

A Mini Project On

SMILE CARE

Submitted in Partial Fulfilment of Requirement
For the Award of the Degree
Bachelor of Computer Application

MANGALORE UNIVERSITY



UNDER THE GUIDANCE OF

Mr. Prakash Acharya
Lecture of Computer Science Department
St Mary's College, Shirva

By

Sujith Arun Shrisiddhi
(U05CS21S0033) (U05CS21S0039) (U05CS21S0031)



DEPARTMENT OF COMPUTER SCIENCE

ST. MARY'S COLLEGE, SHIRVA-574116

2023-24

ST. MARY'S COLLEGE, SHIRVA- 574116 2023-24



DEPARTMENT OF COMPUTER SCIENCE

CERTIFICATE

This is to certify that the mini project work entitled “Smile Care” has been carried out by Sujith(U05CS21S0033), Arun (U05CS21S0039), shrisiddhi(U05CS21S0031) students of VI semester BCA, in Partial fulfilment of the requirements to the completion of VI semester, Bachelor of Computer Applications of Mangalore University during the even semester of the academic year of 2023-24.

Internal Guide

Mr. Prakash Acharya

Lecturer of Computer Science Department

St Mary's College, Shirva

Head of the Department
Mr K. Praveen Kumar
Department of Computer Science

Principal
Dr. Herald Ivan Monis

Examiners:

- 1.
- 2.

Date:

ACKNOWLEDGMENT

On the successful completion of our mini project, “Smile Care”, we feel proud to express our gratitude to the mini project to the people who have helped us directly and indirectly for bringing the source of help and boosting us throughout our work.

First of all, we thank Almighty for showing his abundant blessings on us.

We are grateful to Dr. Herald Ivan Monis, Principal, St. Mary's College, Shirva for providing all the required facilities for completion of our project.

We would like to express our immense gratitude to our HOD Mr. K Praveen Kumar Department of Computer Science who helped us to do the mini project.

We would like to express our immense gratitude to Mr Prakash Acharya, lecturer of the Department of Computer Science and our guide, without whose help we would not have seen the light of the day and wouldn't have completed our mini project work.

Finally, the word of thanks goes to my family and friends for guiding, encouraging and inspiring me all through the mini project.

Place: Shirva

Sujith

Date:

Arun

ShriSiddhi

DECLARATION

We hereby declare that the mini project entitled “Smile Care” by us for the partial fulfillment for the award of degree of Bachelor in Computer Applications, of Mangalore University embodies the result of our mini project work carried out under the direct supervision and guidance of Mr Prakash Acharya, Lecturer of Computer Science Department, St. Mary’s College, Shirva during the even semester of the academic year of 2023-24. We further declare that the part of the project has not been previously submitted to any other examination or institution.

Place: Shirva

Sujith

Date:

Arun

ShriSiddhi

Chapter No	Contents	Page No
1	Synopsis	
	1.1 Title of the project 1.2 Objective 1.3 Project Category 1.4 Languages to be used 1.5 Structure of the project 1.6 Module Description 1.7 Future scope of the project 1.8 Tools and Environment used 1.9 Hardware interface 1.10 Software interface	
2	Literature Survey	
	2.1 HTML 2.2 CSS 2.3 Javascript 2.4 MYSQL 2.5 PHP 2.6 XAMPP	
3	Software Requirements System	
	3.1 Introduction	

	3.2 Purpose 3.3 Scope 3.4 Definition, Acronyms, Abbreviations 3.5 Overall Description 3.6 External Interface requirement 3.7 Functional Requirement 3.8 Non Functional Requirement	
4	System Design 4.1 Introduction 4.2 Context Flow Diagram 4.3 Data Flow Diagram	
5	Database Design 5.1 Introduction 5.2 Schema Description 5.3 Entity Relationship (ER) Diagram	
6	Detailed Design 6.1 Introduction 6.2 Structured English	
7	System Coding	
8	Testing	
9	Snapshots	
10	Future Enhancement	
11	Conclusion	
12	Bibliography	

Chapter – 1

Synopsis

1.1 Title :

The title of our project is “Smile Care”.

1.2 Main objective of the project: 1. Simplify

appointment scheduling for Smile Care services.

2. Provide real-time tracking of appointments.

3. Send diagnosis details to patients through the website.

4. Improve overall efficiency and accessibility of Smile Care services.

1.3 Project category:

Website for end users.

1.4 Languages to be used:

- Front end: HTML,CSS,JS.
- Back end: PHP.

1.5 Structure of the project:

Problem analysis :

○ Manual Processes:

Many dental practices still rely on manual methods for appointment scheduling, patient records, and billing, leading to inefficiencies and potential errors.

○ Limited Online Presence:

Lack of a comprehensive digital platform limits the practice's ability to attract new patients and offer convenient services such as online appointment booking and virtual consultations.

Features:

○ Appointment Management:

Efficient scheduling and management of patient appointments, reducing wait times and optimizing dentist schedules.

○ Online Presence:

User registration and login functionality for patients to access their profiles and book appointments online.

○ **Administrator Dashboard:**

Admin login for managing clinic operations, including staff schedules, patient records, and financial reporting.

○ **User-Friendly Interface:**

Intuitive interface designed for easy navigation by dentists, receptionists, and patients, enhancing user experience and satisfaction.

1.6 Module Description :

○ **User Registration and Login:**

Patients and doctors can register and log in to their respective accounts.

○ **Appointment Booking:**

Users can schedule dental appointments based on their preferred time and date.

○ **Real-Time Tracking:**

Patients can track the status of their appointments in real-time.

○ **Diagnostic Details:**

After the appointment, diagnostic details are sent to the patients via the website.

1.7 Future scope of the Project:

- Integration of telemedicine features for virtual consultations.
- Implementation of an electronic health record (EHR) system.
- Expansion to include additional features like prescription management.
- Direct communication between patients and dental practitioners.

1.8 Tools and Environment used:

Operating System	Window
Front End	Html, CSS, JS
Back End	MySQL, PHP

1.9 Hardware interface:

System with 4GB ram with intel i3 processor

1.10 Software interface:

Operating System: Windows, macOS, Linux

Web Server: XAMPP control panel

Database Management System: MySQL

Chapter-2

Literature Survey

2.1 HTML:

HTML stands for Hyper Text Markup Language. It uses markup tags to structure web pages.

HTML tags are keywords surrounded by angle brackets like <html>.

HTML documents contain HTML tags and plain text, and they are also called Web Pages.

2.2 CSS:

CSS (Cascading Style Sheets) is a style sheet language used to describe the presentation of a document written in HTML or XML.

CSS defines how to display HTML elements and helps create fast-loading, standards-compliant, and easily modifiable web pages.

External Style Sheets are stored in CSS files.

CSS3 is the latest standard for CSS and is backward-compatible with earlier versions.

2.3 JavaScript:

JavaScript (JS) is a dynamic programming language commonly used for client-side scripting to control the browser, communicate with users, and alter document content.

It is also used in server-side programming, game development, and creating desktop and mobile applications.

JavaScript's syntax is inspired by Java and is relatively easy to learn.

2.4 MySQL:

MySQL is a database system used on the web that runs on a server.

It is ideal for both small and large applications.

MySQL is fast, reliable, easy to use, and uses standard SQL.

2.5 PHP:

PHP stands for Hypertext Preprocessor. It is a popular server-side scripting language used to create dynamic web pages.

PHP scripts are executed on the server, and the result is returned to the browser as plain HTML.

PHP can be embedded within HTML and is widely used in web development.

2.6 XAMPP:

XAMPP is a free and open-source cross-platform web server.

XAMPP stands for Cross Platform (X), Apache (A), MariaDB (M), PHP (P), and Perl (P).

It is a simple, lightweight Apache distribution that makes it easy for developers to create a local web server for testing and deployment purposes

Chapter - 3

Software Requirements Specification

3.1 Introduction:

Software Requirement Specification(SRS) is the starting point of the software development activity. Software Requirement Specification is focused specifically on the functioning of the system. The basic purpose of the SRS is to build the communication between the parties involved in the development project. It allows developer to understand the system, functions to be carried out.

3.2 Purpose:

The purpose of this Software Requirements Specification (SRS) document is to outline the functional and non-functional requirements of the Smile Care Website project. It serves as a guideline for the development team to understand the project objectives, scope, constraints, and interfaces.

3.3 Scope:

The scope of the document is to provide sufficient detail functional and non-functional requirements of the project to see that all the functionalities are properly implemented. Any changes made to the requirement will have to go through a formal approval process

3.4 Definition , Acronyms, Abbreviations

Abbreviation	Definition
HTML	Hypertext Markup Language
CSS	Cascading Style Sheets
JS	Javascript
SQL	Structured Query Language

3.5 Overall Description

The 'SmileCare' Dental Management System is a comprehensive software solution for managing dental care services, offering intuitive modules for appointment scheduling, patient records management, billing, and more. This section of the SRS outlines the general features and requirements of the product.

3.5.1 Product Perspective

The SmileCare Dental Management System serves as a standalone software dedicated to streamlining dental care services. It provides a user-friendly interface for dentists, receptionists, and patients to interact with, facilitating efficient management of dental practices.

3.5.2 Product Function

- Patient appointment scheduling and management.
- Reporting and analytics features for practice performance evaluation.

3.5.3 User Classes and Characteristics interface:

- Dentist: Manages patient appointments, diagnoses, and treatments.
- Patient: Makes appointments, views medical records, and manages personal information.
- Admin: Oversees system configuration, user access, and data management.

3.5.4 General constraints

- Compatibility with standard web browsers.
- Operating system compatibility: Windows, macOS, Linux.
- Dependency on a reliable internet connection for accessing cloud-based features.

3.5.5 Assumptions and Dependencies

- Assumptions:
Reliable internet connectivity is available for accessing cloud-based features.
- Dependencies:
Integration with cloud-based servers for data storage and synchronization.
Compatibility with industry-standard dental imaging and diagnostic tools.

3.6 External Interface requirement

This section provides a detailed description of all inputs into and outputs from the system, along with hardware and software interface specifications.

3.6.1 User Interface

The user interface should be intuitive and easy to navigate, facilitating seamless interaction for dentists, receptionists, and patients.

3.6.2 Hardware and Software interface

- Operating System: Windows, macOS, Linux.
- Web Server: Cloud-based servers for data storage and synchronization.
- Database Management System: Integration with MySQL or equivalent for data management.

3.7 Function Requirements:

User Registration and Authentication:

- Users should be able to register for an account as patients or dental practitioners.
- Registered users should be able to log in securely using their credentials.

Appointment Management:

- Patients should be able to view available appointment slots.
- Patients should be able to schedule, reschedule, and cancel appointments.
- Dental practitioners should be able to view their appointment schedules.
- Dental practitioners should be able to confirm or reject appointment requests.

Real-Time Tracking:

- Patients should be able to track the status of their appointments in real-time.
- Dental practitioners should have access to real-time updates on their appointment schedules.

Communication:

- Patients should be able to communicate directly with their assigned dental practitioners. ▪
- Dental practitioners should be able to communicate with their patients regarding appointments and treatment plans.

Diagnostic Details:

- After appointments, patients should receive diagnostic details and treatment plans through the system.

3.8 Non-Functional Requirements:

Performance:

- The system should be responsive and able to handle multiple concurrent users. ▪ Response times for actions such as appointment scheduling and communication should be minimal.

Security:

- User authentication and sensitive data transmission should be secured using encryption protocols.
- Access control mechanisms should be in place to ensure that users can only access data relevant to them.

Reliability:

- The system should be available 24/7 with minimal downtime for maintenance.
- Backup and recovery mechanisms should be implemented to safeguard against data loss.

Constraints:

- The system's performance may be affected by the hardware and network infrastructure.
- The system is dependent on internet connectivity for both users and the server.

Chapter – 4

System Design

4.1 Introduction

4.1.1 System Analysis:

The system analysis approach emphasizes a closed look on all parts of the system. The analyst must consider all the system elements, their inputs, outputs, control, feedback and the environment when the system is being constructed.

4.1.2 System Design:

The goal of system design phase is to produce a model or representation of the system, which can be used to build the system. Here the emphasis is on translating the requirements of the system into design specification.

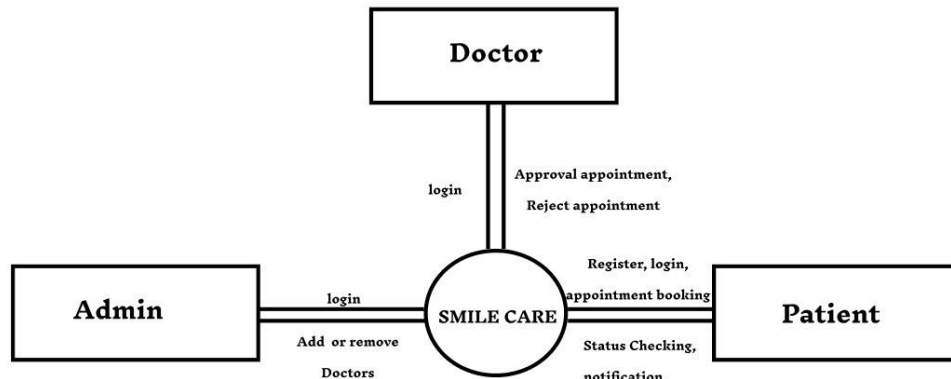
4. 1.3Applicable_Documents:

The document used in system design is Software Requirement Specification Document.

4.2 Context Flow Diagram:

Context flow diagram is a top-level data flow diagram. It only contains one process node that generalizes the function of the entire system in relationship to external entities. In context diagram the entire system is treated as a single process and all its inputs, outputs, sinks and sources are identified and shown.

- **Context Flow Diagram(0):**



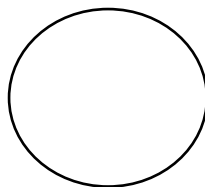
4.3 Data Flow Diagram:

A data flow diagram is a graphical representation of the flow of data through an information system. A data flow diagram can also be used for the visualization of the data processing. It is common practice for a designer to draw a context level DFD. It shows the interaction between the system and the outside entities. This context level DFD, is then exploded to show more detail of the system being modelled.

A DFD represents flow of data through a system. Data flow diagrams are commonly used during problem analysis. It views a system as a function that performs the input into the desired output. A DFD shows movement of data through the different transformations or processes in the system.

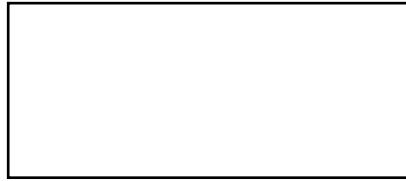
- **Notations in the DFD:**

1. **Circle**



The Circle represents a process. A process is named and each process is represented by a named Circle.

2. Rectangle



The source or sink is represented as a rectangular box. The source or sink is the net originator or the consumer of the data that flows in the system.

3. Parallel Lines



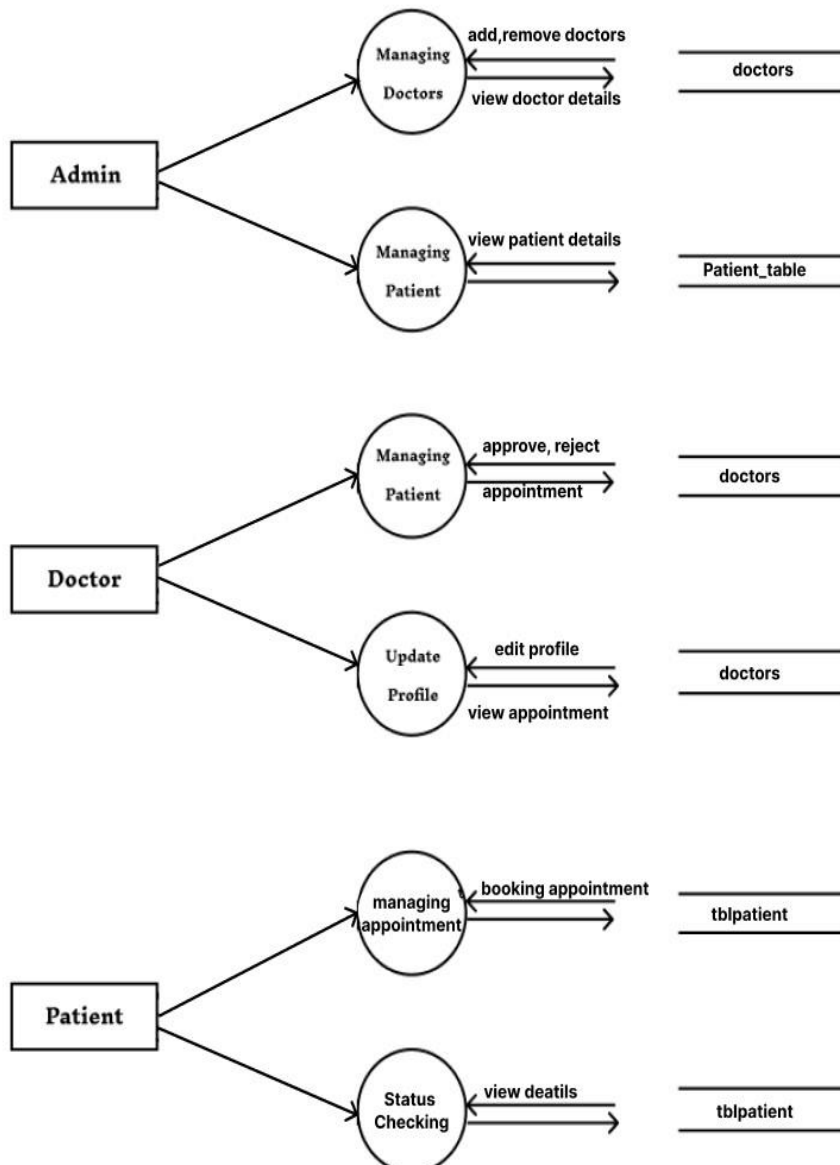
The table is represented with the parallel lines.

4. Arrow

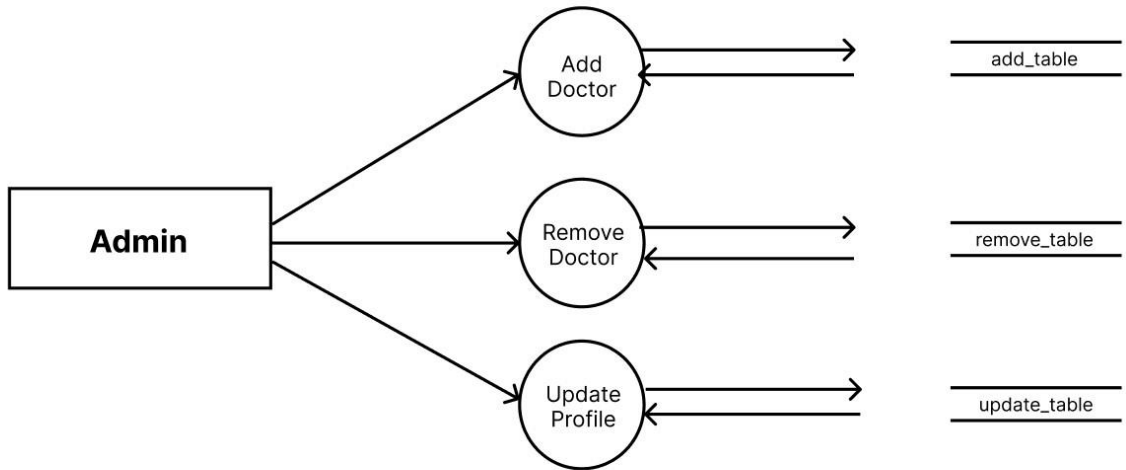


The arrow represents the flow of data through the system. The labeled arrows enter or leave the bubbles

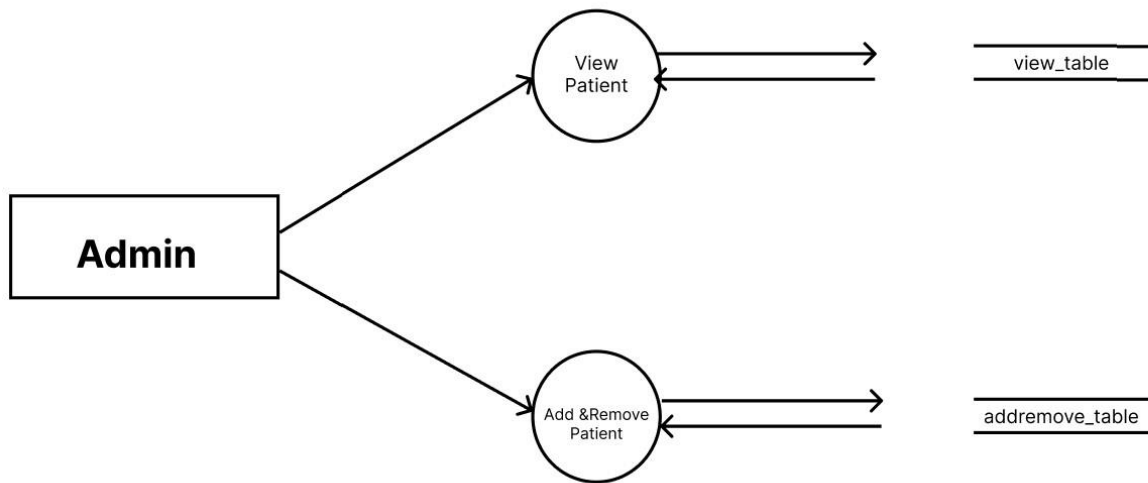
I. DFD Level 1



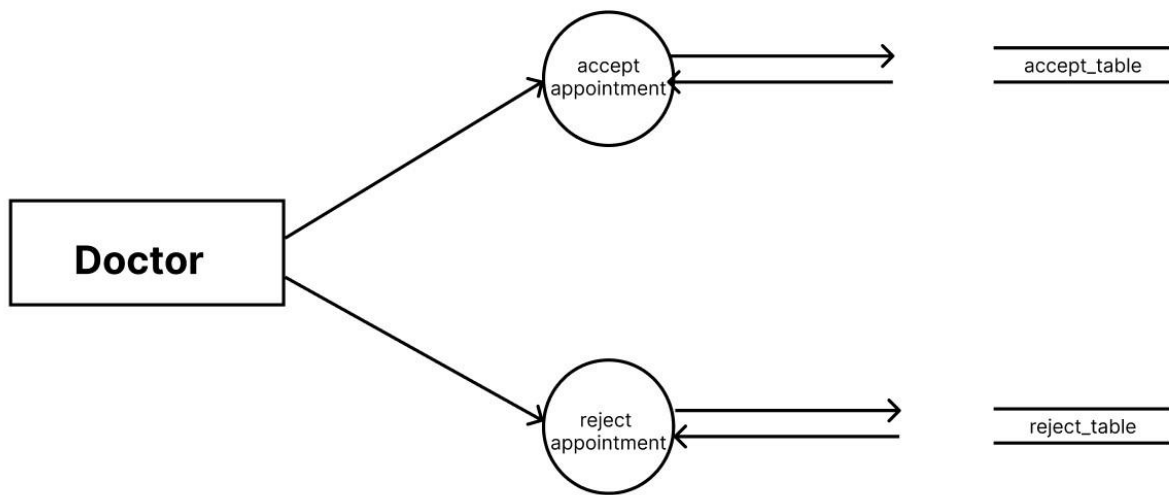
II. DFD Level 2.1



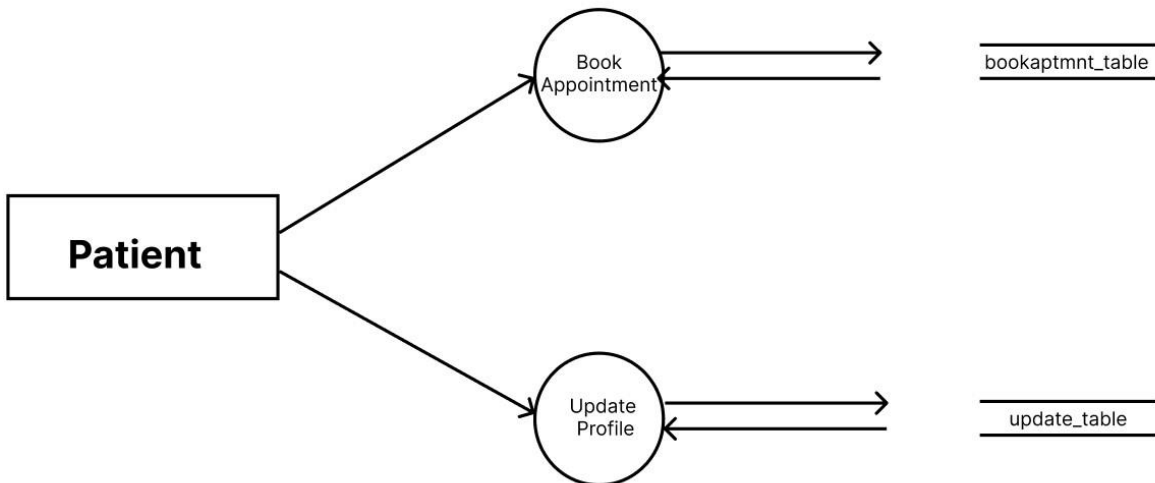
III. DFD Level 2.2



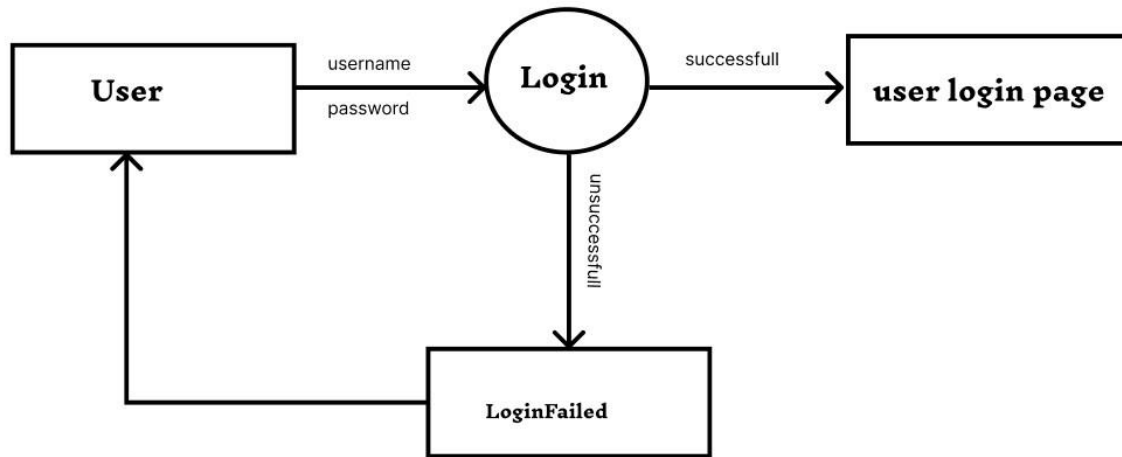
IV. DFD Level 3.1



V. DFD Level 4.1



VI. DFD Level 5.1



Chapter – 5

Database Design

5.1 Introduction:

Database Design maintains the data required by the System One of the key design issues involved in the database to design is the distribution of data in a way that minimizes transaction traffic. Another key design issue is the choice of the database management system. Database tables used are described in the following sections.

5.2 Schema Representation

Admin Table

First Name	Data Type	Constraints	Description
Id	int(11)	Primary key	id
username	varchar(100)	Not null	Name
password	varchar(100)	Not null	Login password
updatetime	varchar(100)	Not null	Update date

Doctor Table

Field Name	Data Type	Constraints	Description
Id	Int(10)	Primary key	id
specialization	Varchar(100)	Not null	specialization
doctorname	Varchar(100)	Not null	name
Address	longtext	Not null	address
Docfees	Varchar(100)	Not null	fees
contactno	Bigint(10)	Not null	number
docEmail	Varchar(100)	Not null	email
Password	Varchar(100)	Not null	Login password
creationdate	timestamp	Current_timestamp()	Create a date
updatetime	timestamp	Not null	Update a date

Patient Table

Field Name	Data Type	Constraints	Description
Id	Int	Primary key	id
Docid	Int	Not null	doctorid
Patient name	varchar(100)	Not null	name
Contactno	bigint(10)	Not null	Contact number
Email	varchar(100)	Not null	Email
Gender	varchar(100)	Not null	Gender
Age	Int	Not null	Age
creationdate	timestamp	Current_timestamp()	Created date
updatedate	timestamp	Not null	Updated date

Feedback table

Field Name	Data Type	Constraints	Description
Name	Varchar(100)	Not null	name
Email	Varchar(100)	Not null	email
Contactno	bigint(10)	Not null	Contact number
Message	varchar(100)	Not null	message

Appointment Table

Field Name	Data Type	Constraints	Description
Id	int	Primary key	id
doctorspecialization	Varchar(100)	Not null	specialization
doctorname	varchar(100)	Not null	name
Fees	int(10)	Not null	Fees
appointmentdate	varchar(100)	Not null	Appointment date
Appointmenttime	varchar(100)	Not null	Appointment time
userStatus	int(10)	Not null	User Status

doctorstatus	t int(10)	Not Null	Doctor Status
updatedate	timestamp	Not null	Updated date

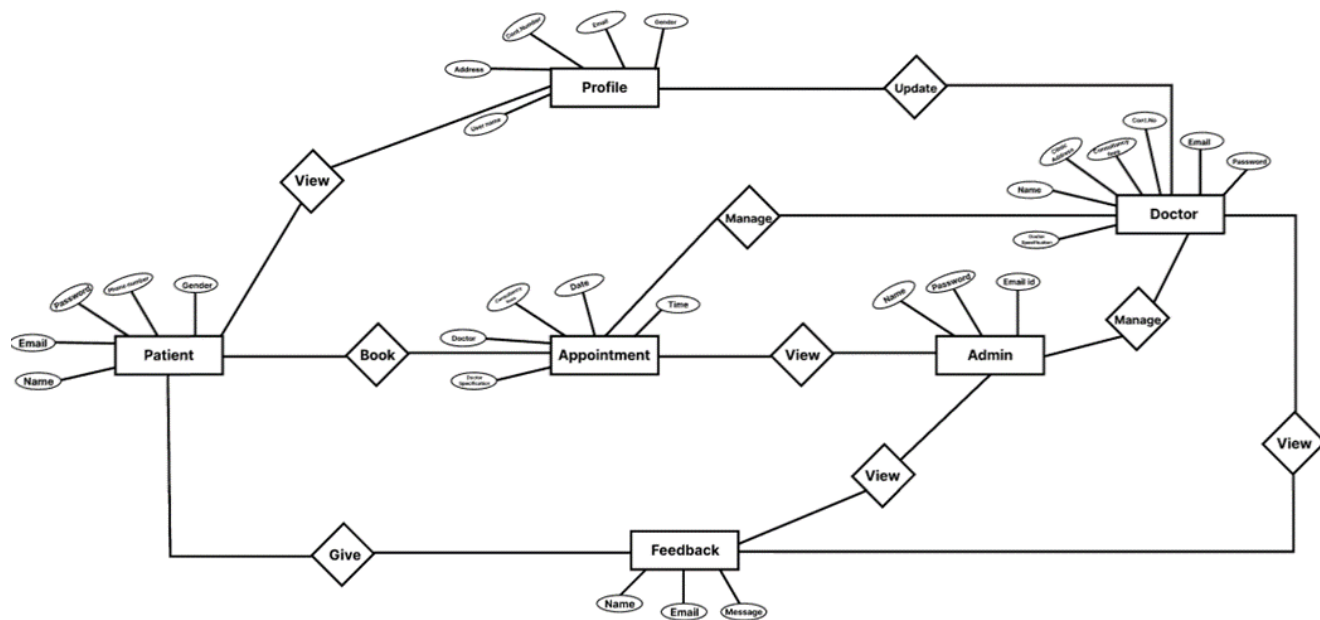
Doctor specialization table

Field Name	Data Type	Constraints	Description
Id	int	Primary key	id
specialization	Varchar(255)	Not null	specialization
createDate	timestamp	current_timestamp()	Create date
updateDate	timestamp	Not null	Update date

Login

Field Name	Data Type	Constraints	Description
Id	int(11)	Primary key	id
Username	Varchar(255)	Not null	User name
Login time	timestamp	current_timestamp()	Login time
Logout	timestamp	current_timestamp()	Logout time
Status	int(11)	Not null	status

5.3 ER(Entity Relationship) Diagram



Chapter – 6

Detailed Design

6.1 Introduction

The detailed design will decide the internal logic for the module ,which implements the given specification. Detailed design is one of the design processes for software items. Detailed design, which is known as login design ,involves the internal sign of the module and how the specification of the module can be satisfied. This phase include the specification of the table structure, which makes us understand the internal logic of the system . The table are described with their field names, data type, table constraints, description of each field etc...

6.2 Structured English

ADMIN LOGIN

BEGIN

IF (click on Admin Login) then Check the login and password of the
admin exists then redirect to the home page

ELSE

Display a message “failed to login”

END IF

END

USER REGISTRATION FORM

BEGIN

IF(click on submit) Then

The user information from the form is inserted into the
database and a success message is displayed END IF

END

PATIENT LOGIN FORM

BEGIN

IF (click on User Login) then Check the login and password of the user exists then redirect to the home page

ELSE

Display a message “failed to login”

END IF

END

DOCTOR LOGIN FORM

BEGIN

IF (click on Employee Login) then Check the login and password of the doctor exists then redirect to the home page

ELSE

Display a message “failed to login”

END IF

END

APPOINTMENT FORM

BEGIN

IF(click on submit) Then

The appointment information from the form is inserted into the database and a success message is displayed END IF

END

ADD DOCTOR FORM

BEGIN

IF(click on submit) Then

The employee information from the form are inserted into the database and a success message is displayed

END IF

END

Chapter – 7

System Coding

7.1 Introduction:

The main goal of the coding of programming phase is to translate the design of the system produced during the design phase into code in a given programmatically language. It is then executed on a mobile or emulator to verify whether the design is correct or not. In coding phase, the output document is code can be extremely useful in enhancing the understandability. Internal documentation of code is done using comments in the program. Comments are textual statements that are meant for the program reader and are not executed.

The coding phase affect both testing and maintenance phases.

Well written code can reduce the testing and maintenance effort.

Book-appointment.php

```
<?php
session_start(); //error_reporting(0);
include('include/config.php');
include('include/checklogin.php');
check_login();

if(isset($_POST['submit']))
{
    $specilization=$_POST['Doctorspecialization'];
    $doctorid=$_POST['doctor'];
    $userid=$_SESSION['id'];
    $fees=$_POST['fees'];
    $appdate=$_POST['appdate'];
    $time=$_POST['apptime'];
    $userstatus=1;
    $docstatus=1;
    $query=mysqli_query($con,"insert into
    appointment(doctorSpecialization,doctorId,userId,consultancyFees,appointmentDate,appointment
    Time,userStatus,doctorStatus)
    values('$specilization','$doctorid','$userid','$fees','$appdate','$time','$userstatus','$docstatus')");
    if($query)
    {
        echo "<script>alert('Your appointment successfully booked');</script>";
    }
}
```

```

    }

}
?>
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>User | Book Appointment</title>

        <link
href="http://fonts.googleapis.com/css?family=Lato:300,400,400italic,600,700|Raleway:300,400,500,600,700|Crete+Round:400italic" rel="stylesheet" type="text/css" />
        <link rel="stylesheet" href="vendor/bootstrap/css/bootstrap.min.css">
        <link rel="stylesheet" href="vendor/fontawesome/css/font-awesome.min.css">
        <link rel="stylesheet" href="vendor/themify-icons/themify-icons.min.css">
        <link href="vendor/animate.css/animate.min.css" rel="stylesheet" media="screen">
        <link href="vendor/perfect-scrollbar/perfect-scrollbar.min.css" rel="stylesheet" media="screen">
        <link href="vendor/switchery/switchery.min.css" rel="stylesheet" media="screen">
        <link href="vendor/bootstrap-touchspin/jquery.bootstrap-touchspin.min.css" rel="stylesheet"
media="screen">
        <link href="vendor/select2/select2.min.css" rel="stylesheet" media="screen">
        <link href="vendor/bootstrap-datepicker/bootstrap-
datepicker3.standalone.min.css" rel="stylesheet" media="screen">
        <link href="vendor/bootstrap-timepicker/bootstrap-timepicker.min.css" rel="stylesheet"
media="screen">
        <link rel="stylesheet" href="assets/css/styles.css">
        <link rel="stylesheet" href="assets/css/plugins.css">
        <link rel="stylesheet" href="assets/css/themes/theme-1.css" id="skin_color" />
        <script>
function getdoctor(val) {
    $.ajax({        type: "POST",
    url: "get_doctor.php",
    data:'specilizationid='+val,
    success: function(data){
        $("#doctor").html(data);
    }
    });
}
</script>

```

```

<script> function
getfee(val) {
    $.ajax({      type:
"POST", url:
"get_doctor.php",
    data:'doctor='+val,
        success: function(data){
            $("#fees").html(data);
        }
    });
}
</script>

```

```

</head>
<body>
    <div id="app">
<?php include('include/sidebar.php');?>
        <div class="app-content">

```

```

            <?php include('include/header.php');?>

```

```

            <!-- end: TOP NAVBAR -->

```

```

            <div class="main-content" >

```

```

                <div class="wrap-content container" id="container">

```

```

                    <!-- start: PAGE TITLE -->

```

```

                    <section id="page-title">

```

```

                        <div class="row">

```

```

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

```

```

                                <h1 class="mainTitle">User |

```

```

                                Book Appointment</h1>

```

```

                            </div>

```

```

                        <ol class="breadcrumb">

```

```

                            <li>

```

```

                                <span>User</span>

```

```

                            </li>

```

```

                            <li class="active">

```

```

                                <span>Book

```

```

                                Appointment</span>

```

```

                            </li>

```

```

                        </ol>

```

```

</section>
<!-- end: PAGE TITLE -->
<!-- start: BASIC EXAMPLE -->
<div class="container-fluid container-fullw bg-white">
    <div class="row">
        <div class="col-md-12">

            <div class="row margin-top-
30">

                <div class="col-lg-8 colmd-
12">

                    <div class="panel
panel-white">

                        <div class="panel-heading">

                            <h5 class="panel-title">Book Appointment</h5>

                                </div>
                                <div
class="panel-body">

                                    <p style="color:red;"><?php echo
htmlentities($_SESSION['msg1']);?>

                                        <?php echo
htmlentities($_SESSION['msg1']="");?></p>

                                    <form role="form" name="book" method="post" >

                                        <div class="form-group">

                                            <label for="DoctorSpecialization">

                                                Doctor Specialization

                                            </label>

                                                <select name="Doctorspecialization"
class="form-control" onChange="getdoctor(this.value);" required="required">

                                                    <option value="">Select Specialization</option>
<?php $ret=mysqli_query($con,"select * from doctorspecilization");
while($row=mysqli_fetch_array($ret))

```



```
{  
?>
```

```
<option value="<?php echo htmlentities($row['specilization']);?>">
```

```
<?php echo htmlentities($row['specilization']);?>
```

```
</option>
```

```
<?php } ?>
```

```
</select>
```

```
</div>
```

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

```
<label for="doctor">
```

```
Doctors
```

```
</label>
```

```
<select name="doctor" class="form-control" id="doctor" onChange="getfee(this.value);" required="required">
```

```
<option value="">Select
```

```
Doctor</option>
```

```
</select>
```

```
</div>
```

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

<label for="consultancyfees">

Consultancy Fees

</label>

<select name="fees" class="form-control" id="fees" readonly>

</select>

</div>

<div class="form-group">

<label for="AppointmentDate">

Date

</label>

<input class="form-control datepicker" name="appdate" required="required" data-date-format="yyyy-mm-dd">

</div>

<div class="form-group">

<label for="Appointmenttime">

Time

</label>

<input class="form-control" name="apptime" id="timepicker1" required="required">eg : 10:00 PM

</div>

```
primary">
    <button type="submit" name="submit" class="btn btn-o btn-
```

```
        Submit
    </button>
```

```
</form>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
<!-- end: BASIC EXAMPLE -->
```

```
<!-- end: SELECT BOXES -->
```

```
</div>
```

```
</div>
```

```
</div>
```

```
<!-- start: FOOTER -->
```

```
<?php include('include/footer.php');?>
```

```
<!-- end: FOOTER -->
```

```
<!-- start: SETTINGS -->
```

```
<?php include('include/setting.php');?>
```

```
<!-- end: SETTINGS -->
```

```
</div>
```

```

        <!-- start: MAIN JAVASCRIPTS -->
        <script src="vendor/jquery/jquery.min.js"></script>
        <script src="vendor/bootstrap/js/bootstrap.min.js"></script> <script
src="vendor/modernizr/modernizr.js"></script>
        <script src="vendor/jquery-cookie/jquery.cookie.js"></script>
        <script src="vendor/perfect-scrollbar/perfect-
scrollbar.min.js"></script>
        <script src="vendor/switchery/switchery.min.js"></script>
        <!-- end: MAIN JAVASCRIPTS -->
        <!-- start: JAVASCRIPTS REQUIRED FOR THIS PAGE ONLY -->
        <script
src="vendor/maskedinput/jquery.maskedinput.min.js"></script>
        <script src="vendor/bootstrap-touchspin/jquery.bootstrap-touchspin.min.js"></script>
        <script src="vendor/autosize/autosize.min.js"></script>
        <script src="vendor/selectFx/classie.js"></script>
        <script src="vendor/selectFx/selectFx.js"></script>
        <script src="vendor/select2/select2.min.js"></script>
        <script src="vendor/bootstrap-datepicker/bootstrap-
datepicker.min.js"></script>
        <script src="vendor/bootstrap-timepicker/bootstrap-
timepicker.min.js"></script>
        <!-- end: JAVASCRIPTS REQUIRED FOR THIS PAGE ONLY -->
        <!-- start: CLIP-TWO JAVASCRIPTS -->
        <script src="assets/js/main.js"></script>
        <!-- start: JavaScript Event Handlers for this page -->
        <script src="assets/js/form-elements.js"></script>
        <script>
            jQuery(document).ready(function() {
                Main.init();
                FormElements.init();
            });

            $('#datepicker').datepicker({
format: 'yyyy-mm-dd',
startDate: '-3d'
});
        </script>
        <script type="text/javascript">
            $('#timepicker1').timepicker();
        </script>
        <!-- end: JavaScript Event Handlers for this page -->

```

```
<!-- end: CLIP-TWO JAVASCRIPTS -->
```

```
<script  
src="http://ajax.googleapis.com/ajax/libs/jquery/1.8.0/jquery.min.js"></script>
```

```
</body>
```

```
</html>
```

Chapter – 8

TESTING

8.1 Introduction:

Testing is the phase where the errors remaining from the earlier phases also must be detected. It is the process of examining something with the intension of finding errors.

Testing a program consists of providing program with a set of test inputs and observing if the program behaves as expected. If the program fails, then the conditions under which a failure occurs are noted for debugging and correction.

8.2 Psychology of testing

The aim of testing is often to demonstrate that program works by showing that it has no errors. The basic purpose testing phase is to detect the errors that may be present in the programs, hence one should not start testing with the internal to show that a program does not work.

8.3 Levels of testing

Before the implementation of the system, tesing has been carried out thoroughly to eliminate any bugs, Which may be present? The test has been conducted based on some levels of testing.

- * Unit testing
- *Integration testing
- *Output testing
- *Validation testing
- * User acceptance testing

8.3.1 Unit testing

Testing at the unit level is very much essential because the error is accurate. Unit testing is testing of different units or modules of a system in isolation. Testing is done to check whether each module in the software works properly so that it gives desired outputs to the given inputs.

8.3.2Integration testing

In Integration testing all the code modules are put together and tested for desired outputs. The modules unit tested are integrated and tested. All the modules are combined in this testing step. Then the entire program is tested as a whole. The integration testing is carried out using integrated test plans prepared in the design of the system development as a guide. All the errors found in the system are corrected for the next testing steps. The modules which are tested show expected results.

8.3.3Output testing

Feeding sample valid input image and then comparing the ratio obtained in the compressed output image with the expected ratio to conduct it. The correctness of the output depends on the inputted image.

8.3.4Validation testing:

At the culmination of the integration testing, the software was completely assembled as the package. Here we test the system in a manner that can be reasonably expected by the customer. The system was tested against system requirements specification.

8.3.5User acceptance testing:

Acceptance testing is performed with realistic data of the client to demonstrate that the software is working satisfactorily. Testing here is focused on the external behavior of the system. The internal logic of the program is not emphasized. Test cases should be selected so that the largest number of attributes of an equivalence class is exercised.

8.4 Testing:

Patient registration form

Sl.no	Condition to be tested	Tested data	Expected output	Remarks
1	If any field in form is empty	Value of form fields	Alter the user to enter all fields and then process	<u>successful</u>
2	If patient name field is empty	Patient name	Alter the user to enter all fields and then process	<u>successful</u>
3	If patient address field is empty	address	Alter the user to enter all fields and then process	<u>successful</u>
4	If patient city field is empty	City name	Alter the user to enter all fields and then process	<u>successful</u>
6	If gender field is empty	Gender name	Alter the user to enter all fields and then process	<u>successful</u>

7	If patient email field is empty	Email address	Alter the user to enter all fields and them process	<u>successfull</u>
8	If patient email is invalid	Email address	Please include an '@' in the email address	<u>successfull</u>
9	If password field is empty	password	Alter the user to enter all fields and them process	<u>successfull</u>

Patient login page

Sl.no	Condition to be tested	Tested data	Excepted output	Remarks
1	If any field inform is empty	Vale of form fields	Alter the user to enter all fields and them process	<u>successfull</u>
2	If email address and password are not valid	Email address and password	Invalid username and password	<u>successfull</u>
3	If all fields are entered	All mandatory fields	Form is directed to home page	<u>successfull</u>

Patient update profile

Sl.no	Condition to be tested	Tested data	Excepted output	Remarks
1	If any field inform is empty	Vale of form fields	Alter the user to enter all fields and them process	<u>successfull</u>
2	If patient name field is empty	Patient name	Alter the user to enter all fields and them process	<u>successfull</u>
3	If patient address field is empty	address	Alter the user to enter all fields and them process	<u>successfull</u>
4	If patient city field is empty	City name	Alter the user to enter all fields and them process	<u>successfull</u>
6	If gender field is empty	Gender name	Alter the user to enter all fields and them process	<u>successfull</u>
7	If patient email field is empty	Email address	Alter the user to enter all fields and them process	<u>successfull</u>
8	If patient email is invalid	Email address	Please include an '@' in the email address	<u>successfull</u>
9	If password field is empty	password	Alter the user to enter all fields and them process	<u>successfull</u>

Admin login form

Sl.no	Condition to be tested	Tested data	Excepted output	Remarks
-------	------------------------	-------------	-----------------	---------

1	If any field inform is empty	Vale of form fields	Alter the user to enter all fields and them process	<u>successfull</u>
2	If email address and password are not valid	Email address and password	Invalid username and password	<u>successfull</u>
3	If all fields are entered	All mandatory fields	Form is directed to home page	<u>successfull</u>

Doctor login form

Sl.no	Condition to be tested	Tested data	Excepted output	Remarks
1	If any field inform is empty	Vale of form fields	Alter the user to enter all fields and them process	<u>successfull</u>
2	If email address and password are not valid	Email address and password	Invalid username and password	<u>successfull</u>
3	If all fields are entered	All mandatory fields	Form is directed to home page	<u>successfull</u>

Doctor update profile

Sl.no	Condition to be tested	Tested data	Excepted output	Remarks
1	If any field inform is empty	Vale of form fields	Alter the user to enter all fields and them process	<u>successfull</u>
2	If doctor specialization is empty	specialization	Alter the user to enter all fields and them process	<u>successfull</u>
2	If doctor name field is empty	Doctor name	Alter the user to enter all fields and them process	<u>successfull</u>
3	If doctor address field is empty	address	Alter the user to enter all fields and them process	<u>successfull</u>
4	If doctor contact field is empty	Contact number	Alter the user to enter all fields and them process	<u>successfull</u>
6	If doctor email is invalid	Email address	Please include an '@' in the email address	<u>successfull</u>
7	If doctor email field is empty	Email address	Alter the user to enter all fields and them process	<u>successfull</u>

Doctor change password

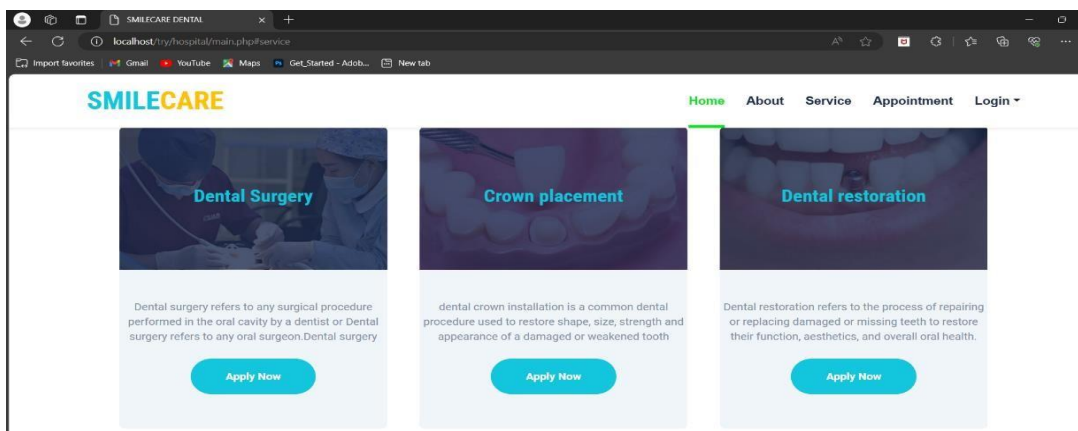
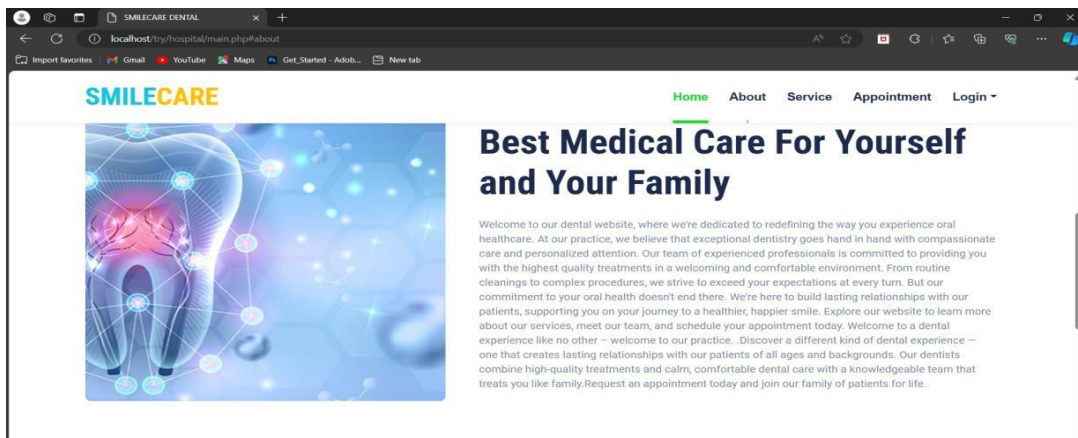
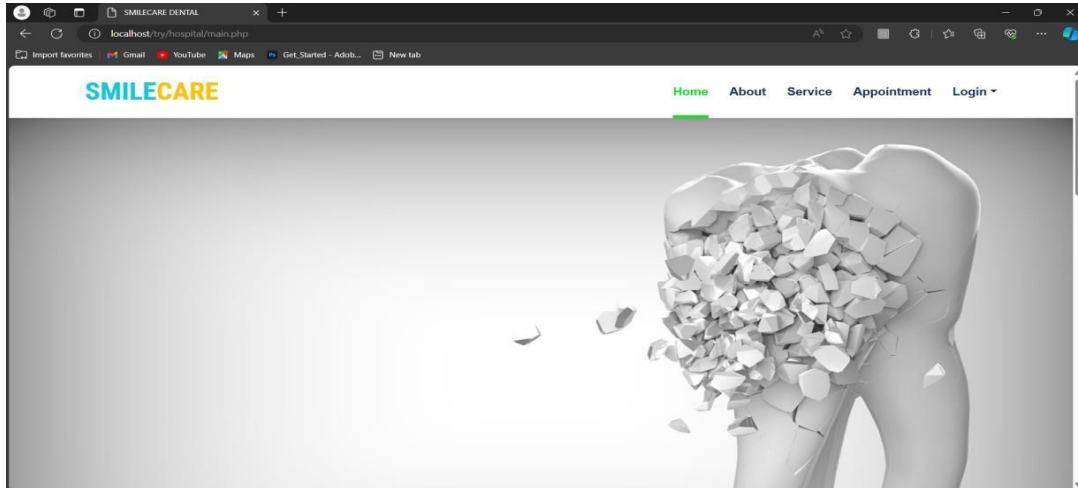
Sl.no	Condition to be tested	Tested data	Excepted output	Remarks
-------	------------------------	-------------	-----------------	---------

1	If any field inform is empty	Vale of form fields	Alter the user to enter all fields and them process	<u>successfull</u>
2	If password field is empty	password	Invalid username and password	<u>successfull</u>
3	If password andconfirm password does not match	Password and confirm password	Alter the user to enter all fields by displaying”please fill the field”	<u>successfull</u>
4	If all fields are entered	All mandatory fields	Form is directed to home page	<u>successfull</u>

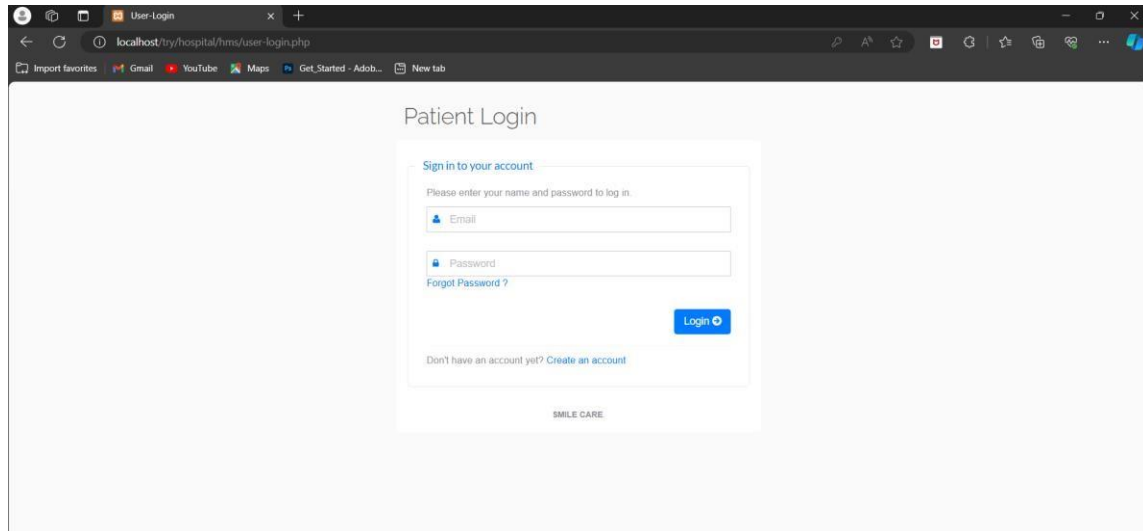
Chapter – 9

Snapshots

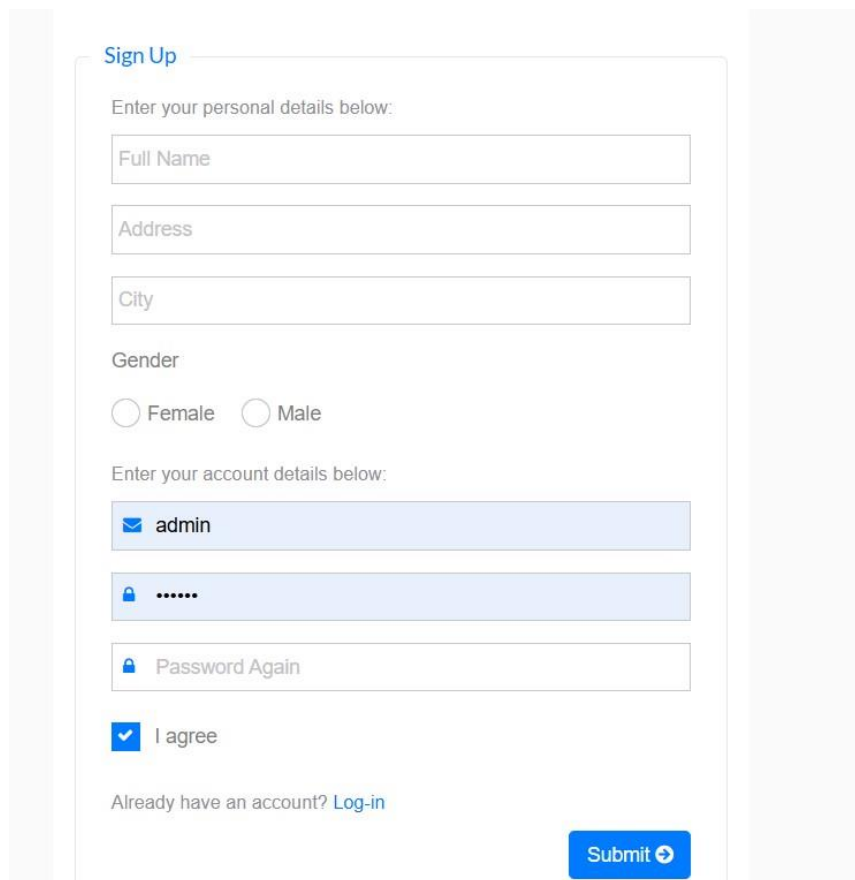
HOME PAGE



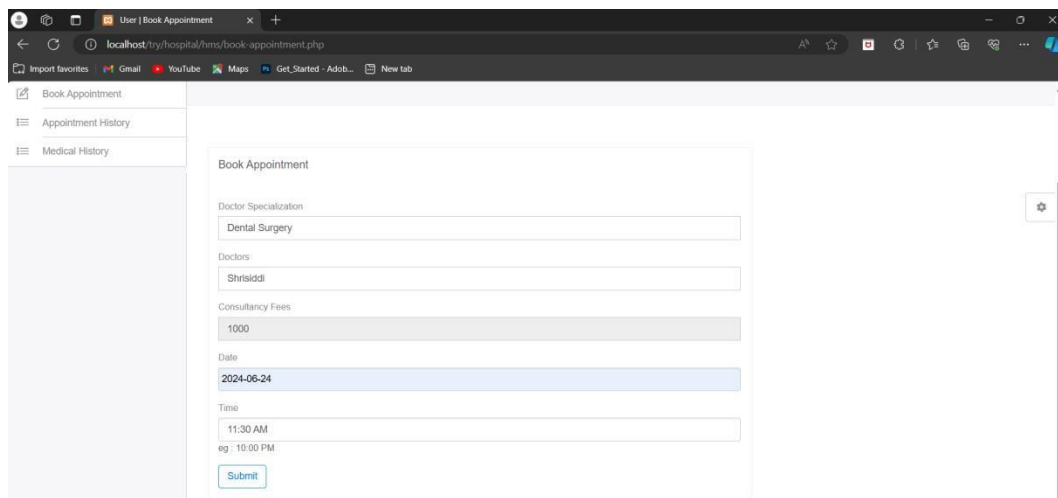
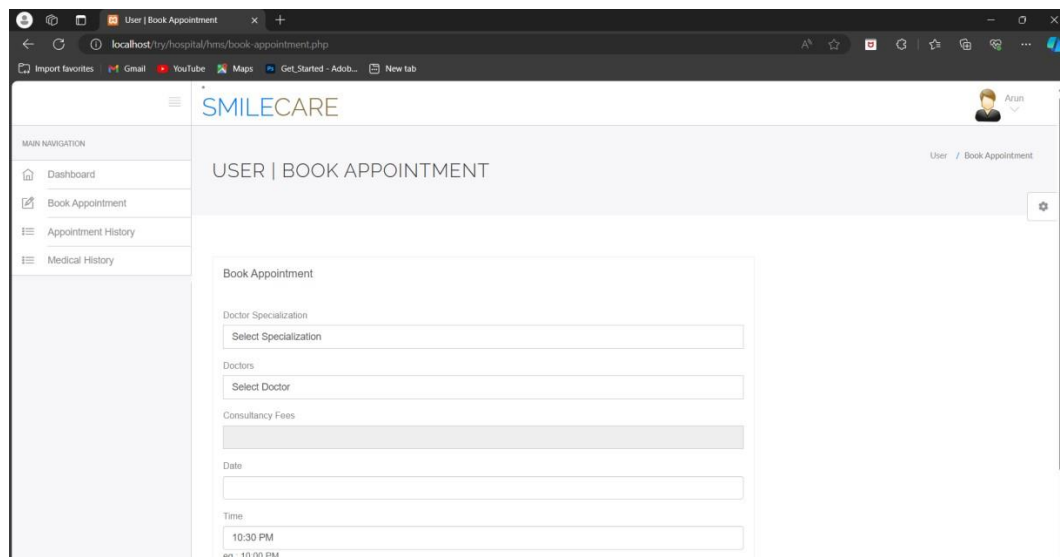
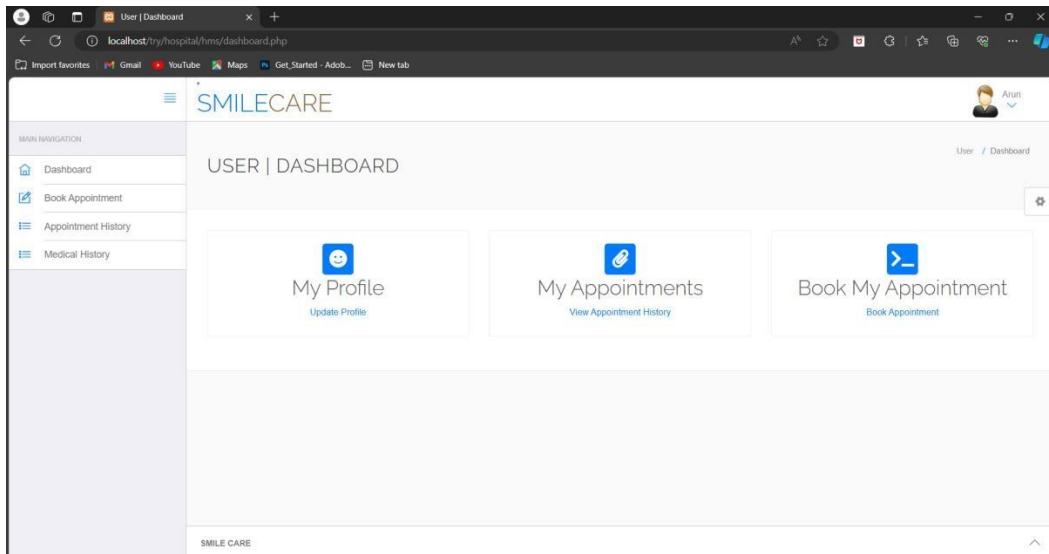
USER LOGIN AND APPOINTMENT:

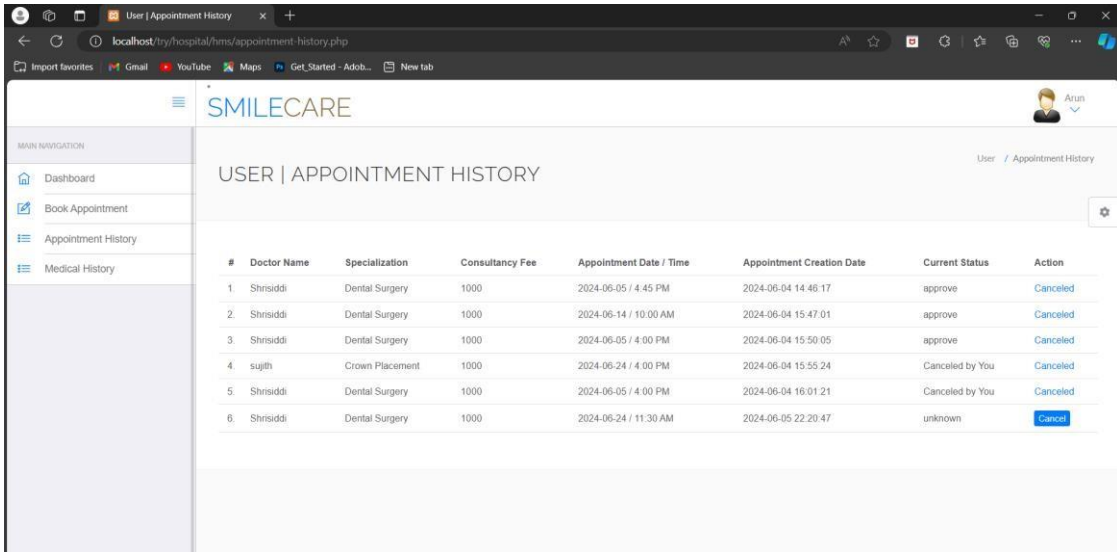
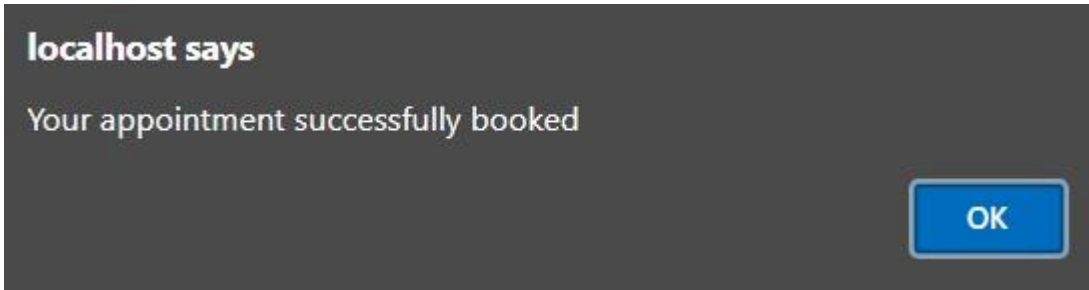


A screenshot of a web browser displaying a "Patient Login" form. The browser's address bar shows "localhost:try/hospital/fms/user-login.php". The form is titled "Patient Login" and contains a sub-header "Sign in to your account". Below this, it says "Please enter your name and password to log in:". There are two input fields: "Email" and "Password". A "Forgot Password?" link is located below the password field. A blue "Login" button is positioned to the right of the password field. At the bottom of the form, it says "Don't have an account yet? Create an account". The text "SMILE CARE" is visible at the very bottom of the page.

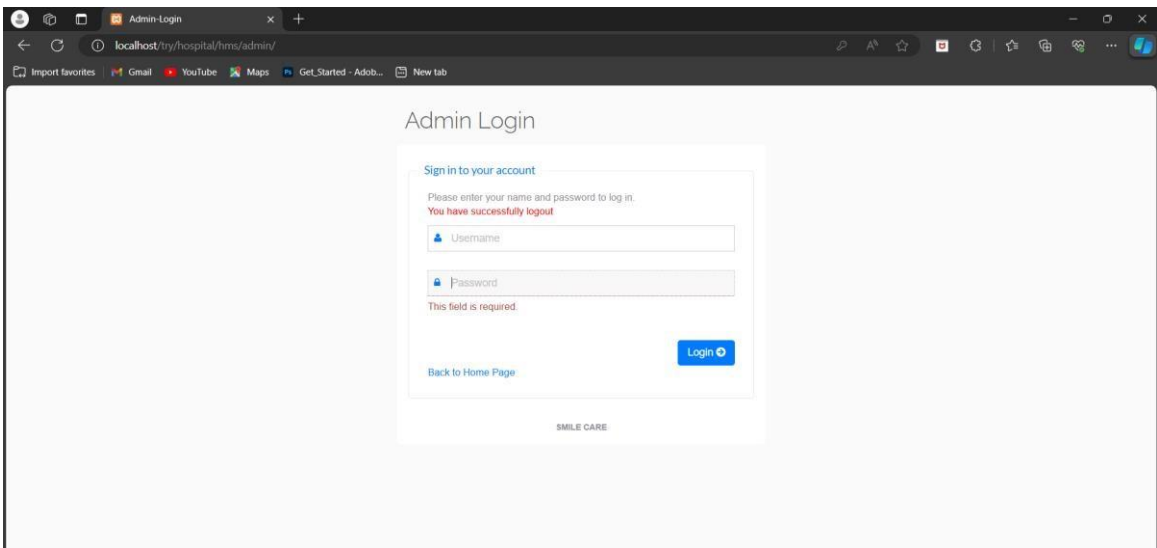


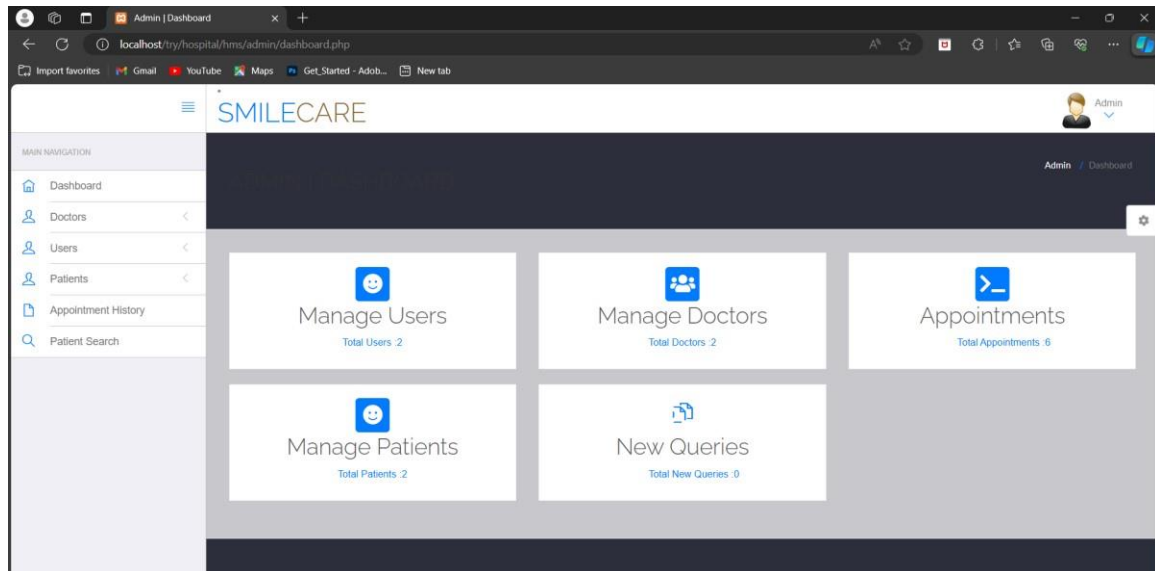
A screenshot of a "Sign Up" form. The form is titled "Sign Up" and contains the instruction "Enter your personal details below:". There are three input fields: "Full Name", "Address", and "City". Below these is a "Gender" section with two radio buttons: "Female" and "Male". The next section is "Enter your account details below:". It contains three input fields: "Email" (with the value "admin"), "Password" (with masked characters "*****"), and "Password Again". Below the password fields is a checkbox labeled "I agree". At the bottom, it says "Already have an account? Log-in". A blue "Submit" button is located at the bottom right of the form.



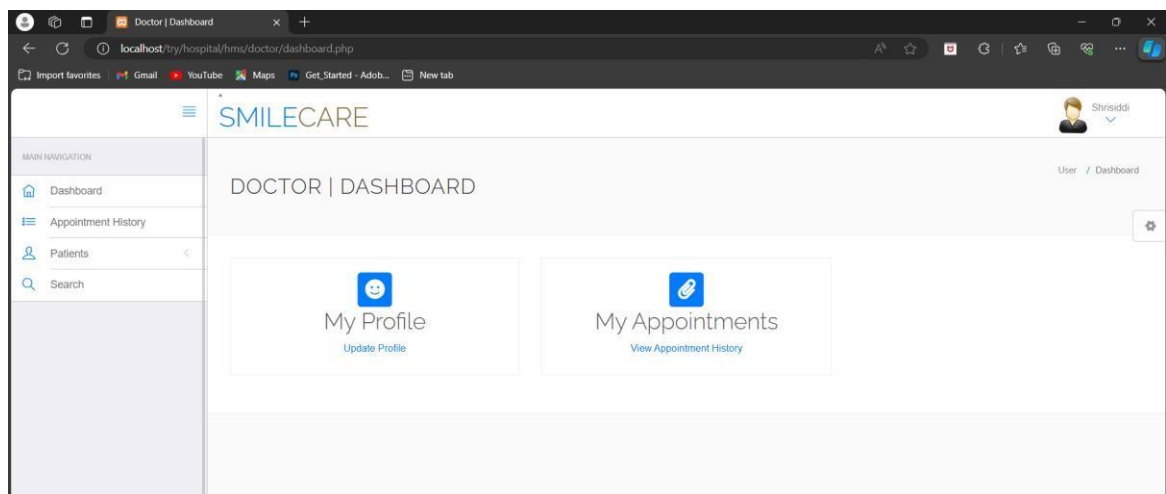
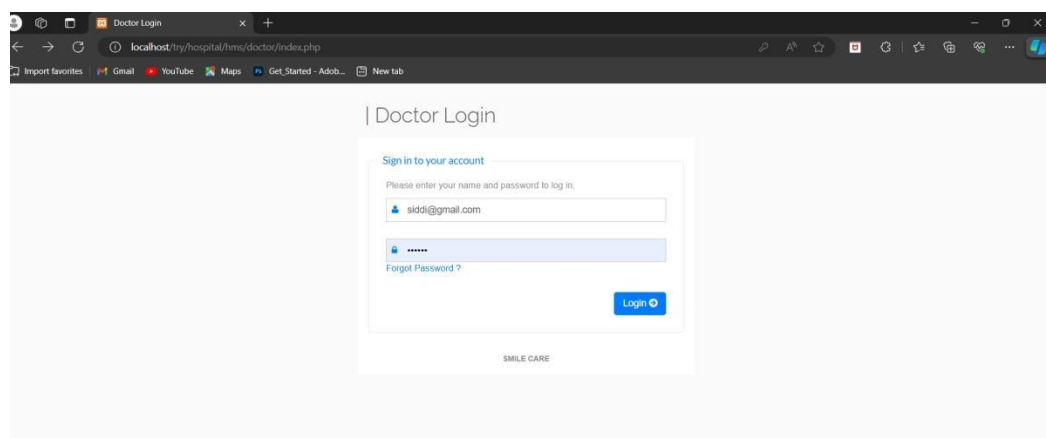


ADMIN





Doctor



Chapter – 10

Future Enhancement

- Integration of telemedicine features for virtual consultations.
- Implementation of an electronic health record (EHR) system.
- Expansion to include additional features like prescription management.

Chapter – 11

Conclusion

In conclusion, the development of the SmileCare Dental Management System has been aimed at enhancing the efficiency and effectiveness of dental care services. This system was designed to address the needs of dental practices by providing robust features for appointment scheduling, patient management, billing, and administrative tasks.

Chapter – 12

Bibliography

12.1 Book references:

- Software engineering by Pankaj Jalote.
- HTML and CSS made simple by Ivan Bayross.
- Database Management System by Sudarshan Prabhu

12.2 Web reference:

- www.w3schools.com
- www.geeksforgeeks.org