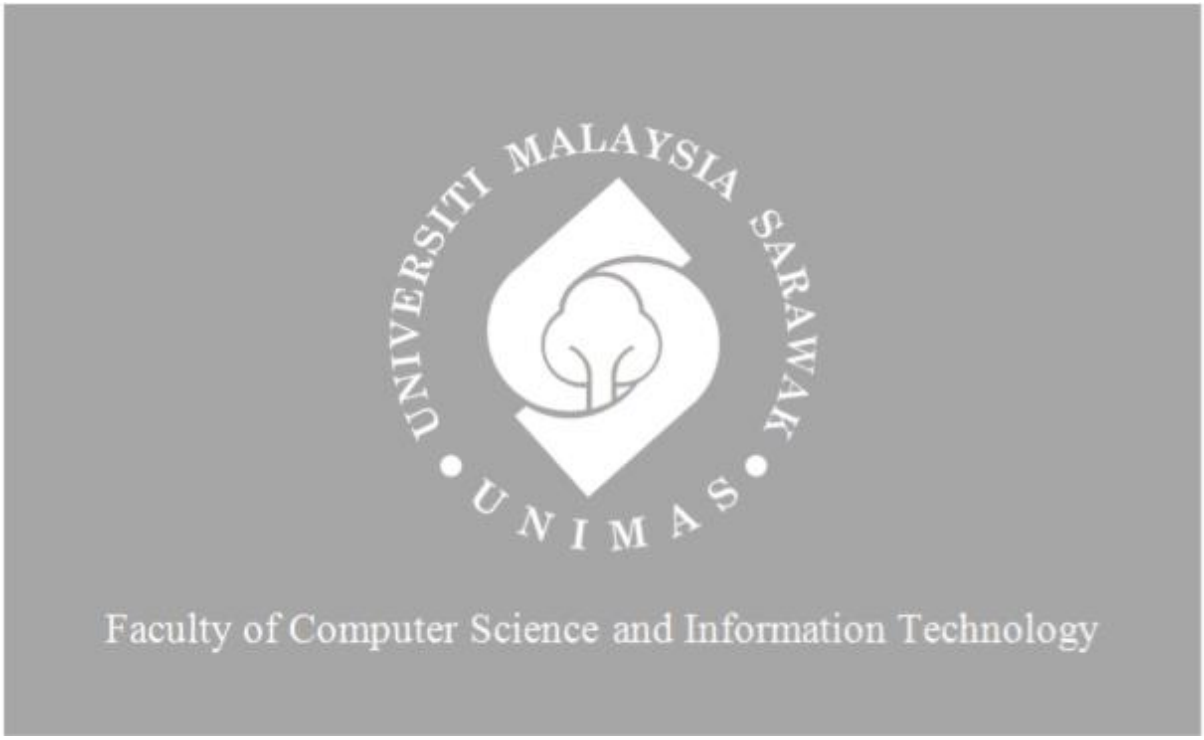


CERVISCAN: CERVICAL CANCER DETECTION USING CONVOLUTIONAL NEURAL NETWORK (CNN)

Ling Yan Ting

Bachelor of Software Engineering with Honours

2024/2025

CERVISCAN: CERVICAL CANCER DETECTION USING CONVOLUTIONAL NEURAL NETWORK (CNN)

LING YAN TING

This project is submitted in partial fulfilment of the
requirements for the degree of
Bachelor of Software Engineering with Honours

Faculty of Computer Science and Information Technology
UNIVERSITI MALAYSIA SARAWAK
2024/2025

CERVISCAN: CERVICAL CANCER DETECTION USING CONVOLUTIONAL NEURAL NETWORK (CNN)

LING YAN TING

Projek ini merupakan salah satu keperluan untuk
Ijazah Sarjana Muda Kejuruteraan Perisian dengan Kepujian

Fakulti Sains Komputer dan Teknologi Maklumat
UNIVERSITI MALAYSIA SARAWAK
2024/2025

UNIVERSITI MALAYSIA SARAWAK

THESIS STATUS ENDORSEMENT FORM

TITLE CerviScan: Cervical Cancer Detection Using Convolutional Neural Network (CNN)

ACADEMIC SESSION: 2024/2025

LING YAN TING

(CAPITAL LETTERS)

hereby agree that this Thesis* shall be kept at the Centre for Academic Information Services, Universiti Malaysia Sarawak, subject to the following terms and conditions:

1. The Thesis is solely owned by Universiti Malaysia Sarawak
2. The Centre for Academic Information Services is given full rights to produce copies for educational purposes only
3. The Centre for Academic Information Services is given full rights to do digitization in order to develop local content database
4. The Centre for Academic Information Services is given full rights to produce copies of this Thesis as part of its exchange item program between Higher Learning Institutions [or for the purpose of interlibrary loan between HLI]
5. ** Please tick (√)

☐	CONFIDENTIAL	(Contains classified information bounded by the OFFICIAL SECRETS ACT 1972)
☐	RESTRICTED	(Contains restricted information as dictated by the body or organization where the research was conducted)
☑	UNRESTRICTED	

yc

(AUTHOR'S SIGNATURE)

Permanent Address

NO 1B, LORONG 1B1,
JALAN JERRWIT BARAT,
96000 SIBU, SARAWAK, MALAYSIA

Date: 20 JUNE 2025

Validated by

(SUPERVISOR'S SIGNATURE)

Ts. Nurfauza Binti Jali
Senior Lecturer
Faculty of Computer Science and Information Technology
Universiti Malaysia Sarawak

Date: 21 July 2025

Note * Thesis refers to PhD, Master, and Bachelor Degree
** For Confidential or Restricted materials, please attach relevant documents from relevant organizations / authorities

ACKNOWLEDGEMENT

I would like to thank Ts. Nurfauza Binti Jali, my supervisor, for all of her help and support during this project. Her patience and insightful feedback have been crucial in shaping the direction and outcome of this project. Her dedication and knowledge sharing have greatly enhanced my learning experience, for which I am sincerely grateful. In addition, I would like to thank my examiner, Dr. Norfadzlan bin Yusup, for insightful criticism that helped me improve my content, as well as my FYP coordinator, Prof. Dr. Wang Ying Chai, for providing clear instructions and guidelines.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

Abbreviations	Meaning
AI	Artificial Intelligence
API	Application Program Interface
Approx.	Approximate
CCD	Charge-coupled device
CNN	Convolutional Neural Network
CSS	Cascading Style Sheets
DL	Deep Learning
DOB	Date of Birth
DT	Decision Tree
e.g.	For example
FN	False Negative
FP	False Positive
HTML	Hypertext Markup Language
IC	Identity Card
ML	Machine Learning
N/A	Not Applicable
NB	Naive Bayes
ONNX	Open Neural Network Exchange
PCA	Principal Component Analysis
RAD	Rapid Application Development
RF	Random Forest
SLR	Simple Logistic Regression
SPP	Spatial Pyramid
SVM	Support Vector Machine
TCT	ThinPrep cytologic test
TN	True Negative
TP	True Positive
UI	User Interfaces
WSI	Whole Slide Imager

ABSTRACT

Cervical cancer remains a major factor in cancer-related deaths among women in the world, especially in areas with limited healthcare resources. The current screening methods, like Pap smear tests, take time and demand expertise and resources due to the need for manual identification of abnormal cervical cells. However, cervical cancer has the potential to be treated if it is detected in the early phases. Therefore, this project, CerviScan, proposes an automated cervical cancer detection system integrating a Convolutional Neural Network (CNN) model trained on the publicly accessible SIPaKMeD Dataset. The CNN model utilises transfer learning with ResNet-50 to classify the cervical Pap smear images into five categories, such as Dyskeratotic, Koilocytotic, Metaplastic, Parabasal, and Superficial-Intermediate. Metrics like accuracy, precision, recall, and F1-score are used to evaluate the model's performance and effectiveness. Moreover, a web-based system that integrates this model is developed to allow medical practitioners to upload Pap smear images, generate and download classification results, as well as manage the list of patients. Last but not least, by reducing the dependency on manual analysis of abnormal cervical cells, CerviScan aims to enhance early detection of cervical cancer, particularly in low-resource settings. Thus, improving the clinical efficiency and treatment success rates.

ABSTRAK

Kanser serviks ialah salah satu kanser yang banyak menyebabkan kematian dalam kalangan wanita, terutamanya di tempat yang kekurangan sumber kesihatan. Ujian saringan yang tersedia sekarang, seperti ujian Pap smear, memerlukan usaha pegawai kesihatan untuk mengenal pasti sel-sel yang mempunyai simptom prakanser. Walau bagaimanapun, kanser serviks ini berpotensi untuk dirawat dan sembuh jika ia dapat dikesan pada peringkat awal. Oleh itu, projek ini, iaitu CerviScan, akan mencadangkan satu sistem pengesanan kanser serviks dengan menggabungkan model Rangkaian Neural Berlingkaran (CNN) yang dilatih menggunakan SIPaKMeD Dataset. Model CNN ini menggunakan 'transfer learning' dengan ResNet-50 untuk mengelaskan imej Pap smear kepada lima kategori, iaitu Dyskeratotic, Koilocytotic, Metaplastic, Parabasal, dan Superficial-Intermediate. Seterusnya, prestasi dan keberkesanan model ini dinilai menggunakan pelbagai metrik, seperti 'accuracy', 'precision', 'recall', dan 'F1-score'. Selain itu, aplikasi web yang menggabungkan model ini dicipta untuk membolehkan pegawai kesihatan memuat naik imej Pap smear, memperoleh dan memuat turun keputusan klasifikasi, serta menguruskan senarai pesakit. Dengan mengurangkan pergantungan kepada analisis manual, CerviScan bertujuan untuk meningkatkan kadar pengesanan awal kanser serviks dan rawatan yang berjaya, terutamanya di kawasan yang kekurangan sumber.

CHAPTER 1: INTRODUCTION

1.1. Introduction

Cervical cancer is among the most typical deadly cancers affecting women. It occurs in the cervix, the lower part of the uterus that opens into the vagina, and it is caused by the persistent infection of Human papillomavirus (HPV). Normally, it takes 15 to 20 years for the abnormal cells to develop into cervical cancer. However, if the women have weaker immune systems, the progress may be faster (World Health Organization, 2024).

The Pap smear test is the preferred screening method for determining precancerous and cancerous cervical cells. However, it might take a significant amount of time due to the involvement of skilled cytologists to take the sample from the cervix surface and pathologists to perform detailed analysis of the samples under a microscope (Kalbhor et al., 2023).

Therefore, this project explores a public dataset of cervical Pap smear images, namely SIPaKMeD, to study and train a CNN model capable of analysing these images and detecting the precancerous and cancerous conditions. A CNN is a deep learning algorithm that is suitable for analysing visual data as it uses principles from linear algebras, namely convolution operations, to extract features and find patterns in images (Awati & Craig, 2024). Additionally, a web-based system integrating the CNN model is be developed in this project, allowing the users to automate the process of cervical cancer detection by uploading images while obtaining classification results through user-friendly interfaces.

1.2. Problem Statement

Cervical cancer is the fourth most common cancer among women globally, especially in low- and middle-income countries where 94% of cervical cancer deaths occurred in the year 2022 (World Health Organization, 2024). Early detection of cervical cancer plays a vital role

in clinical practices as it increases the opportunities for effective treatment. However, traditional screening methods like Pap smear tests or HPV tests rely heavily on skilled medical professionals and are often time-consuming to obtain accurate classification results as the cancerous conditions or cells are hard to differentiate with microscopes and raw eyes.

Recent developments in deep learning technology have created new opportunities to improve cervical cancer detection through automated solutions. While traditional machine learning techniques like Support Vector Machines (SVMs) and Random Forests (RFs) have been widely applied for cancer diagnosis and prediction, they often require human experts to manually remove unnecessary features before training the models (Park et al., 2021). In contrast, Convolutional Neural Networks (CNNs) consist of convolution, non-linear, and pooling layers that learn and extract relevant features through the analysis of spatial information. This ability to process the highly correlated sub-regions of the data makes CNNs useful in domains such as medical imaging and multi-omics, resulting in better performance in tasks like cervical cancer detection (Gupta et al., 2022).

To address the issues, this project proposes to develop a cost-effective and automated solution for cervical cancer detection by analysing and classifying Pap smear images using the trained CNN model. The model is trained using supervised learning on the public SIPaKMeD dataset, which is the largest dataset of cervical cancer images available on Kaggle. The dataset consists of approximately 4049 isolated cell images that were manually cropped from 966 cluster cell images of Pap smear slides and captured through a CCD camera attached to an optical microscope. The images are classified into five categories as explained in Table 1.1.

Table 1.1. List of Cells Images Categories and Their Respective Types and Characteristics

Categories of cell images	Cell types	Characteristics
Superficial-Intermediate	Normal	The majority of cells found in Pap smear test.
Parabasal		Immature squamous cells and smallest epithelial cells.
Koilocytotic	Abnormal	The cells are usually seen in mature skin cells and can sometimes be found in metaplastic cells.
Dyskeratotic		Squamous cells that have started to grow abnormally and undergo keratinisation.
Metaplastic	Benign	Cells from the transformation zone. If left untreated, the cells will become cancerous.

Next, the web-based system integrating the CNN model is designed to allow users to upload Pap smear images and obtain accurate results while serving as a valuable tool, especially in regions with limited healthcare practitioners and infrastructure.

1.3. Scope

This project, CerviScan, involves the development of a web-based system integrating the CNN model for analysing cervical Pap smear images and providing accurate classification results. This system is designed and developed for healthcare practitioners to upload images and receive classification results while improving the efficiency of cervical cancer detection in real-life practices. Table 1.2 summarises the deliverables and exclusions of the project scope.

Table 1.2. Deliverables and Exclusions of the Project Scope

Deliverables	1. The development of a CNN model using PyTorch for healthcare practitioners to analyse and classify cervical Pap smear images according to their five diagnostic categories. 2. The development of a web-based system integrating the trained CNN model to allow healthcare practitioners to upload cervical Pap smear images and obtain downloadable classification results. Django, a Python web framework, is used for back-end development, PostgreSQL for database, and HTML, CSS, and JavaScript for front-end development. 3. The evaluation of the CNN model's performance using accuracy, precision, recall, and F1-score indicators to ensure that it provides accurate cervical cancer detection. 4. The testing of web-based system to ensure smooth user interaction and proper system functionalities.
Exclusions	1. Mobile application development for the system. The system will only be accessible via browsers. 2. Multilingual support for the web-based system. The system will only be supporting English. 3. Advanced clinical diagnostics that require specialised medical expertise or equipment. The system will only serve as a supporting tool for healthcare practitioners by providing the classification results based on the uploaded cervical Pap smear images.

1.4. Research Questions

The following outlines three research questions (RQs) to be solved by accomplishing this project:

a. How can a CNN model be developed to accurately classify cervical Pap smear images into specific diagnostic categories?

b. What evaluation metrics can effectively assess the classification performance of the CNN model?

c. How can a user-friendly web-based system be implemented to facilitate real-time cervical cancer detection using the CNN model?

1.5. Aims and Objectives

This project is conducted primarily to develop an efficient and user-friendly web-based system for cervical cancer detection while integrating a Convolutional Neural Network (CNN) model. The aims and objectives of the project are outlined as follows:

a. To review and synthesise existing research on cervical cancer detection techniques using image processing and CNN-based methods.
b. To design and develop a CNN model using transfer learning for classifying cervical Pap smear images into five diagnostic categories.
c. To integrate the trained CNN model into a web-based system that supports image upload, classification and result management.
d. To evaluate the model's performance using metrics including accuracy, precision, recall, and F1-score on a publicly available dataset.
e. To assess the usability and functionality of the web-based system through structured user testing.

1.6. Brief Methodology (Rapid Application Development)

Rapid Application Development (RAD), which emphasises on quick and iterative development cycles, is selected for this project. In contrast to the Waterfall model, which adheres a strict structure where each phase must be accomplished before proceeding to the next, RAD model offers more flexibility for this project (Egeonu, 2022). This is because it allows continuous refinement of the system development throughout the project based on ongoing feedback while eliminating the need to rework from the start and minimising the risk of

potential additional costs or delays. Figure 1.1 shows the phases of the Rapid Application Development (RAD) methodology.

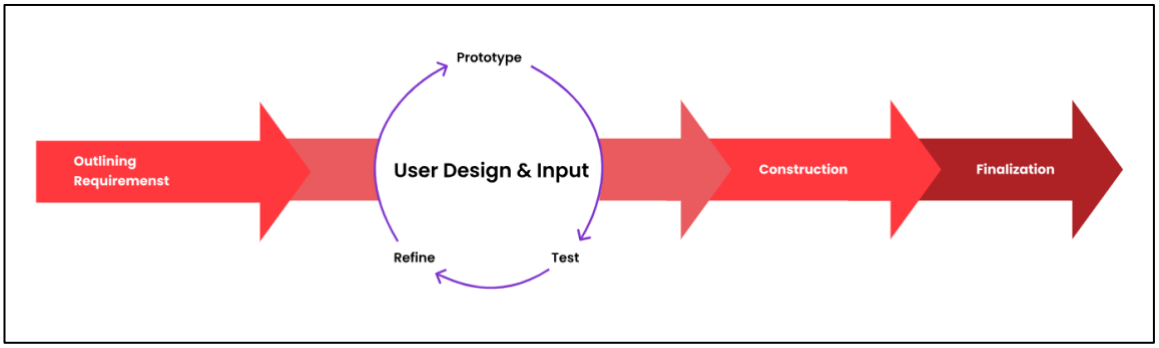


Figure 1.1. Rapid Application Development (RAD) Phases (Iurev, 2022)

1.6.1. Outline Requirements

The first phase of RAD methodology outlines a loose set of requirements for the project, including the main objectives, scope, limitations and functionalities. These requirements are derived from the project goals and domain knowledge while leveraging available insights to ensure they align seamlessly with the project's objectives. The term “loose” refers to the flexible definition of project requirements as RAD allows modifications at any stage of the development cycle. Besides that, this phase also involves exploring and selecting the publicly accessible cervical Pap smear image datasets that are suitable for training the CNN model.

1.6.2. User Design and Input

User Design and Input phase is the second phase of RAD which focuses on designing and developing the prototype of the system with all or most of the requirements covered. This phase involves two primary tasks. First, the layers of the CNN model are designed, ensuring an appropriate selection of layers and configurations to optimise the model’s performance for cervical Pap smear image classification. Second, a prototype of the web-based system is developed. This includes mock-ups that outline the key features, such as uploading images, viewing classification results, and accessing patient information. The prototype will serve as a

visual representation of the system's functionality. Additionally, early coding is carried out in this phase to create a basic and interactive version of the system, allowing preliminary testing and refinement. Iterative improvements to the design is made based on the insights gained during the prior testing, ensuring the design meets the intended functional requirements.

1.6.3. Development / Construction

In the development phase, the final working web-based system integrating the trained CNN model is developed using several technical stacks like HTML, CSS, JavaScript, Python, PyTorch, and PostgreSQL. The CNN model is trained using a selected publicly available cervical Pap smear images dataset, SIPaKMeD, to ensure accuracy in cervical cancer detection. Concurrently, the system's front-end and back-end is also developed. With the early coding completed in the previous phase, the development of the web-based system using RAD methodology is expected to progress faster than the traditional Waterfall model. Moreover, to ensure that all system components work seamlessly together, feedback is welcomed, and continuous testing will be conducted throughout this phase.

1.6.4. Finalisation

In the finalisation phase, comprehensive testing of the final working product is carried out. The testing may include the performance test of the functionalities of the web-based system and the CNN model accuracy test using a validation cervical Pap smear images dataset. After successful testing and potential debugging, the system is deployed and hosted for real-life usage.

1.7. Significance of Project

The completion of this project can offer a system that enhances the efficiency of cervical cancer detection in real-life clinical practices by automating the analysis of cervical

Pap smear images. By integrating the CNN model, the system will accelerate the cervical cancer detection process and reduce the dependency on manual analysis by medical experts, especially in regions with limited healthcare infrastructure and resources.

1.8. Project Schedule

Referring to the Final Year Project I and Final Year Project II courses, this project should be able to finish within two semesters, which are approximately eight months. Final Year Project I is more towards project planning as well as requirements analysis and design, whereas Final Year Project II focuses more on the implementation of the project. Table 1.3 lists the tasks to be completed for this project and their respective estimated completion dates, whereas Figure 1.2 and Figure 1.3 visualise the project timeline using a Gantt chart.

Table 1.3. List of Tasks for CerviScan with Their Estimated Completion Dates

Tasks	**Estimated Dates**
Phase 1: Outline Requirements	
Understand the proposed topic	10/10/24 – 15/10/24
Identify the problem	15/10/24 – 19/10/24
Determine the project scope and goals	20/10/24 – 25/10/24
Explore and decide on available datasets and tools	26/10/24 – 01/11/24
Conduct a literature review and explore related studies on cervical cancer detection and CNNs	02/11/24 – 15/11/24
Analyse and outline the project requirements	16/11/24 – 21/11/24
Phase 2: User Design and Input	
Plan dataset preprocessing methods	03/12/24 – 06/12/24
Plan CNN model training	07/12/24 – 11/12/24
Requirements specification for CerviScan system	12/12/24 – 20/12/24
Design mock-up for CerviScan’s UI	21/12/24 – 31/12/24

Do early coding for the system's front-end	02/01/25 – 17/01/25
Prepare for the FYP I presentation and submit the final report	17/01/25 – 24/01/25
Phase 3: Development / Construction	
Implement and train the CNN model	17/03/25 – 20/03/25
Develop CerviScan system	21/03/25 – 15/05/25
Integrate the trained CNN model into the web system	05/05/25 – 15/05/25
Perform regular system testing and refinements	17/03/25 – 15/05/25 (weekly)
Phase 4: Finalisation	
Comprehensive testing of the system and CNN model	16/05/25 – 20/05/25
Prepare a user manual and test case documentation	21/05/25 – 31/05/25
Deploy CerviScan on a live server	04/06/25 – 10/06/25
Prepare for the FYP II presentation and submit the final report	11/06/25 – 02/07/25

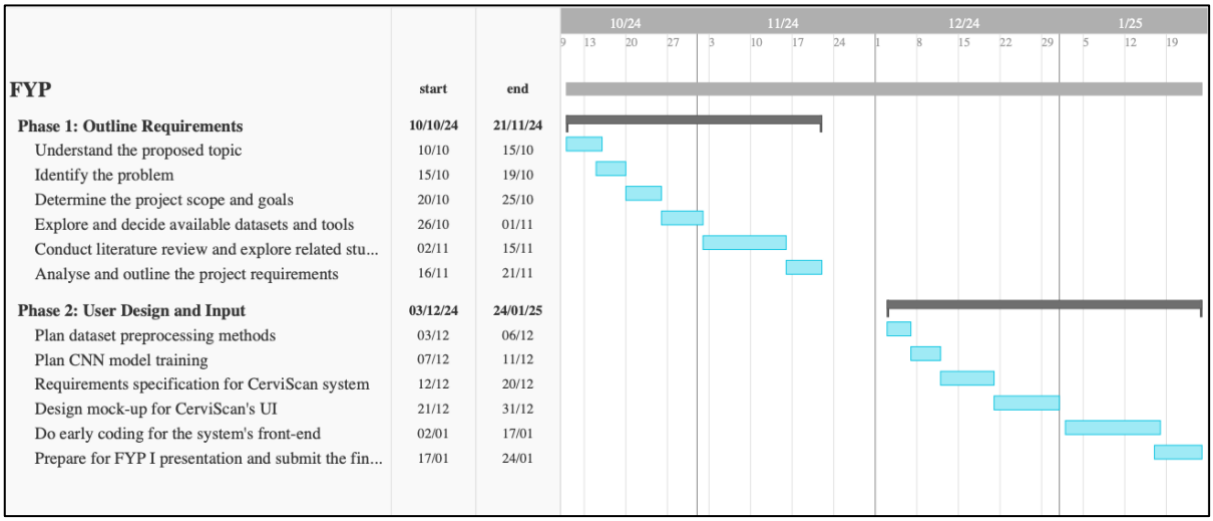


Figure 1.2. Gantt Chart Visualising Phase 1 and Phase 2 (FYP I) of CerviScan Project

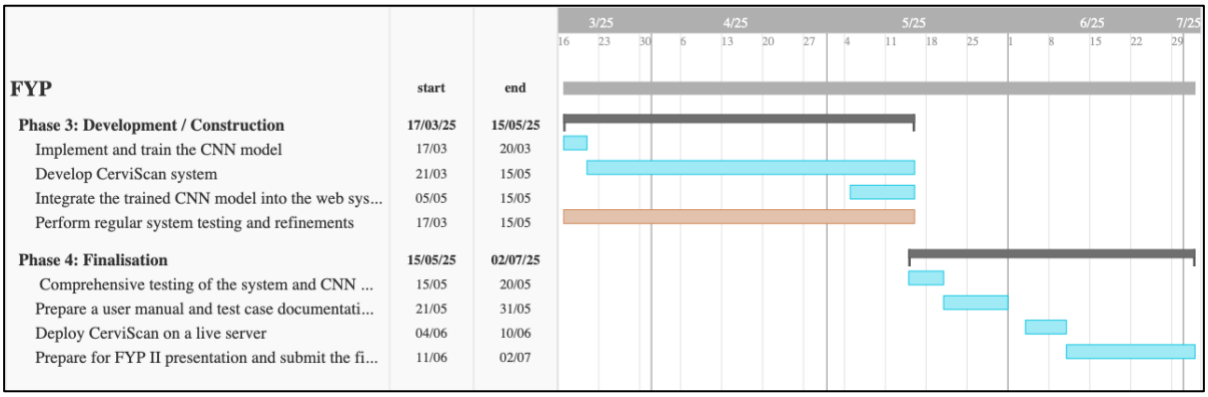


Figure 1.3. Gantt Chart Visualising Phase 3 and Phase 4 (FYP II) of CerviScan Project

1.9. Expected Outcomes

The expected outcome of this project is the development of a reliable web-based system for cervical Pap smear image classification through a trained CNN model. The detailed expected outcomes are listed as follows:

a. A well-trained and validated Convolutional Neural Network (CNN) model using PyTorch and transfer learning (ResNet-50) that can accurately classify cervical Pap smear images into five diagnostic categories (Dyskeratotic, Koilocytotic, Metaplastic, Parabasal, and Superficial-Intermediate) with performance evaluated using accuracy, precision, recall, and F1-score.

b. A functional and user-friendly web-based system developed with Django (backend), Bootstrap 5/HTML/CSS/JavaScript (frontend), and PostgreSQL (database), which allows healthcare practitioners to securely upload Pap smear images, receive automated classification results, view result details, and download the reports in PDF format.

c. A demonstrable improvement in efficiency for cervical cancer detection workflows by automating the image classification process, thereby reducing the diagnostic workload of medical practitioners and supporting faster decision-making, especially in settings with limited medical infrastructure.

1.10. Project Report Outline

There are six chapters to be completed in this project report. The chapters are listed and explained as follows:

1. **Chapter 1: Introduction**

This chapter introduces the project by providing an overview of the selected problem, the project's significance, and the expected outcomes of the proposed solution. This chapter covers the introduction, problem statement, project scope, aims and objectives, brief methodology, project schedule, and expected outcomes.

2. **Chapter 2: Literature Review**

Chapter two includes the study of existing research and CNN models or systems related to cervical cancer detection and Pap smear image analysis. Comparative analysis of related systems or research with the proposed web-based system, CerviScan, is done to produce a better and more comprehensive solution.

3. **Chapter 3: Methodology**

This chapter explains the phases of the applied methodology for this project as well as identifies and analyses the requirements needed for the proposed web-based system. Then, the design for system architecture, CNN model, and user interfaces are created based on the analysed requirements.

4. **Chapter 4: Implementation**

This chapter details the development of the CerviScan system, including building and training the CNN model, developing the website, and integrating the model. The CNN model is built using transfer learning with ResNet-50 and trained on the SIPaKMed dataset. Then, the

website is developed using Django, PostgreSQL, HTML, CSS, Bootstrap 5, and JavaScript, following the predefined UML diagrams and mock-ups.

5. **Chapter 5: Testing**

This chapter focuses on verifying system functionality and evaluating the CNN model's classification accuracy before deployment. Functional testing ensures that key features work as intended, while the model's performance is assessed using accuracy, precision, recall, and F1-score.

6. **Chapter 6: Conclusion and Future Work**

The final chapter reviews the overall progress and work completed throughout the project. The conclusion highlights the project's achievements, its limitations, as well as suggestions, recommendations, and any potential opportunities for future work that could serve as an extension of this project.

1.11. Summary

This chapter provides an overview of the proposed system, CerviScan, a web-based system for detecting cervical cancer through Pap smear image analysis and classification using a CNN model. It addresses the limitations of traditional screening methods, particularly Pap smear tests, and the purposes of the project. Besides that, this chapter also briefly explains the project scope, project schedule, as well as the chosen methodology, Rapid Application Development (RAD), to develop the proposed system.

CHAPTER 2: LITERATURE REVIEW

2.1. Introduction

Cervical cancer remains a significant global health problem and detecting it in the early phases can help reduce the mortality rates. As analysing the cervical Pap smear images can be time-consuming and require much manpower or effort from medical practitioners, developing a computer-aided solution has become crucial, especially in low-resource regions. Therefore, with the rapid advancement in artificial intelligence and deep learning (DL) fields, Convolutional Neural Networks (CNNs) have become a useful tool for medical image analysis and classification. This chapter reviews the basic CNN architecture and discuss the related works on its applications in cervical cancer detection as well as two of the most widely used deep learning frameworks, PyTorch and TensorFlow.

2.2. Overview of Convolutional Neural Networks (CNNs)

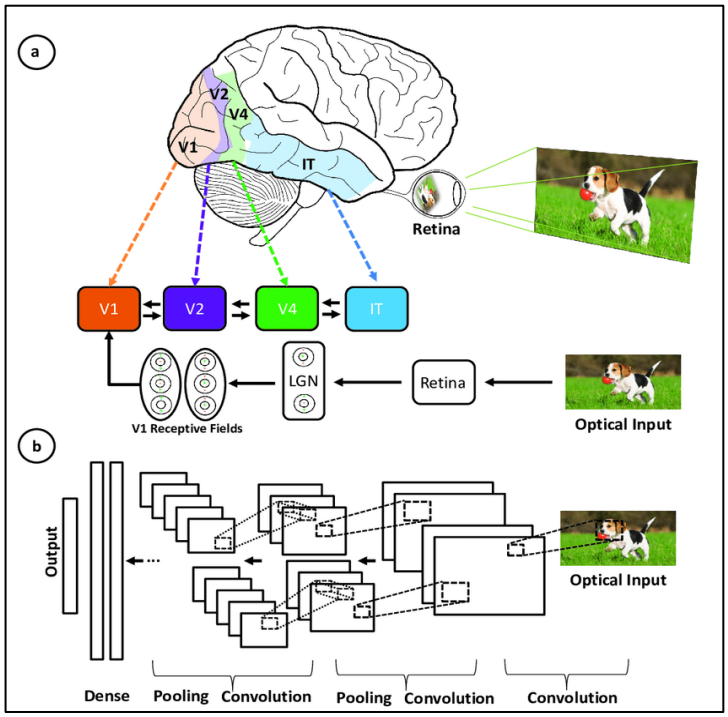


Figure 2.1. Example of the association between the regions related to the visual cortex of the human brain and the layers of a CNN (Roffo, 2017)

Convolutional Neural Network (CNN) is a type of deep learning algorithm that specialises in image recognition tasks like classification, detection and segmentation. Unlike traditional machine learning techniques which require manual feature extraction, CNN is a biologically inspired algorithm which extracts and learns features from images through convolution operations that mimic how the human brain processes the visual information. Figure 2.1 illustrates the association between the regions related to the visual cortex of the human brain and the layers of a CNN model. Besides that, CNNs are composed of a series of layers, each of which performs a specific function to process the input data. Figure 2.2 depicts the example of a basic CNN architecture with each layer being explained in Table 2.1.

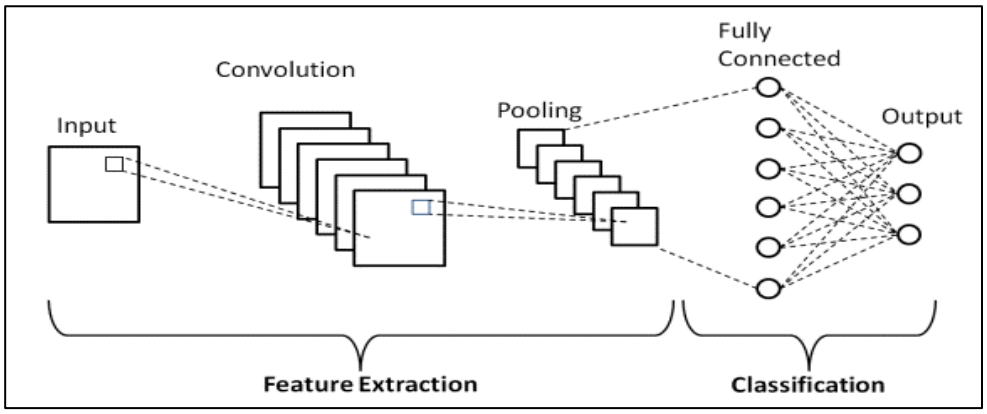


Figure 2.2. Diagram of the Basic CNN Architecture (Gurucharan, 2020)

Table 2.1. Layers of Basic CNN Architecture and Their Functions

CNN layers	Purposes	Processes
Convolutional Layer	Extracts unique spatial features from the input data at different scales and orientations.	Applies programmable filters, or kernels, to the input data by iteratively multiplying each element and adding the results to generate a feature map (output) for detecting the visual patterns in the input data.
Pooling Layer	Reduces the spatial dimensions of feature maps while preserving only the essential features. This can help to reduce the	Performs down sampling of the feature maps using pooling operations like max pooling or average pooling. While max pooling identifies the highest value within each region of the feature maps,

	computational complexity and the risk of overfitting.	average pooling computes the mean value for those regions.
Fully Connected Layer	Aggregates features learned from convolutional and pooling layers to produce the output.	Performs a linear transformation towards the flattened features obtained from the previous layers, followed by a non-linear activation function.

Note: Information adapted from (Jadeja et al., 2023).

2.3. Review on Related Works of CNNs in Cervical Cancer Detection

In addition to training CNNs from scratch, pre-trained CNN models are often used to improve performance and reduce the computational burden. A pre-trained model refers to a network that has been previously trained on a large dataset, often for large-scale image classification tasks and is capable of recognising a wide range of features. Transfer learning is the process of fine-tuning a pre-trained model to a new task by applying the knowledge it obtained during the initial training. This approach is useful because it requires less training data and computational resources, making it suitable for tasks with small datasets or in environments with limited resources. Recent studies on cervical cancer detection highlight the effectiveness of pre-trained CNN models and hybrid approaches.

Promworn et al. (2019) carried out a comparison study using three CNN models (AlexNet, DenseNet-161, and ResNet-101) for Pap Smear image classification. The researchers used the DTU/Herlev Pap Smear Database, which consists of 917 images divided into normal and abnormal cells with 245 images and 675 images respectively. They separated the dataset into three groups, that were 70% for training, 20% for validation, and 10% for testing. Moreover, PyTorch was utilised for implementing the models. They also evaluated the performance of these models based on metrics like accuracy, sensitivity, specificity, and computation time. According to the results of the analysis, the DenseNet-161 model achieved the highest accuracy (94.38%) and sensitivity (100%), making it the most suitable model for

the implementation of Whole Slide Imager (WSI) as well as early detection of cervical abnormalities, despite having the longest computation time.

Besides that, Yu et al. (2021) studied the application of deep learning models for the automatic classification of normal and abnormal cervical cells. They used the ThinPrep cytologic test (TCT) dataset collected from Baoding No.4 Hospital in China. The dataset has a total of 2504 cell samples, which are 1202 abnormal and 1302 normal samples. They then developed four models with PyTorch, including a simple 10-layered CNN model, a CNN model with a Spatial Pyramid Pooling (SPP) layer, a CNN model with an inception module, and a CNN model that combines both the SPP layer and inception module. The study showed that the final model (CNN + SPP + Inception) had the best performance for cervical cell classification with 0.997 area under the curve (AUC) and 98% accuracy.

Furthermore, in a similar study conducted by Remya & Raimond (2023), they fine-tuned and compared five types of pre-trained CNN models, like VGG-16, VGG-19, ResNet-50, ResNet-101 and Efficient Net v3, for classifying Pap smear images. The datasets used were SIPaKMeD and Herlev datasets whereas the Keras module of TensorFlow was used to implement the models. Additionally, to enhance the models' performance, the researchers applied several augmentation techniques to the images, including horizontal flipping, vertical flipping, horizontal shifting, vertical shifting, zooming and rotation. The study demonstrated that ResNet-50 outperformed the other five models by achieving the highest accuracy of 95.25% and 94.4% for the SIPaKMeD and Herlev datasets respectively.

Lastly, the study carried out by Mathivanan et al. (2024) introduced a hybrid approach using deep learning (DL) and machine learning (ML) to enhance cervical cancer classification. The study also utilised the Pap smear images from the SIPaKMeD dataset with 60% for training and 40% for testing. Not only this, the pre-trained CNN models such as AlexNet, ResNet-101,

ResNet-152, and InceptionV3 were used for feature extraction while various ML algorithms were utilised for classification purposes, including Simple Logistic Regression (SLR), Decision Tree (DT), Random Forest (RF), Naive Bayes (NB), and Principal Component Analysis (PCA). According to the findings, ResNet-152 showed the highest performance by achieving 98.08% accuracy when combined with the Simple Logistic classifiers, which also outperformed other ML classifiers.

2.3.1. Comparison on Related Works of CNNs in Cervical Cancer Detection

Table 2.2 compares various studies on the applications of CNN models in cervical cancer detection. It includes information about the datasets used, CNN techniques, and the performance metrics for each study.

Table 2.2. Comparison on the CNNs applications in Cervical Cancer Detection

<table>
<tr><th>No.</th><th>Authors, Year</th><th>Datasets</th><th>CNN Techniques</th><th>Epochs</th><th>DL Frameworks</th><th>Results</th></tr>
<tr><td rowspan="2">1</td><td rowspan="2">Promworn et al. (2019)</td><td rowspan="2">DTU/Herlev Pap Smear Database</td><td rowspan="2">AlexNet, DenseNet-161, ResNet-101</td><td rowspan="2">20</td><td rowspan="2">PyTorch</td><td>Training
<table>
<tr><th>Models</th><th>Accuracy (%)</th><th>Time taken</th></tr>
<tr><td>AlexNet</td><td>88.95</td><td>4 m 49s</td></tr>
<tr><td>ResNet-101</td><td>90.61</td><td>26 m 5s</td></tr>
<tr><td>DenseNet-161</td><td>92.82</td><td>31 m 0s</td></tr>
</table></td></tr>
<tr><td>Prediction
<table>
<tr><th>Models</th><th>Accuracy (%)</th><th>Sensitivity (%)</th><th>Specificity (%)</th></tr>
<tr><td>AlexNet</td><td>93.26</td><td>96.97</td><td>73.91</td></tr>
<tr><td>ResNet-101</td><td>91.01</td><td>98.48</td><td>65.56</td></tr>
<tr><td>DenseNet-161</td><td>94.38</td><td>100</td><td>78.26</td></tr>
</table></td></tr>
<tr><td rowspan="3">2</td><td rowspan="3">Yu et al. (2021)</td><td rowspan="3">TCT dataset collected from Baoding No.4 Hospital, China</td><td>Model A: 10-layered CNN model</td><td>400</td><td rowspan="3">PyTorch</td><td rowspan="3">
<table>
<tr><th rowspan="2">Metrics</th><th colspan="4">Models</th></tr>
<tr><th>A</th><th>B</th><th>C</th><th>D</th></tr>
<tr><td>Precision (%)</td><td>87.6</td><td>93.5</td><td>93.4</td><td>97.5</td></tr>
<tr><td>Sensitivity (%)</td><td>97.1</td><td>95.9</td><td>94.6</td><td>98.3</td></tr>
<tr><td>Specificity (%)</td><td>87.4</td><td>93.9</td><td>93.9</td><td>97.7</td></tr>
<tr><td>Accuracy (%)</td><td>92.0</td><td>94.8</td><td>94.3</td><td>98.0</td></tr>
<tr><td>F1 Score (%)</td><td>92.1</td><td>94.7</td><td>94.0</td><td>97.9</td></tr>
<tr><td>H-mean (%)</td><td>92.0</td><td>94.8</td><td>94.2</td><td>98.0</td></tr>
</table></td></tr>
<tr><td>Model B: CNN model with SPP layer</td><td>450</td></tr>
<tr><td>Model C: CNN model with an</td><td>600</td></tr>
</table>

<table>
<tr><td></td><td></td><td></td><td>inception module</td><td></td><td></td><td><table><tr><td>AUC</td><td>0.953</td><td>0.967</td><td>0.966</td><td>0.997</td></tr></table></td></tr>
<tr><td></td><td></td><td></td><td>Model D: CNN model with SPP layer and inception module</td><td>600</td><td></td><td></td></tr>
<tr><td>3</td><td>Remya & Raimond (2023)</td><td>1. SIPaKMeD dataset
2. Herlev dataset</td><td>VGG-16, VGG-19, ResNet-50, ResNet-101, Efficient Net v3</td><td>50</td><td>TensorFlow (with Keras module)</td><td><table><tr><th rowspan="2">Models</th><th colspan="2">Accuracy (%)</th></tr><tr><th>SIPaKMeD</th><th>Herlev</th></tr><tr><td>VGG-16</td><td>87.5</td><td>82.5</td></tr><tr><td>VGG-19</td><td>90</td><td>90.5</td></tr><tr><td>ResNet-50</td><td>95.25</td><td>94.4</td></tr><tr><td>ResNet-101</td><td>93</td><td>90.06</td></tr><tr><td>Efficient Net v3</td><td>91.2</td><td>89.94</td></tr></table></td></tr>
<tr><td>4</td><td>Mathivanan et al. (2024)</td><td>SIPaKMeD dataset</td><td>Hybrid CNN models and ML algorithms for feature extraction and classification purposes respectively.</td><td>N/A</td><td>N/A</td><td>Accuracy (%)<table><tr><th rowspan="2">ML classifiers</th><th colspan="4">CNN models</th></tr><tr><th>AlexNet</th><th>ResNet-101</th><th>ResNet-152</th><th>Inception V3</th></tr><tr><td>SLR</td><td>96.31</td><td>95.81</td><td>98.08</td><td>95.01</td></tr><tr><td>DT</td><td>90.59</td><td>90.09</td><td>94.29</td><td>89.29</td></tr><tr><td>RF</td><td>92.23</td><td>91.73</td><td>95.93</td><td>90.93</td></tr><tr><td>NB</td><td>87.15</td><td>86.65</td><td>90.85</td><td>85.85</td></tr><tr><td>PCA</td><td>92.52</td><td>92.02</td><td>95.22</td><td>91.22</td></tr></table></td></tr>
</table>

			CNN models: AlexNet, ResNet-101, ResNet-152, InceptionV3 **ML classifiers:** SLR, DT, RF, NB, PCA			

2.3.2. Comparison on Several Pre-trained CNN Models

Several pre-trained CNN models that demonstrated high performance in the related research are selected for comparison, including AlexNet, DenseNet-161, ResNet-50, ResNet-101, ResNet-152, and VGG-19. Table 2.3 compares the models in the aspects of their number of parameters, architecture, strengths, and limitations.

Table 2.3. Comparison on Several Pre-trained CNN Models

Models	Number of Parameters (Approx.)	Architecture	Strengths	Limitations
AlexNet	61.1M	Consists of five convolutional layers and three fully connected layers (Promworn, 2019).	Efficient on small datasets with relatively fast training due to its shallow architecture.	Considered outdated as it has lower accuracy compared to modern deep learning models and lacks advanced feature extraction capabilities.
DenseNet-161	28.7M	Consists of dense blocks that connect each layer to every other layer (Promworn, 2019).	Highly efficient in parameter usage, ensuring strong feature propagation, improved gradient flow, and stable training.	Its complex connectivity requires high memory, making it computationally expensive and challenging to train in resource-limited environments.
ResNet-50	25.0M	Consists of 50 layers with residual blocks.	Provides a good balance between depth and efficiency, offering strong classification performance while mitigating the degradation problem in deeper networks.	Has fewer layers than deeper models like ResNet-101 or ResNet-152, which may limit its ability to learn highly complex features.

ResNet-101	44.5M	Consists of 101 layers with residual blocks.	Offers improved accuracy compared to ResNet-50 while benefiting from residual learning, which enhances gradient flow and training stability.	The increased depth leads to higher training time and computational demands, making it less efficient for real-time applications.
ResNet-152	60.2M	Consists of 152 layers with residual blocks.	Able to achieve excellent accuracy and effectively mitigate the vanishing gradient problem, ensuring stable training in deep architectures.	Its high depth makes it computationally expensive, requiring significant processing power and memory, which can slow down both training and inference.
VGG-19	143.7M	Consists of 16 convolutional layers and three fully connected layers (Remya & Raimond, 2023).	Widely used due to its straightforward structure, making it easy to implement and fine-tune for various image classification tasks.	Has a large number of parameters, increasing computational cost and memory requirements for both training and inference.

2.4. Review on Deep Learning Frameworks

This section provides an overview of two popular deep learning frameworks, PyTorch and TensorFlow, then followed by a comparison of the characteristics relevant to cervical Pap smear image classification in this project. These frameworks were also used in the papers reviewed above, with two of the studies utilising PyTorch and one implementing the Keras module of TensorFlow.

2.4.1. PyTorch

PyTorch is an open-source deep learning framework developed by Meta AI, formerly known as Facebook's AI Research Lab (FAIR), that has grown in popularity in the research community due to its flexibility and ease of use. Unlike TensorFlow, which uses a static computational graph, PyTorch uses a dynamic computational graph that allows developers to adjust the model during execution. This flexibility makes it an ideal framework for research and rapid prototyping. Moreover, PyTorch is also known for its user-friendly Pythonic interface, which is compatible with the Python ecosystem while simplifying model building and debugging (Pandya, 2024).

Besides that, PyTorch offers comprehensive machine learning functionalities such as data loading, random image loading, geometric transformations, calculations of accuracy and loss, as well as backward learning processes like backpropagation and adjustable optimiser for AI learning rate (Promworn et al., 2019). In addition, PyTorch has a powerful library and Python APIs that support the development of deep learning models for image classification purposes, such as cervical Pap smear image classification in this project. For example, the torch.nn module provides essential components like layers, loss functions, and optimisers, that are important for building and training the model. Furthermore, torchvision offers pre-trained models (e.g., ResNet-50) and image transformation utilities, which help in processing image data and improving the model's performance through techniques like transfer learning.

Furthermore, PyTorch is equipped with TorchServe, an efficient and intuitive tool for large-scale model deployments. For instance, TorchServe supports features like multi-model serving, logging, and metrics, and the creation of RESTful endpoints for easy integration with applications. PyTorch also provides native Open Neural Network Exchange (ONNX) support, which allows the export of trained models to a standard format compatible with ONNX-

supported platforms, runtimes, and visualiser (*PyTorch*, n.d.). This feature ensures that models developed in PyTorch can be used across a range of frameworks and devices, hence increasing its interoperability and flexibility.

2.4.2. TensorFlow

TensorFlow is an open-source deep learning framework developed by the Google Brain Team, which is widely used for tasks like natural language processing, image recognition, handwriting recognition and more. As previously mentioned, TensorFlow formerly used a static computational graph whereby the graph and variables are defined upfront and remain fixed during training. However, with the release of TensorFlow 2.0 onwards, dynamic computational graphs were introduced to allow more adaptable and flexible model building (*TensorFlow and PyTorch*, n.d.).

Moreover, TensorFlow is known for its ability to perform low-level operations on a wide range of acceleration platforms, including CPUs, and GPUs, its efficient automatic gradient computation, its scalability for production environments, and its support for exporting computational graphs to enhance interoperability across multiple platforms and devices. For instance, TensorFlow includes tools, such as TensorFlow Serving, TensorFlow Lite and TensorFlow.js for production environments, mobile devices and web applications, respectively.

Besides that, TensorFlow also includes the Keras API, a high-level and user-friendly interface designed to simplify the development and training of neural network models. Keras can assist in reducing the complexity of DL model implementations as it allows rapid prototyping by offering pre-built layers, optimisers, and loss functions. Furthermore, TensorFlow includes TensorFlow Hub, a repository of pre-trained models that are ready for fine-tuning and deployment.

2.4.3. Comparison on PyTorch and TensorFlow

Table 2.4 compares two reviewed deep learning frameworks, that are PyTorch and TensorFlow. It outlines their differences in architecture, ease of use, flexibility, and scalability.

Table 2.4. Comparison on PyTorch and TensorFlow

Criteria	PyTorch	TensorFlow
Architecture	Dynamic computational graph.	• TensorFlow 1.x: Static computational graph • TensorFlow 2.0+: Dynamic computational graph
Ease of Use	User-friendly and Pythonic interface that eases debugging tasks. Hence, it is great for quick prototyping.	Higher learning curve but can be simplified by using Keras API for easier model development.
Flexibility	Highly flexible as it is easy to modify during development.	Less flexible but can be enhanced with Keras for prototyping.
Scalability	Suitable for small to medium-scale models but also supports large-scale production with tools like TorchServe for deployment and ONNX for interoperability.	Highly scalable with built-in tools like TensorFlow Serving, TensorFlow Lite or TensorFlow.js for deployment.

2.5. Summary

This chapter reviews the fundamentals of CNN, its basic architecture and layers, as well as the previous studies on its applications in cervical cancer detection. The studies above have shown that pre-trained CNN models like ResNet models, VGG-19, DenseNet-161, and a hybrid approach that combines DL and ML had good potential in enhancing the accuracy of cervical cell image classification.

Based on these findings, this project performs transfer learning of ResNet-50 using PyTorch to build and train a CNN model for classifying the cervical Pap smear images

collected from the SIPaKMed dataset. PyTorch is chosen for its flexibility, ease of use, and dynamic computation graph, which simplifies the process of fine-tuning the pre-trained ResNet-50 model for cervical Pap smear image classification.

Furthermore, ResNet-50 is chosen for its efficient architecture, which uses residual blocks to facilitate the direct flow of information and mitigate the vanishing gradient problem while maintaining high accuracy. It also has fewer layers than ResNet-101 and ResNet-152 which allows it to provide a balance between computational cost and classification accuracy while making it ideal for integration into the proposed web-based system, CerviScan. Finally, when considering CerviScan's aim of serving as an efficient tool for cervical cancer detection in regions with limited resources, ResNet-50 is a better choice than VGG-19, despite the latter's popularity in image classification tasks. This is because ResNet-50 has a significantly lower parameter count than VGG-19 (25.0M vs. 143.7M), which reduces memory usage and computational costs, making it more suitable for deployment in low-resource environments.

CHAPTER 3: METHODOLOGY

3.1. Introduction

An overview of the methodology used in the development of CerviScan, a web-based system designed to assist in the detection of cervical cancer through the classification of Pap smear images using CNN, is explained in this chapter. The main goal of this project is to create an efficient and accurate system that supports medical professionals in diagnosing abnormalities of cervical cells. To achieve this, a structured yet flexible development approach is essential. Therefore, the Rapid Application Development (RAD) methodology has been adopted for its iterative and adaptive nature, allowing continuous improvements and quicker delivery of the system. This section discusses the key aspects of the project, such as requirements specifications, technologies used, the ResNet-50 model, the SIPaKMeD dataset, and the user interface prototype. Additionally, a research process flowchart is included to illustrate the overall research process, distinguishing it from the system development process, which specifically follows the RAD methodology.

3.2. Research Process Flowchart

This section presents a research process flowchart (Figure 3.1) to illustrate the overall steps involved in training the CNN model. The process starts with collecting images from the SIPaKMeD Dataset, which is then divided into specific sizes of subsets for training, testing and validation. Image preprocessing techniques, such as resizing, augmentation, and normalisation, are applied to enhance the images and improve model performance. Then, the ResNet-50 model undergoes transfer learning and fine-tuning using PyTorch, with the training process leveraging free cloud GPU services provided by Kaggle. Next, the model's performance is evaluated using various metrics, including accuracy, precision, recall, and F1-score, to assess its effectiveness in classifying the cervical Pap smear images into the five pre-

defined categories. Once the model demonstrates satisfactory performance, it is deployed and integrated into the web-based system.

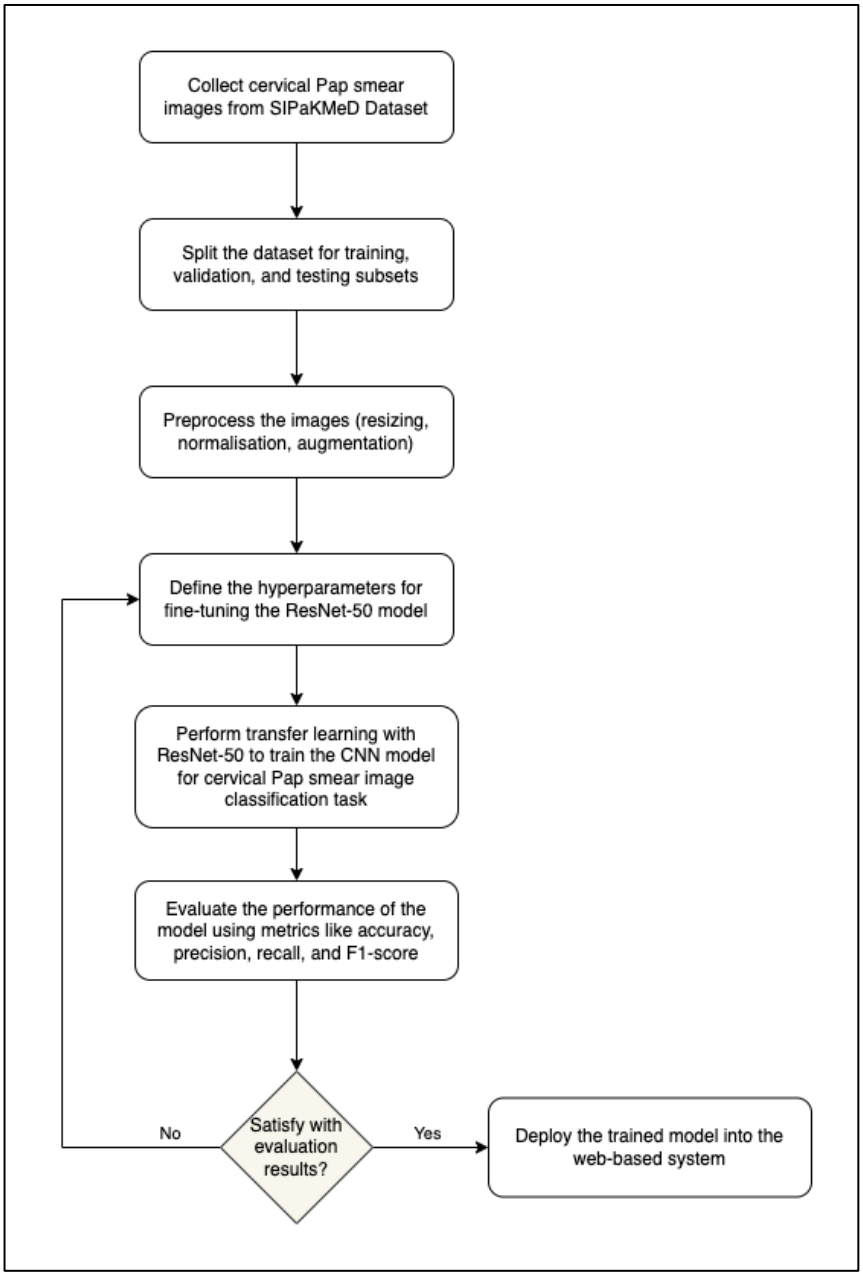


Figure 3.1. Research Process Flowchart

The following sub-sections provides a detailed explanation of the dataset, including its division size and image preprocessing techniques, along with an overview of ResNet-50's architectural structure, hyperparameter specifications for fine-tuning, as well as the purpose and formulae of evaluation metrics that are used in the project.

3.2.1. SIPaKMeD Dataset

The SIPaKMeD dataset is a collection of cervical Pap smear images that can be obtained from Kaggle and is used to train the CNN model for CerviScan. This dataset consists of over 4,000 images from various categories of cervical cells, namely Dyskeratotic, Koilocytotic, Metaplastic, Parabasal, and Superficial-Intermediate (Figure 3.2). Each category of the cervical cells is explained in Table 1.1.

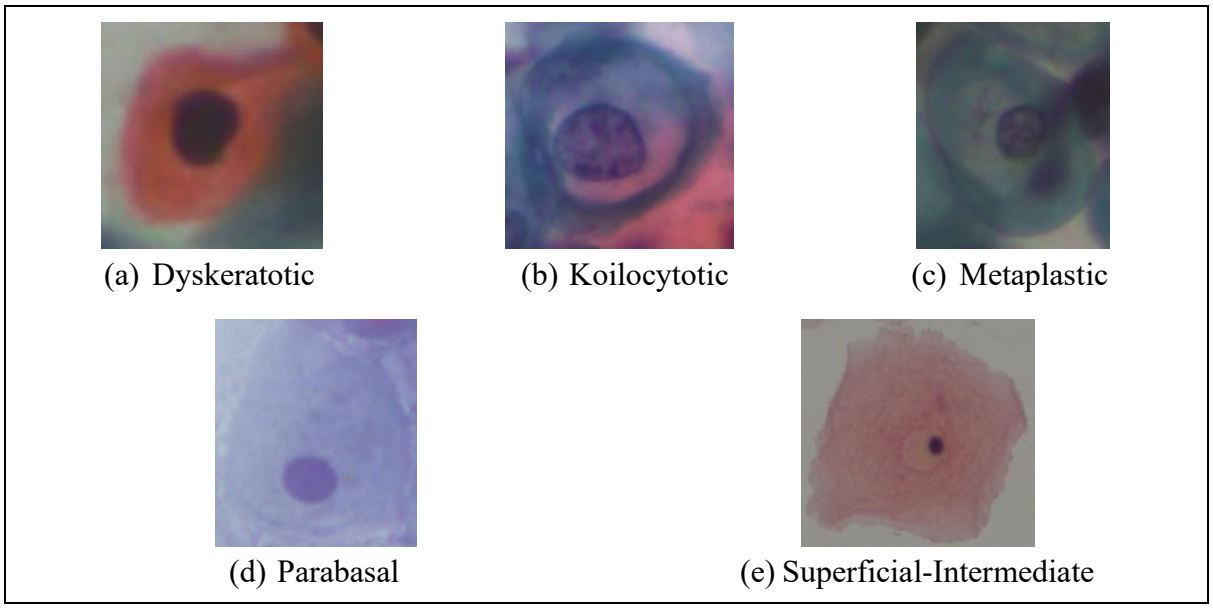


Figure 3.2. Example Cervical Pap Smear Images from SIPaKMeD Dataset

Besides that, the images are labelled and planned to be divided into training, validation, and testing subsets. The division of the dataset is helpful for training a comprehensive model without overfitting issues, whereby it is capable of classifying new and unseen cervical Pap smear images. Table 3.1 lists the size of the division for each subset and their purposes in CNN model training.

Table 3.1. Division of SIPaKMeD Dataset for Training, Validation, and Testing

Dataset Sub-sets	Division Size (%)	Explanation
Training	70	The largest portion of the dataset that will be used to train the CNN model and allow it to learn the features or patterns of each category of cells.
Validation	15	The validation set will be utilised during training to evaluate the model's performance on unseen data. It helps in fine-tuning and optimising the model by adjusting hyperparameters and preventing overfitting.
Testing	15	The testing set is used after training to assess the final performance of the model on unseen data. It utilises evaluation metrics, like accuracy, precision, recall, and F1-score in this project, to assess the model's capabilities in generalising to new data.

Note: Information for the explanation is adapted from (Gen, 2023)

Furthermore, several image preprocessing techniques are applied before training the CNN model to ensure consistency and enhance model performance. The techniques are listed and explained as follows:

- **Resizing**: Images are resized to 232 pixels using bilinear interpolation, standardising input dimensions while preserving structural details.
- **Central Cropping**: A 224×224 pixels central crop is applied to focus on the most relevant image regions, ensuring uniform input dimensions for ResNet-50.
- **Augmentation**: Techniques such as random rotation, random horizontal flipping, and random sharpness adjustment are used to improve generalisation and reduce overfitting.
- **Normalisation**: Standard normalisation is performed using mean = [0.485, 0.456, 0.406] and standard deviation = [0.229, 0.224, 0.225].

3.2.2. CNN Model Structure (ResNet-50)

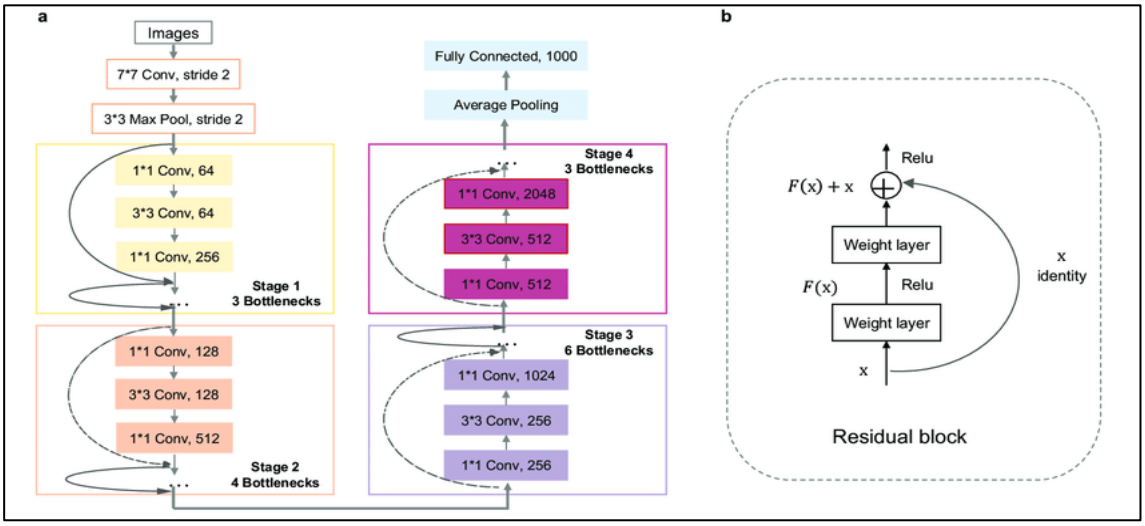


Figure 3.3. Example of ResNet-50 Architecture. (a) Layers of ResNet-50. (b) The Basic Structure of ResNet-50 Residual Block. (Yang et al., 2022)

To train the CNN model for this project, transfer learning with ResNet-50 is implemented to improve the model's performance for classifying cervical Pap smear images in the five predefined categories. ResNet-50 is a 50-layer network made up of convolutional layers, batch normalisation layers, pooling layers, and fully connected layers. Components of ResNet-50 Architecture as shown in Figure 3.3(a) are briefly explained in Table 3.2.

Table 3.2. Brief Explanation on the Components of ResNet-50 Architecture

Components	Explanation
Initial convolution layer	Use a 7x7 kernel to capture low-level features, such as edges and textures.
Initial max pooling layer	3x3 kernel is applied to retain the most important features by reducing the spatial dimensions (height and width) of the feature maps (Kundu, 2023).
Stage 1 – 4	Stages 1 – 4 altogether contain 16 bottlenecks, also known as residual blocks. Each block is equipped with several convolution layers of 1x1 and 3x3 kernel sizes. Each kernel has a number of filters that vary from 64, 128, 256, 524 to 1024. The 1x1 convolutions reduce and restore dimensions, while 3x3 convolutions act as bottlenecks with smaller input/output dimensions (Yang et al., 2022).

ResNet-50 is chosen for its efficient architecture, which incorporates 'residual blocks' (Figure 3.3(b)) that help to mitigate the vanishing gradient problem that may occur with deeper neural networks (Yang et al., 2022). Moreover, its lower parameter count (25.0M) compared to VGG-19 (143.7M) and fewer layers than ResNet-101 and ResNet-152 provide a balance between computational cost and classification accuracy, making it ideal for integration into the CerviScan system. This is because the lower parameter counts and fewer layers of ResNet-50 significantly reduce the memory usage and computational demand, thus ensuring faster processing and making it more suitable for deployment in low-resource settings.

Furthermore, the pre-trained ResNet-50 model, which was initially trained on large-scale image datasets, is fine-tuned and adapted to this image classification task. This approach not only reduces the training time but also enhances its performance on a smaller dataset, that is the SIPaKMeD dataset chosen for this project. The hyperparameters that will be used for fine-tuning the model are specified in Table 3.3.

Table 3.3. Specification of Key Hyperparameters

Hyperparameters	**Value**	**Explanation**
Optimiser	Adam (Adaptive Moment Estimation)	Adam is used with a learning rate of 0.0001 to ensure stable and efficient training of the ResNet-50 model during transfer learning. This low learning rate allows fine-tuning of pre-trained weights without disrupting them, while Adam's adaptive optimisation algorithm accelerates convergence and reduces sensitivity to hyperparameter choices (Agarwal, 2023).
Learning Rate	0.0001	
Batch Size	64	A batch size of 64 is chosen to balance accuracy and computational efficiency. It provides stable gradient estimates, reduces overfitting, and is compatible with most GPU configurations, ensuring efficient training without compromising model performance.
Epochs	50 (with early stopping)	The number of epochs is set to 50 so that the model can learn the features of input adequately. Moreover, early stopping is applied with a patience of 10 epochs, which

		means that the training will stop if the validation loss does not improve for 10 consecutive epochs. The application of early stopping will help to promote data generalisation while preventing overfitting (Kashyap, 2024).

3.2.3. Evaluation Metrics Used in CNN Model Training

For the multi-class classification task in this project, the performance of the CNN model are measured using several metrics during the training and testing phases. These metrics can help to evaluate the model's ability to correctly classify the cervical Pap smear images across the five categories.

1. **Confusion Matrix for Five (5) Categories Classification**

A confusion matrix is a useful tool for visualising the performance of a classification model. It is a square matrix (NxN matrix), where N is the number of classes or categories (Barathi, 2024). Moreover, the rows of the matrix represent the actual values while the columns represent the predicted values. In the case of this project, the selected dataset has five categories of cervical cell images. Therefore, it is called a multi-class classification, and the confusion matrix is a 5x5 matrix. Additionally, there are four key terms for the confusion matrix, including True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN). Table 3.4 explains the terms based on binary and multi-class classification.

Table 3.4. Explanation of Four Key Terms Based on Binary and Multi-class Classification

Key Terms	Explanation based on Binary Classification	Explanation based on Multi-class Classification
True Positive	Correctly predicts the positive class.	Correctly predicts a specific class.
False Positive	Incorrectly predict the negative class.	Incorrectly predicts a specific class. For example, 'Dyskeratotic' is predicted as

False Negative	Fails to predict the positive class.	'Koilocytotic'. Then, FP is Koilocytotic while FN is Dyskeratotic.
True Negative	Correctly predicts the negative class.	Correctly predicts all classes other than the one being evaluated.

2. Accuracy, Precision, Recall, and F1-Score

Next, based on the four metrics obtained from the confusion matrix (refer Table 3.4), other evaluation metrics like accuracy, precision, recall and F1-score can be calculated to evaluate the model's performance. Table 3.5 briefly explains the evaluation metrics and lists their respective formulae.

Table 3.5. Description, Purpose and Formulae of Utilised Evaluation Metrics

Metrics	Description	Purpose
Accuracy	Measure of how well the model performs, that is the ratio of correct predictions over a total number of predictions. $$\frac{TP + TN}{TP + TN + FP + FN}$$	Provides a basic understanding of model performance and effectiveness in classifying cervical Pap smear images.
Precision	The ratio of true positive predictions to the total number of positive predictions (regardless of correct or false positive predictions). $$\frac{TP}{TP + FP}$$	Important for minimising false positive predictions that would incorrectly classify healthy cells as abnormal or benign. Therefore, precision ensures that when a cell is predicted to be abnormal or benign, the classification is more likely to be accurate.
Recall	The ratio of true positive predictions to the total of actual positives. $$\frac{TP}{TP + FN}$$	It evaluates the model's capability to detect actual positive samples. In other words, a higher recall indicates that more positive samples are correctly detected.

F1-Score	Represents the harmonic mean of precision and recall, offering a balanced measure of both. $2 \times \frac{Recall \times Precision}{Precision + Recall}$	It is particularly useful in cases of uneven class distribution. It provides a more comprehensive evaluation of model performance to detect the cervical cell types.

Note: Information adapted from (Barathi, 2024; Gad & Skelton, 2024).

3.3. Rapid Application Development (RAD)

Rapid Application Development (RAD) is a software development methodology that emphasises iterative development, prototyping, and active user feedback to ensure efficient project delivery. This approach prioritises speed and flexibility, making it ideal for the CerviScan project, which involves both deep learning and web-based system development. RAD allows for continuous refinement of system components based on continuous feedback while ensuring the final product meets the specified requirements. The following sub-sections discusses the four phases of RAD methodology in detail and how they contribute to the planning and development of this project.

3.3.1. Outline Requirements

3.3.1.1. Project Description

Previously, identifying abnormalities in cervical Pap smear images required much manual effort from professional medical practitioners like pathologists or cytologists. This has led to the need for a more efficient, accurate and computer-aided method of image analysis. To address these challenges, CerviScan is developed as a tool for clinicians and researchers to analyse Pap smear images effectively.

CerviScan is a cervical cancer detection system designed to provide reliable classification of cervical Pap smear images into five categories, which are Dyskeratotic,

Koilocytotic, Metaplastic, Parabasal, and Superficial-Intermediate. A detailed explanation of each cell type can refer to Table 1.1. This project is developed as a solution to the challenges faced in cervical cancer detection by offering a web-based system that integrates image analysis and classification capabilities using a CNN model. The model is trained on the SIPaKMed dataset using transfer learning with ResNet-50 to enhance accuracy and efficiency in image classification tasks.

The CerviScan web-based system allows users to upload cervical Pap smear images for classification and generates the classification results. Users can also download a detailed analysis report in PDF format, which includes the classification result and confidence score. Furthermore, the system also provides features like user authentication, user profile management, analytics dashboard, classification record tracking, and patient management.

To access the CerviScan system, users require a stable internet connection, which enables them to use the platform anytime, anywhere, from devices like laptops or personal computers. User-friendliness and user experience of CerviScan are also prioritised to minimise any potential difficulties and hassle that may arise among the target users, primarily medical practitioners.

3.3.1.2. Requirements for CNN Model Training

The CNN model in CerviScan forms the core of its image classification capabilities. The following are the key requirements to train the CNN model effectively:

- **Cervical Pap Smear Images Dataset**: Utilise the SIPaKMed dataset, which contains labelled cervical Pap smear images with five categories.
- **Dataset Splitting**: Divide the dataset into three subsets, such as 70% for training, 15% for testing, and 15% for validation. The purpose of dataset splitting is to enhance the

performance of the model and prevent overfitting, whereby the model may fail to analyse new and unseen data.

- **Transfer Learning with ResNet-50**: Use the pre-trained ResNet-50 model for transfer learning to reduce training time and improve classification accuracy. Necessary fine-tuning of the model is required to achieve higher performance for the cervical Pap smear image classification tasks in this project.
- **Data Preprocessing**: Apply preprocessing techniques such as image resizing, normalisation, and augmentation to ensure consistency and improve the generalisation of the model.
- **Performance Metrics**: Metrics such as accuracy, precision, recall, and F1-score are used to evaluate the model's effectiveness and reliability.
- **Model Deployment Preparation**: Deploy the trained model into a format compatible with CerviScan, the web-based system, to ensure smooth integration and accurate predictions.

3.3.1.3. Requirements for CerviScan

CerviScan is designed to be an efficient and user-friendly tool for cervical cancer detection. The following are the requirements essential for developing the system:

- **User Login & Account Registration**: Allow users to register a new account and log in to their accounts to access the system.
- **User Profile Management**: Allow users to manage their profile and modify their personal information as needed.
- **Analytics Dashboard**: Allow users to view the summary and analytics charts of the uploaded images and export the charts in PDF report.

- **Image Upload and Classification**: Enable users to upload cervical Pap smear images for classification into five predefined categories, while assigning them to a specific patient.
- **View Classification Results**: Display classification results along with confidence score in a clear and simple format.
- **Download Report**: Provide the option to download the report in PDF format, which contains the associated patient's information as well as the classification results and confidence score of the uploaded image.
- **Classification Record Tracking**: Enable users to view and manage records of all classification results linked to the corresponding images and patients. Users can also delete unwanted results or modify the assigned patients as necessary.
- **Manage Patients**: Allow users to view and manage the list of registered patients by updating the patient information and removing selected patients when needed.

3.3.1.4. Software Requirements

The software requirements for the CerviScan system are essential for its functionality and operation. These requirements specify the necessary frameworks, platforms, and development tools that are used to build and deploy the web-based system. Some cloud-based GPU resources are also required for model training. Table 3.6 explains the list of software requirements for this project.

Table 3.6. List of Software Requirements

Software Requirements	Explanation
Backend Framework	Django, an open-source Python web framework, is used for backend development of the system. It handles functionalities like user authentication, database interactions, image uploading, and patient and

	results management, as well as integrate with the trained CNN model for classification tasks.
Front-end Framework	Bootstrap 5 is used for the front-end development of CerviScan. It is a popular open-source front-end framework that provides pre-designed, responsive UI components and layout structures, which allows faster and more responsive web development.
Deep Learning Framework	PyTorch is used for developing and deploying the Convolutional Neural Network (CNN). It also includes a library of pre-trained models, such as ResNet-50, which is utilised for transfer learning to improve classification accuracy in this project.
Kaggle	Kaggle provides free cloud-based GPUs that can be used for processing deep learning tasks, like CNN model training in this project.
Code Editor	Visual Studio Code is used to write and test the code for developing the web-based system as it supports Python
Web Browser	A modern web browser such as Google Chrome or Safari is required to access the CerviScan system for both development and deployment processes. These browsers are compatible with HTML5 and JavaScript, ensuring smooth interaction with the web-based system.
Database	DBeaver is a free database management tool that is used throughout the development of CerviScan to connect, manage and query the PostgreSQL database.

3.3.1.5. Hardware Requirements

Table 3.7 lists the hardware requirements for the development of CerviScan and CNN model training. These include the processors used for web-based system development, the cloud-based resources required for model training, and the devices that can access the web application.

Table 3.7. List of Hardware Requirements

Hardware Requirements	Explanation
Processor	An Intel Core i5 processor is used in this project. This processor is capable of handling the development of CerviScan.

Graphics	Intel HD Graphics 6000 used in this project is only sufficient for running the web application. Therefore, free GPU resources from Kaggle is utilised for model training tasks.
Devices	Laptops or desktops are required for developing and running the system. The devices should have internet connectivity and be able to run modern web browsers that support HTML5 and JavaScript.

3.3.2. User Design and Input

This phase offers an overview of the design and input elements that are used for the development of CerviScan. It includes Unified Modelling Language (UML) diagrams such as the use case, activity, and sequence diagrams to visually represent the system's workflow and user interactions. Next, mock-ups of the user interfaces are presented to illustrate the design and functionality of the CerviScan web-based system.

3.3.2.1. Use Case Diagram

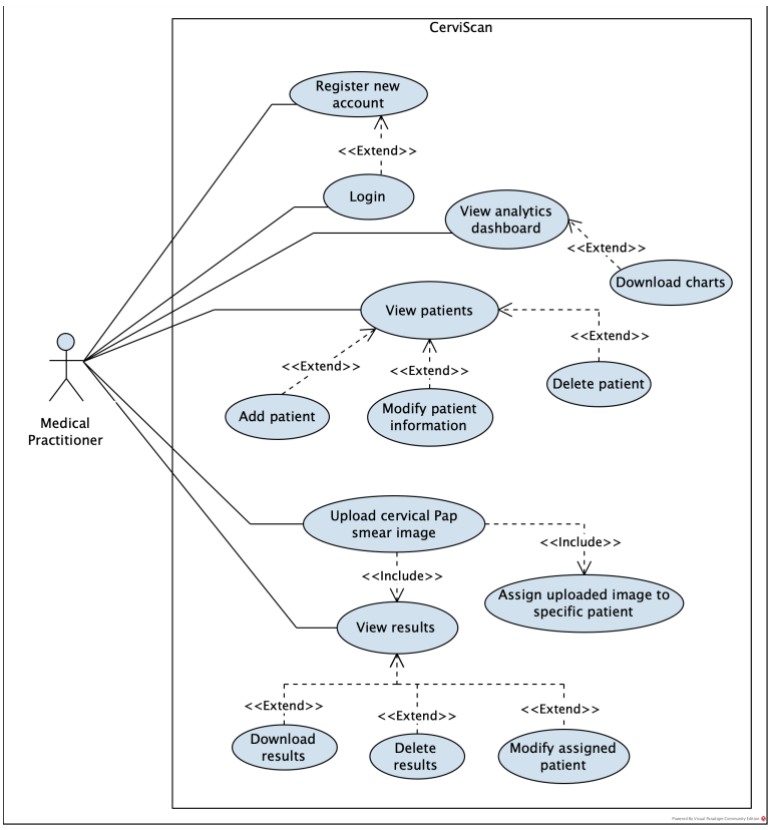


Figure 3.4. Use Case Diagram of CerviScan Web-based System

Figure 3.4 shows the overall view of the use case diagram for the CerviScan website. The actor involved in this use case diagram is a Medical Practitioner and there are a total of 14 use cases. Besides that, a base use case represents the core functionality of the system, while an include use case is a mandatory action that always occurs as part of the base use case, and an extended use case is an optional action that extends the behaviour of the base use case under specific conditions. The detailed use case specifications for each use case are outlined in the tables below.

3.3.2.1.1. Register New Account

Table 3.8. Use Case Specification for 'Register New Account'

Use Case	Register New Account
Actor	Medical Practitioner
Pre-condition(s)	The user is new and unregistered.
Main Flow	1. The user registers a new account. 2. A registration form will be displayed to the user. 3. The user enters the required information (username, full name, email password, confirm password) for new account registration. 4. The system validates the entered information. 5. Upon success, a new account is created for the user and a success registration message will be displayed.
Postcondition(s)	A new account is created and ready for login.
Alternative Flow	1. If validation of provided information fails, an error message will be displayed, and the user is prompted to return to Main Flow Step 3. 2. If the entered username or email is already registered, the user will be notified and redirected to Main Flow Step 3.

3.3.2.1.2. Login

Table 3.9. Use Case Specification for 'Login'

Use Case	Login
Actor	Medical Practitioner
Pre-condition(s)	The user has a registered account.
Main Flow	1. The user enters a registered username and password to log in. 2. The system validates the input. 3. Upon validation passes, the user is redirected to the system's main dashboard page.
Postcondition(s)	The user gains access to the system.
Alternative Flow	1. If validation of provided information fails, the system displays the error, and the user repeats Main Flow Step 1 onwards. 2. If the user forgets the password, a password reset option is provided.

3.3.2.1.3. View Analytics Dashboard

Table 3.10. Use Case Specification for 'View Analytics Dashboard'

Use Case	View Analytics Dashboard
Actor	Medical Practitioner
Pre-condition(s)	The user has logged into the system
Main Flow	1. The user navigates to the dashboard page. 2. The system displays the summary cards and various analytics charts. 3. The user may filter the analytics charts based on their preferred date range. Extends: a. Download charts
Postcondition(s)	The user views relevant analytical insights through summary cards and analytics charts.
Alternative Flow	N/A

3.3.2.1.4. Download Charts

Table 3.11. Use Case Specification for 'Download Charts'

Use Case	Download Charts
Actor	Medical Practitioner
Pre-condition(s)	1. The user has logged into the system. 2. The user has access to the dashboard page.
Main Flow	1. The user selects the 'Download Charts' option from the dashboard page. 2. The system presents a list of available charts. 3. The user selects one or more charts to download. 4. The selected charts are exported in PDF format and the download begins.
Postcondition(s)	The selected charts are successfully downloaded in PDF format.
Alternative Flow	N/A

3.3.2.1.5. View Patients

Table 3.12. Use Case Specification for 'View Patients'

Use Case	View Patients
Actor	Medical Practitioner
Pre-condition(s)	The user has logged into the system.
Main Flow	1. The user chooses to view all patients. 2. The list of registered patients is displayed to the user. 3. The user may select the option to add a patient, modify the patient's information, or delete the selected patient. Extends: a. Add Patient b. Modify Patient Information c. Delete Patient
Postcondition(s)	The list of patients is displayed.
Alternative Flow	N/A

3.3.2.1.6. Add Patient

Table 3.13. Use Case Specification for 'Add Patient'

Use Case	Add Patient
Actor	Medical Practitioner
Pre-condition(s)	The user is logged in and has access to the patients listing page.
Main Flow	1. The user selects to add a new patient. 2. The patient registration form is shown. 3. The user completes the form with the patient's information (full name, date of birth, identity card number, contact number, email and address). 4. The entered information is validated by the system. 5. Once validated, the system saves new patient entries and displays a success message to users.
Postcondition(s)	A new patient is added to the system.
Alternative Flow	1. If required fields are missing, the user will be prompted to complete them. 2. If the patient already exists, the system notifies the user.

3.3.2.1.7. Modify Patient Information

Table 3.14. Use Case Specification for 'Modify Patient Information'

Use Case	Modify Patient Information
Actor	Medical Practitioner
Pre-condition(s)	1. The user has logged into the system. 2. Specific patient does exist in the system.
Main Flow	1. The user selects a specific patient and chooses the edit feature. 2. The system displays the editable patient's information. 3. The user makes necessary changes. 4. The system validates the modification submitted. 5. The updated information is saved, and a success modification message is displayed to the user.
Postcondition(s)	Patient information is updated.
Alternative Flow	If validation fails, an error message is displayed, and the user is prompted to return Main Flow Step 2.

3.3.2.1.8. Delete Patient

Table 3.15. Use Case Specification for 'Delete Patient'

Use Case	Delete Patient
Actor	Medical Practitioner
Pre-condition(s)	1. The user has logged into the system. 2. Specific patient does exist in the system.
Main Flow	1. The user selects a specific patient and chooses the delete function. 2. The system requests confirmation from users. 3. The user confirms the deletion. 4. The patient record is removed, and a success deletion message is displayed.
Postcondition(s)	The selected patient is removed from the system.
Alternative Flowthe	If the user cancels deletion, the system returns to the patients listing page.

3.3.2.1.9. Upload Cervical Pap Smear Image

Table 3.16. Use Case Specification for 'Upload Cervical Pap Smear Image'

Use Case	Upload Cervical Pap Smear Image
Actor	Medical Practitioner
Pre-condition(s)	The user has logged into the system.
Main Flow	1. The user chooses and uploads a cervical Pap smear image. 2. The system validates the image format. 3. The user assigns the image to the specific patient. 4. Then, the user chooses to generate the classification results. 5. The system classifies the image using the integrated CNN model and displays the generated result. Includes: a. Assign Uploaded Image to Specific Patient b. View Results
Postcondition(s)	1. The image is uploaded and associated with the patient. 2. The classification result of the image is obtained.

Alternative Flow	1. If the file format is invalid, the system inhibits the upload process. 2. If the image fails to be uploaded, the system prompts to retry. 3. If the patient does not exist, the system prompts the user to add a new patient record.

3.3.2.1.10. View Results

Table 3.17. Use Case Specification for 'View Results'

Use Case	View Results
Actor	Medical Practitioner
Pre-condition(s)	1. The user has logged into the system. 2. Has records of results in the system.
Main Flow	1. The user chooses to view results. 2. The list of available results is displayed. 3. The user may select a specific result to view it in detail. 4. Besides that, the user may select the option to download the result, delete the result, or modify the assigned patient. Extends: a. Download Result b. Delete Result c. Modify Assigned Patient
Postcondition(s)	A list of results is displayed.
Alternative Flow	If no record of results is found, the system displays an appropriate message to indicate the empty list.

3.3.2.1.11. Download Results

Table 3.18. Use Case Specification for 'Download Results'

Use Case	Download Results
Actor	Medical Practitioner
Pre-condition(s)	1. The user has logged into the system. 2. The system contains a record of results.
Main Flow	1. The user selects a result and chooses the download feature. 2. The system processes the result into a PDF report. 3. The system initiates the download.
Postcondition(s)	Results are downloaded to the user's device.
Alternative Flow	If the download fails, the system prompts the user to retry.

3.3.2.1.12. Delete Results

Table 3.19. Use Case Specification for 'Delete Results'

Use Case	Delete Results
Actor	Medical Practitioner
Pre-condition(s)	1. The user has logged into the system. 2. The system contains a record of results.
Main Flow	1. The user selects a result and chooses the delete function. 2. The system requests deletion confirmation. 3. The user confirms the deletion. 4. The result is removed, and a success deletion message is displayed.
Postcondition(s)	If the user cancels deletion, the system returns to the results listing page.
Alternative Flow	N/A

3.3.2.1.13. Modify Assigned Patient

Table 3.20. Use Case Specification for ‘Modify Assigned Patient’

Use Case	Modify Assigned Patient
Actor	Medical Practitioner
Pre-condition(s)	The user has logged into the system
Main Flow	1. The user selects a result and chooses the modify patient function. 2. The patient selection interface is displayed. 3. The user chooses the new patient and saves the changes. 4. The system updates the reassignment of patient.
Postcondition(s)	The selected result is reassigned to a new patient.
Alternative Flow	If the new patient does not exist, the system prompts the user to add a new patient record.

3.3.2.2. Activity Diagram

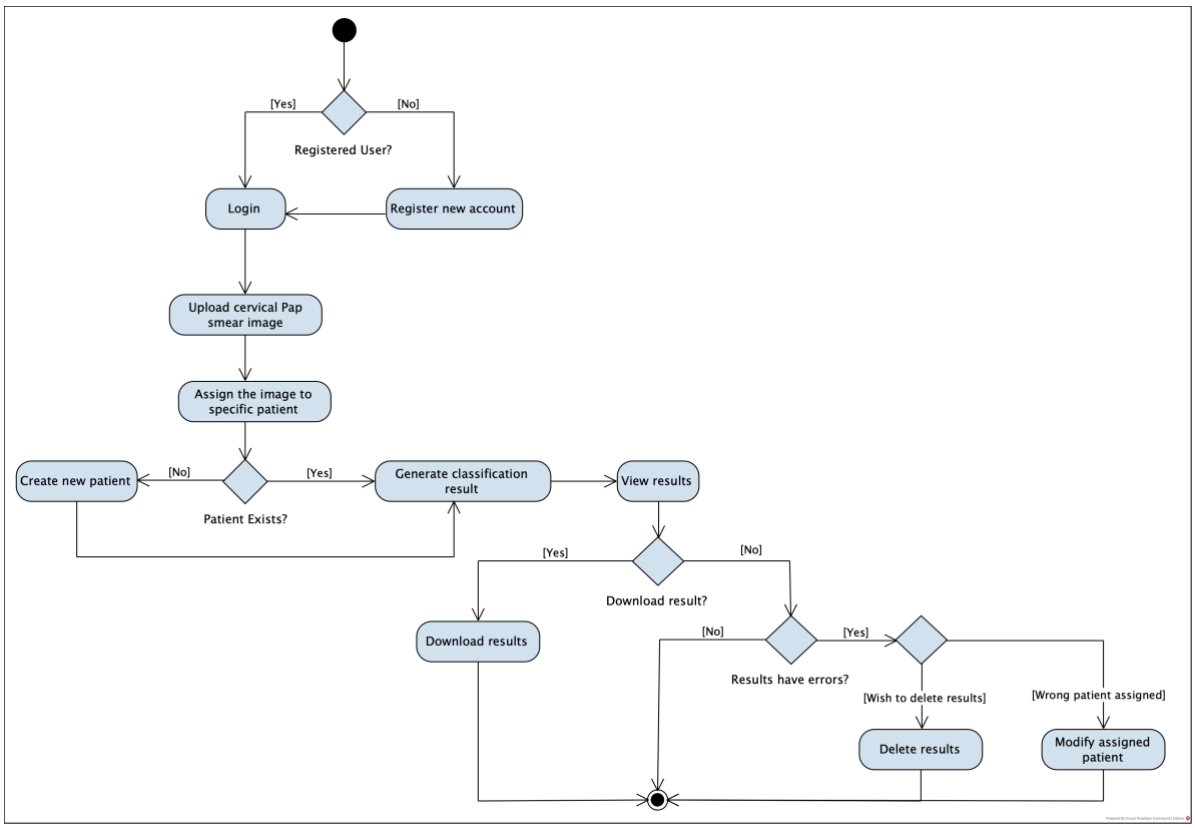


Figure 3.5. Activity Diagram of CerviScan

Figure 3.5 illustrates the activity diagram that explains the workflow of the CerviScan system. The flow is started with user authentication. If the user is new to the system, they are required to register a new account before being able to access it. Otherwise, they can directly log in to the system and proceed with the cervical Pap smear image classification task. First, the user uploads the relevant image and assigns it to a specific patient. If the patient does not already exist in the system, the user can create a new patient entry. After the image is assigned, the system will generate the classification results using the integrated and trained CNN model. Then, once the generation of results is completed, they will be displayed to the users. Additionally, the user is given several options to handle the results, including downloading the results as a PDF report, modifying the assigned patient, or deleting the results if unwanted errors arise.

3.3.2.3. Sequence Diagrams

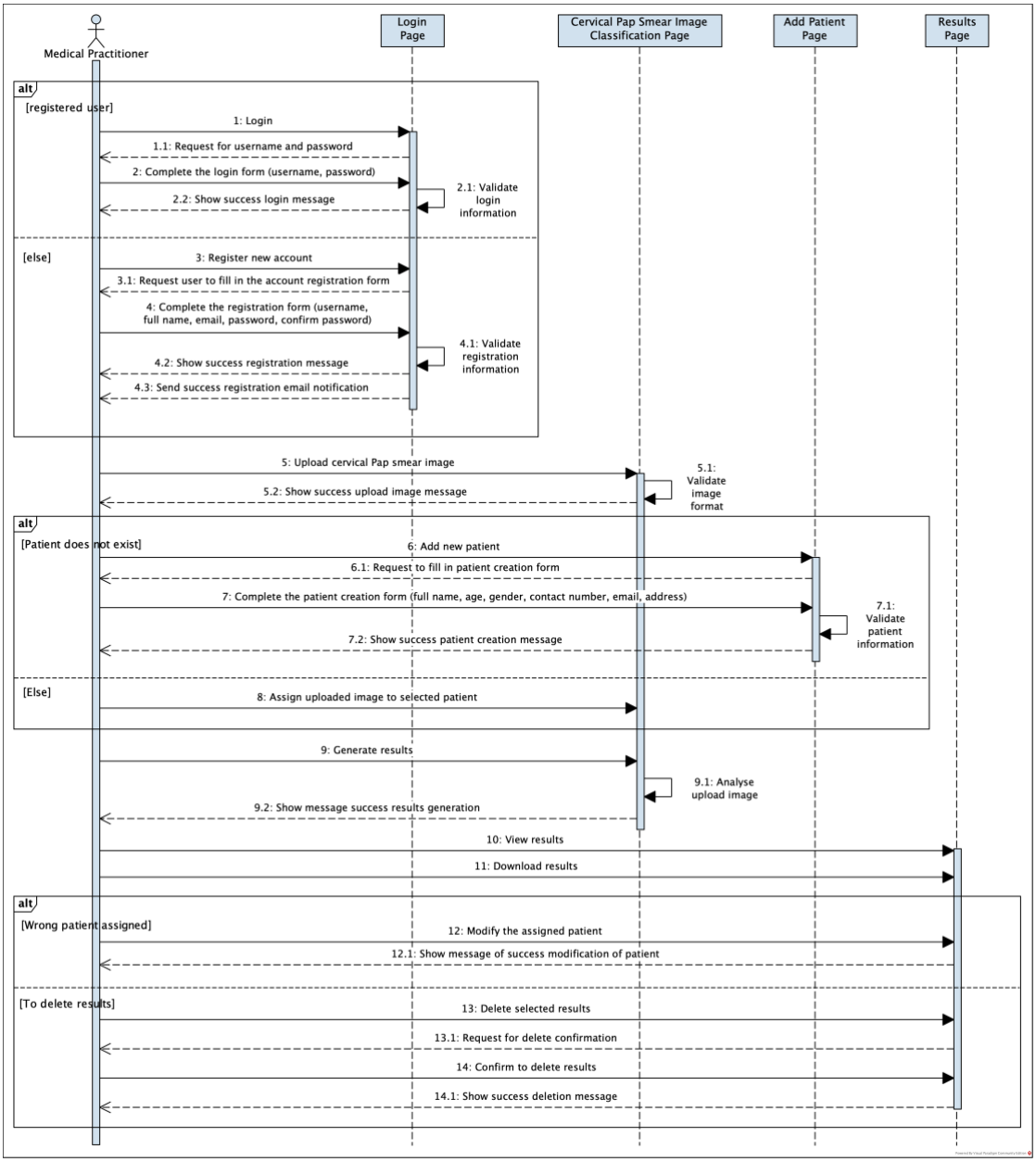


Figure 3.6. Sequence Diagram of CerviScan (Flow for Generating Cervical Pap Smear Image Classification Results)

The sequence diagram in Figure 3.6 shows the interactions between the user (a medical practitioner) and the CerviScan system across multiple modules for generating cervical Pap smear image classification results. First of all, the user either logs in to the system or registers a new account if they are a new user. Upon successful login or account registration, the user will proceed to the page that contains the functionality of cervical Pap smear image

classification. Then, the user uploads an image and assigns it to a specific patient. If there is no entry record for the patient, the user may navigate to the Add Patient page/module to create a new patient record. The detailed flow of patient management is further illustrated and explained in Figure 3.7 below.

Once the new patient is added to the database, they can be assigned to the uploaded image and the user can request to generate the classification results. After the results are generated, the user can view them on the results page. The user can choose to download the results in PDF format as needed. If errors are identified, the user can either delete the incorrect results or modify the patient assignment associated with the uploaded image.

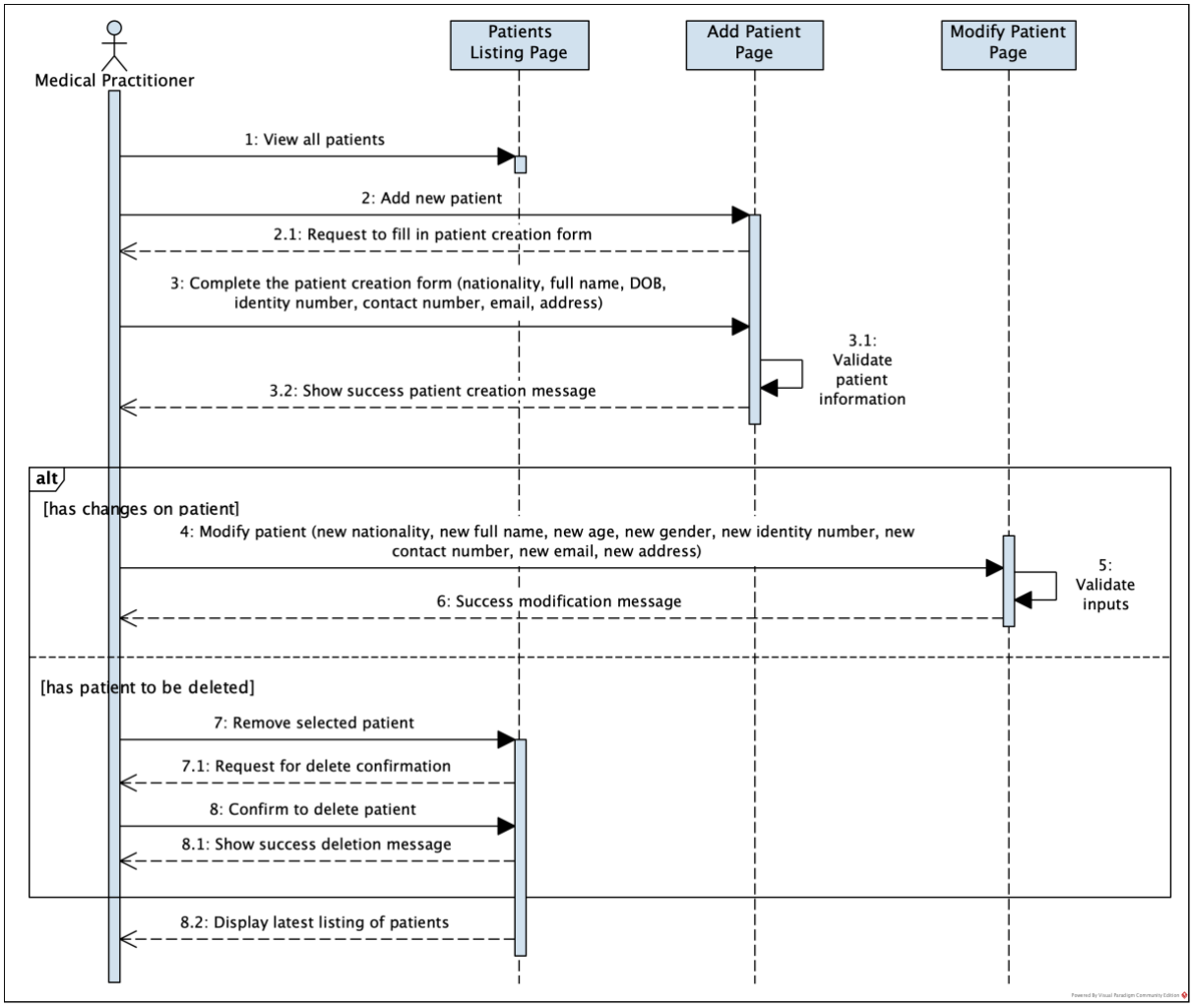


Figure 3.7. Sequence Diagram of CerviScan (Flow for Patients Management)

Figure 3.7 depicts the sequence diagram that illustrates the interaction of users with the CerviScan system for patients' management. A few modules or pages are involved in this interaction, such as the Patients Listing page, Add Patient page, and Modify Patient page. Firstly, the process begins with the user viewing all available patients on the Patients Listing page. If there is a new patient, the user may add the patient entry by filling up the patient creation form with details like nationality, full name, date of birth (DOB), identity number (IC or passport number), contact number, address, and email. Once the form is submitted, the system will validate the information and display a success message upon the successful addition of a new patient.

Besides that, if modification of patient information is needed, the user will be redirected to the Modify Patient page and update the relevant details. After submitting the changes, the system will validate the inputs and display a success message if the modification is passed. Lastly, the user is also given an option to delete unwanted patients from the listing. When the user chooses to delete a particular patient, the system will prompt for a deletion confirmation. Upon confirmation from the user, the patient will be deleted from the database and the latest list of patients will be displayed to the users.

3.3.2.4. Mock-Up for CerviScan's User Interfaces

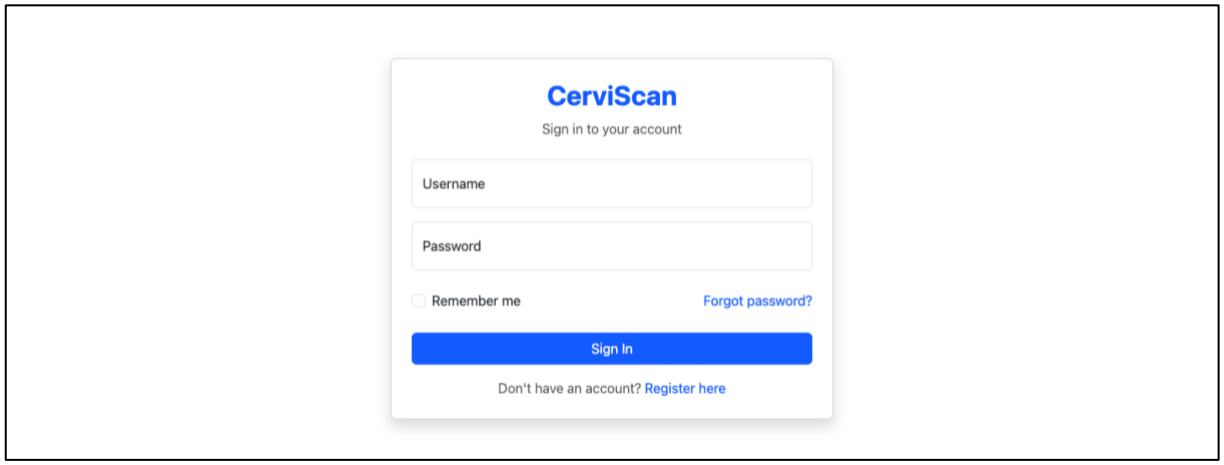


Figure 3.8. Mock-up for CerviScan's Login Page

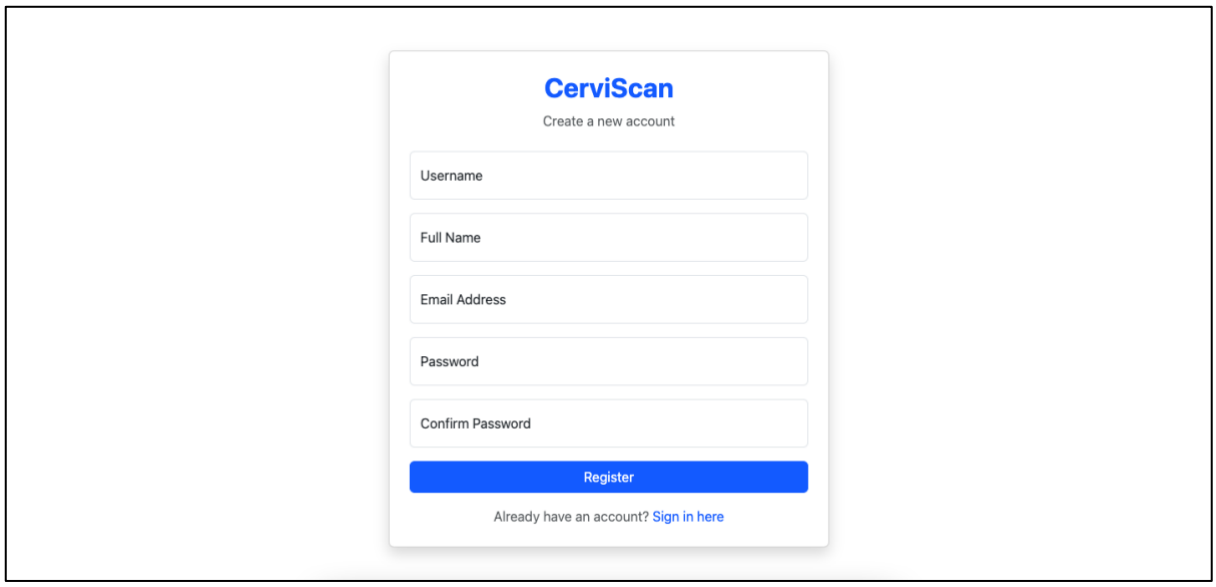


Figure 3.9. Mock-up for CerviScan's Account Registration Page

Figure 3.8 and Figure 3.9 show the login page and account registration page of CerviScan, respectively. To log into the website, the user is requested for their registered username and password whereas username, email, password, and confirm password are required to create a new account.

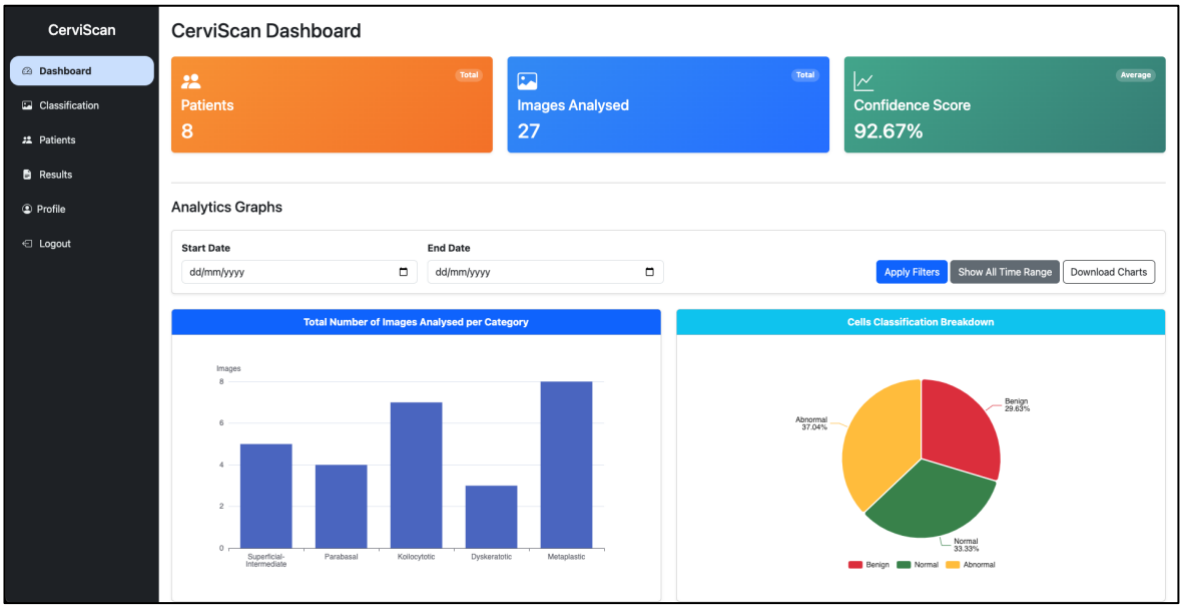


Figure 3.10. Mock-up for CerviScan's Analytics Dashboard Page

Figure 3.10 shows the mock-up of analytics dashboard page, which displays the summary cards and various analytics charts that can provide the analytical insights for the users. Moreover, the user may filter the data of analytics charts based on their preferred and export the charts in PDF report.

Figure 3.11. Mock-up for CerviScan's Cervical Pap Smear Image Classification Page

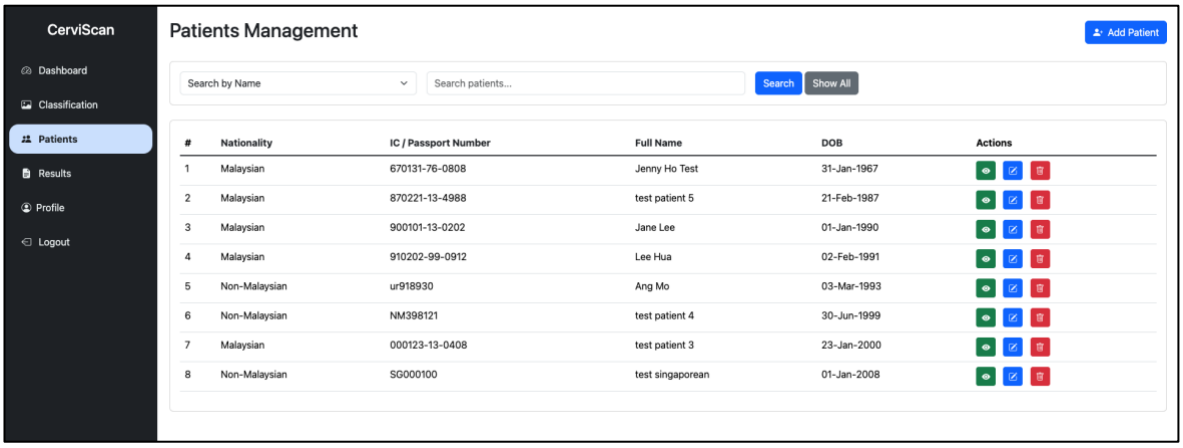


Figure 3.12. Mock-up for CerviScan's Patients Management Page

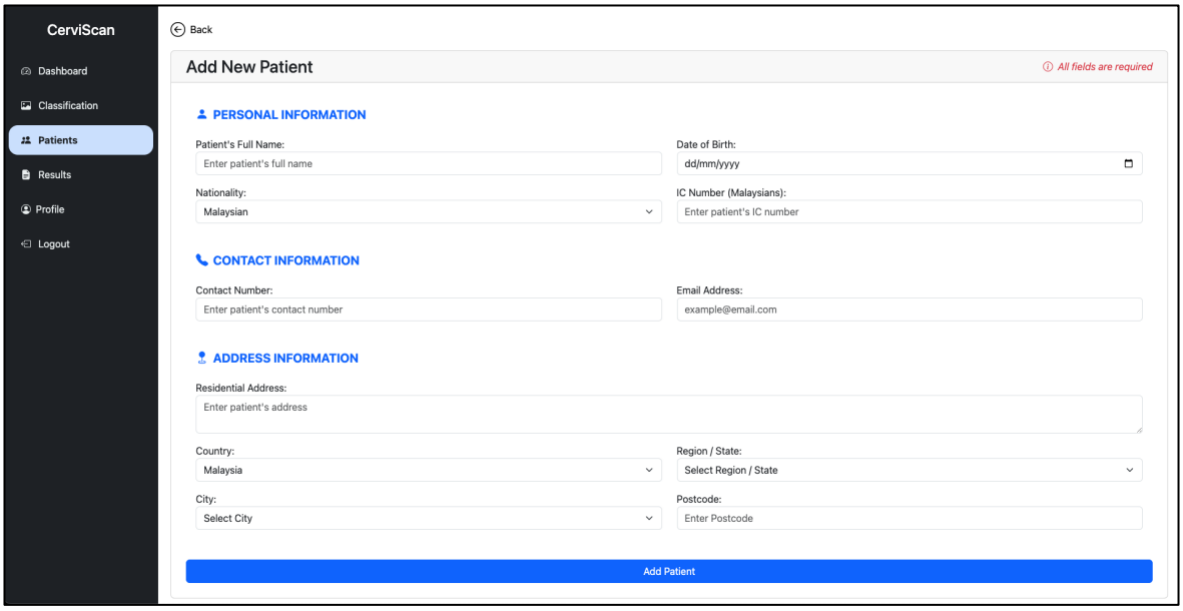


Figure 3.13. Mock-up for CerviScan's Add Patient Page

Next, Figure 3.11 depicts the interface mock-up for cervical Pap smear image classification tasks. On this page, the user can upload one Pap smear image and assign it to a specific patient before generating the result. If the patient does not exist in the selection list, the user may click on the 'Add new patient here' to add a new patient entry into the system. After results are generated, the user will be redirected to a detailed result page as shown in Figure 3.15.

The mock-up for patients' management page is shown in Figure 3.12. This page displays the list of patients registered by the user in the system. The list of patients will be distinct for every user who uses the system, assuming that they represent different clinics or hospitals. Next, features like viewing the patient's information in detail, updating the patient's details, deleting selected patient, and adding new patient, are provided on this page too. For example, Figure 3.13 illustrates the example of the add patient page, which will be redirected when the user clicks the 'Add Patient' button.

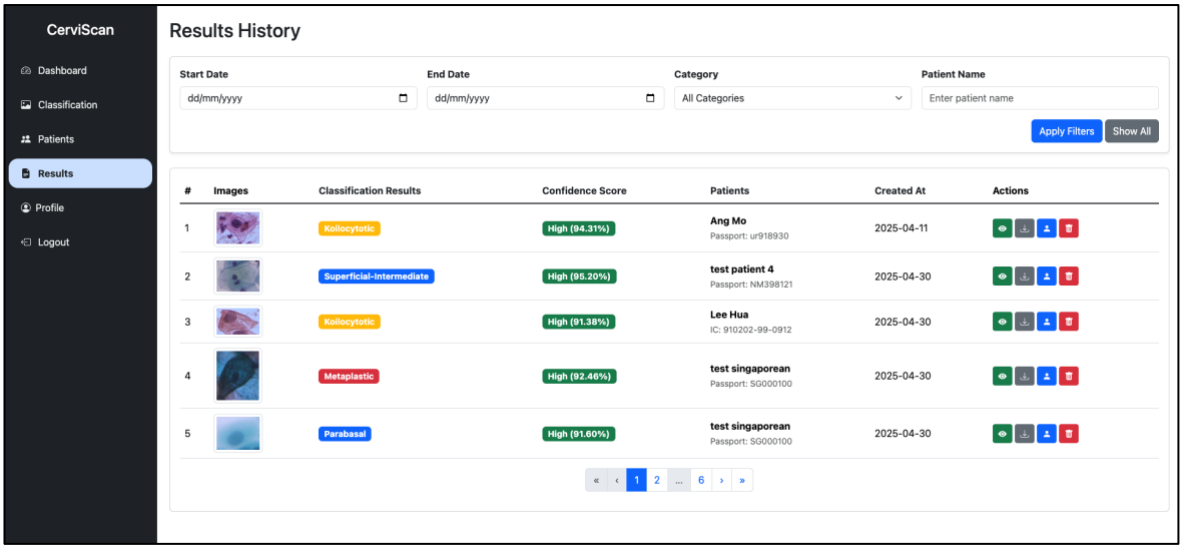


Figure 3.14. Mock-up for CerviScan's Results History Page

Figure 3.14 shows the results history page, where the list of all results generated will be displayed here. Moreover, functionalities, such as viewing detailed results, downloading results in PDF format, modifying the patient assignment, as well as deleting a specific result, are shown in the actions column. So, to view the detailed results as depicted in Figure 3.15, the user may click on the green 'Eye' button. Next, the 'Reassign Patient' modal in Figure 3.16 will appear when the user clicks on the blue 'Reassign Patient' button in either Figure 3.14 or Figure 3.15.

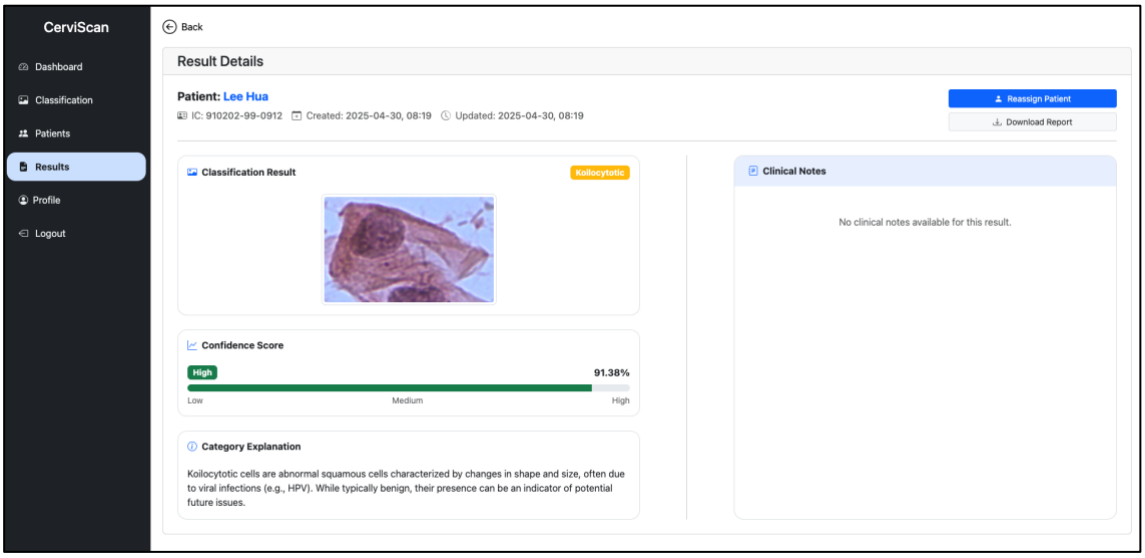


Figure 3.15. Mock-up for CerviScan's Detailed Result View

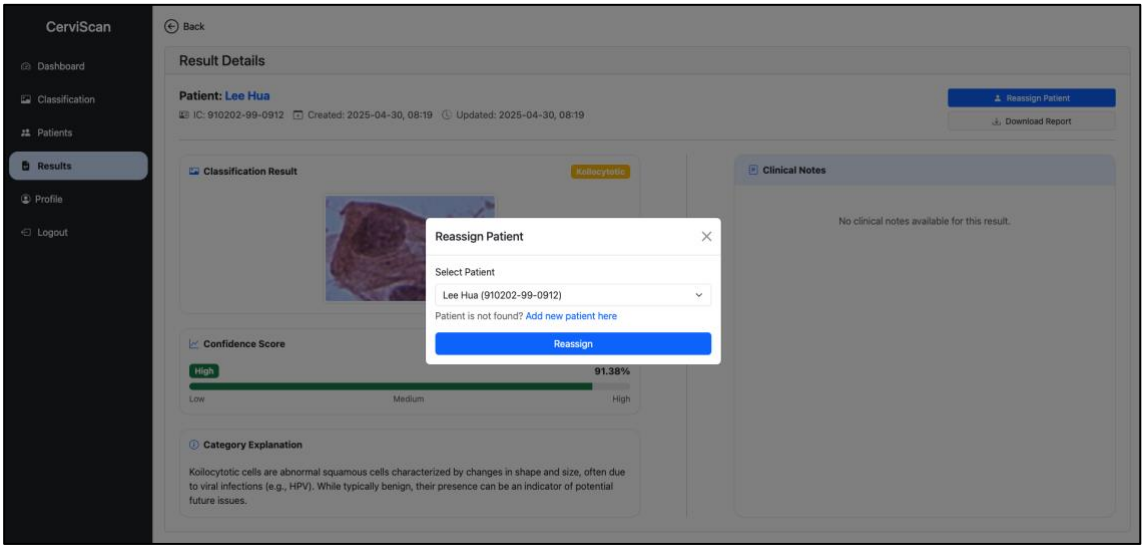


Figure 3.16. Mock-up for CerviScan's Reassign Patient Modal

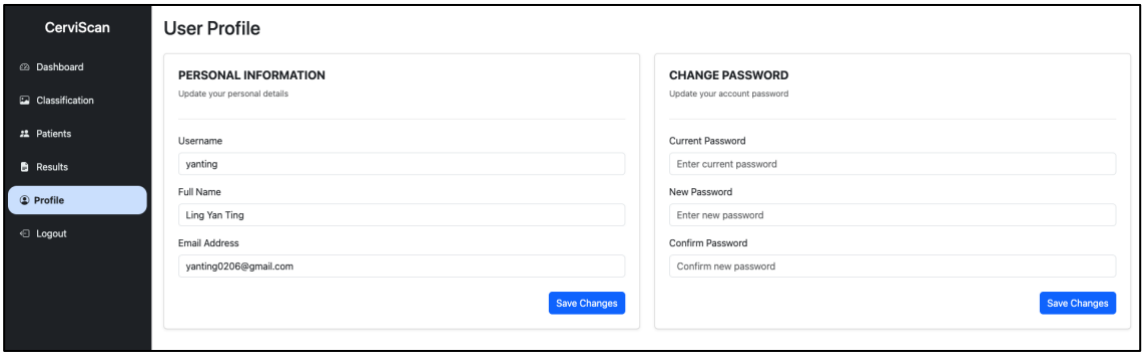


Figure 3.17. Mock-up for CerviScan's User Profile Page

Figure 3.17 illustrates the mock-up for CerviScan's User Profile page, where the user can view their personal information, edit the information and change their password as needed. For privacy and safety purposes, the current password of the user will not be displayed on the website.

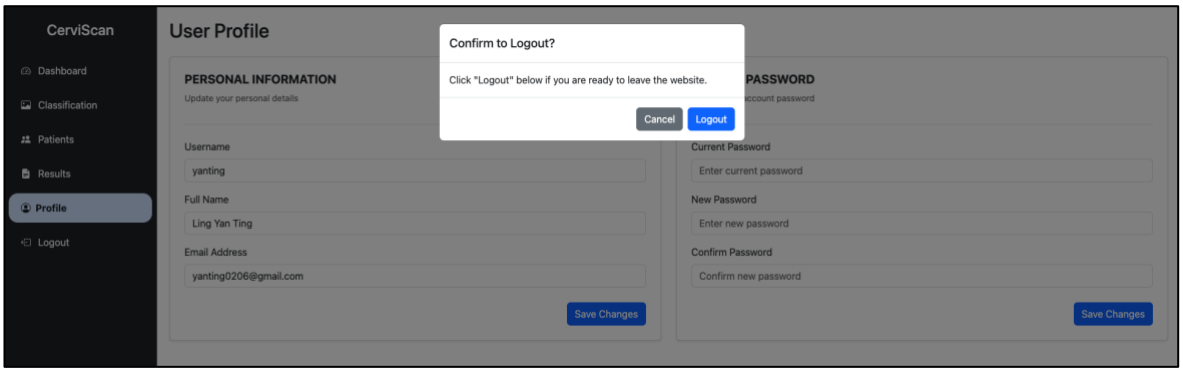


Figure 3.18. Mock-up for CerviScan's Logout Modal

Figure 3.18 shows the 'Logout' modal that will be shown when the user clicks on the 'Logout' link on the sidebar. Once the user confirms to log out, their session at the website will be terminated.

3.3.3. Development/Construction

The development phase of CerviScan involve sthe front-end and back-end development of the web-based system. The back-end of the system is developed using Django, and the trained CNN model is integrated to handle the cervical Pap smear image classification task. PostgreSQL is used to manage the database and store data like the user's personal information or login credentials, the list of patients and the record of all classification results. On the other hand, the front-end is developed using HTML, CSS, JavaScript and supported by Bootstrap 5, which offers pre-defined UI components and layouts while allowing responsive web development to enhance user experience across different devices and screen sizes. Since early front-end coding based on the designed prototype has been completed in Phase 2: User Design

and Input, this phase focuses on the back-end functionalities and improving the system's user interfaces.

Lastly, iterative testing is conducted throughout this phase to verify the system's functionality. For example, unit testing to ensure individual components, such as database operations, work correctly, and integration testing to confirm the smooth interaction between the CNN model, back-end, and front-end modules.

3.3.4. Finalisation

The finalisation phase of CerviScan focuses on preparing the system for deployment, ensuring its usability, and conducting comprehensive testing. The fully integrated system undergoes thorough functional and performance testing to ensure it works well and meets the project's goals. Moreover, deployment preparations, like setting up the cloud environment for hosting the application and database, are carried out.

Furthermore, documentation, such as user manuals and test cases, are also created during this phase to support both users and future developers. The user manual guides the users through key features of CerviScan, such as uploading Pap smear images, generating classification results, downloading PDF reports, and managing patient records. Meanwhile, the test cases verify the system's functionality, ensuring that each feature works as expected under various scenarios, and any issues are identified and addressed before deployment.

3.4. Summary

This chapter has discussed the overall research process and application of Rapid Application Development (RAD) methodology in the development of CerviScan. It highlights the structure of the CNN model built with transfer learning of ResNet-50 and evaluation metrics for cervical Pap smear image classification, as well as the specification of project

requirements, and the mock-up of CerviScan's user interfaces. Additionally, it also briefly explains the development and finalisation phases to be completed in Final Year Project II. For example, the development phase of CerviScan focuses on training the CNN model using transfer learning of ResNet-50 with PyTorch, along with the development of both the back-end with Django and PostgreSQL and the front-end using HTML, CSS, JavaScript, and Bootstrap 5. The finalisation phase ensures the system is ready for deployment through comprehensive testing, system integration, and the creation of supporting documentation, including user manuals and test cases for future maintenance.

CHAPTER 4: IMPLEMENTATION

4.1. Introduction

This chapter covers the implementation process of the CerviScan system, which consists of two main components. For instance, a Convolutional Neural Network (CNN) model for cervical Pap smear image classification and a web-based system for user interaction, image analysis, and result presentation. It begins with the installation and configuration steps for setting up the development environments, such as using PyTorch and related libraries to build the CNN model on Kaggle's cloud GPU environment. Meanwhile, the web-based system is developed with Django for the backend, Bootstrap 5 for the front-end, and PostgreSQL for database management. Next, the chapter further explains the CNN model development process, including dataset preparation, model training, validation, and evaluation. Finally, the modules and functionalities of the web-based system are discussed, along with the integration of the trained CNN model into the system for real-time image analysis and result presentation.

4.2. Development Environment Setup

4.2.1. CNN Model Environment Setup

4.2.1.1. Kaggle

Kaggle (https://www.kaggle.com/) is a popular online platform that hosts data science projects and machine learning competitions. It provides a collaborative environment where users can access public datasets, develop models in interactive notebooks, and share their work with the community. Kaggle also supports Python and a wide range of machine learning frameworks, making it a valuable resource for both beginners and experienced data scientists.

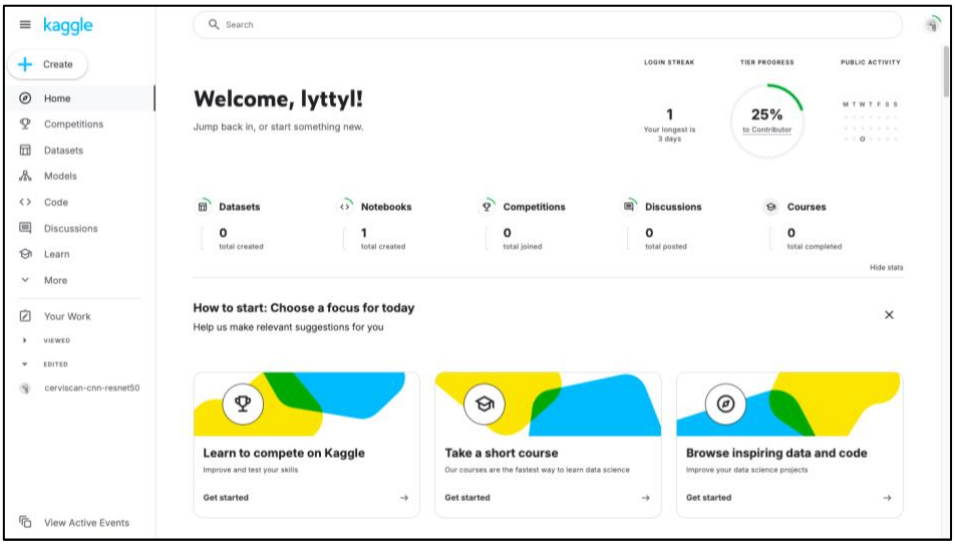


Figure 4.1. Kaggle's Home Page

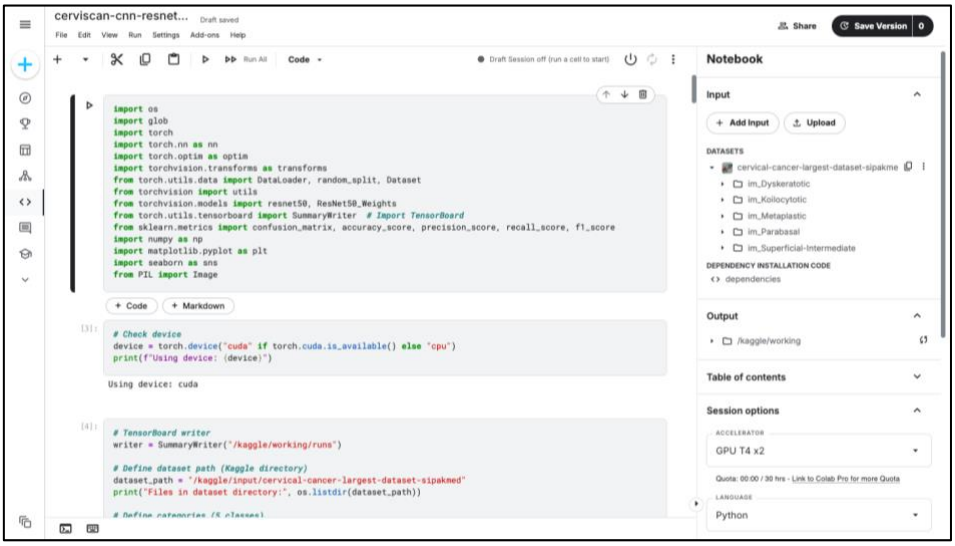


Figure 4.2. Example Page for Kaggle's Notebook

For this project, Kaggle has been chosen as the development environment for the CNN model for several key reasons. Firstly, Kaggle offers free access to cloud GPUs for up to 30 hours per week. Training deep learning models, such as the CNN model used for cervical Pap smear image classification, is computationally intensive. Normally, this would mean the need to use a high-performance local machine with a dedicated GPU, which can be expensive to purchase and maintain. By using Kaggle's cloud GPU, the project avoids these costs while still allowing the model to be trained efficiently and effectively.

Secondly, the SIPaKMeD dataset required for this project is available directly on Kaggle as an open-source resource. This means that the dataset can be linked directly within the Kaggle environment without needing the extra steps of downloading and uploading, thus simplifying the process of data preparation.

Thirdly, Kaggle provides an integrated notebook interface where all coding, testing, and documentation can be performed in one place. This notebook environment supports real-time collaboration and version control, making it easier to experiment with different model architectures and track changes during model development.

4.2.1.2. Installation of Libraries and Dependencies

Several Python libraries and dependencies are required to develop and train the CNN model for cervical Pap smear image classification. Each library serves a specific purpose to support various parts of the machine learning workflow, from model construction to evaluation and visualisation.

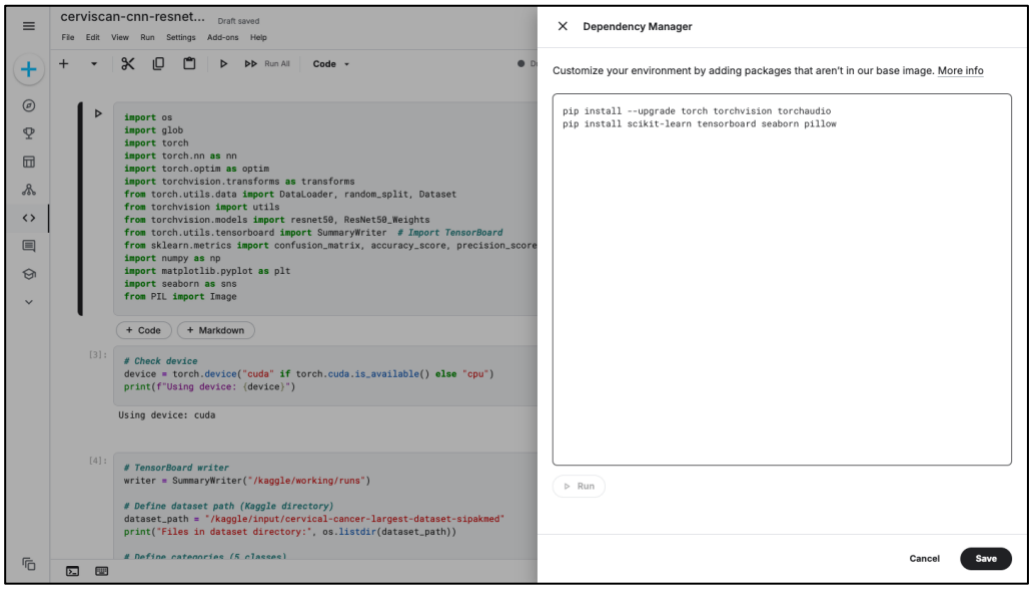


Figure 4.3. Installation of Needed Libraries and Dependencies in Kaggle Notebook

1. **PyTorch**: An open-source deep learning framework used to build and train neural networks. In this project, PyTorch is used to construct the CNN model, manage the training process, and handle forward and backward propagation.
2. **TensorBoard**: A visualisation tool originally for TensorFlow but also supports PyTorch through *"torch.utils.tensorboard"*. It is used to monitor training and validation performance by showing metrics like loss and accuracy over time.
3. **scikit-learn (sklearn)**: A machine learning library that provides tools for data preprocessing and model evaluation. It is used to generate metrics such as confusion matrices, accuracy, precision, recall and F1-score to assess model performance.
4. **Seaborn**: A visualisation library built on top of Matplotlib. It is used to improve the appearance of statistical plots, such as confusion matrices, making results clearer and easier to interpret.

4.2.2. Web-Based System Environment Setup

4.2.2.1. Visual Studio Code (VS Code)

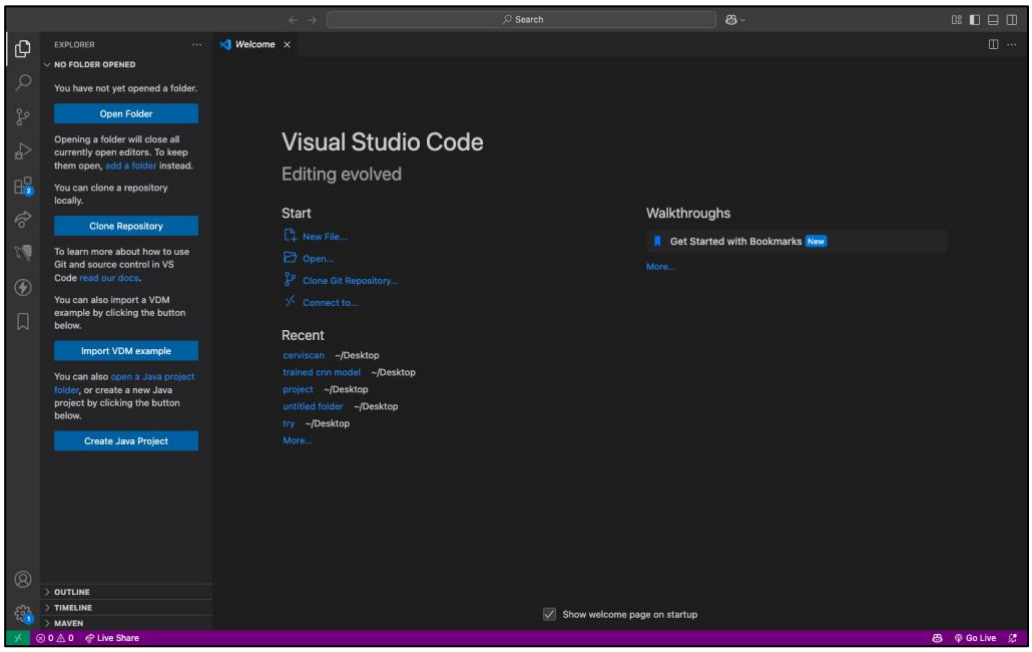


Figure 4.4. Visual Studio Code

VS Code (https://code.visualstudio.com/) is a lightweight, open-source code editor developed by Microsoft. It supports many programming languages and comes with features like syntax highlighting, debugging tools, extensions, and Git integration. In this project, VS Code is used as the main code editor for developing and managing the web-based system using Django, HTML, CSS, JavaScript and Bootstrap 5.

4.2.2.2. Django

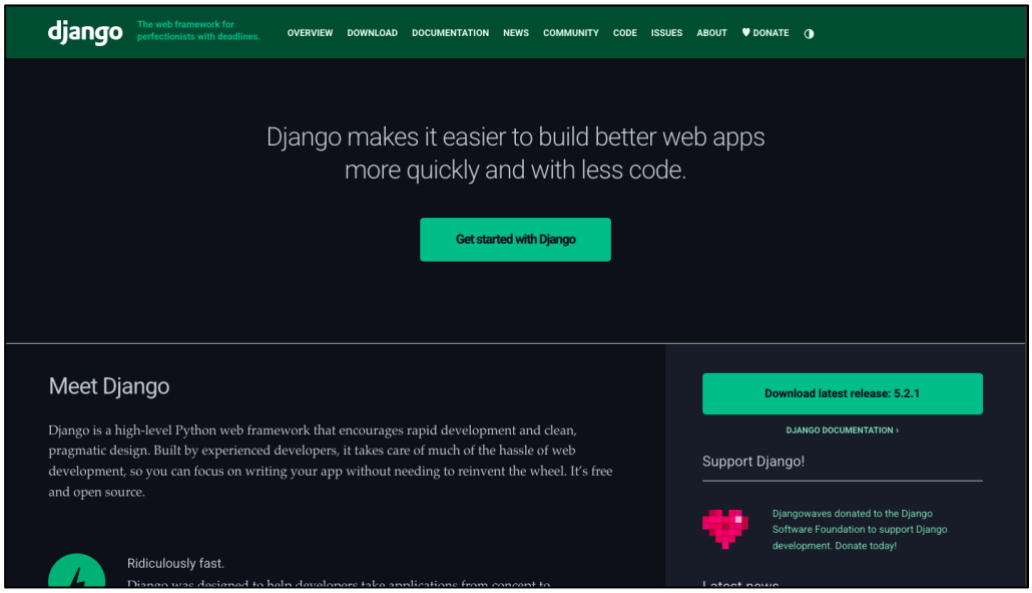


Figure 4.5. Screenshot of Django's official website

Django (https://www.djangoproject.com/) is a high-level Python web framework that encourages rapid development and clean, maintainable code. It includes built-in features such as database integration and an admin interface. In this project, Django is used to build the backend of the web application, handling user interactions, image upload, patients management, result display, and integration with the trained CNN model.

4.2.2.3. Bootstrap 5

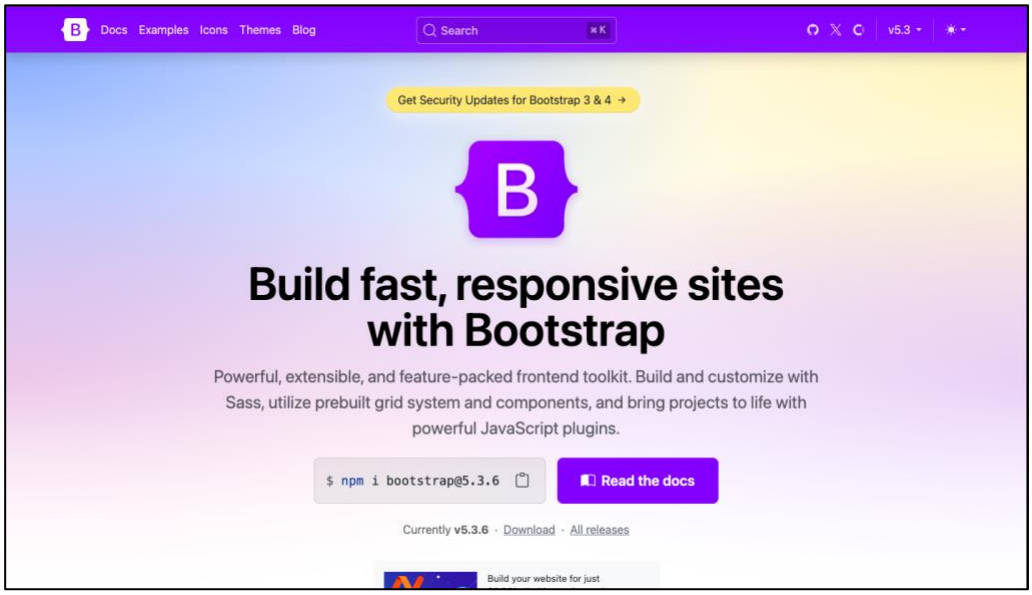


Figure 4.6. Screenshot of Bootstrap 5's official website

Bootstrap 5 (https://getbootstrap.com/) is a widely used open-source frontend framework that helps in creating responsive and user-friendly web interfaces. It works alongside HTML, CSS, and JavaScript, providing ready-to-use components such as buttons, forms, modals, and navigation bars. In this project, Bootstrap 5 is integrated into the Django templates to design the layout and styling of the web-based system. By using Bootstrap 5 together with custom HTML, CSS, and JavaScript, the user interface becomes more interactive, consistent, and suitable for different screen sizes and devices.

4.2.2.4. DBeaver

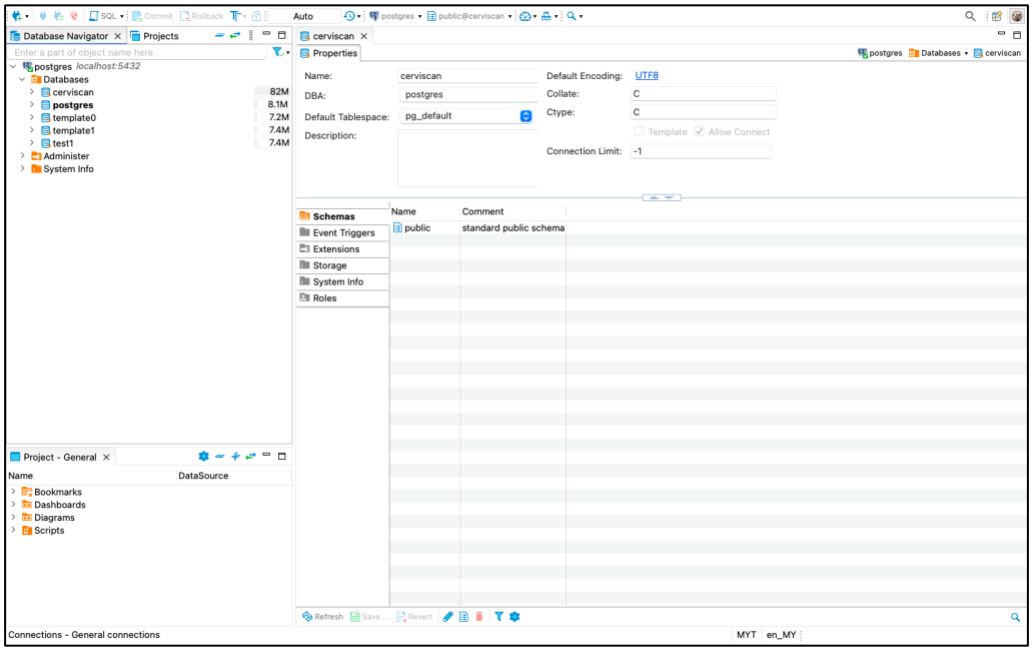


Figure 4.7. DBeaver

DBeaver (https://dbeaver.io/) is a free, universal database management tool that supports a wide range of databases, including PostgreSQL. It provides a graphical interface to manage, view, and query databases easily. In this project, DBeaver is used to manage the PostgreSQL database, allowing easy inspection of stored patient data, image records, prediction results, and registered users.

4.2.2.5. Installation of Python Libraries and Dependencies

To manage and install the required Python packages for the CerviScan project, a *"requirements.txt"* file is created. This file lists all the necessary libraries and frameworks for the development of the web-based system and integration of the trained CNN model (Figure 4.8). Then, these dependencies are installed using the command shown in Figure 4.9.

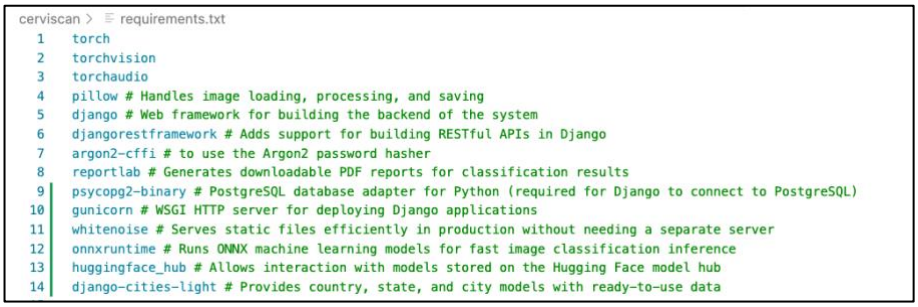


Figure 4.8. Screenshot of *"requirements.txt"* file

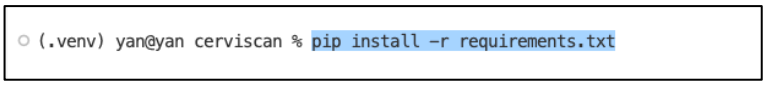


Figure 4.9. Command to install the dependencies listed in the *"requirements.txt"* file

4.2.2.6. Creating Django Project and App

To begin the web-based system development, a Django project, named *"cerviscan"*, is created to serve as the foundation. This is done using the command shown in Figure 4.10. This then generates the main project directory containing configuration files such as *"settings.py"*, *"urls.*py*"*, and *"wsgi.py"* (Figure 4.11).

Figure 4.10. Command to Create Django Project

Figure 4.11. A list of files that are automatically created in the project directory

Then, inside the project, a single app named *"cerviscanApp"* is created to handle all the system's functionalities, including user authentication, image uploads, patients' management, CNN model integration, result display, and dashboard analytics. The app is

created using the command shown in Figure 4.12. Next, the app, together with the *"djangorestframework (rest_framework)"* and *"django-cities-light (cities_light)"*, are added to the *"INSTALLED_APPS"* list in *"settings.py"* to activate them within the Django project (Figure 4.13).

```
(.venv) yan@yan cerviscan % python manage.py startapp cerviscanApp
```

Figure 4.12. Command to create a Django app

```
cerviscan > cerviscan > settings.py > ...
31
32
33  # Application definition
34
35  INSTALLED_APPS = [
36      "django.contrib.admin",
37      "django.contrib.auth",
38      "django.contrib.contenttypes",
39      "django.contrib.sessions",
40      "django.contrib.messages",
41      "django.contrib.staticfiles",
42      "cerviscanApp",
43      "rest_framework",
44      "cities_light",
45  ]
46
```

Figure 4.13. Screenshot of *"INSTALLED_APPS"* in *"settings.py"*

4.2.2.7. Email Configuration Setup

To enable email functionalities such as password reset requests and notifications, the Django application is configured to use the Simple Mail Transfer Protocol (SMTP) via Gmail. This setup is defined in the project's *"settings.py"* file using the following parameters as shown in Figure 4.14.

```
cerviscan > cerviscan > settings.py > ...
143  # Email
144
145  EMAIL_BACKEND = "django.core.mail.backends.smtp.EmailBackend"
146  EMAIL_HOST = "smtp.gmail.com"
147  EMAIL_PORT = 587
148  EMAIL_USE_TLS = True
149  EMAIL_HOST_USER = "yannotian@gmail.com"
150  EMAIL_HOST_PASSWORD = "aity cffr aaks jcwl"
151  DEFAULT_FROM_EMAIL = "CerviScan <yannotian@gmail.com>"
152
```

Figure 4.14. Screenshot of the email configuration in *"settings.py"*

4.2.2.8. Setting up PostgreSQL Connection

For database management, PostgreSQL is used to store registered user accounts, patient information, image analysis records, and other relevant data. A local PostgreSQL server is set up, and tools like DBeaver can be used to manage the database through a graphical interface. In the *"settings.py"* file of the Django project, the database configuration is updated as in Figure 4.15. Once configured, the commands shown in Figure 4.16 and Figure 4.17 are used sequentially to apply the database schema based on the models defined in *"cerviscanApp"*. After completing the setup of the Django project, creating the cerviscanApp, and configuring the PostgreSQL database, the development server can be started using the command in Figure 4.18.

Figure 4.15. Screenshot of *"DATABASES"* in *"settings.py"*

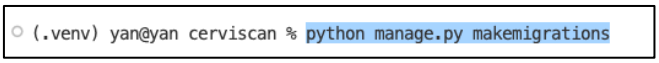


Figure 4.16. Command to generate migration files from model changes

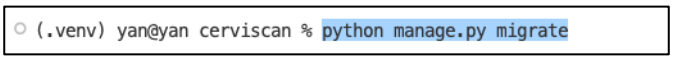


Figure 4.17. Command to apply migrations to the database

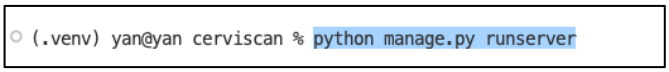


Figure 4.18. Command to run the development server

4.2.2.9. Installation of Front-end Libraries and Dependencies

For the front-end, several libraries and frameworks are included via CDN in the HTML templates to enhance the UI and interactivity of the web-based system. Figure 4.19 shows the key client-side dependencies used in this project. Examples of the dependencies include Bootstrap 5, Echarts, SweetAlert2, Html2Canvas and jsPDF.

Figure 4.19. Client-side Dependencies Used

4.3. CNN Model Implementation

This section explains the development of a cervical Pap smear image classification model using the SIPaKMeD dataset. The model is implemented in PyTorch using a transfer learning approach with the pre-trained ResNet-50 model. The implementation encompasses several processes like data preprocessing, model fine-tuning, training, and performance evaluation.

4.3.1. Dataset Configuration and Transformations

The dataset used in this project is the SIPaKMeD dataset, which contains cropped images of cervical cells categorised into five types, such as Parabasal, Dyskeratotic,

Metaplastic, Superficial-Intermediate, and Koilocytotic. These categories represent normal, abnormal, and benign cell types, which are crucial for early detection of precancerous changes.

```
# Define dataset path (Kaggle directory)
dataset_path = "/kaggle/input/cervical-cancer-largest-dataset-sipakmed"
print("Files in dataset directory:", os.listdir(dataset_path))

# Define categories (5 classes)
categories = ["im_Parabasal", "im_Dyskeratotic", "im_Metaplastic", "im_Superficial-Intermediate", "im_Koilocytotic"]
```

Figure 4.20. Code to set dataset path and define class labels

To enhance model generalisation and reduce overfitting, different image transformations are applied. These include resizing, cropping, horizontal flipping, rotation, and sharpness adjustment for the training set, while the validation and test sets are standardised using consistent resizing and normalisation.

```
# Image transformations
train_transform = transforms.Compose([
    transforms.Resize(232), # default interpolation mode is bilinear
    transforms.CenterCrop(224),
    transforms.RandomHorizontalFlip(),
    transforms.RandomRotation(10),
    transforms.RandomAdjustSharpness(sharpness_factor=2),
    transforms.ToTensor(),
    transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])
])

val_test_transform = transforms.Compose([
    transforms.Resize(232),
    transforms.CenterCrop(224),
    transforms.ToTensor(),
    transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])
])
```

Figure 4.21. Code to define image transformations for training, testing, and validation sets

4.3.2. Custom Dataset Loader

A custom PyTorch *"Dataset"* class named *"SIPaKMedDataset"* is implemented to load image paths and corresponding labels from the folder structure. Each image is read using PIL and converted to RGB format to ensure consistency. The class supports applying transformations dynamically during data loading.

```python
# Custom dataset loader
class SIPaKMedDataset(Dataset):
    def __init__(self, root, categories, transform=None):
        self.samples = []
        self.labels = []
        self.transform = transform

        for label, category in enumerate(categories):
            cropped_path = os.path.join(root, category, category, "CROPPED")
            image_paths = glob.glob(os.path.join(cropped_path, "*.bmp"))

            for img_path in image_paths:
                self.samples.append(img_path)
                self.labels.append(label)

    def __len__(self):
        return len(self.samples)

    def __getitem__(self, idx):
        img_path = self.samples[idx]
        label = self.labels[idx]

        image = Image.open(img_path).convert("RGB")
        if self.transform:
            image = self.transform(image)

        return image, label
```

Figure 4.22. Code to load and label SIPaKMeD images using a custom dataset class

4.3.3. Dataset Splitting and Loading

The dataset is divided into training (70%), validation (15%), and testing (15%) sets using *"random_split"*. Each subset is then wrapped using a helper class *"TransformDataset"* to apply corresponding image transformations for augmentation and normalization. Next, DataLoaders with a batch size of 64 are created for each subset to enable efficient batch processing, with shuffling enabled during training to improve model generalisation.

```python
# Split dataset
train_size = int(0.7 * len(full_dataset))  # 70% for training
test_size = int(0.15 * len(full_dataset))   # 15% for testing
val_size = len(full_dataset) - train_size - test_size  # 15% for validation
print(f"train size: {train_size}, test size: {test_size}, val size: {val_size}")

train_dataset, val_dataset, test_dataset = random_split(full_dataset, [train_size, val_size, test_size])
```

Figure 4.23. Code to split the dataset into training, testing, and validation sets

```python
# Apply transformations using a wrapper
class TransformDataset(Dataset):
    def __init__(self, dataset, transform):
        self.dataset = dataset
        self.transform = transform

    def __len__(self):
        return len(self.dataset)

    def __getitem__(self, idx):
        image, label = self.dataset[idx]
        if self.transform:
            image = self.transform(image)
        return image, label


# Wrap datasets with their respective transformations
train_dataset = TransformDataset(train_dataset, train_transform)
val_dataset = TransformDataset(val_dataset, val_test_transform)
test_dataset = TransformDataset(test_dataset, val_test_transform)
```

Figure 4.24. Code to apply specific transformations to training, validation, and test datasets using a custom wrapper

```python
# DataLoaders
batch_size = 64
dataloaders = {
    "train": DataLoader(train_dataset, batch_size=batch_size, shuffle=True),
    "val": DataLoader(val_dataset, batch_size=batch_size, shuffle=False),
    "test": DataLoader(test_dataset, batch_size=batch_size, shuffle=False)
}
```

Figure 4.25. Code to create DataLoaders for training, validation, and testing with a batch size of 64

4.3.4. Transfer Learning with ResNet-50

To reduce training time and utilise the pre-learned features, a pre-trained ResNet-50 model is used. The initial layers are frozen to preserve general visual features, while deeper layers (layer 3 and layer 4) are unfrozen to allow fine-tuning specific to cervical Pap smear image classification. The final fully connected layer is replaced to output predictions for the five target categories.

```python
# Model definition
def get_model():
    model = resnet50(weights=ResNet50_Weights.IMAGENET1K_V2)

    for param in model.parameters():
        param.requires_grad = False
    for param in model.layer3.parameters():
        param.requires_grad = True
    for param in model.layer4.parameters():
        param.requires_grad = True

    num_ftrs = model.fc.in_features
    model.fc = nn.Linear(num_ftrs, 5)

    return model.to(device)

# Initialize model
model = get_model()
```

Figure 4.26. Code to define and fine-tune the ResNet-50 model for 5-class classification

4.3.5. Training with Early Stopping

The model is trained using the cross-entropy loss function with label smoothing to improve generalisation and reduce overfitting. The Adam optimizer is selected for efficient gradient-based optimisation with a low learning rate of 0.0001. A step learning rate scheduler reduces the learning rate by a factor of 0.1 every 7 epochs to improve convergence. Early stopping is implemented to terminate training if the validation loss does not improve for 10 consecutive epochs, thus preventing unnecessary training and overfitting.

```python
# Loss function and optimizer
criterion = nn.CrossEntropyLoss(label_smoothing=0.1)
optimizer = optim.Adam(model.parameters(), lr=0.0001)

# Learning rate scheduler
scheduler = optim.lr_scheduler.StepLR(optimizer, step_size=7, gamma=0.1)

num_epochs = 50
early_stop_patience = 10
```

Figure 4.27. Code to set loss function, optimizer, learning rate scheduler, and training parameters

The complete *"train_model"* function is shown in Figure 4.28. It manages the full training and validation cycle over multiple epochs, calculating and logging loss and accuracy for both phases to TensorBoard. Besides that, it updates model weights using backpropagation during training and evaluates performance on validation data without weight updates. Finally, the best model based on validation loss is saved, ensuring the most effective weights are preserved for later use.

```
# Training function with early stopping
def train_model(model, device, dataloaders, criterion, optimizer, num_epochs, writer, early_stop_patience):
    best_val_loss = float("inf")
    patience_counter = 0

    # Move model to device
    model.to(device)

    print("{0:>15} | {1:>15} | {2:>15} | {3:>15} | {4:>15}".format(
        "Epoch", "Train Loss", "Train Acc (%)", "Val Loss", "Val Acc (%)"
    ))

    for epoch in range(num_epochs):
        # Training Phase
        model.train()
        total_train_loss = 0.0
        train_correct = 0  # Track correct predictions
        total_train_samples = 0

        for batch_idx, (images, labels) in enumerate(dataloaders['train']):
            images, labels = images.to(device), labels.to(device)
            optimizer.zero_grad()
            outputs = model(images)
            loss = criterion(outputs, labels)
            loss.backward()
            optimizer.step()

            total_train_loss += loss.item() * images.size(0)

            # Training Accuracy Calculation
            _, preds = torch.max(outputs, dim=1)
```

```
        train_correct += preds.eq(labels).sum().item()
        total_train_samples += labels.size(0)

    train_loss = total_train_loss / len(dataloaders['train'].dataset)
    train_accuracy = (train_correct / total_train_samples) * 100

    writer.add_scalar("Training Loss", train_loss, epoch)
    writer.add_scalar("Training Accuracy", train_accuracy, epoch)

    # Validation Phase
    model.eval()
    val_loss = 0.0
    val_correct = 0
    total_val_samples = 0

    with torch.no_grad():
        for batch_idx, (images, labels) in enumerate(dataloaders['val']):
            images, labels = images.to(device), labels.to(device)
            outputs = model(images)
            loss = criterion(outputs, labels)

            val_loss += loss.item() * images.size(0)

            # Validation Accuracy Calculation
            _, preds = torch.max(outputs, dim=1)
            val_correct += preds.eq(labels).sum().item()
            total_val_samples += labels.size(0)

    val_loss = val_loss / len(dataloaders['val'].dataset)
    val_accuracy = (val_correct / total_val_samples) * 100

    writer.add_scalar("Validation Loss", val_loss, epoch)
    writer.add_scalar("Validation Accuracy", val_accuracy, epoch)
    writer.flush()

    # Early Stopping
    if val_loss < best_val_loss:
        best_val_loss = val_loss
        patience_counter = 0
        torch.save(model.state_dict(), "/kaggle/working/best_model.pth")
    else:
```

```
            patience_counter += 1
            print(f"Patience counter: {patience_counter}/{early_stop_patience}")
            if patience_counter >= early_stop_patience:
                print("Early stopping triggered.")
                break

        print("{0:>15} | {1:>15.4f} | {2:>15.2f} | {3:>15.4f} | {4:>15.2f}".format(
            epoch+1, train_loss, train_accuracy, val_loss, val_accuracy
        ))

    # Save final model
    torch.save(model.state_dict(), "/kaggle/working/final_model.pth")
    print("Training complete. Best model saved.")
```

Figure 4.28. Code for the *"train_model"* function with early stopping, validation, and TensorBoard logging

4.3.6. Model Testing and Performance Evaluation

After training, the model is evaluated on the testing set to assess its performance and effectiveness on unseen data (Figure 4.29). Key metrics including confusion matrix, accuracy, precision, recall, and F1-score are calculated using the *"compute_metrics"* function in Figure 4.30. Additionally, a confusion matrix is visualised as a heatmap through *"plot_confusion_matrix"* to offer clear insights into the model's classification performance across all categories.

```python
# Testing function
def test(model, device, test_loader, criterion, categories):
    model.eval()
    test_loss = 0.0
    all_preds = []
    all_labels = []

    with torch.no_grad():
        for batch_idx, (images, labels) in enumerate(test_loader):
            images, labels = images.to(device), labels.to(device)
            outputs = model(images)

            # Calculate loss
            loss = criterion(outputs, labels)
            test_loss += loss.item() * images.size(0)

            # Store predictions & labels
            _, preds = torch.max(outputs, dim=1)
            all_preds.extend(preds.cpu().numpy())
            all_labels.extend(labels.cpu().numpy())

    # Calculate average test loss
    test_loss = test_loss / len(test_loader.dataset)

    # Calculate evaluation metrics
    conf_matrix, acc, precision, recall, f1 = compute_metrics(all_labels, all_preds)
    plot_confusion_matrix(conf_matrix, categories)

    print(f"Test Loss: {test_loss:.4f}, Accuracy: {acc:.4f}, Precision: {precision:.4f}, Recall: {recall:.4f}, F1-score: {f1:.4f}")

    return test_loss, conf_matrix, acc, precision, recall, f1
```

Figure 4.29. Code to evaluate the trained model on the test dataset

```python
def compute_metrics(all_labels, all_preds):
    conf_matrix = confusion_matrix(all_labels, all_preds)
    acc = accuracy_score(all_labels, all_preds)
    precision = precision_score(all_labels, all_preds, average="macro")
    recall = recall_score(all_labels, all_preds, average="macro")
    f1 = f1_score(all_labels, all_preds, average="macro")

    return conf_matrix, acc, precision, recall, f1

def plot_confusion_matrix(conf_matrix, categories):
    plt.figure(figsize=(8, 6))
    sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=categories, yticklabels=categories)
    plt.xlabel('Predicted Labels')
    plt.ylabel('True Labels')
    plt.title('Confusion Matrix')
    plt.show()
```

Figure 4.30. Code to calculate evaluation metrics and visualise the confusion matrix

4.3.7. Deployment of CNN Model

To make the trained CNN model portable, efficient and suitable for integration with the web-based system, the model is converted from PyTorch (.pth) format to ONNX (.onnx) format. The ONNX (Open Neural Network Exchange) format allows models to be used across different frameworks and runtimes, thus improving compatibility and deployment speed.

4.3.7.1. Exporting the Trained CNN Model to ONNX Format

The trained CNN model, named *"final_model.pth"*, is downloaded from the Kaggle notebook. To convert the model, a separate Python script, named *"convert_onnx.py"*, is created. This script loads the trained model, applies the same architecture, and exports it to ONNX format.

convert_onnx.py > ...

```python
import torch
from torchvision.models import ResNet50_Weights, resnet50

# Define model
model = resnet50(weights=ResNet50_Weights.IMAGENET1K_V2)
num_ftrs = model.fc.in_features
model.fc = torch.nn.Linear(num_ftrs, 5)

# Load weights
model.load_state_dict(torch.load("final_model.pth", map_location="cpu"))
model.eval()

# Dummy input with batch size 1 and 3x224x224 image size
dummy_input = torch.randn(1, 3, 224, 224)

# Export to ONNX
torch.onnx.export(
    model,
    dummy_input,
    "final_model.onnx",
    input_names=["input"],
    output_names=["output"],
    opset_version=11,
    dynamic_axes={"input": {0: "batch_size"}, "output": {0: "batch_size"}},
)
```

Figure 4.31. Code to convert *"final_model.pth"* to *"final_model.onnx"*

4.3.7.2. Uploading the ONNX Model to Hugging Face Hub

To make the trained model accessible for deployment, sharing, and inference across different environments, the ONNX version of the model is uploaded to the Hugging Face Hub. Hugging Face provides a convenient and centralised platform to store and distribute machine learning models, with built-in versioning, hosting, and integration capabilities. This makes it easier to use the model in web applications without hosting the model file manually.

Figure 4.32 shows the short Python script that is written to upload the model using the Hugging Face Hub's Python API.

```python
from huggingface_hub import HfApi

api = HfApi()
api.upload_file(
    path_or_fileobj="final_model.onnx",  # Local path
    path_in_repo="final_model.onnx",  # Destination path in model repository
    repo_id="lyttyl/cerviscan-resnet50",  # Model repository on Hugging Face
    repo_type="model",  # Repository type
    token="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",  # Hugging Face access token
)
```

Figure 4.32. Code to upload *"final_model.onnx"* to Hugging Face Hub

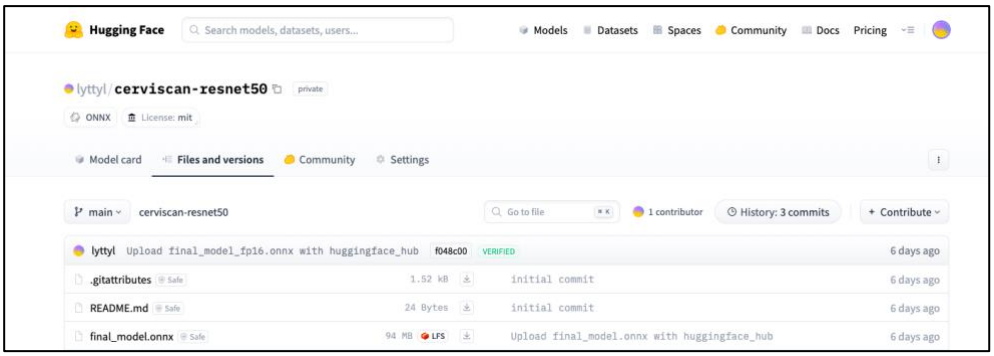


Figure 4.33. Screenshot of *"cerviscan-resnet50"* repository in Hugging Face Hub

4.4. Web-based System Implementation

This section explains the implementation of the web-based system, CerviScan, which enables cervical Pap smear image classification by integrating the trained CNN model developed in this project. The implementation covers both front-end and back-end development, including user interface design, database integration, model interaction workflows, and essential system functionalities. Besides that, the system is structured into six main modules, which are User Authentication, User Profile, Analytics Dashboard, Image Upload and Classification, Classification Result Management, and Patients Management.

4.4.1. Entity Relationship (ER) Diagram

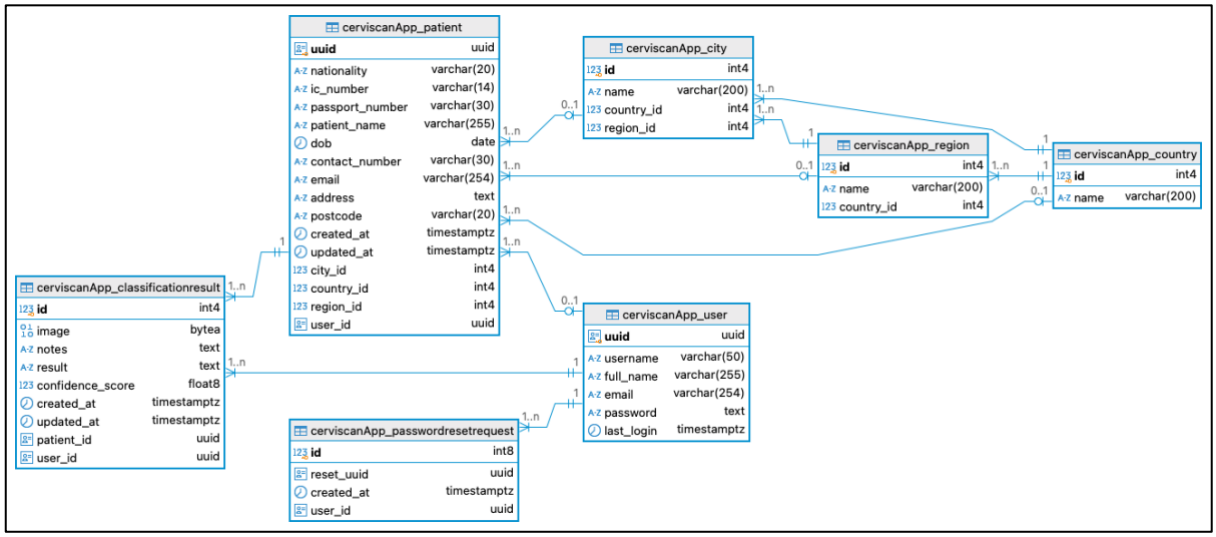


Figure 4.34. Entity Relationship (ER) Diagram for CerviScan

The ER diagram above illustrates the database structure of the CerviScan system, which is designed to manage users, patients, and classification results. The ER diagram uses Crow's Foot notation to represent entities, their attributes, and their relationships, with a clear indication of cardinality and data types.

At the core of the system is the *"cerviscanApp_user"* entity, which represents users such as medical professionals. Each user has a unique UUID as the primary key, along with attributes like username, full name, email, password, and timestamp of the last login. Additionally, a single user can be associated with many patients, classification results, and password reset requests. This design ensures that each user can manage several patients, perform more than one cervical Pap smear image classification task, and initiate multiple password recovery actions.

The *"cerviscanApp_passwordresetrequest"* table holds password reset records initiated by users. Each reset entry contains an auto-incremented ID, a unique *"reset_uuid"*

token, a timestamp, and a foreign key *"user_id"* linking it back to the requesting user. This allows the system to manage multiple concurrent or historical reset attempts by a single user.

Besides that, the *"cerviscanApp_patient"* table stores detailed information about the patients, with each record uniquely identified by a UUID primary key. Examples of the information include IC number, passport number, full name, date of birth, nationality, email, contact number, address, and postcode. Then, the geographical information (country, region/state, city) is managed through the foreign key relationships with the *"cerviscanApp_country"*, *"cerviscanApp_region"*, and *"cerviscanApp_city"* tables. These custom tables were populated with essential geographical data, specifically *"geoname_id"* and *"name"*, which were imported from the *"django-cities-light"* package. In addition, each patient is mandatorily linked to a registered user through the *"user_id"* as a foreign key, representing the medical practitioner responsible for managing their profile.

Next, the *"cerviscanApp_classificationresult"* table records the classification results of the uploaded cervical Pap smear images. Each result includes a unique ID, the binary Pap smear image, optional notes, a result field indicating the classification category, and a confidence score that reflects the model's certainty. Moreover, the table also includes timestamp fields, like *"created_at"* and *"updated_at"*, to track record history. To ensure traceability and accountability, each classification result is linked to both a user and a patient through the *"user_id"* and *"patient_id"* foreign keys. This shows that a single user or patient can be linked to multiple classification results.

4.4.2. User Authentication Module

4.4.2.1. Login

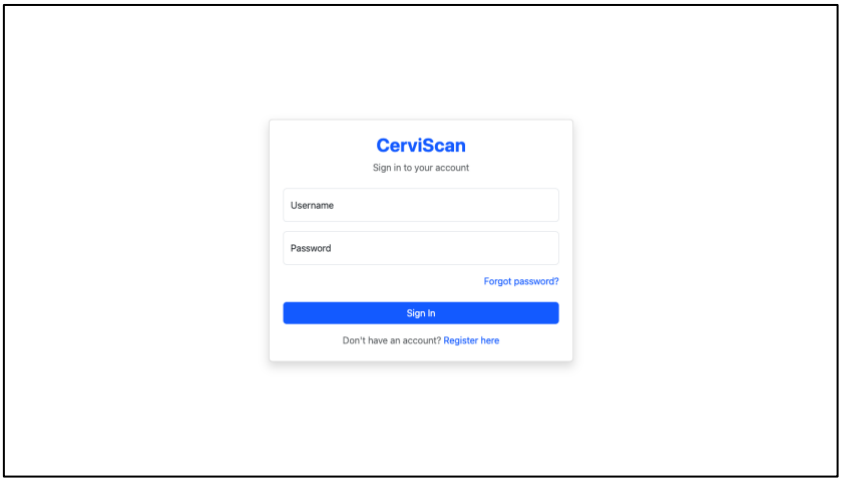


Figure 4.35. CerviScan's Login Page

When a non-logged-in user accesses the CerviScan website, they are directed to the login page (Figure 4.35). To access the system, the user must enter their username and password. Upon successfully login, they will be redirected to the analytics dashboard page. If any login credentials are incorrect, an error message will be displayed using the SweetAlert2 library (Figure 4.36).

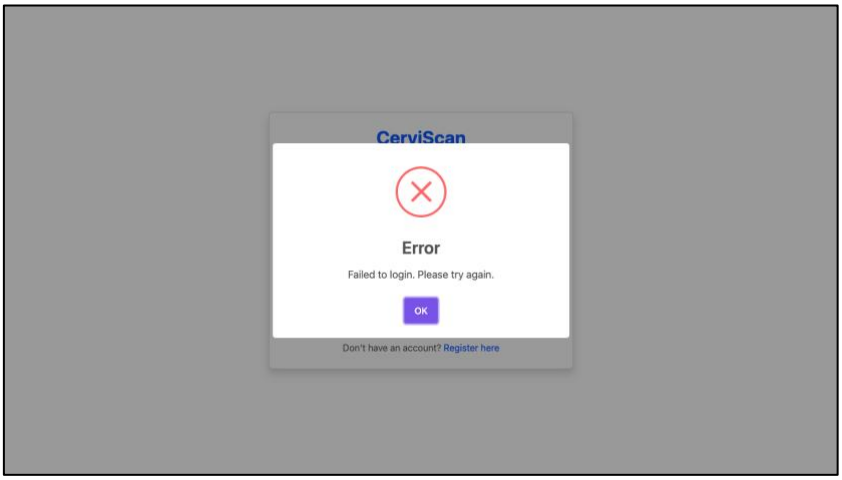


Figure 4.36. "Failed to login" Error Pop-up Message

4.4.2.2. Register

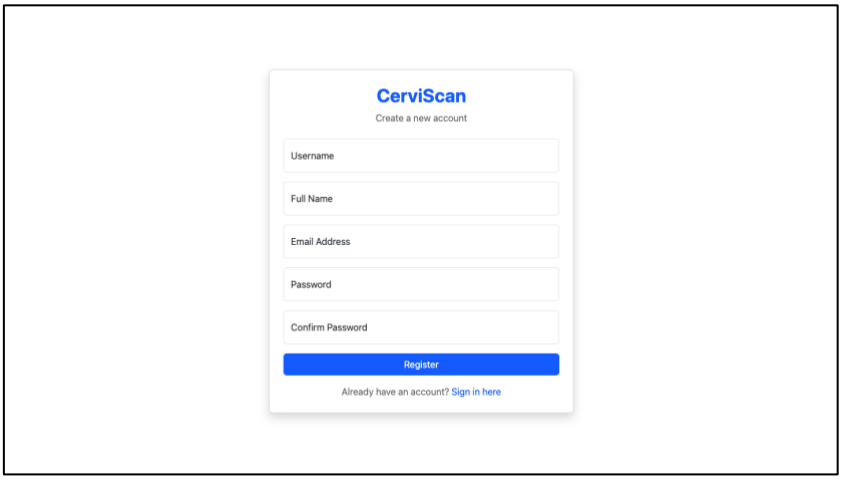


Figure 4.37. CerviScan's Register Page

If the user is new to the system, they can click the *"Register here"* link on the login page, which redirects them to the account registration page as shown in Figure 4.37. Information like username, full name, email address, password and confirm password are required to successfully create a new account. If the chosen username or email address already exists in the system, registration will fail, and an error message will be displayed. However, upon successful registration, an email will be sent to the registered email address while redirecting back to the login page with a success pop-up message displayed.

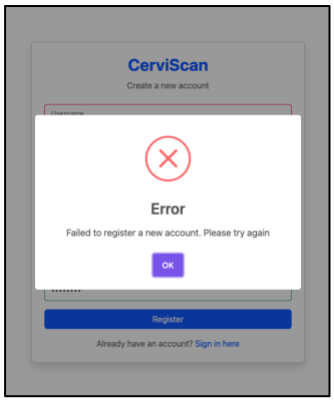


Figure 4.38. "Failed Registration" Error Pop-up Message

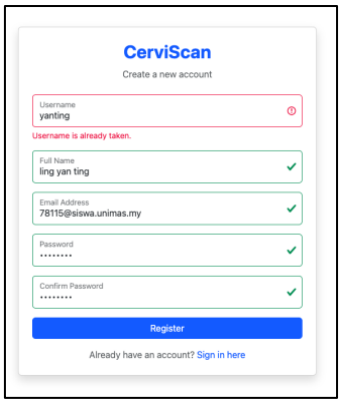


Figure 4.39. Sample Error Message at the Username Field

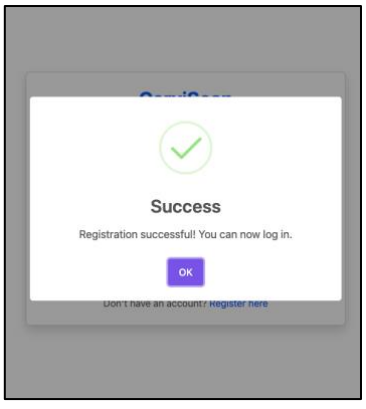


Figure 4.40. Registration Success Pop-up Message

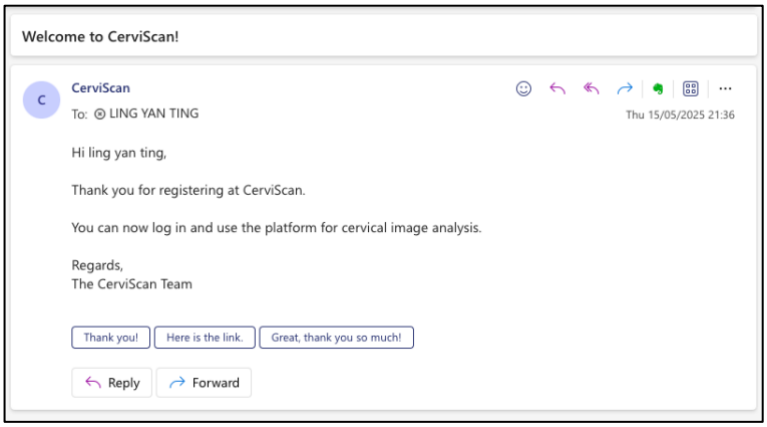


Figure 4.41. Sample Email Received by User Upon Successful Registration

4.4.2.3. Forgot Password

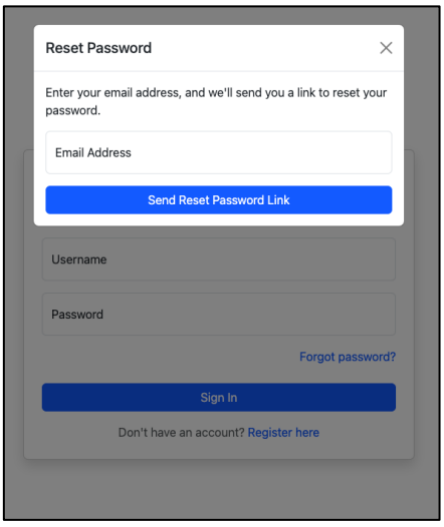


Figure 4.42. Password Reset Pop-up Modal

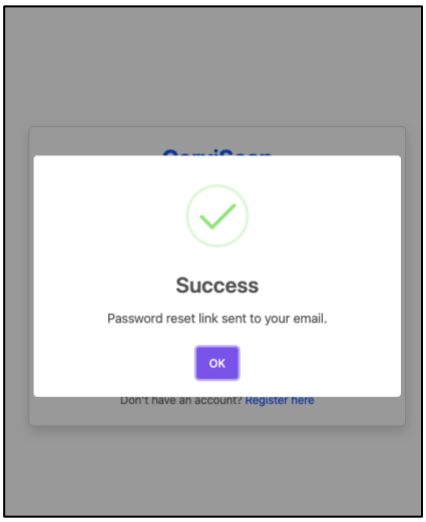


Figure 4.43. Pop-up Message for Password Reset Link Sent Successfully

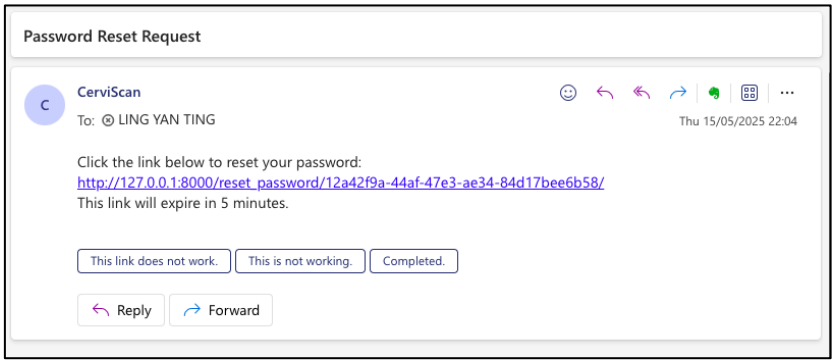


Figure 4.44. Sample "Password Reset Request" Email Received by User

If the user forgets their password, they can click the *"Forgot password?"* link on the login page and a reset password modal will be shown (Figure 4.42). To request for password reset, the user must enter their registered email address. Once the email address is validated and exists in the system, a password reset link will be sent to the email (Figure 4.44) and a success pop-up message will be displayed (Figure 4.43).

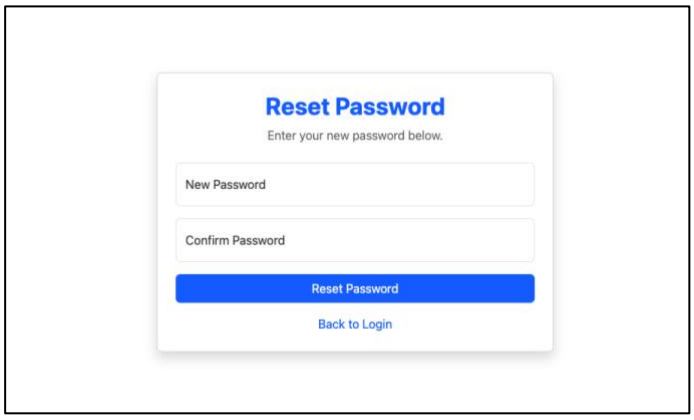


Figure 4.45. Password Reset Page

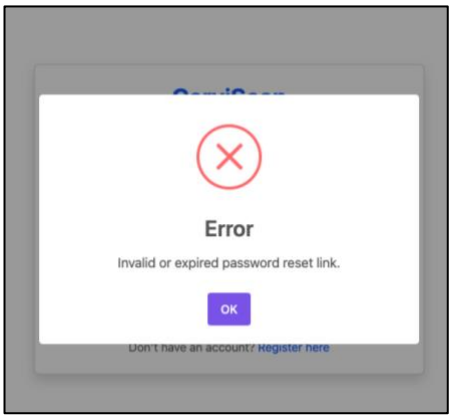


Figure 4.46. Error Message for Invalid or Expired Password Reset Link

After clicking the link provided in the received email, the user will be redirected to the password reset page as shown in Figure 4.45, where they can set a new password. Moreover, several security measures were considered in developing this feature to enhance the system's integrity. For example, enforcing a 5-minute expiration time for the password reset link, ensuring that no important user credentials are exposed in the URL, preventing repeated use of the link once the password has been successfully changed, and invalidating the old password request for the user if a new request is made. Figure 4.46 shows the sample error message for an invalid or expired password reset link.

4.4.2.4. Logout

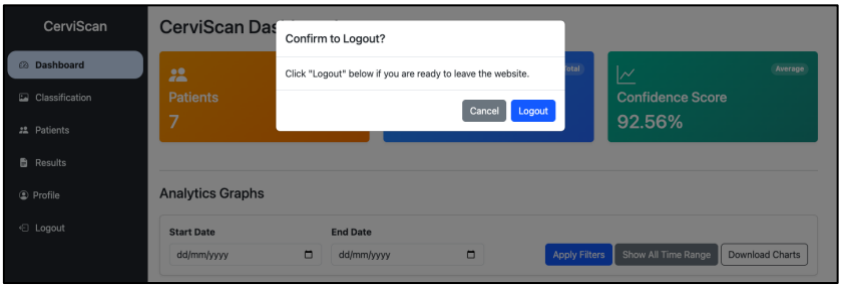


Figure 4.47. Logout Confirmation Modal

The user can initiate the logout function by clicking the *"Logout"* tab on the sidebar, which will trigger a logout confirmation modal (Figure 4.47). To proceed with logging out, the user must click the blue *"Logout"* button in the bottom right of the modal. Upon confirmation, the user session will be terminated, and they will be redirected to the login page.

4.4.3. User Profile Module

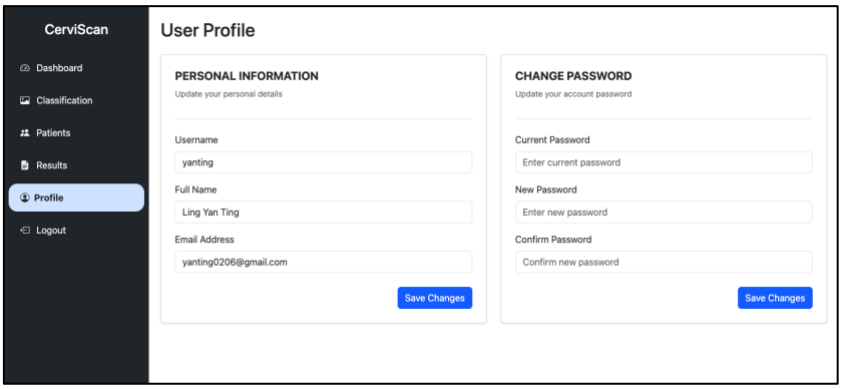


Figure 4.48. User Profile Page

Figure 4.48 shows the user profile page, which is divided into two sections, *"Personal Information"* and *"Change Password"*. To update their username, full name, or email address, the user can edit the respective fields and click the *"Save Changes"* button to apply the modifications. Meanwhile, for more security, the *"Change Password"* feature requires the user to their current password correctly before a new password can be set.

4.4.4. Analytics Dashboard Module

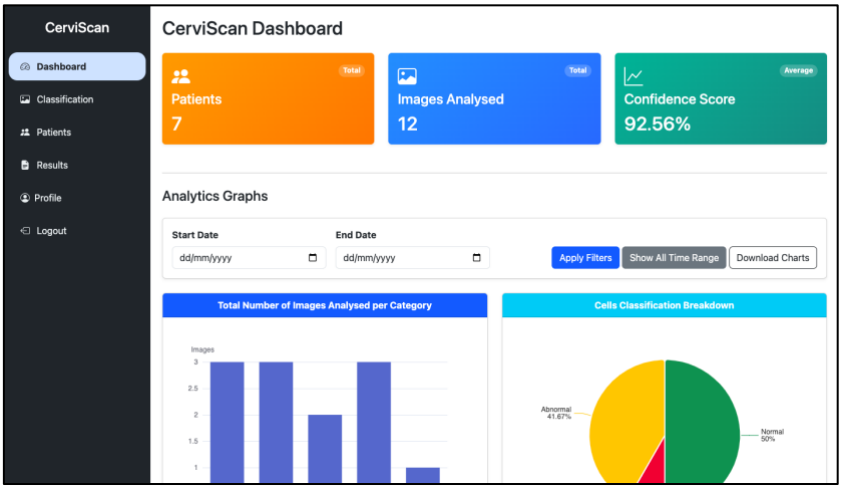


Figure 4.49. Dashboard Page

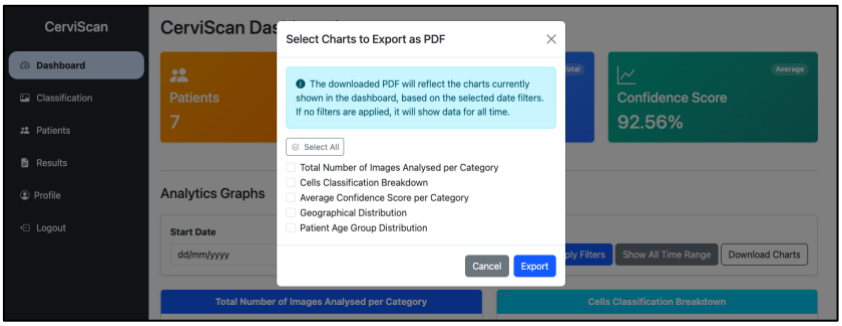


Figure 4.50. Pop-up Modal for Selecting Charts to Export as PDF Report

Upon successful login, the user will be redirected to the dashboard page as shown in Figure 4.49. The upper section of the dashboard includes three summary cards that display the total number of patients, the total number of images analysed in the system, and the average confidence score. Then, the lower section presents five types of analytical graphs using the ECharts library to provide the users with a more comprehensive and visual understanding of the overall data (Figure 4.51).

Besides that, a time filter feature is provided in the analytical graphs section, allowing users to view the charts based on a specific date range. This also offers the users more focused insights and analysis of the image classifications by category, age, geographical location, and

confidence score. Not only this but a download feature is also implemented to allow users to export the displayed analytics charts into a PDF format using Html2Canvas and jsPDF libraries. The user can click the *"Download Charts"* button, and a pop-up modal will be displayed for them to select the desired charts to be downloaded.

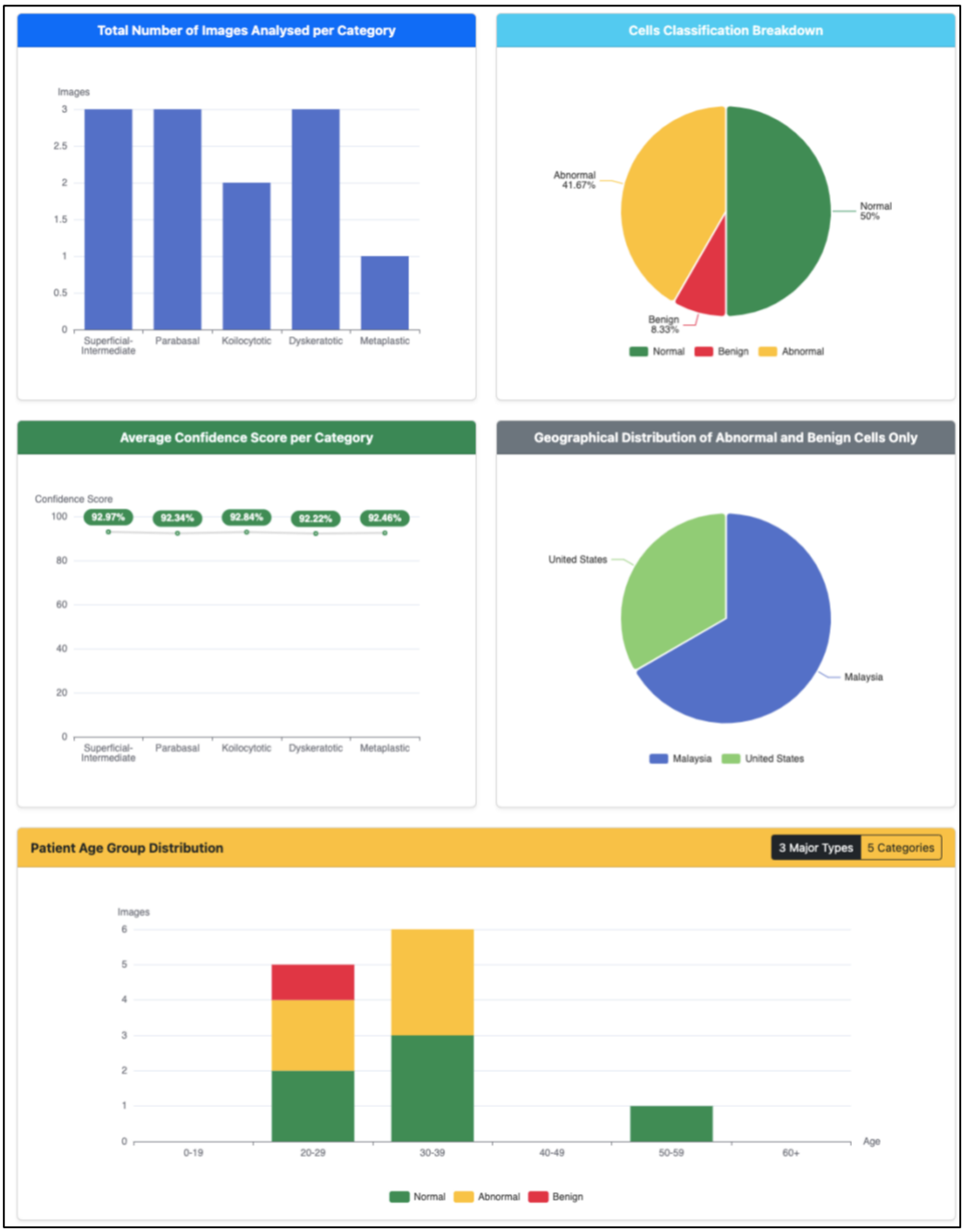


Figure 4.51. Sample of the Five Analytical Charts on the Dashboard

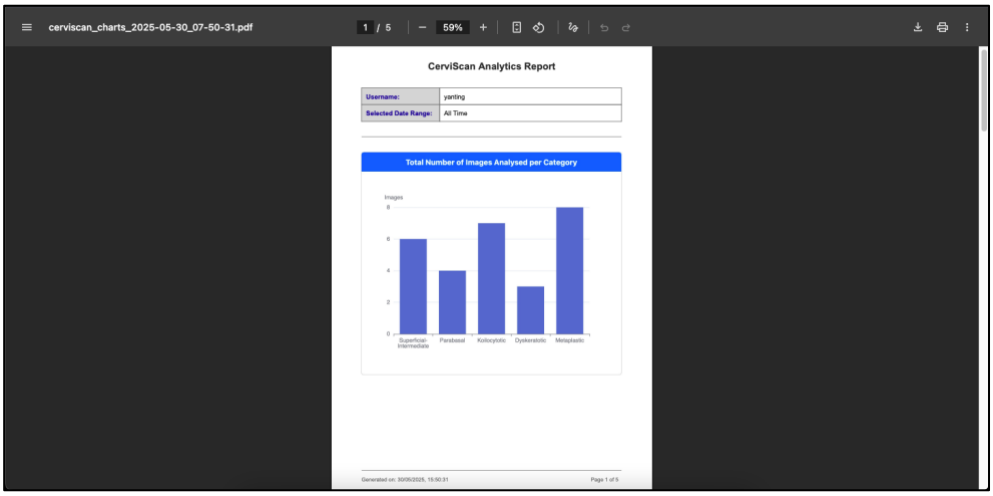


Figure 4.52. Sample Analytics Report

Figure 4.52 shows the sample analytics report generated using the download feature. The exported PDF file is automatically named with a timestamp to reflect the date and time of the download, allowing users to organise and reference their reports more easily. Additionally, the first page of the report includes information, like the username of the logged-in user and the selected date range, to provide clear context and traceability for the data visualised in the subsequent charts.

4.4.5. Image Upload and Classification Module

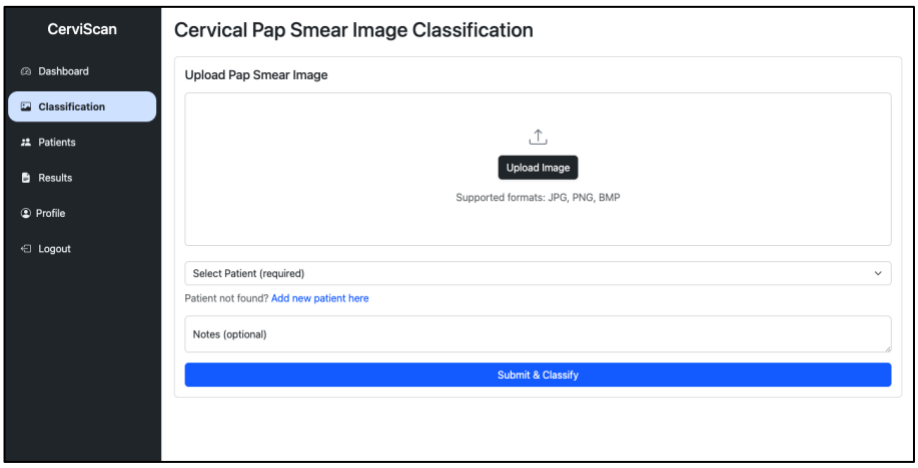


Figure 4.53. Cervical Pap Smear Image Classification Page

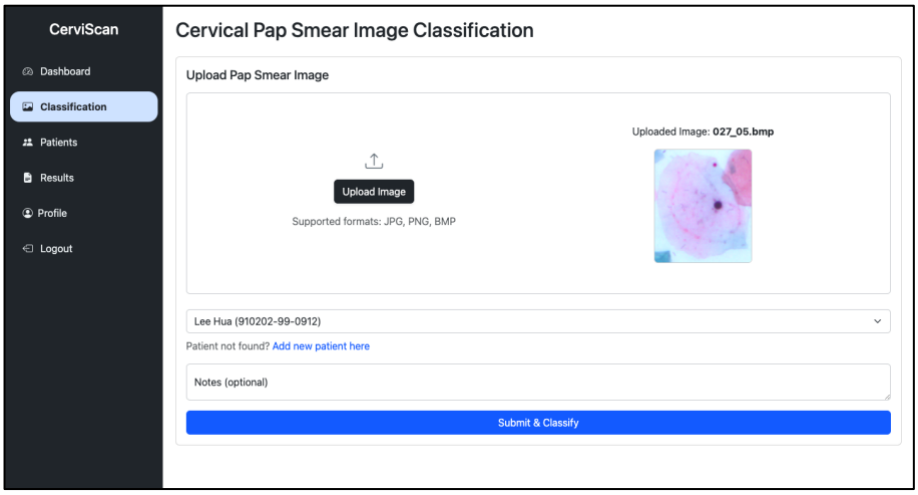


Figure 4.54. Example of Uploaded Image Preview Before Submission

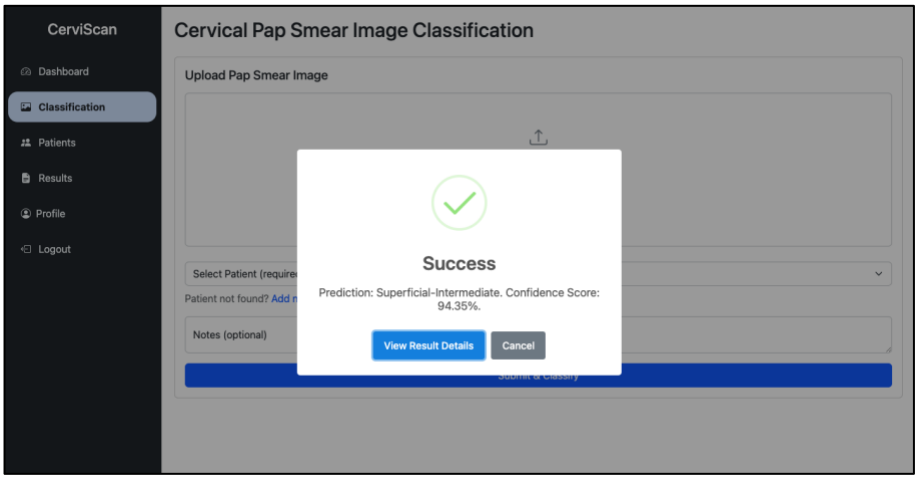
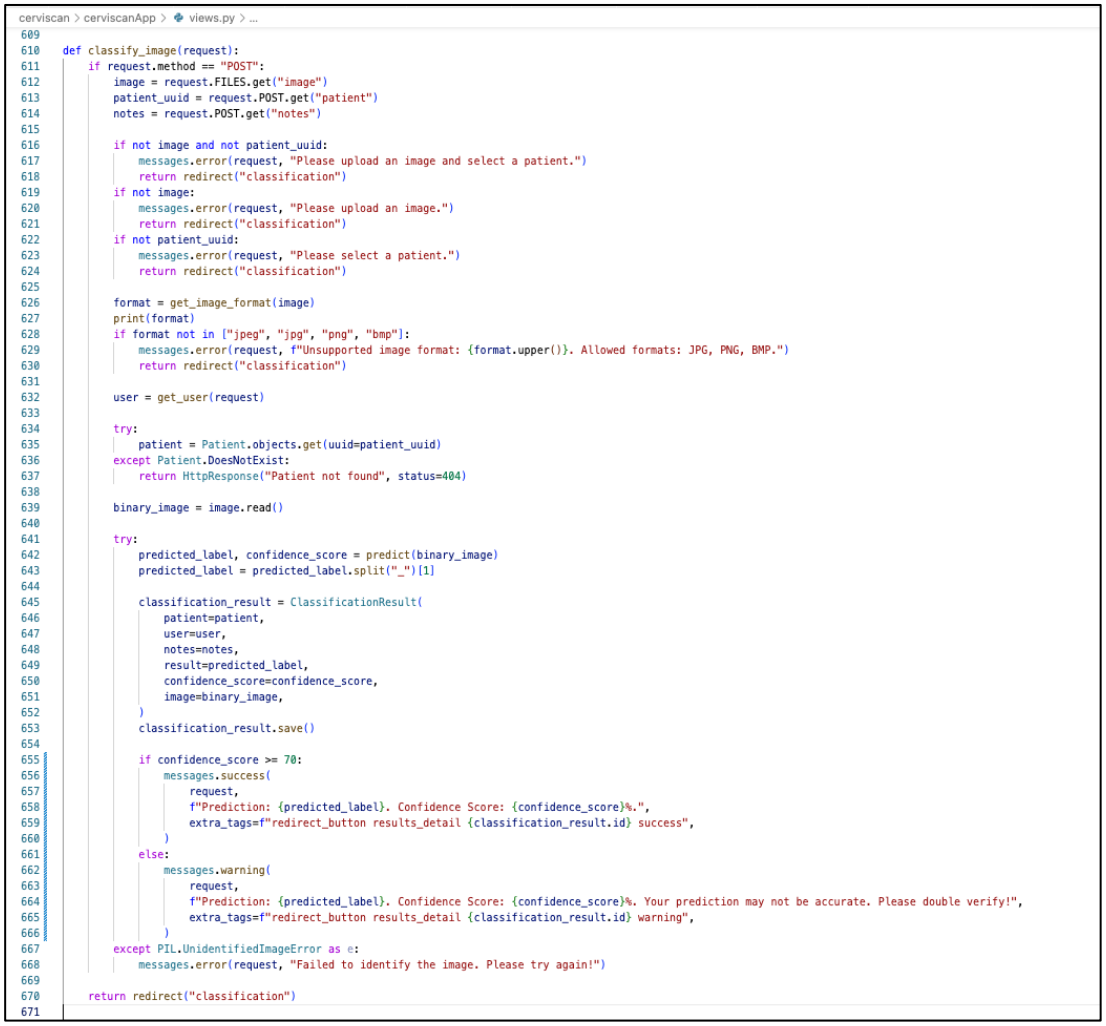


Figure 4.55. Successful Pop-up Alert with Classification Result and Confidence Score

Next, to upload a cervical Pap smear image and obtain the classification result, the user can navigate the *"Classification"* tab, which opens the page as shown in Figure 4.53. To provide instant feedback on the upload process, a preview of the uploaded image along with its file name will be displayed on the screen as illustrated in Figure 4.54. Additionally, the formats of the image files are limited to jpg, png, and bmp.

Besides uploading an image, the user must assign the image to a specific patient. If the patient is not found in the existing list, they can click the *"Add new patient here"* link to add a new patient. Finally, by clicking the *"Submit & Classify"* button, the image is submitted for

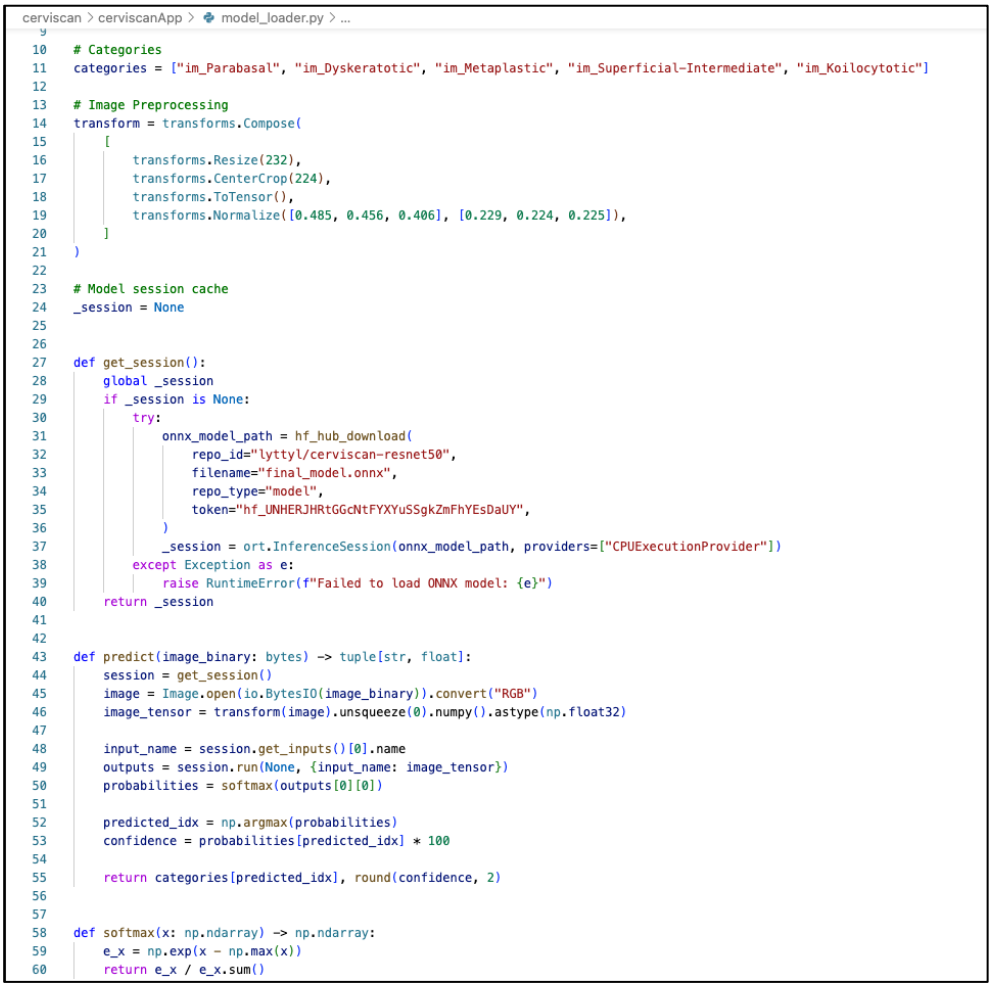
analysis, and the classification result along with its confidence score will be displayed as in Figure 4.55.



```
cerviscan > cerviscanApp > views.py > ...
609
610  def classify_image(request):
611      if request.method == "POST":
612          image = request.FILES.get("image")
613          patient_uuid = request.POST.get("patient")
614          notes = request.POST.get("notes")
615
616          if not image and not patient_uuid:
617              messages.error(request, "Please upload an image and select a patient.")
618              return redirect("classification")
619          if not image:
620              messages.error(request, "Please upload an image.")
621              return redirect("classification")
622          if not patient_uuid:
623              messages.error(request, "Please select a patient.")
624              return redirect("classification")
625
626          format = get_image_format(image)
627          print(format)
628          if format not in ["jpeg", "jpg", "png", "bmp"]:
629              messages.error(request, f"Unsupported image format: {format.upper()}. Allowed formats: JPG, PNG, BMP.")
630              return redirect("classification")
631
632          user = get_user(request)
633
634          try:
635              patient = Patient.objects.get(uuid=patient_uuid)
636          except Patient.DoesNotExist:
637              return HttpResponse("Patient not found", status=404)
638
639          binary_image = image.read()
640
641          try:
642              predicted_label, confidence_score = predict(binary_image)
643              predicted_label = predicted_label.split("_")[1]
644
645              classification_result = ClassificationResult(
646                  patient=patient,
647                  user=user,
648                  notes=notes,
649                  result=predicted_label,
650                  confidence_score=confidence_score,
651                  image=binary_image,
652              )
653              classification_result.save()
654
655              if confidence_score >= 70:
656                  messages.success(
657                      request,
658                      f"Prediction: {predicted_label}. Confidence Score: {confidence_score}%.",
659                      extra_tags=f"redirect_button results_detail {classification_result.id} success",
660                  )
661              else:
662                  messages.warning(
663                      request,
664                      f"Prediction: {predicted_label}. Confidence Score: {confidence_score}%. Your prediction may not be accurate. Please double verify!",
665                      extra_tags=f"redirect_button results_detail {classification_result.id} warning",
666                  )
667          except PIL.UnidentifiedImageError as e:
668              messages.error(request, "Failed to identify the image. Please try again!")
669
670      return redirect("classification")
671
```

Figure 4.56. Code for *"classify_image"* function in *"views.py"*

Upon submission, the uploaded cervical Pap smear image is passed to the backend where the integration with the trained CNN model takes place. The image is first processed by the function *"classify_image"* (Figure 4.56), which retrieves the image file, the selected patient ID, and any accompanying notes from the request. The system ensures that both an image and a patient selection are provided. If not, appropriate error messages are shown to guide the user.



```
cerviscan > cerviscanApp > model_loader.py > ...
# Categories
categories = ["im_Parabasal", "im_Dyskeratotic", "im_Metaplastic", "im_Superficial-Intermediate", "im_Koilocytotic"]

# Image Preprocessing
transform = transforms.Compose(
    [
        transforms.Resize(232),
        transforms.CenterCrop(224),
        transforms.ToTensor(),
        transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225]),
    ]
)

# Model session cache
_session = None


def get_session():
    global _session
    if _session is None:
        try:
            onnx_model_path = hf_hub_download(
                repo_id="lyttyl/cerviscan-resnet50",
                filename="final_model.onnx",
                repo_type="model",
                token="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",
            )
            _session = ort.InferenceSession(onnx_model_path, providers=["CPUExecutionProvider"])
        except Exception as e:
            raise RuntimeError(f"Failed to load ONNX model: {e}")
    return _session


def predict(image_binary: bytes) -> tuple[str, float]:
    session = get_session()
    image = Image.open(io.BytesIO(image_binary)).convert("RGB")
    image_tensor = transform(image).unsqueeze(0).numpy().astype(np.float32)

    input_name = session.get_inputs()[0].name
    outputs = session.run(None, {input_name: image_tensor})
    probabilities = softmax(outputs[0][0])

    predicted_idx = np.argmax(probabilities)
    confidence = probabilities[predicted_idx] * 100

    return categories[predicted_idx], round(confidence, 2)


def softmax(x: np.ndarray) -> np.ndarray:
    e_x = np.exp(x - np.max(x))
    return e_x / e_x.sum()
```

Figure 4.57. Code in the *"model_loader.py"* file

Once validated, the image is read in binary format and passed to the predict function defined in the *"model_loader.py"* file (Figure 4.57). In this function, the image is converted to a standard RGB format and transformed using the same preprocessing steps that were applied during the model training phase. These transformations include resizing, cropping, normalising, and converting the image to a tensor suitable for PyTorch input. The transformed image is then passed through the trained CNN model.

The model outputs raw prediction scores (logits), which are converted into probabilities using the *"softmax"* function. The system identifies the class with the highest probability as the final predicted label and retrieves its corresponding confidence score. The result is then stored in the database as a new *"ClassificationResult"* object, which includes details such as

the predicted label, confidence score, original image, associated patient, and the user who performed the classification task.

4.4.6. Classification Result Management Module

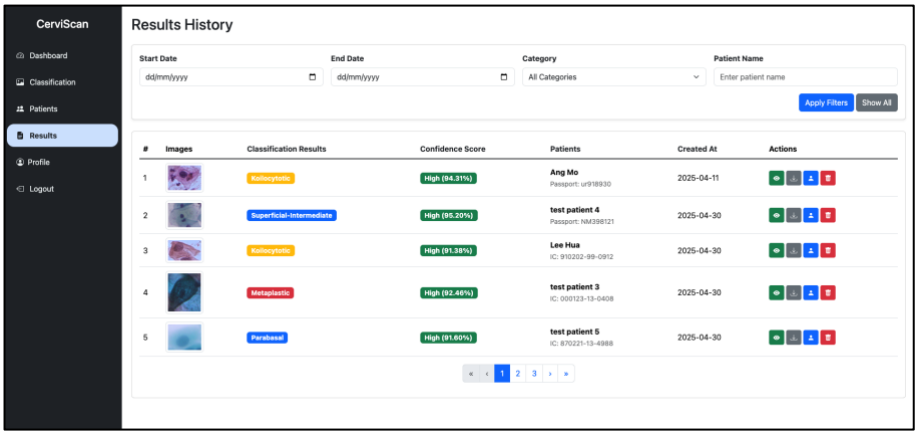


Figure 4.58. Classification Results Page

Figure 4.58 shows the classification results page which can be accessed through the *"Results"* tab on the sidebar. By default, the page will show five rows of results per view. If more results are available, users can navigate through them using the pagination bar located at the bottom of the page.

Next, the results table supports sorting functionality, allowing users to organise the list in either ascending or descending order by clicking on the column headers. Sorting can be performed based on attributes such as the category of classification results, confidence score, patient's name, or the result's creation date.

Additionally, a filter function is also implemented on this page to enhance results navigation. For example, the user can narrow down the results based on a specific time range, category or patient's name for quicker access to their desired results.

Moreover, each of the four buttons in the *"Actions"* column indicates a different feature, including viewing the result's details, downloading results as a PDF report via

ReportLab library, reassigning the patient for the selected result, and deleting a result from the system.

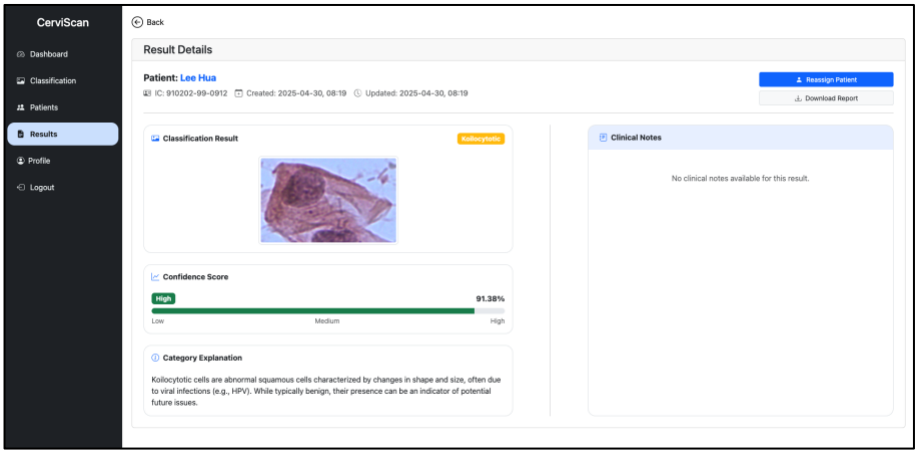


Figure 4.59. Example of Result Details Page

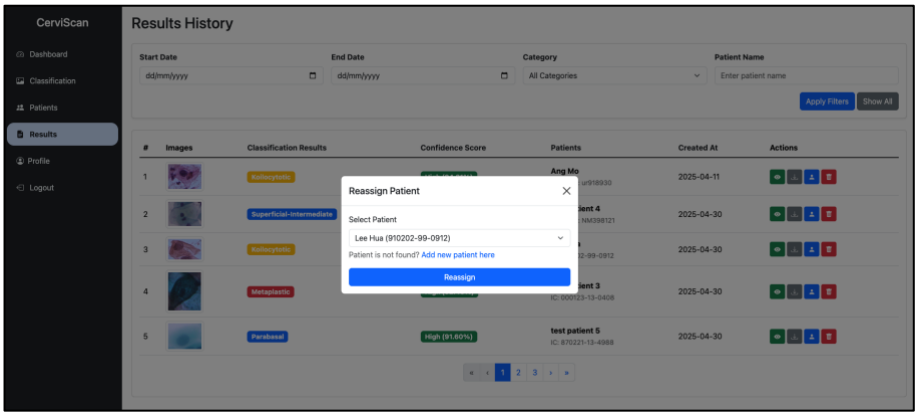


Figure 4.60. Pop-up Modal to Reassign Patient for Selected Result

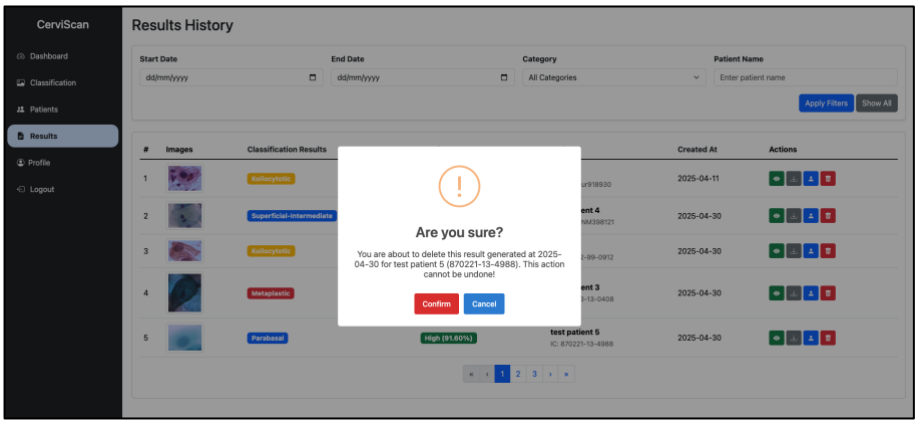


Figure 4.61. Confirmation Modal to Delete Selected Result

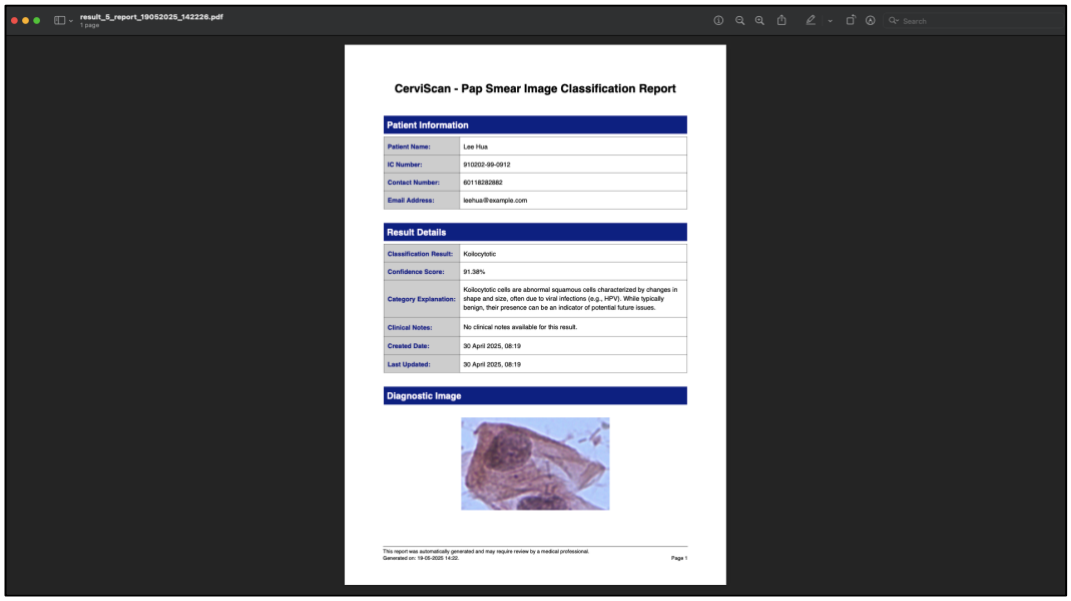


Figure 4.62. Sample PDF Report for Selected Classification Result

4.4.7. Patients Management Module

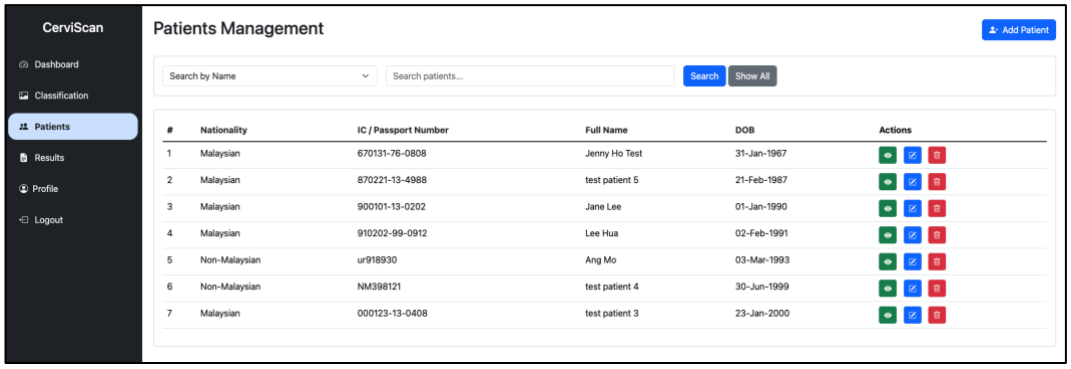


Figure 4.63. Patients Management Page

Figure 4.63 shows the patients management page which can be accessed through the "Patients" tab on the sidebar. By default, the page will show ten rows of patients per view. If the number of patients exceeds ten, users can navigate through additional records using the pagination bar located at the bottom of the page. Note that the pagination bar will be hidden if the total number of patients is less than ten. Similar to the results table, the patients table also supports sorting functionality in either ascending or descending order based on various attributes, such as nationality, IC or passport number, full name, and date of birth.

Additionally, a search function is also implemented to help users quickly look for specific patients. For example, the user can search the patients based on their IC number, passport number or full name. Not only this, but the user can also filter the list of patients according to their nationality by selecting either Malaysian or Non-Malaysian.

Besides that, the user can click the *"Add Patient"* button on the top right corner of the page to access the Add New Patient page as shown in Figure 4.64. The patient register form mandatorily requires the patient's personal information, contact information, and address information.

Finally, each of the four buttons in the *"Actions"* column indicates a different feature, including viewing the patient's details and the associated results, editing the selected patient's information, and deleting a patient from the system.

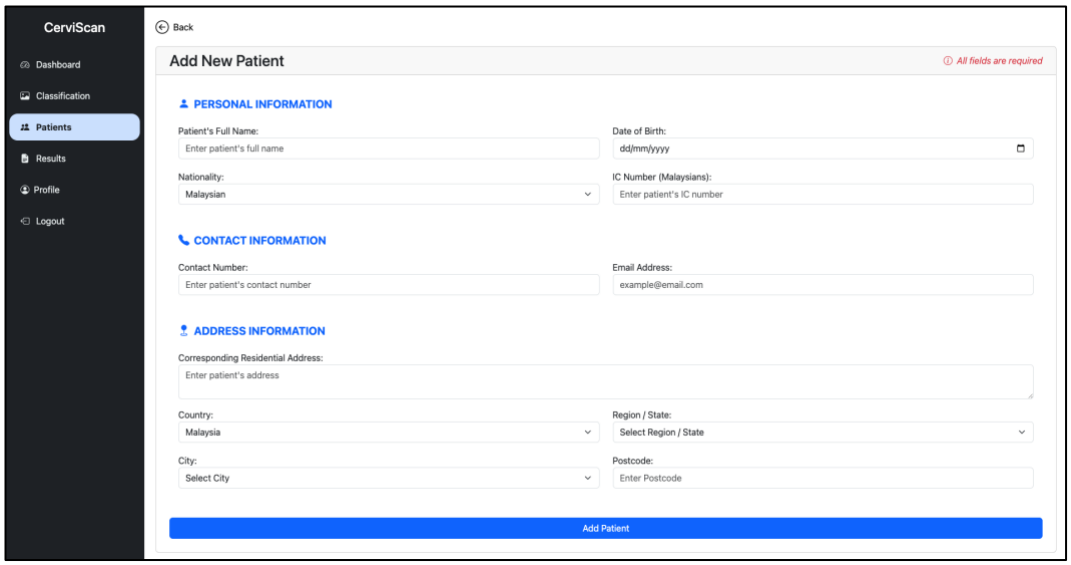


Figure 4.64. Add New Patient Page

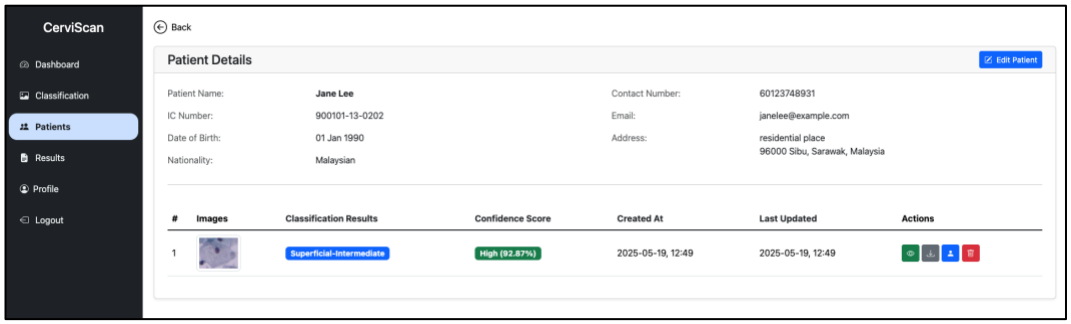


Figure 4.65. Patient Details Page

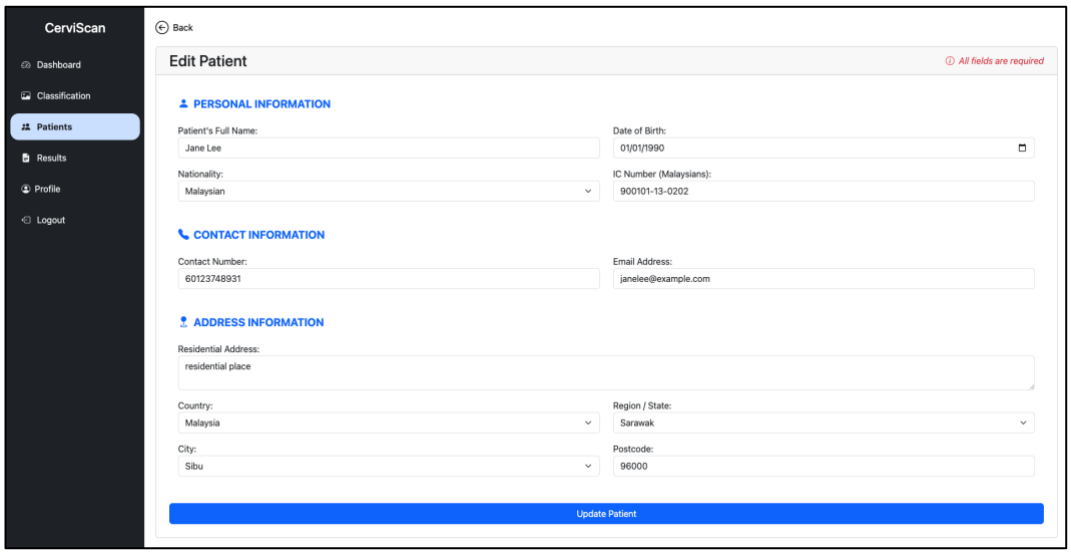


Figure 4.66. Edit Patient Page

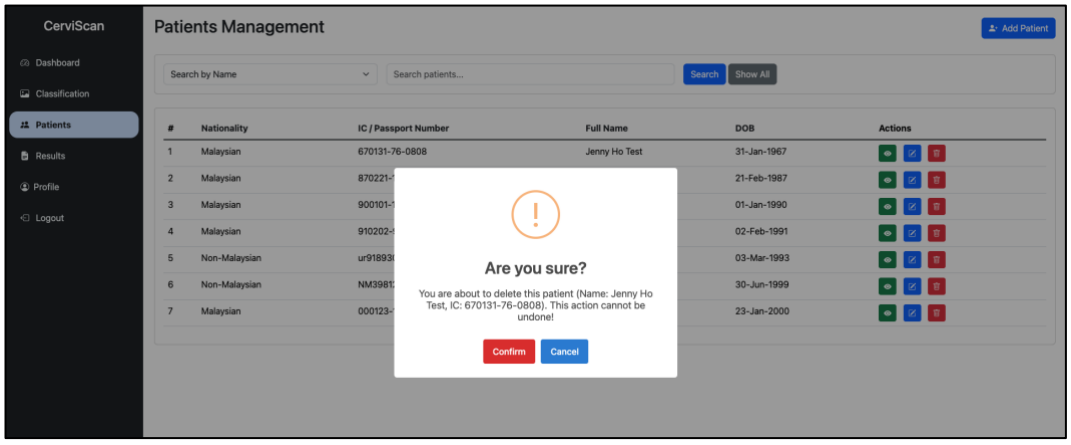


Figure 4.67. Confirmation Modal to Delete Selected Patient

4.5. Summary

This chapter provides a detailed overview of the implementation of the CerviScan project, covering both the CNN model training and the development of a web-based system. The model is trained using a transfer learning approach with ResNet-50 in Kaggle's cloud GPU environment and the SIPaKMeD dataset. Additionally, the training process is supported by essential Python libraries like PyTorch and scikit-learn to support data preprocessing, augmentation, model fine-tuning, and performance evaluation. Next, the web-based system is developed using Django for the backend, Bootstrap 5 for the frontend, and PostgreSQL for database management. The system is also structured into six main modules, including User Authentication, User Profile, Analytics Dashboard, Image Upload and Classification, Classification Result Management, and Patients Management, while the user interface, core functionalities, and integration with the trained model are being discussed in this chapter.

CHAPTER 5: Evaluation and Testing

5.1. Introduction

This chapter discusses the evaluation and testing activities conducted for the CerviScan project. The evaluation section focuses on the performance of the trained CNN model. It includes a justification of the accuracy and loss trends observed during the model's training and validation. Then, it also presents an analysis of the model's effectiveness on the test dataset using confusion matrix along with evaluation metrics such as accuracy, precision, recall and F1-score. On the other hand, the testing section verifies the functionality and robustness of the web-based system. It includes a functional testing to ensure that each feature works as expected, and a non-functional testing to examine aspects such as usability, reliability, and efficiency. Additionally, a user acceptance testing (UAT) is carried out to ensure that the system meets the expectations and requirements of end users, particularly medical practitioners.

5.2. CNN Model Evaluation

This section presents the evaluation of the trained CNN model using the SIPaKMeD dataset. The evaluation comprises three key components, including the training and validation performance, quantitative evaluation metrics on the testing set, and visual interpretation using a confusion matrix. The dataset is split into 70% training, 15% validation, and 15% testing to ensure a balanced and robust evaluation of the model's generalisation capabilities.

5.2.1. Training and Validation Performance

To assess the learning progression of the CNN model, training and validation accuracy and loss are monitored over a maximum of 50 epochs with an early stopping patience of 10

epochs to prevent overfitting. The final training process halts at epoch 28, where no further improvements are detected within the early stopping threshold.

5.2.1.1. Training Accuracy and Loss Trends

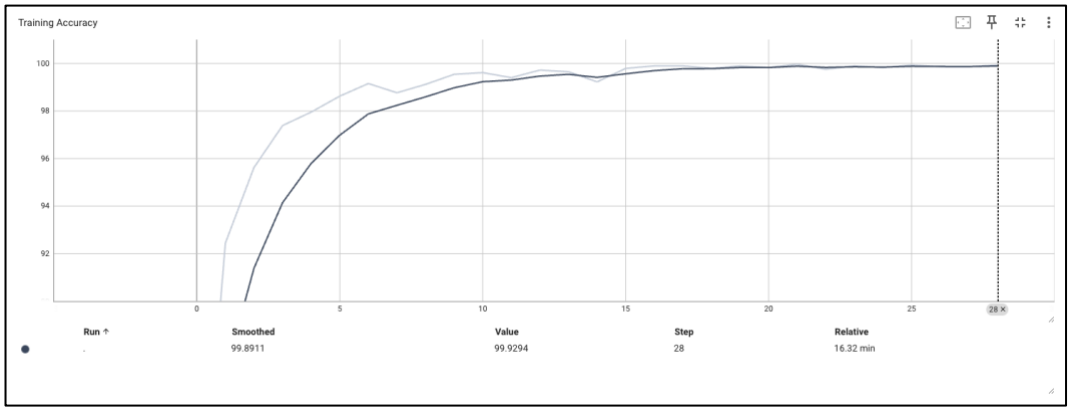


Figure 5.1. Training Accuracy Trend

As illustrated in Figure 5.1, the training accuracy exhibits a rapid increase in the initial epochs, rising from approximately 91% to over 99.23% within the first 10 epochs. This early improvement suggests that the model rapidly learns the distinguishing features present in the training images. After this initial surge, the accuracy continues to increase gradually, eventually stabilising at 99.93% by epoch 28. This exceptionally high training accuracy suggests that the model is able to achieve strong memorisation of the training data.

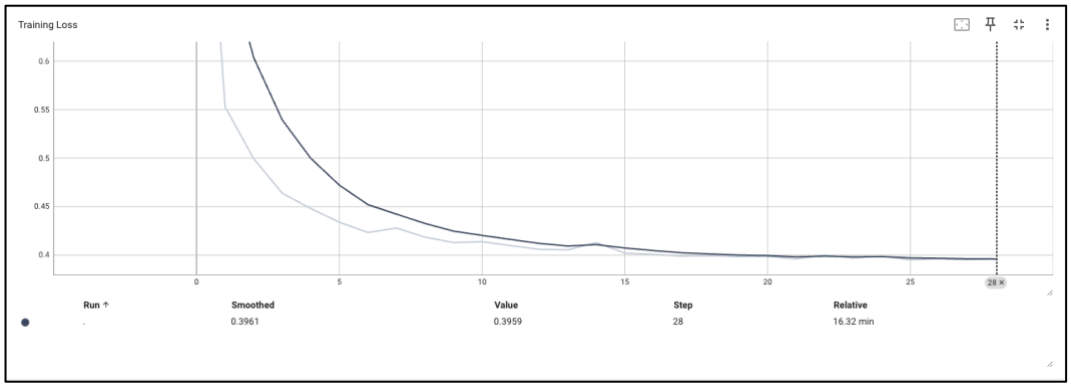


Figure 5.2. Training Loss Trend

Simultaneously, the training loss in Figure 5.2 shows a consistent downward trend. Starting from a value above 0.6, the loss declines steadily to 0.3959 by the end of training. The steady decrease in training loss indicates that the model is optimising its parameters effectively through backpropagation, gradually reducing the error between predicted and actual class labels.

5.2.1.2. Validation Accuracy and Loss Trends

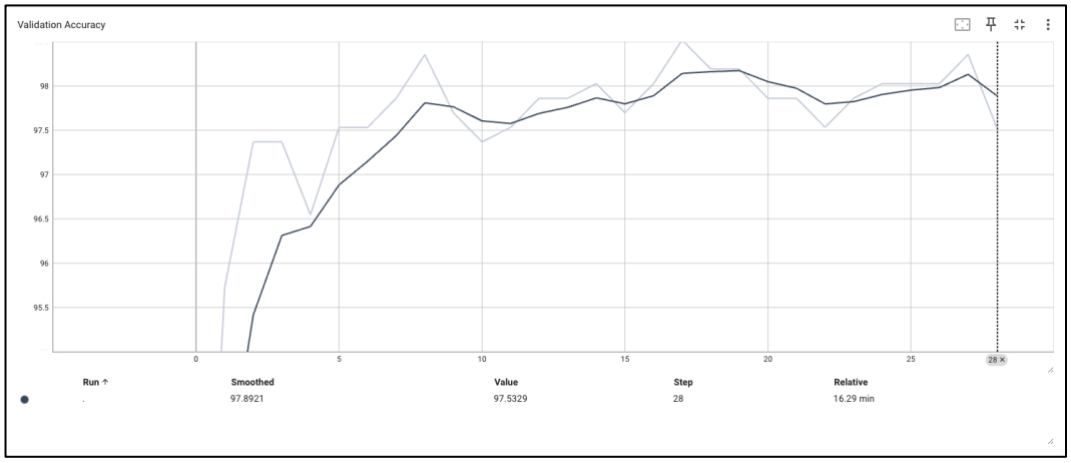


Figure 5.3. Validation Accuracy Trend

According to the validation accuracy trend shown in Figure 5.3, the validation process exhibits two distinct phases of learning behaviour. In the initial phase (epochs 0–8), accuracy rapidly improves from around 95% to 98.36%, highlighting the model's ability to quickly learn the core patterns within the dataset. From epochs 9 to 28, the validation accuracy plateaus and fluctuates slightly between 97% and 98%. This showed that the model is undergoing finer adjustments and learning more subtle features. These minor fluctuations are expected since they result from local optimiser adjustments and variations in data complexity. Therefore, the final accuracy of 97.53% at epoch 28 reflects a stable performance while demonstrating a good generalisation without the signs of overfitting.

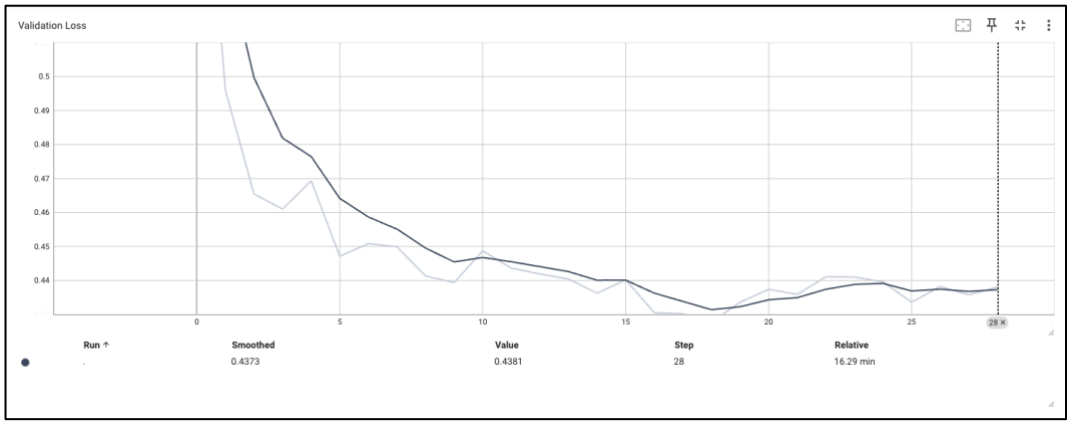


Figure 5.4. Validation Loss Trend

The validation loss trend in Figure 5.4 shows a consistent and effective optimisation of the model throughout the validation process. The validation loss steadily decreases from approximately 0.55 to 0.4381 by the last epoch, indicating reduced prediction errors on unseen data. The trend also exhibits an exponential decay pattern, where the steepest decline occurred in the early epochs, followed by a slower decrease as the model approached convergence. Importantly, the absence of sharp spikes in the validation loss confirms that the model generalised well to unseen data without overfitting.

5.2.2. Testing Performance

After training and validation, the model's generalisation is assessed using the test set, comprising 15% of the SIPaKMeD dataset. This testing phase provides an unbiased measure of the model's real-world performance on previously unseen data.

Figure 5.5 shows the confusion matrix generated to display the number of correct and incorrect predictions made by the model for each of the five cell categories. Meanwhile, Table 5.1 summarises the confusion matrix in numerical form, highlighting the total number of correct predictions and the corresponding misclassifications.

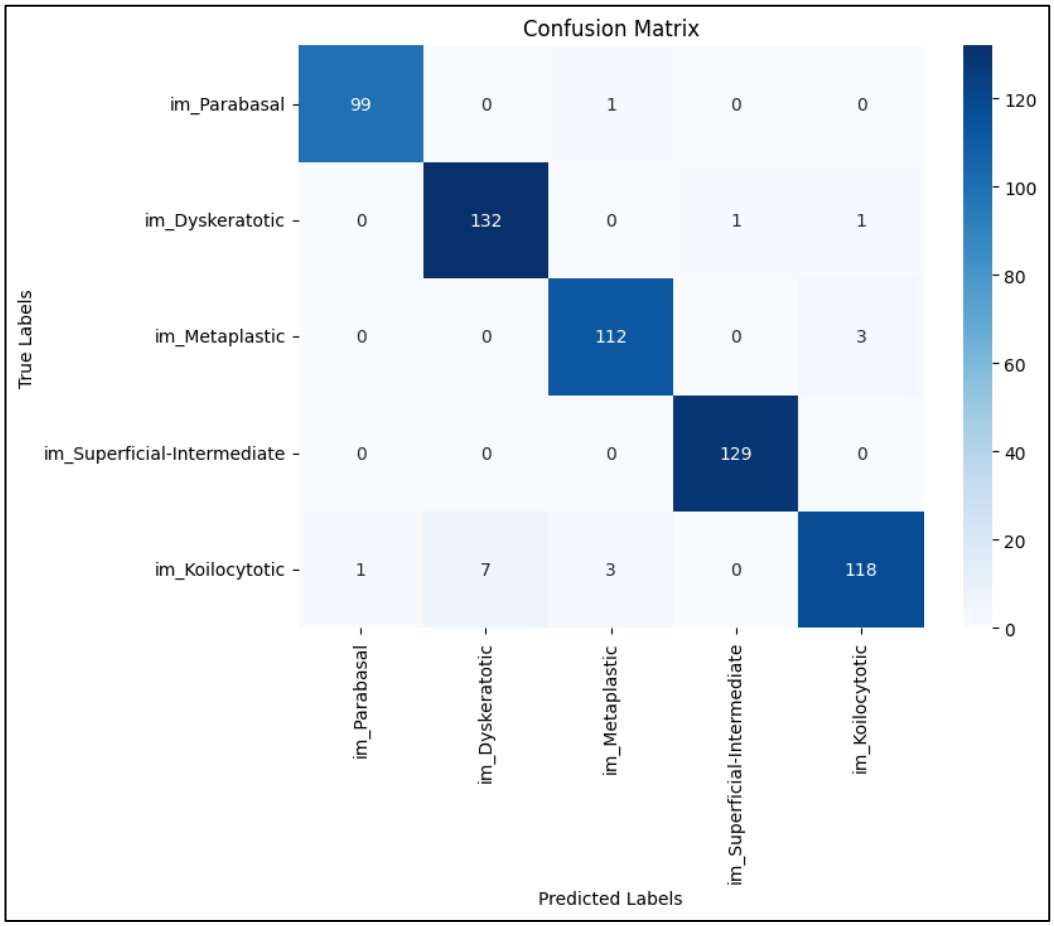


Figure 5.5. Confusion Matrix of Test Results

Table 5.1. Confusion Matrix Summary of Test Set Classification Results

Cell category	Correct Classifications	Accuracy (%)	Misclassifications
Parabasal	99/100	99.00	1 misclassified as Metaplastic.
Dyskeratotic	132/134	98.51	1 misclassified as Superficial-Intermediate, and 1 misclassified as Koilocytotic.
Metaplastic	112/115	97.39	3 misclassified as Koilocytotic.
Superficial-Intermediate	129/129	100.00	None.
Koilocytotic	118/129	91.47	7 misclassified as Dyskeratotic, 1 misclassified as Parabasal, and 3 misclassified as Metaplastic.

The confusion matrix results demonstrate strong overall classification performance, particularly with high specificity and sensitivity for the Superficial-Intermediate and Dyskeratotic categories. The Superficial-Intermediate class achieves perfect accuracy (100%), reflecting its distinctive features, while Dyskeratotic cells also performs excellently with 98.51% accuracy. Parabasal (99%) and Metaplastic (97.39%) cells experiences minimal misclassifications, mostly involving closely related categories.

In contrast, Koilocytotic cells are the most challenging, achieving 91.47% accuracy with greater confusion involving Dyskeratotic and Metaplastic cells. This is likely due to visual similarities in their cellular structures. These results also underscore the biological complexity and overlapping features in cervical cell classification, particularly for HPV-related changes that present subtle diagnostic challenges for both automated systems and pathologists.

Table 5.2. Evaluation Metrics of the CNN Model on the Test Set

Evaluation Metric	**Value (%)**
Accuracy	97.20
Precision	97.29
Recall	97.27
F1-Score	97.26

Table 5.2 presents the evaluation results on the test set. The model achieves a test accuracy of 97.20%, indicating a strong ability to correctly classify most cervical cell images. Next, the precision score of 97.29% shows the model's effectiveness in minimising false positives, which occur when a cell is wrongly identified as belonging to a certain category. In other words, when the model predicts a specific cell type, it is correct in 97.29% of those instances. The recall value of 97.27% reflects the model's success in capturing true positives, meaning it correctly identifies 97.27% of all actual instances for each category. Additionally,

the F1-score, records at 97.26%, combines precision and recall into a single metric and offers a balanced measure of the model's overall classification performance.

Although none of these scores reaches 100%, this is expected in the context of medical image classification. This main reason is the biological complexity and visual similarity of certain cervical cell types that make perfect classification extremely difficult. Additionally, the model iss trained and evaluated using only the SIPaKMeD dataset. While this dataset is high quality, it may not fully capture the diversity of cervical cytology variations encountered in clinical settings. This limitation in dataset variety can affect the model's generalisability and contributes to occasional misclassifications.

However, the model is still considered highly reliable because it consistently performed well on unseen data with very few errors. Its strong accuracy, precision, recall, and F1-score indicate that the model can make correct predictions across most categories. Moreover, misclassifications typically occurs between categories that are biologically and visually similar, which are also known to challenge human experts. The training process also uses early stopping to prevent overfitting, helping the model generalise effectively. Altogether, these results demonstrate that the model is both effective and suitable for practical applications in cervical Pap smear image classification.

5.3. Web-based System Testing

This section outlines the testing procedures and results for the CerviScan web-based system. The goal of system testing is to verify that each module performs its intended functions accurately and reliably. The testing phase is divided into two main categories, that are **functional testing**, which focuses on verifying the correctness of each system feature, and **non-functional testing**, which evaluates the system's usability, reliability, and efficiency. Each test

case was executed using sample user accounts, patient records, and image uploads in a local environment to simulate real-world usage conditions.

5.3.1. Functional Testing

Functional testing is conducted to verify that each core module of the CerviScan system performs according to its specified requirements. This process involves systematically executing predefined test cases designed to cover all key functionalities within the following modules:

- User Authentication
- User Profile Management
- Analytics Dashboard
- Image Upload and Classification
- Classification Results Management
- Patients Management

5.3.1.1. Functional Testing for User Authentication Module

Table 5.3. Functional Test Cases for User Authentication Module

Module Name: User Authentication **Test Case Description:** To verify user registration, login, logout, and password reset functionalities. **Testing Objective:** To ensure secure and correct user access management. **Remarks:** Tests cover valid and invalid inputs and error handling.							
Test Case ID	**Test Scenario**	**Test Steps**	**Test Data**	**Expected Results**	**Actual Results**	**Status**	**Severity of Failure**
Functionality: Register User Account							
UA-01	User registration with valid information	1. Navigate to the registration page 2. Fill in all required fields 3. Click "Register" button to submit the form.	a. Username: yanting b. Full name: Ling Yan Ting c. Email: 78115@siswa.unimas.my d. Password: Abcd1@34 e. Confirm password: Abcd1@34	1. User account is created. 2. Success message is shown. 3. User is redirected to the login page.	1. User account is created. 2. A success registration pop-up message is shown. 3. User receives a success registration email. 4. User is redirected to the login page.	Pass	High
UA-02	User registration with taken username or email	1. Navigate to the registration page. 2. Fill in the taken username or email. 3. Click "Register" button.	Invalid Input Variations: a. Taken username: yanting (already registered in UA-01) b. Taken email: 78115@siswa.unimas.my (already registered in UA-01)	1. User fails to register a new account. 2. Appropriate error messages are shown on the corresponding taken input.	1. Registration fails as expected. 2. A failed registration pop-up message will be shown. 3. Specific error message will be shown on the username or email input field too: a. Taken username: *"Username is already taken."*		Medium

				3. User remains on the registration page to retry.	b. Take email: *"An account with this email already exists."* 4. User remains on the registration page to retry.		
UA-03	User registration with invalid information	1. Navigate to the registration page. 2. Fill in invalid fields based on test conditions. 3. Click "Register" button to submit the form.	Invalid Input Variations: a. Username too short: yan b. Empty full name c. Invalid email: 78115@siswa d. Weak password: Abcde e. Mismatched passwords: Abcd1@34, Abcd1@78	1. User account fails to be registered. 2. Appropriate error messages are shown per invalid input. 3. User remains on the registration page to retry.	1. Before submission, specific inline error message will be shown on the input field: a. Username too short: *"Username must be at least 6 characters and contain no spaces."* b. Empty full name: *"Full name cannot be empty or contain only spaces."* c. Invalid email: *"Enter a valid email address."* d. Weak password: *"Password must be at least 8 characters, with at least 1 uppercase letter, 1 number, and 1 special character."* e. Password mismatch: *"Passwords do not match."* 2. If the user insists to submit the registration form, registration will fail as expected and a failed registration pop-up message will be shown. 3. User remains on the registration page to retry.	Pass	Medium
Functionality: Login							

UA-04	Login with valid information	1. Navigate to the login page. 2. Enter correct username and password. 3. Click “Login” button.	a. Username: yanting b. Password: Abcd1@34	1. User is successfully logged in. 2. User is redirected to the dashboard.	User is successfully logged in as expected and redirected to the dashboard.	Pass	High
UA-05	Login with invalid credentials	1. Navigate to the login page. 2. Enter valid username and incorrect password. 3. Click “Login” button.	a. Valid username: yanting b. Wrong password: Abcd1234#	1. User fails to log in. 2. Error message is displayed to indicate invalid credential.	1. User fails to log in as expected. 2. A login failure pop-up message is shown. 3. Error message indicating wrong password is shown at the password field: *“Invalid password.”* 4. User remains on the login page to retry.	Pass	Medium
Functionality: Forgot Password / Password Reset							
UA-06	Request for password reset link	1. Click “Forgot password?” link on the login page. 2. Enter registered email address. 3. Click “Send Reset Password Link” button.	Registered email in UA-01: 78115@siswa.unimas.my	Password reset email is sent to the user’s email.	1. A success pop-up message is shown to indicate that the email has been sent to the user’s email. 2. User receives the email containing the password reset link.	Pass	High
UA-07	Reset password	1. User clicks the password reset link received in the email [UA-06] and is redirected to the password reset page. 2. User enters the new password and confirms password again.	a. New password: Newde@909 b. Confirm password: Newde@909	1. Password is successfully modified. 2. User is redirected	1. A success pop-up message is shown to indicate that the password has been changed successfully. 2. User is redirected to the login page.	Pass	High

		3. Click “Reset Password” button to submit the new password change.		to the login page.			
UA-08	Attempt to reset password using an expired reset link.	1. Click the password reset link again after completed test case UA-07 or wait for the new password reset link to be expired after 5 minutes. 2. Observe the system response.	Expired or invalid password reset link.	The system inhibits the user to reset password.	1. An error pop-up message will be displayed to notify user on the invalid or expired link. Error message: *“Invalid or expired password reset link”.* 2. User is redirected to the login page.	Pass	Medium
Functionality: Logout **Prerequisite**: User has logged into the system.							
UA-09	Logout	1. Click the “Logout” tab on the sidebar. 2. A confirmation pop-up modal appears. 3. Click the “Logout” button on the modal to confirm logout. 4. Observer system behaviour.	N/A	1. A confirmation modal is displayed when the “Logout” tab is clicked. 2. Upon confirmation, the user session is terminated. 3. User is redirected to the login page.	1. Confirmation modal is shown as expected. 2. User is logged out successfully and redirected to the login page.	Pass	Medium

5.3.1.2. Functional Testing for User Profile Module

Table 5.4. Functional Test Cases for User Profile Module

Module Name: User Profile **Test Case Description:** To verify user profile view and update capabilities. **Testing Objective:** To ensure users can successfully view and modify their profile information. **Remarks:** Includes validation and error handling tests.							
Test Case ID	**Test Scenario**	**Test Steps**	**Test Data**	**Expected Results**	**Actual Results**	**Status**	**Severity of Failure**
Functionality: View Profile							
UP-01	View profile details	1. Login [UA-04] 2. Click "Profile" on the sidebar to navigate to the profile page.	N/A	User profile information is displayed correctly.	User can view their personal information (full name, username, and email) as well as a form to change password.	Pass	High
Functionality: Update Personal Information **Prerequisite:** User has logged in to the system and navigated to the profile page and "personal information" section.							
UP-02	Update personal information with valid inputs	1. Update username, full name, and/or email. 2. Click "Save Changes" to submit the modification.	User may choose to edit one or more fields of personal information. Example below shows modification of username and full name while email remains unchanged. a. Username: yantingtest b. Full name: Ling Yan Ting Test	1. Personal information of user is updated successfully. 2. The new changes are immediately visible to the user.	1. A success pop-up message is shown to indicate the personal information has been modified successfully. 2. The user can view the updated profile immediately.	Pass	Medium

			c. Email: 78115@siswa.unimas.my				
UP-03	Update personal information with invalid inputs.	1. Enter the invalid inputs based on test conditions. 2. Click "Save Changes" to submit the modification.	Invalid Input Variations: a. Username too short: yan b. Empty full name c. Invalid email: 78115@siswa	1. User's personal information fails to be modified. 2. Appropriate error messages are shown on per invalid input field.	1. Before submission, specific inline error messages will already be shown under the input field: a. Username too short: *"Username must be at least 6 characters and contain no spaces."* b. Empty full name: *"Full name cannot be empty or contain only spaces."* c. Invalid email: *"Enter a valid email address."* 2. If the user attempts to submit the form despite the errors, an error pop-up message will be shown to notify user on the failure to update the personal information, while the inline error messages remain visible.	Pass	Low
Functionality: Change Password **Prerequisite:** User has logged in to the system and navigated to the profile page and "change password" section.							
UP-04	Change password with correct current password	1. Enter correct current password and valid new password. 2. Click "Save Changes" button to submit the change password request.	a. Current password [based on UA-06]: Newde@909 b. New password: Abcd1@34 c. Confirm password: Abcd1@34	Password is changed successfully.	A success pop-up message is shown to indicate user that the password has been changes successfully.	Pass	High
UP-05	Change password with wrong	1. Enter wrong current password and new	a. Current password: Random@123	1. Password fails to be changed.	1. An error pop-up message is displayed to notify user on the failure to change password.	Pass	Medium

	current password	password with valid format. 2. Click “Save Changes” button to submit the change password request.	b. New password: Abcd1234# c. Confirm password: Abcd1234#	2. Appropriate error message will be shown on the corresponding input field.	2. Error message, *“Current password is incorrect”* will be shown on the current password input field.		
UP-06	Change password with invalid format	1. Enter the correct current password but new password with invalid format. 2. Click “Save Changes” button to submit the change password request.	a. Current password [based on UP-04]: Abcd1@34 Invalid Input Variations: b. New password: random2 c. Confirm password: random3	1. Password fails to be changed. 2. Appropriate error message will be shown on the corresponding input fields.	1. Before submission, an inline error message stating, *“Password must be at least 8 characters long and include at least one uppercase letter, one number, and one special character,”* is displayed beneath the New Password and Confirm Password input fields. 2. If user insists to submit the form, an error pop-up message is displayed to notify user that the password change has failed, while the inline error message remain visible on the respective input fields.	Pass	Low

5.3.1.3. Functional Testing for Analytics Dashboard Module

Table 5.5. Functional Test Cases for Analytics Dashboard Module

Module Name: Analytics Dashboard
Test Case Description: To confirm all summary cards and charts load correctly with accurate and up-to-date data.
Testing Objective: To ensure visualisation tools represent underlying data properly and interactively.
Remarks: Includes date filtering and PDF export features.
Prerequisite: User logs in and is redirected to the dashboard page.

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Results	Actual Results	Status	Severity of Failure
Functionality: View Summary Cards and Analytical Graphs							
AD-01	View summary cards and analytical charts when no data exists	User views the dashboard page on as a newly registered account with no patient or result records.	No patient is recorded, or image is analysed in the system.	1. Summary cards show value "0" for all: a. Total patients b. Total images analysed c. Average confidence score 2. All five charts are rendered but appear empty.	1. Summary cards correctly show zeros. 2. Charts are displayed with labels but visually empty.	Pass	Low
AD-02	View summary cards and analytical charts when data exists	User views and observes the dashboard page with existing data (patients and results)	At least one patient is added, and one image is analysed.	1. The summary cards show the total numbers of recorded patients, total number of images analysed, and average confidence score for all categories. 2. Five types of charts are displayed, including: a. Bar chart: Total Number of Images Analysed per Category b. Drill-down pie chart: Cells Classification Breakdown c. Line chart: Average Confidence Score per Category	1. Summary cards display correct values. 2. Charts are rendered and populated with data. 3. Drill-down and tooltip interactivity of	Pass	High

				d. Drill-down pie chart: Geographical Distribution e. Stacked bar chart: Patient Age Group Distribution	the charts work as expected.		
Functionality: Filter Analytical Graphs by Date Range							
AD-03	Filter charts by date range	1. Select a date range using the date picker filter. Options include: a. Start date only b. End date only c. Both dates 2. Click “Apply Filters” button.	a. Start date only: 15 May 2025 b. End date only: 30 May 2025 c. Both: 15 May 2025 – 30 May 2025	All charts will update to reflect only the data within the specified date range criteria: a. Start date only: shows data from 15 May 2025 to the most recent available data. b. End date only: shows data from the beginning up to 30 May 2025. c. Both: shows data between 15 May 2025 and 30 May 2025 inclusive.	1. The charts and summary cards rendered correctly based on selected date input. 2. Filtering works for all three cases: start only, end only, and both. 3. Drill-down and tooltip interactivity remain working after filtering.	Pass	Medium
AD-04	Reset filtered charts	1. Complete test case AD-03 to apply a date range filter and update the charts. 2. Click “Show All Time Range” button.	N/A	1. Page reloads automatically. 2. All date filters are cleared. 3. All charts are updated to show data for the full date range (all-time). 4. Summary cards remain unchanged as they always show all-time data.	1. Charts are correctly reset to show all-time data after page reloads. 2. Summary cards remain the same. 3. Date filters are cleared.	Pass	Medium
AD-05	Filter charts with invalid date range	1. Select a start date that is later than the end date via the date pickers. 2. Click “Apply Filters” button.	a. Start date: 30 May 2025 b. End date: 15 May 2025	1. No filters are applied. 2. An error message is displayed to indicate the start date is later than the end date. 3. The graphs on the dashboard page are reset to all-time data.	1. An error pop-up message is shown: *“Start date (2025-05-30 00:00:00) must not be later than end date (2025-05-15 23:59:59)!”*	Pass	Low

					2. The dashboard is refreshed to display all data with no filters applied.		
Functionality: Export Charts as PDF Report							
AD-06	Download Charts	1. Click “Download Charts” button on the dashboard. 2. A pop-up modal appears and displays a list of 5 available charts to be selected. 3. Select one or more charts to be exported. 4. Click “Export” button to download the charts in PDF report.	N/A	1. After clicking “Export”, the modal closes and a toast notification appears, showing the download progress. 2. The toast disappears once the download is completed. 3. The selected charts are exported and downloaded in a single PDF report. 4. The PDF report includes the logged-in user’s username, selected date range, and clear chart visuals with correct titles. 5. If no chart is selected, an error prompt is shown, and export is not triggered.	1. Modal closes upon clicking “Export” button. 2. Toast appears and shows download progress. 3. Toast disappears after download completes. 4. Charts are correctly selected and downloaded as PDF. 5. The PDF report contains correct username, date range, and charts. 6. If no chart is selected, an error pop-up message appears, and the modal remains open for correction.	Pass	High

5.3.1.4. Functional Testing for Image Upload and Classification Module

Table 5.6. Functional Test Cases for Image Upload and Classification Module

Module Name: Image Upload and Classification **Test Case Description:** To verify the functionality of uploading and classifying cervical Pap smear images. **Testing Objective:** To ensure valid image files can be uploaded, assigned to patients, classified correctly, and appropriate feedback is given for invalid inputs. **Remarks:** Includes successful image classification and validation of input errors. **Prerequisite:** User has logged into the system and navigates to the "Classification" page from the sidebar.							
Test Case ID	**Test Scenario**	**Test Steps**	**Test Data**	**Expected Results**	**Actual Results**	**Status**	**Severity of Failure**
IM-01	Upload and classify cervical Pap smear image	1. Click "Upload Image" button to select and upload a cervical Pap smear image. 2. Select a patient from the dropdown list to assign the image. 3. Optionally write a clinical note. 4. Click "Submit & Classify" button.	a. Valid image file in either jpg, png, or bmp format b. One existing patient selected c. Optional notes	1. The selected image is previewed immediately after upload. 2. Form submits successfully when required fields are filled. 3. A loading overlay appears while the image is being classified by the integrated CNN model. 4. Upon receiving the result, the overlay is hidden. 5. A success pop-up alert appears to display the classification category and confidence score. 6. The classification result is saved in the database and linked to the selected patient. 7. User has the option to click "View Result Details" to see the result in detail.	1. Image preview appears correctly after selection from file explorer. 2. Form submission succeeds when a valid image and patient are provided. 3. A loading overlay is displayed while processing. 4. The overlay is removed once classification task completes. 5. A success alert shows the classification result and confidence score. 6. User is able to view detailed result.	Pass	Critical

IM-02	Validate image upload and classification with missing or invalid input	1. Attempt to upload and classify an image under the following invalid conditions: a. No image file selected b. Unsupported image file format (e.g., gif, tiff and avif) c. No patient selected d. No image and patient selected 2. Click the "Submit & Classify" button.	Invalid conditions: a. No image file only b. Image file in an invalid format (e.g., gif) c. Missing patient only d. Missing both image and patient	1. Form submission is blocked. 2. Appropriate error message will be displayed according to the testing conditions. 3. Image classification is not triggered.	1. Form submission is blocked for all three conditions. 2. An error pop-message will be shown with the relevant text: a. No image file only: *"Please upload an image."* b. Image with invalid format: *"Unsupported image format: GIF Allowed formats: JPG, PNG, BMP."* c. Missing patient only: *"Please upload an image."* d. Missing both image and patient: *"Please upload an image and select a patient."* 3. Image is not being classify and user will not receive any result. 4. User remains on the image classification page to retry.	Pass	High

5.3.1.5. Functional Testing for Classification Results Management Module

Table 5.7. Functional Test Cases for Classification Results Management Module

Module Name: Classification Results Management
Test Case Description: To verify the functionalities related to viewing, downloading, reassigning, and deleting classification results.
Testing Objective: To ensure classification results are accurately displayed and that all actions (view, download, reassign, delete) perform correctly and update the database or UI accordingly.
Remarks: Includes validation of result ownership, confidence score display, patient reassignment logic, and secure deletion.
Prerequisite: The user has logged into the system and navigated to the "Classification Results" page from the side bar. [UA-04]

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Results	Actual Results	Status	Severity of Failure
Functionality: View All Results							
CR-01	View all classification results	Observe the listing of all classification results displayed.	N/A	The system displays a table of all classification results, showing their images, category results, confidence score, associated patients, and created At.	1. The table of results is correctly displayed. 2. If there is no result, *"No results found"* message will be displayed. 3. Other actions like view result's details, download result, reassign patient, and delete result can be found on this page too. Title of the action buttons will be shown when the mouse hovers on them.	Pass	Medium
CR-02	Sort results table by column	1. Click on the header of a sortable column:	Sort by: a. Classification Results (category)	1. The table is sorted in ascending order based on the selected column.	1. Results table sorts correctly based on the clicked column. 2. Toggling between the ascending and descending orders works well.	Pass	Low

		a. Classification Results (category b. Confidence Score c. Patients d. Created At 2. Observe the sorted order of all results. 3. Click again the column header to toggle sorting in a different order.	b. Confidence Score c. Patients (name) d. Created At	2. Clicking the same header again toggles between ascending and descending order. 3. Only one column can be sorted at a time.	3. Sorting is indeed limited to one column at a time.		
Functionality: Search and Filter Results							
CR-03	Apply combined filters and search (date range, classification category, and patient name)	1. Select desired date range via the date pickers provided. 2. Select a classification category from the dropdown. 3. Type a patient's name (partial or full) in search box. 4. Click "Apply Filters" button. *Note: User may use one or more filter and search criteria in combination.	a. Start date: 01 May 2025 b. End date: 28 May 2025 c. Category: Koilocytotic d. Patient's Name: Mary	Only classification results that match all criteria are displayed.	1. Classification results are filtered and displayed correctly based on the search and filter criterion. 2. Search and filter criteria can be cleared, and the results listing will be refreshed when "Show All" button is clicked.	Pass	Medium
Functionality: View Selected Result's Details							
CR-04	View selected result's details	1. Three scenarios to trigger view result's details functionality: a. Click the "View Result Details" button from the pop-up alert after	Valid result's ID (automatically handle by the system).	1. The result's details are shown correctly. 2. Reassign patient and download	1. The result's details and associated patient are displayed correctly. 2. The functionalities like reassign patient and	Pass	High

		uploading and classing a cervical Pap smear image. [IM-01] b. On the main results page, click the "View" (eye) icon on the row of selected result. c. From the selected patient's details page. [PM-07] 2. Observe the result's details.		report functions work as expected.	download report can work as expected.		
Functionality: Download Result in PDF Report							
CR-05	Download results	Two scenarios to trigger download results function: a. On result's details page, click "Download report" button. [CR-04] b. On the main results page, click the "Download" icon on the row of selected result.	Valid result's ID (automatically handle by the system).	The results report will be download successfully.	Correct results report is downloaded successfully in PDF format.	Pass	High
Functionality: Reassign Associated Patient							
CR-06	Reassign patient for specific result	1. Two scenarios to trigger download results function: a. On result's details page, click "Reassign patient" button. [CR-04] b. On the main results page, click the "Reassign patient" button on the row of selected result.	a. Valid result's ID (automatically handle by the system). b. Selected patient	1. Upon submitting the changes, the patient will be reassigned successfully. 2. A success pop-up message will be shown. 3. The changes are saved in the databased and	1. The reassignment of patient works correctly. 2. A success pop-up message is also shown to indicate that the patient has been reassigned successfully. 3. The reassigned patient will be shown	Pass	Critical

		2. A pop-up modal appears and displays a dropdown list of existing patients. 3. Select another associated patient. 4. Click “Reassign” button to submit the changes.		reflected immediately on the UI.	correctly on the selected result too.		
Functionality: Delete Selected Result							
CR-07	Delete result	1. Click the “Delete” (trash) icon on the row of selected result. 2. Click “Confirm” button to delete the result.	Valid result’s ID (automatically handle by the system).	1. A confirmation pop-up should be displayed when delete icon is clicked [Test step 1]. 2. Once “Confirm” button on the modal is clicked [Test step 2], the selected result will be deleted successfully. 3. A success deletion message will be shown. 4. The results table will be updated.	1. A delete confirmation pop-up is shown for the selected result. 2. Upon confirmation, the selected result will be deleted successfully. 3. A success pop-message will be displayed with the text *“Result has been deleted successfully*”. 4. The list of results is updated correctly.	Pass	High

5.3.1.6. Functional Testing for Patients Management Module

Table 5.8. Functional Test Cases for Patients Management Module

Module Name: Patients Management
Test Case Description: To verify the functionalities related to viewing, adding, searching, editing, and deleting patient records.
Testing Objective: To ensure that patient management features operate correctly, allowing users to manage patient information efficiently and securely.
Remarks: Includes validations, error handling, sorting and correct database updates or deletions.
Prerequisite: The user has logged into the system and navigated to the "Patients Management" page from the sidebar.

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Results	Actual Results	Status	Severity of Failure
Functionality: View All Patients							
PM-01	View all patients	Observe the listing of existing patients displayed.	N/A	The system displays a table of all recorded patients, showing their nationality, IC or passport number, full name and date of birth.	1. The table of patients is correctly displayed. 2. If there is no patient record, *"No patients found"* message will be shown. 3. Other actions like view patient's details, add new patient, edit, search, and delete can be found on this page too.	Pass	High
PM-02	Sort patients table by column	1. Click on the header of a sortable column: a. Nationality b. IC/Passport number c. Full name d. DOB	Sort by: a. Nationality b. IC/Passport number c. Full name d. Date of Birth (DOB)	1. The table is sorted in ascending order based on the selected column. 2. Clicking the same header again toggles	1. Table sorts correctly based on the clicked column. 2. Toggling between the ascending and descending orders works as expected. 3. Sorting is indeed limited to one column at a time.	Pass	Medium

		2. Observe the sorted order of patient records. 3. Click again the header to toggle sorting in a different order.		between ascending and descending order. 3. Only one column can be sorted at a time.			
Functionality: Add New Patient							
PM-03	Add a new Malaysian patient with valid data	1. Click the "Add Patient" button. 2. Fill in all required fields in the patient form. 3. Click "Add Patient" to save the record.	e. Name: Mary Chin f. Nationality: Malaysian g. DOB: 10/06/1998 h. IC Number: 980610-13-0102 i. Contact number: 0191234567 j. Email: marychin@example.com k. Residential address: No 1A, Taman Muhibah l. Country: Malaysia m. State: Sarawak n. City: Sibu o. Postcode: 96000	1. User is redirected to the "Patients Management" page and a success message is shown. 2. The new patient is added to the database and appears in the patient list.	1. User is correctly redirected to the "Patients Management" page. 2. The success pop-up message is shown to indicate that the patient has been added. 3. The new patient record is also visible in the list of patients.	Pass	High
PM-04	Add a non-Malaysian patient with valid data		Assuming is non-Malaysian but stay in Malaysia: a. Name: Jane Doe b. Nationality: Non-Malaysian c. DOB: 10/05/1999 d. Passport Number: NM123456	1. When the nationality selection is switched to non-Malaysian, the IC input field should be changed to passport number input field. 2. Once form submission is successful, user will be redirect to the	1. The IC input field is switched to passport number when "non-Malaysian" is chosen for the nationality. 2. Upon successful submission of the	Pass	High

			e. Contact number: 011234567 f. Email: janedoe@example.com g. Residential address: No 2A, Taman Muhibah h. Country: Malaysia i. State: Sarawak j. City: Sibu k. Postcode: 96000	"Patients Management" page and a success message will be shown. 3. The new patient is added to the database and appears in the patient list.	new patient record, the user is redirected to the "Patients Management" page. 3. The success pop-up message is also shown. 4. The new patient record is also listed in the table of patients.		
PM-05	Add a new patient with mismatch DOB and IC number	1. Click the "Add Patient" button. 2. Fill in all required fields in the patient form with mismatch DOB and IC number. 3. Click "Add Patient" to save the record.	Test data may follow the format in test case PM-03, except: a. DOB: 05/05/1996 b. IC Number: 960101-13-0202	1. New patient fails to be created. 2. An error message will be shown. 3. User remains on the "Add Patient" page to retry.	1. New patient fails to be created as expected. 2. An error pop-up message will be shown, indicating the failure to add new patient. 3. A specific inline error message will also be displayed on the IC input field: *"The first 6 digits of the IC number must match the date of birth."* 4. User remains on the same page and can retry with correct inputs.	Pass	High
PM-06	Add a new Malaysian patient record with taken IC number	1. Click "Add Patient" button. 2. Fill in required fields. 3. Submit form.	Follow sample input variations in Test Case PM-03, except with a taken IC number. a. DOB: 10/06/1998	1. Form submission is prohibited by an error-pop message and patient fails to be added.	1. The error alert will be displayed with text, *"Failed to add new patient. Please try again"*.	Pass	High

			b. IC Number: 980610-13-0102	2. Appropriate inline error message will be shown on the IC input filed.	2. Correct inline error message is shown, "*This IC number is registered.*". 3. User remains on the "Add Patient" page to retry.		
Functionality: Search and Filter Patients							
PM-07	Search and filter patients by name, IC number, passport number, and nationality	1. Select "Search by Name" from the dropdown and enter part of the patient's name. Click "Search" button. 2. Change to "Search by IC Number" and enter a valid partial or full IC number. Click "Search" button. 3. Change to "Search by Passport Number" and enter a valid partial or full passport number. Click "Search" button. 4. Change to "Search by Nationality", select "Malaysian" or "Non-Malaysian" from the dropdown. Click "Search" button. 5. Verify that results match the input query for each type.	a. Name: chin b. IC: 9906 c. Passport: NM123 d. Nationality: Malaysian	The patients list displays matching results for each search criterion: a. Name contains "chin" b. IC contains "9906" c. Passport number contains "NM123" d. Nationality equals "Malaysian"	1. The search results match correctly with every search criterion. 2. Search and filter criteria can be cleared, and the patients listing will be refreshed when "Show All" button is clicked.	Pass	Medium
Functionality: View Selected Patient's Details							
PM-08	View the details of a specific patient and their classification results	1. On the selected patient's row, click the "View" (eye) icon to view the patient's details. 2. Observe patient's details displayed.	Valid UUID of selected patient (automatically handle by the system).	1. The patient's personal details are shown correctly. 2. The classification results are listed correctly.	1. Correct patient's details are displayed. 2. All classification results associated to the patient are shown.	Pass	High

		3. Observe the classification results list associated to the selected patient. 4. Test sorting for results table (Classification Results, Confidence Score, Created At, Updated At) 5. Click on “View Result”, “Download Result”, “Reassign Patient”, and “Delete Result” buttons for each result.		3. Sorting of the classification results and each button (view, download, reassign, delete) works as expected.	3. The classification results can be sorted correctly by clicking one column header at a time. 4. The actions button for each classification can work as expected. Title of the buttons will be shown when mouse hovers on them.		
Functionality: Edit Patient							
PM-09	Update patient’s information	1. Two scenarios to navigate to the “Edit Patient” page: a. On the main “Patients Management” page, click the “Edit” (pencil) icon on the specific row of selected patient. b. On the selected patient’s details page [PM-08], click the “Edit Patient” button 2. Modify the patient’s information with valid values. 3. Click “Update Patient” to submit the form.	User may modify one or more fields of patient’s information. Example below only modifies patient’s name and contact number. a. Name: Mary Chin Test b. Contact number: 0191432122	The system should save the updated patient record and redirect the user to the previous page with a success message.	1. Modification of patient’s details is updated successfully. 2. User is redirected to the main “Patients Management” page. 3. A success pop-up message is shown to indicate the successful update of information. *Note: Invalid inputs and error handling for “Edit Patient” functionality is similar to “Add Patient”.	Pass	High
Functionality: Delete Patient							

PM-10	Delete selected patient	1. Click the “Delete” (trash) icon on the specific row of selected patient. 2. Click “Confirm” button to delete the patient.	Valid UUID of selected patient (automatically handle by the system).	1. A confirmation pop-up should be displayed when delete icon is clicked [Test step 1]. 2. Once “Confirm” button on the modal is clicked [Test step 2], the selected patient will be deleted successfully. 3. A success deletion message will be shown. 4. The latest list of patients will be updated.	1. A delete confirmation pop-up is shown with the selected patient’s name and IC/Passport number. 2. Upon confirmation, the selected patient will be deleted successfully. 3. A success pop-message will be displayed with the text *“Patient deleted successfully”*. 4. The list of patients is updated correctly.	Pass	High

5.3.2. Non-functional Testing

Non-functional testing is conducted to evaluate the overall quality attributes of the CerviScan system beyond its core functionalities. The focus areas include reliability, usability, and efficiency to ensure the system performs well under real-world conditions. Reliability testing targets the critical modules, such as “User Authentication” and “Image Upload and Classification”, to verify their stability and error handling under repeated and edge-case scenarios. Next, usability testing is performed to assess user interaction and ease of use for key modules including “Patients Management”, “Classification Results Management”, and “Analytics Dashboard” with further feedback and improvements planned to be gathered during upcoming User Acceptance Testing (UAT). Furthermore, efficiency testing is also conducted to measure the system’s responsiveness and performance metrics across the entire platform, including email communication, report generation, and image classification processing times. Detailed test plans and cases for these areas are presented in the tables below.

5.3.2.1. Reliability Testing

Table 5.9. Reliability Test Plan for User Authentication Module

Module Name: User Authentication
Test Case Description: To verify the system's reliability when handling repeated invalid inputs, valid registrations after multiple failures, and secure handling of expired password reset links.
Testing Objective: To ensure the authentication system can withstand repeated edge-case operations, maintain stability, provide consistent feedback, and preserve security integrity under various user actions.
Remarks: Tests validate robustness against repeated invalid inputs, correctness of user registration flow, and secure password reset handling to prevent unauthorised access or system crashes.

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Results	Actual Results	Status	Severity of Failure
R-01	Repeated registration attempts with invalid inputs	1. Attempt to register using invalid inputs. 2. Repeat the attempt 10 times consecutively.	Follow test cases UA-02 and UA-03 for the invalid input variations.	1. Each attempt is rejected with a specific error message. 2. System remains stable with no crashes or delays.	1. All 10 registration attempts were rejected with a pop-up error alert appeared immediately. 2. Correct error messages are displayed. 3. The system remained responsive and stable with no crashes or slowdowns.	Pass	High
R-02	Register with valid inputs	Register an account with all valid inputs after completing test case R-01.	Follow the format in test case UA-01 for the sample valid inputs.	Registration succeeds and user may proceed to login.	1. New user account is registered successfully with a success pop-up message shown. 2. User is redirected to the login page.	Pass	High
R-03	Handling expired reset links	1. Trigger a password reset for a valid account.	Valid email used during password reset request, but	1. System should reject the attempt with a clear error message such as	1. When the expired reset link was accessed, the system	Pass	Critical

		2. Wait five minutes until the reset link expires. 3. Attempt to use the expired link to reset the password.	the reset link is used after expiration.	*"Invalid or expired password reset link"* 2. User is redirected to the login page to try again.	displayed the correct error message. 2. The user was redirected back to the login page, allowing them to generate a new link through the "Forgot password?" link. 3. No system errors or crashes during the process.		

Table 5.10. Reliability Test Plan for Image Upload and Classification Module

Module Name: Image Upload and Classification **Test Case Description:** To verify the stability and consistent behaviour of the image upload and classification process under repeated and boundary conditions. **Testing Objective:** To ensure the module reliably accepts valid images, handles errors for invalid formats, and maintains stability during repeated use. **Remarks:** Tests focus on single image upload flow, repeated classification, and rejection of unsupported formats.							
Test Case ID	**Test Scenario**	**Test Steps**	**Test Data**	**Expected Results**	**Actual Results**	**Status**	**Severity of Failure**
R-04	Repeated image upload and classification	1. Upload one valid Pap smear image. 2. Trigger classification task. 3. Repeat the process 10 times.	Valid .jpg, .png, or .bpm cervical Pap smear image.	1. All uploads succeed without crashing. 2. Classification results are returned each time correctly.	1. All 10 attempts succeeded. 2. Classification was stable and consistent. 3. No system errors or delays.	Pass	High
R-05	Upload unsupported file format	1. Attempt to upload a file in unsupported format. 2. Observe system response.	A .gif or .svg file.	1. Upload is rejected with clear error. 2. System remains stable and responsive.	1. Upload was blocked with an error pop-up message: *"Unsupported image format: SVG. Allowed formats: JPG, PNG, BMP."* 2. No crash or UI glitch observed.	Pass	High

5.3.2.2. Usability Testing

Table 5.11. Usability Test Plan for Analytics Dashboard Module

Module Name: Analytics Dashboard **Test Case Description:** To validate the readability, interactivity, and accessibility of analytics visualisations and summary cards. **Testing Objective:** To ensure users can easily interpret data through graphs, apply filters, and export charts. **Remarks:** Focuses on clarity of graphs, responsiveness, and ease of filtering.							
Test Case ID	**Test Scenario**	**Test Steps**	**Test Data**	**Expected Results**	**Actual Results**	**Status**	**Severity of Failure**
U-01	Interpret and interact with analytics charts	1. View all visual charts, including bar chart, pie chart, line chart, age distribution, and geographical pie chart. 2. Hover over each chart element to trigger tooltips. 3. Observe legends and labels.	N/A	1. All charts are visually clear with appropriate titles, labels, legends, and colour coding. 2. Tooltip displays accurate and detailed information on hover.	1. All charts render cleanly with readable labels and legends. 2. Tooltips show precise values on hover.	Pass	Medium
U-02	Apply date filter to update dashboard	1. Select a date range. 2. Observe chart updates across dashboard.	From 2025-01-01 to 2025-05-31	1. Filter is easy to use with date picker. 2. Charts refresh correctly to reflect selected date range.	1. Charts update accurately after applying date filter. 2. Date pickers are intuitive and quick, as well as selection of future dates are prohibited.	Pass	High
U-03	Export one or more selected charts as PDF	1. Click “Download charts” button. 2. Choose either a single chart or multiple charts using checkboxes on the pop-up modal.	Any one or more dashboard charts	1. System allows selection of one or more charts to download. 2. PDF contains logged-in user’s username, selected date range, and selected	1. User can flexibly choose charts. 2. PDF downloads correctly with selected content only and all chart formatting retained.	Pass	Medium

		3. Click “Export” button to save the file.		chart with correct formatting and title. 3. File downloads with an appropriate name.	3. File name is descriptive by containing the download timestamp.		
U-04	Drill down in both classification and geographical pie charts	1. In Cell Classification Breakdown Chart: Click on “Abnormal” to reveal subcategories like Koilocytotic and Dyskeratotic. 2. In Geographical Distribution Chart: Click from country → state → city. 3. Navigate back up levels.	a. Cell Types: Normal, Abnormal, Benign. b. Locations: Malaysia → Sarawak → Sibu	1. Sub-categories appear after clicking. 2. Navigation is intuitive with visible breadcrumbs or back button. 3. No loading issues or misalignment.	1. Correct breakdowns are displayed by the drill-down charts. 2. Navigation between layers is smooth and user-friendly.	Pass	High

Table 5.12. Usability Test Plan for Patients Management Module

Module Name: Patients Management **Test Case Description:** To evaluate the ease of managing patient records including adding, editing, viewing, and searching patient data. **Testing Objective:** To ensure the module is intuitive, clearly labelled, and provides responsive user feedback for all actions. **Remarks:** Focuses on input form usability, clarity of messages, and navigation experience.							
Test Case ID	**Test Scenario**	**Test Steps**	**Test Data**	**Expected Results**	**Actual Results**	**Status**	**Severity of Failure**
U-05	Add a new patient record with all valid inputs.	1. Click "Add Patient" button on the "Patients Management" page. 2. Fill in required fields. 3. Submit form.	Follow sample valid input variations in Test Case PM-03.	1. Form fields are clearly labelled. 2. Upon submission, a success message appears, and patient is added to the list.	1. Input fields are easy to understand and correctly validated. 2. Success alert is display for successful addition of patient. 3. New patient is reflected in the listing.	Pass	High
U-06	Edit existing patient details	1. Click the edit icon for a patient. 2. Modify the data. 3. Save changes.	Follow sample valid input variations in Test Case PM-09.	1. User can edit fields with ease. 2. Success pop-up message appears upon successful update.	1. Changes are saved smoothly with appropriate success alert shown. 2. UI flow is intuitive and similar to "add patient" feature.	Pass	High
U-07	Form validation feedback	Submit "Add Patient" form with one required field empty.	Follow sample valid input variations in Test Case PM-03 but omit corresponding residential address field.	1. Form prevents submission. 2. An error alert is displayed. 3. Inline error appears with a clear message.	1. Patient fails to be added. 2. The error pop-up message shows *"Failed to update patient's information. Please try again"*. 3. Correct inline error message is displayed directly under the field: *"This field is required."*	Pass	Medium

					4. Inline error message will be cleared or updated when new input is keyed in.			
U-08	Search for a patient by name	1. Select “Search by name” from the dropdown list. 2. Enter a full or partial name in the search bar. 3. Click “Search” button or press “Enter” on the keyboard.		“Mary”	Matching records are shown in real-time or quickly after search.	Search results updated dynamically with expected entries.	Pass	Medium

Table 5.13. Usability Test Plan for Classification Results Management Module

Module Name: Classification Results Management **Test Case Description:** Tests the ease of understanding classification results, confidence scores, detailed explanations, and the usability of pagination when browsing multiple results. **Testing Objective:** To ensure users can clearly interpret result details and navigate pages smoothly without losing applied filters or context. **Remarks:** Covers clarity of data presentation, consistency of filtering and sorting across pages, and support for informed user interaction.							
Test Case ID	**Test Scenario**	**Test Steps**	**Test Data**	**Expected Results**	**Actual Results**	**Status**	**Severity of Failure**
U-09	Navigate results using pagination while preserving search, sort, or filter criteria	1. Go to the "Classification Results" page. 2. Apply a search keyword, sorting (e.g., by date), or a filter (e.g., by cell type or date range). 3. Navigate to the next page using pagination controls.	⩾15 results with varied cell categories and dates to allow filtering and pagination	1. Pagination works smoothly and is clearly labelled. 2. When user moves to the next/previous page, the applied search, sort, and filter settings remain active, and the results reflect these criteria. 3. No resets or data loss occur when navigating pages.	1. Search, sort, and filter settings were preserved as the user paginated through results. 2. The system displayed only the matching records across pages.	Pass	High
U-10	View detailed result with confidence score and category explanation	1. Click the "View" icon on a result entry to view details. 2. Observe the display of predicted category and confidence score.	Valid result's ID (automatically handled by the system).	1. Classification result is shown as a badge. 2. Confidence score is clearly displayed with percentage, colour-coded badge, progress bar, and interpretation (Low/Medium/High). 3. A readable explanation of the predicted category is displayed in its own space.	All sections (classification badge, score bar, and explanation) are clearly visible in well-separated cards, improving understanding.	Pass	Medium

5.3.2.3. Efficiency Testing

Table 5.14. Efficiency Test Plan for Whole System

Module Name: System-Wide Efficiency (Model, Download, Email) **Test Case Description:** To measure the time efficiency of core operations including image classification, report generation/download, and email communication. **Testing Objective:** To ensure all performance-critical operations complete within acceptable time thresholds under normal conditions. **Remarks:** Helps verify responsiveness of key functions from both back-end processing and user interaction perspectives.							
Test Case ID	**Test Scenario**	**Test Steps**	**Test Data**	**Expected Results**	**Actual Results**	**Status**	**Severity of Failure**
E-01	Image classification completion time	1. Upload an image and link it to a specific patient. 2. Start classification. 3. Measure the duration from submission to result display.	One valid cervical Pap smear image and selected patient.	Result is displayed within 1–2 seconds (preferably under 1s).	Result displayed less than 1s.	Pass	High
E-02	Classification report (PDF) download generation time	1. Click “Download Result” on any selected result. 2. Measure time from click to file download prompt.	Any completed classification result	Report should be generated within 3 seconds.	Results PDF report is downloaded within 1s.	Pass	Medium
E-03	Export time for analytics charts from dashboard	1. Click “Download Charts” button. 2. Select one or more charts to be exported. 3. Click “Export” button. 4. Measure the completion time for the download.	One or more selected charts	The export process should complete within 3 to 5 seconds, depending on the number of selected charts.	The charts were successfully exported to a PDF report within 1.5 to 4 seconds, varying by the number of charts selected.	Pass	Medium

E-04	Email send and receive time (e.g., reset link)	1. Trigger password reset request by entering the registered email. 2. Monitor inbox for reset email. 3. Record time from request to email arrival.	Registered user's email	Email should be received within 1 minute.	Email is received in less than 1 minute.	Pass	High

5.4. User Acceptance Testing (UAT)

User Acceptance Testing (UAT) is conducted to validate that the CerviScan system meets user expectations in terms of functionality, usability, and overall user experience. The full questionnaire used during the UAT is shown in Appendix A**:** CerviScan User Acceptance Testing (UAT) Feedback Form. A total of 30 participants are involved in the UAT process. These participants include individuals with varying degrees of technical proficiency and backgrounds, including students, lecturers, and individuals simulating roles of medical personnel.

UAT Methodology

Due to technical limitations (i.e., limited memory provided on free tier hosting platform), it is not possible to host the full CerviScan system on a live server. As an alternative approach, two demonstration videos were created to guide the participants through the system's features. The videos include a narrated walk-through video explaining each module and key functionality in detail, as well as a sped-up version of the system demonstration with background music only.

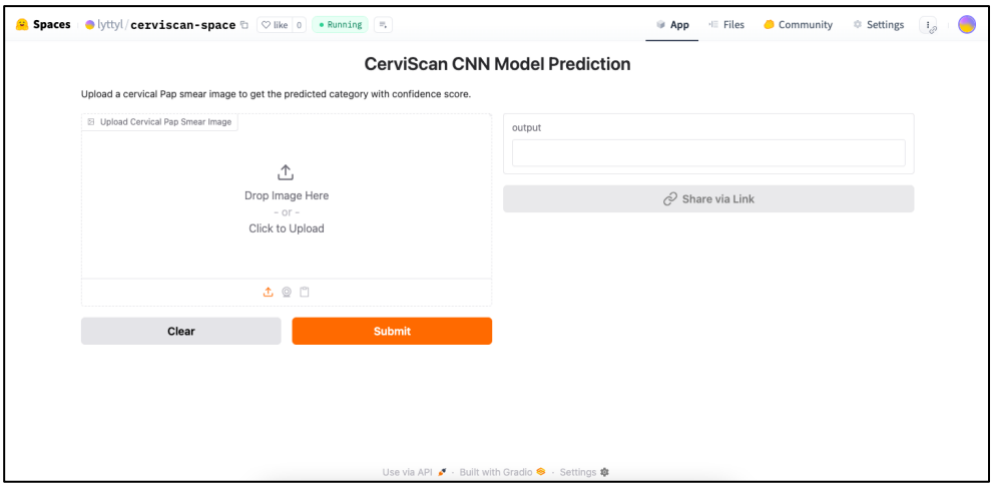


Figure 5.6. Pap smear image upload and classification feature deployed on Hugging Face Spaces

Besides that, to allow participants to interact with the main classification functionality, the trained CNN model is deployed on Hugging Face Spaces. Then, participants are provided with the link to the live Hugging Face deployment where they can upload Pap smear images and observe real-time classification results.

After that, participants are asked to complete a UAT feedback form. The form includes 5-star Likert-scale questions to evaluate the system aspects such as usability, responsiveness, visual design, and result clarity. Optional open-ended questions are also included to collect qualitative comments and suggestions for improvement.

UAT Results

5.4.1. Model Accuracy Testing

The first section of the UAT form aims to gather participants' perspectives on the performance of the CNN model based on their interaction with the classification feature.

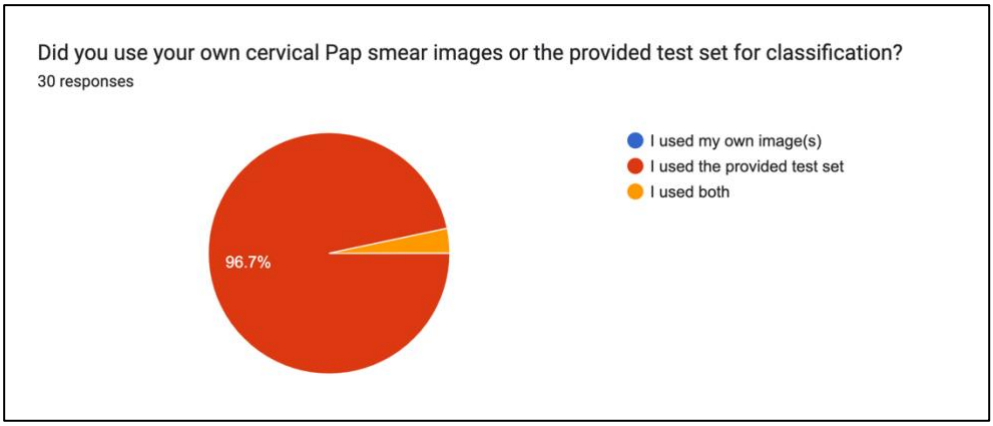


Figure 5.7. Pie Chart Distribution of Participants Using Own Cervical Pap Smear Images or Provided Test Set

Firstly, participants are asked whether they used their own cervical Pap smear images, or the test images provided. The majority of users (96.7%) report using the provided test set, which included pre-labelled images taken from SIPaKMeD dataset, while a small portion

(3.3%) experiment with both their own and provided images (Figure 5.7). Although this approach ensures consistency in evaluation and allows users to fairly assess the model's accuracy, it also introduces a potential limitation in real-world representativeness. Since most users do not upload their own images, the model is not tested against more varied or imperfect samples. This may pose an unforeseen risk, as real-world data could include noise, artefacts, or different staining conditions that the model has not encountered. Therefore, future evaluations should encourage testing with a wider range of images to better assess the model's generalisation and reliability.

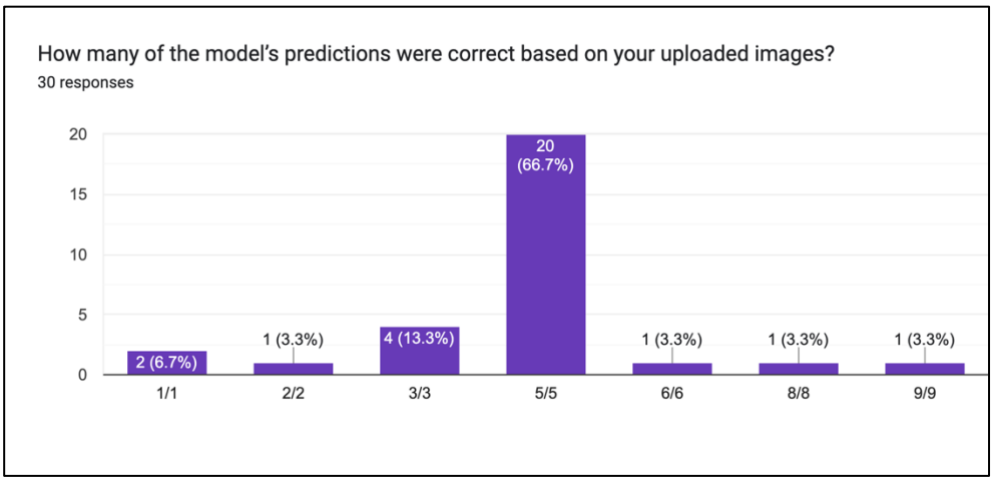


Figure 5.8. Total Correct Predictions Based on Uploaded Pap Smear Images

According to the participants' responses shown in Figure 5.8, the model correctly predicts the cell category for every image uploaded by each participant. All users report perfect prediction rates (e.g., 1/1, 2/2, 3/3, 5/5, 8/8, 9/9), indicating that the model consistently produce correct classifications based on the images tested during the UAT.

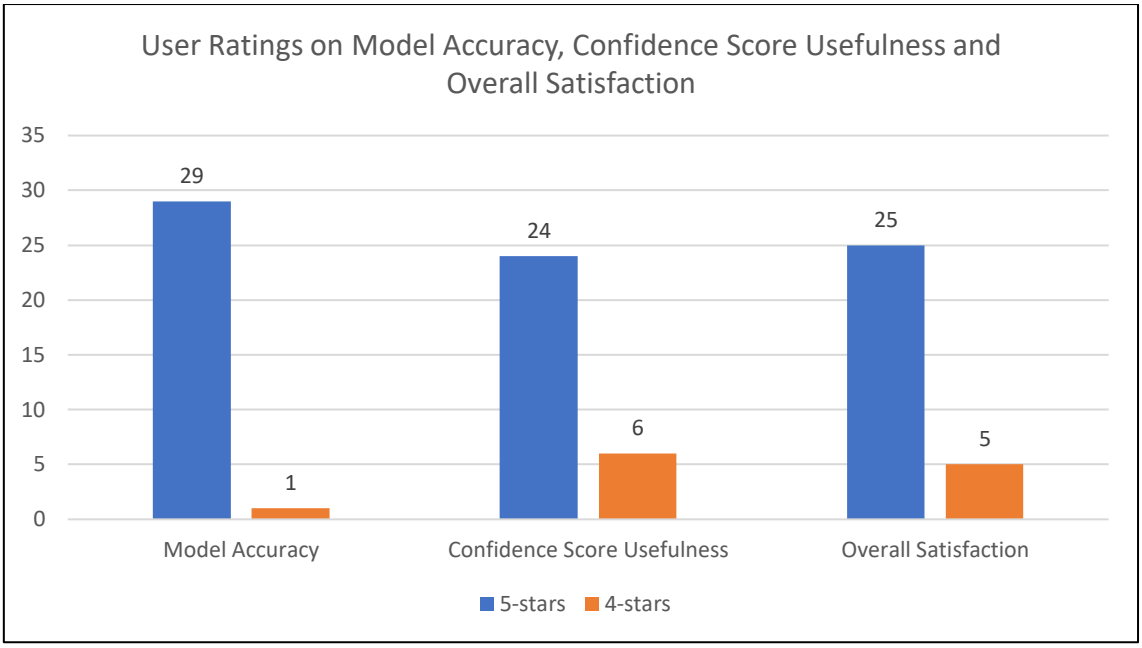


Figure 5.9. Bar Chart of User Ratings on Model Accuracy, Confidence Score Usefulness and Overall Satisfaction

The bar chart in Figure 5.9 presents the distribution of user ratings for three core aspects of the model's performance, that are model's prediction accuracy, confidence score usefulness, and overall satisfaction. All three metrics receive positive feedback, with no rating falling below four stars. For model accuracy, 29 out of 30 participants rate the model with 5 stars, while one participant gives it 4 stars, resulting in an average rating of 4.97. This indicates that users find the model's classifications highly accurate.

In terms of the confidence score's usefulness, the average rating is 4.80, with 24 participants rating 5 stars and six of them assigning 4 stars. While still strongly positive, the slightly lower rating suggests that some users may have been less familiar with interpreting confidence scores or found them slightly unclear.

Next, overall satisfaction with the model's performance receive an average score of 4.83, with 25 participants giving 5 stars and five giving 4 stars. These ratings show that the majority of users are highly satisfied with the model and its integration within the system. The

consistency of high scores across all three questions reflects strong user confidence, trust, and acceptance of the model's capabilities, validating its suitability for deployment within the CerviScan system.

5.4.2. Usability and Visual Design

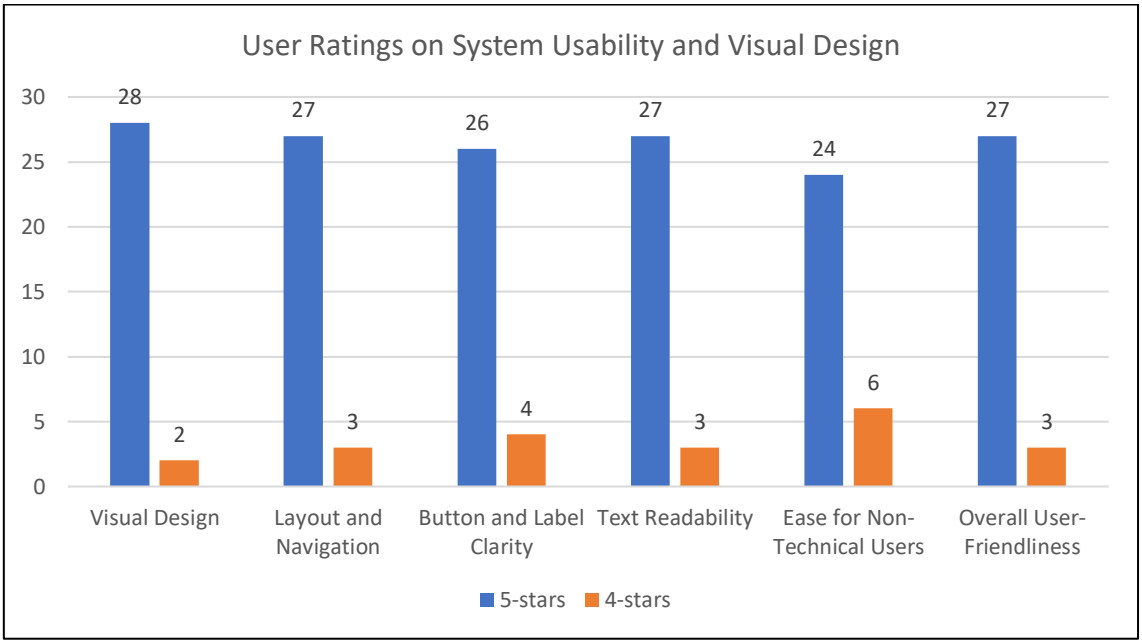


Figure 5.10. Bar Chart of User Ratings on System Usability and Visual Design

Figure 5.10 illustrates user feedback on the system's usability and visual design, based on a set of six 5-stars Likert-scale questions. The responses are consistently high, with no rating falling below 4 stars, and indicate a strong approval across all visual and usability aspects.

The visual design of the user interface receives an average score of 4.93, with 28 participants rating it 5 stars. This suggests that the users find the interface clean and well-structured. Similarly, the system's layout and navigation flow are rated 4.90 on average, showing that the users are able to understand and find the navigation easy.

Moreover, feedback on the clarity and placement of buttons, icons, and labels achieve an average rating of 4.87, while text readability, considering font size, colour, and contrast,

receive an average score of 4.90. These results confirm that essential UI elements are designed to be visually accessible and easy to interpret.

Besides that, the only slightly lower score is for the question on ease of use for non-technical users, which still scores a solid 4.80. This reflects the system's broad accessibility, although it suggests that a small number of participants may have found certain features slightly less intuitive when approached without a technical background. Then, the overall user-friendliness rating is 4.90, indicating that the interface meets general usability standards.

5.4.3. Functionality and Workflow

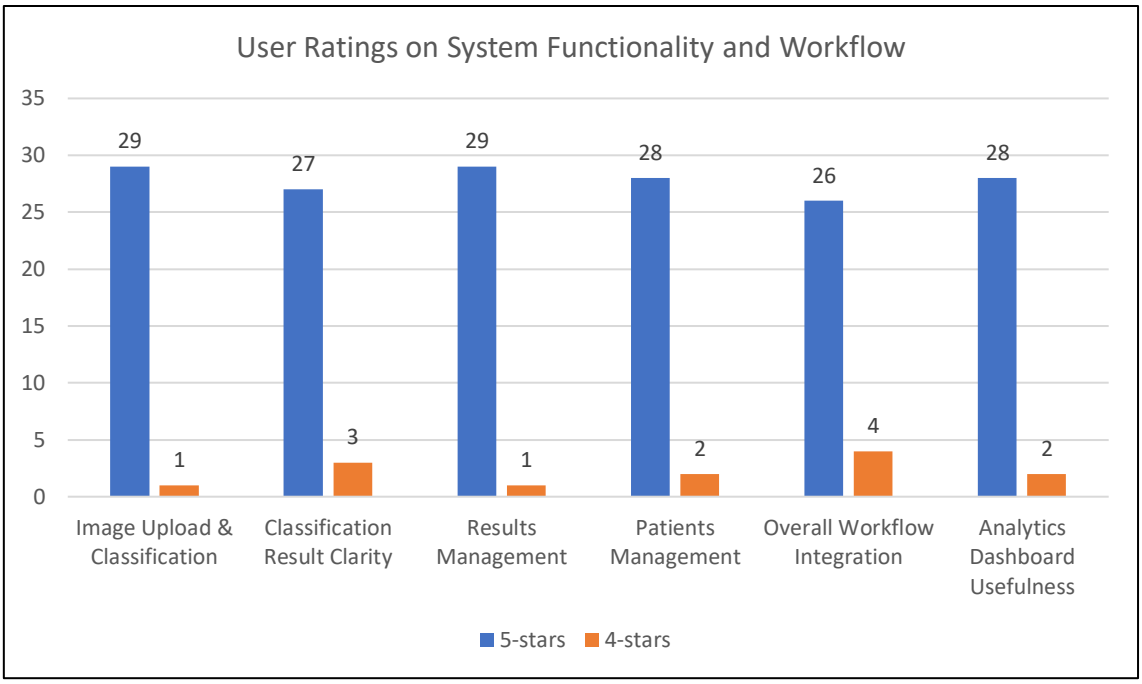


Figure 5.11. Bar Chart of User Ratings on System Functionality and Workflow

The bar chart in Figure 5.11 shows the user ratings on the core functionalities and workflow integration within the CerviScan system. The responses are overwhelmingly positive, with all average scores ranging between 4.87 and 4.97, which indicates a high level of user satisfaction and overall usability.

The image upload and classification process receive the highest average rating of 4.97 with 29 participants give it 5 stars. This confirms that users find this key feature both clear and easy to use. Similarly, the “Classification Results Management” module also achieves a 4.97 rating, which shows that participants valued the ability to view, filter, and manage classification results effectively.

Understanding of the classification output, including cell category and confidence score, is rated at 4.90, and this suggests that most users are able to interpret the results correctly and confidently. Next, the “Patients Management” module receives an average of 4.93, which means it is perceived as practical and straightforward to use for handling patient records.

Furthermore, the analytics dashboard, while still receiving positive feedback with a 4.87 average, has the lowest score among all categories. This may indicate that some users find data interpretation slightly more complex or needed clearer guidance to fully understand its features.

Overall, the workflow integration, from uploading images to analysing results, scores 4.93, and this highlights that users feel all the system components work well together to support the diagnostic process. The consistently high ratings across all aspects reinforce the system’s capability to provide a seamless and user-friendly experience.

5.4.4. Clinical and Technical Relevance

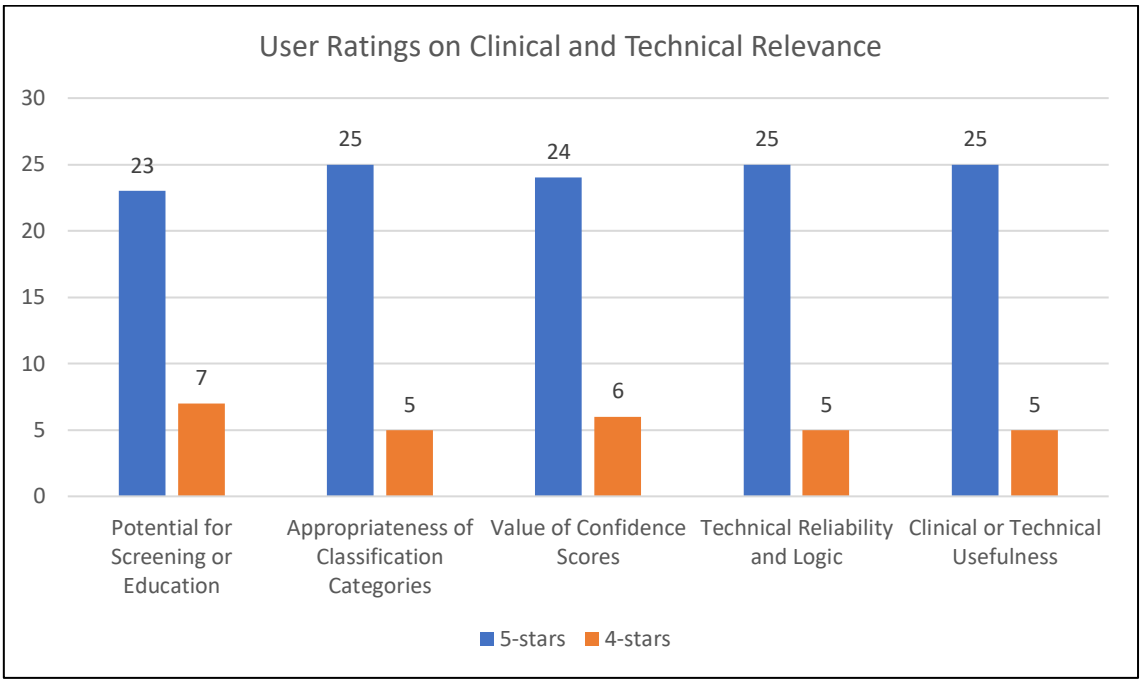


Figure 5.12. Bar Chart of User Ratings on Clinical and Technical Relevance

Figure 5.12 presents the user feedback regarding the clinical and technical relevance of the CerviScan system. The responses show strong agreement with the system's applicability and reliability, as all five questions receive average ratings above 4.75.

The highest number of users, 25 out of 30, give 5-star ratings to the appropriateness of the classification categories used in the system, such as Superficial-Intermediate, Dyskeratotic, Metaplastic, Parabasal, and Koilocytotic. This yields an average score of 4.83, which suggests that the chosen categories are considered clinically meaningful and relevant.

Similarly, the technical reliability and logic of the system's design also receive a 4.83 average rating. This shows that users find the system to be well-structured and dependable in its operations. The same average rating is also given to the question regarding the clinical or technical usefulness of the features and outputs, indicating that participants think that the system to have a practical value for real-world applications.

Additionally, the use of confidence scores in the classification results receives a 4.80 average. This shows that the users generally find the confidence scores helpful in assessing the reliability of the predictions. However, a few participants may have needed greater familiarity or clearer guidance to fully interpret the meaning and implications of these values.

Besides that, the system's potential to support early cervical cancer screening or be used for educational purposes scores a 4.77 average, which is still very positive. This slightly lower score may reflect the limitations of testing through a demo environment rather than a fully integrated clinical setting.

5.4.5. Feedback and Suggestions

Several open-ended questions are included in the UAT form to gather qualitative feedback from participants regarding their experiences and suggestions on the CerviScan system. As these questions are optional, not all participants provided answers. However, the responses receive still offer valuable insights.

For the question on the most useful or impressive features, the majority of responses highlighted the image upload and classification functionality. Participants specifically note the classification process, the ease of image upload, and the presence of detailed classification categories along with confidence scores. This reflects that users are particularly impressed with the system's core feature.

Regarding future improvements, one participant suggests to enable multiple image uploads. This comment is found constructive as the current system allows only one image at a time. Supporting batch uploads could enhance efficiency and usability, especially for clinical or research use cases where multiple samples may need to be analysed.

5.5. Summary

This chapter provides a complete evaluation and testing of the CerviScan system, including both the CNN model and the web-based system. The CNN model is assessed using important performance metrics such as accuracy, recall, precision, and F1-score, all of which were above 97%. These results show that the model is highly accurate and reliable in classifying cervical Pap smear images into the correct cell categories. Next, functional testing of the CerviScan system shows that all six main modules functioned correctly and passed every test case, including handling invalid inputs and missing data through proper error handling. Meanwhile, the non-functional testing demonstrates that the system is stable, efficient, and user-friendly, with smooth performance and a well-structured interface. Lastly, User Acceptance Testing (UAT) reveals strong user satisfaction, particularly with the classification feature, and includes suggestions for future improvements such as enabling multiple image uploads.

CHAPTER 6: CONCLUSION AND FUTURE WORK

6.1. Introduction

This chapter concludes the CerviScan project by summarising the work completed, answering the research questions, evaluating the achievement of the stated objectives, discussing the project's limitations, and proposing potential areas for improvement and future enhancements. The CerviScan system aimed to assist in the early detection of cervical cancer by providing an intelligent, user-friendly web-based platform for Pap smear image classification using a trained Convolutional Neural Network (CNN) model.

6.2. Findings on Research Questions

This section reflects on the key findings from the project in relation to the research questions presented in Chapter 1.4. Each research question is addressed individually, based on the implemented solution and the results obtained.

RQ1: How can a CNN model be developed to accurately classify cervical Pap smear images into specific diagnostic categories?

To address this question, a CNN model based on the ResNet-50 architecture is developed using transfer learning. The model is trained on the SIPaKMeD dataset, which contains labelled images corresponding to five diagnostic categories, such as Superficial-Intermediate, Parabasal, Koilocytotic, Dyskeratotic, and Metaplastic. Transfer learning enables the model to leverage pre-learned image features from large-scale datasets and adapt them to the cervical cell classification task, significantly improving training efficiency and accuracy. The successful implementation of the model, and its ability to distinguish between visually similar cell types, demonstrates the effectiveness of CNNs in automating the classification of Pap smear images.

RQ2: What evaluation metrics can effectively assess the classification performance of the CNN model?

The model's performance is evaluated using several metrics, including accuracy, precision, recall, and F1-score, along with a detailed confusion matrix. On the test set of the SIPaKMeD dataset, the model achieves a high overall accuracy of 97.20%, with a precision of 97.29%, recall of 97.27%, and F1-score of 97.26%, indicating strong and consistent classification performance across categories.

The confusion matrix provides further insights into category-level classification effectiveness. The Superficial-Intermediate class achieves a perfect classification rate with 100% accuracy (129/129), while Parabasal achieves 99.00% accuracy (99/100), with only one image misclassified as Metaplastic. Dyskeratotic and Metaplastic also record high accuracies of 98.51% and 97.39%, respectively, with only minor misclassifications. The Koilocytotic category has the lowest classification accuracy at 91.47%, with most errors involving confusion with visually similar categories like Dyskeratotic and Metaplastic.

These results highlight the importance of using multiple evaluation metrics to capture the model's overall performance and category-specific strengths and weaknesses. Notably, the F1-score is particularly effective in balancing precision and recall, especially for classes with morphological complexity or fewer samples. Together, these metrics provide a comprehensive understanding of the CNN model's classification capabilities.

RQ3: How can a user-friendly web-based system be implemented to facilitate real-time cervical cancer detection using the CNN model?

A web-based system is developed using the Django framework to integrate the trained CNN model. The system supports image upload, real-time classification, result management,

and data visualisation. It includes six core modules, that are user authentication, profile management, image upload and classification, classification results management, analytics dashboard, and patients management. The front-end is built using Bootstrap 5 to ensure a responsive and intuitive user interface across various devices.

Although formal clinical usability testing is not feasible, the system underwent extensive internal testing. Structured User Acceptance Testing (UAT) is also conducted with at least 1–2 domain experts, supported by feedback from general users (e.g., students and lecturers). The results confirm the system's usability, practicality, and effectiveness in delivering a real-time cervical cancer detection solution.

6.3. Objective Achievements

The following table summarises the aims and objectives outlined in Chapter 1.5 alongside the corresponding achievements.

Table 6.1. Summary of Project Objectives and Their Achievements

Aims and Objectives	Achievements
To review and synthesise existing research on cervical cancer detection techniques using image processing and CNN-based methods.	A comprehensive literature review is conducted, covering the principles and architecture of Convolutional Neural Networks (CNNs). Related studies utilising pre-trained CNN models and hybrid approaches for cervical cancer detection are critically analysed, with comparisons made between various models. Additionally, the deep learning frameworks, such as PyTorch and TensorFlow, are evaluated based on their architecture, ease of use, flexibility, and scalability. PyTorch is ultimately selected as the preferred framework, and ResNet-50 is chosen as the base architecture for the CNN model implemented in this project.
To design and develop a CNN model using transfer learning for classifying cervical Pap smear images into five diagnostic categories.	A CNN model based on the ResNet-50 architecture is implemented using transfer learning. It is trained and fine-tuned on the SIPaKMeD dataset to classify images into five categories, that are Superficial-Intermediate, Parabasal, Koilocytotic, Dyskeratotic, and Metaplastic.

To integrate the trained CNN model into a web-based system that supports image upload, classification and result management.	The trained CNN model is successfully integrated into a Django-based web application. The system allows users to upload Pap smear images, classify them using the CNN model, and manage classification results through features such as viewing, deleting, exporting as PDF report, and reassigning patient. In addition, the system includes an interactive analytics dashboard for visualising classification trends and insights, as well as a patient management module that supports adding, editing, and tracking patient records.
To evaluate the model's performance using metrics including accuracy, precision, recall, and F1-score on a publicly available dataset.	The CNN model's performance is assessed on the SIPaKMeD test set, achieving an accuracy of 97.20%, precision of 97.29%, recall of 97.27%, and F1-score of 97.26%, indicating strong classification capability.
To assess the usability and functionality of the web-based system through structured user testing.	The complete system is internally tested for functionality and non-functionality (i.e., usability, efficiency, reliability) across all modules. Structured User Acceptance Testing (UAT) was conducted with domain experts and supported by informal feedback from general users, confirming the system's practicality and ease of use.

6.4. Project Limitations

While the project has successfully met its primary objectives, several limitations are identified that constrain its current scope and potential applicability:

- **Limited dataset diversity**: The CNN model is trained exclusively on the SIPaKMeD dataset, which may not represent the full diversity of Pap smear images in clinical settings. Differences in staining techniques, imaging equipment, and patient demographics across different populations could affect the model's generalisability.
- **Single image upload**: The system currently supports the upload and classification of only one image at a time. This limitation may hinder usability in high volume or clinical batch-processing scenarios where multiple samples are analysed simultaneously.

- **Lack of clinical validation**: The model's predictions have not undergone formal validation by certified medical professionals. Thus, the system is not yet suitable for clinical deployment and should be considered as experimental or educational usage only.
- **No multilingual support**: The web-based system supports only English, which may limit accessibility for non-English speaking users.
- **No mobile or desktop application support**: The system is currently available only through web browsers and lacks dedicated mobile or desktop applications. This limitation may reduce accessibility and convenience for users in mobile-first or offline-preferred environments. In low-resource settings or areas with unreliable internet connectivity, the absence of installable applications may hinder consistent usage and limit the platform's reach.
- **Inability to recognise non-cellular images**: The system is not designed to distinguish between cervical Pap smear images and unrelated objects, like t-shirts, bottles or drawings. Although such inputs typically result in low confidence score and the system will issue warnings to the users, there is no explicit detection or error handling mechanism for rejecting non-cellular images.

6.5. Conclusion

The CerviScan project successfully demonstrates the feasibility of integrating a Convolutional Neural Network (CNN) model with a web-based system for the automated classification of cervical Pap smear images. Leveraging transfer learning with the ResNet-50 architecture and the PyTorch framework, the model achieves strong classification performance, attaining 97.20% accuracy, 97.29% precision, 97.27% recall, and a 97.26% F1-score on the

SIPaKMeD test set. These results confirm the model's capability to distinguish between five diagnostic cell types with high reliability.

The trained model is deployed in a fully functional Django-based web platform that supports image upload, real-time classification, results management, and visual analytics. The system's usability is enhanced through a responsive Bootstrap 5 interface and structured internal testing. Although formal clinical validation is not conducted, feedback from domain experts and general users confirms the system's accessibility, functionality, and potential for educational or preliminary diagnostic use. Overall, the project achieves its core objectives and established a strong foundation for future advancements in AI-assisted cervical cancer screening tools.

6.6. Future Works

While this project lays important groundwork, several areas can be explored to expand its capabilities and improve its practical impact:

- **Clinical validation and collaboration with experts**: Future development should involve collaboration with medical professionals to clinically validate the model's predictions. This step is essential for gaining regulatory approval and for safe deployment in real-world clinical settings.
- **Expansion and diversification of dataset**: Incorporating additional datasets from different sources, populations and imaging conditions will help improve the model's robustness and ensure better performance across various clinical scenarios.
- **Multiple image upload**: Supporting multiple uploads at a time will enhance usability for high-volume users and enable more efficient workflows in labs and research centres.

- **Mobile and desktop application development**: Creating dedicated mobile and desktop apps will improve accessibility, especially in low-resource or remote environments where stable internet access is not always available.
- **Multilingual support**: Adding support for multiple languages, such as Malay and Mandarin, will broaden the system's reach and usability.
- **Detection of irrelevant or non-cellular images**: Enhancing the model to automatically flag or reject non-cytological images, like photographs of everyday objects, would prevent misuse and improve reliability. This could involve training an auxiliary classifier or incorporating a pre-filtering functionality to distinguish valid Pap smear images from unrelated inputs rather than relying completely on low confidence scores.

REFERENCES

Agarwal, R. (2023, September 13). *Complete Guide to the Adam Optimization Algorithm*. Built In. https://builtin.com/machine-learning/adam-optimization

Awati, R., & Craig, L. (2024, January). *What is a convolutional neural network (CNN)?* TechTarget. https://www.techtarget.com/searchenterpriseai/definition/convolutional-neural-network

Barathi. (2024, December 18). *Latest Guide on Confusion Matrix for Multi-Class Classification.* Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/06/confusion-matrix-for-multi-class-classification/

Egeonu, E. (2022, January 5). *The Advantages and Disadvantages of the RAD Model*. DistantJob - Remote Recruitment Agency. https://distantjob.com/blog/rad-model-advantages-and-disadvantages/

Gad, A. F., & Skelton, J. (2024, August 29). *Evaluating Deep Learning Models: The Confusion Matrix, Accuracy, Precision, and Recall*. Digital Ocean. https://www.digitalocean.com/community/tutorials/deep-learning-metrics-precision-recall-accuracy

Gen, D. L. (2023, December 2). *Five Methods for Data Splitting in Machine Learning*. Medium. https://medium.com/@tubelwj/five-methods-for-data-splitting-in-machine-learning-27baa50908ed

Gupta, A., Parveen, A., Kumar, A., & Yadav, P. (2022). Advancement in Deep Learning Methods for Diagnosis and Prognosis of Cervical Cancer. *Current genomics*, *23*(4), 234–245. https://doi.org/10.2174/1389202923666220511155939

Gurucharan, M. (2020, December 7). *Basic CNN Architecture: Explaining 5 Layers of Convolutional Neural Network*. UpGrad. https://www.upgrad.com/blog/basic-cnn-architecture/

Iurev, A. (2022, January 11). *What Is Rapid Application Development?* Pocketfied. https://pocketfied.com/what-is-rapid-application-development/

Jadeja, V., Rao, A. L. N., Srivastava, A., Singh, S., Chaturvedi, P., & Bhardwaj, G. (2023). Convolutional Neural Networks: A Comprehensive Review of Architectures and Application. *Proceedings of International Conference on Contemporary Computing and Informatics, IC3I 2023*. https://doi.org/10.1109/IC3I59117.2023.10397695

Kalbhor, M., Shinde, S., Wajire, P., & Jude, H. (2023). CerviCell- detector: An object detection approach for identifying the cancerous cells in pap smear images of cervical cancer. *Heliyon,9(11).* https://doi.org/10.1016/j.heliyon.2023.e22324

Kashyap, P. (2024, October 30). *Early Stopping in Deep Learning: A Simple Guide to Prevent Overfitting*. Medium. https://medium.com/@piyushkashyap045/early-stopping-in-deep-learning-a-simple-guide-to-prevent-overfitting-1073f56b493e

Kundu, N. (2023, January 23). *Exploring ResNet50: An In-Depth Look at the Model Architecture and Code Implementation*. Medium. https://medium.com/@nitishkundu1993/exploring-resnet50-an-in-depth-look-at-the-model-architecture-and-code-implementation-d8d8fa67e46f

Mathivanan, S. K., Francis, D., Srinivasan, S., Khatavkar, V., P, K., & Shah, M. A. (2024). Enhancing cervical cancer detection and robust classification through a fusion of deep learning models. *Scientific Reports, 14(1),* 10812. https://doi.org/10.1038/s41598-024-61063-w

Pandya, Y. (2024, January 24). *What are the strengths and use cases of TensorFlow and PyTorch in the context of AI automation?* Medium. https://medium.com/@yagnesh.pandya/what-are-the-strengths-and-use-cases-of-tensorflow-and-pytorch-in-the-context-of-ai-automation-009f3ffbd0e7

Park, Y. R., Kim, Y. J., Ju, W., Nam, K., Kim, S., & Kim, K. G. (2021). Comparison of machine and deep learning for the classification of cervical cancer based on cervicography images. *Scientific Reports*, *11*(1), 16143. https://doi.org/10.1038/s41598-021-95748-3

Promworn, Y., Pintavirooj, C., & Piyawattanametha, W. (2019). A comparative study of 3 deep learning models for Pap smear screening. *BMEiCON 2018 - 11th Biomedical Engineering International Conference*. https://doi.org/10.1109/BMEiCON.2018.8609945

PyTorch. (n.d.). PyTorch. https://pytorch.org/features/

Remya, R., & Raimond, K. (2023). Cervical cell classification using Deep Learning Techniques. *2023 International Conference on Computer Communication and Informatics, ICCCI 2023*. https://doi.org/10.1109/ICCCI56745.2023.10128238

Roffo, G. (2017). *Ranking to Learn and Learning to Rank: On the Role of Ranking in Pattern Recognition Applications*. http://arxiv.org/abs/1706.05933

TensorFlow and PyTorch. (n.d.). CS230 Stanford. https://cs230.stanford.edu/section/5/

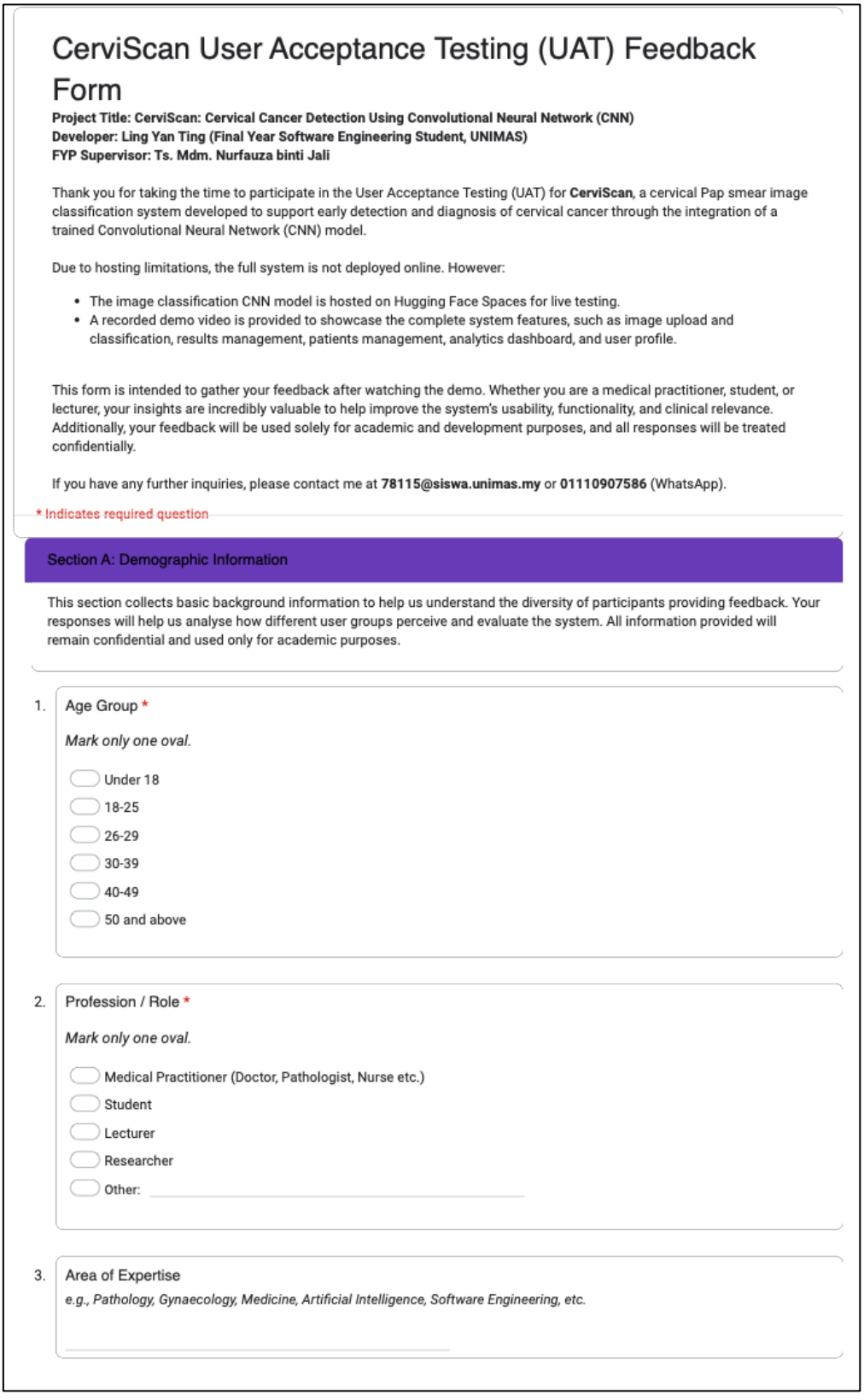
World Health Organization. (2024, March 5). *Cervical Cancer*. World Health Organization. https://www.who.int/news-room/fact-sheets/detail/cervical-cancer

Yang, J., Ju, J., Guo, L., Ji, B., Shi, S., Yang, Z., Gao, S., Yuan, X., Tian, G., Liang, Y., & Yuan, P. (2022). Prediction of HER2-positive breast cancer recurrence and metastasis risk from histopathological images and clinical information via multimodal deep learning. *Computational and Structural Biotechnology Journal*, *20*. https://doi.org/10.1016/j.csbj.2021.12.028

Yu, S., Feng, X., Wang, B., Dun, H., Zhang, S., Zhang, R., & Huang, X. (2021). Automatic classification of cervical cells using deep learning method. *IEEE Access, 9*. https://doi.org/10.1109/ACCESS.2021.3060447

APPENDICES

Appendix A: CerviScan User Acceptance Testing (UAT) Feedback Form



CerviScan User Acceptance Testing (UAT) Feedback Form

Project Title: CerviScan: Cervical Cancer Detection Using Convolutional Neural Network (CNN)
Developer: Ling Yan Ting (Final Year Software Engineering Student, UNIMAS)
FYP Supervisor: Ts. Mdm. Nurfauza binti Jali

Thank you for taking the time to participate in the User Acceptance Testing (UAT) for **CerviScan**, a cervical Pap smear image classification system developed to support early detection and diagnosis of cervical cancer through the integration of a trained Convolutional Neural Network (CNN) model.

Due to hosting limitations, the full system is not deployed online. However:

- The image classification CNN model is hosted on Hugging Face Spaces for live testing.
- A recorded demo video is provided to showcase the complete system features, such as image upload and classification, results management, patients management, analytics dashboard, and user profile.

This form is intended to gather your feedback after watching the demo. Whether you are a medical practitioner, student, or lecturer, your insights are incredibly valuable to help improve the system's usability, functionality, and clinical relevance. Additionally, your feedback will be used solely for academic and development purposes, and all responses will be treated confidentially.

If you have any further inquiries, please contact me at **78115@siswa.unimas.my** or **01110907586** (WhatsApp).

* Indicates required question

Section A: Demographic Information

This section collects basic background information to help us understand the diversity of participants providing feedback. Your responses will help us analyse how different user groups perceive and evaluate the system. All information provided will remain confidential and used only for academic purposes.

1. Age Group *

Mark only one oval.

- ◯ Under 18
- ◯ 18-25
- ◯ 26-29
- ◯ 30-39
- ◯ 40-49
- ◯ 50 and above

2. Profession / Role *

Mark only one oval.

- ◯ Medical Practitioner (Doctor, Pathologist, Nurse etc.)
- ◯ Student
- ◯ Lecturer
- ◯ Researcher
- ◯ Other: _____

3. Area of Expertise

e.g., Pathology, Gynaecology, Medicine, Artificial Intelligence, Software Engineering, etc.

Section B: Model Accuracy Testing

Due to hosting limitations, the full CerviScan website is not currently available online. However, the core image classification model has been hosted separately on Hugging Face Spaces to allow you to test the cervical Pap smear image classification feature directly.

You can try uploading your own sample images or test images on the hosted model via the link provided. This section gathers your feedback on the model's accuracy, usability, and overall performance based on your experience interacting with the live model.

Your input here is valuable for validating the AI classification and guiding future improvements.

Please note that this hosted model demo is provided for academic and testing purposes only.

💡 **Test the model here:** https://huggingface.co/spaces/lyttyl/cerviscan-space
📂 **Sample images set:** https://drive.google.com/drive/folders/1cjDUbZ2sfd3Xqlx4DahSx9K6yjhZ-Qmp?usp=sharing

Instructions: After testing the model using the link above, please rate the following:
(1 star = Very Poor, 5 stars = Excellent)

4. Did you use your own cervical Pap smear images or the provided test set for classification? *

Mark only one oval.

- ◯ I used my own image(s)
- ◯ I used the provided test set
- ◯ I used both
- ◯ Other: _____

5. How many of the model's predictions were correct based on your uploaded images? *

Please indicate in the format: Total correct predictions / Total images uploaded (e.g., 3/5)

_____

6. How accurate the model's prediction was based on the image you uploaded *

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

7. How helpful was the confidence score in understanding the model's certainty? *

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

8. Overall, how satisfied are you with the model's performance? *

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

9. If there were any incorrect predictions, please specify the actual category and the model's predicted result.
e.g., Actual: Koilocytotic, Predicted: Dyskeratotic

Demo Video – CerviScan System Overview

To evaluate the full system (including image upload, results management, patients management, analytics dashboard, and user profile), please watch one of the following recorded demo videos:

Demo Video Link

1. **with narration / explanation (10:21):** https://youtu.be/iHX5tJGhHzE
2. **sped up with bgm only (04:07):** https://youtu.be/PaKP6SNDC0s

Section C: Usability & Interface

This section aims to evaluate the overall look, feel, and ease of use of the CerviScan system as presented in the demo video. Your feedback will help determine whether the interface is user-friendly, intuitive, and visually appropriate for its intended users.

Instructions: Please rate each of the following aspects of the system's usability and interface based on what you observed in the demo video.
(1 star = Very Poor, 5 stars = Excellent)

10. How would you rate the overall visual design and professionalism of the interface? *

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

11. How intuitive and easy was it to understand the system's layout and navigation flow? *

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

12. How clear and well-placed were the buttons, icons, and labels? *

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

13. How readable was the text in terms of font size, colour, and contrast? *

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

14. How easy does the system appear to use for someone with little to no technical background? *

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

15. Overall, how user-friendly is the system interface based on your observation? *

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

16. Do you have any suggestions or comments regarding the system's interface or usability?

Section D: Functionality & Workflow

This section focuses on how well the system's core features and workflows meet user expectations. Your feedback will help evaluate how intuitive and practical the system appears for real-world use based on the demo video.

Instructions: Please rate each aspect based on your observation.
(1 star = Very Poor, 5 stars = Excellent)

17. How clear and straightforward does the image upload and classification process appear? *

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

18. How understandable are the classification results (e.g., cell category and confidence score)? *

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

19. How well does the results management module support reviewing, searching, and managing classification results? *

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

20. How practical and easy-to-use does the patient management module appear to be? *

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

21. How useful and informative are the analytics dashboards for monitoring classification trends and statistics? *

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

22. Overall, how well do the system features work together to support the workflow from image upload to result analysis? *

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

23. Do you have any suggestions to improve the system's functionality or workflow?

Section E: Clinical and Technical Relevance

This section evaluates whether the system appears relevant, reliable, and beneficial for clinical or technical use. Your feedback will help ensure the system aligns with its intended purpose and target users.

Instructions: Please rate the following based on your professional or academic perspective.
(1 star = Strongly Disagree, 5 stars = Strongly Agree)

24. The system appears to have potential for supporting early cervical cancer screening or educational use. *

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

25. The classification categories used (e.g., superficial-intermediate, dyskeratotic, metaplastic, parabasal, koilocytotic) are appropriate and meaningful. *

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

26. The use of confidence scores in the results adds value to decision-making or interpretation. *

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

27. The system demonstrates a reasonable level of technical reliability and logical design. *

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

28. The features and outputs appear clinically or technically useful for the intended users. *

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

29. In your opinion, how can the system be made more clinically or technically relevant?

Section F: General Feedback & Suggestions

This final section invites open feedback about your overall impression of the CerviScan system and any ideas for improvement. Your input is greatly appreciated and will help refine the system further.

30. What features did you find most useful or impressive in the system?

31. Were there any features or aspects that seemed unnecessary or confusing?

32. Do you have any suggestions for features or improvements that should be added in future versions?

33. Any other comments, questions, or concerns?