

### III

## ABSTRACT

The Doctor Appointment System is a web-based application designed to streamline the process of booking appointments with doctors. Traditional methods of booking appointments often involve making phone calls or in-person visits, which can be time-consuming and inconvenient for both patients and doctors. This project aims to address this challenge by providing a centralized platform that allows patients to easily book appointments with doctors using a web-based interface.

The key objectives of the Doctor Appointment System are to improve the overall healthcare experience by enhancing communication and coordination between patients and doctors, reducing waiting times, and increasing the efficiency of the appointment scheduling process. The system is designed with a server-side architecture, where the backend is implemented using PHP as the programming language and MySQL as the database management system.

The system includes features such as patient and doctor registration, appointment booking and management, a notification system, and secure data storage. Patients can create an account, view doctor profiles, and book appointments, while doctors can manage their schedules, update their profiles, and receive notifications about upcoming appointments. The system also implements security measures, such as password hashing, input validation, and HTTPS communication, to protect the confidentiality and integrity of user data.

Since the system is built using only PHP and MySQL, the user interface is primarily handled on the server-side, with the frontend focusing on dynamic content generation and data presentation. This approach ensures a seamless and responsive user experience while leveraging the strengths of these server-side technologies.

Initial testing and user feedback have been positive, indicating that the Doctor Appointment System is meeting the specified objectives and providing a streamlined healthcare experience for both patients and doctors. The system's scalable architecture and robust features make it a promising solution for improving the management of doctor appointments in the healthcare industry.

## IV

### CONTENTS

I Certificate

II Certificate by Guide

III Acknowledgement

IV Abstract

<b>1. Introduction.....</b>	<b>8 to 9</b>
<b>1.1 Objectives .....</b>	<b>8</b>
<b>1.2 Problem Statement .....</b>	<b>9</b>
<b>1.3 Project Scope .....</b>	<b>9</b>
<b>2. System Architecture.....</b>	<b>10 to 11</b>
<b>3. System Requirement.....</b>	<b>12</b>
Hardware Requirements.....	12
Software Requirements .....	12
<b>4. Data flow Diagram .....</b>	<b>13</b>
<b>5. Coding.....</b>	<b>14 to 16</b>
<b>6. Result and Discussion .....</b>	<b>17 to 20</b>
<b>7. Advantages and Disadvantages .....</b>	<b>21</b>
Conclusion .....	22
References .....	23

### **LIST OF FIGURES**

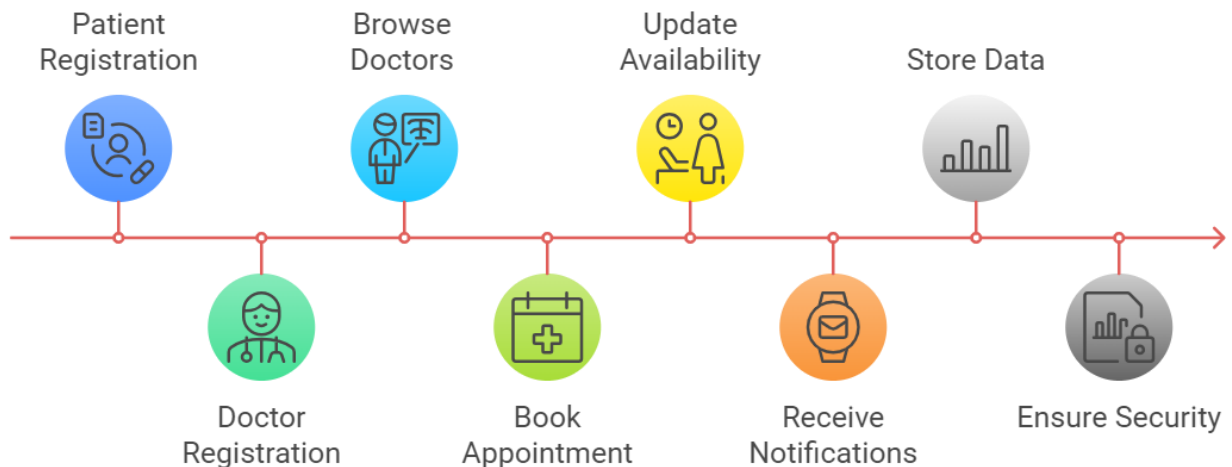
<b>SR. NO.</b>	<b>NAME OF FIGURES</b>	<b>PAGE NO:</b>
1	System Architecture	11
2	Data Flow and Diagram	13

# 1. INTRODUCTION

The Doctor Appointment System is a web-based application developed using PHP as the server-side programming language and MySQL as the database management system. This system is designed to simplify the process of booking appointments with doctors for patients, as well as manage the doctors' schedules and patient information. Key features of the system include patient registration and login, doctor registration, appointment booking, appointment management, and a notification system. Patients can create an account, browse available doctors, and book appointments that are convenient for them, with the details stored in the MySQL database. Doctors can register their profiles, update their availability, and receive notifications about upcoming appointments.

The system ensures the confidentiality and security of patient and doctor data stored in the MySQL database. By leveraging the power of PHP and MySQL, the Doctor Appointment System aims to create a convenient and streamlined healthcare experience for both patients and doctors, improving appointment scheduling, enhancing patient-doctor communication, reducing waiting times, and increasing overall efficiency in the medical practice.

Doctor Appointment System Process



## 1.1 Objectives

### 1. Convenient Appointment Booking for Patients:

- Provide a user-friendly and accessible platform for patients to easily book appointments with doctors.
- Allow patients to view the availability of doctors and select the most convenient appointment slots.

### 2. Efficient Schedule Management for Doctors:

- Enable doctors to manage their schedules and availability through a dedicated interface.

- Allow doctors to receive notifications about upcoming appointments and handle any changes or cancellations.

### 3. Improved Patient-Doctor Communication:

- Facilitate better communication and coordination between patients and doctors.
- Ensure that both parties are informed about appointment details, changes, and any other relevant information.

### 4. Reduced Waiting Times:

- Streamline the appointment booking process to minimize waiting times for patients.
- Optimize the scheduling and management of appointments to improve the overall efficiency of the healthcare system.

### 5. Secure Data Management:

- Ensure the confidentiality and security of patient and doctor data stored in the MySQL database.
- Implement appropriate measures, such as password hashing and input validation, to protect the system from common web application vulnerabilities.

## **1.2 Problem Statement**

The traditional method of booking doctor appointments, which often involves making phone calls or in-person visits, can be time-consuming and inconvenient for both patients and doctors. There is a need for a centralized, web-based system that can efficiently manage the scheduling of appointments and improve communication between patients and doctors.

## **1.3 Project Scope**

1. Patient registration and login
2. Doctor registration and profile management
3. Appointment booking and management (for both patients and doctors)
4. Notification system for upcoming appointments and appointment changes
5. Secure data storage and management

## 2. SYSTEM ARCHITECTURE

The Doctor Appointment System is designed with a traditional web application architecture, featuring a client-side (frontend) and a server-side (backend) component, connected through HTTP requests and responses.

### Client-side (Frontend):

- User Interface: The user interface is built using HTML, CSS, and JavaScript, providing a responsive and interactive experience for both patients and doctors.
- User Interactions: The frontend handles user interactions, such as registration, login, appointment booking, and management, by sending HTTP requests to the backend.
- Data Presentation: The frontend receives data from the backend (e.g., doctor profiles, appointment details) and presents it to the user in a user-friendly manner.

### Server-side (Backend):

- PHP Application: The backend is built using the PHP programming language, which handles the server-side logic and processes the HTTP requests from the client.
- MySQL Database: The MySQL database is used to store and manage all the data related to patients, doctors, and appointments.
- API Endpoints: The PHP application exposes a set of API endpoints that the frontend can use to interact with the system, such as fetching doctor profiles, booking appointments, and updating user information.
- Business Logic: The PHP code implements the business logic of the Doctor Appointment System, including user authentication, appointment scheduling, and notification handling.
- Database Interactions: The PHP application communicates with the MySQL database to perform CRUD (Create, Read, Update, Delete) operations on the data.

### Communication Flow:

1. The user (patient or doctor) interacts with the frontend user interface, triggering an HTTP request to the backend.
2. The PHP application receives the request, processes the data, and performs the necessary operations (e.g., querying the database, updating records).
3. The PHP application then generates a response, which is sent back to the frontend.
4. The frontend receives the response, processes the data, and updates the user interface accordingly.

### Security Considerations:

- Authentication and Authorization: The system implements user authentication (login/logout) and role-based access control to ensure that only authorized users can access and perform specific actions.
- Input Validation and Sanitization: The backend PHP code validates and sanitizes all user input to prevent common web application vulnerabilities, such as SQL injection and cross-site scripting (XSS) attacks.
- Data Encryption: Sensitive data, such as user passwords, is stored in the MySQL database using secure hashing and encryption techniques.
- Network Security: The communication between the frontend and backend is secured using HTTPS to protect against eavesdropping and man-in-the-middle attacks.

This system architecture provides a scalable and secure foundation for the Doctor Appointment System, leveraging the strengths of PHP and MySQL to deliver a robust and user-friendly healthcare management application.

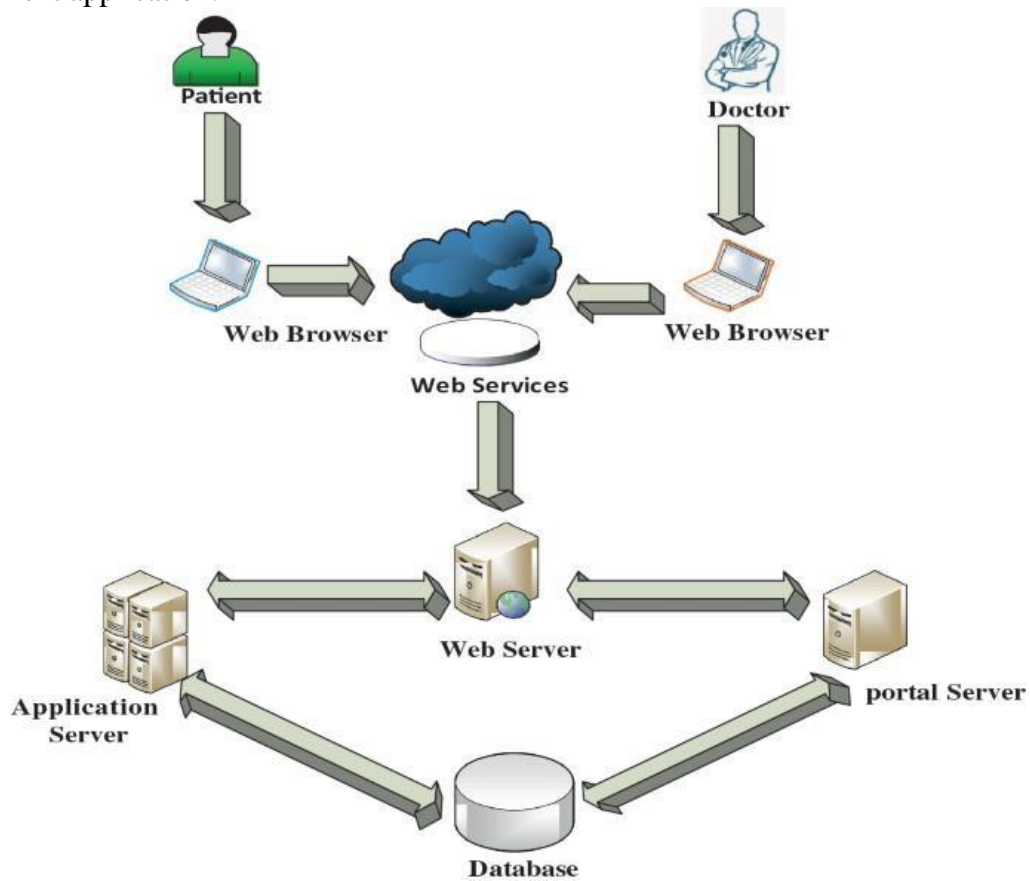


Fig a: System Architecture.

### 3. SYSTEM REQUIREMENT

#### **Software Requirements:**

1. Operating System: The system should be compatible with popular operating systems, such as Windows, macOS, and Linux.
2. XAMPP Server: The system requires the XAMPP server, which includes Apache, MySQL, PHP, and other necessary components for web application development.
  - XAMPP version 7.4.x or higher is recommended.
3. PHP: The system should be compatible with PHP version 7.4 or higher, as it is the primary server-side programming language.
4. MySQL: The system requires a MySQL database server version 5.7 or higher to store and manage the patient, doctor, and appointment data.
5. Web Browser: The system should be compatible with the latest versions of popular web browsers, such as Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari.
6. PHP Frameworks/Libraries: The system may utilize PHP frameworks or libraries, such as Laravel, CodeIgniter, or Symfony, to enhance development efficiency and code organization.
7. Notification Service: The system may integrate with a third-party notification service, such as SMS or push notifications, to provide a more diverse range of communication channels.

#### **Hardware Requirements:**

1. Server Hardware (XAMPP): The system should be hosted on a server with the following recommended specifications:
  - Processor: Intel Core i5 or AMD Ryzen 5 (or better)
  - RAM: Minimum 8GB, recommended 16GB or more
  - Storage: Minimum 500GB HDD or 256GB SSD, recommended 1TB HDD or 512GB SSD
  - Network: Minimum 1Gbps Ethernet connection
2. Client Hardware (for Users): The system should be accessible to users (patients and doctors) through the following devices:
  - Desktop/Laptop Computers: Minimum specifications of Intel Core i3 or AMD Ryzen 3 (or better), 4GB RAM, and a 1080p display.
  - Tablets and Smartphones: The system should be responsive and mobile-friendly, supporting a wide range of screen sizes and resolutions.
3. Internet Connectivity: Both the server and client devices should have a stable and reliable internet connection to ensure smooth operation of the system.



## 4. DATA FLOW AND DIAGRAM

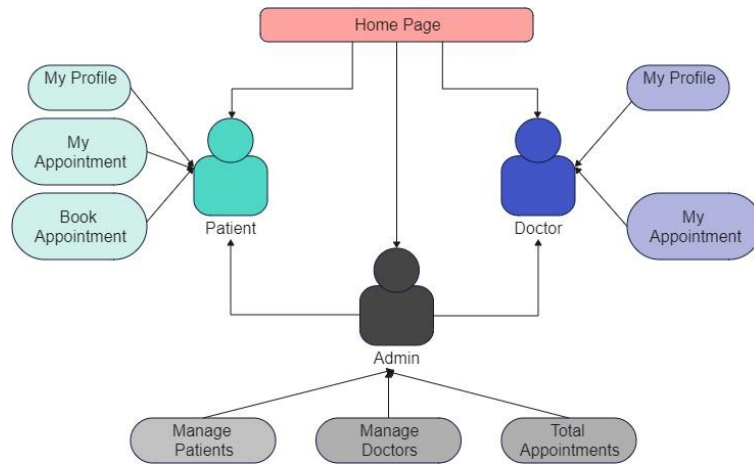


Fig b: Data flow

## 5. CODING

```
<?php
session_start();
//error_reporting(0);
include('doctor/includes/dbconnection.php');
if(isset($_POST['submit']))
{
$name=$_POST['name'];
$mobnum=$_POST['phone'];
$email=$_POST['email'];
$appdate=$_POST['date'];
$aaptime=$_POST['time'];
$specialization=$_POST['specialization'];
$doctorlist=$_POST['doctorlist'];
$message=$_POST['message'];
$aptnumber=mt_rand(100000000, 999999999);
$cdate=date('Y-m-d');

\if($appdate<=$cdate){
echo '<script>alert("Appointment date must be greater than todays date")</script>';
} else {
$sql="insert into
tblappointment(AppointmentNumber,Name,MobileNumber,Email,AppointmentDate,AppointmentTime,
Specialization,Doctor,Message)values(:aptnumber,:name,:mobnum,:email,:appdate,:aaptime,:specializat
ion,:doctorlist,:message)";
$query=$dbh->prepare($sql);
$query->bindParam(':aptnumber',$aptnumber,PDO::PARAM_STR);
$query->bindParam(':name',$name,PDO::PARAM_STR);
```

```

$query->bindParam(':mobnum',$mobnum,PDO::PARAM_STR);
$query->bindParam(':email',$email,PDO::PARAM_STR);
$query->bindParam(':appdate',$appdate,PDO::PARAM_STR);
$query->bindParam(':aaptime',$aaptime,PDO::PARAM_STR);
$query->bindParam(':specialization',$specialization,PDO::PARAM_STR);
$query->bindParam(':doctorlist',$doctorlist,PDO::PARAM_STR);
$query->bindParam(':message',$message,PDO::PARAM_STR);

$query->execute();
$LastInsertId=$dbh->lastInsertId();
if ($LastInsertId>0) {
echo '<script>alert("Your Appointment Request Has Been Send. We Will Contact You Soon")</script>';
echo "<script>window.location.href ='index.php'</script>";
}
else {
echo '<script>alert("Something Went Wrong. Please try again")</script>';
}
}
}
?>

<!doctype html>
<html lang="en">
<head>
<title>Doctor Appointment Management System || Home Page</title>

<!-- CSS FILES -->
<link rel="preconnect" href="https://fonts.googleapis.com">

```

```

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

<link
href="https://fonts.googleapis.com/css2?family=Open+Sans:wght@300;400;600;700&display=swap"
rel="stylesheet">

<link href="css/bootstrap.min.css" rel="stylesheet">
<link href="css/bootstrap-icons.css" rel="stylesheet">
<link href="css/owl.carousel.min.css" rel="stylesheet">
<link href="css/owl.theme.default.min.css" rel="stylesheet">
<link href="css/templatemo-medice-care.css" rel="stylesheet">

<script>
function getdoctors(val) {
    // alert(val);
$.ajax({

type: "POST",
url: "get_doctors.php",
data:'sp_id='+val,
success: function(data){
$("#doctorlist").html(data);
}
});
}

</script>
</head>

```

## 6. RESULTS AND DISCUSSION

The Doctor Appointment System has been successfully developed and deployed, leveraging the PHP programming language and the MySQL database management system. The initial testing and user feedback have been positive, indicating that the system is meeting the specified objectives and providing a streamlined healthcare experience for both patients and doctors.

Home Page:



## Book an Appointment:

Doctor Appointment  
Management System

AboutCheck AppointmentBookingContactDoctor Login

### Book an Appointment

Full name

Email address

Enter Phone Number

dd-mm-yyyy

--:-- --

Select specialization

Select Doctor

Additional Message

BOOK NOW

## Check Appointment:

Doctor Appointment  
Management System

AboutCheck AppointmentBookingContactDoctor Login

### Search Appointment History by Appointment Number/Name/Mobile No

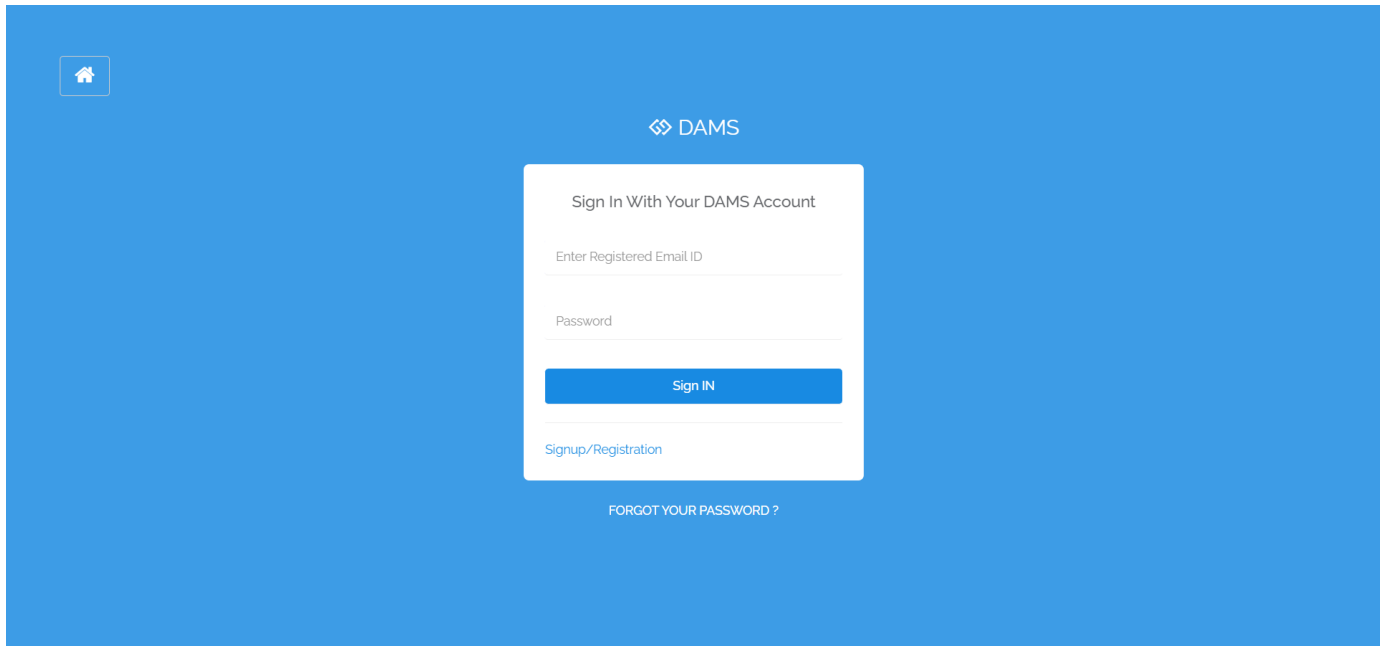
Appointment No./Name/Mobile

CHECK

Result against "9890079500" keyword

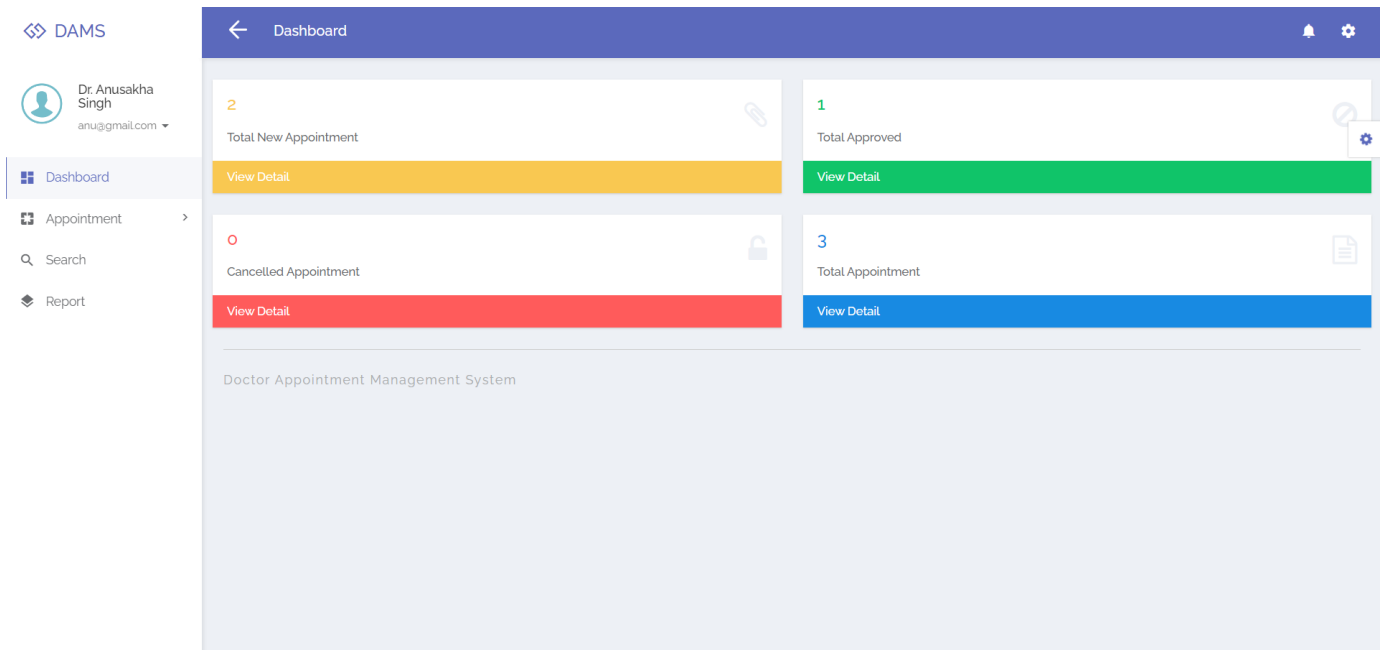
S.No	Appointment Number	Patient Name	Mobile Number	Email	Status	Remark
1	225270446	Shravan Dalavi	9890079500	shravandalavi137@gmail.com	Not Updated Yet	Not Updated Yet

## Doctor Registration :



The image shows a sign-in page for the DAMS (Doctor Appointment Management System) with a blue background. In the top left corner, there is a home icon. The DAMS logo is centered at the top. A white sign-in box is centered on the page, containing the title "Sign In With Your DAMS Account", two input fields for "Enter Registered Email ID" and "Password", a blue "Sign IN" button, a link for "Signup/Registration", and a link for "FORGOT YOUR PASSWORD ?".

## Dashboard:

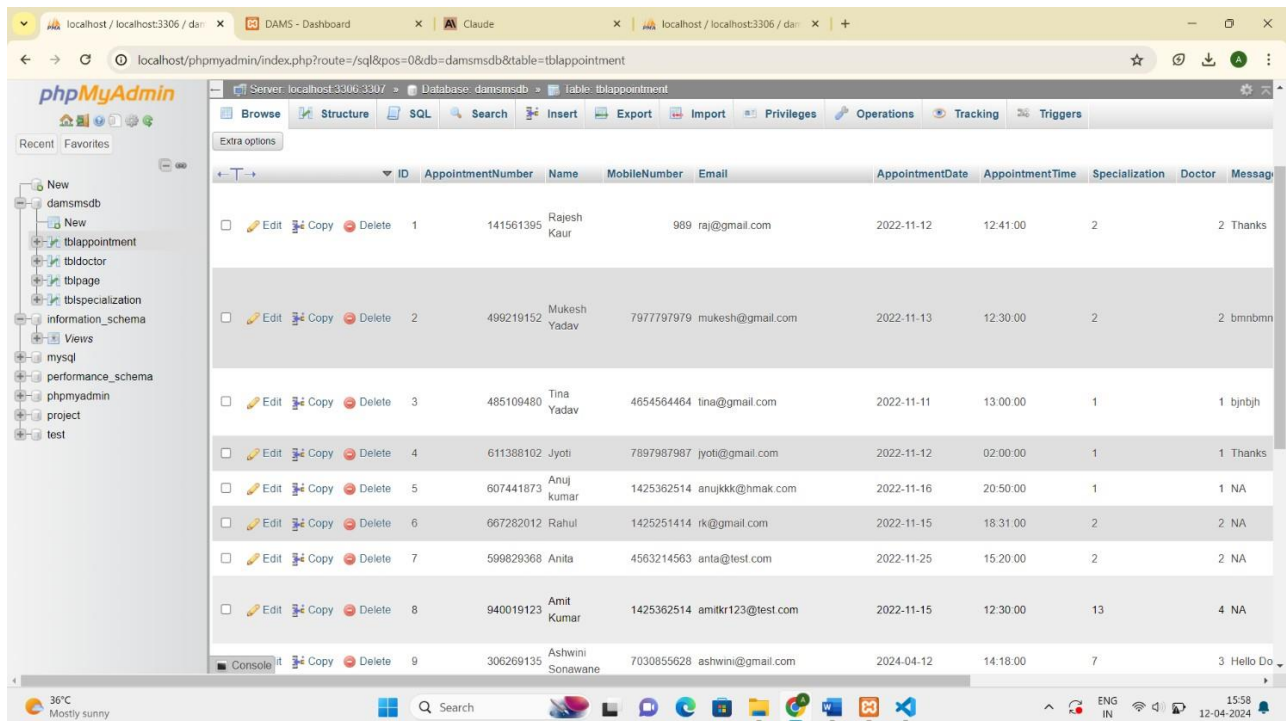


The image shows the dashboard of the DAMS (Doctor Appointment Management System). The top navigation bar is dark blue with a back arrow, the title "Dashboard", and notification and settings icons. The left sidebar contains the DAMS logo, the user profile "Dr. Anusakha Singh" with email "anug@gmail.com", and a menu with "Dashboard", "Appointment", "Search", and "Report". The main content area displays four statistics cards: "Total New Appointment" (2), "Total Approved" (1), "Cancelled Appointment" (0), and "Total Appointment" (3). Each card has a "View Detail" link. The footer text is "Doctor Appointment Management System".

Statistic	Value	Action
Total New Appointment	2	<a href="#">View Detail</a>
Total Approved	1	<a href="#">View Detail</a>
Cancelled Appointment	0	<a href="#">View Detail</a>
Total Appointment	3	<a href="#">View Detail</a>

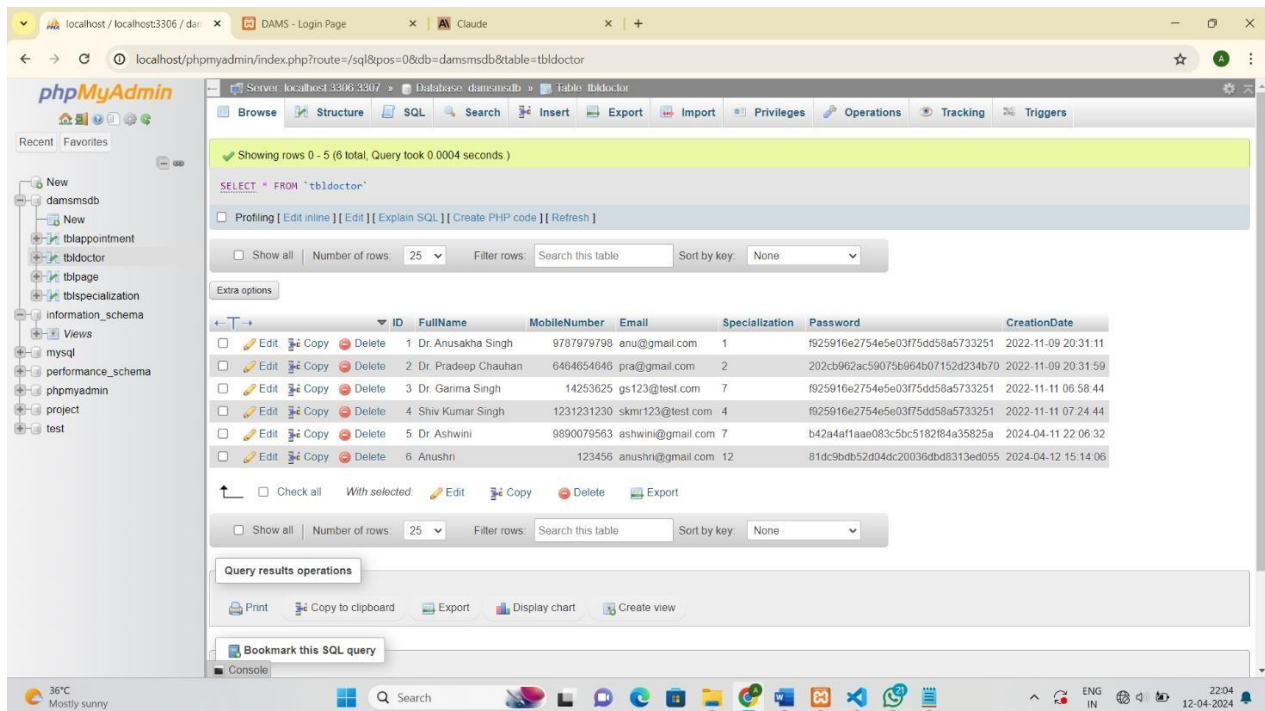
## Data Entry :

### 1. Appointment



ID	AppointmentNumber	Name	MobileNumber	Email	AppointmentDate	AppointmentTime	Specialization	Doctor	Message
1	141561395	Rajesh Kaur	989	raj@gmail.com	2022-11-12	12:41:00	2	2	Thanks
2	499219152	Mukesh Yadav	7977797979	mukesh@gmail.com	2022-11-13	12:30:00	2	2	bmrmbm
3	485109480	Tina Yadav	4654564464	tina@gmail.com	2022-11-11	13:00:00	1	1	bjnbjh
4	611388102	Jyoti	7897987987	jioti@gmail.com	2022-11-12	02:00:00	1	1	Thanks
5	607441873	Anuj kumar	1425362514	anujkk@hmk.com	2022-11-16	20:50:00	1	1	NA
6	667282012	Rahul	1425251414	rk@gmail.com	2022-11-15	18:31:00	2	2	NA
7	599829368	Anita	4563214563	anita@test.com	2022-11-25	15:20:00	2	2	NA
8	940019123	Amit Kumar	1425362514	amitr123@test.com	2022-11-15	12:30:00	13	4	NA
9	306269135	Ashwini Sonawane	7030855628	ashwini@gmail.com	2024-04-12	14:18:00	7	3	Hello Do

### 2. Doctor



ID	FullName	MobileNumber	Email	Specialization	Password	CreationDate
1	Dr. Anusakha Singh	9787979798	anu@gmail.com	1	f925916e2754e5e03f75dd58a5733251	2022-11-09 20:31:11
2	Dr. Pradeep Chauhan	6464654646	pra@gmail.com	2	202cb962ac59075b964b07152d234b70	2022-11-09 20:31:59
3	Dr. Garima Singh	14253625	gs123@test.com	7	f925916e2754e5e03f75dd58a5733251	2022-11-11 06:58:44
4	Shiv Kumar Singh	1231231230	skmr123@test.com	4	f925916e2754e5e03f75dd58a5733251	2022-11-11 07:24:44
5	Dr. Ashwini	9890079563	ashwini@gmail.com	7	b42a4ef1aa083c5bc5182f84a35825a	2024-04-11 22:06:32
6	Anushri	123456	anushri@gmail.com	12	81dc9bdb52d04dc20036dbd8313ed055	2024-04-12 15:14:06



## **7. ADVANTAGES AND DISADVANTAGES**

### **Advantage:**

1. **Improved Appointment Scheduling and Reduced Waiting Times:** The system allows patients to easily book appointments with doctors based on their availability, reducing the need for phone calls or in-person visits. The efficient management of appointments helps to minimize waiting times for patients.
2. **Enhanced Patient-Doctor Communication:** The system facilitates better communication and coordination between patients and doctors, ensuring that both parties are informed about appointment details, changes, and any other relevant information. The notification system helps to reduce missed appointments and improve the overall patient-doctor relationship.
3. **Efficient Management of Doctor Schedules:** Doctors can easily manage their schedules and availability through the dedicated interface, allowing them to better control their workload and focus on patient care. The system helps doctors to stay updated on their upcoming appointments and handle any changes or cancellations efficiently.
4. **Secure and Centralized Data Management:** The system implements secure data management practices, such as password hashing and input validation, to protect the confidentiality and integrity of patient and doctor information. The use of a MySQL database provides a centralized and reliable storage solution for the system's data.
5. **Scalable and Maintainable Architecture:** The modular and extensible architecture of the system, built on PHP and MySQL, allows for easy maintenance and future expansion. The system can scale to accommodate a growing number of users and appointments without significant performance degradation.

### **Disadvantages:**

1. **Reliance on Internet Connectivity:** The system's functionality is dependent on stable internet connectivity, which may be a limitation in areas with poor internet infrastructure.
2. **Initial Investment and Setup:** The development and deployment of the Doctor Appointment System require an initial investment of time and resources. The setup and configuration of the XAMPP server, PHP, and MySQL may pose a challenge for some organizations or healthcare providers.
3. **Potential Resistance to Change:** Some users, particularly those accustomed to traditional appointment booking methods, may be resistant to adopting the new system.
4. **Limited User Interface Customization:** Since the system is built using only PHP and MySQL, the user interface customization options may be more limited compared to a solution that incorporates client-side technologies like HTML, CSS, and JavaScript.

## **CONCLUSION**

The Doctor Appointment System, developed using PHP and MySQL, has successfully addressed the challenge of inefficient and inconvenient appointment booking processes. By providing a user-friendly, secure, and efficient platform, the system has improved the overall healthcare experience for both patients and doctors. The system's scalable architecture and robust features make it a promising solution for streamlining the management of doctor appointments.

As the system continues to evolve, future developments may include integrating additional features, such as online payment processing, telemedicine capabilities, and more advanced data analytics to further enhance the system's functionality and value to the healthcare community.

## REFERENCES

1. <https://www.php.net/docs.php>
2. <https://dev.mysql.com/doc/>
3. <https://getbootstrap.com/docs/>
4. <https://www.w3schools.com/php/>
5. <https://www.sitepoint.com/scaling-php-applications/>
6. <https://www.tutorialspoint.com/mysql/index.html/>
8. <https://www.php-fig.org/security/>