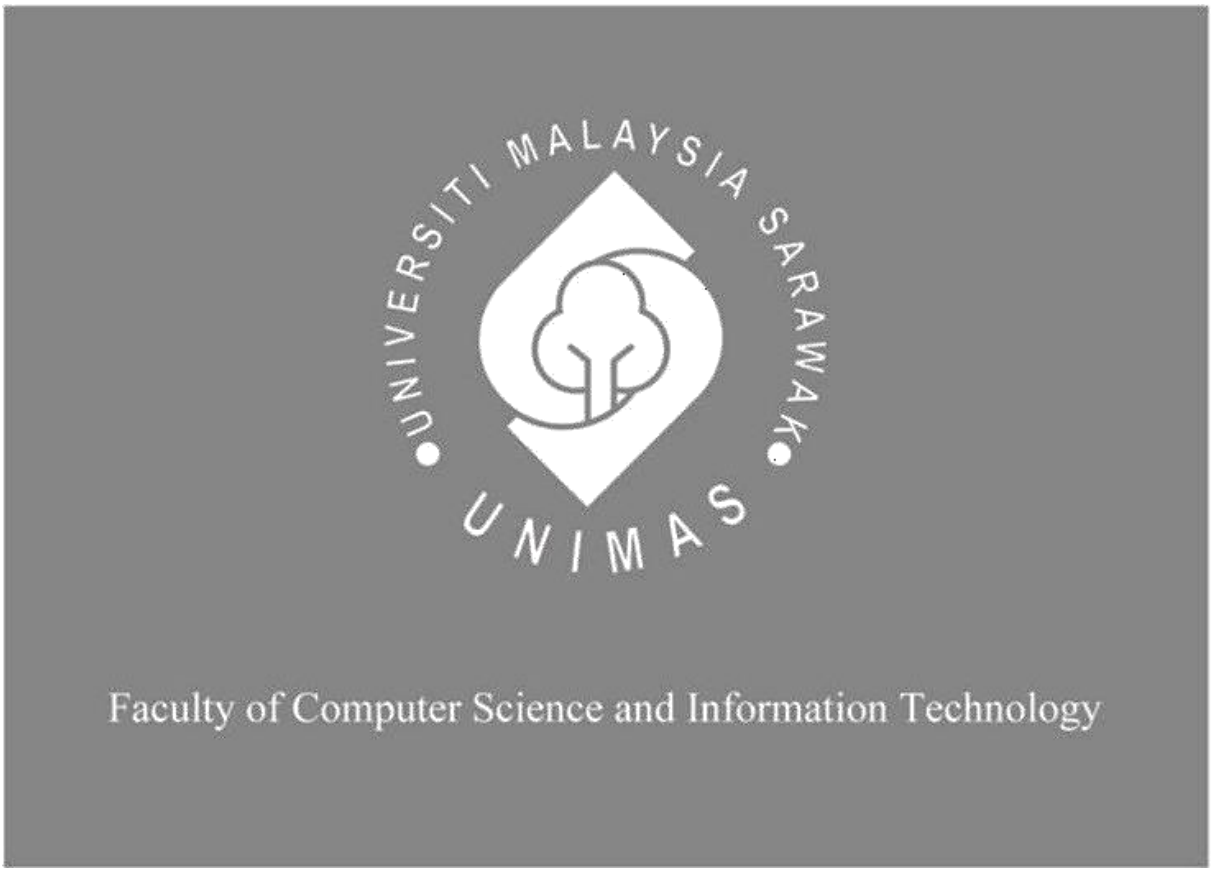


DOG BREED CLASSIFICATION USING DEEP LEARNING APPROACH

Annastasha Chang See May

Bachelor of Computer Science with Honours (Network Computing)

2025

DOG BREED CLASSIFICATION USING DEEP LEARNING APPROACH

ANNASTASHA CHANG SEE MAY

This project is submitted in partial fulfillment of the
requirements for the degree of
Bachelor of Computer Science with Honours (Network Computing)

Faculty of Computer Science and Information Technology
UNIVERSITI MALAYSIA SARAWAK
2025

PENGKLASIFIKASIAN BAKA ANJING MENGGUNAKAN PENDEKATAN PEMBELAJARAN MENDALAM

ANNASTASHA CHANG SEE MAY

Project ini merupakan salah satu keperluan untuk
Ijazah Sarjana Muda Sains Komputer dan Teknologi
Maklumat (Pengkomputeran Rangkaian)

Fakulti Sains Komputer dan Teknologi Maklumat
UNIVERSITI MALAYSIA SARAWAK
2025

UNIVERSITI MALAYSIA SARAWAK

THESIS STATUS ENDORSEMENT FORM

TITLE DOG BREED CLASSIFICATION USING DEEP LEARNING APPROACH

ACADEMIC SESSION: 2024/2025

ANNASTASHA CHANG SEE MAY
(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	

Validated by

(AUTHOR'S SIGNATURE)

Assoc. Prof. Dr. Johari Abdullah
Faculty of Computer Science & Information Technology
Universiti Malaysia Sarawak
94300 Kota Samarahan

(SUPERVISOR'S SIGNATURE)

Permanent Address

No.15-I, Lorong Rubber 7, Rubber Road, 93400, Kuching, Sarawak, Malaysia

Date: 19/7/2025

Date: 19/7/2025

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

DECLARATION

I, Annastasha Chang See May with the matric number 78855, hereby solemnly declare that the report titled '**Dog Breed Classification Using Deep Learning Approach**' is prepared and completed authentically by me in partial fulfilment of the requirements for the degree of Bachelor of Computer Science with Honours (Network Computing) is a record of an original work carried out by me under the guidance of Associate Professor Dr. Johari bin Abdullah. I further declare that the work reported in this project has not been previously or concurrently submitted for any other degree at UNIVERSITI MALAYSIA SARAWAK (UNIMAS) or any other institution. I declare that I have not copied or plagiarised from any other sources except for quotations and citations which have been duly acknowledged.

ANNASTASHA CHANG SEE MAY (78855)
Faculty of Computer Science and Information Technology
UNIVERSITI MALAYSIA SARAWAK

ACKNOWLEDGEMENT

First and foremost, I would like to express my heartfelt gratitude to my supervisor, **Associate Professor Dr. Johari bin Abdullah**, for his invaluable guidance, encouragement, and unwavering support throughout this Final Year Project. His expertise and insights have been instrumental in shaping this project into its current form.

I am deeply grateful to my examiner, **Associate Professor Dr. Lee Nung Kion**, for his invaluable feedback on my Final Year Project. His evaluation and suggestions were crucial in refining my work and improving its quality.

I extend my gratitude to **Professor Dr. Wang Yin Chai** and **Associate Professor Dr. Chiew Kang Leng**, the Final Year Project coordinators, for his remarkable guidance and clear direction. His updates and organisation have kept me on track and met all deadlines.

A special thank you to my family and friends, whose constant motivation and understanding have been a source of strength throughout this journey.

Finally, I would like to express my gratitude to all the respondents who took the time to participate in the questionnaire and share their valuable insights, which were instrumental in the development of this project.

This project would not have been possible without the support of everyone mentioned above, and I am sincerely grateful to each of them.

ABSTRACT

Identifying dog breeds from images is crucial for understanding their specific traits, such as behaviour, health, and care needs. However, distinguishing similar or mixed breeds remains challenging. To tackle this issue, a web-based application utilising deep learning approach is proposed for dog breed classification. The proposed system employs transfer learning, adapting a pre-trained EfficientNetB3 model (initially trained on ImageNet) to the Kaggle Dog Breed Dataset. This approach ensures high accuracy and efficiency in breed identification. The web application utilises computer vision technology to process user-uploaded dog images and predict the top three similar breeds, along with their corresponding similarity percentages. It offers a user-friendly interface, built with Django for the backend and HTML, CSS, and JavaScript for the frontend, ensuring accessibility and responsiveness. TensorFlow's pre-trained models are employed to enhance the system's deep learning capabilities, enabling precise and efficient classification. The system is designed to benefit dog owners, breeders, and veterinarians by providing accurate breed predictions and valuable insights. This project showcases the potential of deep learning in solving real-world classification problems. Future improvements may include features such as mixed-breed analysis, multilingual support, and integration with pet adoption databases, broadening the system's usability. By offering an innovative approach to dog breed classification, this application aims to make a meaningful contribution to AI-driven image recognition and its practical applications.

ABSTRAK

Mengenal pasti baka anjing melalui gambar sangat penting untuk mengetahui ciri-ciri setiap baka, seperti tingkah laku, masalah kesihatan, dan cara penjagaan yang sesuai. Tetapi, mengenal pasti baka yang hampir sama atau anjing kacukan boleh jadi sukar. Oleh itu, satu sistem aplikasi web telah dibina menggunakan kaedah pembelajaran mesin yang dipanggil pembelajaran pindahan (transfer learning). Sistem ini menggunakan model EfficientNetB3 yang sudah dilatih dengan set data ImageNet, kemudian ditambah latihan menggunakan data baka anjing dari Kaggle supaya ia lebih tepat. Pengguna boleh memuat naik gambar anjing ke laman web ini, dan sistem akan meramal tiga baka yang paling hampir, bersama peratus perbezaan. Sistem ini dibina menggunakan TensorFlow untuk model pembelajaran mesin, Django sebagai sistem belakang (backend), dan HTML, CSS, serta JavaScript untuk paparan laman web (frontend). Reka bentuk laman ini adalah mudah dan mesra pengguna supaya semua orang boleh menggunakannya tanpa masalah. Sistem ini berguna untuk pemilik anjing, doktor haiwan, dan penternak untuk mendapatkan maklumat tentang baka anjing mereka. Pada masa hadapan, sistem ini boleh ditambah baik dengan ciri-ciri lain seperti ramalan untuk anjing baka campuran, sokongan bahasa lain, dan pautan ke laman pengambilan haiwan peliharaan. Projek ini menunjukkan bagaimana teknologi AI boleh digunakan untuk menyelesaikan masalah dalam dunia sebenar..

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AI	Artificial Intelligent
API	Application Programming Interface
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
CSS	Cascading Style Sheets
DFD	Data Flow Diagram
DL	Deep Learning
ERD	Entity Relationship Diagram
HTML	Hypertext Markup Language
JS	JavaScript
ML	Machine Learning
SQL	Structured Query Language

CHAPTER 1: INTRODUCTION

1.1 Project Title

Dog Breed Classification Using Deep Learning Approach

1.2 Introduction/Background

Dog ownership is increasing worldwide, with more people seeking accurate breed identification to provide tailored care and management (Varghese & Remya, 2021). Knowing a dog's breed can help predict behaviour, address health needs, and identify inherited traits. However, traditional breed identification methods rely heavily on visual assessments by professionals, which can be subjective and unreliable, especially for mixed breeds with complex genetic backgrounds (Varghese & Remya, 2021).

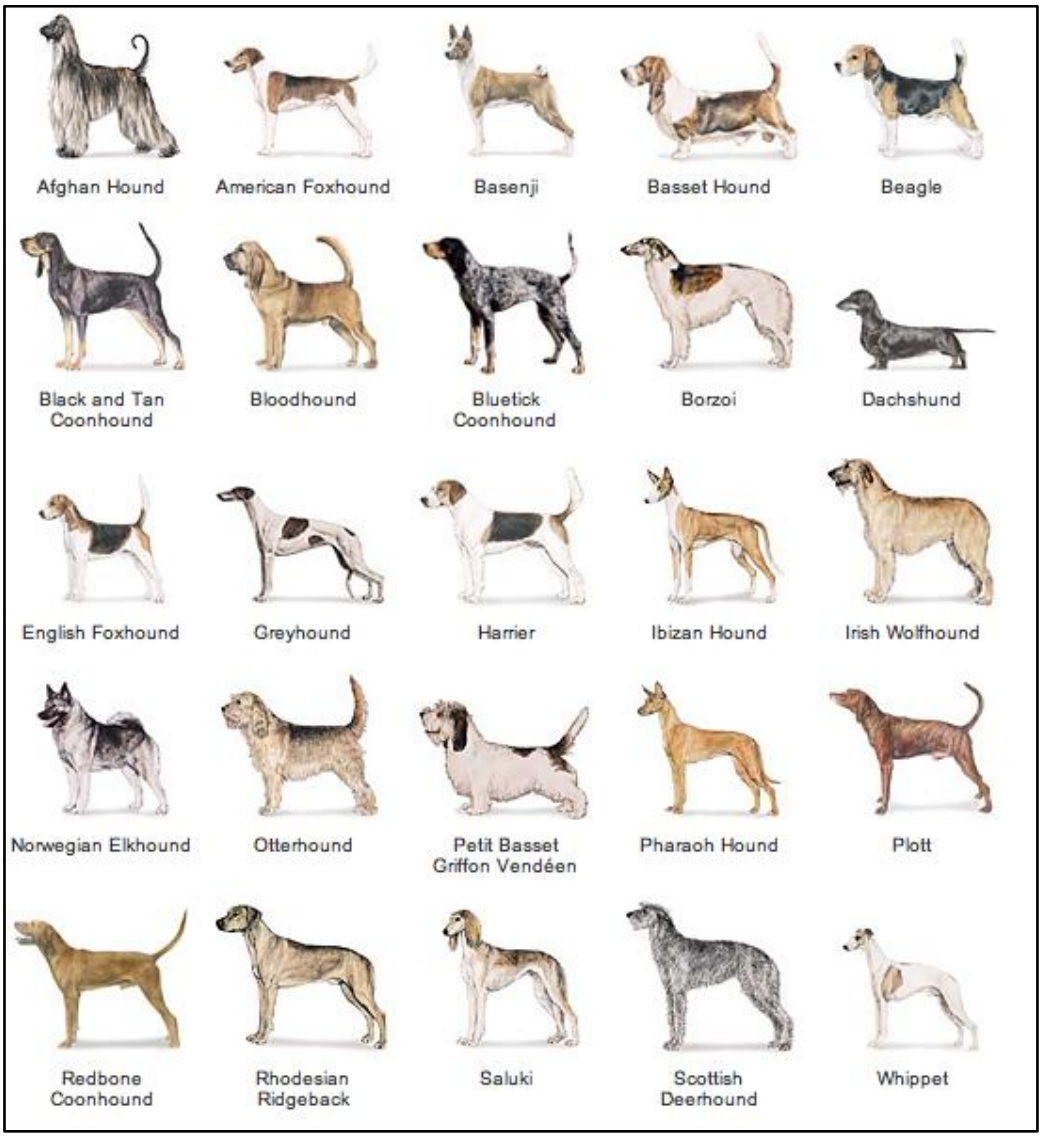


Figure 1. 1: Dog Breeds

Source: https://medium.com/@wadhai.piyush/the-data-science-algorithmic-classification-of-dog-breed-be38eb3ceab0 (Wadhai, 2021)

The limitations of traditional methods underscore the need for a more precise and accessible solution for identifying dog breeds. Deep learning (DL), a subset of machine learning, has proven highly effective in image recognition tasks across various fields. Convolutional neural networks (CNNs) excel at analysing visual data with minimal pre-processing (Acharya et al., 2022). For this project, transfer learning—an approach that leverages pre-trained models—has been selected to improve both accuracy and efficiency in dog breed classification.

Transfer learning utilises models trained on extensive datasets, allowing adaptation to specific tasks while minimising training time and resource needs.

This DL-based system will power a web application that allows users to upload an image of a dog to identify its breed. This web application can offer several advantages, including:

1. **Classify Dog Breed:** Users can upload a photo of their dog to identify its breed**.**
2. **Enhancement of Animal Shelter Management**: Shelters can identify breeds to manage care, find homes, and connect with adopters.
3. **Supporting Veterinarians**: The system provides breed information that can assist veterinarians in diagnosis and treatment, particularly in areas where a dog's history is unknown.

1.3 Problem Statement

Despite technological advancements, accurately identifying dog breeds remains a challenge. Traditional methods, which rely on manual identification by specialists, are often time-consuming and challenging for the average dog owner or animal shelter worker (Varghese & Remya, 2021). This highlights the need for an accessible, efficient tool that can accurately identify dog breeds from images. By implementing transfer learning, this project aims to leverage pre-trained deep learning models, which have demonstrated strong performance in image classification, to address this need. The application of transfer learning can enhance accuracy and efficiency by building on existing, robust models rather than training one from scratch. The issue can be categorised into the following challenges:

1. **Limitations of Traditional Methods**: Nowadays, many practices rely on visual assessments, but these can sometimes be subjective and may lead to errors, especially when evaluating dogs that are similar or mixed-breeds (Haswar et al., 2022). Methods

like microchipping and tattooing, while more reliable, tend to be costly and are not widely adopted. (Varghese & Remya, 2021).

2. **The Importance of Precision and Accessibility**: Accurate identification of dog breeds is crucial for understanding specific behaviours, health risks, and training requirements. (Borwarnginn et al., 2020). A solution must be both precise and accessible to a range of users, including dog owners, shelter workers, and veterinarians.
3. **Challenges of Fine-grained Classification**: Identifying dog breeds can be tricky due to the small differences in their traits, especially among similar-looking breeds. This detailed classification proves challenging, as certain breeds appear nearly identical or are classified as sub-breeds, making visual identification unreliable. (Manivannan & Venkateswaran, 2023). Transfer learning can help with these issues by using pre-trained models designed to recognise subtle differences. This boosts the model's strength and usefulness in real-world applications.

1.4 Scope

This project aims to develop a web application that enables users to upload an image of a dog. A deep learning model trained using transfer learning will be applied in this proposed web application. Transfer learning allows the system to use models that have already been trained. This saves time and makes the system more accurate, as there is no need to start training from the beginning.

The target audience are pet owners, dog breeders, veterinarians, and animal shelters. The web application helps identify different breeds. The system will use TensorFlow for the deep learning model. Python will handle backend tasks such as image processing and interacting with the model. HTML, CSS, and JavaScript will create a responsive frontend, allowing users to upload images and view the results.

The application will focus on common dog breeds from existing datasets. It will not recognize rare or mixed breeds, nor will it analyse things that are not dogs. The system will consider different image qualities, angles, and lighting to make accurate predictions. It will also show percentage similarities between breeds, helping users understand potential matches when confidence is low.

1.5 Aims and Objectives

The primary objective of this project is to develop a web application that can identify dog breeds uploaded by users. Using transfer learning, the model can take advantage of knowledge from large, pre-trained models. This improves its performance and reduces the need for a lot of training on new data. The system is designed to provide a fast, user-friendly, and accurate tool for recognizing dog breeds, focusing on these key objectives:

1. **To implement a transfer learning model for classifying dog breeds:** A pre-trained Convolutional Neural Network (CNN) model will be adapted to classify specific dog breeds using transfer learning accurately.
2. **To develop a user-friendly web interface for dog breed identification:** Users will be able to upload pictures of dogs through the web interface, which will analyze and identify the breed. It will show the top three breeds that are most similar based on the model's predictions. The design of the interface will focus on being easy to use and accessible, making it suitable for pet owners, veterinarians, and shelter workers.
3. **To integrate and test the full system for performance, reliability, and scalability:** The last goal is to test the whole system fully, including how the model works with both the backend and the frontend, to make sure it works at its best. This step will make sure that the web-based system meets the project's goals for high accuracy, responsiveness,

and user engagement, as well as being able to work with different types of devices and user settings.

1.6 Brief Methodology

The Agile-SCRUM technique has been selected to direct the development of a web-based canine breed classification system. Agile is optimal for this project, as it facilitates flexibility, iterative advancement, and ongoing assessment—crucial for enhancing the deep learning model, user interface, and overall functionality. The project will employ transfer learning by utilising a pre-trained convolutional neural network (CNN) model in TensorFlow to improve picture classification, hence optimising time and processing resources. Python will serve as the principal programming language for backend development, but HTML, CSS, and JavaScript (or possibly Django for a more dynamic frontend) will be employed to provide an accessible and responsive interface.

The iterative and incremental Agile methodology facilitates ongoing enhancement of the deep learning model, backend API, and user interface, permitting real-time modifications to accommodate changing user requirements. The incorporation of transfer learning within Agile's sprint-based framework offers a methodical strategy to evaluate and enhance every component of the system, ranging from the picture classification model to the user experience of the web application. Through continuous user feedback and modifications, Agile guarantees that the final product is both functionally sound and user-centric.

1.7 Significance of Project

This study on classifying dog breeds shows that computer science can use cutting edge technologies like deep learning and transfer learning to solve problems in the real world. Through the use of transfer learning, the project improves model performance and shortens

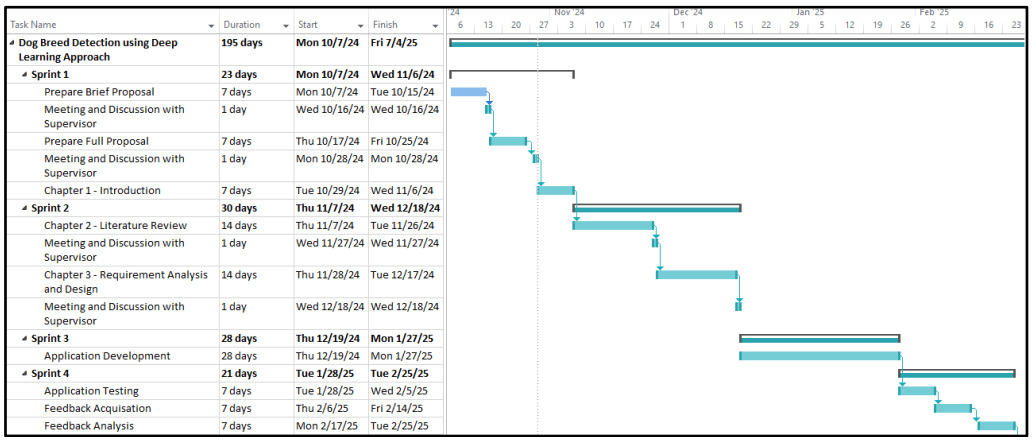
training time, showing an effective and scalable way to describe images. This is important for computer science because it shows how pre-trained models can be used for specific jobs. It also shows how useful machine learning frameworks like TensorFlow are.

The project's web-based design, potentially utilising HTML, CSS, JavaScript, or Django, delivers an extensive learning experience in both front-end and back-end programming, while also ensuring an accessible and intuitive user interface. The program assists pet owners, veterinarians, breeders, and shelters by facilitating the identification and understanding of dog breeds, hence enhancing breed-specific treatment. The combination of computer vision, machine learning, and user-focused web design underscores the capacity of computer science to develop significant, user-friendly, and socially advantageous applications.

By linking complicated AI theory to real-world applications, this work shows how important computer science is for making answers that are both technically advanced and open to everyone.

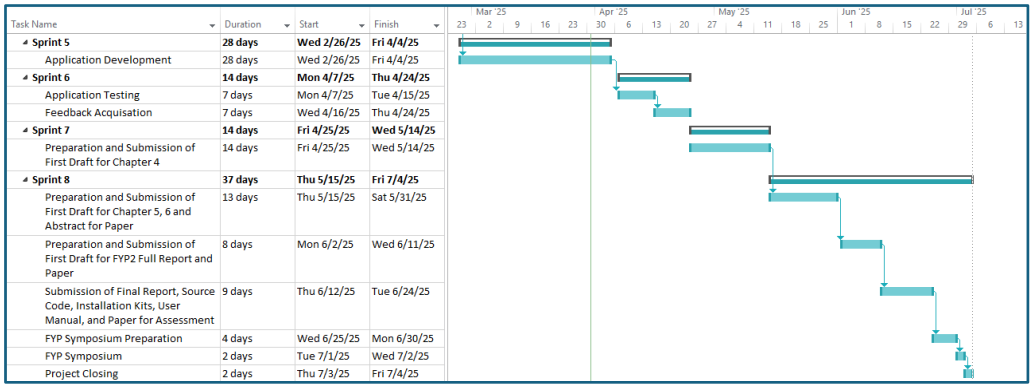
1.8 Project Schedule

The project is expected to be completed within two semesters following the completion of Final Year Project I and Final Year Project II, as demonstrated below.



Task Name	Duration	Start	Finish
Dog Breed Detection using Deep Learning Approach	**195 days**	**Mon 10/7/24**	**Fri 7/4/25**
Sprint 1	**23 days**	**Mon 10/7/24**	**Wed 11/6/24**
Prepare Brief Proposal	7 days	Mon 10/7/24	Tue 10/15/24
Meeting and Discussion with Supervisor	1 day	Wed 10/16/24	Wed 10/16/24
Prepare Full Proposal	7 days	Thu 10/17/24	Fri 10/25/24
Meeting and Discussion with Supervisor	1 day	Mon 10/28/24	Mon 10/28/24
Chapter 1 - Introduction	7 days	Tue 10/29/24	Wed 11/6/24
Sprint 2	**30 days**	**Thu 11/7/24**	**Wed 12/18/24**
Chapter 2 - Literature Review	14 days	Thu 11/7/24	Tue 11/26/24
Meeting and Discussion with Supervisor	1 day	Wed 11/27/24	Wed 11/27/24
Chapter 3 - Requirement Analysis and Design	14 days	Thu 11/28/24	Tue 12/17/24
Meeting and Discussion with Supervisor	1 day	Wed 12/18/24	Wed 12/18/24
Sprint 3	**28 days**	**Thu 12/19/24**	**Mon 1/27/25**
Application Development	28 days	Thu 12/19/24	Mon 1/27/25
Sprint 4	**21 days**	**Tue 1/28/25**	**Tue 2/25/25**
Application Testing	7 days	Tue 1/28/25	Wed 2/5/25
Feedback Acquisation	7 days	Thu 2/6/25	Fri 2/14/25
Feedback Analysis	7 days	Mon 2/17/25	Tue 2/25/25

Figure 1.2: Gantt Chart of Project Timeline 1



Task Name	Duration	Start	Finish
Sprint 5	**28 days**	**Wed 2/26/25**	**Fri 4/4/25**
Application Development	28 days	Wed 2/26/25	Fri 4/4/25
Sprint 6	**14 days**	**Mon 4/7/25**	**Thu 4/24/25**
Application Testing	7 days	Mon 4/7/25	Tue 4/15/25
Feedback Acquisation	7 days	Wed 4/16/25	Thu 4/24/25
Sprint 7	**14 days**	**Fri 4/25/25**	**Wed 5/14/25**
Preparation and Submission of First Draft for Chapter 4	14 days	Fri 4/25/25	Wed 5/14/25
Sprint 8	**37 days**	**Thu 5/15/25**	**Fri 7/4/25**
Preparation and Submission of First Draft for Chapter 5, 6 and Abstract for Paper	13 days	Thu 5/15/25	Sat 5/31/25
Preparation and Submission of First Draft for FYP2 Full Report and Paper	8 days	Mon 6/2/25	Wed 6/11/25
Submission of Final Report, Source Code, Installation Kits, User Manual, and Paper for Assessment	9 days	Thu 6/12/25	Tue 6/24/25
FYP Symposium Preparation	4 days	Wed 6/25/25	Mon 6/30/25
FYP Symposium	2 days	Tue 7/1/25	Wed 7/2/25
Project Closing	2 days	Thu 7/3/25	Fri 7/4/25

Figure 1.3: Gantt Chart of Project Timeline 2

1.9 Expected Outcome

When this project is finished, a fully functional web application that correctly recognises dog breeds from user-uploaded photos should be produced. The system will be optimised for high accuracy and efficiency in dog breed recognition, driven by a deep learning model enhanced by transfer learning and created using TensorFlow. Users will gain from a simplified, user-friendly interface that shows breed predictions, confidence scores, and pertinent details about each breed and is made for simple image uploads and instant feedback.

The web application will benefit pet owners, veterinarians, and animal welfare professionals by offering comprehensive insights on identified breeds, including possible traits, behaviours, and care information. By showing how transfer learning can increase model accuracy while lowering the requirement for large amounts of training data, this project seeks to advance the field of computer vision and pave the way for comparable applications in other fields.

1.10 Project Outline

This work consists of six chapters covering the creation and implementation of a web application intended to precisely identify dog species..

Chapter 1: Introduction provides a thorough overview of the project together including its background, issue description, objectives, methods, scope, significance, timeline, expected results, and outline. Defining the present social issues and offering the required background, the problem statement. While methodology clarifies the chosen strategy, objectives define exact goals. Scopes define project fields, whereas significance highlights strengths and impact. The schedule provides a comprehensive timeline, while the expected outcome emphasises the web application's achievements.

Chapter 2: Literature Review examines commercial applications similar to the proposed web application. It evaluates functions, conducts a comparative study, and investigates the aspects of the proposed system. The chapter clarifies how the recommended application solves the limitations of present applications and defines their shortcomings. It covers development methods, technical instruments, and knowledge gained from a thorough reading of published materials, including conference papers, journals, and books.

Chapter 3: Requirement Analysis and Design specifies stakeholder participation in the requirements-gathering process, guiding the development of the intended application. The chapter systematically defines both system and user requirements and emphasises the design of the application, thus justifying its following requirements. Architectural design, scenario illustrations, context diagrams, TensorFlow stages, entity relationship diagrams (ERD), data flow diagrams, and flow charts help to clearly show the structure of the application. Additionally, it shows user interface prototypes, giving a first impression of the design and functionality of the web application.

Chapter 4: Implementation gives a thorough overview of the proposed system's implementation, including the rationale for the chosen methodology. It delves into the technical

intricacies of the system, including the processes and tools employed during development. The chapter also focuses on the major characteristics and functionalities of the proposed system, ensuring that it meets the required requirements.

Chapter 5: Testing discusses the testing techniques used to evaluate the system's performance. It defines the test circumstances that will be utilised to accurately assess the system's functionality. Thorough testing was performed, and appropriate improvements were applied based on the results to improve the system's effectiveness. The chapter also investigates the impact of testing results on system changes, ensuring that the final product fulfils the required requirements.

Chapter 6: Conclusions and Future Work outlines the planned online application and presents conclusive conclusions. It stresses the successful attainment of project objectives and describes the constraints encountered. The chapter discusses the project's lessons learnt, using realistic instances to illustrate key ideas. Furthermore, it investigates potential future work fields, suggesting enhancements and developing system functionalities. Discussions about novel concepts and areas for improvement provide valuable guidance for researchers and developers working to advance the system. The chapter acts as a roadmap for future initiatives, ensuring continuous progress and adaptability to meet new needs and problems.

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

This chapter presents an analysis of comparable web and mobile applications, examining their characteristics, functionalities, and primary procedures. This section presents a literature review of three analogous systems: DogBreedAI, Dog Visual Classification (DogVC), and Identify Dog Breeds Pro. A comparative table will be developed to demonstrate the distinctions between the existing and proposed web applications based on the analysis of these systems. This table presents the advantages and disadvantages of each system. This section will review three Convolutional Neural Networks (CNNs): EfficientNet, InceptionV3, and ResNet50. This chapter analyses the selection of software and hardware components necessary for developing the proposed web application. This evaluation seeks to determine the optimal software and hardware resources for developing the proposed web application. The identified special features from the analysed systems will be incorporated into the proposed system, thereby enabling a comprehensive discussion of its distinctive attributes. This chapter presents a summary of the comparative analysis and consolidates the collected insights and recommendations.

2.2 Review on Similar Existing Systems

In this section, a total of three applications—two web applications and one mobile application with features similar to the suggested web application—are reviewed and analysed. DogBreedAI, Dog Visual Classification (DogVC), and Identify Dog Breeds Pro are the apps that have been chosen for examination. Because of their distinctive and creative qualities that set them apart in their respective fields, these applications have been chosen for in-depth investigation and evaluation.

2.2.1 DogBreedAI

DogBreedAI evaluates uploaded dog photos using sophisticated AI algorithms. It recognises unique characteristics of the breed, such as body size, fur pattern, muzzle length, and ear shape. To correctly identify the breed, these characteristics are then compared to a large database of recognised dog breeds. From the uploaded photo, the breed analysis extracts important physical characteristics like body proportions, coat colour, and facial structure. The system accurately determines the dog's breed by looking at these characteristics. The dog breed detection service is a useful tool for dog owners and enthusiasts, and it is completely free. The service is maintained with little advertising to guarantee free and easy access..

In particular, the breed analysis is optimised for images that only show one dog. When multiple dogs appear in an image, the AI may encounter difficulties in accurately identifying and analyzing each dog individually, which could affect the precision of the analysis. The precision of breed analysis is significantly affected by the quality of the image. It is recommended to use a well-lit, clear image with the dog facing the camera directly and free from obstructions such as collars or toys. High-quality images enable a more precise analysis.

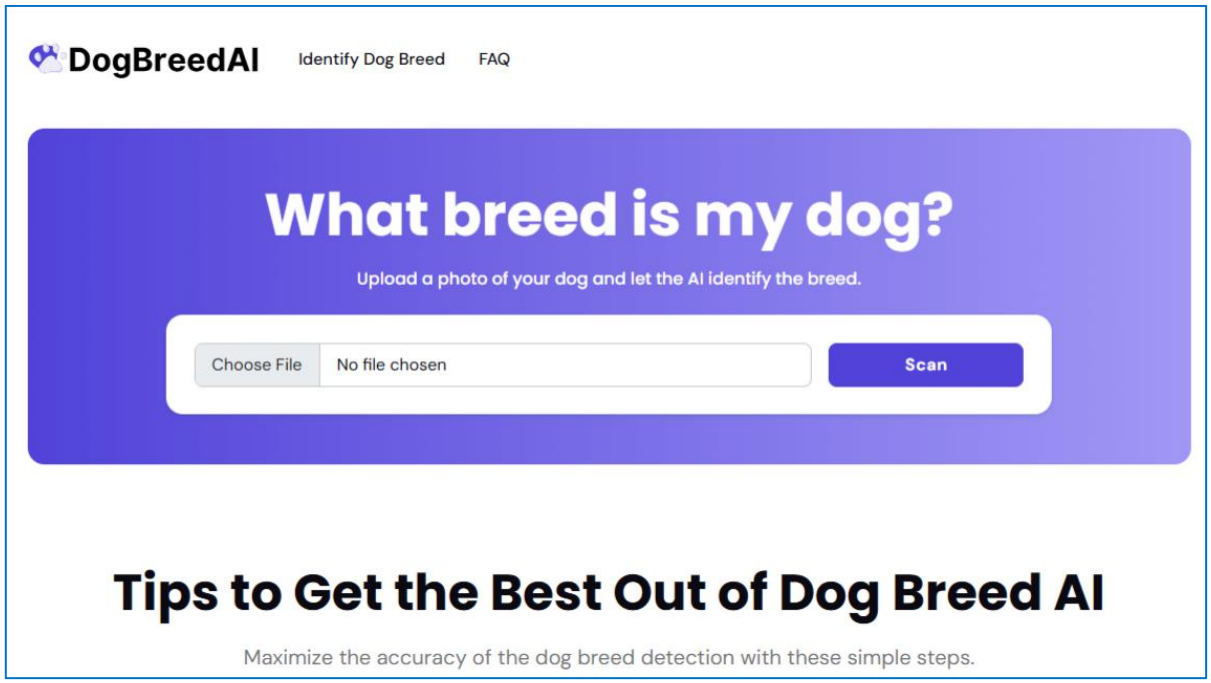


Figure 2. 1: DogBreedAI Main Page

Source: https://www.whatbreedismydog.com/

Figure 2.1 shows the DogBreedAI main page. User can upload an image of a dog breed by selecting "Choose File" or drag and drop into the "No file chosen" text bar. After uploading the image, user can click on "Scan" and the result for the possible dog breed will be displayed for the user.

2.2.2 Dog Visual Classification (DogVC)

Dog Visual Classification (DogVC), hosted by the National Genomics Data Center, leverages a robust dataset tied to the iDog phenotype database. Its strengths lie in providing detailed genetic and health information about the identified breeds. However, it relies heavily on high-resolution photos, making it less effective in real-world scenarios.

Dog Visual Classification (DogVC) is an online tool created to identify a dog's breed from an uploaded image. It utilises deep learning methodologies to forecast the probability of different breeds. Furthermore, it provides a detailed link to the iDog phenotype dataset, which

includes comprehensive information about breeds, registration details, genetic variations, and associated diseases.

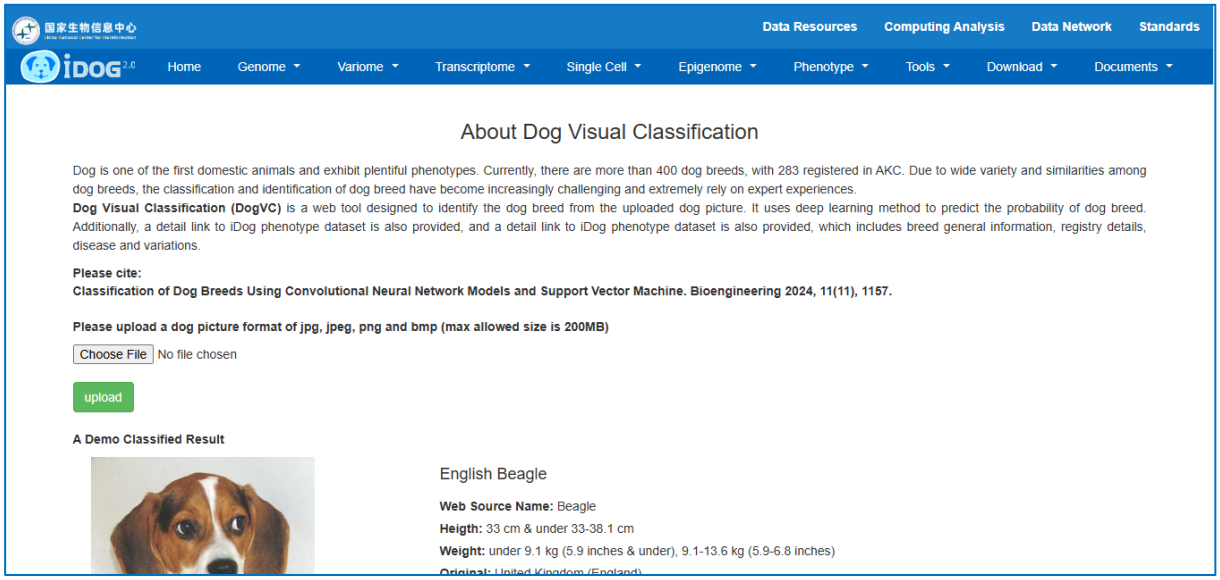


Figure 2.2: Dog Visual Classification (DogVC) Main Page

Source: https://ngdc.cncb.ac.cn/dogvc/

Figure 2.2 shows the main page of the Dog Visual Classification (DogVC) web application. Users can upload a dog breed image by either selecting “Choose File” or drag and drop it into the “No file chosen” area. Once the image is uploaded, they can click “upload” to display the results showing the possible dog breed.

2.2.3 Identify Dog Breeds Pro

Identify Dog Breeds Pro uses a convolutional neural network to classify breeds, designed for offline use and optimised for mobile platforms. Its strengths include minimal latency and portability, but it has limited support for less common breeds and mixed-breed identifications. This app allows users to identify more than 200 dog breeds directly from their smartphone. Users can either take a photo using the camera or select an image from the gallery for analysis. The app provides a detailed list of detected breeds, which can be accessed by tapping on the results.

It also supports continuous predictions by focusing on the dog within a specified circle for real-time analysis via video stream. The app offers up to three breed predictions it considers the most accurate, though the results should be regarded as suggestions. Figure 2.3 shows the user interface of the Identify Dog Breeds Pro mobile application.

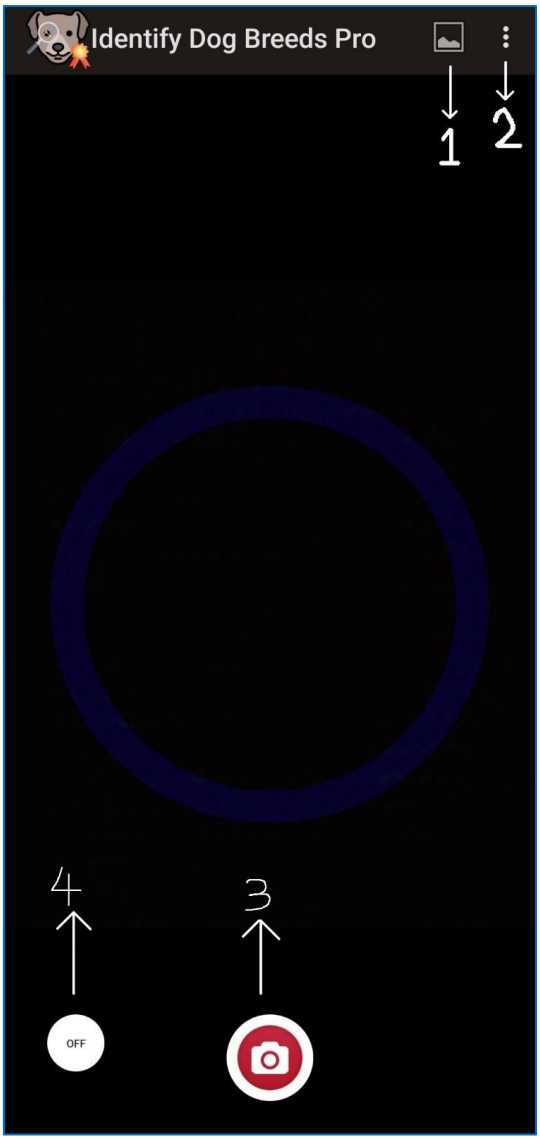


Figure 2.3: Identify Dog Breeds Pro User Interface

Table 2.1 is the description of the features of the mobile application based on the numbers written in Figure 2.3.

Table 2.1: Features of Identify Dog Breeds Pro

Features		Explanation
1	Gallery	User can upload an image of the dog from their gallery.
2	Other settings	Inside the other settings, there are selection such as “Supported Breeds”, “Change language”, “High Accuracy”, “About” and “Exit”.
3	Camera button	User can click the camera button to take a picture of the dog.
4	Live detection	User can choose to “On” or “Off” the live detection.

According to Table 2.1, inside the other settings, “Supported Breeds” shows the image of the dogs that can be detected by the system. Next, if user wants to change the language of the mobile application, they can select the “Change language” option. The “High Accuracy” is to let the application to detect the dog breed with a higher accuracy. For “About”, it is a short explanation about the functionality of this application and the “Exit” is to quit the application.

2.3 Features Comparison of the Reviewed System

A comparative table, which is Table 2.2, summarises the features of these systems and the proposed system.

Table 2.2: Comparison of Features Between 3 Existing System

Features	**Existing Systems**			**Proposed System**
	DogBreedAI	**Dog Visual Classification (DogVC)**	**Identify Dog Breeds Pro**	
Web application	✓	✓	✗	✓
Mobile application	✗	✗	✓	✗
Uses AI/Deep Learning	✓	✓	✓	✓

Supports Real-Time Analysis	✗	✗	✓	✗
Subscription fees	✗	✗	✗	✗
Requires High-Resolution Images	✓	✓	✓	✓
Take picture from camera	✗	✗	✓	✗
Upload from gallery	✓	✓	✓	✓
Analyse single dog	✓	✓	✓	✓
Dog breed detail	✗	✓	✗	✓

Table 2.2 shows the comparison between the three existing systems—DogBreedAI, Dog Visual Classification (DogVC), and Identify Dog Breeds Pro, and the proposed system highlights several key features. The proposed system, DogBreedAI and Dog Visual Classification are web applications, whereas Identify Dog Breeds Pro is a mobile application. All the systems, including the proposed system, utilise AI or deep learning techniques for breed identification. However, only Identify Dog Breeds Pro supports real-time analysis, while the others, including the proposed system, do not. None of the systems impose subscription fees, ensuring accessibility to users. All of the systems rely on high-resolution images for accurate analysis.

Identify Dog Breeds Pro uniquely allows users to capture photos directly using the camera, while the other systems, including the proposed system, require users to upload images from the gallery. All systems support single-dog analysis. Dog Visual Classification (DogVC) provide detailed breed information. The proposed web application aims to address the limitations of existing systems by combining the accessibility of web-based platforms with the

capability to process photos regardless of resolution constraints, thereby ensuring a user-friendly experience.

2.4 Review on Convolutional Neural Network (CNN) Pre-Trained Models

This section examines three selected CNN (Convolutional Neural Networks) pre-trained models commonly used for image classification. CNNs enable computer vision tasks such as object detection and image classification (Klingler, 2024). Their ability to learn from raw images has led to breakthroughs in autonomous vehicles, medical diagnosis, and facial recognition (Klingler, 2024). However, as the size and complexity of datasets increase, CNNs must become deeper and more complex to maintain high accuracy.

The selected models are: EfficientNet, InceptionV3, and ResNet50. These pre-trained models have been selected for comprehensive research and assessment because of their performance, efficiency, and widespread use in image classification tasks, making them ideal candidates for review and analysis in this study.

2.4.1 EfficientNet

EfficientNet comprises a series of convolutional neural network architectures designed to deliver exceptional performance while using significantly fewer parameters and computational resources than conventional convolutional neural networks (CNNs) (Product, 2024). Created by Mingxing Tan and Quoc V. Le at Google Research in 2019, EfficientNet marks a significant advancement in deep learning model architecture, especially for applications such as image classification and object detection (Product, 2024).

EfficientNet introduces a novel compound scaling technique that proportionally adjusts the network's depth, width, and resolution to optimise performance (Product, 2024). This

strategy strikes a balance between model size and accuracy, enabling EfficientNet to achieve greater accuracy while using fewer parameters (Product, 2024).

According to Product (2024), the key components and characteristics of EfficientNet are compound scaling, efficient building blocks, and Neural Architecture Search (NAS).

1. **Compound Scaling**

   EfficientNet utilises compound scaling to harmonize model size and accuracy by concurrently adjusting the network's depth (layer count), width (channel count), and resolution (input image dimensions). This approach guarantees that the model is fine-tuned for both accuracy and efficiency across various tasks and datasets.

2. **Efficient Building Blocks**

   EfficientNet utilizes optimized building components, such as Mobile Inverted Bottleneck Convolution (MBConv), to reduce computational demands while maintaining its ability to represent complex information. These components enable EfficientNet to deliver exceptional performance with a reduced number of parameters compared to conventional CNN frameworks.

3. **Natural Architecture Search (NAS)**

   The EfficientNet architecture was identified using neural architecture search, a method that automatically finds the best neural network designs for a specific task. Utilizing NAS, EfficientNet investigates a large range of potential architectures to determine the most efficient and effective model setup.

EfficientNet is made up of 8 different models, ranging from EfficientNet-B0 to EfficientNet-B7 (Klingler, 2024). The EfficientNet-B0 architecture is built upon the inverted bottleneck residual blocks found in MobileNetV2, along with the incorporation of squeeze-and-

excitation blocks. It's the smallest and most efficient model within the EfficientNet variants.

2.4.2 InceptionV3

For image recognition and classification, InceptionV3 is a sophisticated deep learning model that enables accurate and effective processing of intricate visual data (Activeloop, 2023). It is well known for its ability to accurately analyse complex visual data and produce results as a member of the Inception model family. Activeloop (2023) claims that InceptionV3 has been used in a number of domains, including the detection of skin cancer, the quality-based classification of defective parts, and the identification of diseases in agricultural environments.

Activeloop (2023) claims that new research has demonstrated InceptionV3's effectiveness across a range of applications. One study, for instance, used InceptionV3 in conjunction with other deep learning models to correctly categorise various skin lesion types in order to detect skin cancer. Another use was for InceptionV3, which achieved high accuracy in detecting problems like short forming and weaving faults in plastic components made by injection moulding.

Transfer learning, a method that uses previously trained models to address new tasks with limited data, has shown promise with InceptionV3 (activeloop, 2023). For example, InceptionV3 was used to create a face mask detection system that uses transfer learning to identify people in public places without masks with high accuracy.

2.4.3 ResNet50

A well-liked convolutional neural network (CNN) architecture, ResNet-50 is renowned for its depth and image classification capabilities (Potrimba, 2024). It is a member of the ResNet (Residual Networks) family, which also includes models like ResNet-18 and ResNet-32 with different depths (Potrimba, 2024). ResNet-50, created by Microsoft Research Asia, was

intended to address the difficulties associated with training extremely deep neural networks. ResNet-50, a medium-sized model in the ResNet family, has demonstrated exceptional performance across a range of image classification tasks (Potrimba, 2024).

ResNet-50 is an effective image classification model that can be trained on sizable datasets to achieve exceptional performance, according to Kundu (2023). The use of residual connections, which enable the network to learn residual functions that correlate inputs with outputs, is a key feature of this architecture. Deeper networks can be trained thanks to these connections without running into the vanishing gradient problem.

Convolutional layers, identity blocks, convolutional blocks, and fully connected layers are the four main components of ResNet-50 (Kundu, 2023). The input image's features are first extracted by the convolutional layers. These extracted features are then processed and transformed by identity and convolutional blocks. Lastly, the processed data is categorised by the fully connected layers (Kundu, 2023).

A sequence of convolutional operations make up ResNet-50's first layers. Batch normalisation and Rectified Linear Unit (ReLU) activation are applied following each convolution (Kundu, 2023). The purpose of these layers is to extract characteristics from the input image, such as edges, textures, and shapes. The feature maps are then downsampled by max pooling layers, which reduces their spatial size while maintaining the most crucial information (Kundu, 2023).

The identity and convolutional blocks are the fundamental components of ResNet-50. Bypassing certain layers and adding the original input straight to the output is made possible by the identity block (Kundu, 2023). The network learns residual functions more efficiently thanks to this "shortcut connection" (Kundu, 2023). In order to improve computational

efficiency and reduce the number of filters, the convolutional block is similar but has an additional 1x1 convolutional layer before the main 3x3 convolutional layer (Kundu, 2023).

The final stage of ResNet-50 consists of fully connected layers. These layers produce the final classification by transforming the processed features (Kundu, 2023). The output of the last fully connected layer is then passed through a softmax function, which calculates the probability of the input image belonging to each class (Kundu, 2023).

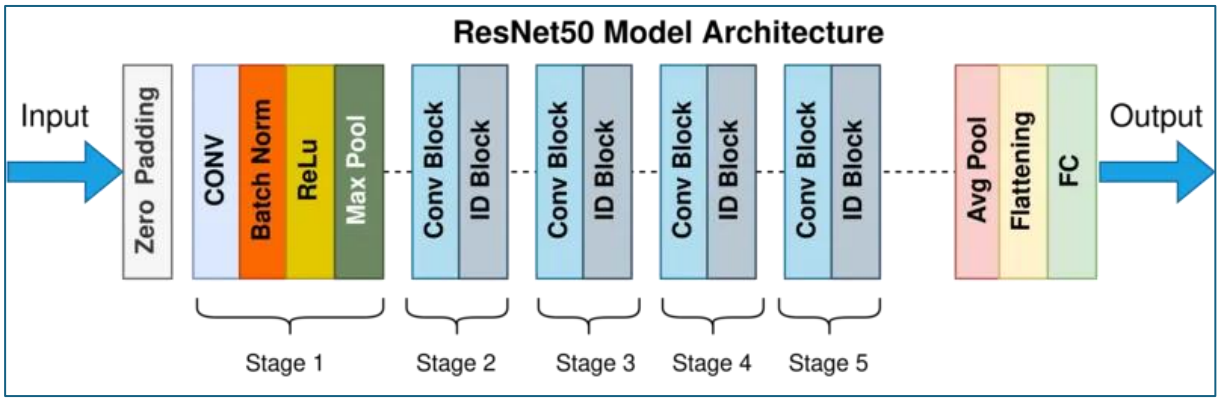


Figure 2.4: ResNet50 Model Architecture

Source: https://medium.com/@nitishkundu1993/exploring-resnet50-an-in-depth-look-at-the-model-architecture-and-code-implementation-d8d8fa67e46f

Figure 2.4 illustrates the architecture of the ResNet-50 model. Trained on the ImageNet dataset, which contains over 14 million images spanning 1,000 categories, ResNet-50 has demonstrated cutting-edge performance in various benchmarks (Kundu, 2023). It achieved an error rate of 22.85% on ImageNet, which is remarkably close to the reported human error rate of 5.1%.

2.5 Review on Tools & Technology

This part aims to deliver a thorough analysis and discussion on the technological requirements and tools essential for the development of the proposed system. It explores the definition, background, and characteristics of both software and hardware requirements. This

section can be defined as the "justification" segment, as it explains the rationale for the system's customised software and hardware requirements among various alternative options.

2.5.1 Software

The software requirements that are essential to the creation of the suggested system will be covered in detail in this section. The front-end and back-end are the main elements required for the development of the web application.

2.5.1.1 Deep Learning (DL)

Artificial Neural Networks (ANNs) are the foundation of Deep Learning (DL), which was first presented by Hinton et al. in 2006. Sarker (2021) points out that DL's remarkable capacity to learn from and derive significant patterns from given data has made it a prominent area in machine learning, artificial intelligence, and data science.

Deep Learning (DL) is a subfield of both Machine Learning (ML) and Artificial Intelligence (AI), according to Sarker (2021). It functions as an AI method that simulates the information processing capabilities of the human brain.

Holdsworth and Scapicchio (2024) claim that the complexities of neural network architectures are the primary distinction between deep learning and conventional machine learning. Simpler neural networks with usually one or two layers are used in traditional machine learning models, which are sometimes referred to as "non-deep" models. Deep learning models, on the other hand, have a more complex structure with three or more layers; to achieve the best training results, these models frequently have hundreds or even thousands of layers (Holdsworth & Scapicchio, 2024).

Deep learning models can make use of unsupervised learning strategies, in contrast to supervised learning models that need structured and labelled data (Holdsworth & Scapicchio,

2024). This makes it possible for them to analyse unstructured, raw data and find important traits, features, and connections in order to produce precise results. Additionally, to increase accuracy, deep learning models can assess and improve their predictions (Holdsworth & Scapicchio, 2024).

A key component of data science, deep learning powers a variety of autonomous, task-automating apps and services. Digital assistants, voice-activated TV remote controls, credit card fraud detection systems, self-driving cars, and generative AI tools are just a few of the commonplace goods and services that are powered by this technology.

2.5.1.2 Transfer Learning

A machine learning technique called transfer learning uses insights from one task or dataset to increase the effectiveness of a model on a related task or dataset (Olivas et al., 2009). It makes use of knowledge gained in one context to improve generalisation in another (Goodfellow et al., 2016). This method has many applications, including training deep learning models and solving regression problems in data science (Ph.D & Kavlakoglu, 2024). Deep learning, which usually requires large datasets to build effective neural networks, benefits greatly from it.

Depending on the available labelled data, a unique model is created for every task in traditional learning (Ph.D & Kavlakoglu, 2024). This is because conventional machine learning algorithms are predicated on the idea that test and training data come from the same feature space. Therefore, if the data distribution alters or the trained model is applied to a new dataset, the model must be retrained entirely, regardless of the similarity of the new task to the original (e.g., developing a sentiment analysis classifier for movie reviews compared to one for song reviews) (Ph.D & Kavlakoglu, 2024). Transfer learning starts with a pre-trained model or

network, utilizing the knowledge acquired from a source task or dataset (e.g., classifying movie reviews) to address a new, related target goal or dataset (Han et al., 2012).

2.5.1.3 Convolutional Neural Networks (CNNs)

A CNN, or Convolutional Neural Network, is a type of artificial intelligence that ranks among the most used deep learning algorithms. It is mostly employed for image analysis. It functions by dividing the image into smaller segments and examining them for patterns (Mohd, 2020). These patterns facilitate the identification of elements within the image, including edges and forms. Convolutional Neural Networks (CNNs) are utilised in several applications, such as facial recognition, driverless vehicles, and medical image analysis (Mohd, 2020).

IBM (2024) states that neural networks constitute a subset of machine learning and are essential to deep learning methodologies. The structure has node levels, which include an input layer, one or more hidden layers, and an output layer, as illustrated in Figure 2.5 (IBM, 2024).

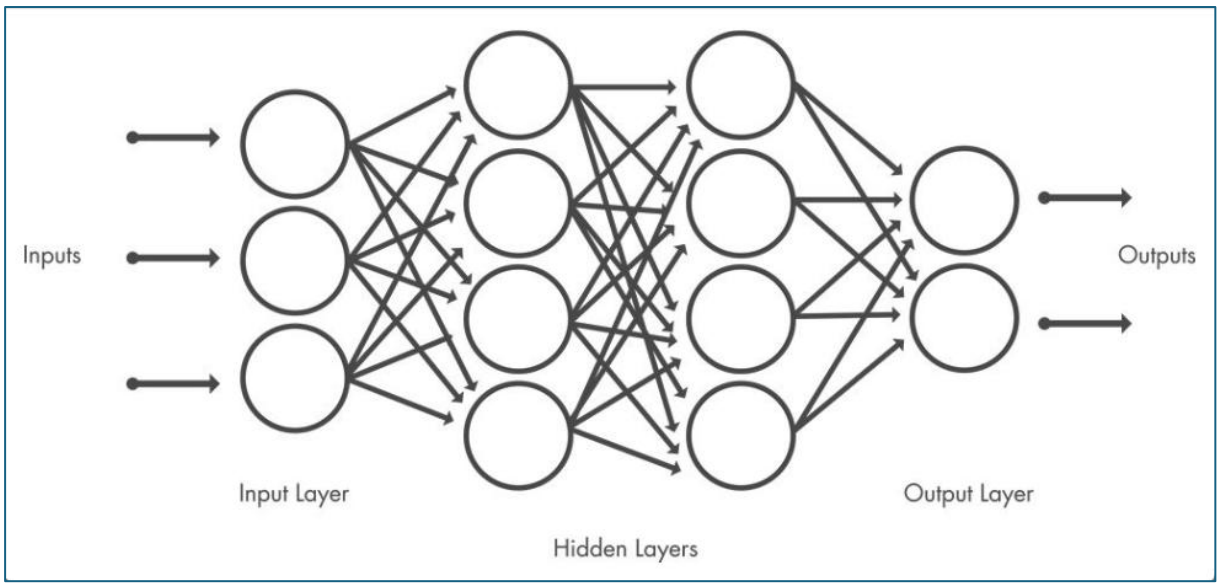


Figure 2.5: Convolutional Neural Network Layer

Source: https://www.mathworks.com/discovery/convolutional-neural-network.html

Each node is interconnected with another, having a specific weight and threshold (IBM, 2024). When a single node's output surpasses the established threshold, it activates and relays

data to the subsequent layer of the network (IBM, 2024). Without this, no data is conveyed to the succeeding layer of the network.

IBM (2024) identifies three primary types of layers: Convolutional layer, Pooling layer, and Fully-connected (FC) layer. The convolutional layer functions as the initial component of a convolutional neural network (CNN). Although further convolutional or pooling layers may precede it, the fully linked layer often occupies the last position (IBM, 2024). As the CNN analyses data sequentially, its complexity escalates, allowing it to discern increasingly greater segments of the image (IBM, 2024). The network primarily emphasises fundamental attributes such as edges and colours. As the data progresses through the layers, the network increasingly recognises more prominent shapes and elements until it ultimately identifies the target object (IBM, 2024).

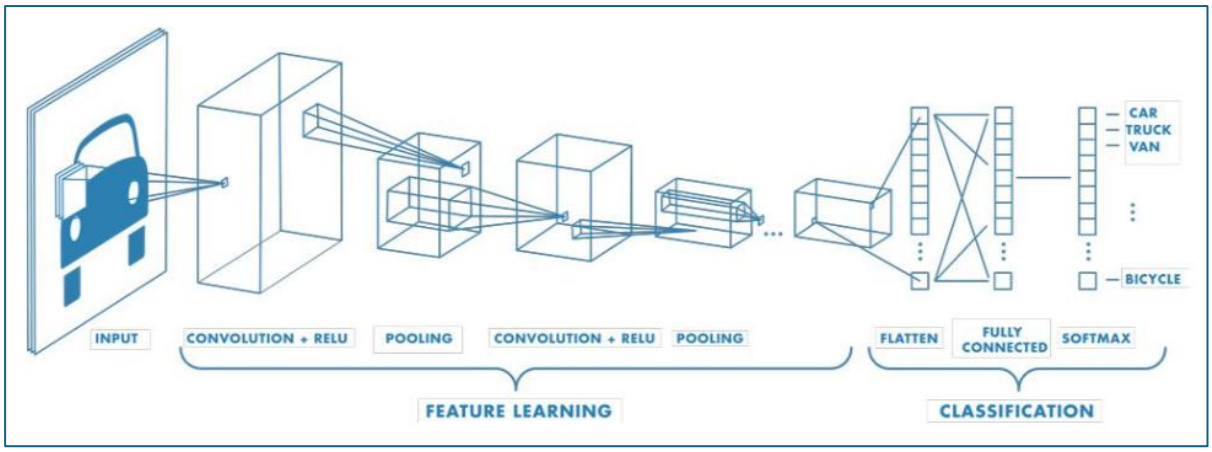


Figure 2.6: Architecture of a CNN

Source: https://www.mathworks.com/discovery/convolutional-neural-network.html

Figure 2.6 shows a network of numerous convolutional layers, which entails the application of filters to each training image at diverse resolutions (IBM, 2024). The result of each convolution operation functions as the input for the following layer, establishing a hierarchical processing framework.

As stated by MathWorks (2024), engineers and scientists often prefer to begin with a pre-trained model when working with CNNs, as it can be utilised to learn and recognise features from a new dataset. A pre-trained AI model is a deep learning model that has undergone training on large datasets to execute a certain task (Lee, 2022). It can be used in its original form or adapted to meet the requirements of various applications across different industries (Lee, 2022). These models are already trained to identify a diverse set of features and patterns in images. The importance of pre-trained CNN models lies in their capacity to provide a strong foundation for various computer vision tasks. According to Gupta (2024), while a pre-trained model might not achieve complete accuracy for your specific application, it significantly reduces the effort needed to develop a solution from scratch.

2.5.1.4 Datasets for Deep Learning

Deep learning data relates to the datasets employed for the training, validation, and testing of deep learning models. Deep learning, a subset of machine learning, utilises deep neural networks with numerous layers to learn from input and provide predictions (clickworker, 2024). These models necessitate significant quantities of labelled data for training to generalise efficiently to novel, unseen data.

Deep learning data typically comprises input-output pairs, where the input represents the features or attributes of the data, and the output signifies the corresponding labels or target values (clickworker, 2024). In image recognition tasks, the input comprises images, while the output labels indicate the class or category of items included in those images.

Clickworker (2024) claims that the quality and quantity of data are essential for the effectiveness of deep learning models. Deep learning models can acquire complex patterns and representations; nonetheless, they necessitate diverse and comprehensive datasets to perform effectively in practical applications (clickworker, 2024). Inadequate or biased data can result in

poor model performance, while high-quality and diversified datasets facilitate the creation of strong and accurate models.

2.5.1.5 TensorFlow

TensorFlow is an open-source library compatible with Python, employed for developing machine learning applications and neural networks. The fundamental elements of TensorFlow are defined here (Yegulalp, 2024).

TensorFlow is an open-source library developed by the Google Brain team and made publicly available in 2015, designed for numerical computing and extensive machine learning applications (Yegulalp, 2024). It integrates many machine learning and deep learning models and algorithms (neural networks) and makes them available through standard programming paradigms. TensorFlow features a user-friendly front-end API that allows developers to create applications using Python or JavaScript, while the core platform runs these applications using high-performance C++ (Yegulalp, 2024). Although TensorFlow provides libraries for several programming languages, Python remains the most used.

TensorFlow is a leading deep learning framework that competes with platforms like PyTorch and Apache MXNet. It excels in training and deploying deep neural networks across a wide range of applications, including natural language processing, image recognition, handwritten digit classification, word embeddings, recurrent neural networks, sequence-to-sequence models for machine translation, and simulations involving partial differential equations (PDEs) (Yegulalp, 2024). One of TensorFlow's significant advantages is its ability to support scalable production predictions, using the same models trained during development.

2.5.1.6 Django

Django is a framework constructed upon the Python programming language, designed with the "Don't Repeat Yourself" (DRY) principle at its foundation (Chen et al., 2020). This principle promotes writing concise, maintainable code and discourages redundancy, encouraging developers to build streamlined and efficient applications (Dekkati et al., 2019).

Django handles many essential aspects of web development, such as user authentication, site mapping, and content management (Madurapperuma et al., 2022). One of Django's standout features is its built-in administrative capabilities. The framework may autonomously produce a meticulously structured collection of administrative pages to execute create, read, update, and delete (CRUD) activities on models (Madurapperuma et al., 2022). These pages also support searching, filtering, and sorting lists. Every model registered with the admin class is displayed on the admin landing page, making it a highly efficient tool for managing application data (Madurapperuma et al., 2022).

A traditional data-driven website depends on a web application that receives HTTP requests from a web browser or other client (Mozilla, 2024). Upon receiving a request, the program determines the necessary steps based on the URL and any additional data sent via POST or GET methods (Mozilla, 2024). Subsequently, it may interact with a database to retrieve or save data or execute additional actions to fulfil the request (Mozilla, 2024). Ultimately, the application conveys a response to the browser, often generating an HTML page dynamically by inserting acquired data into specified placeholders within an HTML template (Mozilla, 2024).

2.5.1.7 Visual Studio Code

Visual Studio Code (VS Code) is a free, lightweight, and versatile source code editor compatible with Windows, macOS, Linux, Raspberry Pi OS, and available as a web version. It provides integrated support for JavaScript, TypeScript, and Node.js, and features an extensive

array of extensions for various programming languages (including C++, C#, Java, Python, PHP, and Go), runtimes (such as .NET and Unity), development environments (like Docker and Kubernetes), and cloud services (including Amazon Web Services, Microsoft Azure, and Google Cloud Platform).

VS Code is designed for swift initialization and optimal resource utilization, offering a variety of advanced features such as IntelliSense for variable, method, and imported module completion, graphical debugging, linting, multi-cursor editing, parameter hints, and other refined editing functionalities (Heller, 2022). It has code navigation, refactoring, and integrated source control with Git support, all of which were influenced by Visual Studio.

VS Code is constructed using Electron, Node.js, TypeScript, and the Language Server Protocol, and it undergoes monthly updates (Heller, 2022). Extensions are updated as needed, providing varying degrees of support for multiple programming languages, from basic syntax highlighting and bracket matching to extensive debugging and refactoring capabilities (Heller, 2022).

The source code for Visual Studio Code is accessible as open source under the MIT License (Heller, 2022). The product is provided under a standard Microsoft product license, featuring a restricted amount of Microsoft-specific customisations. Nevertheless, VS Code remains accessible free of charge.

2.5.2 Hardware

This section presents a summary of the hardware used in the development of the proposed web application.

2.5.2.1 Asus A556U

The development will primarily utilise an Asus A556U laptop, equipped with a 12GB RAM, Intel i5 7th Gen processor, and NVIDIA 930MX GPU. These specifications ensure adequate performance for model training and system integration.

2.6 Summary

In conclusion, this chapter offers a thorough examination of the main ideas, tools, and frameworks that are pertinent to the project. The chapter opens with an overview of deep learning and machine learning, emphasising its importance in solving complex problems. The chapter highlights the advantages of transfer learning, a technique that completes similar tasks by using pre-trained models.

The layered architecture of artificial neural networks, which makes it possible to extract significant patterns from data, is then covered. The significance of convolutional neural networks (CNNs), which are widely used in image processing applications due to their ability to recognise spatial hierarchies in visual data, is covered in this chapter.

After that, the chapter shifts to an explanation of web development technologies, emphasising the Django framework. It lists the features that make Django a popular option for creating reliable web applications, including built-in administration, scalability, and support for clean URL designs.

The chapter concludes by reviewing related works that have already been done and examining earlier studies and applications in image recognition and classification. The foundation for the project's strategy is laid by these references, which provide insightful information about the strategies and resources that have worked well for dealing with related problems. This project will use a deep learning approach with transfer learning to develop a web application using the Django framework. Because of its effectiveness and performance in

handling image classification tasks, EfficientNetB3, a member of the EfficientNet family, has been chosen as the model.

CHAPTER 3 REQUIREMENTS ANALYSIS & DESIGN

3.1 Introduction

This chapter describes the approach taken to create the suggested Dog Breed Classification system, a web application that uses deep learning methods. SCRUM, the chosen methodology, was picked because it can boost teamwork and speed up development. According to Drumond (2024), SCRUM is an Agile project management framework that offers teams a methodical way to plan and coordinate their work using a predetermined set of values, principles, and procedures.

Higher-quality code and more stable releases are the result of its iterative nature and frequent sprint testing, which allow for quick issue identification and resolution. SCRUM is ideally suited for this endeavour because of its capacity to break down big projects into smaller, more manageable tasks.

This chapter will explain the application of Agile-SCRUM in developing the Dog Breed Classification system, thoroughly exploring the stages of task planning, execution, and testing. It will also detail the system's design, all guided by Agile-SCRUM principles to ensure a structured yet adaptable approach to development.

3.2 Agile-SCRUM

The Agile Manifesto, composed in 2000 by a collective of developers seeking an innovative approach to software development, was the earliest document to discuss agility. Agile is a process that allows a team to implement a project more efficiently by segmenting it into various phases (Peek, 2024). Each phase facilitates continuous stakeholder participation to foster incremental enhancements at every level (Peek, 2024).

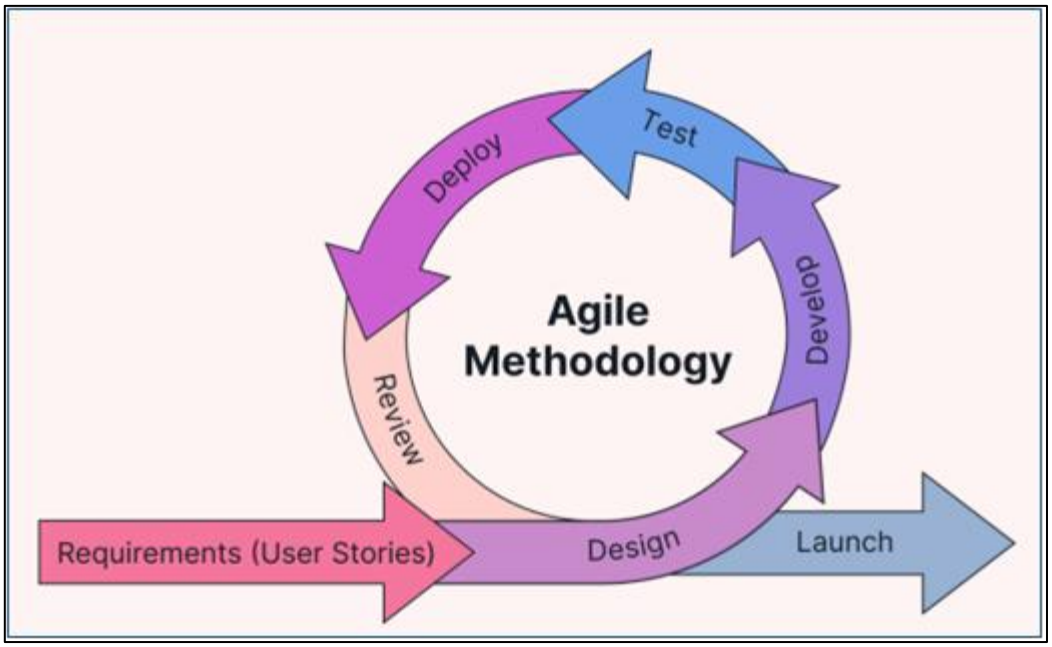


Figure 3.1: Agile Methodology Process

Source: https://www.linkedin.com/pulse/agile-methodologies-navigating-change-software-development-ejie-eukdf/

Figure 3.1 illustrates the Agile methodology process. There are six main steps in this process, which are requirements (often written as user stories), design, development, testing, review, and deployment. The cycle stresses continuous input and adaptation. The team starts by defining requirements and then moves on to user needs. This helps with the Design step, where ideas for solutions are formed. During the Develop step, the software is built. The Test step checks for quality and finds mistakes. During the Review step, stakeholders give their thoughts. Finally, the software is made available during the Deploy step. The circular flow shows that once the process is deployed, it goes back to the Requirements step and adds new requirements and feedback for the next round. This lets the process keep getting better and adapt to new needs.

Scrum is a widely used framework for developing complex products, which Ken Schwaber and Jeff Sutherland created. It is a type of Agile framework. Scrum is defined by the principles of empiricism and minimalist thinking, as per Drumond (2024). Empiricism claims

that knowledge is acquired through experience and that decisions are predicated on observations. Lean thinking prioritises necessities and reduces waste. The scrum framework is logic, relying on ongoing learning and adaptation to variable elements. It recognises that the team lacks complete knowledge at the project's beginning and will develop through experience. Scrum is designed to enable teams to seamlessly adjust to evolving conditions and user demands, incorporating re-prioritization inside the framework and utilizing brief release cycles to facilitate continuous learning and enhancement.

The scrum framework is a list of values, rules, and procedures that scrum teams must follow to create a product or service. It lists the people on a scrum team and what they are responsible for, the "artifacts" that describe the product and the work that goes into making it, and the scrum routines that help the team do its work.

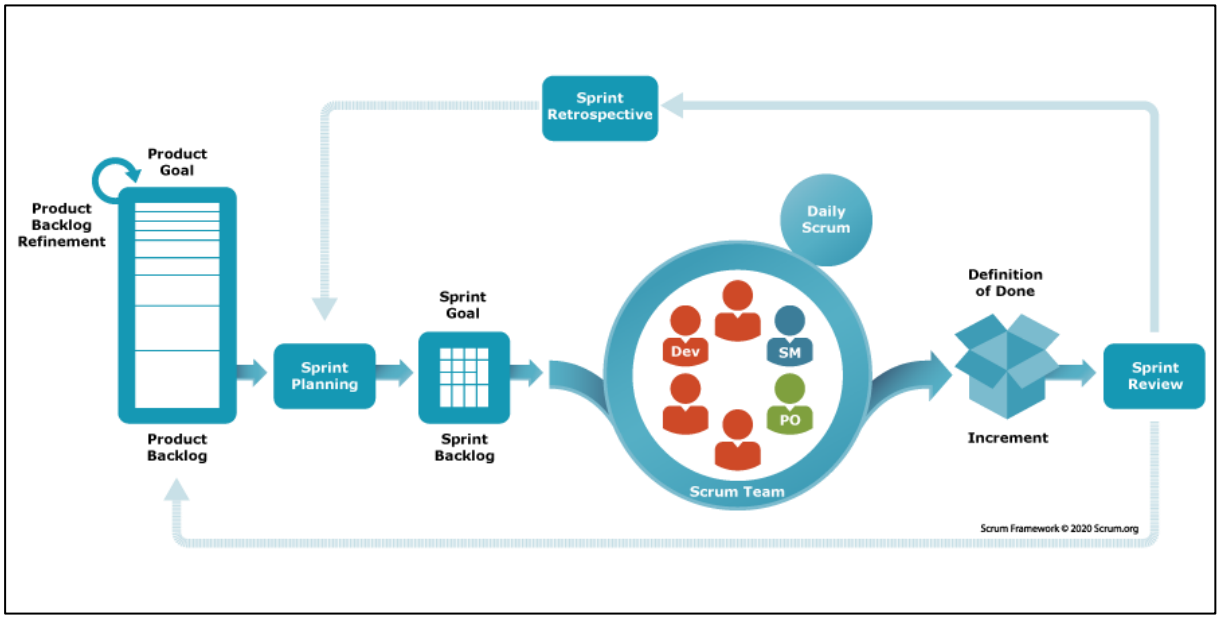


Figure 3.2: SCRUM Framework

Source: https://www.scrum.org/resources/what-scrum-module

Figure 3.2 illustrates the Scrum framework, a widely used Agile methodology for managing and developing complex products. It highlights the cyclical nature of Scrum, and the

key components involved in the process. At the core of Scrum is the Product Backlog, which serves as a prioritised list of all necessary items for the product, acting as the single source of requirements for any product changes. The Product Backlog Refinement is an ongoing process of detailing, estimating, and ordering items in the backlog. From this, The Sprint Planning meeting occurs, during which the Scrum Team chooses items from the Product Backlog to be delivered in the forthcoming Sprint, culminating in the Sprint Backlog and a defined Sprint Goal.

The Scrum Team consists of Developers (Dev), who perform the work to create the Increment, the Scrum Master (SM), who ensures that Scrum principles are followed and helps the team adhere to practices, and the Product Owner (PO), who maximises the value of the product. The Daily Scrum is a brief meeting, typically lasting 15 minutes, in which Developers synchronise their activities and plan for the next 24 hours. At the end of each Sprint, the team generates a Potentially Shippable Increment, which includes all completed Product Backlog items that meet the Definition of Done—a set of quality standards verifying the Increment's preparedness for delivery.

The Sprint Review is a meeting held at the conclusion of the Sprint to assess the Increment and modify the Product Backlog as needed, whereas the Sprint Retrospective takes place subsequent to the Sprint Review to appraise the Sprint’s effectiveness concerning personnel, processes, tools, and the Definition of Done. The arrows in the diagram show the flow of work and information throughout the framework, emphasizing its iterative and incremental process, where each Sprint produces a potentially shippable Increment, continuously improving the product. The figure thus visually encapsulates the Scrum framework, showcasing the interactions among events, artifacts, and roles within the Scrum process.

To conclude, the Agile Scrum technique integrates the Agile philosophy with the Scrum framework. Agile denotes an "incremental" approach, enabling teams to advance projects through small increments. Scrum is an adaptation of the Agile technique. It is renowned for breaking down projects into sizable sections known as "sprints." Requirements, Design, Development, Testing, Deployment, and Review are the six steps that make up each sprint of Scrum, an Agile methodology. Each step will be thoroughly explained in the upcoming subsections.

3.3 Requirements

The first stage of project development is requirements planning, which focusses on determining the main business issue and establishing precise goals. Creating an initial project plan, conducting interviews, and brainstorming sessions are just a few of the activities that take place during this phase. To make sure the suggested web application meets the needs and expectations of users, obtaining application requirements from them is a crucial part of this phase. In order to properly plan business processes for the new web application, it also entails carefully analysing and comprehending current systems. To reduce the possibility of misunderstandings between developers and stakeholders, all requirements must be finalised with the user before proceeding to the user design phase.

A set of target users will receive a questionnaire during the requirements phase. This is done in order to ascertain their familiarity with related applications, their expectations of a web application that classifies dog breeds, and their knowledge of dog breeds. Following the collection of questionnaire responses, a thorough analysis of the responses will be carried out. User requirements will be established in light of this analysis. At this stage, the system's hardware and software requirements will also be determined.

3.3.1 Analysis of Similar Existing Systems

Numerous dog breed classification models and applications created with deep learning techniques are highlighted in the review of existing systems. Convolutional neural networks (CNNs), which use pre-trained models like MobileNet, ResNet, or VGGNet, are primarily used in these systems. To improve classification accuracy, they are trained on large datasets, such as the Kaggle dog breed dataset.

While some systems provide web-based platforms or mobile applications for user accessibility, they are either functionally limited or do not offer the level of simplicity needed for non-technical users. These results highlight the need for a system that provides educational content while achieving high classification accuracy and usability.

3.3.2 Analysis of Proposed Application

The proposed web application for classifying dog breeds addresses the drawbacks of the existing systems while enhancing their positive aspects. To guarantee precise breed identification, the system will make use of transfer learning, utilising the EfficientNetB3 model and the Kaggle dog breed dataset. The web-based platform of the application, which makes it easy to upload dog photos and displays the percentage of dog breeds that are similar, is what makes it special.

In order to ensure that the application serves a diverse population, the emphasis is on improving accessibility and interaction. Furthermore, in comparison to other current methods, the suggested system is more effective and simpler to implement due to its optimised design and integration of the training and prediction phases.

3.3.3 Questionnaire

A questionnaire was administered to 30 respondents to gather their requirements. This was undertaken to better understand the expectations of potential users for the planned dog breed classification web application. Additionally, the questionnaire aimed to assess whether the proposed application meets the needs of potential users. The respondents included university students and professionals from various fields. A sample of the questionnaire used for this purpose is provided in Appendix A. Once the responses were collected, a comprehensive analysis was conducted and summarised. The questionnaire consists of four sections which are Demographics and Background, General Features, Result Display, and User preferences and Needs.

3.3.3.1 Demographics

The questionnaire is completed by respondents aged between 18 and 35 years old. Figure 3.3 shows the pie chart for the survey.

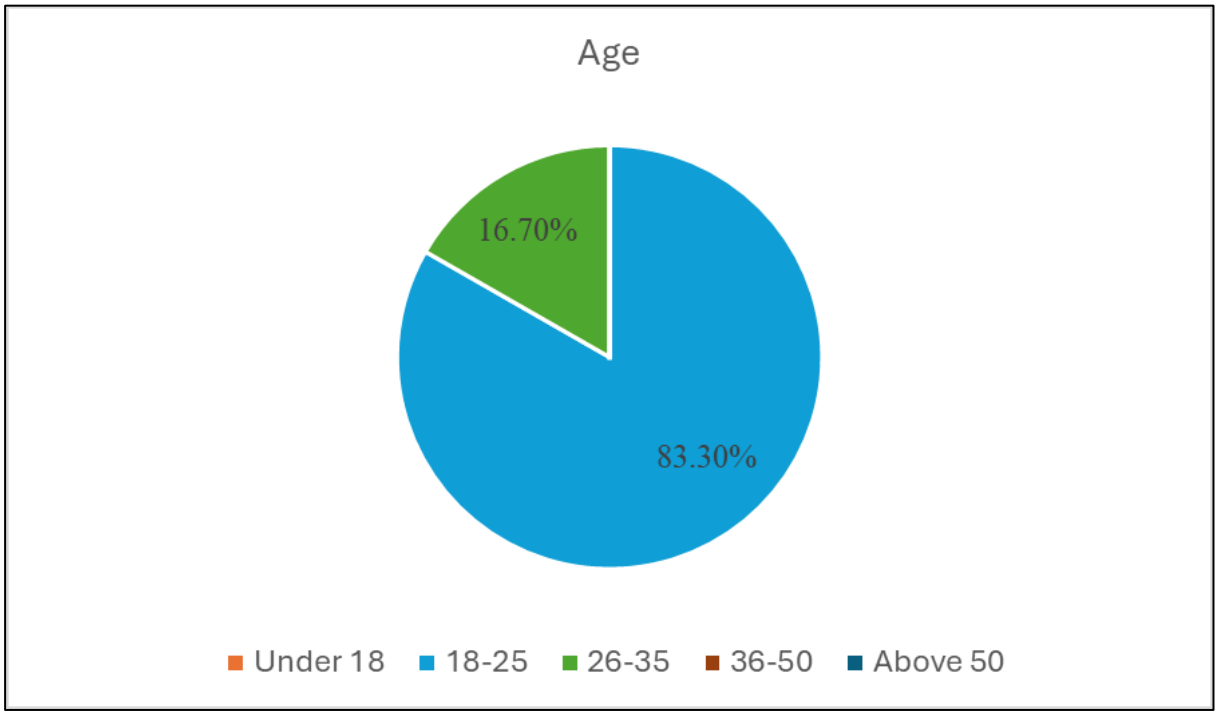


Figure 3.3: Age of Respondent

Figure 3.3 illustrates the distribution of individuals across different age groups. The 18-25 age group dominates, comprising 25 people (83.30%) of the respondents, while the 26-35 age group makes up 5 people (16.70%). There is no respondent for 26-35, 36-50, and above 50 age groups.

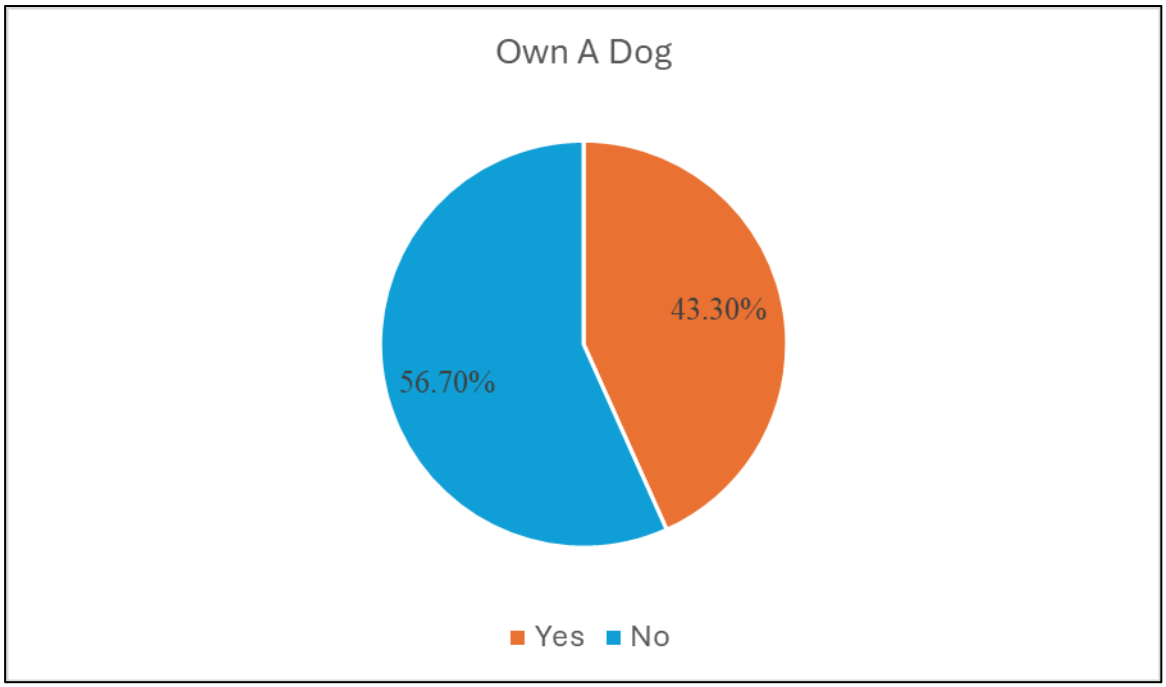


Figure 3.4: Percentage of Dog Owners

Figure 3.4 illustrates the percentage of people who own a dog versus those who do not. A significant majority of the population, representing 56.70%, does not own a dog, while the remaining 43.30% owns a dog.

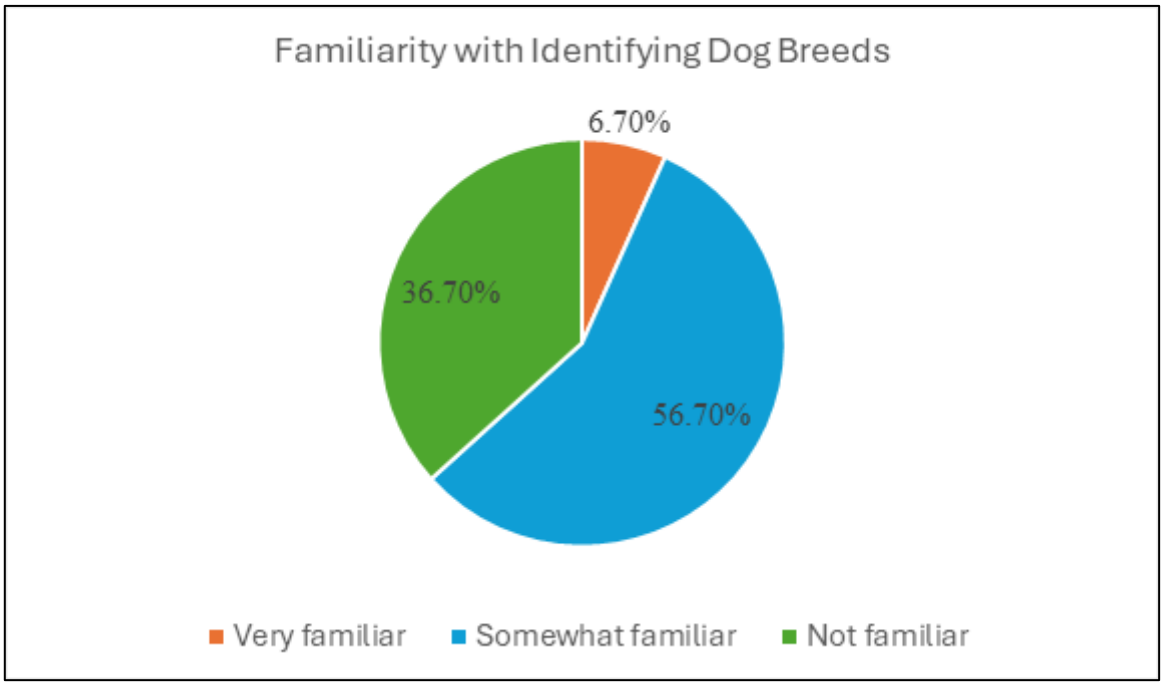


Figure 3.5: Familiarity with Identifying Dog Breeds

Figure 3.5 illustrates the level of familiarity people have with identifying dog breeds. A significant majority, 17 people (56.70%), feel somewhat familiar with recognizing different breeds. There are 11 people (36.70%) who are not familiar with identifying dog breeds. Only a small percentage, 2 people (6.70%), consider themselves very familiar with recognizing dog breeds. This suggests that while many people have some knowledge of dog breeds, a deep understanding of breed identification is not widespread.

3.3.3.2 General Features

This section will show the general features that the respondents wish to have in the web application.

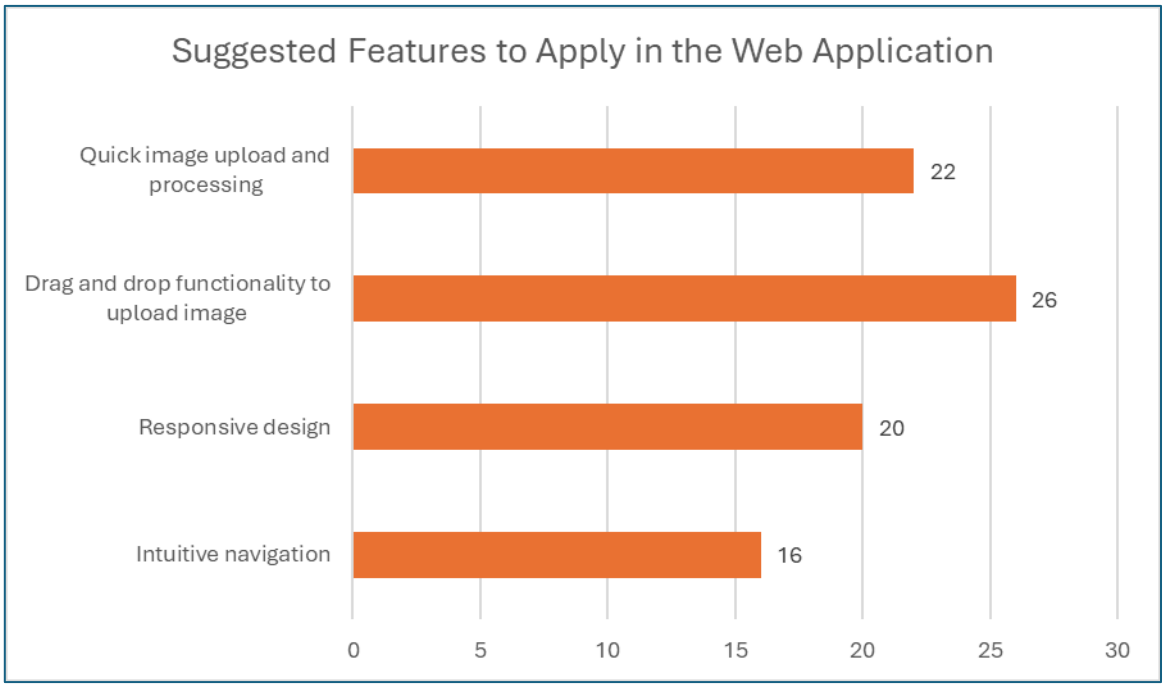


Figure 3.6: Suggested Features to Apply in the Web Application

Figure 3.6 shows the results of suggested features to apply in the proposed web application. The most popular suggestion is "Drag and drop functionality to upload image" with 26 votes, followed by "Quick image upload and processing" with 22 votes, "Responsive design" with 20 votes, and "Intuitive navigation" with 16 votes. This data suggests that users prioritise features that make image uploading and navigation easier and more efficient in the web application.

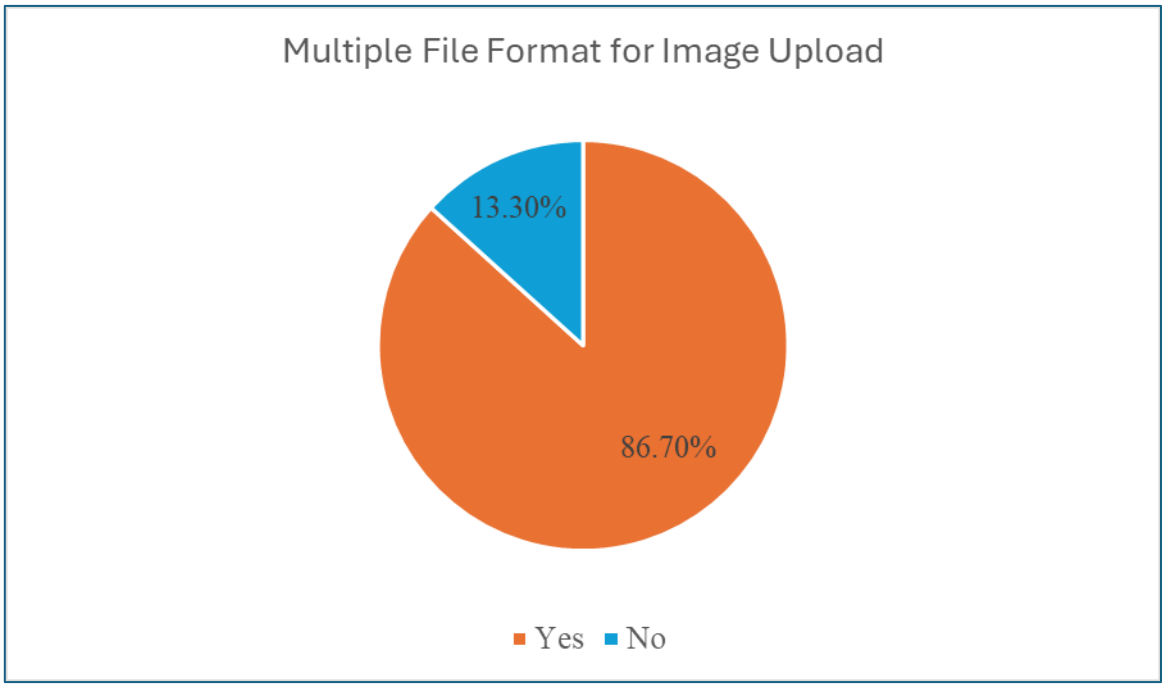


Figure 3.7: Multiple File Format Preference

Figure 3.7 illustrates the preference for multiple file formats when uploading images. A significant majority, 26 people (86.70%), support the inclusion of multiple file formats, while only 4 people (13.30%) oppose it. This suggests that users would benefit from the flexibility to upload images in different formats, which would likely improve compatibility and convenience when working with various image types.

3.3.3.3 Results Display

This section will outline the expected results interface that respondents hope to achieve in the web application.

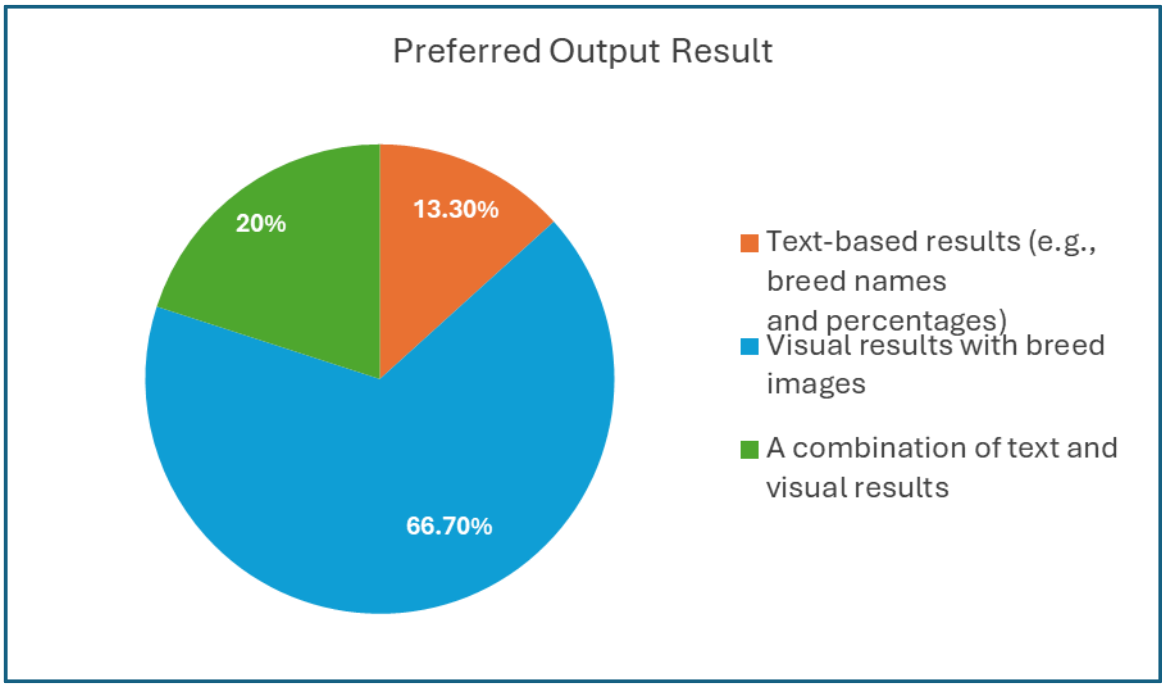


Figure 3.8: Preferred Output Result

Figure 3.8 illustrates the preferred output result for a dog breed identification tool. Most users, 20 people (66.70%), prefer a combination of text-based results (breed names and percentages) and visual results (breed images). A smaller portion, 6 people (20%), are satisfied with text-based results alone, while 4 people (13.30%) would prefer visual results with breed images only. This data suggests that users value both accuracy and visual confirmation when identifying dog breeds, indicating a preference for a comprehensive and engaging output format.

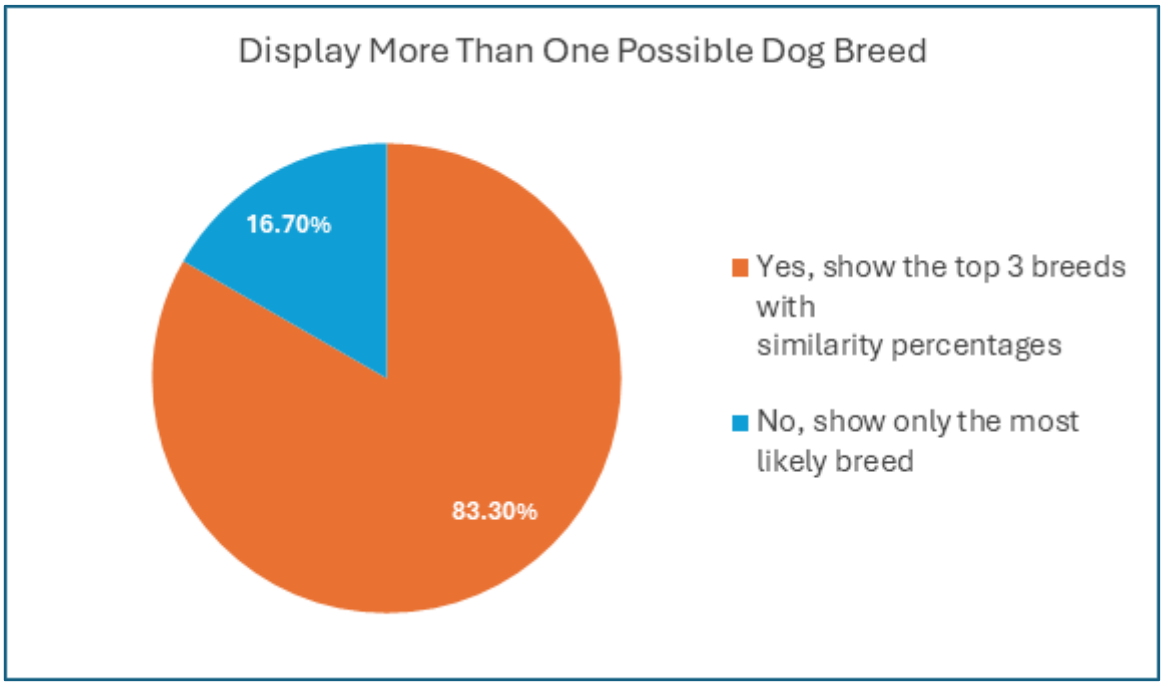


Figure 3.9: Display More Than One Possible Dog Breed

Figure 3.9 shows the user preference for displaying multiple possible dog breeds in a breed identification tool. A significant majority, 25 people (83.30%), favour displaying only the most likely breed. A smaller portion, 5 people (16.70%), prefer to see the top 3 breeds with similarity percentages. This suggests that most users prioritise clear and concise results, while a minority values exploring potential alternative breed matches.

3.3.3.4 User Preferences and Needs

This section will display the user's preferences and needs within the web application.

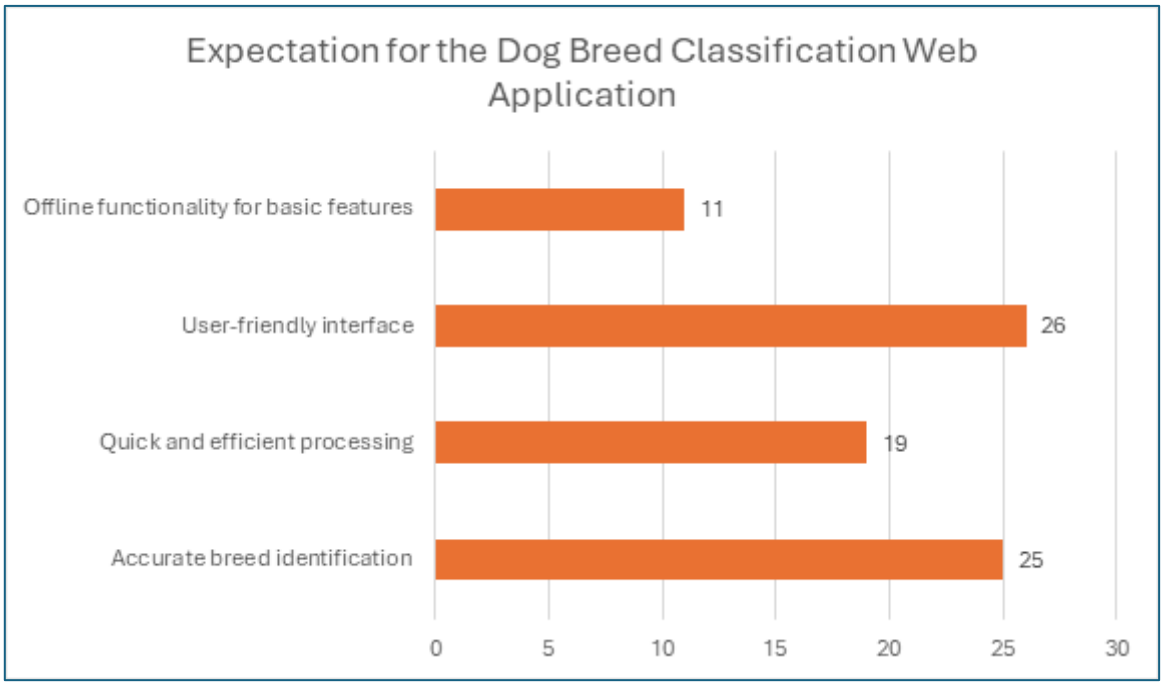


Figure 3.10: Expectation for the Dog Breed Classification Web Application

Figure 3.10 shows the expectations for a dog breed classification web application. The most important expectations are for a "User-friendly interface" (with 26 responses), followed by "Accurate breed identification" (with 25 responses), and "Quick and efficient processing" (with 19 responses). "Offline functionality for basic features" is the least important expectation, with 11 responses. This data suggests that users prioritise ease of use, accurate results, and fast processing speed in a dog breed classification web application, while offline functionality is less of a concern.

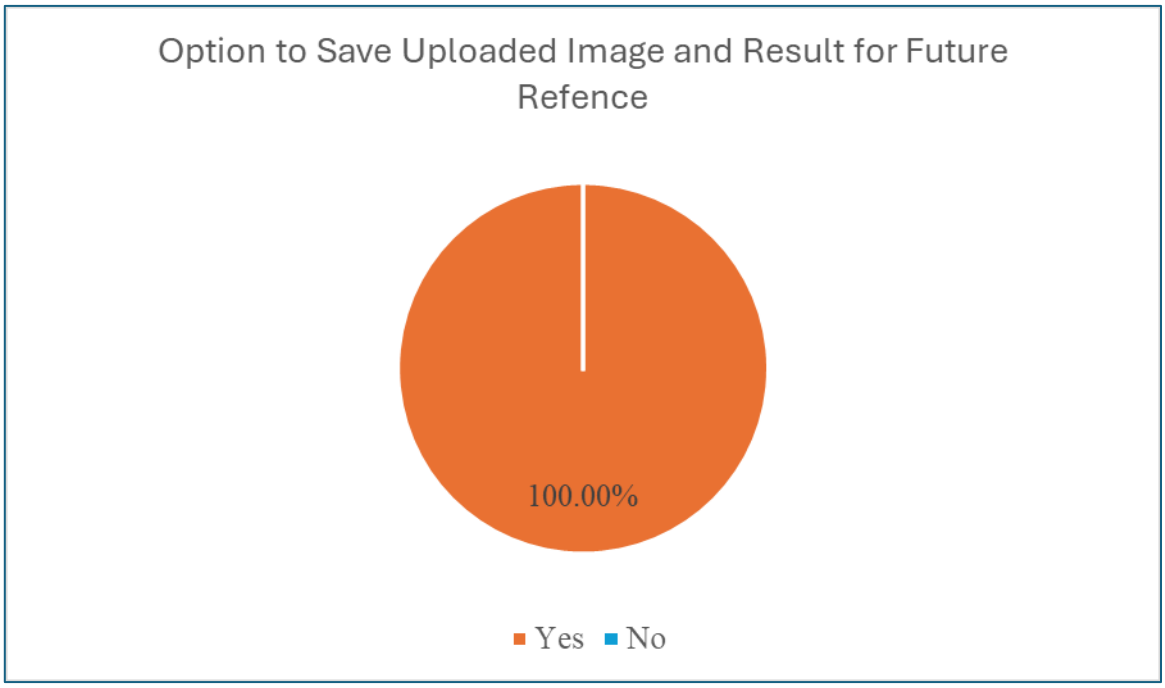


Figure 3.11: Option to Save Uploaded Image and Result for Future Reference

Figure 3.11 illustrates the overwhelming support for an option to save uploaded images and their corresponding breed identification results for future reference. All 30 respondents (100%) expressed a desire for this feature. This indicates a strong user preference for a tool that allows them to easily revisit and refer to previous breed identifications, potentially for personal records, research, or comparison purposes.

3.3.4 Software Requirements

The software requirements for the proposed system are shown in Table 3.1.

Table 3.1: Software requirements for development

No.	Software	Description
1	Microsoft Windows 10 64-bit Operating System	For development and testing.
2	Visual Studio Code	As the primary code editor for web application development.

3	Deep Learning Framework	TensorFlow / Keras: For implementing and training the deep learning model using transfer learning with EfficientNetB3.
4	Jupyter Notebook / Google Colab / Kagle Notebook	For model experimentation and training.
5	Google Chrome / Mozilla Firefox / Microsoft Edge	Serves as a testing platform for evaluating the interface and functionality of the proposed web application.

3.3.5 Hardware Requirements

The following Table 3.2 provides a concise overview of the hardware components to be employed in the development.

Table 3. 2: Hardware requirement for development

No.	**Hardware**	**Description**
1	Intel i5/i7 or AMD Ryzen 5/7 (or equivalent)	For smooth development and model training.
2	Minimum 6 GB RAM	For handling model training and running the development environment simultaneously.
3	NVIDIA GeForce 930MX or higher	For accelerated training and inference.

3.4 Design

Following the requirements phase in the Agile iteration process, the design phase ensues. At this stage, there are two types of design: logical design and physical design. The following diagrams depict logical designs:

- Overall System Architecture Diagram
- Flowchart
- Data Flow Diagram (DFD)
- Entity Relationship Diagram (ERD)
- Data Dictionary

On the other hand, physical design is represented by using wireframes.

3.4.1 Overall System Architecture Diagram

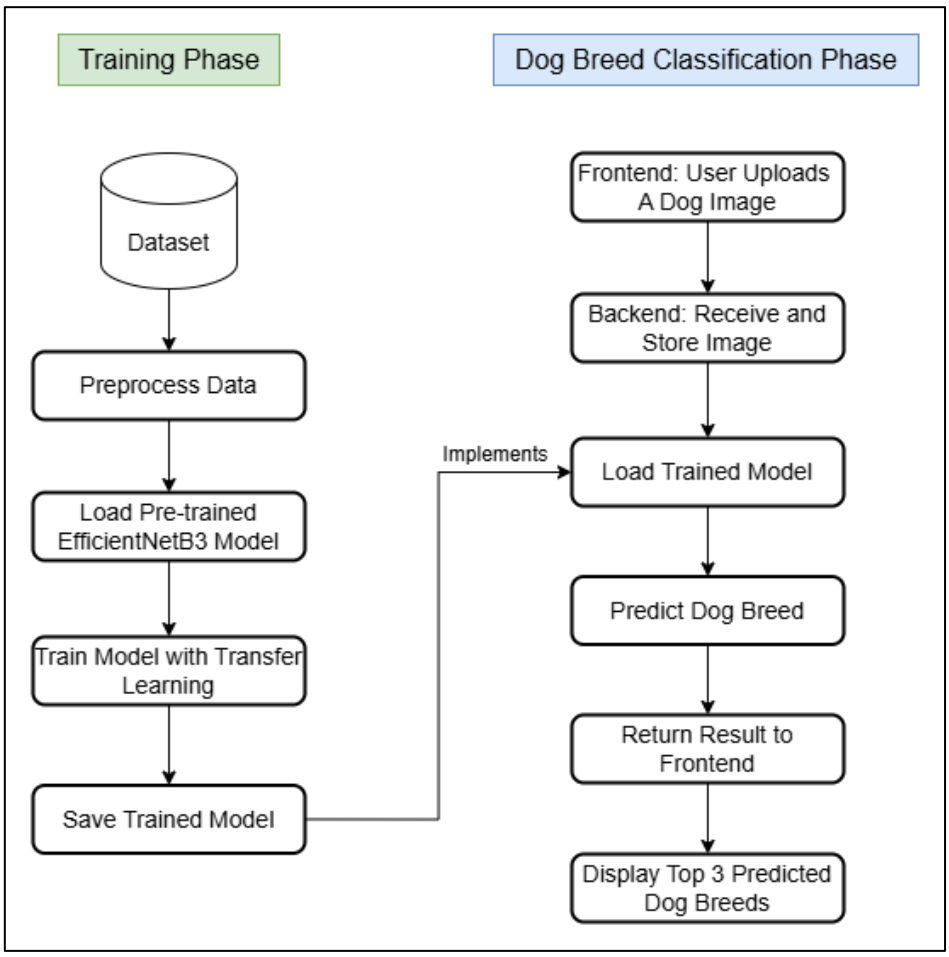


Figure 3.12: Overall System Architecture Diagram

Figure 3.12 illustrates the overall system architecture for the proposed dog breed classification web application, divided into two distinct phases: the Training Phase and the Dog Breed Classification Phase (Deployment Phase). The Training Phase begins with a "Dataset," which is then "Preprocessed." A "Pre-trained EfficientNetB3 Model" is loaded. The model is then "Train with Transfer Learning." Finally, the resulting "Trained Model" is saved. The Dog Breed Classification Phase starts with the "Frontend," where the "User Uploads a Dog Image." The "Backend" then "Receives and Preprocesses Image." The previously saved "Trained Model" is loaded, and the "Dog Breed" is "Predicted." The "Result" is then "Returned to Frontend," where the "Top 3 Predicted Dog Breeds" are displayed to the user.

3.4.2 Flowchart

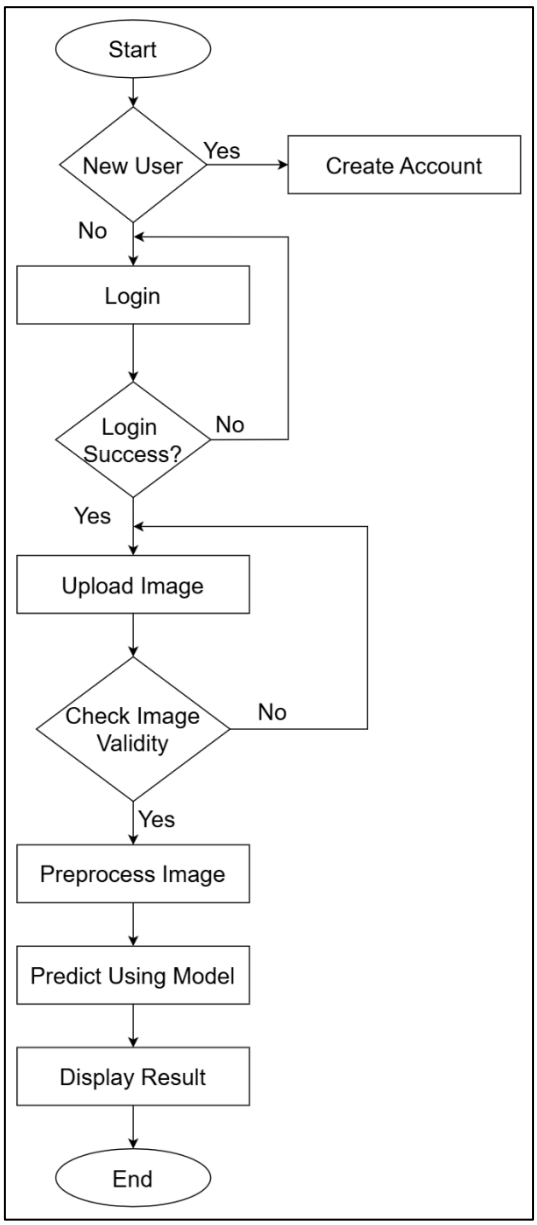


Figure 3.13: Flowchart of classifying dog breeds

Figure 3.13 shows the flow of classifying a dog breed in the proposed web application. The process begins with the "Start" point. The user will then need to create a new account. If the user has an account, they can proceed to the next stage, which is "Upload Image." A decision point follows: "Check if Image Contains a Dog." If the image does not contain a dog ("No"), the process loops back to the "Upload Image" step, prompting the user to upload a different image. If the image does contain a dog ("Yes"), the process proceeds to "Preprocess Image,"

where the image is prepared for the model (e.g., resizing, normalization). Next, the "Predict Using Model" step utilises the trained deep learning model to classify the dog breed. Finally, the "Display Result" step presents the predicted breed(s) to the user, and the process ends at the "End" point.

3.4.3 Data Flow Diagram (DFD)

Data flow diagrams are often used to illustrate the flow of data within a system. The components of a data flow diagram comprise external entities and processes.

3.4.3.1 Context Diagram / Level 0 Diagram

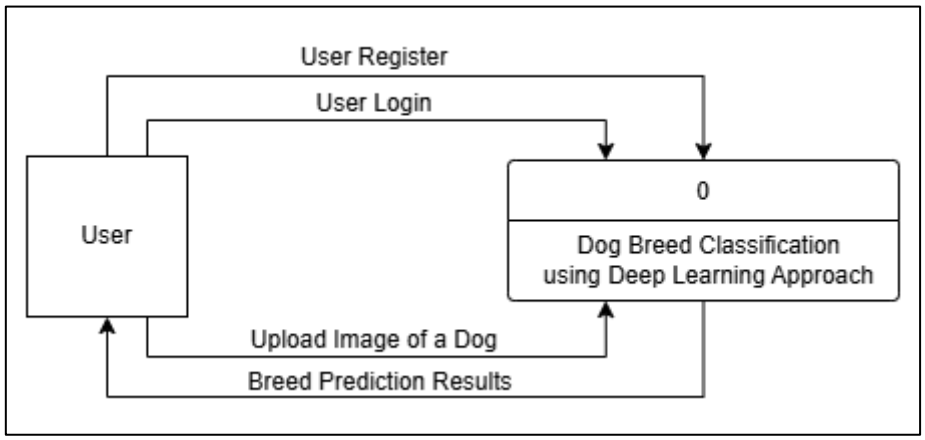


Figure 3.14: Context Diagram of Dog Breed Classification Web Application

Figure 3.14 is a Context Diagram, sometimes referred to as a Level 0 Data Flow Diagram, depicting the interaction between a user and a dog breed classification system. The diagram illustrates a "User" as an external entity engaging with the primary process, designated as "Dog Breed Classification using Deep Learning Approach." The user inputs data into the system by uploading a photograph of a dog. The system subsequently delivers output to the user as "Breed Prediction Results." The figure illustrates the system's boundaries and its interaction with the external user, delineating the flow of information between the two entities. It represents a high-level overview of the system's purpose and its external interactions.

3.4.3.2 Level 1 Diagram

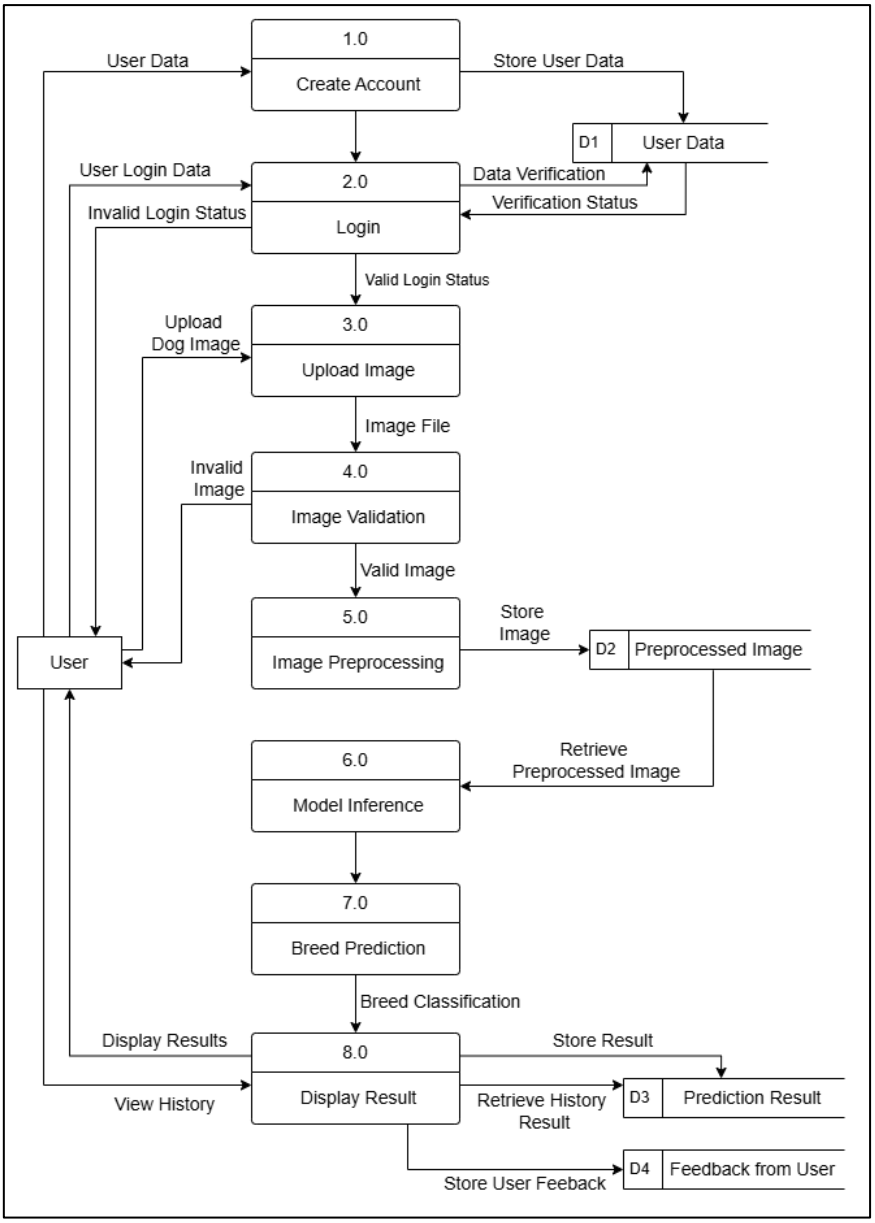


Figure 3.15: Level 1 Diagram of Dog Breed Classification Web Application

Figure 3.15 illustrates the Level 1 Diagram of the Dog Breed Classification Web Application. This figure offers a comprehensive summary of the dog breed identification system. The process starts with the user engaging with the system by creating an account and logging in. Upon logging in, the user may upload an image of a dog. The system subsequently authenticates the submitted image and performs image preparation procedures. Thereafter, the preprocessed image is input into a model inference procedure to determine the dog breed. The

system subsequently presents the forecast results to the user, allowing them to review their prediction history and provide feedback on the accuracy of the outcomes. The system stores user data, preprocessed photos, prediction outcomes, and user feedback for subsequent reference or study.

3.4.3.3 Level 2 Diagram

The Level 2 diagram offers a refined view of the system by further dissecting the sub-processes detailed in the Level 1 Data Flow Diagram (DFD) into supplementary sub-processes (GeeksforGeeks, 2019). Each sub-process is represented as a separate process in the level 2 DFD. The data flows and repositories relevant to each sub-process are also illustrated (GeeksforGeeks, 2019). According to GeeksforGeeks (2019), a 2-Level Data Flow Diagram (DFD) provides a more thorough examination of the components present in a 1-Level DFD. It can be utilised to prepare or document essential information regarding the system's performance.

3.4.3.3.1 DFD Level 2 for Process 1.0

Figure 3.16 below illustrates the Data Flow Diagram Level 2 for Process 1.0, which is the process for registering a user.

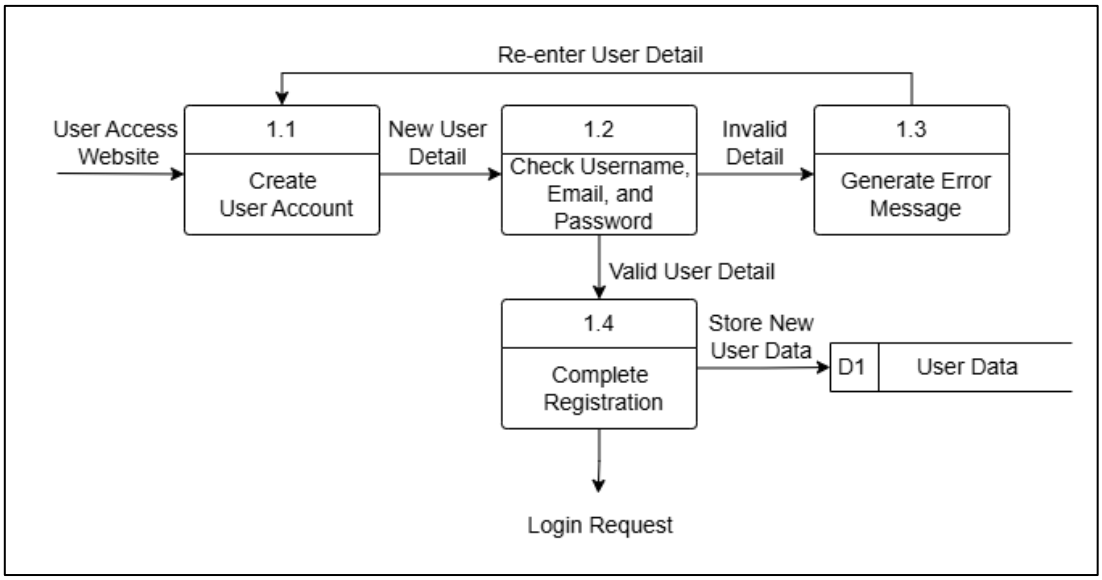


Figure 3.16: Data Flow Diagram Level 2 for Process 1.0

Process 1 outlines the user registration procedure. After accessing the website, users must enter their username, email address, and password to register for a new account. The system checks for appropriate email format, password difficulty, and duplicate email addresses, therefore validating this input. If the input is invalid, an error notice is shown to the user. Once the details have been validated, the system records the user's data in the database, therefore completing the registration process. The system then guides the user to the home page, allowing them to access the web application using their recently registered account.

3.4.3.3.2 DFD Level 2 for Process 2.0

Figure 3.17 below illustrates the Data Flow Diagram Level 2 for Process 2.0, which is the process for User Login.

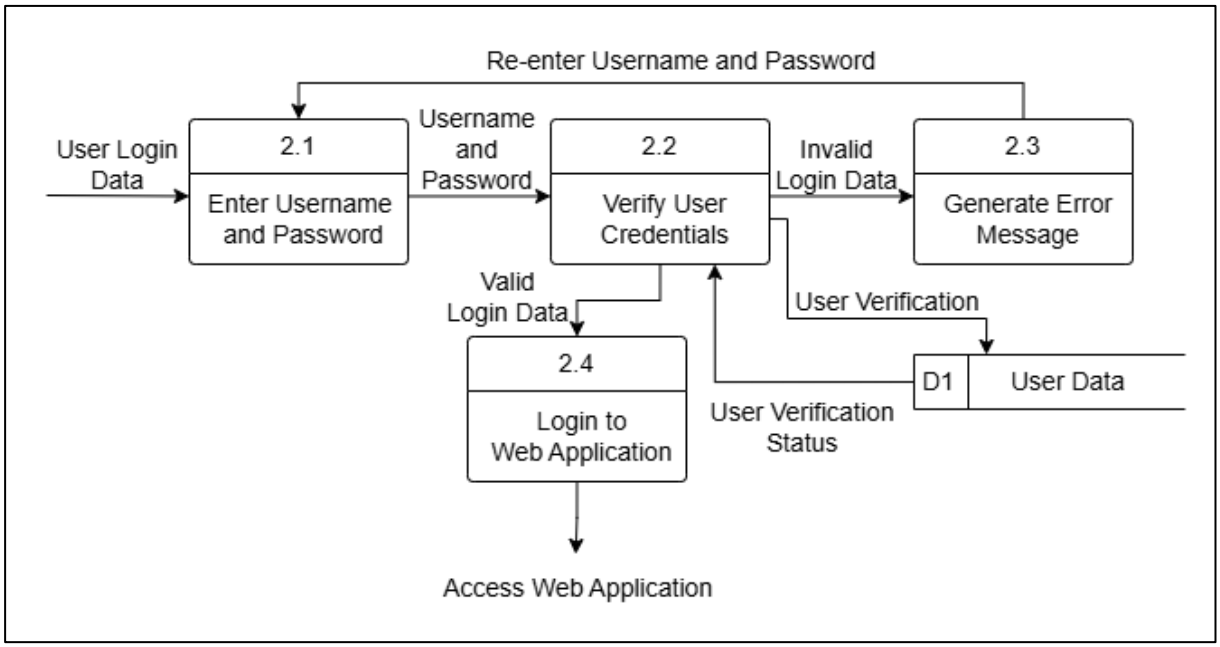


Figure 3.17: Data Flow Diagram Level 2 for Process 2.0

Process 2 illustrates the user login process. The user is required to fill in their username and password to log in. If the login credentials are valid, the user will successfully log into the web application. Subsequently, if the login credentials do not match the data stored in the database, the user will be required to enter the correct credentials.

3.4.3.3.3 DFD Level 2 for Process 3.0

Figure 3.18 below illustrates the Data Flow Diagram Level 2 for Process 3.0, which is the process for Upload Image.

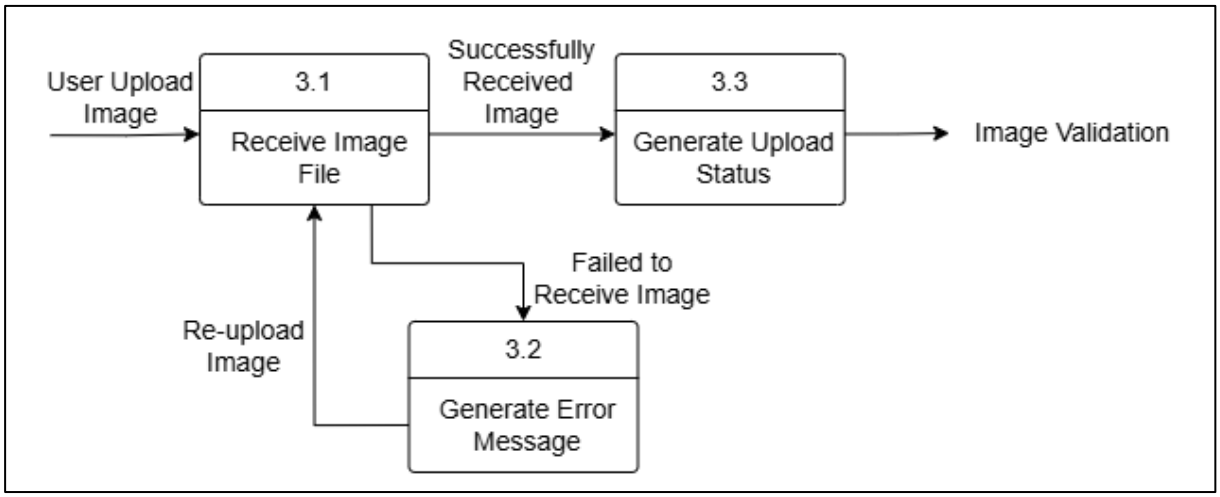


Figure 3.18: Data Flow Diagram Level 2 for Process 3.0

Process 3.0 shows the process of receiving an image file from a user. The system starts trying to obtain the file when a user uploads an image. If the upload is successful, the system records this with a "Ready to upload!" notice. Whereby, if the upload fails due to reasons such as a missing file or connection problem, an error notice appears, guiding the user to attempt uploading the image again. Once the image is acquired, it moves on to the next level for additional checks, including content and image format validation.

3.4.3.3.4 DFD Level 2 for Process 4.0

Figure 3.19 below illustrates the Data Flow Diagram Level 2 for Process 4.0, which is the process for Image Validation.

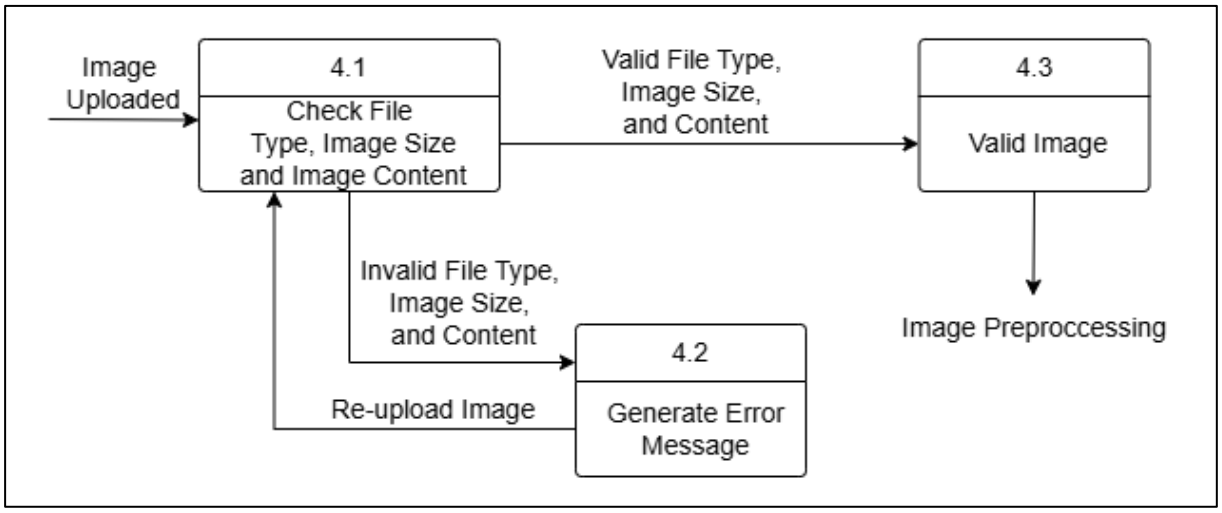


Figure 3.19: Data Flow Diagram Level 2 for Process 4.0

Process 4.0 shows the process of validating an uploaded image. A picture is evaluated immediately upon upload to ensure it meets the image standards. The uploaded photo will be examined to ensure it contains the correct dog breed, is in the correct file type (PNG, JPEG, or WEBP), and meets the specified size requirements. If any of these tests fail, due to inaccurate size, improper file type, or unsuitable content, the user receives an error message and is directed to resubmit the photo. The picture can proceed with image preprocessing, as long as every tick passes.

3.4.3.3.5 DFD Level 2 for Process 5.0

Figure 3.20 below illustrates the Data Flow Diagram Level 2 for Process 5.0, which is the process for Image Preprocessing.

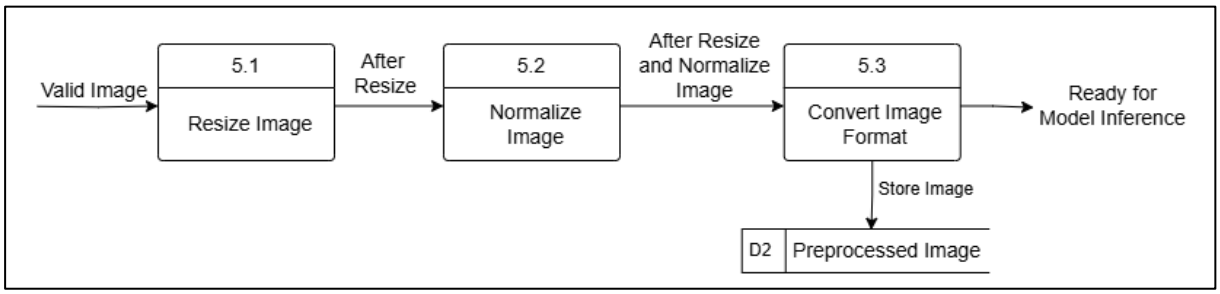


Figure 3.20: Data Flow Diagram Level 2 for Process 5.0

Process 5.0 shows the image preprocessing stage. The uploaded valid image will be resized first. After the picture has been resized, it is normalised by setting each pixel's value to a specific range, usually between 0 and 1. Lastly, the picture is converted into a format compatible with the deep learning model used for inference, such as a NumPy array. The preprocessed picture is then saved so that it can be used again in the model inference process.

3.4.3.3.6 DFD Level 2 for Process 6.0

Figure 3.21 below illustrates the Data Flow Diagram Level 2 for Process 6.0, which is the process for Model Inference.

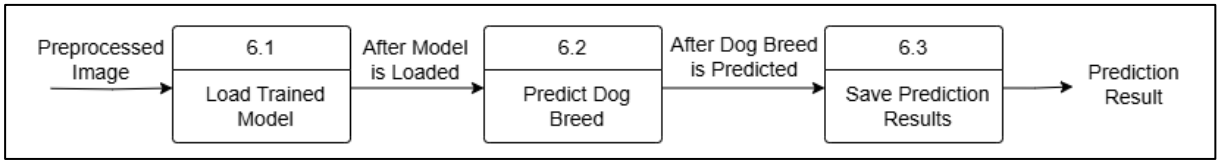


Figure 3.21: Data Flow Diagram Level 2 for Process 6.0

Process 6.0 describes the process of predicting dog breeds. The preprocessed picture is first sent to the trained model that has been loaded. The model then processes the picture and, using what it has learnt, guesses what kind of dog it is. The prediction result is then saved so that it can be used or analysed again in the future.

3.4.3.3.7 DFD Level 2 for Process 7.0

Figure 3.22 below illustrates the Data Flow Diagram Level 2 for Process 7.0, which is the process for Breed Prediction.

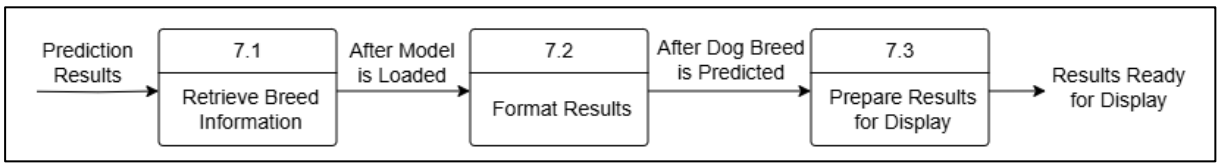


Figure 3.22: Data Flow Diagram Level 2 for Process 7.0

Process 7.0 shows the Breed Prediction process. The system obtains more details about the predicted dog breed, including its traits, personality, and grooming needs, after the model has made the prediction. Finally, the formatted results are prepared to be displayed to the user, providing them with all the necessary information about the expected dog breed.

3.4.3.3.8 DFD Level 2 for Process 8.0

Figure 3.23 below shows the Data Flow Diagram Level 2 for Process 8.0, which is the process for Display Result.

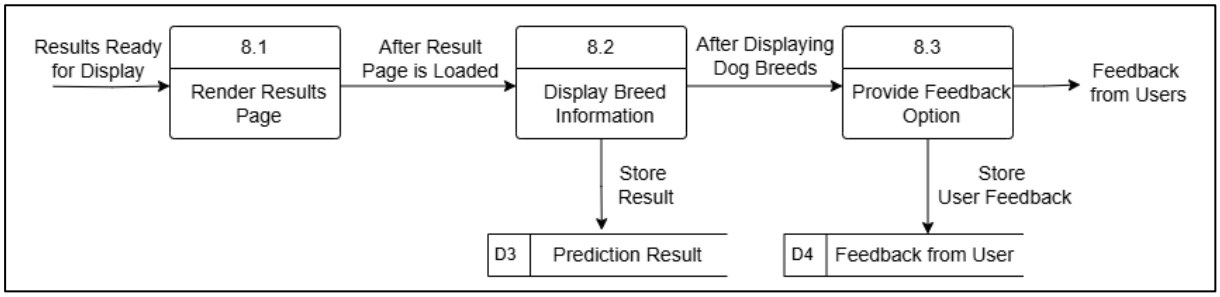


Figure 3.23: Data Flow Diagram Level 2 for Process 8.0

Process 8.0 shows the final stages of the user interface for the dog breed classifier web application. Once the prediction results are ready, the system displays a results page that shows the predicted dog breed and their information, such as breed characteristics. The results are then stored in the database. After displaying the dog breed result, there will be a feedback section at the bottom of the result page. The feedback from the user is then stored in the database.

3.4.4 Entity Relationship Diagram (ERD)

An entity relationship diagram (ER diagram or ERD) visually represents the relationships among components within a database (Belcic & Stryker, 2024). Entity-Relationship Diagrams (ERDs) are a distinct type of flowchart that defines the relationships among different entities inside a system (Belcic & Stryker, 2024).

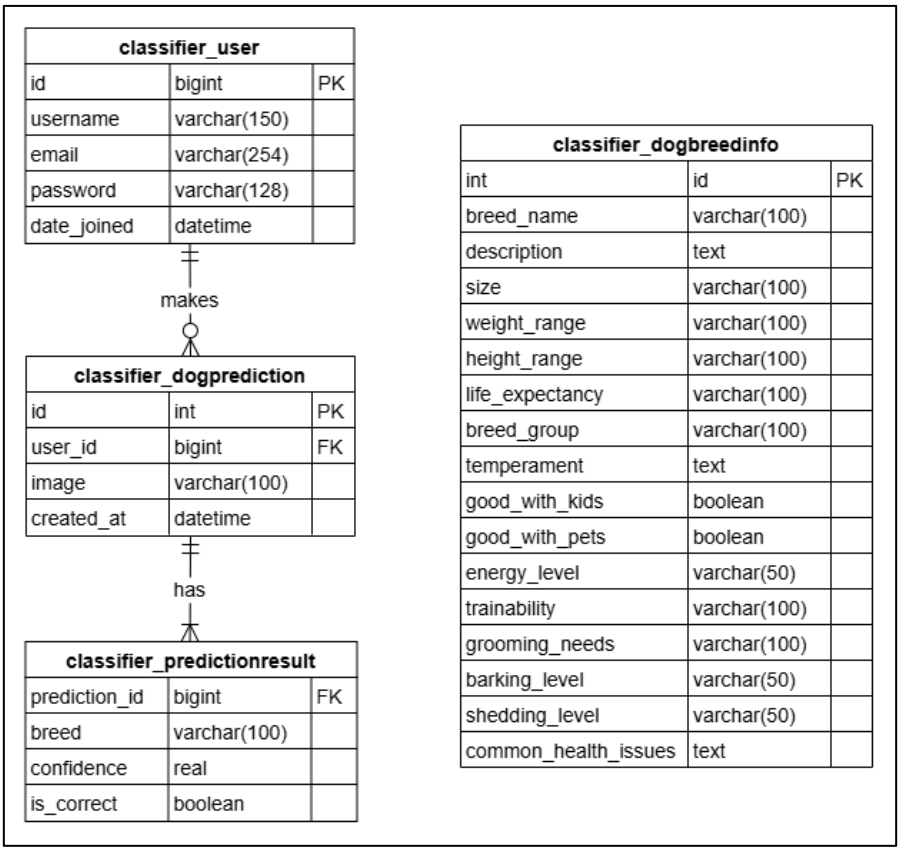


Figure 3.24: Entity Relationship Diagram (ERD)

To represent the relationships among the key components of the proposed web-based dog breed classification system, an Entity Relationship Diagram (ERD) using Crow's Foot notation is presented in Figure 3.24. The system consists of four main entities:

- User
- Dog Prediction
- Prediction Result
- Dog Breed Info

The database design includes the following four tables:

- classifier_user
- classifier_dogprediction
- classifier_predictionresult

- classifier_dogbreedinfo

As shown in Figure 3.24, a single user can make zero or more dog breed predictions, establishing a one-to-many relationship between classifier_user and classifier_dogprediction. Each prediction can produce multiple results, forming a one-to-many relationship between classifier_dogprediction and classifier_predictionresult.

The classifier_dogbreedinfo table functions independently, storing detailed information about various dog breeds. It does not have direct relational links with the other tables but is used by the application to present breed-related data.

3.4.5 Data Dictionary

A Data Dictionary is a collection of names, definitions, and characteristics about data components employed or documented in a database, information system, or research initiative (UC Merced Library, 2023). This document defines the meanings and objectives of data elements within a project framework, encompassing guidance on interpretation, permissible definitions, and formatting (UC Merced Library, 2023). A Data Dictionary also provides metadata concerning data elements. The information in a Data Dictionary clarifies the scope and characteristics of data items, together with the rules regulating their use and application (UC Merced Library, 2023).

3.4.5.1 Data Dictionary for User Information Table

Table 3.3 illustrates the data objects inside the User Information table. The field name, data type, and description for the data are listed down in this table.

Table 3.3: User Information Table

No.	Field Name	Data Type	Description
1	id	INT (PK)	Unique identifier for each user.
2	username	VARCHAR (50)	Username chosen by the user.

3	email	VARCHAR (50)	User's email address.
4	password	VARCHAR (225)	Hashed password for security.
5	date_joined	DATETIME	Date and time of user registration.

There are five fields in Table 3.3:

- **User ID**: This field is a BIGINT and serves as the primary key for the table, ensuring unique identification for each user.
- **Username**: This field is a VARCHAR with a maximum length of 150 characters, representing the username chosen by the user.
- **Email**: This field is a VARCHAR with a maximum length of 254 characters, storing the user's email address.
- **Password**: This field is a VARCHAR with a maximum length of 128 characters, storing the user's password in a hashed format for security purposes.
- **Date Joined**: This field is of DATETIME type, recording the date and time when the user registered on the platform.

3.4.5.2 Data Dictionary for Uploaded Images Table

Table 3.4 illustrates the data objects inside the Dog Prediction table. The field name, data type, and description for the data are listed down in this table.

Table 3.4: Dog Prediction Table

No.	Field Name	Data Type	Description
1	id	INT (PK)	Unique identifier for each prediction entry.
2	user_id	BIGINT (FK)	ID of the user who uploaded the image.
3	image	VARCHAR(100)	The uploaded image.
4	created_at	DATETIME	Timestamp of when the prediction was made.

There are five fields in Table 3.4:

- **ID**: This field is an integer and serves as the primary key for the table, uniquely identifying each prediction.
- **User ID**: A BIGINT field that acts as a foreign key, linking to the ID of the user who made the prediction.
- **Image**: A VARCHAR(100) field that stores the filename or path of the uploaded image.
- **Created At**: A DATETIME field recording the date and time when the prediction was submitted.

3.4.5.3 Data Dictionary for Prediction Results Table

Table 3.5 illustrates the data objects inside the Prediction Results table. The field name, data type, and description for the data are listed down in this table.

Table 3.5: Prediction Result Table

No.	Field Name	Data Type	Description
1	prediction_id	BIGINT (FK)	ID referencing the related prediction entry.
2	breed	VARCHAR (100)	Name of the predicted dog breed.
3	confidence	REAL	Confidence score of the prediction (e.g., similarity%).
4	is_correct	BOOLEAN	User feedback indicating if the prediction was accurate.

There are nine fields in Table 3.5:

- **Prediction ID**: A BIGINT field serving as a foreign key that links each result to a corresponding dog prediction entry.
- **Breed**: A VARCHAR(100) field representing the name of the predicted breed.
- **Confidence**: A REAL data type field showing the similarity percentage of the prediction.
- **Is Correct**: A BOOLEAN field that records user feedback on whether the prediction was accurate.

3.4.5.4 Data Dictionary for Dog Breed Information Table

Table 3.6 illustrates the data objects inside the Dog Breed Information table. The field name, data type, and description for the data are listed down in this table.

Table 3.6: Dog Breed Information Table

No.	Field Name	Data Type	Description
1	id	INT (PK)	Unique identifier for each dog breed.
2	breed_name	VARCHAR(100)	Name of the dog breed.
3	description	TEXT	General description of the breed.
4	size	VARCHAR(100)	Breed size category (e.g., Small, Medium, Large).
5	weight_range	VARCHAR(100)	Average weight range of the breed.
6	height_range	VARCHAR(100)	Average height range of the breed.
7	life_expectancy	VARCHAR(100)	Average life expectancy of the breed.

8	breed_group	VARCHAR(100)	Group classification (e.g., Working, Toy).
9	temperament	TEXT	Typical behavior and personality traits of the breed.
10	good_with_kids	BOOLEAN	Indicates if the breed is good with children.
11	good_with_pets	BOOLEAN	Indicates if the breed is good with other pets.
12	energy_level	VARCHAR(50)	Describes the energy level (e.g., Low, Medium, High).
13	trainability	VARCHAR(100)	Indicates how easy it is to train the breed.
14	grooming_needs	VARCHAR(100)	Level of grooming care required.
15	barking_level	VARCHAR(50)	Frequency or intensity of barking.
16	shedding_level	VARCHAR(50)	Amount of fur shedding.
17	common_health_issues	TEXT	Common illnesses or conditions found in the breed.

There are seventeen fields in Table 3.6:

- **ID**: This field is an integer and serves as the primary key for the table, uniquely identifying each dog breed.
- **Breed Name**: A VARCHAR(100) field that stores the name of the dog breed.
- **Description**: A TEXT field that provides a general overview or summary of the breed.
- **Size**: A VARCHAR(100) field indicating the general size category of the breed (e.g., small, medium, large).
- **Weight Range**: A VARCHAR(100) field that states the average weight range of the breed.
- **Height Range**: A VARCHAR(100) field representing the average height range of the breed.
- **Life Expectancy**: A VARCHAR(100) field indicating the average lifespan of the breed.
- **Breed Group**: A VARCHAR(100) field that categorizes the breed into groups like Sporting, Working, etc.
- **Temperament**: A TEXT field describing the typical behavioral traits and personality of the breed.
- **Good with Kids**: A BOOLEAN field indicating whether the breed is generally suitable for families with children.
- **Good with Pets**: A BOOLEAN field indicating whether the breed typically gets along well with other pets.

- **Energy Level**: A VARCHAR(50) field describing the breed's activity level (e.g., Low, Medium, High).
- **Trainability**: A VARCHAR(100) field that indicates how easy the breed is to train.
- **Grooming Needs**: A VARCHAR(100) field that specifies the level of grooming required for the breed.
- **Barking Level**: A VARCHAR(50) field that reflects how frequently or loudly the breed tends to bark.
- **Shedding Level**: A VARCHAR(50) field that describes how much the breed typically sheds.
- **Common Health Issues**: A TEXT field listing known health problems common to the breed.

3.4.6 Wire Frame

A wireframe serves as a schematic representation for a website or application. This is an initial idea of a digital interface developed prior to finalizing the optimal design aesthetic for planning user interaction with fundamental elements and navigation of a product (Khrupa, 2022).

3.4.6.1 Wireframe for Home / Landing Page

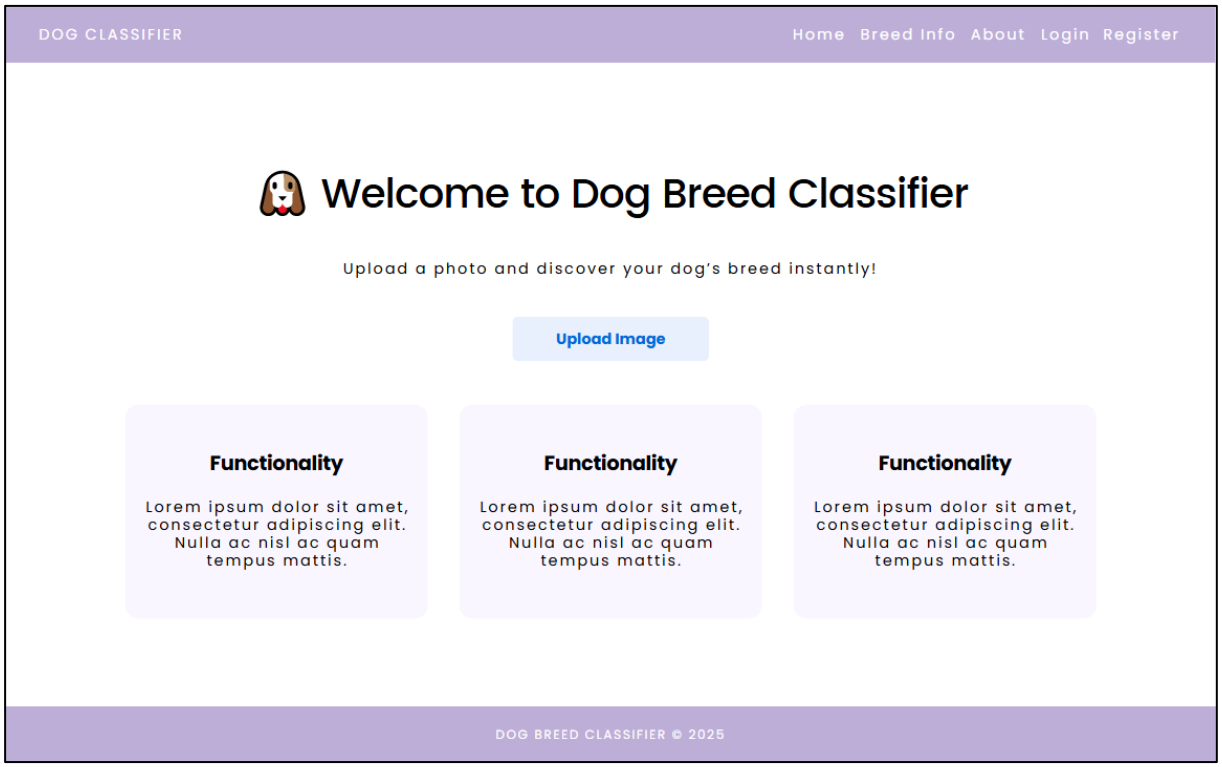


Figure 3.25: Wireframe for Home / Landing Page

Figure 3.25 displays the main landing page of the "Dog Breed Classifier" web application. The navigation bar at the top features the web application name, "DOG CLASSIFIER." The right side includes navigation links for "Home," "Breed Info," "About," "Login," and "Register," indicating comprehensive user functionalities. The central area of the page presents a clear value proposition: "Welcome to Dog Breed Classifier", followed by the inviting prompt, "Upload a photo and discover your dog's breed instantly!" An "Upload Image" button that will navigate the user to the login page. Below this, three distinct sections labeled "Functionality" are presented in cards, suggesting key features or benefits of the application.

3.4.6.2 Wireframe for Breed Info Page

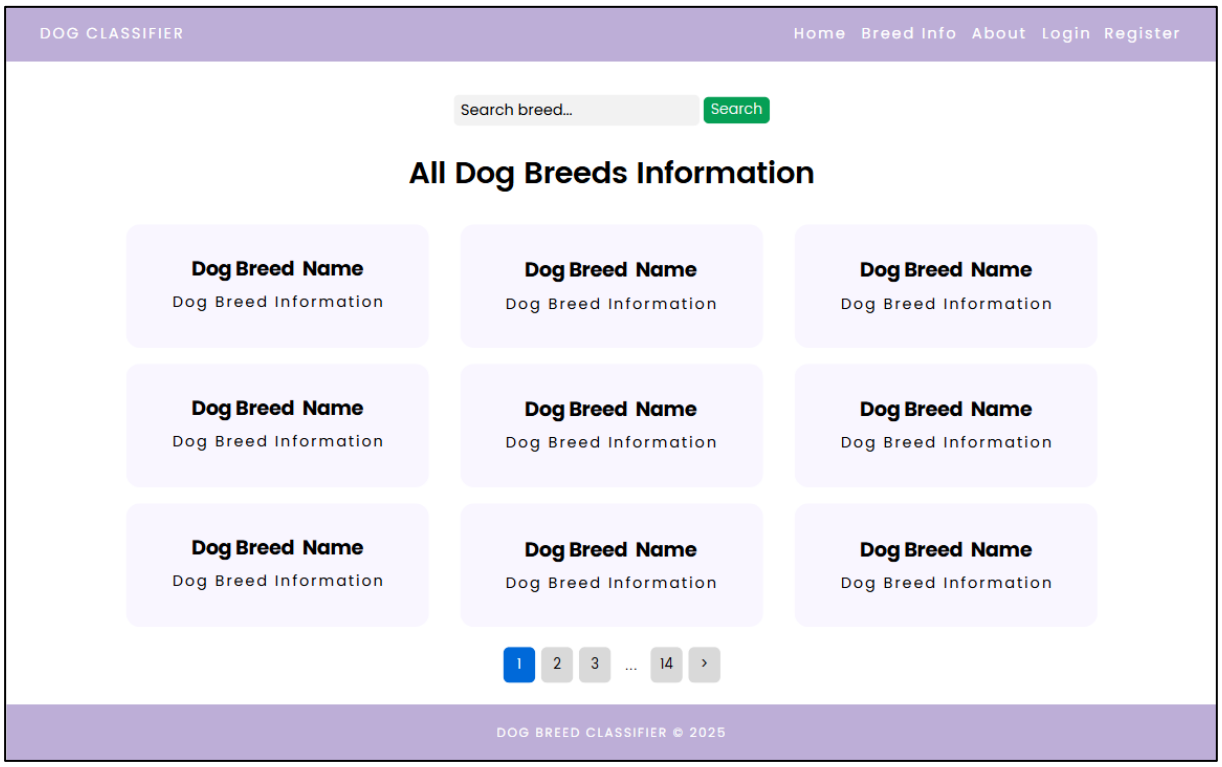


Figure 3.26: Wireframe for Breed Info Page

Figure 3.26 shows the wireframe for the Breed Info page. There are a total of 120 dog breeds included in this web application. Each page will display 9 different dog breeds with their information. If the user wants to look for a specific dog breed, a search bar is provided where they can type in the breed they would like to find. The user can navigate to other pages by using the page button provided at the bottom of the page.

3.4.6.3 Wireframe for About Page

Figure 3.27: Wireframe for About Page

Figure 3.27 shows the 'About' section of the proposed dog breed classification web application. This page will likely encompass information such as the application's purpose, key features, methodology, and any relevant acknowledgements.

3.4.6.4 Wireframe for Sign Up Page

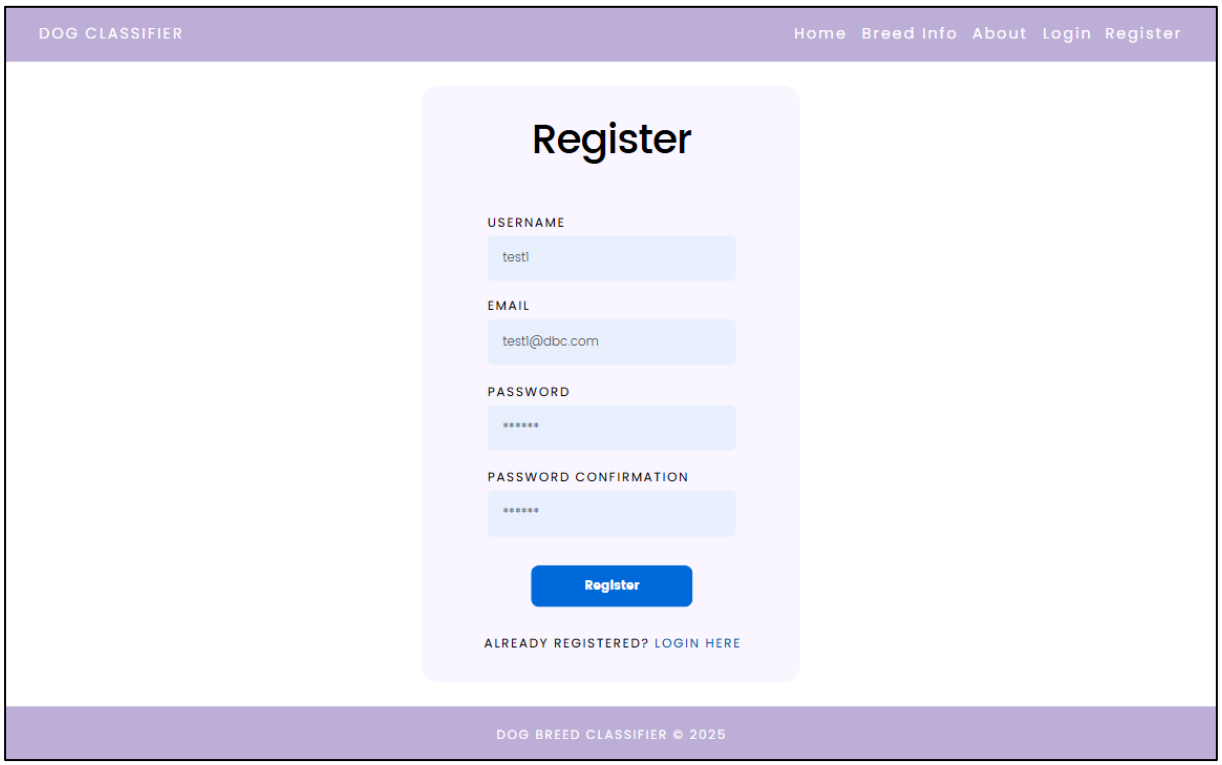


Figure 3.28: Wireframe for Sign Up Page

Figure 3.28 shows the registration page for the proposed dog breed classification web application. The central element is a rounded rectangle containing the registration form. The form includes fields for entering a username, email, password, and password confirmation with clear labels guiding the user. When the user had filled in all the required fields, they can proceed by clicking on the “Register” button. Then, the user will be redirected to the home page and proceed with using the functionality of the website.

3.4.6.5 Wireframe for Login Page

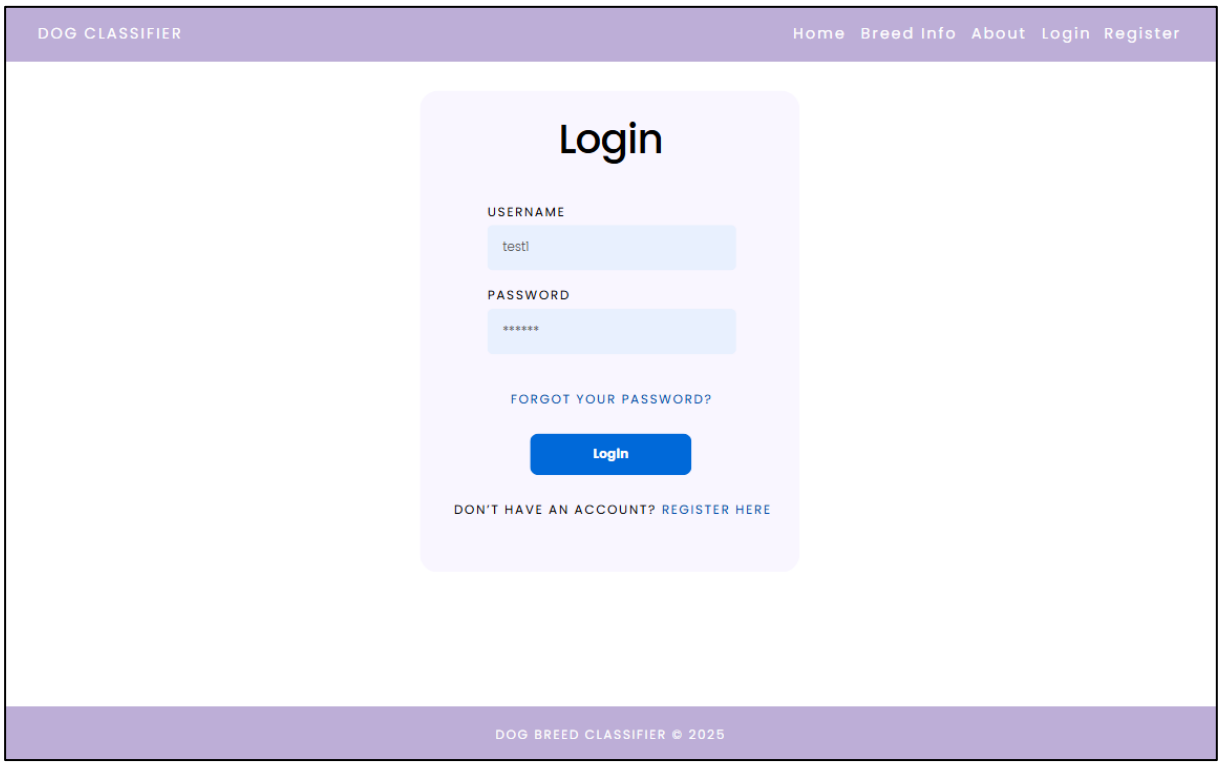


Figure 3.29: Wireframe for Login Page

Figure 3.29 shows the login page for the proposed dog breed classification web application. Whenever the user accesses the website, they will always be required to log in to access the web application. The user must complete the username and password fields to proceed. If the user does not have an account, they can proceed by clicking on the "REGISTER HERE" text under the login button. If the user forgot their password, they could click on the "FORGOT YOUR PASSWORD?" text. After the user logs in, they will be redirected to the home page.

3.4.6.6 Wireframe for Upload Page

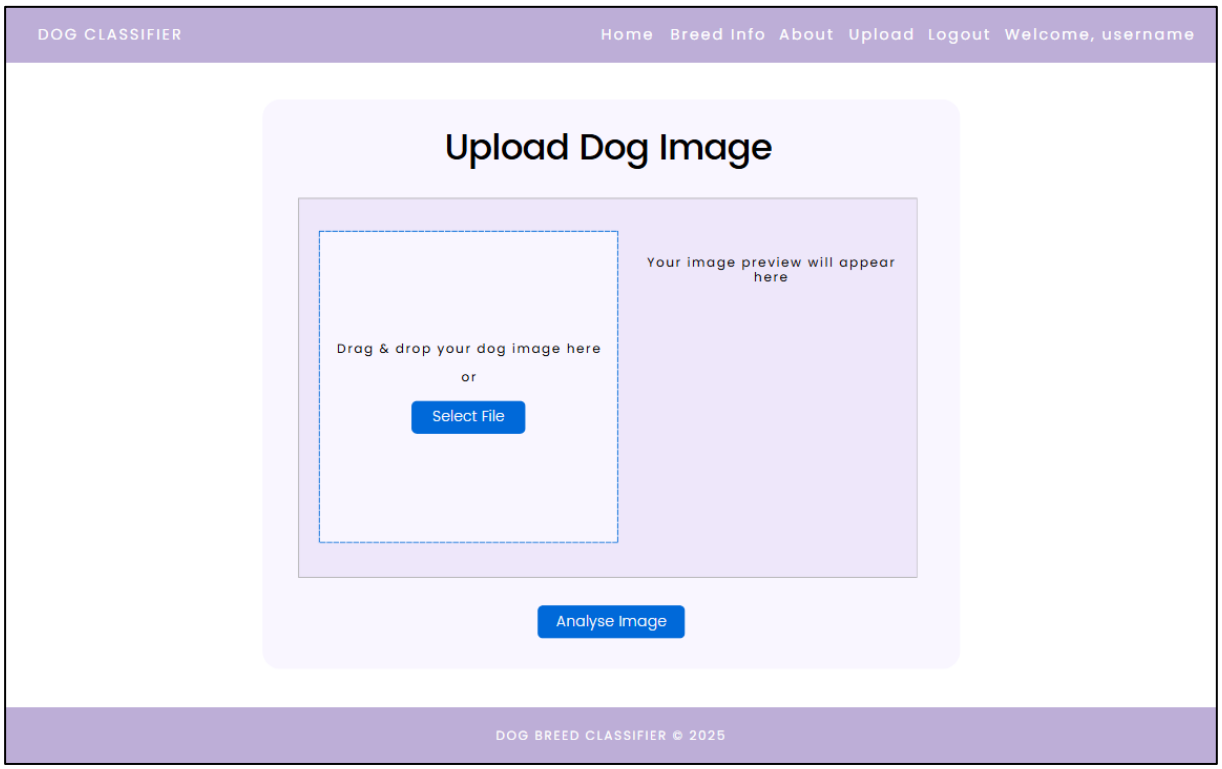


Figure 3.30: Wireframe for Upload Page

Figure 3.30 shows the upload page for the proposed Dog Breed Classification web application. Once the user has logged in to the web application, they can navigate to the upload page and upload their dog breed image. There are 2 ways to upload the image: drag and drop the image into the provided box, or click on the "Select File" button to upload the dog image. After the user has uploaded the image, it will be previewed on the right side of the upload box. Then, the user can click the "Analyse Image" button to start analyzing the uploaded image.

3.4.6.7 Wireframe for Prediction Result Page

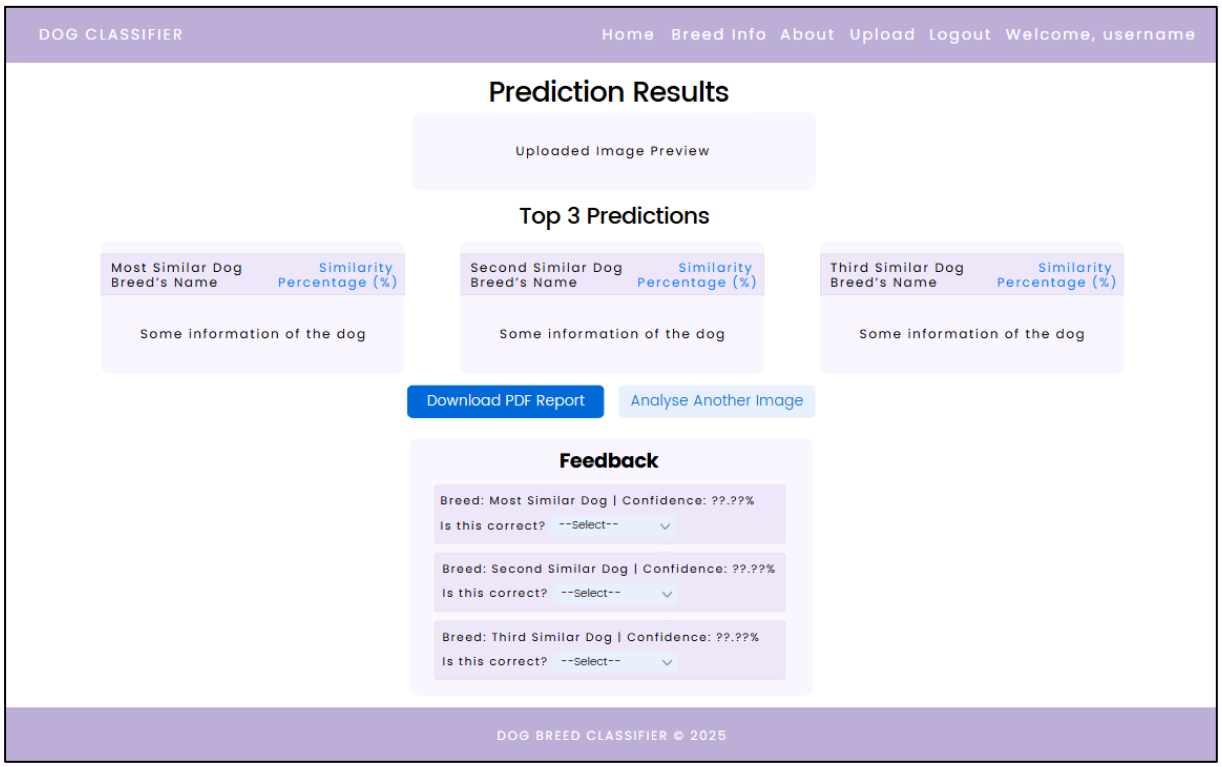


Figure 3.31: Wireframe for Prediction Result Page

Figure 3.31 shows the prediction result interface for the proposed Dog Breed Classification system. At the top, it provides a section for previewing the uploaded dog image, followed by a "Top 3 Predictions" section that shows the names of the most similar dog breeds along with their respective similarity percentages and brief breed information. Below the predictions, users can choose to either download a PDF report of the results or analyse another image. The page also includes a "Feedback" section, where users can confirm the accuracy of each predicted breed by selecting whether the result was correct, thereby helping to improve the model's performance.

3.4.6.8 Wireframe for Prediction History Page

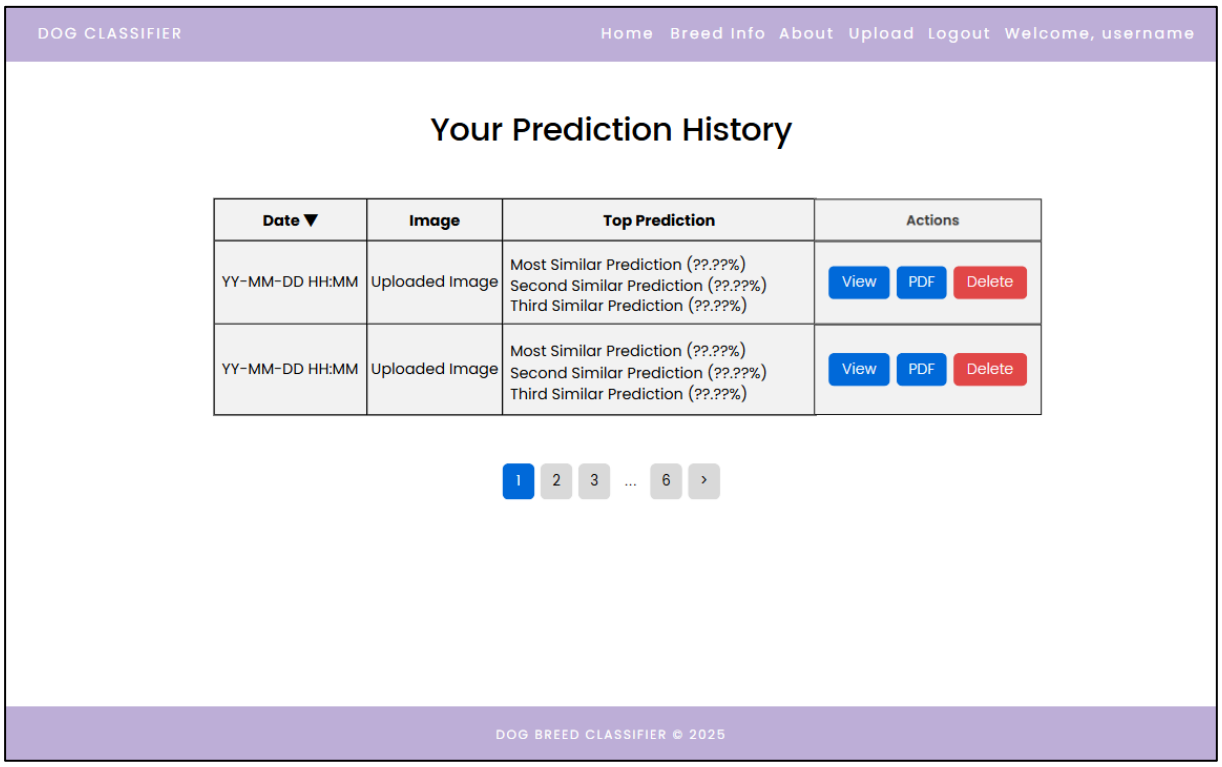


Figure 3.32: Wireframe for Prediction History Page

Figure 3.32 shows the prediction history interface for the proposed Dog Breed Classification web application, which displays a user's past dog breed classification results in a tabular format. Each row in the table shows the date and time of the prediction, a thumbnail or label of the uploaded image, and the top three predicted dog breeds along with their similarity percentages. The "Actions" column provides three buttons for each entry: "View" to revisit the prediction details, "PDF" to download the result as a report, and "Delete" to remove the entry. The page also includes pagination controls at the bottom, allowing users to easily navigate through multiple pages of prediction records.

3.5 Development

During the development phase, the web application will primarily be programmed in Python for backend operations. The development environment will be established, and the system will undergo real-time testing using local servers. The process starts with the design and coding of the screen layouts for the user interface, including image upload sections and result display sites. The subsequent integration will cover functionalities such as file uploads and the prediction process. The trained TensorFlow model will connect with the backend for real-time dog breed classification. The application will undergo testing with example inputs to verify the proper functionality of both frontend and backend components. The complete codebase will be cleaned, refactored, and adequately commented on to guarantee maintainability and clarity.

3.6 Testing

Testing represents a critical phase in the Agile iteration workflow, requiring nearly as much execution time as development. All functionalities of the dog breed classification web application must be fully operational prior to proceeding to the deployment phase. During the testing stage, the system will be verified and validated to ensure that all requirements are satisfied. According to Indeed Editorial Team (2024), there are five phases of testing. The software testing life cycle typically consists of five phases, each serving a distinct purpose in ensuring the software's quality and functionality. The phases are described in Table 3.8.

Table 3.7: Five Phases of Testing

No.	Phases	Description
1	Static Testing	During static testing, developers attempt to identify and resolve issues before they occur. During this phase, the code is not run; instead, developers either read supporting documents manually or automatically, such as requirement specifications. The goal is to identify any unclear sections, errors, or unnecessary information in these documents as soon as possible. This will make it less likely that bugs will be integrated to the software.
2	Unit Testing	Unit testing checks that different parts, functions, or procedures of software work the way they are supposed to. During this step, developers often use white-box testing to examine the code's internal structure and logic. Unit testing can be performed every time the code is changed, which helps writers identify problems quickly and resolve them. This step is typically taken before the software is sent to quality assurance teams for formal testing.
3	Integration Testing	Integration testing examines how the various components of the program interact with each other. This step helps developers find problems in how software components work together. As a result, the system functions well as a whole, rather than as separate parts. This is crucial for the program's overall functionality.
4	System Testing	When software is tested for the first time as a thoroughly combined application, this is called system testing. Independent testers who didn't work on the development carry out this step, making sure the reviews are fair and unbiased. The goal of system testing is to ensure that the application meets the client's needs and functions adequately as a whole.
5	Acceptance Testing	The final step is acceptance testing, which verifies whether the software is ready for release and use. In this step, testers collaborate with users to determine if the software meets their needs and expectations. The feedback from these people helps ensure the

		software meets the needs of users. As soon as the program passes acceptance testing, it is sent to production.

3.7 Deployment

The second-last step of each sprint is the deployment stage. At this point, the web application for classifying dog breeds is complete and ready for use. The application will be made available to users after a safe build, ready for production, is deployed to the web server. Now, the system will process real data (images shared by users) instead of dummy data used for testing.

During this phase, technical and user documentation for the web app will also be sent over. These papers will guide users on how to use the system and provide technical information that can be utilised for future maintenance or updates. Regular maintenance will also be planned to ensure the application operates smoothly and reliably. To minimise problems, users will be informed about any planned work.

3.8 Review

The sprint review marks the concluding step in the Agile iteration process for the dog breed classification web application. At this point, feedback is collected from various sources, including user interactions, internal assessments, and external evaluations. Internal assessments include insights from those who participated in gathering requirements and testing, while external assessments come from users beyond this group who interacted with the live application.

After collecting feedback, a detailed analysis is done to identify useful suggestions and improvement areas. These findings become requirements for the next sprint, helping to improve the application continuously. This stage involves reviewing the finished system to see what it

does well, what it doesn't do well, and how it performs overall. Moreover, factors like time, budget, and potential features are evaluated to plan for the next iteration, guaranteeing continuous development and enhancement of the system.

3.9 Summary

In summary, this chapter describes how the Agile-SCRUM method was used to create a web application for classifying dog breeds. User surveys indicated a desire for user-friendly features, including drag-and-drop file support and responsive design. In the design process, both logical and physical parts are involved. For instance, system architecture and data flow diagrams illustrate how the system operates, while wireframes display the appearance of key interfaces. A Python backend works with a trained deep learning model during development, using tools like TensorFlow and Kaggle Notebook. The chapter ends with a review that uses feedback to improve the application.

CHAPTER 4: IMPLEMENTATION

4.1 Introduction

This chapter details the implementation of the proposed system, building upon the requirements and design outlined in Chapter 3, as well as the software and hardware discussed earlier. The implementation occurs during the prototype development phase, which follows an iterative process involving continuous development, demonstration, and refinement. Each cycle creates and enhances a working prototype to better meet the system's requirements. This stage focuses on building and preparing the system for deployment. This chapter also emphasises the importance of both front-end and back-end development for the successful implementation of the system.

4.2 Feature Implementation

Before creating the Dog Breed Classification system, several installations and setup are needed to be done. Setting up the development environment is important to make sure the system runs properly and all parts work well together. With everything ready, it became easier to build the system, run the model correctly, and manage user data and prediction history.

This system functions as a web application. For the front end, HTML, CSS, and JavaScript work together to deliver a responsive user interface. The backend relies on Django as its main framework, managing server-side logic, user authentication, and database interactions. Central to the system is a deep learning model, specifically EfficientNetB3, which is implemented with TensorFlow utilising transfer learning to identify and categorise dog breeds from uploaded images.

The system uses SQLite to store user data and prediction history. Development took place in a Python virtual environment to keep everything organized and prevent conflicts. With

all these parts working together, the system can effectively provide important features like image uploading, breed prediction, displaying results, and tracking history.

4.2.1 Environment Setup and Tools Installation

4.2.1.1 Setup Virtual Environment and Dependency Installation

A Python virtual environment was created with the venv module to keep a clean and separate development space. This method helps manage dependencies specific to a project and prevents issues with other Python environments. After activation, key libraries such as Django, TensorFlow, NumPy, and Pillow were manually installed using pip to facilitate web development, integrate deep learning models, and perform image processing tasks required by the system. Additionally, a requirements.txt file was created to maintain a uniform setup across various machines.

4.2.1.2 Django Web Framework Setup

Django was selected as the backend framework due to its strong features and ability to enhance speed. This project created an app called Classifier. It manages important features like the dog breed classification model, user registration and login, and storing users' prediction history. Django's built-in structure made the code clear and simplified the development process.

4.2.1.3 Front-End Setup (HTML/CSS/JS) in Visual Studio Code (VS Code)

The front-end was created using HTML for page structure, CSS for styling, and JavaScript for providing interactive elements. The online pages are built as templates, allowing users to upload dog photographs, check prediction results, and review previous predictions result.

4.2.1.4 Database Setup and Configuration

SQLite was used as the database during development since it integrates well with Django and is lightweight. Database models were created using Django's Object-Relational Mapping (ORM) system to store important data, including user profiles, uploaded images, prediction results, and timestamps for each activity. Database migrations were performed to build the actual database tables from these models.

4.3 System Hierarchy

Figure 4.1 below shows the system hierarchy of the proposed web application.

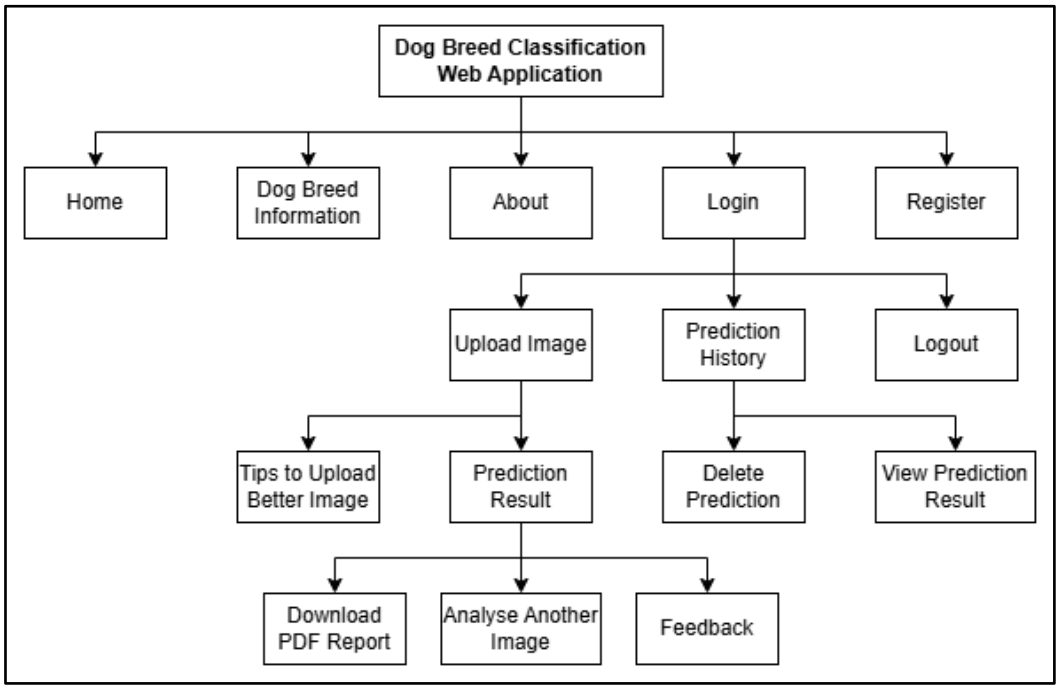


Figure 4.1: System Hierarchy of the Proposed Web Application

According to Figure 4.1 above, at the top level, the main application serves as the central platform that integrates all functionalities. The primary navigation options available to users include **Home, Dog Breed Information, About, Login, and Register**. The Home page acts as the landing page, while the Dog Breed Information section provides users with general

knowledge about various dog breeds. The About page explains the application's purpose and its core features.

Registered users can log in to access personalised features. Upon logging in, users are given access to the **Upload Image** functionality, which allows them to upload an image of a dog for breed classification. In the top navigation bar, there is a section called "Upload Tips" that provides tips on how to take or select high-quality photos. After uploading a picture, the Prediction Result screen appears, showing the breed that was detected along with extra choices like "**Download PDF Report**" and "**Analyse Other Image**." On the Prediction Result page, a feedback form is available for users to complete, allowing them to indicate whether the predictions for each dog breed were accurate.

Additionally, users can check past prediction outcomes or remove prediction records they no longer need by visiting their Prediction History. A "**Logout**" function is also provided to securely end the session.

4.4 Integration of Deep Learning Model

This section describes how to integrate a trained deep learning model into a web application built using Django. It explains the steps used to train the model, save it in the correct format, and integrate it into the backend code so that it can make predictions.

4.4.1 EfficientNetB3 Transfer Learning

This project's deep learning model is based on EfficientNetB3, a cutting-edge convolutional neural network design that is known for striking a good balance between accuracy and speed. EfficientNetB3 was chosen because it excels at classifying images while remaining compact enough to be utilized in web-based applications.

For training, the Kaggle Dog Breed Identification dataset was used. This dataset has thousands of labelled pictures of 120 different dog types. Kaggle Notebook was used for the training. It is a cloud-based system that allows users to use GPUs for free and is already integrated with Kaggle datasets. These are the main things that were done during the training:

- **Data Preprocessing:** All of the pictures that were sent were resized to 300x300 pixels so that they would fit the size requirements of EfficientNetB3.
- **Data Augmentation:** To get better generalisation and less overfitting, methods like random flipping, rotation, zooming, and brightness changes were used.
- **Model Architecture:** The base layers of EfficientNetB3 were loaded with ImageNet weights that had already been trained and then frozen. This enabled the model to function as a fixed feature extractor. A custom classification head was then added to map extracted features to the corresponding dog breeds.
- **Compilation and Training:** The model was compiled using the Adam optimiser with categorical cross-entropy as the loss function. Accuracy was used as the primary evaluation metric during training.

Fine-tuning of the EfficientNetB3 base is not applied, because the prediction accuracy is already quite good at around 91.54%. Therefore, the base was frozen to preserve the pre-trained weights and maintain stability during training. The trained model showed reasonable accuracy for the task and was saved for integration into the Django web application.

4.4.2 Model Conversion for Web Use

After the training and testing of the model was done, the next step was to turn and get the model ready to be added to the Django web application. It was decided to save the model

in the keras format because it is lightweight, works better with current TensorFlow workflows, and is easy to load in production settings.

Checkpointing was used to make sure that during training, only the best-performing model was saved. During training, the validation accuracy was checked, and the model was only saved when a significant improvement was observed. This method ensured the best performance while avoiding overwriting models that were already better. To save the model, the following important steps were taken:

- **Model Saving Strategy:** The Keras ModelCheckpoint callback was used to save the model only when the validation accuracy improved on its own. The following line was used to save the model in the Keras format:

```
checkpoint = ModelCheckpoint(
            'best_model.keras',
            monitor='val_accuracy',
            save_best_only=True,
            mode='max',
            verbose=1)
```

After training, the best version of the model was automatically saved as 'best_model.keras' in the Kaggle environment. This file was then downloaded and renamed to "dog_breed_classifier.keras," and it was placed in the Django project's directory (e.g., inside the classifier app folder) to be accessible by the backend.

4.4.3 Prediction Logic in Django View

The core prediction functionality was implemented in the Django view. When a user uploads an image via the web interface, the backend performs the following:

1. **Image Preprocessing:** The uploaded image is resized and converted to an array format that matches the model's expected input.

2. **Prediction:** The pre-processed image is passed to the loaded model to predict the dog breed.
3. **Result Display:** The top three predicted breeds, based on probability scores, are extracted and displayed to the user along with their corresponding confidence percentages.

The predicted results are also stored in the database for future viewing, contributing to the history feature located at the navigation bar.

4.5 Proposed Web Application Implementation

This section talks about the most important features of the web program. It starts with the user onboarding process, where new users can sign up for an account. Once the user has registered, they are redirected to the home page. From the home page, users can click the "Upload" button to navigate to the upload page. The interface allows users to upload an image of a dog for analysis. The web application utilises a deep learning model to classify breeds based on images and then displays the most likely breeds along with their corresponding confidence scores. Users can view information about each predicted breed to gain a deeper understanding of the findings. The web application also keeps track of images that have already been submitted, allowing users to review their past submissions.

4.5.1 Launching the Web Application

Figure 4.2 below illustrates the process for launching the web application.

```
PS C:\Users\annas\Desktop\FYP_78855> myenv\Scripts\activate
(myenv) PS C:\Users\annas\Desktop\FYP_78855> cd dog_breed_classification
(myenv) PS C:\Users\annas\Desktop\FYP_78855\dog_breed_classification> python manage.py runserver
System check identified no issues (0 silenced).
May 17, 2025 - 10:47:43
Django version 5.1.6, using settings 'dog_breed_classification.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

Figure 4.2: Launching the Web Application in VS Code Terminal

First, the user should set up a virtual environment to handle the necessary software libraries for this project, so that it doesn't interfere with other Python projects on their computer. Next, the user needs to navigate to the folder that contains the core code for the web application. Finally, the user must execute the command `python manage.py runserver`. This command starts a local development web server, allowing the user to open a web browser on their computer and navigate to http://127.0.0.1:8000/ to access the web application.

4.5.2 User Onboarding

The user onboarding process is designed to provide a smooth entry point into the system, allowing users to create an account, securely log in, and manage their session. This section outlines the core features that enable users to begin interacting with the application.

4.5.2.1 Home Page

The Home Page serves as the main interface of the web application, where users can access the core functionalities. After logging in, users are greeted with a clean and user-friendly layout that allows them to upload dog images for breed prediction.

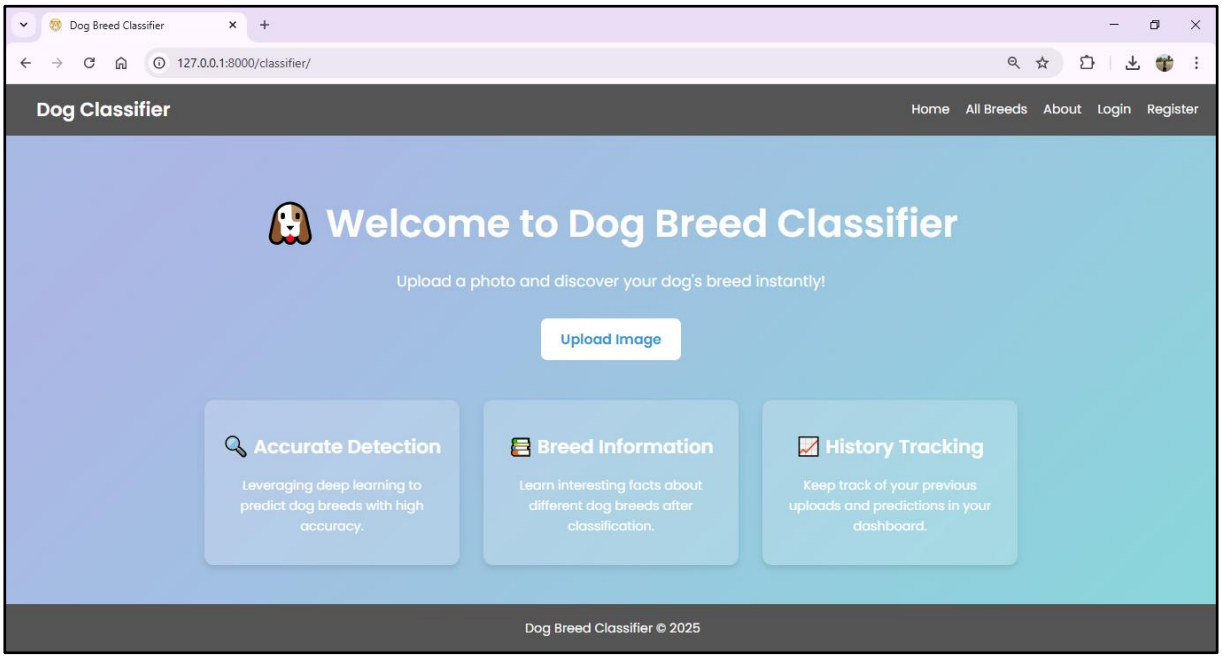


Figure 4.3: Home Page (Haven't logged in)

Figure 4.3 shows the interface when the user initially accesses the web application. The navigation bar is displayed at the top, consisting of links to the Home, All Breeds, About, Login, and Register pages. This home page showcases the brief functionality provided by this web application.

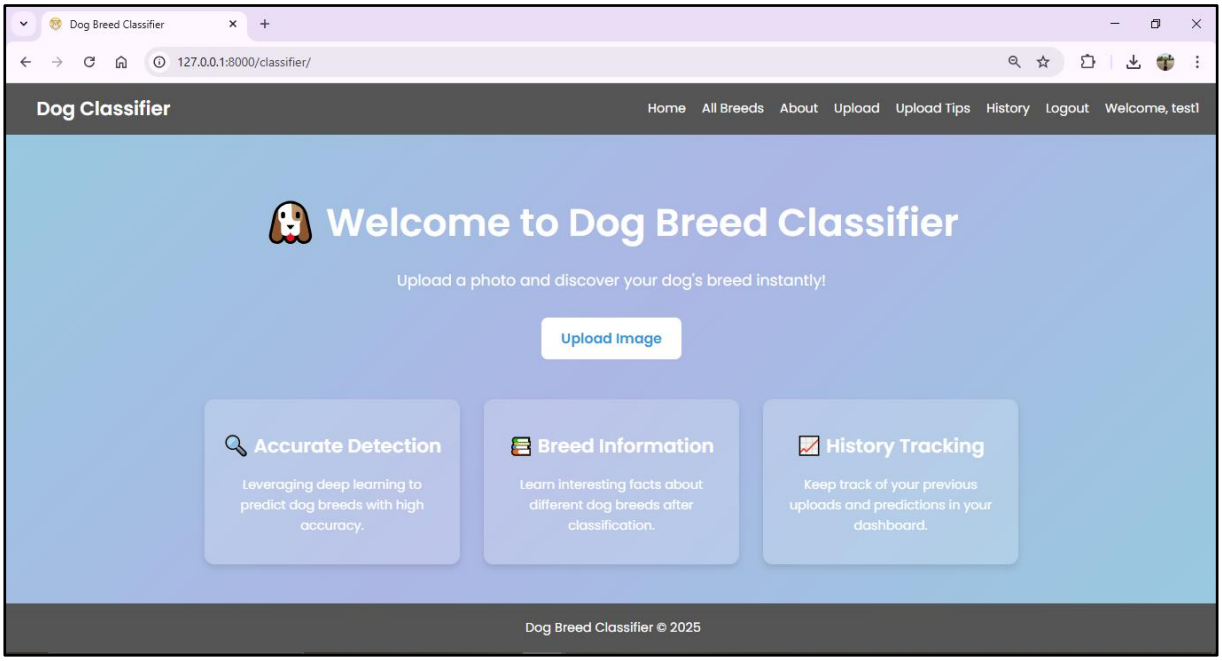


Figure 4.4: Home Page (Logged in)

Figure 4.4 illustrates the interface when the user has successfully registered or logged in to the web application. The navigation bar displayed at the top is slightly different from what the user had seen before they logged in. After the user has logged in, the navigation bar now features additional columns, including "**Upload**", "**Upload Tips**", "**History**", and a simple welcome message to indicate which user is accessing the webpage.

4.5.2.2 All Dog Breeds Information Page

This page contains all the dog breed information included in the dataset used. The dog breed information is sourced from the American Kennel Club, Purina Malaysia, and PetMD. The content of the information consists of description, group, size, weight, height, breed group, temperament, whether the dog is suitable with children or other pets, trainability, grooming needs, barking level, shedding level, and common health issues.

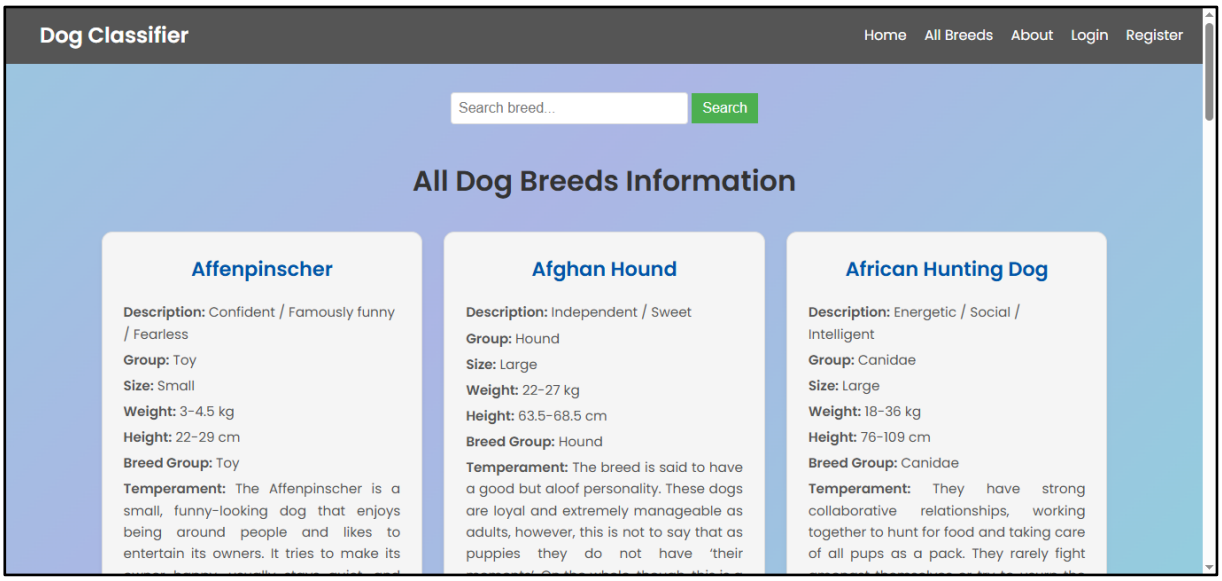


Figure 4.5: All Dog Breeds Information Page

Figure 4.5 above shows all the information about dog breeds to the user. If the user wants to search for specific dog breeds, they can use the provided search bar to find the desired breeds.

4.5.2.3 About Page

Figure 4.6 illustrates the About Page, which shows an overview of the Dog Breed Classification web application, including its purpose, operation, and the technologies used in its development. It explains the project's objective, highlights the use of deep learning (EfficientNetB3) and the Django framework, and informs users about the system's capabilities

and limitations. This page aims to help users understand the background and value of the application.

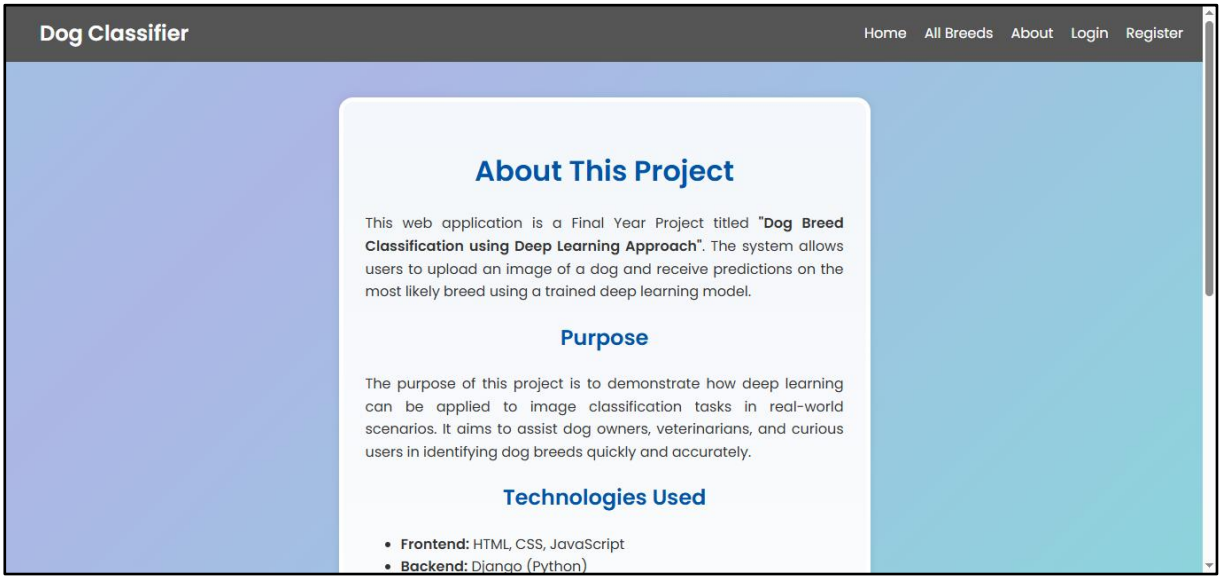


Figure 4.6: About Page

4.5.2.4 User Registration Page

The registration page enables new users to create an account by providing their username, email address, and password details. Upon successful submission, the user's data is stored securely in the database, and the system redirects the user to the home page. Alternatively, if the user already had an account, they could click "**Login here**" to navigate to the login page.

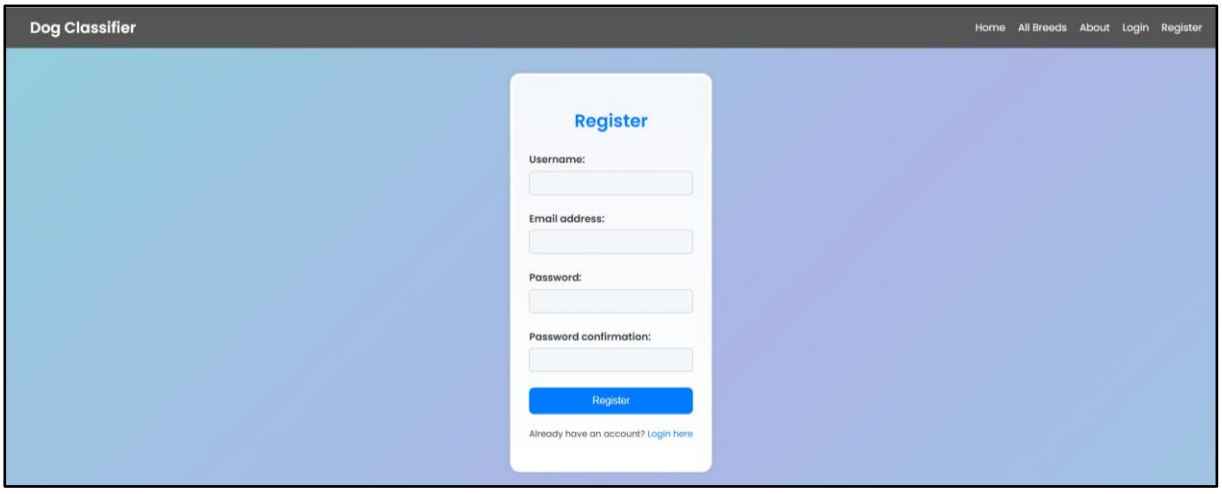


Figure 4.7: Register Page

The interface of the user registration page is shown in Figure 4.7. A validation message will be shown to the user if the inserted credentials do not meet the requirements. The validation messages are as shown in Figure 4.8.

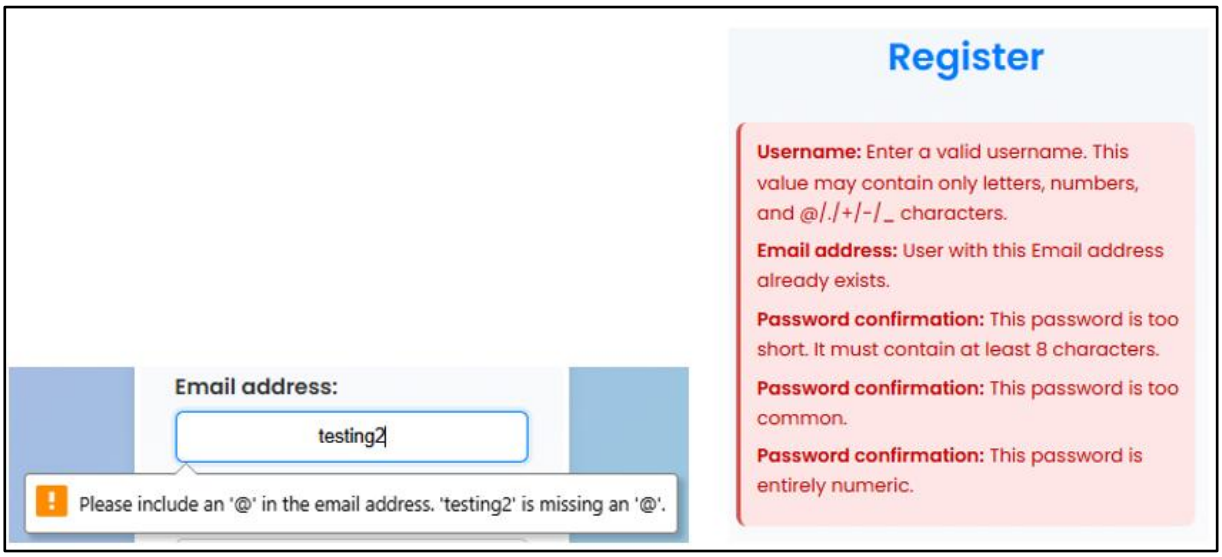


Figure 4.8: Register Page Validation

4.5.2.5 Login Page

The login page lets existing users access their accounts using their registered credentials. The authentication system verifies the submitted username and password, granting access to the dashboard if the credentials are valid, or displaying an error message if not.

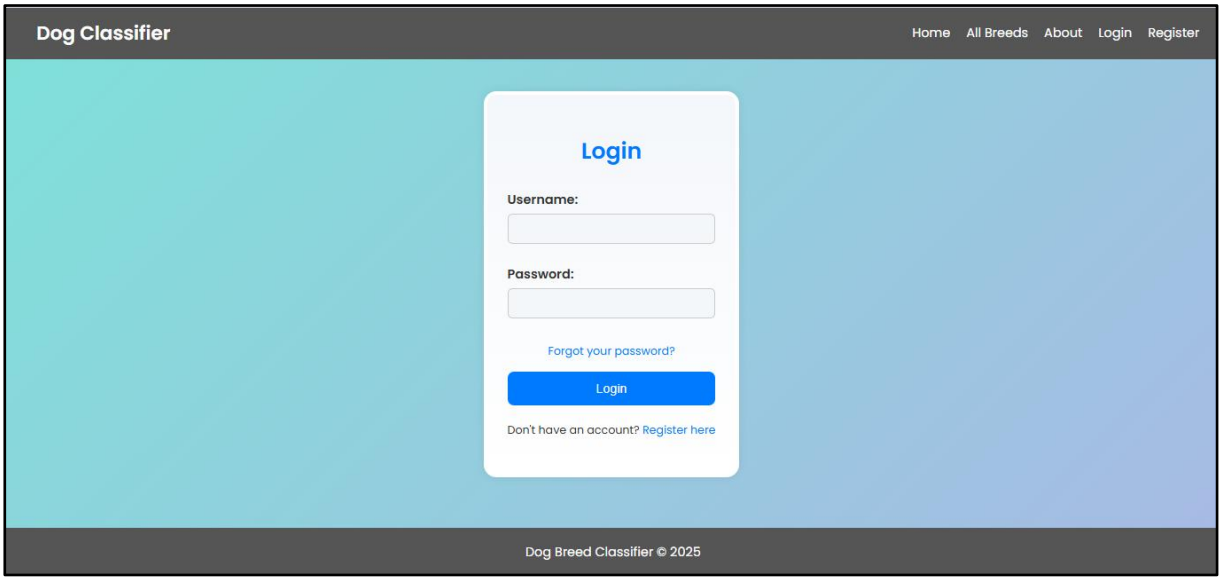


Figure 4.9: Login Page

Figure 4.9 shows the login page. The user must enter the correct username and password to log in and access the features of this web application. If the user does not have an account yet, they can click the "Register here" link to create one. The "Forgot your password?" choice is also provided to the user.

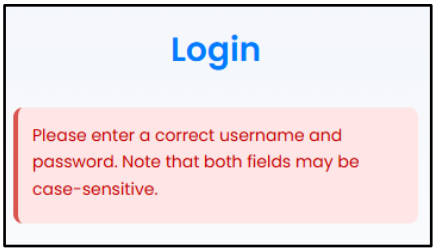


Figure 4.10: Login Page Validation

Figure 4.10 illustrates the validation process for the user to ensure they have entered the correct username and password.

Figure 4.11: Forget Password

Figure 4.11 shows the forgot password page to the user. If the user forgets their password, they can contact the administrator or register a new account to continue using the web application.

4.5.2.6 User Profile

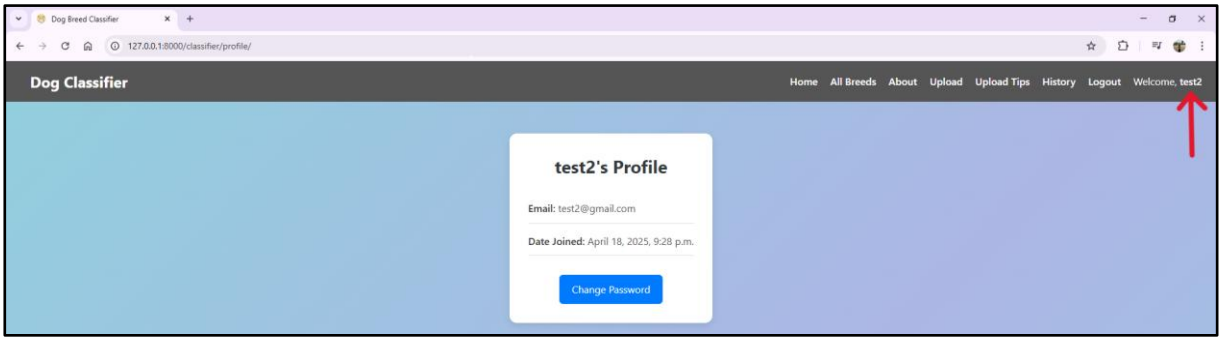


Figure 4.12: User Profile

Figure 4.12 above shows the user profile. The user can navigate to their profile by clicking on the username in the navigation bar at the top. This page will display basic information, including their email address and the date they joined. Users can also change their password by clicking the "Change Password" button.

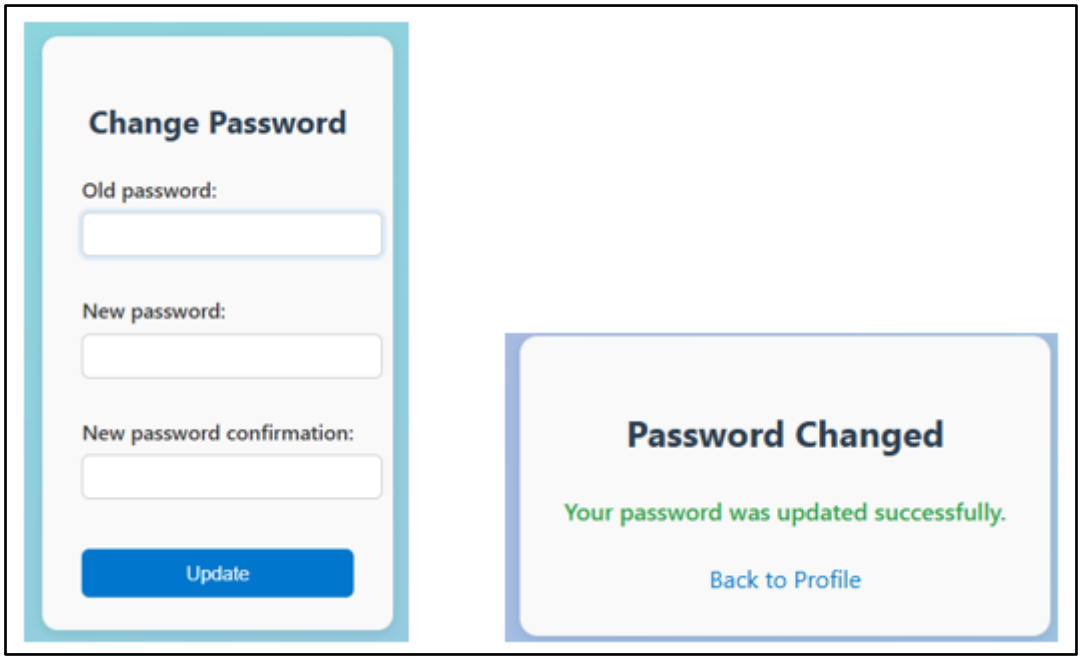


Figure 4.13: Change Password

Figure 4.13 shows the change password section and a message indicating a successful password change. The user must fill in their old password to create a new one. Based on the validations triggered by the user, the user will be shown validations.

4.5.2.7 Logout Functionality

The logout feature ensures secure session management, allowing users to end their session at any time.

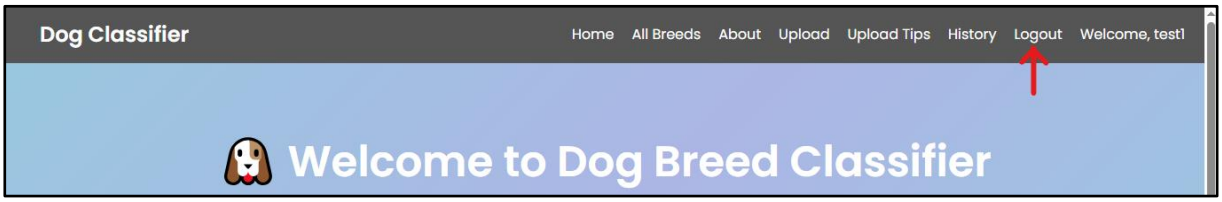


Figure 4.14: Register Page Validation

Figure 4.14 illustrates the Logout function, located in the navigation bar at the top. When users log out, the system clears their session data. It redirects them to the login page, protecting user information and maintaining privacy.

4.5.3 Dog Breed Prediction Module

This module enables users to upload dog images, process them through a deep learning model, and view the predicted breed along with its top 3 confidence scores. It also includes features to validate uploads, provide tips for better image inputs, and download results as PDFs for recordkeeping.

4.5.3.1 Upload Image Page

The Upload Dog Image Page allows users to submit a dog image for prediction. The interface includes a file input field that accepts .jpg, .jpeg, and .png formats. Users are encouraged to upload clear, front-facing photos of their dogs to improve prediction accuracy.

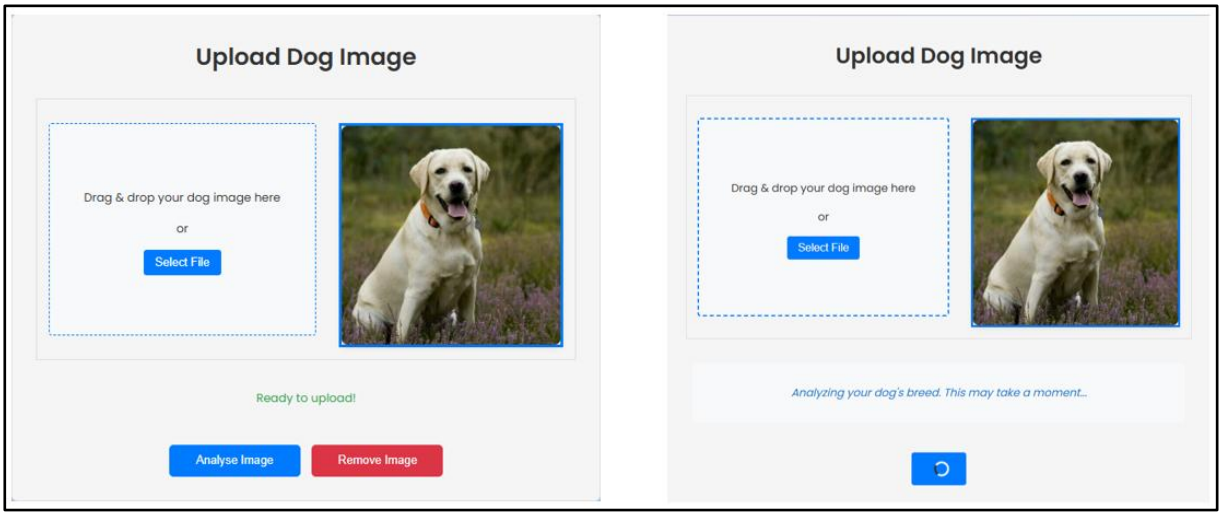


Figure 4.15: Upload Page

Figure 4.15 illustrates the user interface for uploading an image. The design is minimal and user-friendly, ensuring ease of use. The user can drag and drop their image into the container provided on the left side or click on the "**Select File**" button to select the image they want to classify. After the user uploads their image, it will be shown on the right side of the upload image container. Then, the user can click "Remove Image" if they want to change to another image, or they can drag and drop a new image, or click "Select File" to select an

alternative image. Suppose the user had decided which image to classify. In that case, the user can click on the "**Analyse Image**" button, and the "**Analysing your dog's breed. This may take a moment...**" will be shown to the user. After the classification is done, the user will be able to navigate to the result page.

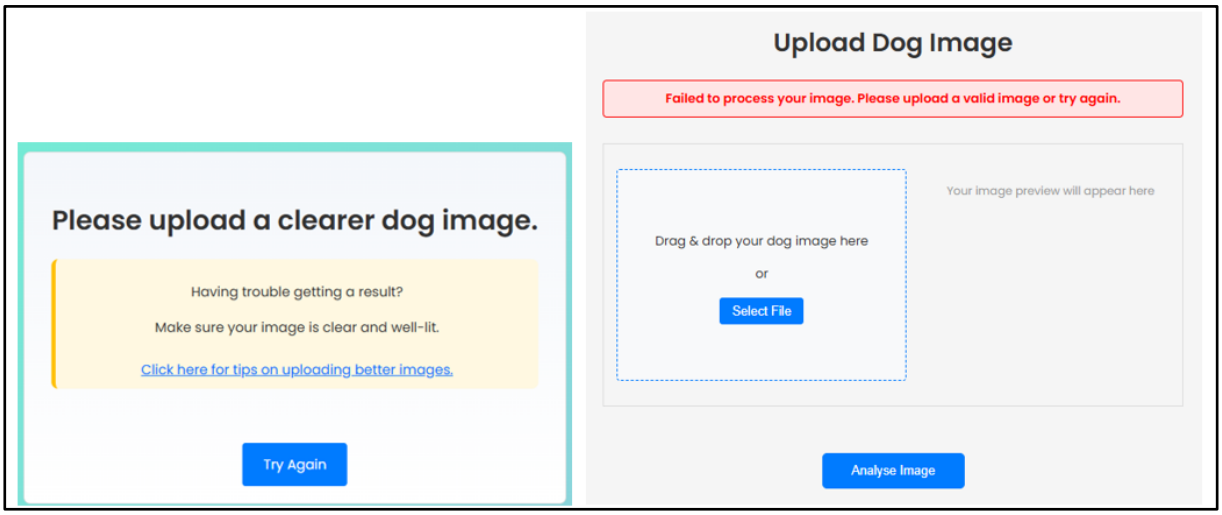


Figure 4.16: Uploaded Image Validation

Based on Figure 4.16, if the user uploads an unclear image, a notice will be displayed to the user. The user can either try again or look for tips on uploading a better photo.

4.5.3.2 Prediction Result Page

Once the image passes validation, it is sent to the server, where the deep learning model predicts the top 3 likely dog breeds. The Prediction Result Page displays the uploaded image, predicted breed names, respective confidence percentages, and user feedback.

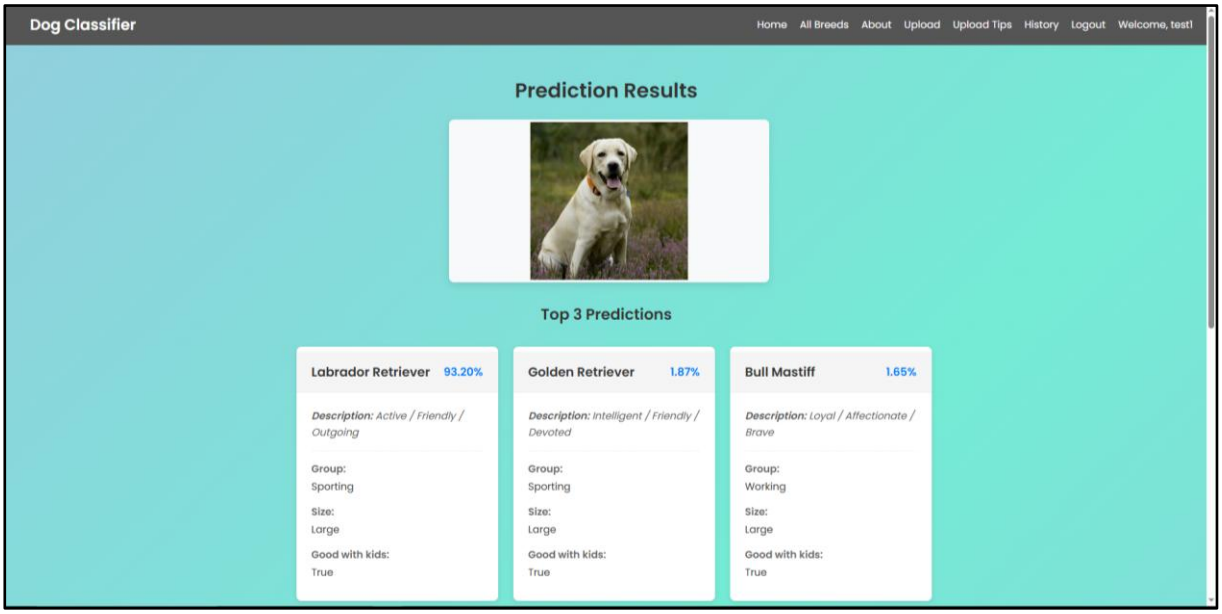


Figure 4.17: Prediction Result Page

Figure 4.17 above illustrates the prediction results of the uploaded dog breed image. It will show the top three predicted dog breeds and the similarity percentage to the user. It will also show some information about the dog breed.

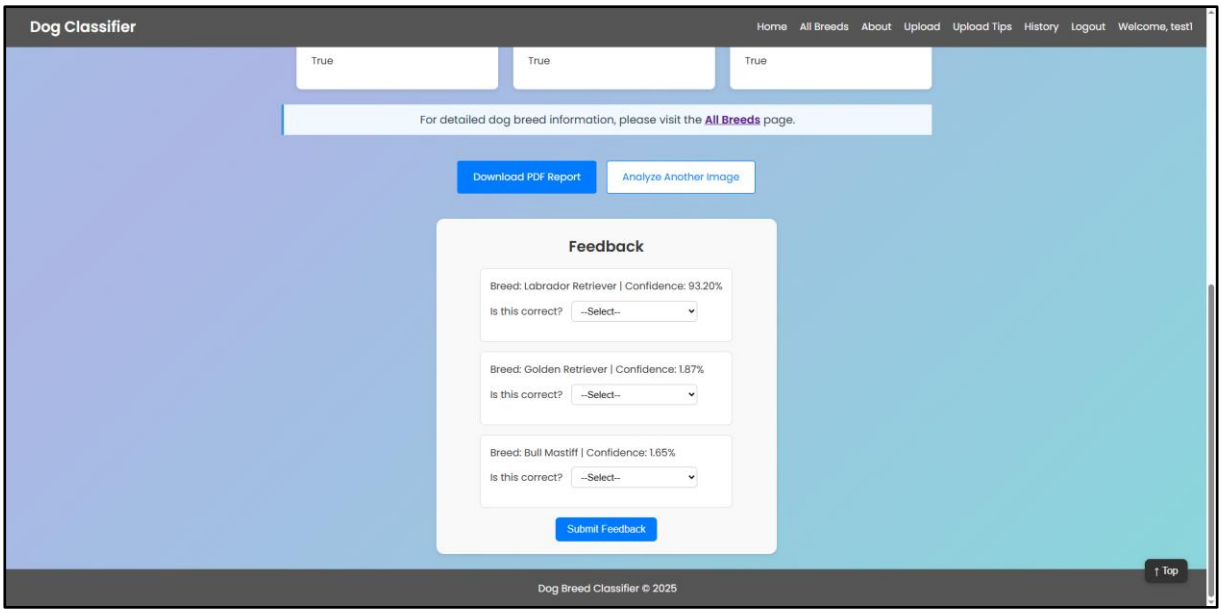


Figure 4.18: Prediction Result Feedback

Figure 4.18 above shows the user feedback located at the lower part of the prediction result page.

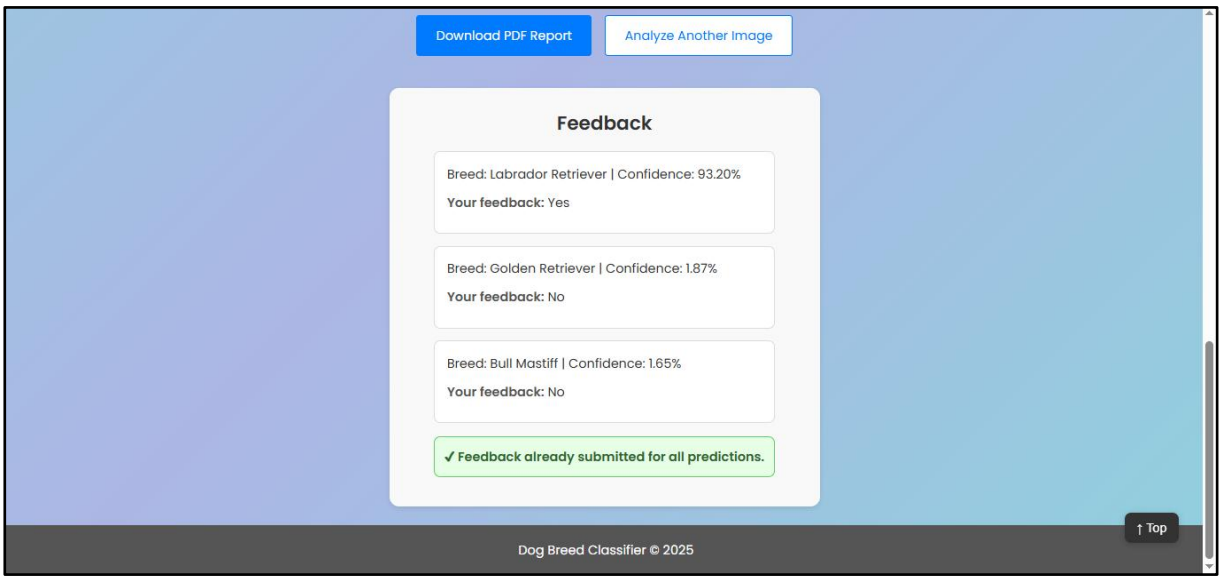


Figure 4.19: Result Feedback Form Filled

Figure 4.19 illustrates the feedback form that the user completed. It will tell the user that the feedback for all the predictions has already been submitted.

4.5.3.3 Tips to Upload Better Images Page

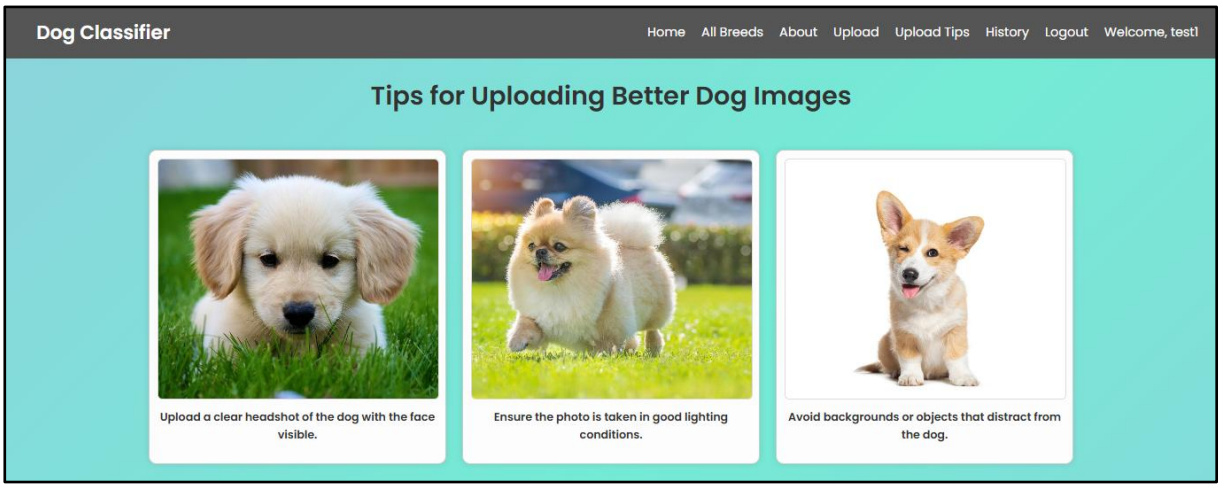


Figure 4.20: Tips to Upload Better Images Page

This page provides guidelines for uploading images that result in more accurate predictions. Figure 4.20 illustrates three tips for uploading a clearer picture. The first one is to upload a clearer headshot of the dog with its face visible. Second, ensure the photo is taken in good lighting conditions. The third one is avoiding background or objects that distract the dog. The figure provides helpful instructions designed to enhance user submissions.

4.5.3.4 Result in PDF

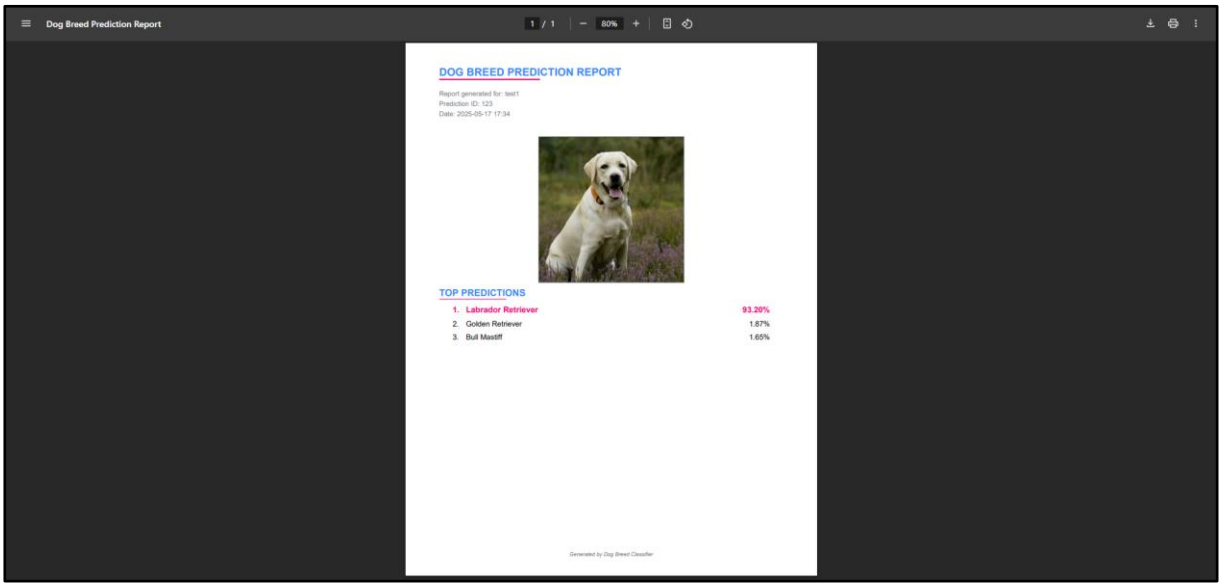


Figure 4.21: Prediction Result PDF

The web application allows users to download the prediction results in PDF format. This includes the prediction ID, image upload date, uploaded image, the top 3 prediction results, and confidence scores. Figure 4.21 displays the layout of the exported PDF document, which contains all relevant information.

4.5.4 Prediction History Module

This section lets users view past predictions in a structured and accessible format. Each entry includes a timestamp, the image uploaded, and the associated prediction results. The module enhances the user experience by allowing users to track their prediction activities and access past results at any time.

4.5.4.1 History Page

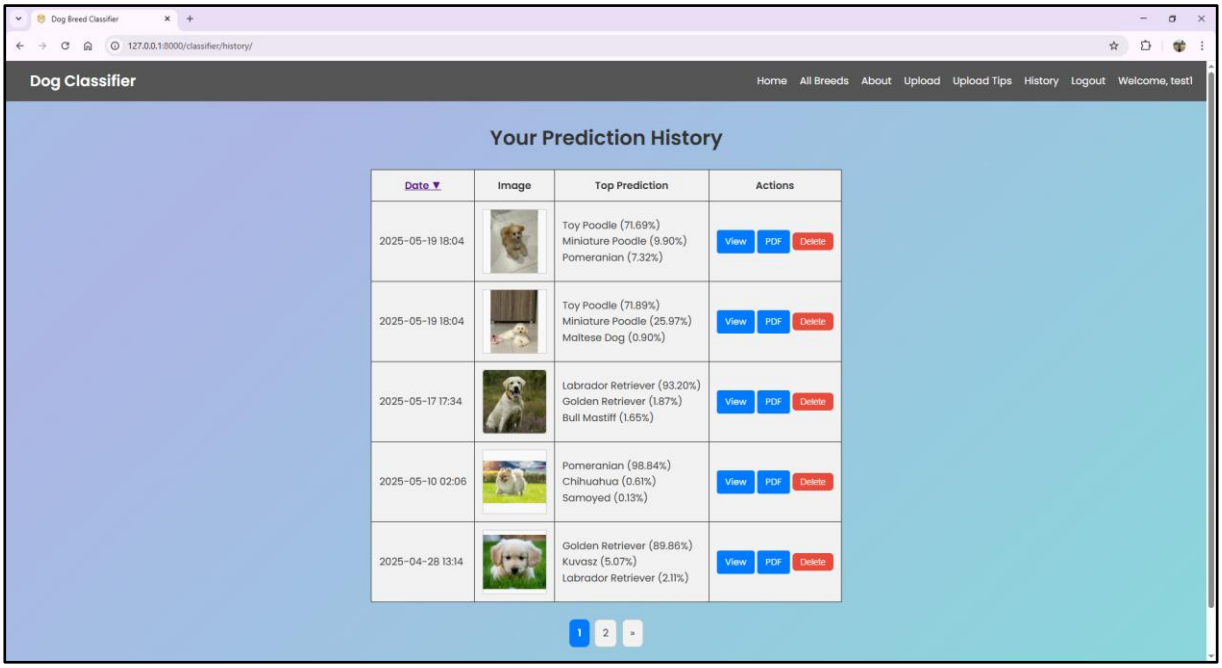


Figure 4.22: History Page

Figure 4.22, the history page, displays all predictions made by the logged-in user in reverse chronological order. Each record shows the uploaded image, the date and time of the prediction, and the top predicted dog breed. A "**View Result**" button is available for each entry, allowing users to access the full details of the prediction again. This functionality ensures that users can easily find and revisit their previous uploads without having to repeat the process.

The "**View Result**" button redirects the user to the same detailed result page shown during the initial prediction.

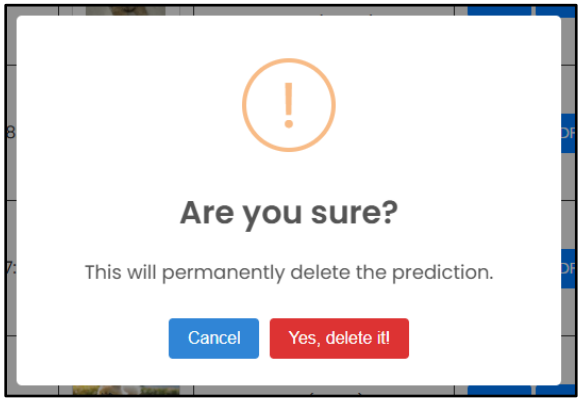


Figure 4.23: Delete History Record Confirmation Pop-up

Users are also given control over their data through the ability to delete history records. When a user chooses to delete an entry, a confirmation pop-up is shown to prevent accidental deletion. This adds a layer of safety while giving users flexibility in managing their data. Figure 4.23 above shows the pop-up message that will be displayed to the user.

4.5.5 Admin Panel

The Admin Panel is a key system feature that enables administrative users to manage and monitor the platform's data effectively. Django's powerful built-in admin interface provides access to all major system components, including user data, uploaded dog images, prediction results, and detailed information about dog breeds. This panel ensures that system maintenance, data updates, and user management can be handled in a secure and user-friendly environment.

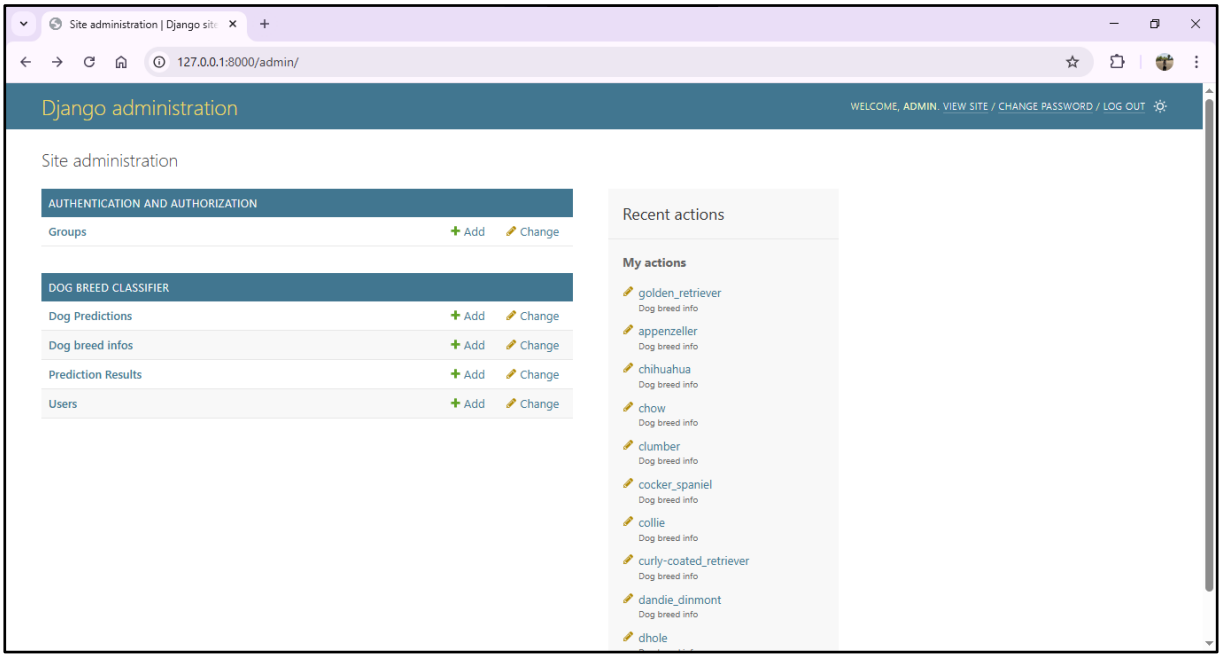


Figure 4.24: Admin Dashboard

Figure 4.24 shows the existing table for the web application. This page also displays the admin's recent actions.

4.5.5.1 Users Data

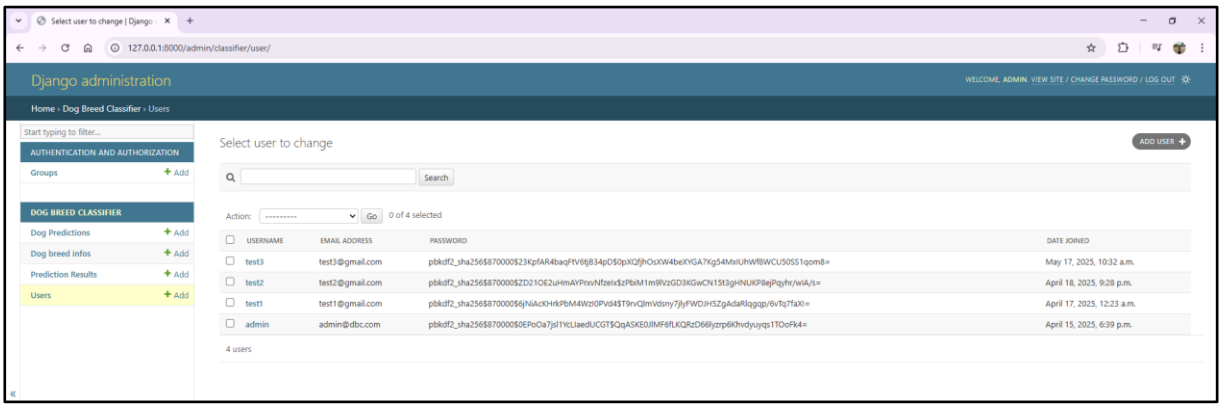


Figure 4.25: User Data Table

Figure 4.25 shows the Users Data table within the admin panel, which allows administrators to view, edit, and manage information about registered users. While usernames and email addresses are stored in plaintext for display purposes, Django securely encrypts user

passwords using a robust hashing algorithm (PBKDF2 by default) to ensure account security. This means that even administrators cannot view user passwords directly, as they are stored in a hashed and salted format to prevent unauthorised access.

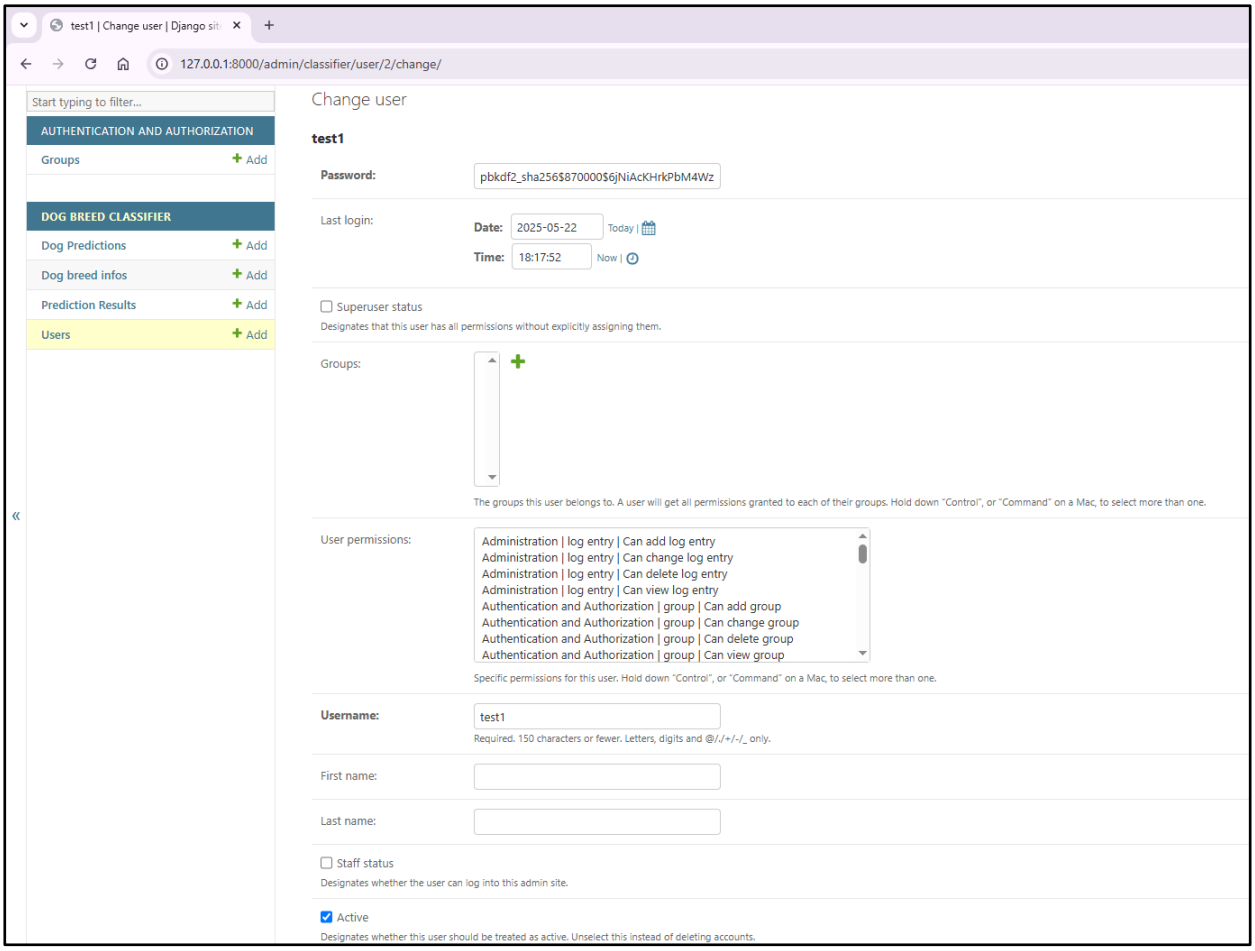


Figure 4.26: Edit User Table

Figure 4.26 illustrates the content of the user table, where administrators can modify user credentials, such as usernames and email addresses, or remove users from the system as necessary.

4.5.5.2 Dog Prediction Table

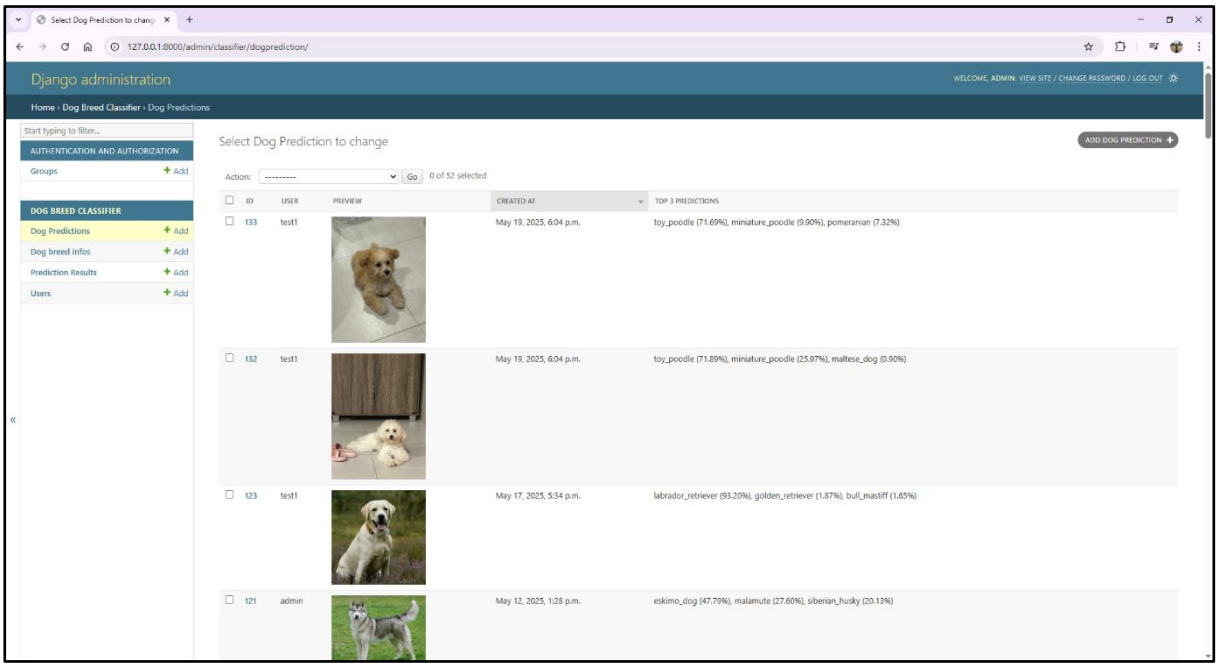


Figure 4.27: Dog Prediction Table

Figure 4.27 displays the Dog Prediction Table, which shows all images that users submit for breed prediction. Each entry includes the username of the user, the image, and the date and time of upload.

4.5.5.3 Prediction Result Table

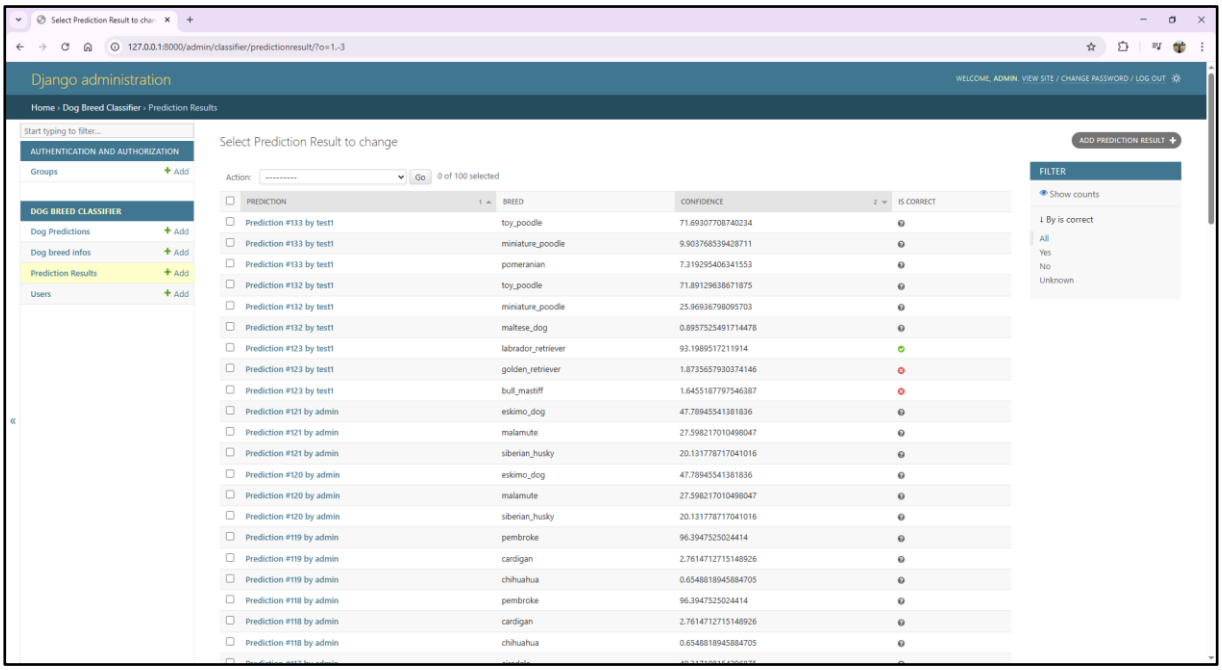


Figure 4.28: Prediction Result Table

Figure 4.28 displays the Prediction Result Table, which contains the output from the prediction model. It lists each prediction's breed, confidence score, and user feedback on the accuracy.

4.5.5.4 Dog Breed Info Table

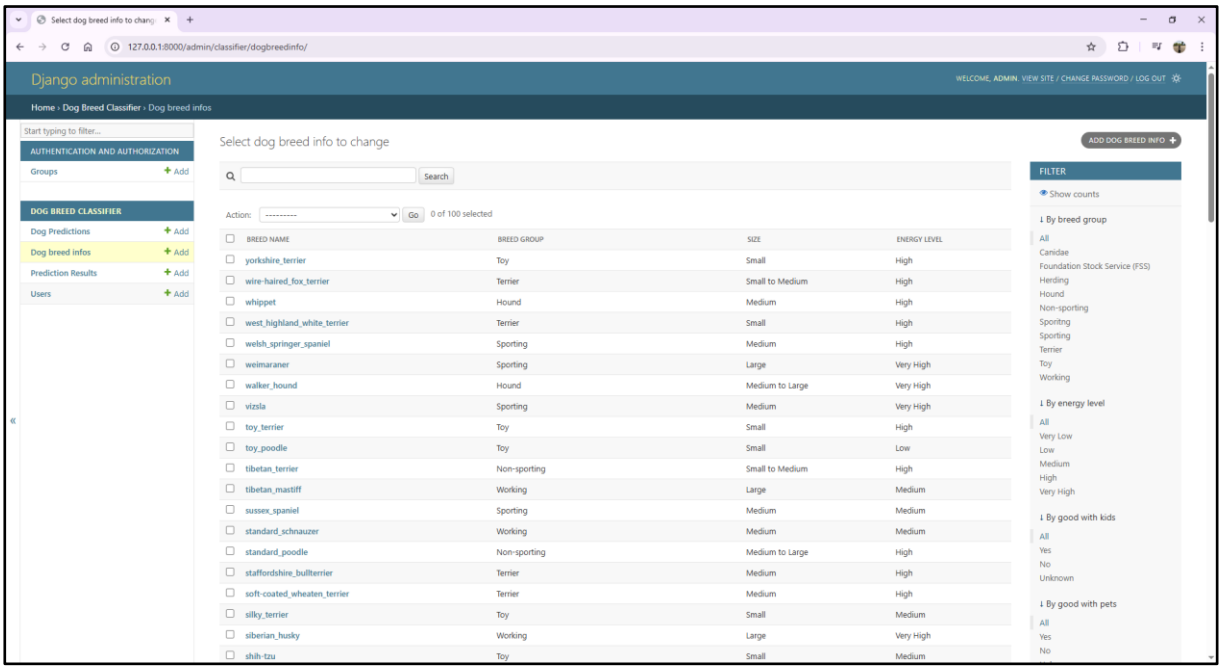


Figure 4.29: Dog Breed Info Table

Figure 4.29 shows the Dog Breed Info Table. This includes size, weight, height range, life expectancy, breed group, temperament, and other relevant characteristics. The admin can click on the name of the dog breed that they want to edit. Not just that, the admin can delete the dog breed info that is not used.

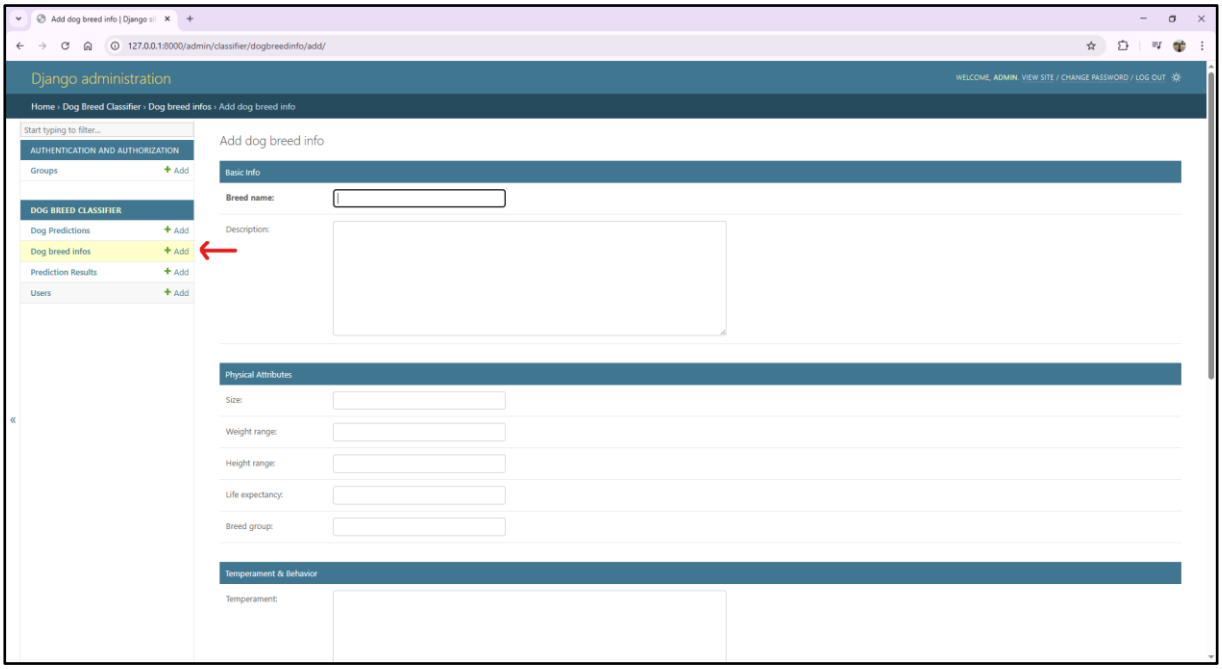


Figure 4.30: Add Dog Breed Info

Figure 4.30 shows the interface where administrators can add, edit, or delete entries using a form-based interface within the admin panel. The admin is required to type in the relevant details of the breed if they wanted to add a new dog breed info into the web application.

4.6 Summary

In summary, this chapter explains the implementation section of the Dog Breed Classification Web Application. It covers the tools and frameworks used, including Django for backend development and TensorFlow for the prediction model. The chapter presents the system structure, including modules for image upload, prediction results, history tracking, and feedback collection. Screenshots of each feature are included to show how the system works. The Django Admin Panel has also been introduced to manage users, predictions, and dog breed data. This chapter outlines the development of each system component, laying the groundwork for testing in the next chapter.

CHAPTER 5 TESTING

5.1 Introduction

This chapter outlines the testing approaches applied during the system's development, focusing on functional and non-functional aspects. Functional testing was conducted to verify the correctness and integration of individual components within the system. These include unit and integration testing to ensure system stability and feature accuracy. Non-functional testing was also conducted to assess the system's performance, reliability, and usability under different conditions. These tests helped ensure the system delivers smooth and efficient user experience, even under varying operational loads. The following sections will provide a detailed discussion of each testing type, including its purpose, implementation process, and the results obtained.

5.2 Functional Testing

Functional testing evaluates whether the system's outputs align with the expected results, offering a more comprehensive view than testing individual components alone (Schmitt, 2022). Since errors often arise from the interactions between different modules, this type of testing focuses on those connections. It may also involve examining specific parts of the code to ensure proper functionality.

5.2.1 Unit Testing

Unit test cases focus on testing individual functions or components of the software separately (SM, 2025). This allows developers to detect and resolve issues early in the development cycle, ensuring that each piece of code works as intended before it is combined with other application parts. The test cases for the proposed web application are shown in Table 5.1 to 5.7, and the test case for the proposed web application Django admin page is shown in Table 5.8.

5.2.1.1 Test Cases for the Proposed Web Application

Table 5.1 shows the test case for the unit testing of Register in the Dog Breed Classification Web Application.

Table 5.1: Test Case for Registering an Account

ID	1	**Module Name**	Register an account			
Objective		To test the register functionality				
Name		*Annastasha Chang See May*	***Date***	*23rd May 2025*	***Status***	*Pass*

ID	Test Case	Input Data	Expected Result	Actual Result	Status
1	Navigate to the Register page	Click "Register"	The Register page is displayed.	As Expected	Pass
2	Valid registration	Valid username, email, password, and checked terms	Redirect to Home page.	As Expected	Pass
3	Invalid username format	Username: `test#`	"Enter a valid username. This value may contain only letters, numbers, and @/./+/-/_ characters."	As Expected	Pass
4	Existing username	Username: `test1`	"A user with that username already exists."	As Expected	Pass
5	Invalid email format	Email: `test1gmailcom`	"Please include an '@' in the email address. 'test1gmailcom' is missing an '@'."	As Expected	Pass
6	Email already registered	Email: `test1@gmail.com`	"User with this Email address already exists."	As Expected	Pass
7	Password too similar with email	Password: `Test.1234`	"The password is too similar to the email address."	As Expected	Pass
8	Password too short, too common, entire numeric	Password: `123456`	"This password is too short. It must contain at least 8 characters." "This password is too common."	As Expected	Pass

			"This password is entirely numeric."		
9	Password and password confirmation does not match	Password: `test.123` Password Confirmation: `test.1234`	"The two password fields didn't match."	As Expected	Pass
10	Field-by-field empty (username only)	Username: (empty), others valid	At Username input textbox will show "Please fill out this field."	As Expected	Pass
11	Field-by-field empty (email only)	Email: (empty), others valid	At Email input textbox will show "Please fill out this field."	As Expected	Pass
12	Field-by-field empty (password only)	Password: (empty), others valid	At Password input textbox will show "Please fill out this field."	As Expected	Pass
13	Field-by-field empty (password confirmation)	Password Confirmation: (empty), others valid	At Password Confirmation input textbox will show "Please fill out this field."	As Expected	Pass

Table 5.2 shows the test case for unit testing for logging into the Dog Breed Classification Web Application.

Table 5.2: Unit Test Case for Login

ID	2	**Module Name**	Login			
Objective		To test the login functionality				
Name		*Annastasha Chang See May*	***Date***	*23rd May 2025*	***Status***	*Pass*

ID	Test Case	Input Data	Expected Result	Actual Result	Status
1	Open the web application and navigate to Login screen	Launch web application	Login page is displayed	As Expected	Pass
2	Login with valid credentials	Username: `test1` Password: `Abcd.1234`	Redirect to Home page	As Expected	Pass

3	Login with invalid credentials	Username: `test1` Password: `123456`	"Please enter a correct username and password. Note that both fields may be case-sensitive."	As Expected	Pass
4	Attempt login with empty fields	Leave fields blank	On empty input textbox will show "Please fill out this field."	As Expected	Pass

Table 5.3 shows the test case for uploading an image into the Dog Breed Classification Web Application.

Table 5.3: Unit Test Case for Upload Image

ID	3	**Module Name**	Upload Image		
Objective		To test image upload functionality			
Name		*Annastasha Chang See May*	***Date***	*23rd May 2025*	***Status***	*Pass*

ID	Test Case	Input Data	Expected Result	Actual Result	Status
1	Upload supported image format	`.jpg` image of a Labrador	Image is accepted and previewed on screen	As Expected	Pass
2	Upload an unsupported image format	`.avif` image	Shows message: "Only JPEG/PNG/WEBP images are allowed."	As Expected	Pass
3	Upload a large image file	8.6MB image	Shows message: "Image too large (8.6MB). Max 5 MB allowed."	As Expected	Pass
4	Remove the uploaded image	Click on 'Remove' button after upload	Image is removed from preview area	As Expected	Pass

Table 5.4 shows the test case for viewing the prediction result in the Dog Breed Classification Web Application.

Table 5.4: Unit Test Case for Prediction Result

ID	4	**Module Name**	Prediction Result			
Objective		To test the accuracy and display of results				
Name		*Annastasha Chang See May*	***Date***	*23rd May 2025*	***Status***	*Pass*

ID	Test Case	Input Data	Expected Result	Actual Result	Status
1	Predict dog breed from valid image	Upload clear image of a Golden Retriever	Result shows "Golden Retriever (95.11%) Labrador Retriever (0.62%) Cocker Spaniel (0.34%)"	As Expected	Pass
2	Upload an unsupported image format	`.avif` image	Shows message: "Only JPEG/PNG/WEBP images are allowed."	As Expected	Pass
3	Predict without uploading image	Click "**Analyse Image**" with no image selected	Shows message: "Please select a valid image file"	As Expected	Pass

Table 5.5 shows the test case for viewing the prediction history in the Dog Breed Classification Web Application.

Table 5.5: Unit Test Case for View History

ID	5	**Module Name**	View History			
Objective		To test viewing of prediction history				
Name		*Annastasha Chang See May*	***Date***	*23rd May 2025*	***Status***	*Pass*

ID	Test Case	Input Data	Expected Result	Actual Result	Status
1	Access the history page from the dashboard	Click on "**History**" in the navigation bar	History page is displayed	As Expected	Pass
2	View records of past predictions	History contains previous predictions	List of images with prediction results and timestamps is shown	As Expected	Pass

3	View history when no predictions made	User has never uploaded an image	Shows message: “You haven't made any predictions yet.” is displayed	As Expected	Pass
4	Check image and breed details in record	Click on a previous predicted image	Prediction Result page is shown	As Expected	Pass
5	Ensure correct user history is shown	Logged-in user has 2 past predictions	Only that user’s history is displayed	As Expected	Pass

Table 5.6 shows the test case for downloading the prediction result in the Dog Breed Classification Web Application.

Table 5.6: Unit Test Case for Download PDF Result

ID	6	**Module Name**	Download PDF Prediction Result		
Objective		To test the functionality of downloading prediction results as a PDF			
Name	*Annastasha Chang See May*	***Date***	*26th May 2025*	***Status***	*Pass*

ID	Test Case	Input Data	Expected Result	Actual Result	Status
1	Click on "Download PDF Report" after a successful prediction	Prediction result exists	A PDF file containing the image, predicted breeds, and confidence levels is downloaded	As Expected	Pass
2	PDF content check	Valid prediction data exists	PDF file contains image, top 3 breed predictions, confidence %, and timestamp	As Expected	Pass
3	Check PDF file name format	Prediction completed	PDF named as "`Dog_Prediction_<date>_<timestamp>.pdf`"	As Expected	Pass
4	Check image and breed details in record	Click on a previous predicted image	Prediction Result page is shown	As Expected	Pass

5	Open the downloaded file	Downloaded PDF	File opens without error and content is readable	As Expected	Pass

Table 5.7 shows the test case for logout in the Dog Breed Classification Web Application.

Table 5.7: Unit Test Case for Logout

ID	7	**Module Name**	Logout			
Objective		To test the logout functionality				
Name		*Annastasha Chang See May*	***Date***	*26th May 2025*	***Status***	*Pass*

ID	Test Case	Input Data	Expected Result	Actual Result	Status
1	Logout from dashboard	Click on "**Logout**" button from navigation	User is logged out and redirected to login screen	As Expected	Pass
2	Attempt to access dashboard after logout	Use browser back button after logout	Redirects to login screen with session expired	As Expected	Pass

5.2.1.2 Test Cases for the Proposed Web Application Django Admin

Table 5.8 shows the test case for the admin module of the Dog Breed Classification Web Application by using the Django Admin page.

Table 5.8: Unit Test Case for Admin Panel

ID	8	**Module Name**	Admin Panel			
Objective		To test admin functionalities and access control				
Name		*Annastasha Chang See May*	***Date***	*26th May 2025*	***Status***	*Pass*

ID	Test Case	Input Data	Expected Result	Actual Result	Status
1	Admin login with valid credentials	Username: admin, Password: Admin@123	Admin dashboard is displayed	As Expected	Pass
2	Admin login with invalid credentials	Username: admin, Password: wrongpass	Error message: "Invalid credentials."	As Expected	Pass

3	Access admin panel as non-admin user	Logged in as regular user	Access denied	As Expected	Pass
4	View all registered users	Click on "Manage Users"	List of all users with details is displayed	As Expected	Pass
5	View prediction history by all users	Click on "View Predictions"	List of prediction records with usernames, images, and results	As Expected	Pass
6	Search for a specific user	Enter keyword in user search bar	Matching users are displayed	As Expected	Pass
7	Logout from admin panel	Click on "Logout"	Redirected to login screen	As Expected	Pass

5.2.2 Integration Testing

Integration test cases evaluate how different modules or components of the system interact with one another (SM, 2025). Their primary purpose is to detect any problems that may occur when these parts function together.

Table 5.9 shows the integration test cases for the Dog Breed Classification Web Application.

Table 5.9: Integration Test Cases for Dog Breed Classification Web Application

ID	9	**Application Name**	Dog Breed Classifier		
Objective		Ensure the app functions smoothly and that various modules interact correctly.			
Name		*Annastasha Chang See May*	***Date***	*29th May 2025*	***Status***	*Pass*

Test Scenario: Integration Testing for Register and Login					
ID	**Test Case**	**Input Data**	**Expected Result**	**Actual Result**	**Status**
1	Verify Sign Up and Login integration	New user details	User signs up successfully and can log in	As Expected	Pass
2	Verify error handling during Sign Up	Empty email, weak password	Shows proper validation errors	As Expected	Pass

3	Verify successful Registration redirects to Login	Tap "**Register**", then submit	Redirects to Home page	As Expected	Pass

Test Scenario: Integration Testing for Login and Home					
ID	**Test Case**	**Input Data**	**Expected Result**	**Actual Result**	**Status**
4	Verify Login redirects to Home	Valid credentials	Redirects to Home page	As Expected	Pass
5	Verify failed login blocks access to Home	Invalid credentials	Stays on Login with error	As Expected	Pass

Test Scenario: Integration Testing for Logout and Login					
ID	**Test Case**	**Input Data**	**Expected Result**	**Actual Result**	**Status**
6	Verify Logout returns to Login screen	Tap Logout	Redirects to Login screen	As Expected	Pass
7	Verify re-login after Logout	Valid credentials	Logs in again successfully	As Expected	Pass

Test Scenario: Integration Testing for Image Upload and Prediction					
ID	**Test Case**	**Input Data**	**Expected Result**	**Actual Result**	**Status**
8	Verify valid image triggers prediction	JPG image of dog	Shows top 3 breed prediction result	As Expected	Pass
9	Verify unsupported file format	Upload PDF file	Shows message: "Only JPEG/PNG/WEBP images are allowed."	As Expected	Pass
10	Verify empty image upload	No file selected	Shows message: "Please select a valid image file."	As Expected	Pass

Test Scenario: Integration Testing for Prediction and History Logging					
ID	**Test Case**	**Input Data**	**Expected Result**	**Actual Result**	**Status**
11	Verify successful prediction is logged	Image file	Shows message: "Ready to upload!"	As Expected	Pass
12	Verify failed prediction not logged	Invalid file	Not logged in history	As Expected	Pass
13	Verify viewing history after login	Navigate to history page	Shows user's previous predictions	As Expected	Pass

Test Scenario: Integration Testing for Database and Authentication					
ID	**Test Case**	**Input Data**	**Expected Result**	**Actual Result**	**Status**
14	Verify registered user stored in DB	New user details	User appears in DB	As Expected	Pass
15	Verify session created after login	Valid login	Session active	As Expected	Pass
16	Verify session destroyed on logout	Tap logout	Session ends	As Expected	Pass

Test Scenario: Integration Testing for Navigation Across Modules					
ID	**Test Case**	**Input Data**	**Expected Result**	**Actual Result**	**Status**
17	Navigate from Home to Upload	Click "**Upload**"	Opens Upload page	As Expected	Pass
18	Navigate from History to one of the past prediction result	Click on a past prediction	Opens prediction result	As Expected	Pass
19	Navigate from Prediction Result to Home	Click "**Home**"	Redirects to Home page	As Expected	Pass
20	Access restricted page without login	Open /history without logging in	Redirect to Login	As Expected	Pass

Test Scenario: Integration Testing for Navigation Across Modules					
ID	**Test Case**	**Input Data**	**Expected Result**	**Actual Result**	**Status**
21	Upload large image file	Image > 5MB	Shows message: "Image too large (8.6MB). Max 5 MB allowed."	As Expected	Pass
22	Submit empty Sign-Up form	No data	Shows "Please fill out this field."	As Expected	Pass
23	Upload non-image file	.txt file	Shows message: "Only JPEG/PNG/WEBP images are allowed."	As Expected	Pass

Test Scenario: Integration Testing for Prediction Performance and Loading					
ID	**Test Case**	**Input Data**	**Expected Result**	**Actual Result**	**Status**
24	Check prediction response time	Upload image	Result shown within 5 seconds	As Expected	Pass

25	Check loading indicator	During prediction	Loading spinner is shown	As Expected	Pass

5.3 Non-functional Testing

Non-functional testing evaluates aspects of an application that, while not directly tied to its core functionality, significantly impact the user experience (Schmitt, 2022). Factors like performance and reliability under stress are not part of the software's functional features but play a crucial role in user satisfaction. A failure in non-functional testing might not be immediately noticeable to users, but it can reveal underlying issues in the system, particularly when operating on a scale.

5.3.1 Reliability Testing

Reliability testing is a crucial aspect of software testing that evaluates the stability and dependability of a system by assessing its performance under various stress conditions, data loads, and environmental scenarios over time (testsigma, 2023). The main objective is to ensure the software operates consistently without unexpected failures, identifying and addressing potential flaws or vulnerabilities before deployment. By simulating real-world usage patterns, reliability testing helps prevent system breakdowns, downtime, or data loss, ultimately contributing to a stable and dependable user experience.

Table 5.10 shows the reliability test case for the Dog Breed Classification Web Application.

Table 5.10: Test Case for Reliability Testing

ID	10	**Web Application Name**	Dog Breed Classifier			
Objective		To test the reliability and stability of the web application.				
Name		*Annastasha Chang See May*	***Date***	*29th May 2025*	***Status***	*Pass*

Test Scenario: Verify the reliability and stability of the web application under various operating and edge case conditions.					
ID	**Test Case**	**Input Data**	**Expected Result**	**Actual Result**	**Status**
1	Continuous Usage Test	Use web application for 4 hours straight	No crashes, stable performance	As Expected	Pass
2	Upload small size image	Upload image < 10KB	No crash	As Expected	Pass
3	Network Disconnection	Predict during network loss	No crash	As Expected	Pass
4	Server Restart	Restart Django server during use	App reconnects, no data loss	As Expected	Pass
5	Invalid File Type	Upload PDF instead of image	Shows message: "Only JPEG/PNG/WEBP images are allowed."	As Expected	Pass
6	Large Image Upload	Upload 8MB image	Shows message: "Image too large (8.6MB). Max 5 MB allowed."	As Expected	Pass
7	Browser Compatibility	Use Chrome, Firefox, Edge	App works on all tested browsers	As Expected	Pass
8	Simultaneous Users	2 users predict simultaneously	App handles parallel users	As Expected	Pass
9	Invalid Image Content	Upload photo of non-dog object	Displays “Please upload a clearer dog image.” web page	As Expected	Pass
10	Prediction Result Recovery	Refresh after prediction	Result remains accessible	As Expected	Pass

5.3.2 Performance Testing

A key aspect of non-functional testing is performance testing, which verifies that a software system responds quickly and efficiently to user requests (Schmitt, 2022). High latency can significantly degrade the user experience, but effective performance testing can identify potential issues early on, allowing them to be addressed before they impact users.

In this project, the performance evaluation of the dog breed classification model focuses primarily on the model's ability to include the correct breed within the top three predicted results (top-3 accuracy). Unlike traditional single-label accuracy metrics that require a fixed confidence threshold, top-3 accuracy provides a more practical and user-friendly measure of performance, reflecting the realistic usage scenario of the system.

As such, this evaluation method prioritises whether the true breed appears in the top three predictions rather than demanding a strict numerical confidence value for a single prediction. This approach better accommodates the inherent variability in image quality, lighting conditions, and breed similarity that affect classification. The focus is on whether the system correctly suggests the true breed among the top three predictions, ensuring a balance between accuracy and usability for end-users. Do take note that the first image classification after starting the application may take more than 5 seconds to generate a prediction. This delay occurs because the deep learning model needs to be loaded into memory and initialised for inference. Subsequent predictions will be slightly faster as the model remains in memory.

Table 5.11 shows the performance test case for the Dog Breed Classification Web Application.

Table 5.11: Performance Test Case for Dog Breed Classification Model's Accuracy

ID	11	**Web Application Name**	Dog Breed Classifier			
Objective		To test the reliability and stability of the web application.				
Name		*Annastasha Chang See May*	***Date***	*30th May 2025*	***Status***	*Pass*

ID	Test Case	Input Data	Expected Result	Actual Result	Accurate
1	Detect clear image of Labrador Retriever	Centered image of Labrador Retriever	Correct breed appears in top 3 predictions	As Expected	Yes
2	Detect dog in low-light	Dim-lit photo of Golden Retriever	Correct or similar breed appears in top 3	As Expected	Yes
3	Detect dog in side profile	Profile shot of Pomeranian	Correct breed prediction appears in top 3	As Expected	Yes
4	Detect dog in motion	Slightly blurred image of moving dog	Acceptable breed or related breed in top 3	As Expected	Yes
5	Detect puppy of known breed	Image of young German Shepherd	Correct or parent breed detected	As Expected	Yes
6	Detect mixed breed dog	Image of mixed-breed dog	One parent breed included in top 3	As Expected	Yes
7	Detect different coat colors	Same breed (Poodle) in different colors	Still correctly detects breed	As Expected	Yes
8	Detect different dog sizes	Same breed in toy vs standard size	Maintains consistent breed prediction	As Expected	Yes
9	Detect breed in crowded background	Dog in a busy park photo	Accurate or near-accurate prediction	As Expected	Yes
10	Detect rare breed	Image of Saluki	Predicts correctly or includes in top 3	As Expected	Yes
11	Detect breed from low-resolution image	128x128px image of Husky	Identifies breed within top 3 predictions	As Expected	Yes

12	Detect dog wearing clothes	Golden Retriever in jacket or hat	Breed still identified accurately	As Expected	Yes
13	Detect breed from grayscale image	Black and white image of Golden Retriever	Acceptable accuracy, not color-dependent	As Expected	Yes
14	Detect breed with background removal	Dog cropped on white background	High accuracy maintained	As Expected	Yes
15	Predict top-3 breeds	Random test image	Returns top-3 likely breeds	As Expected	Yes
16	Predict with rotated image	Rotated 90° image of a dog	Accurate prediction maintained	As Expected	Yes
17	Predict breed under color filter	French Bulldog with warm tone filter	Breed still correctly detected	As Expected	Yes
18	Detect dog with human in frame	Irish Wolfhound beside person	Model focuses on dog and includes correct breed	As Expected	Yes
19	Misclassified non-dog image	Upload image of a cat	Model returns incorrect predictions or handles error	As Expected	No

5.3.3 User Acceptance Testing

Usability testing evaluates how effective and satisfying the user's experience is when interacting with a system (Schmitt, 2022). Although it is generally a manual process that does not easily scale, skipping usability testing, particularly during application localisation, can result in confusing or difficult-to-use interfaces. Therefore, it is beneficial to dedicate time to this type of testing during software development.

A total of fifteen (15) participants were involved in this testing phase. The survey aimed to gather feedback on aspects such as the accuracy of the breed predictions, ease of navigation, interface design, and overall user experience. A sample of the survey form is included in

Appendix C for reference. The responses collected from all fifteen testers were compiled and analysed. The summary of the findings is presented in the following subsection.

5.3.3.1 Analysis on User Acceptance Survey

The questionnaire consists of 3 sections: User Demographics, Usability and Functionality of the Dog Breed Classification web application, and Overall Experience of using the Dog Breed Classification application. Below is a detailed analysis of the responses received from 15 testers. For the first section, user demographics, there is a total of 53.3% (8 respondents) between 18 and 25 years old, and 46.7% (7 respondents) between 26 and 35 years old. Among the 15 respondents, 7 respondents (46.7%) had used a dog breed classification system before, while 8 respondents (53.3%) had never used it.

Moving on to the second section, the usability and functionality of the dog breed classification web application were evaluated through several key aspects, including ease of navigation, performance speed, clarity of output, usefulness of information, and prediction accuracy. The results are shown below:

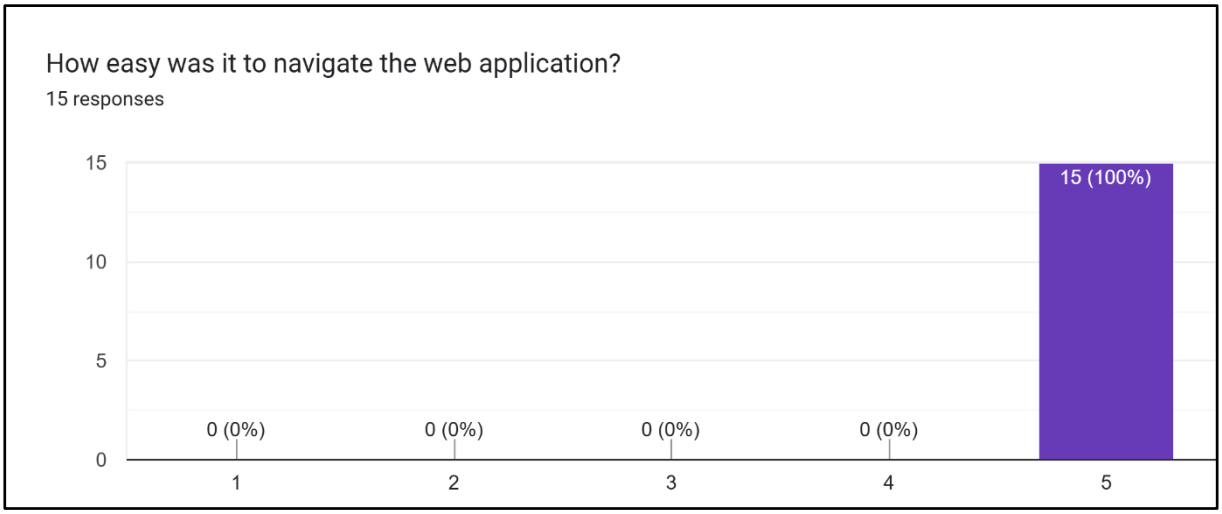


Figure 5.1: Ease of Navigation

Based on Figure 5.1 above, all 15 respondents (100.0%) rated the application as very easy to navigate, highlighting the system's user-friendly nature and intuitive interface.

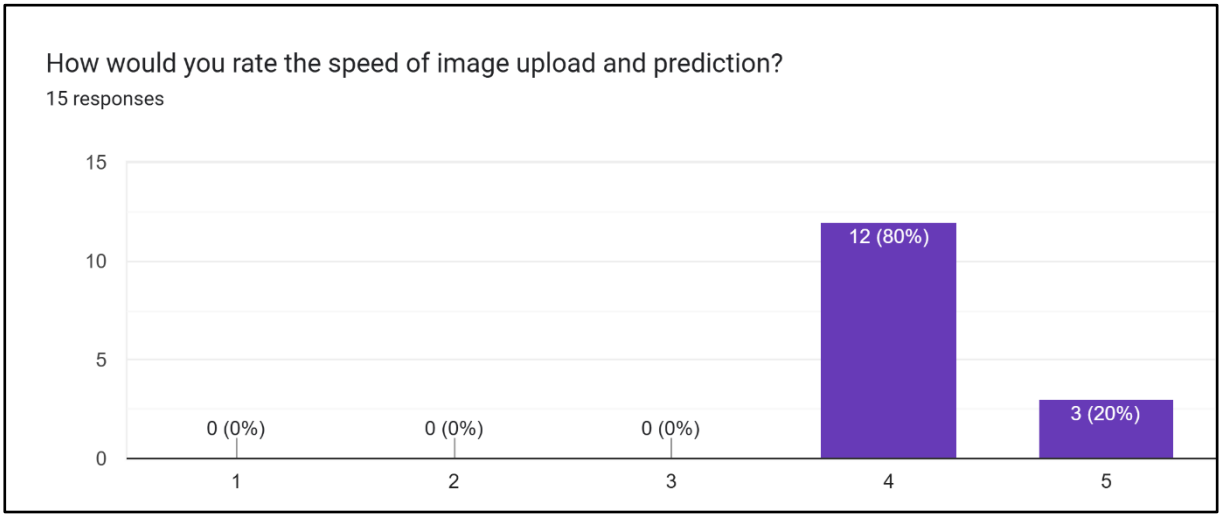


Figure 5.2: Speed of Image Upload and Prediction

Based on Figure 5.2 above, regarding the performance, 12 respondents (80.0%) rated the speed of image upload and prediction as fast. In comparison, the remaining 3 respondents (20.0%) rated it as very fast, indicating that the application delivered a smooth and efficient user experience with minimal delays.

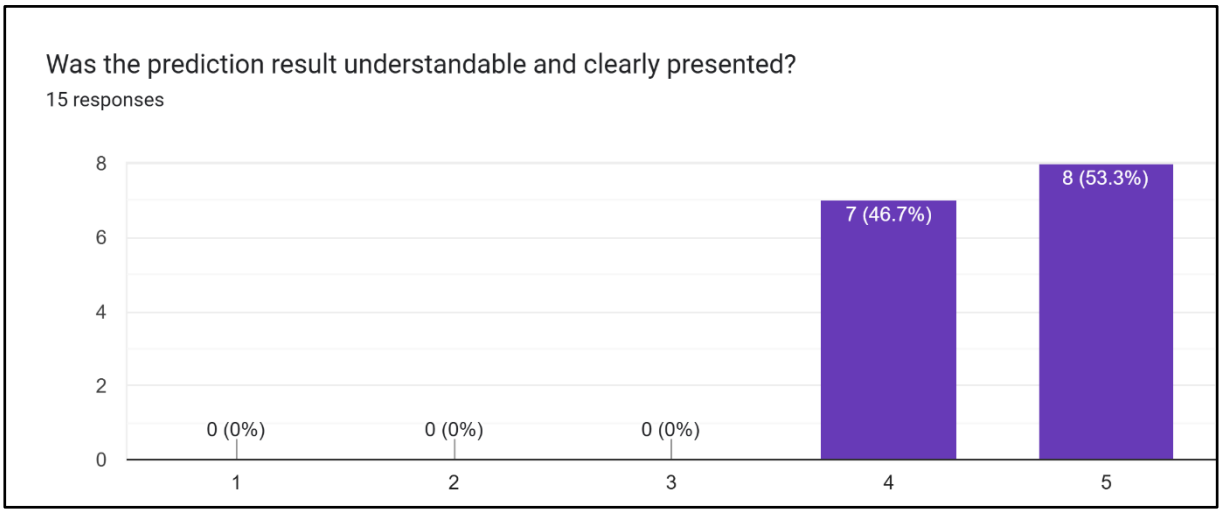


Figure 5.3: Clarity of Prediction Result

Based on Figure 5.3 above, 8 participants (53.3%) strongly agreed that the prediction output was understandable and clearly displayed, while 7 participants (46.7%) agreed. This suggests that all users were able to interpret the prediction results easily.

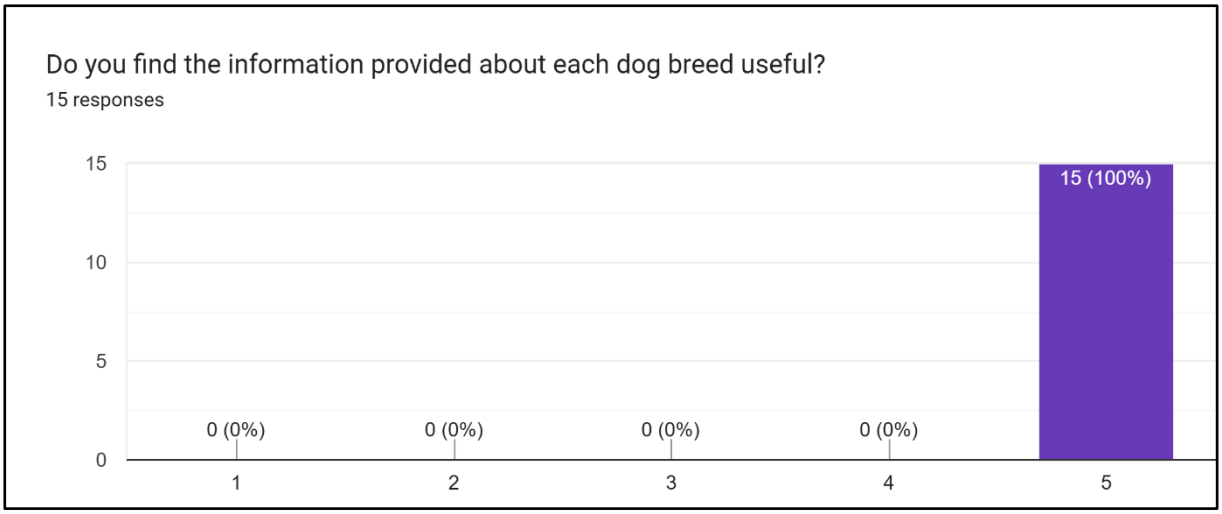


Figure 5.4: Usefulness of Dog Breed Information

Based on Figure 5.4 above, the usefulness of the additional breed information provided after classification was rated as very useful by every respondent (100.0%), reflecting the relevance and value of the content delivered by the application.

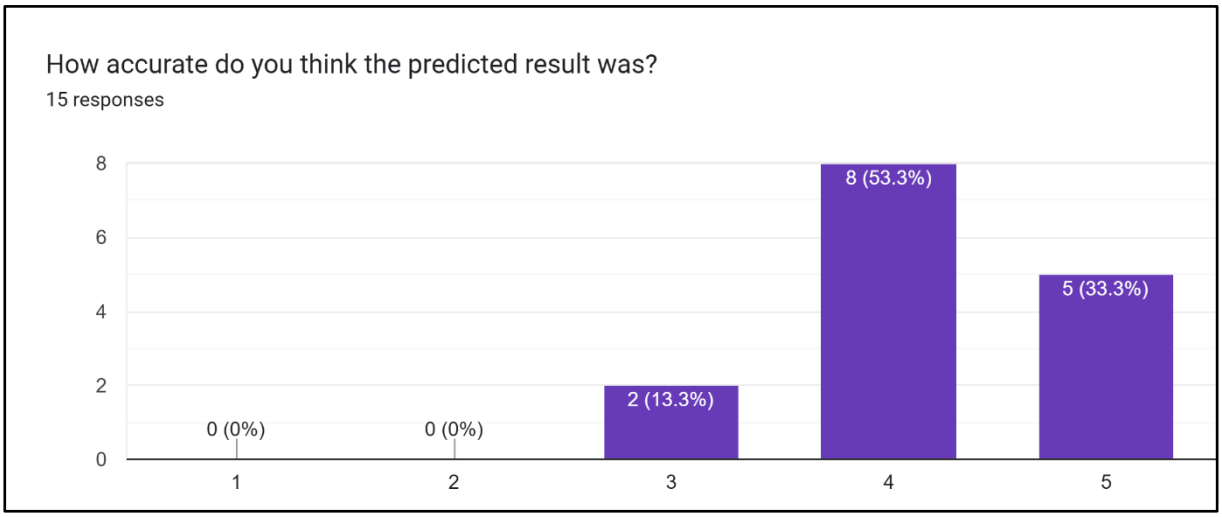


Figure 5.5: Accuracy of Prediction

Based on the Figure 5.5 above, 5 participants (33.3%) found the results to be very accurate, 8 participants (53.3%) considered them accurate, and 2 participants (13.3%) rated them as moderately accurate. While the majority were satisfied with the model's performance, the feedback suggests some opportunities for refining the prediction precision further.

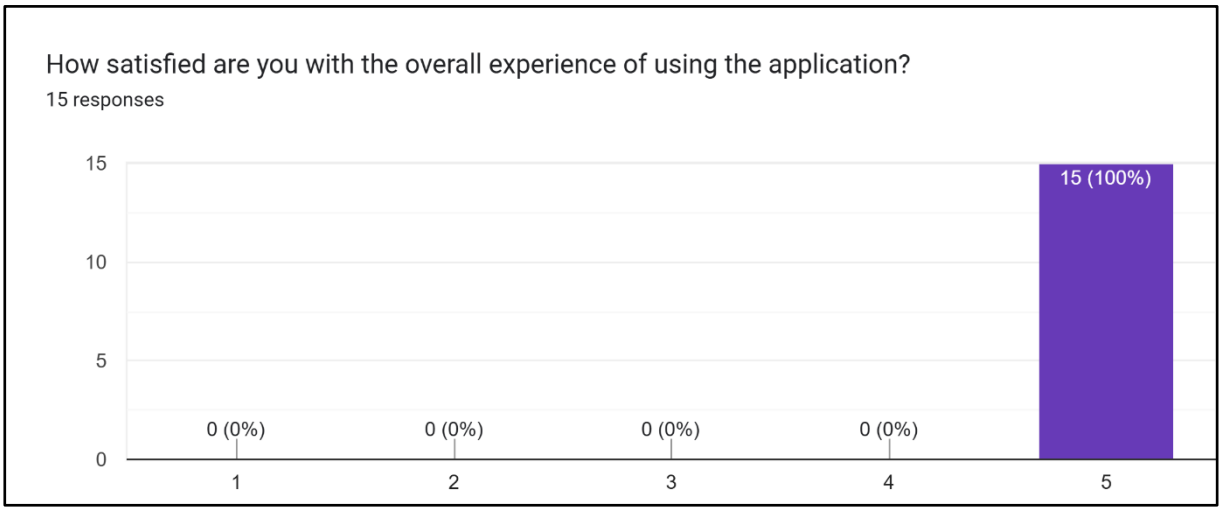


Figure 5.6: Overall Satisfaction

Based on Figure 5.6 above, the response to the web application was overwhelmingly positive. All 15 participants (100.0%) reported being very satisfied with their experience using the application.

So, to conclude, this consistent feedback across all surveyed areas confirms the application's effectiveness in delivering functional performance and a high-quality user experience.

5.4 Discussion on Results

The dog breed classification system successfully met its primary objective of identifying dog breeds from uploaded images using a deep learning approach. By leveraging a pre-trained model (EfficientNetB3) with transfer learning, the application demonstrated the ability to classify images with a relatively high accuracy. Integration with a Django web framework

enabled the deployment of a functional web application that allows users to upload images and view breed predictions, including a top 3 suggestion list and relevant breed information.

5.5 Summary

In summary, this chapter discusses the functional and non-functional testing conducted on the proposed web application. For functional testing, unit testing, and integration testing were performed on the web application. As for non-functional testing, reliability testing, performance testing, and user-acceptance testing are performed. There are 15 respondents for the user acceptance questionnaire.

CHAPTER 6: CONCLUSION AND FUTURE WORK

6.1 Introduction

This final chapter summarises the results of the developed dog breed classification web application using a deep learning approach. It assesses the degree of accomplishment of the initial objectives mentioned earlier in Chapter 1, examines the restrictions faced during the project, and suggests possible directions for future improvements to enhance the system's performance and applicability.

6.2 Objectives and Achievements

This project has effectively achieved all the objectives stated in Chapter 1. The objectives and achievements are listed in Table 6.1 below.

Table 6.1: Achievements of the Dog Breed Classification Web Application

No.	Objectives	Achievements
1	To implement a transfer learning model for classifying dog breeds	A pre-trained CNN model (EfficientNetB3) was successfully adapted using transfer learning. It was fine-tuned on the dog breed dataset, achieving high accuracy and effectively identifying over 100 different dog breeds.
2	To develop a user-friendly web interface for dog breed identification	A web-based system was developed for users to upload pictures of dogs. The interface shows the top three predicted breeds with their similarity percentages. It's user-friendly, making it easy for pet owners and shelter staff to use.
3	To integrate and test the whole system for performance, reliability, and scalability	The deep learning model was added to the Django web application. The system is tested to ensure it performs well and responds quickly on various browsers and devices. It can handle many images at once without losing accuracy or speed.

6.3 Project Limitations and Constraints

This project met its main goals and showed that deep learning can be used to categorise dog breeds. However, some limitations and challenges came up during its development and evaluation. While these issues did not affect the system's effectiveness, they highlight areas for future improvement and refinement:

1. **Dataset Imbalance**

   The Kaggle dog breed dataset used in this experiment had an uneven number of photos for different breeds. Some popular breeds had many examples, while others had very few. This imbalance led to biased learning, making the model more accurate for the common breeds but less effective for the rare ones. This could affect the fairness and general usefulness of the predictions.

2. **High Visual Similarity Among Breeds**

   Some dog breeds share similar physical traits, including coat color, size, and facial shape. For example, the Alaskan Malamute and Siberian Husky, as well as various types of Spaniels, share these characteristics. Although the model's overall accuracy was high, it occasionally misclassified visually similar breeds due to slight variances that are difficult to detect, even with transfer learning.

3. **Limited to Purebred Classification**

   Only images of purebred dogs were used to train the model. Because of this, it is unable to correctly identify or depict mixed-breed dogs, which make up a significant proportion of the dog population. This restriction limits the system's practicality in environments where mixed-breed dogs are common, such as animal shelters or adoption websites. Furthermore, the web application is designed to detect only one breed per image, which

reduces accuracy in cases where an image contains multiple dogs or dogs of mixed heritage.

4. **Computational Resource Requirements**

   Training deep learning models, especially during the fine-tuning of large pre-trained networks, requires access to GPU-enabled environments such as Kaggle or Google Colab. These systems have limited compute time and session constraints, even if they are helpful. The demand for high-performance resources may restrict scalability in real-world deployment situations unless specifically targeted for inference efficiency.

5. **Initial Prediction Delay Due to Model Loading**

   Another notable limitation observed during system usage was the delay experienced during the first image classification request. When the application is first accessed or restarted, the deep learning model requires several seconds to load into memory and initialise, often taking more than 5 seconds before returning the first prediction result. This initial delay is due to the need to load the model weights and allocate necessary system resources, particularly in local or non-persistent hosting environments.

6. **Sensitivity to Real-World Image Variations**

   When tested with images outside the clean, high-quality training images, such as those with poor lighting, cluttered backgrounds, motion blur, or partial dog visibility, the model showed decreased performance.

7. **Misclassification of Non-Dog Images**

   One major drawback noted was the model's inability to distinguish dogs from other visually similar animals. For example, the system still attempted to classify images of cats or other furry animals with traits similar to dogs, that is, foxes or small mammals, as a dog breed. This is so because the model lacked a more general classification

mechanism to first confirm whether the input image indeed included a dog and was trained just on dog images. Any animal image, even if obviously not a dog to the human eye, was thus pushed into one of the known dog breed categories, producing erroneous and maybe confusing results.

6.4 Future Works

While the system has successfully met its core objectives, several areas for future enhancement can significantly expand its functionality, robustness, and societal impact. These proposed improvements aim to refine classification accuracy, enhance the system's real-world usability, and broaden its relevance to pet owners, shelters, and the general public.

1. **Dog vs. Non-Dog Image Verification**

   The system's current inability to distinguish dogs from visually similar animals, such as cats, is a central area that needs improvement. This causes non-dog images to be wrongly identified as particular dog breeds. Future iterations should include a first binary classification step to confirm whether an uploaded image features a dog. By doing so, it would raise general accuracy and prevent incorrect breed predictions.

2. **Deployment to a Web Hosting Platform**

   The web application currently runs locally, limiting accessibility and usability for testing or public use. Future work involves deploying it to a cloud hosting platform like Heroku, AWS, or Render. This would enable broader access, facilitate user feedback, and allow integration with services like mobile apps or third-party APIs.

3. **Dataset Enhancement and Mixed-Breed Support**

   The web application deployed EfficientNetB3 architecture for accurate classification. However, its performance is influenced by the quality and variety of the dataset. Including more images, especially of mixed-breed dogs, underrepresented breeds, and

photos from various real-world environments with different lighting and backgrounds, will significantly enhance the model's ability to generalize and be robust.

4. **Lost and Found Dog Module**

   It would be ideal to add a "Lost Dog" section where pet owners can share details and a photo of their missing dog, helping them to reunite them with their furry friends. People nearby will be able to view the information about the lost dog, and the owner can check the notifications to see if anyone has spotted the dog. If someone finds a dog that matches the description in the lost dog post, they can contact the owner through the web application. This feature would help create a supportive community and increase the chances of lost pets being safely returned to their owners.

5. **Adoption Support and Pet Shelter Integration**

   To enhance the platform's usability, a module linking users to local pet shelters and adoption centers could be developed. Once a dog breed is identified, the system can provide information about nearby locations where similar breeds are available for adoption. This would support pet adoption efforts and promote responsible pet ownership.

6. **Real-Time Prediction and Model Warm-Up**

   The system currently experiences a noticeable delay during the first prediction due to model initialisation. Future deployments could benefit from implementing a background model warm-up mechanism or using a lightweight serverless architecture that maintains the model in memory. This would provide real-time predictions from the very first request, improving the user experience.

7. **Improved Real-World Performance**

To handle diverse image qualities (e.g., motion blur, low lighting, partial occlusions), future development can include more varied and realistic images to help the model generalise better to user-submitted content.

8. **Enhancing the User Feedback Loop**

   The current system already includes a feedback section where users have answered whether the top 3 predictions are accurate or not. As a future enhancement, this feedback mechanism could be extended to support active learning, where user inputs are systematically collected and used to retrain or fine-tune the model over time. This would ensure that the system evolves based on real-world user interactions, thereby improving accuracy and adaptability.

6.5 Conclusion

The project successfully developed a deep learning-based system for dog breed classification, achieving high accuracy and demonstrating the potential of AI in pet identification applications. While the system performs well under controlled conditions, addressing the identified limitations through the proposed future works will be crucial for enhancing its robustness and applicability in diverse real-world scenarios. Continued research and development in this area hold promise for more advanced and accessible solutions in animal breed identification.

REFERENCES

Acharya, G., Borade, R., Shejwal, P., & Gaikwad, S. (2022, April). *Dog breed prediction system (web)*. International Research Journal of Engineering and Technology. https://mail.irjet.net/archives/V9/i4/IRJET-V9I4297.pdf

activeloop. (2023). *What is InceptionV3*. Activeloop.ai. https://www.activeloop.ai/resources/glossary/inception-v-3/

Belcic, I., & Stryker, C. (2024, June). *What is an entity relationship diagram? | IBM*. Www.ibm.com. https://www.ibm.com/think/topics/entity-relationship-diagram

Borwarnginn, P., Kusakunniran, W., Karnjanapreechakorn, S., & Thongkanchorn, K. (2020). Knowing your dog breed: Identifying a dog breed with deep learning. *International Journal of Automation and Computing*, *18*, 45–54. https://doi.org/10.1007/s11633-020-1261-0

Chen, S., Ahmmed, S., Lal, K., & Deming, C. (2020). Django web development framework: Powering the modern web. *American Journal of Trade and Policy*, *7*, 99–106. https://doi.org/10.18034/ajtp.v7i3.675

clickworker. (2024, February). *All about deep learning & datasets for deep learning*. Clickworker.com. https://www.clickworker.com/ai-glossary/deep-learning/

Dekkati, S., Lal, K., & Desamsetti, H. (2019). React native for android: Cross-platform mobile application development. *Global Disclosure of Economics and Business*, *8*, 153–164. https://doi.org/10.18034/gdeb.v8i2.696

Drumond, C. (2024). *What is scrum and how to get started*. Atlassian. https://www.atlassian.com/agile/scrum

Ejie, C. (2024, January). *Agile methodologies: Navigating change in software development*. Www.linkedin.com. https://www.linkedin.com/pulse/agile-methodologies-navigating-change-software-development-ejie-eukdf/

GeeksforGeeks. (2019, March). *Levels in data flow diagrams (DFD) - GeeksforGeeks*. GeeksforGeeks. https://www.geeksforgeeks.org/levels-in-data-flow-diagrams-dfd/

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

Gupta, D. (2024, July). *Master transfer learning by using pre-trained models in deep learning*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2017/06/transfer-learning-the-art-of-fine-tuning-a-pre-trained-model/

Han, J., Kamber, M., & Pei, J. (2012). *Data mining : Concepts and techniques*. Elsevier.

Heller, M. (2022, July). *What is visual studio code? Microsoft's extensible code editor*. InfoWorld. https://www.infoworld.com/article/2335960/what-is-visual-studio-code-microsofts-extensible-code-editor.html

Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, *18*, 1527–1554. https://doi.org/10.1162/neco.2006.18.7.1527

Holdsworth, J., & Scapicchio, M. (2024, June). *What is deep learning?* Www.ibm.com; IBM. https://www.ibm.com/topics/deep-learning

IBM. (2024). *What are convolutional neural networks? | IBM*. Www.ibm.com; IBM. https://www.ibm.com/topics/convolutional-neural-networks

Khrupa, A. (2022, March). *What is wireframe in software development what is wireframe in software development? Benefits & TipsWhat is wireframe in software development | QArea blog*. Software Development Company. https://qarea.com/blog/what-is-wireframe-in-software-development

Klingler, N. (2024, March 18). *EfficientNet: Optimizing Deep Learning Efficiency - viso.ai*. Viso.ai. https://viso.ai/deep-learning/efficientnet/

Kundu, N. (2023, January). *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

Lee, A. (2022, December). *What is a pretrained AI model?* NVIDIA Blog. https://blogs.nvidia.com/blog/what-is-a-pretrained-ai-model/

Madurapperuma, I. H., Shafana, M. S., & Sabani, M. J. A. (2022, August). *State-of-art frameworks for front-end and back-end web development*. http://192.248.66.13/bitstream/123456789/6339/3/62-67.pdf

Manivannan, S., & Venkateswaran, N. (2023). *Dog breed classification using inception-resnet-v2*. 1–5. https://doi.org/10.1109/ICONAT57137.2023.10080065

MathWorks. (2024). *What is a convolutional neural network? | 3 things you need to know*. Www.mathworks.com; MathWorks. https://mathworks.com/discovery/convolutional-neural-network.html

Mohd,. (2020, February). *Image classification using CNN*. Analytics Vidhya. https://analyticsvidhya.com/blog/2020/02/learn-image-classification-cnn-convolutional-neural-networks-3-datasets/

Mozilla. (2024, November). *Django introduction*. MDN Web Docs. https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Introduction

Olivas, E. S., David, J., Sober, M. M., Rafael, J., & Jose, A. (2009). *Handbook of research on machine learning applications and trends : algorithms, methods, and techniques*. Igi Global (701 E. Chocolate Avenue, Hershey, Pennsylvania, 3, Usa.

Peek, S. (2024, June). *What is agile scrum methodology?* Business News Daily. https://www.businessnewsdaily.com/4987-what-is-agile-scrum-methodology.html

Ph.D, J. M., & Kavlakoglu, E. (2024, January). *Transfer learning*. Ibm.com. https://www.ibm.com/topics/transfer-learning#f03

Potrimba, P. (2024, March). *What is ResNet-50?* Roboflow Blog. https://blog.roboflow.com/what-is-resnet-50/

Product, A. (2024, January 3). *Product Teacher*. Product Teacher. https://www.productteacher.com/quick-product-tips/efficientnet-for-ai-product-managers?utm_source=chatgpt.com

Purwono, P., Ma'arif, A., Rahmaniar, W., Haris, F., Kusuma, Z., & Haq,. (2023). Understanding of convolutional neural network (CNN): A review. *International Journal of Robotics and Control Systems*, *2*, 739–748. https://doi.org/10.31763/ijrcs.v2i4.888

Sarker, I. H. (2021). Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions. *SN Computer Science*, *2*. https://doi.org/10.1007/s42979-021-00815-1

Schmitt, J. (2022, April). *Functional vs non-functional software testing*. CircleCI. https://circleci.com/blog/functional-vs-non-functional-testing/

Scrum.org. (2024). *What is scrum?* Scrum.org. https://www.scrum.org/resources/what-scrum-module

SM, R. (2025, February). *Test case template with explanation*. Software Testing Material. https://www.softwaretestingmaterial.com/test-case-template-with-explanation/

Team, I. E. (2024, August). *The phases of software testing: Explanation and steps*. Indeed. https://www.indeed.com/career-advice/career-development/phases-of-testing

testsigma. (2023, August). *What is reliability testing? (with examples)*. Testsigma Blog. https://testsigma.com/guides/reliability-testing/

UC Merced Library. (2023). *What is a data dictionary? | UC merced library*. Library.ucmerced.edu. https://library.ucmerced.edu/data-dictionaries

Varghese, S., & Remya, S. (2021). Dog breed classification using CNN. In S. David, A. R. S, V. Jeyakrishnan, & M. Niranjanamurthy (Eds.), *Security Issues and Privacy Concerns in Industry 4.0 Applications* (pp. 195–205). Wiley Data and Cybersecurity. https://ieeexplore.ieee.org/document/9822355

Wadhai, P. (2019, September). *The data science algorithmic classification of dog breed.* Medium. https://medium.com/@wadhai.piyush/the-data-science-algorithmic-classification-of-dog-breed-be38eb3ceab0

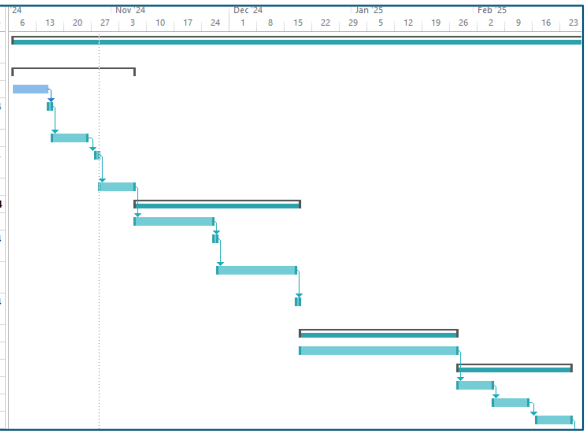
Yegulalp, S. (2024, January). *What is TensorFlow? The machine learning library explained*. InfoWorld. https://www.infoworld.com/article/2255099/what-is-tensorflow-the-machine-learning-library-explained.html

APPENDICES

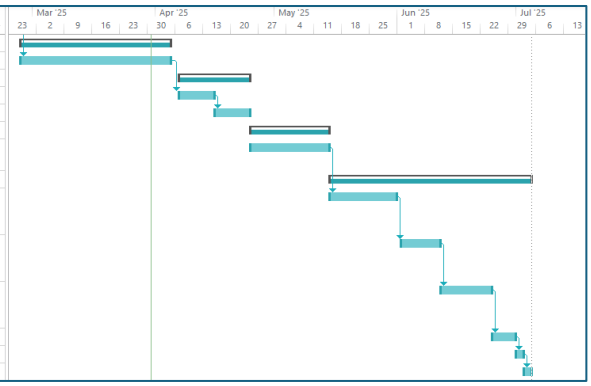
Appendix A

Project Schedule (Gantt Chart)

Task Name	Duration	Start	Finish
Dog Breed Detection using Deep Learning Approach	**195 days**	**Mon 10/7/24**	**Fri 7/4/25**
Sprint 1	**23 days**	**Mon 10/7/24**	**Wed 11/6/24**
Prepare Brief Proposal	7 days	Mon 10/7/24	Tue 10/15/24
Meeting and Discussion with Supervisor	1 day	Wed 10/16/24	Wed 10/16/24
Prepare Full Proposal	7 days	Thu 10/17/24	Fri 10/25/24
Meeting and Discussion with Supervisor	1 day	Mon 10/28/24	Mon 10/28/24
Chapter 1 - Introduction	7 days	Tue 10/29/24	Wed 11/6/24
Sprint 2	**30 days**	**Thu 11/7/24**	**Wed 12/18/24**
Chapter 2 - Literature Review	14 days	Thu 11/7/24	Tue 11/26/24
Meeting and Discussion with Supervisor	1 day	Wed 11/27/24	Wed 11/27/24
Chapter 3 - Requirement Analysis and Design	14 days	Thu 11/28/24	Tue 12/17/24
Meeting and Discussion with Supervisor	1 day	Wed 12/18/24	Wed 12/18/24
Sprint 3	**28 days**	**Thu 12/19/24**	**Mon 1/27/25**
Application Development	28 days	Thu 12/19/24	Mon 1/27/25
Sprint 4	**21 days**	**Tue 1/28/25**	**Tue 2/25/25**
Application Testing	7 days	Tue 1/28/25	Wed 2/5/25
Feedback Acquisation	7 days	Thu 2/6/25	Fri 2/14/25
Feedback Analysis	7 days	Mon 2/17/25	Tue 2/25/25



Task Name	Duration	Start	Finish
Sprint 5	**28 days**	**Wed 2/26/25**	**Fri 4/4/25**
Application Development	28 days	Wed 2/26/25	Fri 4/4/25
Sprint 6	**14 days**	**Mon 4/7/25**	**Thu 4/24/25**
Application Testing	7 days	Mon 4/7/25	Tue 4/15/25
Feedback Acquisation	7 days	Wed 4/16/25	Thu 4/24/25
Sprint 7	**14 days**	**Fri 4/25/25**	**Wed 5/14/25**
Preparation and Submission of First Draft for Chapter 4	14 days	Fri 4/25/25	Wed 5/14/25
Sprint 8	**37 days**	**Thu 5/15/25**	**Fri 7/4/25**
Preparation and Submission of First Draft for Chapter 5, 6 and Abstract for Paper	13 days	Thu 5/15/25	Sat 5/31/25
Preparation and Submission of First Draft for FYP2 Full Report and Paper	8 days	Mon 6/2/25	Wed 6/11/25
Submission of Final Report, Source Code, Installation Kits, User Manual, and Paper for Assessment	9 days	Thu 6/12/25	Tue 6/24/25
FYP Symposium Preparation	4 days	Wed 6/25/25	Mon 6/30/25
FYP Symposium	2 days	Tue 7/1/25	Wed 7/2/25
Project Closing	2 days	Thu 7/3/25	Fri 7/4/25



Appendix B

Sample Questionnaire

Final Year Project: Survey on Functional Requirements for Dog Breed Classification Web Application using Deep Learning Approach

Dear Respondent,

My name is Annastasha Chang See May , and I am an undergraduate student in Network Computing at University Malaysia Sarawak (UNIMAS). For my Final Year Project, I am developing a web application that can identify dog breeds from uploaded images using Artificial Intelligence (AI).

To ensure this application meets the needs of users like you, I kindly request your participation in this short survey. Your feedback on desired features and functionalities will be invaluable in making the application user-friendly and helpful.

All information you provide will be kept strictly confidential and used solely for this research project.

Thank you for your time and valuable input.

Sincerely,

Annastasha Chang See May

Network Computing, FCSIT, UNIMAS

78855@siswa.unimas.my

Section 1: Demographics and Background

1. What is your age group? *

- ◯ Under 18
- ◯ 18 - 25
- ◯ 26 - 35
- ◯ 36 - 50
- ◯ Above 50

2. What is your occupation? *

- ◯ Student
- ◯ Working Professional
- ◯ Retired
- ◯ Other: _____

3. Do you own a dog? *

- ◯ Yes
- ◯ No

4. If yes, how many dogs do you have? *

- ◯ 1
- ◯ 2 - 3
- ◯ More than 3
- ◯ None

5. Are you familiar with identifying dog breeds? *

- ◯ Very familiar
- ◯ Somewhat familiar
- ◯ Not familiar

Section 2: General Features

1. What features should the application provide to make it user-friendly? (Select any that apply) *

- [ ] Intuitive navigation
- [ ] Responsive design
- [ ] Drag and drop functionality to upload image
- [ ] Quick image upload and processing
- [ ] Other: _____

2. Should the application support multiple file formats for image uploads (e.g., JPG, PNG)? *

- ( ) Yes
- ( ) No

3. What visual design style do you prefer for the application? *

- ( ) Minimalistic
- ( ) Modern and colorful
- ( ) Other: _____

4. How important is ease of use when uploading and classifying images? *

- ◯ Very Important
- ◯ Somewhat Important
- ◯ Not Important

5. Would you like to see a step-by-step guide for using the application? *

- ◯ Yes
- ◯ No

6. Do you have experience using image-based classification applications? *

- ◯ Yes
- ◯ No

Section 3: Results Display

1. What format would you prefer for the prediction results? *

- ◯ Text-based results (e.g., breed names and percentages)
- ◯ Visual results with breed images
- ◯ A combination of text and visual results

2. Should the application display more than one possible breed? *

- ◯ Yes, show the top 3 breeds with similarity percentages
- ◯ No, show only the most likely breed

3. How should the similarity percentages be presented? *

- ◯ A numerical percentage (e.g., 85%)
- ◯ A bar graph or chart
- ◯ Other: _____

4. How important is the speed of displaying prediction results? *

- ◯ Extremely important (results should be instant)
- ◯ Moderately important (a few seconds delay is acceptable)
- ◯ Not important (accuracy matters more than speed)

5. What should the application do if the uploaded image does not contain a dog? *

- ◯ Display an error message and ask for a new image
- ◯ Suggest tips for better image uploads
- ◯ Automatically discard and prompt the user to try again

Section 4: User Preferences and Needs

1. Why would you use a dog breed classification web application? (Select any that apply) *

- [ ] For personal interest
- [ ] To identify my dog's breed
- [ ] To learn about other breeds
- [ ] For professional purposes (e.g., vet, breeder)
- [ ] Other: _____

2. What would you expect from a dog breed classification application? (Select any that apply) *

- [ ] Accurate breed identification
- [ ] Quick and efficient processing
- [ ] User-friendly interface
- [ ] Offline functionality for basic features
- [ ] Other: _____

3. What would make you recommend such an application to others? (Select any that apply) *

- [ ] High accuracy in breed classification
- [ ] Easy-to-use interface
- [ ] Fast image upload and processing time
- [ ] No registration required to use the application
- [ ] Other: _____

4. Are there any additional features or functionalities you think should be included in this application? (Select any that apply) *

- [ ] Ability to save and download classification results
- [ ] A section with tips on dog care and training
- [ ] Integration with a database for finding lost dogs
- [ ] Offline mode for basic functionality
- [ ] Other: _____

5. Should the application allow users to save their uploaded images and results for future reference? *

- ( ) Yes
- ( ) No

Appendix C

User Acceptance Questionnaire

Usability Testing Survey – Dog Breed Classification Web Application

Hello! My name is Annastasha Chang See May (78855), a final year Computer Science (Network Computing) student at University of Malaysia Sarawak (UNIMAS). This survey is part of the usability testing for my **Final Year Project** titled ***Dog Breed Classification Web Application***, which uses deep learning to identify dog breeds from uploaded images.

Your feedback will help evaluate the system's ease of use, design, and effectiveness. Kindly answer the questions below honestly. Your responses will remain confidential and will only be used for academic purposes.

Rest assured, your responses will be kept confidential and will be used solely for the purposes of this Final Year Project. No data will be shared with third parties. Should you have any questions or require further information, feel free to email at **78855@siswa.unimas.my**.

I sincerely appreciate your time and support. Your participation is valuable and contributes significantly to the success of this project.

Warm regards,
Annastasha Chang See May

annastashachang@gmail.com Switch account

Not shared

* Indicates required question

Section 1: User Demographics

What is your age? *

- ◯ Under 18
- ◯ 18 - 25
- ◯ 26 - 35
- ◯ 36 - 50
- ◯ Above 50

What is your occupation? *

- ◯ Student
- ◯ Working
- ◯ Retired

Have you used a dog breed classification system before? *

- ◯ Yes
- ◯ No
- ◯ Maybe

Next Clear form

Section 2: Usability and Functionality

Instructions: Please rate the following statements based on your experience using the system.

How easy was it to navigate the web application? *

	1	2	3	4	5	
Very difficult	◯	◯	◯	◯	◯	Very easy

How would you rate the speed of image upload and prediction? *

	1	2	3	4	5	
Very slow	◯	◯	◯	◯	◯	Very fast

Was the prediction result understandable and clearly presented? *

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

Do you find the information provided about each dog breed useful? *

	1	2	3	4	5	
Not useful at all	○	○	○	○	○	Very useful

How accurate do you think the predicted result was? *

	1	2	3	4	5	
Very inaccurate	○	○	○	○	○	Very accurate

Back Next Clear form

Section 3: Overall Experience

Instructions: Please rate the following statements based on your experience using the system.

How satisfied are you with the overall experience of using the application? *

	1	2	3	4	5	
Very dissatisfied	○	○	○	○	○	Very satisfied

Would you recommend this web application to others? *

- ○ Yes
- ○ No
- ○ Maybe

Back Submit Clear form