Real-Time Bus ID Verification and Tracking

**24-FYP-204**



**Session 2021-2025**

**BACHELOR OF SCIENCE IN** **COMPUTER SCIENCE**

**SUBMITTED BY**

|  |  |
| --- | --- |
| Abubakar Nadeem | 21-NTU-CS-1291 |
| Imran Ali Niaz | 21-NTU-CS-1321 |
| Huzaifa Ahmad | 21-NTU-CS-1318 |

**SUPERVISED BY**

Supervisor: Zahid Javed

Co-Supervisor: Muhammad Naeem

**Department of Computer Science**

**National Textile University, Faisalabad**

# Certification

This is to certify that this project titled “**Real-Time Bus ID Verification and Tracking**” was found to satisfy the requirement for the award of a “**Bachelor of Sciences in Software Engineering**” degree by the Department of Computer Science, National Textile University.

|  |  |
| --- | --- |
| **Supervisor** |  |
| Zahid Javed | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_  Signature Date |
| **Co-Supervisor** |  |
| Muhammad Naeem | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_  Signature Date |
| **External Examiner** |  |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_  Signature Date |
| **Internal Examiner** |  |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_  Signature Date |
| **FYP-Convener** |  | |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_  Signature Date |
| **Head of Department** |  |
| Dr. | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_  Signature Date |

# Declaration

We hereby declared that this document is completely written by us, and it is totally our effort and none of anyone from outside of our group has copied it. This Report is purely written technically in accordance with our project.

**Group Members:**

Abubakar Nadeem \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_

21-NTU-CS-1291 Signature Date

E-Mail: abubakaranjum066@gmail.com

Imran Ali Niaz \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_

21-NTU-CS-1321 Signature Date

E-Mail:

Huzaifa Ahmad \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_

21-NTU-CS-1318 Signature Date

E-Mail:

**Acknowledgement**

First of all, In the Name of Allah, the Most Beneficent, the Most Merciful. All the praises and thanks be to Allah. A lot of love for our beloved Holy Prophet MUHAMMAD (S.A.W), his guidance always helps us to get the right path. After that, the preparation of this Software Requirements Specification (SRS) document was a collaborative effort involving Abubakar, Imran, and Huzaifa. The author would like to acknowledge and express gratitude to all individuals who directly or indirectly contributed to the completion of this document. Special thanks are extended to our Supervisor Zahid Javed and our Co-Supervisor Muhammad Naeem, who provided invaluable guidance and support throughout the SRS preparation process. The author would also like to acknowledge the invaluable support and suggestions provided by Zahid Javed. This SRS document would not have been possible without the support and cooperation of all involved individuals. The author is confident that this document will serve as a comprehensive guide for the development and implementation of the Real-Time Bus ID Verification and Tracking.

Signed by:

Abubakar Nadeem,

Imran Ali Niaz,

Huzaifa Ahmad,

Date:

00/00/2025

**Abstract**

Efficient and secure transportation is vital for educational institutions, where thousands of students rely on daily commutes. However, current systems face significant challenges, including inefficiencies, security vulnerabilities, and a lack of real-time tracking and communication. Issues such as overcrowding, unauthorized access, and poor resource utilization arise when students board buses without proper verification. Additionally, the absence of real-time bus monitoring and effective communication channels leads to confusion, delays, and suboptimal transport management.

The *Real-Time Bus ID Verification and Tracking* system addresses these challenges by integrating advanced technologies like RFID-based ID verification, live GPS tracking, and automated communication tools. The system ensures only authorized students access transport services, provides real-time visibility of bus locations for students, parents, and administrators, and enables timely updates regarding delays, route changes, or emergencies. It also tracks driver performance, monitors bus occupancy, and generates insights for optimizing route planning and capacity utilization.

By modernizing transport operations, the *Real-Time Bus ID Verification and Tracking* system enhances safety, boosts operational efficiency, and delivers a reliable and user-friendly commuting experience. This solution is an ideal choice for educational institutions aiming to transform their transportation infrastructure with secure and efficient technology.

**Table of Contents**

**Title**  **Page #**

[Certification II](#_Toc185949113)

[Declaration III](#_Toc185949114)

[List of Figures IX](#_Toc185949115)

[List of Tables XI](#_Toc185949116)

[List of Abbreviations XII](#_Toc185949117)

[CHAPTER 1 1](#_Toc185949118)

[1 Introduction 1](#_Toc185949119)

[1.1 Real-Time Bus ID Verification and Tracking System 1](#_Toc185949120)

[1.2 Reason to Develop 1](#_Toc185949121)

[1.2.1 Addressing Unique Institutional Needs 2](#_Toc185949122)

[1.2.2 Enhancing Safety and Security 2](#_Toc185949123)

[1.2.3 Bridging Communication Gaps 2](#_Toc185949124)

[1.2.4 Optimizing Resource Management 2](#_Toc185949125)

[1.2.5 Scalability and Adaptability 2](#_Toc185949126)

[1.3 Problem Statement 2](#_Toc185949127)

[1.4 Purpose 3](#_Toc185949128)

[1.5 Project Goals 3](#_Toc185949129)

[1.6 Objectives 3](#_Toc185949130)

[1.7 Project Scope 3](#_Toc185949131)

[1.8 Proposed Solution 4](#_Toc185949132)

[1.9 Project Scheduling 4](#_Toc185949133)

[CHAPTER 2 4](#_Toc185949134)

[2 Literature Review 4](#_Toc185949135)

[2.1 Related Work 4](#_Toc185949136)

[2.2 Overview of YOLO 6](#_Toc185949137)

[*2.2.1* *History of YOLO* 6](#_Toc185949138)

[*2.2.2* *Description and Working of YOLO* 8](#_Toc185949139)

[2.3 Types and Differences in YOLO Versions 9](#_Toc185949140)

[2.4 Benefits of Using YOLO 10](#_Toc185949141)

[CHAPTER 3 11](#_Toc185949142)

[3 System Requirements 11](#_Toc185949143)

[3.1 Functional Requirements: 12](#_Toc185949144)

[3.2 Non-Functional Requirements 13](#_Toc185949145)

[*3.2.1* *Security:* 13](#_Toc185949146)

[Implement robust authentication and authorization checks. 13](#_Toc185949147)

[*3.2.2* *Availability:* 13](#_Toc185949148)

[The system should be available 24/7. 13](#_Toc185949149)

[*3.2.3* *Scalability:* 13](#_Toc185949150)

[The system must be scalable to handle an increasing number of users and data. 13](#_Toc185949151)

[*3.2.4* *Usability:* 13](#_Toc185949152)

[User interface should be intuitive and easy to navigate. 13](#_Toc185949153)

[*3.2.5* *Maintainability:* 13](#_Toc185949154)

[The system should be easy to maintain and update with minimal downtime. 13](#_Toc185949155)

[3.3 Use Case Diagram 13](#_Toc185949156)

[3.3.1 Use Case of Sign In 14](#_Toc185949157)

[3.3.2 Use Case of Sign Up 14](#_Toc185949158)

[3.3.3 Use Case of Bus and Route Management 15](#_Toc185949159)

[3.3.4 Use Case of Journey Management 16](#_Toc185949160)

[3.3.5 Use Case of Bus card Management 16](#_Toc185949161)

[3.3.6 Use Case of Student & Session Management 17](#_Toc185949162)

[3.3.7 Use Case of Bus Staff Management 17](#_Toc185949163)

[3.3.8 Use Case of Complaint Management 18](#_Toc185949164)

[3.3.9 Use Case of Report and Analysis 18](#_Toc185949165)

[3.3.10 Use Case of Announcement Management 19](#_Toc185949166)

[3.3.11 Use Case of Complete System 20](#_Toc185949167)

[3.4 Use Case Description. 20](#_Toc185949168)

[3.4.1 Description of Sign In: 20](#_Toc185949169)

[3.4.2 Description of Sign Up: 21](#_Toc185949170)

[3.4.3 Description of Patient Account Creation: 21](#_Toc185949171)

[3.4.4 Description of Doctor Management: 22](#_Toc185949172)

[3.4.5 Description of Patient Admission: 22](#_Toc185949173)

[3.4.6 Description of Nursing and Patient Care: 22](#_Toc185949174)

[3.4.7 Description of Laboratory and Radiology: 23](#_Toc185949175)

[3.4.8 Description of Dispensary and Drugs Management: 23](#_Toc185949176)

[3.4.9 Description of Patient Payment System: 24](#_Toc185949177)

[3.4.10 Description of Drug and Inventory Management: 24](#_Toc185949178)

[3.4.11 Description of General Functionalities: 25](#_Toc185949179)

[Chapter 4 26](#_Toc185949180)

[4.1 Agile Software Development Methodology 26](#_Toc185949181)

[4.2 Selected Methodology: Agile 26](#_Toc185949182)

[4.3 Reasons for Selecting Agile Methodology 26](#_Toc185949183)

[4.4 Agile Project Planning and Execution for HME 27](#_Toc185949184)

[Chapter 5 28](#_Toc185949185)

[5 System Architecture 28](#_Toc185949186)

[5.1 Architecture 28](#_Toc185949187)

[5.2 Activity Diagram 29](#_Toc185949188)

[5.3 Sequence Diagram 34](#_Toc185949189)

[Chapter 6 36](#_Toc185949190)

[6 AI-Model and Algorithms 36](#_Toc185949191)

[6.1 General Disease Detection: 36](#_Toc185949192)

[6.1.1 Key Concepts of Decision Tree: 36](#_Toc185949193)

[6.1.2 Steps in Building a Decision Tree 36](#_Toc185949194)

[6.1.3 Training Epochs 37](#_Toc185949195)

[6.1.4 Architecture 37](#_Toc185949196)

[6.1.5 Confusion Matrix 38](#_Toc185949197)

[6.1.6 Accuracy 38](#_Toc185949198)

[6.1.7 Activity diagram of program 39](#_Toc185949199)

[6.2 Fractured Bone Detection: 40](#_Toc185949200)

[6.2.1 Dataset 40](#_Toc185949201)

[6.2.2 YOLO Working and Explanation 40](#_Toc185949202)

[6.2.3 Training Epochs 41](#_Toc185949203)

[6.2.4 Architecture Diagram: 41](#_Toc185949204)

[6.2.5 Confusion Matrix: 42](#_Toc185949205)

[6.2.1 Accuracy 42](#_Toc185949206)

[6.2.2 Activity Diagrams 42](#_Toc185949207)

[Chapter 7: System Implementation 44](#_Toc185949208)

[7.1 System Tools and Technology 44](#_Toc185949209)

[7.1.1 Django 44](#_Toc185949210)

[7.1.2 React 45](#_Toc185949211)

[7.1.3 MySQL 45](#_Toc185949212)

[7.1.4 VS Code 45](#_Toc185949213)

[7.2 Module Implementation 46](#_Toc185949214)

[7.2.1 Dashboard 46](#_Toc185949215)

[7.3 System Integration and Deployment 50](#_Toc185949216)

[7.3.1 Frontend-Backend Communication 50](#_Toc185949217)

[7.3.2 Database Management 52](#_Toc185949218)

[7.3.3 Deployment 52](#_Toc185949219)

[Chapter 8 53](#_Toc185949220)

[8 Screenshots 53](#_Toc185949221)

# List of Figures

**Description Page #**

[Figure 1.1 Gantt Chart .................................................................................................. 4](file://Users/multi/Documents/FYP.docx%23_Toc137710140)

[Figure 2.1](file://Users/multi/Documents/FYP.docx%23_Toc137710141) Historical Timeline of Research on HRMS [................................................ 5](file://Users/multi/Documents/FYP.docx%23_Toc137710141)

Figure 3.3.1 Use Case of Sign In.……………………. ……........................................14

Figure 3.3.2 Use Case of Sign Up ................................................................................14

Figure 3.3.3 Use Case of Patient Management .............................................................15

Figure 3.3.4 Use Case of Doctor Management ............................................................ 15

Figure 3.3.5 Use Case of Admission Management ..................................................... 16

Figure 3.3.6 Use Case of Nursing Care ....................................................................... 16

Figure 3.3.7 Use Case of Laboratory and Radiology .................................................. 17

Figure 3.3.8 Use Case of Dispensary and Medicine Management ............................. 18

Figure 3.3.9 Use Case of Patient Payment .................................................................. 18

Figure 3.3.10 Use Case of Payroll System ................................................................. 19

Figure 3.3.11 Use Case of Inventory Management ..................................................... 19

Figure 3.3.12 Use Case of General Functionality ........................................................ 20

Figure 3.3.13 Use Case of Complete System .............................................................. 21

Figure 5.1 Model, View and Template ........................................................................ 30

Figure 5.2.1 Activity of Admin Control ...................................................................... 31

Figure 5.2.2 Activity of Doctor, Nursing, Laboratory and Radiology Centre ............ 32

Figure 5.2.3 Activity of Dispensary, Pharmacy, HMO Authorization, Cashier ........... 33

Figure 5.2.4 Activity of Complete System .................................................................. 34

Figure 5.3.1 Sequence of Whole System ..................................................................... 35

Figure 5.3.2 Sequence of Interaction among APIs and Model .................................... 36

Figure 6.1.4.1 Architecture Diagram of Decision Tree …........................................... 38

Figure 6.1.7.1 Activity Diagram of G.D.D …............................................................. 40

Figure 6.2.1.1 Architecture diagram of working of YOLO......................................... 37

Figure 6.2.3.1 Activity diagram of YOLO................................................................... 44

Figure 7.3.1.1 Class Diagram of Hospital management System ................................. 52

Figure 7.3.1.2 Component Diagram of System ........................................................... 52

Figure 8.1 Main Dashboard.......................................................................................... 54

Figure 8.2 Patient Dashboard....................................................................................... 54

Figure 8.3 Patient Detail.............................................................................................. 55

Figure 8.4 Patient Appointment................................................................................... 55

Figure 8.5 Patient Report............................................................................................. 56

Figure 8.6 Nurse Vitals……….................................................................................... 56

Figure 8.7 Patient Data Update.................................................................................... 56

Figure 8.8 Pregnancy Report........................................................................................ 57

Figure 8.9 Rooms and Beds Allocation........................................................................ 57

Figure 8.10 OPD Clinic............................................................................................... 58

Figure 8.11 Authorization……………........................................................................ 58

Figure 8.12 General Disease AI Model....................................................................... 59

Figure 8.13 Patient Vitals Detail.................................................................................. 59

Figure 8.14 Laboratory................................................................................................. 60

Figure 8.15 Laboratory Tests....................................................................................... 60

Figure 8.16 Laboratory Tests Reports.......................................................................... 61

Figure 8.17 User Management..................................................................................... 61

# List of Tables

**Description Page#**

Table 3.4.1 Description of Sign Up ............................................................................. 22

Table 3.4.2 Description of Sign In .............................................................................. 22

Table 3.4.3 Description Patient Account Creation ....................................................... 23

Table 3.4.4 Description of Doctor Management ......................................................... 23

Table 3.4.5 Description of Patient Admission …………............................................ 24

Table 3.4.6 Description of Nursing Care .................................................................... 24

Table 3.4.7 Description of Laboratory and Radiology ................................................ 25

Table 3.4.8 Description of Dispensary and Medicine Management ........................... 25

Table 3.4.9 Description of Patient Payment ................................................................ 26

Table 3.4.10 Description of Inventory Management System ..................................... 26

Table 3.4.11 Description General Functionalities ....................................................... 27

Table 6.1.5.1 Confusion Matrix of General Disease Detection .................................. 39

Table 6.2.5.1 Confusion Matrix of YOLO ................................................................. 43

# 

# List of Abbreviations

|  |  |
| --- | --- |
| RT-BIVT | Real-Time Bus ID Verification and Tracking |
| RFID | Radio Frequency Identification |
| RT-BIVT system | Admin Panel, Driver and Student App’s |
|  |  |
|  |  |

# CHAPTER 1

# Introduction

Efficient and secure transportation systems are crucial for educational institutions of all sizes, where thousands of students rely on daily commutes. However, many institutions face significant challenges, including inefficiencies, security vulnerabilities, and a lack of real-time tracking and communication. Students often board buses without proper verification, leading to overcrowding, unauthorized access, and inequitable usage of transport resources. Additionally, the absence of real-time monitoring for buses and driver performance, coupled with ineffective communication channels, results in confusion, delays, and poor management of transport operations.

To address these challenges, we propose a **RT-BIVT** system tailored for educational institutions. This system combines advanced technologies like **RFID**-based ID verification, live **GPS** tracking, and automated communication tools to streamline transportation management. It ensures that only authorized students access transportation services, provides real-time visibility of bus locations for students, parents, and administrators, and facilitates timely updates on delays, route changes, or emergencies. Additionally, the system tracks driver performance, monitors bus occupancy, and generates alerts to optimize route planning and capacity utilization.

This innovative solution goes beyond solving existing challenges—it transforms transportation management into a secure, efficient, and user-friendly ecosystem. By integrating modern technologies, the **RT-BIVT** system enhances safety, improves operational efficiency, and delivers a reliable transportation experience, making it an ideal choice for any educational institution seeking to modernize its transport infrastructure.

## Real-Time Bus ID Verification and Tracking System

The **RT-BIVT** system is an innovative solution designed to enhance transportation efficiency and safety for educational institutions. It integrates cutting-edge technologies like **RFID** for ID verification, live **GPS** tracking, and an advanced communication framework to create a streamlined, user-friendly transportation management ecosystem. This system empowers institutions to optimize bus routes, manage capacity, and monitor driver performance, ensuring a secure and efficient transport experience for students, parents, and administrators.

## Reason to Develop

The development of the RT-BIVT system is driven by several compelling factors, despite the existence of other transportation solutions:

### Addressing Unique Institutional Needs

Educational institutions often face specific challenges like unauthorized access, overcrowding, and inefficient communication. Generic transport management systems fail to meet these tailored needs, prompting the creation of a solution that directly addresses the unique requirements of such environments.

### Enhancing Safety and Security

Ensuring that only authorized students use transport facilities is crucial for security and fairness. RFID-based verification provides a reliable, real-time solution to mitigate misuse and enhance safety for students and drivers.

### Bridging Communication Gaps

A lack of effective communication between transport departments, parents, and students leads to confusion and inefficiency. The proposed system integrates real-time notifications for updates like delays, route changes, and emergencies, ensuring timely and clear communication.

### Optimizing Resource Management

Overcrowding or underutilization of buses is a common issue. By integrating occupancy monitoring and route optimization, this system helps administrators allocate resources more efficiently, reducing costs and enhancing comfort.

### Scalability and Adaptability

The system is designed to adapt to the specific challenges of any educational institution, whether managing a small fleet or a large-scale transport network. It is scalable, ensuring that institutions can continue to benefit as their transportation needs evolve.

This project is more than a technological upgrade; it is a strategic initiative to modernize transport systems, improve safety, and deliver a seamless experience for all stakeholders in an educational environment.

## Problem Statement

Educational institutions face significant challenges in managing their transportation systems effectively. Common issues include overcrowded or underutilized buses, unauthorized access, inefficient communication, and a lack of real-time tracking. These problems result in confusion, operational inefficiencies, and safety concerns for students, parents, and administrators. Existing solutions often lack the integration and adaptability required to address these specific institutional needs.

## Purpose

The primary purpose of the RT-BIVT system is to provide a seamless, all-in-one solution for transportation management in educational institutions. By combining secure access verification, live GPS tracking, and advanced communication tools, the system aims to improve operational efficiency, enhance student safety, and streamline communication among all stakeholders.

## Project Goals

* Implement secure ID verification to prevent unauthorized access.
* Provide live bus location tracking for real-time visibility and improved coordination.
* Enable real-time notifications for updates such as delays, route changes, or emergencies.
* Track key metrics like speed, stop intervals, and adherence to schedules.
* Streamline bus scheduling, capacity management, and route planning.
* Ensure an intuitive interface for administrators, parents, and students.

## Objectives

Objectives of the project are as follows:

* Integrate RFID-based ID verification to ensure only authorized users board the buses.
* Provide GPS-enabled tracking for buses accessible to students, parents, and administrators.
* Enable alerts for overcrowding or underutilization to optimize bus capacity.
* Monitor driver behaviour to ensure adherence to safety and efficiency standards.
* Develop a mobile app for notifications and updates to keep all stakeholders informed.

## Project Scope

Although designed for educational institutions, this system can be extended to other domains such as corporate transport, public transit, or private bus fleets. Its modular design allows for customization and scalability, making it suitable for varying transportation requirements and operational complexities.

## Proposed Solution

The system offers a modular and comprehensive approach to transportation management. It features real-time ID verification using **RFID**, live **GPS** tracking for buses, occupancy alerts, driver performance monitoring, and mobile apps for seamless communication. By adopting this system, institutions can significantly enhance operational efficiency, reduce resource wastage, and ensure a safer, more reliable transport experience for all users.

## Project Scheduling

Here is the Gantt chart for the RT-BIVT project. This chart visually represents the project timeline, including the start and end dates for each activity. It provides a high-level overview of how the project tasks are scheduled over time, aiding in effective project management and tracking. The timeline of the project is shown by the Gantt chart in Figure 1.1.

*Figure 1.1 Gantt Chart*

**CHAPTER 2**

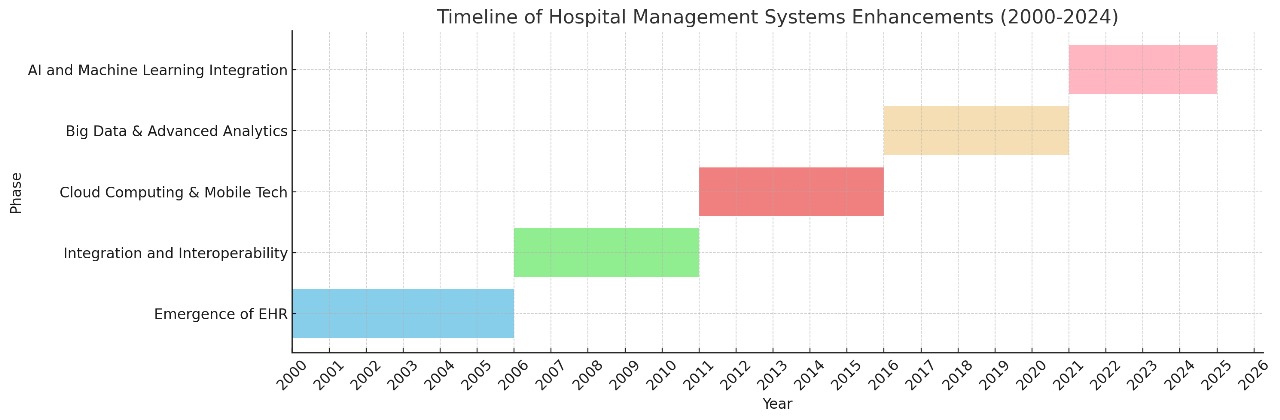
1. **Literature Review**

In the area of hospital management, the integration of synthetic intelligence (AI) in healthcare systems represents a large development. This bankruptcy evaluations the literature surrounding AI packages in health facility control, focusing on AI-driven diagnostics, patient information management, and operational performance. The relevance of AI in healthcare is comparable to the importance of biometric credentials, which includes handwritten signatures, in record and individual authentication [1].

* 1. **Related Work**

The evolution and development of AI-pushed health center control systems may be traced again several decades, paralleling the studies trajectory of offline handwritten signature reputation [2]. Since the early integration of facts generation in healthcare, there were continuous efforts to decorate medical institution control thru numerous technological method, inclusive of AI [3]. Below, a timeline is presented to highlight significant milestones in the development of AI applications in hospital management Figure 2.1.

Timeline of Significant Events:



*Figure 2.1 Historical Timeline of Research of HRMS*

***2.1.1 Early 1990s:***

The preliminary ranges of IT integration in healthcare start, focusing normally on digital report-maintaining [4].

***2.1.2 Late 1990s:***

Emergence of virtual equipment for handling patient statistics, scheduling, and primary health facility operations [5].

***2.1.3 Early 2000s:***

Introduction of extra state-of-the-art clinic information structures (HIS), incorporating database management for large records units [6].

***2.1.4 Mid-2000s:***

Initial AI implementations in diagnostic equipment, basically in imaging and laboratory tests [7].

***2.1.5 Late 2000s:***

Expansion of AI packages in predictive analytics for patient care and operational planning [2].

***2.1.6 Early 2010s:***

Development of AI fashions for precise disease detection, along with the first instances of AI in radiology and pathology [8].

***2.1.7 Mid-2010s:***

Integration of AI in digital health facts (EHR) for more suitable information analysis and decision assist [9].

***2.1.8 Late 2010s:***

Widespread adoption of AI for various clinic control functions, such as aid allocation, affected person drift optimization, and customized treatment plans [3].

***2.1.9 Early 2020s:***

Introduction of superior AI algorithms, which includes deep learning, in diagnostics, appreciably improving accuracy and efficiency [10].

***2.1.10 Present:***

The Hospital Management Ecosystem and AI-Diagnostics represents the end result of those trends, integrating diverse AI functionalities into a comprehensive, consumer-friendly machine for hospital administration and affected person care [11].

This timeline illustrates the revolutionary integration of AI in clinic control, highlighting how technological advancements have incrementally converted healthcare operations, much like the evolution visible in offline signature popularity research [2]. It presents a ancient context to the development of the Hospital Management Ecosystem and AI-Diagnostics, underscoring its significance as a brand new answer in healthcare era [12].

* 1. **Overview of YOLO**
     1. ***History of YOLO***

YOLO (You Only Look Once) is a circle of relatives of item detection fashions famend for his or her velocity and accuracy. The evolution of YOLO has substantially impacted the field of computer imaginative and prescient, specifically in actual-time item detection. Here is an in depth records of its development:

* **YOLOv1 (2015)**:

Introduced by way of Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi in their seminal paper "You Only Look Once: Unified, Real-Time Object Detection." YOLOv1 proposed a singular approach to object detection via framing it as a unmarried regression trouble. Unlike conventional methods like R-CNN, which worried complicated pipelines with more than one tiers (region proposal, feature extraction, and class), YOLOv1 carried out item detection in a single forward skip through the network. This significantly improved the rate of detection, allowing real-time processing [13].

* **YOLOv2 (2016)**:

Also known as YOLO9000, this version introduced numerous enhancements to the authentic version. The improvements protected batch normalization, high-resolution classifier, anchor packing containers, and size clusters. These adjustments stepped forward both the accuracy and speed of the version. YOLO9000 additionally introduced the idea of the usage of both classified and unlabeled facts to increase the detection abilities to over 9000 item categories [14].

* **YOLOv3 (2018)**:

YOLOv3 similarly stepped forward the architecture by way of introducing a deeper and extra efficient spine network called Darknet-53. This model additionally applied multi-scale predictions, allowing better detection of objects at special scales. Logistic regression become used for class prediction, enhancing the version's performance on complicated datasets [15].

* **YOLOv4 (2020)**:

Developed via Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao, YOLOv4 incorporated a large number of advancements, including mosaic records augmentation, self-antagonistic training, and go-stage partial connections (CSP). These improvements substantially boosted the model's accuracy and efficiency [16].

* **YOLOv5 (2020)**:

Although not formally released by the unique authors, YOLOv5, developed by using Ultralytics, became popular for its user-pleasant implementation and enhanced performance. YOLOv5 targeted on ease of training and deployment, making it a popular choice for each researchers and practitioners [17].

* **YOLOv6, YOLOv7, and YOLOv8**:

These subsequent variations persisted to construct on the improvements in their predecessors. Each version brought optimizations in structure, education strategies, and alertness-precise diversifications, in addition solidifying YOLO's recognition as a main item detection framework [18].

* + 1. ***Description and Working of YOLO***

YOLO's key innovation lies in treating object detection as a unmarried regression trouble. This technique appreciably simplifies the detection pipeline and complements speed. Here is an in depth clarification of the way YOLO works:

**1. Input Image Division**

The enter photograph is split into an S×SS instances SS×S grid. Each grid cell is accountable for detecting gadgets whose center falls in the cell. This division enables the version recognition on unique regions of the image concurrently.

**2. Bounding Box Prediction**

* Each grid cellular predicts a fixed quantity of bounding boxes (B). For every bounding container, the model predicts the subsequent:
* Four coordinates representing the middle (x, y), width, and peak of the container.
* A self belief rating indicating the presence of an object within the box and the accuracy of the bounding field.

**3. Class Prediction**

For each grid cell, a hard and fast of class probabilities is predicted. These possibilities suggest the likelihood of each magnificence being gift within the bounding field. The class prediction is unbiased of the wide variety of bounding bins predicted through the cell.

**4. Output Tensor**

The very last output is a tensor of shape S×S×(B×5 C)S times S instances (B times five C)S×S×(B×5 C), in which:

• S×SS instances SS×S represents the grid cells.

• B×5B times 5B×5 represents the bounding box predictions (4 coordinates 1 self assurance score).

• CCC represents the wide variety of classes.

The model outputs predictions for each grid mobile, and submit-processing steps (like non-max suppression) are used to filter out and refine those predictions.

* 1. **Types and Differences in YOLO Versions**

Each version of YOLO has introduced significant improvements and optimizations. Here's a detailed comparison of the different YOLO versions:

**YOLOv1**

* **Architecture:** Single convolutional community for stop-to-cease predictions.
* **Strengths:** Fast and capable of actual-time detection.
* **Limitations:** Lower accuracy for smaller items due to coarse grid division and the version's incapacity to handle more than one items nicely within a single grid mobile.

**YOLOv2 (YOLO9000)**

**• Improvements:** Introduced batch normalization, anchor containers, better decision classifier, and size clusters.

**• Strengths:** Better accuracy and speed, able to detecting over 9000 item categories.

**• Limitations:** While advanced, still confronted challenges with very small items and overlapping gadgets in densely packed scenes.

**YOLOv3**

**• Architecture:** Darknet-53 backbone, multi-scale predictions.

**• Improvements:** Introduced logistic regression for class prediction and used three one-of-a-kind scales for predictions, enhancing the detection of small, medium, and large items.

**• Strengths:** Significantly progressed accuracy and robustness in numerous detection scenarios.

**• Limitations:** Increased computational complexity in comparison to YOLOv2.

**YOLOv4**

**• Improvements:** Advanced strategies like mosaic statistics augmentation, self-adversarial schooling, and CSPDarknet53 backbone.

**• Strengths:** Achieved state-of-the-art overall performance with a terrific stability among velocity and accuracy.

**• Limitations:** More complex architecture requiring extra computational sources for training.

**YOLOv5**

**• Developed via:** Ultralytics.

**• Improvements:** Focused on ease of use, higher support for training and deployment, and improvements in each velocity and accuracy.

**• Strengths:** Lightweight, notably efficient, and smooth to teach and set up.

**• Limitations:** Not an legitimate continuation from the unique YOLO authors, leading to a few debate within the community.

**YOLOv6, YOLOv7, YOLOv8**

**• Improvements:** Continued optimizations in structure, schooling techniques, and application-particular variations.

**• Strengths:** Enhanced adaptability and performance throughout special tasks and devices.

**• Limitations:** Increased complexity and computational necessities, relying at the unique model and application.

* 1. **Benefits of Using YOLO**

YOLO's innovative approach to object detection offers several significant benefits over traditional methods:

**1. Speed**

* **Real-time Detection**: YOLO's unmarried-pass detection permits it to technique snap shots in real-time, making it one of the fastest object detection algorithms available. This is critical for packages like autonomous using, robotics, and stay video analysis.

**2. Simplicity**

* **End-to-End Training**: YOLO simplifies the object detection pipeline with the aid of the usage of a unmarried convolutional network to are expecting bounding boxes and class chances immediately. This reduces the complexity and capability mistakes associated with multi-stage pipelines used by different techniques.

**3. Accuracy**

* **Enhanced Performance**: With every model, YOLO has integrated new techniques and optimizations that enhance its accuracy. YOLOv4 and YOLOv5, as an example, offer today's overall performance in terms of each precision and don't forget.

**4. Single Model for Multiple Tasks**

* **Unified Approach**: YOLO can concurrently predict a couple of bounding bins and their respective magnificence chances, casting off the want for separate fashions or complex integration steps.

**5. Real-time Applications**

* **Low Latency**: YOLO's capability to carry out real-time detection makes it best for packages wherein low latency is essential, such as surveillance systems, independent motors, and augmented truth.

**6. Flexibility**

* **Adaptability**: YOLO's non-stop development and style of variations make it notably adaptable. It may be tailored to one-of-a-kind tasks and environments, from detecting small gadgets to running in aid-restrained settings.

**CHAPTER 3**

1. **System Requirements**

In this bankruptcy, all of the useful requirements of the application and the overall requirement of the stockholders are documented as it’s an important a part of a mission or product that allows to satisfy stakeholder’s necessities. Now, we can speak system necessities, practical necessities, software program development, and present and selected methodology with the purpose of technique. These sections describe software program methodologies which are present and decided on for this assignment with the glide of machine and alertness detail depicted.

* 1. **Functional Requirements:**

***3.1.1 Dashboard Management:***

Display real-time health facility operations metrics and critical notifications. Provide a summary view of each day appointments, admissions, and discharges.

***3.1.2 Patient Management:***

Enable complete seek capability for patient records. Facilitate appointment scheduling, monitoring, and updates. Manage designated affected person come across histories inclusive of go to notes and treatment records.

***3.1.3 Medical Records Management:***

Securely upload and store patient files and medical documents. Ensure quick retrieval of patient encounter reports and medical histories.

***3.1.4 Nursing Station Functionality:***

Record and monitor patient vitals with alert structures for abnormal readings. Manage and log nursing sports consisting of injections, dressings, and procedures. Handle requisitions for medical substances and song inventory.

***3.1.5 Physician and Clinical Services:***

Support outpatient clinic operations with scheduling and patient waft management. Facilitate the creation of virtual patient stumble upon reviews. Integrate with laboratory and radiology statistics systems for check ordering and result retrieval.

***3.1.6 Laboratory Information System:***

Manage incoming test requests and track pending orders. Record and disseminate laboratory outcomes to applicable departments. Provide functionalities for take a look at reporting, management, and categorization.

***3.1.7 Radiology Department Integration:***

Track radiology requests including X-rays, CT scans, ultrasounds, and ECGs. Manage the workflow from take a look at requests to end result reporting.

***3.1.8 Pharmacy and Dispensary Operations:***

Oversee outpatient and inpatient dispensary features. Manage pharmacy inventory inclusive of drug expiry and utilization reports. Handle buy orders and affirm receipt of medicine.

***3.1.9 Financial Transactions Handling:***

Process billing for healthcare services and generate comprehensive reviews. Manage cashier operations together with deposits, costs, and sales tracking.

***3.1.10 Disease Diagnostics***:

Implement AI algorithms to analyse affected person data for diagnostics and remedy guidelines.

***3.1.11 Predictive Analytics:***

Utilize AI for predictive analytics to get better diagnostic consequences.

* 1. **Non-Functional Requirements** 
     1. ***Security:***

Implement robust authentication and authorization checks.

* + 1. ***Availability:***

The system should be available 24/7.

* + 1. ***Scalability:***

The system must be scalable to handle an increasing number of users and data.

* + 1. ***Usability:***

User interface should be intuitive and easy to navigate.

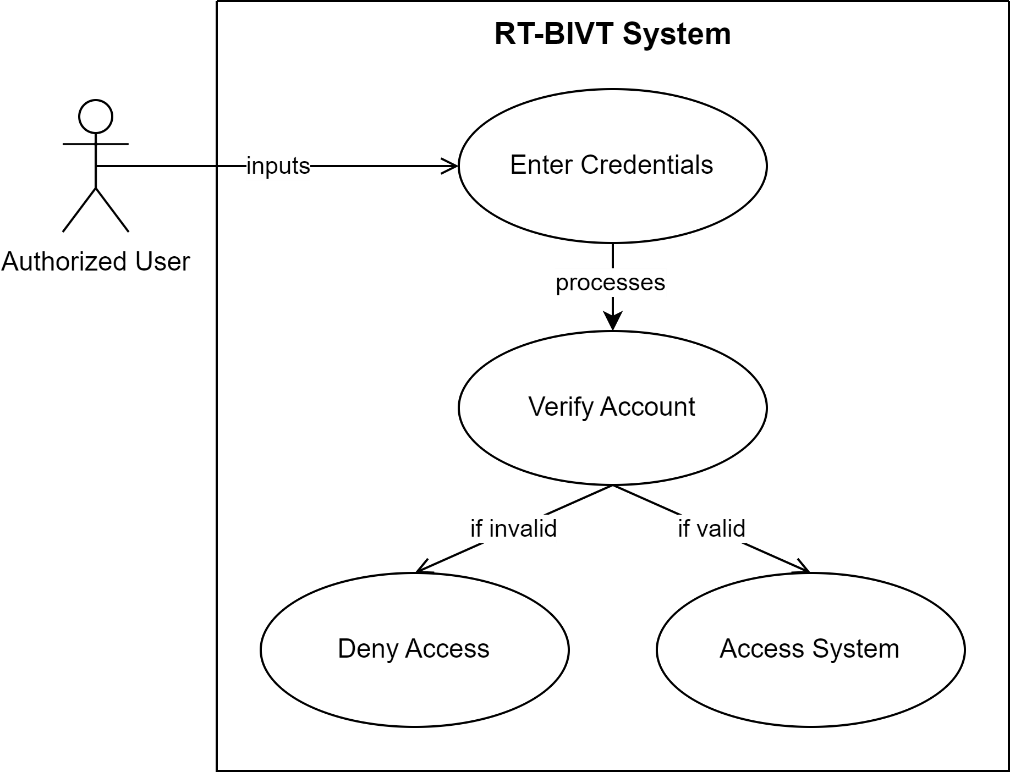
* + 1. ***Maintainability:***

The system should be easy to maintain and update with minimal downtime.

## Use Case Diagram

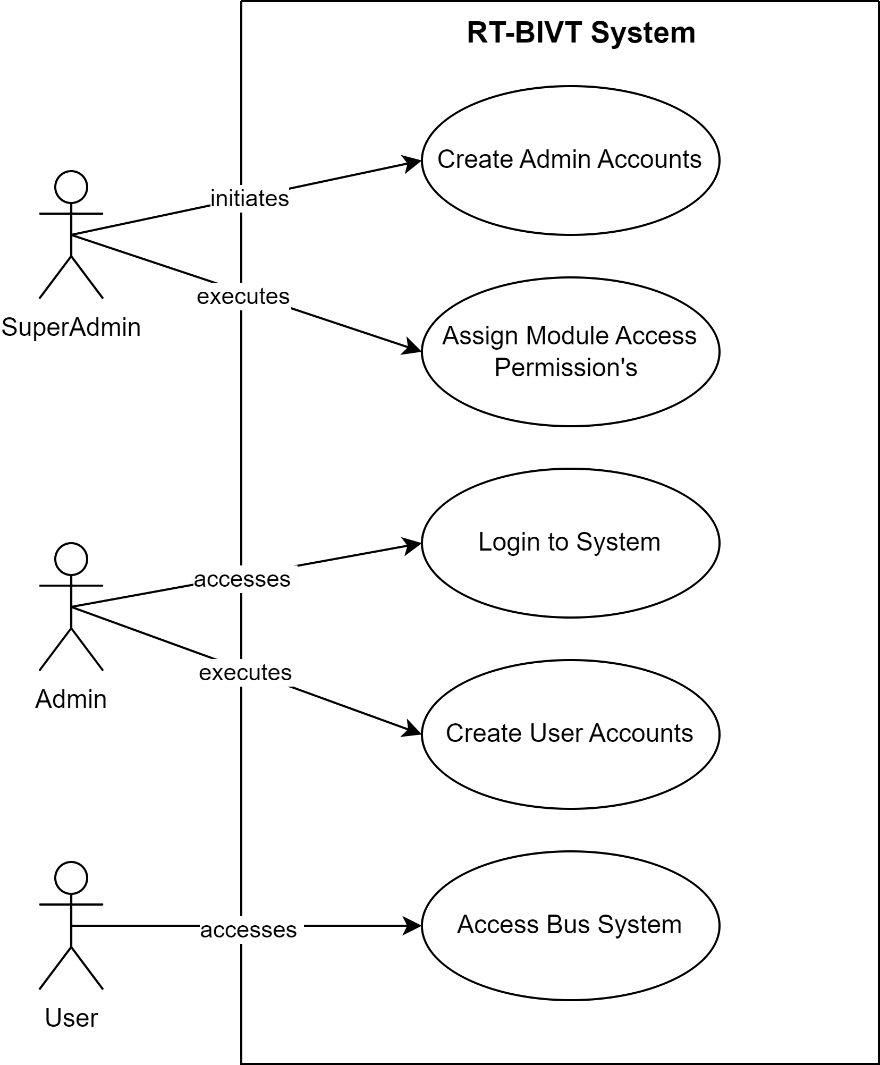
For graphical visualization of actor interaction with the components of the systems, the most appropriate approach is to use case diagrams that graphically represent which actor may perform or access which functionality or component of the system.

### Use Case of Sign In



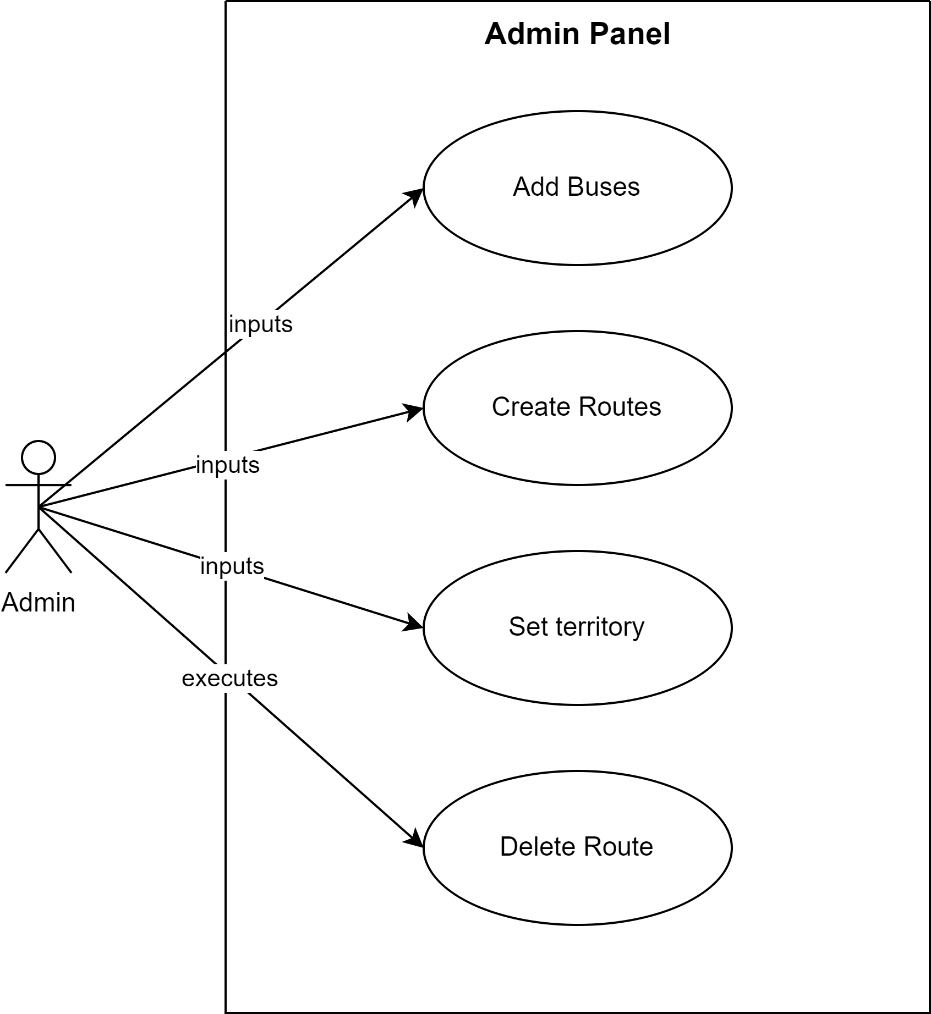
*Figure 3.3.1 Use Case of Sign In*

### Use Case of Sign Up



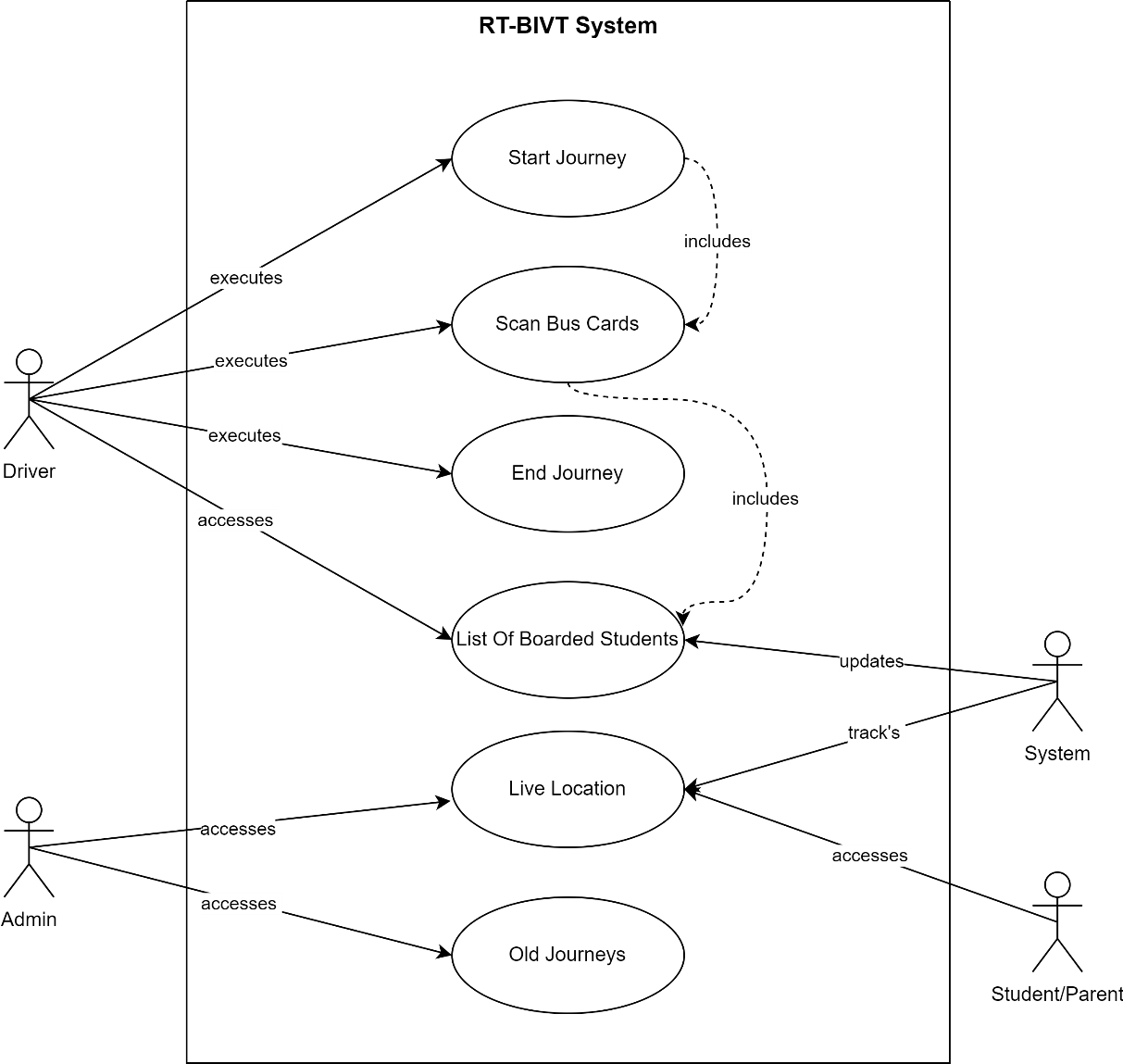
*Figure 3.3.2 Use Case of Sign Up*

### Use Case of Bus and Route Management

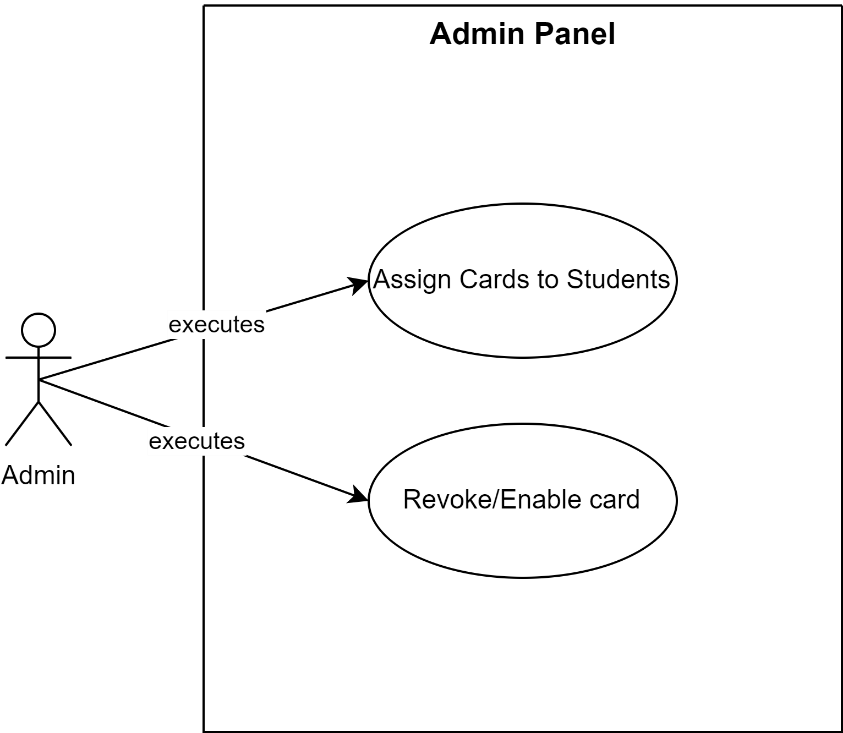


*Figure 3.3.3 Use Case of Bus and Route Management*

### Use Case of Journey Management

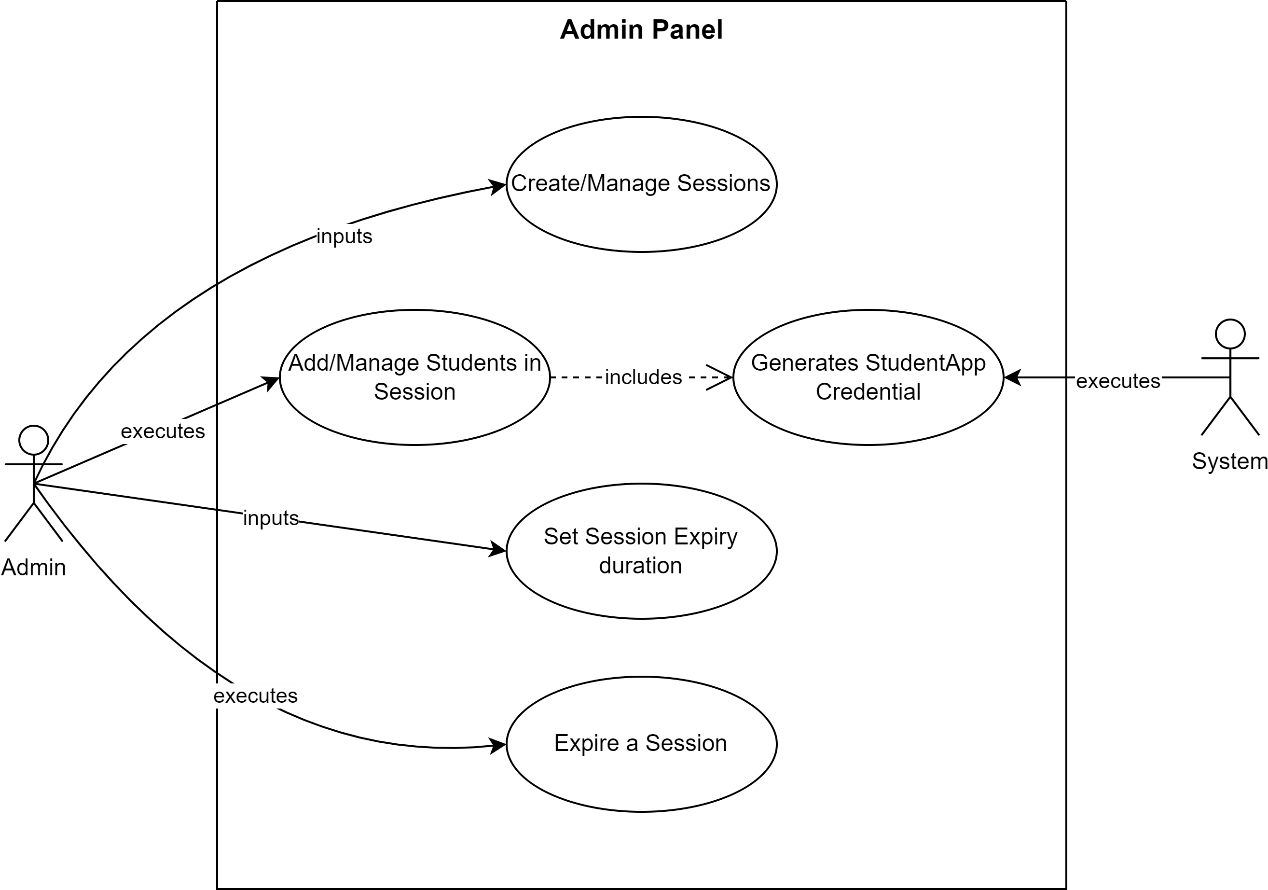
*Figure 3.3.4 Use Case of Journey Management*

### Use Case of Bus card Management



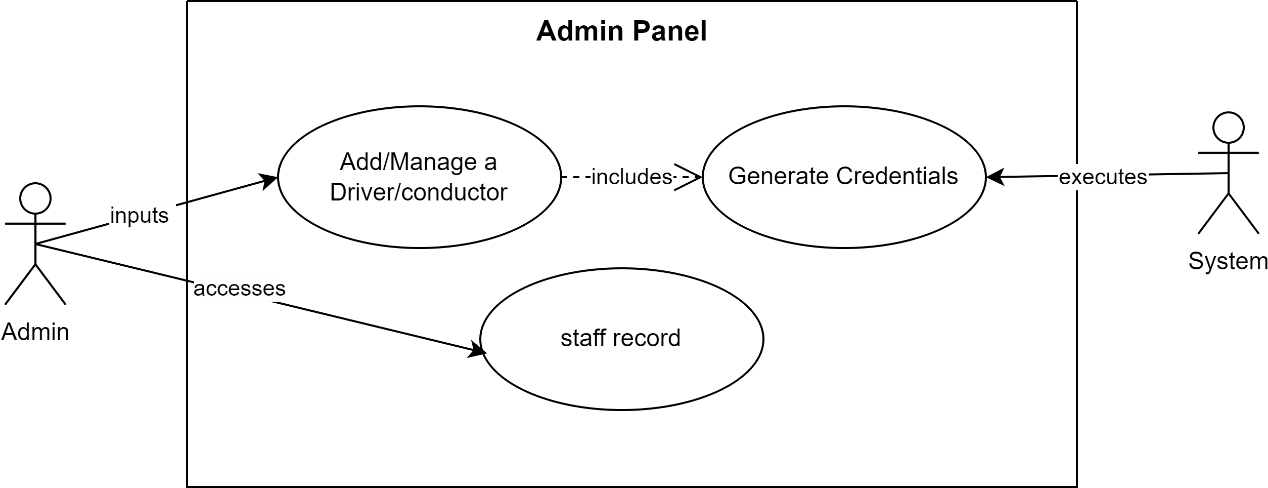
*Figure 3.3.5 Use Case of Bus Card Management*

### Use Case of Student & Session Management



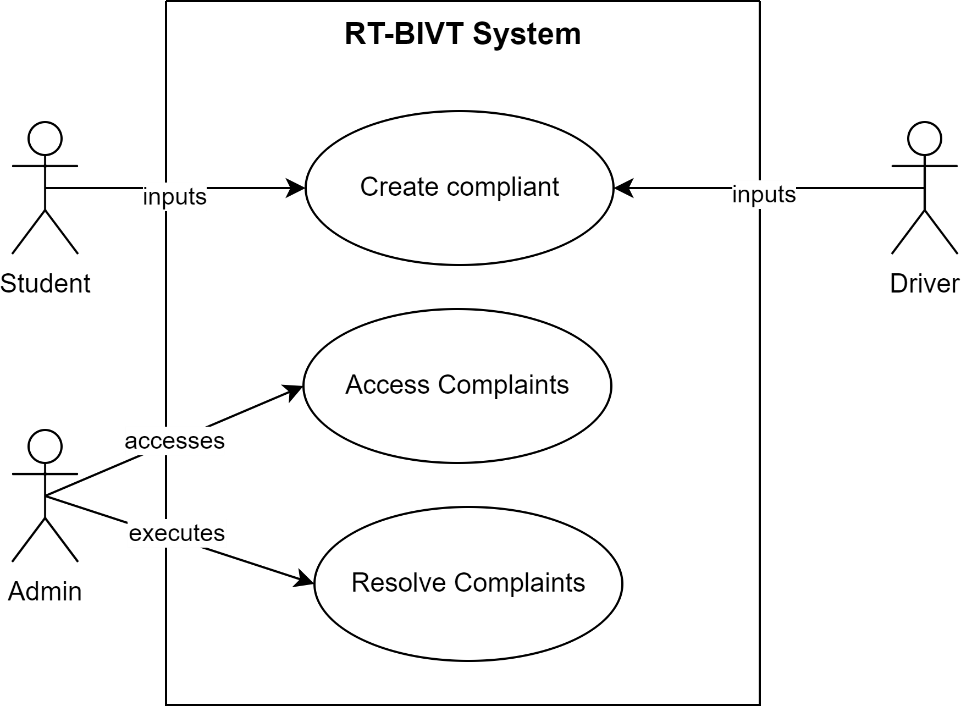
*Figure 3.3.6 Use Case of Student & Session Management*

### Use Case of Bus Staff Management



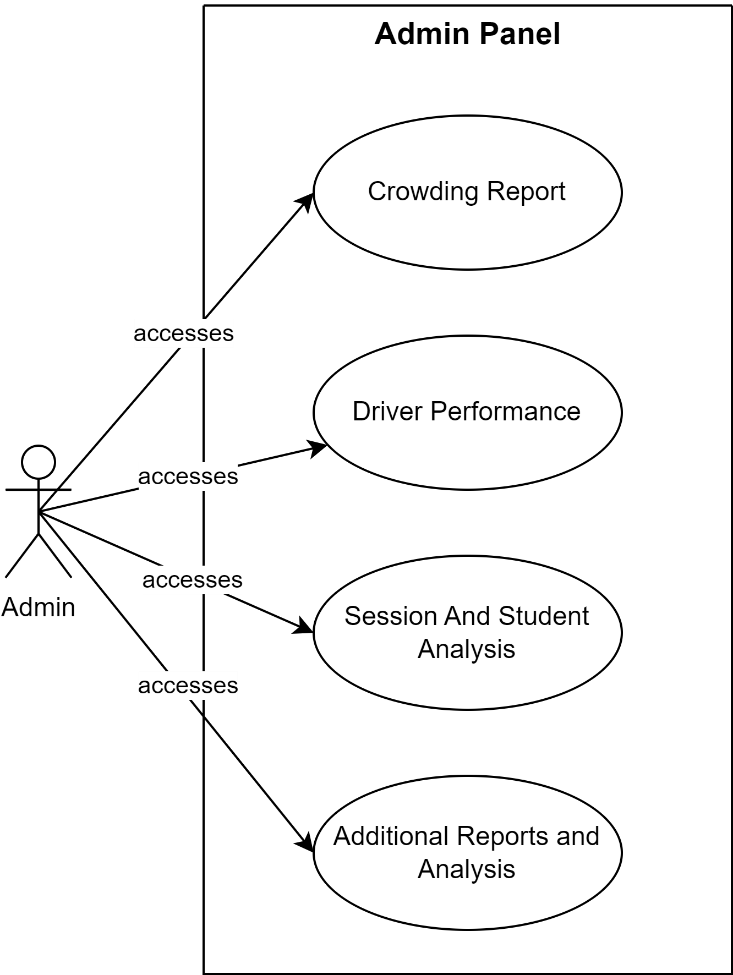
*Figure 3.3.7 Use Case of Bus Staff Management*

### Use Case of Complaint Management

**

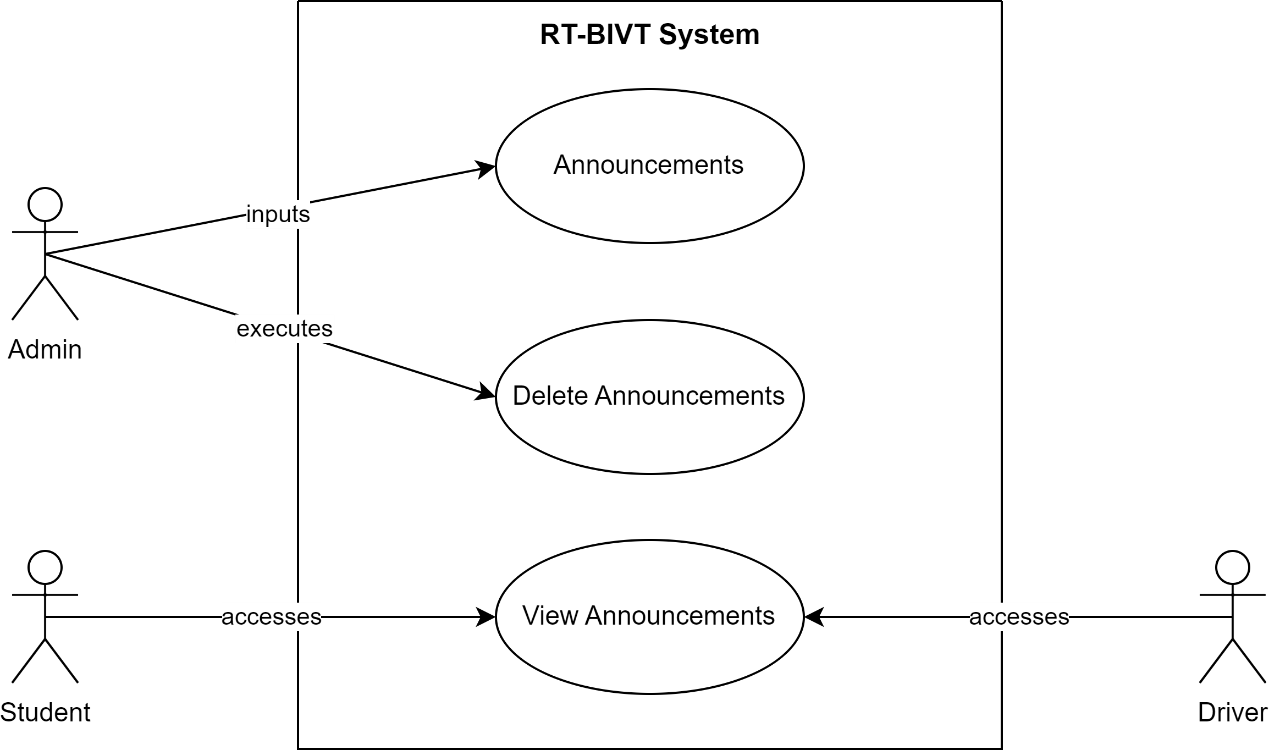
*Figure 3.3.8 Use Case of Complaint Management*

### Use Case of Report and Analysis



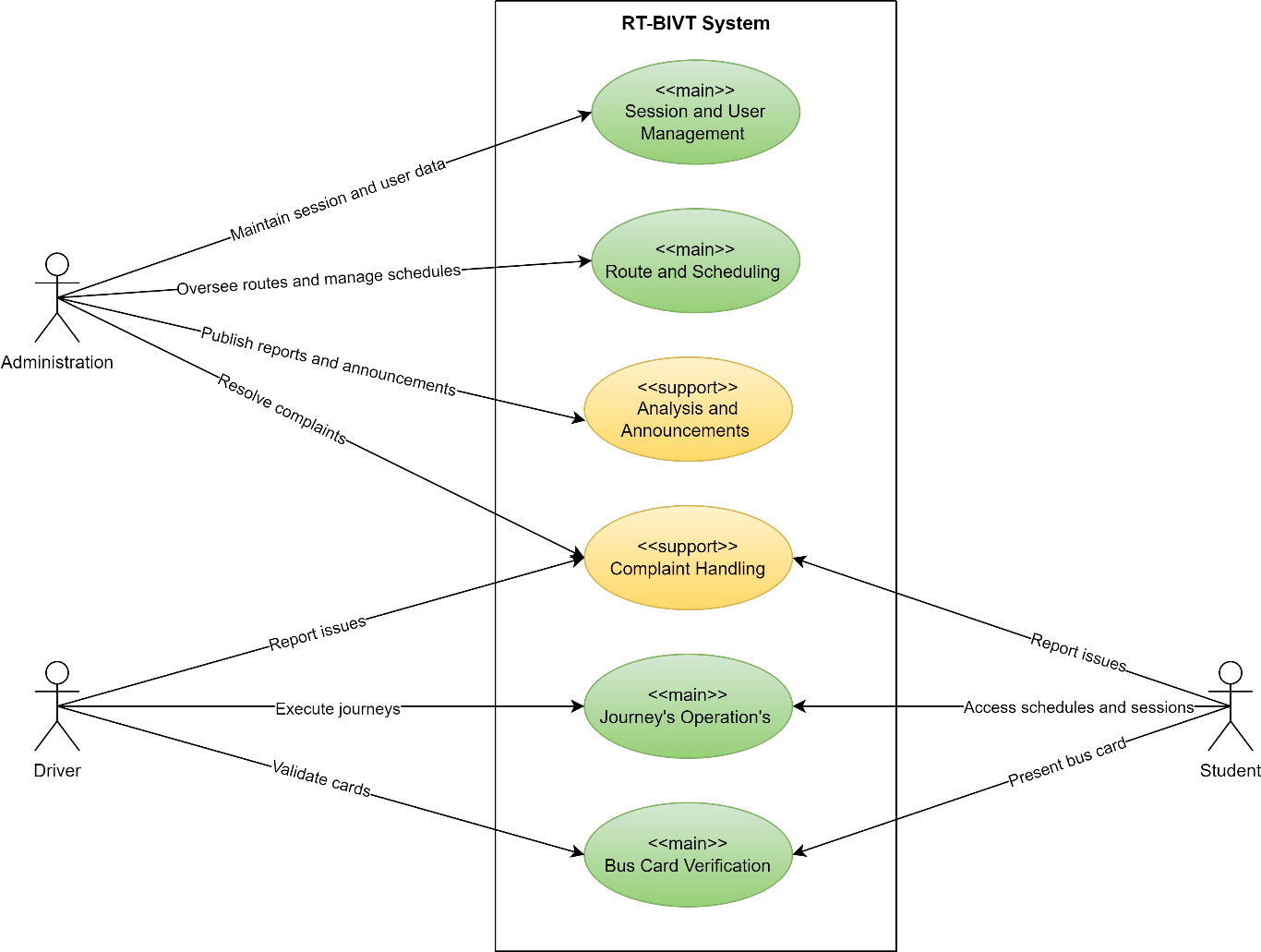
*Figure 3.3.9 Use Case of Report and Analysis*

### Use Case of Announcement Management



*Figure 3.3.10 Use Case of Announcement Management*

### Use Case of Complete System

**

*Figure 3.3.12 Use Case of Complete System*

## Use Case Description.

Use case description contains every piece of information (use case id, use case name, description, pre- and post-conditions) of each use case.

### Description of Sign In:

|  |  |
| --- | --- |
| Field | Details |
| Use Case Id | 01 |
| Use Case Name | Sign In |
| Actors | Authorized User |
| Description | Authorized users log in to the RT-BIVT via provided applications. Access is denied if the user does not have an account or proper credentials. |
| Pre-condition | User must have an account and credentials. |
| Post-condition | User logs into the RT-BIVT system or is denied access. |

*Table: 3.4.1 Description of Sign In*

### Description of Sign Up:

|  |  |
| --- | --- |
| Field | Details |
| Use Case Id | 02 |
| Use Case Name | Sign Up |
| Actors | Super Admin, Authorized User |
| Description | A Super Admin has the ability to create admins and assign specific roles and permissions to each. These admins are tasked with managing the creation of user accounts, including those for bus staff and students. |
| Pre-condition | User must have an account, enough permissions and credentials to access system. |
| Post-condition | A new user is created with specific roles and permissions to operate with in RT-BIVT. |

*Table: 3.4.2 Description of Sign Up*

### Description of Bus and Route Management:

*Table: 3.4.3 Description of Bus and Route Management*

|  |  |
| --- | --- |
| Field | Details |
| Use Case Id | 03 |
| Use Case Name | Bus and Route Management |
| Actors | Admin |
| Description | The admin is responsible for adding buses, defining routes, and assigning territories. |
| Pre-condition | Admin must be authenticated and have enough permissions to manage buses and routes. |
| Post-condition | The admin effectively oversees the management of buses, routes, and territories. |

### Description of Journey Management:

*Table: 3.4.4 Description of Journey Management*

|  |  |
| --- | --- |
| Field | Details |
| Use Case Id | 04 |
| Use Case Name | Journey Management |
| Actors | Admin, Driver, Student |
| Description | The driver starts and ends journeys while verifying student bus cards during the trip. The system logs journey-related data and tracks the bus's live location, enabling administration, students, and parents to access journey details and monitor the bus's real-time location. |
| Pre-condition | The driver must be authenticated and have specified the route and the bus they are assigned to. The student must possess a valid and active bus card. |
| Post-condition | The driver successfully operates the journeys, while the system tracks and updates the journey details and live bus location. This allows both administrators and students/parents to access the journey information and the live location of the bus. In the event of a failure, the system displays relevant error messages and logs the errors. If a card is invalid or unreadable, the system shows an error and does not mark the student as boarded. If the system goes offline, both the administration and students will be unable to access the journey data. |

### Description of Patient Admission:

*Table: 3.4.5 Description of* *Patient Admission*

|  |  |
| --- | --- |
| Field | Details |
| Use Case Id | 05 |
| Use Case Name | Patient Admission Management |
| Actors | Authorized User (Admission Management) |
| Description | Manage patient admissions by assigning beds and rooms and performing CRUD operations on them. |
| Pre-condition | Patient is admitted. |
| Post-condition | Patient is assigned a bed and room in the hospital. |

### Description of Nursing and Patient Care:

*Table: 3.4.6 Description of Nursing and Patient Care*

|  |  |
| --- | --- |
| Field | Details |
| Use Case Id | 06 |
| Use Case Name | Nursing and Patient Care |
| Actors | Nurse |
| Description | Nurses add patient vitals, manage injections, dressings, and immunizations. They redirect patients to doctors if needed. |
| Pre-condition | Patient is admitted under nursing care. |
| Post-condition | Nursing care provided, and relevant updates made to the patient's record. |

### Description of Laboratory and Radiology:

*Table: 3.4.7 Description of* *Laboratory and Radiology*

|  |  |
| --- | --- |
| Field | Details |
| Use Case Id | 07 |
| Use Case Name | Laboratory and Radiology |
| Actors | Doctor, Laboratory/Radiology Technicians |
| Description | Doctors send patients for lab tests. Technicians conduct tests and upload results. AI models predict bone fractures from X-rays. |
| Pre-condition | Doctor refers a patient to laboratory/radiology. |
| Post-condition | Tests conducted and results uploaded for doctor and nurse access. |

### Description of Dispensary and Drugs Management:

|  |  |
| --- | --- |
| Field | Details |
| Use Case Id | 08 |
| Use Case Name | Dispensary and Drug Management |
| Actors | Dispensary Staff |
| Description | After patient discharge, the patient is sent to the dispensary where prescribed medicines are provided. A smooth supply chain from the pharmacy to various departments ensures medication availability. |
| Pre-condition | Patient is discharged and requires medication. |
| Post-condition | Medicines provided to the patient and sent for payment. |

*Table: 3.4.8 Description of Dispensary and Drugs Management*

### Description of Patient Payment System:

*Table: 3.4.9 Description of* *Patient Payment System*

|  |  |
| --- | --- |
| Field | Details |
| Use Case Id | 09 |
| Use Case Name | Patient Payment System |
| Actors | Cashier |
| Description | Patients are directed for payment post-discharge. HMO authorized patients are discharged directly, while others are sent to the cashier for payment processing. Discounts and wallet system for future use are available. |
| Pre-condition | Patient completes treatment and needs to settle bills. |
| Post-condition | Payment processed, discount applied if applicable, and patient discharged. |

### Description of Drug and Inventory Management:

*Table: 3.4.10 Description of Drug and Inventory Management*

|  |  |
| --- | --- |
| Field | Details |
| Use Case Id | 11 |
| Use Case Name | Drug and Inventory Management |
| Actors | Admin, Authorized Users |
| Description | Admin or authorized users add and manage drugs, dressings, and other medical supplies in the pharmacy store. They create requisitions, handle inventory, and manage stock levels, including low stock alerts. They also handle purchasing from suppliers and update expiry dates. |
| Pre-condition | User has admin or relevant permissions. |
| Post-condition | Inventory updated, requisitions made, and stock levels managed. |

### Description of General Functionalities:

*Table: 3.4.11 Description of General Functionalities*

|  |  |
| --- | --- |
| Field | Details |
| Use Case Id | 10 |
| Use Case Name | General Functionalities |
| Actors | Users |
| Description | The app sends automated birthday wishes to users. It also facilitates real-time chat among users, database and chat backup, and options for forced patient discharge and comprehensive reporting. |
| Pre-condition | Users registered in the app. |
| Post-condition | Effective communication maintained, and app functionalities utilized for hospital management. |

# Chapter 4

## 4.1 Agile Software Development Methodology

Agile methodology is an iterative and incremental approach to software development that emphasizes flexibility, collaboration, and customer feedback. Unlike traditional methodologies like the Waterfall model, Agile allows for adaptive planning, evolutionary development, and continual improvement, enabling rapid and flexible responses to change. It is particularly effective in managing the complexity and unpredictability of software projects.

## 4.2 Selected Methodology: Agile

A software development methodology is a way to improve development work with the help of dividing the development process into distinct phases to make a system with better productivity. It also helps to structure and control the whole system. It involves different methodologies, also called the Software Development Life Cycle, that are stages for software development with a certain set of rules. Generically, we categorized the methodologies into Rapid application development and planned-driven. Waterfall, spiral is planned driven while agile is Rad based.

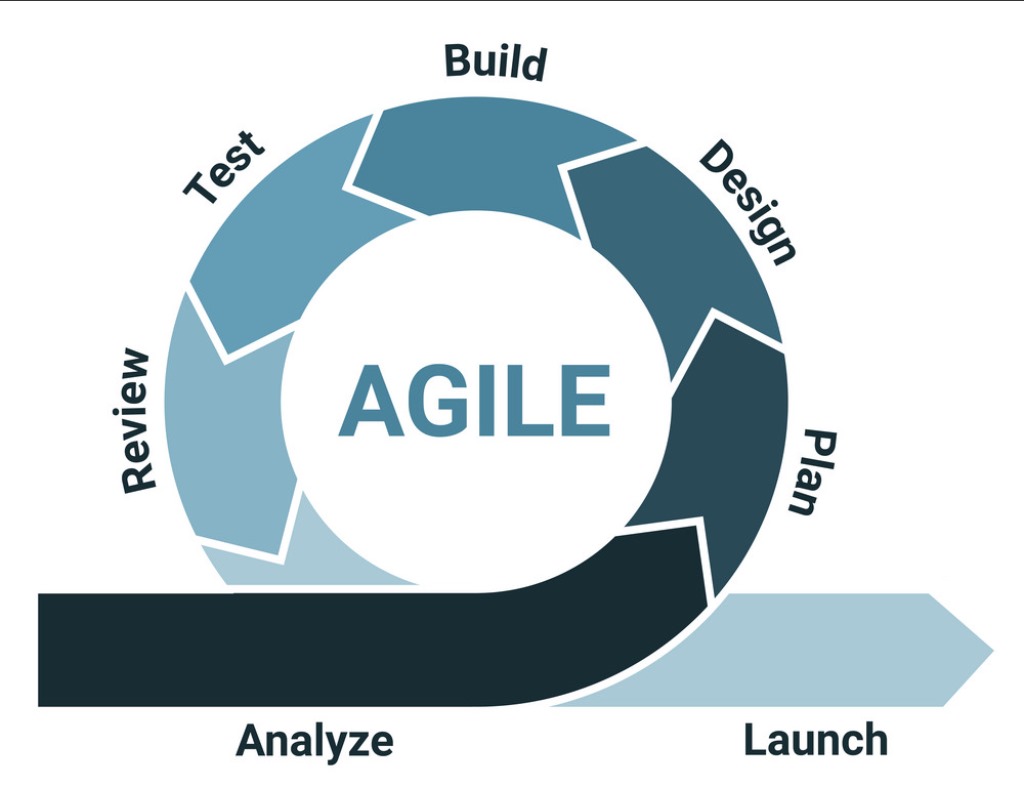
## 4.3 Reasons for Selecting Agile Methodology

1. **Flexibility and Adaptability**: Agile allows the project to adapt to changes in requirements and technology swiftly.
2. **Customer Collaboration**: Regular feedback from users ensures that the development aligns with the user's needs and expectations.
3. **Incremental Delivery**: Agile facilitates the delivery of small, workable segments of the project, ensuring a faster time-to-market and continuous improvement.
4. **Risk Management**: Regular reviews and iterations help in early identification and resolution of issues, reducing the overall risk.

## 4.4 Agile Project Planning and Execution for HME

Agile project planning and execution involve the division of the project into sprints, with each sprint aimed at delivering a potentially shippable product increment. The key phases include:

1. **Product Backlog Creation**: Gather and prioritize necessities for the HME undertaking, growing a product backlog.
2. **Sprint Planning**: At the start of every dash, pick a hard and fast of capabilities from the product backlog and plan their delivery.
3. **Daily Stand-ups**: Conduct day by day meetings to speak about development, demanding situations, and plan the day's paintings.
4. **Sprint Execution**: Develop, take a look at, and combine features inside the dash.
5. **Sprint Review**: At the quit of every sprint, reveal the finished paintings to stakeholders and collect comments.
6. **Sprint Retrospective**: Reflect at the sprint to identify successes and regions for development.
7. **Release Planning**: Plan releases based totally at the undertaking progress, stakeholder comments, and marketplace situations.



*Figure 4.2 Agile Model*

# Chapter 5

# 5 System Architecture

## 5.1 Architecture

In widespread, structure is the manner of product planning, layout, and construction. The design section of the structure comes in the solution phase of the lifestyles cycle as it defines the machine because the primary software program components.

For our task, we've got decided on a Model-View-Template (MVT) structure, appropriate for applications the use of Django and React. This architecture is defined as follows:

* **Model**: Represents the application's information structures, normally mapped to database tables. It is liable for managing facts, processing user inputs, and managing interactions with the database or other statistics assets.
* **View (Django)**: In Django, perspectives handle the enterprise common sense and interact with the version to carry statistics and render a template. They act as a bridge among the Model and the Template.
* **Template (React)**: Templates are liable for rendering the records acquired from the views in a layout suitable for interaction with the user. In our challenge, this position is fulfilled via React, where components are used to create the user interface. These components may consist of HTML, CSS, and JavaScript documents, creating a dynamic and interactive consumer in revel.

By MVT architecture in this way, the undertaking benefits from improved modularity, maintainability, and scalability. It enables less complicated collaboration between developers, as every element - Model, View, and Template - can be advanced independently. Additionally, keeping apart the user interface (React Templates) from the underlying business common sense and data (Django Model and View) makes the utility extra adaptable to special platforms and technology.

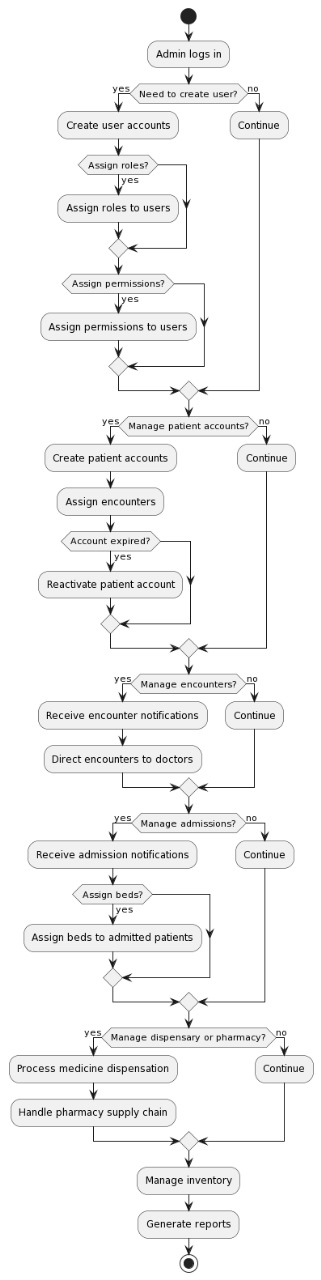
Diagram of a diagram of a computer

Description automatically generated

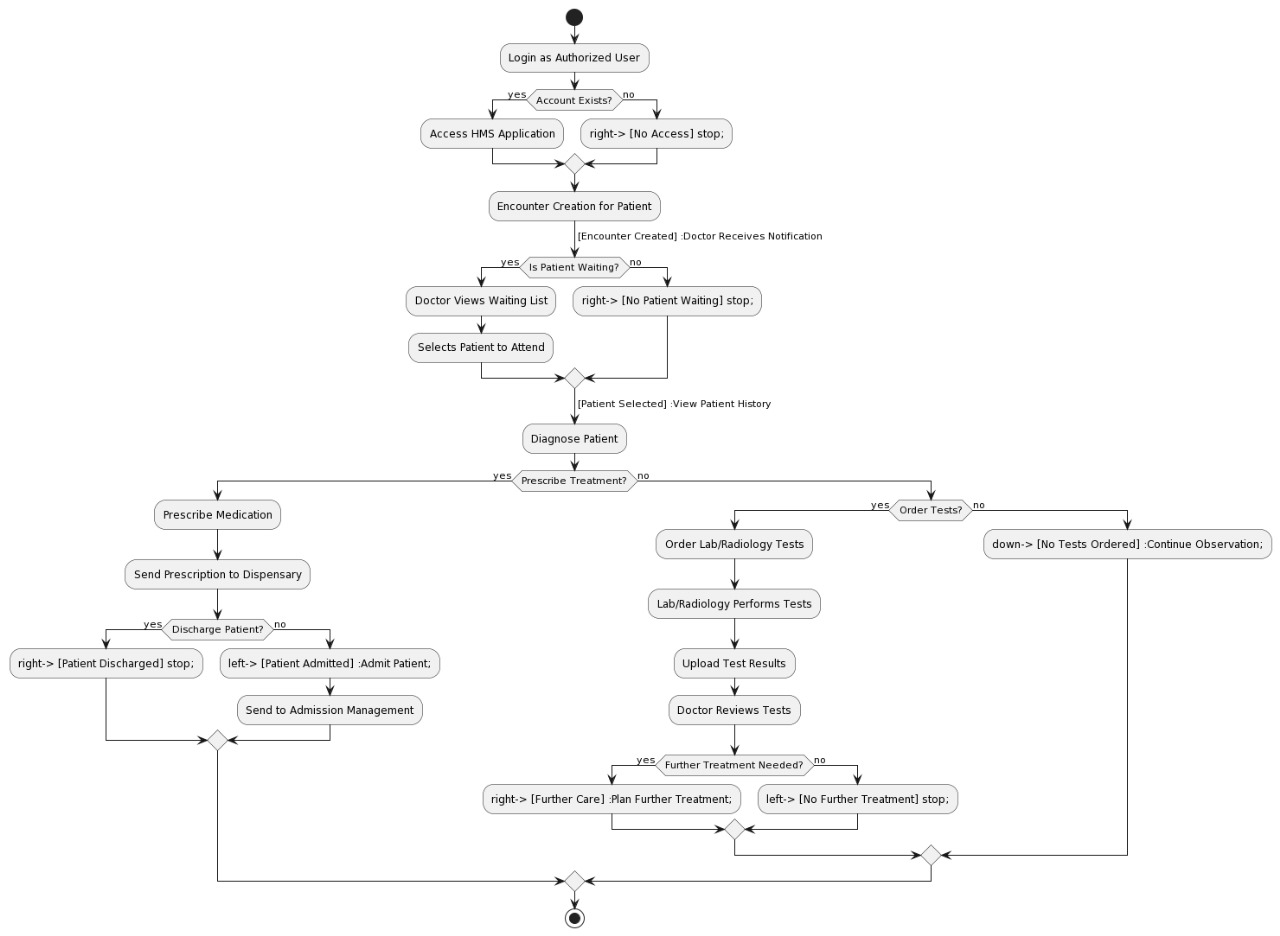
*Figure 5.1 Model, View and Template*

## 5.2 Activity Diagram

Activity Diagram shows the flow of the system from one activity to another. An activity is any set of action or set of actions that compare needs to perform any specific task, so mapping the overall actions of the system we can easily estimate the overall flow of the system, and that’s where activity diagrams are applicable.



*Figure 5.2.1 Activity of Admin Control*

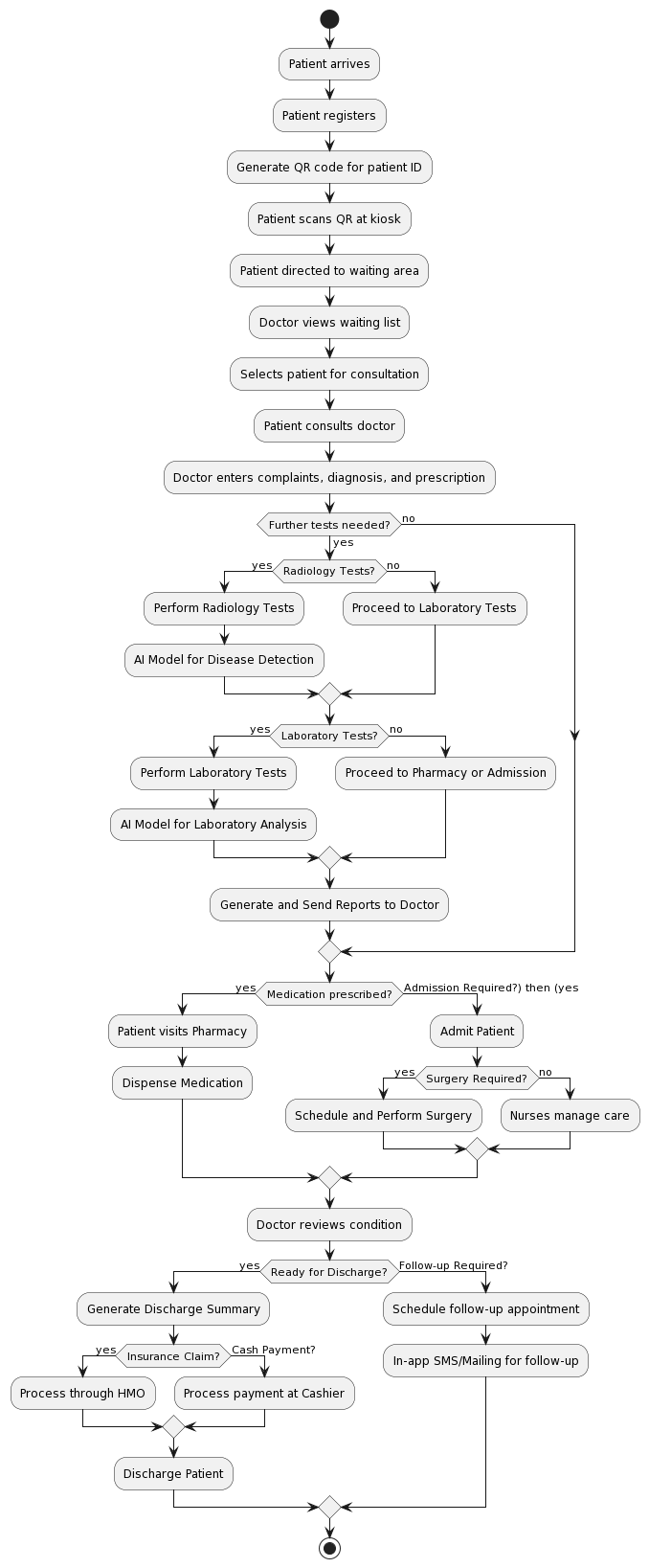


*Figure 5.2.2 Activity of Doctor, Nursing, Laboratory and Radiology Centre*

A diagram of a flowchart

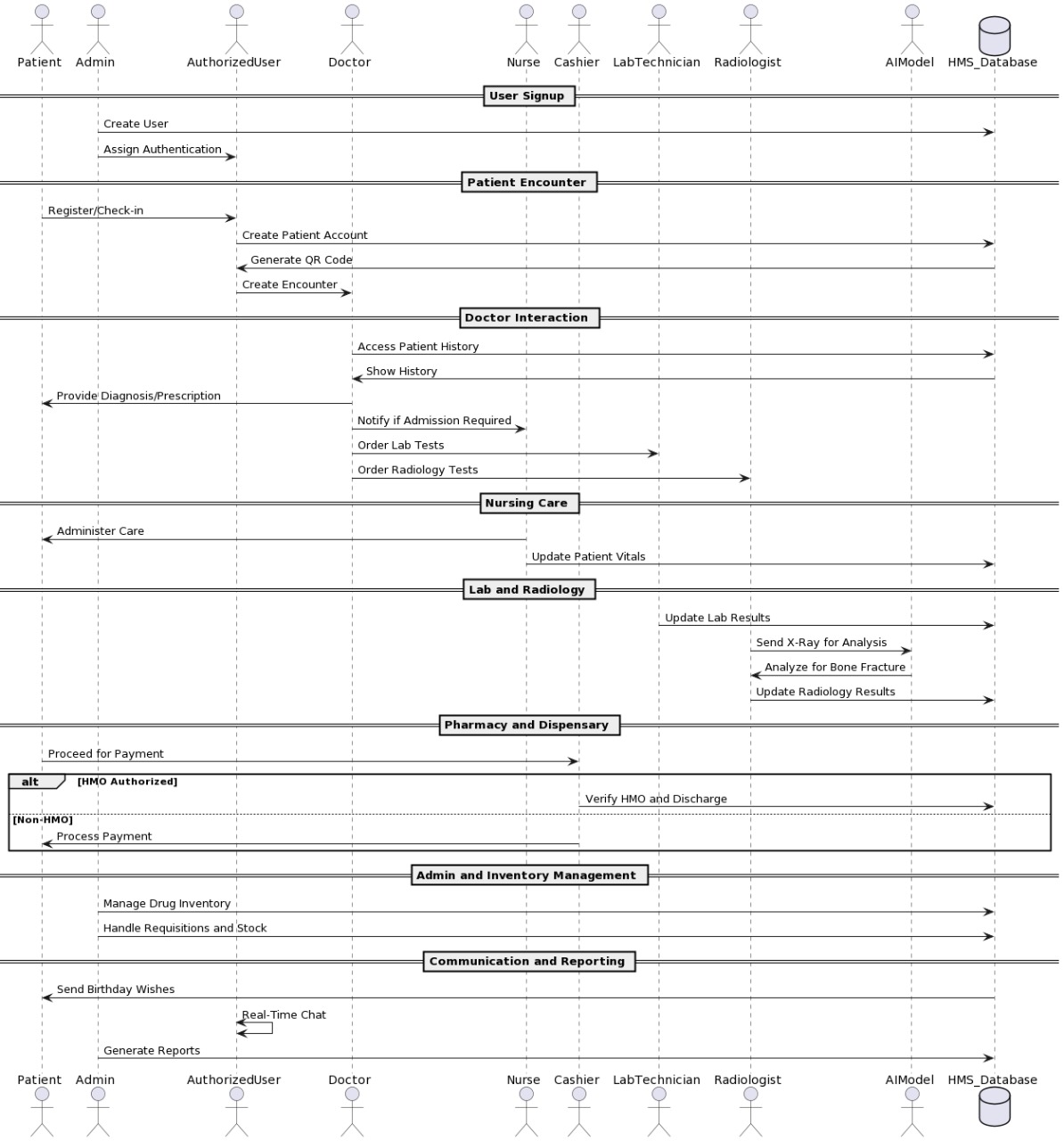
Description automatically generated

*Figure 5.2.3 Activity of Dispensary, Pharmacy, HMO Authorization, Cashier*

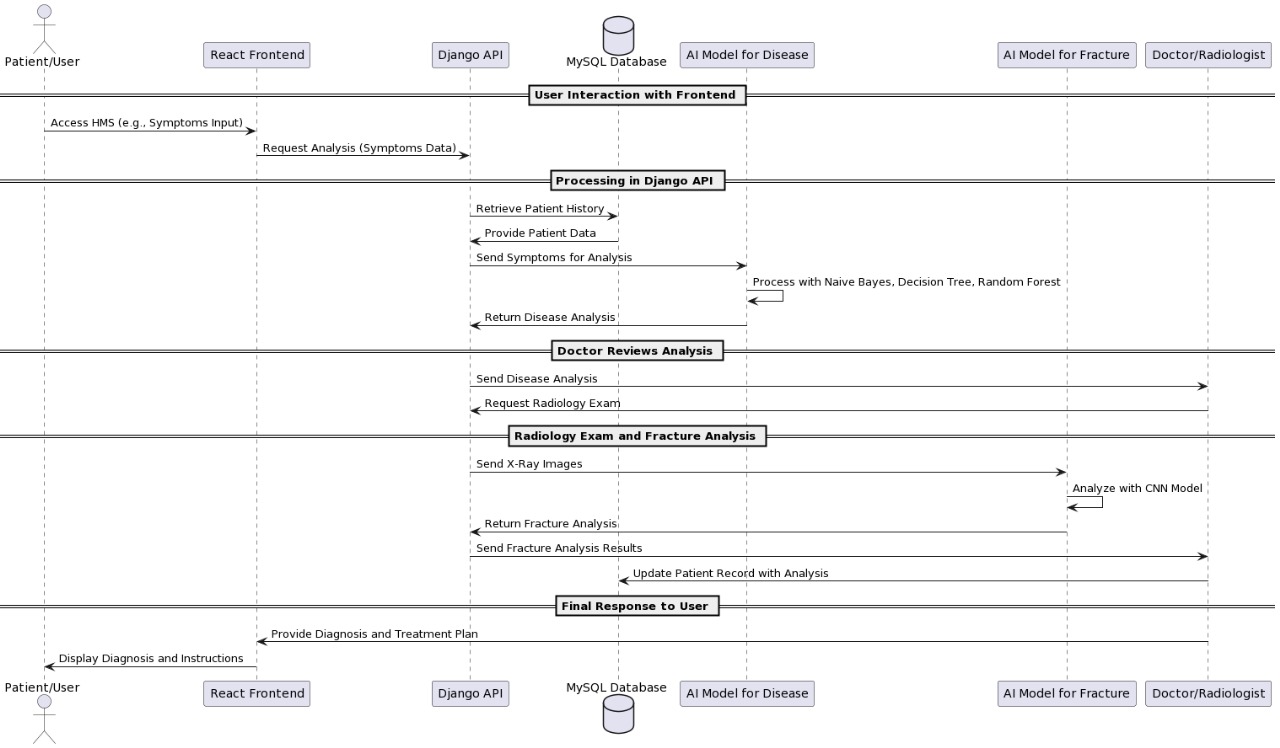


*Figure 5.2.4 Activity of Complete System*

## 5.3 Sequence Diagram

****

*Figure 5.3.1 Sequence of whole system*



*Figure 5.3.2 Sequence of Interaction among APIs and Model*

# Chapter 6

# AI-Model and Algorithms

## General Disease Detection:

We integrate the disease detection application that is primarily based at the system learning algorithm that is decision tree. We integrate this matching software for the assist of docs. Doctor will ask the signs and symptoms from patients and supply to that application a good way to are expecting the sickness from the given dataset.

A Decision Tree is a famous system gaining knowledge of set of rules used for classification and regression duties. Here is an in depth clarification of the way a selection tree works, including the important thing standards, steps, and attributes concerned in constructing and the use of a choice tree.

### Key Concepts of Decision Tree:

1. **Node**: Each point of decision in the tree.
   * **Root Node**: The topmost node, representing the entire dataset.
   * **Internal Node**: Nodes that represent decisions based on attributes.
   * **Leaf Node**: Terminal nodes that represent the output of the decision.
2. **Edge**: Connects nodes, representing the outcome of a decision.
3. **Attribute/Feature**: A characteristic of the data used to split the nodes.
4. **Gini Index / Entropy**: Measures used to determine the quality of a split. Lower values indicate a better split.
   * **Gini Index**: Measures the impurity of a dataset. Used in CART (Classification and Regression Trees).
   * **Entropy**: Used in ID3 and C4.5 algorithms to measure the disorder in the dataset.

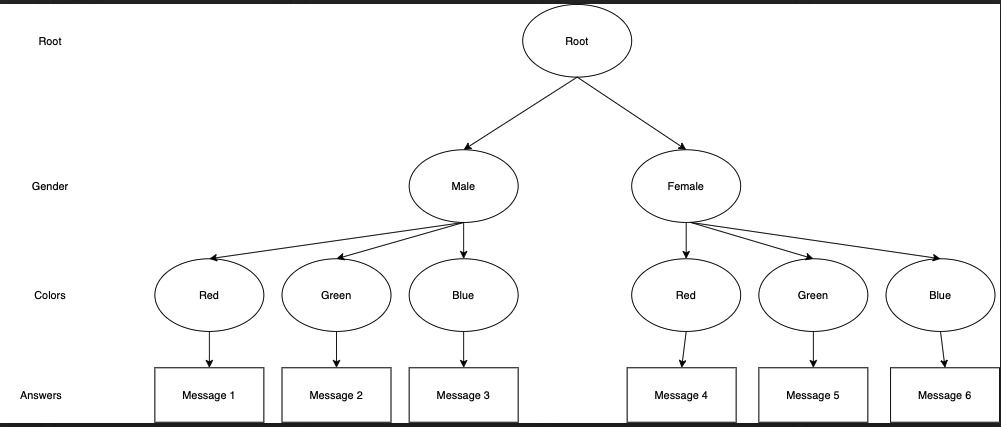
### Steps in Building a Decision Tree

1. **Selecting the Best Attribute**:
   * The process begins at the root node. The algorithm evaluates all possible features and selects the one that best splits the data into subsets with the most homogeneous target variable (class label).
   * This selection is based on the impurity measures such as Gini Index, Entropy, or Information Gain.
2. **Splitting the Dataset**:
   * Once the best attribute is selected, the dataset is split into subsets. Each subset corresponds to a unique value or range of values of the selected attribute.
3. **Creating Sub-nodes**:
   * Each subset created from the split becomes a new node in the tree. This node will be further split using the same process recursively.
4. **Stopping Criteria**:
   * The splitting process stops when one of the following criteria is met:
     + All the data in the node belongs to a single class.
     + There are no remaining attributes to split the data.
     + A predefined stopping condition such as maximum depth of the tree or minimum number of samples per node is reached.
5. **Pruning**:
   * Pruning is done to improve the tree's generalization ability by reducing its size. It removes branches that have little importance and can lead to overfitting.
   * **Pre-pruning**: Stops the tree growth early.
   * **Post-pruning**: Removes branches from a fully grown tree.

### Training Epochs

During an epoch, the learning algorithms updates the model’s parameters based on data. It took approximately 80 epochs to train.

### Architecture



*Figure 6.1.4.1 Architecture Diagram of Decision Tree*

### Confusion Matrix

*Figure 6.1.5.1 Confusion Matrix of G.D.D*

|  |  |  |
| --- | --- | --- |
|  | Predicted Positive | Predicted Negative |
| Actual Positive | 700 | 40 |
| Actual Negative | 60 | 635 |

Accuracy = (700 + 635) / (700 + 635 + 60 + 40)

Accuracy = 0.89

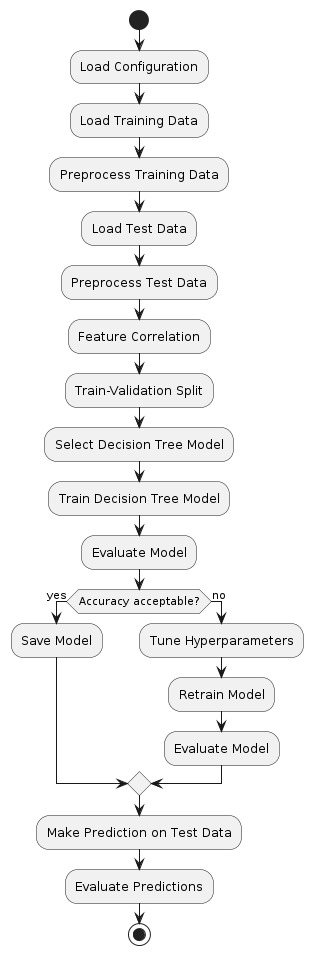
Accuracy = 0.89 x 100

Accuracy = 89 %

### Accuracy

Model accuracy is 89 %.

### Activity diagram of program



*Figure 6.1.1 Activity diagram of program*

## Fractured Bone Detection:

We have additionally integrate AI model for detection of fractured bones of wrists. This is done with the help of YOLO set of rules this is used for the detection of objects. Doctor will give the X-Ray image as enter to the model then model will predict, it's far fractured or now not as output.

### Dataset

The [GRAZPEDWRI-DX](https://www.nature.com/articles/s41597-022-01328-z) is an open dataset containing 20327 annotated pediatric trauma wrist radiograph images of 6091 patients, treated at the Department for Pediatric Surgery of the University Hospital Graz between 2008 and 2018. Several pediatric radiologists annotated the images by placing bounding boxes to mark 9 different classes:

* boneanomaly (276 boxes),
* bonelesion (45 boxes),
* foreignbody (8 boxes),
* fracture (18090 boxes),
* metal (818 boxes),
* periostealreaction (3453 boxes),
* pronatorsign (567 boxes),
* softtissue (464 boxes),
* text (23722 boxes).

### YOLO Working and Explanation

YOLO (You Only Look Once) is an object detection algorithm that has gained significant popularity in the field of computer vision. It is both an algorithm and a model architecture designed for real-time object detection tasks. YOLO stands out for its speed and accuracy in detecting objects in images and videos.

1. **Algorithm/Model**: YOLO is both an algorithm and a model architecture. The YOLO algorithm applies a deep convolutional neural network (CNN) model to perform object detection tasks.
2. **Real-Time Object Detection**: YOLO is designed for actual-time item detection, that means it could system pictures or video frames swiftly and hit upon gadgets within them with excessive accuracy.
3. **Single Pass**: Unlike conventional object detection techniques that apply a sliding window approach or vicinity notion networks, YOLO takes a single bypass thru the enter photograph or frame and without delay predicts bounding packing containers and class probabilities for all items detected.

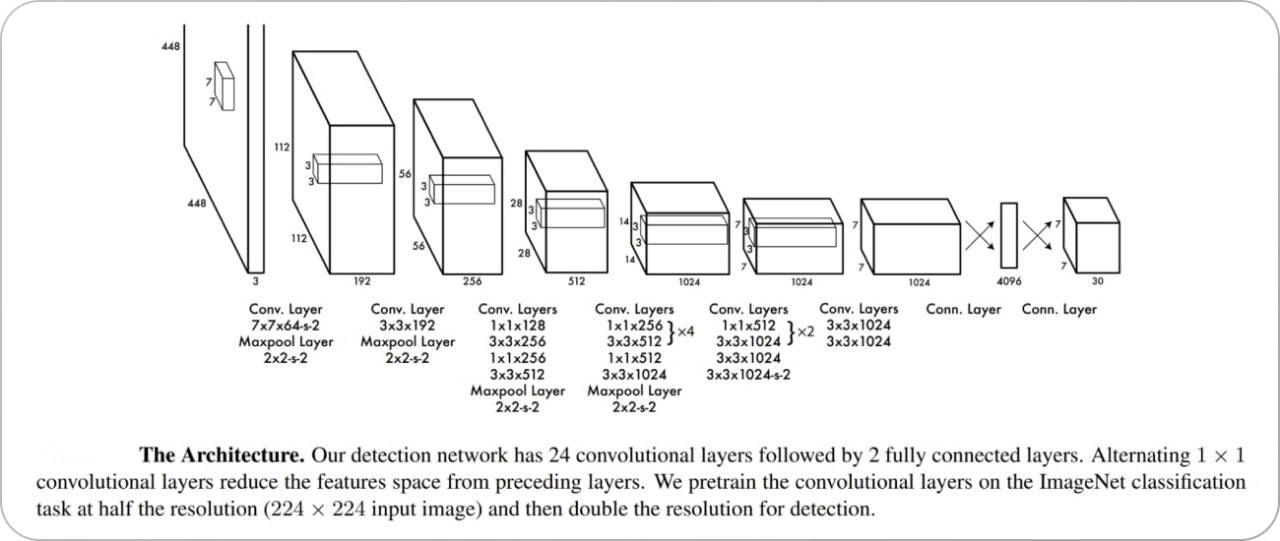
**YOLO Working**

1. **Grid-based Detection**:
   * YOLO divides the input image into a grid of cells. Each cellular is chargeable for predicting bounding boxes and sophistication possibilities for objects that fall inside it.
2. **Bounding Box Prediction**:
   * Each grid cell predicts multiple bounding containers (with predefined sizes and styles) along with self belief scores that imply the chance of each box containing an object.
3. **Class Prediction**:
   * Each bounding field predicts elegance chances for the items it incorporates. YOLO makes use of softmax activation to are expecting the opportunity distribution across more than one training.
4. **Non-Maximum Suppression (NMS)**:
   * YOLO applies non-maximum suppression to remove redundant or overlapping bounding boxes with decrease self belief rankings, retaining only the maximum assured detections.

### Training Epochs

During an epoch, the learning algorithms updates the model’s parameters based on data. It took approximately 120 epochs to train.

### Architecture Diagram:



*Figure 6.2.1 Architecture Diagram*

### Confusion Matrix:

*Figure 6.2.5.1 Confusion Matrix of Yolo Model*

|  |  |  |
| --- | --- | --- |
|  | Predicted Positive | Predicted Negative |
| Actual Positive | 2800 | 200 |
| Actual Negative | 300 | 2060 |

Accuracy = (2800 + 2060) / (2800 + 2060 + 300 + 200)

Accuracy = 0.81

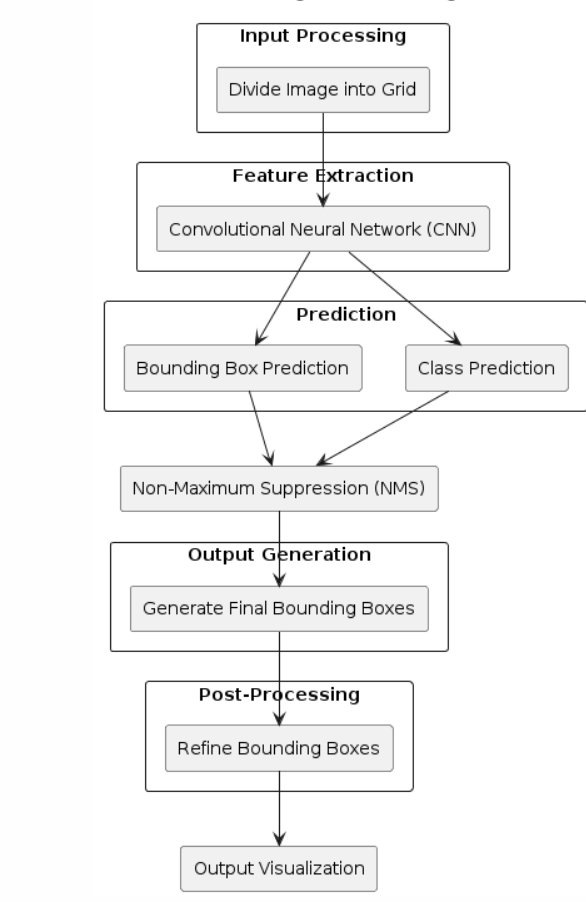
Accuracy = 0.81 x 100

Accuracy = 81 %

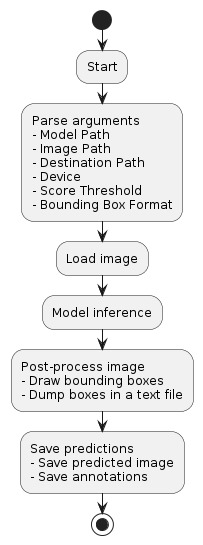
### Accuracy

Model Accuracy is 81%.

### Activity Diagrams



*Figure 6.2.2 Activity diagram of working of YOLO*



*Figure 6.2.3 Activity diagram of program*

# Chapter 7: System Implementation

After successfully completing the initial phases of development, including requirement analysis, design, and planning, we now proceed to the final stage: the actual implementation of the hospital management system. This chapter provides a comprehensive overview of the tools, technologies, and processes involved in the development of the system. We will detail each module and component, as well as the integration of an AI model for disease detection.

## 7.1 System Tools and Technology

The tools and technologies used in this project are as follows:

* **Django**: A high-level Python web framework that encourages rapid development and clean, pragmatic design.
* **React**: A JavaScript library for building user interfaces, particularly single-page applications.
* **MySQL**: An open-source relational database management system.
* **VS Code**: A source-code editor made by Microsoft with support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git.
* **AI Model (Decision Tree)**: Used for disease detection based on user input.

### 7.1.1 Django

Django is a excessive-stage Python web framework that lets in fast improvement and clean, pragmatic layout. It takes care of much of the problem of net improvement, so developers can cognizance on writing their programs while not having to reinvent the wheel.

* **Routing**: Django's URL dispatcher allows clean, readable URLs.
* **Templating**: The Django template engine facilitates the separation of presentation and business logic.
* **ORM**: Django's Object-Relational Mapping (ORM) system makes it easy to interact with the database.
* **Security**: Django provides built-in protection against many security threats, including SQL injection, cross-site scripting, and cross-site request forgery.

### 7.1.2 React

React is a JavaScript library for building user interfaces. It enables developers to build complex UIs from small, isolated pieces of code called "components."

* **Component-Based**: React promotes the development of reusable components.
* **Virtual DOM**: React uses a virtual DOM to optimize rendering and improve performance.
* **JSX**: A syntax extension for JavaScript that looks similar to HTML, making it easier to write and understand components.

### 7.1.3 MySQL

MySQL is a widely used relational database management system. It is known for its reliability, high performance, and ease of use.

* **Data Storage**: MySQL stores data in tables, which are efficient for storing structured data.
* **Query Language**: MySQL uses Structured Query Language (SQL) for accessing and managing data.
* **Transactions**: MySQL supports transactions, ensuring data integrity and consistency.

### 7.1.4 VS Code

Visual Studio Code, commonly known as VS Code, is a source-code editor developed by Microsoft. It includes support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git.

* **Extensions**: VS Code has a rich ecosystem of extensions that enhance its functionality.
* **Terminal**: The terminal allows running commands and scripts directly within the editor.
* **IntelliSense**: Provides smart completions based on variable types, function definitions, and imported modules.

## 7.2 Module Implementation

### 7.2.1 Dashboard

The Dashboard module provides an overview of the hospital's operations. It includes various sub-modules for different departments and functionalities.

***7.2.1.1 Front-Desk***

* Allows searching for patient records.
* Manages patient appointments.
* Displays the history of patient encounters.
* Allows uploading patient-related documents.
* Tracks the number of patient encounters.
* Generates reports on patient registrations.
* Provides reports on ward activities.
* Generates reports on patient encounters.

***7.2.1.2 Nursing***

* Records patient vital signs.
* Manages injection administration.
* Handles dressing and medical procedures.
* Manages immunization records.
* Manages antenatal clinic activities.
* Handles requisitions for medical supplies.
* Manages nursing inventory.
* Manages patient admissions.
* Generates reports on antenatal care.
* Provides reports on pregnancy cases.

***7.2.1.3 Doctors / Department***

* Manages outpatient department clinics.
* Handles patient admissions.
* Generates reports on patient encounters.
* Provides reports on patient diagnoses.

***7.2.1.4 Laboratory***

* Manages pending lab requests.
* Records lab results.
* Manages parameters for test reporting.
* Handles administration of lab tests.
* Manages categories of lab tests.
* Handles requisition of consumables.
* Manages lab inventory.
* Tracks the count of lab tests.
* Records lab test results.

***7.2.1.5 Radiology***

* Manages pending X-ray requests.
* Manages pending CT scan requests.
* Manages pending ultrasound requests.
* Manages pending ECG requests.
* Records radiology test results.
* Manages parameters for radiology test reporting.
* Handles administration of radiology tests.
* Tracks the count of radiology tests.

***7.2.1.6* *Dispensary***

* Manages outpatient dispensary.
* Manages inpatient dispensary.
* Handles purchase orders.
* Manages requisition for drugs and supplies.
* Manages dispensary inventory.
* Generates reports on dispensary activities.
* Manages the pharmacy store.
* Handles management of drugs.
* Manages consumables inventory.
* Manages store operations.
* Manages vendor information.
* Handles drug purchases.
* Confirms receipt of purchased drugs.
* Generates reports on drug expiry.
* Manages overall inventory.
* Manages consumables inventory.
* Confirms purchases.

***7.2.1.7* *HMO Authorization***

* Manages pending authorizations.
* Generates approval reports.
* Provides summaries of bills.
* Generates payment reports.
* Manages patient bills.
* Manages company-related information.
* Reviews patient bills.
* Generates patient bill reports.
* Manages admissions and discharges.
* Manages hospital services.

***7.2.1.8* *Cashier***

* Manages cashier activities.
* Handles deposits.
* Manages expenses.
* Generates daily cashier reports.
* Provides revenue reports.
* Generates payment reports.

***7.2.1.9* *Reports***

* Generates lab result reports.
* Provides appointment reports.
* Generates diagnosis reports.
* Manages drug inventory reports.
* Manages consumable inventory reports.
* Generates reports on admissions and discharges.
* Tracks patient SMS history.
* Provides payment reports.
* Generates encounter reports.
* Provides dispensary reports.
* Generates revenue reports.
* Manages ward reports.
* Provides antenatal care reports.
* Generates pregnancy reports.
* Tracks birth reports.
* Manages immunization reports.
* Tracks patient encounter count.
* Tracks laboratory test count.
* Tracks radiology test count.
* Provides registration reports.
* Manages expense reports.
* Confirms purchase reports.
* Tracks user login logs.

***7.2.1.10* *User Management***

* Manages vaccine information.
* Handles ward management.
* Manages bed availability.
* Manages user accounts.
* Controls user access permissions.
* Configures patient information.
* Manages patient discharge.
* Handles admissions and discharges.
* Manages admissions.
* Handles database backups.

***7.2.1.11* *SMS***

* Composes SMS messages.
* Tracks SMS history.
* Manages SMS settings.
* Generates SMS reports.

***7.2.1.12* *Administration***

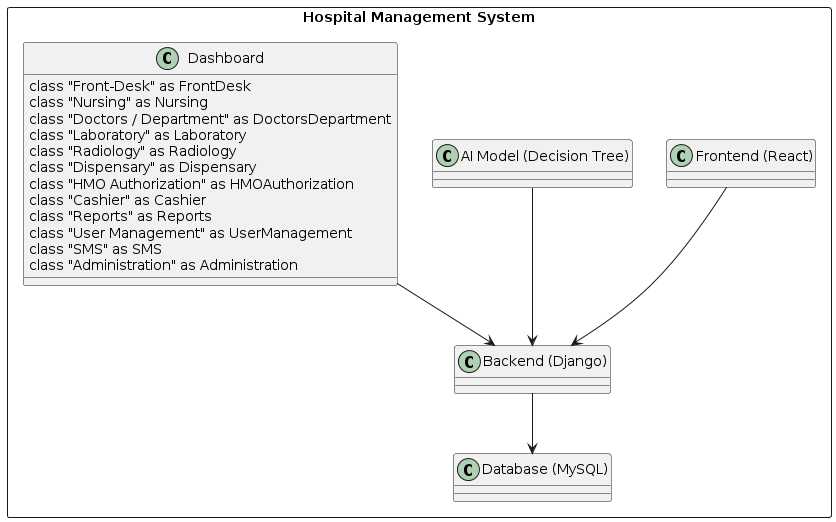
* Manages administrative activities.
* Configures encounter settings.
* Handles database backups.
* Manages antenatal care settings.

## 7.3 System Integration and Deployment

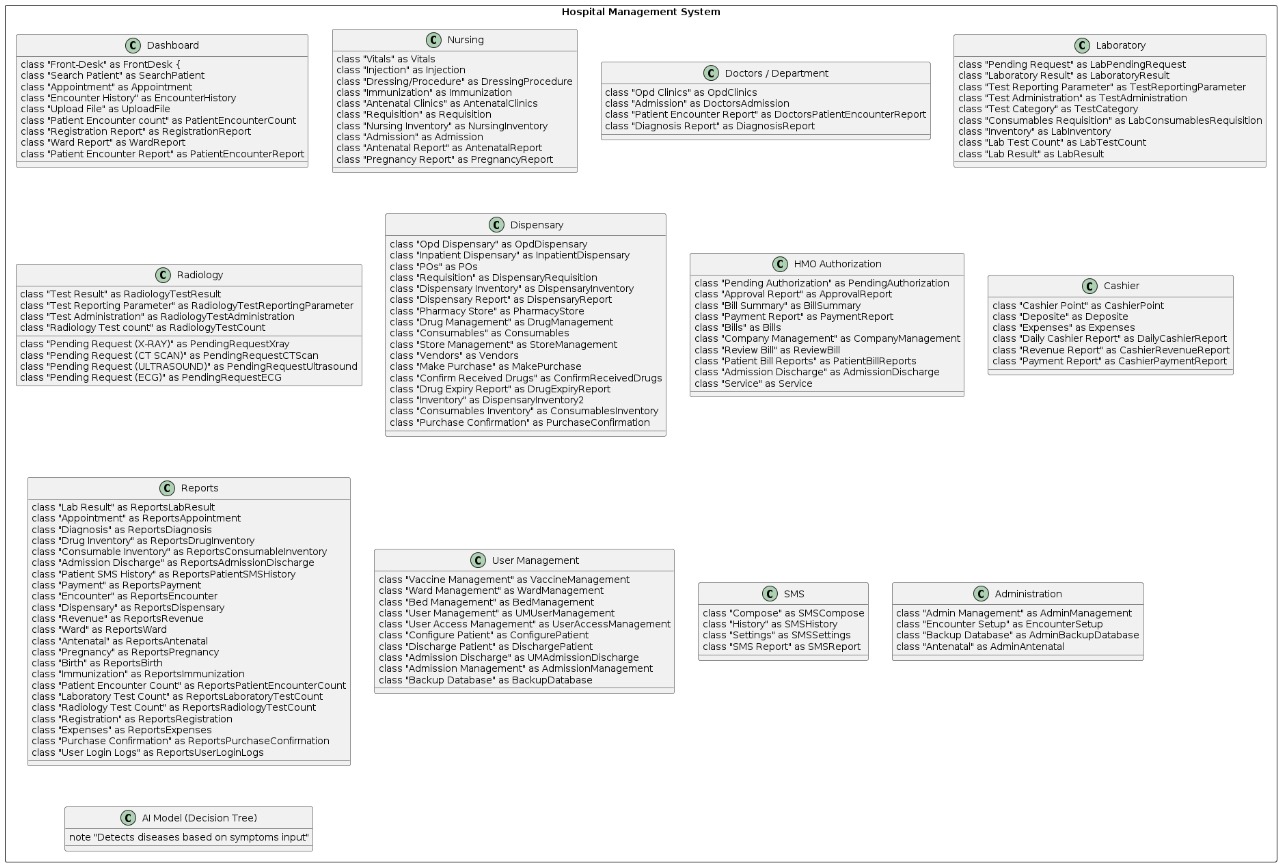
The health center management system is built the usage of Django for the backend, React for the frontend, and MySQL for the database. The integration of those technology ensures a seamless and efficient workflow.

### 7.3.1 Frontend-Backend Communication

React additives speak with the Django backend thru RESTful APIs. These APIs cope with diverse requests inclusive of fetching patient data, updating records, and processing bills.



*Figure 7.3.1.1 Class Diagram of Hospital Management System*



*Figure 7.3.1.2 Component Diagram of System*

### 7.3.2 Database Management

MySQL is used to save all the records related to the health facility's operations. Django's ORM makes it smooth to have interaction with the database, ensuring information integrity and consistency.

### 7.3.3 Deployment

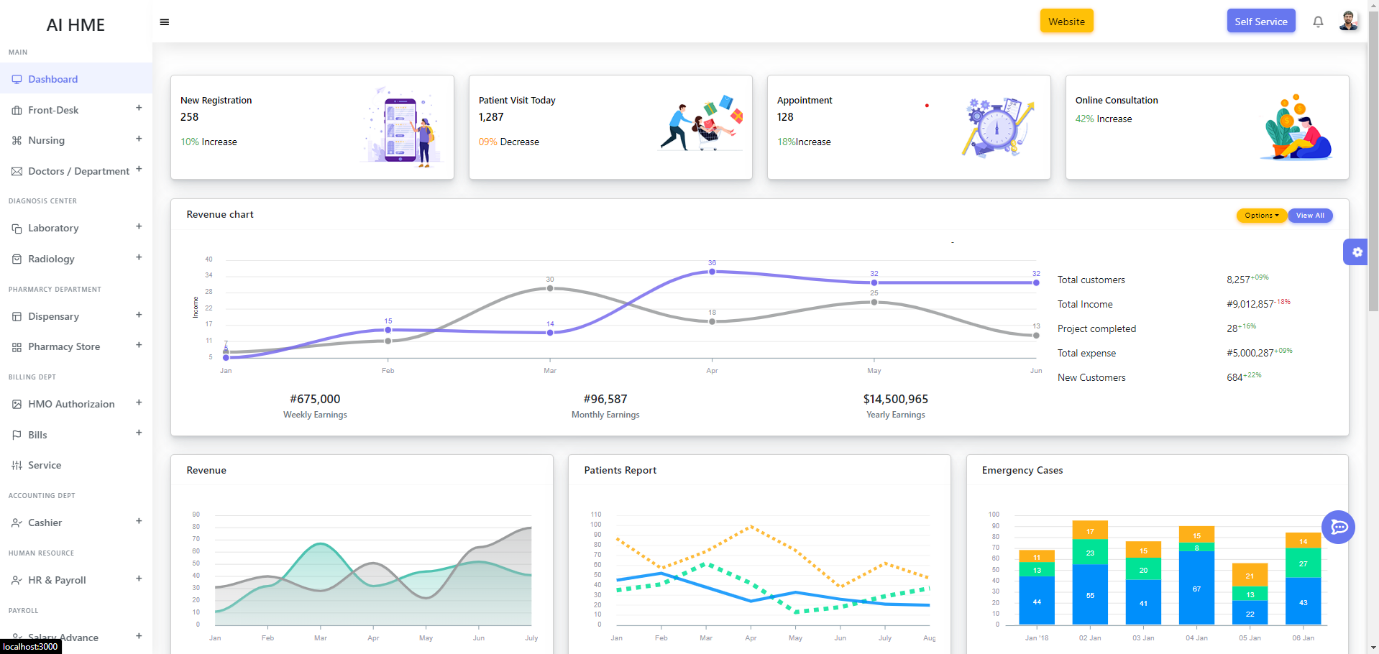
The system is deployed on a cloud platform, making sure excessive availability and scalability. Regular backups are taken to prevent statistics loss, and security features are carried out to shield touchy data.

**Conclusion**

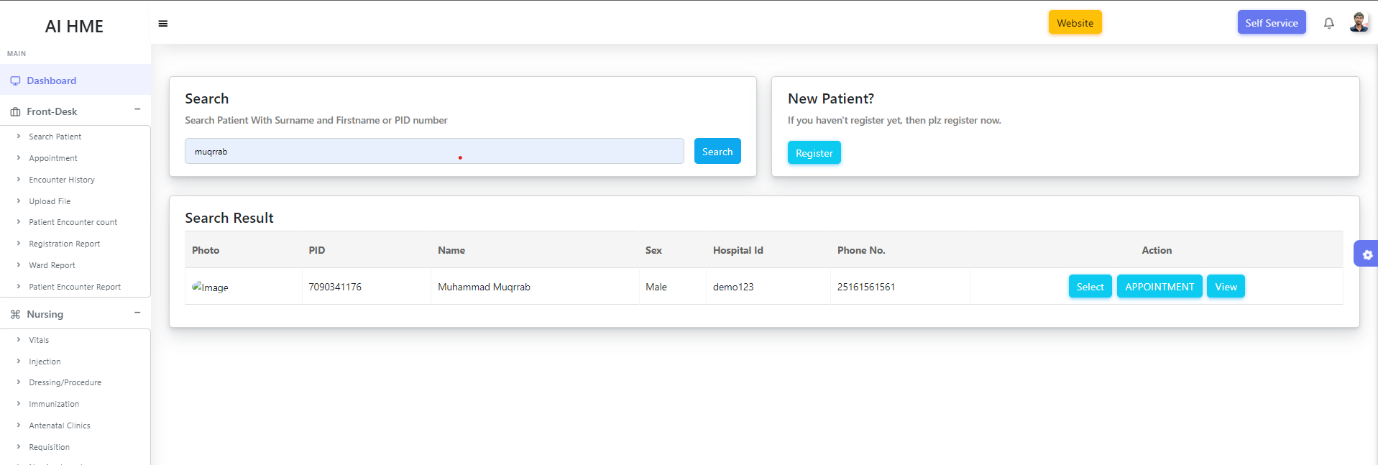
The implementation of the medical institution management machine entails the integration of numerous tools and technologies to create a complete answer for managing hospital operations. Each module and component is carefully designed and advanced to ensure a seamless and efficient workflow. The integration of an AI model for sickness detection adds a precious function to the device, supplying accurate and timely diagnoses primarily based on affected person input. This chapter provides an in depth assessment of the implementation method, highlighting the important thing components and capabilities of the machine.

# Chapter 8

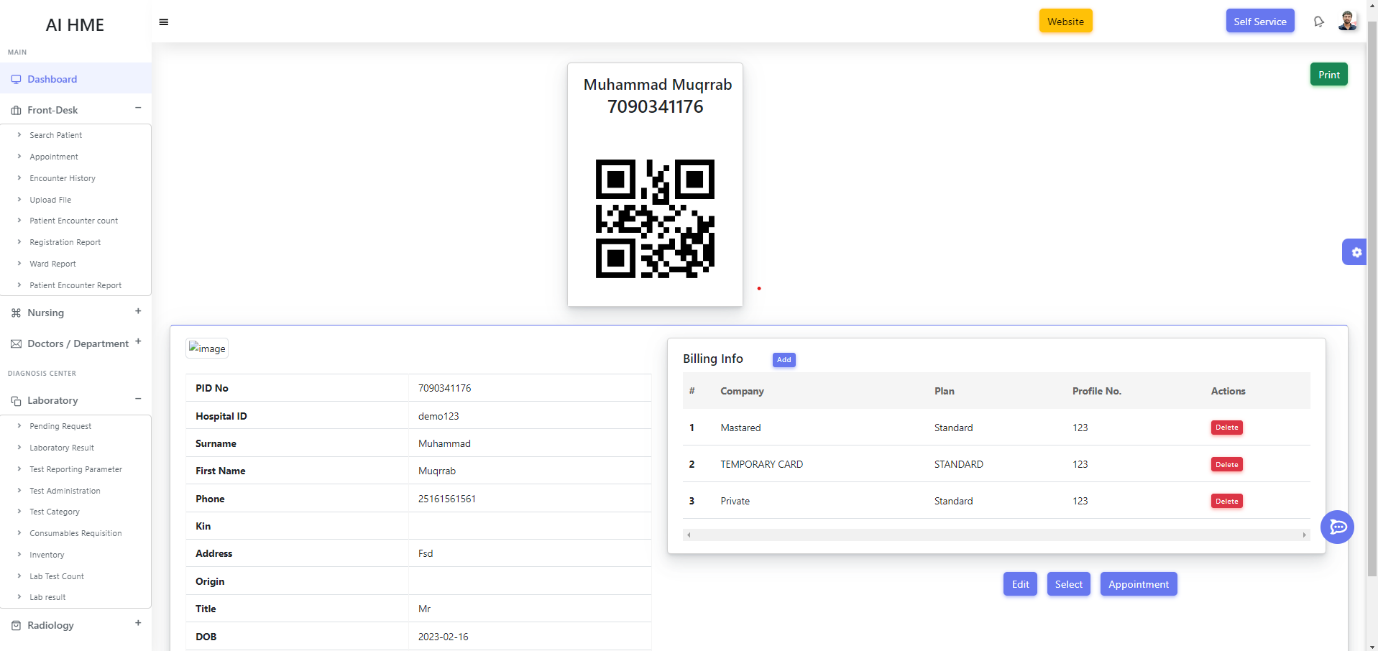
# 8 Screenshots



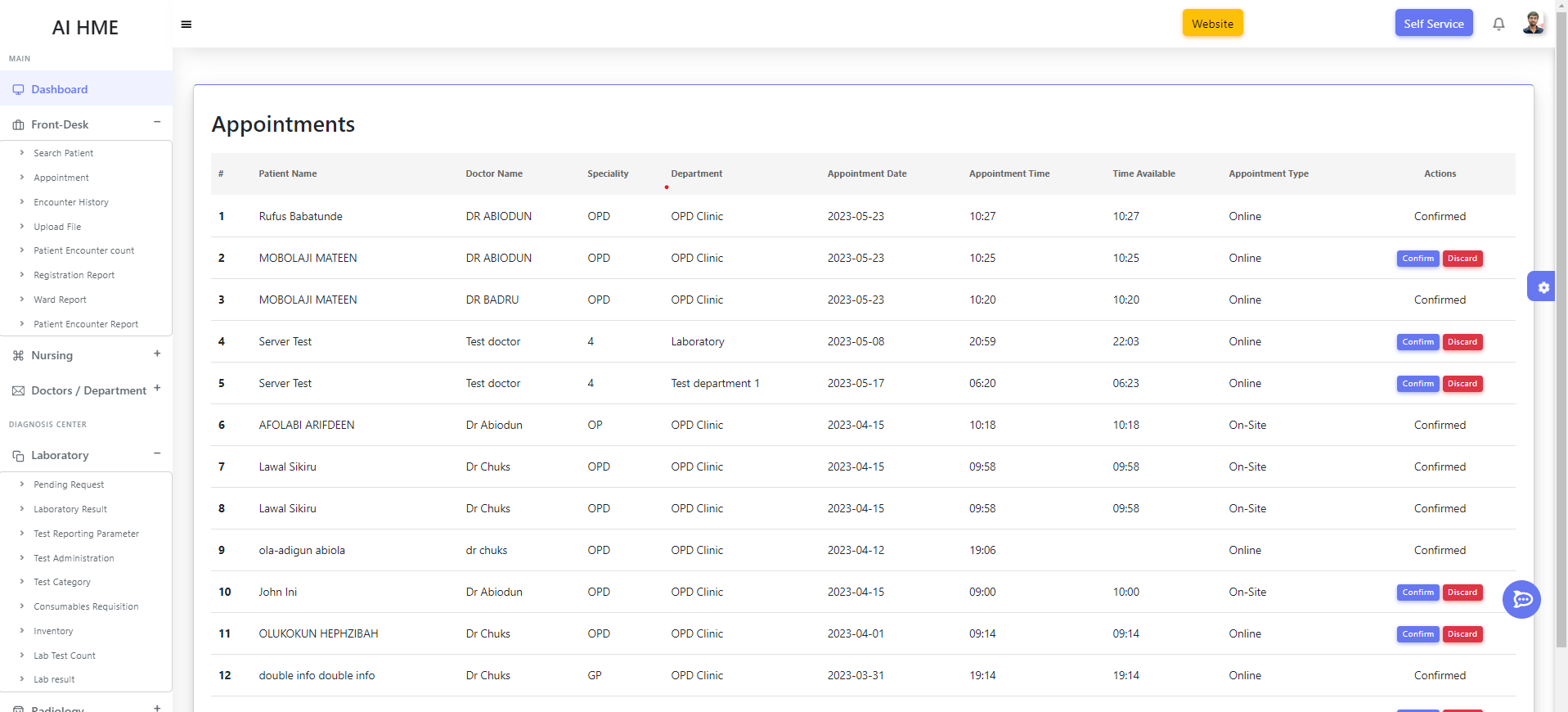
*Figure 8.1 Main Dashboard*



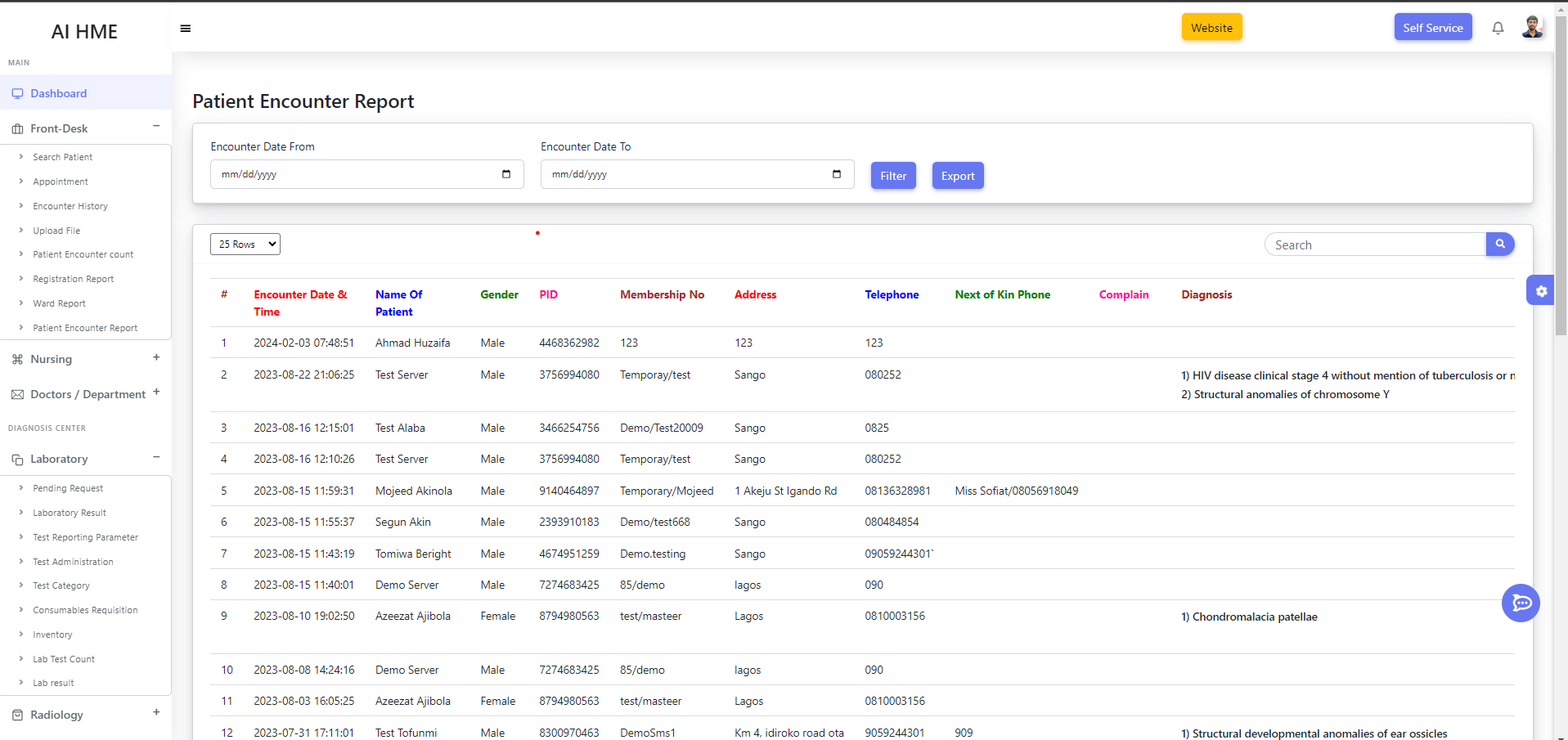
*Figure 8.2 Patient Dashboard*



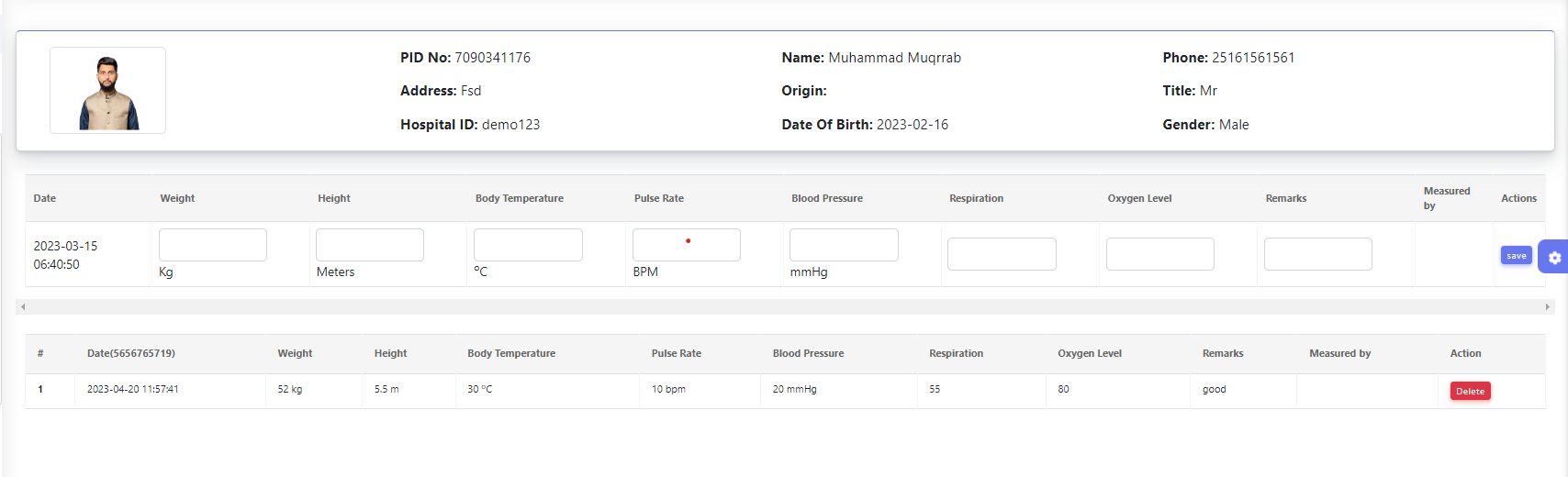
*Figure 8.3 Patient Detail*



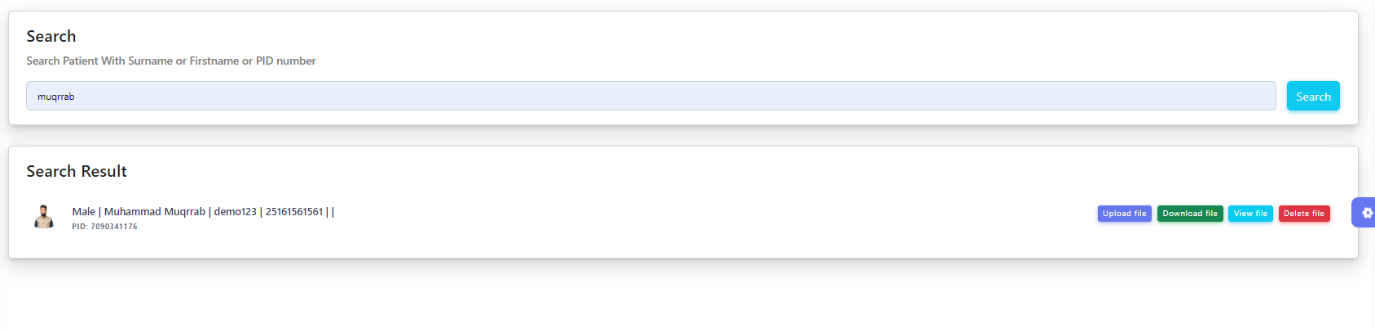
*Figure 8.4 Patient Appointment*



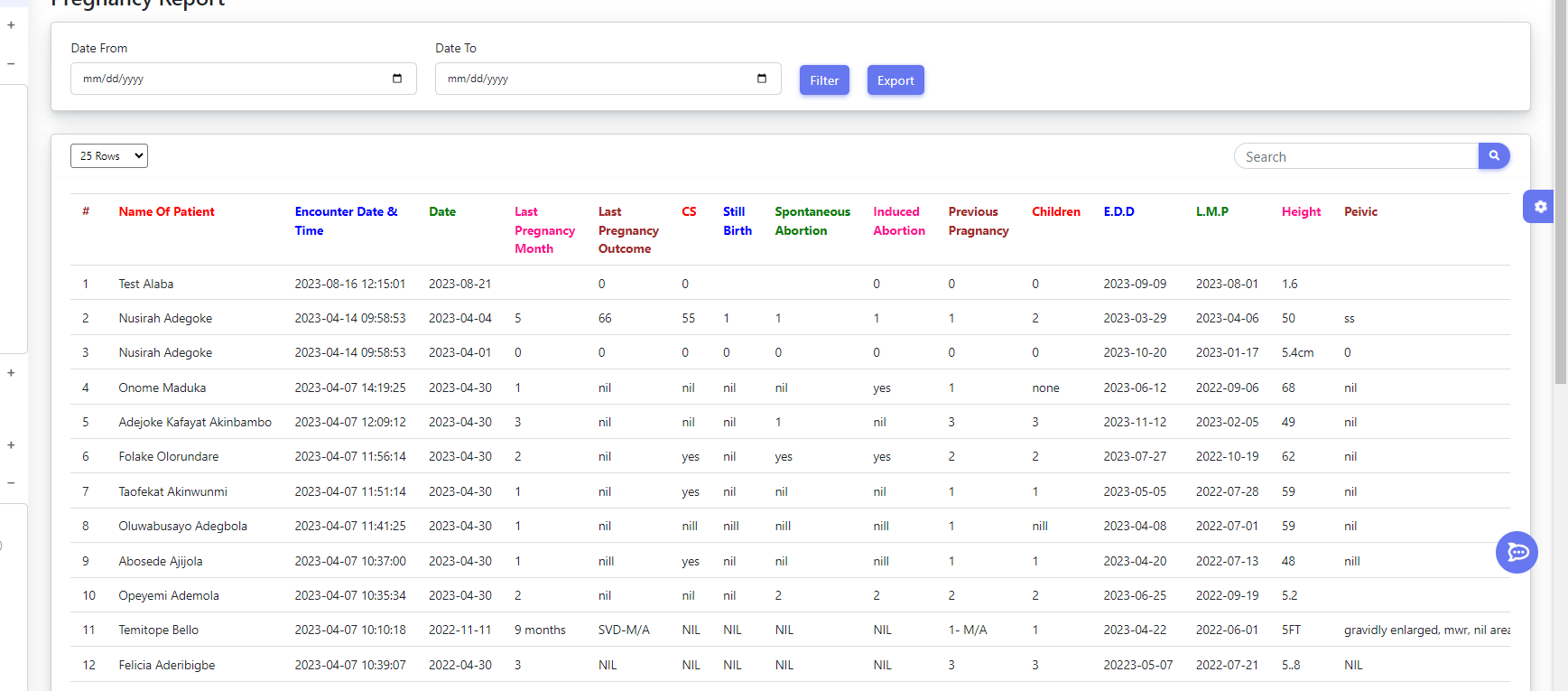
*Figure 8.5 Patient Report*



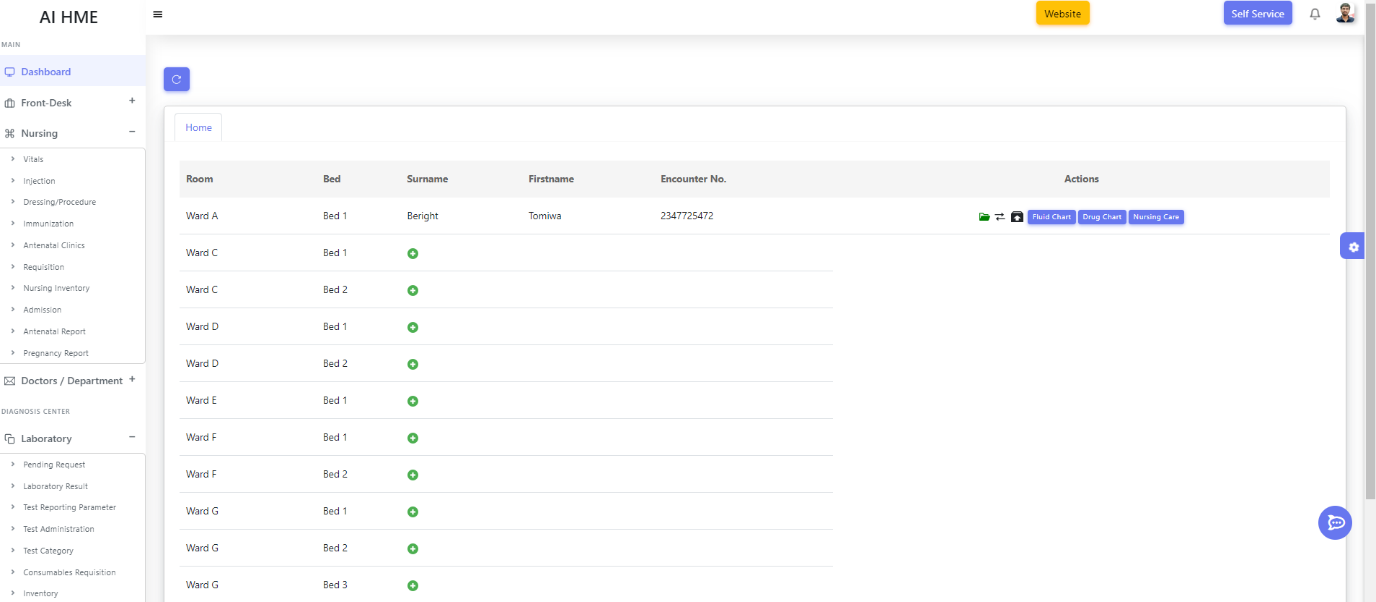
*Figure 8.6 Nurse Vital*



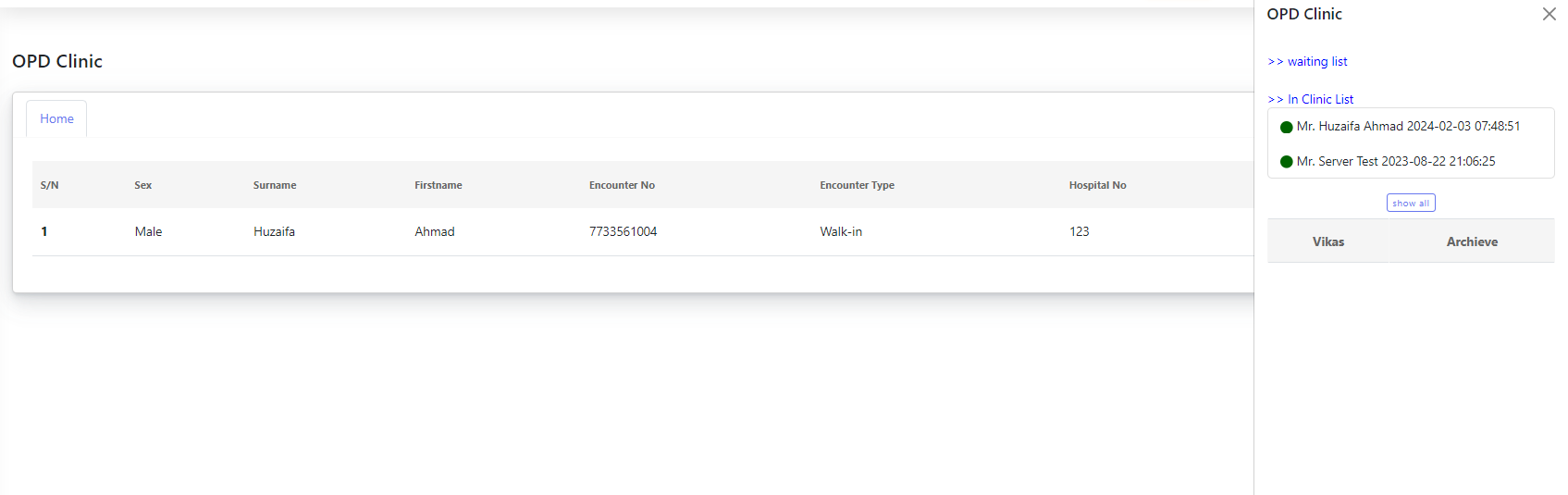
*Figure 8.7 Patient Data Update*



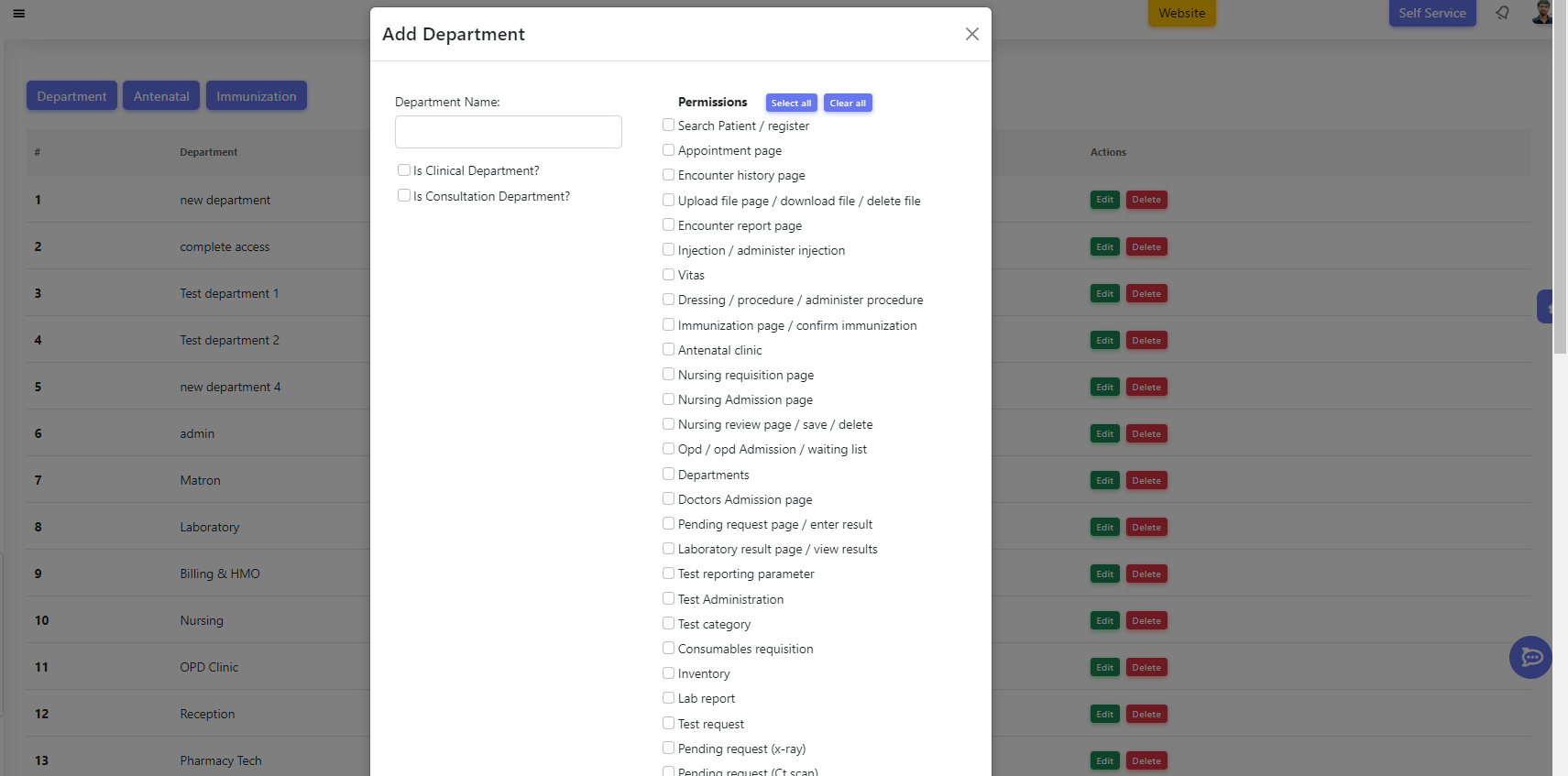
*Figure 8.8 Pregnancy Reports*



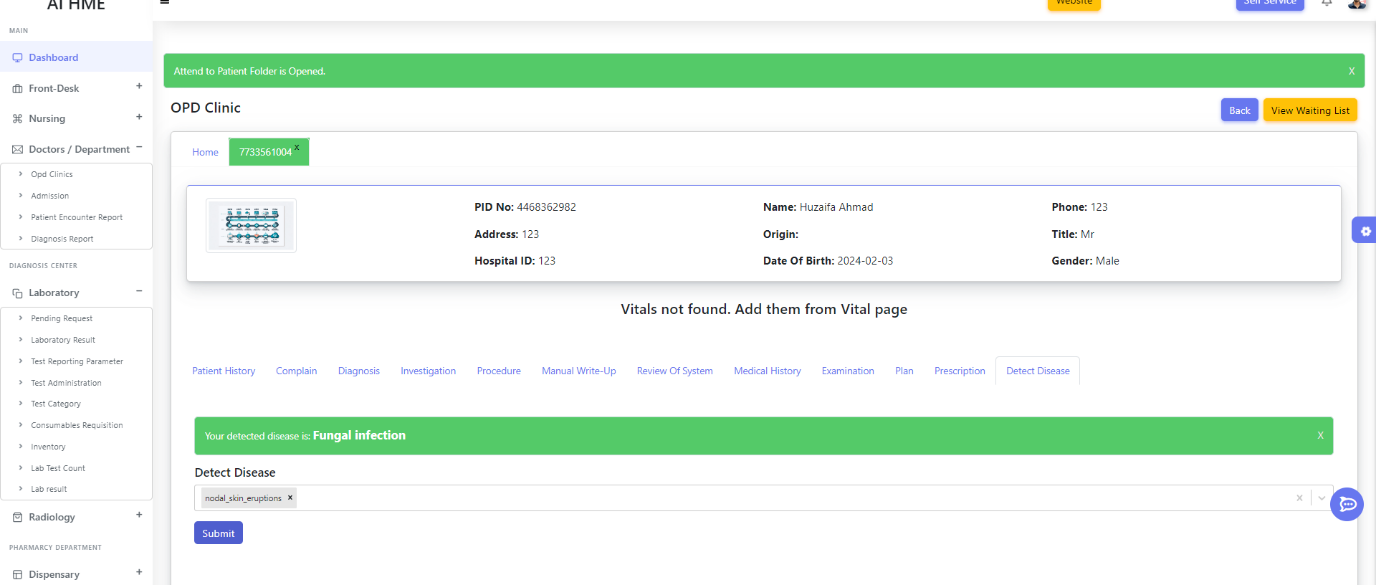
*Figure 8.9 Rooms and Bed Allocation*



*Figure 8.10 OPD Clinic*



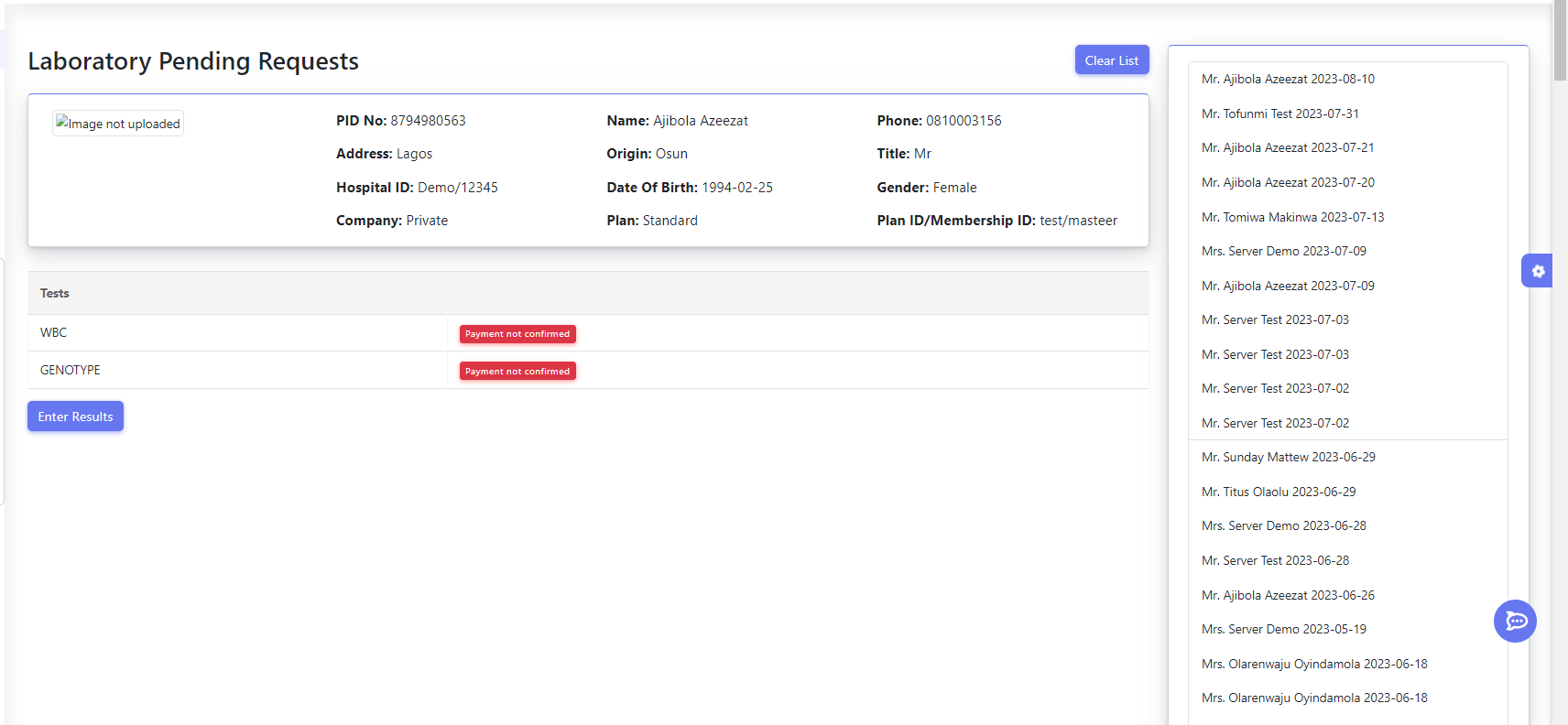
*Figure 8.11 Authorization*



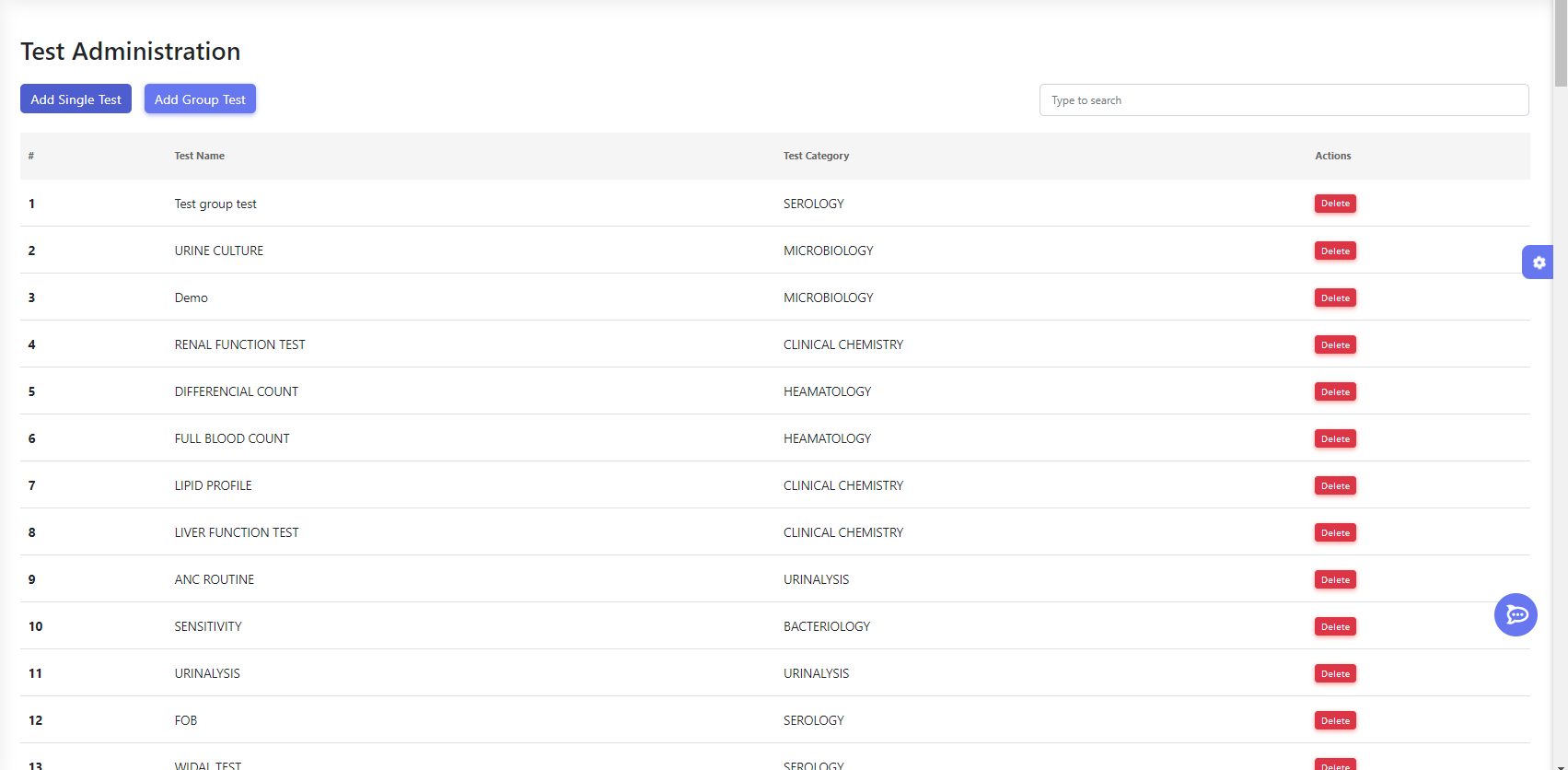
*Figure 8.12 General Disease AI Model*



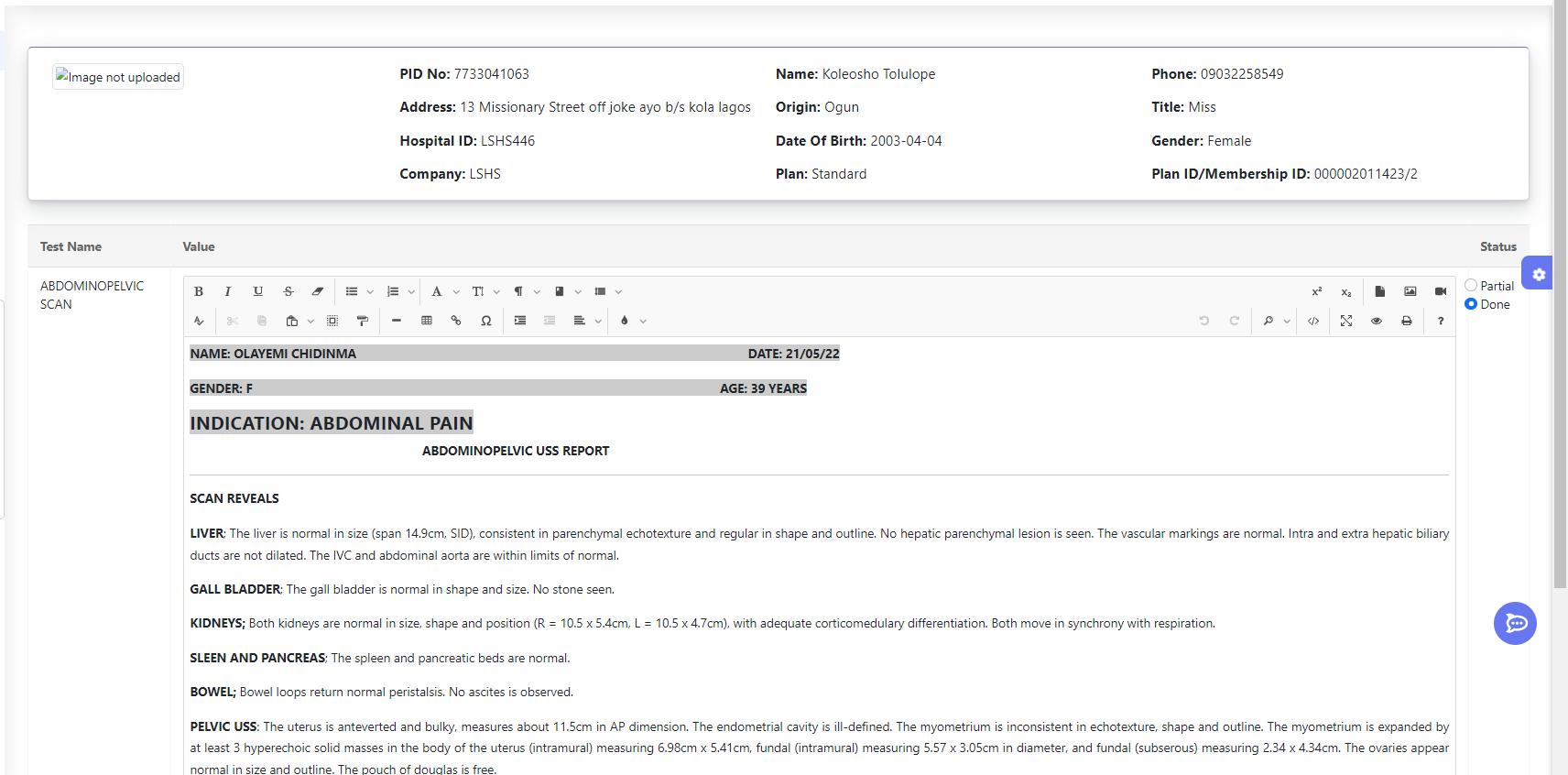
*Figure 8.13 Patient Vitals Details*



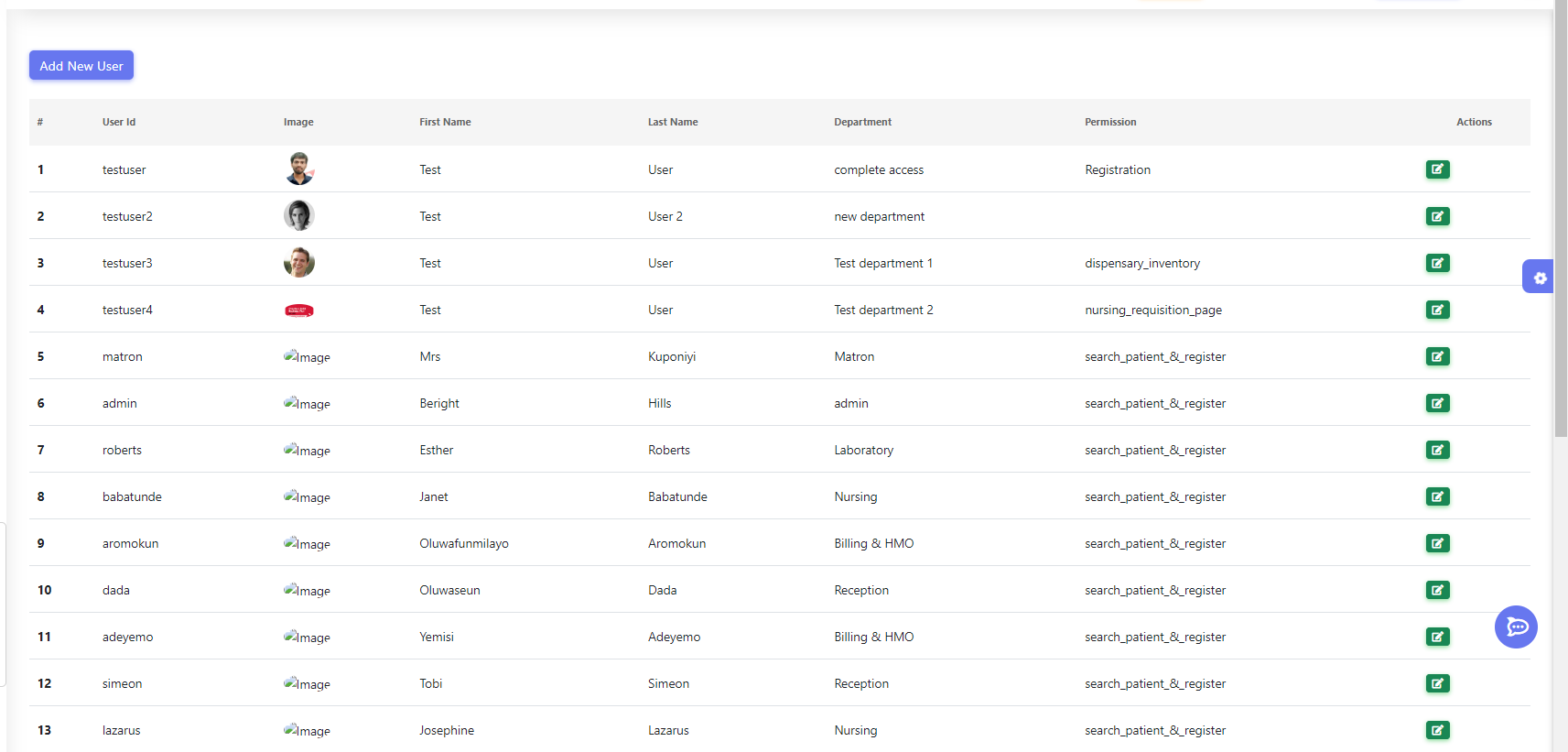
*Figure 8.14 Laboratory*



*Figure 8.15 Laboratory Tests*



*Figure 8.16 Laboratory Test Reports*



*Figure 8.17 User Management*

**References:**

1. Davenport, T., & Kalakota, R. (2019). The potential for artificial intelligence in healthcare. Future healthcare journal, 6(2), 94.
2. Jiang, F., Jiang, Y., Zhi, H., Dong, Y., Li, H., Ma, S., ... & Wang, Y. (2017). Artificial intelligence in healthcare: past, present and future. Stroke and vascular neurology, 2(4), 230-243.
3. Patel, V. L., Arocha, J. F., & Amador, P. (2001). Cognitive models of medical problem-solving: Role of illness scripts. Studies in Health Technology and Informatics, 84, 751-755.
4. Ramesh, A. N., Kambhampati, C., Monson, J. R. T., & Drew, P. J. (2004). Artificial intelligence in medicine. Annals of the Royal College of Surgeons of England, 86(5), 334.
5. Topol, E. (2019). Deep Medicine: How Artificial Intelligence Can Make Healthcare Human Again. Basic Books.
6. A. C. &. G. S. Müller, Introduction to Machine Learning with Python: A Guide for Data Scientists, Sebastopol, CA: O'Reilly Media, 2016.
7. Davenport, T., & Kalakota, R. (2019). The potential for artificial intelligence in healthcare. Future healthcare journal, 6(2), 94.
8. Jiang, F., Jiang, Y., Zhi, H., Dong, Y., Li, H., Ma, S., ... & Wang, Y. (2017). Artificial intelligence in healthcare: past, present and future. Stroke and vascular neurology, 2(4), 230-243.
9. Patel, V. L., Arocha, J. F., & Amador, P. (2001). Cognitive models of medical problem-solving: Role of illness scripts. Studies in Health Technology and Informatics, 84, 751-755.
10. Ramesh, A. N., Kambhampati, C., Monson, J. R. T., & Drew, P. J. (2004). Artificial intelligence in medicine. Annals of the Royal College of Surgeons of England, 86(5), 334.
11. Topol, E. (2019). Deep Medicine: How Artificial Intelligence Can Make Healthcare Human Again. Basic Books.
12. A. C. &. G. S. Müller, Introduction to Machine Learning with Python: A Guide for Data Scientists, Sebastopol, CA: O'Reilly Media, 2016.
13. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Retrieved from <https://arxiv.org/abs/1506.02640>
14. Redmon, J., & Farhadi, A. (2017). YOLO9000: Better, Faster, Stronger. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Retrieved from <https://arxiv.org/abs/1612.08242>
15. Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incremental Improvement. Retrieved from <https://arxiv.org/abs/1804.02767>
16. Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. Retrieved from <https://arxiv.org/abs/2004.10934>
17. Ultralytics. (2020). YOLOv5. Retrieved from <https://github.com/ultralytics/yolov5>
18. Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Advances in Neural Information Processing Systems (NIPS). Retrieved from <https://arxiv.org/abs/1506.01497>
19. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. Proceedings of the European Conference on Computer Vision (ECCV). Retrieved from <https://arxiv.org/abs/1512.02325>
20. Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollar, P. (2017). Focal Loss for Dense Object Detection. Proceedings of the IEEE International Conference on Computer Vision (ICCV). Retrieved from <https://arxiv.org/abs/1708.02002>
21. <https://reactjs.org/docs/getting-started.html>
22. <https://reactjs.org/docs/getting-started.html>
23. <https://stackoverflow.com/>
24. <https://www.youtube.com/watch?v=8Q_QQVQ1HZA&list=PLfFghEzKVmjuhQwKhYXvdU94GSU-6Jcjr>
25. <https://www.youtube.com/watch?v=8kv_njhOeZ0&pp=ygUaQUkgYmFzZSBkaXNlYXNlIHByZWRpY3Rpb24%3D>
26. [www.kaggle.com](http://www.kaggle.com)
27. [www.chat.openai.com](http://www.chat.openai.com)
28. [www.youtube.com](http://www.youtube.com)
29. Collaborations with Doctors and related staffs