

**ALZHEIMER'S DISEASE PREDICTION WITH AI  
PERSONALIZED TREATMENT RECOMMENDATION  
SYSTEM**

**A MAJOR PROJECT REPORT**

*submitted by*

**AVRIL MARTIN (KH.EN.I5MCA22022)**

**CHANDANA VG (KH.EN.I5MCA22028)**

**GAGANA SREENIVASAN (KH.EN.I5MCA22034)**

**GAYATHRI GOPAKUMAR (KH.EN.I5MCA22036)**

**MEGHA MAHESH (KH.EN.I5MCA22056)**

**NIA V KRISHNA (KH.EN.I5MCA22072)**

*Under the supervision of*

**AISWARYA VIJAYAKUMAR**

**FACULTY ASSOCIATE**

**Department of Computer Science and IT**

*in partial fulfilment of the requirement of*

**AMRITA VISHWA VIDYAPEETHAM**

*for the award of the degree of*

**FIVE YEAR INTEGRATED MASTER OF COMPUTER APPLICATIONS**



**AMRITA VISHWA VIDYAPEETHAM, KOCHI  
CAMPUS**

**April 2025**



## **BONAFIDE CERTIFICATE**

This is to certify that the project report entitled **Alzheimer's Disease Prediction with AI Personalized Treatment Recommendation System** submitted by Avril Martin (KH.EN.I5MCA22022), Chandana VG (KH.EN.I5MCA22028), Gagana Sreenivasan(KH.EN.I5MCA22034),GayathriGopakumar (KH.EN.I5MCA22036), Megha Mahesh (KH.EN.I5MCA22056) and Nia V Krishna (KH.EN.I5MCA22072) in partial fulfilment of the requirements for the award of the **DEGREE OF FIVE YEAR INTEGRATED MASTER OF COMPUTER APPLICATIONS** is a Bonafide record of the work carried out under my guidance and supervision at School of Computing, Amrita Vishwa Vidyapeetham, Kochi Campus.

**Aiswarya Vijayakumar**

Project Advisor  
Faculty Associate  
Department of Computer Science  
School of Computing  
Amrita Vishwa Vidyapeetham  
Kochi Campus

**Dr. Sangeetha J**

Head of the Department  
Department of Computer Science  
School of Computing  
Amrita Vishwa Vidyapeetham  
Kochi Campus

The project was evaluated as on: \_\_\_\_\_

**Internal Examiner**

**External Examiner**

# DECLARATION

We affirm that the project work entitled “**Alzheimer’s Disease Prediction with AI Personalized Treatment Recommendation System**” being submitted in partial fulfilment for the award of the **DEGREE OF FIVE YEAR INTEGRATED MASTER OF COMPUTER APPLICATIONS** is the original work carried out by us. It has not formed the part of any other project work submitted for the award of any degree or diploma, either in this or any other University.

Place: Kochi

Date: 28 – 04 - 2025

Avril Martin- KH.EN.I5MCA22022

Chandana VG- KH.EN.I5MCA22028

Gagana Sreenivasan- KH.EN.I5MCA22034

Gayathri Gopakumar-KH.EN.I5MCA22036

Megha Mahesh- KH.EN.I5MCA22056

Nia V Krishna- KH.EN.I5MCA22072



**Syntax Soft-Tech India Pvt. Ltd**

**Corporate Office**

#16, 30th Cross  
4th T Block, Opp. Sagar Hospital  
Jayanagar, Bangalore -560 041, India  
Tel: +91-80-41380700

April 16, 2025

**To Whom So Ever It May Concern**

This is to certify that **AVRIL MARTIN** (KH.EN.I5MCA22022), **CHANDANA VG** (KH.EN.I5MCA22028), **GAGANA SREENIVASAN** (KH.EN.I5MCA22034), **GAYATHRI GOPAKUMAR** (KH.EN.I5MCA22036), **MEGHA MAHESH** (KH.EN.I5MCA22056) and **NIA V KRISHNA** (KH.EN.I5MCA22072), 6th semester **DEGREE OF INTEGRATED MASTER OF COMPUTER APPLICATIONS**, Students of **AMRITA VISHWA VIDYAPEETHAM, KOCHI** have successfully completed a project titled **"Alzheimer's Disease Prediction with AI Personalized Treatment Recommendation System"** from our organization for a duration of 4 months (January 2025 to April 2025)

The project was done in Python, Django using SQLite3 database and was successfully completed for **SYNTAX SOFT-TECH PVT LTD**. All necessary details were provided from our side for the establishment of this project. The work has been completed to the best of our satisfaction.

Thanking you,

For **Syntax Soft-Tech India Pvt Ltd**



**Ramaswamy - Manager**  
**Kerala Operations**

# **DEDICATION**

To

My parents, all my teachers and the eternal God.  
Thank you for believing in me and encouraging me throughout.

## ACKNOWLEDGEMENT

A venture can't be completely by itself. We take this opportunity to gratefully acknowledge various people who acted as guides along the way.

The success of any work requires the blessings of the Lord Almighty. We thank our God for aiding us in our travel to success.

We express our thankfulness to **Dr. U. Krishnakumar**, Director, Amrita Vishwa Vidyapeetham, Kochi Campus for giving us an opportunity to do our major project.

We would like to express our deep gratitude to **Dr. Sangeetha J**, Head of the Department, Computer Science and IT, School of Computing, Amrita Vishwa Vidyapeetham, Kochi Campus for her valuable guidance and for the support she rendered to us.

We would also like to thank our guide **Aiswarya Vijayakumar**, Faculty Associate, who revised our ideas and helped us to successfully complete the major project.

We would also like to express our gratitude to the staff of the ICTS Department, School of Computing, Amrita Vishwa Vidyapeetham, Kochi Campus for the technical support they gave.

This Thanksgiving cannot be complete without mentioning our friends and parents who gave us the mental strength that we almost lost in between the journey.

*Avril Martin*

*Chandana VG*

*Gagana Sreenivasan*

*Gayathri Gopakumar*

*Megha Mahesh*

*Nia V Krishna*

## TABLE OF CONTENTS

|                                       | Page No |
|---------------------------------------|---------|
| 1.0 Introduction                      | 1       |
| 1.1 About the System                  | 1       |
| 2.0 Need for the System               | 2       |
| 3.0 Background Study                  | 3       |
| 3.1 Existing System                   | 3       |
| 3.2 Drawbacks                         | 3-4     |
| 3.3 Proposed System                   | 4-5     |
| 4.0 Problem Formulation               | 5       |
| 4.1 Main Objectives                   | 5       |
| 4.2 Specific Objectives               | 5-6     |
| 4.3 Methodology                       | 6       |
| 4.4 Platform Selection                | 7       |
| 5.0 System Analysis & Design          | 7       |
| 5.1 System Analysis                   | 7-8     |
| 5.2 Feasibility Analysis              | 8       |
| 5.2.1 Technical Analysis              | 8       |
| 5.2.2 Economical Analysis             | 9       |
| 5.2.3 Performance Analysis            | 10      |
| 6.0 System Design                     | 11      |
| 6.1 Input Design                      | 10-12   |
| 6.2 Output Design                     | 12-13   |
| 6.3 Architecture Design               | 14      |
| 6.3.1 Flow Diagrams                   | 15-18   |
| 6.3.2 ER Diagram                      | 19      |
| 6.3.3 Table Design                    | 20-25   |
| 7.0 System Testing and Implementation | 25      |
| 7.1 System Testing                    | 25-26   |
| 7.2 Maintenance                       | 26      |
| 8.0 Conclusion                        | 26-27   |
| 9.0 Scope for Further Development     | 27      |

|                   |         |
|-------------------|---------|
| 10.0 Bibliography | 27      |
| 11.0 Appendix     | 28      |
| 11.1 Screen Shots | 28 - 35 |
| 11.2 Sample Code  | 36 – 97 |



## **1.0 INTRODUCTION**

### **1.1 About the System**

The proposed system is an integrated, web-based platform that leverages deep learning technologies—specifically Convolutional Neural Networks (CNNs) and Deep Neural Networks (DNNs)—to detect, predict, and manage Alzheimer’s Disease (AD). By analyzing brain imaging data such as MRI scans, the system enables early and accurate diagnosis, which is essential for timely medical intervention and slowing the progression of the disease.

Beyond detection, the platform includes a powerful prediction feature that assesses the risk and stage of Alzheimer’s. This allows patients and clinicians to take proactive steps in managing the disease before it worsens. The system also offers personalized medicine recommendations tailored to each patient's medical history and current condition, ensuring more targeted and effective treatment plans.

To improve accessibility and convenience, the platform allows users to schedule online consultations with neurologists and other healthcare professionals. This is particularly beneficial for patients who have mobility challenges or live in remote locations. In addition, the system provides secure access to medical reports, making it easy for patients and doctors to view, download, and share diagnostic information as needed.

The deep learning models are trained on a high-quality dataset provided by Syntax Soft – Tech India Pvt. Ltd., which includes well-curated MRI scans and relevant clinical data. This ensures greater accuracy in predictions and reduces the chances of misdiagnosis.

By combining advanced AI with practical healthcare features, the platform serves as a comprehensive solution for early Alzheimer’s detection, personalized care, and streamlined medical support—ultimately improving the quality of life for patients and aiding healthcare providers in delivering better outcomes.

## **2.0 NEED FOR THE SYSTEM**

Alzheimer's disease (AD) is a progressive neurological disorder that leads to cognitive decline, affecting memory, decision-making, and overall brain function. As the disease advances, individuals experience increasing difficulty in performing daily activities, ultimately requiring full-time care. Early diagnosis is crucial in slowing disease progression, enabling timely medical intervention, and improving patients' quality of life. However, current diagnostic methods face significant challenges in providing accurate and timely detection, especially in the early stages of the disease.

Traditional diagnostic approaches, such as cognitive tests, clinical assessments, and manual interpretation of MRI scans, often suffer from subjectivity, time-consuming procedures, and inconsistent accuracy in detecting early-stage Alzheimer's. Many cases remain undiagnosed until severe cognitive impairment sets in, limiting the effectiveness of available treatments. Additionally, the lack of standardized and automated screening tools contributes to delays in diagnosis, leading to missed opportunities for early intervention.

Beyond diagnostic challenges, patients and caregivers often struggle with fragmented healthcare services. Difficulties in accessing specialists, managing medications, and retrieving medical reports create additional burdens, leading to delays in treatment and care coordination. The absence of integrated and easily accessible platforms for Alzheimer's management further complicates the process for both patients and healthcare professionals.

Given these limitations, there is a growing need for advanced, technology-driven solutions that can enhance early detection, improve diagnostic accuracy, and streamline patient care. A system that leverages modern advancements in medical imaging and digital healthcare services can help bridge the gap, ensuring faster, more precise, and accessible Alzheimer's detection and management.

## 3.0 BACKGROUND STUDY

### 3.1 Existing System

Several advanced systems leverage AI and deep learning for Alzheimer's detection, analyzing neuroimaging and medical data. Notable systems include ADNI, Deep Brain Network, and IBM Watson for Health, but they face limitations in real-time accessibility and comprehensive solutions.

#### 1. Alzheimer's Disease Neuroimaging Initiative (ADNI)

ADNI uses machine learning, including SVMs and CNNs, to analyze MRI and PET scan data, identifying Alzheimer's-related brain changes.

Limitation: Primarily research-focused, lacks real-time clinical application and early intervention prediction.

#### 2. Deep Brain Network

Uses deep learning models like CNNs and GCNs to extract patterns from MRI scans, offering improved accuracy.

Limitation: Does not include disease prediction or continuous patient care features.

#### 3. IBM Watson for Health

Applies AI techniques such as NLP, DNNs, and Reinforcement Learning to analyze medical records and aid diagnosis.

Limitation: Lacks a user-friendly interface and essential features like online consultations or real-time predictive analytics.

### 3.2 Drawbacks

- **Lack of Real-Time and Automated Predictions:** Most systems do not provide instant, automated diagnosis, delaying early intervention.
- **Absence of Continuous Monitoring:** There is no mechanism for follow-up tracking of patients over time to observe disease progression.
- **Limited Accessibility for Patients and Healthcare Providers:** The platforms are primarily research-focused or complex for direct clinical use.

- **Fragmented Decision-Making and Healthcare Services:** No integrated system combines diagnosis, patient care, medication tracking, and consultations in a single platform.

### 3.3 Proposed System

The proposed system is designed to revolutionize Alzheimer's disease detection by leveraging advanced deep learning algorithms to enhance the way Alzheimer's is diagnosed and managed. Traditional diagnostic methods, such as cognitive assessments and manual interpretation of brain scans, often suffer from delays, subjectivity, and inconsistencies. This system aims to address these limitations by using AI-powered analysis to improve diagnostic accuracy and facilitate early detection, which are essential for timely intervention and improved patient outcomes. Central to the system is a comprehensive dataset of brain imaging, including MRI scans from both healthy individuals and those diagnosed with Alzheimer's disease. It incorporates a diverse range of data that captures various stages of the disease: from healthy individuals (control group) to early-stage Alzheimer's patients, where subtle changes in brain structure begin to appear, and moderate to severe Alzheimer's patients, where significant brain atrophy and cognitive decline have occurred. By using MRI scans across these different stages, the system can recognize differentiated patterns of brain abnormalities, enabling early detection that is critical for timely intervention. Deep learning models, such as Convolutional Neural Networks (CNNs) and Deep Neural Networks (DNNs), are employed to analyse these scans, extracting complex patterns and features that are crucial for identifying early signs of Alzheimer's.

#### Merits of the Proposed System

- **Comprehensive Data Utilization:** Incorporates MRI scans from both healthy individuals and Alzheimer's patients for better training.
- **Efficient Data Processing:** Uses grayscale conversion and preprocessing for simplified and enhanced feature extraction.
- **Advanced Feature Extraction:** Leverages CNNs and DNNs for accurate detection.
- **Optimized Performance:** Potential integration of transfer learning to improve accuracy.

- **Reliable Evaluation:** Assesses performance using key metrics like accuracy and specificity.
- **Ethical and Medical Collaboration:** Developed in coordination with healthcare professionals for practical implementation.
- **User-Friendly Interface:** Designed for clinicians to ensure ease of use and accessibility in medical settings.

## 4.0 PROBLEM FORMULATION

### 4.1 Main Objectives

The main objectives of the proposed Alzheimer's Detection System are to enhance early diagnosis, improve patient care, and enable continuous monitoring throughout the progression of the disease. By leveraging deep learning algorithms to analyse brain imaging data (using MRI scans), the system aims to provide early detection of Alzheimer's, facilitating timely interventions and improving patient outcomes. It also focuses on offering personalized treatment recommendations based on the patient's stage of the disease, helping healthcare providers make informed decisions. The system's continuous monitoring feature will allow healthcare professionals to track the disease's progression, ensuring that care plans can be adjusted as needed. Additionally, the system will be designed with a user-friendly interface, making it easy for healthcare providers to seamlessly integrate into their workflow and effectively manage Alzheimer's disease.

### 4.2 Specific Objectives

The specific objectives of the Alzheimer's Detection System are to enhance early detection, provide personalized care, and improve disease management. The system aims to use deep learning algorithms to analyse brain imaging data, using MRI scans, to detect early signs of Alzheimer's disease, enabling timely and accurate diagnosis. It focuses on providing personalized treatment recommendations, helping healthcare providers make informed decisions for targeted and effective care. Additionally, the system continuously monitors changes in brain structure and cognition, ensuring that care plans can be adjusted as needed throughout the disease's progression. The user-friendly interface ensures ease of use for

clinicians when reviewing patient data and making treatment decisions. These objectives collectively work to improve the early diagnosis, management, and ongoing care of Alzheimer's patients.

### 4.3 Methodology

The methodology for developing the Alzheimer's Detection System follows a structured approach involving both hardware and software components to ensure a smooth, efficient, and effective operation.

Hardware requirements:

- **Operating System:** The system is designed to run on Windows 10, providing a stable platform for development and deployment.
- **Processor:** The system requires a Core i5 processor, offering the necessary computational power to handle deep learning algorithms and large datasets.
- **RAM:** A minimum of 8GB of RAM is required to ensure smooth processing of data and uninterrupted operation, especially when analysing brain imaging data.
- **Storage:** A 100 GB ROM is needed for storing MRI scans, system files, and other data essential for the application.

Software requirements:

- **Frontend:** The user interface is developed using HTML, CSS, and JavaScript to create a responsive, interactive, and accessible platform for healthcare professionals.
- **Backend:** The backend of the system is powered by Python and Django, providing a robust framework for handling deep learning models and processing large datasets.
- **Database:** **SQLite3** is used for managing and storing patient data, allowing efficient querying and data management without complex configurations.

## 4.4 Platform Selection

The proposed system is designed as a web-based application to ensure accessibility, scalability, and ease of use for healthcare professionals and patients. A web-based approach allows users to access the system from any location, eliminating the need for specialized hardware and enabling remote diagnosis and monitoring.

To ensure robustness and efficiency, the system is developed using modern frameworks and technologies:

- **Frontend:** The user interface is built using HTML, CSS, and JavaScript, providing a responsive and intuitive design for healthcare professionals and patients.
- **Backend:** The backend is powered by Python and Django, offering a secure and scalable framework to process MRI scan data and manage user interactions.
- **Database:** SQLite3 is used as the primary database for storing patient records, diagnostic reports, and medical history, ensuring efficient data management.

## 5.0 SYSTEM ANALYSIS AND DESIGN

### 5.1 System Analysis

The proposed system uses deep learning and AI to automate Alzheimer's detection from MRI scans. It provides a web-based platform for medical professionals to upload MRI images and receive accurate diagnostic results.

#### **Aims To:**

- Reduce diagnostic time
- Enhance accuracy
- Store and manage records securely

#### **Functional Requirements:**

- User Management: Authentication and authorization

- MRI Image Upload: Secure upload
- Deep Learning Analysis: CNN-based detection
- Diagnosis Reports: Auto-generated results
- Patient Record Management: Store past reports

#### **Nonfunctional Requirements:**

- Scalability: Supports multiple users
- Usability: User-friendly interface
- Performance: Fast processing and response time

## **5.2 Feasibility Analysis**

### **5.2.1 Technical Analysis**

Hardware requirements:

- **Operating System:** The system is designed to run on Windows 10, providing a stable platform for development and deployment.
- **Processor:** The system requires a Core i5 processor, offering the necessary computational power to handle deep learning algorithms and large datasets.
- **RAM:** A minimum of 8GB of RAM is required to ensure smooth processing of data and uninterrupted operation, especially when analysing brain imaging data.
- **Storage:** A 100 GB ROM is needed for storing MRI scans, system files, and other data essential for the application.

Software requirements:

- **Frontend:** The user interface is developed using HTML, CSS, and JavaScript to create a responsive, interactive, and accessible platform for healthcare professionals.
- **Backend:** The backend of the system is powered by Python and Django, providing a robust framework for handling deep learning models and processing large datasets.
- **Database:** SQLite3 is used for managing and storing patient data, allowing efficient querying and data management without complex configurations.



## **5.2.2 Economical Analysis**

Economic analysis assesses the cost-effectiveness of the Alzheimer's Detection System, ensuring that development and maintenance costs are justified by its benefits in improving diagnosis and patient care.

### **1. Deployment Costs**

- Involves software setup, system integration, and staff training.
- Minimal physical infrastructure reduces implementation costs.

### **2. Reduced Diagnosis Costs**

- Automates MRI analysis, lowering labor and repeat test costs.
- Speeds up diagnosis, reducing reliance on manual assessments.

### **3. Improved Efficiency in Healthcare**

- Enables early intervention, lowering long-term care expenses.
- Supports faster diagnosis and increases patient handling capacity.

### **4. Scalability & Accessibility**

- Cloud-based system reduces hardware costs.
- Accessible to smaller clinics and remote areas.
- Enables real-time access to reports.

### **5. Long-term ROI**

- Fewer misdiagnoses and better patient outcomes.
- Reduces hospital admissions and resource strain.
- Supports research and may attract funding or insurance integration.

### 5.2.3 Performance Analysis

Performance analysis assesses the efficiency, speed, and reliability of the Alzheimer's Detection System to ensure it meets healthcare and technological standards.

#### 1. Accuracy & Reliability

- Assesses how consistently the system delivers clinically valid results by comparing with expert diagnoses and testing with varied datasets.

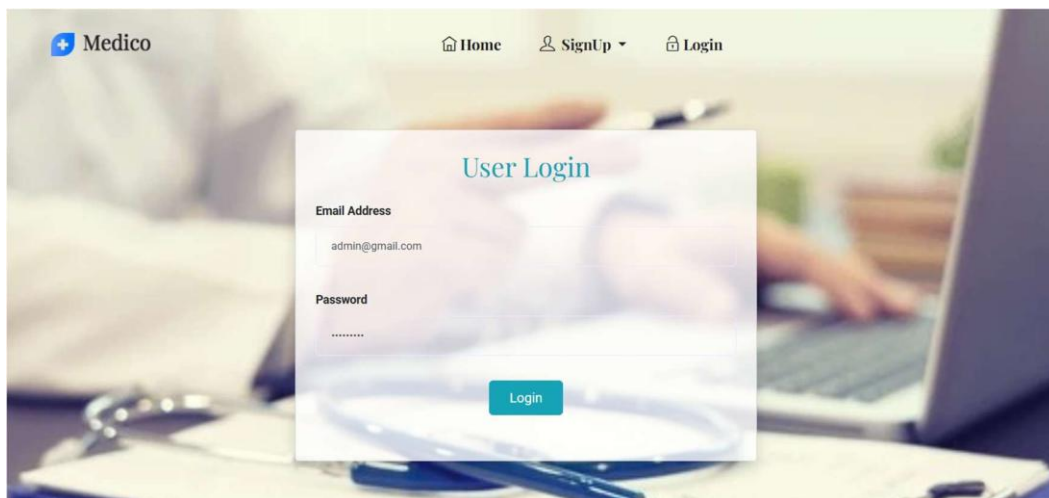
#### 2. Scalability

- Tests system performance under increased data and user load to confirm its stability and future-readiness.

## 6.0 SYSTEM DESIGN

### 6.1 Input Design

#### Admin Panel



## Hospital Reception

The screenshot shows the Medico website interface for Hospital Reception. The background is a blurred image of a doctor's desk with a stethoscope and a laptop. The website header includes the Medico logo (a blue square with a white plus sign) and the text "Medico". To the right of the logo are three navigation links: "Home" with a house icon, "SignUp" with a person icon and a dropdown arrow, and "Login" with a lock icon. In the center of the page is a white "User Login" form. The form has a title "User Login" in blue. It contains two input fields: "Email Address" with the text "emer@gmail.com" and "Password" with masked characters ".....". Below the password field is a teal "Login" button.

Medico

Home SignUp Login

### User Login

Email Address  
emer@gmail.com

Password  
.....

Login

## Doctor

This screenshot is identical to the one above, showing the Medico website interface for Hospital Reception. The background is a blurred image of a doctor's desk with a stethoscope and a laptop. The website header includes the Medico logo (a blue square with a white plus sign) and the text "Medico". To the right of the logo are three navigation links: "Home" with a house icon, "SignUp" with a person icon and a dropdown arrow, and "Login" with a lock icon. In the center of the page is a white "User Login" form. The form has a title "User Login" in blue. It contains two input fields: "Email Address" with the text "ema@gmail.com" and "Password" with masked characters ".....". Below the password field is a teal "Login" button.

Medico

Home SignUp Login

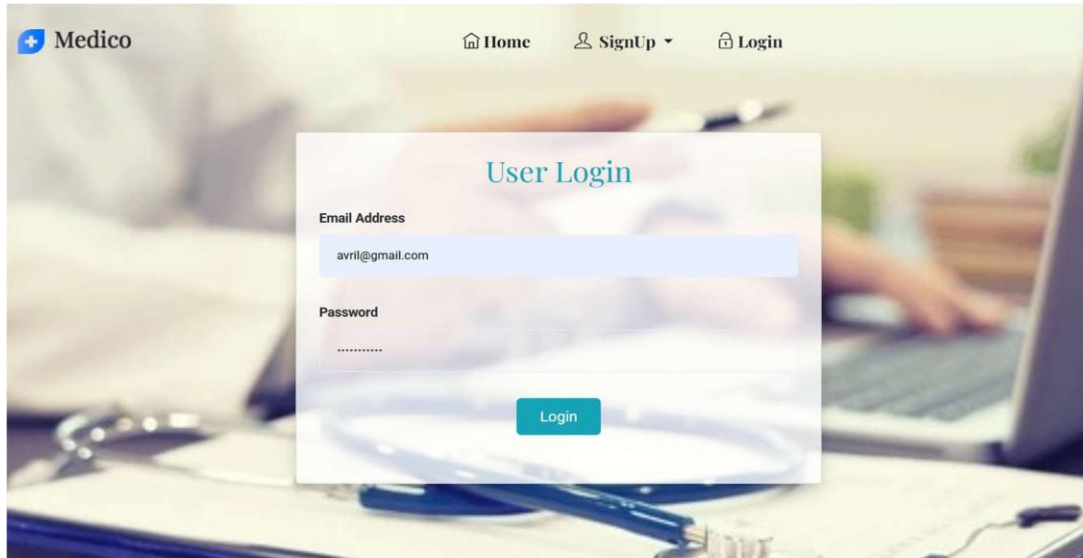
### User Login

Email Address  
ema@gmail.com

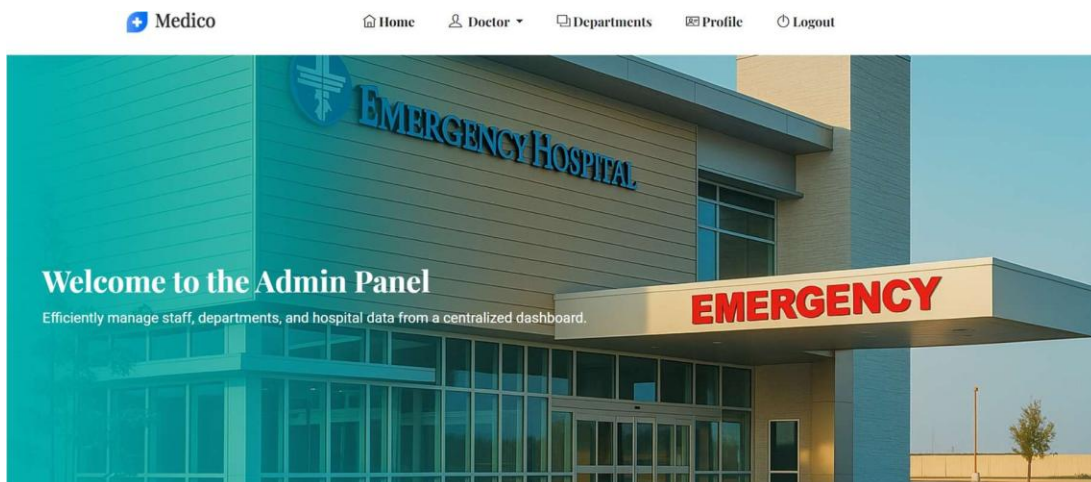
Password  
.....

Login

## Patient



## 6.2 Output Design





## Welcome to the Hospital Reception

Manage your appointments, view prescriptions, and access personalized care plans.



## Welcome, Dr. Ema j

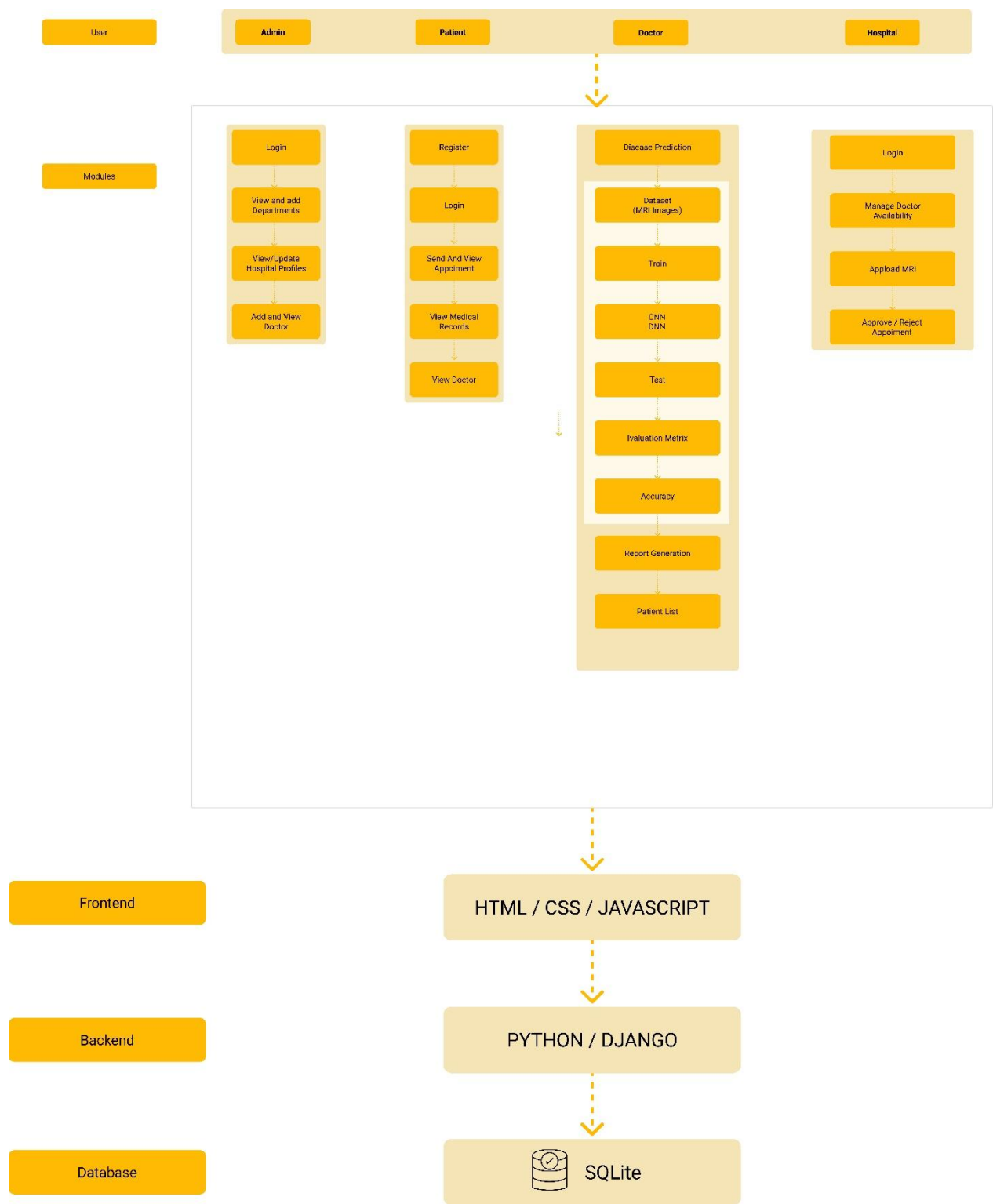
Your central hub for managing appointments, reviewing patient cases, and staying informed.



## Welcome, Avril Martin

Manage your appointments, view prescriptions, and access personalized care plans.

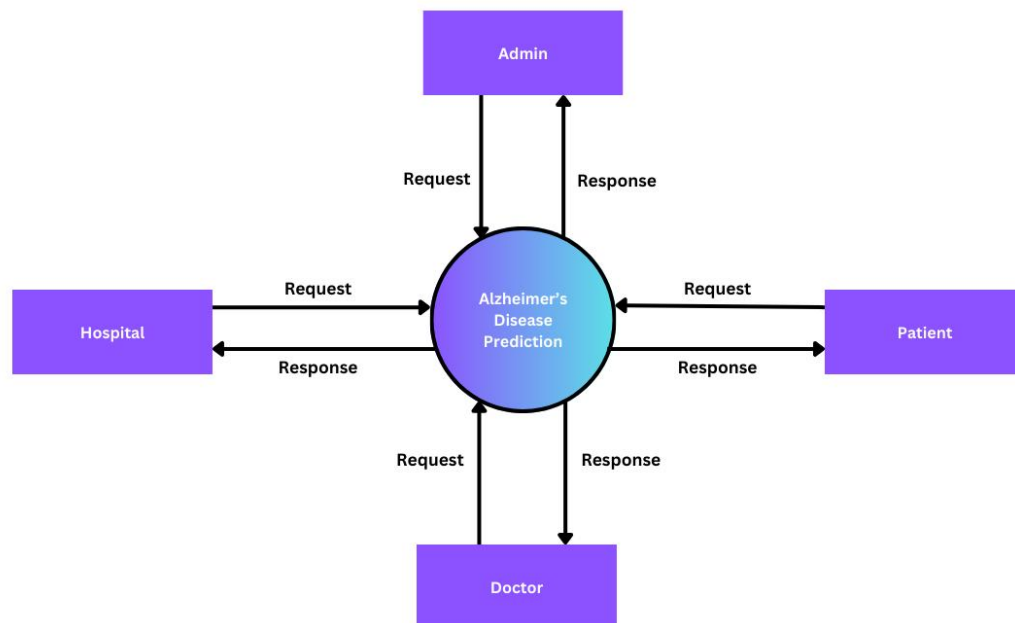
# 6.2 Architecture Design



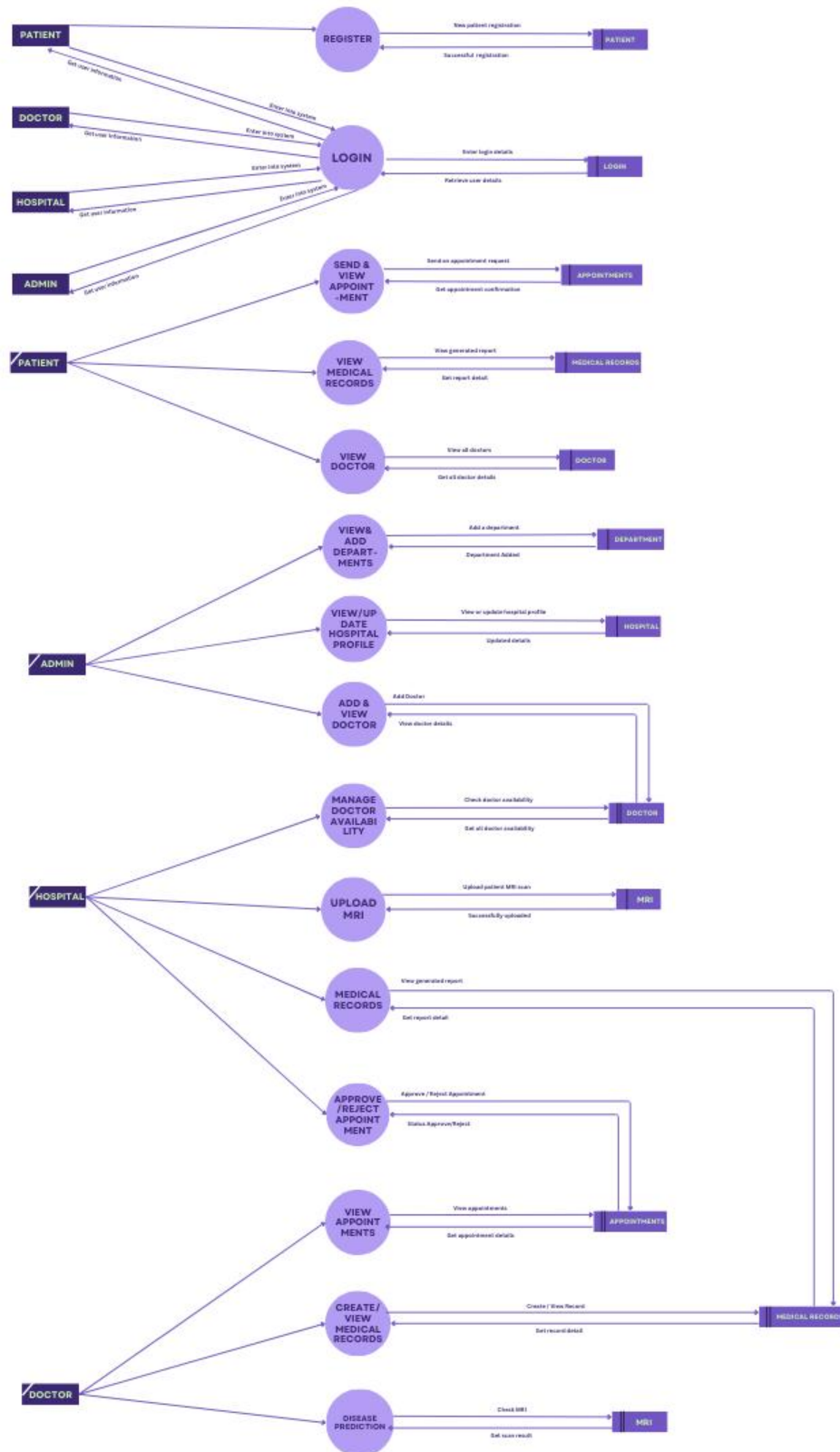


## 6.2.1 Flow Diagrams

### DFD Level-0

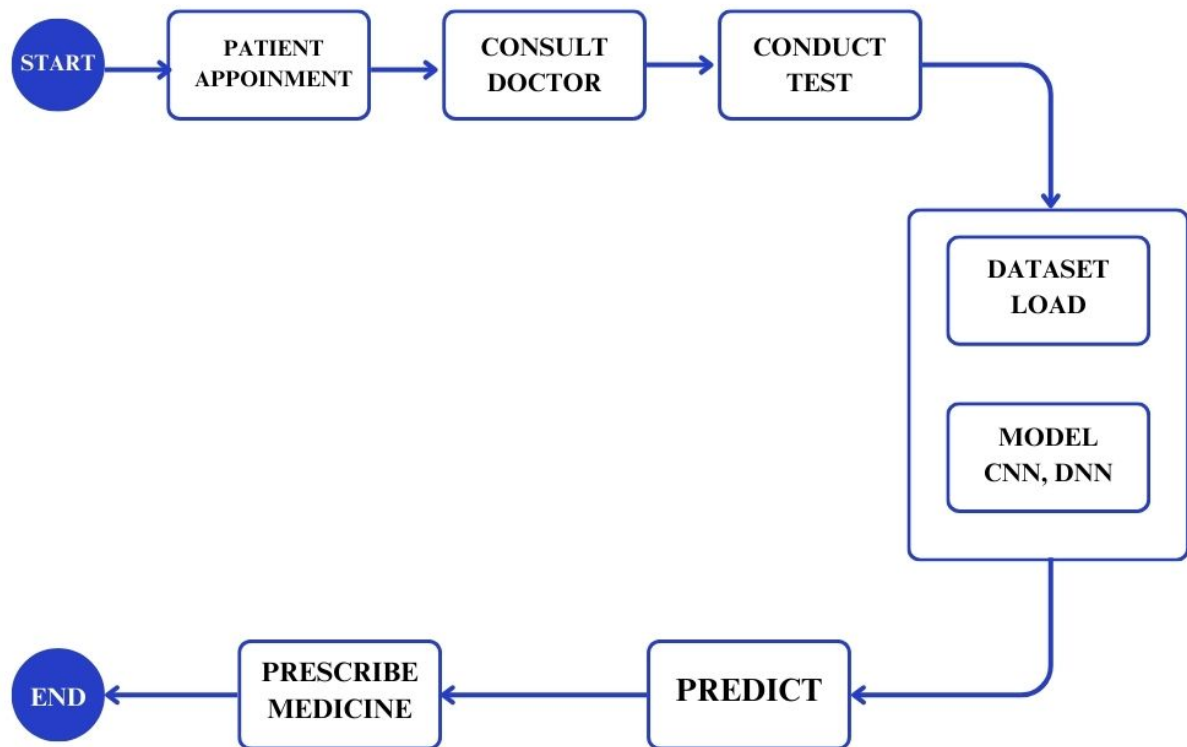


## Level 1 DFD

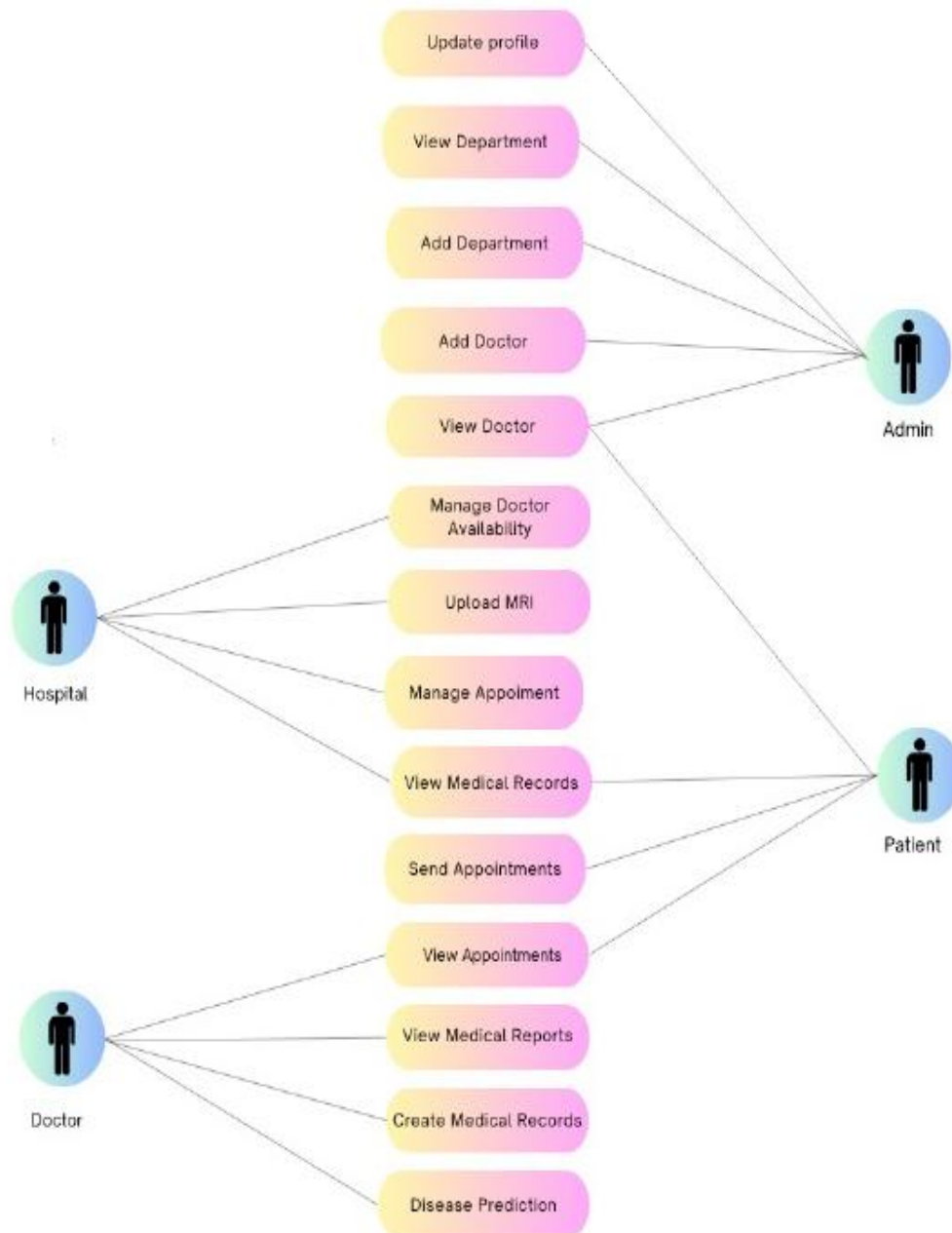




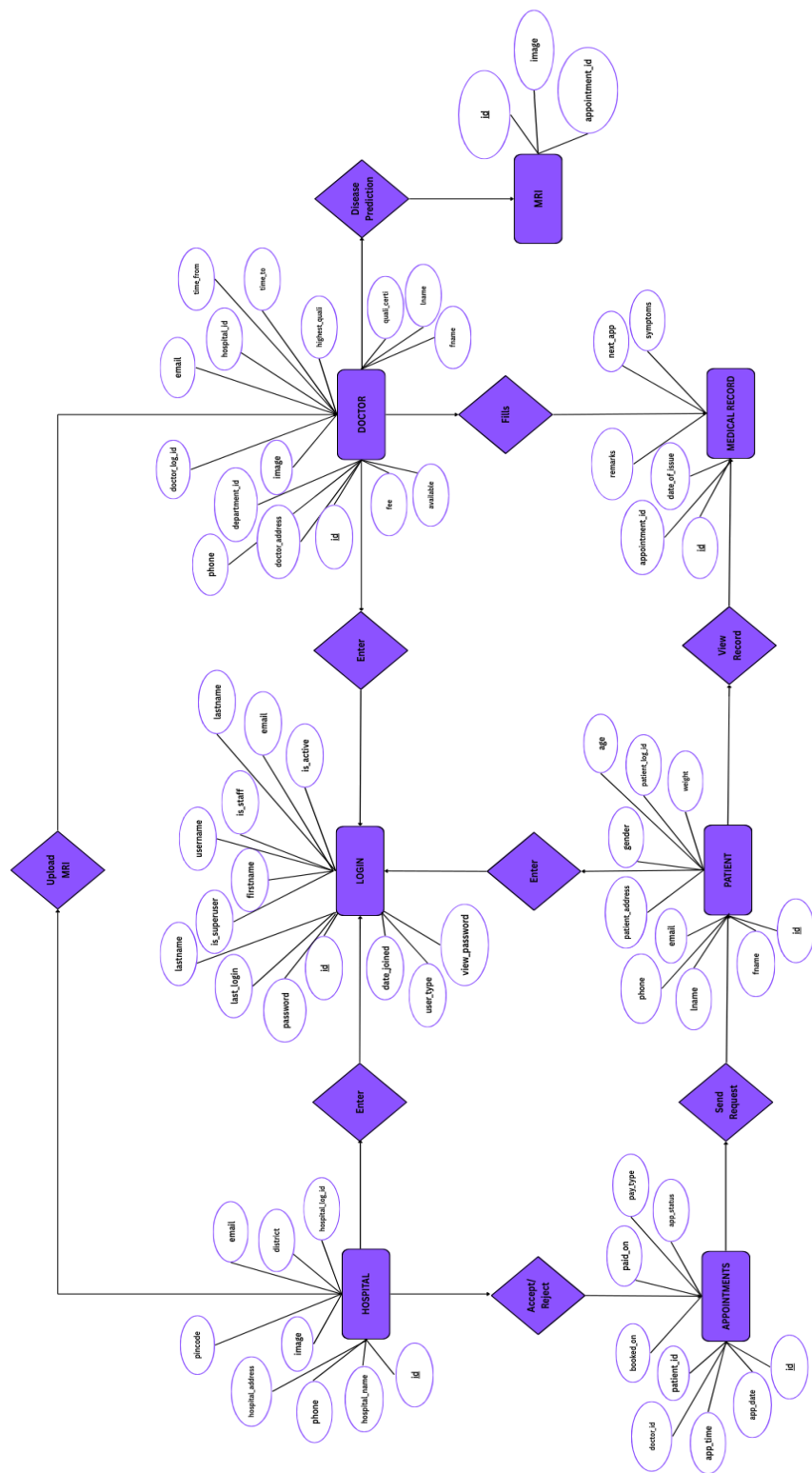
## Project Workflow Diagram



## Use Case Diagram



## 6.2.2 ER Diagram



## 6.2.3 Table Design

### Medico\_app\_appointment

| Attributes | Data Type                         | Constraints                           |
|------------|-----------------------------------|---------------------------------------|
| id         | INT                               | PRIMARY KEY, AUTO INCREMENT           |
| app_date   | DATE                              |                                       |
| app_time   | TIME                              |                                       |
| doctor_id  | INT                               | FOREIGN KEY REFERENCES doctorreg(id)  |
| patient_id | INT                               | FOREIGN KEY REFERENCES patientreg(id) |
| booked_on  | DATE                              |                                       |
| paid_on    | DATE                              |                                       |
| paytype    | ENUM("debit card", "credit card") |                                       |
| app_status | ENUM("pending", "approved")       |                                       |

### Medico\_app\_department

| Attributes      | Data Type    | Constraints                 |
|-----------------|--------------|-----------------------------|
| id              | INT          | PRIMARY KEY, AUTO INCREMENT |
| department_name | VARCHAR (30) |                             |

### **Medico\_app doctor**

| Attributes     | Data Type     | Constraints                                       |
|----------------|---------------|---|
| id             | INT           | PRIMARY KEY, AUTO INCREMENT                       |
| phone          | VARCHAR (15)  |   |
| email          | VARCHAR (50)  |   |
| doctor_address | TEXT          |   |
| department_id  | INT           | FOREIGN KEY REFERENCES medico_app_department (id) |
| doctor_log_id  | INT           | FOREIGN KEY REFERENCES login (id)                 |
| hospital_id    | INT           | FOREIGN KEY REFERENCES hospitalreg(id)            |
| image          | VARCHAR (255) |   |
| time_from      | TIME          |   |
| time_to        | TIME          |   |
| highest_quali  | VARCHAR (30)  |   |
| quali_certi    | VARCHAR (255) |   |
| fname          | VARCHAR (30)  |   |
| lname          | VARCHAR (30)  |   |
| fee            | VARCHAR (30)  |   |
| available      | VARCHAR (30)  |   |

### **Medico app MRI**

| Attributes     | Data Type     | Constraints                                       |
|----------------|---------------|---|
| id             | INT           | PRIMARY KEY, AUTO INCREMENT                       |
| image          | VARCHAR (255) |   |
| appointment_id | INT           | FOREIGN KEY REFERENCES medico_app_appointment(id) |

### **Medico app hospital**

| Attributes       | Data Type     | Constraints                                 |
|------------------|---------------|---|
| id               | INT           | PRIMARY KEY, AUTO INCREMENT                 |
| hospital_name    | VARCHAR (50)  |   |
| phone            | VARCHAR (15)  |   |
| email            | VARCHAR(50)   |   |
| hospital_address | TEXT          |   |
| pincode          | VARCHAR(10)   |   |
| district         | VARCHAR(50)   |   |
| hospital_log_id  | INT           | FOREIGN KEY REFERENCES medico_app_login(id) |
| image            | VARCHAR (255) |   |

## Medico app login

| Attributes    | Data Type     | Constraints                 |
|---------------|---------------|-----------------------------|
| id            | INT           | PRIMARY KEY, AUTO INCREMENT |
| password      | VARCHAR (128) |                             |
| last_login    | DATETIME      |                             |
| is_superuser  | BOOL          |                             |
| username      | VARCHAR(150)  |                             |
| firstname     | VARCHAR(30)   |                             |
| lastname      | VARCHAR(30)   |                             |
| email         | VARCHAR (50)  |                             |
| is_staff      | BOOL          |                             |
| is_active     | BOOL          |                             |
| date_joined   | DATETIME      |                             |
| user_type     | VARCHAR (255) |                             |
| view_password | VARCHAR (30)  |                             |

### **Medico\_app\_medicalrecord**

| Attributes     | Data Type | Constraints                                       |
|----------------|-----------|---|
| id             | INT       | PRIMARY KEY, AUTO INCREMENT                       |
| date_of_issue  | DATE      |   |
| remarks        | TEXT      |   |
| appointment_id | INT       | FOREIGN KEY REFERENCES medico_app_appointment(id) |
| next_app       | DATE      |   |
| symptoms       | TEXT      |   |

### **Medico\_app\_medicine\_details**

| Attributes        | Data Type    | Constraints  |
|-------------------|--------------|--|
| id                | INT          | PRIMARY KEY, AUTO INCREMENT                          |
| m_name            | VARCHAR (30) |  |
| m_dosage          | VARCHAR (30) |  |
| m_quantity        | VARCHAR(30)  |  |
| m_directions      | VARCHAR(30)  |  |
| m_days            | VARCHAR(30)  |  |
| medical_record_id | INT          | FOREIGN KEY REFERENCES medico_app_medical_record(id) |



## **Medico app Patient**

| Attributes      | Data Type    | Constraints                                 |
|-----------------|--------------|---|
| id              | INT          | PRIMARY KEY, AUTO INCREMENT                 |
| fname           | VARCHAR (30) |   |
| lname           | VARCHAR(30)  |   |
| phone           | VARCHAR(15)  |   |
| email           | VARCHAR(50)  |   |
| patient_address | TEXT         |   |
| gender          | VARCHAR(10)  |   |
| age             | VARCHAR 30)  |   |
| weight          | VARCHAR(30)  |   |
| patient_log_id  | INT          | FOREIGN KEY REFERENCES medico_app_login(id) |

## **7.0 SYSTEM TESTING AND IMPLIMENTATION**

### **7.1 System testing**

System testing ensures that the Alzheimer’s Detection System functions correctly, meets all requirements, and is ready for deployment. It covers the system's functionality, performance, security, compatibility, and usability.

#### **1. Functional Testing**

- **Objective:** Verify that core features like image processing and user access work correctly.
- **Approach:** Test MRI uploads, disease detection accuracy, and login/data retrieval.
- **Expected Outcome:** Accurate detection, secure login, and error-free report generation.

## **2. Compatibility Testing**

- **Objective:** Confirm the system works across different platforms and devices.
- **Approach:** Run tests on various OS, browsers, and screen sizes.
- **Expected Outcome:** Smooth operation without UI issues on all supported platforms.

## **7.2 Maintenance**

Maintenance is vital for ensuring the Alzheimer's Detection System remains efficient, reliable, and up-to-date. It involves fixing issues, adapting to changes, and improving performance post-deployment.

1. Fixing Issues – Solving bugs, errors, or system crashes that occur during use.
2. Check User Access – Make sure only authorized people can access sensitive data.
3. Maintain Compatibility – Ensure the system works well with different browsers.

## **8.0 CONCLUSION**

The proposed system offers a robust platform for early prediction and management of Alzheimer's disease, leveraging advanced deep learning algorithms, particularly Convolutional Neural Networks (CNNs) and Deep Neural Networks (DNNs), to analyze MRI scans and neuroimaging data. This enables early detection of Alzheimer's, even in its initial stages, by identifying subtle brain changes often missed by traditional methods, facilitating timely interventions that can slow disease progression and improve outcomes.

Beyond diagnosis, the system includes features for continuous monitoring of cognitive function and disease progression, providing healthcare professionals with real-time data for decision-making. The platform also offers an intuitive interface for healthcare providers to access medical records, track treatment plans, and collaborate with specialists, streamlining patient care.

Designed for ease of use, the system ensures both patients and healthcare professionals can easily navigate its features. Patients benefit from personalized care plans and real-time health tracking, empowering them to manage their condition actively.

By combining early detection, continuous monitoring, and accessible care, the system transforms Alzheimer's diagnosis and management. It improves diagnostic accuracy, tailors interventions to individual needs, and empowers both healthcare providers and patients, potentially enhancing quality of life, reducing costs, and improving patient outcomes.

## **9.0 SCOPE FOR FUTURE DEVELOPMENT**

1. Mobile App Support – Create a mobile app for patients and doctors to access reports and appointments easily.
2. Remote Monitoring – Add features to monitor patients from home using wearable devices.
3. Real-time Alerts – Notify doctors or caregivers instantly if abnormal patterns are detected in MRI or behavior data.
4. Multilingual Support – Include different languages to help users from various regions.
5. Voice Assistants – Use voice commands for elderly users to interact easily with the system.

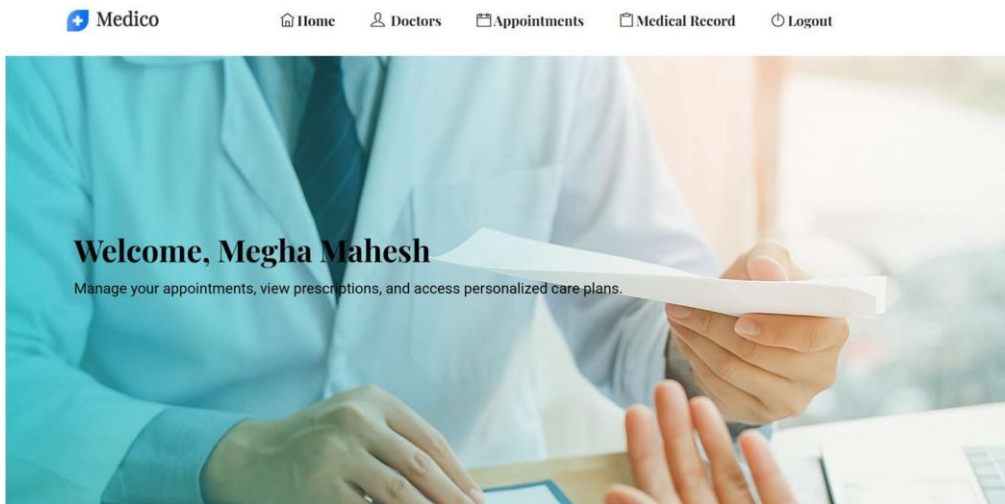
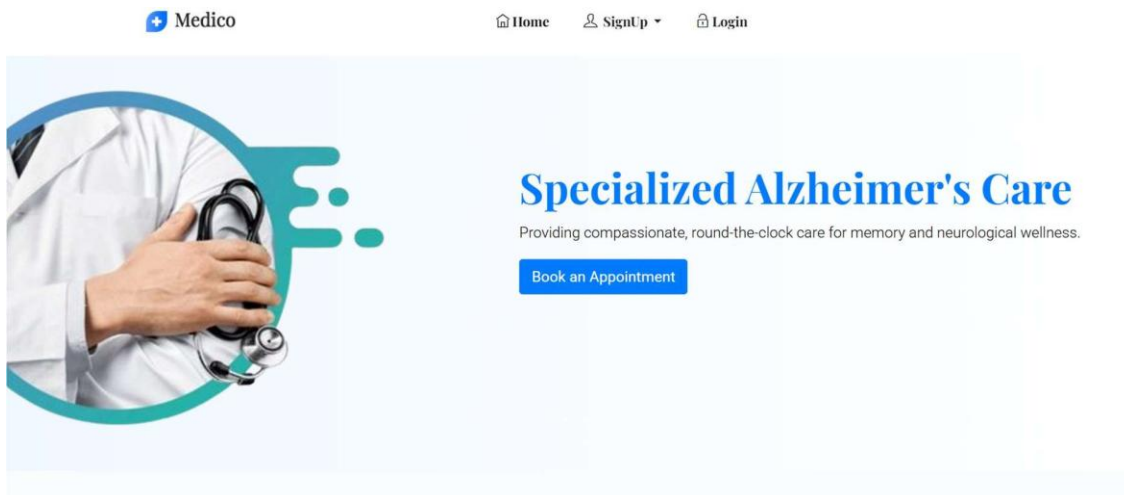
## **10.0 BIBLIOGRAPHY**

### **Websites Reference:**

<https://www.mayoclinic.org/diseases-conditions/alzheimers-disease/symptoms-causes> <https://www.medicalnewstoday.com/articles>  
<https://viso.ai/deep-learning/deep-neural-network-three-popular-types/>  
<https://www.w3schools.com/>  
<https://www.geeksforgeeks.org/>



## 11.0 APPENDIX

### 11.1 Screenshots



## Patient Signup

|  |                                     |
|--|-------------------------------------|
| <b>First Name</b>                                      | <b>Last Name</b>                    |
| <input type="text" value="Megha"/>                     | <input type="text" value="Mahesh"/> |
| <b>Phone</b>   |                                     |
| <input type="text" value="8075426767"/>                |                                     |
| <b>Email Address</b>                                   |                                     |
| <input type="text" value="megha2004mahesh@gmail.com"/> |                                     |
| <b>Password</b>  |                                     |
| <input type="password" value="....."/>                 |                                     |
| <b>Address</b>   |                                     |
| <input type="text" value="Skyline apartments, Petta"/> |                                     |
| <b>Gender</b>  |                                     |
| <input type="text" value="Female"/>                    |                                     |
| <b>Age</b>   | <b>Weight</b>                       |
| <input type="text" value="21"/>                        | <input type="text" value="51"/>     |
| <input type="button" value="Register"/>                |                                     |

|   |  |
|---|--|
|  <p><b>Vignesh H R</b><br/>BDS<br/>Neurology</p> | <p><b>Email</b><br/>vignesh@gmail.com</p> <p><b>Phone</b><br/>9085201479</p> <p><b>Address</b><br/>vignesh address</p> <p><b>Consultation Time</b><br/>10 a.m. - 3:30 p.m.</p> <p><b>Consultation Fee</b><br/>300 Rs.</p> <p><input type="button" value="Make Appointment"/></p>       |
|  <p><b>Nia V</b><br/>MBBS<br/>Geriatician</p>    | <p><b>Email</b><br/>nia@gmail.com</p> <p><b>Phone</b><br/>9123456780</p> <p><b>Address</b><br/>Elamakkara,Kochi,Kerala</p> <p><b>Consultation Time</b><br/>2:43 p.m. - 2:49 p.m.</p> <p><b>Consultation Fee</b><br/>200 Rs.</p> <p><input type="button" value="Make Appointment"/></p> |

### Appointment Request

**Name**  
Megha Mahesh

**Email**  
megha2004mahesh@gmail.com

**Phone**  
8075426767

**Department**  
Geriatrician

**Preferred Doctor**  
Ema j

**Preferred Date**  
16-04-2025

**Preferred Time**  
02:09 PM

Send Request

Medico

[Home](#) [Doctors](#) [Appointments](#) [Medical Record](#) [Logout](#)

### PAYMENT DETAILS

Doctor  
Priya S. Neurology

Date  
March 26, 2025

Hospital  
Emergency hospital

Consultation Fee  
200 Rs.

Choose Payment Type  
Debit Card

Proceed to Pay

| # | PATIENT      | DOCTOR                                   | APPOINTMENT DATE | STATUS             | APPROVE/REJECT                                | MEDICAL RECORD       | ADD MRI SCAN  |
|---|--------------|--|------------------|--------------------|---|----------------------|---|
| 1 | Megha Mahesh | Ema j, Geriatrician Emergency hospital   | April 16, 2025   | Paid               | <a href="#">Accept</a> <a href="#">Reject</a> | <a href="#">View</a> | <input type="button" value="Choose File"/> No file chosen<br><input type="button" value="Upload MRI Scan"/> |
| 2 | Avril Martin | Ema j, Geriatrician Emergency hospital   | April 15, 2025   | Visited            | no action                                     | <a href="#">View</a> | <input type="button" value="Choose File"/> No file chosen<br><input type="button" value="Upload MRI Scan"/> |
| 3 | Avril Martin | neena a, Geriatrician Emergency hospital | April 13, 2025   | Visited            | no action                                     | <a href="#">View</a> | <input type="button" value="Choose File"/> No file chosen<br><input type="button" value="Upload MRI Scan"/> |
| 4 | meera a      | neena a, Geriatrician Emergency          | April 13, 2025   | Prescription added | no action                                     | <a href="#">View</a> | <input type="button" value="Choose File"/> No file chosen<br><input type="button" value="Upload MRI Scan"/> |

### Doctor Availability

| # | DOCTOR      | DEPARTMENT   | AVAILABILITY                        | ACTION                                |
|---|-------------|--------------|-------------------------------------|---------------------------------------|
| 1 | Vignesh H R | Neurology    | <input checked="" type="checkbox"/> | <input type="button" value="Submit"/> |
| 2 | Priya S     | Neurology    | <input type="checkbox"/>            | <input type="button" value="Submit"/> |
| 3 | Nia V       | Geriatrician | <input checked="" type="checkbox"/> | <input type="button" value="Submit"/> |
| 4 | appu kuttan | Geriatrician | <input checked="" type="checkbox"/> | <input type="button" value="Submit"/> |
| 5 | Iva a       | Geriatrician | <input checked="" type="checkbox"/> | <input type="button" value="Submit"/> |
| 6 | Ema j       | Geriatrician | <input checked="" type="checkbox"/> | <input type="button" value="Submit"/> |
| 7 | neena a     | Geriatrician | <input checked="" type="checkbox"/> | <input type="button" value="Submit"/> |

## Appointments

dd-mm-yyyy 

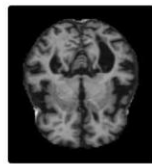
[FILTER](#)

[ALL](#)

Booking List for April 16, 2025

| Patient Name | Age | Booked On      | Visiting Time | Status                  | Medical Record              | Actions                          | MRI Scan                      |
|--------------|-----|----------------|---------------|-------------------------|-----------------------------|----------------------------------|-------------------------------|
| Megha Mahesh | 21  | April 16, 2025 | 02:09 PM      | <a href="#">Visited</a> | <a href="#">View Record</a> | <a href="#">Add Prescription</a> | <a href="#">View MRI Scan</a> |

## MRI



[CHECK](#)

[Diagnosis Result](#)

NonDemented



## Megha Mahesh

**Email:**  
megha2004mahesh@gmail.com

**Phone:** 8075426767

**Address:** Skyline apartments, Petta

**Prescription Date:** April 16, 2025

**Prescribed By:** Ema J, Geriatrician

**Hospital:** Emergency hospital

**Contact:** 9567217436

### Symptoms:

memory problems, difficulty with language, spatial issues, and changes in mood

### Medicines:

| Medicine  | Dosage | Quantity | Directions  | Days |
|-----------|--------|----------|-------------|------|
| Donepezil | 10mg   | 10       | Before Food | 5    |

### Remarks:

Non Demented

### Tests:

| Test Name    | Test Result   |
|--------------|---|
| Non Demented | <div> <input type="button" value="Choose File"/> No file chosen                 </div> <div> <input type="button" value="Upload Test Result"/> </div> |

## Welcome to the Admin Panel

Efficiently manage staff, departments, and hospital data from a centralized dashboard.

Add  
View

EMERGENCY

## Register Doctor

Doctor Image  
 anita.jpg

First Name

Last Name

Highest Qualification

Qualification Certificate  
 cer.jpg

Phone

Email

Password

Department

Address

Doctor Available From

Doctor Available To

Consultation Fee

Register





Add

View

Choose

Filter

All

|  |  |
|--|--|
|  <p>Vignesh H R</p> <p>BDS</p> <p>Neurology</p> | <p>Email<br/>vignesh@gmail.com</p> <p>Phone<br/>9085201479</p> <p>Address<br/>vignesh address</p> <p>Consultation Time<br/>10 a.m. - 3:30 p.m.</p> <p>Consultation Fee<br/>300 Rs.</p> |
|  <p>Priya S</p> <p>BDS</p> <p>Neurology</p>     | <p>Email<br/>priya@gmail.com</p> <p>Phone<br/>7894562013</p> <p>Address<br/>priya address</p> <p>Consultation Time<br/>9:30 a.m. - 5:30 p.m.</p> <p>Consultation Fee<br/>200 Rs.</p>   |
|  <p>Vignesh H R</p> <p>BDS</p> <p>Neurology</p> | <p>Email<br/>vignesh@gmail.com</p> <p>Phone<br/>9085201479</p> <p>Address<br/>vignesh address</p> <p>Consultation Time<br/>10 a.m. - 3:30 p.m.</p> <p>Consultation Fee<br/>300 Rs.</p> |
|  <p>Priya S</p> <p>BDS</p> <p>Neurology</p>     | <p>Email<br/>priya@gmail.com</p> <p>Phone<br/>7894562013</p> <p>Address<br/>priya address</p> <p>Consultation Time<br/>9:30 a.m. - 5:30 p.m.</p> <p>Consultation Fee<br/>200 Rs.</p>   |

## ADD NEW DEPARTMENT

[Add Department](#)

## VIEW DEPARTMENTS

| Department Name | Action                                      |
|-----------------|---|
| Geriatrician    | <a href="#">Edit</a> <a href="#">Delete</a> |
| Neurology       | <a href="#">Edit</a> <a href="#">Delete</a> |



Emergency hospital  
Emakulam

### Information

#### Email

emer@gmail.com

#### Phone

9567217436

#### Address

XXXXXX

#### Pincode

682024

#### Action

[Update Profile](#)

## 11.2 Sample Code

index.html

```
{% load static %}

{% for message in messages %}

<script>alert('{{ message }}')</script>

{% endfor %}

<!doctype html>

<html lang="en">
<head>

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

    <title>medico</title>

    <link rel="icon" href="{% static 'img/favicon.png' %}">

    <link rel="stylesheet" href="{% static 'css/bootstrap.min.css' %}">

    <link rel="stylesheet" href="{% static 'css/animate.css' %}">

    <link rel="stylesheet" href="{% static 'css/owl.carousel.min.css' %}">

    <link rel="stylesheet" href="{% static 'css/themify-icons.css' %}">

    <link rel="stylesheet" href="{% static 'css/flaticon.css' %}">

    <link rel="stylesheet" href="{% static 'css/magnific-popup.css' %}">

    <link rel="stylesheet" href="{% static 'css/nice-select.css' %}">
```

```
<link rel="stylesheet" href="{% static 'css/slick.css' %}">
```

```
<link rel="stylesheet" href="{% static 'css/style.css' %}">
```

```
<style>
```

```
html, body {
```

```
    height:
```

```
    100%;
```

```
    margin: 0;
```

```
}
```

```
body {
```

```
    display: flex;
```

```
    flex-direction:
```

```
    column;    min-
```

```
    height: 100vh;
```

```
}
```

```
main {
```

```
    flex: 1;
```

```
}
```

```
footer.footer-area {
```

```
    background-color:
```

```
    #f8f9fa;
```

```
    padding: 20px 0;
```

```
}
```

```
</style>
```

```
</head>
```

```

<body>

<header class="main_menu home_menu" style="background-color: rgb(255, 255, 255);">

  <div class="container">

    <div class="row align-items-center">

      <div class="col-lg-12">

        <nav class="navbar navbar-expand-lg navbar-light">

          <a class="navbar-brand" href="/">  </a>

          <button class="navbar-toggler" type="button" data-toggle="collapse"

            data-target="#navbarSupportedContent"                aria-
controls="navbarSupportedContent"

            aria-expanded="false" aria-label="Toggle navigation">

            <span class="navbar-toggler-icon"></span>

          </button>
          <div class="collapse navbar-collapse main-menu-item justify-
            content-center" id="navbarSupportedContent">
            <ul class="navbar-nav align-items-center" style="font-size:xx-large;">

              <li class="nav-item active">

                <a style="font-size: large;" class="nav-link" href="/"><i
class="ti- home pr-1"></i>Home</a>

              </li>
              <li class="nav-item dropdown">

                <a style="font-size: large;" class="nav-link dropdown-

```

```

toggle" href="blog.html" id="navbarDropdown"

        role="button"        data-toggle="dropdown"        aria-
haspopup="true" aria-expanded="false"><i class="ti-user pr-1"></i>

        SignUp

    </a>

    <div class="dropdown-menu" aria-labelledby="navbarDropdown">

        <!-- <a class="dropdown-item" href="/hospital-signup">Hospital
Signup</a> -->

        <a style="font-size: large;" class="dropdown-item"
href="/patient-signup">Patient Signup</a>

    </div>

</li>

<li class="nav-item">

    <a style="font-size: large;" class="nav-link" href="/users-
login"><i class="ti-unlock pr-1"></i>Login</a>

</li>

</ul>

</div>

<!-- <a class="btn_2 d-none d-lg-block" href="#">HOT LINE- 09856</a> -->

</nav>

</div>

</div>

```

</div>

</header>

<main>

<section class="banner\_part d-flex align-items-center" style="background: linear-gradient(to right, #f0f8ff, #ffffff); height: 650px;">

<div class="container-fluid px-0">

<div class="row no-gutters align-items-center">

<div class="col-md-6">



</div>

<div class="col-md-6" style="margin-left: -70px;">

<h1 class="display-4 font-weight-bold text-primary">Specialized Alzheimer's Care</h1>

<p class="lead" style="color: #000;">Providing compassionate, round-the-clock care for memory and neurological wellness.</p>

<a href="/users-login" class="btn btn-primary btn-lg mt-3">Book an Appointment</a>

</div>

</div>

</div>



</section>

<section class="py-5" style="background-color: #f9fbfc;">

<div class="container">

<div class="row justify-content-center text-center mb-4">

<div class="col-lg-10">

<h2 class="display-5 font-weight-bold text-primary mb-3">Expertise You Can Trust</h2>

<p class="lead mb-0" style="color: #000;">

At Medico Alzheimer's Hospital, we deliver patient-centric, evidence-based care through our core departments — <strong>Neurology</strong> and <strong>Geriatric Medicine</strong>. With cutting-edge technology and a compassionate approach, we support individuals and families facing Alzheimer's and related conditions.

</p>

</div>

</div>

<div class="row mt-4">

<div class="col-md-6 mb-4">

div class="p-4 shadow-sm rounded bg-white h-100 border-left border-primary">

<h4 class="mb-3 text-primary">Geriatric Medicine</h4>

<p style="color: #000; font-size: 1.05rem;">

Our Geriatricians specialize in enhancing the health, independence, and dignity

of older adults. Services include senior wellness programs, mobility and medication support, and cognitive and behavioral assessments — all tailored for holistic elder care.

</p>

</div>

</div>

<div class="col-md-6 mb-4">

<div class="p-4 shadow-sm rounded bg-white h-100 border-left border-primary">

<h4 class="mb-3 text-primary">Neurology</h4>

<p style="color: #000; font-size: 1.05rem;">

Our Neurology team offers advanced brain imaging, personalized memory clinic consultations, and long-term neurological support. We focus on individualized care plans for Alzheimer’s and related cognitive disorders.

</p>

</div>

</div>

</div>

<div class="row mt-5">

<div class="col-md-12 text-center mb-4">

<h3 class="text-info font-weight-bold">Why Choose Medico Alzheimer's Hospital?</h3>

<p class="text-muted mb-0">

Because your loved ones deserve care that’s as specialized and heartfelt as it

is scientifically advanced.

</p>

</div>

div class="col-md-4 text-center mb-4">

<i class="ti-medall icon" style="font-size: 3rem; color: #17a2b8;"></i>

<h5 class="mt-3 font-weight-bold">Accredited Specialists</h5>

<p class="text-muted">A team of certified professionals with deep expertise in Alzheimer's care.</p>

</div>

<div class="col-md-4 text-center mb-4">

<i class="ti-bar-chart-alt icon" style="font-size: 3rem; color: #17a2b8;"></i>

<h5 class="mt-3 font-weight-bold">Personalized, Data-Driven Treatment</h5>

<p class="text-muted">Every treatment plan is guided by real-time health data and research insights.</p>

</div>

<div class="col-md-4 text-center mb-4">

<i class="ti-support icon" style="font-size: 3rem; color: #17a2b8;"></i>

<h5 class="mt-3 font-weight-bold">24/7 Patient Support</h5>

<p class="text-muted">Compassionate support teams available anytime — for both medical and emotional needs.</p>

</div>

</div>

```


44


```

```

<script src="{% static 'js/popper.min.js' %}"></script>

<script src="{% static 'js/bootstrap.min.js' %}"></script>

<script src="{% static 'js/owl.carousel.min.js' %}"></script>

<script src="{% static 'js/jquery.nice-select.min.js' %}"></script>

<script src="{% static 'js/jquery.ajaxchimp.min.js' %}"></script>

<script src="{% static 'js/jquery.form.js' %}"></script>

<script src="{% static 'js/jquery.validate.min.js' %}"></script>

<script src="{% static 'js/mail-script.js' %}"></script>

<script src="{% static 'js/contact.js' %}"></script>

<script src="{% static 'js/custom.js' %}"></script>

</html>

```

### patient\_signup.html

```

{% extends 'hospital_signup.html' %}

{% load static %}
{% block index %}

<style>

.signup-background {

background-image:          url('{%          static
"img/reg_form.jpeg" %}'); background-size: cover;
background-position: center;

padding: 100px 0 50px;

```

```

    min-height: 100vh;

}

.card {

    background-color: rgba(255, 255, 255, 0.8); padding: 30px;
    border-radius: 10px;

}

.form-control {

    background-color: #f7f7f7; /* Lighter background for better
    contrast */ color: #333333; /* Dark text color */
    border: 1px solid #888888; /* Darker border
    color */ font-size: 1rem; /* Make text more
    readable */ padding: 10px; /* Increase padding
    for easier typing */

}

.form-control:focus {

    border-color: #007bff; /* Change the border color when focused */

    box-shadow: 0 0 5px rgba(0, 123, 255, 0.5); /* Add a subtle shadow on focus */

}

.font-weight-bold {

    color: #333333; /* Dark text color for labels */

}

.btn-info {

```

```

        background-color: #007bff; /* A deeper blue color for better
        visibility */ border: none;
    }
    .btn-info:hover {

        background-color: #0056b3; /* Darker blue on hover */

    }
    {
        background-color:
        #f7f7f7; color: #333333;

        padding: 10px;

        border: 1px solid #888888; /* Darker border color */

    }
    {
        background-color:
        #f7f7f7; color: #333333;

        padding: 10px;

        border: 1px solid #888888; /* Darker border color */

    }

</style>

<div class="container mt-3">

    {% for message in messages %}

        <div class="alert alert-info text-center" role="alert" style="max-width: 500px;
margin: auto;">

            {{ message }}

```

```

</div>

{ % endfor % }

</div>

<section class="contact-section section_padding signup-background">

    <div class="container">

<div    class="row    justify-content-
center">

<div class="col-md-8">

    <div class="card shadow-lg p-4 rounded">

        <h3 class="text-info text-center mb-4" style="font-size: 2rem;">Patient
Signup</h3>

        <form    class="form-contact    contact_form"
method="post" enctype="multipart/form-data">

            { % csrf_token % }

            <div class="row">

                <div class="col-sm-6">

<div class="form-group">

                    <label class="font-weight-bold">First Name</label>

                    <input class="form-control" name="fname" pattern="[a-zA-Z ]+"
required type="text"

                        title="Only alphabets and spaces are allowed." placeholder="Enter
first name">

```



```

        </div>

    </div>

    <div class="col-sm-6">

        <div class="form-group">

            <label class="font-weight-bold">Last Name</label>

            <input class="form-control" name="lname" pattern="[a-zA-Z ]+"
required type="text"

                title="Only alphabets and spaces are allowed." placeholder="Enter
last name">

        </div>

    </div>

    <div class="col-sm-12">

        <div class="form-group">

            <label class="font-weight-bold">Phone</label>

            <input          class="form-control"          name="phone"
pattern="[6789][0-9]{9}" maxlength="10" required type="text"

                title="Phone number must start with 6, 7, 8, or 9 and
contain 10 digits." placeholder="Enter phone">

        </div>

    </div>

    <!-- Email -->

    <div class="col-sm-12">

```

```

<div class="form-group">

    <label class="font-weight-bold">Email Address</label>

    <input class="form-control" name="email" type="email"
pattern="[a-z0-9._%+-]+@[a-z0-9.-]+\.[a-z]{2,4}$" required

        title="Enter a valid email format (e.g.,
example@mail.com)." placeholder="Enter email address">

    </div>

</div>

<div class="col-sm-12">

    <div class="form-group">

        <label class="font-weight-bold">Password</label>

        <input class="form-control" name="password"
type="password" pattern="^(?=.*[A-Za-z])(?=.*\d)(?=.*[@$!%*#?&])[A-Za-
z\d@$!%*#?&]{8,}$" required

            title="Password must be at least 8 characters long and
include at least one letter, one number, and one special character (@, $, !, %, *, #,
?, &)." placeholder="Enter password">

        </div>

    </div>

<div class="col-12">

    <div class="form-group">

        <label class="font-weight-bold">Address</label>

        <textarea class="form-control" name="address" cols="30"

```

```
rows="3" required placeholder="Enter Address"></textarea>
```

```
</div>
```

```
</div>
```

```
<div class="col-sm-12">
```

```
<div class="form-group">
```

```
<label class="font-weight-bold">Gender</label>
```

```
<select class="form-control" name="gender" required>
```

```
<option selected disabled>Choose gender</option>
```

```
<option value="Male">Male</option>
```

```
<option value="Female">Female</option>
```

```
</select>
```

```
</div>
```

```
</div>
```

```
<div class="col-sm-6">
```

```
<div class="form-group">
```

```
<label class="font-weight-bold">Age</label>
```

```
<input class="form-control" name="age" pattern="[0-9]+"
```

```
maxlength="3" required type="text"
```

```
title="Enter a valid age (numeric value only)." placeholder="Enter  
age">
```

```
</div>
```

```

</div>

<div class="col-sm-6">

    <div class="form-group">

        <label class="font-weight-bold">Weight</label>

        <input class="form-control" name="weight" pattern="[0-9]+"
maxlength="3" required type="text"
        title="Enter a valid weight (numeric value only)."
placeholder="Enter weight">

    </div>

</div>

</div>

<div class="form-group text-center mt-4">

    <button type="submit" class="btn btn-info px-4 py-2" style="border-radius:
5px;">Register</button>

</div>

</form>

</div>

</div>

</div>

</div>

</section>
{% endblock index %}

```

## hospital\_adddoctor.html

```
{% endblock hospital %}{% extends 'admin/admin_dashboard.html' %}

{% for message in messages %}

<script>alert('{{ message }}')</script>

{% endfor %}

{% block hospital %}

<section class="breadcrumb_part breadcrumb_bg " style="background-image:
url(../static/img/5183184.jpg);

background-position: center;

background-repeat: no-repeat;

background-size: cover;

margin-top: 70px; width: 100%; height: 500px;" >

<div class="container">

<div class="row">

<div class="col-lg-12">

<div class="breadcrumb_iner">

<div class="breadcrumb_iner_item" >

</div>

</div>

</div>

</div>

</div>

</div>

</section>
```

```
<h3 style=" color: #17a2b8; font-size: 3rem; margin-bottom: 5px; text-align: center;">Add Doctor</h3>
```

```
<section class="contact-section section_padding" style="margin-top: -65px;">
```

```
<div class="container">
```

```
<div class="row ">
```

```
<div class="col-lg-8 mx-auto">
```

```
<form class="form-contact contact_form" method="post" enctype="multipart/form-data">
```

```
{% csrf_token %}
```

```
<div class="row">
```

```
<div class="col-sm-12">
```

```
<div class="form-group">
```

```
<p>Enter doctor image</p>
```

```
<input class="form-control" name="image" type="file" onfocus="this.placeholder  
= ""
```

```
onblur="this.placeholder = 'Enter doctor image'" placeholder='Enter doctor image'>
```

```
</div>
```

```
</div>
```

```
<div class="col-sm-6">
```

```
<div class="form-group">
```

```
<input class="form-control" name="fname" pattern="[a-zA-Z ]+" required  
type="text" onfocus="this.placeholder = ""
```

```
onblur="this.placeholder = 'Enter first name'" placeholder='Enter first name'>
```

```
</div>
```

```
</div>
```

```

<div class="col-sm-6">

    <div class="form-group">

        <input class="form-control" name="lname" pattern="[a-zA-Z ]+" required
type="text" onfocus="this.placeholder = ""
        onblur="this.placeholder = 'Enter last name'" placeholder='Enter last name'>

    </div>

</div>

<div class="col-sm-12">

    <div class="form-group">

        <input class="form-control" name="hqualification" pattern="[a-zA-Z ]+" required
type="text" onfocus="this.placeholder = ""
        onblur="this.placeholder = 'Enter highest qualification'" placeholder='Enter highest
qualification'>

    </div>

</div>

<div class="col-sm-12">

    <div class="form-group">

        <p>Enter qualification certificate</p>

        <input class="form-control" name="qcertificate" type="file"
onfocus="this.placeholder = ""
        onblur="this.placeholder = 'Enter qualification certificate'" placeholder='Enter
qualification certificate'>

    </div>

</div>

<div class="col-sm-12">

```

```

        <div class="form-group">

            <input class="form-control" name="phone" pattern="[6789][0-9]{9}"
maxlength="10" required type="text" onfocus="this.placeholder = ""
onblur="this.placeholder = 'Enter phone'" placeholder='Enter phone'>

        </div>

    </div>

    <div class="col-sm-12">

        <div class="form-group">

            <input class="form-control" name="email" pattern="[a-z0-9._%+-]+@[a-z0-9.-]+\.[a-
z]{2,4}$" required type="email" onfocus="this.placeholder = ""
onblur="this.placeholder = 'Enter email address'" placeholder='Enter email address'>

        </div>

    </div>

    <div class="col-sm-12">

        <div class="form-group">

            <input class="form-control" name="password" pattern="^(?=.*[A-Za-
z])(?=.*\d)(?=.*[@$!%*#?&])[A-Za-z\d@$!%*#?&]{8,}$" required type="password"
onfocus="this.placeholder = ""
onblur="this.placeholder = 'Enter password'" placeholder='Enter password'>

        </div>

    </div>

    <div class="col-sm-12">

        <div class="form-group">

            <select class="form-control" name="department" aria-label="Default select example"
style="display: block;

```



width: 100%;

padding: 0.375rem 0.75rem;

font-weight: 400;

line-height: 1.5;

color: #6c757d;

transition: border-color .15s ease-in-out,box-shadow .15s ease-in-out;">

<option selected disabled style="color: #495057;">Choose Department</option>

{% for department in departments %}

<option value="{{ department.id }}">{{ department.department\_name }}</option>

{% endfor %}

</select>

</div>

</div>

<div class="col-12">

<div class="form-group">

<textarea class="form-control w-100" name="address" cols="30" rows="9"

onfocus="this.placeholder = ''" onblur="this.placeholder = 'Enter Address'"

placeholder='Enter Address'></textarea>

</div>

</div>

<div class="col-sm-6">

<div class="form-group">

<p>Doctor available from:</p>

<input class="form-control" name="tfrom" type="time" onfocus="this.placeholder =

'''

```
        onblur="this.placeholder = 'Enter '" placeholder='Enter '>

    </div>

</div>

<div class="col-sm-6">

    <div class="form-group">

        <p>Doctor available to:</p>

        <input class="form-control" name="tto" type="time" onfocus="this.placeholder = '"

            onblur="this.placeholder = 'Enter '" placeholder='Enter '>

    </div>

</div>

<div class="col-sm-12">

    <div class="form-group">

        <input class="form-control" name="fee" pattern="[0-9]+" required type="text"
onfocus="this.placeholder = '"

            onblur="this.placeholder = 'Enter consultation fee'" placeholder='Enter consultation
fee'>

    </div>

</div>

<div class="form-group mt-3" style="text-align: center;">

    <button type="submit" class="button button-contactForm btn_1" style="border-radius:
5px; margin-left: 20px;">Register</button>

</div>

</form>
```

```

    </div>

    </div>

</div>

</section>

{% endblock hospital %}

```

### hospital\_viewappointments.html

```

{% extends 'hospital/hospital_dashboard.html' %}

{% for message in messages %}

<script>

    alert('{{ message }}');

</script>

{% endfor %}

{% block hospital %}

<style>
body {
    background-color: #ffffff;

    font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

}
.gradient-custom {
    background: linear-gradient(to right bottom, rgb(139, 132, 228), rgb(133, 145, 253));

}
.section-header
h3 { font-size:
3rem;    font-
weight: 600;

```

```

color:
#1e263d;
margin-top:
50px;
margin-bottom: 30px;
}
.container {

padding: 30px 15px;
}
.table {

width:      100%;
background-color:
#fff;      border-
radius: 10px;
box-shadow: 0 4px 8px rgba(0, 0, 0,
0.1); margin-top: 30px;
border-collapse: collapse;
}
.table th, .table
td { padding:
15px; text-
align: left;
border-bottom: 1px solid #ddd;
}
.table thead {

background-color:
#4e6ab1; color: #fff;
font-weight: 600;

```

```

    text-transform: uppercase;
}
.table tbody {
    background-color:
    #fff;
}
.table tbody tr:nth-
child(even) {
    background-color:
    #f7f7f7;
}
.table tbody tr:hover {

    background-color: #e9e9e9;
}
.btn-gray {

    background-color:
    #4e6ab1; color: #fff;
    padding: 6px
    12px; border-
    radius: 5px;
    border: none;
    text-decoration:
    none; font-
    weight: 600;
    margin-right:
    8px; display:
    inline-block;
    transition: background-color 0.3s ease;
}

```

```

.btn-gray:hover      {
  background-color:
  #6a8ce8; color: #fff;
  text-decoration: none;
}

.upload-button {

  background-color:
  #4e6ab1; color: #fff;
  padding: 3px
  12px; border-
  radius: 5px;
  border: none;
  text-decoration:
  none; font-
  weight: 600;
  margin-right:
  8px; display:
  inline-block;
  transition: background-color 0.3s ease;
}

h3.text-info
{ font-size:
  3rem;
  font-weight: 600;
  color: #1e263d;
  margin-top:
  50px; margin-
  bottom: 20px;
}

</style>

```

```

<div class="container mt-5">

  <div class="row justify-content-center">

    <div class="col-md-8 text-center">

      <h3 class="text-info">
        </h3>

    </div>

  </div>

  <div class="table-responsive mt-4">

    <table class="table">

      <thead>

        <tr>

          <th>#</th>

          <th>Patient</th>

          <th>Doctor</th>

          <th>Appointment Date</th>

          <th>Status</th>

          <th>Approve/Reject</th>

          <th>Medical Record</th>

          <th>Add MRI Scan</th>

        </tr>

      </thead>

      <tbody>

        { % for data in appointments % }

        <tr>

```

```

<th scope="row">{{ forloop.counter }}</th>

<td>{{ data.patient.fname }} {{ data.patient.lname }}</td>

<td>{{ data.doctor.fname }} {{ data.doctor.lname }}, {{
data.doctor.department.department_name }} <br>{{
data.doctor.hospital.hospital_name
}}</td>

<td>{{ data.app_date }}</td>

<td>

{% if data.app_status == 'Confirm'
%} Paid
{% else %}

{{ data.app_status }}

{% endif %}

</td>

<td>

{% if data.app_status == 'Confirm' %}

<a href="/doctor-markstatus?appointmentid={{ data.id
}}&status=Visited" class="btn-gray" style="margin-bottom: 10px;" ><i
class="far fa-times-circle mr- 2"></i>Accept</a>

<a href="/doctor-markstatus?appointmentid={{ data.id }}&status=Not
Visited" class="btn-gray" ><i class="far fa-check-circle mr-2"></i>Reject</a>

{% else %}

<p>no action</p>

{% endif %}

</td>

<td><a href="/hospital-viewmedicalrecord?appointmentid={{ data.id }}"

```



```

class="btn- gray">View</a></td>
<td>
    {% if data.mri_scan %}
        <a href="{{ data.mri_scan.url }}" target="_blank">View MRI</a>
    {% else %}
        <form method="POST" enctype="multipart/form-data" action="{% url 'upload_mri'
data.id
%}">
            {% csrf_token %}
            <input type="file" name="image" required>
            <input type="hidden" name="appointment_id" value="{{ data.id }}">
            <button type="submit" class="upload-button">Upload MRI Scan</button>
        </form>
    {% endif %}
</td>
</tr>
{% endfor %}
</tbody>
</table>
</div>
</div>
{% endblock hospital %}

<u>doctor_viewmri.html</u>

{% extends 'doctor/doctor_dashboard.html' %}

```

```

{% for message in messages %}

<script>alert('{{ message }}')</script>

{% endfor %}

{% block doctor %}

<style>

  body {

    margin-top: 20px;

  }

  .rounded-3 {

    border-radius: 0.3rem !important;

  }


a, a:active, a:focus
{
    color:
    #616161;

    text-decoration:    none;
    transition: all 0.2s ease-
    in-out;
}

.text-secondary,      .text-secondary-
  hover:hover { color:  #59b73f
  !important;
}

.display-25 {

  font-size: 1.4rem;

}

.text-primary, .text-primary-hover:hover

```

```

    { color: #ff712a !important;
  }
p {
    margin: 0 0 20px;
}
.mb-1-6, .my-1-6 {
    margin-bottom: 1.6rem;
}
.scan-card img {
    max-height:
    300px; object-
    fit: contain;
    margin: 0 auto;
}
.scan-card {
    display:
    flex;
    flex-direction:
    column; align-
    items: center;

    padding:
    20px;
    width:
    100%;
}
.middle-portion {
    background-color: #dae1e7; /* Slightly darker grayish-
    blue */ padding: 40px 0;

```

```

}

.card {

    border: 1px solid
    #ddd;    border-
    radius:    8px;
    background-color:
    #fff;
    box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);

}

.btn-info {

    background-color:
    #017c7b; border-color:
    #017c7b;
}

.btn-info:hover {

    background-color:
    #015f5d; border-color:
    #015f5d;
}

</style>

<div class="container mt-5">

    <div class="row justify-content-center">

        <div class="col-md-8 text-center">

            <h3 class="text-info" style="font-size: 3rem; margin-top: 50px; margin-
bottom: 20px;">

                MRI

            </h3>

```

```

</div>

</div>

</div>

<div class="container middle-portion">

  <div class="row justify-content-center">

    {% if data %}

      {% for d in data %}

        <div class="col-md-6 col-lg-4 mb-4 d-flex justify-content-center">

          <div class="card shadow scan-card text-center">

            <a href="/Reques?id={{ d.id }}" class="btn btn-info mt-2">CHECK</a>

          </div>

        </div>

      {% endfor %}

    {% else %}

      <div class="col-12 text-center">

        <h2>NO FILE UPLOADED</h2>

      </div>

    {% endif %}

  </div>

</div>

{% endblock doctor %}

checkalzhim.html

```

```

{% extends 'doctor/doctor_dashboard.html' %}

{% for message in messages %}

<script>alert('{{ message }}')</script>

{% endfor %}

{% block doctor %}

<style>

    body {

        margin-top: 20px;

        background-color: #e0e6f0; /* Slightly darker gray-blue background */

    }

    .bg-light-blue {

        background-color: #d6ecf8 !important; /* Darker than
        before */ color: #2a6c8d;
        padding: 7px
        18px; border-
        radius: 4px;

    }

    .bg-light-green {

        background-color:  rgba(40, 167, 69, 0.25)
        !important; padding: 7px 18px;
        border-radius: 4px;

        color: #218838 !important;

    }

    .buttons-to-right {

        position:
        absolute; right:
        0;


```

```

    top: 40%;
}
.btn-gray {
    color:
    #444;
    background-color:
    #d9d9d9; padding: 7px
    18px;
    border-radius: 4px;
}

.booking:hover .buttons-to-right .btn-
gray { opacity: 1;
transition: .3s;
}

.buttons-to-right .btn-
gray { opacity: 0;
transition: .3s;
}

.btn-gray:hover {
    background-color:
    #1b7fc3; color: #fff;
}

.booking {
    margin-bottom: 30px;

    border-bottom: 1px solid
    #ccc; padding-bottom:
    30px;
}

```

```

.booking:last-child
{
    margin-
bottom: 0px;
border-bottom:
none; padding-
bottom: 0px;
}
@media screen and (max-width: 575px) {

    .buttons-to-right
    { top: 10%;
    }
    .buttons-to-right a {
        display: block;

        margin-bottom: 20px;
    }
    .bg-light-blue,
    .bg-light-green,
    .btn-gray {
        padding:
        7px;
    }
}
.card {
    margin-bottom:
    20px; background-
color: #ffffff; border-
radius: 4px; border:
none;
padding: 25px;

```



```

        box-shadow: 0 4px 12px rgba(0, 0, 0, 0.08); /* Deeper shadow for a more pro look */
    }
    .mb-5, .my-5 {
        margin-bottom: 3rem !important;
    }
    .msg-img {
        margin-right: 20px;
    }
    .msg-img img
    {
        width:
        60px;
        border-radius: 50%;
    }
    .fs-5 {
        font-size: 1.25rem;
    }

    .content-area {
        background-color: #dce3ed; /* Slightly darker blue-
        gray */ padding-top: 30px;
        padding-bottom: 30px;
    }
</style>

<div class="container mt-5">

    <div class="row justify-content-center">

        <div class="col-md-8 text-center">

            <h3 class="text-info" style="font-size: 3rem; margin-top: 50px; margin-

```

```

bottom: 20px;">
    </h3>

</div>

</div>

</div>

{% if data %}

<div class="container d-flex justify-content-center align-items-center
content-area" style="min-height: 60vh;">
    <div class="col-md-8 col-lg-6">

        <div class="card shadow-lg border-0 rounded-4 p-4 text-center">

            <h4 class="mb-3 text-primary" style="font-weight: 600;">Diagnosis Result</h4>

            <hr class="mb-4">

            <p class="fs-5" style="color: #333;">{{ data }}</p>

        </div>

    </div>

</div>

{% endif %}

<div class="container content-area">

    <div class="row">

        <div class="col-md-12">

<div class="card card-white mb-5">

            <div class="card-heading clearfix border-bottom mb-4"></div>

            <div class="card-body">

<ul class="list-unstyled">

```

```

{% for appointment in appointments %}

<li class="position-relative booking">

  <div class="media">

    <div class="media-body">

      <h5 class="mb-4">

        {{ appointment.patient.fname }} {{ appointment.patient.lname }}

        {% if appointment.app_status == "Pending" %}

          <span class="badge badge-primary mx-3">Pending</span>

          {% elif appointment.app_status == "Approve" %}

            <span class="badge badge-warning">Approve</span>

            {% elif appointment.app_status == "Reject" %}

              <span class="badge badge-danger">Reject</span>

              {% elif appointment.app_status == "Not Visited" %}

                <span class="badge badge-danger">Not Visited</span>

                {% elif appointment.app_status == "Visited" %}

                  <span class="badge badge-success ml-3">Visited</span>

                  {% elif appointment.app_status == "Prescription added" %}

                    <span class="badge badge-success ml-3">Prescription added</span>

                  {% elif appointment.app_status == "Cancel" %}

                    <span class="badge badge-danger">Cancel</span>

                    {% elif appointment.app_status == "Confirm" %}

                      <span class="badge badge-success ml-3">Confirm</span>

                  {% endif %}

      </h5>

```

```

        <div class="mb-3">

        <span class="mr-2 d-block d-sm-inline-block mb-2 mb-sm-
0">Booked On:</span>

        <span class="bg-light-blue">{{ appointment.booked_on }}</span>

        </div>

        <div class="mb-3">

        <span class="mr-2 d-block d-sm-inline-block mb-2 mb-sm-
0">Visiting Time:</span>

        <span class="bg-light-blue">{{ appointment.app_time }}</span>

        </div>

        <div class="mb-3">

        <span class="mr-2 d-block d-sm-inline-block mb-1 mb-sm-
0">Basic Info:</span>
Years</span>
}}</span>
<span class="border-right pr-2 mr-2">{{ appointment.patient.age }}
<span class="border-right pr-2 mr-2"> {{ appointment.patient.email
<span>{{ appointment.patient.phone }}</span>

        </div>

        {% if appointment.paid_on %}

        <div class="mb-3">

        <span class="mr-2 d-block d-sm-inline-block mb-2 mb-sm-
0">Payment on:</span>

        <span class="bg-light-blue">{{ appointment.paid_on }}</span>

        </div>

```

```

    {% endif %}

    {% if appointment.paytype %}

    <div class="mb-3">

        <span class="mr-2 d-block d-sm-inline-block mb-2 mb-sm-
0">Payment type:</span>

        <span class="bg-light-blue">{{ appointment.paytype }}</span>

    </div>

    {% endif %}

    <a href="/doctor-viewmedicalrecord?appointmentid={{
appointment.id }}" class="btn-gray">Medical Record</a>
    </div>

</div>

<div class="buttons-to-right">

    {% if appointment.app_status == "Pending" %}

    <a href="/doctor-rejectappointment?appointmentid={{ appointment.id
}}" class="btn-gray mr-2"><i class="far fa-times-circle mr-2"></i> Reject</a>

    <a href="/doctor-approveappointment?appointmentid={{ appointment.id
}}" class="btn-gray"><i class="far fa-check-circle mr-2"></i> Approve</a>

    {% elif appointment.app_status == "Approve" %}

    <p>Payment action Pending from user</p>

    {% elif appointment.app_status == "Reject" %}

    <p>Rejected by you</p>

    {% elif appointment.app_status == "Cancel" %}

    <p>Cancelled by user</p>

    {% elif appointment.app_status == "Confirm" %}

```

```

        {% if app_date == today %}

        <a href="/doctor-markstatus?appointmentid={{ appointment.id
}}&status=Visited" class="btn-gray mr-2"><i class="far fa-times-circle mr-
2"></i> Visited</a>

        <a href="/doctor-markstatus?appointmentid={{ appointment.id
}}&status=Not Visited" class="btn-gray"><i class="far fa-check-circle mr-2"></i>
Not Visited</a>

        {% else %}

        <p>Confirmed</p>

        {% endif %}

        {% elif appointment.app_status == "Not Visited" %}

        <p>Patient not visited</p>

        {% elif appointment.app_status == "Visited" %}

        <a href="/doctor-addmedical?appointmentid={{
appointment.id }}" class="btn-gray mr-2"><i class="far fa-times-circle mr-
2"></i> Add Prescription</a>

        {% elif appointment.app_status == "Prescription added" %}

        <p>Prescription added</p>

        {% endif %}

    </div>

</li>

{% endfor %}

</ul>

</div>

</div>

```

</div>

</div>

</div>

{% endblock doctor %}

### Views.py

```
from django.shortcuts import render, redirect
from .models import *
from datetime import date, datetime, timedelta
from django.contrib.auth import authenticate
from django.contrib import messages
import alzimers as al

def index(request):
    return render(request, 'index.html')

def users_login(request):
    if request.POST:
        email=request.POST['email']
        password=request.POST['password']
        user=authenticate(username=email,password=password)
        if user is not None:
            if user.user_type=='admin':
                msg=messages.success(request,'Welcome to admin dashboard')
                return redirect('/admin-dashboard')
            elif user.user_type == 'hospital':
                hospital = HospitalReg.objects.get(hospital_log=user) # Fetch the hospital object
                messages.success(request, f'Welcome {hospital.hospital_name} to the hospital
dashboard') # Use f-string for formatting
                request.session['hid'] = hospital.id # Store hospital ID in session
                return redirect('/hospital-dashboard')
            elif user.user_type == 'patient':
                patient = PatientReg.objects.get(patient_log=user) # Fetch the patient object
```

```

        messages.success(request, f'Welcome {patient.fname} {patient.lname} to the patient
dashboard') # Use f-string
        request.session['pid'] = patient.id # Store patient ID in session
        return redirect('/patient-dashboard')

    elif user.user_type == 'doctor':
        doctor = DoctorReg.objects.get(doctor_log=user) # Fetch the doctor object
        messages.success(request, f'Welcome Dr. {doctor.fname} {doctor.lname} to the doctor
dashboard') # Use f-string
        request.session['did'] = doctor.id # Store doctor ID in session
        return redirect('/doctor-dashboard')

    else:
        msg=messages.success(request,'Invalid Login again')
        return redirect('/users-login')

    return render(request,'users_login.html')

def Reques(request):
    data=""
    if request.GET:
        id=request.GET.get("id")
        data=MRI.objects.get(id=id)
        print("*****",type,"*****")
        ss=str("static/media/") +str(data.image)
        data=al.alzhim(ss)

        return render(request,"doctor/checkalzhim.html",{"data":data})

def admin_dashboard(request):
    return render(request,'admin/admin_dashboard.html')

def admin_approvehospital(request):
    hospitals=HospitalReg.objects.filter(hospital_log__is_active=0)
    return render(request,'admin/admin_approvehospital.html',{'hospitals':hospitals})

def admin_approvesingehospital(request):
    hid=request.GET.get('hid')
    hospital=HospitalReg.objects.get(id=hid).hospital_log.id

```



```

hlogin=Login.objects.filter(id=hospital).update(is_active=1)
msg=messages.success(request,'Hospital approved sucessfully')
return redirect('/admin-viewhospitals')
def admin_viewhospitals(request):
    hospitals=HospitalReg.objects.filter(hospital_log__is_active=1)
    return render(request,'admin/admin_viewhospitals.html',{'hospitals':hospitals})
def admin_rejectsinglehospital(request):
    hid=request.GET.get('hid')
    hospital=HospitalReg.objects.get(id=hid).hospital_log.id
    hlogin=Login.objects.filter(id=hospital).update(is_active=0)
    msg=messages.success(request,'Hospital rejected sucessfully')
    return redirect('/admin-viewhospitals')
def admin_viewdoctors(request):
    departments = Department.objects.all().order_by('department_name')
    doctors = DoctorReg.objects.all() # Fetch all doctors, ignoring hospital_id
    if request.method == "POST":
        department_id = request.POST.get('department', "").strip()
        if department_id.isdigit(): # Validate department ID before filtering
            doctors = doctors.filter(department_id=int(department_id))
    return render(request, 'admin/admin_viewdoctors.html', {
        "doctors": doctors,
        "departments": departments
    })
def admin_adddepartments(request):
    if request.POST:
        department=request.POST['department']
        if Department.objects.filter(department_name__iexact=department).exists():
            msg=messages.success(request,'Department already added')
            return redirect('/admin-adddepartments')
        else:
            department=Department.objects.create(department_name=department)

```

```

        department.save()

        msg=messages.success(request,'Department name added sucessfully')

        return redirect('/admin-adddepartments')

departments=Department.objects.all().order_by('department_name')

return render(request,'admin/admin_adddepartments.html',{'departments':departments})

def admin_adddoctor(request):

    departments=Department.objects.all().order_by('department_name')

    hospital=request.session['hid']

    print(hospital)

    if request.POST:

        fname=request.POST['fname']

        lname=request.POST['lname']

        phone=request.POST['phone']

        fee=request.POST['fee']

        image=request.FILES['image']

        email=request.POST['email']

        password=request.POST['password']

        hqualification=request.POST['hqualification']

        qcertificate=request.FILES['qcertificate']

        address=request.POST['address']

        department=request.POST['department']

        tfrom=request.POST['tfrom']

        tto=request.POST['tto']

        if Login.objects.filter(username=email).exists():

            print('exists,,,,,')

            msg=messages.success(request,'Already Taken')

            return redirect('/admin-adddoctor')

        else:

            d_login=Login.objects.create_user(user_type='doctor',view_password=password,username=e

mail,password=password)

            d_login.save()

```

```

        dadd=DoctorReg.objects.create(doctor_log=d_login,hospital_id=hospital,department_id=department,fname=fname,fee=fee,lname=lname,image=image,phone=phone,
                                     email=email,highest_quali=hqualification,quali_certi=qcertificate,doctor
        _address=address,time_from=tfrom,
                                     time_to=tto)

        dadd.save()

        msg=messages.success(request,'Doctor added sucessfully')

        return redirect('/admin-viewdoctors')

    return render(request,'admin/admin_adddoctor.html',{'departments':departments})

def admin_updatedepartment(request):
    did=request.GET.get('did')

    department=Department.objects.get(id=did)

    if request.POST:
        department=request.POST['department']

        department=Department.objects.filter(id=did).update(department_name=department)

        msg=messages.success(request,'Department name updated sucessfully')

        return redirect('/admin-adddepartments')

    return render(request,'admin/admin_updatedepartment.html',{'department':department})

def admin_deletedepartment(request):
    did=request.GET.get('did')

    department=Department.objects.filter(id=did).delete()

    msg=messages.success(request,'Department name deleted sucessfully')

    return redirect('/admin-adddepartments')

def admin_viewappointments(request):
    appointments=Appointment.objects.all().order_by("-id")

    return render(request,'admin/admin_viewappointments.html',{'appointments':appointments})

def admin_viewmedicalrecord(request):
    appointmentid=request.GET.get('appointmentid')

    patient=Appointment.objects.get(id=appointmentid).patient.id

    pdata=PatientReg.objects.get(id=patient)

    prescriptiondatas=Medical_record.objects.filter(appointment__patient_id=patient).order_by('-id')

```

```

print(prescriptiondatas)
medicines=Medicie_details.objects.all()
print(medicines)
tests=Test_details.objects.all()
print(tests)
return
render(request,'admin/admin_viewmedicalrecord.html',{'prescriptions':prescriptiondatas,"medicines"
:medicines,"tests":tests,"pdata":pdata})
def admin_viewprofile(request):
    hospital=request.session['hid']
    print(hospital)
    hdata=HospitalReg.objects.get(id=hospital)
    return render(request,'admin/admin_viewprofile.html',{'data':hdata})
def admin_updateprofile(request):
    hospitalid=request.GET.get('hid')
    hospital=HospitalReg.objects.get(id=hospitalid)
    if request.POST:
        phone=request.POST['phone']
        if 'image' in request.FILES:
            image=request.FILES['image']
        else:
            image=hospital.image
        if 'district' in request.POST:
            district=request.POST['district']
        else:
            district=hospital.district
        address=request.POST['address']
        pin=request.POST['pincode']
        hup=HospitalReg.objects.filter(id=hospitalid).update(image=image,phone=phone,district=distric
t,hospital_address=address,pincode=pin)
        msg=messages.success(request,'Hospital details updated sucessfully')

```

```

        return redirect('/admin-viewprofile')
    return render(request,'admin/admin_updateprofile.html',{'data':hospital})
def hospital_signup(request):
    if request.POST:
        name=request.POST['name']
        phone=request.POST['phone']
        image=request.FILES['image']
        email=request.POST['email']
        password=request.POST['password']
        district=request.POST['district']
        address=request.POST['address']
        pin=request.POST['pincode']
        licence=request.FILES['licence']
        if Login.objects.filter(username=email).exists():
            print('exists,,,,,')
            msg=messages.success(request,'Already Taken')
            return redirect('/')
        else:
            h_login=Login.objects.create_user(user_type='hospital',view_password=password,username=
email,password=password,is_active=0)
            h_login.save()
            hadd=HospitalReg.objects.create(hospital_log=h_login,hospital_name=name,image=image,ph
one=phone,
                                     email=email,district=district,hospital_address=address,pincode=pin,
                                     licence=licence)
            hadd.save()
            msg=messages.success(request,'Hospital added sucessfully, Wait for approval')
            return redirect('/')
    return render(request,'hospital_signup.html')
def hospital_dashboard(request):
    return render(request,'hospital/hospital_dashboard.html')

```

```

def update_doctor(request):
    hospital=request.session['hid']
    doctor_id = request.GET.get('doctor_id')
    availability = request.GET.get('availability')
    print(doctor_id)
    print(availability)
    available=DoctorReg.objects.filter(id=doctor_id).update(available=availability)
    doctors=DoctorReg.objects.filter(hospital_id=hospital)
    departments=Department.objects.all().order_by('department_name')
    print(doctors)
    return
    render(request,'hospital/hospital_viewdoctors.html',{'doctors':doctors,"departments":departments})

def hospital_viewdoctors(request):
    hospital=request.session['hid']
    print(hospital)
    print("hi1")
    departments=Department.objects.all().order_by('department_name')
    if request.POST:
        print("hi2")
        department=request.POST['department']
        doctors=DoctorReg.objects.filter(hospital_id=hospital,department_id=department)
        doctors=DoctorReg.objects.filter(hospital_id=hospital)
        print(doctors)
    return
    render(request,'hospital/hospital_viewdoctors.html',{'doctors':doctors,"departments":departments})

def hospital_viewdocappointments(request):
    doctor=request.GET.get('docid')
    print(doctor)
    appointments=Appointment.objects.filter(doctor_id=doctor).order_by("-id")
    print(appointments,'appointmentsffffffff')
    return render(request,'hospital/hospital_viewdocappointments.html',{'appointments':appointments})

```

```

def hospital_viewappointments(request):
    appointments=Appointment.objects.all().order_by("-id")
    return render(request,'hospital/hospital_viewappointments.html',{'appointments':appointments})
def hospital_viewmedicalrecord(request):
    appointmentid=request.GET.get('appointmentid')
    patient=Appointment.objects.get(id=appointmentid).patient.id
    pdata=PatientReg.objects.get(id=patient)
    prescriptiondatas=Medical_record.objects.filter(appointment__patient_id=patient).order_by('-id')
    print(prescriptiondatas)
    medicines=Medicie_details.objects.all()
    print(medicines)
    tests=Test_details.objects.all()
    print(tests)
    return
render(request,'hospital/hospital_viewmedicalrecord.html',{'prescriptions':prescriptiondatas,"medicin
es":medicines,"tests":tests,"pdata":pdata})
from django.shortcuts import render, redirect
from django.contrib import messages
from .models import Login, PatientReg # Make sure models are correctly imported
def patient_signup(request):
    if request.method == "POST": # Use request.method instead of request.POST
        fname = request.POST['fname']
        lname = request.POST['lname']
        phone = request.POST['phone']
        email = request.POST['email']
        password = request.POST['password']
        address = request.POST['address']
        gender = request.POST['gender']
        age = request.POST['age']
        weight = request.POST['weight']
        if Login.objects.filter(username=email).exists():

```

```

        messages.error(request, 'Email is already taken') # Changed message type to error
        return redirect('/patient-signup')
    else:
        p_login = Login.objects.create_user(user_type='patient', view_password=password,
        username=email, password=password)
        p_login.save()
        padd = PatientReg.objects.create(
            patient_log=p_login, gender=gender, age=age, fname=fname, lname=lname,
            weight=weight, phone=phone, email=email, patient_address=address
        )
        padd.save()
        messages.success(request, 'Patient added successfully') # No need to assign msg
        return redirect('/users-login')
    return render(request, 'patient_signup.html')

def patient_dashboard(request):
    patient_id = request.session.get('pid') # Get patient ID from session
    full_name = "Patient"
    if patient_id:
        try:
            patient = PatientReg.objects.get(id=patient_id)
            full_name = f"{patient.fname} {patient.lname}" if patient.lname else patient.fname
        except PatientReg.DoesNotExist:
            pass
    return render(request, 'patient/patient_dashboard.html', {'patient_name': full_name})

def patient_viewhospitals(request):
    hospitals=HospitalReg.objects.filter(hospital_log__is_active=1)
    return render(request,'patient/patient_viewhospitals.html',{'hospitals':hospitals})

def patient_viewdoctors(request):
    departments = Department.objects.all().order_by('department_name')
    doctors = DoctorReg.objects.filter(available=1) # Fetch all doctors, ignoring hospital_id
    if request.method == "POST":

```



```

    department_id = request.POST.get('department', "").strip()
    if department_id.isdigit(): # Validate department ID before filtering
        doctors = doctors.filter(department_id=int(department_id))
    return render(request, 'patient/patient_viewdoctors.html', {
        "doctors": doctors,
        "departments": departments
    })
def patient_makeappointment(request):
    period_list=[]
    doctorid=request.GET.get('docid')
    print(doctorid)
    bdate = request.GET.get('bdate')
    print(bdate,'bdateIIIIIIII')
    formatted_date = None
    if bdate:
        bdate = bdate.replace(".", "").replace('Sept', 'Sep').replace('March', 'Mar').replace('April',
'Apr').replace('June', 'Jun').replace('July', 'Jul')
        print(bdate)
        try:
            date_obj = datetime.strptime(bdate, '%b %d, %Y')
            formatted_date = date_obj.strftime('%Y-%m-%d')
            print(formatted_date, 'formatted_dateIIIIII')
        except ValueError:
            print("Invalid date format:", bdate)
    doctor=DoctorReg.objects.get(id=doctorid)
    patient=request.session['pid']
    print(patient)
    patient=PatientReg.objects.get(id=patient)
    from datetime import datetime, timedelta
    def split_time_range(start_time, end_time, interval_minutes):
        time_periods = []

```

```

    current_time = start_time
    while current_time <= end_time:
        time_periods.append(current_time)
        current_time += timedelta(minutes=interval_minutes)
    return time_periods
now = datetime.now()
print("TTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTss")
    end_time = now.replace(hour=17, minute=0, second=0, microsecond=0) # Generate for the next
24 hours
print(end_time)
interval_minutes = 30
time_periods = split_time_range(now, end_time, interval_minutes)
period_list = []
for period in time_periods:
    formatted_time = period.strftime("%I:%M %p")
    print(formatted_time, 'bbbbbbbbbbbbbb')
    period_list.append(formatted_time)
if request.POST:
    app_date=request.POST['app_date']
    app_time=request.POST['inlineRadioOptions']
    print(app_date,app_time,'app_')
    if Appointment.objects.filter(doctor=doctor,app_date=app_date,app_time=app_time):
        msg=messages.success(request,'Appointment time already taken')
        return redirect('/patient-viewappointments')
    else:
        app_add=Appointment.objects.create(patient=patient,doctor=doctor,app_date=app_date,app_t
ime=app_time)
        app_add.save()
        #msg=messages.success(request,'Your appoint request has been send, wait for doctors action')
        return redirect('/patient-viewappointments')

```

return

```

render(request, 'patient/patient_makeappointment.html', {"doctor":doctor, "patient":patient, "periods":period_list, "formatted_date":formatted_date})
def patient_viewappointments(request):
    patient=request.session['pid']
    print(patient)
    print('hiiiiiiiiiii')
    appointments=Appointment.objects.filter(patient_id=patient)
    print(appointments)
    return render(request, 'patient/patient_viewappointments.html', {"appointments":appointments})
def patient_cancelappointment(request):
    appointmentid=request.GET.get('appointmentid')
    print(appointmentid)
    appointment=Appointment.objects.filter(id=appointmentid).update(app_status='Cancel')
    msg=messages.success(request, 'You cancelled the appointment')
    return redirect('/patient-viewappointments')
def patient_payfeeappointment(request):
    appointmentid=request.GET.get('appointmentid')
    print(appointmentid)
    appointment=Appointment.objects.get(id=appointmentid)
    if request.POST:
        return redirect('/patient-addcarddetails?appointmentid='+str(appointmentid))
    return render(request, 'patient/patient_payfeeappointment.html', {"appointment":appointment})
def patient_addcarddetails(request):
    appointmentid=int(request.GET.get('appointmentid'))
    if request.POST:
        pdate=date.today()
        ptype='Debit Card'
        appointment=Appointment.objects.filter(id=appointmentid).update(paid_on=pdate, paytype=ptype, app_status="Confirm")
        msg=messages.success(request, 'Payment success, Appointment confirmed')
        return redirect('/patient-viewappointments')

```

```

        return render(request,'patient/patient_addcarddetails.html')
def patient_viewprescription(request):
    appointmentid=request.GET.get('appointmentid')
    print(appointmentid)
    appointmentdata=Appointment.objects.get(id=appointmentid)
    prescriptiondata=Medical_record.objects.get(appointment=appointmentid)
    medicines=Medicie_details.objects.filter(medical_record=prescriptiondata)
    tests=Test_details.objects.filter(medical_record=prescriptiondata)
    return
render(request,'patient/patient_viewprescription.html',{ "appointment":appointmentdata,"prescription"
:prescriptiondata,"medicines":medicines,"tests":tests})
def patient_viewmedicalrecord(request):
    patient=request.session['pid']
    print(patient)
    pdata=PatientReg.objects.get(id=patient)
    if request.POST:
        tres=request.FILES['tupload']
        testid=request.POST['tid']
        print(tres,testid)
        tresult=Test_details.objects.filter(id=testid).update(test_upload=tres)
        msg=messages.success(request,'Test Result uploaded sucessfully')
        prescriptiondatas=Medical_record.objects.filter(appointment__patient_id=patient).order_by('-id')
        print(prescriptiondatas)
        medicines=Medicie_details.objects.all()
        print(medicines)
        tests=Test_details.objects.all()
        print(tests)
    return
render(request,'patient/patient_viewmedicalrecord.html',{ "prescriptions":prescriptiondatas,"medicines
":medicines,"tests":tests,"pdata":pdata})
def doctor_dashboard(request):

```

```

doctor_id = request.session.get('did') # Get doctor ID from session
full_name = "Doctor"
if doctor_id:
    try:
        doctor = DoctorReg.objects.get(id=doctor_id)
        full_name = f"Dr. {doctor.fname} {doctor.lname}" if doctor.lname else f"Dr. {doctor.fname}"
    except DoctorReg.DoesNotExist:
        pass
    return render(request, 'doctor/doctor_dashboard.html', {'doctor_name': full_name})
def doctor_viewappointments(request):
    doctor=request.session['did']
    print(doctor)
    doctor=DoctorReg.objects.get(id=doctor)
    today = date.today()
    print(today,'datemmmmz')
    appointments=Appointment.objects.filter(doctor=doctor,app_date=today,app_status__in=['Visited']
)
    ddate=today
    if request.POST:
        ddate=request.POST['ddate']
        appointments=Appointment.objects.filter(doctor=doctor,app_date=ddate)
    return
    render(request,'doctor/doctor_viewappointments.html',{'appointments':appointments,'app_date':ddate,'
today':today})
def doctor_approveappointment(request):
    appointmentid=request.GET.get('appointmentid')
    print(appointmentid)
    appointment=Appointment.objects.filter(id=appointmentid).update(app_status='Approve')
    msg=messages.success(request,'You approved the appointment wait for users payment')
    return redirect('/doctor-viewappointments')
def doctor_rejectappointment(request):

```

```

appointmentid=request.GET.get('appointmentid')
print(appointmentid)
appointment=Appointment.objects.filter(id=appointmentid).update(app_status='Reject')
msg=messages.success(request,'You rejected the appointment')
return redirect('/doctor-viewappointments')
def doctor_markstatus(request):
    appointmentid=request.GET.get('appointmentid')
    print(appointmentid)
    status=request.GET.get('status')
    print(status)
    appointments=Appointment.objects.all().order_by("-id")
    if status=='Visited':
        appointment=Appointment.objects.filter(id=appointmentid).update(app_status='Visited')
        msg=messages.success(request,'Successfully Approved')
        return render(request,'hospital/hospital_viewappointments.html',{'appointments':appointments})
    elif status=='Not Visited':
        appointment=Appointment.objects.filter(id=appointmentid).update(app_status='Not Visited')
        msg=messages.success(request,'Rejected')
        return render(request,'hospital/hospital_viewappointments.html',{'appointments':appointments})
    return render(request,'hospital/hospital_viewappointments.html',{'appointments':appointments})
def doctor_addmedical(request):
    appointmentid=request.GET.get('appointmentid')
    appointment=Appointment.objects.get(id=appointmentid)
    if request.method == 'POST':
        symptoms = request.POST.get('symptoms')
        remarks = request.POST.get('remarks')
        print(remarks,'remarksd')
        medicine_names = request.POST.getlist('medicine_name')
        medicine_dosages = request.POST.getlist('medicine_dosage')
        medicine_quantitys = request.POST.getlist('medicine_quantity')
        medicine_directionss = request.POST.getlist('medicine_directions')

```

```

medicine_dayss = request.POST.getlist('medicine_days')
test_names = request.POST.getlist('test_name')
bdate = request.POST.get('bdate')
medicines = [{ 'name': name, 'dosage': dosage, 'quantity': quantity, 'directions': directions, 'days':
                days} for name, dosage, quantity, directions, days in zip(medicine_names,
medicine_dosages,
                medicine_quantitys, medicine_directionss, medicine_dayss)]
tests=[{ 'name':name} for name in (test_names)]
print(remarks,medicines,tests,bdate)
if bdate:
    prescription_data=Medical_record.objects.create(appointment=appointment,symptoms=sympt
oms,
                remarks=remarks,next_app=bdate)
    prescription_data.save()
else:
    prescription_data=Medical_record.objects.create(appointment=appointment,symptoms=sympt
oms,
                remarks=remarks)
    prescription_data.save()
medicineslist_length=(len(medicines))
for item in range(medicineslist_length):
    medicine_data=Medicie_details.objects.create(medical_record=prescription_data,m_name=
    medicines[item]['name'],m_dosage=medicines[item]['dosage'],m_quantity=medicines[item]['q
uantity'],
    m_directions=medicines[item]['directions'],m_days=medicines[item]['days'])
    medicine_data.save()
    testslist_length=(len(tests))
    for item in range(testslist_length):
        test_data = Test_details.objects.create(medical_record=prescription_data,
test_name=tests[item]['name'])
        test_data.save()

```

```

        appointment=Appointment.objects.filter(id=appointmentid).update(app_status='Prescription
added')

        msg=messages.success(request,'Prescription added')

        return redirect('/doctor-viewappointments')

        return render(request,'doctor/doctor_addmedical.html')
def doctor_viewmedicalrecord(request):
    appointmentid=request.GET.get('appointmentid')
    patient=Appointment.objects.get(id=appointmentid).patient.id
    pdata=PatientReg.objects.get(id=patient)
    prescriptiondatas=Medical_record.objects.filter(appointment__patient_id=patient).order_by('-id')
    print(prescriptiondatas)
    medicines=Medicie_details.objects.all()
    print(medicines)
    tests=Test_details.objects.all()
    print(tests)
    return
    render(request,'doctor/doctor_viewmedicalrecord.html',{'prescriptions':prescriptiondatas,"medicines"
:medicines,"tests":tests,"pdata":pdata})
    return
    render(request,'patient/patient_viewmedicalrecord.html',{'prescriptions':prescriptiondatas,"medicines
":medicines,"tests":tests,"pdata":pdata})
def upload_mri(request, appointment_id):
    if request.method == 'POST' and request.FILES.get('image'):
        appointment = Appointment.objects.get(id=appointment_id)
        mri_scan = MRI.objects.create(
            appointment=appointment,
            image=request.FILES['image']
        )
        mri_scan.save() # Save the uploaded MRI image to the database
        return redirect('/hospital-viewappointments/') # Redirect to where you want to view the
appointments

```



```
        return HttpResponseRedirect("Failed to upload MRI scan", status=400)
def doctor_viewmri(request):
    id=request.GET.get("appointmentid")
    data=MRI.objects.filter(appointment_id=id)
    return render(request,"doctor/doctor_viewmri.html",{"data":data})
```