Project Report

for

Smart Clinic Management System with Telemedicine Integration

By

Student Name: Ahmad Amin Student Num: St20204605

Table of Contents

Abstract

The Smart Clinic Management System with Telemedicine Integration is a comprehensive web-based solution designed to facilitate efficient communication and management within healthcare facilities. This project aims to address the growing need for seamless patient-doctor interactions, particularly in the context of telemedicine.

The system includes essential functionalities such as user authentication, appointment scheduling, telemedicine integration, patient records management, prescription management, a unique patient ID system, billing and invoicing, and emergency features. These features are crucial for ensuring the smooth operation of a modern healthcare facility while enhancing patient care and accessibility.

Implementation of the system involves the use of the current web programming language PHP and integration with a secure database to store patient information and other relevant data. The user interface is designed to be user-friendly, providing easy access to various features and ensuring a positive user experience.

The formal report accompanying the project provides a detailed description of each component, including screenshots and source code, along with critical analysis and discussion. Additionally, a video presentation demonstrates the functionality of the system, focusing on telemedicine features, emergency functionalities, and overall user experience.

Overall, the Smart Clinic Management System with Telemedicine Integration offers a robust solution for healthcare facilities to streamline their operations, improve patient care, and adapt to the changing landscape of healthcare delivery.

1 Introduction

Specify In the rapidly evolving landscape of healthcare, the integration of technology has become imperative for improving patient care, enhancing accessibility, and streamlining administrative processes. One area where technological advancements hold significant promise is in the realm of clinic management systems, particularly those that incorporate telemedicine features. Telemedicine, defined as the remote delivery of healthcare services using telecommunications technology, offers unprecedented opportunities to overcome geographical barriers and increase access to medical care.

This project focuses on the development of a Smart Clinic Management System with Telemedicine Integration, aimed at revolutionizing the way healthcare is delivered and managed. By leveraging web-based technologies and incorporating telemedicine functionalities, this system seeks to address the challenges faced by healthcare facilities in providing efficient and effective care to patients.

The Smart Clinic Management System aims to offer a comprehensive solution that encompasses various aspects of clinic management, including user authentication, appointment scheduling, patient records management, prescription management, billing and invoicing, and emergency.

features. These functionalities are designed to enhance the overall efficiency of healthcare operations while ensuring seamless communication between patients and healthcare providers.

In this introduction, we provide an overview of the project, highlighting the need for a modern clinic management system with telemedicine integration. We delve into the significance of each component of the system and outline the objectives and goals of the project. Furthermore, we discuss the potential impact of this system on healthcare delivery and patient outcomes, underscoring its importance in the current healthcare landscape.

2 Problem Statement

The proposed Smart Clinic Management System with Telemedicine Integration aims to address several key challenges faced by healthcare facilities and patients in the current healthcare landscape. The core targeted problem that drives the development of this project includes:

One of the primary problems the system aims to solve is the limited accessibility to healthcare services, particularly for individuals residing in remote or underserved areas.

Traditional clinic management systems often face inefficiencies in appointment scheduling, patient records management, and prescription management.

3 Problem Solution for the Proposed System

3.1 Accessibility to Healthcare

By integrating telemedicine features, the system seeks to bridge the gap between patients and healthcare providers, enabling remote consultations and reducing the need for physical visits to healthcare facilities.

3.2 Efficient Clinic Management

The Smart Clinic Management System aims to streamline these processes through automated scheduling, comprehensive patient records management, and secure prescription management, thereby improving the overall efficiency of clinic operations.

3.3 Enhanced Patient Care

With the integration of telemedicine capabilities, the system aims to enhance patient care by facilitating real-time video consultations between patients and healthcare providers. This not only improves access to timely medical advice but also enables remote monitoring of patients' health conditions, leading to better health outcomes.

3.4 Unique System Features

While existing systems may offer some functionalities related to clinic management and telemedicine, the Smart Clinic Management System distinguishes itself by its focus on emergency features, such as quick access to critical patient information and the ability to escalate telemedicine sessions to in-person care if necessary. These features ensure comprehensive patient care and safety in emergency situations, setting the system apart from existing solutions.

4 Advantages/Benefits of Proposed System

The proposed Smart Clinic Management System with Telemedicine Integration offers several advantages and benefits:

4.1 Enhanced Accessibility

Patients can access healthcare services remotely, reducing the need for physical visits to clinics or hospitals. This improves access to healthcare for individuals with mobility limitations or those residing in remote areas.

4.2 Improved Patient Experience

The system provides a user-friendly interface for appointment scheduling, prescription management, and access to medical records. Patients can conveniently schedule appointments, receive reminders, and consult with healthcare providers from the comfort of their homes.

4.3 Efficient Appointment Management

Healthcare providers can manage appointments more efficiently with features such as real-time availability updates, automated reminders, and easy rescheduling options. This reduces scheduling conflicts and minimizes appointment no-shows.

4.4 Seamless Communication

Telemedicine integration enables real-time video consultations between patients and healthcare providers. Secure communication channels ensure privacy and confidentiality, allowing for effective diagnosis, treatment, and follow-up care.

4.5 Comprehensive Patient Records

The system facilitates comprehensive management of patient records, including personal information, medical history, treatment plans, and prescription details. Centralized access to patient data ensures continuity of care and enables informed decision-making by healthcare professionals.

4.6 Secure Prescription Management

Healthcare providers can securely create, update, and track patient prescriptions within the system. Electronic prescription delivery ensures timely access to medications, while prescription history tracking enhances medication management and adherence.

4.7 Unique Patient ID System

The system generates a unique patient ID at registration, serving as a secure identifier for accessing medical records and participating in telemedicine consultations. This streamlines the patient identification process and ensures data integrity.

4.8 Transparent Billing and Invoicing

Integrated billing and invoicing features facilitate transparent and accurate financial transactions between patients and healthcare providers. Patients can easily review invoices, make payments, and track payment history within the system.

4.9 Emergency Features

The system includes emergency features such as quick access to critical patient information and emergency contact details. Healthcare providers can escalate telemedicine sessions to in-person care if necessary, ensuring timely intervention during emergencies.

4.10 Continuous Improvement

The system's iterative development approach allows for continuous improvement based on user feedback, emerging healthcare trends, and technological advancements. Regular updates and enhancements ensure that the system remains relevant and effective in meeting evolving healthcare needs.

Overall, the proposed system enhances healthcare delivery by leveraging technology to improve accessibility, communication, efficiency, and patient outcomes.

5 System Architecture

The architecture described is a three-tier architecture. Here's how it aligns with the tiers:

5.1 Presentation Tier (Client Tier)

The presentation tier encompasses the client-side components responsible for interacting with users and presenting information to them.
In this architecture, HTML, CSS, and JavaScript handle the presentation logic, including the structure, style, and interactivity of the web pages.
Users interact with the system through their web browsers, which render the HTML content and execute client-side scripts.

5.2 Application Tier (Middle Tier)

The application tier, also known as the middle tier or logic tier, contains the server-side components responsible for processing user requests, executing business logic, and accessing data resources.
PHP serves as the primary language for implementing server-side logic in this architecture. It handles tasks such as user authentication, appointment scheduling, telemedicine integration, and database interactions.
The application tier communicates with both the presentation tier (client-side) and the data tier (database) to retrieve and process data and generate dynamic content for the users.

5.3 Data Tier (Database Tier)

The data tier consists of the database server and related components responsible for storing and managing the system's data.
MySQLi (MySQL Improved) serves as the database management system (DBMS) in this architecture, handling tasks such as storing patient records, appointment details, prescription data, billing information, and other relevant data.

The application tier interacts with the data tier to perform CRUD (Create, Read, Update, Delete) operations on the database, ensuring data integrity and consistency.

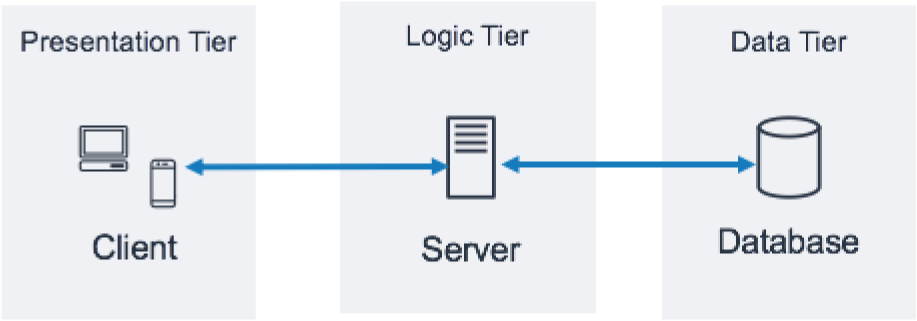


Figure 1: System Architecture of Smart Clinic Management System with Telemedicine Integration

6 Description of the System

The system is composed of several unique modules, each with its own functionality and importance it serves for the system. These modules are listed as follows:

6.1 Module 1: User Authentication/Login

One of the core features of the system is authorization and authentication. With a system allowing access to multiple roles. The system contains the following roles:

1. Admin
2. Doctor
3. Patient

Based on each role, each user has different rights and along with those different restrictions imposed while using the system. This allows security for the system so that the destined person performs their respective duties and does not violate the use of the system. The admin

The admin has all the privileges which includes from adding doctors, patients, viewing appointments and managing schedules to checking the billings and payments. So basically, admit can almost do anything.

The doctor has the right to add patients, schedule appointments for them, view scheduled appointments and prescribe medicines to them. They record and maintain the complete medical treatment and history of the patient. They can also conduct lab tests and see the patient's lab reports, they can also assign lab tests to the patient. The doctors can also set their schedules which is basically their time slots and their availability.

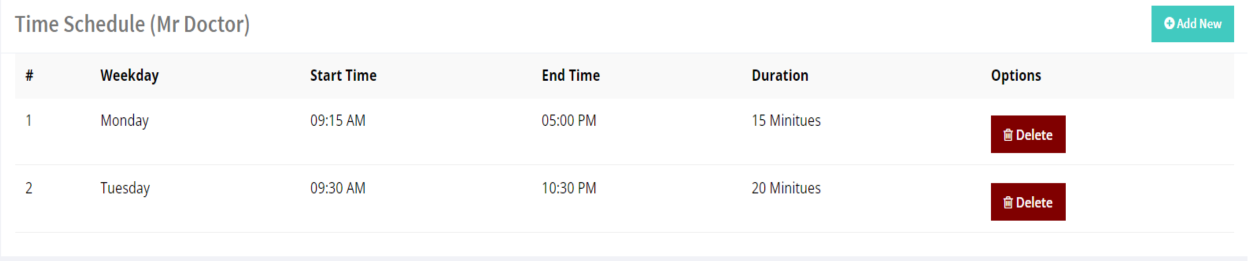
6.1.1 Screen Shot(s)

6.2 Module 2: Appointment Scheduling

In this module the doctors set their schedules, that is their available slots for booking appointments. They set their availability by selecting the day and time range, then they themselves or the patient can simply select the day, date and time from the available slots and schedule their appointment.

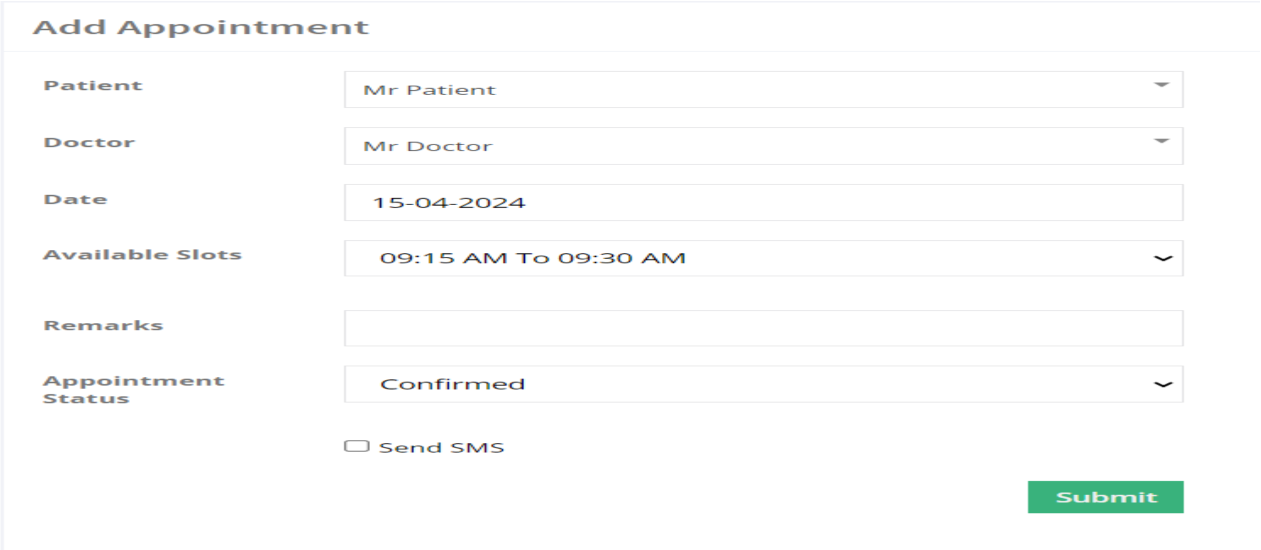
The doctors can select from all available patients to appoint their schedules whereas the patients can select from available doctors for their appointments.

6.2.1 Screen Shot(s)



Time Schedule (Mr Doctor)

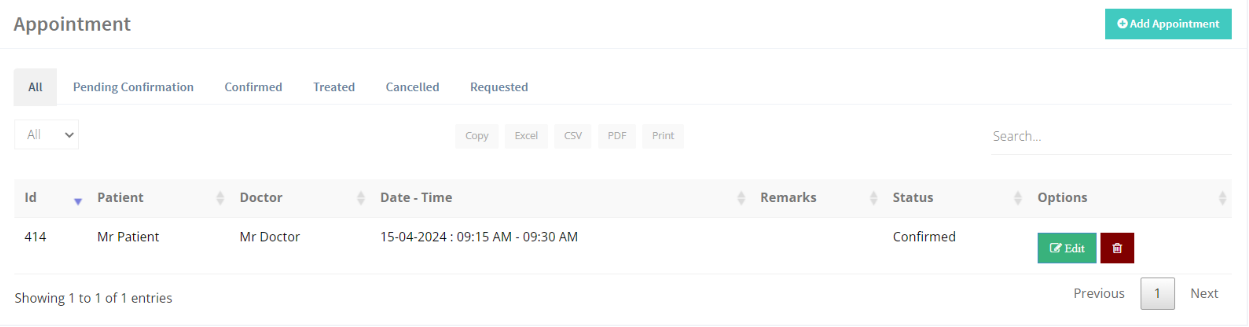
#	Weekday	Start Time	End Time	Duration	Options
1	Monday	09:15 AM	05:00 PM	15 Minitues	Delete
2	Tuesday	09:30 AM	10:30 PM	20 Minitues	Delete



Add Appointment

Patient	Mr Patient
Doctor	Mr Doctor
Date	15-04-2024
Available Slots	09:15 AM To 09:30 AM
Remarks	
Appointment Status	Confirmed

☐ Send SMS



Appointment

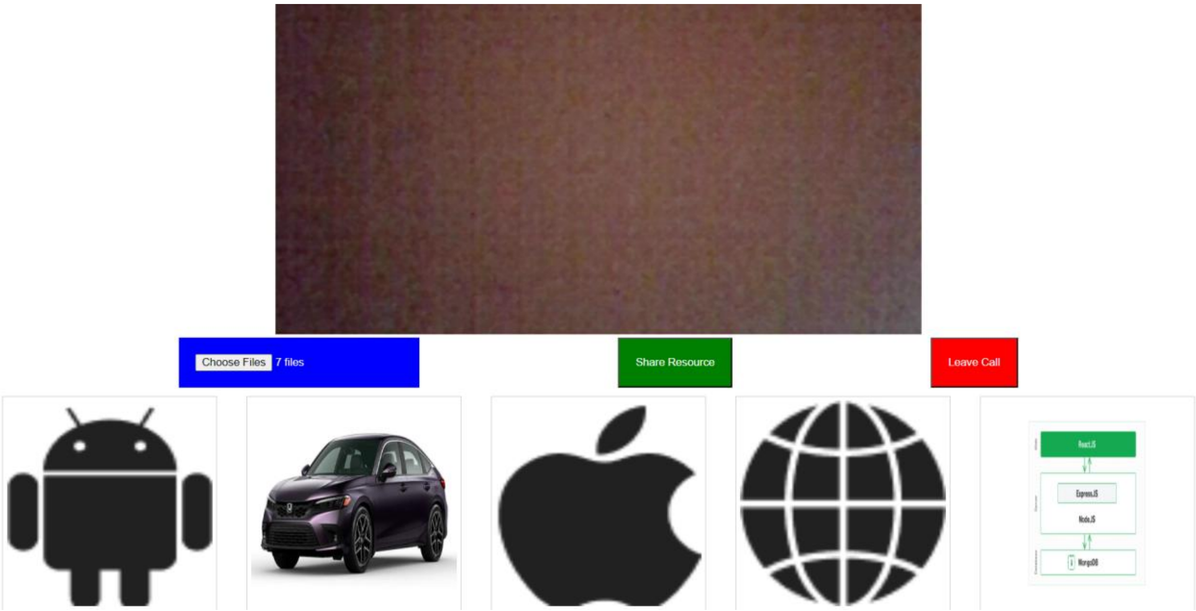
Id	Patient	Doctor	Date - Time	Remarks	Status	Options
414	Mr Patient	Mr Doctor	15-04-2024 : 09:15 AM - 09:30 AM		Confirmed	Edit

Showing 1 to 1 of 1 entries

6.3 Module 3: Telemedicine Integration

This module deals with video conference calls between the doctor and the patient. A video chat room is available for both the doctor and the patient to join so that they can exchange conversations and so that the doctor and diagnose the patient in a much more efficient way.

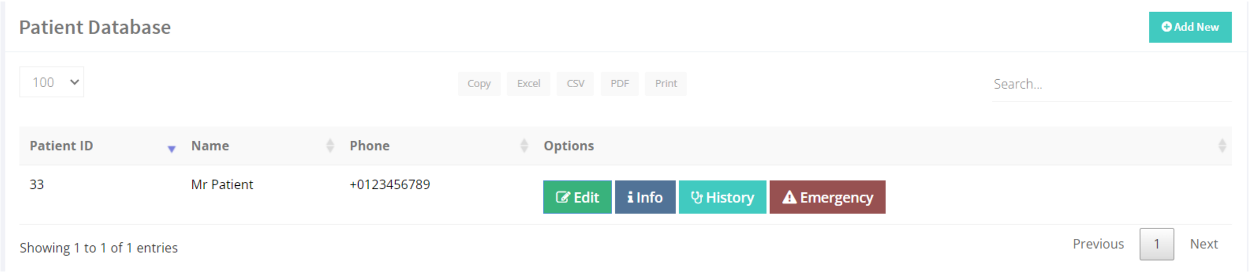
6.3.1 Screen Shot(s)



6.4 Module 4: Patient Record Management

This module deals with patient management, which includes CRUD operations performed on the patients, keeping their medical record, which includes their diagnosis, their reports and test along with the prescriptions prescribed by the doctors as well as their transactions performed for billing.

6.4.1 Screen Shot(s)



Patient Database

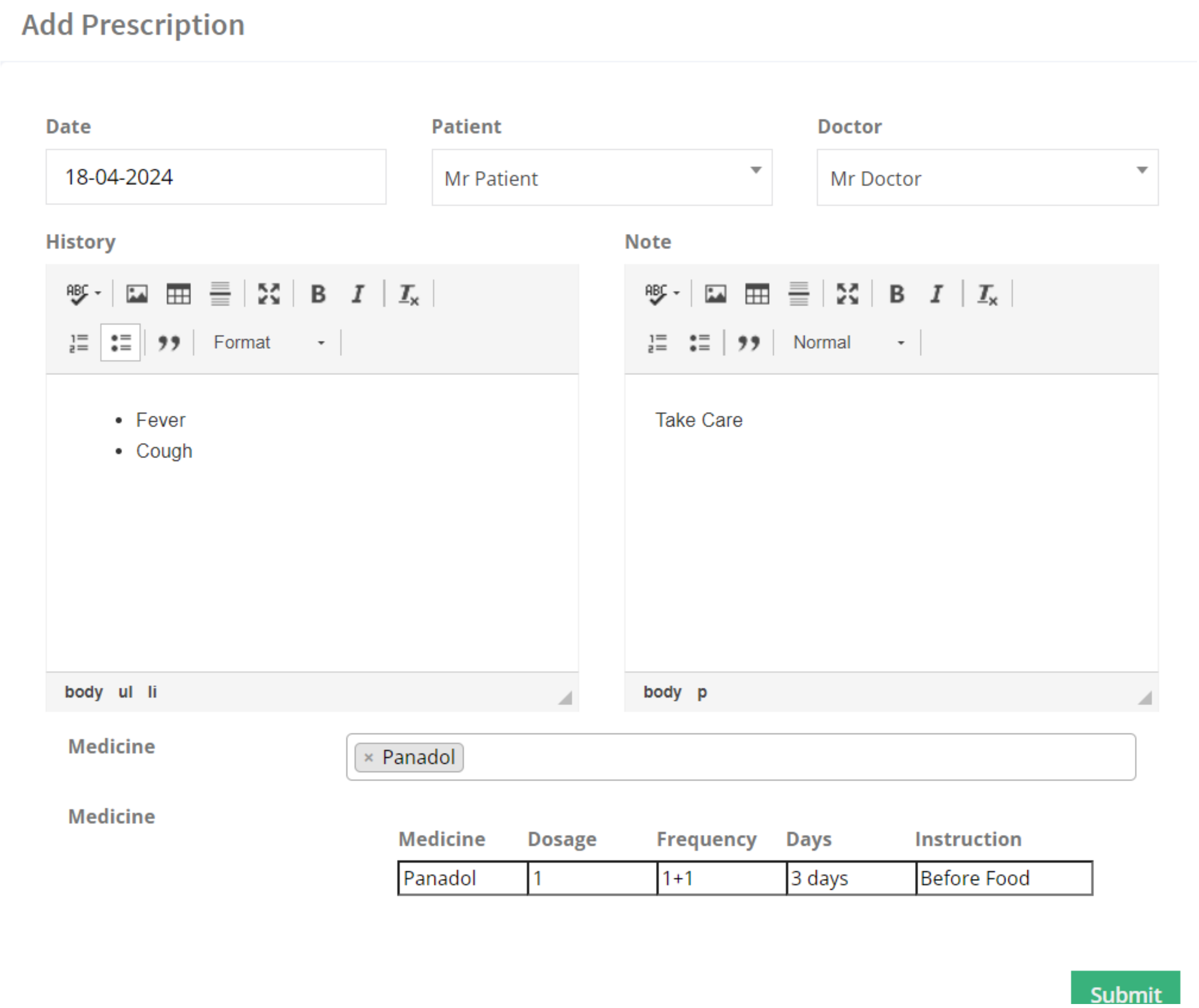
Patient ID	Name	Phone	Options
33	Mr Patient	+0123456789	Edit Info History Emergency

Showing 1 to 1 of 1 entries

6.5 Module 5: Prescription Management

This module allows the admin and doctors to add medicines which will be used for prescriptions. Then after the doctor has diagnosed the patient. The doctor will write a prescription for the patient. This prescription will contain a list of medications along with instructions. The patient will also be able to print or download the prescription.

6.5.1 Screen Shot(s)



Add Prescription

Date: 18-04-2024

Patient: Mr Patient

Doctor: Mr Doctor

History:

- Fever
- Cough

Note: Take Care

Medicine: × Panadol

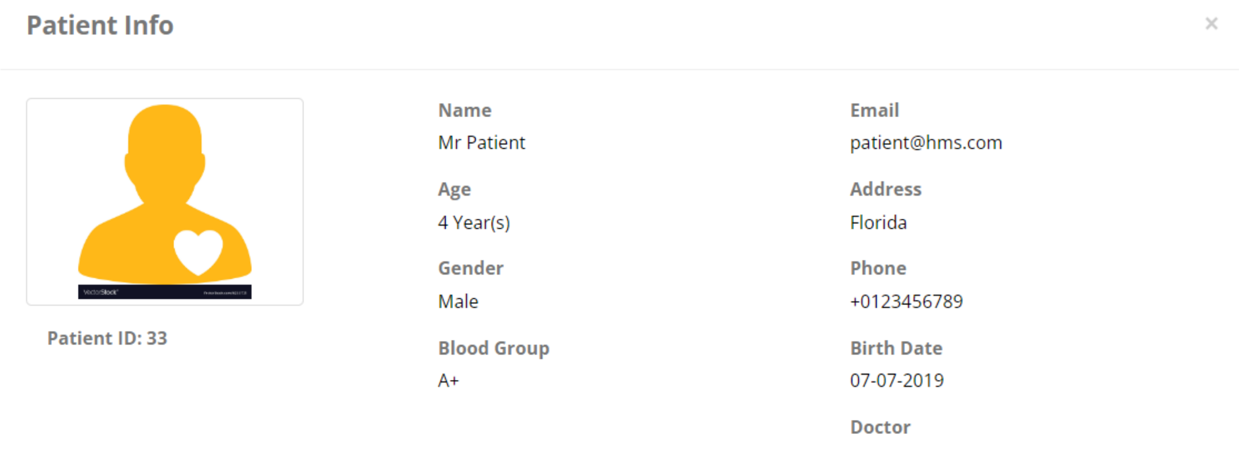
Medicine

Medicine	Dosage	Frequency	Days	Instruction
Panadol	1	1+1	3 days	Before Food

6.6 Module 6: Unique Patient ID System

The logic for having a unique patient ID is very simple. The database has the patient table which has the constraint of a unique id which in this case is the patient ID also this is set to auto increment so that every record gets an incremented patient ID, and the constraint unique makes sure that it stays unique.

6.6.1 Screen Shot(s)



Patient Info

Patient ID: 33

Name
Mr Patient

Age
4 Year(s)

Gender
Male

Blood Group
A+

Email
patient@hms.com

Address
Florida

Phone
+0123456789

Birth Date
07-07-2019

Doctor

6.7 Module 7: Billing and Invoicing

In this PHP code, PayPal's Payments Pro API is integrated to facilitate direct credit card payments on a website. Let's break down the key functionalities:

Constructor Function:

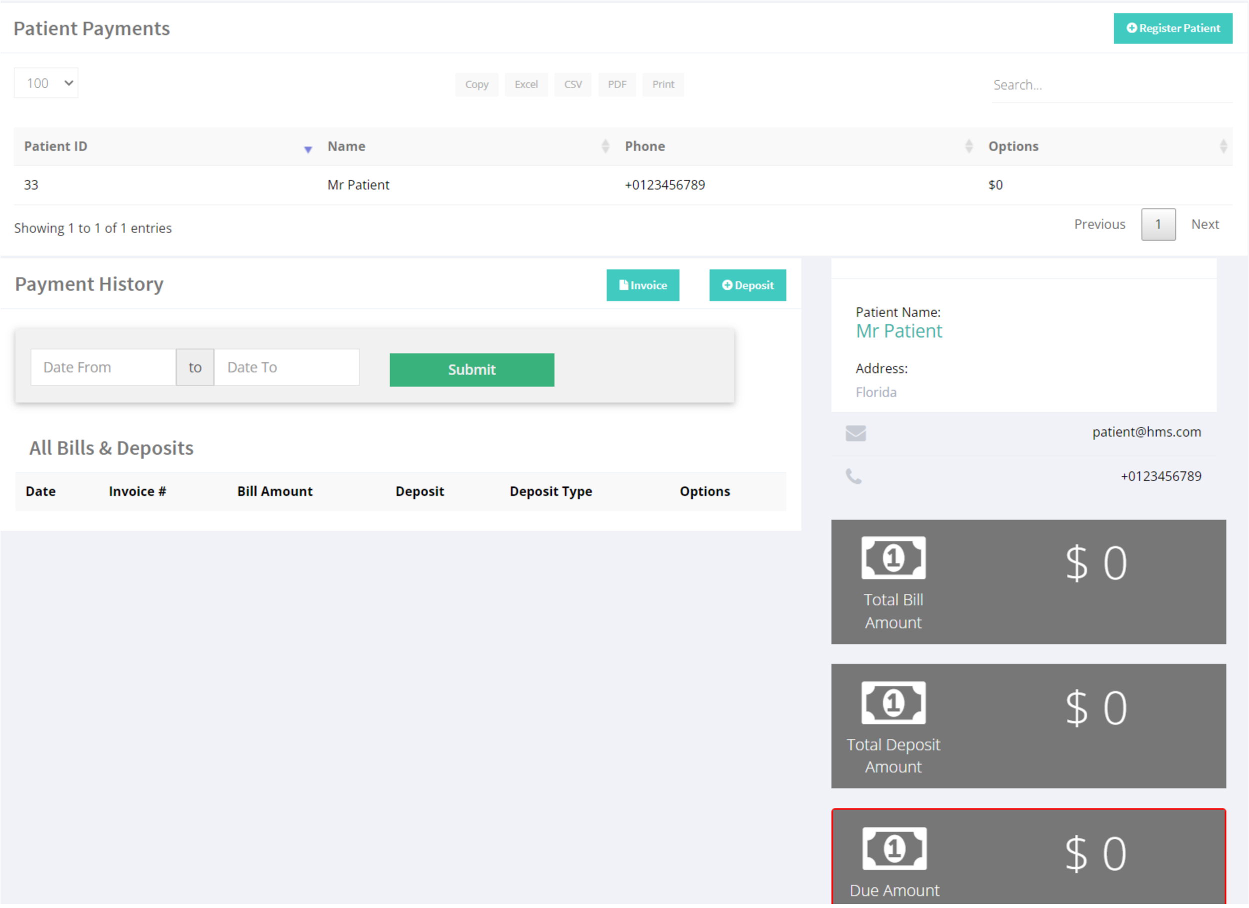
The __construct() function initializes the PayPal library and sets up necessary configurations. It loads required models, helpers, and the PayPal library.
It checks if the PayPal gateway is in test mode or live mode and sets the configuration accordingly.
Index Function:

The index() function loads a view file (paypal/samples/payments_pro.php) which might contain a form or UI elements for initiating payments.
Do_direct_payment Function:

This function handles the direct payment process using PayPal's Payments Pro API.
It receives payment details as input parameters.
Constructs arrays with necessary payment information such as credit card details, payer information, billing address, payment details, etc.
Calls the DoDirectPayment() method of the PayPal Pro library (loaded in the constructor) with the prepared payment data.
Checks if the API call was successful using the APICallSuccessful() method provided by the PayPal Pro library.
If the payment is successful, it updates the payment status in the database and redirects the user to appropriate pages based on the user's role (e.g., patient or finance staff).
Get_balance Function:

This function is responsible for retrieving the balance from the PayPal account.
It constructs the necessary request data.
Calls the GetBalance() method of the PayPal Pro library.
Handles the response, either displaying errors or showing the balance in the view.
Overall, this controller provides the backend functionality to interact with PayPal's Payments Pro API for processing direct credit card payments on the website. It handles payment processing, error handling, and updating payment status in the database.

6.7.1 Screen Shot(s)



Patient Payments

Patient ID	Name	Phone	Options
33	Mr Patient	+0123456789	$0

Showing 1 to 1 of 1 entries

Payment History

All Bills & Deposits

Date	Invoice #	Bill Amount	Deposit	Deposit Type	Options

Patient Name:
Mr Patient

Address:
Florida

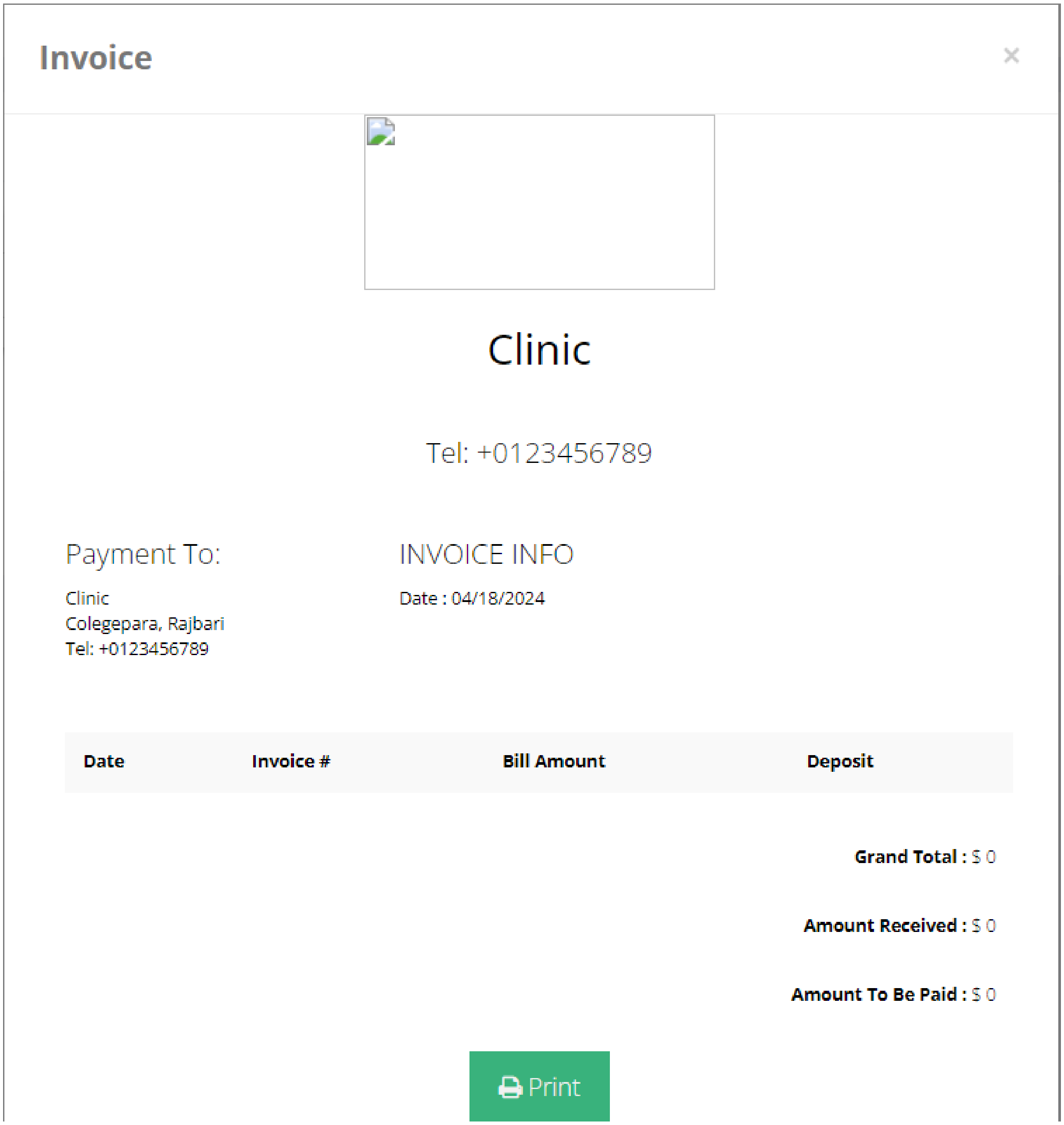
patient@hms.com

+0123456789

$ 0
Total Bill Amount

$ 0
Total Deposit Amount

$ 0
Due Amount



Clinic

Tel: +0123456789

Payment To:

Clinic
Colegepara, Rajbari
Tel: +0123456789

INVOICE INFO

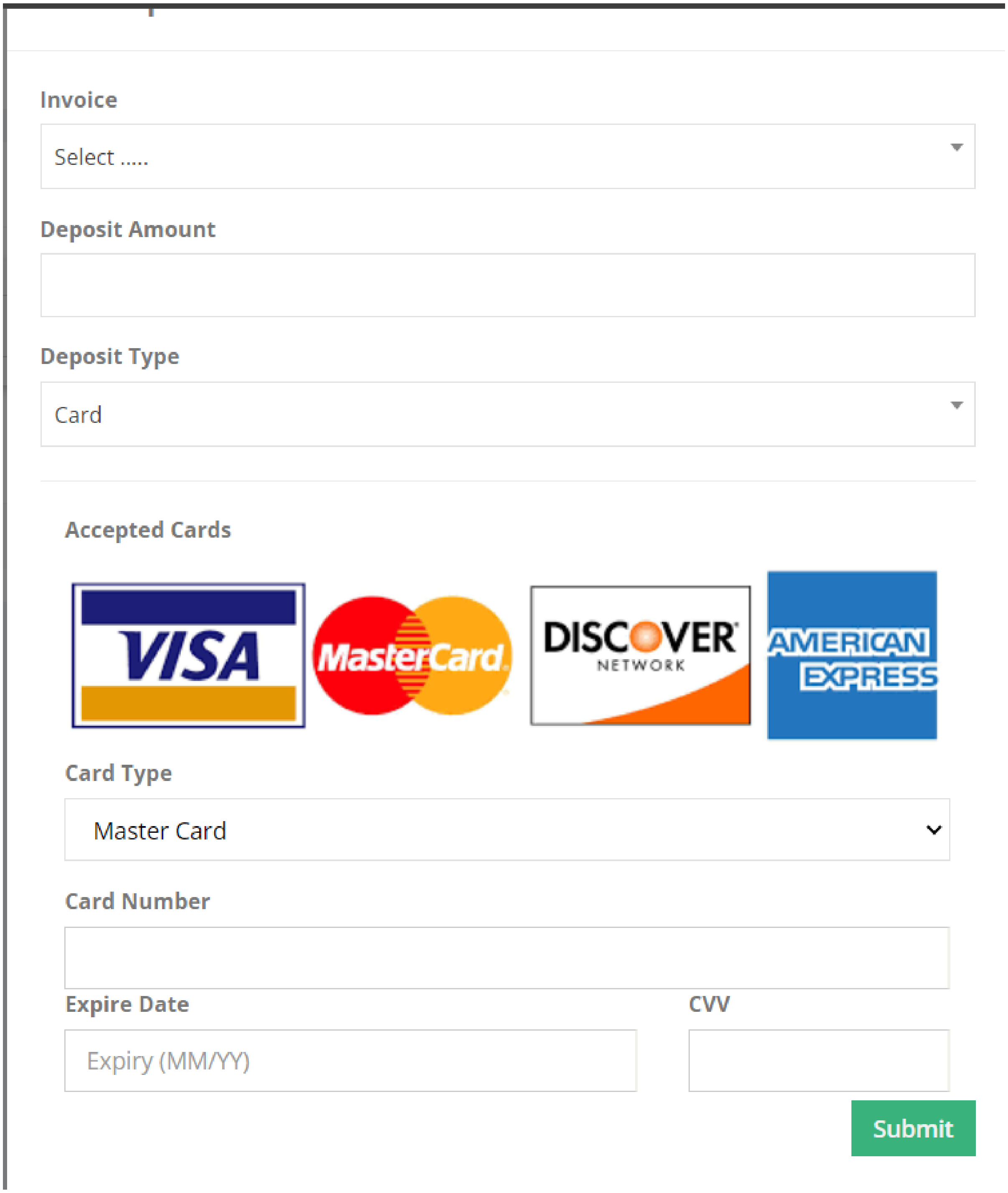
Date : 04/18/2024

Date	Invoice #	Bill Amount	Deposit

Grand Total : $ 0

Amount Received : $ 0

Amount To Be Paid : $ 0



Invoice: Select .....

Deposit Amount:

Deposit Type: Card

Accepted Cards

Card Type: Master Card

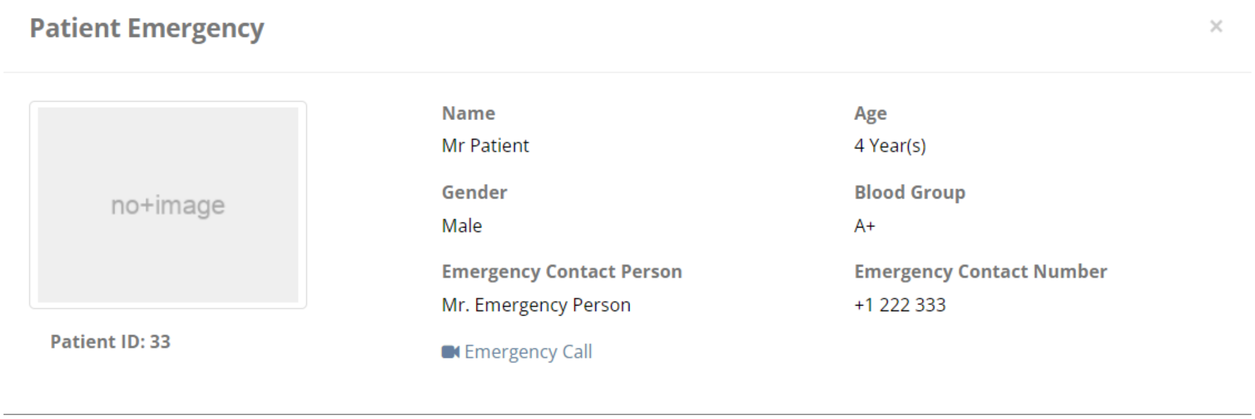
Card Number:

Expire Date: Expiry (MM/YY)

CVV:

6.8 Module 8: Emergency Features

6.8.1 Screen Shot(s)



Patient Emergency

Patient ID: 33

Name
Mr Patient

Age
4 Year(s)

Gender
Male

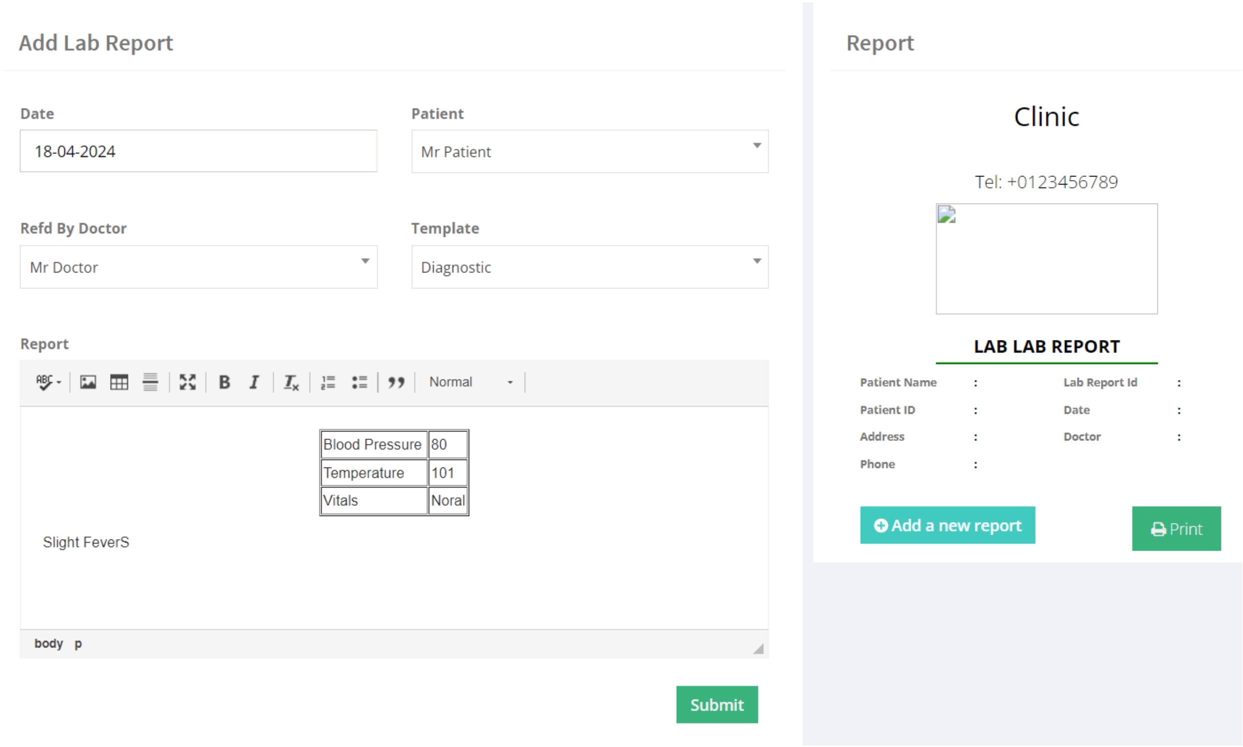
Blood Group
A+

Emergency Contact Person
Mr. Emergency Person

Emergency Contact Number
+1 222 333

6.9 Module 9: Reports

6.9.1 Screen Shot(s)



Add Lab Report

Date
18-04-2024

Patient
Mr Patient

Refd By Doctor
Mr Doctor

Template
Diagnostic

Report

Blood Pressure	80
Temperature	101
Vitals	Noral

Slight FeverS

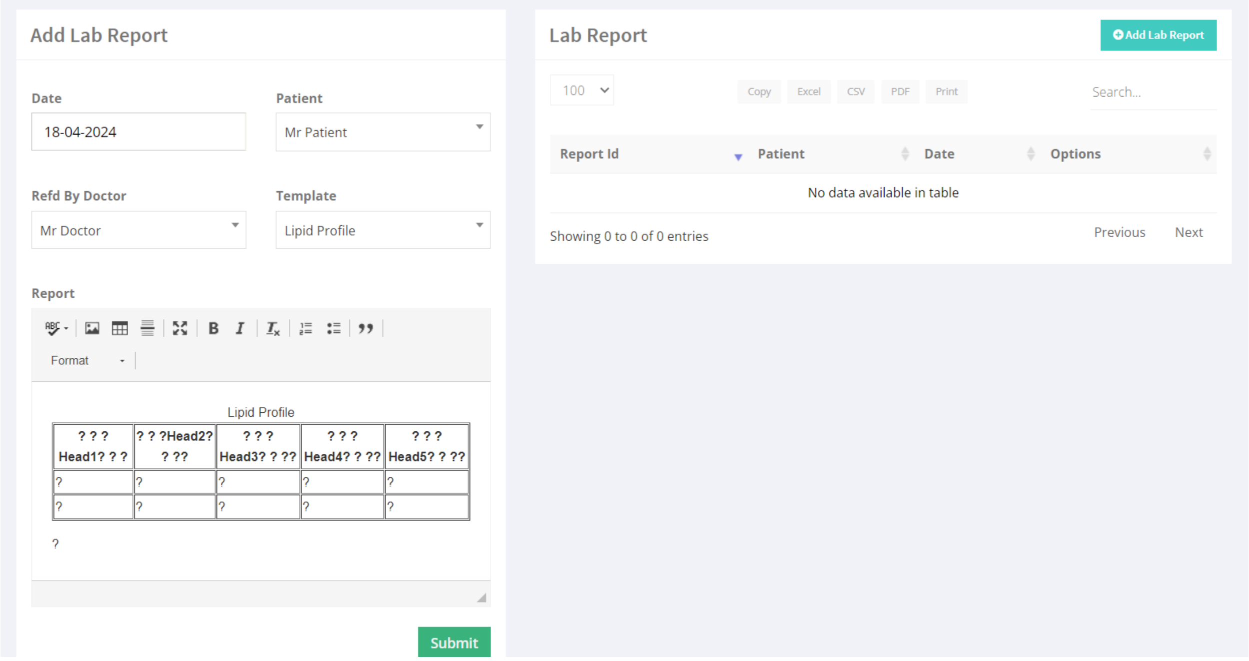
Report

Clinic

Tel: +0123456789

LAB LAB REPORT

Patient Name :
Lab Report Id :
Patient ID :
Date :
Address :
Doctor :
Phone :



Add Lab Report

Date: 18-04-2024

Patient: Mr Patient

Refd By Doctor: Mr Doctor

Template: Lipid Profile

Report

Lipid Profile

? ? ? Head1? ? ?	? ? ?Head2? ? ??	? ? ? Head3? ? ??	? ? ? Head4? ? ??	? ? ? Head5? ? ??
?	?	?	?	?
?	?	?	?	?

?

Lab Report

Report Id	Patient	Date	Options
No data available in table			

Showing 0 to 0 of 0 entries

7 Telemedicine Integration

WebRTC (Web Real-Time Communication) is used for telemedicine purposes. Here's how WebRTC is used in the provided code:

Accessing Media Devices: The startVideo() function uses navigator.mediaDevices.getUserMedia() to access the user's webcam and microphone. This function prompts the user for permission to use these devices and returns a media stream containing audio and video tracks.

Creating Peer Connection: The createPeerConnection() function creates an RTCPeerConnection object, which represents a WebRTC connection between the local device and remote peers. It configures ICE (Interactive Connectivity Establishment) servers for handling network traversal. ICE servers help in establishing a connection between peers even if they are behind NATs (Network Address Translators) or firewalls.

Handling Track Events: The handleTrackEvent() function is called when a remote peer adds a new media track to the connection. It creates a new <video> element for displaying the remote video stream and appends it to the #videos div.

Handling ICE Candidate Events: The handleICECandidateEvent() function is called whenever the local peer generates a new ICE candidate. ICE candidates are network information used by peers to establish direct peer-to-peer connections. These candidates are sent to the remote peer via signaling to establish connectivity.

Signaling: The sendMessage() and receiveMessage() functions handle signaling between peers. Signaling is the process of exchanging metadata (like session descriptions, ICE candidates) between peers to establish and manage a WebRTC connection. In this code, the signaling process is abstracted, and the actual implementation (using WebSocket, HTTP requests, etc.) is not provided.

Starting the Call: The startCall() function initiates the video call by starting the local video stream, creating a peer connection, creating an offer (description of the local session), setting the local description, and sending the offer to the remote peer.

Leaving the Call: The leaveCall() function redirects the user to another URL, possibly indicating the end of the telemedicine session.

Sharing Resource: The shareResource() function allows users to share images during the call. It listens for changes in the file input, reads the selected image files using FileReader, converts them to data URLs, and displays them as previews in the #resourcePreviews div.

In summary, this code establishes a basic video conferencing application using WebRTC, allowing users to establish peer-to-peer connections, share video/audio streams, exchange signaling data, and share images during the call. However, it lacks a complete signaling implementation and backend server interaction for a fully functional application.

8 Emergency Features and Billing System

The emergency feature lists the emergency contact number of the patient focal person. This allows the doctor to get connected with the emergency focal person and guide them or with the option of emergency call, this emergency call connects the doctor to the critical patient through conference call. This conference call is carried out using WebRTC.

WebRTC (Web Real-Time Communication) for telemedicine purposes. Here's how WebRTC is used in the provided code:

Accessing Media Devices: The startVideo() function uses navigator.mediaDevices.getUserMedia() to access the user's webcam and microphone. This function prompts the user for permission to use these devices and returns a media stream containing audio and video tracks.

Creating Peer Connection: The createPeerConnection() function creates an RTCPeerConnection object, which represents a WebRTC connection between the local device and remote peers. It configures ICE (Interactive Connectivity Establishment) servers for handling network traversal. ICE servers help in establishing a connection between peers even if they are behind NATs (Network Address Translators) or firewalls.

Handling Track Events: The handleTrackEvent() function is called when a remote peer adds a new media track to the connection. It creates a new <video> element for displaying the remote video stream and appends it to the #videos div.

Handling ICE Candidate Events: The handleICECandidateEvent() function is called whenever the local peer generates a new ICE candidate. ICE candidates are network information used by peers to establish direct peer-to-peer connections. These candidates are sent to the remote peer via signaling to establish connectivity.

Signaling: The sendMessage() and receiveMessage() functions handle signaling between peers. Signaling is the process of exchanging metadata (like session descriptions, ICE candidates) between peers to establish and manage a WebRTC connection. In this code, the signaling process is abstracted, and the actual implementation (using WebSocket, HTTP requests, etc.) is not provided.

Starting the Call: The startCall() function initiates the video call by starting the local video stream, creating a peer connection, creating an offer (description of the local session), setting the local description, and sending the offer to the remote peer.

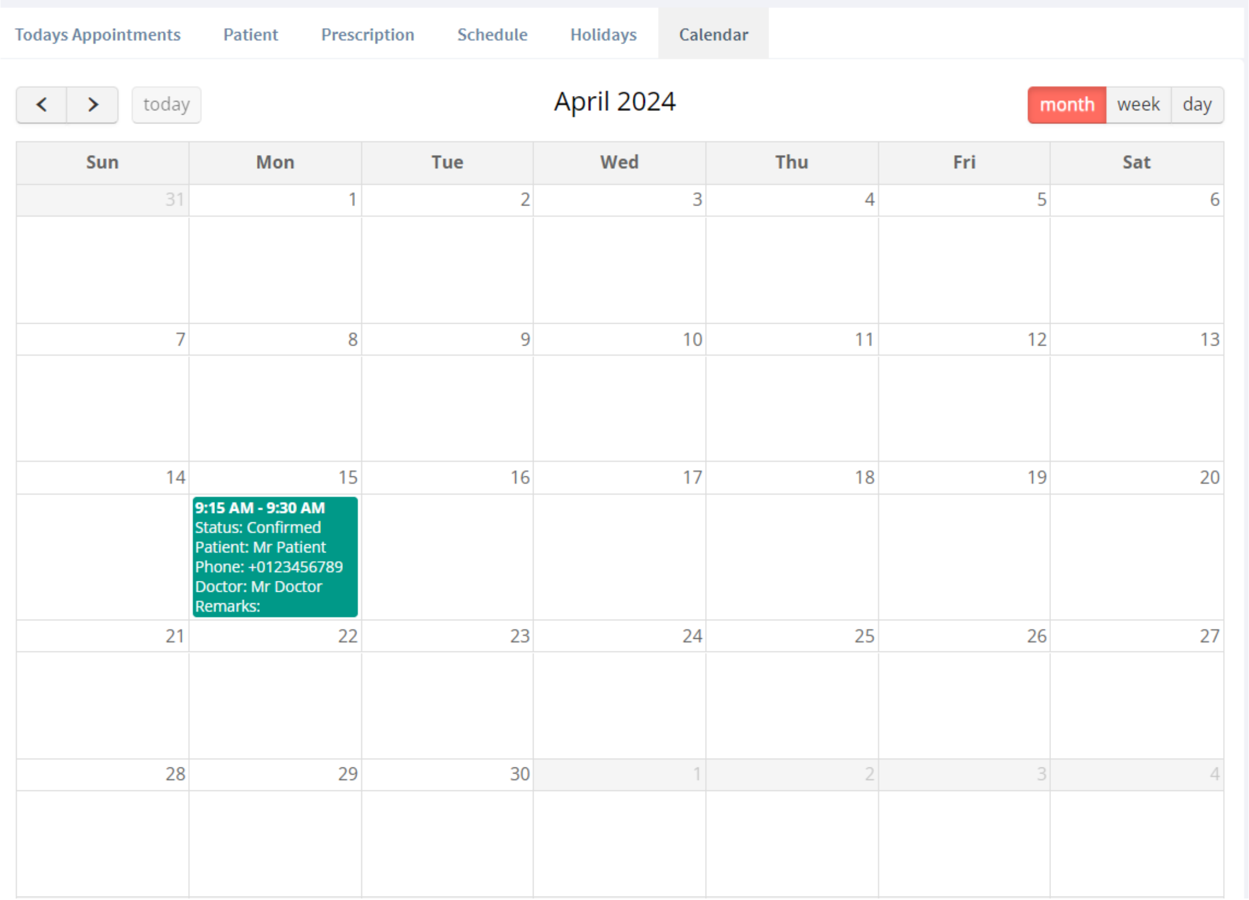
Leaving the Call: The leaveCall() function redirects the user to another URL, possibly indicating the end of the telemedicine session.

Sharing Resource: The shareResource() function allows users to share images during the call. It listens for changes in the file input, reads the selected image files using FileReader, converts them to data URLs, and displays them as previews in the #resourcePreviews div.

In summary, this code establishes a basic video conferencing application using WebRTC, allowing users to establish peer-to-peer connections, share video/audio streams, exchange signaling data, and share images during the call. However, it lacks a complete signaling implementation and backend server interaction for a fully functional application.

9 Calendar

The calendar feature allows the smooth and easier way to keep track of the appointments through a calendar view. This calendar gives the users a quick insight of the appointments and the schedule, which are highlighted with a color and text with information regarding the appointment.



April 2024

Sun	Mon	Tue	Wed	Thu	Fri	Sat
31	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15 **9:15 AM - 9:30 AM** Status: Confirmed Patient: Mr Patient Phone: +0123456789 Doctor: Mr Doctor Remarks:	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	1	2	3	4

10 Tools and Technologies

Table 2: Tools and Technologies for the Targeted Project.

	Tools	**Version**	**Rationale**
Tools And Technologies	MS Visual Studio	2022	IDE
	XAMPP	3.3.0	Apache Server
	MS Word	2019	Documentation
	Technology	**Version**	**Rationale**
	PHP	7.0	Programming language
	MySQL	2013	Query Language
	Html	5	Web Development
	CSS	5	Styling
	JavaScript	ES6	Scripting Language

11 Conclusion

In conclusion, the development of the Smart Clinic Management System with Telemedicine Integration addresses critical needs in the healthcare industry by leveraging modern web technologies to enhance patient care, streamline administrative processes, and facilitate remote medical consultations. Through this project, we have achieved the following key objectives:

Improved Patient Care: By providing patients with a user-friendly platform for scheduling appointments, accessing medical records, and engaging in telemedicine consultations, the system enhances the overall patient experience and ensures timely access to healthcare services.

Efficient Clinic Management: The integration of features such as appointment scheduling, patient records management, prescription management, and billing and invoicing streamlines clinic operations, reduces administrative overhead, and enables healthcare providers to focus more on delivering quality care.

Enhanced Telemedicine Capabilities: The telemedicine integration enables real-time video consultations between patients and healthcare providers, breaking down geographical barriers and improving access to healthcare, especially in remote or underserved areas. The inclusion of a virtual waiting room and document-sharing capabilities further enriches the telehealth experience.

Data Security and Compliance: With a focus on data privacy and compliance with healthcare regulations, the system ensures the secure handling of sensitive patient information. By implementing robust authentication mechanisms, encryption protocols, and access controls, the confidentiality and integrity of patient data are safeguarded.

Learning and Skill Development: The development of this project has provided valuable opportunities for learning and skill development. Through the implementation of various functionalities using technologies such as HTML, CSS, JavaScript, PHP, and MySQLi, students have honed their programming, database management, and web development skills. Moreover, the project has fostered critical thinking, problem-solving, and project management abilities, preparing students for future endeavors in the field of web programming and healthcare informatics.

In summary, the Smart Clinic Management System with Telemedicine Integration represents a significant step forward in modernizing healthcare delivery systems, leveraging the power of technology to improve patient outcomes, enhance operational efficiency, and promote accessible and inclusive healthcare services for all.

12 References

1. PayPal Developer Documentation: https://developer.paypal.com/docs - Provides comprehensive documentation, guides, and resources for integrating PayPal APIs into web applications.
2. PayPal Payments Pro API Reference: https://developer.paypal.com/docs/paypal-payments-pro/integration-guide/paypal-payments-pro-overview - Detailed documentation on the functionalities and usage of PayPal Payments Pro API for processing direct payments.
3. CodeIgniter User Guide: https://codeigniter.com/user_guide - Offers extensive documentation and guides for working with the CodeIgniter framework, providing insights into best practices for developing PHP applications.
4. PHP Documentation: https://www.php.net/docs.php - Official PHP documentation offering reference materials, tutorials, and examples for PHP developers.
5. Stack Overflow: https://stackoverflow.com - Community-driven platform for asking questions and finding solutions to programming-related queries, often containing valuable insights and troubleshooting tips for integrating payment gateways like PayPal.
6. WebRTC Official Website: https://webrtc.org/
7. The official website provides comprehensive documentation, guides, and resources for understanding and implementing WebRTC technology.
8. MDN Web Docs - WebRTC API: https://developer.mozilla.org/en-US/docs/Web/API/WebRTC_API
9. Mozilla Developer Network (MDN) provides detailed documentation on the WebRTC API, including interfaces, methods, and usage examples.
10. Google Developers - WebRTC Samples: https://webrtc.github.io/samples/
11. WebRTC Glossary: https://webrtcglossary.com/
12. WebRTC Experiments by Muaz Khan: https://www.webrtc-experiment.com/
13. WebRTC Weekly Newsletter: https://webrtcweekly.com/
14. WebRTC Hacks: https://webrtchacks.com/
15. A blog and resource hub focusing on practical tips, tricks, and hacks for working with WebRTC, including tutorials, case studies, and best practices.
16. GitHub WebRTC Organization: https://github.com/webrtc