

**OneHealth**  
**E-diagnostic laboratory management system**  
*Mini Project Report*

*Submitted by*

**Anjali Raj**

**Reg. No.: AJC22MCA-2019**

*In Partial fulfillment for the Award of the Degree of*

**MASTER OF COMPUTER APPLICATIONS**

**(RMCA)**

**APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY**



**AMAL JYOTHI COLLEGE OF ENGINEERING**

**KANJIRAPPALLY**

[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by AICTE, Accredited by NAAC with 'A' grade. Koovappally, Kanjirappally, Kottayam, Kerala – 686518]

**2023-2024**

**DEPARTMENT OF COMPUTER APPLICATIONS**  
**AMAL JYOTHI COLLEGE OF ENGINEERING**  
**KANJIRAPPALLY**



**CERTIFICATE**

This is to certify that the Project report, “OneHealth” is the bonafide work of **ANJALI RAJ (Regno: AJC22MCA-2019)** in partial fulfillment of the requirements for the award of the Degree of Integrated Master of Computer Applications under APJ Abdul Kalam Technological University during the year 2023-24.

**Dr.Bijimol T.K**

**Internal Guide**

**Ms. Meera Rose Mathew**

**Coordinator**

**Rev. Fr. Dr. Rubin Thottupurathu Jose**

**Head of the Department**

## **DECLARATION**

I hereby declare that the project report “**OneHealth**” is a bona-fide work of done at Amal Jyothi College of Engineering, towards the partial fulfillment of the requirements for the award of the Integrated Master of Computer Applications ( RMCA) from APJ Abdul Kalam Technological University, during the academic year 2023-2024.

**Date:**

**ANJALI RAJ**

**Reg: AJC22MCA-2019**

**KANJIRAPPALLY**

## ACKNOWLEDGEMENT

First and foremost, I thank God almighty for his eternal love and protection throughout the project. I take this opportunity to express my gratitude to all who helped me in completing this project successfully. It has been said that gratitude is the memory of the heart. I wish to express my sincere gratitude to our Manager **Rev. Fr. Dr. Mathew Paikatt** and Principal **Dr. Lillykutty Jacob** for providing good faculty for guidance.

I owe a great depth of gratitude towards our Head of the Department **Rev.Fr.Dr. Rubin Thottupurathu Jose** for helping us. I extend my whole hearted thanks to the project coordinator **Ms. Meera Rose Mathew** for her valuable suggestions and for overwhelming concern and guidance from the beginning to the end of the project. I would also express sincere gratitude to my guide **Dr . Bijimol T.K** for her inspiration and helping hand.

I thank our beloved teachers for their cooperation and suggestions that helped me throughout the project. I express my thanks to all my friends and classmates for their interest, dedication, and encouragement shown towards the project. I convey my hearty thanks to my family for the moral support, suggestions, and encouragement to make this venture a success.

**ANJALI RAJ**

## ABSTRACT

**OneHealth** is an online diagnostic lab management website that brings up various diagnoses working online. Here patients get an option to access the laboratory and register on their site and login using registered details. After registration the patients may now see a variety of tests conducted by the lab along with their costs. The system allows patients to do the bookings for various tests like CBC, Blood Glucose, KFT, LFT, Hemoglobin, WBC, etc. The system allows users to book any test needed in available slots and after successful booking, system calculates costs and allows users to pay online. After this the patient may service in 2 ways: One can go to laboratory for sample extraction or lab may now collect samples from patient's registered address by staffs. After successful testing, the patient gets a notification of test result in their profile. Also the second function of the system is to provide an e-commerce system for purchasing of health monitoring devices which is implemented on main project.

# CONTENT

SL. NO	TOPIC	PAGE NO
<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
<b>1.1</b>	<b>PROJECT OVERVIEW</b>	<b>2</b>
<b>1.2</b>	<b>PROJECT SPECIFICATION</b>	<b>2</b>
<b>2</b>	<b>SYSTEM STUDY</b>	<b>4</b>
<b>2.1</b>	<b>INTRODUCTION</b>	<b>5</b>
<b>2.2</b>	<b>EXISTING SYSTEM</b>	<b>5</b>
<b>2.3</b>	<b>DRAWBACKS OF EXISTING SYSTEM</b>	<b>6</b>
<b>2.4</b>	<b>PROPOSED SYSTEM</b>	<b>7</b>
<b>2.5</b>	<b>ADVANTAGES OF PROPOSED SYSTEM</b>	<b>7</b>
<b>3</b>	<b>REQUIREMENT ANALYSIS</b>	<b>9</b>
<b>3.1</b>	<b>FEASIBILITY STUDY</b>	<b>10</b>
<b>3.1.1</b>	<b>ECONOMICAL FEASIBILITY</b>	<b>10</b>
<b>3.1.2</b>	<b>TECHNICAL FEASIBILITY</b>	<b>10</b>
<b>3.1.3</b>	<b>BEHAVIORAL FEASIBILITY</b>	<b>11</b>
<b>3.1.4</b>	<b>FEASIBILITY STUDY QUESTIONNAIRE</b>	<b>12</b>
<b>3.2</b>	<b>SYSTEM SPECIFICATION</b>	<b>15</b>
<b>3.2.1</b>	<b>HARDWARE SPECIFICATION</b>	<b>15</b>
<b>3.2.2</b>	<b>SOFTWARE SPECIFICATION</b>	<b>15</b>
<b>3.3</b>	<b>SOFTWARE DESCRIPTION</b>	<b>16</b>
<b>3.3.1</b>	<b>DJANGO</b>	<b>16</b>
<b>3.3.2</b>	<b>SQLITE</b>	<b>16</b>
<b>4</b>	<b>SYSTEM DESIGN</b>	<b>17</b>
<b>4.1</b>	<b>INTRODUCTION</b>	<b>17</b>
<b>4.2</b>	<b>UML DIAGRAM</b>	<b>18</b>
<b>4.2.1</b>	<b>USE CASE DIAGRAM</b>	<b>19</b>
<b>4.2.2</b>	<b>SEQUENCE DIAGRAM</b>	<b>20</b>
<b>4.2.3</b>	<b>STATE CHART DIAGRAM</b>	<b>22</b>
<b>4.2.4</b>	<b>ACTIVITY DIAGRAM</b>	<b>23</b>
<b>4.2.5</b>	<b>CLASS DIAGRAM</b>	<b>24</b>
<b>4.2.6</b>	<b>OBJECT DIAGRAM</b>	<b>25</b>

<b>4.2.7</b>	<b>COMPONENT DIAGRAM</b>	<b>26</b>
<b>4.2.8</b>	<b>DEPLOYMENT DIAGRAM</b>	<b>27</b>
<b>4.3</b>	<b>USER INTERFACE DESIGN USING FIGMA</b>	<b>29</b>
<b>4.4</b>	<b>DATABASE DESIGN</b>	<b>32</b>
<b>5</b>	<b>SYSTEM TESTING</b>	<b>39</b>
<b>5.1</b>	<b>INTRODUCTION</b>	<b>40</b>
<b>5.2</b>	<b>TEST PLAN</b>	<b>40</b>
<b>5.2.1</b>	<b>UNIT TESTING</b>	<b>41</b>
<b>5.2.2</b>	<b>INTEGRATION TESTING</b>	<b>41</b>
<b>5.2.3</b>	<b>VALIDATION TESTING</b>	<b>42</b>
<b>5.2.4</b>	<b>USER ACCEPTANCE TESTING</b>	<b>42</b>
<b>5.2.5</b>	<b>AUTOMATION TESTING</b>	<b>42</b>
<b>5.2.6</b>	<b>SELENIUM TESTING</b>	<b>43</b>
<b>6</b>	<b>IMPLEMENTATION</b>	<b>52</b>
<b>6.1</b>	<b>INTRODUCTION</b>	<b>53</b>
<b>6.2</b>	<b>IMPLEMENTATION PROCEDURE</b>	<b>53</b>
<b>6.2.1</b>	<b>USER TRAINING</b>	<b>54</b>
<b>6.2.2</b>	<b>TRAINING ON APPLICATION SOFTWARE</b>	<b>54</b>
<b>6.2.3</b>	<b>SYSTEM MAINTENANCE</b>	<b>55</b>
<b>7</b>	<b>CONCLUSION &amp; FUTURE SCOPE</b>	<b>56</b>
<b>7.1</b>	<b>CONCLUSION</b>	<b>57</b>
<b>7.2</b>	<b>FUTURE SCOPE</b>	<b>57</b>
<b>8</b>	<b>BIBLIOGRAPHY</b>	<b>58</b>
<b>9</b>	<b>APPENDIX</b>	<b>60</b>
<b>9.1</b>	<b>SAMPLE CODE</b>	<b>61</b>
<b>9.2</b>	<b>SCREEN SHOTS</b>	<b>97</b>

## List of Abbreviation

IDE	Integrated Development Environment
HTML	Hyper Text Markup Language.
CSS	Cascading Style Sheet
SQLite	Relational Database Management System
UML	Unified Modeling Language
AJAX	Asynchronous JavaScript and XML
JS	Java Script



# **CHAPTER 1**

## **INTRODUCTION**

## 1.1 PROJECT OVERVIEW

The Online Diagnostic Lab Management System is a comprehensive web platform designed to streamline the process of diagnostic testing and health monitoring for patients. This system seamlessly integrates laboratory services, test bookings and sample collection. Users can register on the platform, access a range of diagnostic tests, book appointments, and receive test results, providing a user-friendly and efficient healthcare management experience.

## 1.2 PROJECT SPECIFICATION

The OneHealth is a web application designed to facilitate seamless interactions between Admin and Customers. It provides a comprehensive set of functionalities for an efficient online laboratory test booking experience.

### ✧ **Lab Management:**

Admin can add, update, or remove Test to manage inventory.

### ✧ **Details of Registered users:**

Admin accesses and reviews detailed information about registered members.

### ✧ **User Authentication:**

Admin enjoys secure access through a login system.

### ✧ **Payment Processing:**

Secure and user-friendly payment gateway for convenient purchases.

### ✧ **Mode of Sample collection**

Efficient way of sample collection by choosing laboratory or as home service.

### ✧ **Staff Management**

Admin can add staff and make a flexible interaction between them.

✧ **Technologies Used:**

Frontend: HTML, CSS, JavaScript, AJAX.

Backend: Django.

Database: SQLite.

Payment Gateway Integration: Razor pay.

## **CHAPTER 2**

### **SYSTEM STUDY**

## **2.1 INTRODUCTION**

In the dynamic landscape of healthcare, the Online Diagnostic Lab Management System, known as OneHealth, emerges as a pivotal solution that embraces the digital revolution to revolutionize diagnostic services. As the traditional paradigms of healthcare shift towards online platforms, OneHealth is strategically crafted to bridge the gap between patients and diagnostic laboratories. This innovative web application is dedicated to optimizing the diagnostic journey, providing a seamless interface for patients to access a myriad of tests, book appointments, and receive results, all within the convenience of a digital ecosystem.

## **2.2 EXISTING SYSTEM**

The current online booking laboratory system in our locality represents a significant advancement in healthcare accessibility and management. This digital platform allows users to register seamlessly over the phone, streamlining the process of scheduling diagnostic tests. The system features a user-friendly interface during telephone interactions, enabling patients to explore a diverse range of tests, understand associated costs, and efficiently book appointments in available time slots. The integration of secure online payment processing ensures a hassle-free financial transaction experience, offering comprehensive support for those who choose to book appointments via telephone. Patients have the flexibility to opt for sample collection by visiting the laboratory or by having staff members collect samples at their registered address, all initiated through a telephone call. The online booking laboratory system prioritizes user convenience, real-time notifications, and personalized profiles, reflecting a commitment to efficient and transparent healthcare services tailored to the contemporary needs of a digitalized healthcare landscape.

### **2.2.1 NATURAL SYSTEM STUDIED**

The natural laboratory management system recognizes and addresses the existing challenges in the traditional laboratory booking and result retrieval process, emphasizing the urgent need for a more patient-centric and streamlined approach. Aiming to alleviate the cumbersome experience of waiting in laboratories and the inconvenience of finding an alternative when labs are busy, this system prioritizes efficiency and accessibility. In contrast to the requirement for patients to physically visit the lab for bill retrieval and the absence of developed home services in conventional practices, the natural laboratory management system offers transformative

alternatives. The integration of health monitoring devices and the facilitation of home-based routine checkups emerge as progressive solutions, allowing individuals to conduct daily health assessments at home. This not only eases the burden on laboratories but also encourages a more convenient and proactive healthcare approach. Such innovations signify a significant evolution in healthcare services, placing a heightened emphasis on efficiency, well-being, and patient comfort.

### **2.2.2 DESIGNED SYSTEM STUDIED**

The meticulously designed online laboratory system stands as a testament to the future of healthcare management. With a focus on user-centricity, this digital platform has redefined the way diagnostic services are accessed and experienced. Seamlessly integrating modern technology, the system enables users to effortlessly register online, navigate through a diverse array of tests, understand associated costs, and conveniently book appointments in available time slots. The introduction of secure online payment processing ensures a hassle-free financial transaction experience, eliminating the need for in-person payments. Offering flexibility, users can choose between visiting the laboratory or opting for sample collection at their registered address, all with a few clicks. Real-time notifications and personalized profiles further enhance the efficiency and transparency of healthcare services. This innovatively designed online laboratory system not only simplifies the diagnostic journey but also reflects a commitment to shaping a healthcare experience that is efficient, accessible, and tailored to the evolving needs of a digitalized world.

### **2.3 DRAWBACKS OF EXISTING SYSTEM**

- ✧ Time Consuming: The current system involves significant waiting times in laboratories and moving from one lab to another, making the overall healthcare process time-consuming for patients.
- ✧ Expensive Health Monitoring Device Purchases: The system becomes costly, particularly when individuals purchase health monitoring devices from traditional shops, leading to higher healthcare expenses.
- ✧ Additional Fuel Cost: The requirement to physically visit labs or shops introduces an extra expense in terms of fuel cost, impacting the overall financial burden on patients.
- ✧ Less User-Friendly: The existing system is perceived as less user-friendly, potentially

causing confusion and inconvenience for individuals navigating through the complex processes.

- ✧ Waiting Between Labs: Patients often need to wait from lab to lab, adding to the time consumption and posing challenges to accessibility and patient comfort.
- ✧ Limited Payment Options
- ✧ Insufficient Personalization
- ✧ Complex Checkout Process

## 2.4 PROPOSED SYSTEM

OneHealth is poised to revolutionize healthcare management with its state-of-the-art E-Laboratory system, offering an unprecedented level of efficiency, accessibility, and user satisfaction. This innovative platform envisions a seamless healthcare journey for users, eliminating the existing demerits of traditional systems. The proposed system prioritizes time efficiency by providing swift and convenient online booking options, eradicating the need for extensive waiting periods in labs. With cost-effectiveness in mind, OneHealth introduces budget-friendly options for purchasing health monitoring devices directly through the platform, reducing financial burdens on individuals. The system's integration of multiple sample collection modes, including doorstep collection, ensures flexibility and convenience for users, mitigating additional fuel costs associated with physical visits. Boasting an intuitive and user-friendly interface, OneHealth simplifies the entire process from booking to result retrieval, making healthcare management more accessible to a diverse user base. With OneHealth, the future of e-laboratories is envisioned as a seamless, efficient, and user-centric healthcare solution.

## 2.5 ADVANTAGES OF PROPOSED SYSTEM

- ✧ Reduced Expenses: By offering cost-effective options for health monitoring device purchases and minimizing traditional costs associated with physical lab visits, OneHealth strives to make healthcare more affordable for individuals.
- ✧ No Fuel Cost: With the elimination of the need for patients to physically travel to laboratories, the OneHealth system significantly reduces or eliminates fuel costs, contributing to both cost savings and environmental sustainability.

- ✧ Flexible Health Monitoring Device Purchases: OneHealth empowers users by allowing them to purchase health monitoring devices at any time, providing convenience and flexibility.
- ✧ 24/7 Booking Availability: The proposed system breaks free from conventional time constraints by offering round-the-clock booking availability. Users can schedule diagnostic tests at any time, choosing from available slots, enhancing accessibility, and accommodating diverse schedules.
- ✧ User-Friendly Interface: OneHealth places a strong emphasis on user-friendliness, ensuring that the interface is intuitive and easy to navigate.
- ✧ Comprehensive Health Record Access: One of the standout features of OneHealth is its capability to provide users with access to their complete test details, both present and past.
- ✧ payment options. This instills trust in users, addressing concerns related to online payment security and encouraging more confident purchasing behavior
- ✧ Flexible Location Selection: OneHealth empowers users with the flexibility to choose their preferred location for diagnostic services. Whether it's a specific laboratory branch or the comfort of their home, individuals can effortlessly select the location that best suits their needs, enhancing convenience and accessibility.
- ✧ Effortless New Address Registration: For users with new addresses or changes in location, OneHealth streamlines the process of updating and registering new addresses. This ensures that individuals can seamlessly transition to a new address without disruptions in sample collection or result notifications, promoting a smooth and user-centric experience.



## **CHAPTER 3**

### **REQUIREMENT ANALYSIS**

### **3.1 FEASIBILITY STUDY**

Feasibility is the degree to which a project can be carried out successfully. A feasibility study is conducted to assess the solution's viability, which establishes whether it is viable and implementable in the program. The feasibility study considers details like the availability of resources, software development costs, the advantages of the software to the business once it is built, and the costs associated with maintaining it. The outcome of the feasibility study should be a report recommending whether the requirements engineering, and system development process should be continued. A system is of no real value to a corporation if it does not serve its goals. Even though this may seem obvious, many organizations create systems that do not support their goals, either because they lack a clear statement of these goals, because they fail to specify the system's business requirements, or because other organizational or political factors have an impact on the procurement of the system.

#### **3.1.1 ECONOMIC FEASIBILITY**

The economic feasibility of an e-diagnostic laboratory management system, such as the proposed OneHealth, is marked by its potential to generate substantial cost savings and optimize resource utilization. By reducing expenses associated with physical infrastructure, paper documentation, and manual processes, the system inherently contributes to operational efficiency. The elimination of fuel costs for patients, streamlined processes for health monitoring device purchases, and flexible location choices for home services all work synergistically to minimize financial burdens on both healthcare providers and users. Moreover, the investment in technology infrastructure can lead to long-term cost-effectiveness through automated processes, reduced administrative overheads, and enhanced overall resource allocation. The economic feasibility of this system is not just a reflection of immediate cost savings but also a strategic investment in a sustainable and technologically advanced healthcare management solution, creating value for both providers and users alike.

#### **3.1.2 TECHNICAL FEASIBILITY**

The technical feasibility of your online bookstore project is crucial to determine whether it can be practically implemented from a technological standpoint. Evaluating various aspects is essential for its success. Firstly, ensure that your organization has the necessary technology infrastructure or can acquire it to support the platform, including web hosting, databases, and server capacity. Check for skilled software developers who can build and maintain the system

effectively. Integration with third-party services like payment gateways and shipping providers must be seamless. The architecture should be scalable to accommodate increasing traffic and data as the platform grows. Security measures to protect user data and transactions must be robust. Ensure the platform is mobile-friendly for users on various devices, and its performance is fast and responsive. Compliance with data protection and privacy regulations is essential. Lastly, evaluate the financial resources required for development, hosting, maintenance, and support. A comprehensive assessment of technical feasibility will help identify potential challenges and enable you to make informed decisions to create a successful online bookstore platform

### **3.1.3 BEHAVIORAL FEASIBILITY**

The behavioral feasibility of the proposed OneHealth e-diagnostic laboratory management system is pivotal for its successful integration into the healthcare landscape. To ensure its viability and effectiveness, several key considerations must be evaluated. Firstly, assess the availability of essential resources, including skilled human capital, necessary technology, and infrastructure within the project's allocated time frame and budget. Recognize the existing skill set of healthcare professionals and identify potential needs for additional training or recruitment. Evaluate how seamlessly OneHealth will integrate with current healthcare processes, encompassing sample collection, result retrieval, and patient management. Proactively plan for change management strategies to mitigate potential resistance and foster a smooth transition. Confirm the compatibility of third-party services and suppliers with operational requirements. Ensure compliance with legal and regulatory frameworks governing healthcare data and patient privacy. Solicit user feedback to gauge acceptance and pinpoint areas for improvement, ensuring that the system aligns with the behavioral expectations of both healthcare providers and patients. Finally, conduct a comprehensive analysis of operational costs to guarantee harmony with the healthcare organization's budget. A meticulous examination of these behavioral feasibility aspects will be instrumental in ensuring the successful integration of the OneHealth platform into the existing healthcare operations.

### **3.1.4 FEASIBILITY STUDY QUESTIONNAIRE**

#### **1.Project Overview?**

The Online Diagnostic Lab Management System is a comprehensive web platform designed to streamline the process of diagnostic testing and health monitoring for patients. This system seamlessly integrates laboratory services, test bookings and sample collection. Users can register on the platform, access a range of diagnostic tests, book appointments, and receive test results, providing a user-friendly and efficient healthcare management experience.

#### **2.To what extend the system is proposed for?**

The proposed OneHealth system is designed to address the comprehensive spectrum of diagnostic healthcare services, offering a holistic solution that goes beyond the limitations of traditional systems. OneHealth aims to streamline the entire diagnostic process, from test bookings to result notifications, making it a full-service e-diagnostic laboratory management system. This system is not confined solely to the booking aspect but extends to diverse features, including flexible location choices, the purchase of health monitoring devices at any time, and home-based routine checkups. It caters to the diverse needs and preferences of users, providing a 24/7 booking availability and a user-friendly interface. By offering these extensive functionalities, OneHealth aspires to transform and enhance the overall healthcare experience, promoting accessibility, efficiency, and user empowerment.

#### **3.Specify the Viewers/Public which is to be involved in the System?**

OneHealth is designed to serve a diverse range of users within the healthcare ecosystem, creating a user-friendly platform for both healthcare providers and patients. Patients seeking diagnostic services, caregivers, and healthcare professionals such as lab technicians and physicians form a crucial part of the user base. Administrative staff managing appointments and billing, healthcare administrators overseeing operations, and technology support staff ensuring system maintenance are also included. Additionally, third-party service providers and regulatory authorities are part of the audience. This inclusive approach aims to provide a seamless and accessible healthcare experience, making OneHealth a valuable tool for a broad spectrum of users involved in the healthcare journey.

#### **4.List the Modules included in your System?**

✧ Admin Module:

- User Management: Handling permissions, appointment details.
- Laboratory Management: Overseeing lab operations and test services.
- Booking Management: Managing appointment bookings, secure payments.
- Staff Management: Handling staff personnel details.

✧ Patient Module:

- Registration and Login: User account creation and authentication.
- Appointment Booking: Browsing available services, scheduling appointments, and making secure payments.
- Account Management: Updating personal information.

✧ Staff Module:

- Sample Collection: Confirming and managing sample collection from patient addresses.
- Appointment Management: Viewing and managing patient appointments.

#### **5. Identify the users in your project?**

The Readify book involves three types of users: admin,staff and patients.

#### **6. Who owns the system?**

Administrator

#### **7. System is related to which firm/industry/organization?**

The OneHealth system is related to the healthcare and diagnostics industry. It is designed to streamline and enhance the diagnostic laboratory management process, providing online access for patients to book appointments, receive test results, and purchase health monitoring devices. The system caters to healthcare providers, administrative staff, patients, and delivery personnel, offering a comprehensive solution for managing diagnostic services and e-commerce of health monitoring devices. With a focus on improving accessibility, efficiency, and user empowerment in the healthcare domain, OneHealth is tailored for individuals seeking convenient and effective healthcare management solutions.

**8. Details of person that you have contacted for data collection.**

Name: Arya P

Position: Lab Assistant, Elister Healthcare, Kollam Sasthamcotta

**9. Questionnaire to collect details about the project?**

a) What are the prime details collected from the patients?

Name, Age, Mobile no and which test need to be test.

b) How there details are filed and included in the laboratory?

Details are stored using different softwares (named “grapes”).

c) On What basis the patients are treated?

One who is registered with the lab and has a file .For their details and booking requirements MR no: is needed. It is a 6 digit number. Instead of MR no: Mobile phone number can be used which is used at the registration time. MR no: is a permanent number. By that no: complete test details of the patient can be viewed by the lab manger.

One who is not a regular customer they provided a bill with OP no:.. It is a temporary 4 digit no:.. and always change.

d) How often the results of test are issued?

Based on the situation the results are issued. That is for an early morning lab test almost 1 hr waiting is requested to patients for their result. Other cases manage it by based on test. For outside testing reports it might takes days for getting results.

e) How the results are informed?

Through email or whatsapp. The test result updated by the lab technicians can be automatically updated in the reception system also. By then the receptionist issue reports.

f) How the testing procedure is scheduled and managed?

One who come to lab for testing first they need to take the bill from reception. Based on that billed details the lab technicians guide the patients. The report is only issued after the bill registration scenario.

g) How the payment is managed? Is it necessary for paying complete bill for getting report?

By the wish of patient, one can pay a minimum amount at beginning of billing or not and after the complete test they can pay the balance amount.

If someone not able to pay the complete amount and they need the report urgently they will included in the pending bill. Pending will always be red marked.

h) How the home service is managed?

The patient enquires to the lab through phone call for home service and the name, address, place, test details are collected from the patients. The amount is combined with the testing amount and service charge. For minimum distances service charges are not taken.

i) How their bill are managed?

After taking the samples, their bill is to be included in the pending section. The result is issued only after complete payment.

j) What are the basic requirements needed for home services?

The home services is based on customer priority. But the time schedule and staff availability is also analysed for it.

k) working time of labs?

24 hrs. Staff are scheduled with regular interval of days and times.

## 3.2 SYSTEM SPECIFICATION

### 3.2.1 Hardware Specification

Processor Intel core i5

RAM 8GB

Hard disk 512 SSD

### 3.2.2 Software Specification

Front End HTML5, CSS, Bootstrap, Vuejs

Back End Django, SQLite

Database SQLite

Client on PC Windows 7 and above.

Technologies used Django, HTML5, AJAX, Bootstrap, JS, jQuery

### **3.3 SOFTWARE DESCRIPTION**

#### **3.3.1 DJANGO**

Django, a leading open-source web framework, epitomizes the pinnacle of modern web development with its efficiency and versatility. Built on Python, Django follows the Model-View-Controller architecture, prioritizing simplicity and flexibility. Noteworthy features include its Object-Relational Mapping system, streamlined URL routing, and a user-friendly template engine. The built-in admin interface simplifies data management, while robust security measures and middleware support enhance application integrity. Django scales effortlessly, accommodating growing datasets and traffic demands. Its REST Framework extension further extends its utility to API development. Supported by an active community and extensive documentation, Django stands as a powerful toolkit, seamlessly shaping the development of dynamic and secure web applications for diverse purposes, from content management systems to Restful API. In essence, Django empowers developers with a comprehensive and adaptable framework for crafting sophisticated and salable web solutions.

#### **3.3.2 SQLite**

SQLite, a lightweight and server less relational database management system, is celebrated for its simplicity, portability, and efficiency. Operating as a self-contained, single-file database, SQLite requires minimal setup and eliminates the need for a separate server process. This design makes it a preferred choice for embedded systems, mobile applications, and small to medium-scale projects.

With zero configuration and a server less architecture, SQLite streamlines the database management process. It supports ACID transactions, ensuring data integrity and reliability, even in the face of system interruptions. The dynamic typing feature allows flexible data storage, accommodating various data types within the same column.

Noteworthy is SQLite's cross-platform compatibility, making it versatile across different operating systems and environments. Its wide adoption is evident in applications ranging from mobile apps to web browsers, where it is utilized for local data storage. SQLite stands out as a go-to solution for projects requiring a lightweight, self-contained, and easily deployable database system, embodying simplicity without compromising on functionality and reliability.



## **CHAPTER 4**

### **SYSTEM DESIGN**

## 4.1 INTRODUCTION

Any designed system or product's development process begins with the design phase. An efficient system depends on a well-executed design, which is a creative process. It entails utilizing a variety of techniques and concepts to completely specify a procedure or system for it to be implemented. Regardless of the development paradigm used in software engineering, the design phase is essential. It seeks to produce the architectural detail needed to design a system or product and acts as the process's technical backbone.

To maximize every aspect of efficiency, performance, and accuracy, this program underwent a comprehensive design phase. A user-oriented document is converted into a document for programmers or database workers throughout the design phase.

## 4.2 UML DIAGRAM

Software engineering uses the Unified Modeling Language (UML), a standardized visual language, to model, develop, and document software systems. UML diagrams provide a concise and organized approach to represent different facets of a software system, serving as a common communication tool for developers, stakeholders, and designers. There are many different varieties of UML diagrams, including class diagrams, sequence diagrams, use case diagrams, and more. Each is designed to communicate a particular piece of knowledge about the design, operation, and interactions of the system. UML diagrams are essential to the software development process because they help with the visualization, analysis, and design of complex systems, which in turn makes the process more effective and efficient.

- Use case diagram
- Sequence diagram
- State diagram
- Activity diagram
- Class diagram
- Object diagram
- Component diagram
- Deployment diagram

### 4.2.1 USE CASE DIAGRAM

A use case diagram is a visual representation of the interactions between system components. An approach for identifying, outlining, and organizing system requirements is called a use case. The word "system" here refers to a thing that is being created or run, like a website for mail-order product sales and services. UML (Unified Modeling Language), a standard language for the modelling of real-world objects and systems, uses use case diagrams. The planning of general requirements, the validation of a hardware design, the testing and debugging of a software product in development, the creation of an online help reference, or the completion of a job focused on customer support are all examples of system objectives. For instance, use cases in a product sales context can involve customer service, item ordering, catalogue updating, and payment processing. There are four elements in a use case diagram.

- The boundary, which defines the system of interest in relation to the world around it.
- The actors, usually individuals involved with the system defined according to their roles.
- The use cases, which are the specific roles are played by the actors within and around the system.
- The relationships between and among the actors and the use cases. Use case diagrams are drawn to capture the functional requirements of a system. After identifying the above items, we must use the following guidelines to draw an efficient use case diagram.
- The name of a use case is very important. The name should be chosen in such a way so that it can identify the functionalities performed.
- Give a suitable name for actors.
- Show relationships and dependencies clearly in the diagram.
- Do not try to include all types of relationships, as the main purpose of the diagram is to identify the requirements.
- Use notes whenever required to clarify some important points.

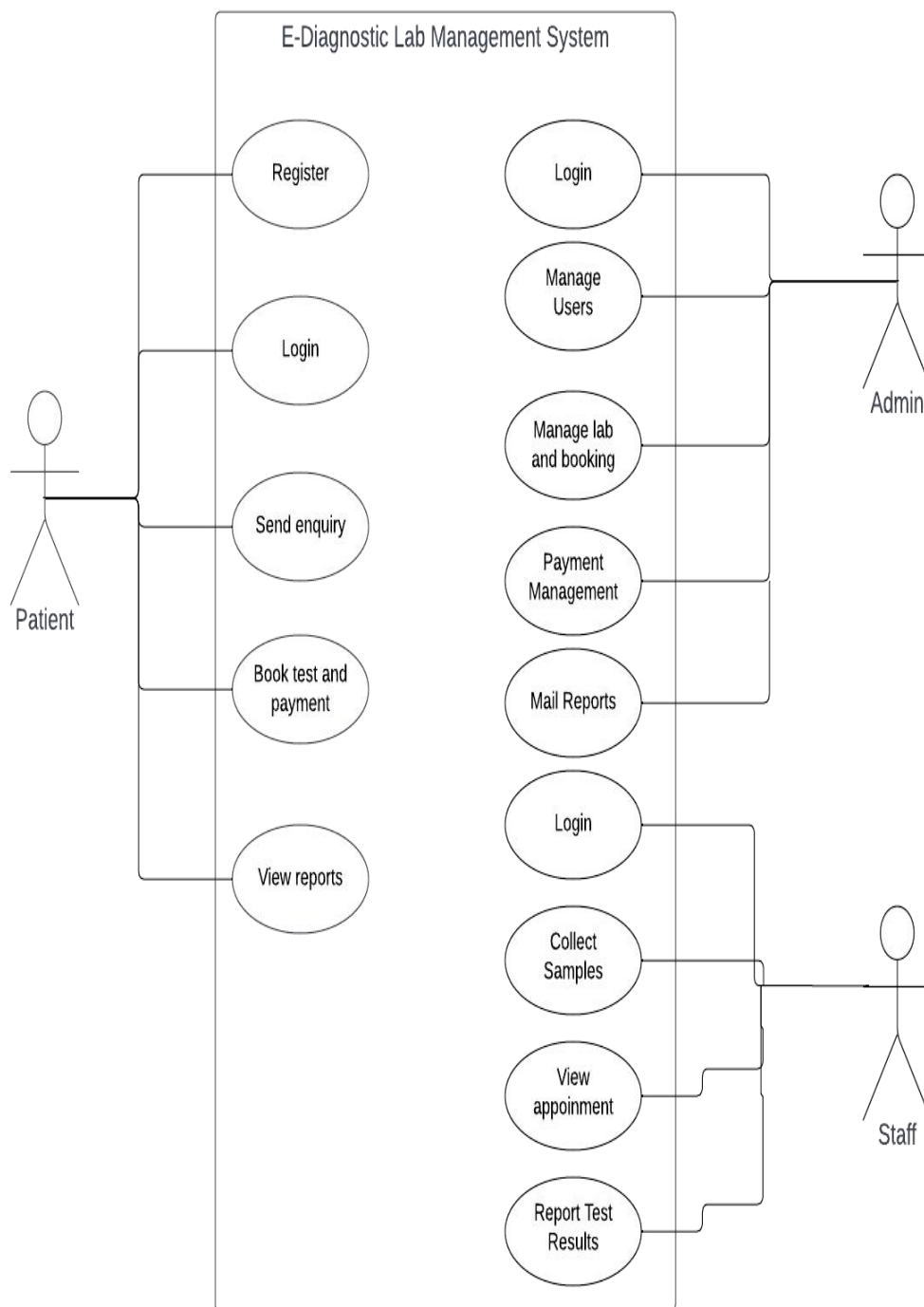


Figure 1: Use Case diagram

#### 4.2.2 SEQUENCE DIAGRAM

A sort of Unified Modeling Language (UML) diagram called a sequence diagram is used in software engineering to depict the communications and interactions that take place across time between various system components or objects. It gives a live view of the interactions between

objects as they work together to complete a certain job or situation.

Objects are shown as lifelines in a sequence diagram, and vertical lines indicate their existence over time. The flow of calls or messages between objects is shown by arrows and lines between lifelines. These diagrams are very helpful for demonstrating the interactions within a particular use case, assisting in determining the sequence of operations and the functions that each object performs.

Sequence diagrams are essential for understanding the behavior of a system and for ensuring that the software functions as intended, as they offer a clear and detailed depiction of how different components or objects collaborate to achieve a particular functionality.

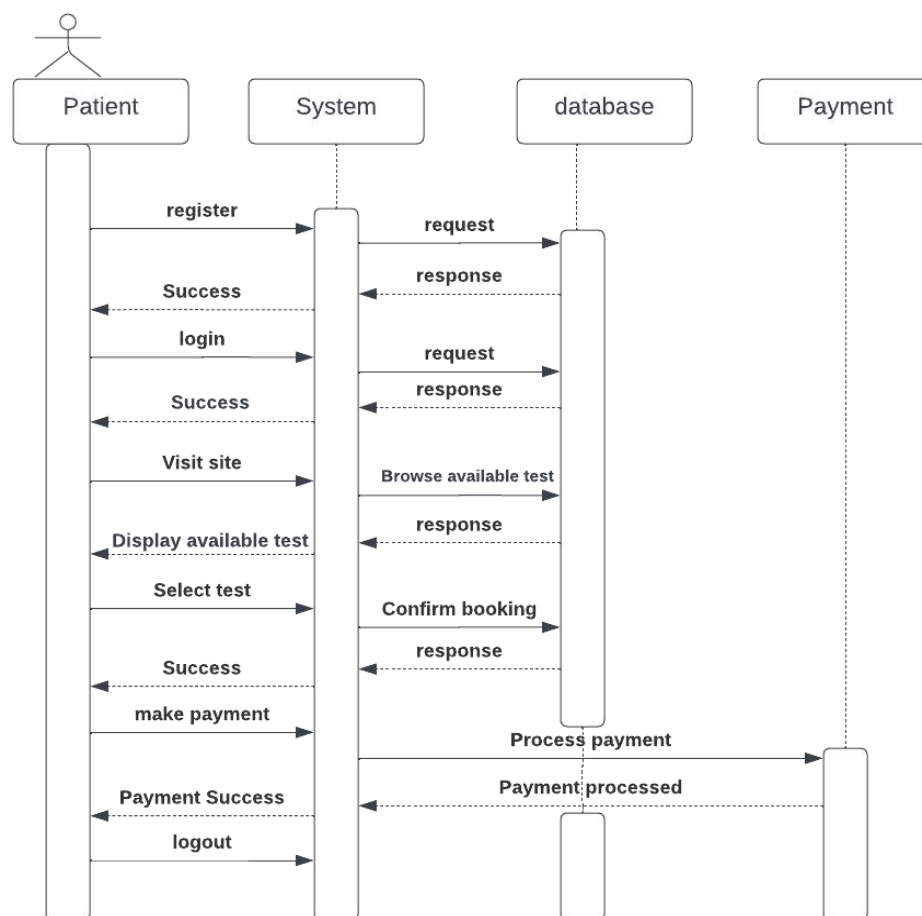


Figure 2: Sequence diagram

### 4.2.3 STATE CHART DIAGRAM

A state diagram, also known as a state machine diagram or state chart diagram, is an illustration of the states an object can attain as well as the transitions between those states in the Unified Modeling Language (UML). In this context, a state defines a stage in the evolution or behavior of an object, which is a specific entity in a program or the unit of code representing that entity. A state diagram resembles a flowchart in nature; however, a flowchart shows the processes within a system that alters the state of an object rather than the actual state changes themselves. The first step to creating a state chart diagram is identifying the initial and final states of a system. Then, all the possible existing states are placed in relation to the beginning and the end. Lastly, all of the events that trigger state changes are labeled as transition elements.

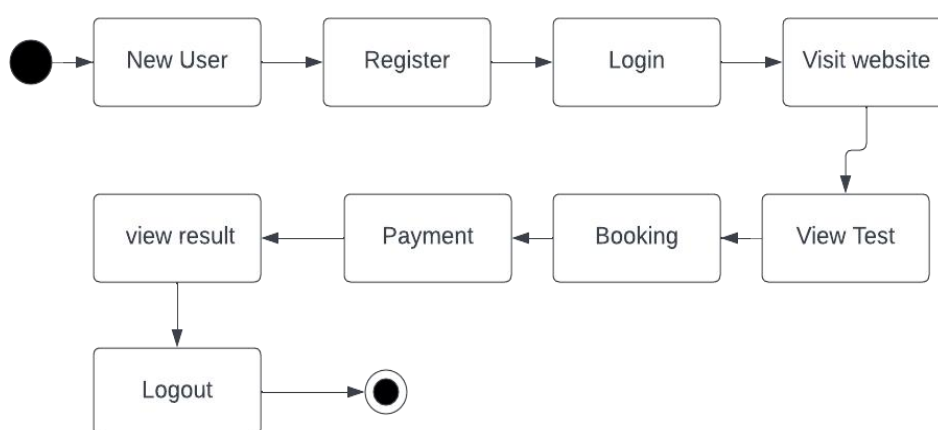


Figure 3: State Chart diagram

### 4.2.4 ACTIVITY DIAGRAM

An activity diagram is essentially a flowchart that shows how one activity leads to another. The action might be referred to as a system operation. One operation leads to the next in the control flow. This flow may be parallel, contemporaneous, or branched. Activity diagrams use many features, such as fork, join, etc., to cope with all types of flow control. An activity diagram is a behavioral diagram i.e., it depicts the behavior of a system. An activity diagram portrays the control flow from a start point to a finish point showing the various decision paths that exist while the activity is being executed.

Activity diagram is another important behavioral diagram in UML diagram to describe dynamic aspects of the system. Activity diagram is essentially an advanced version of flow chart that modeling the flow from one activity to another activity.

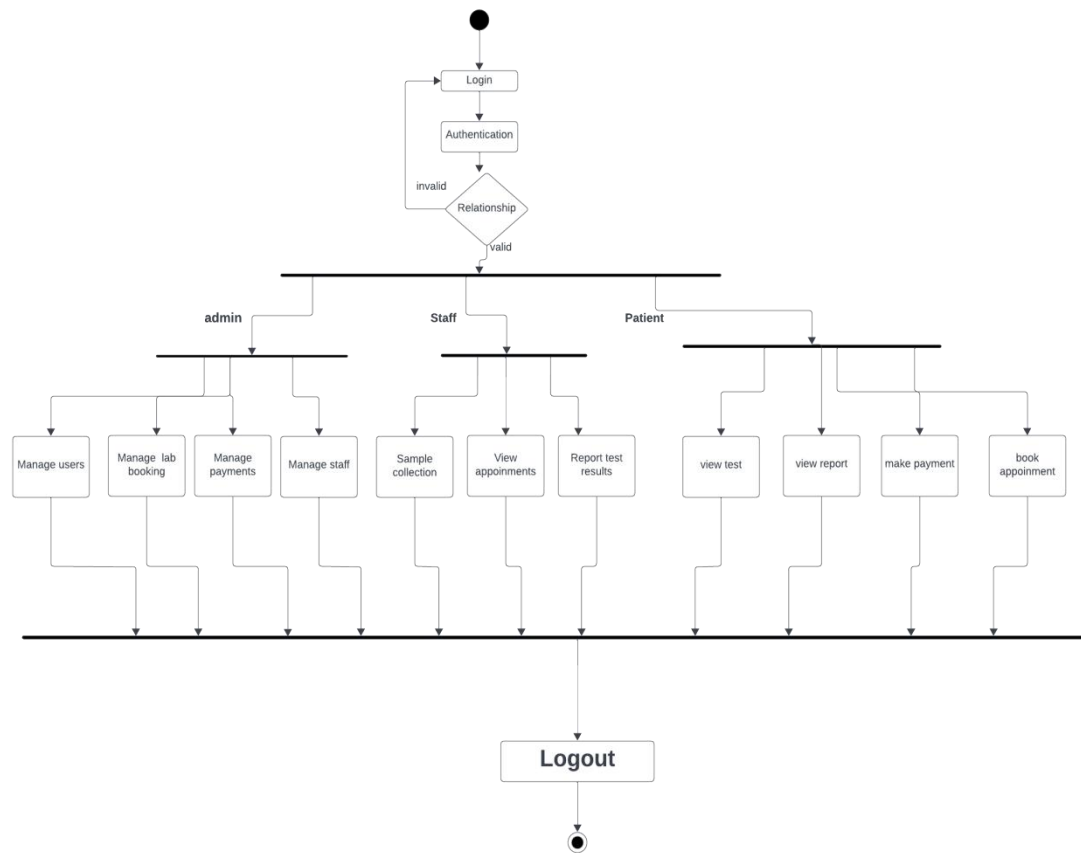


Figure 4: Activity diagram

### 4.2.5 CLASS DIAGRAM

The class diagram is the main building block of object-oriented modeling. It is used for general conceptual modeling of the structure of the application, and for detailed modeling, translating the models into programming code. Class diagrams can also be used for data modeling. Class diagrams are the blueprints of your system or subsystem. You can use class diagrams to model the objects that make up the system, to display the relationships between the objects, and to describe what those objects do and the services that they provide. Class diagrams are useful in many stages of system design. In the analysis stage, a class diagram can help you to understand the requirements of your problem domain and to identify its components. In an object-oriented software project, the class diagrams that you create during the early stages of the project contain classes that often translate into actual software classes and objects when you write code. Later, you can refine your earlier analysis and conceptual models into class diagrams that show the specific parts of your system, user interfaces, logical implementations, and so on. Your class diagrams then become a snapshot that describes exactly how your system works, the relationships between system components at many levels, and how you plan to implement those components.



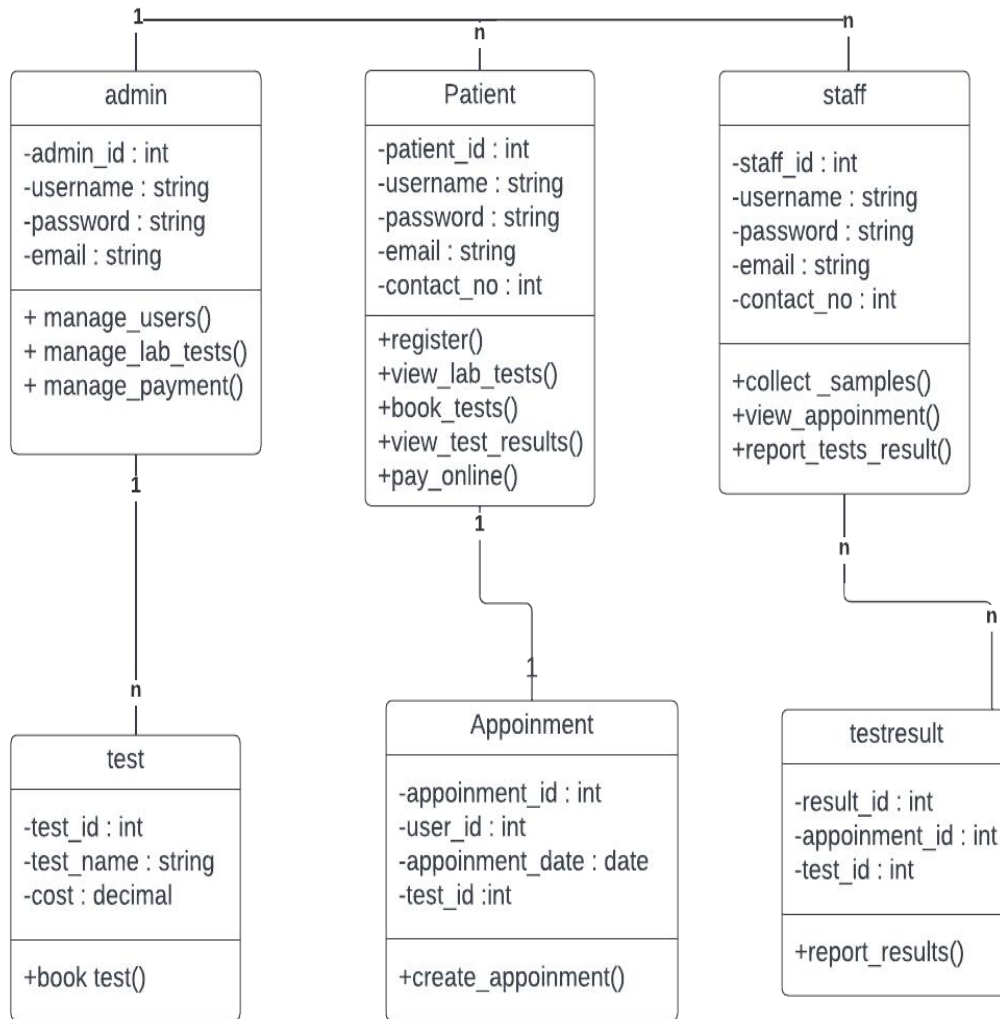


Figure 5: Class diagram

#### 4.2.6 OBJECT DIAGRAM

Since class diagrams are the source of object diagrams, class diagrams are a prerequisite for object diagrams. An instance of a class diagram is represented by an object diagram. Class and object diagrams both use the same fundamental ideas. The static view of a system is also represented by object diagrams, but this static view represents a momentary snapshot of the system. To represent a group of items and their connections as an instance, object diagrams are employed.

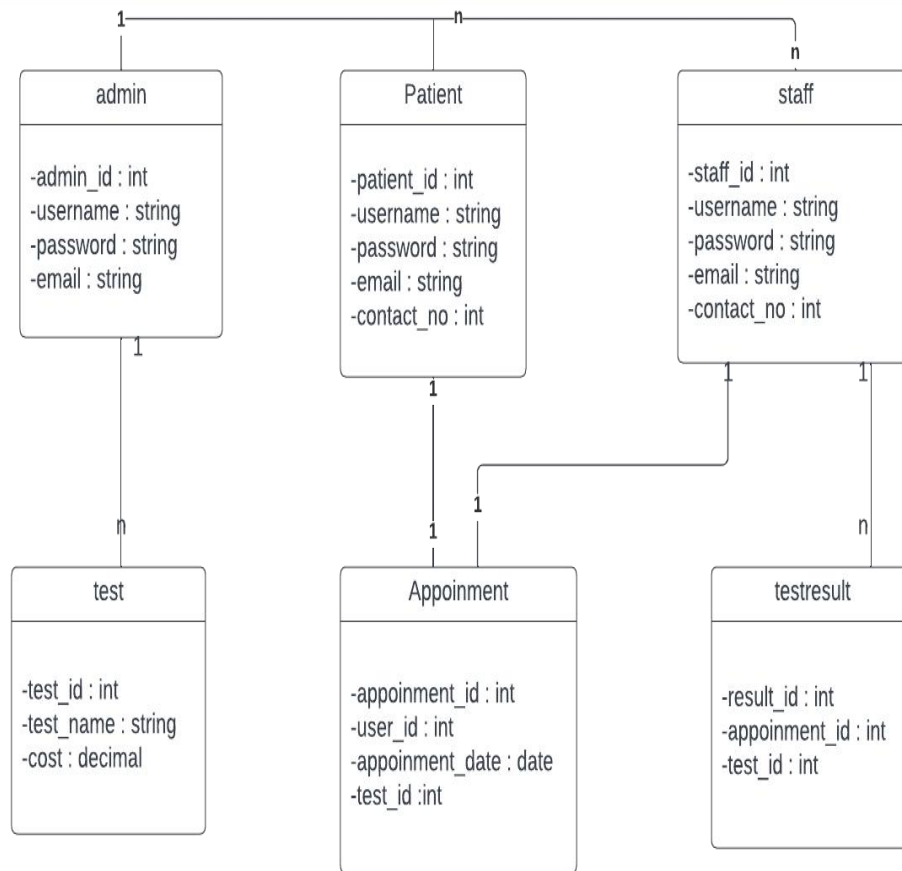


Figure 6: Object diagram

#### 4.2.7 COMPONENT DIAGRAM

UML diagrams are used to represent and describe the behavior of object-oriented systems, assisting in visualization, explanation, and thorough documentation. Particularly in class diagrams, where they record the static behavior of a system, these diagrams play a vital role. On the other hand, graphics can impair the efficiency of real multitasking systems. To maintain coordination, each part within the broader process has a distinct duty and only communicates with other essential components.

A component diagram depicts how components are wired together to form larger components or software systems. They are used to illustrate the structure of arbitrarily complex systems.

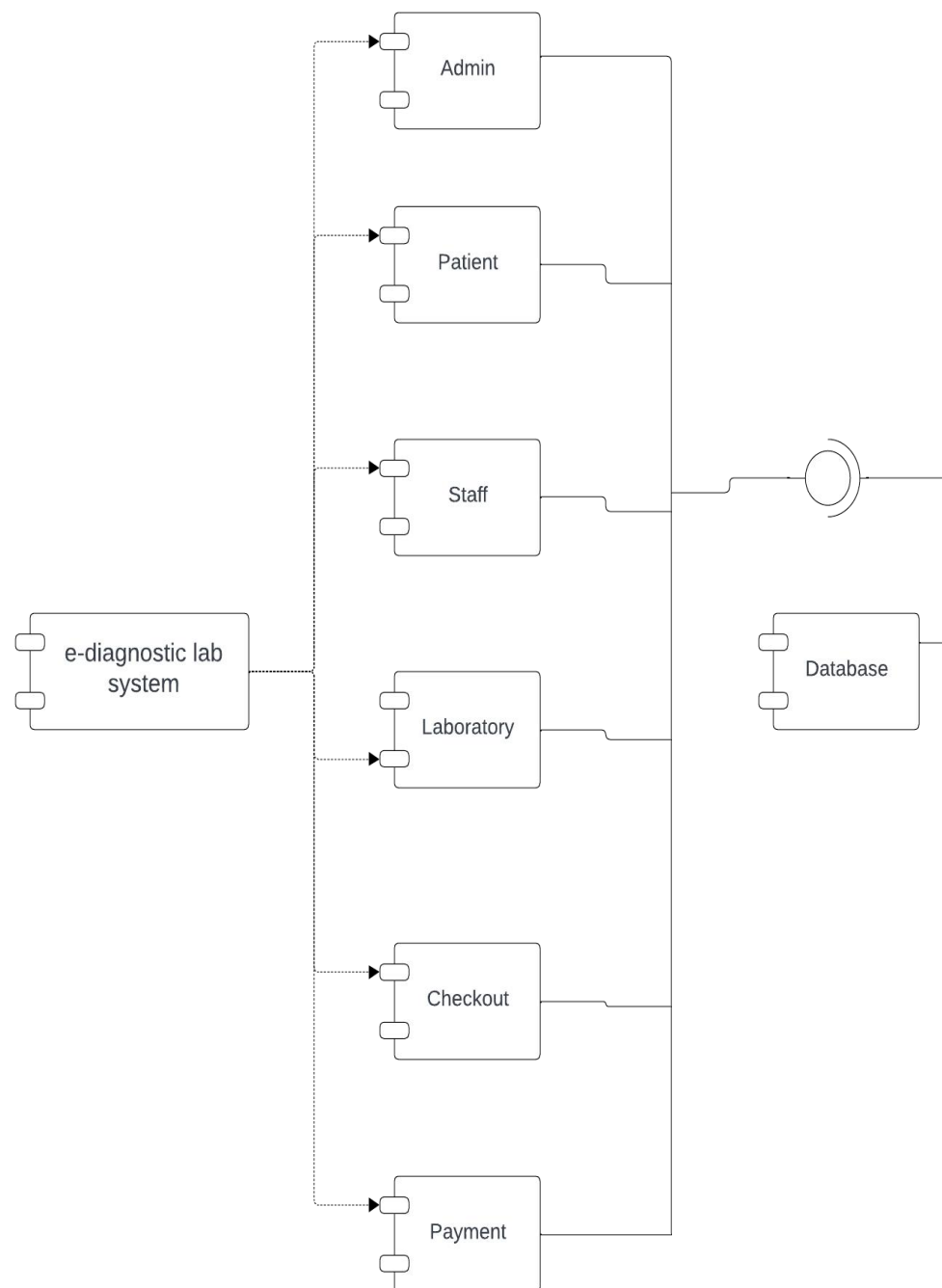


Figure 7: Component diagram

#### 4.2.8 DEPLOYMENT DIAGRAM

The deployment diagram visualizes the physical hardware on which the software will be deployed. It portrays the static deployment view of a system. It involves the nodes and their relationships. It ascertains how software is deployed on the hardware. It maps the software architecture created in

design to the physical system architecture, where the software will be executed as a node. Since it involves many nodes, the relationship is shown by utilizing communication paths. In contrast to other UML diagram types, which primarily depict the logical components of a system. The deployment diagram does not focus on the logical components of the system, but it put its attention on the hardware topology.

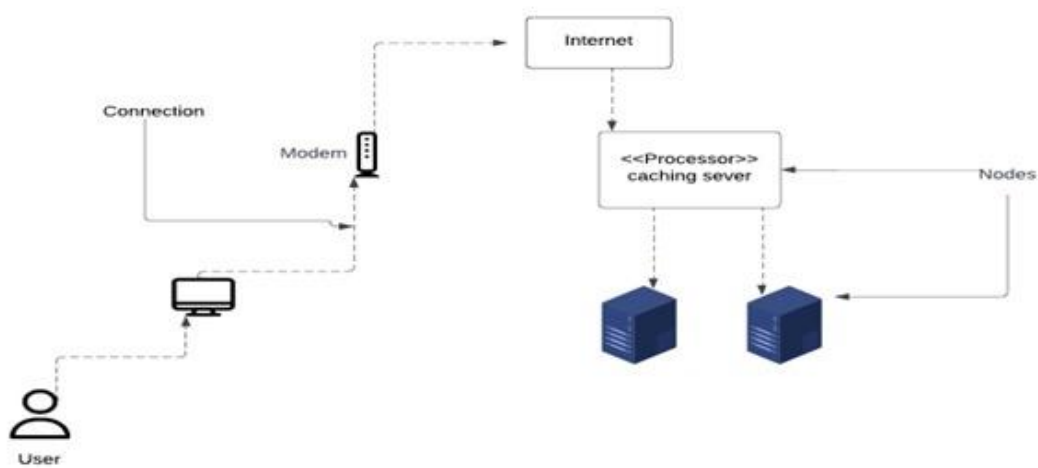
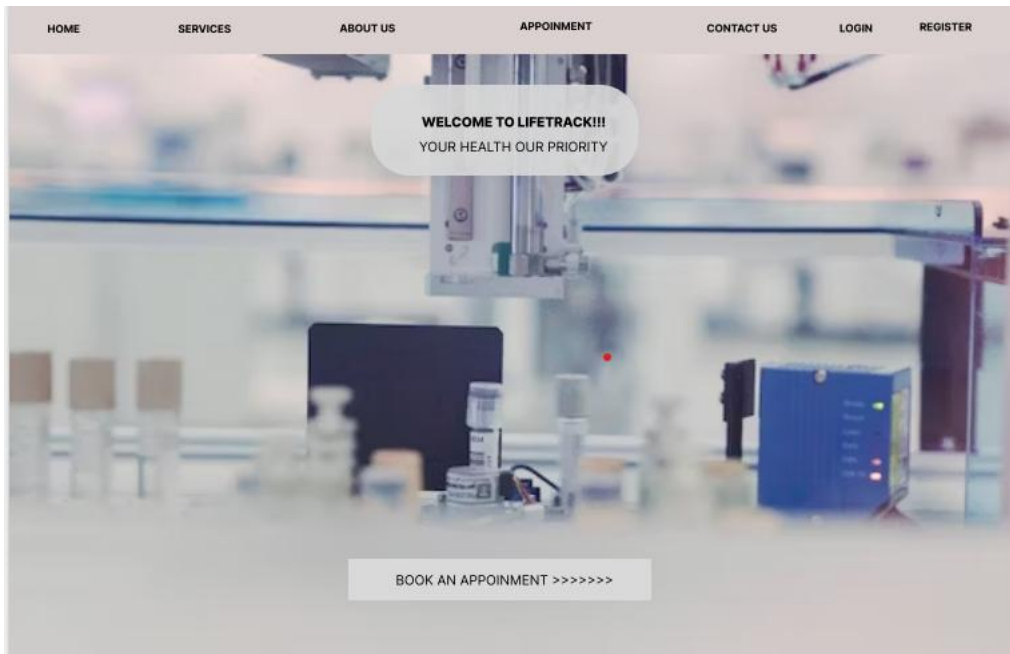


Figure 8: Deployment diagram

## 4.3 USER INTERFACE DESIGN USING FIGMA

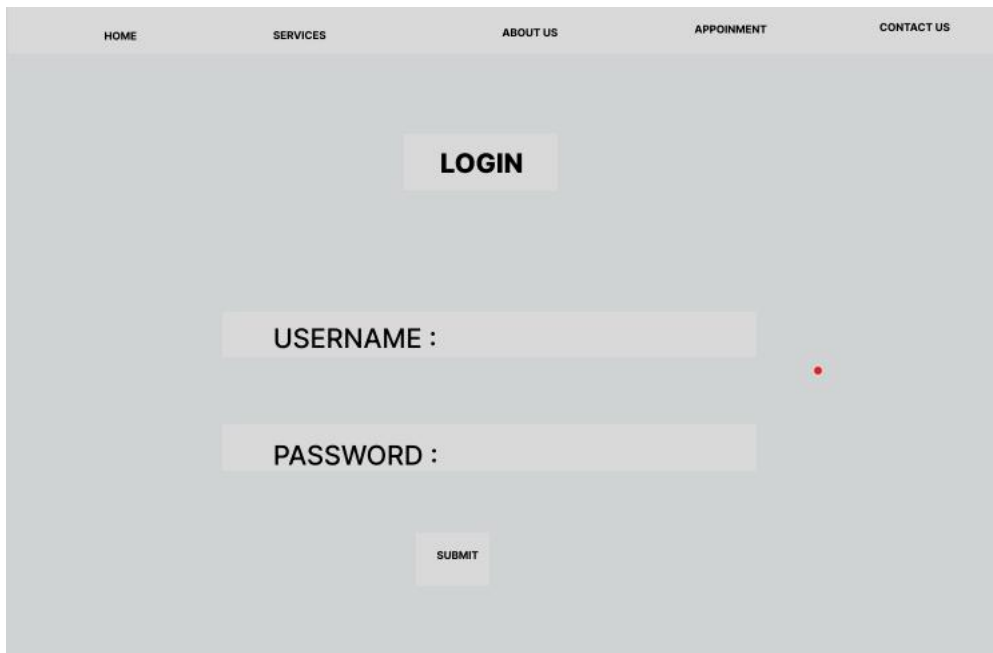
### Home page



### Registration Form

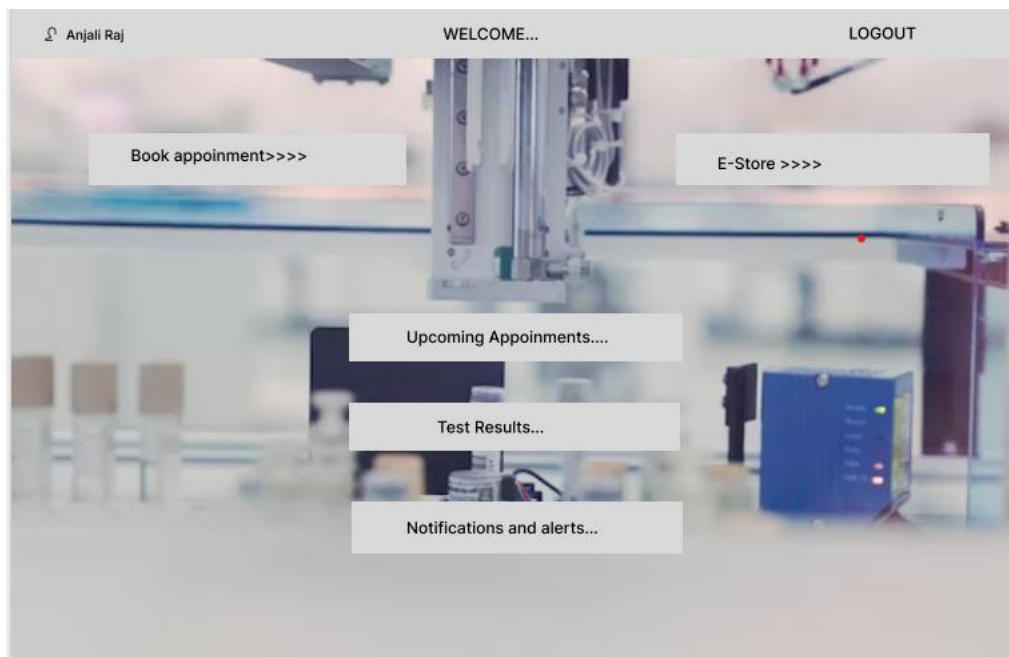
The screenshot shows the registration form of the web application. At the top, there is a navigation bar with links: HOME, SERVICES, ABOUT US, APPOINTMENT, and CONTACT US. The main content area has a light gray background. On the right side, there is a white rounded rectangle with the text "REGISTER !!". On the left side, there are labels for the registration fields: "Full name :", "Age :", "Gender :", "Email :", "Phone no :", "Username:", "Password:", and "Confirm password :". Below these labels, there is a blue checkbox followed by the text "I agree all the terms and conditions .". At the bottom right, there is a blue button with the text "SUBMIT".

## Login page

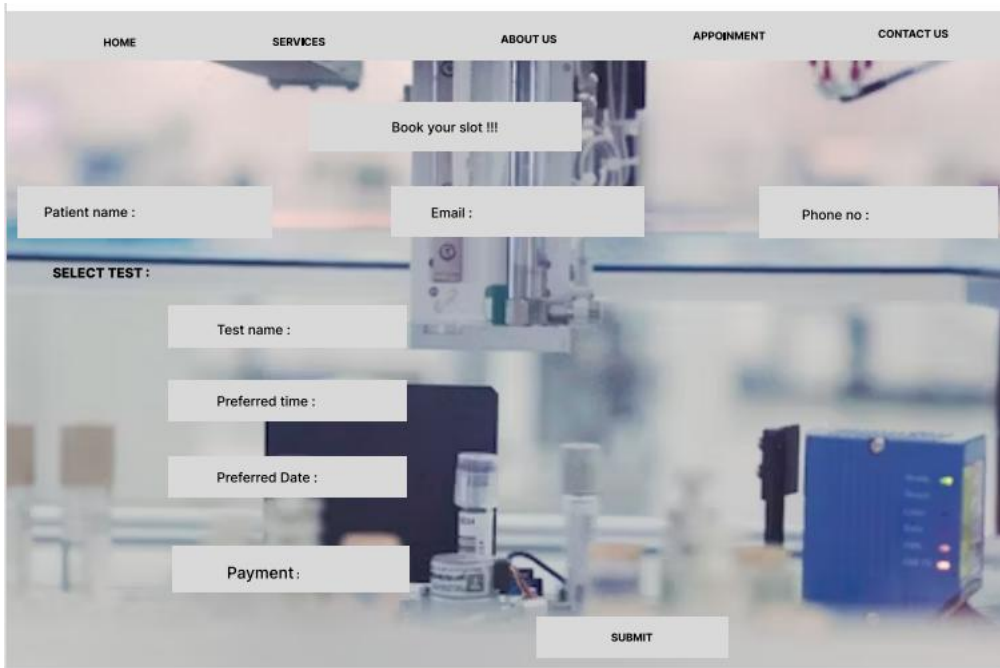


The screenshot shows a web application's login page. At the top, there is a navigation bar with five links: HOME, SERVICES, ABOUT US, APPOINTMENT, and CONTACT US. Below the navigation bar, the word "LOGIN" is displayed in a large, bold, black font. Underneath "LOGIN", there are two input fields: "USERNAME :" and "PASSWORD :". To the right of the "PASSWORD :" field, there is a small red dot. Below the input fields, there is a "SUBMIT" button.

## User page:



The screenshot shows a user profile page. At the top, there is a navigation bar with three links: "Anjali Raj" (with a user icon), "WELCOME...", and "LOGOUT". Below the navigation bar, there is a background image of a laboratory setting. Overlaid on the background image are four buttons: "Book appointment>>>>", "E-Store >>>>", "Upcoming Appoinments....", "Test Results...", and "Notifications and alerts...".

**Booking page:**

The booking page features a light gray navigation bar at the top with links: HOME, SERVICES, ABOUT US, APPOINTMENT, and CONTACT US. The main content area has a blurred background of a laboratory. At the top center, a button says "Book your slot !!!". Below this, there are three input fields for "Patient name :", "Email :", and "Phone no :". A section titled "SELECT TEST :" contains four input fields for "Test name :", "Preferred time :", "Preferred Date :", and "Payment :". A "SUBMIT" button is located at the bottom right of the form area.

HOME SERVICES ABOUT US APPOINTMENT CONTACT US

Book your slot !!!

Patient name : Email : Phone no :

SELECT TEST :

Test name : Preferred time : Preferred Date : Payment :

SUBMIT

## 4.3 DATABASE DESIGN

A database is a collection of data that has been organized to make it simple to manage and update. Information security could be one of the main aims of any database. The database design process is divided into two steps. The goal of the first step is to gather user requirements to create a database that as clearly as possible satisfies user needs. It is known as information-level design and is carried out without the aid of any DBMS. An information-level design is changed to a specific DBMS design that will be used to develop the system in the following stage. The physical-level design phase is where the characteristics of the specific DBMS are considered.

### 4.4.1 RELATIONAL DATABASE MANAGEMENT SYSTEM (RDBMS)

A Relational Database Management System (RDBMS) is a critical software application for organizing and managing data in a structured manner. It stores data in tables with rows and columns, where each row represents a record, and each column is a specific attribute or field. RDBMS systems like MySQL, Oracle, or Microsoft SQL Server ensure data integrity, enforce relationships between tables, and allow for efficient data retrieval and manipulation through Structured Query Language (SQL). These systems are widely used in various applications, such as e-commerce websites, financial systems, and inventory management, due to their robustness, scalability, and ability to handle complex data structures, making them essential for modern data-driven environments.

A Relational Database Management System (RDBMS) is a cornerstone of modern data management. It structures data into tables, where each row represents a unique entry, and each column corresponds to a specific attribute, ensuring a logical and organized storage system. RDBMS systems employ complex algorithms for data retrieval and manipulation, guaranteeing data consistency and adherence to predefined relationships between tables. Popular RDBMS software, including PostgreSQL, MySQL, and Microsoft SQL Server, provides robust features for transactions, data security, and scalability. These systems are integral to a myriad of applications, from healthcare records and customer databases to inventory control and financial platforms, supporting the efficient storage, retrieval, and analysis of vast datasets, making them essential tools in today's data-driven world.

### 4.4.2 NORMALIZATION

Normalization is a database design technique used in relational database management systems (RDBMS) to eliminate data redundancy and improve data integrity. It involves organizing data in a way that reduces data duplication and ensures that relationships between tables are defined and maintained.

The primary goals of normalization are:



**Minimizing Data Redundancy:** By breaking down data into separate tables and ensuring that each piece of data is stored only once, normalization helps reduce the chances of data inconsistencies or errors.

**Ensuring Data Integrity:** Normalization enforces the rules of referential integrity, which means that relationships between tables are well-defined and maintained. This ensures that data remains accurate and consistent.

Normalization typically involves dividing a database into multiple related tables and using primary keys and foreign keys to establish relationships between these tables. There are several normal forms, from First Normal Form (1NF) to Fifth Normal Form (5NF), each with specific rules and requirements. The level of normalization achieved depends on the specific needs of the database and the trade-off between data redundancy and query performance.

By applying normalization principles, designers can create efficient and reliable database structures that support data integrity and ease data maintenance and manipulation.

There are several normal forms (NF) that define specific rules and requirements for achieving progressively higher levels of normalization. Here are the most common normal forms, from First Normal Form (1NF) to Fifth Normal Form (5NF):

**First Normal Form (1NF):**

- Each table has a primary key.
- All columns contain atomic (indivisible) values.
- There are no repeating groups or arrays in columns.

**Second Normal Form (2NF):**

- The table is in 1NF.
- All non-key attributes are fully functionally dependent on the entire primary key. In other words, all non-key attributes must be dependent on the entire primary key, not just part of it.

**Third Normal Form (3NF):**

- The table is in 2NF.
- There is no transitive dependency, meaning that non-key attributes are not dependent on other non-key attributes.

**Boyce-Codd Normal Form (BCNF):**

- A stricter version of 3NF.
- It enforces that every non-trivial functional dependency involves a super key.

**Fourth Normal Form (4NF):**

- Addresses multi-valued dependencies.

- Eliminates any multi-valued dependencies within the data.

Fifth Normal Form (5NF):

- Also known as Project-Join Normal Form (PJ/NF).
- Handles cases where data can be derived by joining multiple tables.

#### 4.4.3 SANITIZATION

In Python Django, sanitization refers to the process of cleaning and validating data to ensure that it is safe and free from malicious content before it is used or stored in a database. Sanitization is a crucial security practice to prevent various forms of attacks, such as cross-site scripting (XSS) and SQL injection, which can compromise the security and integrity of a web application.

#### 4.4.4 INDEXING

Indexing in Django is a fundamental database optimization technique. It involves creating data structures, or indexes, on specific fields to expedite data retrieval. These indexes act as signposts that enable the database to swiftly locate and fetch relevant data, especially in tables with substantial amounts of information. Django offers automatic index creation for primary keys, unique fields, and foreign keys. Additionally, developers can define custom indexes for fields frequently used in filtering, sorting, or searching. The choice of proper indexing plays a pivotal role in improving query performance, resulting in more responsive and scalable web applications. It's essential to consider application-specific query patterns and create indexes accordingly, as well as to understand the capabilities and limitations of the chosen database backend. Effective indexing is a cornerstone of efficient database operations in Django, contributing to enhanced application performance.

### 4.5 TABLE DESIGN

#### 1.Tbl\_user

Primary key: **user\_id**

No:	Field name	Data type	Key constraints	Description
1	<b>user_id</b>	int	Primary key	Unique identifier for user
2	Username	varchar(50)	Not Null	User name for login
3	password	varchar(50)	Not Null	password
4	email	varchar(50)	Not Null	email

5	Phonenumber	int	Not Null	Contactnumber
6	Role	Enum(Patient,Staff)	Not null	Indicates whether the user is patient or staff.
7	Full name	Varchar(50)	Not Null	Full name of user
8	Address id	int	Foreign Key	References to the address of user

## 2.Tbl\_appointment

Primary key: **appointment\_id**

Foreign key: **user\_id** references table **Tbl\_user**

No:	Field name	Data type	Key constraints	Description
1	<b>Appointment_id</b>	int	Primary key	Unique identifier for each appointment
2	User_id	int	foreignkey	Referenced to the user who booked
3	Appointment_date	dateTime	Not Null	Date and time for appointment
5	Appointment_type	varchar(50)	Not Null	Online/Offline
6	Testid	int	Foreign key	Referenced to the test booked
7	Prescription	FileField	Not Null	Patient's prescription regarding test
8	Additional_Testid	int	Foreign key	Referenced to additional the test booked
9	Location_id	int	Foreign Key	Referenced to the location for sample collection

**3.Tbl\_test**Primary key: **test\_id**

No:	Field name	Data type	Key constraints	Description
1	<b>Test_id</b>	int	Primary Key	Unique identifier for each test
2	TestName	varchar(50)	Not Null	Name of test
3	Cost	decimal	Not Null	Cost of test

**4.Tbl\_appointment**Primary key: **appointment\_id**Foreign Key : **address\_id** referencing referenceds **Tbl\_address****user\_id** references table **Tbl\_user**

No:	Field name	Data type	Key constraints	Description
1	<b>address_id</b>	int	Primary key	Unique identifier for each address
2	user_id	int	Foreign key	Referencing the user associated with the address
3	Streetaddress	varchar(50)	Not Null	Street Address
4	city	Varchar(50)	Not Null	City of user
5	Postalcode	Varchar(50)	Not Null	Postal code of user

**5.Tbl\_payment**Primary Key: **payment\_id**Foreign Key : **appointment\_id** referencing referenceds **Tbl\_appointment****user\_id** references table **Tbl\_user**

No:	Field name	Data type	Key constraints	Description
1	<b>Paymentid</b>	int	Primary key	Unique identifier for payment

2	Appoinmentid	int	foreignkey	References appoinment associated with payment
3	Paymentdate	Datetime	Not Null	Date and time of payment
4	Amount(test+service)	decimal	Not Null	Amount payed
5	Razorpay_id	varchar(50)	Not Null	Gateway id of payment
6	Userid	int	Foreign Key	References to the user who doing payment

## 6.Tbl\_location

Primary Key: **location\_id**

No:	fieldname	Datatype	Key Constraints	Description
1	<b>Location_id</b>	int	Primary Key	Unique identifier for a location
2	Distance	Decimal	Not Null	Distance from lab to locations for selecting sample
3	Address	Text	Not Null	Address of the selected Location
4	Latitude	Varchar(50)	Not Null	Latitude of the selected location
5	Longitude	Varchar(50)	Not Null	Logitude of the selected location

## 7.Tbl\_Patient

Primary Key: **patient\_id**

Foreign Key : **appointment\_id** referencing referenceds **Tbl\_appointment**

No:	fieldname	Datatype	Key constraints	Description
1	<b>Patient_id</b>	int	Primary Key	Unique identifier for patient
2	appointmentid	int	Foreign key	References to the appointment
3	pname	Varchar(50)	Not Null	Patient Name
4	age	int	Not Null	Patient age
5	gender	Enum(male,female,others)	Not Null	Patient gender

## 7. Tbl\_Review

Primary Key: **review\_id**

No:	fieldname	Datatype	Key constraints	Description
1	reviewid	int	Primary Key	Unique identifier for a review
2	comment	Text	Not Null	Review by a user
3	rating	int	Not Null	Rating by a user
4	Created_at	dateTime	Not Null	Date and time of review created
5	userid	int	ForeignKey	Referenced to the user who make review

## **CHAPTER 5**

### **SYSTEM TESTING**

## 5.1 INTRODUCTION

Software testing is a way to check if the computer program works like it's supposed to. We use testing to make sure the software does what it is supposed to do. Validation means checking or testing things like software to make sure they meet the requirements and standards they are supposed to follow. Software testing is a way to check if a program works well. It goes along with other methods like checking and walking through the program. Validation means making sure that what the user wanted is what they got. There are several rules that can serve as testing objectives.

They are:

Testing is a process of executing a program with the intent of finding an error.

- A good test case is one that has high possibility of finding an undiscovered error.
- A successful test is one that uncovers an undiscovered error.

If a test works well and follows its goals, it can find mistakes in the software. The test showed that the computer program is working like it's supposed to and is doing well.

There are three ways to test program.

- For correctness
- For implementation efficiency
- For computational complexity

## 5.2 TEST PLAN

A test plan suggests several required steps that need be taken to complete various testing methodologies. The activity that is to be taken is outlined in the test plan. A computer program, its documentation, and associated data structures are all created by software developers. It is always the responsibility of the software developers to test each of the program's separate components to make sure it fulfills the purpose for which it was intended. To solve the inherent issues with allowing the builder evaluate what they have developed, there is an independent test group (ITG). Testing's precise goals should be laid forth in quantifiable language. So that the mean time to failure, the cost to find and fix the defects, remaining defect density or frequency of occurrence and test work-hours per regression test all should be stated within the test.

The levels of testing include:

- Unit testing
- Integration Testing
- Data validation Testing & Output Testing



### 5.2.1 UNIT TESTING

Unit testing concentrates verification efforts on the software component or module, which is the smallest unit of software design. The component level design description is used as a guide when testing crucial control paths to find faults inside the module's perimeter. The level of test complexity and the untested area determined for unit testing. Unit testing is white-box focused, and numerous components may be tested simultaneously. To guarantee that data enters and exits the software unit under test properly, the modular interface is tested. To make sure that data temporarily stored retains its integrity during each step of an algorithm's execution, the local data structure is inspected. Boundary conditions are tested to ensure that all statements in a module have been executed at least once. Finally, all error handling paths are tested.

Before starting any other test, tests of data flow over a module interface are necessary. All other tests are irrelevant if data cannot enter and depart the system properly. An important duty during the unit test is the selective examination of execution pathways. Error circumstances must be foreseen in good design, and error handling paths must be put up to cleanly reroute or halt work when an error does arise. The final step of unit testing is boundary testing. Software frequently fails at its limits.

In the Sell-Soft System, unit testing was carried out by treating each module as a distinct entity and subjecting them to a variety of test inputs. The internal logic of the modules had some issues, which were fixed. Each module is tested and run separately after coding. All unused code was eliminated, and it was confirmed that every module was functional and produced the desired outcome

### 5.2.2 INTEGRATION TESTING

Integration testing is a methodical approach for creating the program's structure while also carrying out tests to find interface issues. The goal is to construct a program structure that has been determined by design using unit tested components. The program is tested. Correction is challenging since the size of the overall program makes it challenging to isolate the causes. As soon as these mistakes are fixed, new ones arise, and the process repeats itself in an apparently unending cycle. All the modules were integrated after unit testing was completed in the system to check for an interface inconsistency. A distinctive program structure also developed when discrepancies in program structures.

### 5.2.3 VALIDATION TESTING OR SYSTEM TESTING

The testing process comes to an end here. This involved testing the entire system in its entirety, including all forms, code, modules, and class modules. Popular names for this type of testing include system tests and black box testing. The functional requirements of the software are the main emphasis of the black box testing approach. That example, using Black Box testing, a software engineer can create sets of input conditions that will fully test every program requirement. The following sorts of problems are targeted by black box testing: erroneous or missing functions, interface errors, data structure or external data access errors, performance errors, initialization errors, and termination errors.

### 5.2.4 OUTPUT TESTING OR USER ACCEPTANCE TESTING

The system considered is tested for user acceptance; here it should satisfy the firm's need. The software should keep in touch with perspective system; user at the time of developing and making changes whenever required.

This done with respect to the following points:

- Input Screen Designs,
- Output Screen Designs,

The above testing is done taking various kinds of test data. Preparation of test data plays a vital role in the system testing. After preparing the test data, the system under study is tested using that test data. While testing the system by which test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future.

### 5.2.5 AUTOMATION TESTING

Automation Testing is a software testing technique that performs using special automated testing software tools to execute a test case suite. Essentially, it's a test to double-check that the equipment or software does exactly what it was designed to do. It tests for bugs, defects, and any other issues that can arise with product development. Although some types of testing, such as regression or functional testing can be done manually, there are greater benefits of doing it automatically. Automation testing can be run at any time of the day. It uses scripted sequences to examine the software. It then reports on what has been found, and this information can be compared with earlier test runs. Automation developers generally write in the following programming languages: C#, JavaScript, and Ruby.

### 5.2.6 SELENIUM TESTING

Selenium is an open-source automated testing framework used to verify web applications across different browsers and platforms. Selenium allows for the creation of test scripts in various programming languages such as Java, C#, and Python. Jason Huggins, an engineer at Thought Works, developed Selenium in 2004 while working on a web application that required frequent testing. He created a JavaScript program called "JavaScriptTestRunner" to automate browser actions and improve testing efficiency. Selenium has since evolved and continues to be developed by a team of contributors. In addition to Selenium, another popular tool used for automated testing is Cucumber. Cucumber is an open-source software testing framework that supports behavior-driven development (BDD). It allows for the creation of executable specifications in a human-readable format called Gherkin. One of the advantages of using Cucumber is its ability to bridge the gap between business stakeholders and technical teams. By using a common language, Cucumber facilitates effective communication and collaboration during the testing process. It promotes a shared understanding of the requirements and helps ensure that the developed software meets the intended business goals. Cucumber can be integrated with Selenium to combine the benefits of both tools. Selenium is used for interacting with web browsers and automating browser actions, while Cucumber provides a structured framework for organizing and executing tests. This combination allows for the creation of end-to-end tests that verify the behavior of web applications across different browsers and platforms, using a business-readable and maintainable format.

#### Test Case 1

##### Code

```
import unittest
from selenium import webdriver
from selenium.webdriver.common.by import By

from selenium.webdriver.support.ui import WebDriverWait

from selenium.webdriver.support import expected_conditions as EC

# ... (your imports)

class LoginTest(unittest.TestCase):
    def setUp(self):
        self.driver = webdriver.Chrome()
        self.base_url = "http://localhost:8000/login.html"
        self.driver.maximize_window()
        self.driver.implicitly_wait(10)

    def wait_for_element(self, by, value, timeout=10):
        return WebDriverWait(self.driver, timeout).until(
```

```

        EC.presence_of_element_located((by, value))
    )

    def test_successful_login(self):
        self.driver.get(self.base_url)

        username = self.wait_for_element(By.ID, 'username')
        password = self.wait_for_element(By.ID, 'password')

        username.send_keys("anu")
        password.send_keys("Anu@12345")

        login_button = self.wait_for_element(By.ID, 'login-button')
        login_button.click()

        self.assertIn("user.html", self.driver.current_url)

    def tearDown(self):
        self.driver.quit()

if __name__ == "__main__":
    unittest.main()

```

## Screenshot

```

● (venv) PS C:\Project\PROJECT> py manage.py test tests
Found 1 test(s).
System check identified no issues (0 silenced).

DevTools listening on ws://127.0.0.1:56249/devtools/browser/d447680b-8983-4d23-a7af-c4e3b1f9bced
.
-----
Ran 1 test in 14.424s

OK
○ (venv) PS C:\Project\PROJECT> 

```

## Test Report

Test Case 1	
<b>Project Name: OneHealth</b>	
<b>Login Test Case</b>	
<b>Test Case ID:</b> Test_1	<b>Test Designed By:</b> Anjali Raj
<b>Test Priority (Low/Medium/High):</b> High	<b>Test Designed Date:</b> 02-12-2023
<b>Module Name:</b> Login Screen	<b>Test Executed By :</b> Dr. Bijimol T K
<b>Test Title :</b> User Login	<b>Test Execution Date:</b> 02-12-2023
<b>Description:</b> Verify login with valid username and	

password					
<b>Pre-Condition</b> :User has valid username and password					
Step	Test Step	Test Data	Expected Result	Actual Result	Status(Pass/Fail)
1	Navigation to Login Page		Login Page should be displayed	Login Page be displayed	Pass
2	Provide valid username	Username:anu	User should be able to login	User is logged in and navigated to corresponding Home Page	Pass
3	Provide Valid Password	Password: Anu@12345			
4	Click on Login button				
<b>Post-Condition:</b> User is validated with database and successfully login into account. The Account session details are logged in database					

## Test Case 2:

### Code

```
import unittest
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time

class AdminDashboardTest(unittest.TestCase):
    def setUp(self):
        self.driver = webdriver.Chrome()
        self.base_url = "http://localhost:8000/admin-dashboard"
        self.driver.maximize_window()
        self.driver.implicitly_wait(10)

    def wait_for_element(self, by, value, timeout=10):
        return WebDriverWait(self.driver, timeout).until(
            EC.presence_of_element_located((by, value))
        )

    def login_admin(self):
        self.driver.get("http://localhost:8000/login.html")
        username = self.wait_for_element(By.ID, 'username')
        password = self.wait_for_element(By.ID, 'password')
        username.send_keys("admin")
        password.send_keys("P@ssw0rd")
        login_button = self.wait_for_element(By.ID, 'login-button')
        login_button.click()

    def test_add_test_and_redirect(self):
```

```

self.login_admin() # Login as admin first
add_test_link = self.wait_for_element(By.ID, 'addTestLink')
add_test_link.click()

name_input = self.wait_for_element(By.ID, 'name')
price_input = self.wait_for_element(By.ID, 'price')
availability_checkbox = self.wait_for_element(By.ID, 'checkbox-availability')
add_test_button = self.wait_for_element(By.ID, 'addTestButton')

name_input.clear()
name_input.send_keys("New Test")

price_input.clear()
price_input.send_keys("50.00")
if not availability_checkbox.is_selected():
    availability_checkbox.click()
time.sleep(1)
add_test_button.click()

self.assertIn("admintest.html", self.driver.current_url)
print("Login with username and password .Then add new test to the existing list.")

```

## Screenshot

```

(venv) PS C:\Project\PROJECT> py manage.py test tests
Login with username and password .Then add new test to the existing list.
Found 1 test(s).
System check identified no issues (0 silenced).

DevTools listening on ws://127.0.0.1:56460/devtools/browser/0f742a6a-95f2-4fdd-b68c-440398b191cd
.
-----
Ran 1 test in 8.453s

OK
(venv) PS C:\Project\PROJECT>

```

## Test report

Test Case 2	
<b>Project Name: OneHealth</b>	
<b>Addtest Test Case</b>	
<b>Test Case ID: Test_2</b>	<b>Test Designed By: Anjali Raj</b>
<b>Test Priority (Low/Medium/High): High</b>	<b>Test Designed Date: 2-12-2023</b>
<b>Module Name: Addtest</b>	<b>Test Executed By : Dr. Bijimol T K</b>
<b>Test Title : Adding test to list</b>	<b>Test Execution Date: 2-12-2023</b>
<b>Description:</b> User diagnosing test is added by admin	
<b>Pre-Condition :User has valid username and password</b>	

Step	Test Step	Test Data	Expected Result	Actual Result	Status(Pass/Fail)
1	Navigation to Login Page		Login Page should be displayed	Login Page be displayed	Pass
2	Provide valid username	username: admin	Admin should be able to login	Admin is logged in and navigated to corresponding Home Page	Pass
3	Provide Valid Password	Password: P@ssw0rd			
4	Click on Login button				
5	Click on Addtest	Let it show		Add test page is displayed	Pass
6	Enter details and submit			Test added to list	Pass
<b>Post-Condition:</b> Test is added to List successfully.					

### Test Case 3:

#### Code

```
import unittest
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time

class AdminDashboardTest(unittest.TestCase):
    def setUp(self):
        self.driver = webdriver.Chrome()
        self.base_url = "http://localhost:8000/admin-dashboard"
        self.driver.maximize_window()
        self.driver.implicitly_wait(10)

    def wait_for_element(self, by, value, timeout=10):
        return WebDriverWait(self.driver, timeout).until(
            EC.presence_of_element_located((by, value))
        )

    def login_admin(self):
        self.driver.get("http://localhost:8000/login.html")

        username = self.wait_for_element(By.ID, 'username')
        password = self.wait_for_element(By.ID, 'password')
        username.send_keys("admin")
        password.send_keys("P@ssw0rd")
        login_button = self.wait_for_element(By.ID, 'login-button')
```

```

        login_button.click()

    def test_add_staff_and_redirect(self):
        self.login_admin() # Login as admin first
        add_staff_link = self.wait_for_element(By.ID, 'addStaffLink')
        add_staff_link.click()
        full_name_input = self.wait_for_element(By.ID, 'first-name')
        username_input = self.wait_for_element(By.ID, 'username')
        email_input = self.wait_for_element(By.ID, 'email')
        password_input = self.wait_for_element(By.ID, 'password')
        submit_button = self.wait_for_element(By.ID, 'submit-button')

        # Fill in the details
        full_name_input.clear()
        full_name_input.send_keys("New Staff")

        username_input.clear()
        username_input.send_keys("newstaff9")

        email_input.clear()
        email_input.send_keys("newstaff@example.com")

        password_input.clear()
        password_input.send_keys("P@ssw0rd")

        submit_button.click()
        time.sleep(2)
        print("Current URL:", self.driver.current_url)
        self.assertIn("admin-dashboard", self.driver.current_url)

if __name__ == "__main__":
    unittest.main()

```

## Screenshot

```

● (venv) PS C:\Project\PROJECT> py manage.py test tests
Login with valid username and password
Add new staff to the laboratory using essential credentials
Found 1 test(s).
System check identified no issues (0 silenced).

DevTools listening on ws://127.0.0.1:56736/devtools/browser/21f0f9e7-901f-4547-9f74-f9244ff82d5a
Current URL: http://localhost:8000/admin-dashboard
.
-----
Ran 1 test in 17.889s

OK
○ (venv) PS C:\Project\PROJECT> 

```



**Test report**

Test Case 3					
Project Name: OneHealth					
Staff Test Case					
Test Case ID: Test_3			Test Designed By:Anjali Raj		
Test Priority (Low/Medium/High): High			Test Designed Date: 2-12-2023		
Module Name: AddingStaff			Test Executed By : Dr. Bijimol T K		
Test Title : Adding staff to laboratory			Test Execution Date: 2-12-2023		
Description: Admin adding staff to laboratory					
Pre-Condition :User has valid username and password					
Step	Test Step	Test Data	Expected Result	Actual Result	Status(Pass/Fail)
1	Navigation to Login Page		Login Page should be displayed	Login Page be displayed	Pass
2	Provide valid username	Email: admin	Admin should be able to login	Admin is logged in and navigated to corresponding Home Page	Pass
3	Provide Valid Password	Password: P@ssw0rd			
4	Click on Login button				
5	Click on addstaff	Let it snow		Add staff page is displayed	Pass
6	Add the details of staff	Let it snow		Staff added to laboratory	Pass
Post-Condition: Staff is added to laboratory successfully.					

**Test Case 4:****Code**

```
import unittest
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time
```

```
class LoginAndSearchTest(unittest.TestCase):
```

```

def setUp(self):
    self.driver = webdriver.Chrome()
    self.base_url = "http://localhost:8000/login.html"
    self.driver.maximize_window()
    self.driver.implicitly_wait(10)

def wait_for_element(self, by, value, timeout=10):
    return WebDriverWait(self.driver, timeout).until(
        EC.presence_of_element_located((by, value))
    )

def test_successful_login_and_search(self):

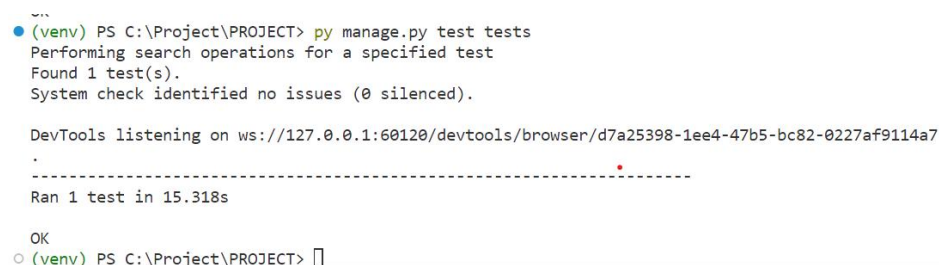
    self.driver.get(self.base_url)
    username = self.wait_for_element(By.ID, 'username')
    password = self.wait_for_element(By.ID, 'password')
    username.send_keys("anu")
    password.send_keys("Anu@12345")
    login_button = self.wait_for_element(By.ID, 'login-button')
    login_button.click()
    self.assertIn("user.html", self.driver.current_url)
    self.driver.get("http://localhost:8000/services.html")
    search_input = self.wait_for_element(By.ID, 'search')
    search_input.send_keys("Thyroid")
    search_button = self.wait_for_element(By.CLASS_NAME, 'book-button')
    search_button.click()
    self.wait_for_element(By.CLASS_NAME, 'test-listing')
    time.sleep(5) # You can adjust the sleep duration as needed
    self.assertIn("Thyroid", self.driver.page_source)

def tearDown(self):
    self.driver.quit()

if __name__ == "__main__":
    unittest.main()
print("Performing search operations for a specified test")

```

## Screenshot



```

~
● (venv) PS C:\Project\PROJECT> py manage.py test tests
Performing search operations for a specified test
Found 1 test(s).
System check identified no issues (0 silenced).

DevTools listening on ws://127.0.0.1:60120/devtools/browser/d7a25398-1ee4-47b5-bc82-0227af9114a7
.
-----
Ran 1 test in 15.318s

OK
○ (venv) PS C:\Project\PROJECT>

```

## Test report

Test Case 4					
Project Name: OneHealth					
Perform search operations					
Test Case ID: Test_4			Test Designed By: Anjali Raj		
Test Priority (Low/Medium/High): High			Test Designed Date: 2-12-2023		
Module Name: Perform Search			Test Executed By : Dr. Bijimol T K		
Test Title : Perform search for test			Test Execution Date: 2-12-2023		
Description:User searching for a specified test					
Pre-Condition :User has valid username and password					
Step	Test Step	Test Data	Expected Result	Actual Result	Status(Pass/Fail)
1	Navigation to Login Page		Login Page should be displayed	Login Page be displayed	Pass
2	Provide validUsername	Email: fablelonappan@gmail.com	User should be able to login	User is logged in and navigated to corresponding user Page	Pass
3	Provide Valid Password	Password: Lonson@2609			
4	Click on Login button	Let it snow			
5	Click on services	Let it snow		The list of tests is displayed	Pass
6	Click on search button for searching	Let it snow		Searched test is visible	Pass
Post-Condition: Test search is done successfully					

## **CHAPTER 6**

### **IMPLEMENTATION**

## 6.1 INTRODUCTION

Implementation is the stage of the project where the theoretical design is turned into a working system. It can be the most crucial stage in achieving a successful new system gaining the users confidence that the new system will work and will be effective and accurate. It is primarily concerned with user training and documentation. Conversion usually takes place about the same time the user is being trained or later. Implementation simply means convening a new system design into operation, which is the process of converting a new revised system design into an operational one.

At this stage the main work load, the greatest upheaval and the major impact on the existing system shifts to the user department. If the implementation is not carefully planned or controlled, it can create chaos and confusion. Implementation includes all those activities that take place to convert from the existing system to the new system. The new system may be a totally new, replacing an existing manual or automated system or it may be a modification to an existing system. Proper implementation is essential to provide a reliable system to meet organization requirements. The process of putting the developed system in actual use is called system implementation. This includes all those activities that take place to convert from the old system to the new system. The system can be implemented only after through testing is done and if it is found to be working according to the specifications. The system personnel check the feasibility of the system. The more complex the system being implemented, the more involved will be the system analysis and design effort required to implement the three main aspects: education and training, system testing and changeover. The implementation state involves the following tasks:

- Careful planning.
- Investigation of system and constraints.
- Design of methods to achieve the changeover.

## 6.2 IMPLEMENTATION PROCEDURES

The implementation procedures for this project involve a systematic and phased approach to bring the online crime reporting portal to fruition. This complex system, designed to cater to the needs of various stakeholders, requires careful planning and execution.

To commence the implementation, the project team will initiate a detailed requirements analysis phase. This stage involves comprehensive discussions with law enforcement officials, control room staff, prison wardens, and potential end-users to determine their specific needs and

expectations. The results of this analysis will inform the development of a detailed system specification and design.

The development phase will follow, involving the creation of the portal's architecture, databases, and user interfaces. It's during this stage that the functionalities outlined in the project's scope, including user management, crime reporting, and communication features, will be integrated into the system. The portal will be designed with a user-friendly interface to ensure ease of use for all categories of users.

Simultaneously, a dedicated team will work on the incorporation of advanced technologies, such as machine learning capabilities for crime trend analysis and predictive features. These technologies will be implemented carefully to ensure the portal's robustness and security.

Once the development phase is complete, rigorous testing will be conducted to identify and rectify any bugs or issues. The portal will undergo extensive quality assurance and user acceptance testing to ensure that it meets all the necessary requirements and is free of vulnerabilities.

Deployment of the system will be carried out in a controlled manner, ensuring minimal disruption to the existing processes, and allowing for gradual adaptation by the involved stakeholders. Comprehensive training sessions will be conducted to familiarize law enforcement personnel, control room staff, prison wardens, and registered users with the portal's functionality and features. Post-deployment, the project team will continue to provide support and maintenance, addressing any issues that may arise and making necessary improvements as the system matures. Regular updates and security measures will be implemented to safeguard user data and maintain the portal's efficiency.

### **6.2.1 USER TRAINING**

User training is designed to prepare the user for testing and converting the system. To achieve the objective and benefits expected from computer-based system, it is essential for the people who will be involved to be confident of their role in the new system. As system becomes more complex, the need for training is more important. By user training the user comes to know how to enter data, respond to error messages, interrogate the database, and call up routine that will produce reports and perform other necessary functions

### **6.2.2 TRAINING ON THE APPLICATION SOFTWARE**

After providing the necessary basic training on computer awareness the user will have to be trained on the new application software. This will give the underlying philosophy of the use of the new system such as the screen flow, screen design type of help on the screen, type of errors while entering the data, the corresponding validation check at each entry and the ways to correct the date

entered. It should then cover information needed by the specific user/ group to use the system or part of the system while imparting the training of the program on the application. This training may be different across different user groups and across different levels of hierarchy.

### **6.2.3 SYSTEM MAINTENANCE**

Maintenance is the enigma of system development. The maintenance phase of the software cycle is the time in which a software product performs useful work. After a system is successfully implemented, it should be maintained in a proper manner. System maintenance is an important aspect in the software development life cycle. The need for system maintenance is for it to make adaptable to the changes in the system environment. Software maintenance is of course, far more than “Finding Mistakes”.

## **CHAPTER 7**

### **CONCLUSION AND FUTURE SCOPE**



---

## 7.1 CONCLUSION

In conclusion, the proposed e-Diagnostic Laboratory Management System, named OneHealth, stands as a groundbreaking solution in the healthcare industry. Offering a seamless and user-centric approach to diagnostic services, OneHealth revolutionizes the way patients access and manage their healthcare journey. The platform's diverse features, including online appointment booking, secure payment processing, health monitoring device purchases, and efficient sample collection, contribute to a holistic and accessible healthcare experience. OneHealth is designed to cater to a wide audience, comprising healthcare providers, administrative staff, patients, and delivery personnel, with a commitment to enhancing efficiency and user empowerment. The advantages of this proposed system lie in its user-friendly interface, flexible appointment scheduling, secure online transactions, and the integration of an e-commerce platform for health monitoring devices. By amalgamating these features, OneHealth aims to redefine healthcare management, fostering accessibility, efficiency, and satisfaction for users in the dynamic landscape of online diagnostics.

## 7.2 FUTURE SCOPE

The future scope of OneHealth is poised for transformative advancements, embracing cutting-edge technologies to elevate the healthcare management experience. Integration of artificial intelligence can revolutionize the recommendation system, offering personalized and nuanced suggestions based on individual health histories and evolving needs. The platform can expand its community engagement features, introducing dynamic forums, support groups, and user-generated content to foster a collaborative and supportive healthcare community. Future iterations may include predictive analytics for proactive health management and telemedicine features for virtual consultations. Global expansion initiatives, including multilingual support, can enable OneHealth to cater to diverse populations, ensuring accessibility and inclusivity on a global scale. Additionally, strategic collaborations with healthcare institutions, research organizations, and technology partners can pave the way for innovative solutions, exclusive services, and continuous improvement, establishing OneHealth as a dynamic and forward-thinking player in the evolving landscape of digital healthcare.

## **CHAPTER 8**

## **BIBLIOGRAPHY**

---

**REFERENCES:**

- ✧ PubMed and Research Journals: Explore databases like PubMed for scientific articles and research papers related to e-Diagnostic Laboratory Management Systems.
- ✧ IEEE Xplore: IEEE Xplore is a digital library that often contains research articles and conference papers related to healthcare systems and diagnostics.
- ✧ Google Scholar: Google Scholar is a freely accessible search engine that indexes scholarly articles. It's a good resource for finding academic publications.
- ✧ Healthcare IT News and Journals: Websites like Healthcare IT News and various healthcare-related journals might feature articles on advancements in diagnostic laboratory management.

**WEBSITES**

- <https://blog.mathieu-leplatre.info/geodjango-maps-with-leaflet.html>
- <https://www.geeksforgeeks.org/django-tutorial/>
- <https://razorpay.com/>

## **CHAPTER 9**

### **APPENDIX**

## 9.1 Sample Code

### Header.html

```
{% load static %}
<header>
<div class="topbar">
<div class="container">
<div class="row">
<div class="col-sm-8 text-sm">
<div class="site-info">
<a href="#"><span class="mai-call text-primary"></span>+91 8330064687</a>
<span class="divider">|</span>
<a href="#"><span class="mai-mail text-primary"></span> onehealth@gmail.com</a>
</div>
</div>

</div>
</div> <!-- .row -->
</div> <!-- .container -->
</div> <!-- .topbar -->

<nav class="navbar navbar-expand-lg navbar-light shadow-sm">
<div class="container">
<a class="navbar-brand" href="#"><span class="text-primary">One</span>-Health</a>

<button class="navbar-toggler" type="button" data-toggle="collapse" datatarget="#navbarSupport"
aria-controls="navbarSupport" aria-expanded="false" aria-label="Toggle navigation">
<span class="navbar-toggler-icon"></span>
</button>

<div class="collapse navbar-collapse" id="navbarSupport">
<ul class="navbar-nav ml-auto">
<li class="nav-item ">
<a class="nav-link" href="{% url 'home' %}">Home</a>
</li>
<li class="nav-item">
<a class="nav-link" href="{% url 'about' %}">About Us</a>
</li>
<li class="nav-item">
<a class="nav-link" href="{% url 'contact' %}">Contact</a>
</li>
{% if not user.is_authenticated %}
<li class="nav-item">
<a class="nav-link" href="{% url 'services1' %}">Services</a>
</li>
{% endif %}
{% if user.is_authenticated %}
<li class="nav-item">
<a class="nav-link" href="{% url 'services' %}">Services</a>
</li>

<li class="nav-item">
<a class="nav-link" href="{% url 'appointment' %}">Appointment</a>
</li>
</ul>
</div>
</div>
</nav>
```

```

</li>
<!-- Updated HTML -->
<div class="dropdown">
<button class="btn btn-secondary dropdown-toggle" type="button" id="userProfileDropdown" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">
<div class="user-profile">

{{ user.username }}
</div>
</button>
<div class="dropdown-menu" aria-labelledby="userProfileDropdown">
<a class="dropdown-item" href="{% url 'userprofile' %}">My Profile</a>
<a class="dropdown-item" href="{% url 'logout' %}">Logout</a>
</div>
</div>

{% endif %}

<!--<li class="nav-item">
<a class="btn btn-primary ml-lg-3" href="login.html">Login/Signup</a>
</li> -->
</ul>
{% if not user.is_authenticated %}
<a href="login.html" class="btn btn-primary ml-lg-3">Login/Signup</a>
{% endif %}
</div>
</div> <!-- .navbar-collapse -->
</div> <!-- .container -->
</nav>
</header>

```

## Footer.html

```

<footer class="page-footer">
<div class="container">
<div class="row px-md-3">
<div class="col-sm-6 col-lg-3 py-3">
<h5>Company</h5>
<ul class="footer-menu">
<li><a href="about.html">About Us</a></li>
<li><a href="services1.html">services</a></li>

<li><a href="#">Products</a></li>
</ul>
</div>
<div class="col-sm-6 col-lg-3 py-3">
<h5>More</h5>
<ul class="footer-menu">
<li><a href="#">Terms & Condition</a></li>
<li><a href="#">Privacy</a></li>

```

```

</ul>
</div>

<div class="col-sm-6 col-lg-3 py-3">
<h5>Contact</h5>
<p class="footer-link mt-2">MC road Kottayam</p>
<a href="#" class="footer-link">8330064687</a>
<a href="#" class="footer-link">onehealth@gmail.com</a>
</div>
<div class="footer-sosmed mt-3">

<h5 class="mt-3">Social Media</h5>
<a href="https://www.facebook.com/your-facebook-page" target="_blank"><span class="mai-logo-
facebook-f"></span></a>
<a href="https://twitter.com/your-twitter-handle" target="_blank"><span class="mai-logo-
twitter"></span></a>
<a href="https://plus.google.com/your-google-plus-page" target="_blank"><span class="mai-logo-google-
plus-g"></span></a>
<a href="https://www.instagram.com/your-instagram-account" target="_blank"><span class="mai-logo-
instagram"></span></a>
<a href="https://www.linkedin.com/in/your-linkedin-profile" target="_blank"><span class="mai-logo-
linkedin"></span></a>

</div>
</div>

<hr>

</div>
</footer>

```

### Base.html

```

{% load static %}

<!DOCTYPE html>
<html lang="en">

<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<meta name="copyright" content="MACode ID, https://macodeid.com/">

<title>One Health - Medical Center HTML5 Template</title>

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

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

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

```

---

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

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

```
<style>
/* Custom CSS for the message toaster */
.message-toaster {
position: fixed;
top: 0;
left: 50%;
text-align: center;
z-index: 9999;
transform: translateX(-50%);
padding: 30px 0;
}
/* Updated CSS */
/* Style for the dropdown button */
.dropdown .btn-secondary {
background-color: #09ae96;
color: #fff;
border: none;
border-radius: 3px;
cursor: pointer;
display: flex;
align-items: center;
justify-content: space-between;
transition: background-color 0.3s ease, color 0.3s ease;
padding: 5px 15px;
}

.dropdown .btn-secondary:hover {
background-color: #035b52; /* Change background color on hover */
color: #fff; /* Change text color on hover */
}

/* Style for the user icon */
.user-icon {
width: 30px;
height: 30px;
margin-right: 10px;
border-radius: 50%;
}

/* Style for the user profile dropdown menu */
.dropdown-menu {
background-color: #333;
color: #fff;
}

.dropdown-item {
color: #fff;
transition: background-color 0.3s ease, color 0.3s ease;
}
```



```

.dropdown-item:hover {
background-color: #042f2d; /* Change background color on hover */
color: #fff; /* Change text color on hover */
}

</style>

{% block extra_css %}
{% endblock %}
</head>

<body>
{% include "layout/messages.html" %}

<!-- Back to top button -->
<div class="back-to-top"></div>

{% include "layout/header.html" %}

{% block content %}
{% endblock %}

{% include "layout/footer.html" %}

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

<script src="{% static '/assets/js/jquery-3.5.1.min.js"%}"></script>

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

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

<script src="{% static '/assets/vendor/wow/wow.min.js"%}"></script>

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

{% block extra_js %}
{% endblock %}

</body>

</html>

```

### index.html

```

{% extends "layout/base.html" %}
{% load static %}
{% block extra_css %}
<style>
.rating {
cursor: pointer;
}

.rating .star {

```

```

display: inline-block;
font-size: 24px;
color: #ccc;
}

.rating .star.selected {
color: #ffcc00;
}

.existing-reviews {
margin-top: 20px;
}

.existing-review {
margin-bottom: 20px;
}
}
</style>
{% endblock %}
{% block content %}

<div class="page-hero bg-image overlay-dark" style="background-image: url('{% static
"assets/img/bg_image_2.jpg" %}');">
<div class="hero-section">
<div class="container text-center wow zoomIn">
<span class="subhead">Let's make your life happier</span>
<h3 class="display-4">Healthy Living</h3>
<!-- <a href="appointment.html" class="btn btn-primary">Let's Consult</a -->
</div>
</div>
</div>

<div class="bg-light">
<div class="page-section py-3 mt-md-n5 custom-index">
<div class="container">
<div class="row justify-content-center">
<div class="col-md-4 py-3 py-md-0">
<div class="card-service wow fadeInUp">
<div class="circle-shape bg-secondary text-white">
<span class="mai-chatbubbles-outline"></span>
</div>
<p><span>Easy</span>Booking</p>
</div>
</div>
<div class="col-md-4 py-3 py-md-0">
<div class="card-service wow fadeInUp">
<div class="circle-shape bg-primary text-white">
<span class="mai-shield-checkmark"></span>
</div>
<p><span>One</span>-Health Protection</p>
</div>
</div>
<div class="col-md-4 py-3 py-md-0">
<div class="card-service wow fadeInUp">
<div class="circle-shape bg-accent text-white">
<span class="mai-basket"></span>
</div>

```

```

<p><span>One</span>-Health Products</p>
</div>
</div>
</div>
</div>
</div> <!-- .page-section -->

<div class="page-section pb-0">
<div class="container">
<div class="row align-items-center">
<div class="col-lg-6 py-3 wow fadeInUp">
<h1>Welcome to OneHealth</h1>
<p class="text-grey mb-4">Welcome to our Health Center, where health and well-being are at the heart of
everything we do. We are more than just a healthcare facility; we are a community dedicated to enhancing
the lives of our patients. Our journey began with a simple yet powerful vision: to provide accessible,
comprehensive, and compassionate healthcare services. Over the years, we have grown and evolved,
expanding our services, and embracing the latest medical advancements to better serve you. Our team of
highly skilled healthcare professionals shares a common commitment: your health and happiness.</p>
<a href="about.html" class="btn btn-primary">Learn More</a>
</div>
<div class="col-lg-6 wow fadeInRight" data-wow-delay="400ms">
<div class="img-place custom-img-1">

</div>
</div>
</div>
</div>
</div> <!-- .bg-light -->
</div> <!-- .bg-light -->

<div class="container">
<h2 class="text-center mb-5 wow fadeInUp">Leave a Review</h2>
<!-- Add the appointment ID to the form -->
<form class="review-form" method="POST" action="review/">
{% csrf_token %}
<input type="hidden" name="appointment_id" value="{{ appointment.id }}">

<div class="form-group">
<label for="user_review">Your Review:</label>
<textarea class="form-control" id="comment" name="comment" rows="4" placeholder="Write your
review here..."></textarea>
</div>

<div class="form-group">
<label for="user_rating">Your Rating:</label>
<div class="rating" id="rating" name="rating">
<span class="star" data-value="5">&#9733;</span>
<span class="star" data-value="4">&#9733;</span>
<span class="star" data-value="3">&#9733;</span>
<span class="star" data-value="2">&#9733;</span>
<span class="star" data-value="1">&#9733;</span>
</div>
<input type="hidden" name="rating" id="rating-value" value="">
</div>

```

```

<button type="submit" class="btn btn-primary">Submit Review</button>
</form>

<!-- Display existing reviews and ratings -->
<div class="existing-reviews">
  {% for review in reviews %}
  <div class="existing-review card mb-3">
    <div class="card-body">
      <h5 class="card-title">{{ review.user.username }}</h5>
      <div class="rating">
        {% for _ in "12345"|slice:review.rating %}
        <span class="star selected">&#9733;</span>
        {% endfor %}
      </div>
      <p class="card-text">{{ review.comment }}</p>
      <p class="card-text text-muted">Created at: {{ review.created_at|date:"F j, Y" }}</p>
    </div>
  </div>
  {% endfor %}
</div>
</div>
{% endblock %}
{% block extra_js %}
<script src="https://code.jquery.com/jquery-3.6.4.min.js"></script>

<script>
$(document).ready(function() {
$('.rating .star').on('click', function() {
const value = $(this).data('value');
const $hiddenInput = $('#rating-value');

// Toggle selection for the clicked star
if ($(this).hasClass('selected')) {
// If the clicked star is already selected, deselect it
$(this).removeClass('selected');
} else {
// If the clicked star is not selected, select it and deselect all higher stars
$('.rating .star:gt(' + (value - 1) + ')).removeClass('selected');
$(this).addClass('selected');
}

// Update the hidden input with the overall rating based on the number of selected stars
$hiddenInput.val($('.rating .star.selected').length);
});
});
</script>
{% endblock %}

```

### appointment.html

```

{% extends "layout/base.html" %}
{% load static %}

{% block extra_css %}
<link rel="stylesheet" href="https://unpkg.com/leaflet@1.9.4/dist/leaflet.css"

```

```
integrity="sha256-p4NxAoJBhIIN+hmNHrzRCf9tD/miZyoHS5obTRR9BMY="
crossorigin=""/>
<style>
body {
font-family: Arial, sans-serif;
background-image: url('{% static "assets/img/bg_image_2.jpg" %}');
/* Replace with the actual path to your background image */
background-size: cover;
/* This will cover the entire viewport */
background-repeat: no-repeat;
background-attachment: fixed;
/* This will keep the background fixed while scrolling */
}

header {
background-color: #fff;
/* Set the background color to white */
}

/* CSS for the form container */
.form-container {
max-width: 600px;
/* Adjust the width as needed */
margin: 0 auto;
/* Center the container horizontally */
padding: 20px;
/* Add padding to the container */
border: 1px solid #ccc;
/* Add a border around the container */
border-radius: 5px;
/* Add rounded corners to the container */
box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
/* Add a subtle box shadow */
background-color: #f9f9f9;
/* Set the background color */
}

/* Apply styles to form elements within the container */
.form-container input[type="text"],
.form-container input[type="number"],
.form-container input[type="date"],
.form-container input[type="button"],
.form-container select,
.form-container textarea,
.form-container label {
width: 100%;
margin-bottom: 10px;
padding: 10px;
border: 1px solid #ccc;
border-radius: 3px;
font-size: 16px;
}

/* Style checkboxes and radio buttons */
.form-container input[type="checkbox"],
```

```
.form-container input[type="radio"] {
  margin-right: 5px;
}

/* Style the submit button */
.form-container input[type="submit"] {
  background-color: #007BFF;
  color: #fff;
  border: none;
  border-radius: 3px;
  padding: 10px 20px;
  cursor: pointer;
  font-size: 18px;
  transition: background-color 0.3s;
}

.form-container input[type="submit"]:hover {
  background-color: #0056b3;
}

.additional-test-container {
  display: none;
  margin-top: 10px;
}

.additional-test-container label {
  display: block;
}

.additional-test-container select {
  width: 100%;
  padding: 10px;
  border: 1px solid #ccc;
  border-radius: 3px;
  font-size: 16px;
}

.radio-box {
  display: flex;
  align-items: center;
}

.radio-label {
  margin-right: 10px;
}

.new-address-fields.visible {
  display: block;
}

.new-address-fields .form-group {
  margin-right: 20px;
}

.error-message {
  color: red;
}
```

```

}

</style>
{% endblock %}

{% block content %}
<div class="form-container">
<!-- Your form content goes here -->
<form method="POST" enctype="multipart/form-data">
{% csrf_token %}
<div class="frminput">

<h3>
<center><b><span style="color: rgb(146, 197, 236);">Book Now!</span></b></center>
</h3> <br><br>
<select name="test" id="testname" required>
{% for test in tests %}
<option value="{{ test.id }}" {% if request.GET.test == test.name %}selected{% endif %}>
{{ test.name }}
</option>
{% endfor %}
</select>
<div>
<label for="prescription" style="display: inline-block; margin-right: 10px;">Upload Prescription (PDF or
Image):</label>
<input type="file" id="prescription" name="prescription" accept=".pdf, .jpg, .jpeg, .png" style="display:
inline-block;" />
</div><br>
<p>Preferred Date: <input type="date" id="preferred-date" name="date" max="{{ fiveDaysLater }}"
min="{{ today }}" pattern="\d{2}/\d{2}/\d{4}" title="Please enter a valid date in mm/dd/yyyy format"
required></p>

<span id="preferred-date-error" class="error-message"></span>
<p>Preferred Time:
<select id="preferred-time" name="preferred-time">
</select>
</p>

<div class="person-fields">
<div>
<input type="text" id="full_name" name="full_name" placeholder="Full name" required>
<span id="full_name_error" class="error-message"></span>
<select id="gender" name="gender" required>
<option value="" selected>Select Gender</option>
<option value="Male">Male</option>
<option value="Female">Female</option>
</select>
<input type="number" id="age" name="age" placeholder="Age" required>
</div>
</div>

<input type="text" id="email" name="email" placeholder="Email" required readonly
value="{{ user.email }}">

```

```

<span id="email_error" class="error-message"></span>
<input type="text" id="phone" name="phone" placeholder="Mobile" required readonly
value="{{user.phone_number}}">
<span id="phone_error" class="error-message"></span>

<div class="radio-box">
<label class="radio-label">
<input type="radio" id="current-address-choice" name="address-choice" value="current-address"
checked>
Use Current Address
</label>
<label class="radio-label">
<input type="radio" id="new-address-choice" name="address-choice" value="new-address">
Enter New Address
</label>
</div>

<div id="current-address-fields">
<!-- Fields for the current address -->
<input type="text" id="current-home-address" name="current-home-address" placeholder="Current Home
Address" readonly value="{{user.address.street_address}}">
<input type="text" id="current-city" name="current-city" placeholder="Current City" readonly
value="{{user.address.city}}">
<input type="text" id="current-pincode" name="current-pincode" placeholder="Current Pincode" readonly
value="{{user.address.pincode}}">
</div>

<div id="new-address-fields" class="new-address-fields" style="display: none;">
<!-- Fields for the new address -->
<input type="text" id="new-home-address" name="new-home-address" placeholder="New Home
Address">
<div class="error-message" id="new-home-address-error" style="color: red; display: none;"></div>
<input type="text" id="new-city" name="new-city" placeholder="New City">
<div class="error-message" id="new-city-error" style="color: red; display: none;"></div>
<input type="text" id="new-pincode" name="new-pincode" placeholder="New Pincode">
<div class="error-message" id="pincode-error" style="color: red; display: none;"></div>
</div>
</div>
<!-- Additional tests checkboxes -->

<label class="agreecheck">
<input type="checkbox" id="additionalTestsCheckbox" name="additional_tests"
onclick="toggleAdditionalTestDropdown();">
<span>Add Additional Tests</span>
</label>

<!-- Additional test dropdown container -->
<div class="additional-test-container" id="additionalTestContainer">

<select id="additionalTestsSelect" name="additional_tests_select">
<option value="" selected>Select Test</option>
{% for test in tests %}
<option data-price="{{ test.price }}" value="{{ test.id }}">{{ test.name }}</option>
{% endfor %}
</select>

```



```
</div>
<br>

<span>Choose mode of sample collection :</span><br><br>
<div>
<label style="display: inline;">
<input type="radio" id="homeAppointment" name="appointmentType" value="Home" checked> Home
</label>
<label style="display: inline;">
<input type="radio" id="labAppointment" name="appointmentType" value="Lab"> Lab
</label>
</div>

<!-- Button trigger modal -->
<button type="button" class="btn btn-primary btn-sm mt-3" data-toggle="modal" data-
target="#mapModal"
id="select-location">
Select Location
</button>

<!-- <select name="location" id="location" class="mt-3" required>
{% for location in locations %}
<option data-price="{{ location.amount }}" value="{{ location.id }}">
{{ location.name }}
</option>
{% endfor %}
</select> -->

<input type="number" hidden name="amount" id="total-amount" value="0.0">
<input type="text" hidden name="location-address" id="location-address">
<input type="text" hidden name="location-distance" id="location-distance">
<input type="text" hidden name="location-lat" id="location-lat">
<input type="text" hidden name="location-lng" id="location-lng">

</form>

<br>

<!-- Total amount calculation -->
<div>
<label style="padding-bottom: 8px; display: block;">
<span id="totalamnt"><strong>Test price: <span id="tprice"> Rs. </span></strong></span>
</label>

<label id="hccrglbl">
<span id="totalamnt"><strong>Home Collection Charge: <span id="hcprice">Rs.
0</span></strong></span>
</label>

<label>
<span id="totalamnt"><strong>Total Amount: <span id="totalprice"> Rs.</span></strong></span>
</label>
</div>
```

```

<br>
<input type="hidden" id="coupontextbox" name="coupontextbox" value="">
<input type="hidden" id="product_id" name="product_id" value="77">
<input type="hidden" id="price" name="price" value="300">
<input type="hidden" name="currenturl" value="https://www.diagnosticcentres.in/health-checkup-packages/cbc-hemogram">
<input id="frmsubbtn" type="submit" class="frmsub" value="Book Now">
<input type="hidden" name="sunday" id="sunday" value="">
</div>
</form>
</div>
</div>

<!-- Modal -->
<div class="modal fade" id="mapModal" tabindex="-1" role="dialog" aria-labelledby="mapModalTitle"
aria-hidden="true">
<div class="modal-dialog modal-dialog-centered" role="document">
<div class="modal-content">
<div class="modal-header">
<h5 class="modal-title" id="exampleModalLongTitle">Select Location</h5>
<button type="button" class="close" data-dismiss="modal" aria-label="Close">
<span aria-hidden="true">&times;</span>
</button>
</div>
<div class="modal-body">
<div id="map" style="height: 240px;"></div>
</div>
<div class="modal-footer">
<button type="button" class="btn btn-primary btn-sm" id="confirm-location">Confirm Location</button>
</div>
</div>
</div>
</div>

{% endblock %}
{% block extra_js %}
<script src="https://unpkg.com/leaflet@1.9.4/dist/leaflet.js"
integrity="sha256-20nQCchB9co0qIjJZRGuk2/Z9VM+kNiyxNV1lvTlZBo="
crossorigin=""></script>
<script>

function toggleAdditionalTestDropdown() {
const additionalTestsCheckbox = document.getElementById("additionalTestsCheckbox");
const additionalTestContainer = document.getElementById("additionalTestContainer");

if (additionalTestsCheckbox.checked) {
additionalTestContainer.style.display = "block";
} else {
additionalTestContainer.style.display = "none";
}
}

// Get the select element by its ID
const selectElement = document.getElementById('preferred-time');
```

```
// Function to add time options to the select field
function addTimeOptions() {
// Start time (09:00 AM) in 24-hour format
let startTime = 9;

// End time (05:30 PM) in 24-hour format
const endTime = 17.5; // 05:30 PM is equivalent to 17.5 hours

// Loop to generate time options
while (startTime <= endTime) {
// Convert the current time to a formatted string (hh:mm AM/PM)
const formattedTime = formatTime(startTime);

// Create an option element
const option = document.createElement('option');
option.value = startTime.toFixed(2).replace(".", ":"); // Use toFixed(2) to ensure two decimal places for half hours
option.text = formattedTime;

// Add the option to the select field
selectElement.appendChild(option);

// Increment time by 0.5 (30 minutes)
startTime += 0.5;
}
}

// Function to format time as hh:mm AM/PM
function formatTime(hours) {
const hour = Math.floor(hours);
const minutes = (hours % 1) * 60;
const ampm = hour >= 12 ? 'PM' : 'AM';
const formattedHour = hour % 12 === 0 ? 12 : hour % 12; // Convert 0 to 12 for 12:00 AM/PM
const formattedMinutes = minutes < 10 ? `0${minutes}` : minutes;

return `${formattedHour}:${formattedMinutes} ${ampm}`;
}

// Call the function to add time options
addTimeOptions();

function toggleAddressFields() {
const currentAddressChoice = document.getElementById("current-address-choice");
const newAddressChoice = document.getElementById("new-address-choice");
const currentAddressFields = document.getElementById("current-address-fields");
const newAddressFields = document.getElementById("new-address-fields");

if (currentAddressChoice.checked) {
currentAddressFields.style.display = "block";
newAddressFields.style.display = "none";
} else if (newAddressChoice.checked) {
currentAddressFields.style.display = "none";
newAddressFields.style.display = "block";
} else {
currentAddressFields.style.display = "none";
newAddressFields.style.display = "none";
}
```

```
}  
}  
  
// Add event listeners to the radio buttons to detect changes  
const currentAddressChoice = document.getElementById("current-address-choice");  
const newAddressChoice = document.getElementById("new-address-choice");  
  
currentAddressChoice.addEventListener("change", toggleAddressFields);  
newAddressChoice.addEventListener("change", toggleAddressFields);  
  
// Call the function to set the initial state  
toggleAddressFields();  
  
// Add an event listener to the radio buttons to detect changes  
const fullNameField = document.getElementById("full_name");  
const fullNameError = document.getElementById("full_name_error");  
fullNameField.addEventListener("input", () => {  
  const fullName = fullNameField.value.trim();  
  const namePattern = /^[A-Za-z\s]+$/; // Regular expression to allow letters and spaces  
  
  if (!namePattern.test(fullName)) {  
    fullNameError.textContent = "Full name must contain only letters and spaces";  
    fullNameField.classList.add("invalid");  
  } else if (fullName.length < 2) {  
    fullNameError.textContent = "Full name must be at least 2 characters";  
    fullNameField.classList.add("invalid");  
  } else {  
    fullNameError.textContent = "";  
    fullNameField.classList.remove("invalid");  
  }  
});  
const emailField = document.getElementById("email");  
const emailError = document.getElementById("email_error");  
  
emailField.addEventListener("input", () => {  
  const email = emailField.value.trim();  
  const emailPattern = /^[a-zA-Z0-9._-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,4}$/;  
  
  if (!emailPattern.test(email)) {  
    emailError.textContent = "Please enter a valid email address";  
    emailField.classList.add("invalid");  
  } else {  
    emailError.textContent = "";  
    emailField.classList.remove("invalid");  
  }  
});  
  
const phoneField = document.getElementById("phone");  
const phoneError = document.getElementById("phone_error");  
  
phoneField.addEventListener("input", () => {  
  const phone = phoneField.value.trim();  
  const phonePattern = /^[0-9]{10}$/; // Regular expression to allow only 10 digits  
  
  if (!phonePattern.test(phone)) {
```

```
phoneError.textContent = "Please enter a valid 10-digit phone number";
phoneField.classList.add("invalid");
} else {
phoneError.textContent = "";
phoneField.classList.remove("invalid");
}
});

const testDropdown = document.getElementById("testname");
const testPriceElement = document.getElementById("tprice");
const homeCollectionChargeElement = document.getElementById("hcprice");
const totalAmountElement = document.getElementById("totalprice");
const additionalTestsSelect = document.getElementById("additionalTestsSelect");

// Function to fetch and update the test price based on the selected test
function updateTestPrice() {
const selectedOption = testDropdown.options[testDropdown.selectedIndex];
const testId = selectedOption.value;

fetch(`/get_test_price/?test_id=${testId}`) // Replace with your actual URL
.then(response => response.json())
.then(data => {
if (data.price !== null) {
testPriceElement.textContent = `Rs. ${parseFloat(data.price) || 0}`;
calculateTotalAmount(); // Recalculate total amount
} else {
testPriceElement.textContent = "Price not available";
totalAmountElement.textContent = "Total Amount: N/A";
}
})
.catch(error => {
console.error("Error fetching test price:", error);
testPriceElement.textContent = "Error fetching price";
totalAmountElement.textContent = "Total Amount: N/A";
});
}

// Function to calculate and update the total amount
function calculateTotalAmount() {
const testPrice = parseFloat(testPriceElement.textContent.replace("Rs. ", " ")) || 0;
const homeCollectionCharge = parseFloat(homeCollectionChargeElement.textContent.replace("Rs. ", " ")) || 0;
const totalAmountInput = document.getElementById("total-amount");

// Calculate the total amount including the selected additional tests
const additionalTestsTotalPrice = Array.from(additionalTestsSelect.selectedOptions)
.map(option => parseFloat(option.dataset.price) || 0)
.reduce((acc, price) => acc + price, 0);

const totalAmount = testPrice + homeCollectionCharge + additionalTestsTotalPrice;

if (!isNaN(totalAmount)) {
totalAmountElement.textContent = `Rs. ${totalAmount}`;
totalAmountInput.value = totalAmount;
} else {
```

```
totalAmountElement.textContent = "Total Amount: N/A";
}
}

// Add an event listener to the test dropdown to update the test price and total amount
testDropdown.addEventListener("change", updateTestPrice);

// Add an event listener to the additional tests dropdown to recalculate total amount on changes
additionalTestsSelect.addEventListener("change", calculateTotalAmount);

// Call the updateTestPrice function initially to set the default price
updateTestPrice();
const homeAppointment = document.getElementById("homeAppointment");
const labAppointment = document.getElementById("labAppointment");
const locationSelect = document.getElementById("select-location");

function updateHomeCollectionCharge() {
  const FIXED_CHARGE = 20;
  const PER_KM = 10;
  let distance = window.kilometer;
  let amount;

  if (!homeAppointment.checked || !distance){
    amount = 0;
  }
  else if (distance <= 1){
    amount = FIXED_CHARGE;
  }
  else {
    amount = (FIXED_CHARGE + (distance - 1) * PER_KM).toFixed(2);
  }

  if (!isNaN(amount)) {
    homeCollectionChargeElement.textContent = ` ${amount}`;
    calculateTotalAmount(); // Recalculate total amount
  } else {
    homeCollectionChargeElement.textContent = " N/A";
    totalAmountElement.textContent = " N/A";
  }
}

// Add an event listener to the location dropdown to update home collection charge
// locationSelect.addEventListener("change", updateHomeCollectionCharge);

// Call the updateHomeCollectionCharge function initially to set the default home collection charge
updateHomeCollectionCharge();

function toggleLocationSelection() {
  if (homeAppointment.checked) {
    locationSelect.style.display = "block";
  } else {
    locationSelect.style.display = "none";
  }
  updateHomeCollectionCharge();
}
```

```
// Add event listeners to the radio buttons
homeAppointment.addEventListener("change", toggleLocationSelection);
labAppointment.addEventListener("change", toggleLocationSelection);

// Call the function to set the initial state
toggleLocationSelection();
const form = document.querySelector('form');
const submitButton = document.getElementById('frmsubbtn');

form.addEventListener('submit', function (event) {
// Prevent the default form submission
event.preventDefault();

// You can perform any additional actions here, such as form validation

// If everything is valid, you can submit the form
form.submit();
});

// Alternatively, you can also handle form submission when the "Book Now" button is clicked
submitButton.addEventListener('click', function (event) {
event.preventDefault();
form.submit();
});
document.addEventListener('DOMContentLoaded', function () {
// Function to validate the preferred date
function validatePreferredDate() {
const preferredDateInput = document.getElementById("preferred-date");
const currentDate = new Date();
const selectedDate = new Date(preferredDateInput.value);

// Calculate the allowed date range (today to 5 days later)
const allowedStartDate = new Date(currentDate);
const allowedEndDate = new Date(currentDate);
allowedEndDate.setDate(currentDate.getDate() + 5);

// Check if the selected date is within the allowed range
if (selectedDate < currentDate || selectedDate > allowedEndDate) {
document.getElementById("preferred-date-error").textContent = "Please select a valid date within today
and the next 5 days";
return false; // Prevent form submission
} else {
document.getElementById("preferred-date-error").textContent = ""; // Clear error message
return true; // Allow form submission
}
}

// Add an input event listener to the "Preferred Date" input field for real-time validation
const preferredDateInput = document.getElementById("preferred-date");
preferredDateInput.addEventListener('input', validatePreferredDate);

// Add an event listener to the form to trigger the date validation on submission
const form = document.querySelector('form');
const submitButton = document.getElementById('frmsubbtn');
```

```
form.addEventListener('submit', function (event) {
// Prevent the default form submission
event.preventDefault();

// Call the date validation function
if (validatePreferredDate()) {
// If validation passes, submit the form
form.submit();
}
});

// Alternatively, you can also handle form submission when the "Book Now" button is clicked
submitButton.addEventListener('click', function (event) {
event.preventDefault();
// Call the date validation function
if (validatePreferredDate()) {
// If validation passes, submit the form
form.submit();
}
});
});

</script>
<script>
var marker, map, kilometer;
const SOURCE = [9.0827879, 76.5650678];
const RADIUS = 5;

function haversineDistance(lat1, lon1, lat2, lon2) {
// Convert latitude and longitude from degrees to radians
const toRadians = (angle) => angle * (Math.PI / 180);
lat1 = toRadians(lat1);
lon1 = toRadians(lon1);
lat2 = toRadians(lat2);
lon2 = toRadians(lon2);

// Haversine formula
const dLat = lat2 - lat1;
const dLon = lon2 - lon1;
const a =
Math.sin(dLat / 2) ** 2 +
Math.cos(lat1) * Math.cos(lat2) * Math.sin(dLon / 2) ** 2;
const c = 2 * Math.asin(Math.sqrt(a));

// Radius of the Earth in kilometers (mean value)
const radiusOfEarth = 6371;

// Calculate the distance
const distance = radiusOfEarth * c;

return distance;
}
```



```

$('#mapModal').on('shown.bs.modal', function (e) {
  if (!map){
    map = L.map('map').setView([9.0827879, 76.5650678], 15);

    L.tileLayer('https://tile.openstreetmap.org/{z}/{x}/{y}.png', {
      maxZoom: 19,
      attribution: '&copy; Anjali Raj'
    }).addTo(map);

    L.circle(SOURCE, {
      color: 'blue',
      fillColor: '#FFFFFF00',
      // fillOpacity: 0.5,
      radius: RADIUS * 1000,
    }).addTo(map);

    var location = navigator.geolocation.getCurrentPosition(
      function success(pos){
        marker = L.marker(
          [pos.coords.latitude, pos.coords.longitude],
          // {draggable:true}
        ).addTo(map);
      }
    );

    map.on('click', function onMapClick(e){
      kilometer = haversineDistance(SOURCE[0], SOURCE[1], e.latlng.lat, e.latlng.lng);
      if (kilometer > RADIUS){
        alert('Select a radius within ${RADIUS} Kilometers.');
```

```

      return;
    }
    if (marker){
      map.removeLayer(marker);
    }
    marker = L.marker(e.latlng).addTo(map);
  });

  $('#confirm-location').on('click', function onConfirmation(e){
    destination = marker.getLatLng();
    kilometer = haversineDistance(SOURCE[0], SOURCE[1], destination.lat, destination.lng);
    $('#mapModal').modal('hide');
    updateHomeCollectionCharge();
    $.get(
      `https://nominatim.openstreetmap.org/reverse?format=jsonv2&lat=${destination.lat}&lon=${destination.lng}`,
      function (data){
        console.log(data)
        $('#location-address').val(data.display_name);
        $('#location-distance').val(kilometer);
        $('#location-lat').val(data.lat);
        $('#location-lng').val(data.lon);
      }
    );
  })
}
```

```
    })
</script>
<script type="text/javascript">
$(document).ready(function () {
// Validation for New Home Address
$('#new-home-address').on('input', function () {
var newHomeAddress = $(this).val();
var validHomeAddress = /^[A-Za-z0-9\s]{3,}$/.test(newHomeAddress); // At least 3 alphanumeric
characters

var errorContainer = $('#new-home-address-error');

if (!validHomeAddress) {
// Display an error message below the input field
errorContainer.text('Address must start with letters or numbers and contain at least 3 characters.');
```

```
errorContainer.show();
} else {
// Clear the error message and hide the error container if the input is valid
errorContainer.text("");
errorContainer.hide();
}
});

// Validation for New City
$('#new-city').on('input', function () {
var newCity = $(this).val();
var validCity = /^[A-Za-z\s]{3,}$/.test(newCity); // At least 3 letters

var errorContainer = $('#new-city-error');

if (!validCity) {
// Display an error message below the input field
errorContainer.text('City must start with letters and contain at least 3 characters.');
```

```
errorContainer.show();
} else {
// Clear the error message and hide the error container if the input is valid
errorContainer.text("");
errorContainer.hide();
}
});

// Validation for New Pincode
$('#new-pincode').on('input', function () {
var newPincode = $(this).val();
var validPincode = /^[1-9]\d{5}$/.test(newPincode); // Starts with a digit (excluding 0) and has 6 digits

var errorContainer = $('#pincode-error');

if (!validPincode) {
// Display an error message below the input field
errorContainer.text('Invalid pincode. It must start with a non-zero digit and contain 6 digits.');
```

```
errorContainer.show();
} else {
// Clear the error message and hide the error container if the input is valid
errorContainer.text("");
}
```

```

errorContainer.hide();
}
});
});
</script>

```

```
{% endblock %}
```

### payment.html

```
{% extends "layout/base.html" %}
```

```
{% block content %}
```

```

<div class="container" id="app">
<h1>Payment Confirmation</h1>

```

```

<p>Thank you for booking your appointment! Please choose your method of payment and enter the
required details:</p>

```

```
<!-- Display appointment details here -->
```

```

<p><strong>Test:</strong> {{ appointment.main_test.name }}</p>
<p><strong>Preferred Date:</strong> {{ appointment.preffered_date }}</p>
<p><strong>Preferred Time:</strong> {{ appointment.preffered_time }}</p>
<p><strong>Full Name:</strong> {{ appointment.user.first_name }}</p>
<p><strong>Email:</strong> {{ appointment.user.email }}</p>
<p><strong>Phone:</strong> {{ appointment.user.phone_number }}</p>
<p><strong>Sample Collection Type:</strong> <span style="text-transform:
capitalize;">{{ appointment.appoinment_type }}</span> Collection</p>

```

```
<!-- Payment Method Form -->
```

```

<button type="submit" class="btn btn-primary my-3" @click="proceedToPayment">Proceed to
Payment</button>

```

```
</div>
```

```
{% endblock %}
```

```
{% block extra_js %}
```

```
<!-- Add JavaScript to show/hide payment details -->
```

```
<script src="https://checkout.razorpay.com/v1/checkout.js"></script>
```

```
<script>
```

```
new Vue({
```

```
el: '#app',
```

```
delimiters: ['[', ']',
```

```
methods: {
```

```
proceedToPayment(){
```

```
var options = {
```

```
"key": "{{ razorpay_key_id }}", // Enter the Key ID generated from the Dashboard
```

```
"amount": "{{ appointment.amount }}", // Amount is in currency subunits. Default currency is INR. Hence,
50000 refers to 50000 paise
```

```
"currency": "INR",
```

```
"name": "E-Diagnostic Lab Management System", //your business name
```

```
"description": "Lab Appoinment",
```

```
// "image": "https://example.com/your_logo",
```

```
"order_id": "{{ order.id }}", //This is a sample Order ID. Pass the `id` obtained in the response of Step 1
```

```
"callback_url": "{{ callback_url }}",
```

```
"prefill": { //We recommend using the prefill parameter to auto-fill customer's contact information
```

```

especially their phone number
"name": "{{ appointment.patients.1.full_name }}", //your customer's name
"email": "{{ appointment.user.email }}",
"contact": "{{ appointment.user.phone_number }}" //Provide the customer's phone number for better
conversion rates
},
};
var rzp1 = new Razorpay(options);
rzp1.open();
}
}
})
</script>
{% endblock %}

```

### Signup.html

```

{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<meta name="copyright" content="MACode ID, https://macodeid.com/">

<title>One Health - Medical Center HTML5 Template</title>

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

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

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

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

<link rel="stylesheet" href="{% static '/assets/css/theme.css' %}">
<style>
body {
font-family: Arial, sans-serif;
background-color: #e8f6f4;
}
header{
background-color: white;
}
/* Styles for the registration form container */
.registration-form-container {
max-width: 600px;
margin: 0 auto;
padding: 20px;
background-color: #dbfcf7;
box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
}

```

```
/* Form Styles (retain the existing form styles) */  
/* ... (form styles from your previous code) ... */
```

```
/* Styles for the registration form container */  
.registration-form-container {  
  max-width: 600px;  
  margin: 0 auto;  
  padding: 20px;  
  background-color: #fff;  
  box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);  
}
```

```
/* Form Styles (retain the existing form styles) */  
/* ... (form styles from your previous code) ... */
```

```
/* Styles for the registration form container */  
.registration-form-container {  
  max-width: 600px;  
  margin: 0 auto;  
  padding: 20px;  
  background-color: #fff;  
  box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);  
}
```

```
/* Form Styles (retain the existing form styles) */  
/* ... (form styles from your previous code) ... */
```

```
/* Styles for the registration form container */  
.registration-form-container {  
  max-width: 600px;  
  margin: 0 auto;  
  padding: 20px;  
  background-color: #fff;  
  box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);  
}
```

```
/* Form Styles (retain the existing form styles) */  
/* ... (form styles from your previous code) ... */
```

```
/* Styles for the registration form container */  
.registration-form-container {  
  max-width: 600px;  
  margin: 0 auto;  
  padding: 20px;  
  background-color: #fff;  
  box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);  
}
```

```
/* Form Styles (retain the existing form styles) */  
/* ... (form styles from your previous code) ... */
```

```
/* Styles for the registration form container */  
.registration-form-container {  
  max-width: 600px;
```

```
margin: 0 auto;
padding: 20px;
background-color: #fff;
box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
}

/* Form Styles (retain the existing form styles) */
/* ... (form styles from your previous code) ... */

/* Styles for the registration form container */
.registration-form-container {
max-width: 600px;
margin: 0 auto;
padding: 20px;
background-color: #fff;
box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
}

/* Form Styles (retain the existing form styles) */
/* ... (form styles from your previous code) ... */

h1 {
text-align: center;
}

/* Form Styles */
form {
margin-top: 20px;
}

label {
display: block;
margin-bottom: 5px;
}

input[type="text"],
input[type="password"],
input[type="email"],
input[type="tel"],
input[type="date"],
textarea {
width: 100%;
padding: 10px;
margin-bottom: 10px;
border: 1px solid #ccc;
border-radius: 4px;
}

input[type="radio"],
input[type="checkbox"] {
margin-right: 5px;
}

/* Form Layout */
.form-row {
```

```
display: flex;
justify-content: space-between;
align-items: center;
}

.form-row label {
flex: 1;
}

.form-row input,
.form-row textarea {
flex: 2;
}

.form-row input[type="radio"],
.form-row input[type="checkbox"] {
flex: unset;
}

/* Consent Checkbox */
#consent {
margin-top: 10px;
}

/* Submit Button */
input[type="submit"] {
background-color: #007BFF;
color: #fff;
padding: 10px 20px;
border: none;
border-radius: 4px;
cursor: pointer;
}
.custom-dropdown-button {
background-color: #4a69bd; /* Change the background color to your desired color */
color: #191717; /* Change the text color to white or your preferred color */
padding: 5px 10px; /* Adjust padding to control the button size */
font-size: 14px; /* Adjust font size as needed */
border: none; /* Remove border if not needed */
border-radius: 3px; /* Apply rounded corners if desired */
}
.error {
color: red; /* Set the text color for error messages to red */
font-size: 14px; /* Adjust font size as needed */
margin-top: 5px; /* Add some space between the input field and the error message */
}
.username-exists {
color: orange; /* Set a different text color for the "Username already exists" message */
font-size: 14px; /* Adjust font size as needed */
margin-top: 5px; /* Add some space between the input field and the error message */
}
.email-exists {
color: orange; /* Set a different text color for the "Username already exists" message */
font-size: 14px; /* Adjust font size as needed */
margin-top: 5px; /* Add some space between the input field and the error message */
}
```

```
}
</style>
</head>
<body>

<!-- Back to top button -->
<div class="back-to-top"></div>

<header>
<div class="topbar">
<div class="container">
<div class="row">
<div class="col-sm-8 text-sm">
<div class="site-info">
<a href="#"><span class="mai-call text-primary"></span> +00 123 4455 6666</a>
<span class="divider"></span>
<a href="#"><span class="mai-mail text-primary"></span> mail@example.com</a>
</div>
</div>

</div>
</div> <!-- .row -->
</div> <!-- .container -->
</div> <!-- .topbar -->

<nav class="navbar navbar-expand-lg navbar-light shadow-sm">
<div class="container">
<a class="navbar-brand" href="#"><span class="text-primary">One</span>-Health</a>

</div>

<button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupport" aria-
controls="navbarSupport" aria-expanded="false" aria-label="Toggle navigation">
<span class="navbar-toggler-icon"></span>
</button>

<div class="collapse navbar-collapse" id="navbarSupport">
<ul class="navbar-nav ml-auto">
<li class="nav-item ">
<a class="nav-link" href="index.html">Home</a>
</li>
<li class="nav-item">
<a class="nav-link" href="about.html">About Us</a>
</li>

<li class="nav-item">
<a class="nav-link" href="contact.html">Contact</a>
</li>
<li class="nav-item active">
<a class="btn btn-primary ml-lg-3" href="login.html">Login</a>
</li>
</ul>

</div> <!-- .navbar-collapse -->
</div> <!-- .container -->
```



```
</nav>
</header>

<br><br>
<div class="container">
<h3><center><b>Join Our Community</b></center></center></h3>

<form action="" method="post" id="registration-form">
{% csrf_token %}
<!-- Personal Information -->
<div class="form-row">
<label for="first-name">Full Name:</label>
<input type="text" id="first-name" name="first-name" required autocomplete="off">
<span id="full-name-error" class="error"></span>

</div>
<div class="form-row">
<label for="email">Email Address:</label>
<input type="email" id="email" name="email" required autocomplete="off">
<span id="email-error" class="error"></span>
<span id="email-availability" class="availability"></span>
</div>
<div class="form-row">
<label for="phone">Phone Number:</label>
<input type="tel" id="phone" name="phone" required autocomplete="off">
<span id="phone-error" class="error"></span>
</div>
<div class="form-row">
<label>Address:</label>
</div>
<div class="form-row">
<input type="text" id="home" name="home" placeholder="home Address" required autocomplete="off">
<span id="home-address-error" class="error"></span>
</div>
<div class="form-row">
<input type="text" id="city" name="city" placeholder="City" required autocomplete="off">
<span id="city-error" class="error"></span>
</div>

<div class="form-row">
<input type="text" id="zip" name="zip" placeholder="Postal Code" required autocomplete="off">
<span id="zip-error" class="error"></span>
</div>

<!-- User Account -->
<div class="form-row">
<label for="username">Username:</label>
<input type="text" id="username" name="username" required autocomplete="off">
<span id="username-error" class="error"></span>
<span id="username-availability" class="availability"></span>

</div>
<div class="form-row">
<label for="password">Password:</label>
```

```

<input type="password" id="password" name="password" required autocomplete="off">
<span id="password-error" class="error"></span>

</div>
<div class="form-row">
<label for="confirm-password">Confirm Password:</label>
<input type="password" id="confirm-password" name="confirm-password" required autocomplete="off">
<span id="confirm-password-error" class="error"></span>
</div>
<!-- Consent -->
<div class="form-row">
<input type="checkbox" id="consent" name="consent" required autocomplete="off">
<label for="consent">I have read and agree to the terms and conditions.</label>
</div>
<!-- Submit Button -->
<div class="form-row">
<input type="submit" value="Submit">
</div>
</form>
</div>
<br><br>
<footer class="page-footer">
<div class="container">
<div class="row px-md-3">
<div class="col-sm-6 col-lg-3 py-3">
<h5>Company</h5>
<ul class="footer-menu">
<li><a href="#">About Us</a></li>
<li><a href="#">Career</a></li>
<li><a href="#">Editorial Team</a></li>
<li><a href="#">Protection</a></li>
</ul>
</div>
<div class="col-sm-6 col-lg-3 py-3">
<h5>More</h5>
<ul class="footer-menu">
<li><a href="#">Terms & Condition</a></li>
<li><a href="#">Privacy</a></li>
<li><a href="#">Advertise</a></li>
<li><a href="#">Join as Doctors</a></li>
</ul>
</div>
<div class="col-sm-6 col-lg-3 py-3">
<h5>Our partner</h5>
<ul class="footer-menu">
<li><a href="#">One-Fitness</a></li>
<li><a href="#">One-Drugs</a></li>
<li><a href="#">One-Live</a></li>
</ul>
</div>
<div class="col-sm-6 col-lg-3 py-3">
<h5>Contact</h5>
<p class="footer-link mt-2">351 Willow Street Franklin, MA 02038</p>
<a href="#" class="footer-link">701-573-7582</a>
<a href="#" class="footer-link">healthcare@temporary.net</a>

```

```

<h5 class="mt-3">Social Media</h5>
<div class="footer-sosmed mt-3">
<a href="#" target="_blank"><span class="mai-logo-facebook-f"></span></a>
<a href="#" target="_blank"><span class="mai-logo-twitter"></span></a>
<a href="#" target="_blank"><span class="mai-logo-google-plus-g"></span></a>
<a href="#" target="_blank"><span class="mai-logo-instagram"></span></a>
<a href="#" target="_blank"><span class="mai-logo-linkedin"></span></a>
</div>
</div>
</div>

<hr>

<p id="copyright">Copyright &copy; 2020 <a href="https://macodeid.com/" target="_blank">MACode
ID</a>. All right reserved</p>
</div>
</footer>

<script src="{% static 'assets/js/jquery-3.5.1.min.js'%}"></script>

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

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

<script src="{% static 'assets/vendor/wow/wow.min.js'%}"></script>

<script src="{% static 'assets/js/theme.js'%}"></script>
<!-- ... (previous HTML code) ... -->

<script>

document.addEventListener("DOMContentLoaded", function () {
const fullNameInput = document.getElementById("first-name");

const emailInput = document.getElementById("email");
const phoneInput = document.getElementById("phone");
const homeAddressInput = document.getElementById("home");
const cityInput = document.getElementById("city");
const zipInput = document.getElementById("zip");
const usernameInput = document.getElementById("username");
const passwordInput = document.getElementById("password");
const confirmPasswordInput = document.getElementById("confirm-password");

const errorMessages = {
fullName: "Full Name is required and only letters",

email: "Invalid email format",
phone: "Invalid phone number format",
homeAddress: "Home Address is required",
city: "City is required",
zip: "Invalid ZIP code format",
username: "Username must be at least 3 characters long and only alphabets",
password: "Password must be at least 6 characters long",

```

```
confirmPassword: "Passwords do not match",
};
function validateFullName() {
const fullName = fullNameInput.value; // Do not trim the input value
const fullNamePattern = /^[A-Za-z][A-Za-z\s]{2,}$/; // Starts with a letter, followed by letters and spaces,
minimum 3 characters
const errorElement = document.getElementById("full-name-error");

if (!fullNamePattern.test(fullName) || /^s/.test(fullName)) {
errorElement.textContent = "Full Name must start with a letter and contain minimum 3 characters";
} else {
errorElement.textContent = "";
}
}

function validateEmail() {
const email = emailInput.value.trim();
const emailPattern = /^[a-z][a-z0-9._%+-]+@[a-z0-9.-]+\.[a-z]{2,}$/; // Updated pattern (no capital letters)
const errorElement = document.getElementById("email-error");

if (!emailPattern.test(email)) {
errorElement.textContent = "Invalid email format";
} else if (!/@(gmail|mca|ajce|yahoo|amaljyothi\.ac)\.[a-z]{2,}$/i.test(email)) { // Updated domain check
errorElement.textContent = "invalid email";
} else {
const parts = email.split('@');
const domainParts = parts[1].split('.');

// Check if the domain is "amaljyothi.ac.in"
if (domainParts.length === 3 && domainParts[1] === 'ac' && domainParts[2] === 'in') {
// No additional checks needed, so clear the error message
errorElement.textContent = "";
} else {
// Check if there is more than one dot in the domain part, except for specific domains
if (
domainParts.length > 2 &&
!/(mca\.ajce|ajce\.in|amaljyothi\.ac)$/i.test(domainParts.slice(-2).join('.'))
) {
errorElement.textContent = "Only one dot allowed in domain part";
} else if (email.length > 320) {
errorElement.textContent = "Email address is too long. Please use a shorter email address.";
} else {
errorElement.textContent = "";
}
}
}

function validatePhone() {
const phone = phoneInput.value.trim();
const phonePattern = /^[6789]d{9}$/; // Starts with 6, 7, 8, or 9 and has 10 digits
const errorElement = document.getElementById("phone-error");

if (!phonePattern.test(phone)) {
errorElement.textContent = "Invalid phone number format";
}
```

```
} else if (/^0{10}$/.test(phone)) {
errorElement.textContent = "Phone number cannot contain all zeros";
} else {
errorElement.textContent = "";
}
}

function validateHomeAddress() {
const homeAddress = homeAddressInput.value.trim();
const errorElement = document.getElementById("home-address-error");
const minLength = 3; // Minimum length

if (homeAddress.length < minLength || /^d/.test(homeAddress)) {
errorElement.textContent = "Home Address must start with a letter and be at least 3 characters long.";
} else {
errorElement.textContent = "";
}
}

function validateCity() {
const city = cityInput.value.trim();
const errorElement = document.getElementById("city-error");
const minLength = 3; // Minimum length

if (!/^([a-zA-Z][a-zA-Z, ]*$/.test(city) || city.length < minLength) {
errorElement.textContent = "City must start with a letter, can contain only letters, spaces, and commas, and be at least 3 characters long.";
} else {
errorElement.textContent = "";
}
}

function validateZIP() {
const zip = zipInput.value.trim();
const zipPattern = /^[1-9]\d{5}$/; // Six digits, starting with 1-9
const errorElement = document.getElementById("zip-error");

if (!zipPattern.test(zip)) {
errorElement.textContent = "invalid post code";
} else {
errorElement.textContent = "";
}
}

function validateUsername() {
const username = usernameInput.value.trim();
const usernamePattern = /^[A-Za-z][A-Za-z0-9]*$/; // Start with a letter, letters and numbers
const errorElement = document.getElementById("username-error");

if (!usernamePattern.test(username)) {
errorElement.textContent = "Username must start with a letter and can only contain letters and numbers";
} else if (username.length < 3) {
errorElement.textContent = "Username must be at least 3 characters long";
} else {
errorElement.textContent = "";
}
}
```

```
}

function validatePassword() {
  const password = passwordInput.value;
  const confirmPassword = confirmPasswordInput.value;
  const passwordError = document.getElementById("password-error");
  const confirmPasswordError = document.getElementById(
    "confirm-password-error"
  );

  const uppercasePattern = /[A-Z]/; // At least one uppercase letter
  const specialCharPattern = /[!@#$%^&*()_+{}\[ \];<>.,?~\\-]/; // At least one special character

  if (password.length < 6) {
    passwordError.textContent = errorMessages.password;
  } else if (!uppercasePattern.test(password)) {
    passwordError.textContent = "Password must contain at least one uppercase letter";
  } else if (!specialCharPattern.test(password)) {
    passwordError.textContent = "Password must contain at least one special character";
  } else {
    passwordError.textContent = "";
  }

  if (password !== confirmPassword) {
    confirmPasswordError.textContent = errorMessages.confirmPassword;
  } else {
    confirmPasswordError.textContent = "";
  }
}

fullNameInput.addEventListener("input", validateFullName);

emailInput.addEventListener("input", validateEmail);
phoneInput.addEventListener("input", validatePhone);
homeAddressInput.addEventListener("input", validateHomeAddress);
cityInput.addEventListener("input", validateCity);
zipInput.addEventListener("input", validateZIP);
usernameInput.addEventListener("input", validateUsername);
passwordInput.addEventListener("input", validatePassword);
confirmPasswordInput.addEventListener("input", validatePassword);
function validateForm() {
  // Check if there are any error messages
  const errorElements = document.querySelectorAll(".error");
  const hasErrors = Array.from(errorElements).some((errorElement) => errorElement.textContent !== "");

  // Enable or disable the submit button based on the presence of errors
  const submitButton = document.getElementById("submit-button");
  submitButton.disabled = hasErrors;
}

// Add an event listener to each input field for real-time validation
fullNameInput.addEventListener("input", validateFullName);
// ... Add event listeners for other input fields ...

// Add an event listener for the form submission
```

```
const registrationForm = document.getElementById("registration-form");
registrationForm.addEventListener("submit", function (event) {
// Prevent form submission if there are validation errors
if (document.querySelector(".error:not(:empty)") {
event.preventDefault();
}
});

// Validate the form whenever any input changes
registrationForm.addEventListener("input", validateForm);
});

// Function to check username availability
function checkUsernameAvailability(username) {
const usernameAvailability = document.getElementById('username-availability');

if (username.length >= 3) {
// Make an AJAX GET request to the server to check username availability
$.get(`/check_username_availability/?username=${username}`, function (data) {
if (data.available) {
usernameAvailability.textContent = "";
usernameAvailability.classList.add('available');
} else {
usernameAvailability.textContent = 'Username already registered';
usernameAvailability.classList.remove('available');
usernameAvailability.classList.add('username-exists');
}
});
} else {
usernameAvailability.textContent = "";
usernameAvailability.classList.remove('available');
}
}

// Add an event listener for the username input
const usernameInput = document.getElementById('username');
usernameInput.addEventListener('input', function () {
const username = usernameInput.value.trim();
checkUsernameAvailability(username);
});

// Function to reset the form fields
function resetForm() {
const registrationForm = document.getElementById("registration-form");
const formInputs = registrationForm.querySelectorAll("input, textarea");

// Clear input values
formInputs.forEach((input) => {
input.value = "";
});

// Clear any error messages
const errorElements = document.querySelectorAll(".error");
errorElements.forEach((errorElement) => (errorElement.textContent = ""));
}

function checkEmailAvailability(email) {
```

```
const emailAvailability = document.getElementById('email-availability');

if (email.length >= 5) { // You can adjust the minimum length as needed
// Make an AJAX GET request to the server to check email availability
$.get(`/check_email_availability/?email=${email}`, function (data) {
  if (data.available) {
    emailAvailability.textContent = "";
    emailAvailability.classList.add('available');

  } else {
    emailAvailability.textContent = 'Email already registered';
    emailAvailability.classList.remove('available');
    emailAvailability.classList.add('email-exists');
  }
});
} else {
  emailAvailability.textContent = "";
  emailAvailability.classList.remove('available');
}
}

// Add an event listener for the email input
const emailInput = document.getElementById('email');
emailInput.addEventListener('input', function () {
  const email = emailInput.value.trim();
  checkEmailAvailability(email);
});

// Function to reset the form fields and availability messages
function resetForm() {
  const registrationForm = document.getElementById("registration-form");
  const formInputs = registrationForm.querySelectorAll("input, textarea");

  // Clear input values
  formInputs.forEach((input) => {
    input.value = "";
  });

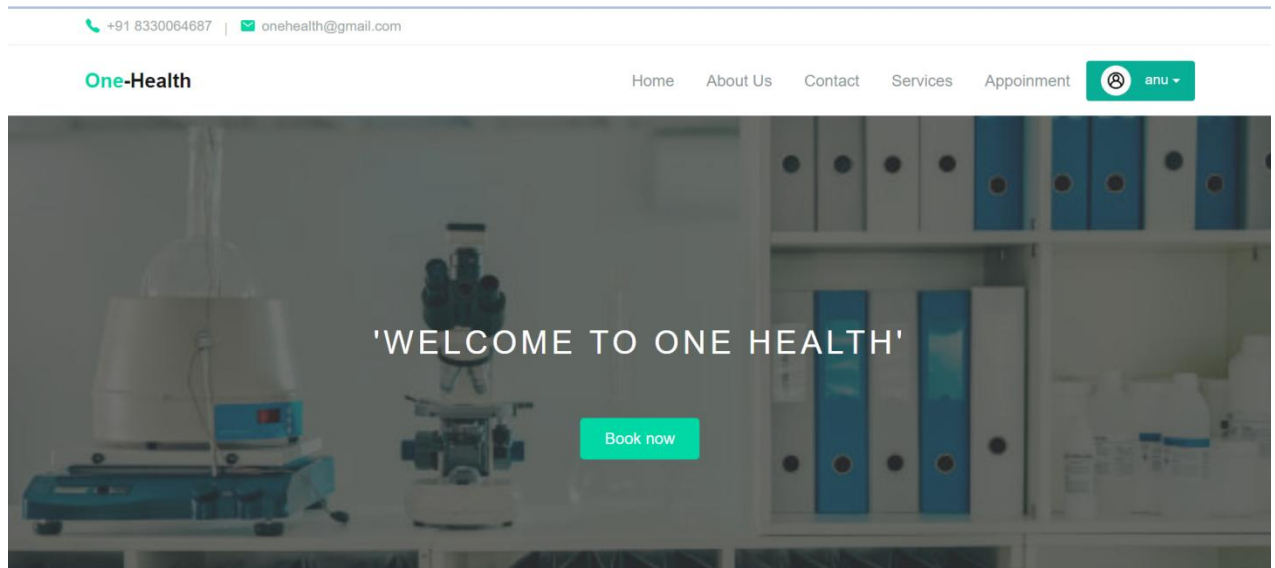
  // Clear any availability messages
  const availabilityElements = document.querySelectorAll(".availability");
  availabilityElements.forEach((availabilityElement) => {
    availabilityElement.textContent = "";
    availabilityElement.classList.remove('available', 'username-exists', 'email-exists');
  });

  // Clear any error messages
  const errorElements = document.querySelectorAll(".error");
  errorElements.forEach((errorElement) => (errorElement.textContent = ""));
}
</script>
<!-- ... (rest of your HTML code) ... -->
</body>
</html>
```

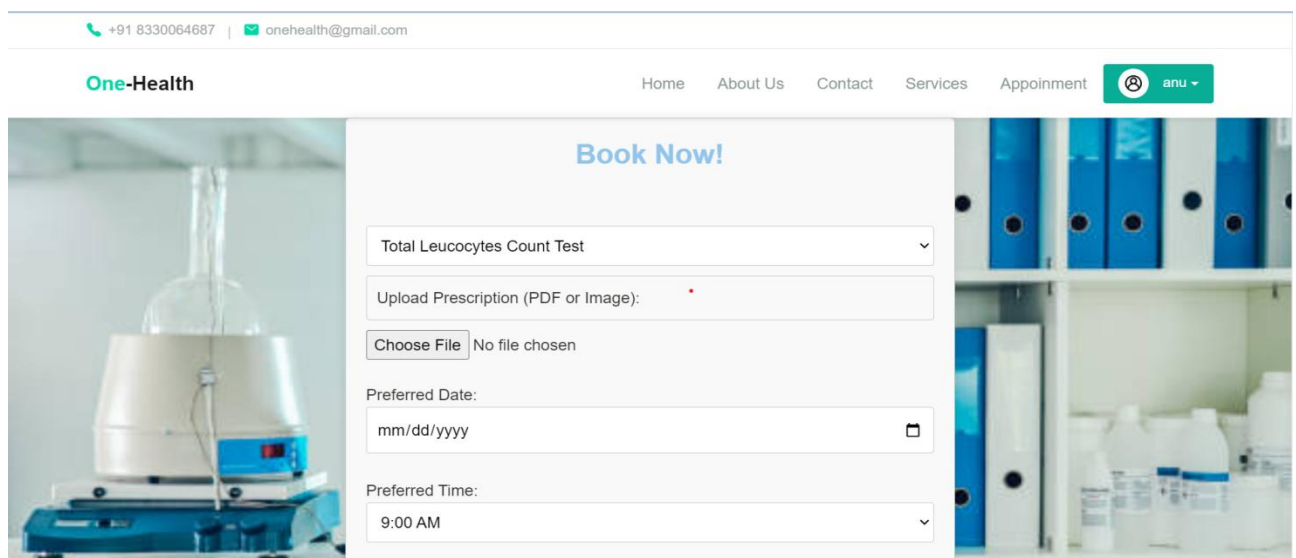




## 9.1 Screen Shots

### Homepage



### Book page





Full name

Select Gender

Age

anu@gmail.com

8330064687



☒ Use Current Address ☐ Enter New Address

anusssssssssss

kochi

987654

☐ Add Additional Tests



Choose mode of sample collection :

☒ Home ☐ Lab

Select Location

Test price: Rs. 200

Home Collection Charge: 0

Total Amount: Rs. 200

Book Now

## Payment page

+91 8330064687 | onehealth@gmail.com

One-Health Home About Us Contact Services Appointment anu

### Payment Confirmation

Thank you for booking your appointment! Please choose your method of payment and enter the required details:

**Test:** Mumps antibody-IGM test

**Preferred Date:** Dec. 4, 2023

**Preferred Time:** 5 p.m.

**Full Name:** Anu

**Email:** anu@gmail.com

**Phone:** 8330064687

**Sample Collection Type:** Home Collection


Proceed to Payment

Services page

+91 8330064687 | onehealth@gmail.com

One-Health

[Home](#) [About Us](#) [Contact](#) [Services](#) [Appointment](#)

 anu

Home / Services

Our Services

Search for Tests (CBC, Lipid,diabetes etc)

Search for Blood Tests (CBC, Lipid, etc)

Search

Total Leucocytes Count Test

Rs. 200.00

Book Now

Hemoglobin

Rs. 100.00

Book Now

Platelet count Test

Rs. 150.00

Book Now

Serum Globulin Test

Rs. 150.00

Book Now

