DENTAL HOSPITAL BOOKING SYSTEM

Project Report Submitted by

AJEL VARGHESE JOHN

Reg. No.: AJC22MCA-2005

In Partial fulfillment for the Award of the Degree Of

MASTER OF COMPUTER APPLICATIONS

(MCA TWO YEAR)

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, "**DENTAL HOSPITAL BOOKING SYSTEM"** is the bona fide work of **AJEL VARGHESE JOHN (AJC22MCA-2005)** in partial fulfillment of the requirements for the award of the Degree of Master of Computer Applications under APJ Abdul Kalam Technological University during the year 2023-24.

Ms. Lisha Varghese	**Ms. Meera Rose Mathew**
Internal Guide	**Coordinator**

Rev. Fr. Dr. Rubin Thottupurathu Jose

Head of the Department

DECLARATION

I hereby declare that the project report **"DENTAL HOSPITAL BOOKING SYSTEM"** is a bona fide work done at Amal Jyothi College of Engineering, towards the partial fulfilment of the requirements for the award of the Master of Computer Applications (MCA) from APJ Abdul Kalam Technological University, during the academic year 2022-2023.

Date:08/12/2023 **AJEL VARGHESE JOHN**

KANJIRAPPALLY **AJC22MCA-2005**

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. Roy Abraham Pazhayaparambil** and Principal **Dr. LillykuttyJacob** 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 **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 **Lisha Varghese** 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.

AJEL VARGHESE JOHN

ABSTRACT

This system revolutionizes the process of scheduling and managing dental appointments, offering a seamless experience for patients, administrators, and dental professionals. Users can effortlessly book appointments, access information on available dental services, and even customize treatment plans to suit their individual needs. The platform goes beyond traditional booking systems by providing personalized recommendations for additional dental services, based on the unique requirements of each patient. This level of customization is achieved through direct consultations with dental professionals, ensuring that the recommended treatments align with the patient's preferences and oral health goals.

Moreover, the system introduces a novel approach to dental appointment scheduling by incorporating events-based searching. Patients can now efficiently schedule appointments tailored to specific dental events or requirements. This feature enhances the overall efficiency of the booking process, allowing individuals to find and secure appointments that align with their dental care needs. In contrast to conventional methods that often require patients to navigate various sources for dental services, this system centralizes information, empowering patients with a comprehensive view of available treatments and practitioners. The result is a patient-centric approach that prioritizes individual preferences, leading to a more tailored and convenient booking experience within the dental hospital ecosystem.
HTML, CSS, DJANGO, JAVA, and MySQL – are the technologies used.

CONTENT

List of Abbreviation

IDE - Integrated Development Environment

HTML - Hyper Text Markup Language. CSS - Cascading Style Sheet

SQL - Structured Query Language

UML - Unified Modeling Language

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

The Dental Appointment Booking System is tailored to meet the diverse needs of patients seeking dental care. This user-friendly platform streamlines the process of scheduling dental appointments, providing anaccessible and efficient solution for patients, administrators, and dental professionals alike. Through its innovative features, the system aims to enhance the overall dental healthcare experience, offering a seamless and personalized approach to appointment management within the dental sector.

1.2 PROJECT SPECIFICATION

There are mainly 3 users in the system and they are:

i. Patients

ii. Clinics

iii. Admin

The Dental Hospital Booking System is designed to create an efficient and user-centric platform for streamlining dental appointment scheduling and service management. The system accommodates three primary user roles: Patients, Dental Professionals, and Admin. Patients can seamlessly schedule dental appointments, customize treatment plans through direct consultations with dental professionals, and access information on available dental services and practitioners. Dental professionals can collaborate with patients to customize treatment plans, provide feedback on treatment plans and appointments, and access patient information for personalized healthcare. Admin oversees and enhances the overall efficiency of dental services within the hospital, managing and optimizing the appointment booking system and providing a centralized hub of information on available dental services and practitioners. The system aims to simplify the dental healthcare experience, fostering direct communication between patients and dental professionals while ensuring comprehensive control and optimization through the Admin interface.

CHAPTER 2

SYSTEM STUDY

2.1 INTRODUCTION

System analysis is the process of acquiring and analyzing data, diagnosing issues, and using the data to suggest system changes. The system users and system developers must communicate extensively during this problem-solving process. A critical stage of every system is the analysis or study phase creation procedure. The system is meticulously examined and assessed. The System analysts act as questioners and delve deeply into the operation of the current framework. The input to the system is seen as a whole and includes identified. The various procedures can be linked to the outputs from the organization. The goal of analysis is to recognize the issue, pinpoint the source of the issue, and analyze and synthesis of the numerous components, consideration of pertinent and decisive variables, deciding on an ideal, or at the very least, a suitable, solution or course of action. The process must be thoroughly studied using a variety of techniques, including questionnaires and interviews. These sources' data must be carefully examined to make a decision. Understanding how the system works. The current system is being forced to shut down. Problem areas are found through research. The designer is currently a problem Solver and makes an effort to resolve the issues the business is facing. The remedies are given as suggestions. The proposal is then analytically compared to the current system. The user was made aware of the proposition and asked to support it.

2.2 NATURAL SYSTEM STUDIED

The current system for dental hospital booking relies on traditional methods, requiring patients to visit or call for appointments, leading to potential inefficiencies and limited accessibility to real-time information. The process lacks a centralized platform for patients, dental professionals, and administrators, hindering customization options and personalized recommendations. The system also lacks modern features, such as event-based searching and collaborative approaches between patients and dental professionals. Transitioning to a more technology-driven and user-centric approach could significantly improve efficiency and enhance the overall patient experience.

DRAWBACKS OF EXISTING SYSTEM

- Human effort is needed
- Wastage of time for booking manually
- No proper data collection and processing in case of doctors

- Lack of validations

2.3 DESIGNED SYSTEM STUDIED

The Dental Hospital Booking System is an intricately designed platform that aims to optimize the processes associated with dental appointment scheduling and service management. Tailored to accommodate the needs of three key user roles—Patients, Dental Professionals, and Admin—the system offers a user-friendly interface for patients to effortlessly book appointments, customize treatment plans through direct consultations with dental professionals, and receive personalized recommendations for additional services. Dental professionals benefit from collaborative tools to efficiently manage appointments and provide personalized healthcare. Admin, on the other hand, oversees and enhances the overall efficiency of dental services within the hospital, implementing trust signals for appointment availability and maintaining stock management. With features like real-time notifications, a centralized information hub, and robust admin controls, the system stands as a comprehensive solution poised to transform the dental healthcare experience.

2.4 ADVANTAGES OF PROPOSED SYSTEM

- Time management
- Not much human effort is needed
- Quality of services
- Money Management
- Always available system

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDY

Planning, organizing, and managing resources to ensure the achievement of particular project goals and objectives is the process of project management. A feasibility study is a preliminary examination of a prospective project or end to determine its merits and viability. A feasibility study aims to provide an objective assessment of the technical, economic, financial, legal, and environmental elements of a proposed project. The information can then be used by decisionmakers to decide whether to proceed with the project or not. The findings of the feasibility study can also be used to develop a practical project plan and budget. It cannot be simple to determine whether or not a proposed project is worthwhile pursuing without a feasibility study. The document provides the feasibility of the project that is being designed and lists. Various areas that were considered very carefully during the feasibility study of this project such as Technical, Economic and Operational feasibility. The following are its features: -

3.1.1 Economical Feasibility

Cost and benefit analyses are required to support the developing system. criteria to make sure that focus is on the project that will yield the best results and return the earliest. The price that would be involved in developing a new system is one of the variables. Some significant financial queries raised during the initial investigation include the following:
The costs conduct a full system investigation.?

✓ The proposed system is developed as part of project work, there is no manual cost to spend for the proposed system.
The cost of the hardware and software.?

✓ Also all the resources are already available

3.1.2 Technical Feasibility

The system needs to be assessed first from a technical standpoint. The outline design of the system requirement in terms of input, output, programs, and procedures must serve as the foundation for the assessment of this feasibility. After determining an outline investigation must continue to identify the necessary equipment kind. Once the system has been designed, there are several ways to run it. Technical issues raised during the investigation are:

1. Is the project feasible within the limits of current technology?

✓ Satisfied

2. Can the technology be easily applied to current problems?

✓ Satisfied

3. Does the technology have the capacity to handle the solution?

✓ Satisfied

3.1.3 Behavioral Feasibility

The proposed system includes the following questions:

Is there sufficient support for the users?

✓ Satisfied

Will the proposed system cause harm?

✓ No

The project would be beneficial because it satisfies the objectives when developed and installed. All behavioral aspects are considered carefully and conclude that the project is behaviorally feasible.

3.2 SYSTEM SPECIFICATION

3.2.1 Hardware Specification

Processor - Intel core i3

RAM- 4 GB

Hard disk - 1 TB

3.2.2 Software Specification

Front End - HTML, CSS

Backend - Django

Client on PC - Windows 7 and above.

Technologies used - JS, HTML, AJAX, J Query, Django,

CSS

3.3 SOFTWARE DESCRIPTION

3.3.1 DJANGO

\

Django is a high-level, open-source web framework for the rapid development of web applications using Python. Following the Model-View-Template (MVT) architecture, Django provides an Object-Object-relational mapping (ORM) system for seamless database interactions, abstracting the database layer and promoting code readability. With a built-in admin interface, URL routing, and a template engine, Django facilitates clean and expressive development. Its middleware components enable global processing of requests and responses, while robust security features guard against common web vulnerabilities. The framework includes authentication and authorization systems, encouraging the development of reusable apps and ensuring scalability from small projects to large, high-traffic websites. Known for its "batteries-included" philosophy, Django is widely used for building web applications with diverse functionalities, making it a popular choice in the development community.

3.3.2 MySQL

MySQL, the most popular open-source SQL database management system, is developed, distributed, and supported by Oracle Corporation. The MySQL Web site provides the latest information about MySQL software.

• **MySQL is a database management system.**

A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

• **MySQL databases are relational.**

A relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables, views, rows, and columns, offers a flexible programming environment. You set up rules governing the relationships between different data fields, such as one-to-one, one-to-many, unique, required, or optional, and "pointers" between different tables. The database enforces these rules, so that with a well-designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data. The SQL part of "MySQL" stands for "Structured Query Language". SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax. SQL is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, "SQL92" refers to the standard released in 1992, "SQL: 1999" refers to the standard released in 1999, and "SQL: 2003" refers to the current version of the standard. We use the phrase "the SQL standard" to mean the current version of the SQL Standard at any time.

• **MySQL software is Open Source.**

Open source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License), to define what you may and may not do with the software in different situations.

If you feel uncomfortable with the GPL or need to embed MySQL code into a commercial application, you can buy a commercially licensed version from us. See the MySQL Licensing Overview for more information.

• **The MySQL Database Server is very fast, reliable, scalable, and easy to use.**

You ought to give it a shot if that is what you're after. In addition to your other apps, web servers, and other software, MySQL Server can function smoothly on a desktop or laptop while requiring little to no maintenance. You can modify the settings to utilize all the RAM, CPU power, and I/O capacity if you dedicate an entire machine to MySQL. MySQL Server works in client/server or embedded systems.

The MySQL Database Software is a client/server system made up of a multi-threaded SQL server that supports several client programs and libraries, administration tools, and a broad range of application programming interfaces(APIs). Additionally, we provide MySQL Server as an integrated multi-threaded library that you can link into your program to create a standalone offering that is smaller, faster, and simpler to operate.

CHAPTER 4

SYSTEM DESIGN

4.1 INTRODUCTION

Any engineered system or product's development process begins with design. A creative process is design. The secret to an efficient system is a decent design. The process of using different methodologies and concepts to specify a process or a system in enough detail to allow for its physical implementation is referred to as "design." One way to describe it is as the process of using different methodologies and concepts to specify a device, a process, or a system in enough detail to allow for its physical reality. Regardless of the development paradigm that is employed, software design forms the technical core of the software engineering process. The architectural detail needed to construct a system or product is developed through the system design. This program has also through the best possible design phase, fine-tuned all efficiency, performance, and accuracy levels, as in the case of any systematic technique. A user-oriented document is converted into a document for programmers or database staff throughout the design phase. The two stages of system design development are logical design and physical design.

4.2 UML DIAGRAM

UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML was created by the Object Management Group (OMG)and the UML 1.0 specification draft was proposed to the OMG in January 1997. UML stands for Unified Modeling Language. UML is different from the other common programming languages such as C++, Java, COBOL, etc. UML is a pictorial language used to make software blueprints. UML can be described as a general-purpose visual modeling language to visualize, specify, construct, and document software systems. Although UML is generally used to model software systems, it is not limited within this boundary. It is also used to model non-software systems as well. For example, the process flow in a manufacturing unit, etc. UML is not a programming language but tools can be used to generate code in various languages using UML diagrams. UML has a direct relation with object-oriented analysis and design. After some standardization, UML has become an OMG standard. All the elements and relationships are used to make a complete UML diagram and the diagram represents a system. The visual effect of the UML diagram is the most important part of the entire process.

All the other elements are used to make it complete.

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

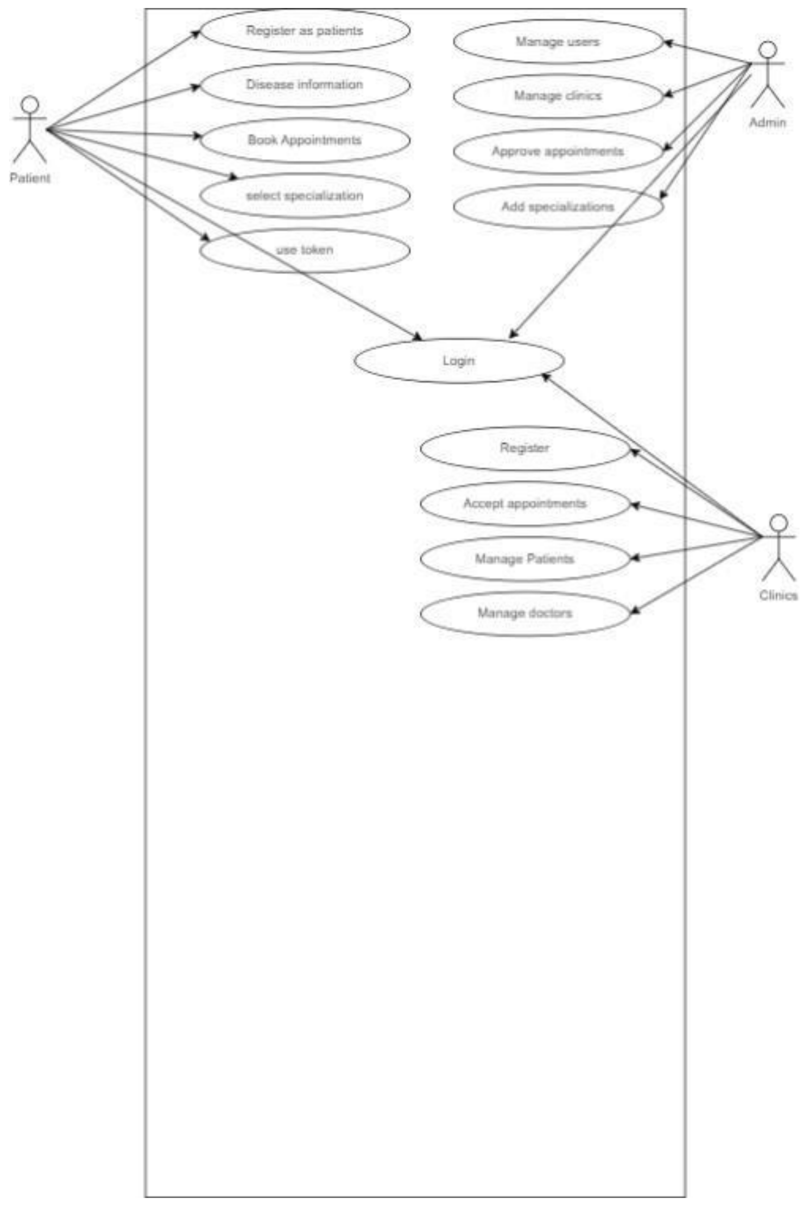
4.2.1 USE CASE DIAGRAM

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 modeling 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, catalog updating, and payment processing. There are four elements in a use case diagram.

- The boundary, which defines the system of interest about 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 have to 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.

USE CASE DIAGRAM



4.2.2 SEQUENCE DIAGRAM

A sequence diagram essentially shows how things interact with one another sequentially, or the order in which these interactions occur. A sequence diagram can also be referred to as event diagrams or event scenarios. Sequence diagrams show the actions taken by the components of a system in chronological order. Business people and software engineers frequently use these diagrams to record and comprehend the requirements for new and current systems.

Sequence Diagram Notations –

i. Actors – In a UML diagram, an actor represents a particular kind of role that interacts with the system and its objects. An actor is always beyond the purview of the system that we want to use the UML diagram to represent. We employ actors to portray a variety of roles, including those of human users and other outside subjects. In a UML diagram, an actor is represented using a stick person notation. In a sequence diagram, there might be several actors.

ii. Lifelines – A lifeline is a named element in a sequence diagram that represents an individual participant. So, in a sequence diagram, each incident is represented by a lifeline. A sequence diagram's lifeline elements are at the top.

iii. Messages – Messages are used to show how objects communicate with one another. The messages are displayed on the lifeline in chronological sequence. Arrows are how messages are represented. A sequence diagram's main components are lifelines and messages. Messages can be broadly classified into the following categories:

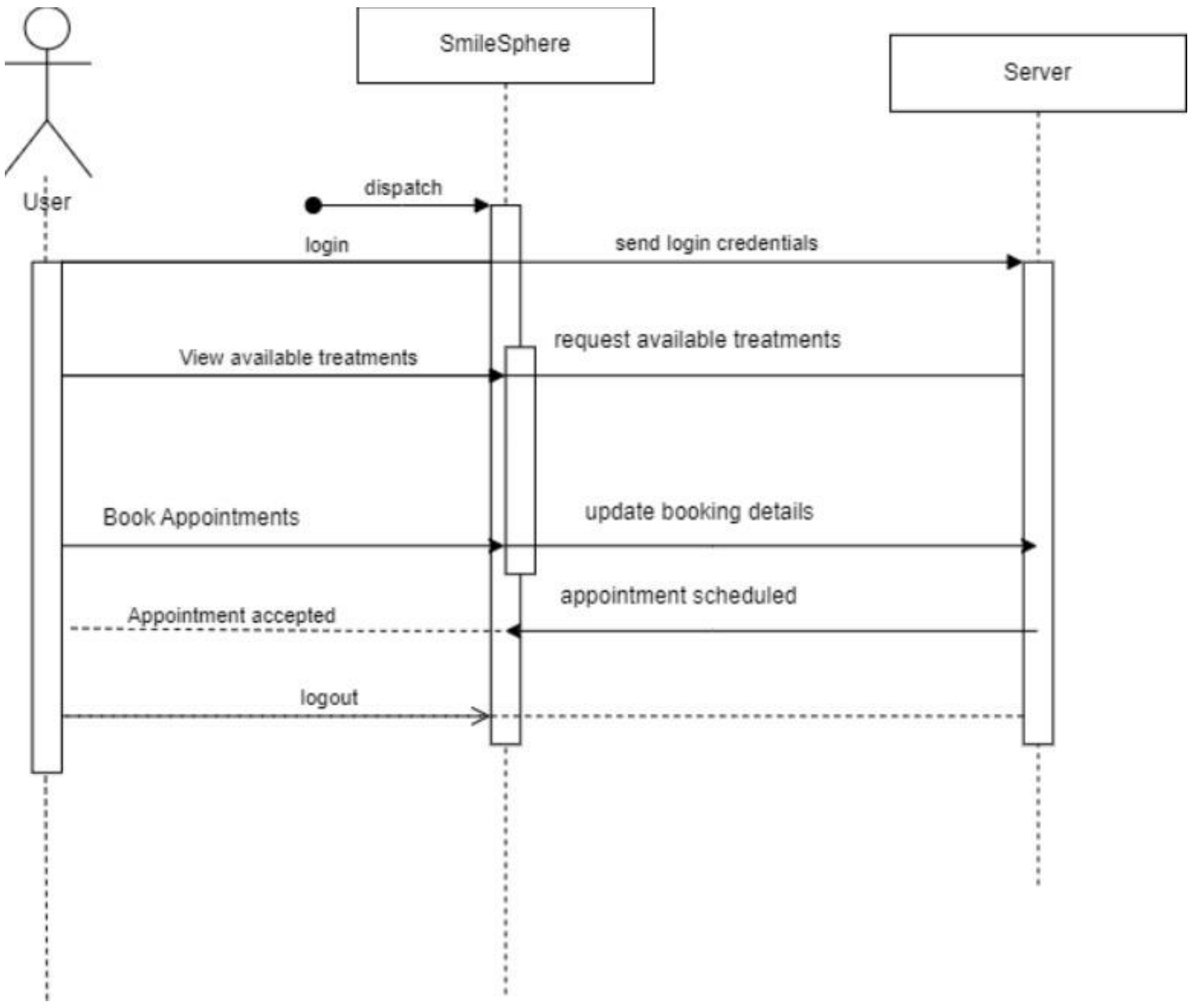
- Synchronous messages
- Asynchronous Messages
- Create message
- Delete Message
- Self-Message
- Reply Message
- Found Message
- Lost Message

iv. Guards – To model conditions we use guards in UML. They are used when we need to restrict the flow of messages on the pretext of a condition being met. Guards play an important role in letting software developers know the constraints attached to a system or a particular process.

Uses of sequence diagrams –

- Used to model and visualize the logic behind a sophisticated function, operation or procedure.
- They are also used to show details of UML use case diagrams.
- Used to understand the detailed functionality of current or future systems.
- Visualize how messages and tasks move between objects or components in a system.

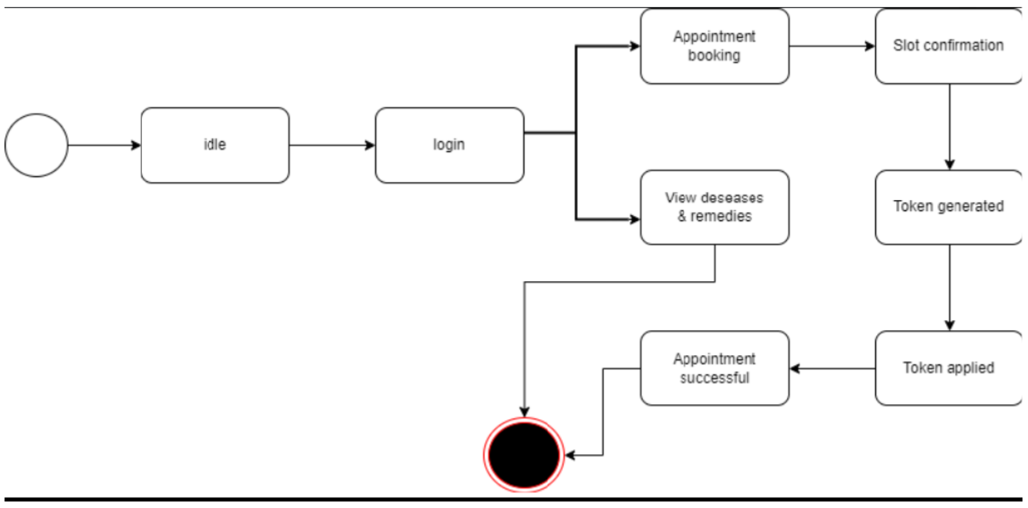
SEQUENCE DIAGRAM



4.2.3 STATE CHART DIAGRAM

A state diagram, also known as a state machine diagram or statechart 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).

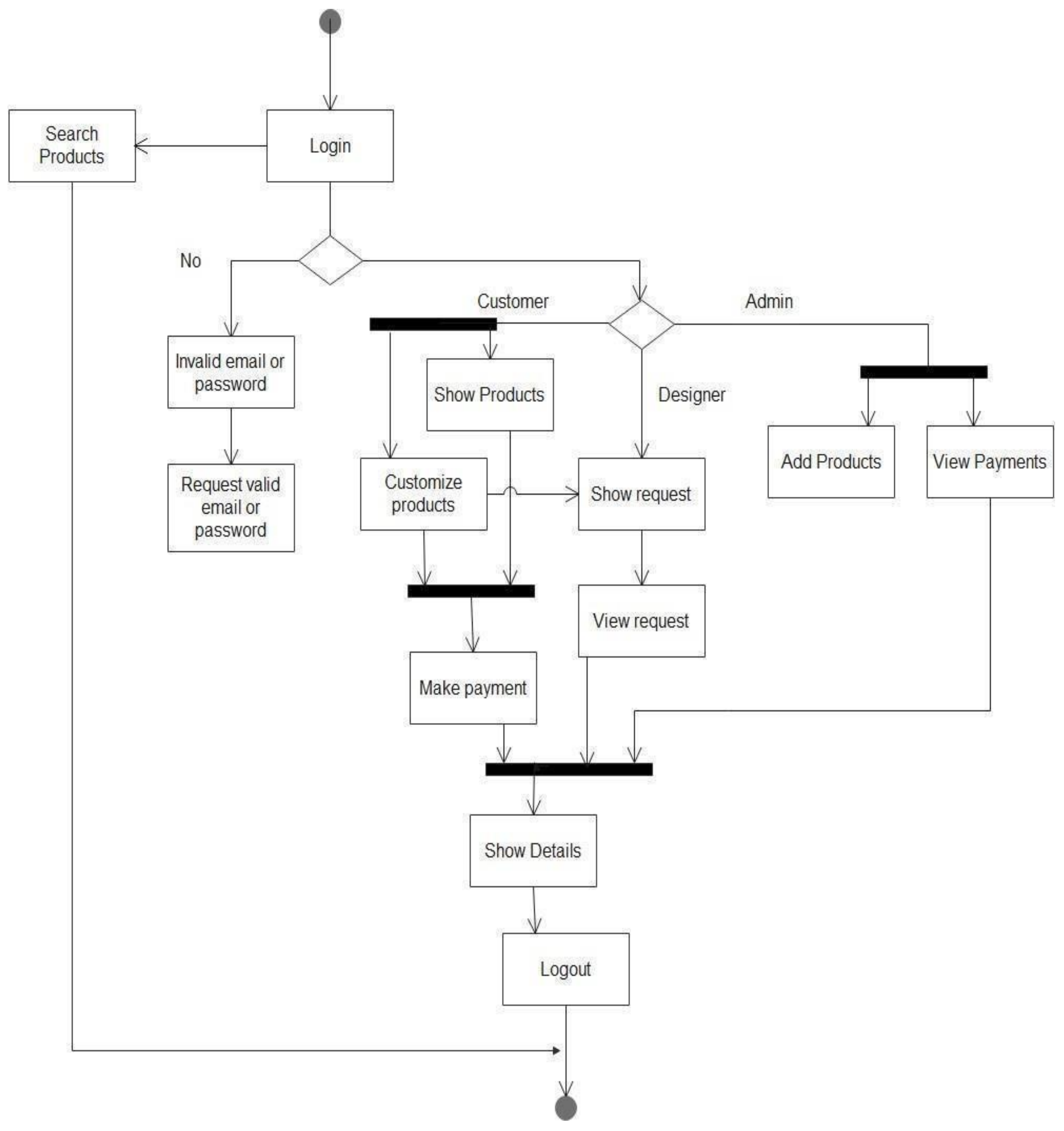
<u>STATE CHART DIAGRAM</u>



4.2.4 ACTIVITY DIAGRAM

Another crucial UML diagram for describing the system's dynamic elements is the 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 execute

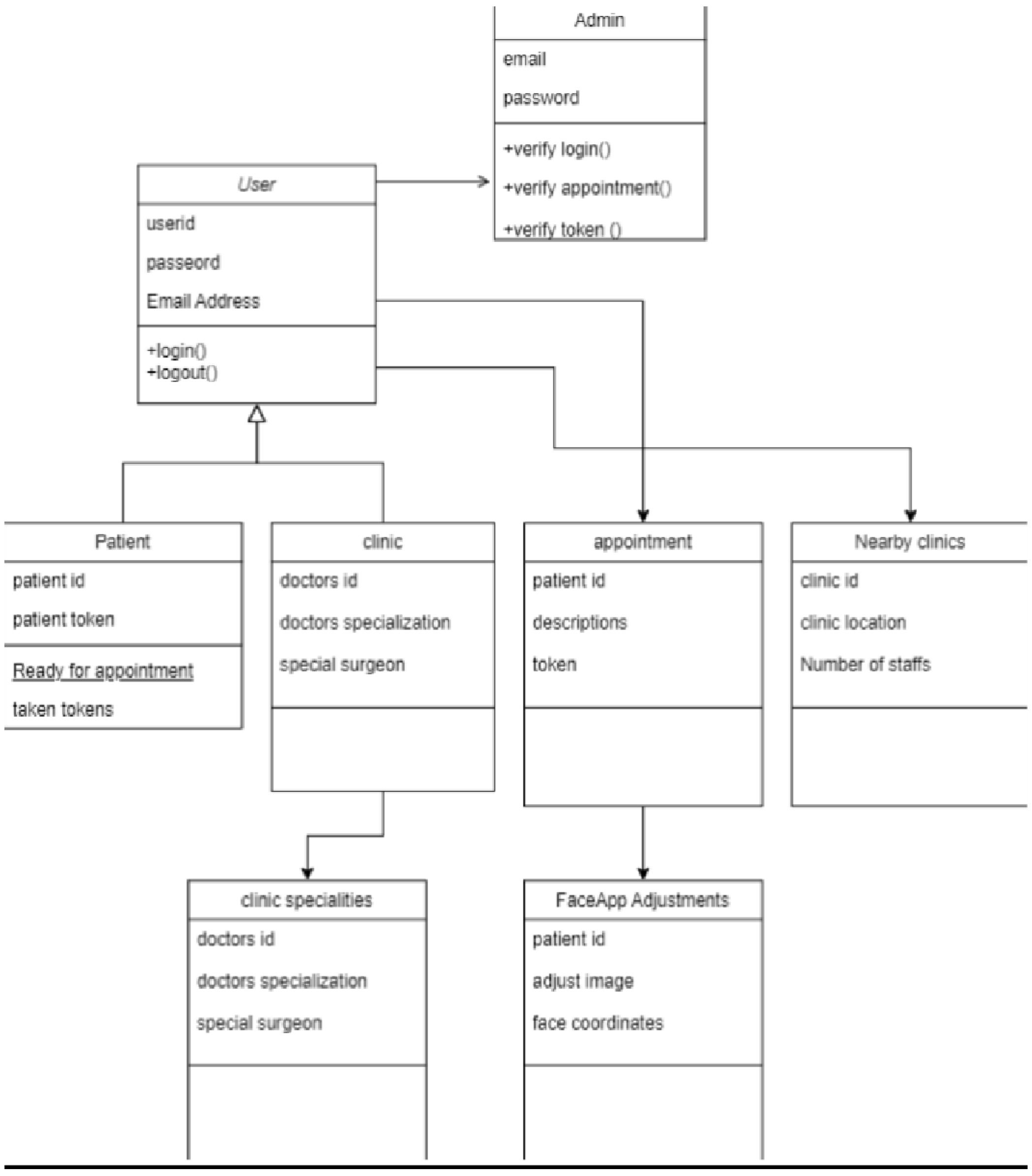
ACTIVITY DIAGRAM



4.2.5 CLASS DIAGRAM

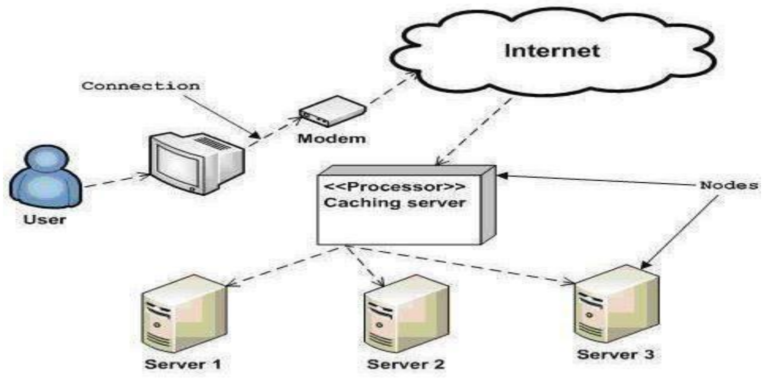
The class diagram is the main building block of object-oriented modeling. It is used for general conceptual modeling of the structure of the application, and for detailed modeling, translating the models into programming code. Class diagrams can also be used for data modeling.

CLASS DIAGRAM



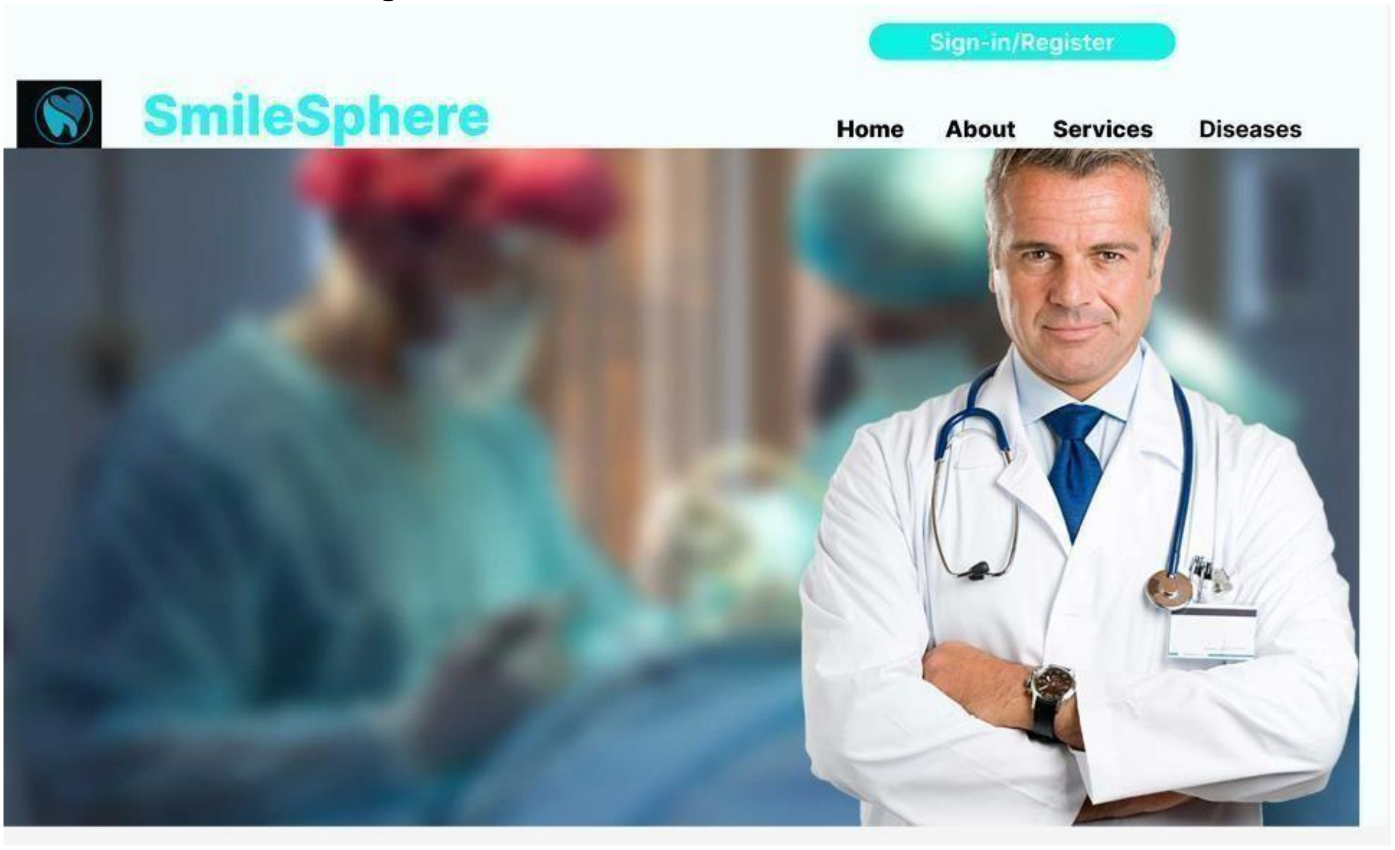
4.2.6 DEPLOYMENT DIAGRAM

An execution architecture of a system, containing nodes like hardware or software execution environments, and the middleware linking them, is shown in a deployment diagram, a form of UML diagram. Typically, deployment diagrams are used to represent the actual hardware and software of a system. By using it, you can comprehend how the hardware will physically deliver the system. In contrast to other UML diagram types, which primarily depict the logical components of a system, deployment diagrams assist describe the hardware structure of a system.

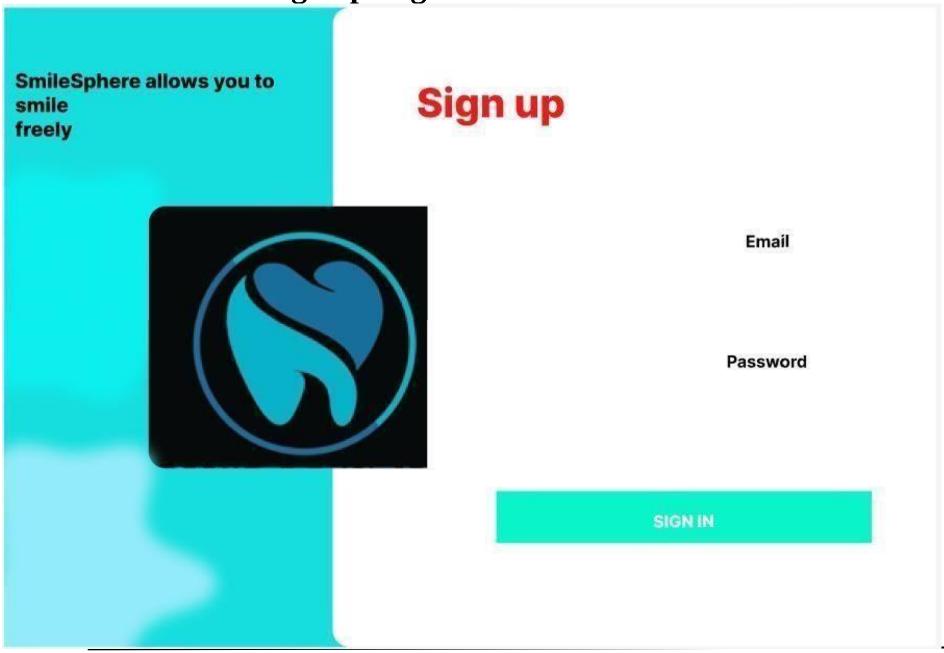


4.3 USER INTERFACE DESIGN USING FIGMA

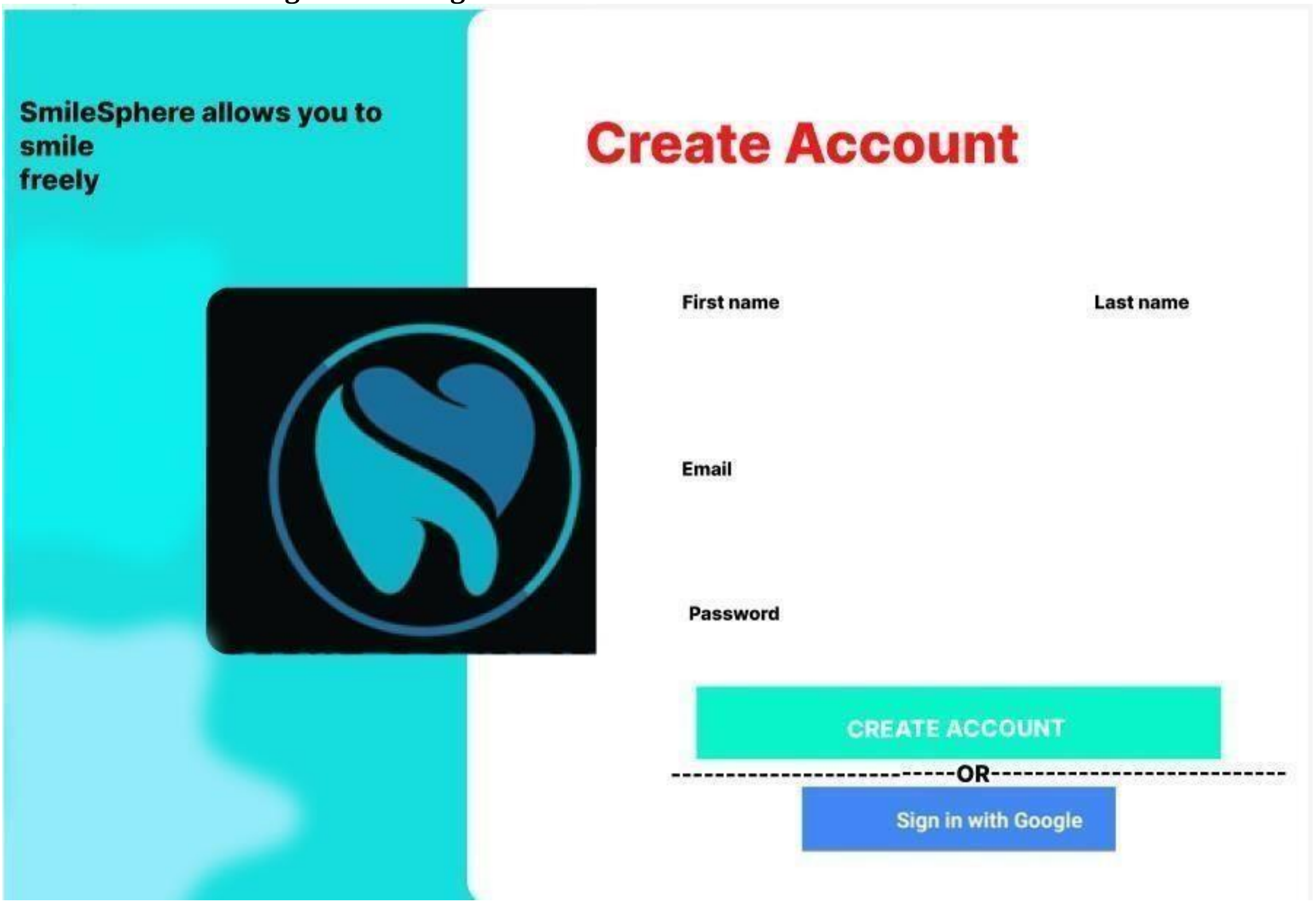
Form Name: Index Page



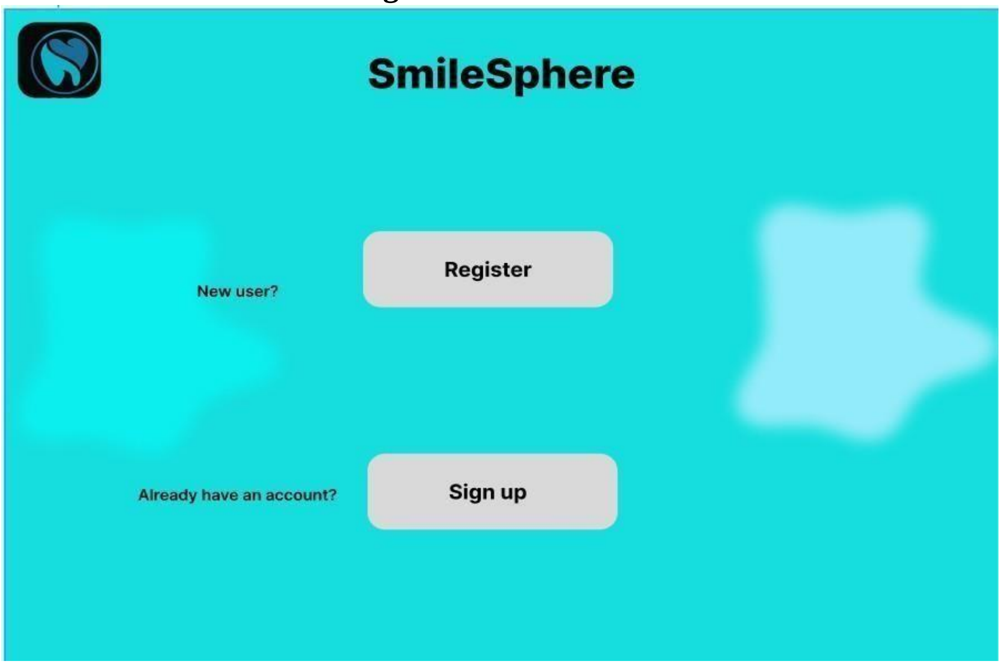
Form Name: Sign up Page



Form Name: Registration Page


SmileSphere allows you to smile freely
Create Account
First name
Last name
Email
Password
CREATE ACCOUNT
------------------------OR----------------------------
Sign in with Google

Form Name: Front Page



4.3 DATABASE DESIGN

A database is a structured system with the capacity to store information and allow users to retrieve stored information quickly and effectively. Any database's primary goal is its data, which demands protection. There are two stages to the database design process. The user needs are obtained in the first step, and a database is created to as clearly as possible meet these criteria. This process, known as information level design, is carried out independently of all DBMS. The design for the specific DBMS that will be used to construct the system in issue is converted from an information-level design to a design in the second stage. Physical Level Design is the stage where the characteristics of the particular DBMS that will be used are discussed. Parallel to the system design is a database design. The database's data arrangement aims to accomplish the two main goals listed below.

• Data Integrity

• Data independence

4.4.1 Relational Database Management System (RDBMS)

In a relational model, the database is shown as a set of relations. Each relation resembles a file or table of records with values. A row is referred to as a tuple, a column heading is referred to as an attribute, and the table is referred to as a relation in formal relational model language. A relational database is made up of many tables, each with its own name. In a story, each row represents a group of associated values. Relations, Domains & Attributes A relation is a table. Tuples are the units of a table's rows. An ordered group of n elements is a tuple. Attributes are referred to as columns. Every table in the database has relationships already established between them. This guarantees the integrity of both referential and entity relationships. A group of atomic values make up a domain D. Choosing a data type from which the domain's data values are derived is a typical way to define a domain. To make it easier to understand the values of the domain, it is also helpful to give it a name. Each value in a relation is atomic and cannot be broken down.

Relationships

• Table relationships are established using Key. The two main keys of prime importance are Primary Key & Foreign Key. Entity Integrity and Referential Integrity Relationships can be established with these keys.

• Entity Integrity enforces that no Primary Key can have null values.

• Referential Integrity enforces that no Primary Key can have null values.

• Referential Integrity for each distinct Foreign Key value, there must exist a matching Primary Key value in the same domain. Other keys are Super Key and Candidate Keys.

4.4.2 Normalization

The simplest possible grouping of data is used to put them together so that future changes can be made with little influence on the data structures. The formal process of normalizing data structures in a way that reduces duplication and fosters integrity. Using the normalization technique, superfluous fields are removed and a huge table is divided into several smaller ones. Anomalies in insertion, deletion, and updating are also prevented by using it. Keys and relationships are two notions used in the standard form of data modeling. A row in a table is uniquely identified by a key. Primary keys and foreign keys are the two different kinds of keys. A primary key is an element, or set of components, in a table that serves as a means of distinguishing between records from the same table. A column in a table known as a foreign key is used to uniquely identify records from other tables. Up to the third normal form, all tables have been normalized. It means placing things in their natural form, as the name suggests. By using normalization, the application developer aims to establish a coherent arrangement of the data into appropriate tables and columns, where names may be quickly related to the data by the user. By removing recurring groups from the data, normalization prevents data redundancy, which puts a heavy strain on the computer's resources. These consist of:

- Normalize the data.
- Choose proper names for the tables and columns.
- Choose the proper name for data.

4.4.3 Sanitization

Sanitizing data means removing any illegal character from the data. Sanitizing user input is one of the most common tasks in a web application. To make this task easier PHP provides a native filter extension that you can use to sanitize the data such as e-mail addresses, URLs, IP addresses, etc. PHP filters are used to sanitize and validate external input. The PHP filter extension has many of the functions needed for checking user input and is designed to do data sanitization easier and quicker. This function, when using the flag in the example, is making sure that the code removes all characters except letters, digits and the following characters !#$%&amp;’*+-=?_`{|}~@.[] . Many web applications receive external input. External input/data can be:

- User input from a form
- Cookies
- Web services data
- Server Variables

- Database query results

4.4.4 Indexing

The index stores the value of a specific field or set of fields, ordered by the value of the field. The ordering of the index entries supports efficient equality matches and range-based query operations. Indexes are used to quickly locate data without having to search every row in a database table every time a database table is accessed. Indexes can be created using one or more columns of a database table, providing the basis for both rapid random lookups and efficient access to ordered records. Indexes support the efficient execution of queries in PHP. An "index" can improve the speed of operation in a table. MySQL automatically creates an index for primary key, foreign key, and unique constraints. In addition, you may want to create "indexes" for other columns that are frequently used in joins or search conditions. The user cannot see indexes. You must have used a "CREATE INDEX" statement to create an index for one or more columns of a table. To create an index, write the table name and column names after the "on" clause. You can also use "UNIQUE" keywords to specify that an "index" has unique values. You can also specify "ASC" and "DESC" keywords with a column name to indicate whether you want the "index" stored in ascending or descending order. If you do not specify "asc" or "desc", then "asc" is the default same as the "order by" keyword (which is also able to sort columns in "asc" or "desc" order.

4.5 TABLE DESIGN

1. LOGIN TABLE

Field Name	Data Type	Constraints
User_ID	Integer(PK)	Auto-increment, Not Null
Username	VARCHAR	Unique, Not Null
Password_Hash	VARCHAR	Not Null
Email	VARCHAR	Unique, Not Null

2. SESSIONS TABLE

Field Name	Data Type	Constraints
Session ID	VARCHAR	Primary key, Not Null
User ID	Integer (FK)	Foreign Key
Expiry	TIMESTAMP	Not Null

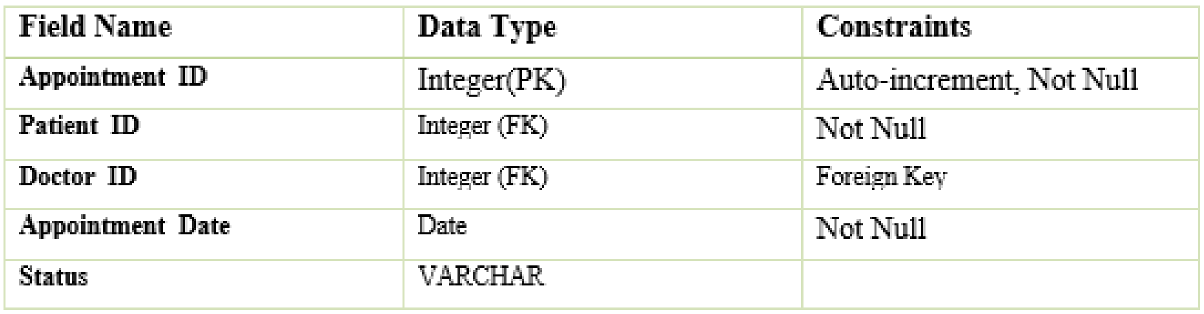
3. PATIENTS TABLE

Field Name	Data Type	Constraints
Patient ID	Integer(PK)	Auto-increment, Not Null
First Name	VARCHAR	Not Null
Last Name	VARCHAR	Not Null
Phone No	VARCHAR	

4. DOCTORS TABLE

Field Name	Data Type	Constraints
Doctor_ID	Integer(PK)	Auto-increment, Not Null
First Name	VARCHAR	Not Null
Last Name	VARCHAR	Not Null
Specialisation	VARCHAR	Not Null
Clinic_ID	VARCHAR	Foreign Key
Awards	VARCHAR	

5. APPOINTMENT TABLE



Field Name	Data Type	Constraints
Appointment ID	Integer(PK)	Auto-increment, Not Null
Patient ID	Integer (FK)	Not Null
Doctor ID	Integer (FK)	Foreign Key
Appointment Date	Date	Not Null
Status	VARCHAR	

CHAPTER 5
SYSTEM TESTING

5.1 INTRODUCTION

Software Testing is the process of executing software in a controlled manner, to answer the question - Does the software behave as specified? Software testing is often used in association with the term verification and validation. Validation is the checking or testing of items, including software, for conformance and consistency with an associated specification. Software testing is just one kind of verification, which also uses techniques such as reviews, analysis, inspections, and walkthroughs. Validation is the process of checking that what has been specified is what the user actually wanted. Other activities which are often associated with software testing are static analysis and dynamic analysis. The static analysis investigates the source code of software, looking for problems and gathering metrics without actually executing the code. Dynamic analysis looks at the behavior of software while it is executing, to provide information such as execution traces, timing profiles, and test coverage information. Several rules can serve as testing objectives.
They are:
Testing is the 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 testing is conducted successfully according to the objectives as stated above, it would uncover errors in the software. Also testing demonstrate that the software function appears to be working according to the specification, that performance requirement appears to have been met. There are three ways to test program.

- For correctness
- For implementation efficiency
- For computational complexity

Test for correctness is supposed to verify that a program does exactly what it was designed to
do. This is much more difficult than it may at first appear, especially for large programs.

5.2 TEST PLAN

A test plan implies a series of desired course of action to be followed in accomplishing various testing methods. The Test Plan acts as a blue print for the action that is to be followed. The software engineers create a computer program, its documentation and related data structures. The software developers are always responsible for testing the individual units of the programs, ensuring that each performs the function for which it was designed. There is an independent test group (ITG) which is to remove the inherent problems associated with letting the builder to test the thing that has been built. The specific objectives of testing should be stated in measurable terms. 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 plan. The levels of testing include:

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

5.2.1 Unit Testing

Unit testing focuses verification effort on the smallest unit of software design – the software component or module. Using the component level design description as a guide, important control paths are tested to uncover errors within the boundary of the module. The relative complexity of tests and uncovered scope established for unit testing. The unit testing is white- box oriented, and step can be conducted in parallel for multiple components. Tests of data flow across a module interface are required before any other test is initiated. If data do not enter and exit properly, all other tests are moot. Good design dictates that error conditions be anticipated and error handling paths set up to reroute or cleanly terminate processing when an error does occur. Unit testing was done in Sell-Soft System by treating each module as separate entity and testing each one of them with a wide spectrum of test inputs. Some flaws in the internal logic of the modules were found and were rectified. After coding each module is tested and run individually. All unnecessary code were removed and ensured that all modules are working, and gives the expected result.

5.2.2 Integration Testing

Integration testing is systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing. The objective is to take unit tested components and build a program structure that has been dictated by design**.** The entire program is tested as whole. Once these errors are corrected, new ones appear and the process continues in a seemingly endless loop. After performing unit testing in the System all the modules were integrated to test for any inconsistencies in the interfaces. Moreover, differences in program structures were removed and a unique program structure was evolved**.**

5.2.3 Validation Testing or System Testing

This is the final step in testing. In this the entire system was tested as a whole with all forms, code, modules and class modules. This form of testing is popularly known as Black Box testing or System tests. Black Box testing method focuses on the functional requirements of the software. That is, Black Box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements for a program. Black Box testing attempts to find errors in the following categories; incorrect or missing functions, interface errors, errors in data structures or external data access, performance errors and 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 the perspective system; and user at the time of developing and make changes whenever required. This is done with respect to the following points: Input Screen Designs,

Output Screen Designs. The above testing is done by 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. In contrast, testing the system by which test data errors are again uncovered and corrected by using the above testing steps and corrections are also noted for future use.

5.2.5 Automation Testing

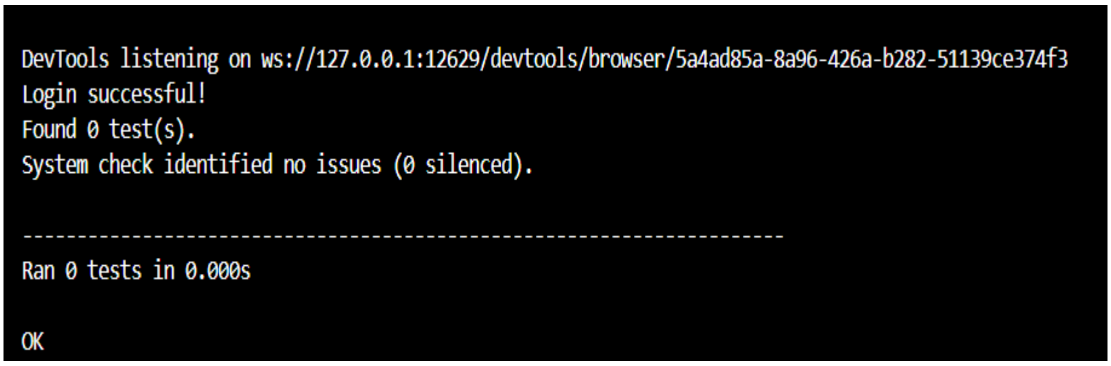
Automated testing is a process that validates if the software is functioning appropriately and meeting requirements before it is released into production. This software testing method uses scripted sequences that are executed by testing tools. UI automation testing is a technique where these testing processes are performed using an automation tool. Instead of having testers click through the application to verify data and action flows visually, test scripts are written for each test case. A series of steps to follow when the verifying data is then added. Automatic testing is required when you want to run the same test cases across multiple machines at the same time.

5.2.6 Selenium Testing

Selenium is an open-source, automated, and valuable testing tool that all web application developers should be well aware of. A test performed using Selenium is usually referred to as Selenium automation testing. However, Selenium is not just a single tool but a collection of tools, each catering to different Selenium automation testing needs. In this tutorial, you will learn all about Selenium and the various types of Selenium automation testing tools. Manual testing, a vital part of the application development process, unfortunately, has many shortcomings, chief of them being that the process is monotonous and repetitive. To overcome these obstacles, Jason Huggins, an engineer at Thought Works, decided to automate the testing process. He developed a JavaScript program called the Java Script Test Runner that automated web application testing. This program was renamed Selenium in 2004.

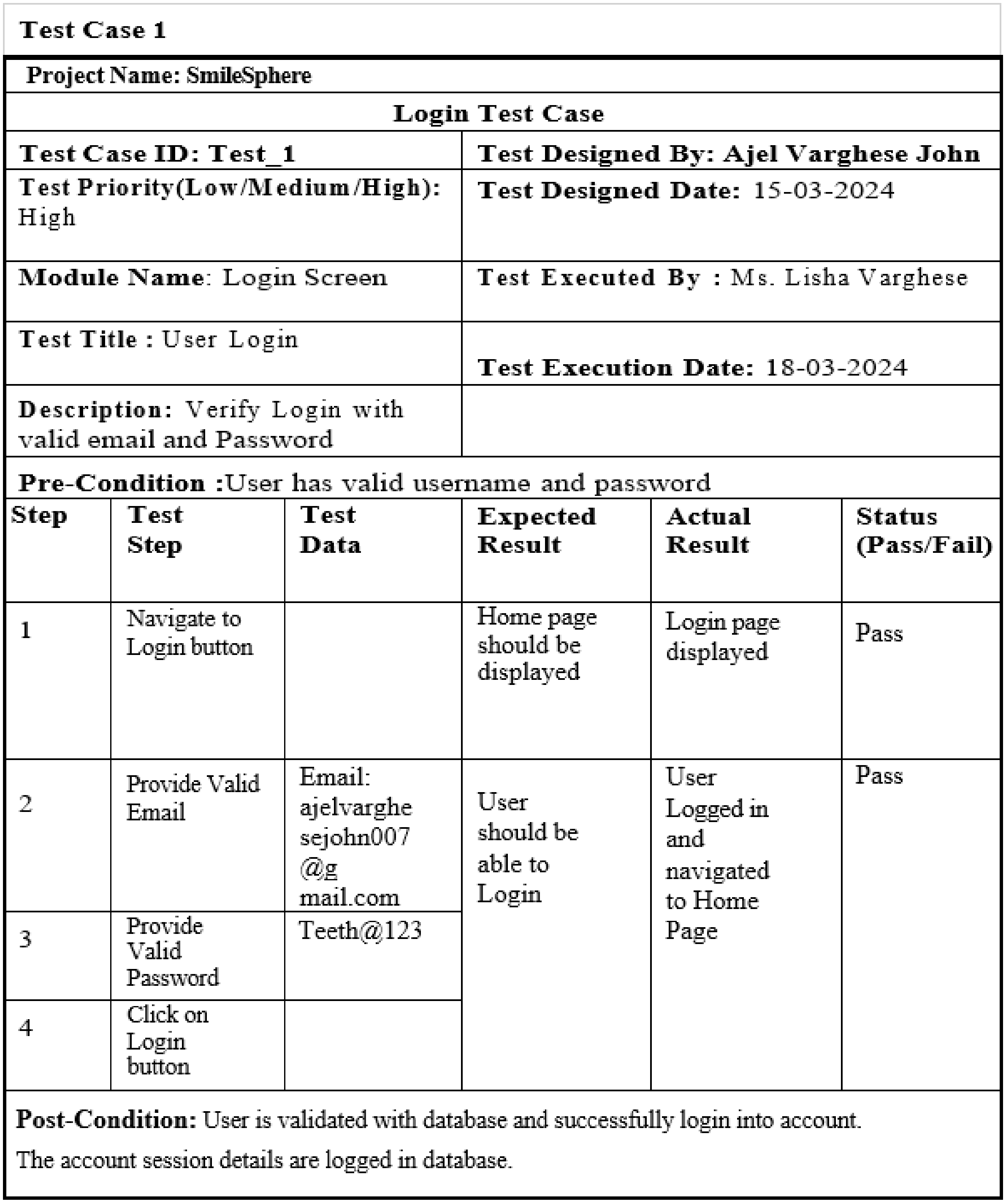
Test Case: clinic_login
Code:

```
import unittest
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.chrome.options import Options
chrome_driver_path = r'C:\Users\ajelv\Downloads\chromedriver-win64\chromedriver.exe'
url = r'C:\Users\ajelv\OneDrive\Documents\SmileSphereApp[1]\template\clinic_login.html'
chrome_options = Options()
driver = webdriver.Chrome(options=chrome_options
driver.get(url)
try:
    clinic_name_input = driver.find_element(By.ID, 'clinic_name')
    password_input = driver.find_element(By.ID, 'password')
    submit_button = driver.find_element(By.XPATH, '//button[contains(text(), "Submit")]')
    cancel_button = driver.find_element(By.XPATH, '//button[contains(text(), "Cancel")]')

    clinic_name_input.send_keys("your_clinic_name")
    password_input.send_keys("your_password")
    submit_button.click()

    print("Login successful!")

except Exception as e:
    print("Error:", e)

finally:
driver.quit()
```



```
DevTools listening on ws://127.0.0.1:12629/devtools/browser/5a4ad85a-8a96-426a-b282-51139ce374f3
Login successful!
Found 0 test(s).
System check identified no issues (0 silenced).

----------------------------------------------------------------------
Ran 0 tests in 0.000s

OK
```

Test Report



Test Case 1

Project Name: SmileSphere	
Login Test Case	
Test Case ID: Test_1	**Test Designed By: Ajel Varghese John**
Test Priority(Low/Medium/High): High	**Test Designed Date:** 15-03-2024
Module Name: Login Screen	**Test Executed By :** Ms. Lisha Varghese
Test Title : User Login	**Test Execution Date:** 18-03-2024
Description: Verify Login with valid email and Password	
Pre-Condition : User has valid username and password	

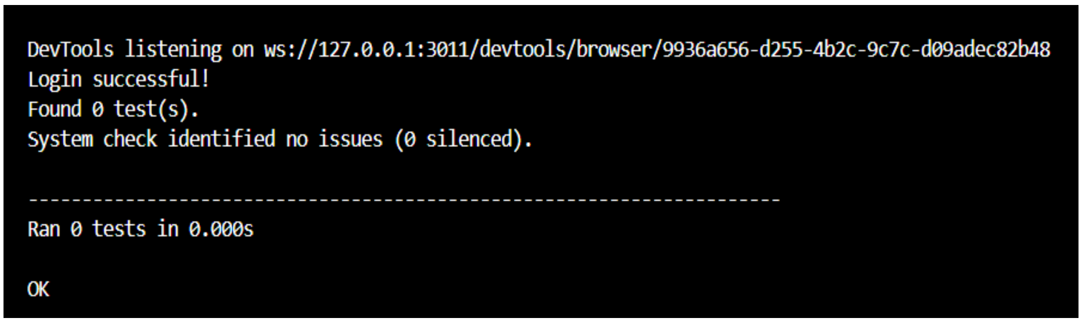
Step	**Test Step**	**Test Data**	**Expected Result**	**Actual Result**	**Status (Pass/Fail)**
1	Navigate to Login button		Home page should be displayed	Login page displayed	Pass
2	Provide Valid Email	Email: ajelvarghesejohn007@gmail.com	User should be able to Login	User Logged in and navigated to Home Page	Pass
3	Provide Valid Password	Teeth@123			
4	Click on Login button				

Post-Condition: User is validated with database and successfully login into account.
The account session details are logged in database.

Test Case 2: patient_login

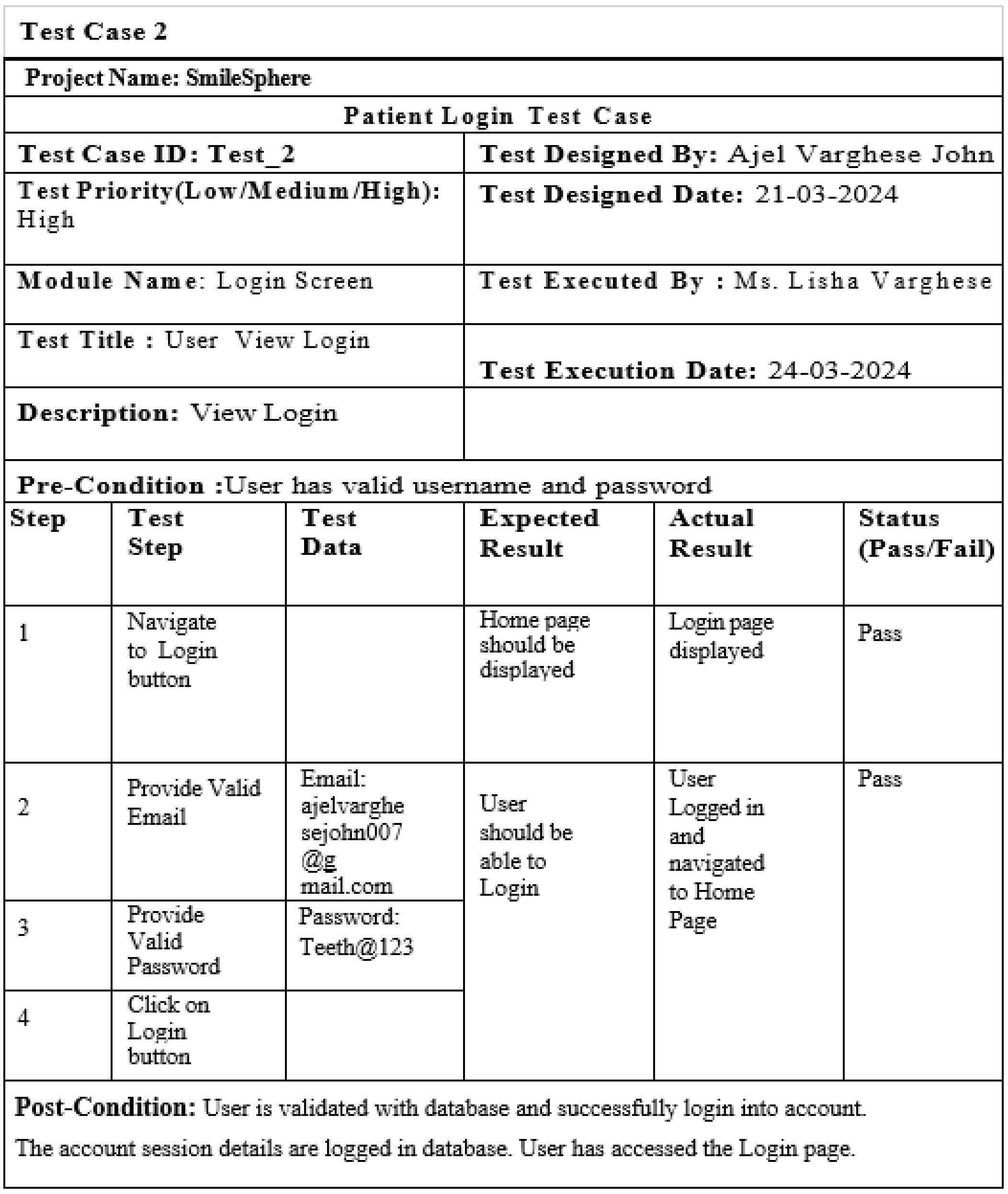
Code:

```
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

chrome_driver_path = r'C:\Users\ajelv\Downloads\chromedriver-win64\chromedriver.exe'
driver = webdriver.Chrome()


url = r'file:///C:/Users/ajelv/OneDrive/Documents/SmileSphereApp[1]/template/patient_login.html'
driver.get(url)

email_input = driver.find_element(By.NAME, "email")
password_input = driver.find_element(By.NAME, "password")

email_input.send_keys("test@example.com")
password_input.send_keys("testpassword")

login_button = driver.find_element(By.XPATH, "//button[@type='submit']")
login_button.click()

try:
    WebDriverWait(driver, 10).until(EC.title_contains("Patient Login"))
    print("Login successful!")
except:
    print("Login failed!")

driver.quit()
```



```
DevTools listening on ws://127.0.0.1:3011/devtools/browser/9936a656-d255-4b2c-9c7c-d09adec82b48
Login successful!
Found 0 test(s).
System check identified no issues (0 silenced).

----------------------------------------------------------------------
Ran 0 tests in 0.000s

OK
```

Test Report



Test Case 2

Project Name: SmileSphere

Patient Login Test Case

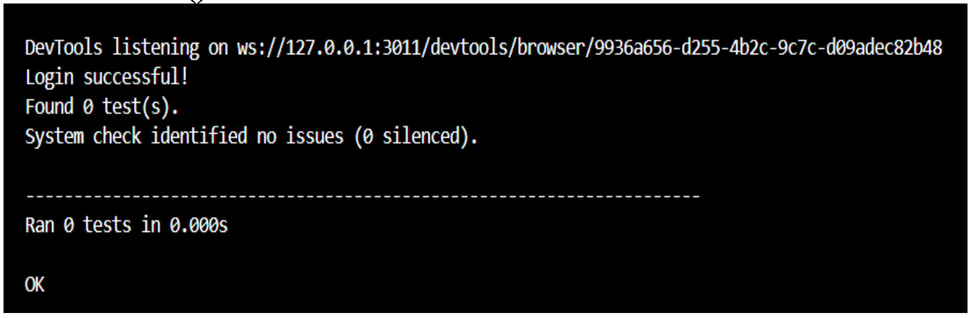
Test Case ID: Test_2			**Test Designed By:** Ajel Varghese John		
Test Priority(Low/Medium/High): High			**Test Designed Date:** 21-03-2024		
Module Name: Login Screen			**Test Executed By :** Ms. Lisha Varghese		
Test Title : User View Login			**Test Execution Date:** 24-03-2024		
Description: View Login					
Pre-Condition : User has valid username and password					
Step	**Test Step**	**Test Data**	**Expected Result**	**Actual Result**	**Status (Pass/Fail)**
1	Navigate to Login button		Home page should be displayed	Login page displayed	Pass
2	Provide Valid Email	Email: ajelvarghesejohn007@gmail.com	User should be able to Login	User Logged in and navigated to Home Page	Pass
3	Provide Valid Password	Password: Teeth@123			
4	Click on Login button				

Post-Condition: User is validated with database and successfully login into account. The account session details are logged in database. User has accessed the Login page.

Test Case 3: Patient appointment

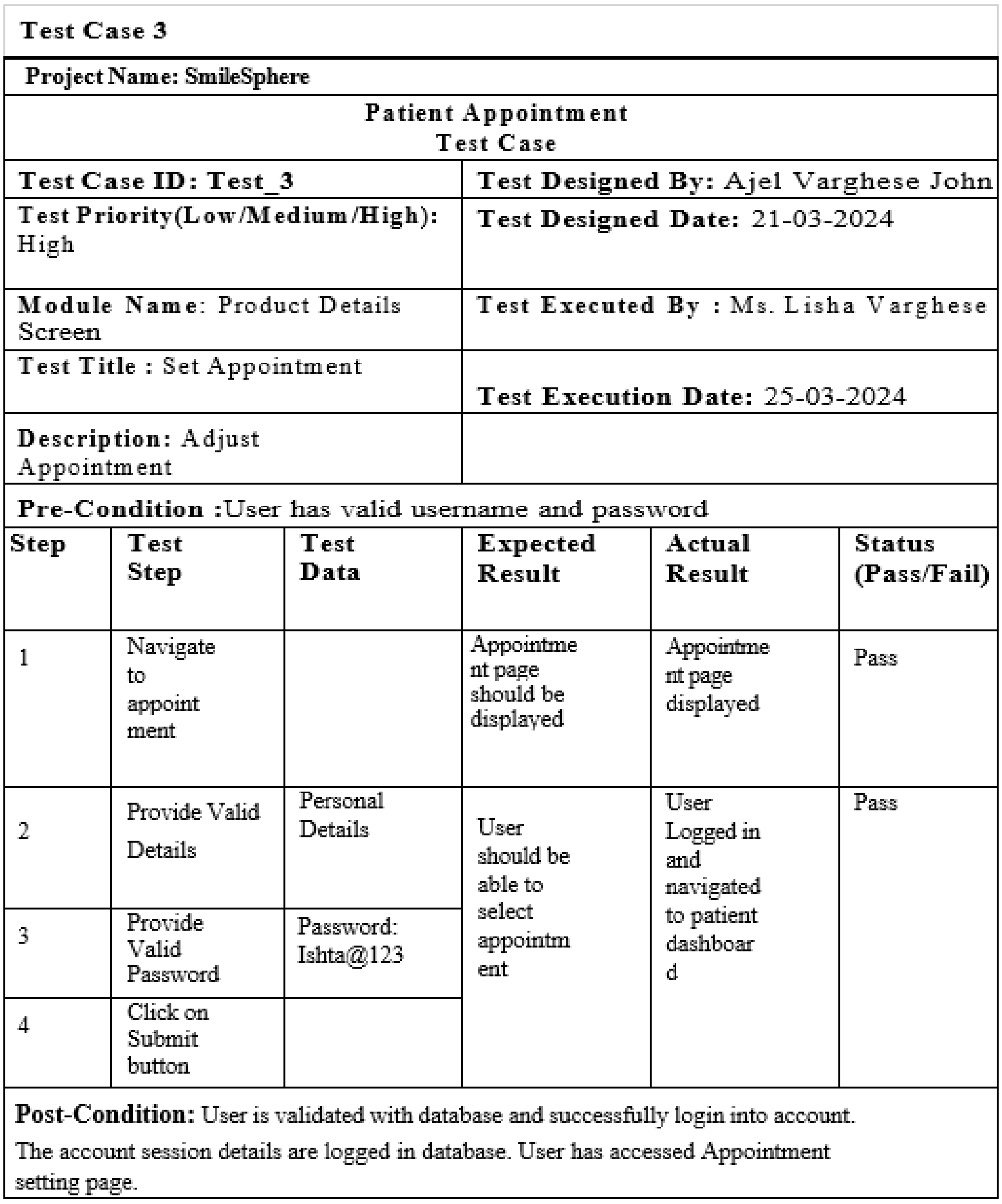
Code:

```
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

class TestAppointmentForm(unittest.TestCase):
def setUp(self):
self.driver = webdriver.Chrome()
self.driver.get("file:///C:/Users/ajelv/OneDrive/Documents/SmileSphereApp[1]
/template/new_appointment.html")

def test_fill_form(self):
name_input = WebDriverWait(self.driver, 30).until(EC.presence_of_element_located((By.ID, "name")))

date_input = self.driver.find_element(By.ID, "date")
clinic_select = self.driver.find_element(By.ID, "clinic")
description_textarea = self.driver.find_element(By.NAME, "description")
submit_button = self.driver.find_element(By.XPATH, "//button[@type='submit']")

name_input.send_keys("John Doe")
date_input.send_keys("2024-04-16") # Change to a valid date
clinic_select.send_keys("Clinic 1") # Choose a clinic option
description_textarea.send_keys("Description of the problem")

submit_button.click()

WebDriverWait(self.driver, 10).until(EC.title_contains("Confirmation"))

self.assertIn("Appointment Created", self.driver.title)



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

if__name__== "_main_":
    unittest.main()
```



```
DevTools listening on ws://127.0.0.1:3011/devtools/browser/9936a656-d255-4b2c-9c7c-d09adec82b48
Login successful!
Found 0 test(s).
System check identified no issues (0 silenced).

----------------------------------------------------------------------
Ran 0 tests in 0.000s

OK
```

Test Report



Test Case 3

Project Name: SmileSphere

Patient Appointment Test Case

Test Case ID: Test_3	**Test Designed By:** Ajel Varghese John
Test Priority(Low/Medium/High): High	**Test Designed Date:** 21-03-2024
Module Name: Product Details Screen	**Test Executed By :** Ms. Lisha Varghese
Test Title : Set Appointment	**Test Execution Date:** 25-03-2024
Description: Adjust Appointment	

Pre-Condition : User has valid username and password

Step	Test Step	Test Data	Expected Result	Actual Result	Status (Pass/Fail)
1	Navigate to appoint ment		Appointme nt page should be displayed	Appointme nt page displayed	Pass
2	Provide Valid Details	Personal Details	User should be able to select appointm ent	User Logged in and navigated to patient dashboar d	Pass
3	Provide Valid Password	Password: Ishta@123			
4	Click on Submit button				

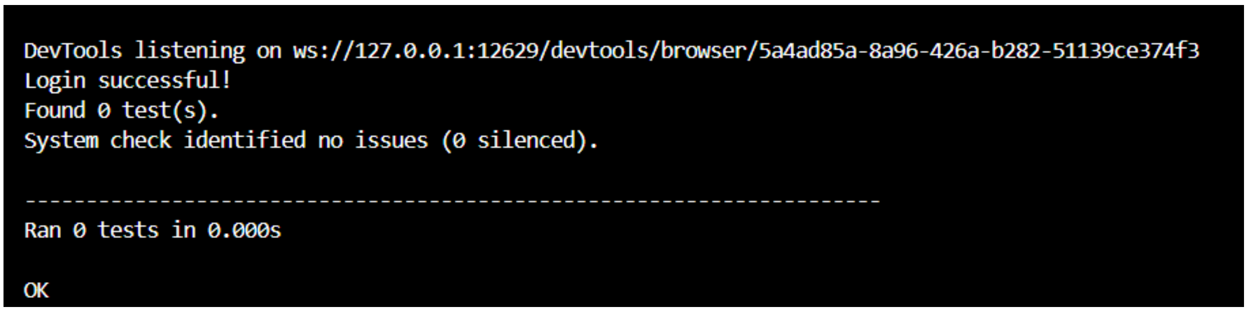
Post-Condition: User is validated with database and successfully login into account. The account session details are logged in database. User has accessed Appointment setting page.

Test Case 4: Patient Register

Code:

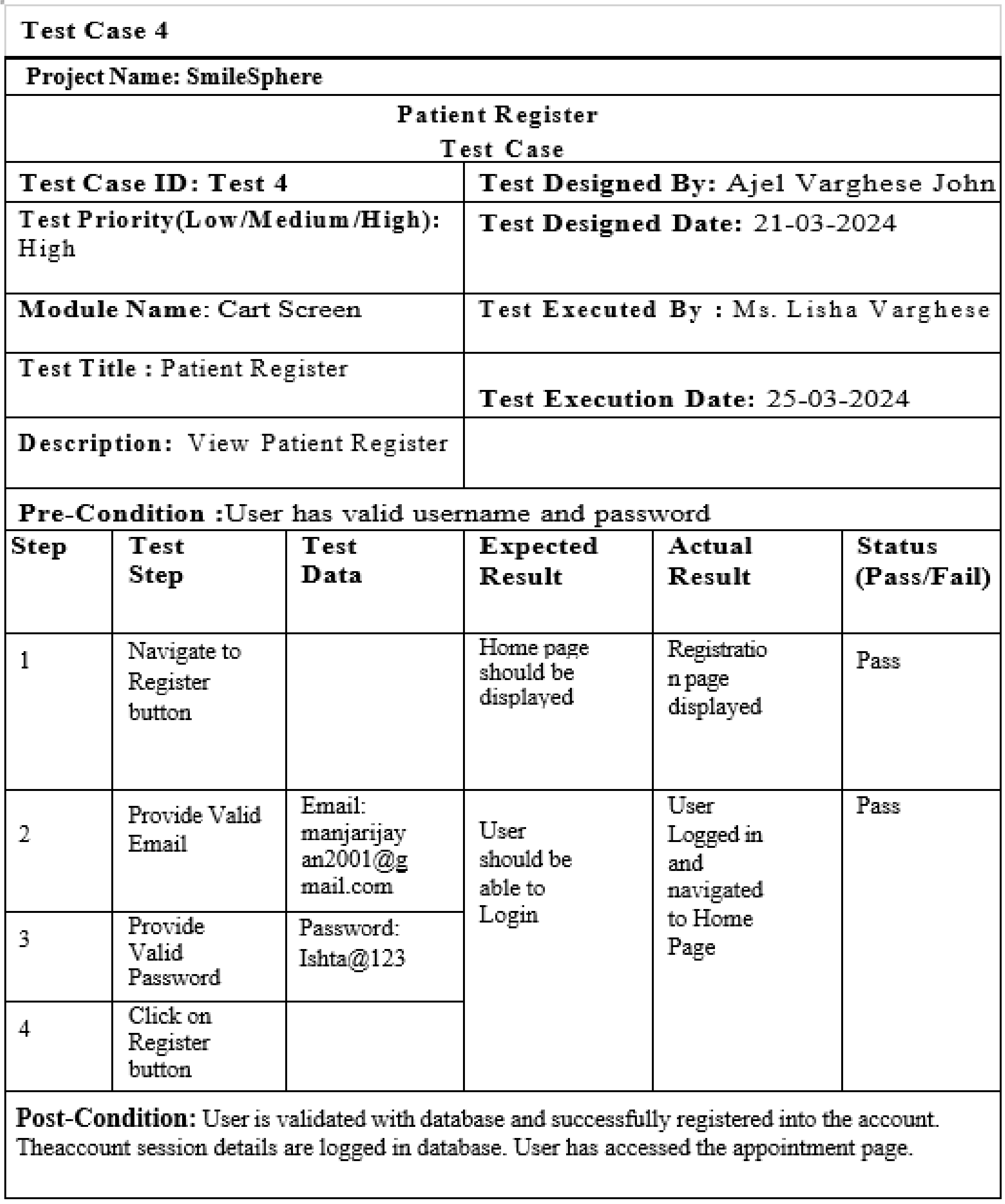
```
import unittest
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC

class TestPatientRegistration(unittest.TestCase):
def setUp(self):
self.driver = webdriver.Chrome()
self.driver.get(r"C:\Users\ajelv\OneDrive\Documents\SmileSphereApp[1]\template
\patient_register.html")

def test_register_patient(self):
WebDriverWait(self.driver, 10).until(EC.visibility_of_element_located((By.TAG_NAME, "form")))
username_input = self.driver.find_element(By.NAME, "username")
nationality_input = self.driver.find_element(By.NAME, "nationality")
email_input = self.driver.find_element(By.NAME, "email")
password_input = self.driver.find_element(By.NAME, "password")
 confirm_password_input = self.driver.find_element(By.NAME, "password_confirm")
 register_button = self.driver.find_element(By.XPATH, "//button[@type='submit']")

  username_input.send_keys("John Doe")
  nationality_input.send_keys("Country")
  email_input.send_keys("test@example.com")
  password_input.send_keys("testpassword")
  confirm_password_input.send_keys("testpassword")
```

```
register_button.click()

WebDriverWait(self.driver, 10).until(EC.title_contains("Confirmation"))

 self.assertIn("Registration Successful", self.driver.title)


 def tearDown(self):

       self.driver.quit()


 if__name__== "_main_":

 unittest.main()
```



```
DevTools listening on ws://127.0.0.1:12629/devtools/browser/5a4ad85a-8a96-426a-b282-51139ce374f3
Login successful!
Found 0 test(s).
System check identified no issues (0 silenced).

----------------------------------------------------------------------
Ran 0 tests in 0.000s

OK
```

Test Report



Test Case 4

Project Name: SmileSphere

Patient Register Test Case

Test Case ID: Test 4	**Test Designed By:** Ajel Varghese John
Test Priority(Low/Medium/High): High	**Test Designed Date:** 21-03-2024
Module Name: Cart Screen	**Test Executed By :** Ms. Lisha Varghese
Test Title : Patient Register	**Test Execution Date:** 25-03-2024
Description: View Patient Register	

Pre-Condition : User has valid username and password

Step	Test Step	Test Data	Expected Result	Actual Result	Status (Pass/Fail)
1	Navigate to Register button		Home page should be displayed	Registration page displayed	Pass
2	Provide Valid Email	Email: manjarijayan2001@gmail.com	User should be able to Login	User Logged in and navigated to Home Page	Pass
3	Provide Valid Password	Password: Ishta@123			
4	Click on Register button				

Post-Condition: User is validated with database and successfully registered into the account. Theaccount session details are logged in database. User has accessed the appointment page.

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 considered to 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

Implementation of software refers to the final installation of the package in its real environment, to the satisfaction of the intended uses and the operation of the system. In many organizations someone who will not be operating it, will commission the software development project. In the initial stage people doubt about the software but we have to ensure that the resistance does not build up, as one has to make sure that:

- The active user must be aware of the benefits of using the new system. Their confidence in the software is built up.
- Proper guidance is imparted to the user so that he is comfortable in using the application.

Before going ahead and viewing the system, the user must know that for viewing the result, the server program should be running in the server. If the server object is not up running onthe server, the actual process won’t take place.

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”.

6.3 HOSTING

Hosting a website entails making it accessible to users on the internet by storing its files and data on a server. This process involves several steps, beginning with choosing a suitable hosting provider that meets your website's requirements for storage, bandwidth, uptime, and support. Once a hosting provider is selected, you typically register a domain name for your website, which serves as its unique address.After domain registration, you set up a hosting account by selecting a plan, providing payment information, and creating an account with the hosting provider. Subsequently, you upload your website files to the server using FTP or a web-based file manager. Organizing these files properly, including HTML, CSS, JavaScript, images, and other assets, ensures smooth functioning.Configuring domain settings to point to the hosting server via DNS records is the next step. Thorough testing of the website for issues such as broken links or formatting problems is crucial before making it live. Once everything is set, you launch the website by updating hosting account settings or additional configurations as needed.Post-launch, regular monitoring of performance, security, and uptime is essential. Many hosting providers offer tools and analytics to help track website metrics and address any arising issues. Consistently updating website content and software ensures security and relevance to the audience, ensuring a seamless online presence.

6.3.1 AMAZON ELASTIC COMPUTE CLOUD (EC2)

Procedure for hosting a website on Amazon EC2:

• **Step1: Create an AWS Account**: Start by registering for an AWS account if you haven't already done so. Once logged in to the AWS Management Console, proceed to the EC2 dashboard.

• **Step2: Launch an EC2 Instance:** Click on the "Launch Instance" button to initiate a new EC2 instance. Select the Ubuntu AMI (Amazon Machine Image) and choose an appropriate instance type according to your project's specifications.

• **Step3: Configure Instance Settings:** Customize instance parameters such as quantity, networking configurations, and storage preferences. Include storage volumes to securely store your project files and data.

• **Step4: Set Up Security Group:** Establish a new security group or utilize an existing one to define firewall regulations. Ensure that ports 22 (SSH) for remote access and 8000 (or any other required port for your application) for web traffic are accessible.

• **Step5: Launch Instance:** Review the instance setup and commence the EC2 instance launch process. Opt for or generate a key pair for SSH access.

- **Step6: Connect to Your Instance:** Upon instance launch, establish an SSH connection. Utilize the public IP address or DNS name of your instance along with the key pair to establish a secure connection.

- **Step7: Clone Project Repository:** Install Git on the EC2 instance and clone the EduSphere Fusion project repository from your Git repository using the git clone command.

- **Step8: Install Python and Django:** Deploy Python and Django on the EC2 instance. Utilize the apt package manager for Python installation and pip to install Django along with any other necessary Python packages.

- **Step9: Install Dependencies:** Install additional packages and dependencies essential for running the website, including database drivers and Django extensions.

- **Step10: Configure Django Settings:** Update the Django settings file (settings.py) with the required database configuration, static file settings, and other project-specific configurations.

- **Step11: Run Django Server:** Initiate the Django development server by executing the command python manage.py runserver 0.0.0.0:8000. This command operates the server on port 8000, accessible on all network interfaces.

- **Step12: Test Your Website:** Access a web browser and navigate to the public IP address or DNS name of your EC2 instance followed by the designated port number (<public_ip>:8000). Confirm that your website is accessible and operates correctly.

- **Step13: Domain Name Configuration (Optional):** If you own a domain name, configure the DNS settings to point to the public IP address of your EC2 instance.

- **Step14: SSL/TLS Certificate Setup (Optional):** Enable HTTPS for your website by setting up an SSL/TLS certificate using AWS Certificate Manager or a third-party provider.

- **Step15: Monitor Your EC2 Instance:** Employ AWS CloudWatch or other monitoring utilities to oversee the performance, security, and uptime of your EC2 instance. Regularly update your instance and software to ensure security and reliability.

Hosted Website: Amazon EC2

Hosted Link: http://18.215.147.211:8000/

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

In conclusion, the Dental Hospital Booking System represents a pivotal advancement in dental care management, optimizing the appointment scheduling process and fostering a user-centric environment for patients, dental professionals, and administrators. The system's commitment to personalized healthcare,

real-time appointment scheduling, and event-based searching enhances efficiency and convenience in dental service delivery. Rigorous testing ensures reliability and security, laying a strong foundation for future developments. Looking ahead, potential advancements include telehealth integration, data analytics, enhanced customization options, mobile application development, EHR integration, machine learning for appointment prediction, patient feedback systems, and internationalization. Embracing these avenues will propel the system towards becoming a comprehensive, cutting-edge solution that not only meets but anticipates the evolving needs of the dental healthcare landscape.

7.1 FUTURE SCOPE

Looking ahead, the future scope of the Dental Hospital Booking System encompasses a comprehensive evolution towards cutting-edge healthcare management. The system is poised to integrate telehealth solutions for virtual consultations, implement advanced data analytics for strategic decision-making, and enhance customization options for patients in collaboration with dental professionals. Further developments include the creation of a dedicated mobile application, integration with electronic health records, and the incorporation of machine learning algorithms for appointment prediction. Introducing patient feedback and ratings systems will contribute to ongoing service improvement, while internationalization and multilingual support aim to broaden accessibility. Collectively, these future enhancements position the system at the forefront of dental healthcare, fostering efficiency, personalization, and technological innovation for an enhanced patient experience.

CHAPTER 8

BIBLIOGRAPHY

REFERENCES:

- PankajJalote, “Software engineering: a precise approach”
- Gary B. Shelly, Harry J. Rosenblatt, “System Analysis and Design”, 2009
- Ken Schwaber, Mike Beedle,Agile Software Development with Scrum Pearson
- Roger S Pressman, “Software Engineering”
- IEEE Std 1016 Recommended Practice for Software Design Descriptions

WEBSITES:

- www.w3schools.com
- www.bootstrap.com
- www.ogaan.com

CHAPTER 9

APPENDIX

1.1 Sample Code

Registration

```
{% load static %}
<!DOCTYPE html>
<html>

<head>
   <title>Clinic Registration</title>
   <style>
      body {
         font-family: Arial,
         sans-serif;background-
         color: #f3f3f3; margin:
         0;
         padding: 0;
      }

      h2 {
         text-align:
         center;
         color: #333;
         margin-top:
         20px;
      }

      form {
         background-
         color: #fff;
         padding: 20px;
         width: 70%;
         margin: 0
         auto; border-
         radius: 10px;
         box-shadow: 0 0 10px rgba(0, 0, 0, 0.2);
      }

      label {
         display:
         block;
         margin-top:
         15px;
      }
```

```
input {
    width:
    100%;
    padding:
    10px;
    margin-
    top: 5px;
    border: 1px
    solid #ccc;
    border-radius:
    5px;
}

h3 {
    margin-top:
    20px;font-
    size: 18px;
    color: #333;
}

input[type="submit"] {
    background-color:
    #4CAF50;color: white;
    padding: 10px
    20px;border:
    none;
    border-
    radius: 5px;
    cursor:
    pointer;
    margin-top:
    20px; font-
    size: 16px;
}

input[type="submit"]:ho
    ver { background-
    color: #45a049;
}

/* Add animation to the
form */@keyframes
fadeIn {
    0% {
        opacity: 0;
        transform: translateY(-10px);
    }

    100% {
        opacity: 1;
        transform: translateY(0);
```

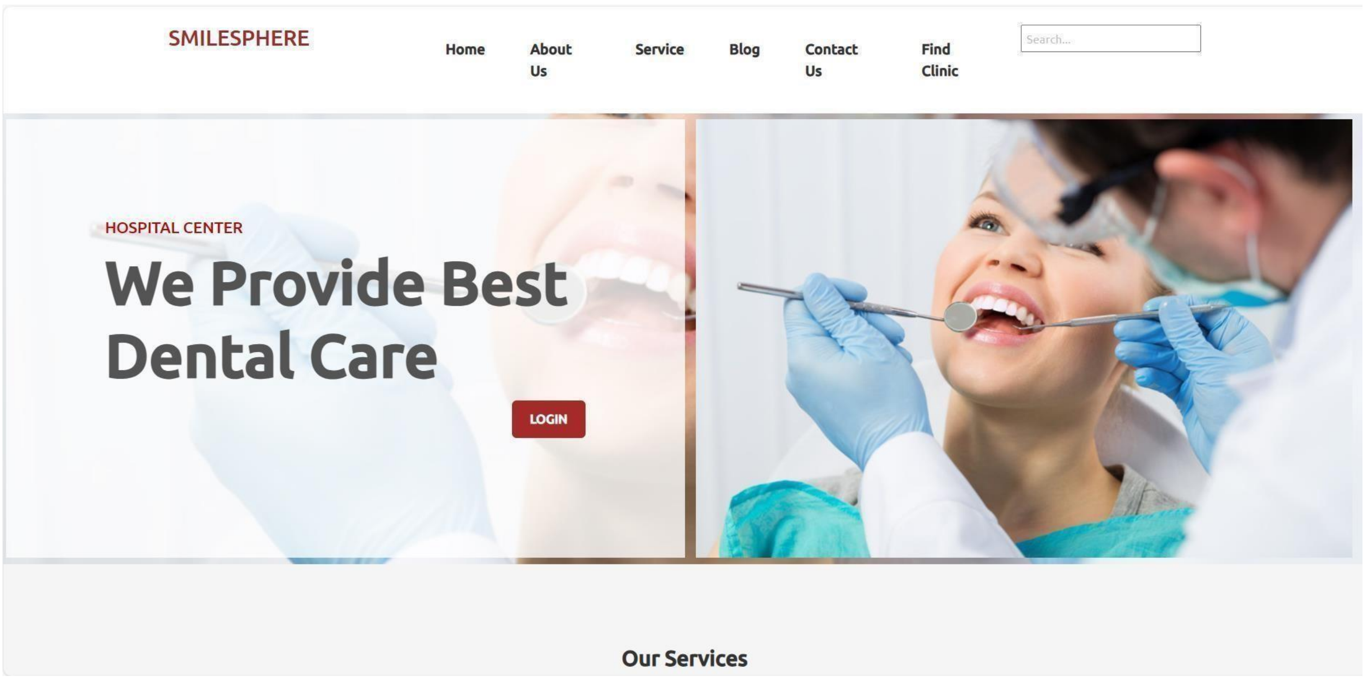
```
            }
        }

        form {
            animation: fadeIn 0.5s ease-in;
        }
    </style>
</head>

<body>
    <h2>Clinic Registration</h2>
```

```
<form action="/clinic-registration-submit" method="post" onsubmit="return validateForm()">
    {% csrf_token %}
    <label for="name">Clinic Name:</label>
    <input type="text" id="name" name="name" required><br><br>

    <label for="address">Address:</label>
    <input type="text" id="address" name="address" required><br><br>

    <label for="phone">Phone Number:</label>
    <input type="tel" id="phone" name="phone" required><br><br>

    <label for="email">Email:</label>
    <input type="email" id="email" name="email" required><br><br>

    <label for="password">Password:</label>
    <input type="password" id="password" name="password" required><br><br>

    <label for="confirm_password">Confirm Password:</label>
    <input type="password" id="confirm_password" name="confirm_password" required><br><br>

    <h3>Doctor Details</h3>
    <label for="doctor_name">Doctor's Name:</label>
    <input type="text" id="doctor_name" name="doctor_name" required><br><br>

    <label for="doctor_specialty">Doctor's Specialty:</label>
    <input type="text" id="doctor_specialty" name="doctor_specialty" required><br><br>

    <label for="doctor_email">Doctor's Email:</label>
    <input type="email" id="doctor_email" name="doctor_email" required><br><br>

    <h3>Guest Surgeons (Specific Day)</h3>
    <label for="surgeon_name">Surgeon's Name:</label>
    <input type="text" id="surgeon_name"><br><br>

    <label for="surgeon_specialty">Surgeon's Specialty:</label>
    <input type="text" id="surgeon_specialty"><br><br>

    <label for="surgeon_email">Surgeon's Email:</label>
    <input type="email" id="surgeon_email"><br><br>\

    <label for="event_date">Event Date (Guest Surgeon):</label>
    <input type="date" id="event_date"><br><br>

    <input type="submit" value="Register">
</form>

<script>
    function validateForm() {
        var name = document.getElementById("name").value;
```

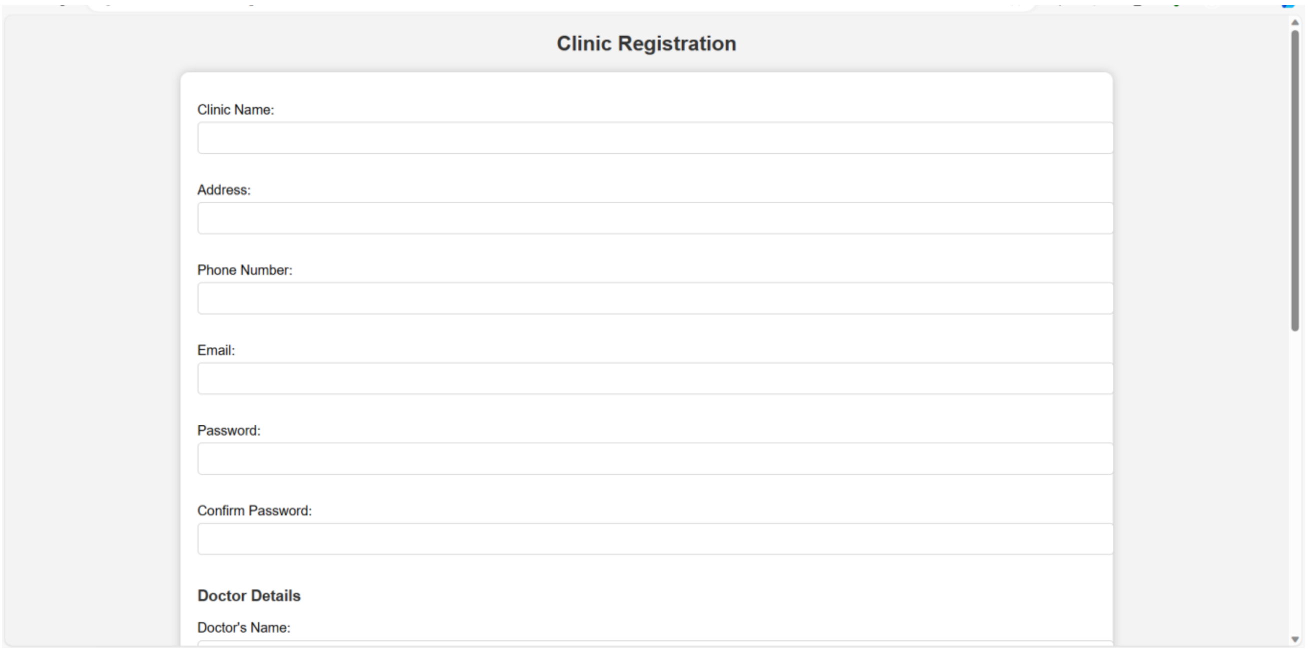
```
            var address =
            document.getElementById("address").value;var
            phone = document.getElementById("phone").value;
            var email =
            document.getElementById("email").value;
            var password = document.getElementById("password").value;
            var confirmPassword =
            document.getElementById("confirm_password").value;var doctorName
            = document.getElementById("doctor_name").value;
            var doctorSpecialty =
            document.getElementById("doctor_specialty").value;var doctorEmail
            = document.getElementById("doctor_email").value;
            var surgeonName = document.getElementById("surgeon_name").value;
            var surgeonSpecialty =
            document.getElementById("surgeon_specialty").value;var surgeonEmail
            = document.getElementById("surgeon_email").value;
            var eventDate = document.getElementById("event_date").value;

            if (name.trim() === "" || address.trim() === "" || phone.trim() === "" || email.trim()
=== "" ||password.trim() === "" || confirmPassword.trim() === "" || doctorName.trim() ===
"" || doctorSpecialty.trim() === "" || doctorEmail.trim() === "") {
                alert("Please fill out all required
                fields.");return false;
            }

            if (password !==
                confirmPassword) {
                alert("Passwords do not
                match."); return false;
            }

            // Add more specific validation if needed

            return true;
        }
    </script>
</body>

</html>
```

9.1 Screen Shots

Index page



Clinic Registration


Clinic Registration

Clinic Name:

Address:

Phone Number:

Email:

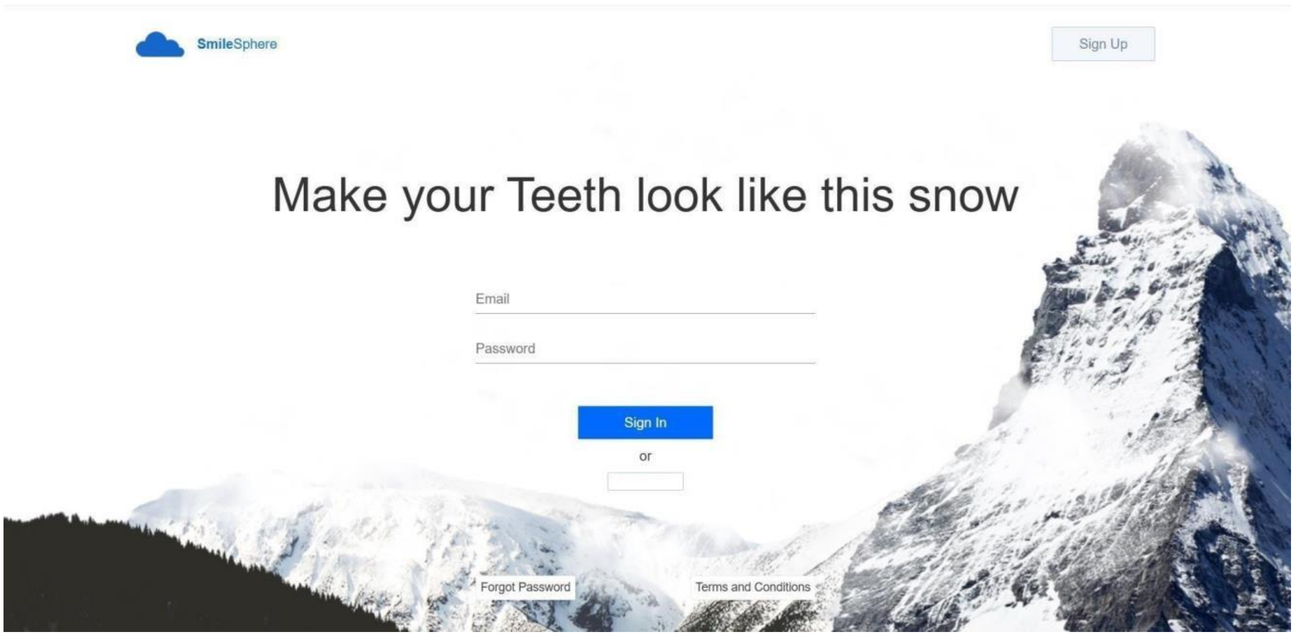
Password:

Confirm Password:

Doctor Details

Doctor's Name:

Patient Login


SmileSphere
Sign Up
Make your Teeth look like this snow
Email
Password
Sign In
or
Forgot Password
Terms and Conditions