**Team**

Talaat Amr Talaat

Abd Alrahman Ahmed Mohamed

Hesham Mohamed Aldaidamony

Alaa Mohamed Mahmoud

Adel Alsaid Abdelazim

Ali Ahmed Ali

Sameh Ahmady Abdelhalem

**Acknowledgments**

First and foremost, we offer our deepest gratitude and praises to God, the Almighty, for His countless blessings throughout the journey of completing this project successfully.

We extend our heartfelt thanks to the Faculty Dean ,Prof. Amal Farouk, for granting us the necessary permissions to visit hospitals, which was crucial for collecting data for our graduation project.

We are also deeply thankful to our professor ,Dr. Osama Elkomy, for his guidance and for providing practical solutions to the challenges we faced during the project. Their expertise and support have been instrumental in our success.

A special thanks to our team members for the stimulating discussions, the long nights working together to meet deadlines, and the enjoyable moments we shared throughout this project.

**Abstract**

In the demanding environment of intensive care units (ICUs), healthcare providers face significant challenges in effectively monitoring patient vital signs and medical data in real-time.

Traditional methods, which rely heavily on manual monitoring and paper-based records, increase workload and the potential for errors.

This approach often hinders communication and coordination between healthcare teams, especially during shift changes, as critical patient information may not be readily accessible.

Additionally, doctors may find it difficult to leave the ICU for essential activities, such as restroom breaks or consultations, due to the need for continuous monitoring and timely decision-making. This situation can lead to increased stress and fatigue among medical staff, potentially compromising patient care quality.

**Table of contents**

Team 1

Acknowledgments 2

Abstract 3

Table of contents 4

**1 chapter one : introduction 8**

* 1. **Main problem 8**
  2. **Solution 9**
  3. **Project idea 9**
  4. **Project goal 9**
  5. **Scope 10**
  6. **The inspiration behind the name 10**
  7. **Project units 10**
  8. **Tools and Technologies 11**

**2 Chapter two : Planning 12**

**2.1 System development life cycle 12**

**2.2 Fisability Analysis 13**

2.2.1 Technical feasibility 13

2.2.2 Economical feasibility 13

2.2.3 Organizational feasibility 13

**2.3 Project plan 14**

**2.4 System request 15**

2.4.1 Sponsor 15

2.4.2 Business need 15

2.4.3 Business requirments 15

2.4.4 Business Value 16

2.4.5 Special issues or constrains 17

**2.5 User flow 18**

**3 Chapter three : Analysis 19**

**3.1 User requirements 20**

3.1.1 Doctors 20

3.1.2 Nurses 20

3.1.3 Health care administrators 20

3.1.4 General users (patients) 21

**3.2 Software requirements 21**

3.2.1 Functional requirements 21

3.2.1.1 Login 21

3.2.1.2 Monitoring 21

3.2.1.3 Streaming 21

3.2.1.4 Computer vision algorithm 21

3.2.1.5 Notification 22

3.2.1.6 Autoinsert vital signs history 22

3.2.2 Nonfunctional requirements 22

3.2.2.1 Performance 22

3.2.2.2 Reliability 22

3.2.2.3 Security 23

3.2.2.4 Usability 23

3.2.2.5 Scalability 23

3.2.2.6 Interoperability 23

**3.3 Business Model 24**

**3.4 Use case diagram 26**

3.4.1 use case login 27

3.4.2 use case monitoring 28

3.4.3 use case vital sign record 29

3.4.4 use case OCR 29

3.4.5 use case streaming by cam 30

3.4.6 use case todolist 30

3.4.7 use case database administration 31

**3.5 Sequence diagram 32**

3.5.1 Login case 32

3.5.2 Monitoring case 33

3.5.3 Todolist case 34

3.5.4 Steaming case 35

**3.6 Component diagram 35**

**3.7 Acivity diagram 36**

**3.8 Class diagram 37**

**3.9 Communication diagram 38**

**3.10 Development digram 39**

**3.11 Dataflow diagram 40**

**3.12 Context diagram 40**

**3.13 Level 0 diagram 41**

**3.14 Entity relationship diagram 42**

**3.15 Physical schema 43**

**4 Chapter four : Design 44**

**4.1 Splash 44**

**4.2 Onboarding 45**

**4.3 Login 45**

**4.4 Shift 46**

**4.5 Patient screens 47**

**5 Chapter five : Implementation 48**

**5.1 Flutter implementation 49**

5.1.1 Choose role 49

5.1.2 Login 51

5.1.3 OTP verification 52

5.1.4 Units 53

5.1.5 Beds 54

5.1.6 Patient personal information 56

5.1.7 Vital sign 57

5.1.8 Todolist 59

5.1.9 Add task 60

5.1.10 Streaming process 61

**5.2 Spring boot implementation 62**

5.2.1 Login service 62

5.2.2 Shift service 62

5.2.3 Medical report service 63

5.2.4 Vital sign report service 63

5.2.5 Patient bed service 64

5.2.6 Todolist service 65

**5.3 Computer vision implementation 66**

**5.4 ESP Cam implementation 67**

**5.5 Database implementation 69**

5.5.1 Doctor 69

5.5.2 ICU 69

5.5.3 Patient 70

5.5.4 Sign 70

5.5.5 Unit 71

5.5.6 Device 71

5.5.7 Coordinate 72

5.5.8 Shift 72

5.5.9 Medicine 72

5.5.10 Nurse 72

5.5.11 Task 73

5.5.12 Medical report 74

5.5.13 image 74

5..5.14 Vital sign report 75

5.5.15 Measure 75

**6 Chapter six : Testing 76**

**6.1 User interface testing 77**

6.1.1 Objective 77

6.1.2 Tools used 77

6.1.3 Approach 77

**6.2 Integration testing 77**

6.2.1 Objective 77

6.2.2 Tools used 77

6.2.3 Approach 77

**6.3 Unit testing 78**

6.3.1 Objective 78

6.3.2 Tools used 78

6.3.3 Approach 78

**7 Chapter seven : Conclusion and Future Work 79**

**7.1 Conclusion 79**

**7.2 Future work 80**

8 Workspaces 82

9 References 82

# Chapter one

Introduction

In this chapter, we will provide a comprehensive overview of the project, delving into its scope and limitations. Additionally, we will explain key terminologies that will be used throughout this document.

**1.1 The Main Problem**

In intensive care units (ICUs), healthcare providers face numerous challenges in effectively monitoring patient vital signs and medical data in real-time.

Traditional methods rely heavily on manual monitoring, where nurses must diligently record vital signs at specific intervals, leading to increased workload and the potential for errors.

Moreover, the reliance on paper-based records can hinder communication and coordination between healthcare teams, especially during shift changes, as critical patient information may not be readily accessible.

doctors may find it challenging to leave the ICU for essential activities, such as restroom breaks or consultations, due to the need for continuous monitoring and timely decision-making.

This situation can lead to increased stress and fatigue among medical staff and may compromise patient care quality. Therefore, there is a need for a solution that allows healthcare providers to remotely monitor ICU patients without compromising the quality of care.

**1.2 Our Solution**

To address these challenges, we propose the development of an innovative ICU management app called VU. VU aims to streamline the monitoring process, enhance communication between healthcare teams, and improve overall patient care outcomes.

By leveraging modern technology and integrating features such as real-time data streaming, automated vital sign monitoring, and electronic record-keeping,

VU seeks to revolutionize the way ICU patients are monitored and managed. By enabling remote monitoring capabilities, VU empowers healthcare providers to monitor patients from anywhere, allowing doctors to attend to other essential activities without compromising patient care quality.

**1.3 Project Idea**

The core idea behind VU is to create a user-friendly mobile application that enables healthcare providers to Manage the ICU and remotely monitor patient vital signs and medical data in real-time.

Through the use of advanced hardware components, such as ESP cams, and sophisticated software solutions, including computer vision algorithms and cloud-based databases, VU will provide doctors and nurses with comprehensive access to patient information anytime, anywhere.

By automating vital sign recording, improving communication between healthcare teams, and centralizing patient data storage,

VU aims to enhance workflow efficiency and patient care delivery in the ICU setting.

**1.4 Project Goals**

Enable real-time monitoring of patient vital signs and medical data through a user-friendly mobile application.

Implement automated vital sign monitoring to reduce manual recording efforts and minimize the risk of errors.

Facilitate seamless communication and collaboration between healthcare teams, including doctors, nurses, and other medical staff.

Provide instant notifications and alerts for critical changes in patient condition to ensure timely intervention and treatment.

Establish a centralized database for storing electronic medical records, enabling easy access and retrieval of patient information during shift changes and emergencies.

Enable doctors to attend to essential activities outside the ICU without compromising patient care quality.

**1.5 Scope**

The VU app is an innovative ICU management solution designed for healthcare institutions and companies across Egypt that prioritize advanced technology. It focuses specifically on enhancing ICU patient monitoring and healthcare delivery.

### 1.6 The Inspiration Behind The Name"VU"

* "VU" derives from the word "view", emphasizing our focus on providing healthcare providers (doctors and nurses) with a clear view and monitoring capability of patient conditions remotely.
* The characters "V" and "U" resemble peaks often seen in ECG (Electrocardiogram) readings as you see : [](https://github.com/heshammohamedaldidamony/VU-ICU-Management-App/blob/main/DOCs/Media/Name%20Inspiration.png)

**1.7 Project Units**

**Frontend Application (Flutter App):** Develop the user interface and functionality of the mobile application using Flutter framework, allowing healthcare providers to access patient data, monitor vital signs, and communicate with other team members.

**Backend Server (Spring Boot):** Create the backend server to handle data processing, user authentication, and communication between the frontend application and the database, ensuring smooth operation and data integrity.

**Database Management (MySQL):** Design and manage the database schema to store patient information, medical records, vital sign data, and other relevant data, providing a centralized and secure storage solution.

**Hardware Integration (ESP Cams):** Integrate ESP cams with the system to capture real-time video streams of patient monitors, allowing for remote monitoring of patient vital signs and medical data.

**Computer Vision Algorithms:** Develop and implement computer vision algorithms to analyze video streams and extract vital sign information, such as heart rate, respiratory rate, and oxygen saturation levels, providing automated monitoring capabilities.

### 1.8 Tools and Techniques

**User Interface Design:**

* Figma

**Front-end Development:**

* Flutter
* Visual Studio Code

**Back-end Development:**

* Spring Boot Framework
* IntelliJ IDEA

**Database Management:**

* SQL
* MySQL Database Management System

**Hardware:**

* Arduino IDE
* ESP32-CAM

**Computer Vision:**

* Open VV

# Chapter Two

Planning

In this chapter, we will delve into the planning phase of the VU project. We will outline the project timeline, key milestones, and deliverables. Additionally, we will discuss the resources required, risk management strategies, and the roles and responsibilities of team members. This chapter aims to provide a clear roadmap for the successful execution of the project, ensuring all stakeholders are aligned and prepared for the upcoming phases.

### 2.1 System Development Life Cycle

In developing this project, we adhered to the System Development Life Cycle (SDLC) methodology. This approach encompasses seven phases: planning, analysis, design, development, testing, deployment, and maintenance. Each phase plays a critical role in ensuring the successful completion and sustainability of the project.



**2.2 Feasibility Analysis**

**2.2.1 Technical Feasibility:**

**Hardware Components (ESP cams):** Assess the availability and compatibility of ESP cams in the market. Ensure that the chosen hardware meets the requirements for real-time monitoring and data capture in ICU environments.

**Software Solutions (Computer Vision Algorithms):** Evaluate the technical capabilities of computer vision algorithms for accurate vital sign monitoring. Ensure that the software can handle the complexities of medical data and deliver reliable results.

**Cloud-Based Databases:** Check the scalability, security, and reliability of cloud-based databases. Ensure that they can handle the large volumes of data generated by continuous monitoring in ICU settings.

**2.2.2 Economical Feasibility:**

Pharmaceutical companies can be contracted to display their drugs in app.

**Cost of Development:** Estimate the costs associated with developing the VU application, including software development, hardware acquisition, and integration with existing healthcare systems.

**Return on Investment (ROI):** Assess the potential benefits in terms of increased efficiency, reduced errors, and improved patient care. Determine whether the expected ROI justifies the initial investment.

**2.2.3 Organizational Feasibility:**

**Integration with Existing Systems:** Consider the compatibility of VU with existing hospital information systems and workflows. Evaluate how easily the application can be integrated into the daily routines of healthcare providers.

**Training and Adoption:** Assess the willingness of healthcare professionals to adopt new technology. Develop a training plan to ensure a smooth transition and user acceptance.

**Regulatory Compliance:** Investigate and comply with healthcare regulations and data privacy laws. Ensure that VU meets the necessary standards for medical applications.

**2.3 Project Plan**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Task Name | Duration | Start | Finish | job |
| Define The Problem ,Initial Solution | 20 day | 9/1/23 | 9/20/23 | All Team |
| Visit And Listening to Medical Stuff Problems | 7 days | 9/23/23 | 10/10/23 | All team |
| Define Real Solutions | 7 days | 10/10/23 | 10/17/23 | All Team |
| Integration With Patient Monitor | 9 days | 10/20/23 | 10/29/23 | All team |
| **Looking For A Replacement For Integration** | 10 days | 11/1/23 | 11/10/23 | Supervisors |
| Planning | 5 days | 11/15/23 | 11/20/23 | All team |
| Analysis | 25 day | 11/22/23 | 12/17/23 | Analyst |
| Assigning Tasks | 2 days | 12/18/23 | 12/20/23 | Leader |
| Design Data Base | 38 day | 12/20/23 | 1/28/24 | Designer |
| Design (ui,ux) | 85 day | 2/1/24 | 3/2/24 | UI,UX |
| Start Backend Implementation | 85 day | 3/5/24 | 6/1/24 | Backend |
| Work With ESP-CAM | 85 day | 3/5/24 | 6/1/24 | IOT Engineer |
| Drawing UML Diagrams | 85 day | 3/5/24 | 6/1/24 | Analyst |
| Implement Computer Vision Algorithm | 85 day | 3/5/24 | 6/1/24 | IOT Engineer |
| Finish APP Screens | 85 day | 3/5/24 | 6/1/24 | Computer vision dev |
| Integrate All Works | 12 day | 6/3/24 | 6/15/24 | Flutter |
| Testing | 15 day | 6/15/24 | 7/1/24 | All Team |
| Finish The Documentation | 15 day | 6/15/24 | 7/1/24 | Analyst |
| Make The Discussion Presentation | 6 days | 7/1/24 | 7/7/24 | All Team |

**2.4 System Request**

**2.4.1 Sponsor:**

Faculty of Computer and Information - Medical Informatics Department – Zagazig University.

**2.4.2 Business need:**

The development of the ICU management app stems from the pressing need to modernize and optimize the processes involved in intensive care units. Current methods of patient monitoring and data management are often manual, inefficient, and prone to errors. Moreover, communication between healthcare teams can be fragmented, leading to delays in critical decision-making and potentially compromising patient care quality. The app aims to address these challenges by providing a comprehensive solution for remote patient monitoring, real-time data access, and streamlined communication.

**2.4.3 Business requirements:**

**Improved Patient Care Delivery:** The primary goal of the VU project is to enhance the delivery of patient care in intensive care units (ICUs) by providing healthcare providers with innovative tools and technologies to monitor patient vital signs, streamline communication, and facilitate timely interventions.

**Enhanced Healthcare Collaboration:** The system aims to promote collaboration and teamwork among healthcare professionals by facilitating seamless communication, task management, and information sharing, thereby improving coordination and decision-making in patient care.

**Efficiency and Productivity:** By automating manual processes and providing intuitive interfaces, the VU system seeks to increase operational efficiency and productivity within healthcare facilities, allowing medical staff to focus more on patient care and less on administrative tasks.

**Data-driven Insights and Decision Support:** Through comprehensive data collection and analysis capabilities, the VU system aims to provide healthcare providers with actionable insights and decision support tools to optimize treatment plans, improve outcomes, and personalize patient care.

**Regulatory Compliance and Data Security:** Ensuring compliance with healthcare regulations and standards, as well as maintaining robust data security measures, is paramount to safeguarding patient privacy and confidentiality, as well as maintaining trust in the system.

**User Training and Support:** The successful adoption and utilization of the VU system rely on providing comprehensive user training and ongoing support to healthcare professionals, empowering them to leverage the system effectively in their daily clinical workflows.

**Scalability and Adaptability:** As healthcare needs evolve and technology advances, the VU system should be scalable and adaptable to accommodate future growth, changes in clinical practices, and emerging healthcare trends, ensuring long-term sustainability and relevance.

**Cost-effectiveness and Return on Investment (ROI):** While delivering significant value in terms of improved patient outcomes and operational efficiency, the VU project should also demonstrate cost-effectiveness and a positive ROI for healthcare organizations, aligning with their financial objectives and constraints.

**Customer Satisfaction and Stakeholder Engagement:** Ultimately, the success of the VU project hinges on the satisfaction of its users – both healthcare professionals and patients – as well as active engagement and support from stakeholders, including hospital administrators, IT personnel, and regulatory bodies

**2.4.4 Business value:**

**Improved Operational Efficiency:** Automation of vital sign monitoring and streamlining of processes lead to increased efficiency in ICU operations, allowing medical staff to focus more on patient care rather than administrative tasks.

**Enhanced Communication and Collaboration:** Improved communication between healthcare teams through real-time data sharing and electronic record-keeping fosters collaboration, leading to better coordination and decision-making.

**Empowerment of Healthcare Providers:** Remote monitoring capabilities empower healthcare providers to monitor patients from anywhere, allowing them to attend to essential activities without compromising patient care quality.

**Enhanced Patient Care Delivery:** The overall solution results in enhanced patient care delivery, with timely interventions, improved treatment plans, and personalized care based on real-time data and historical medical records.

**Cost Savings and Resource Optimization:** Automation and optimization of processes lead to cost savings associated with reduced manual labor, improved resource allocation, and optimized bed utilization.

**Compliance and Data Security:** Implementation of electronic record-keeping ensures compliance with healthcare regulations and standards while maintaining robust data security measures, safeguarding patient privacy and confidentiality.

**User Satisfaction and Stakeholder Engagement:** The solutions aim to increase user satisfaction among healthcare professionals and patients by improving workflow efficiency and patient care outcomes. Active engagement and support from stakeholders further contribute to the success of the ICU management system.

**2.4.5 Special issues or constrains:**

**Integration Challenges with Existing Hospital Systems:**The absence of an established hospital system and database poses integration challenges, potentially delaying the deployment of the ICU management app. Efforts will be required to develop interoperability solutions to seamlessly integrate the app with existing hospital infrastructure and workflows.

**Compliance with Regulatory Standards:**Compliance with healthcare regulations and standards, such as HIPAA (Health Insurance Portability and Accountability Act) in the United States, poses a constraint on the development and deployment of the app. Ensuring that the app adheres to strict data privacy and security requirements is essential to meet regulatory standards and maintain patient confidentiality.

**Scalability and Adaptability:**The app must be designed to scale and adapt to varying hospital environments and operational needs. Considerations for scalability, both in terms of user volume and feature expansion, should be incorporated into the design to accommodate future growth and changes in healthcare practices.

**Limited Budget and Resource Allocation:**Budget constraints and resource limitations may impact the development and implementation of the ICU management app. Effective resource allocation and budget management strategies will be essential to ensure the successful delivery of the project within the allocated budget and timeline.

**Stakeholder Acceptance and Adoption:**Ensuring stakeholder acceptance and adoption, including healthcare providers, administrators, and patients, is critical for the success of the app. Addressing user concerns, providing adequate training and support, and fostering a culture of acceptance towards digital healthcare solutions are key considerations to overcome potential resistance to change.

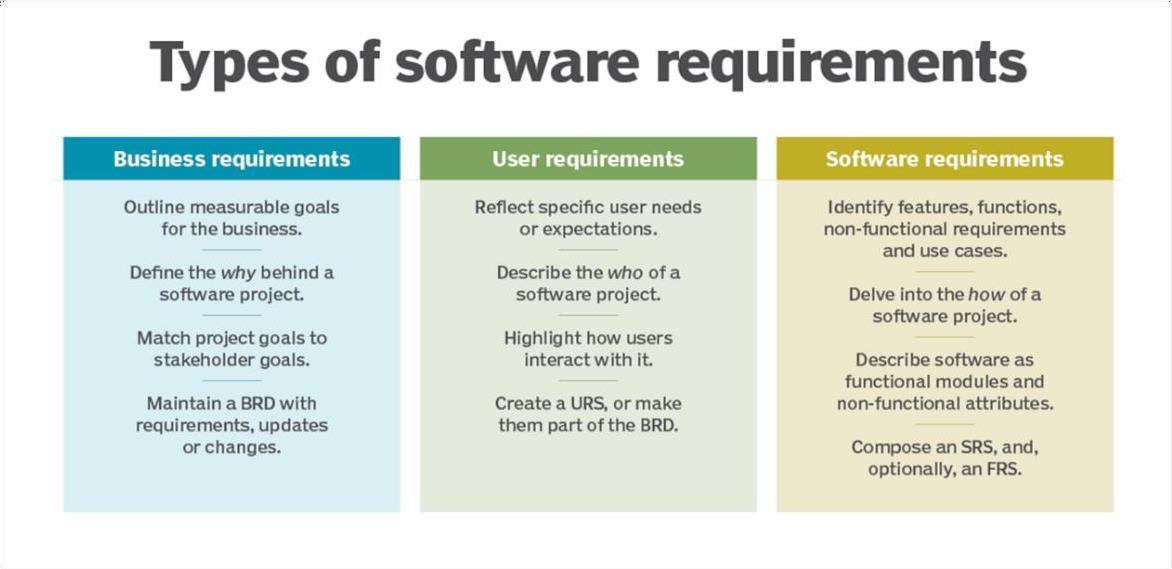
**2.5 User Flow**

* -In the beginning some widgets with panting and tests to explain the power and using of app.
* Entry point: Starting the application either as a doctor or nurser (each one of them have access to different things).
* page to enter your number to authentication with send verifying code OTP to secure that the really doctor or nurse number.
* After sure that is doctor or nurse phone, we check the number in hospital.Then set the profile picture.
* Opening home page that contains units each units have the beds numbers in it, and the doctor either the nurse supervisor of.
* Opening the sections of intensive care which navigate you to the beds in each intensive care showing the fall and available beds with patients.
* Showing beds as a card in drop list each card showing patient monitor streaming -name – diagnosis– current nurse responsible
* Enter any of the bed leads you to page of very specific details of this bed (patient of this bed) his vista functions as details of stating day the patients came to hospital.Where you can find the personal information ,todolist ,vital signs reports ,medical reports.
* In vital signs reports page user will find the reports ordered the date . And the medical reports as well
* In todolist page nurse will find the tasks were setted by the doctor that she can do it in its shift

# Chapter Three

Analysis

In this chapter, we will conduct a thorough analysis of the VU project, examining both functional and non-functional requirements, and identifying key challenges and opportunities. We will present various UML diagrams to visualize the system's structure and behavior. Additionally, we will detail the database analysis and design, including the database schema, entity-relationship diagrams. This comprehensive analysis will provide a solid foundation for the project's design and implementation phases.



**3.1 User Requirements :**

****3.1.1 Doctors:****

* **Authentication:** Secure login via phone number with OTP verification.
* **Shift Management:** Access to real-time shift schedules and patient/unit assignments.
* **Patient Monitoring:** View detailed patient profiles, monitor vital signs via real-time streaming from ESP-Cam.
* **Task Management:** Create and manage to-do lists for patients, assign tasks to nurses.
* **Notifications:** Receive critical alerts for immediate response.
* **Data Automation:** Automatic recording of vital signs data into patient histories.

**3.1.2 Nurses:**

* **Authentication:** Secure login via phone number with OTP verification.
* **Shift Management:** Access to real-time shift schedules and patient/unit assignments.
* **Patient Monitoring:** View detailed patient profiles, monitor vital signs via real-time streaming from ESP-Cam.
* **Task Execution:** View and complete tasks assigned by doctors, update task statuses.
* **Notifications:** Receive alerts for critical patient condition changes.
* **Training and Support:** Access to training resources for effective app utilization.

**3.1.3 Healthcare Administrators:**

* **User Management:** Manage user accounts, roles, and permissions.
* **Data Management:** Ensure data integrity and compliance with healthcare regulations.
* **System Monitoring:** Monitor system performance and responsiveness.
* **Backup and Recovery:** Implement backup and recovery processes to prevent data loss.
* **Compliance and Security:** Ensure adherence to data security and privacy standards.

**3.1.4 General Users (Patients and Families):**

* **Access to Information:** View patient information as permitted.

**3.2 Software Requirements :**

**3.2.1** Functional Requirements

**3.2.1.1** **Logging In:**

Users (doctors and nurses) should be able to log in to the system using their registered phone number authentication methods.

- **OTP Verification:** Upon entering their phone numbers, users should receive a one-time password (OTP) via SMS or another method of communication.

**3.2.1.2** **Monitoring:**

- **ٍSee Shift:**Each doctor and nurse should have access to a personalized view of their assigned shift, displaying the patients and units they are supervising in real time.

- **Care Patients:** Users should be able to access detailed patient profiles, including personal information , Vital signs (by streaming) and history reports , to provide appropriate care.

**- To-Do List:** Doctors should be able to create and manage to-do lists for specific patients, assigning tasks to nurses . And nurse be able to do these tasks.

**3.2.1.3** **Streaming:**

The system should enable doctors and nurses to remotely monitor patient vital signs and medical data in real-time by streaming video from patient monitors by ESP-Cam.

**3.2.1.4** **Computer Vision Algorithm:**

The system should utilize a computer vision algorithm to analyze video frames from patient monitors, extracting vital signs data such as heart rate, oxygen levels, etc.

**3.2.1.5** **Notification:**

The system should send critical alerts to doctors and nurses regarding any critical changes in a patient's condition, enabling them to provide immediate guidance and support.

**3.2.1.6** **Auto-Insert Vital Signs History:**

The system should automatically insert vital signs data extracted by the CV algorithm into the patient's medical history, creating a comprehensive record over time.

**3.2.2** Non-Functional Requirements:

**3.2.2.1** **Performance**:

- The overall system latency, including network transmission, data processing, and user interface responsiveness, should be minimized to provide users with a smooth and uninterrupted experience.

- Real-Time Streaming: The system should ensure seamless real-time streaming of video data captured by the ESP cam, with minimal buffering and delays.

- Computer Vision Algorithm:The processing time for the

computer vision algorithm should be optimized to ensure timely extraction of vital signs data from the video stream.

- Internet:A reliable network ensures minimal latency and buffering during streaming, allowing doctors and nurses to monitor patient vital signs without delays or interruptions.

- High-speed internet ensures that frames captured by the ESP cam are transmitted quickly to the app for processing and display, maintaining real-time monitoring capabilities.

**3.2.2.2** **Reliability**:

- The system should be reliable and available 24/7 to provide continuous service

- The system should have backup and recovery mechanisms to prevent data loss and minimize downtime

- Data Integrity: Patient data stored in the system should be accurate, consistent, and protected against corruption or unauthorized modification.

- Error Handling: The system should provide informative error messages and gracefully handle unexpected errors to prevent system crashes and ensure a seamless user experience.

**3.2.2.3** **Security**:

- Authentication: User authentication should be secure and follow best practices. Users should only have access to data and features appropriate to their roles and permissions.

**3.2.2.4** **Usability**:

- The system should be easy to use and navigate for users of different ages, backgrounds, and abilities .

- The system should have a user-friendly interface that provides clear and concise information to users .

- The system should provide users with guidance and feedback to help them complete tasks efficiently and effectively.

**3.2.2.5** **Scalability**:

- The system should be scalable and able to handle growing demands from users and healthcare organizations.

- The system should be able to accommodate new features and functionalities without compromising performance, reliability, or security .

- The system should use cloud-based infrastructure and technologies to enable flexible and cost effective scaling.

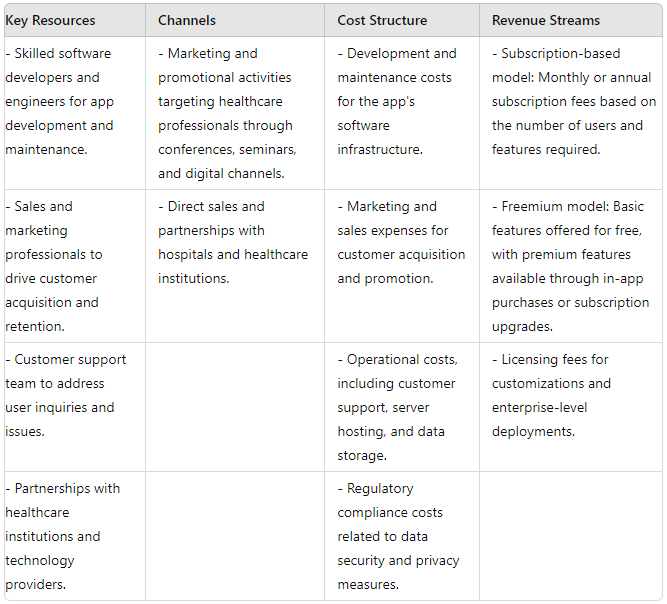
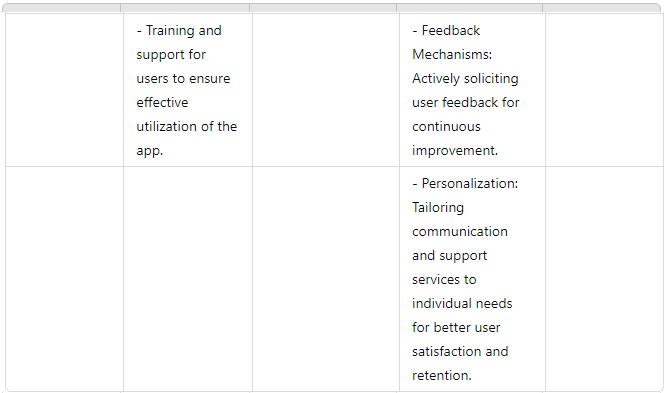
**3.2.2.6** **Interoperability**:

- The system should be interoperable and able to exchange data with other healthcare systems, such as electronic health records (EHRs), medical devices, and telemedicine platforms.

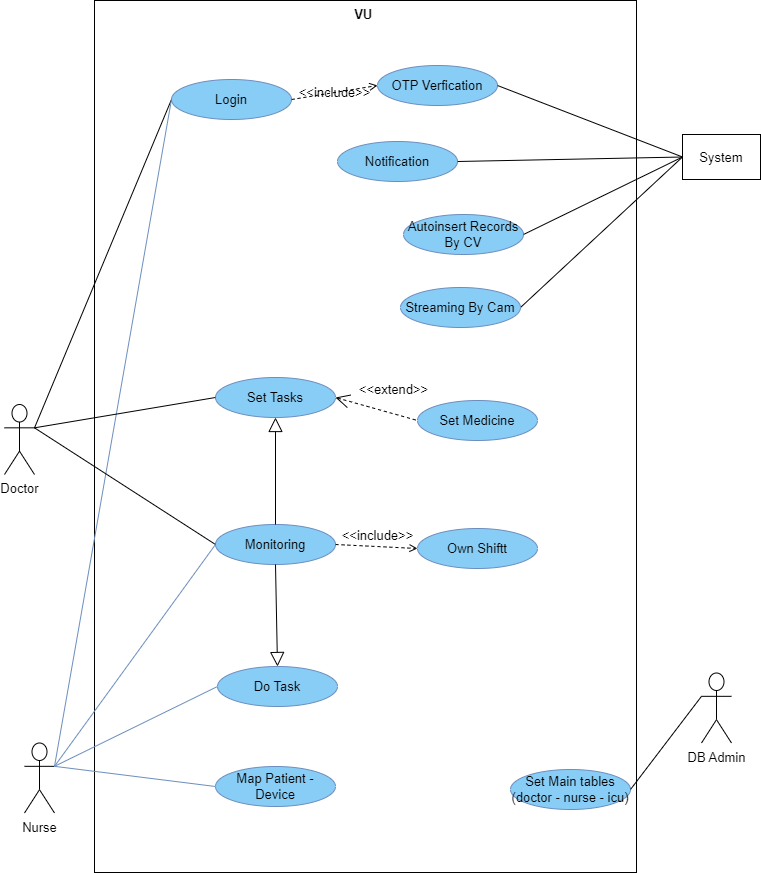
- The system should support different data formats and standards to facilitate data sharing and collaboration among healthcare organizations.

**3.3 Business Model:**

****



**3.4 Use Case Diagram:**

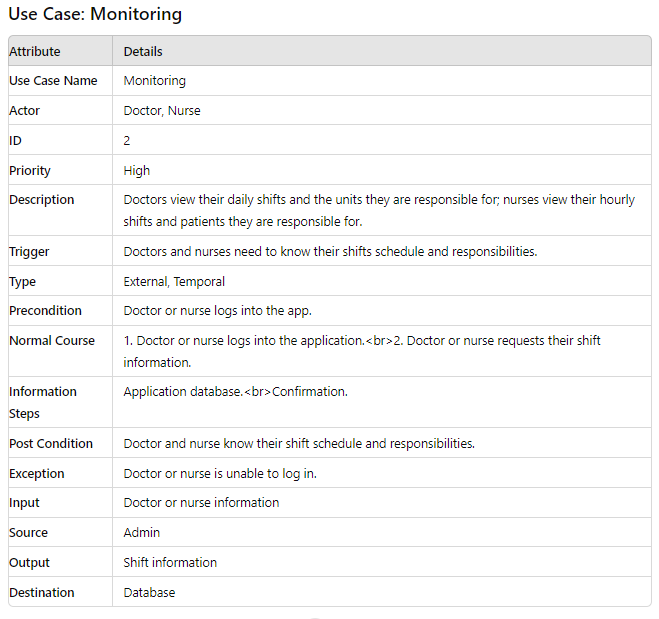


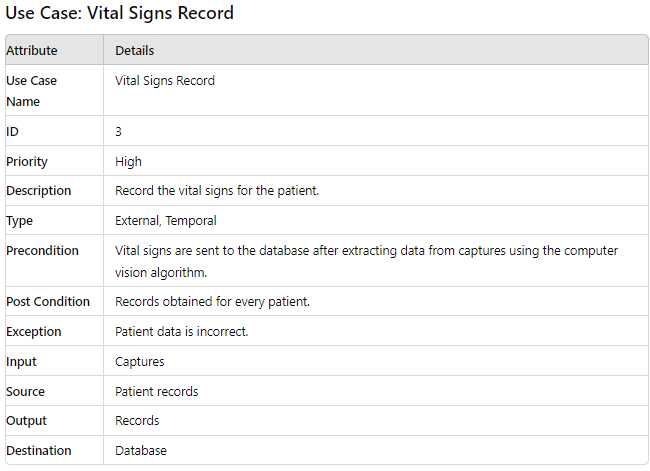
**Use Cases Tables:**

**3.4.1**

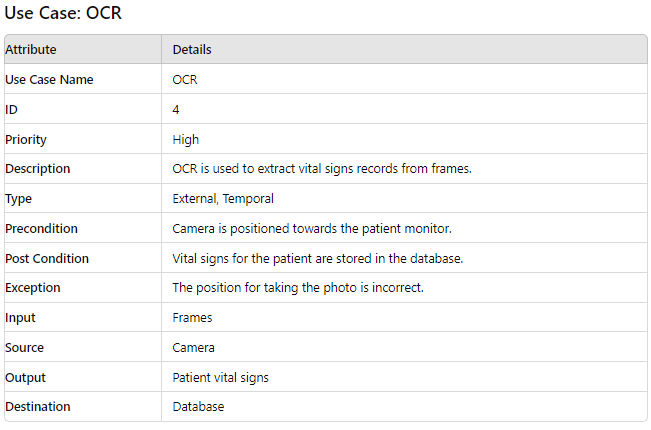


**3.4.2**

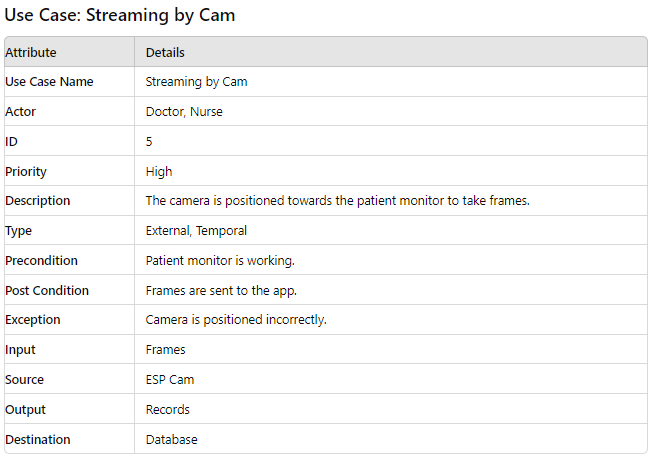


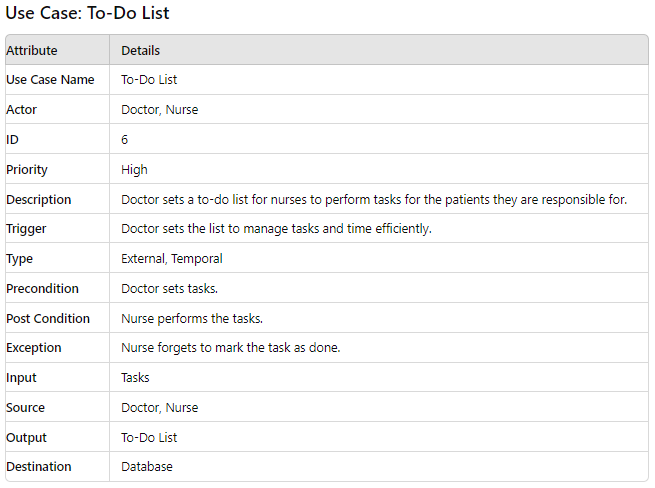
**3.4.3**

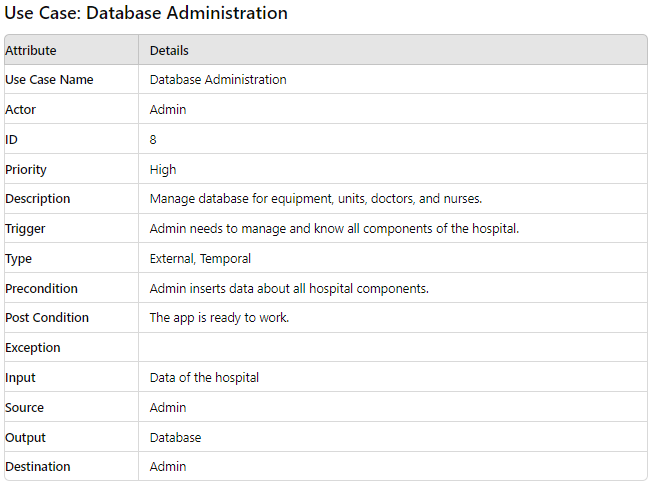
**3.4.4**



**3.4.5**

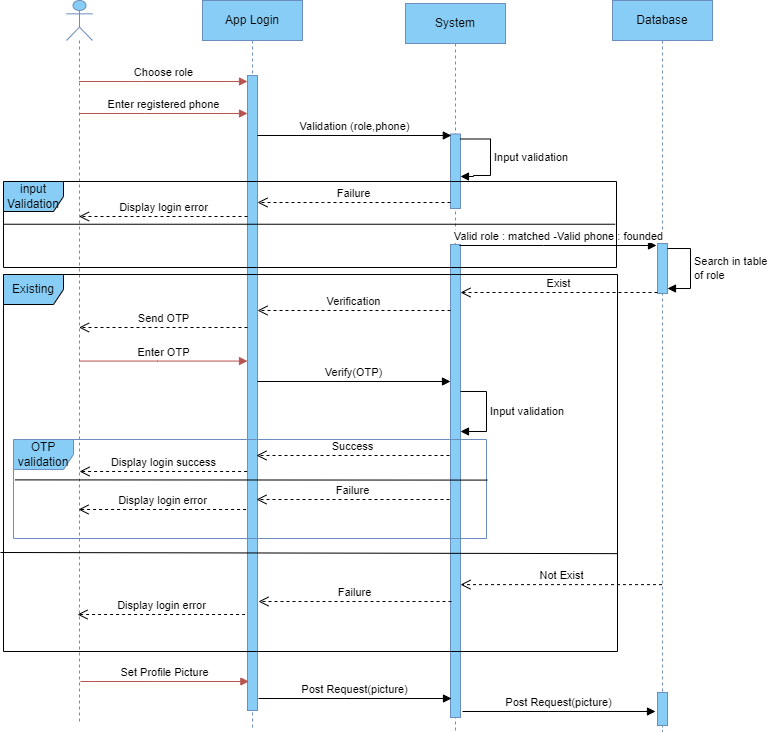


**3.4.6**

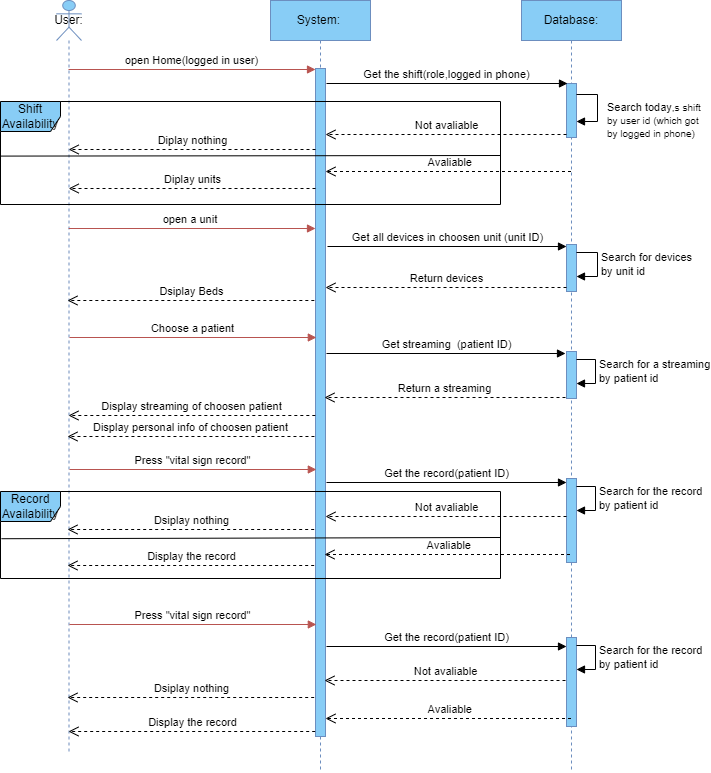
**3.4.7**

**3.5 Sequence Diagram:**

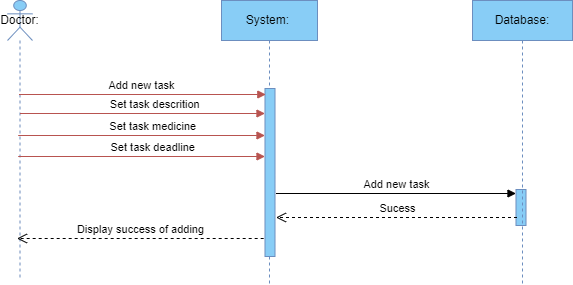
**3.5.1** Login Case

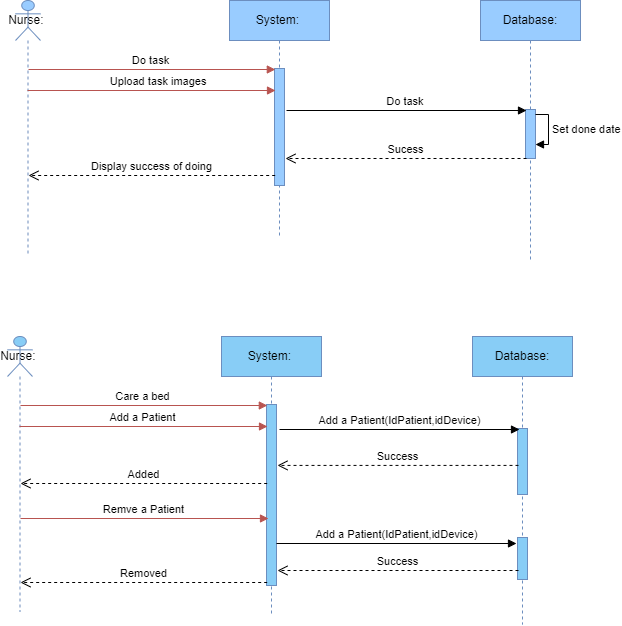


**3.5.2** Monitoring Case

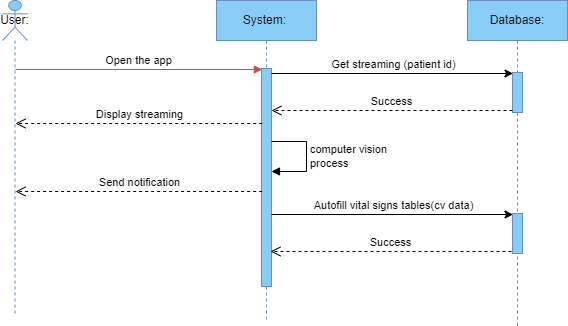


**3.5.3** Todolist Case

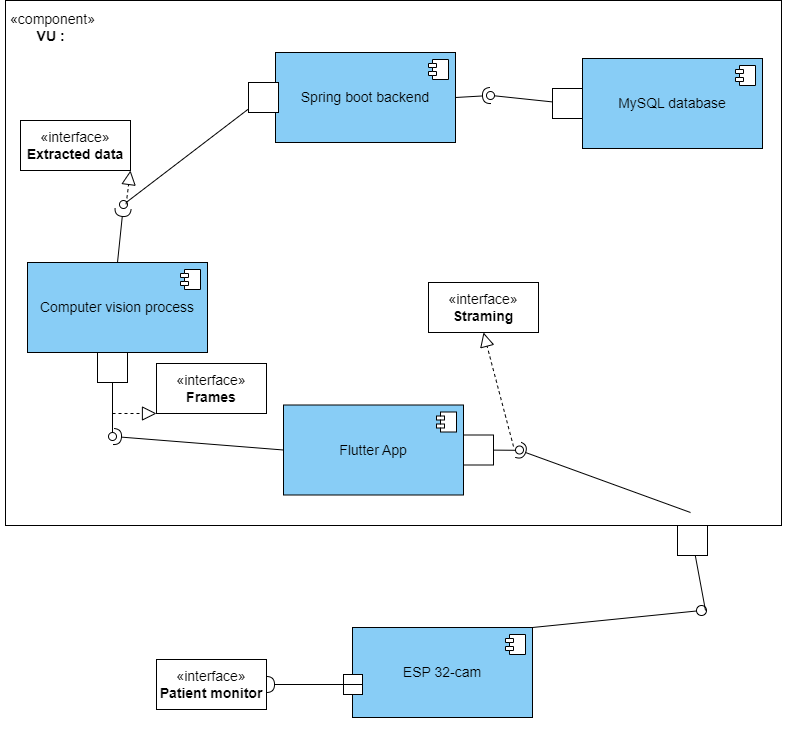




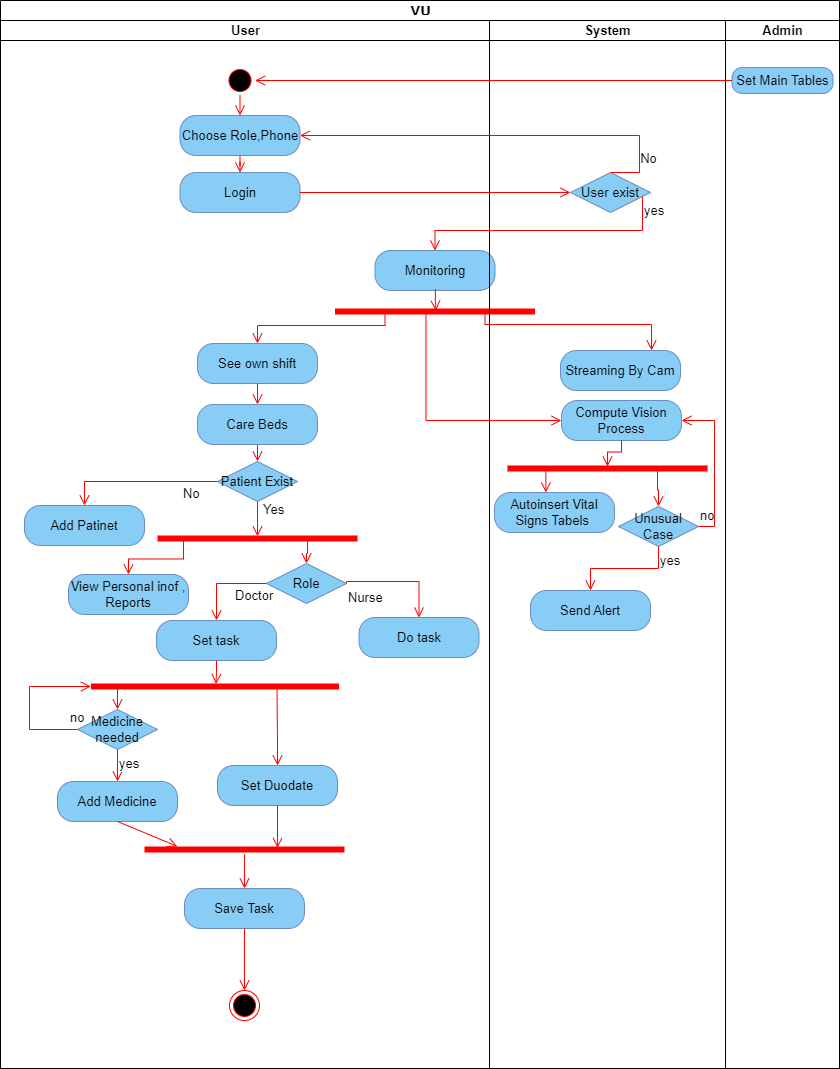
**3.5.4** Streaming Case



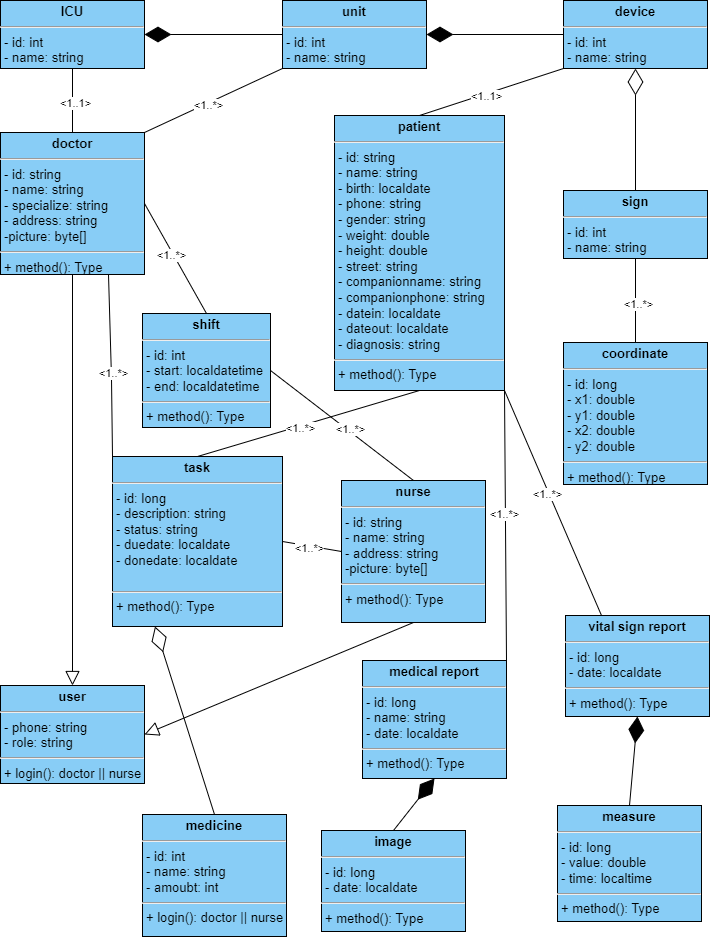
**3.6 Component Diagram:**



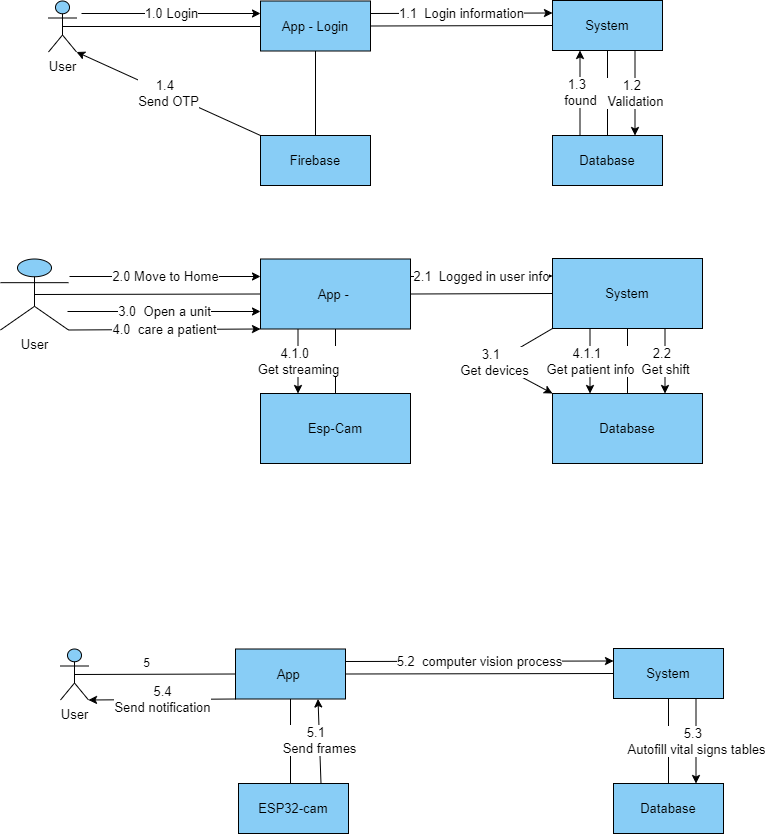
**3.7 Activity Diagram:**



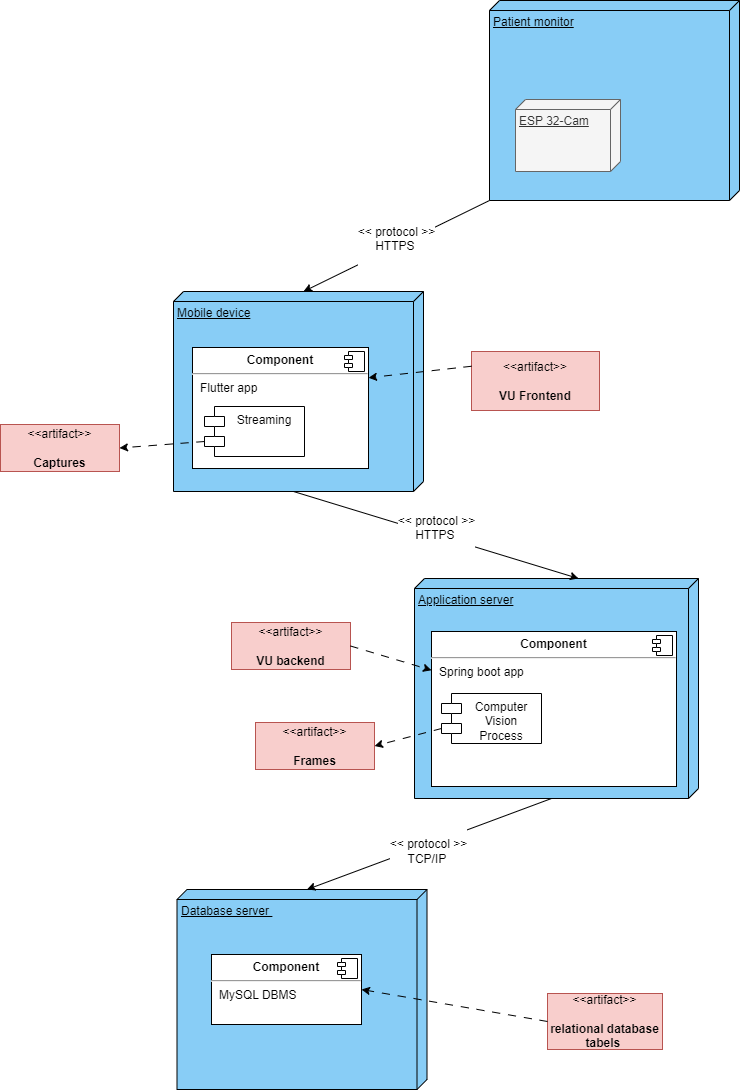
**3.8 Class Diagram:**



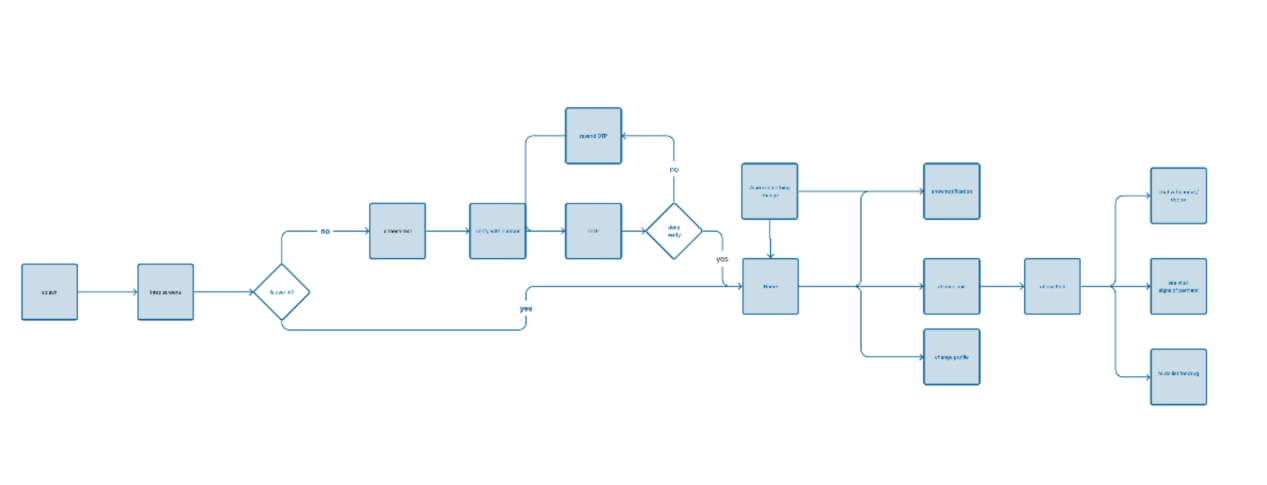
**3.9 Communication Diagram:**



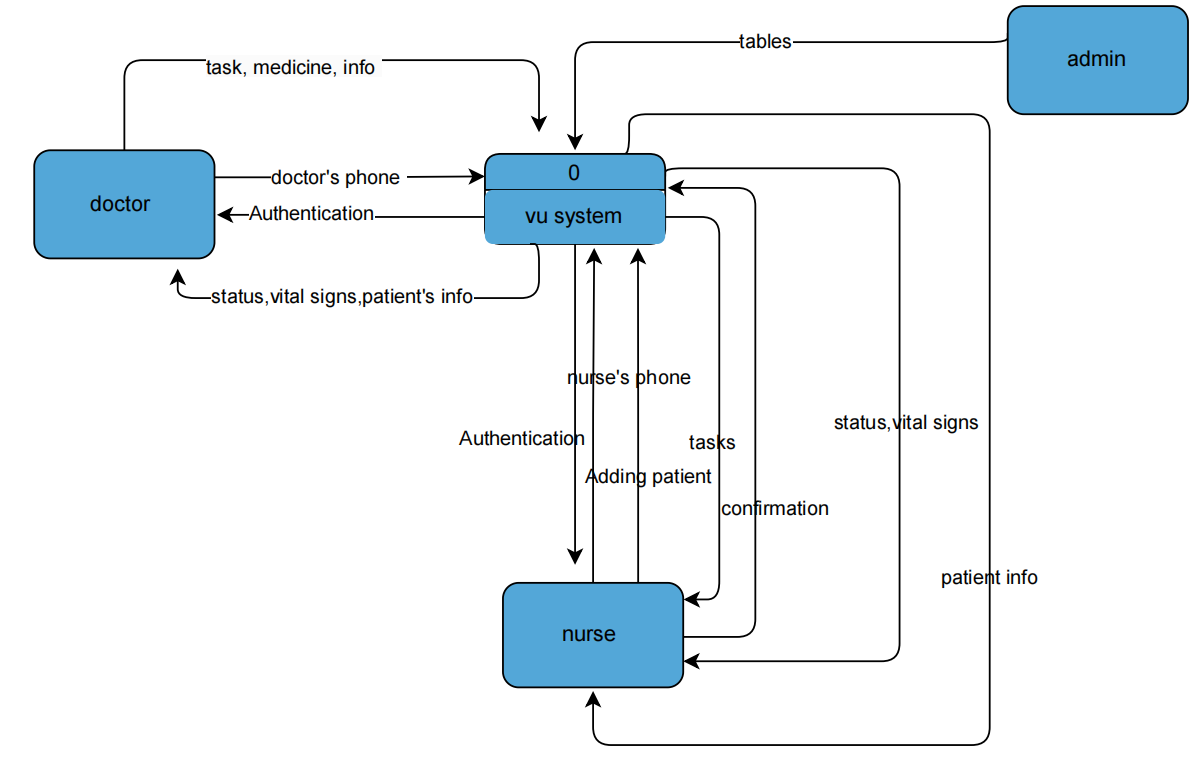
**3.10 Deployment Diagram:**



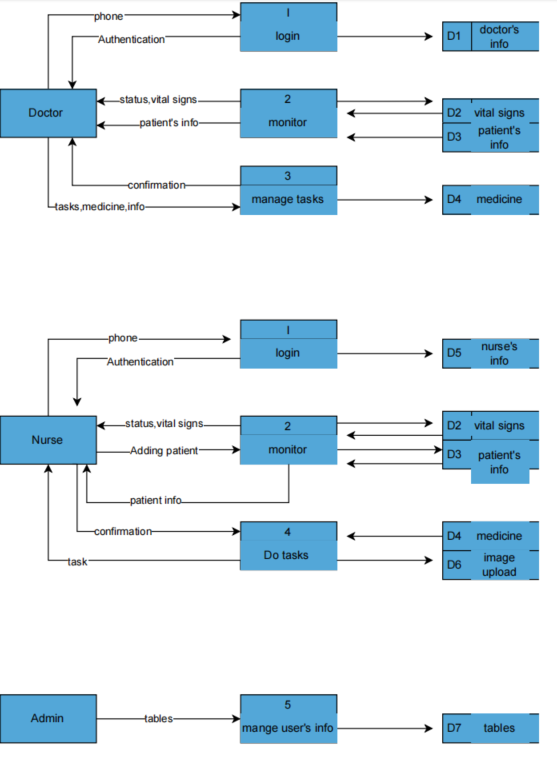
**3.11 Data Flow Diagram:**



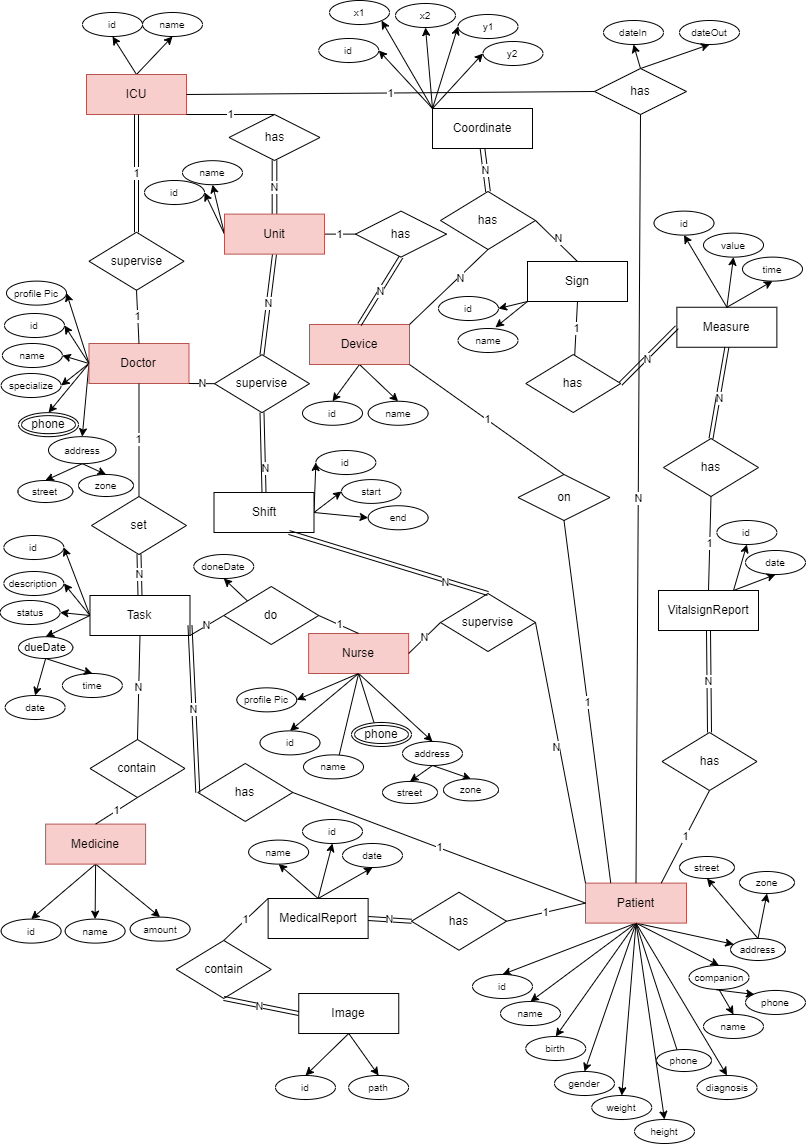
**3.12 Context Diagram:**



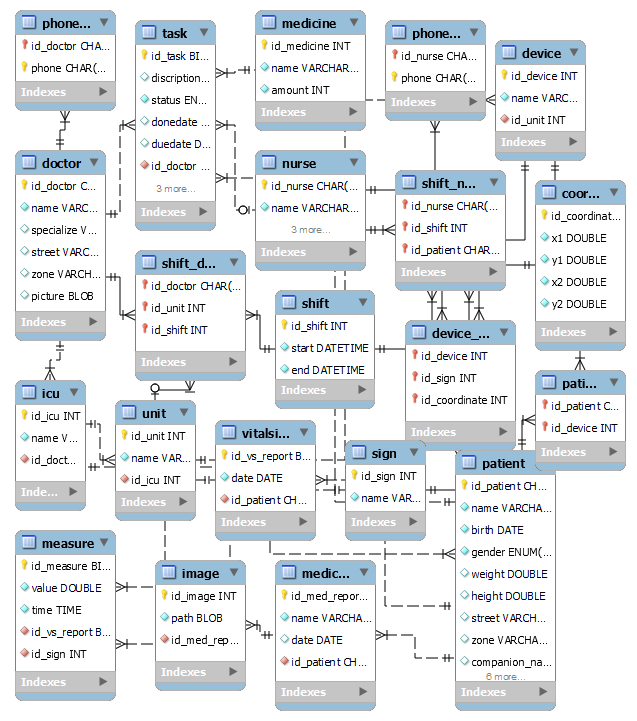
**3.13 Level 0**



**3.14 Entity Relationship Diagram:**



**3.15 Physical Schema:**

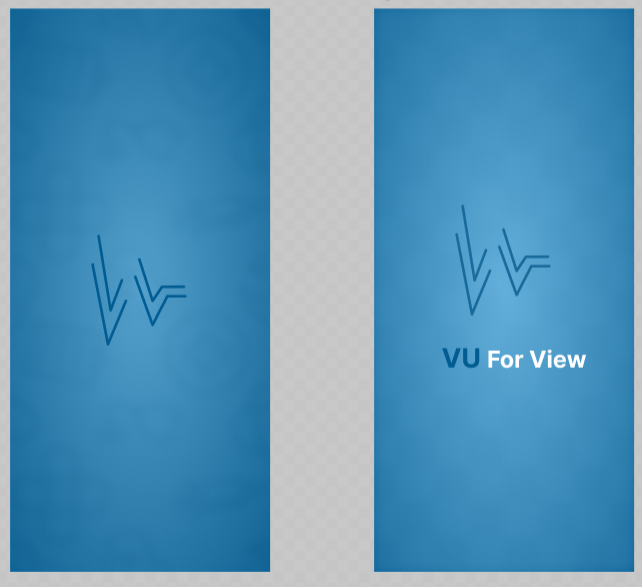


# Chapter four

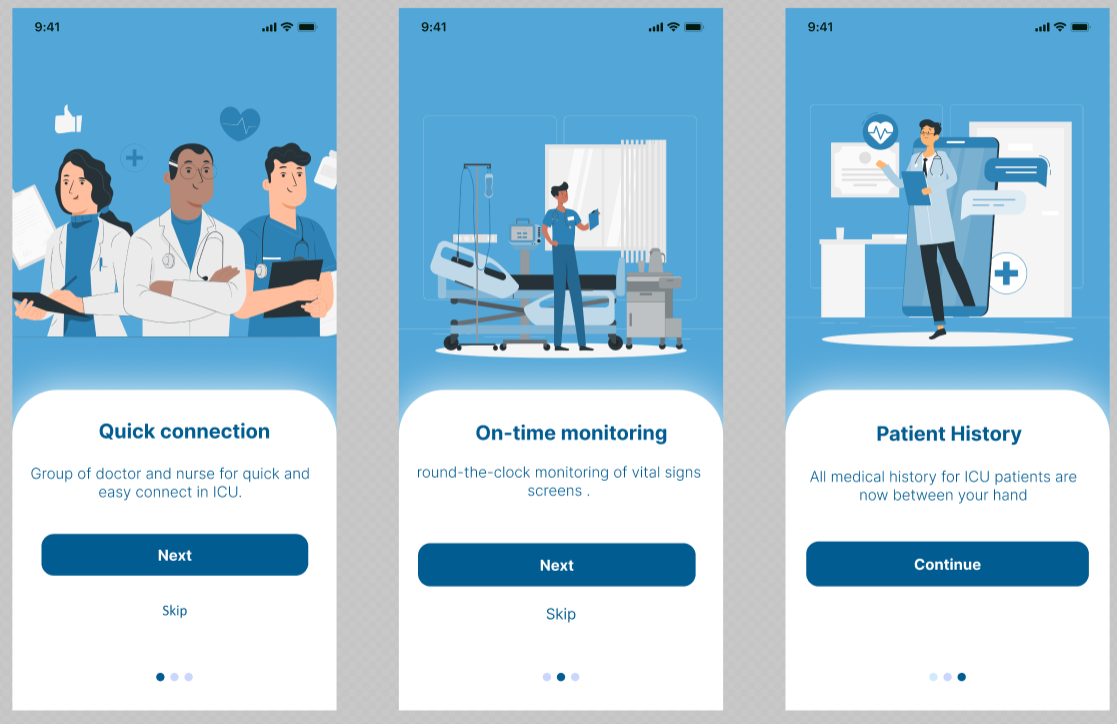
Design

In this chapter, we delve into the design phase of the VU project, laying the foundation for our innovative ICU management system. We will explore the architectural blueprint, user interface design that collectively shape the functionality and user experience of our application

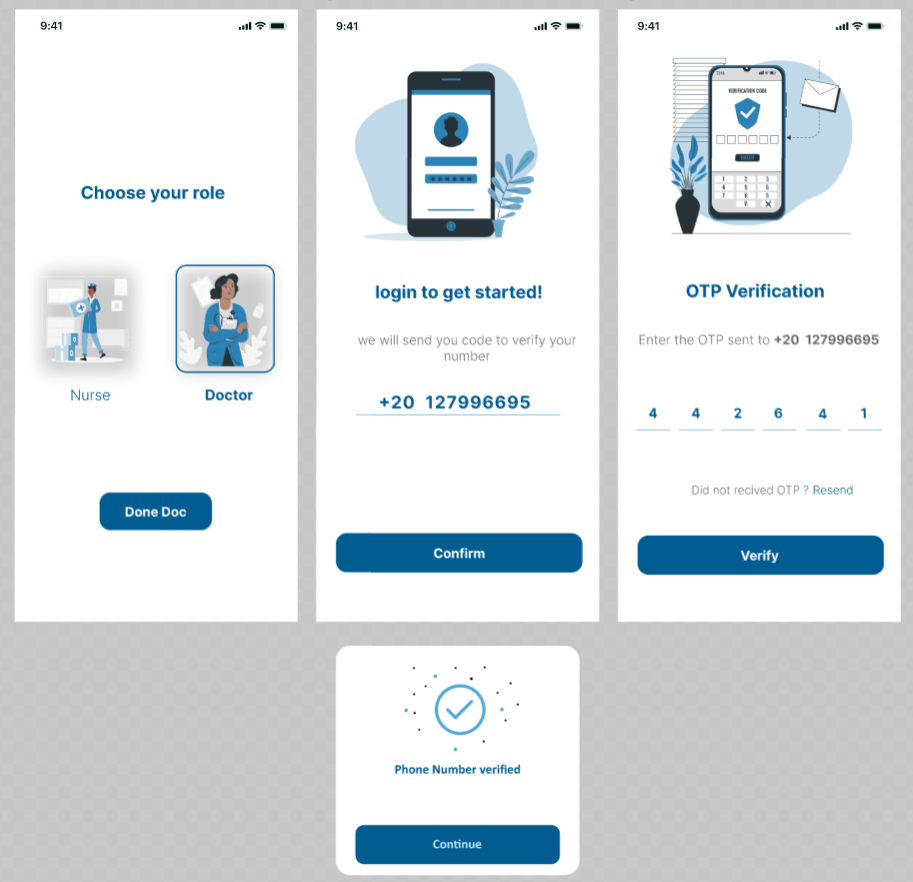
**4.1 Splash**



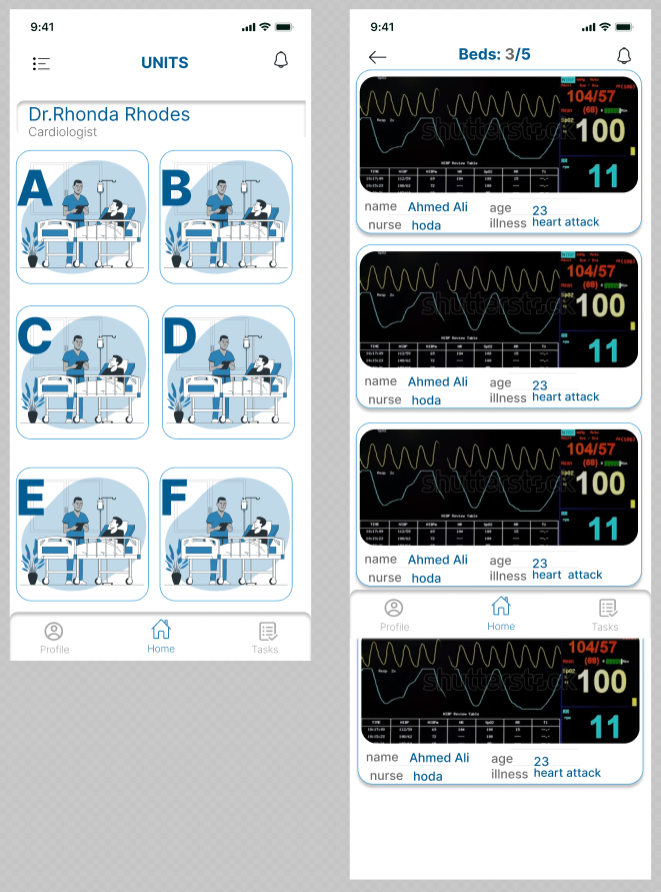
**4.2 On Boarding**



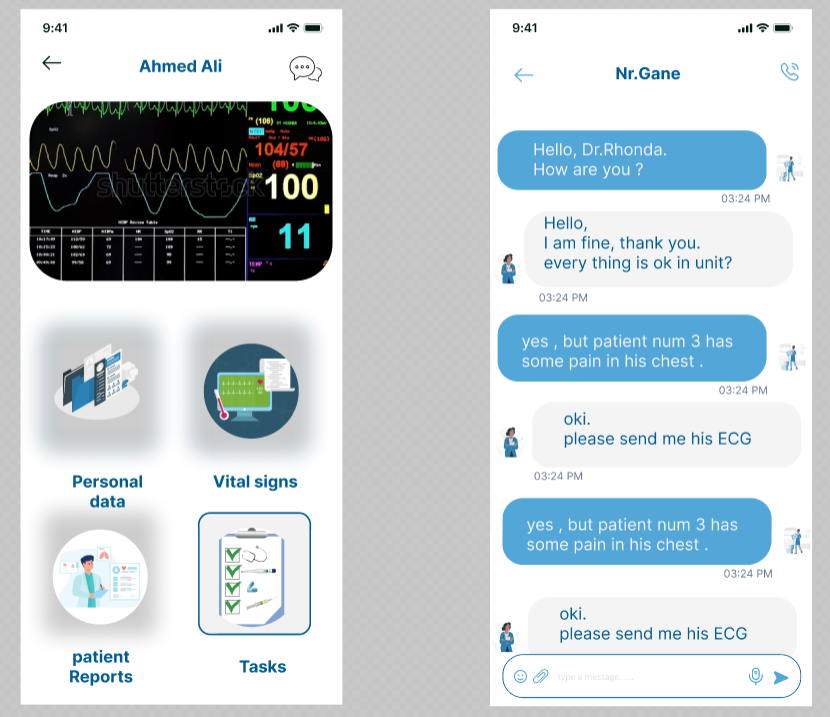
**4.3 Login Process**

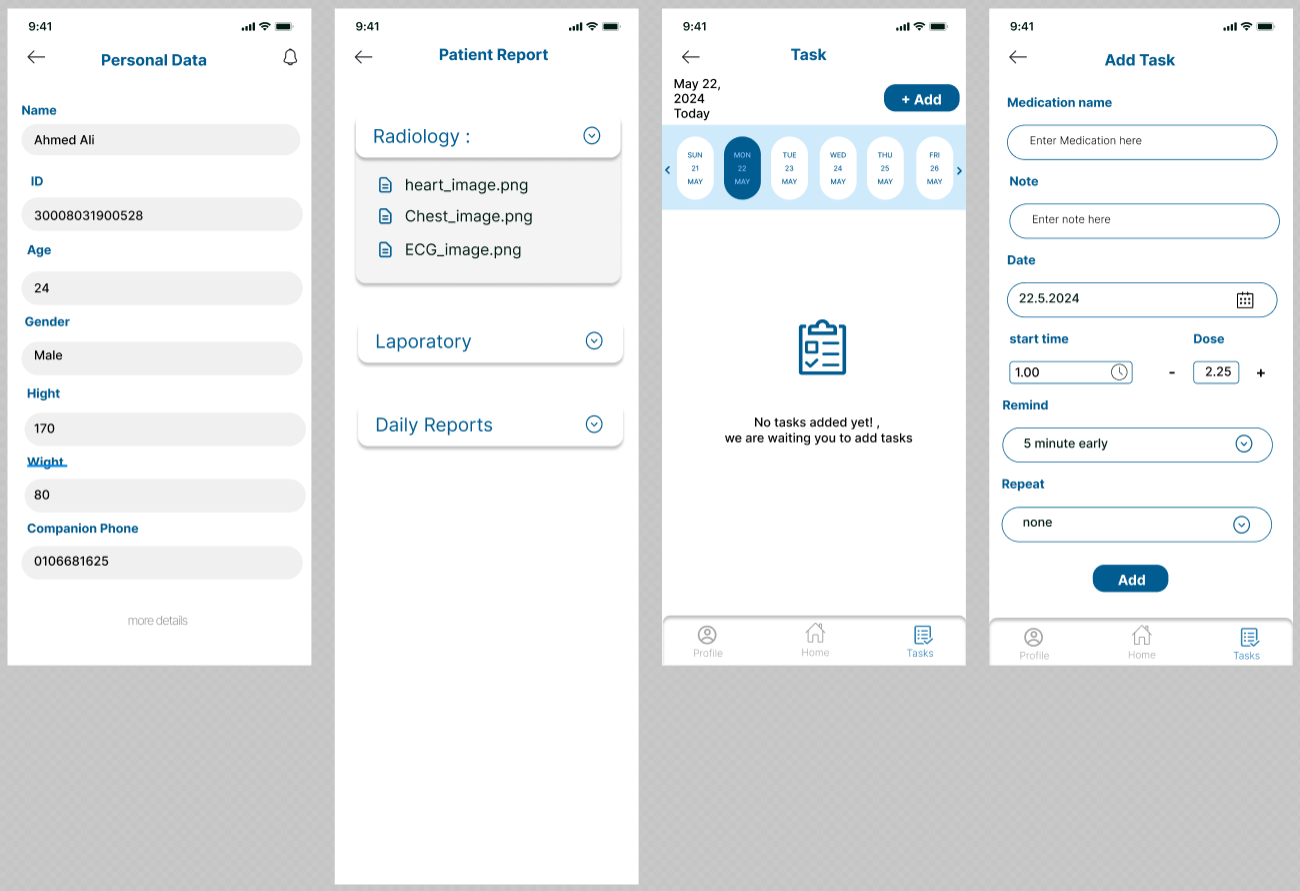


**4.4 Shift**



**4.5 Patient Screens**



# Chapter five

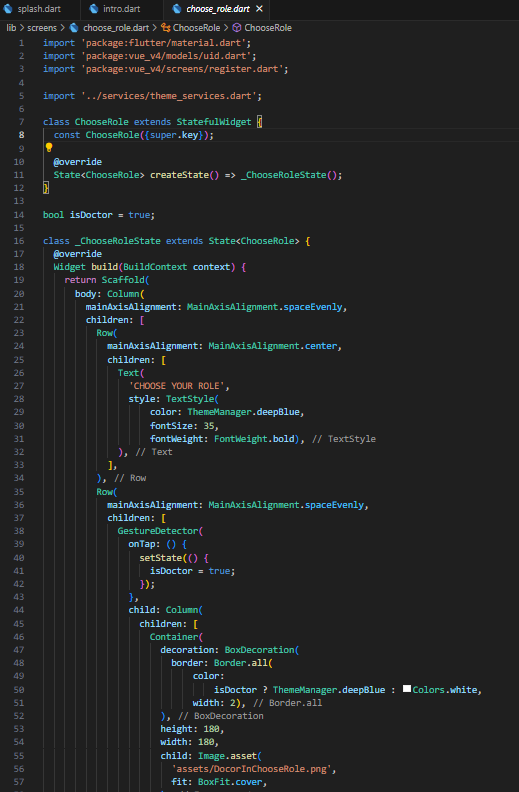
Implementation

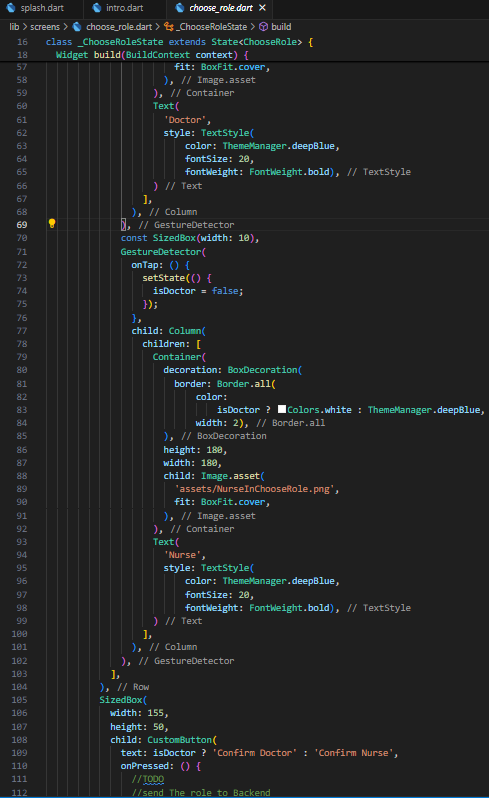
In this chapter, we provide an overview of what we implemented in the VU project using a combination of advanced technologies. Leveraging Flutter for mobile app development, Spring Boot for robust backend services, OpenCV for sophisticated computer vision capabilities, and Arduino IDE for seamless hardware integration, our implementation focuses on revolutionizing ICU management.



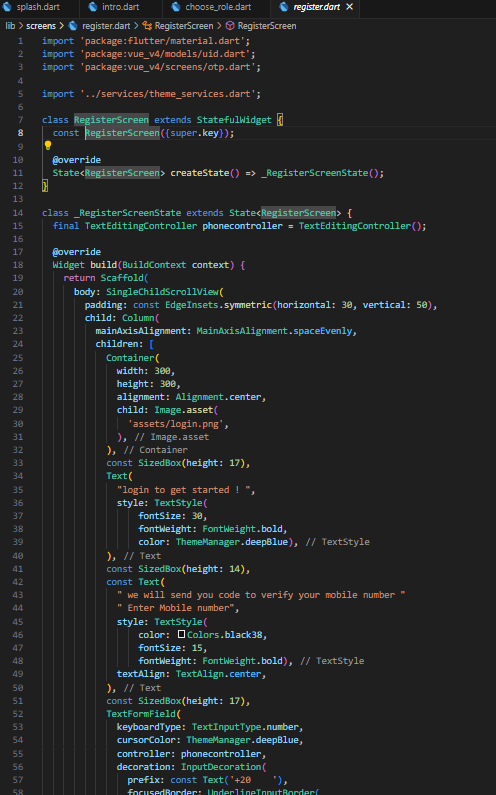
**5.1 Flutter Implementation**

**5.1.1 Choose Role**





**5.1.2 Login**



**5.1.3 OTP Verification**

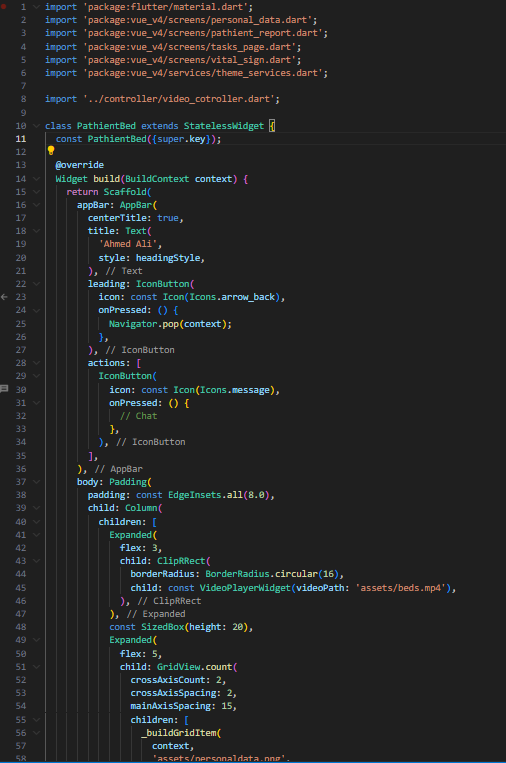


**5.1.4 Units**



**5.1.5 Beds**

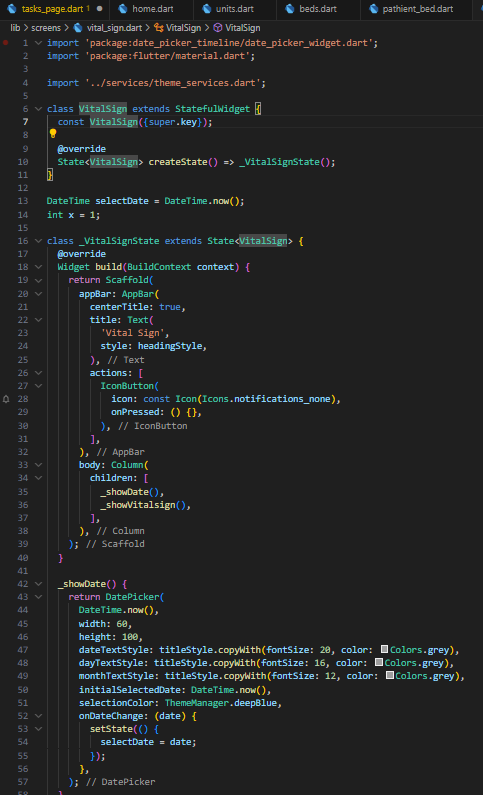


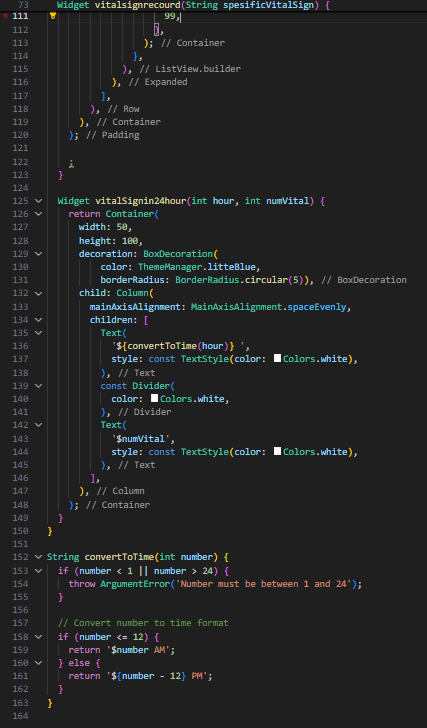


**5.1.6 Patient Personal Information**

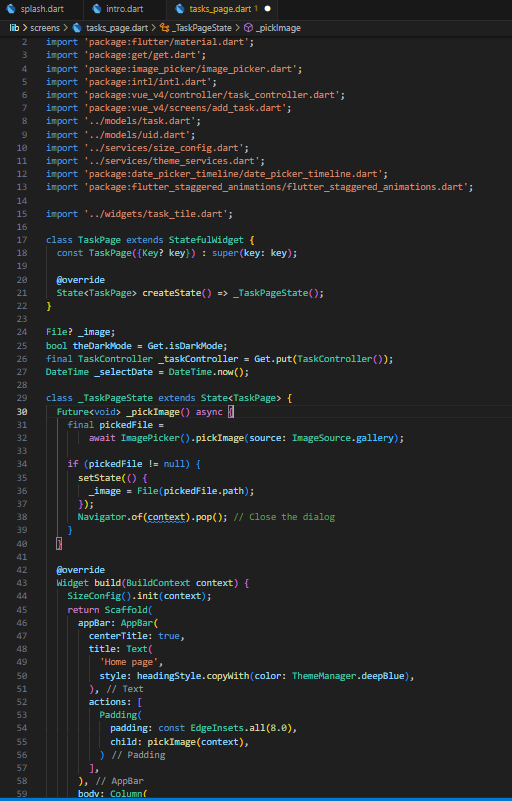


**5.1.7 Vital Signs Report**





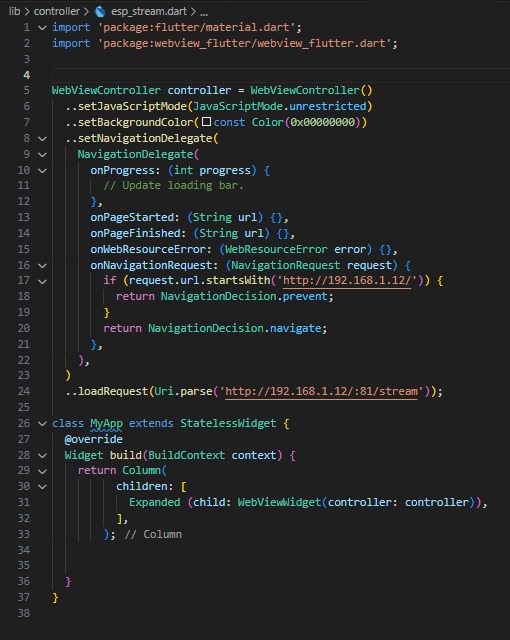
**5.1.8 Todolist**



**5.1.9 Add Task**



**5.1.10 Streaming Process**

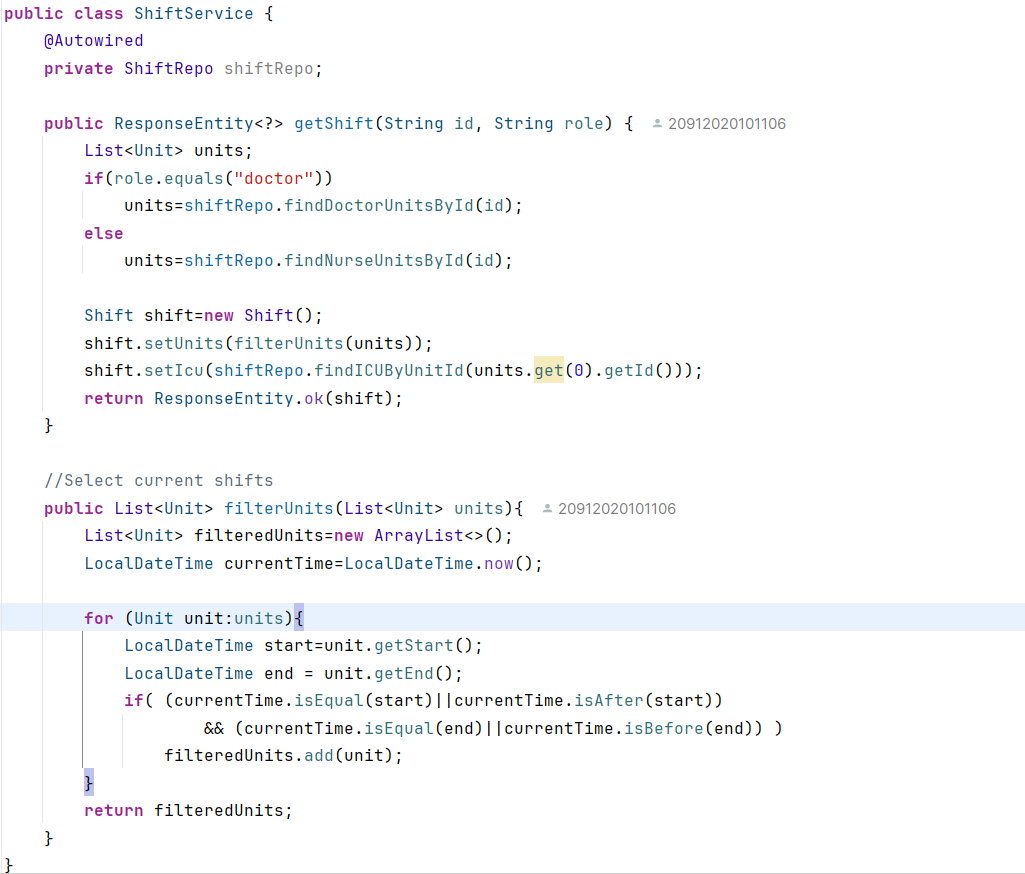


**5.2 Spring Boot Implementation**

**5.2.1 Login Service**

****

**5.2.2 Shift Service**

****

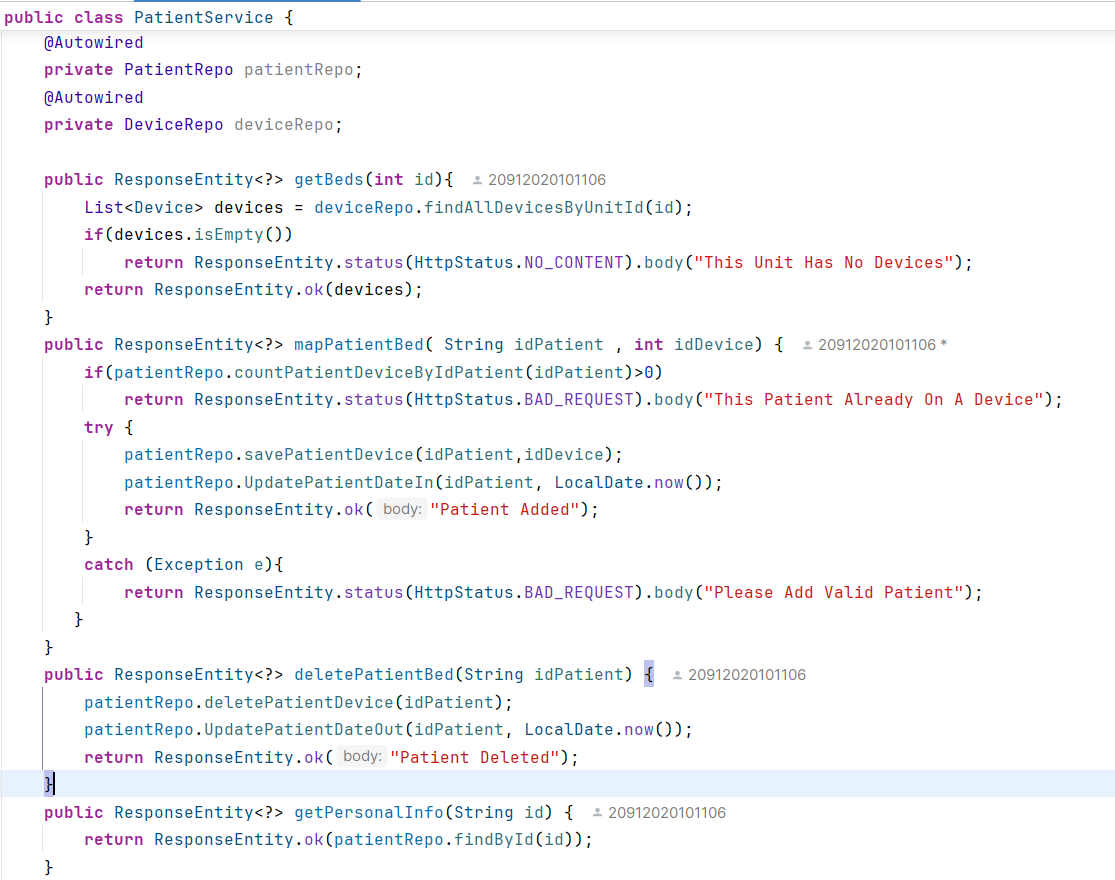
**5.2.3 Medical Reports Service**

****

**5.2.4 Vital Signs Report Service**

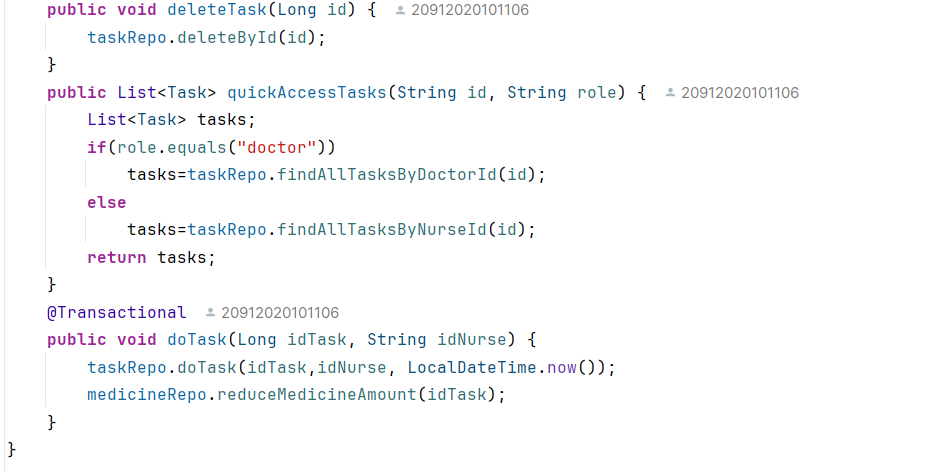
****

**5.2.5 Patient-Beds Service**

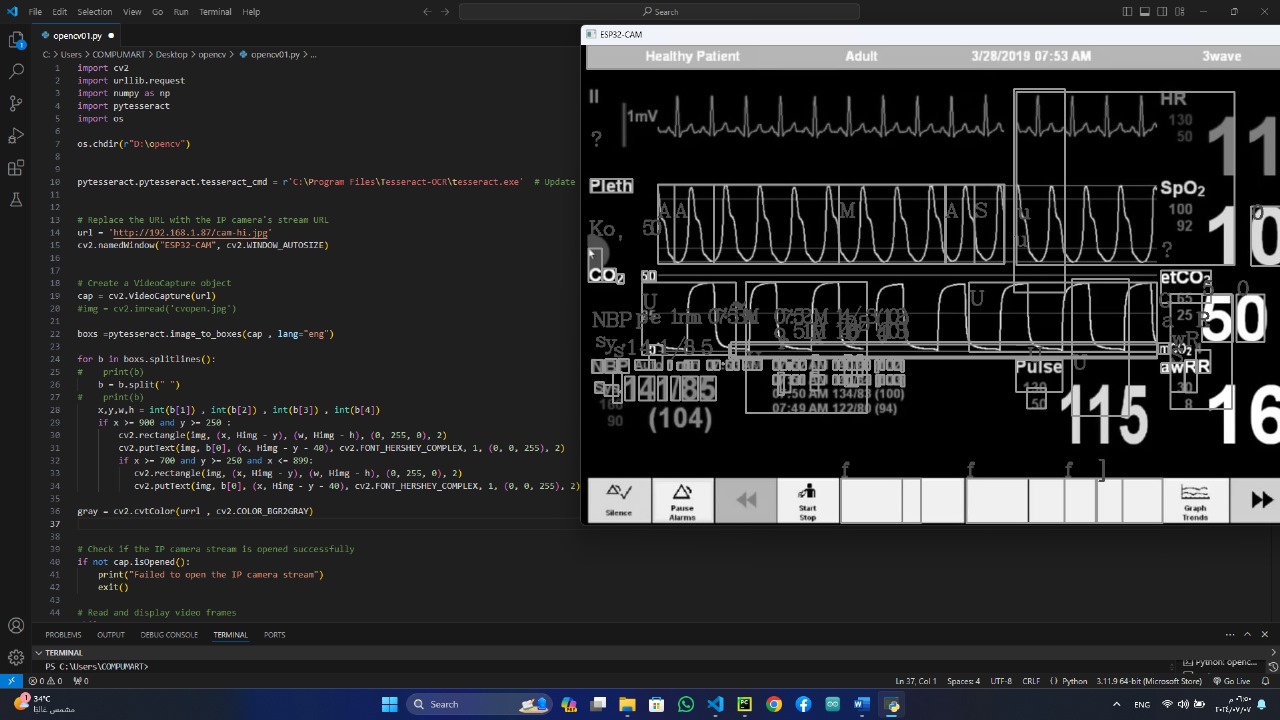
****

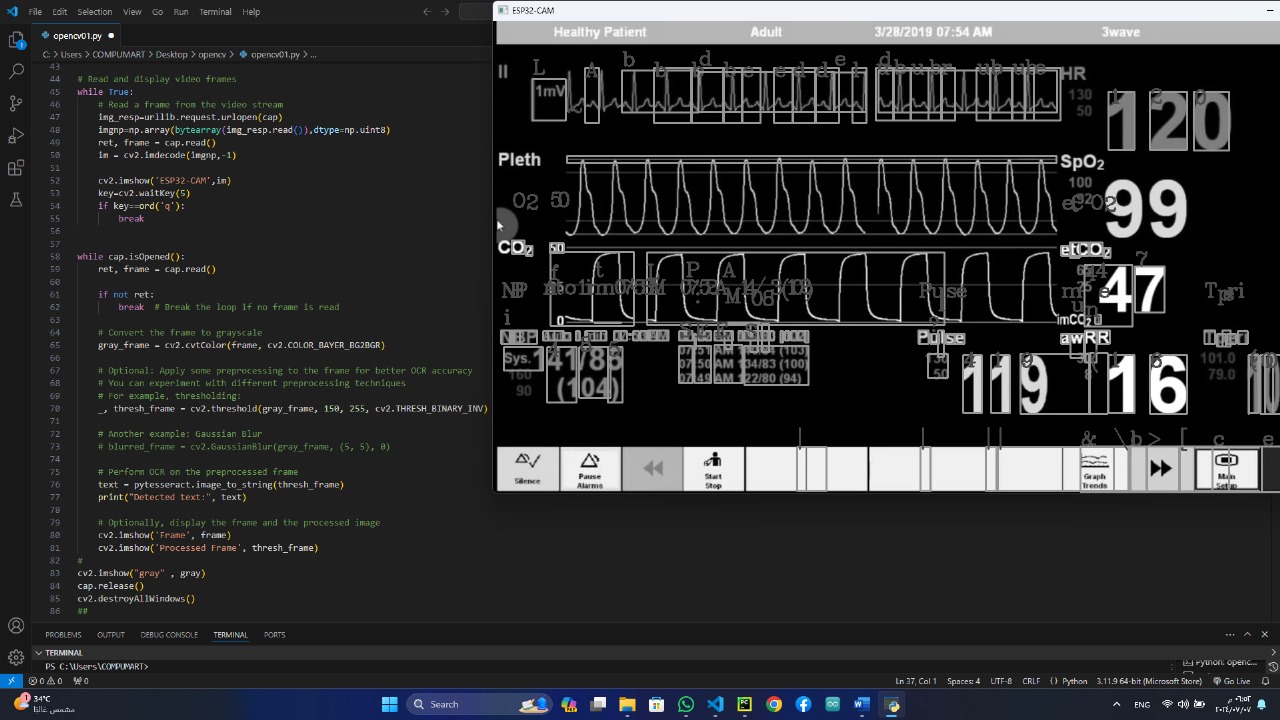
**5.2.6 TodoList Service**

****

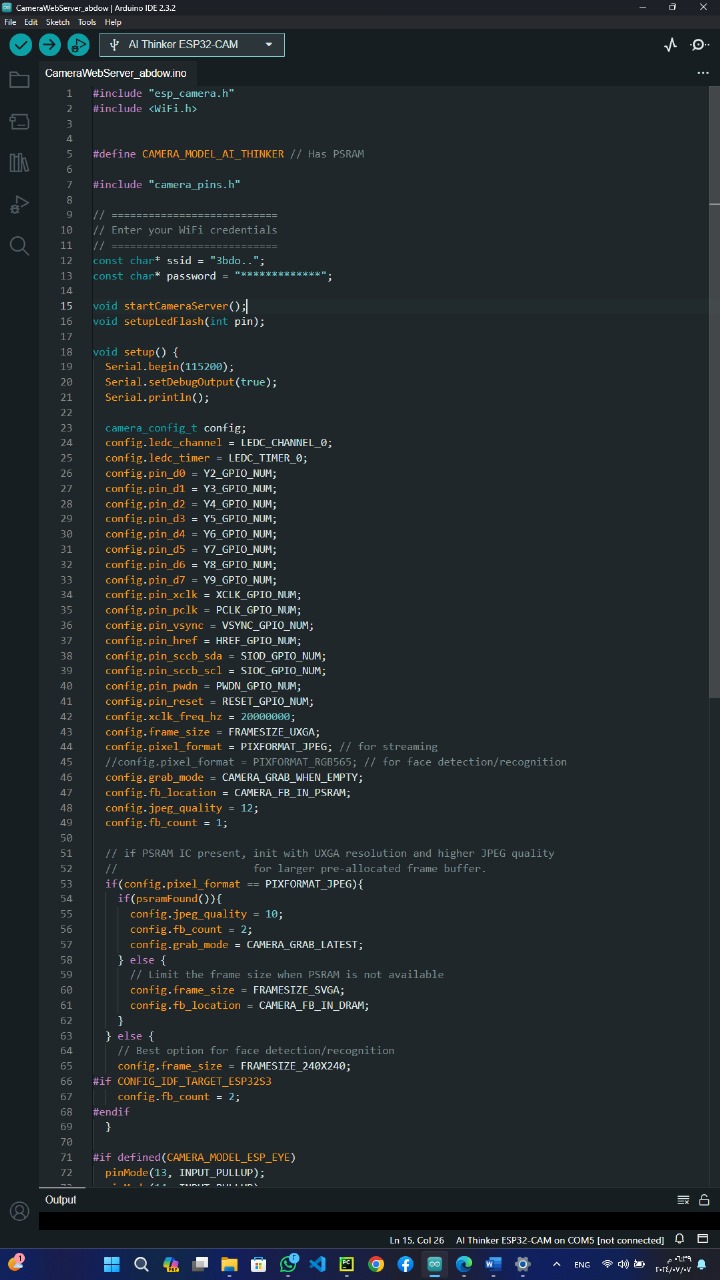
****

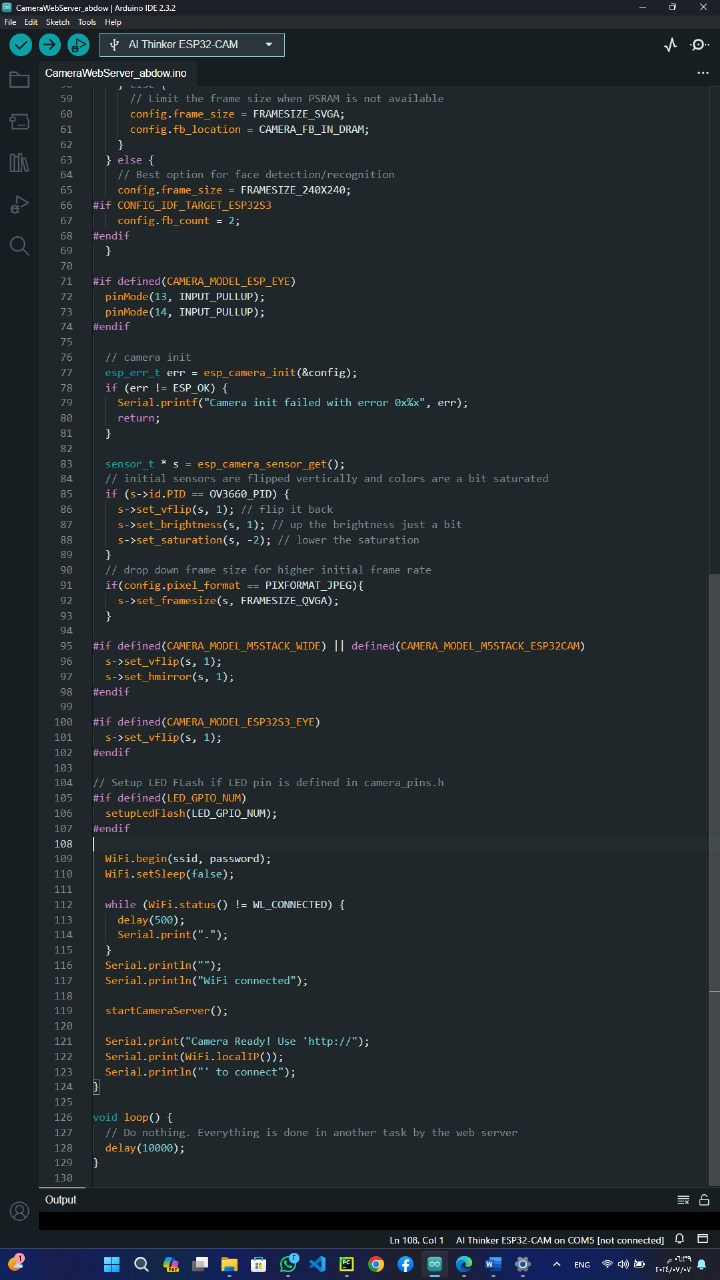
**5.3 Computer Vision Implementation**

**Open CV - Pytesseract**

****

**3.4 ESP-Cam Implementation**

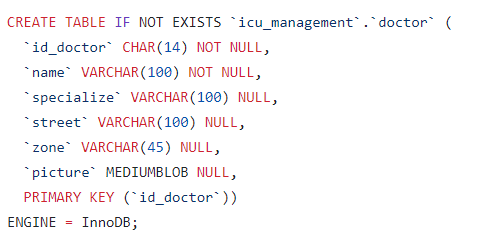
****

****

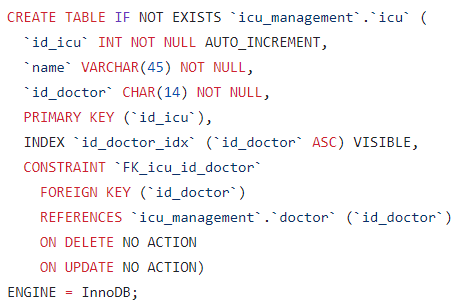
**5.5 Data Base Implementation**

**Entities:**

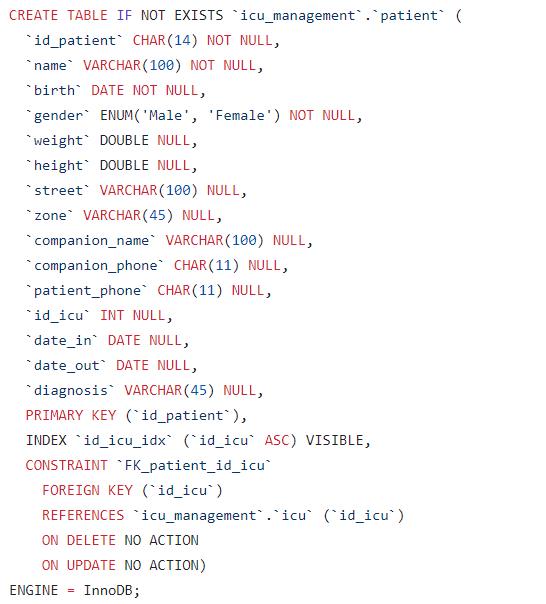
* **5.5.1** Doctor



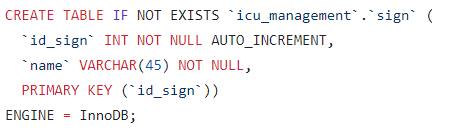
* **5.5.2** ICU



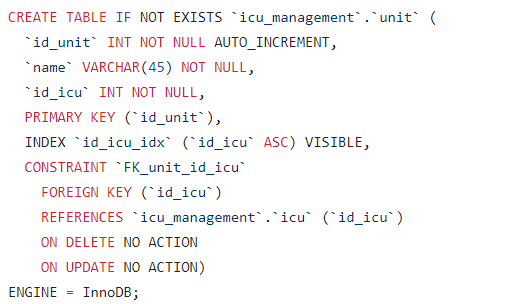
* **5.5.3** Patient



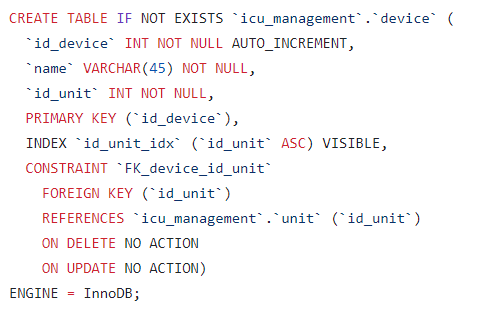
* **5.5.4** Sign



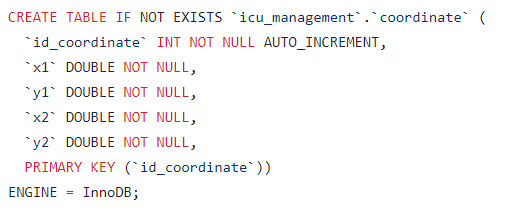
* **5.5.5** Unit



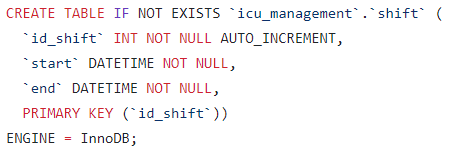
* **5.5.6** Device



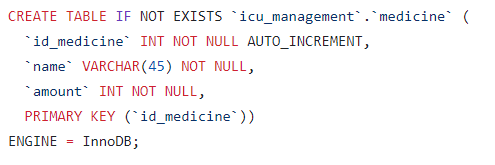
* **5.5.7** Coordinate



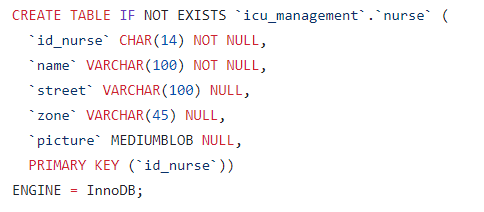
* **5.5.8** Shift



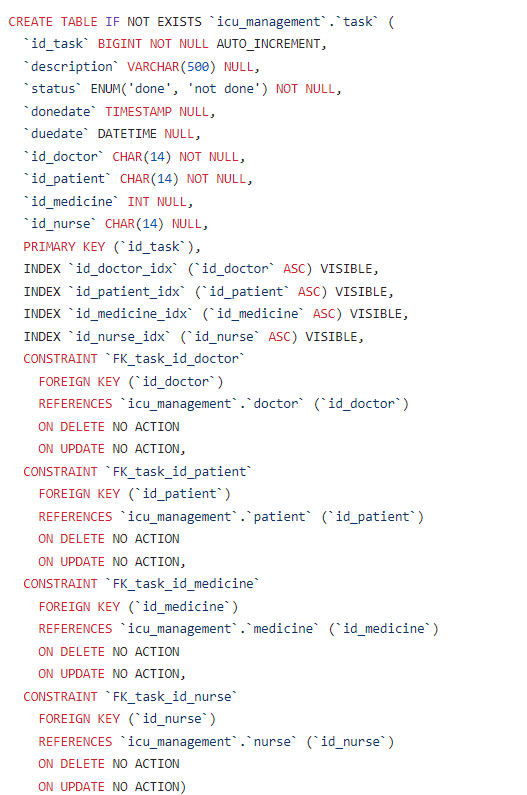
* **5.5.9** Medicine



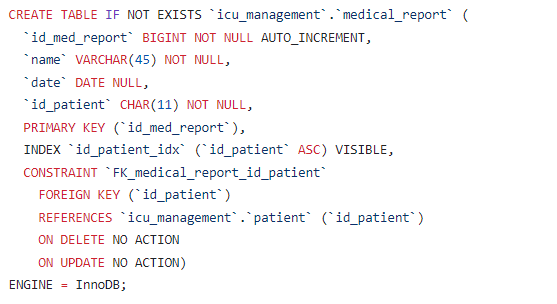
* **5.5.10** Nurse



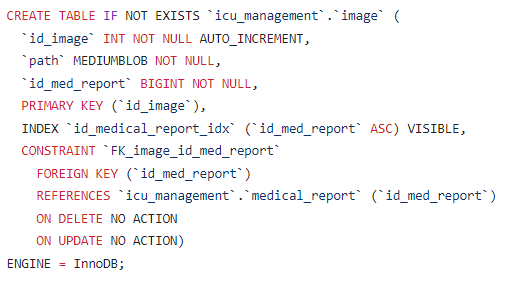
* **5.5.11** Task



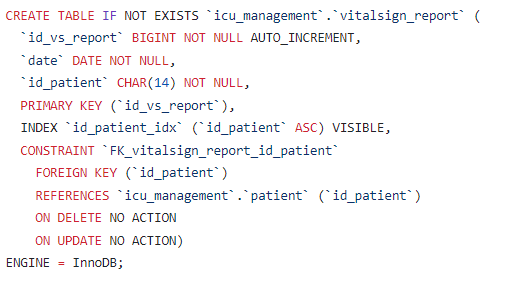
* **5.5.12** Report



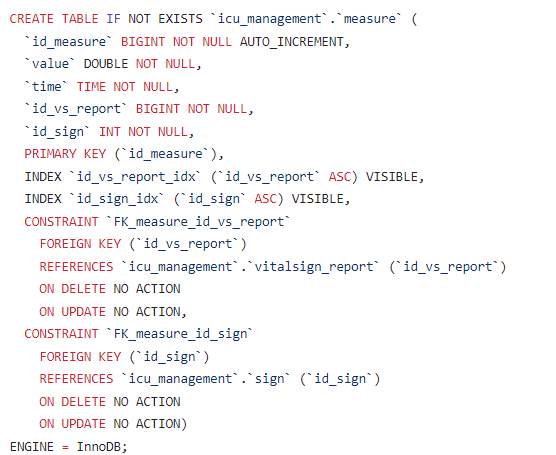
* **5.5.13** Image



* **5.5.14** Vital Signs Report



* **5.5.15** Measure



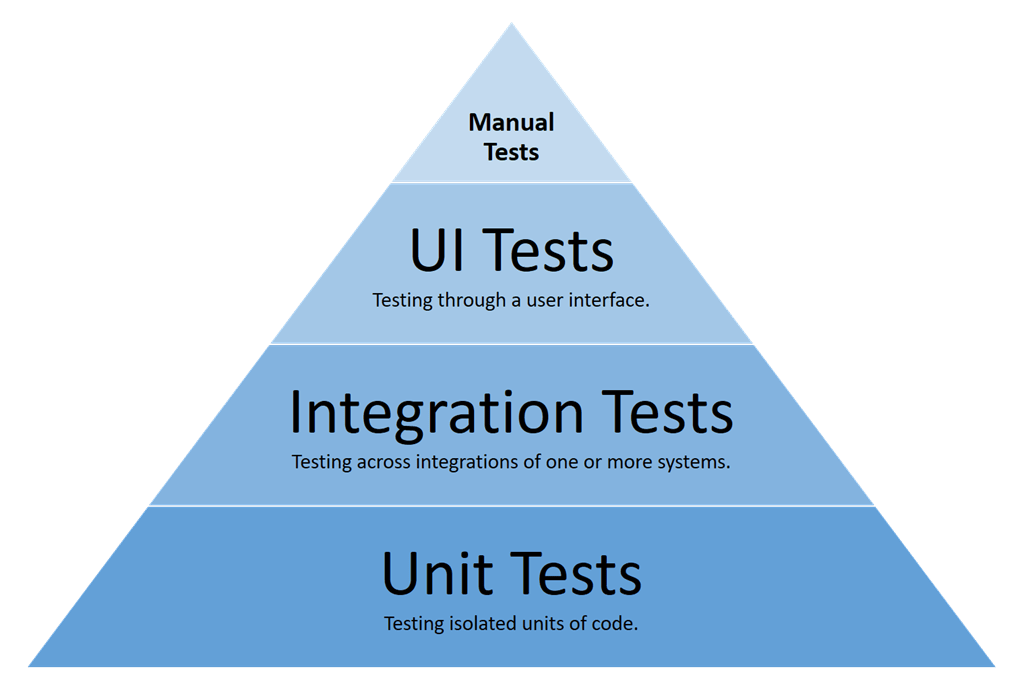
# Chapter six

Testing

In this chapter, we focus on the testing phase of the VU project, a critical step to ensure the reliability, functionality, and performance of our ICU management system. Testing is pivotal in identifying and rectifying any issues, ensuring that the application operates seamlessly under various conditions and meets the high standards required for healthcare environments.

窗体顶端

窗体底端



**6.1 User Interface (UI) Testing**

**6.1.1 - Objective**: Ensure that the Flutter-based mobile application provides a user-friendly and intuitive interface for healthcare providers.

**6.1.2 - Tools Used**: Flutter’s testing framework and manual testing.

**6.1.3 - Approach**:

* **Layout Verification**: Ensure all UI components are correctly positioned and rendered across different devices and screen sizes.
* **Navigation Flow**: Test the navigation between different screens to ensure smooth and logical transitions.
* **Usability Testing**: Conduct usability tests with healthcare professionals to gather feedback and make necessary adjustments.
* **Responsiveness**: Verify that the app responds promptly to user inputs and performs intended actions without lag.

**6.2 Integration Testing**

**6.2.1 - Objective**: Ensure that different components of the system work together seamlessly.

**6.2.2 - Tools Used**: Postman, Selenium, and custom integration test scripts.

**6.2.3 - Approach**:

* **Flutter and Spring Boot Integration**:
  + Verify data exchange between the Flutter app and Spring Boot backend through API calls.
  + Test user authentication, data retrieval, and updates.
* **Backend and Database (MySQL)**:
  + Ensure the backend correctly reads from and writes to the MySQL database.
  + Validate data consistency and integrity across different operations.
* **Computer Vision and Backend**:
  + Test the integration of the CV algorithm with the backend to ensure accurate and timely data processing.
  + Verify that extracted vital signs are correctly stored in the database and displayed in the app.
* **End-to-End Testing**:
  + Simulate real-world scenarios to validate the entire workflow, from logging in and monitoring patients to receiving alerts and updating records.

**6.3 Unit Testing**

**6.3.1 - Objective**: Validate the functionality of individual components and methods within the Flutter app, Spring Boot backend, and computer vision algorithms.

**6.3.2 - Tools Used**:

* **Flutter Unit Testing**: Dart’s built-in test package.
* **Spring Boot Testing**: JUnit and Mockito.
* **Computer Vision Testing**: Custom Python scripts for testing algorithm accuracy.

**6.3.3 - Approach**:

**Frontend (Flutter)**:

* Test individual widgets and functions to ensure they perform as expected.
* Validate form inputs, state management, and error handling.

**Backend (Spring Boot)**:

* Test API endpoints for correct data retrieval and processing.
* Mock external dependencies to isolate and test business logic.

**Computer Vision Algorithm**:

* Test the accuracy of vital sign extraction from video frames.
* Validate the algorithm's performance with various video qualities and conditions.

# Chapter seven

# Conclusion and

# Future Work

In this chapter, we focus on the conclusion and future work of the VU project, summarizing the progress and achievements thus far, and outlining potential directions for further development. The VU app has made significant strides in enhancing ICU management through improved patient monitoring, streamlined communication, and automated vital sign tracking. These advancements have the potential to transform ICU operations, reduce the workload on healthcare professionals, and improve patient outcomes.

#### 7.1 Conclusion

The VU app represents a significant step forward in ICU management, offering an innovative solution that enhances patient monitoring and healthcare delivery. Through the development process, we focused on improving communication between healthcare teams, automating vital sign monitoring, and providing real-time access to patient data. This app has the potential to streamline ICU operations, reduce the workload on healthcare professionals, and ultimately improve patient outcomes.Key achievements include:

* A comprehensive and user-friendly UI/UX design tailored for ICU environments.
* Robust backend infrastructure using Spring Boot and MySQL, ensuring reliable data management.
* Integration with hardware components like the ESP32 cam for remote monitoring.
* Implementation of advanced features such as OCR for extracting data from video streams.

#### 7.2 Future Work

While the VU app has achieved its initial objectives, there are several areas for future improvement and expansion:

**Enhanced Data Analytics**:

* Incorporate machine learning algorithms to predict patient deterioration and suggest proactive measures.
* Develop comprehensive data visualization tools to provide deeper insights into patient health trends.

**Integration with Electronic Health Records (EHR)**:

* Enable seamless integration with existing EHR systems to provide a holistic view of patient data.
* Ensure compliance with healthcare data standards and regulations (e.g., HL7, FHIR).

**Scalability and Performance Optimization**:

* Optimize the backend to handle a larger number of concurrent users and data points.
* Implement load balancing and database sharding techniques to improve scalability.

**User Feedback and Iterative Development**:

* Continuously gather feedback from healthcare professionals to identify pain points and areas for enhancement.
* Regularly update the app with new features and improvements based on user feedback.

**Advanced Security Measures**:

* Enhance security protocols to protect sensitive patient data from cyber threats.
* Implement multi-factor authentication (MFA) and end-to-end encryption.

**Cross-Platform Compatibility**:

* Expand the app’s compatibility to support more devices and operating systems, ensuring wider accessibility.
* Develop a web-based version of the app for use on desktops and laptops.

**Collaboration with Healthcare Institutions**:

* Partner with hospitals and research institutions to conduct pilot studies and validate the app’s effectiveness.
* Gather empirical data to support the app’s impact on ICU management and patient outcomes.

**Integration with IoT for Room Automation**:

* Utilize the ESP32 cam and other IoT devices to automate ICU room controls.
* Implement features for remote monitoring and control of room temperature, bed adjustments, lighting, and other environmental factors.
* Develop an intuitive interface within the app for healthcare professionals to manage these controls efficiently.

By addressing these areas, the VU app can continue to evolve and adapt to the ever-changing landscape of healthcare technology, ensuring it remains a valuable tool for ICU management in the future.

**8** Workspaces

* <https://github.com/heshammohamedaldidamony/VU-ICU-Management-App>
* <https://www.figma.com/board/ZQrooRtj2tLBjVQd0YZJbc/VUE-SDLC?node-id=0-1&t=mkKWM7DL2DmUfKcb-1>

**9** References

* **UI-UX :**

<https://youtu.be/1-S1vTkm1EU?si=QpOPnE7ZLswyJHt2&sfnsn=scwspwa>

* **Flutter :**

<https://pub.dev/>

* **SpringBoot :**

<https://spring.io/projects/spring-boot#learn>

* **CV Algorithm :**

<https://youtu.be/Ypu2OnVcvNM?si=cvKaX6jaybx2yVHQ>

* **ESP-Cam :**

<https://youtu.be/npJsmbFZiMg?si=_5s3xZrPrWHR85ys>