SUBMITTED BY
Saqib-Ur-Rehman
01-235192-091
Hira Majeed
01-235192-029

Underwater Object Detection

Bachelor of Science in Computer Science

Supervisor: Ms. Maryam Khalid Multani

Department of Computer Science
Bahria University, Islamabad

May 2023

Certificate

We accept the work contained in the report titled "Underwater Object Detection", written by Mr. Saqib-Ur-Rehman AND Ms. Hira Majeed as a confirmation to the required standard for the partial fulfillment of the degree of Bachelor of Science in Computer Science.

Approved by . . . :

Supervisor: Ms. Maryam Khalid Multani (Assistant Professor)

____

Internal Examiner: Name of the Internal Examiner (Title)

____

External Examiner: Name of the External Examiner (Title)

____

Project Coordinator: Ms. Zubaria Inayat (Sr. Lecturer)

____

Head of the Department: Dr. Arif ur Rahman (Sr. Associate Professor)

____

May 3rd, 2023

Abstract

Robotics are very expensive nowadays so every new researcher who wants to explore the underwater environment cannot afford such expensive robots, sonars, and remote vehicles in order to detect the underwater species. The underwater environment is one of the most challenging conditions for object detection because the underwater environment is harsh due to the amount of light present in the underwater. The images captured in an underwater environment are probably blurry, and low contrast due to this it is difficult for the human eye to detect the underwater objects/species. This project aims to provide a platform for new researchers to explore the underwater environment. With the help of this application, researchers upload captured image underwater and the application provides enhance image remove low contrast and blur issues, and provide a clear image to the user along with the detected objects or species. This application also provides a description of the species. This application uses a deep learning model which helps to accurately identify underwater objects. This Underwater object detection application provide a valuable tool for researchers to better understand the underwater environment.

Acknowledgments

In the name of Allah Almighty, the Most Gracious and the most merciful, All praises to ALLAH for the strengths and the blessings. We would like to declare our deeper acknowledgment and gratitude towards Mr.Abdul Hanan for his expert guidance, provision of detailed information regarding the project. His motivations to work hard and not to given up really inspired us and we are honored to have him at our side. There is no alternate to the prayers, affections, resignation of Parents. Even in tough circumstance their prayers were always with us. we are also thankful to our Supervisor Ms.Maryam Khalid Multani and the FYP project coordinator Ms.Zubaria Inayat for the exchange of information regarding project milestones and meetings.

SAQIB-UR-REHMAN
HIRA MAJEED
Islamabad, Pakistan

May 2023

“Don’t feel entitled to anything you didn’t sweat and struggle for.”

Marian Wright Edelman

Contents

List of Figures

List of Tables

Acronyms and Abbreviations

FR	Functional Requirement
GUI	Graphical user interface
API	Application Programming Interface
YOLO	You Only Look Once
RDBMS	Relational Database Management System
Max RGB	Maximum Red, Green, Blue
CNN	Convolutional Neural Networks
VGG	Visual Geometry Group
SVM	Support Vector Machines
RCNN	Region-based Convolutional Neural Network
RPN	Region Proposal Network
ROVs	Remotely Operated Vehicles
ER Diagram	Entity-Relationship Diagram
RAM	Random Access Memory

Chapter 1

Introduction

1.1 Project Background/ overview

The ocean covers a large part (about 70 percent) of the Earth's surface and is home to valuable natural resources. People have always been interested in exploring and using these resources. With advances in technology, we can now study and exploit the ocean more efficiently. However, exploring the ocean can be challenging. The underwater environment is often rough and unpredictable. There are many factors that make it difficult to capture clear images underwater for example, the water can be murky or cloudy which reduces visibility. The lighting underwater is also uneven, either because of sunlight or artificial sources making it hard to see clearly. Additionally, the colors underwater are not as vibrant as on land which makes it harder to distinguish objects. The background underwater can be complicated with corals, rocks, and plants which can confuse the camera or imaging system.

The light attenuation in water makes it loss the energy rapidly which results in color depletion. The presence of suspended organic and inorganic particles in water also contributes to light beams randomly reflecting and deflecting before entering the camera sensor resulting in a lower contrast image. Color shifting and degraded contrast make the underwater image segmentation process more difficult and challenging. The amount of light contained within the water is always less than the amount of light present on the water's surface. As a result, images obtained underwater have generally poor visual quality. The lack of light underwater is typically caused by two unavoidable factors. One , light loses its true intensity underwater and two, the chances of light scattering within the water are quite high. The color distortion and illumination of the underwater scene visibility are the immediate effects of this insufficient amount of light [1].

The primary goal of the "Underwater Object Detection" project is to create a system that can accurately identify various objects in the underwater environment, including jellyfish, starfish, Stingray, and other aquatic animals. The underwater environment is one of the most challenging conditions for object detection. The increasing demand for vision-based applications enhances the importance of camera-based object detection methods in underwater scenarios. The modern world is enclosed with gigantic masses of digital visual applications. Many image analysis techniques exist to analyze and understand this huge sea of visual information like image processing and deep learning. Deep learning is a method that automatically detects the object needed to be detected or classified using the provided raw data.

The researchers have underwater images and want to detect objects that are present in the pictures. It will help them to identify and locate one or more effective targets from an image. Secondly, it will also help to detect the objects captured by researchers or photographers which are probably blurred, have uneven illumination, and have low contrast due to harsh underwater environments. This application takes picture from the user then enhance the picture first and detects the objects in it along with the description. We are using image processing methods to improve the underwater image quality to satisfy the requirements of the human visual system and machine recognition has gradually become a hot issue. Although the ocean environment is complex, many unfavorable factors, such as the scattering and absorption of light by water, and the underwater suspended particles have serious interference with image quality [2]. Underwater object detection systems that provide real-time detection like sonar and other technologies. For example, sonar is used to detect objects and provide information about their location, while computer vision is used to classify objects and provide more detailed information about their shape and appearance.

1.2 Problem Description

Our application aims to assist underwater researchers in their efforts to study and conserve marine species by providing an automated object detection system. The researchers are often faced with the challenge of identifying and classifying various species in underwater environments, which can be time-consuming and prone to human error. Therefore, our goal is to develop an efficient and accurate solution that can aid in this process.

The specific problem we are addressing is the detection of underwater species/ object in images captured during research expeditions. These species can include fish, jellyfish, stingray, sharks, puffin, starfish and penguin. Currently, researchers rely heavily on manual observation and identification, which can be subjective, labor-intensive, and hindered by limited human capacity. We will train a model to recognize and localize different species within images, allowing researchers to quickly identify species. The system will provide bounding box annotations and class labels for each detected species, along with the description of every detected species.

1.3 Project Objective

"To design an application for researchers where they could detect and get the information of an underwater object by photo".

1.4 Project Scope

Our application takes only captured pictures from the gallery. For now, we will detect seven Classes = [’fish’, ’jellyfish’, ’penguin’, ’puffin’, ’shark’, ’starfish’, ’stingray’] of underwater objects and also provide description related to detected objects to help user to understand them easily. We will use the aquarium-pretrained dataset as well [3]

Chapter 2

Literature Review

This chapter presents an overview of methods and techniques of previous work done related to underwater object detection.

2.1 Related Work

Fish species identification is mainly performed by morphological identification of gross anatomical features of the whole fish. However, the increasing presence on markets of new little-known species makes morphological identification of species difficult. Fish APP, a cloud-based infrastructure for fish species recognition. Fish APP is composed of a mobile application developed for the Android operating system enabling the user to shot pictures of a whole fish and submit them for remote analysis and a remote cloud-based processing system that implements a complex image processing pipeline and a neural network machine learning system able to analyze the obtained images and to perform classification into predefined fish classes. [4].

2.2 Underwater Image Processing and Analysis

A review of existing relatively mature and representative underwater image processing models, which are classified into seven categories: enhancement, fog removal, noise reduction, segmentation, salient object detection, color constancy and restoration [5].

2.2.1 Underwater Image Enhancement

The primary objective of underwater image enhancement is to enhance the visual quality and applicability of images captured in underwater environments. The ultimate goal is to

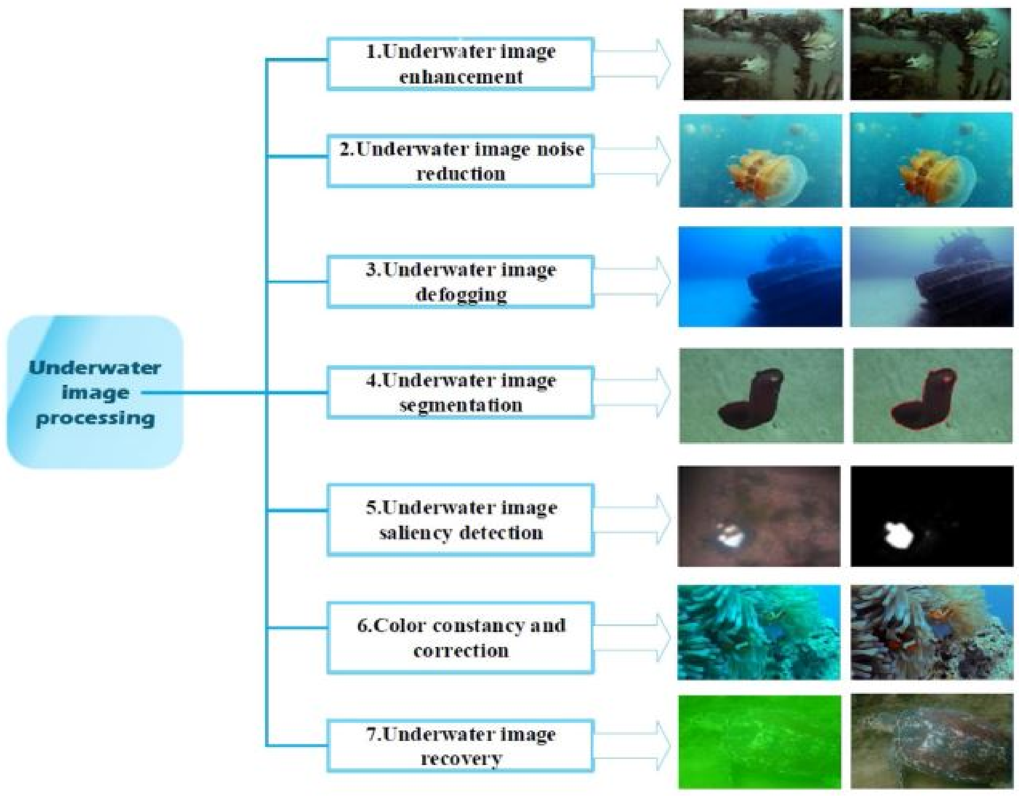

Underwater image processing
1.Underwater image enhancement
2.Underwater image noise reduction
3.Underwater image defogging
4.Underwater image segmentation
5.Underwater image saliency detection
6.Color constancy and correction
7.Underwater image recovery

Figure 2.1: Underwater Image Processing and analysis

improve the overall appearance of underwater images and highlight specific image characteristics that are relevant to the intended application. To achieve this goal, various image enhancement techniques are employed. These techniques may include both traditional and machine learning-based approaches. Traditional techniques include methods such as contrast adjustment, color correction, and sharpening. Machine learning-based techniques, on the other hand, use advanced algorithms to learn the relationship between input images and high-quality reference images.

2.2.2 Underwater Noise Reduction

During the process of capturing and transmitting underwater images, various noises are commonly produced due to factors such as the camera/sensor equipment and harsh underwater environments. These noises can significantly degrade the quality of the images, making them difficult to interpret and analyze.The camera/sensor equipment used in underwater imaging can produce electronic noise, which can appear as speckles or streaks in the images. The harsh underwater environment can also contribute to noise in the form of bubbles, particles, and other disturbances. Various traditional methods have been developed to reduce noise in underwater images. Like filtering, in this technique it removes or reduces unwanted frequencies present in the image. Another technique is denoising, which focuses on eliminating random noise from the image. While these methods are frequently effective in reducing noise levels, they may also result in a reduction of important image details.

2.2.3 Underwater Noise Defogging

Underwater noise defogging is a technique that aims to enhance the visibility of underwater images by reducing the effect of haze, turbidity, and other distortions caused by light scattering and absorption in the water. By reducing the effect of underwater noise and improving the visual clarity of the image, underwater noise defogging can enable better analysis and interpretation of underwater imagery, supporting a wide range of applications in different fields such as oceanography, marine biology, and underwater exploration.

2.2.4 Underwater Image Segmentation

Underwater image segmentation is a technique used to extract meaningful information from underwater images by dividing them into distinct regions or segments based on their visual characteristics. The goal of underwater image segmentation is to identify and isolate objects or regions of interest in the image, such as marine life, underwater structures, or geological features. For example, you have an underwater image containing various underwater animals and plants as well as some background scenery. By segmenting the image, you can isolate the different regions, such as the animals and plants and analyze them separately. This can help in identifying and studying specific marine species or monitoring changes in the underwater environment over time [6].

2.2.5 Underwater Saliency Detection

Underwater saliency detection is used to identify the most visually prominent regions or objects in an underwater image. The goal of underwater saliency detection is to help identify and highlight the most important features or regions in an image, making it easier to analyze and understand. In an underwater image, salient objects or regions could include Underwater life, structures, or geological features that stand out from the background. Underwater saliency detection algorithms typically rely on features such as color, texture, and contrast to identify and highlight these prominent regions.

2.2.6 Color Constancy and Correction

Color constancy and correction are important in underwater image processing that help to maintain color consistency and improve the visual quality of images captured in underwater environments [7]. Color constancy refers to the ability to maintain color consistency in an image despite changes in lighting conditions or other environmental factors. In underwater imaging, the effects of light scattering and absorption can cause colors to shift, making it difficult to accurately capture the true colors of objects in the scene. Color correction, on the other hand, involves adjusting the colors in an image to more accurately reflect the true colors of the objects in the scene. This is typically done by identifying a reference color, such as the color of a white or gray object in the scene, and adjusting the colors in the image based on that reference.

2.2.7 Underwater Image Restoration

Underwater image restoration is a process of recovering or reconstructing degraded images that are caused by adverse factors in the complex underwater environment [8]. These factors include camera and object motion, scattering of light, turbulence, distortion, spectral absorption, and attenuation, among others.

2.3 Region- Based CNN(CNN)

Girshick presented the Regional Based Convolution Neural Network (RCNN) in 2014, which is essentially a combination of the CNN Model and the Region Proposal Network. RCNN creates a set of candidate regions, or "proposals," during the region proposal stage. These proposals are then sent to the classification stage, where the network examines each one to see if it contains an object of interest. The RCNN uses features extracted from the proposals to classify the objects in the image and refine the bounding boxes around the objects during the classification stage. If an object is discovered in a proposal, the RCNN generates a label for the object as well as a refined bounding box around it. This process is repeated for each proposal until all objects in the image have been identified and localized. The RCNN architecture provides several benefits for underwater object detection, including high accuracy, fast processing speed, and the ability to handle the challenges of underwater imaging. The RCNN architecture can handle variations in object scale and aspect ratio by generating multiple proposals and examining each one, making it a powerful tool for underwater object detection. The RCNN model does not work on a large number of regions because it would be computationally expensive and result in lower accuracy. The region proposal stage's goal is to generate a small set of high-quality regions that are likely to contain objects, rather than a large number of regions that may contain many false positives. The following are the steps taken in RCNN for object detection:

2.3.1 Pre-trained Convolutional Neural Network (CNN)

The RCNN begins with a CNN that has already learned to extract useful features from images, such as VGG or ResNet[9].

2.3.2 Feature Extraction

The RCNN uses the pre-trained CNN to extract features from the image at this stage. The features are then forwarded to the classification stage.

2.3.3 Regional Proposal Generation

The RCNN generates a set of candidate regions or "proposals" in the image that may contain objects of interest during this stage. Typically, this step is carried out using techniques such as Selective Search, Edge Boxes, or Faster RCNN.

2.3.4 Feature Pooling

The RCNN pools features from the pre-trained CNN for each region proposal at this stage. The pooled features are then forwarded to the classification stage.

2.3.5 Classification

At this stage, the RCNN employs a classifier, such as Support Vector Machines (SVM) or Logistic Regression, to categorize the objects in the image and refine the bounding boxes around the objects. To learn the features that distinguish different objects from one another, the RCNN network is trained on a large dataset of annotated images.

2.3.6 Object Localization

After classification, the RCNN generates a label for each object and refines the bounding boxes around the objects. This stage is used to locate the objects in the image, and the bounding boxes that result can be used for further analysis or tracking.

2.3.7 Non-Maximum Suppression (NMS)

Following object localization, the RCNN employs non-maximum suppression (NMS) to eliminate overlapping bounding boxes and ensure that each object is detected only once.

2.4 Faster RCNN

Faster RCNN is another improvement over Fast RCNN and RCNN for object detection. Faster RCNN, like Fast RCNN, integrates the region proposal generation and feature extraction steps into a single network. Unlike Fast RCNN, however, the region proposals are generated by a separate subnetwork called the Region Proposal Network (RPN), which allows for more flexible and efficient region proposal generation. The same convolutional neural network as in Fast RCNN is used for feature extraction and object classification.

Faster RCNN has several advantages over Fast RCNN, including improved speed and accuracy, as well as more flexible and efficient region proposal generation. Faster RCNN has been widely adopted and is still one of the most widely used object detection frameworks, particularly in applications where real-time performance is not required.

2.5 Networks based on YOLO (You Only Look Once) framework

YOLO is a real-time object detection system built on a single convolutional neural network. In contrast to RCNN and Fast RCNN, YOLO divides the input image into a grid of cells, each of which predicts multiple bounding boxes for potential objects. For each bounding box, the network predicts the class probabilities, bounding box coordinates, and objectness scores. This enables efficient real-time object detection and reduces the computational overhead associated with the region proposal stage. YOLO performs better than RCNN and Fast RCNN in terms of real-time performance, single-shot detection, and high accuracy. However, YOLO has some drawbacks, including reduced accuracy for small objects and sensitivity to hyper-parameter selection [10] Overall, YOLO is a well-known and widely used framework for real-time object detection and it continues to be one of the most widely used and researched frameworks in the field. The difference of Yolo version is shown in table 2.1

Table 2.1: Yolo Versions

Features	**YOLO v1**	**YOLO v2**	**YOLO v3**	**YOLO v4**
Object Detection Approach	Single-Shot Multi-Box Detection	Improved Version of v1	Improved Version of v2	Improved Version of v3
Architecture	Darknet-19	Darknet-19	Darknet-53	CSPNet
Anchor Boxes	Yes	Yes	Yes	Yes
Batch Normalization	No	Yes	Yes	Yes
Intersection Over Union (IOU)	Used to determine accuracy	Used to determine accuracy	Used to determine accuracy	Used to determine accuracy
Training Speed	Slow	Faster than v1	Faster than v2	Faster than v3
Accuracy	Lower	Higher than v1	Higher than v2	Higher than v3
Memory Efficiency	Lower	Higher than v1	Higher than v2	Higher than v3

Chapter 3

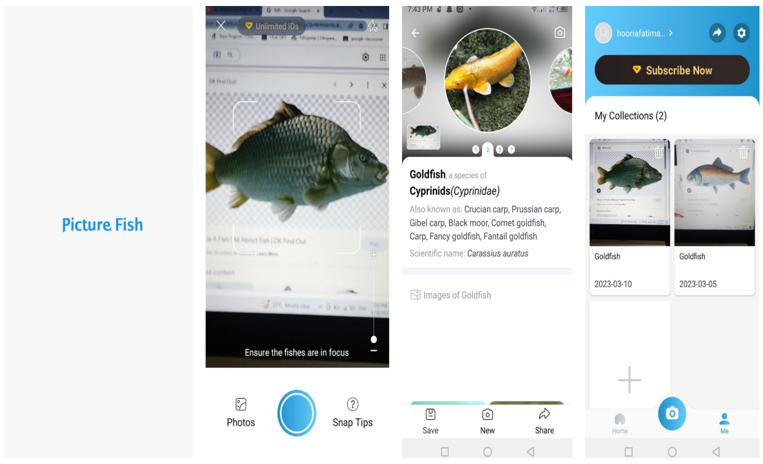
Requirement Specifications

This chapter briefly introduces the existing systems followed by an overview of the proposed system. Describe in detail the requirement specifications of the system.

3.1 Existing System

3.1.1 Picture Fish

Picture fish is a mobile application that use image recognition technology to assist users in identifying various fish species. Anglers, divers, and other enthusiasts who want to learn more about the fish they encounter in the wild may find this app useful. But the drawback of using this application is that, in order to accurately identify fish species, it relies on clear and detailed images. If the image of the fish is too blurry or unclear, the app may struggle to make an accurate identification [4].



Goldfish, a species of Cyprinids(Cyprinidae)
Also known as: Crucian carp, Prussian carp, Gibel carp, Black moor, Comet goldfish, Carp, Fancy goldfish, Fantail goldfish
Scientific name: *Carassius auratus*
Images of Goldfish

Figure 3.1: Picture fish Application

3.1.2 Sound Navigation and Ranging (SONAR)

Sonar (Sound Navigation and Ranging) is used for underwater object detection. Sonar is an acoustic imaging technology that uses sound waves to detect underwater objects. It works by emitting a sound pulse into the water, which then travels through the water and bounces off objects in its path. The reflected sound waves, or echoes, are then detected by the sonar transducer and used to create an image of the underwater environment. The sonar transducer emits a sound pulse into the water. The sound wave travels through the water and bounces off any objects in its path. They provide real time underwater object detection [11].

3.1.3 Remotely Operated Vehicles

Remotely Operated Vehicles (ROVs) are underwater robots that can be used for object detection and a variety of other tasks. ROVs are equipped with cameras, lights, and other sensors that can capture images, video, and data about the underwater environment. They are controlled remotely from the surface and can be used to inspect ships, pipelines, and other underwater structures. ROVs are especially useful for deep-water exploration, where human divers are not practical. They can reach depths that would be dangerous for divers and provide high-resolution images and data about the underwater environment. Additionally, ROVs can be equipped with specialized sensors and tools for specific tasks, such as collecting samples, performing repairs, and deploying instruments. They provide real time detection in underwater environment [12].

In general, academic researchers and government agencies may be able to obtain funding to buy or rent sonar and ROV equipment for research purposes. Smaller research organisations or individual researchers, on the other hand, may have fewer resources and thus cannot afford such research equipment.

Figure 3.2: Remotely Operated Vehicles

3.2 Proposed System

The proposed system aims to develop an application in which user provide underwater pictures which are probably blurred, low contrast and this application will help the researchers to firstly enhance the pictures and then labelled the object present in the picture and then generate the final outcome which will be cleared, and the object are properly labelled and the user will easily download the final result and it also provide the description of labelled objects/underwater species. This system is especially designed for the researcher. For non-real-time underwater object detection, there are several image-based techniques that can be used: These techniques can be combined with deep learning algorithms such as Yolo to improve the accuracy and performance of object detection. Additionally, pre-processing techniques such as color correction, denoising can be used to improve the quality of the underwater images before feeding them into the object detection system.

3.3 Requirement Specification

3.3.1 Functional Requirement

Functional requirements specify what inputs are given to the system, what output is produced, and how the system responds to input. Our system's functional requirements are as follows:

FR 1: Upload Image

The user should be able to upload the image by using the application's "upload Image" button. The system should allow the upload of common image file types like JPG and PNG. If there is an issue with the upload process, such as a file size that exceeds the maximum allowed limit or an unsupported file type, the system should display appropriate error messages. The uploaded image should be displayed in the application's appropriate location.

FR 2 :Enhance Image

When user upload an image from the gallery then it will send to Yolo model for detection but before detection image enhancement techniques will apply on image and then send to Yolo model for detection.

FR 3: Download Image

The download image feature should be accessible to the user from a prominent location within the application. The system should be able to download common image file types such as JPG and PNG. The downloaded image should be securely stored and only accessible to authorised users. If there is a problem with the download process, such as a file that is no longer available or cannot be downloaded, the system should display appropriate error messages.

FR 4: View Generated Image

When the Yolo model finally detected the objects in the image then it will send back to application using Flask API and show it on the user screen so that user can view it and download it into gallery.

3.3.2 Non-functional Requirements

Non-functional requirements are quality characteristics that you include in your system. These specifications define the system's performance. These are the critical parameters for improving system performance. Our system has several non-functional requirements, which are listed below.

3.3.3 Availability

- The application should be available 24/7 with minimal downtime or disruptions.
- The system should be scalable so that it can handle increased traffic and usage without compromising availability.
- The system should have a maintenance schedule in place to reduce downtime and user impact.

3.3.4 Reliability

- The application must be simple to maintain.
- Be available for as much time as possible and be dependable.
- All contents and components must be displayed correctly and in their designated location on the application.

3.3.5 Security

- Mobile applications handle sensitive user data, including personal information. It is crucial to prioritize user privacy and prevent unauthorized access by implementing robust security measures in the application's design.
- To protect user privacy and prevent unauthorised access, application must be designed with strong security measures.
- The system must follow all the latest security standard, help safeguard sensitive user data, maintain user privacy, and mitigate the risks associated with unauthorized access or data breaches.

3.3.6 Performance

- Mobile applications need to be responsive and perform well under various network conditions and device specifications.

- This includes app startup and loading times, as well as responsiveness to user actions.

- The performance of a system can be influenced by the firebase and Yolo model on which it runs.

3.3.7 Maintainability

- The application's code should be modular and organized, facilitating ease of maintenance and future enhancements.

- The application should have comprehensive and up-to-date documentation, including technical specifications, APIs, and user guides.

- The application should be designed in a way that enables effective testing, including unit testing, integration testing, and performance testing.

3.3.8 Compatibility

- Mobile applications must work with a wide variety of devices, operating systems, and network conditions.

- It require testing and development to ensure that the app works properly across multiple platforms and environments.

3.4 Use Cases

A use case is several actions or events which define the interaction between the system and actors to achieve a desire goal. The actors in this use-cases are the persons/entities who will be using the system.

3.4.1 Sign-up Usecase

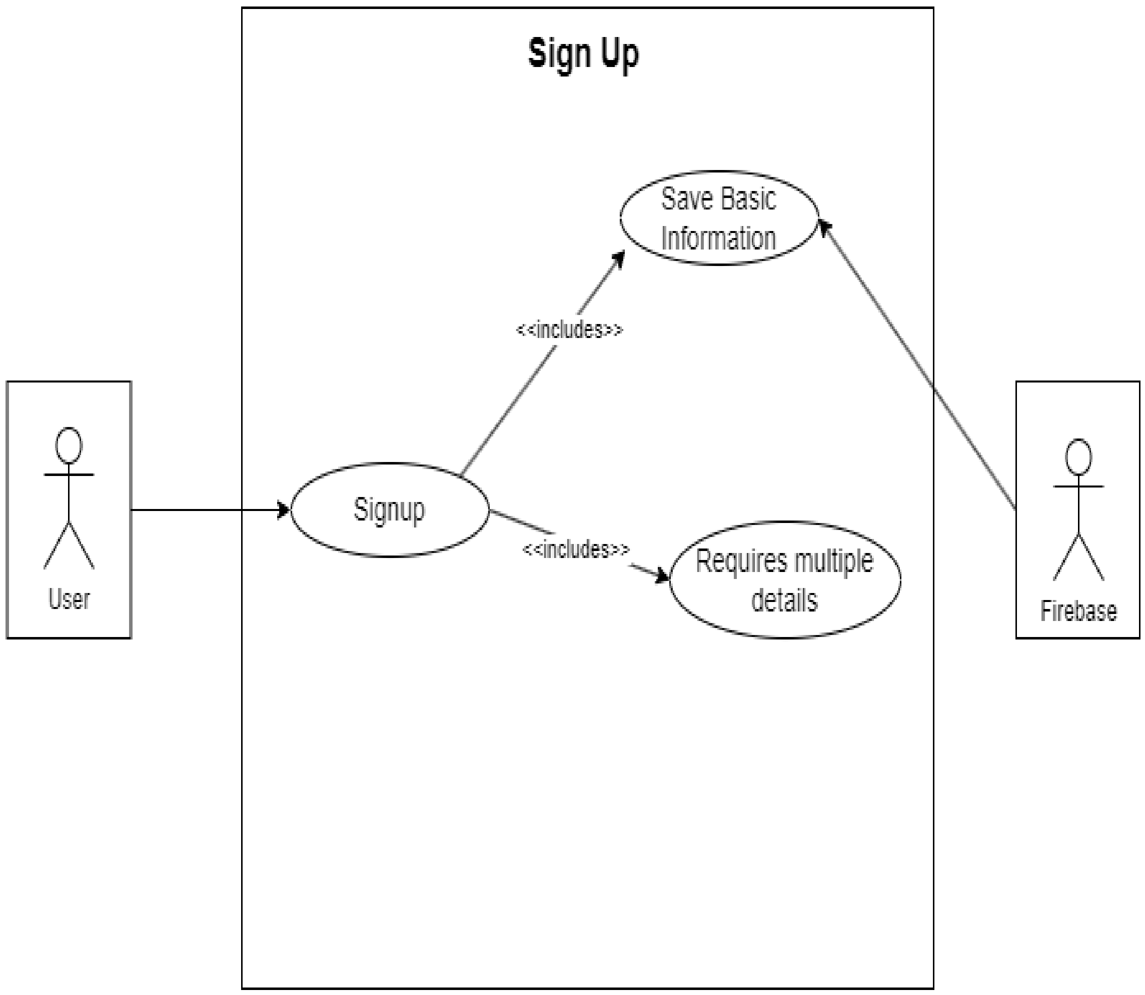


Figure 3.3: Sign-up Usecase

3.4.2 Sign-up Usecase Table

Sign-up usecase table as shown in table 3.1.

Table 3.1: Signup usecase table

Use Case ID	UD001
Use Case Name	Sign Up
Actor	User
Data	User registration details (name, email, company, researcher).
Precondition	The user does not have an existing account on the system.
Steps/Description	a. The user navigates to the registration page. b. The system presents a registration form to the user. c. The user fills in the required registration details, such as name, email, company, researcher. d. The user submits the registration form. e. The system verifies the entered data for completeness and validity. f. If the data passes validation, the system creates a new user account and stores the provided information.
Alternative Flow of Events	• If the user enters an email address that is already associated with an existing account: a) The system displays an error message indicating that the email is already in use. b) The user is prompted to either log in with the existing account or provide a different email address. • If the user enters invalid or incomplete data: a) The system highlights the fields with errors and displays error messages specifying the issues. b) The user corrects the errors and resubmits the registration form.
Post Condition	The user successfully creates a new account and gains access to the system.
Comment	- - - - -

3.4.3 Login Usecase

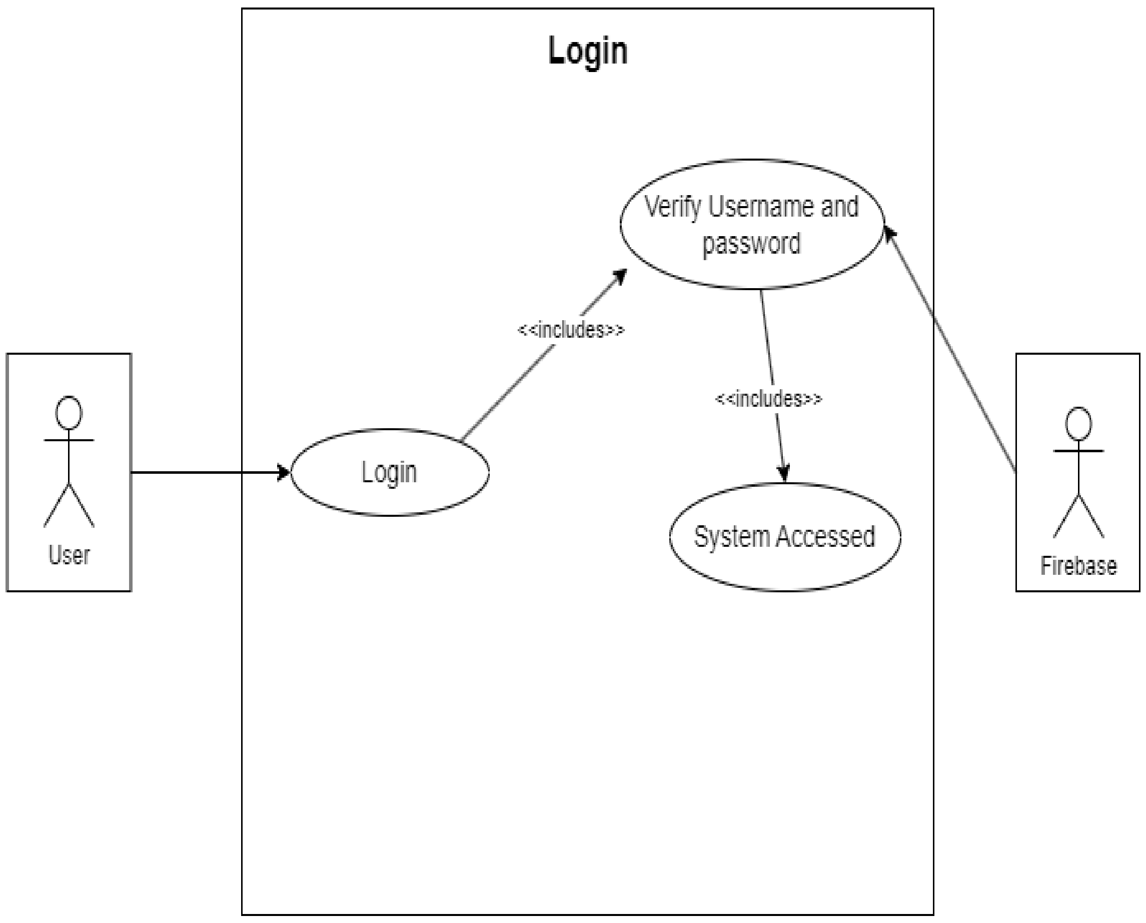


Figure 3.4: Use Case 2: Login

3.4.4 Login Usecase Table

Login usecase table is shown in table 3.2

Table 3.2: Login usecase table

Use Case ID	UD002
Use Case Name	Login
Actor	User
Data	User credentials (Email).
Precondition	The user has registered an account on the system.
Steps/Description	a. The user navigates to the login page. b. The system presents a login form to the user. c. The user enters his/her email into the respective field. d. The user submits the login form. e. The system verifies the provided credentials against the stored user data. f. If the credentials are valid, the system grants access to the user and proceeds to the main page.
Alternative Flow of Events	• If the credentials are invalid, the system displays an error message indicating the login failure. • If the user does not have an account: a. The user clicks on the "Sign Up" button. b. The system redirects the user to the registration page. c. The user fills in the required registration details. d. The user submits the registration form. e. The system creates a new account for the user and proceeds to the main page.
Post Condition	The user is logged into the system and granted access to the main functionality.
Comment	- - - - - -

3.4.5 Upload and View Image Usecase

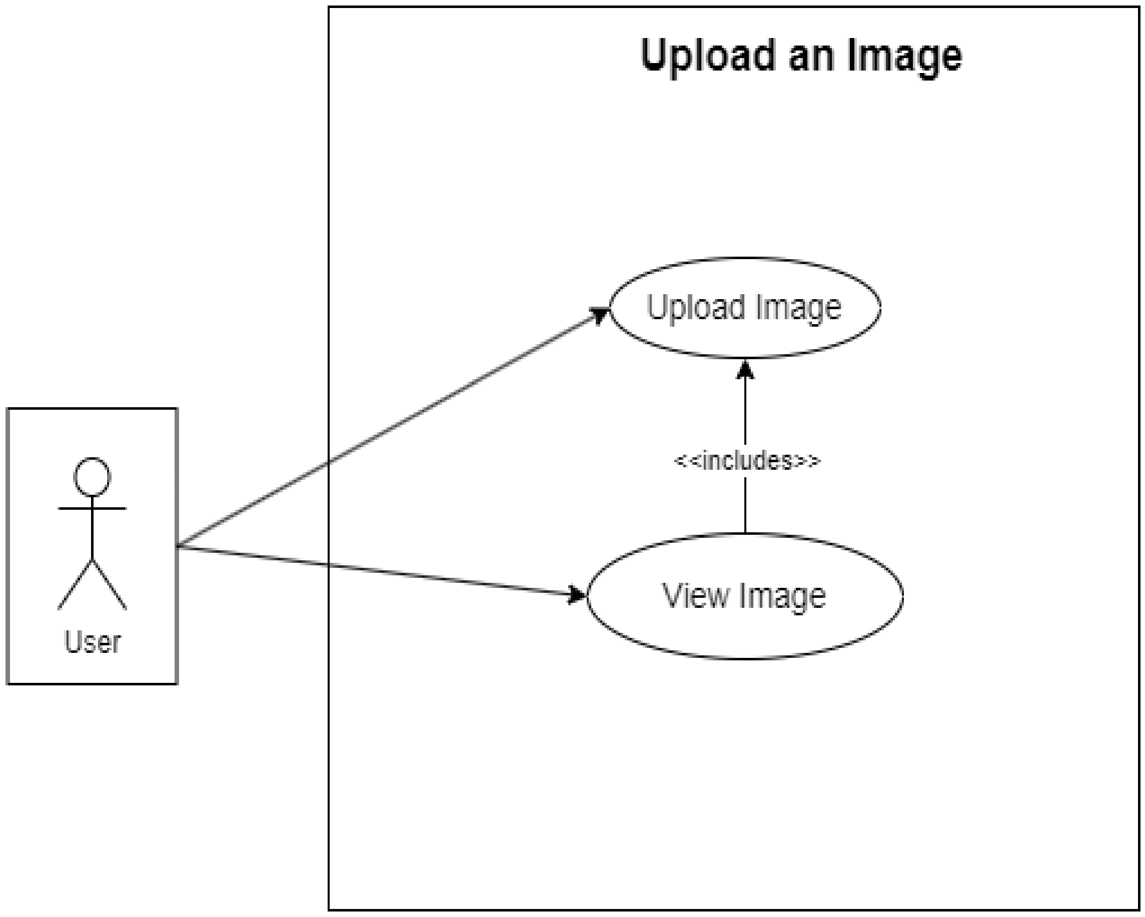


Figure 3.5: Use Case 3: Upload and view image

3.4.6 Upload Image Usecase Table

Upload Image Usecase table is shown in table 3.3.

Table 3.3: Upload image usecase table

Use Case ID	UD003
Use Case Name	Upload an Image.
Actor	User
Data	Image file
Precondition	The user is logged into the system.
Steps/Description	a. The user navigates to the upload page or section. b. The system presents an interface for uploading an image. c. The user selects the image file from their local device using a file picker. d. The user confirms the selection. e. The system verifies that the selected file is a valid image file format (e.g., JPEG, PNG). f. If the file passes validation, The system updates the user interface to display the uploaded image.
Alternative Flow of Events	• If the selected file is not a valid image file format: a. The system displays an error message indicating that only specific image file formats are allowed. b. The user is prompted to select a different file. • If the user cancels the image selection: a. The user closes the file picker. b. The system returns to the previous state without uploading an image.
Post Condition	The image is successfully uploaded and visible to user on the screen.
Comment	- - - - -

3.4.7 Detected and Save Image Usecase

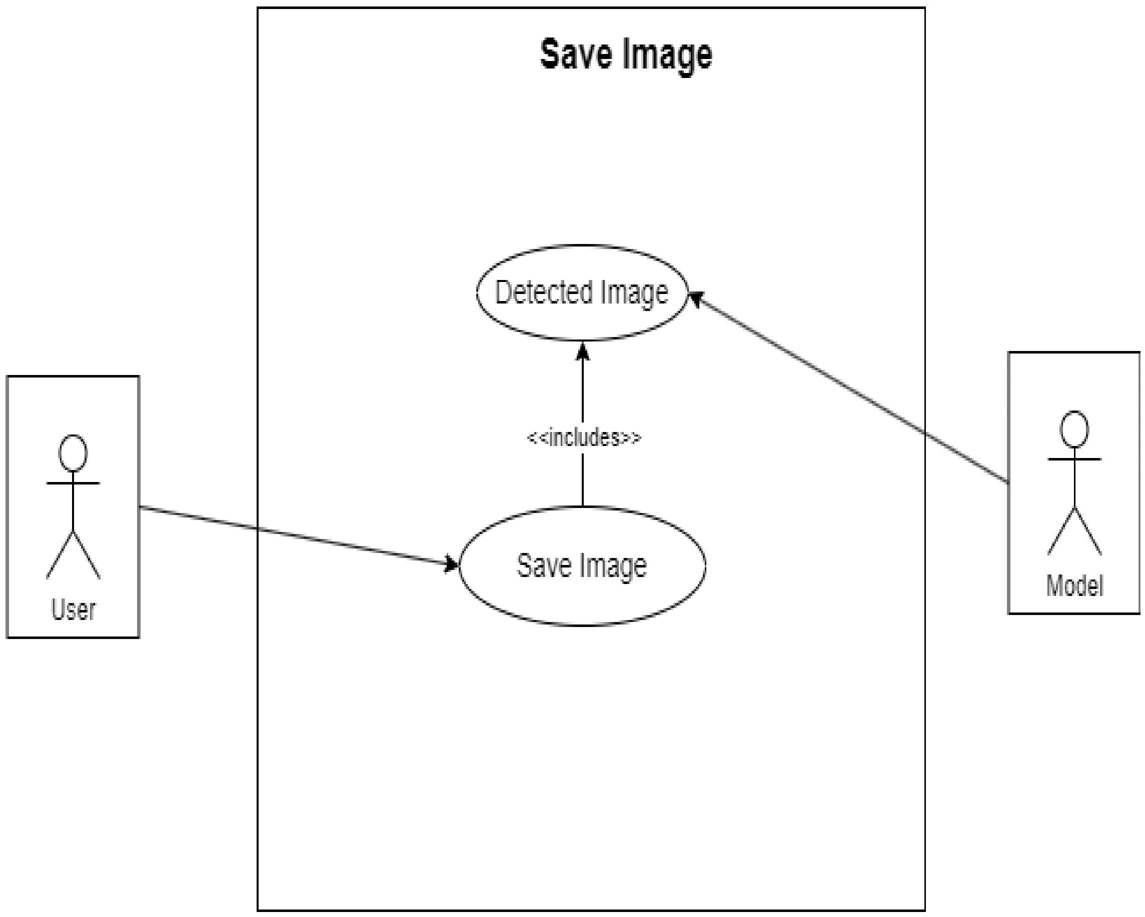


Figure 3.6: Use Case 4: Detected and save image

3.4.8 Detected Image Usecase Table

Detected Image usecase table is shown in table 3.4.

Table 3.4: Detected Image usecase table

Use Case ID	UD004
Use Case Name	Show Detected Image and Save It.
Actor	User
Data	Uploaded image, detected image with objects.
Precondition	The user has uploaded an image and the YOLOv5 model should be running properly.
Steps/Description	a. The system receives the uploaded image and passes it to the YOLOv5 model for object detection. b. The YOLOv5 model processes the image and identifies the objects present in the image. c. The system receives the detected image from the YOLOv5 model. d. The system displays the modified image with the detected objects to the user on their screen. e. The user has the option to save the displayed image with the detected objects. f. If the user chooses to save the image, the system provides a "Save" button. g. The user clicks on the "Save" button. h. The system saves the displayed image with the detected objects to a specified location.
Alternative Flow of Events	• If the YOLOv5 model fails to detect any objects in the uploaded image: a. The system displays a message indicating that no objects were detected. b. The user can choose to upload a different image or proceed without saving. • If there is an error or exception during the image display or saving process: a. The system displays an error message indicating the issue. b. The user can try again.
Post Condition	The user is able to view the uploaded image with detected objects and save it successfully.
Comment	- - - - - - -

3.4.9 View Detail Usecase

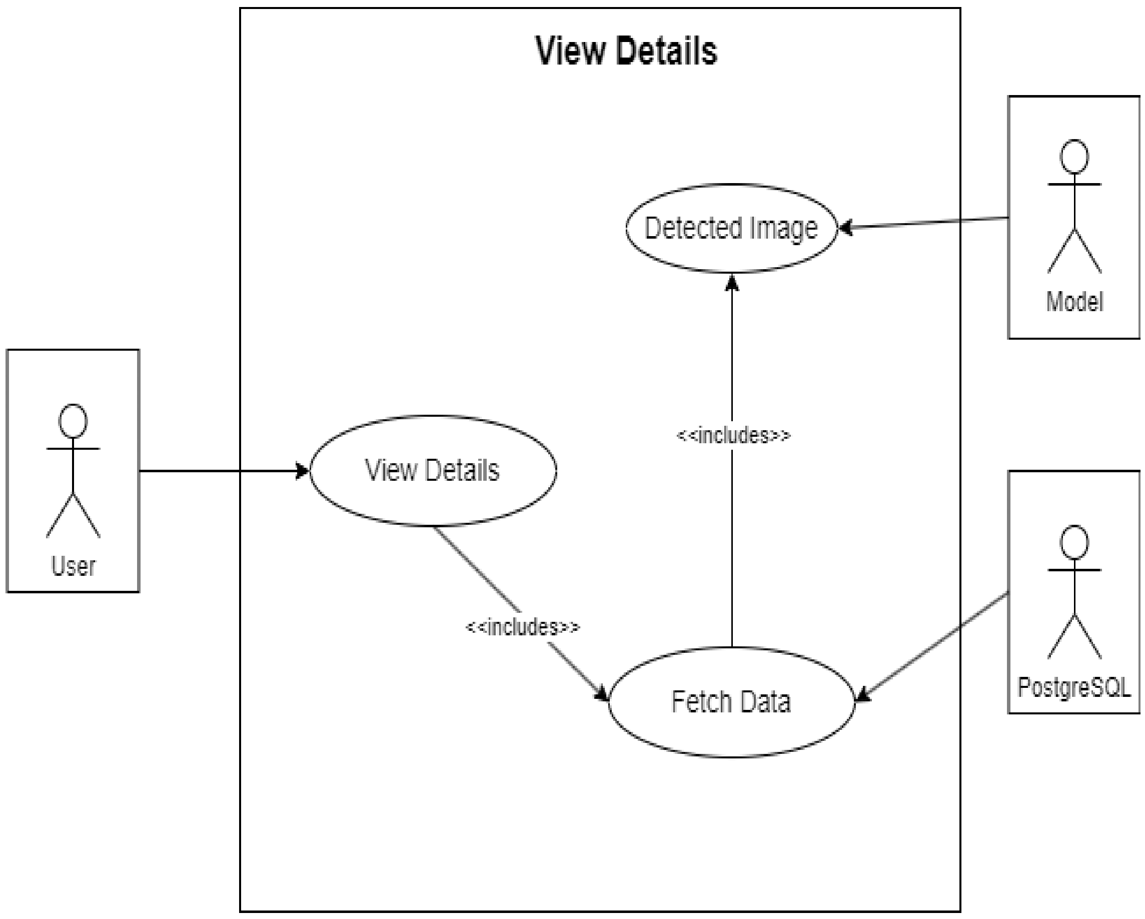


Figure 3.7: Use Case 5: View Detail

3.4.10 View detail Usecase Table

View detail usecase table is shown in table 3.5.

Table 3.5: View detail usecase table

Use Case ID	UD005
Use Case Name	View Details
Actor	User
Data	Uploaded image, detected objects, object details from the PostgreSQL database.
Precondition	The user has uploaded an image, the YOLOv5 model has successfully detected objects in the image, and the object details are stored in a PostgreSQL database.
Steps/Description	a. The system receives the uploaded image and passes it to the YOLOv5 model for object detection. b. The YOLOv5 model processes the image and identifies the objects present in the image. c. The system queries the PostgreSQL database using the detected object information to fetch additional details about the objects. d. The system retrieves the object details from the database. e. The system displays the object details, along with the uploaded image and the detected objects, on the user's screen.
Alternative Flow of Events	• If there is an error during the communication with the PostgreSQL database or the data retrieval process: a. The system displays an error message indicating the issue. b. The user can try again. • If there are no detected objects found by YOLOv5: a. The system displays a message indicating that we are out of range.
Post Condition	The user is able to view the uploaded image, the detected objects, and the associated object details from the PostgreSQL database.
Comment	- - - - - -

3.4.11 Logout Usecase

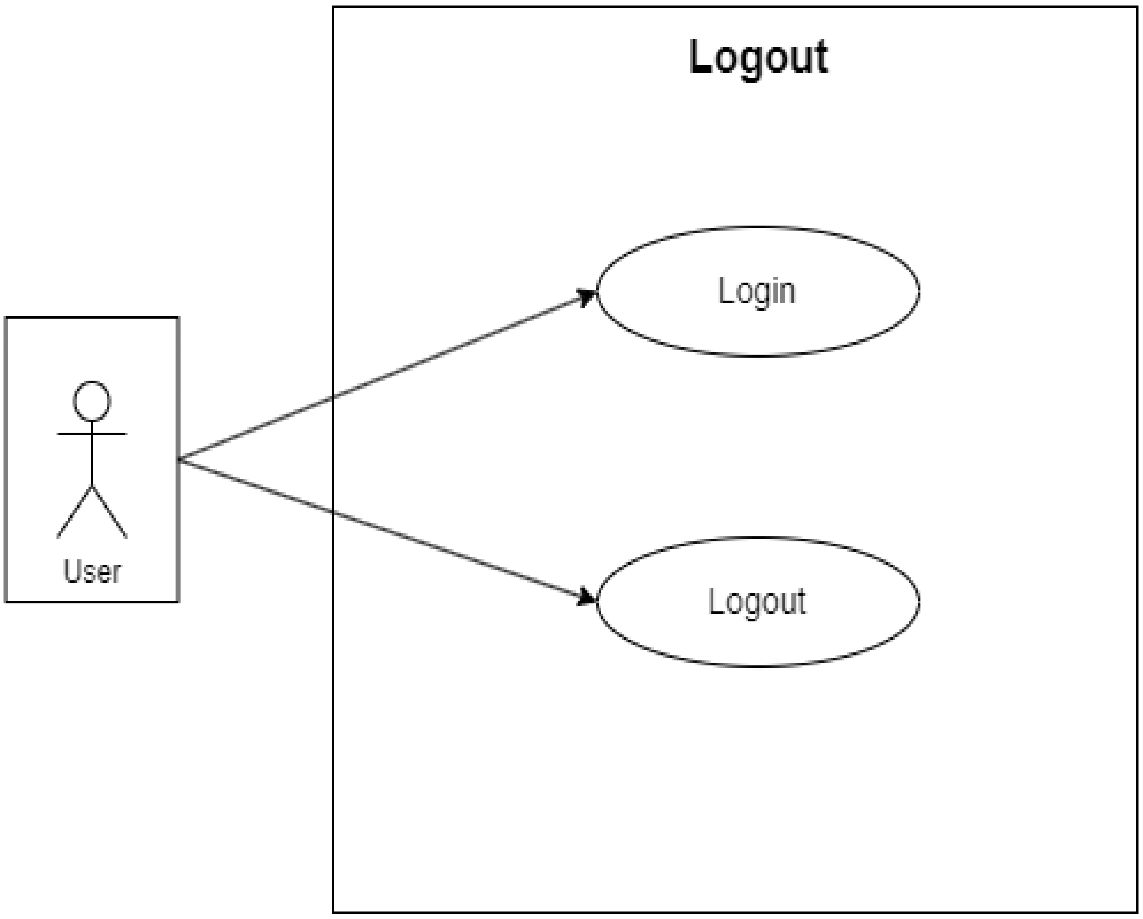


Figure 3.8: Use Case 6: Logout

3.4.12 Logout Usecase Table

Logout Usecase table is shown in table 3.6.

Table 3.6: Logout usecase table

Use Case ID	UC006
Use Case Name	Logout
Actor	User
Data	None
Precondition	The user is logged into the system.
Steps/Description	a. The user navigates to the "Exit" option in the application. b. The system verifies the user's current login status. c. The system terminates the user's session and revokes their access to protected functionality. d. The system redirects the user to the logout confirmation dialog. e. The user sees a confirmation message and if select the yes option then user will successfully logout and redirect to login page .
Alternative Flow of Events	• If the user attempts to access protected functionality after logging out: a. The system detects the user's logged-out state. b. The system redirects the user to the login page or prompts them to log in again. • If the user cancels the logout process: a. The user clicks on a "No" option during the logout confirmation step. b. The system maintains the user's current session and returns them to their previous state.
Post Condition	The user is successfully logged out of the system and redirect to login page.
Comment	- - - - - -

Chapter 4

Design

In this chapter of system design, we take a closer look at the development phases of "Underwater object detection", which is all about defining components, interfaces, and data to meet our defined criteria. The chapter is divided into modules, each of which is addressed in depth below.

A. System Architecture.

B. Activity Diagram.

C. Sequential Diagram.

4.1 System Architecture

System architecture refers to the conceptual structure of a system's behavior and operation. It concisely and clearly demonstrates how the system works, making it easier to understand and enhance.This system architecture includes the initial stage to gather data, which could be images, After gathering the data, it must be pre-processed to ensure it is in a format suitable for the model. This includes cleansing the data, dealing with missing values, and translating the data into a numerical representation that the model can understand. The next stage after Pre-processing is feature extraction. This involves relevant characteristics from data that will be utilized to train the model. In image classification, for example, feature extraction may entail recognizing edges, forms, and colors in a picture.

After the features have been extracted, an algorithm can be used to train the model. The algorithm will learn from the data by modifying its internal parameters so that the differences between the expected and actual output are as little as possible. Testing can then be used to evaluate the model's performance. This involves providing it fresh, previously unseen data and testing how well it predicts the correct outcome. Finally, the model testing results will be examined to evaluate whether the machine learning model is accurate enough for the purpose it was created for. To increase the model's performance, refine the model or make changes to the data pre-processing and feature extraction stages.

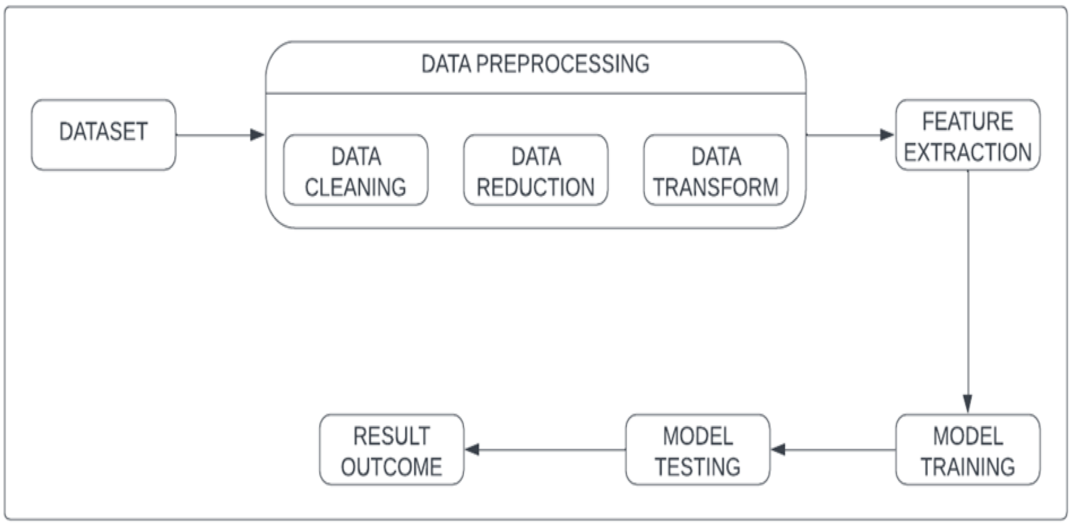


Figure 4.1: System Architecture

4.2 Design Constraints

4.2.1 Software Requirements constraints

- The application must support specific operating systems, such as Android.
- The application's user interface must be intuitive, user-friendly, and accessible to users of varying needs and abilities.

4.2.2 Hardware Requirements constraints

- The hardware must be capable of supporting the application's user interface, including any 2D or 3D graphics.
- Specific network connectivity requirements, such as Wi-Fi, cellular, or Ethernet, might require the use of specific hardware.

4.3 Activity Diagram

Activity diagram of our project is given below, which basically explains the graphical representation of activities.

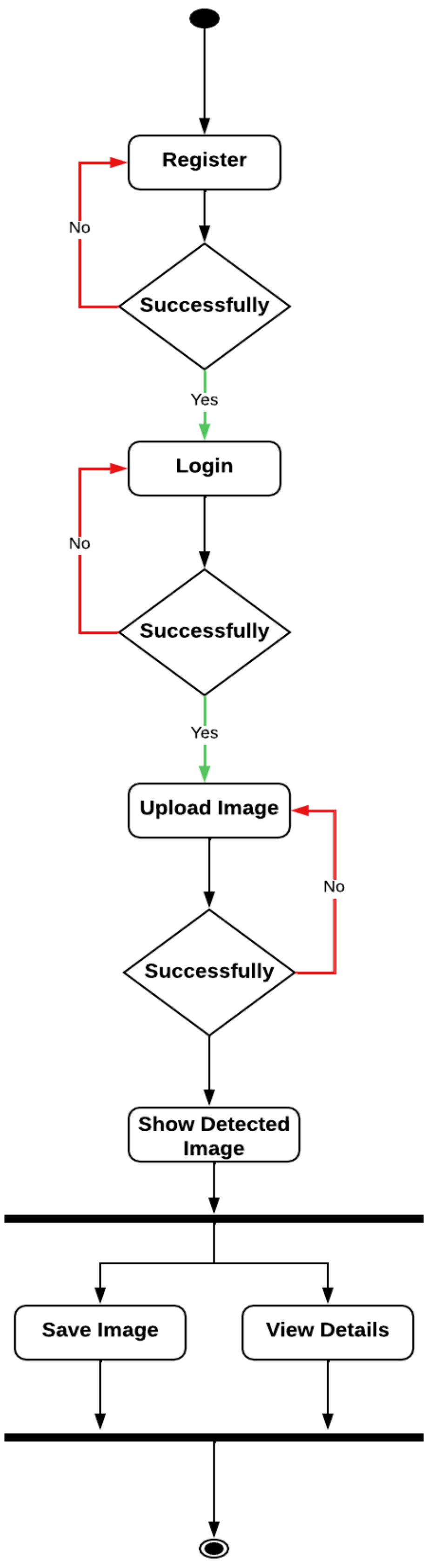


Figure 4.2: Activity Diagram

4.4 Sequential Diagram

Sequence diagram shows the interaction between the objects of the system application. The interaction between two lifelines is shown as a time-ordered sequence of events in the sequence diagram.

4.4.1 Sign-up Sequence Diagram

If the user has provided the sign-up information in the right order, an account will be established; otherwise, the system will show the user an error notice. The sign-up process is depicted in the diagram below.

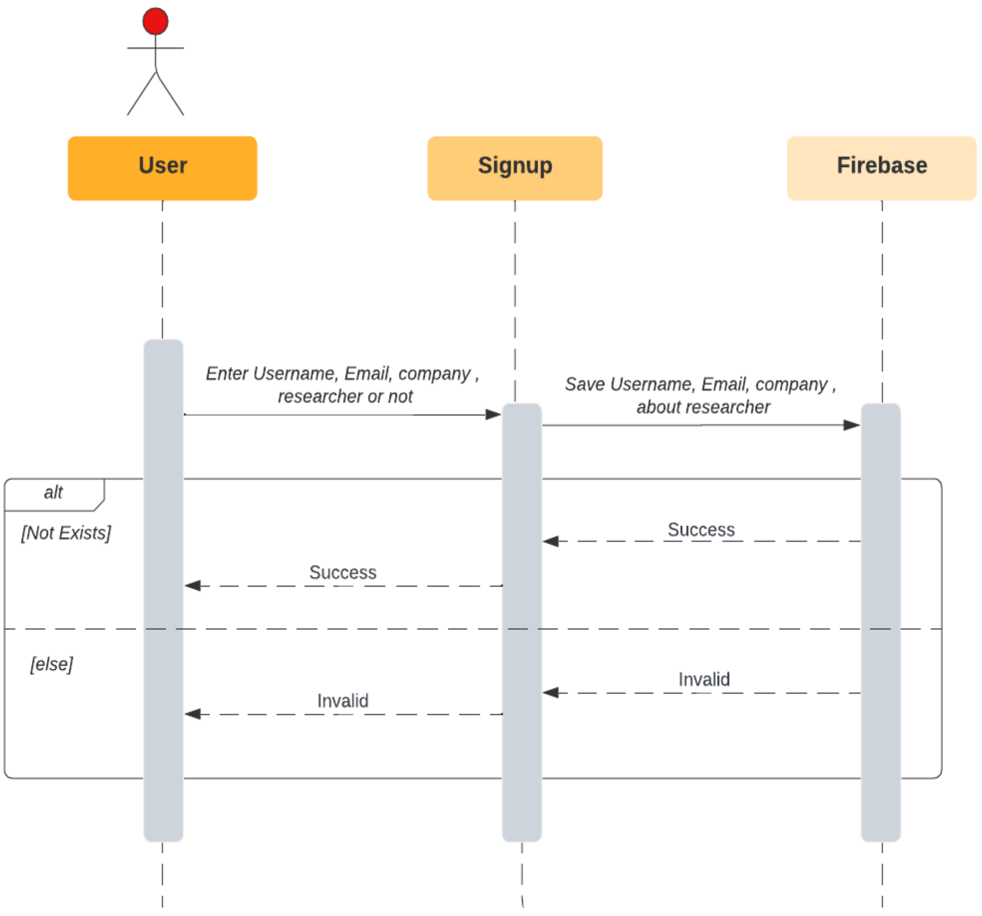

User
Signup
Firebase
Enter Username, Email, company , researcher or not
Save Username, Email, company , about researcher
alt
[Not Exists]
Success
Success
[else]
Invalid
Invalid

Figure 4.3: Sign-up Sequence Diagram

4.4.2 Login Sequence Diagram

If the user has provided the correct login information means Email, the system will allow them to access the application; otherwise, an error notice will be displayed.

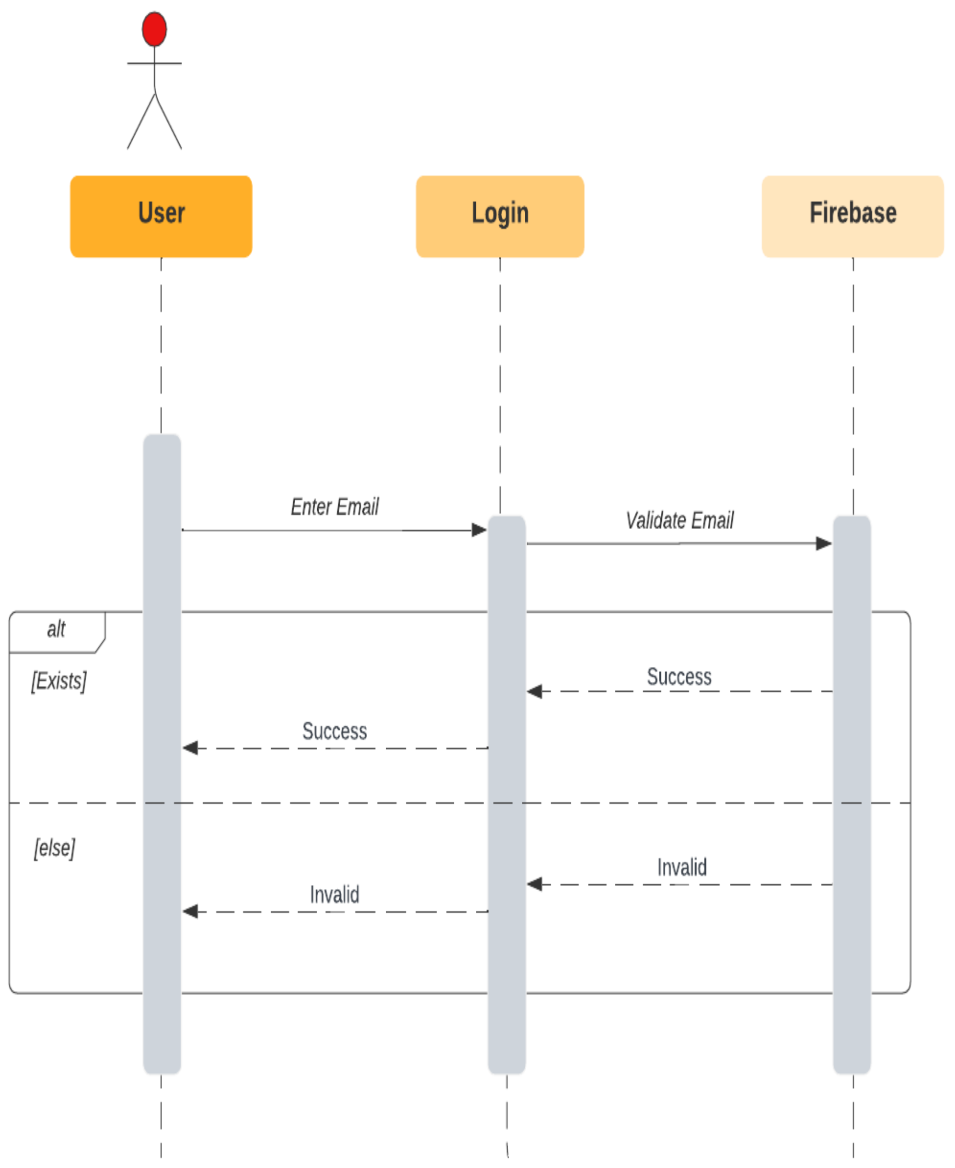


Figure 4.4: Login Sequence Diagram

4.4.3 Upload Image Sequence Diagram

In order to detect the underwater objects user must upload an image. if the image is successfully uploaded the model will start identifying the objects. if the image is not uploaded successfully it will show an error message.

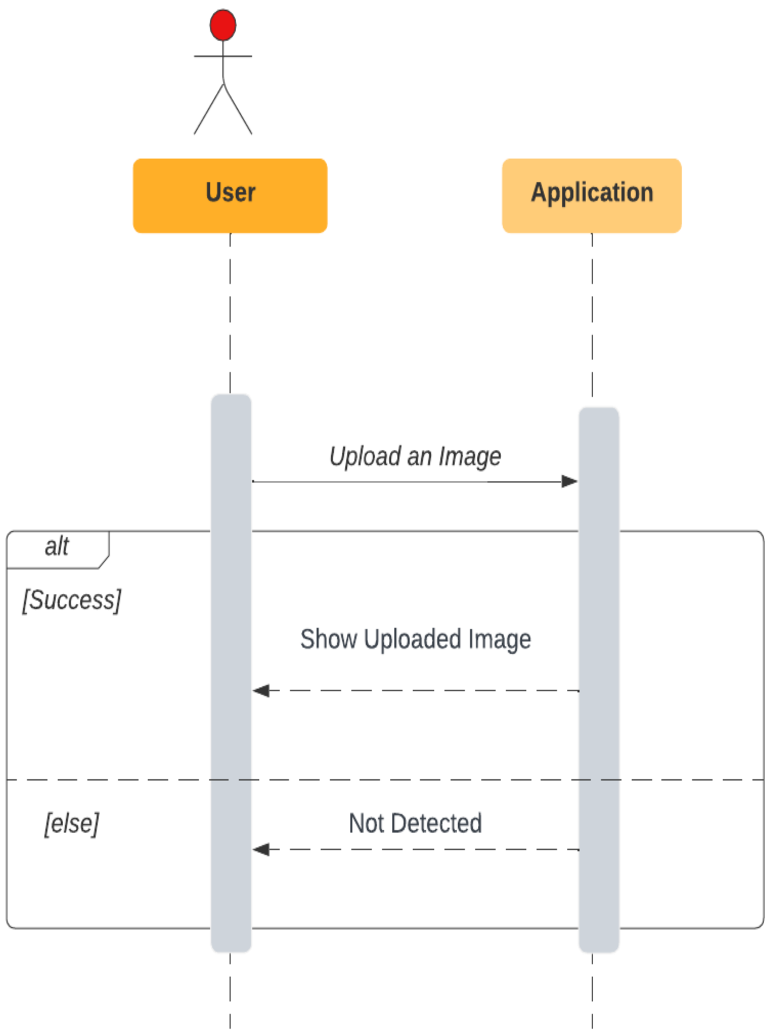


Figure 4.5: Upload Image Sequence Diagram

4.4.4 Show Detected Image Sequence Diagram

Image send to the model by applying deep learning model on the given image show the detected image to the user either their is one object present in the image or two. If the image is not related to the underwater environment it will display an error message.

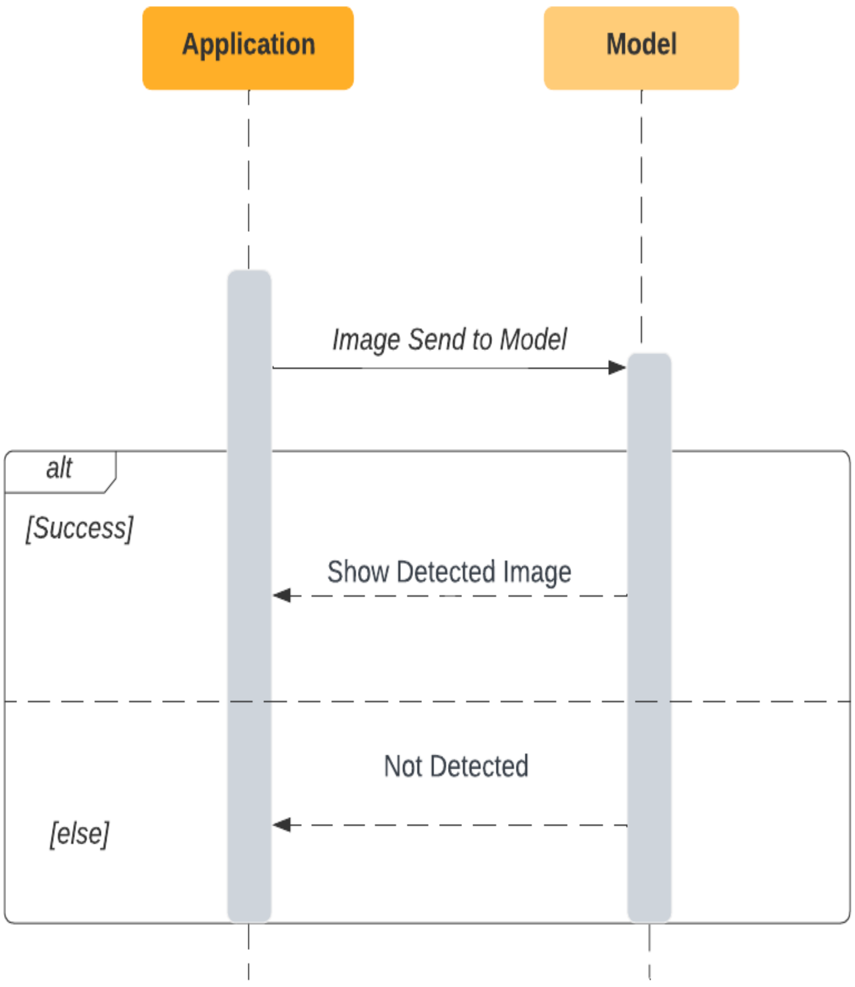


Figure 4.6: Show Detected Image Sequence Diagram

4.4.5 Save Detected Image Sequence Diagram

User can save the detected image in gallery. It provides users with the ability to save the images that contain detected objects, which can be used for future reference or shared with others.

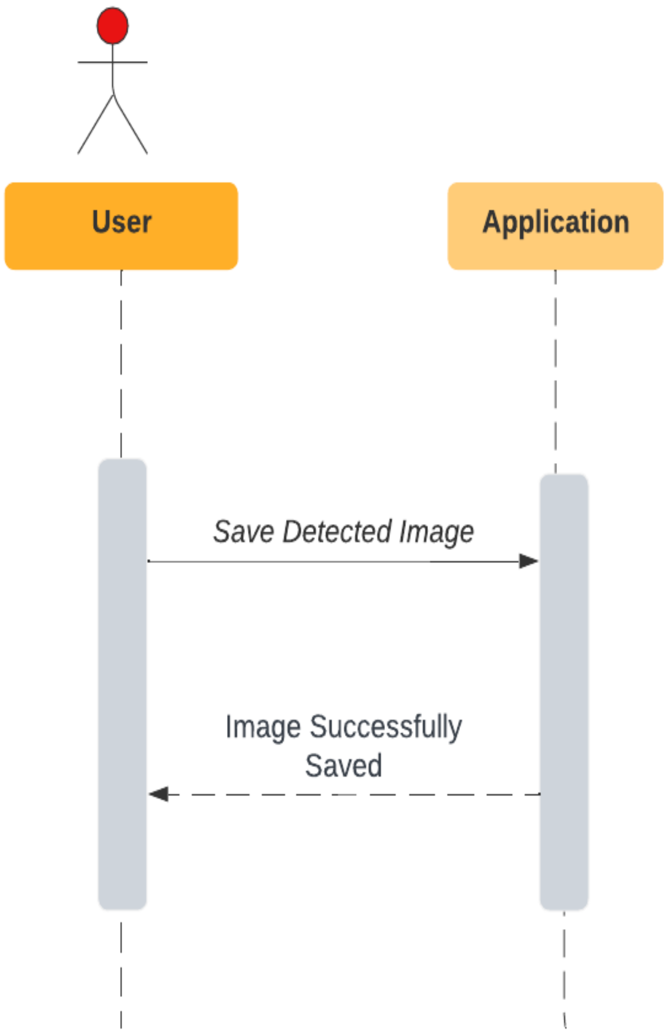


Figure 4.7: Save Detected Image Sequence Diagram

4.4.6 Show Data of detected objects

When the objects are detected successfully. This application displays the brief description of detected objects in order to make it easy to know about the objects. User will easily get to know the objects and its background.

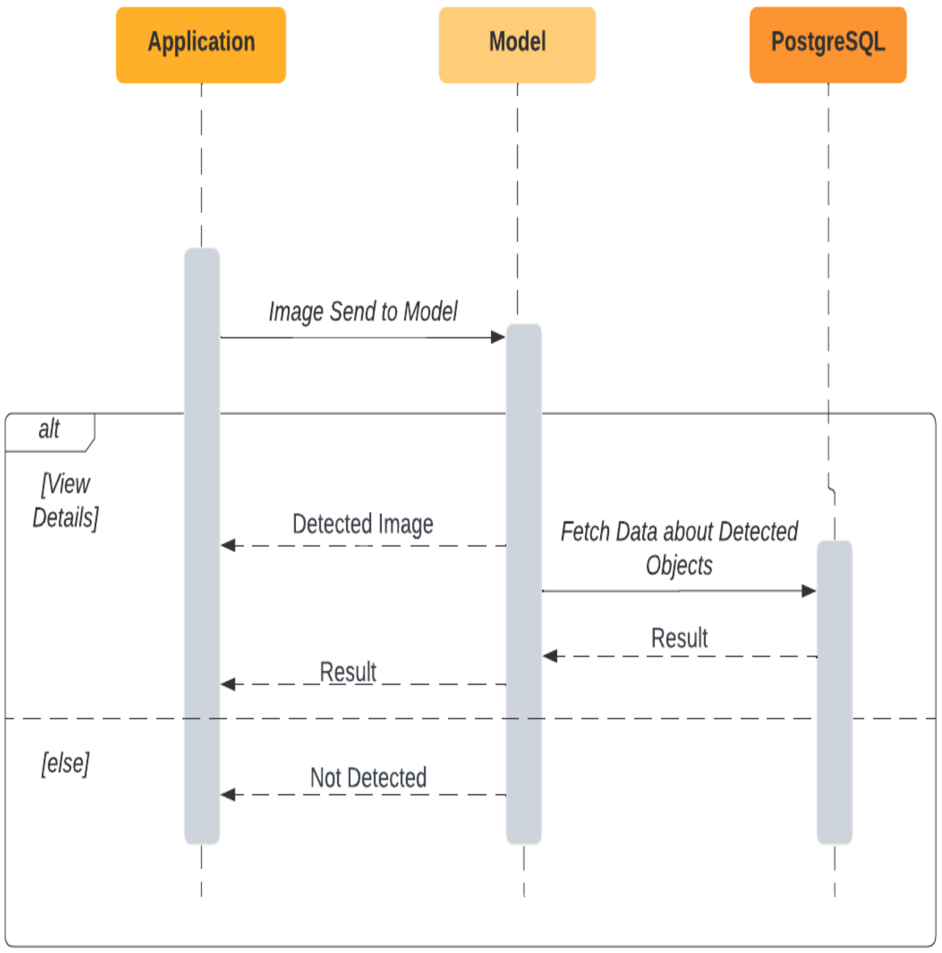


Figure 4.8: Show Data Sequence Diagram

4.4.7 Logout Sequence Diagram

Logout allows a user to securely sign out of their account and end their session. This is important for security reasons, as it helps to prevent unauthorized access to the user's account and personal information.

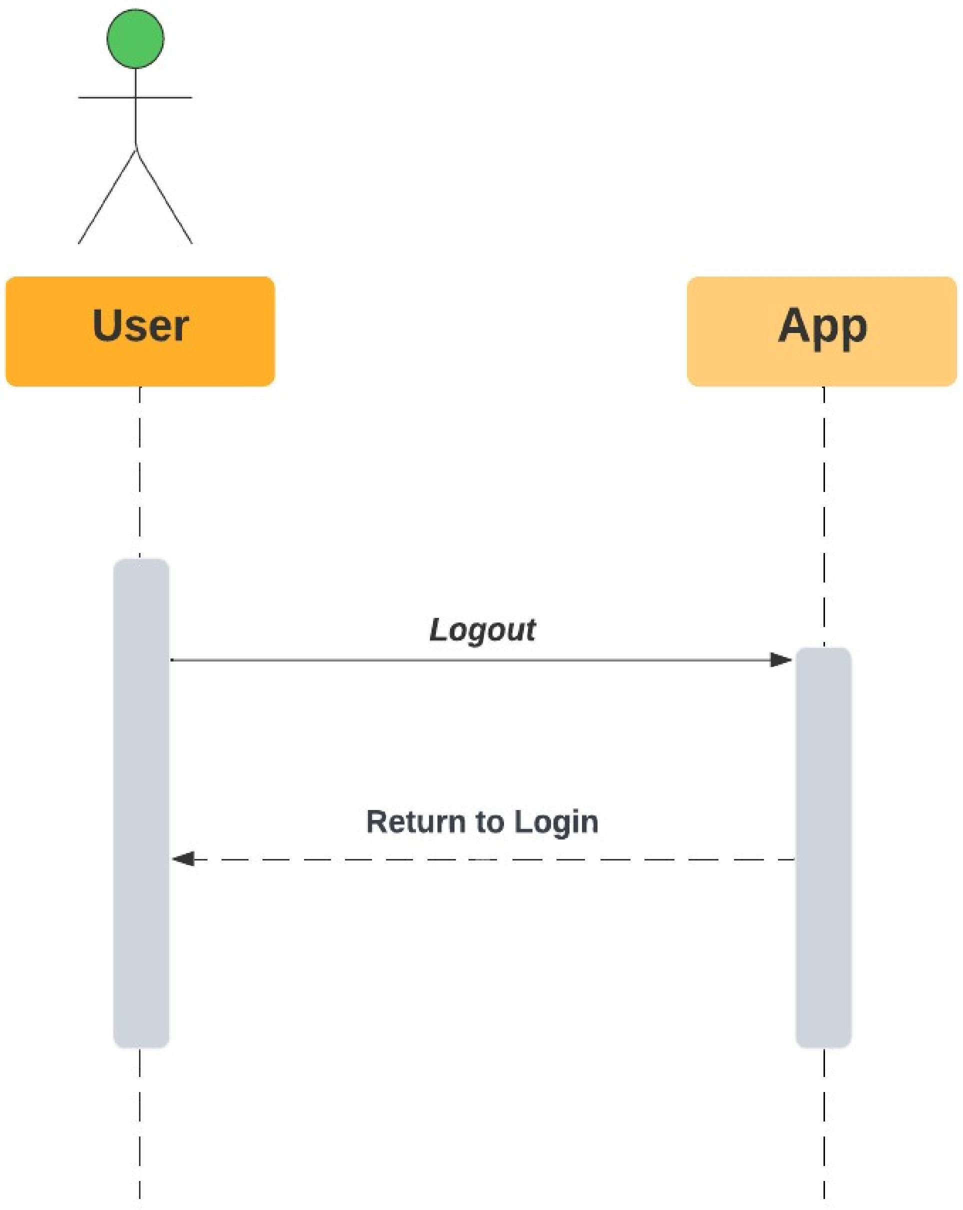


Figure 4.9: Logout Sequence Diagram

Chapter 5

System Implementation

This chapter presents the implementation detail of a system in the following section;

5.1 System Architecture

System architecture refers to the conceptual structure of a system's behavior and operation. It concisely and clearly demonstrates how the system works, making it easier to understand and enhance.. It becomes easier to identify the essential components and their relationships, as well as challenges that must be addressed, by describing the system architecture. Overall, system architecture acts as a road map for creating a system that satisfies the required goals and specifications.

5.1.1 Mobile Application

The system architecture of the underwater object detection mobile application would include numerous components that would work together to provide the desired functionality. The application for mobile devices would serve as the front-end user interface, allowing users to upload images and provide information about the object to be detected. The results of the object detection process would also be displayed by the application. The machine learning model would be trained on a collection of underwater photographs, using deep learning techniques to recognise and name items in the photos.

Overall, the system architecture of the underwater object identification mobile application would have a combination of front-end and back-end components that would work together to deliver an effortless and successful user experience.

5.2 System Internal Components

5.2.1 User Information page

In this, User will give some specific information about themselves. Firstly user will provide ’Name’ then ’Email Address’ this email will also help in login credential then user will answer about “ Are you a researcher”. After that user will enter organization Name.

5.2.2 Upload Image

User must upload an image. User can also crop the image while uploading image from gallery.

5.2.3 Enhance Image

If image needs to be enhanced then the system will enhance it before detection.

5.2.4 Object Detection

Image will send to Yolov5 model for detection and model will return the detected objects.

5.2.5 Download Image

User can download the detected image.

5.2.6 Description of Object

Image display along with the short description of the object.

5.3 Tools and Technologies

5.3.1 Visual Studio Code

Visual Studio Code is an open-source code editor with a wide range of features meant to increase productivity and make coding easier. It is compatible with a variety of programming languages, including C++, Python, Java, and PHP. Intellisense, which provides intelligent code completion and error highlighting, is one of its primary features, as is an integrated terminal that allows you to run scripts and start debuggers without leaving the editor.

5.3.2 Flutter and Dart

Google’s Flutter is a mobile app development platform that uses the Dart programming language. Flutter and Dart, when combined provide a number of features for developing high-performance, cross-platform mobile apps. Flutter also includes an extensive number of pre-built widgets and frameworks, making it simple to design beautiful and useful user interfaces for your project. Flutter also supports hot reloading, which enables rapid and easy testing and debugging of your code without having to recompile the entire app.

5.3.3 Firebase

Firebase is a mobile and online application development platform that offers app developers a variety of back-end services. It was created by Google and has a number of features that make app development faster and easier. One of Firebase's primary features is its real-time database, which enables for real-time data synchronization between clients and the server. This implies that changes to the database are reflected promptly in the app.

5.3.4 Model Yolo v5

Yolo v5 is an object detection model from the You Only Look Once (YOLO) model family. The Yolo v5 model predicts bounding boxes and class probabilities for objects in an input image using a single neural network. Overall, Yolo v5 is an advanced and adaptable object recognition model that can be applied to a wide range of applications.

5.3.5 PostgreSQL

PostgreSQL is a powerful open-source relational database management system (RDBMS) with several capabilities for data storage, organization, and manipulation. PostgreSQL also provides a number of tools for database management and monitoring, such as PgAdmin, a popular graphical user interface (GUI) for administering PostgreSQL databases.

5.3.6 Flask

Flask is a Python-based lightweight and flexible web framework. It includes a minimal set of tools and libraries that allow developers to quickly get started and customize their application as needed. Flask is also extremely flexible, with numerous third-party extensions available to add additional functionality and features.

5.4 Processing Logic/Algorithms

5.4.1 Model Selection

The process of model selection involves choosing an appropriate algorithm or model architecture based on the problem being solved and the characteristics of the dataset. In our case, we choose the pretrained model Yolov5, YOLOv5 is a popular object detection algorithm that has gained significant attention and adoption in computer vision tasks, it achieves a good balance between accuracy and speed. It is designed for fast and efficient object detection and YOLOv5 offers different model variants, including different sizes (such as yolov5s, yolov5m, yolov5l, and yolov5x), which allow users to trade off between speed and accuracy based on their specific requirements. To obtained a good accuracy we selected and trained our dataset on YOLOv5x. The model trained for 20 epochs on the training data with a validation split of 0.2%, making the training dataset of 0.8% and the valdiation dataset of 0.2% of the entire dataset.

5.4.2 Evaluation

The model took almost 15-20 minutes to train and yielding a validation accuracy of 81.9% with the rest of the details of the evalution shown in table 5.1 and in table 5.2 for the validation metrics of our model.

Table 5.1: Evaluation Metrics of Main Model

Evaluation Metrics	**Value**
Accuracy	81.8%
Precision	84.6%
Recall	74.6%
F1 Score	79.2%

Table 5.2: Validation Metrics of Main Model

Evaluation Metrics	**Value**
Validation Accuracy	81.9%
Validation Precision	79.1%
Validation Recall	80%
Validation F1 Score	79.5%

5.5 Development Environment/Languages Used

Flutter dart and Python has been used as the development language while Visual Studio code is the development environment.

5.6 Methodology

In order to implement this we proceed as follows, firstly we analyzed the functionality of this system login/sign-up page, user authentication, upload the image , object detection using yolov5 , database for data storage. In next step we design the system like how data is flow between these components. We developed our front-end using Flutter and integrated it with firebase. We used two features of firebase first one is user authentication and second one is firestore database for storing the user's data. Then developed our back-end using python in which we used Yolov5 model for object detection and image processing techniques for image enhancement and also used PostgreSQL database for storing the description of the detected objects. We used Flask API for integration between front-end and back-end.

When a person who is not yet registered wants to sign up for the application. An invalid message will be shown. And when the user register, the user gets access to the application. After logging in, he/she'll will be redirected to main page where they can upload image for detection when user will upload an image it will send to back-end for enhancement then after enhancement it will be detected by Yolov5 model and fetch the data from the PostgreSQL database and return to the flutter app and will show the user about object with its short description.

Chapter 6

System Testing and Evaluation

After developing an application, the process of testing is there and testing of a system is necessary to check the errors, bugs or requirements. To evaluate whether the requirements are fulfilled or not, system testing is performed. This testing is done on a completed system. The other tests will also be performed accordingly. We have mentioned the tests we are applying to our application.

6.1 Graphical User interface testing

We perform system interface testing to ensure that the graphical user interface works properly. In the GUI testing stage, we examined the visual design, functionality, color scheme, buttons, the message shown, and alignments. GUI testing is critical since it validates the client experience. GUI testing contributes in the delivery of good and user-friendly applications. So, in our instance, our application works flawlessly. Our pages, buttons, and color scheme all have a clear design and function. The text we used is understandable. Buttons are the correct size. The color scheme is perfect.

6.1.1 Splash Screen

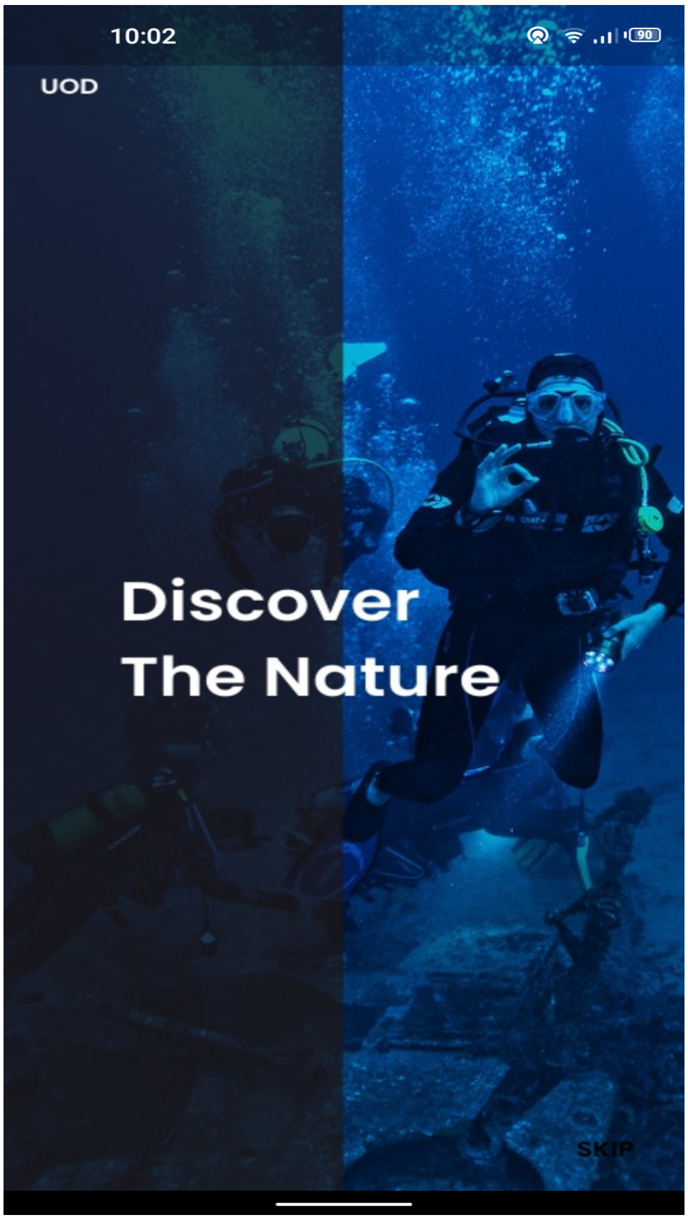


Figure 6.1: Splash Screen GUI

6.1.2 Main Screen

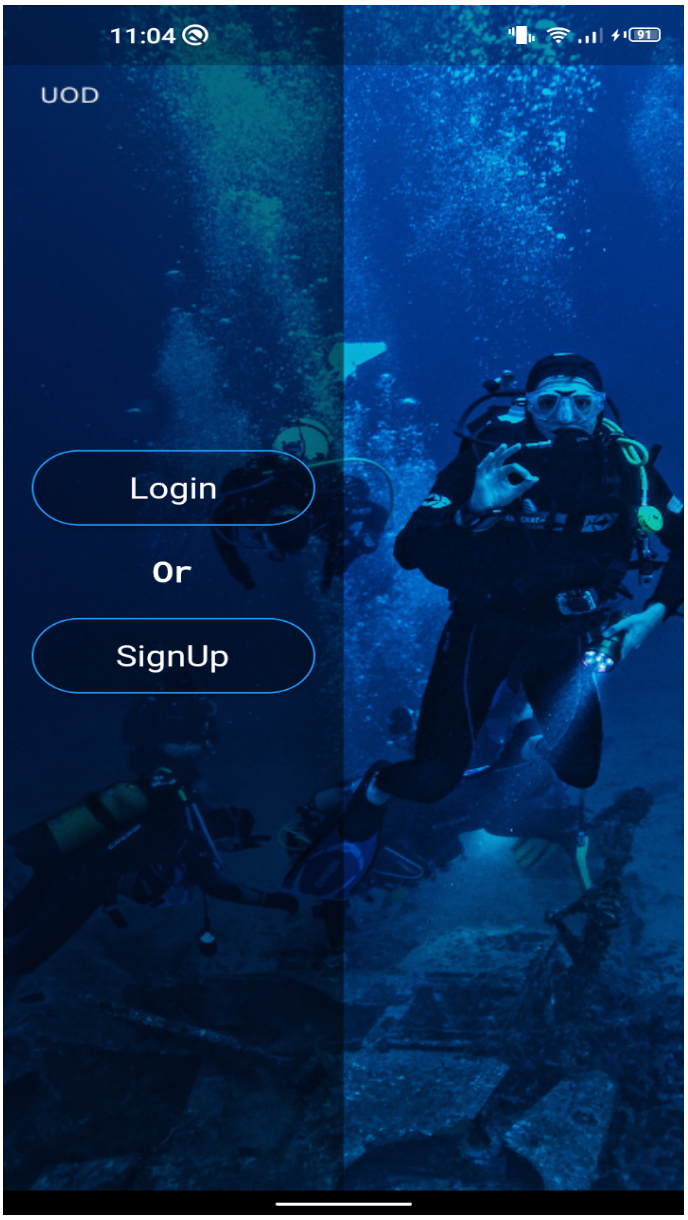


Figure 6.2: Main Screen GUI

6.1.3 Sign-Up

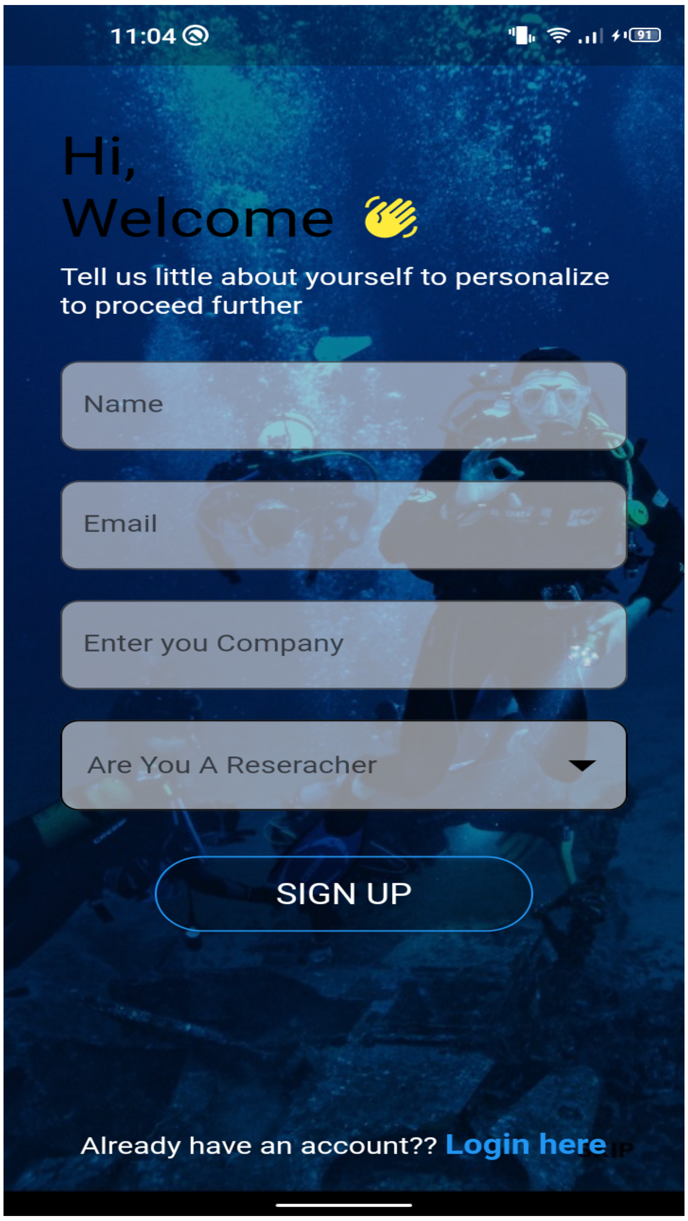

11:04
Hi,
Welcome
Tell us little about yourself to personalize to proceed further
Name
Email
Enter you Company
Are You A Reseracher
SIGN UP
Already have an account?? Login here

Figure 6.3: Signup GUI

6.1.4 Login

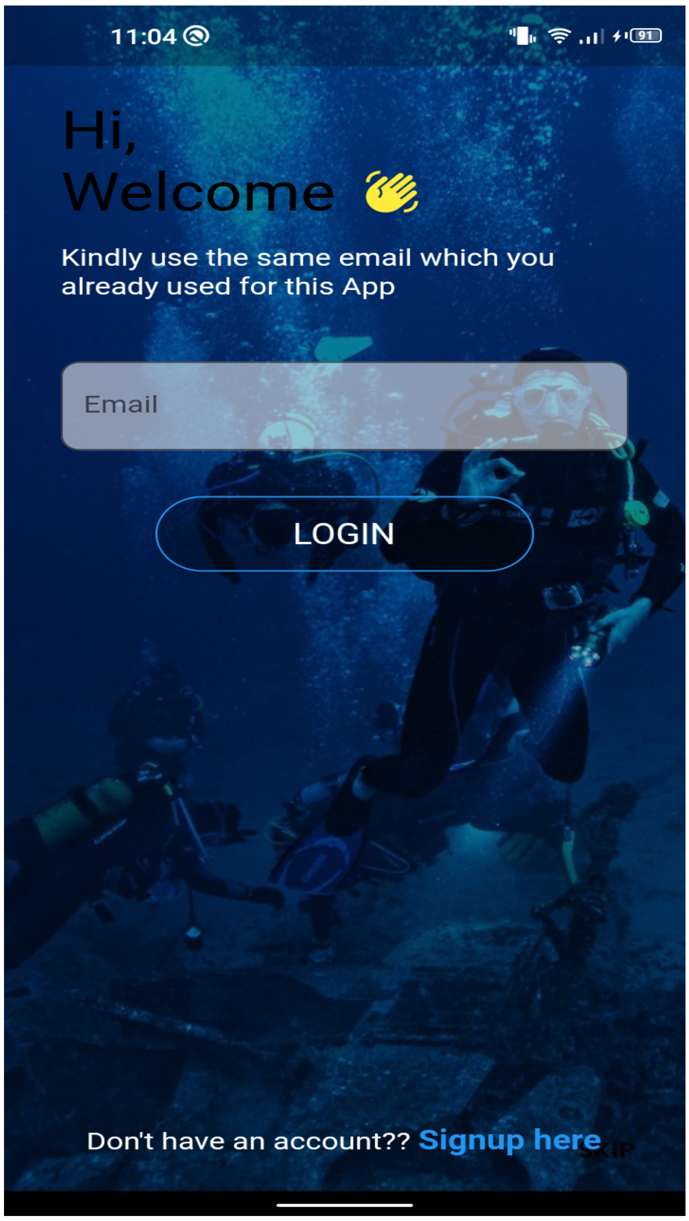

11:04
Hi,
Welcome
Kindly use the same email which you already used for this App
Email
LOGIN
Don't have an account?? Signup here

Figure 6.4: Login GUI

6.1.5 Upload Image

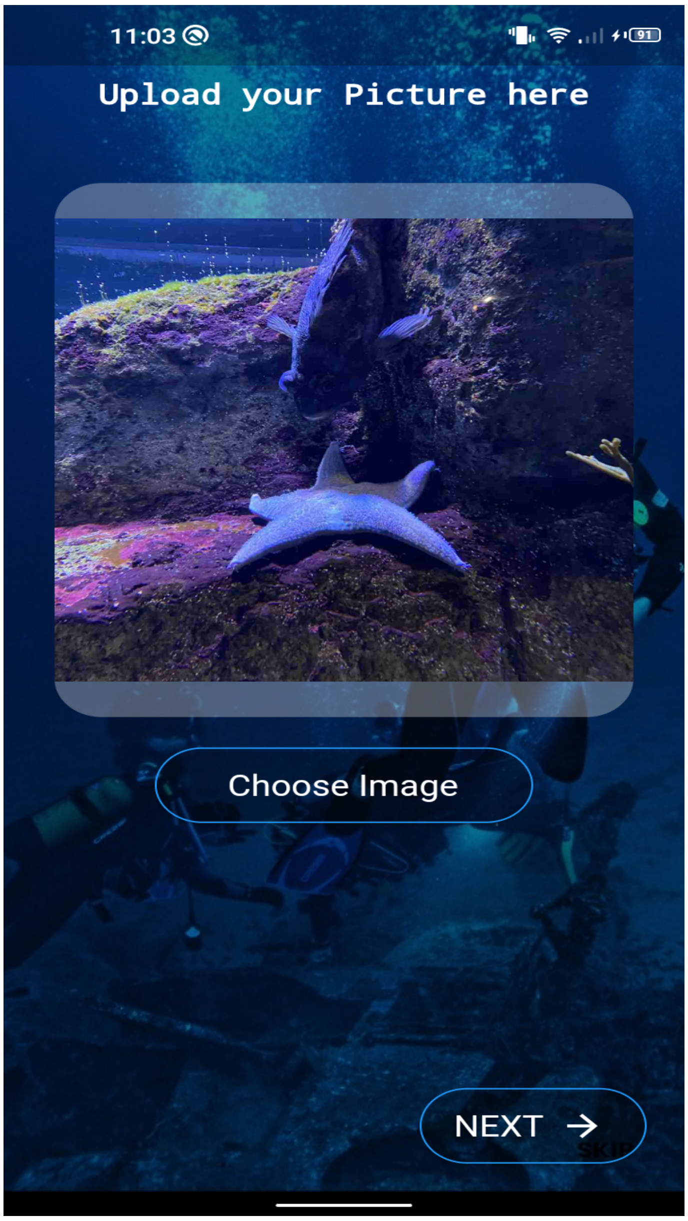


Figure 6.5: Upload Image GUI

6.1.6 Detected Image

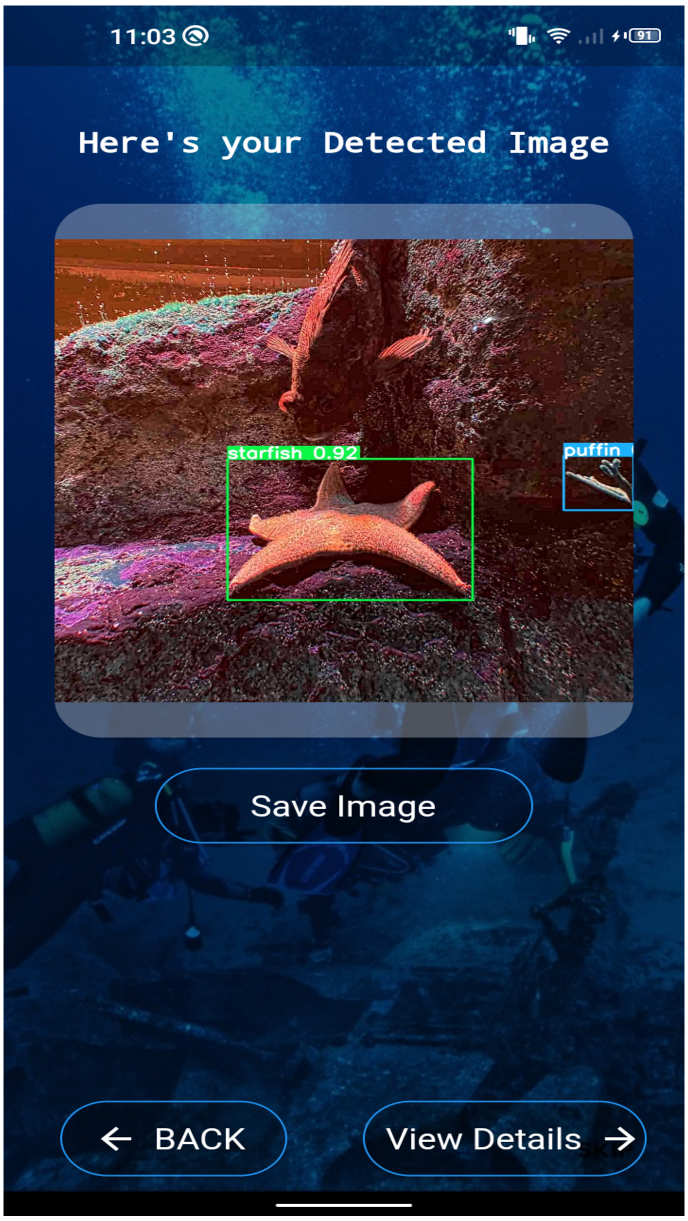


Figure 6.6: Detected Image GUI

6.1.7 Download Image

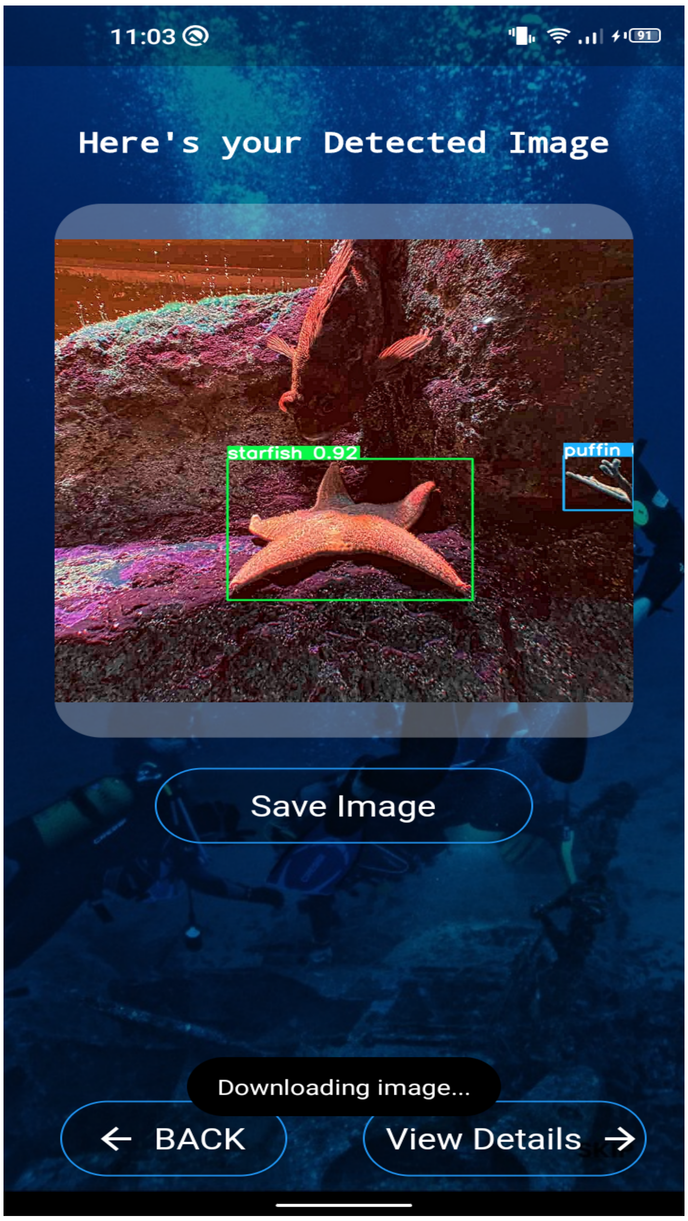


Figure 6.7: Download Image GUI

6.1.8 Image save to Gallery

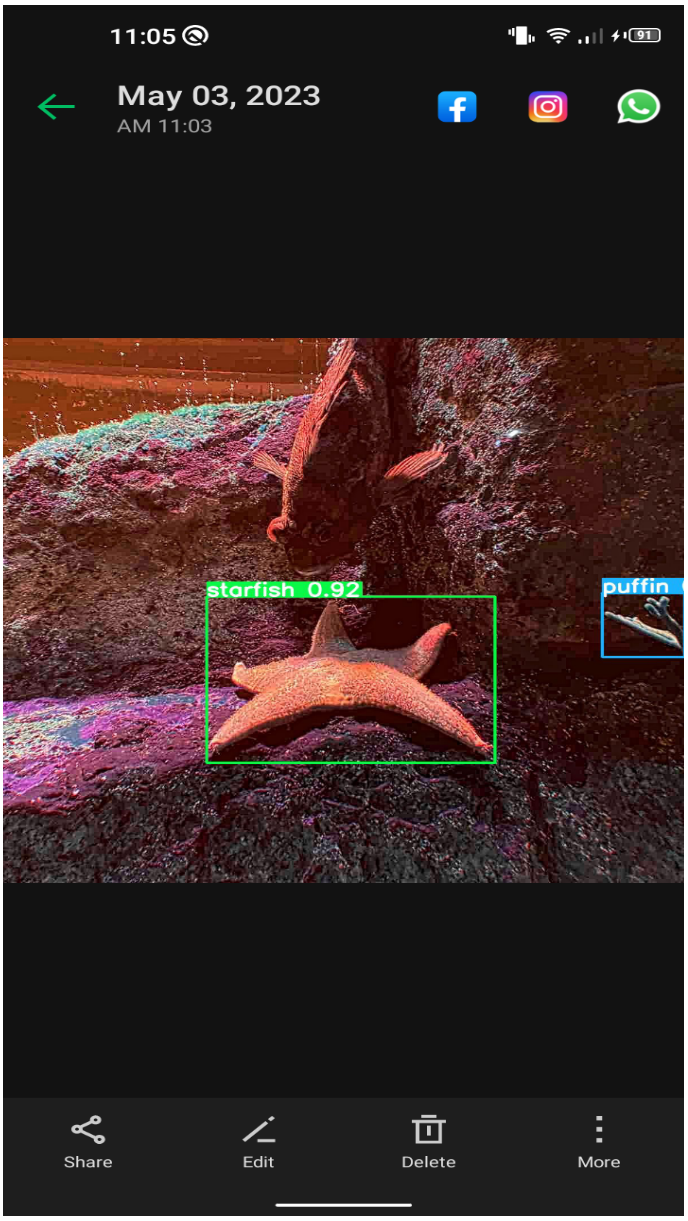
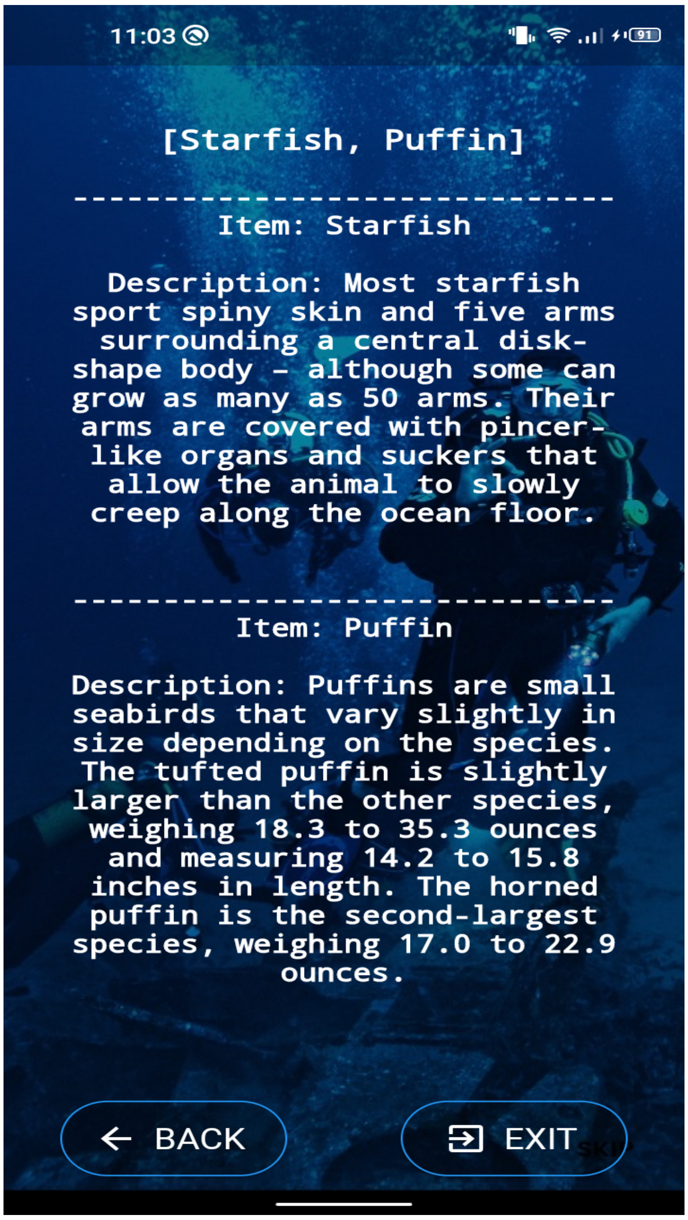


Figure 6.8: Image Save to Gallery GUI

6.1.9 View Detail



[Starfish, Puffin]

Item: Starfish

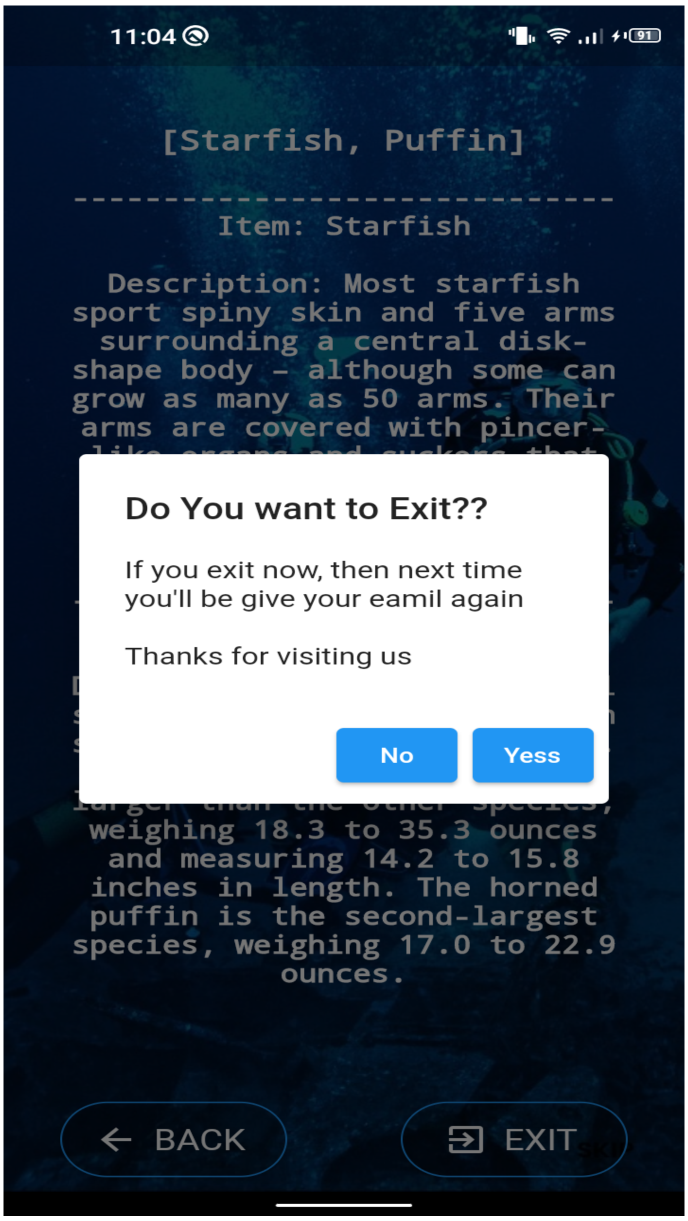
Description: Most starfish sport spiny skin and five arms surrounding a central disk-shape body – although some can grow as many as 50 arms. Their arms are covered with pincer-like organs and suckers that allow the animal to slowly creep along the ocean floor.

Item: Puffin

Description: Puffins are small seabirds that vary slightly in size depending on the species. The tufted puffin is slightly larger than the other species, weighing 18.3 to 35.3 ounces and measuring 14.2 to 15.8 inches in length. The horned puffin is the second-largest species, weighing 17.0 to 22.9 ounces.

Figure 6.9: View Detail GUI

6.1.10 Logout



11:04

[Starfish, Puffin]

Item: Starfish

Description: Most starfish sport spiny skin and five arms surrounding a central disk-shape body - although some can grow as many as 50 arms. Their arms are covered with pincer-

Do You want to Exit??

If you exit now, then next time you'll be give your eamil again

Thanks for visiting us

weighing 18.3 to 35.3 ounces and measuring 14.2 to 15.8 inches in length. The horned puffin is the second-largest species, weighing 17.0 to 22.9 ounces.

Figure 6.10: Logout GUI

6.2 Usability Testing

Usability testing adheres to several guidelines one of which is that the application should be easy to understand and use. Each button's purpose and function must be understood by the user. Users were able to understand the purposes of buttons and how to perform them in this application, and they were able to immediately enhance the objects based on their preferences. The user interface should provide a pleasant experience. The color scheme used in the application is simple but appealing.

6.2.1 Feedback

When buttons are pressed, their color changes to indicate that a certain action has been taken.

6.2.2 Visibility

We choose blue and white color theme for our application's text, buttons, and backdrop, as well as appropriate button sizes, so that everything is more visible and the user has a pleasant experience when using it.

6.2.3 Mapping

In the Home screen and other screens of application, proper mapping is employed.

6.2.4 Inconsistency

Our application is consistent and follows widely accepted procedures.

6.2.5 Affordance

Every action the user makes is accompanied by suggestions and help from the application.

6.2.6 Effectiveness

It is capable of fulfilling its functions and has cleared all the test case situations without crashing.

6.2.7 Efficiency

It is highly fast and efficient, and opening or performing any operation takes no time.

6.2.8 Target Sizes

In our application the sizes of the icons and the options for picking them are placed near each other, visible, and easy to select.

6.3 Security Testing

With the use of security testing, risks to the application may be found and its possible weaknesses can be assessed, allowing for threat resolution without the application being unusable. There are no risks associated with our application. No unauthorized user has access to the system and is not permitted to read or edit any user information or other database data.

6.4 Load Testing

Load testing is a type of non-functional testing. It is used to forecast how the application will perform under varying loads. A load test is a sort of performance test that determines how the system performs while a high number of virtual users execute transactions concurrently over an extended period of time. In other words, the test evaluates how well systems handle tremendous amounts of heavy load. The primary purpose of load testing is to evaluate the speed of the application. its rate of speed. Our application operates without any issues on any of the devices we've tried, and we haven't encountered any thus far. The application loaded rapidly and responded to user input quickly.

6.5 Compatibility Testing

Compatibility testing is the process of determining whether or not an application is compatible with a device. It ensures user satisfaction by determining whether or not the application fits the user's needs. Our application is system-compatible and loads swiftly. It works perfectly. The tasks that our application will perform are as follow: 1. Upload Image 2. Enhance Image 3. Download Image 4. Object Description

6.6 Software Performance Testing

Software performance testing is used to evaluate an application's performance under a variety of workloads. Our application is quick and has a short delay in responding, which benefits users because they do not have to wait. It loads accurately and completely during runtime and does not use a lot of storage or RAM. It works flawlessly.

6.7 Test Cases

We did test checks to verify that our application is working fine.

6.7.1 Register: Test Case 1

To Check if Underwater object Detection application registration page working correctly shown in table 6.1 .

Table 6.1: Test case 1

Test Case 1: Register	
ID	Test case 1
Description	To ensure that the sign-up form is functioning properly and that users can successfully register for the service.
Pre-conditions	Installing and launching an application is required. • Internet connectivity should be accessible.
Test steps	• Check that the required fields are present and appropriately labelled (such as username and email). • Check that sign-up page is properly loaded. • Fill in all the relevant fields with accurate info. • Make sure the email address is typed in the correct format. • Verify that the user is taken to a home page after clicking the sign-up button.
Expected result	•The sign-up page loads successfully • All required fields are present and labeled correctly. • Valid data can be entered in all required fields. • Email address entered in a valid format. • Clicking on the sign-up button successfully redirects the user to a home page.
Post-condition	The user is successfully signed up and can access the service using the provided credentials.
Test status	Pass

6.7.2 Login: Test Case 2

To Check if Underwater object Detection application Login page working correctly shown in table 6.2.

Table 6.2: Test case 2

Test Case 2: Login	
ID	Test case 2
Description	To ensure that the Login form is functioning properly and that users can successfully login into the service.
Pre-conditions	• Installing and launching an application is required. Internet connectivity should be accessible. • User should be registered already.
Test steps	• Check that the required fields are present and appropriately labelled (such as email). • Check that login page is properly loaded. • Enter valid and accurate email. • Make sure the email address is typed in the correct format. • Verify that the user is taken to a home page after clicking the login button. • Check that user should be able to use all features
Expected result	• The login page loads successfully. • All required fields are present and labeled correctly. • Email address entered in a valid format. • Clicking on the login button successfully redirects the user to a home page. • User able to access all features
Post-condition	The user is successfully logged in and can access the service using the provided credentials.
Test status	Pass

6.7.3 Upload Image : Test Case 3

To Check if Underwater object Detection application upload image functionality working correctly shown in table 6.3 .

Table 6.3: Test case 3

Test Case 3: Upload Image	
ID	Test case 3
Description	To ensure that the upload image functionality is working properly and that users can successfully upload an image.
Pre-conditions	• Installing and launching an application is required. Internet connectivity should be accessible. • User should be logged in.
Test steps	• Check that the upload image page is properly loaded. • Check that the upload image button is labeled and present correctly. • Verify that button is working properly. • Verify that the image uploaded successfully
Expected result	• The upload image page loads successfully. • Upload image button is labeled and present correctly. • Upload image button is working properly. • Image uploaded and displayed successfully.
Post-condition	The user is successfully uploaded an image and viewed the image.
Test status	Pass

6.7.4 Image enhancement: Test Case 4

To Check if Underwater object Detection application image enhancement functionality working correctly shown in table 6.4 .

Table 6.4: Test case 4

Test Case 4: Image enhancement	
ID	Test case 4
Description	To ensure that the image enhancement functionality is working properly.
Pre-conditions	• Installing and launching an application is required. Internet connectivity should be accessible. • User should be logged in. • Image uploaded successfully.
Test steps	• Check that the next button is labeled and present correctly. • Verify that the next button is working properly. • After clicking the next button the image should go to back-end for enhancement. • Verify that image enhancement result before prediction
Expected result	• Next button is labeled and present correctly. • Image send to back-end properly. • Image enhancement process completed successfully.
Post-condition	The enhanced image will send to the yolov5 model for prediction.
Test status	Pass

6.7.5 Object detection : Test Case 5

To Check if Underwater object Detection application object detection functionality working correctly shown in table 6.5 .

Table 6.5: Test case 5

Test Case 5: Object detection	
ID	Test case 5
Description	To ensure that the object detection functionality is working properly.
Pre-conditions	• Installing and launching a application is required. Internet connectivity should be accessible. • User should be logged in. • Image enhancement process completed successfully
Test steps	• Check that the outcome after enhancement is successfully send to Yolov5 model. • Verify that yolov5 model is working properly. • Verify that our model give good accuracy. • Verify that it detects objects properly.
Expected result	• Image enhancement result sends to model successfully. • Model is working properly. • Model gives good accuracy. • It detects properly.
Post-condition	Fetch data of detected objects and show it in the screen.
Test status	Pass

6.7.6 Detected Objects description : Test Case 6

To Check if Underwater object Detection application detected object functionality working correctly shown in table 6.6.

Table 6.6: Test case 6

Test Case 6: Detected Objects description	
ID	Test case 6
Description	To ensure that the functionality of fetching data from PostgreSQL is working properly.
Pre-conditions	• Installing and launching an application is required. Internet connectivity should be accessible. • User should be logged in. • Model should already detect objects from the image.
Test steps	• Check that PostgreSQL is working properly. • Verify that PostgreSQL is connected successfully. • Verify that retrieval data is accurate.
Expected result	• PostgreSQL is working properly. • PostgreSQL is connected successfully. • Retrieval data is accurate.
Post-condition	Show this data on the screen.
Test status	Pass

6.7.7 Show data : Test Case 7

To Check if Underwater object Detection application show data functionality working correctly shown in table 6.7.

Table 6.7: Test case 7

Test Case 7: Show data	
ID	Test case 7
Description	To ensure that the functionality of displaying data on the screen is working properly.
Pre-conditions	• Installing and launching an application is required. Internet connectivity should be accessible. • User should be logged in. • Model should already detect objects from the image. • Already fetched data from PostgreSQL.
Test steps	• Check that the display data page is properly loaded. • Verify that integration between front-end and back-end is done accurately and completely. • Check that API runs properly. • Verify that Flask API returns data accurately. • Verify that the application gets data from the API. • Check that if data are shown in the application properly.
Expected result	• Display data page loaded successfully. • Integration is done completely and accurately. • Flask API runs and return data. • Application retrieve data properly. • Data shown in the display page.
Post-condition	User will log out
Test status	Pass

6.7.8 Logout : Test Case 8

To Check if Underwater object Detection application logout functionality working correctly shown in table 6.8 .

Table 6.8: Test case 8

Test Case 8: Logout	
ID	Test case 8
Description	To ensure that the functionality of logout is working properly.
Pre-conditions	• Installing and launching an application is required. Internet connectivity should be accessible. • User should be logged in
Test steps	• Check that the logout button is labeled and present correctly. • Verify that the logout button is working properly
Expected result	• Button is labeled and present correctly. • Logout button is working properly.
Post-condition	User will redirect to login page.
Test status	Pass

Chapter 7

Conclusions

7.1 Conclusion

In this project, we created an application that allows researchers to easily identify objects in the underwater environment. Images obtained under these conditions are usually blurry and low contrast, making object recognition difficult. However, by uploading a picture and providing necessary details, the model correctly recognizes the objects in the underwater situation.

Creating this system was a difficult undertaking, but we approached it with a growth mindset and gained valuable insights throughout the process. We utilized deep learning models and trained them to meet the project's requirements. We encountered different obstacles during the project, which presented us with an opportunity to learn and grow.

We learned valuable skills such as gathering requirements, creating short-term initiatives, and implementing them throughout the project. These skills will be beneficial for future projects and assist us in developing more efficient and effective systems. In summary, this project was a valuable learning opportunity that taught us new methods and techniques for creating machine-learning applications.

7.2 Future Enhancements

Functionalities we tend to add in the future are;

7.2.1 Communication and collaboration feature:

A potential improvement for the underwater object detection application would be to integrate communication and collaboration features, which could greatly benefit researchers.

By enabling researchers to share images, research details, and results, they can collaborate more effectively, potentially leading to new discoveries and insights.

The communication feature could also allow researchers to discuss specific species, share observations or insights, and collaborate on research projects. Such exchanges could promote the exchange of knowledge and ideas, leading to a more comprehensive understanding of underwater ecosystems.

7.2.2 Expanding the number of species:

Improving the underwater object detection model's ability to recognize distinct species is a critical goal for future progress. At the moment, the model can only detect a few underwater species, but it can be expanded to include a more variety of species. To achieve this goal, additional data and photos of various underwater creatures must be collected and integrated into the model's training data set.

Increasing the model's ability to recognize a broader range of species is an important step toward developing a more comprehensive and efficient underwater item identification system.

References

[1] P Srinivas Babu, B Prateek, P Punith, B Pramod, and M Nitin Patel. Underwater object detection using image processing. *International Journal of Research in Engineering, Science and Management*, 4(7):315–319, 2021. Cited on p. 1.

[2] Di Wu, Fei Yuan, and En Cheng. Underwater no-reference image quality assessment for display module of rov. *Scientific Programming*, 2020:1–15, 2020. Cited on p. 2.

[3] Roboflow. Aquarium combined dataset. https://universe.roboflow.com/brad-dwyer/aquarium-combined, feb 2023. visited on 2023-05-02. Cited on p. 3.

[4] Francesco Rossi, Alfredo Benso, Stefano Di Carlo, Gianfranco Politano, Alessandro Savino, and Pier Luigi Acutis. Fishapp: A mobile app to detect fish falsification through image processing and machine learning techniques. In *2016 IEEE international conference on automation, quality and testing, robotics (AQTR)*, pages 1–6. IEEE, 2016. Cited on pp. 4 and 10.

[5] Risheng Liu, Xin Fan, Ming Zhu, Minjun Hou, and Zhongxuan Luo. Real-world underwater enhancement: Challenges, benchmarks, and solutions under natural light. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(12):4861–4875, 2020. Cited on p. 4.

[6] Zhe Chen, Yang Sun, Yupeng Gu, Huibin Wang, Hao Qian, and Hao Zheng. Underwater object segmentation integrating transmission and saliency features. *IEEE Access*, 7:72420–72430, 2019. Cited on p. 6.

[7] Chongyi Li, Jichang Guo, and Chunle Guo. Emerging from water: Underwater image color correction based on weakly supervised color transfer. *IEEE Signal processing letters*, 25(3):323–327, 2018. Cited on p. 6.

[8] Wagner Barros, Erickson R Nascimento, Walysson V Barbosa, and Mario FM Campos. Single-shot underwater image restoration: A visual quality-aware method based on light propagation model. *Journal of Visual Communication and Image Representation*, 55:363–373, 2018. Cited on p. 7.

[9] Fenglei Han, Jingzheng Yao, Haitao Zhu, and Chunhui Wang. Underwater image processing and object detection based on deep cnn method. *Journal of Sensors*, 2020, 2020. Cited on p. 7.

[10] Sheezan Fayaz, Shabir A Parah, and GJ Qureshi. Underwater object detection: architectures and algorithms–a comprehensive review. *Multimedia Tools and Applications*, 81(15):20871–20916, 2022. Cited on p. 9.

[11] Aprameya Satish, Brendan Nichols, David Trivett, and Karim G Sabra. Passive underwater acoustic markers for navigation and information encoding for high frequency sound navigation and ranging (sonar) devices. *The Journal of the Acoustical Society of America*, 142(4):2731–2731, 2017. Cited on p. 11.

[12] Yvan R Petillot, Gianluca Antonelli, Giuseppe Casalino, and Fausto Ferreira. Underwater robots: From remotely operated vehicles to intervention-autonomous underwater vehicles. *IEEE Robotics & Automation Magazine*, 26(2):94–101, 2019. Cited on p. 11.