**MANGALORE UNIVERSITY**

**Project Report On**

**“KamathResidency.com”**

# **Submitted By**

# **SHRIHARI A M**

## Reg.No.P05MG22S038035

**MGM College, Udupi**

## Under the guidance of

**Ms. Krithika Pai Ms. Pallavi**

Techpro Systems, Assistant Professor,

Udupi PG Department of Computer Science,

MGM College, Udupi

## In the partial fulfillment of the requirement for the degree of Master of Science(ComputerScience)

**MAHATMAGANDHIMEMORIALCOLLEGEUDUPI– 576102**

**2023-2024**

### MAHATMA GANDHI MEMORIAL COLLEGEUDUPI– 576102

**POST GRADUATION DEPARTMENT OF COMPUTER SCIENCE**

## CERTIFICATE

This is to certify that the project entitled “**KamathResidency.com**” has been carried out by **SHRIHARI A M** (**Reg.No:P05MG22S038035),** student of fourth semester **M.Sc. (Computer Science)** under the supervision of **Ms.Pallavi,**Assistant Professor, PG Department of Computer Science,Mahatma Gandhi Memorial College,Udupi.This dissertation is submitted in partial fulfillment of the requirement for the award of **Master of Science(ComputerScience)** by Mangalore University during the Academic year2023-2024.

### Internal Guide Head of the Department Principal

**Internal Examiner External Examiner**

**Submitted for the viva-voice examination held on:………………**

## DECLARATION

This project work entitled “KamathResidency.com” has been successfully carried out by me, **SHRIHARI A M** (Reg No: **P05MG22S038035**), student of Fourth semester M.Sc. Computer Science, at **Techpro Systems** Udupi and also under the supervision and guidance of **Ms.Pallavi**, Assistant Prof., Department of Computer Science, Mahatma Gandhi Memorial College, Udupi. This is submitted in partial fulfillment for the award of degree in **Master of Science** in computer science (MSc-Computer Science) by Mahatma Gandhi Memorial College, Udupi during academic year 2023-2024.

**Shrihari A M**

Register No:P05MG22S038035

IV SEM MSc Computer Science Mahatma Gandhi Memorial College Udupi-576102

## ACKNOWLEDGEMENT

The successful completion of my task will be incomplete without complementing those who made it possible and whose guidance and encouragement made my efforts successful.

I like to place on record the valuable guidance and constant encouragement by my guide, Ms. Pallavi, Assistant Prof., Department of Computer Science, Mahatma Gandhi Memorial College, Udupi for giving a nice shape to my project.

My sincere thanks to all the staff members of Computer Science, Mahatma Gandhi Memorial College, Udupi for their help and guidance during my MSc Course.

I would like to express my sincere gratitude to my respected guide Dr .M. Vishwanath Pai, Techpro Systems,Udupi providing constant guidance and encouragement during the course of this dissertation work and timely supports till my successful completion of the project work.

I would like to thank all teaching and non-teaching staff member of Computer Science, Mahatma Gandhi Memorial College, Udupi for their cooperation and constant encouragement, which helped me in successfully completing my project.

## ABSTRACT

This paper presents the development and functionality of the Kamath Residency Booking Website, an advanced online platform designed to facilitate seamless hotel reservations at Kamath Residency.

The Kamath Residency Booking Website features a comprehensive overview of the hotel's offerings, including detailed descriptions of room types, amenities, and services. The platform's core functionality allows users to browse available rooms, check real-time availability, and complete bookings through a secure payment gateway. This ensures that all transactions are conducted safely, maintaining the privacy and security of user information.

A key feature of the website is its user-centric design, which includes easy navigation and access to booking management tools. Guests can modify their reservations, review cancellation and refund policies, and contact customer support directly through the platform. These features are aimed at enhancing the overall user experience by providing transparency and ease of use.

In addition to booking functionalities, the website serves as an informative guide to Kamath Residency. It includes sections dedicated to dining options, local attractions, and wellness services, enabling guests to plan their entire stay comprehensively. The inclusion of customer reviews and testimonials further aids users in making informed decisions regarding their accommodations.

The Kamath Residency Booking Website represents a significant advancement in hotel booking systems, offering a blend of functionality, security, and user engagement. By streamlining the reservation process and providing extensive information about the hotel and its services, the website aims to enhance the guest experience from the point of booking through to the completion of their stay.

## CONTENT

|  |  |  |
| --- | --- | --- |
| **ChapterNo.** | **Chapters Name** | **PageNo.** |
| **1** | **Introduction and Overview**   * 1. Introduction   2. Abstract   3. Problem Statement   4. Objectives   5. Existing System   6. Proposed System   7. Features   8. Language and software tools used | 1-11 |
| **2** | **Software Requirement Specification**  2.1 Introduction  2.2 Need for SRS  2.3 Purpose  2.4 Scope  2.5 Overall Description  2.6 General Constraints   * 1. Specific requirement   2.8 Functional requirement 2.9Performance requirement 2.10 System Attributes. | 12-18 |

|  |  |  |
| --- | --- | --- |
| **3** | **Design of the project**   * 1. System Deign      1. Introduction      2. Overview      3. Logical Design      4. Physical Design      5. Scope and Overview   2. System Overview      1. Context Flow Diagram      2. Data Flow Diagram      3. Entity Relationship Diagram   3. Database Design | 19-34 |
| **4** | **System Testing**   * 1. Introduction   2. Types of Tests 4.3Level of Testing   4.4Test cases | 35-45 |
| **5** | **5.1Appendix** | 46-47 |
| **6** | **Conclusion and Future scope**  6.1Conclusion  6.2 Future Enhancement | 48-58 |
| **7** | **Reference** | 59-60 |

CHAPTER-1

INTRODUCTIONANDOVERVIEW

1. **Introduction and Overview**

### Introduction

Welcome to the Kamath Residency booking website, your one-stop portal for planning and securing your stay at our esteemed hotel. Our user-friendly site offers a seamless experience, allowing you to easily navigate, check real-time room availability, and make secure reservations. Explore detailed descriptions and images of our rooms, and discover our luxurious amenities, dining options, and special packages. We also provide guides to local attractions to enhance your visit. With dedicated customer support available via live chat, email, or phone, your satisfaction is our top priority. Thank you for choosing Kamath Residency.

* 1. **ABSTRACT**

The Kamath Residency Booking Website is an advanced online platform for seamless hotel reservations at Kamath Residency. It offers detailed descriptions of room types, amenities, and services, with real-time availability checks and secure payment options. The user-centric design allows easy navigation, booking management, and direct customer support access. Additionally, the website features information on dining, local attractions, and wellness services, helping guests plan their stay comprehensively. Customer reviews and testimonials provide further insights, ensuring informed decisions. This platform enhances the guest experience by streamlining the booking process and offering extensive hotel information.

### Problem Statement

Kamath Residency faces challenges in optimizing guest experience and operational efficiency due to its reliance on manual booking processes, leading to errors, delays, and guest frustration. The absence of an integrated online booking system and modern payment solutions further exacerbates these issues. Additionally, limited customer engagement strategies and the lack of a CRM system hinder personalized service and repeat business opportunities. The hotel also struggles with data-driven decision-making due to inadequate analytics tools. To overcome these challenges, Kamath Residency plans to implement an integrated booking and management system to enhance operations, guest experience, and customer engagement.

### Objectives

### To Streamline Booking Process: Create a user-friendly platform that simplifies the room reservation process for guests, allowing them to check availability, select rooms, and make bookings online.

### To Increase Revenue: Facilitate online payments through secure payment gateways, encouraging more bookings and boosting the hotel's revenue generation.

### Existing system

The current system at Kamath Residency relies on a static website that provides basic information about the hotel, its amenities, and contact details. This website, while serving as an online presence, falls short in offering an interactive and user-friendly booking experience. The primary functionalities and limitations of the existing system are as follows:

**Limited Functionality**

The static nature of the current website means that it lacks dynamic features that are essential for modern online booking systems. There is no facility for potential guests to check real-time room availability or make reservations directly through the website. Users must contact the hotel via phone or email to inquire about room availability and to make bookings, which is time-consuming and inconvenient.

**Manual Reservation Process**

Reservations are currently managed manually by the hotel staff. This process involves checking room availability in logbooks or spreadsheets, confirming reservations over the phone or email, and manually updating records. This manual handling is prone to errors, such as double bookings or missed reservations, which can lead to guest dissatisfaction and revenue loss.

### Proposed system

To address the limitations of the current static website, the proposed system for Kamath Residency will transform the online experience into a dynamic, user-friendly platform that meets modern standards and enhances operational efficiency. This system will incorporate a range of features designed to streamline the booking process, improve customer satisfaction, and provide valuable insights for management. The key components of the proposed system are as follows:

* **Real-Time Room Availability and Booking**

The proposed system will include a dynamic booking engine that allows potential guests to check real-time room availability directly on the website. This feature will enable users to make instant reservations without the need for phone or email inquiries. The availability calendar will be updated in real-time to reflect current bookings and cancellations, reducing the risk of overbooking.

* **Secure Online Payment Processing**

To facilitate seamless and secure transactions, the new system will integrate with a reliable payment gateway, such as Stripe. This integration will allow guests to pay for their reservations online using various payment methods, including credit/debit cards and digital wallets. Secure payment processing will not only enhance the convenience for guests but also ensure that the hotel receives immediate payment confirmation.

### Features

The proposed system for Kamath Residency incorporates a carefully selected set of features aimed at significantly enhancing the user experience and operational efficiency. The following five key features are designed to address the primary challenges of the current static website:

* **Real-Time Room Availability and Booking**

The new system will feature a dynamic availability calendar, allowing guests to view real-time room availability and make instant reservations. This eliminates the need for manual verification and communication, providing immediate booking confirmations. The seamless process ensures that guests can plan their stays with confidence, knowing that their reservations are secured instantly.

* **Secure Online Payment Processing**

To ensure a smooth and secure transaction experience, the system will integrate with reliable payment gateways like Stripe. Guests will have multiple payment options, including credit/debit cards and digital wallets. All transactions will be encrypted, providing a high level of security and protecting guest data from potential fraud.

* **SMS Notifications**:

To enhance communication and provide better service, the proposed system integrates SMS notification features. Upon successful booking, customers receive a confirmation SMS, ensuring they have immediate proof of their reservation. This feature helps in reducing no-shows and ensures that customers are kept informed about their bookings.

**1.8 Tools and Languages Used**

**1.8.1 Technology Environment**

**1.8.1.1 Laravel 11 Framework:**

Laravel is a widely adopted open-source PHP web application framework that is meticulously crafted to streamline and accelerate web development endeavors. Originated by Taylor Otwell, Laravel introduces an elegant and expressive syntax that adds both enjoyment and efficiency to the process of constructing web applications. Its inclusive set of features equips developers with the tools necessary to create websites and web applications that are not only robust and scalable, but also highly maintainable.

At its core, Laravel operates as a comprehensive web application framework featuring an expressive and elegant syntax. It serves as the bedrock upon which applications are built, alleviating developers from the complexities of intricate coding intricacies. Functioning as a web framework, Laravel provides a structured foundation that allows developers to channel their energy into crafting exceptional projects, while the framework handles the finer details.

Laravel is committed to offering an exceptional developer experience, while simultaneously delivering potent capabilities. These capabilities encompass an extensive array of features, including meticulous dependency injection, a deeply expressive database abstraction layer, support for job queues and scheduled tasks, comprehensive unit and integration testing capabilities, and more.

Important features of Laravel are:

* Configuration in Laravel

The configuration files that drive Laravel's functionality are centralized within the config directory. These files are comprehensively documented, ensuring developers can effortlessly navigate through them and acquaint themselves with the myriad options available. These configuration files empower developers to configure fundamental components, such as database connection specifics, mail server configurations, and core settings like application time zones and encryption keys.

* MVC (Model-View-Controller)

MVC stands for Model-View-Controller, a renowned software architectural pattern extensively employed in web development for structuring applications. The MVC pattern partitions an application into three discrete components, each endowed with distinct responsibilities:

* Model

Responsible for encapsulating the application's data and business logic. The Model handles data management, operations, and enforces business rules. In the context of web applications, it often interfaces with databases for data retrieval, updates, and manipulation.

* View

The View assumes responsibility for presenting data to users and managing user interface interactions. This component manages the visual presentation of information and facilitates user engagement. Views are commonly in the form of templates that format data from the Model into an accessible format for users.

* Controller

Serving as a mediator between the Model and the View, the Controller receives user input from the View, processes it, interacts with the Model to enact actions, and subsequently updates the View with the results. Controllers orchestrate the flow of the application, determining actions based on user interactions.

* Artisan CLI

Artisan is a command-line tool that offers a range of commands to automate repetitive tasks. Developers can use it to create controllers, models, migrations, and more. This streamlines development and reduces the need for manual coding, boosting productivity.

* Eloquent ORM

Laravel's Eloquent is a powerful Object-Relational Mapping (ORM) system that simplifies database interaction by providing an expressive and intuitive way to work with database records. Eloquent supports relationships, querying, and database schema management.

* Database Interaction

Laravel's Eloquent ORM is a powerful feature that simplifies database interactions. It lets developers work with databases using an intuitive object-oriented syntax, minimizing the need for writing raw SQL queries. This abstraction enhances code readability and reduces the potential for SQL injection vulnerabilities.

* Blade Templating

Blade is a templating engine built into Laravel. It allows developers to create dynamic and reusable views using straightforward template files. Blade templates can include control structures, conditionals, and loops, helping developers maintain a clear separation between PHP logic and presentation.

* Routing

Laravel's routing system is responsible for mapping URLs to specific actions within the application. This makes defining and managing routes efficient and organized. With named routes and route groups, developers can easily handle different types of requests and manage complex routing scenarios.

* Middleware

Middleware functions as a series of filters that can be applied to incoming HTTP requests. It's a powerful tool for implementing cross-cutting concerns like authentication, authorization, and request transformation. Middleware intercepts requests before they reach the application logic, enhancing security and allowing for preprocessing.

* Authentication and Authorization

Laravel simplifies user authentication with built-in authentication scaffolding. Developers can easily implement user registration, login, and password reset functionality. The framework also provides tools for handling role-based authorization, ensuring that users have the appropriate access levels to different parts of the application.

* Testing Support

Laravel's testing capabilities contribute to code quality and application stability. The framework supports various types of tests, including unit tests to isolate individual components, integration tests to verify interactions between components, and feature tests to test application behavior from a user's perspective.

**1.8.1.2 PHP:**

PHP is an acronym for "PHP: Hypertext Preprocessor". PHP is a widely-used, open-source scripting language. PHP scripts are executed on the server. PHP is free to download and use. It is powerful enough to be at the core of the biggest blogging system on the web (WordPress). It is deep enough to run large social networks. It is also easy enough to be a beginner's first server-side language. PHP files can contain text, HTML, CSS, JavaScript, and PHP code. PHP code is executed on the server, and the result is returned to the browser as plain HTML.

PHP files have extension ".php".PHP can generate dynamic page content. It can create, open, read, write, delete, and close files on the server. PHP can collect form data. It can send and receive cookies. PHP can add, delete, modify data in your database, can be used to control user-access. PHP can encrypt data With PHP you are not limited to output HTML. You can output images, PDF files, and even Flash movies. You can also output any text, such as XHTML and XML. PHP runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.). PHP is compatible with almost all servers used today (Apache, IIS, etc.). PHP supports a wide range of databases. PHP is free.

PHP 5.6 was the last release branch in the PHP 5 series of releases. It brought in a number of new features, as well as deprecations in preparation for the 7.0 release. PHP 5.6 marked the end of the road for PHP 5 versions and paved the way for PHP 7. Alongside some notable deprecations, it also introduced a handful of new features and a modest performance improvement over PHP 5.4. However, at over 6 years old, the release is out of long-term support and has more than a handful of critical vulnerabilities and exceptions worth mentioning.

PHP 7 is a major release of PHP programming language and is touted to be a revolution in the way web applications can be developed and delivered for mobile to enterprises and the cloud.

Important features of PHP are:

* Simplicity

PHP is particularly famous for its simplicity. It is organized and easy to learn. Even beginners won’t face any hard time learning and using PHP. It is a very well-organized programming language, and it comes with a lot of pre-defined functions, which makes the task of the programmer easy. There is no need to include libraries in PHP like C. With a lot of pre-defined functions, PHP is easy to optimize as well.

* Flexibility

PHP scripts can run on any device- mobile, tablet, or PC. It is very compatible with various databases. It can be easily embedded and integrated into HTML, XML, and JavaScript. Likewise, it is also compatible with almost all servers used today like Apache, IIS, etc.

* Objectiveoriented

PHP supports object-oriented programming features like data encapsulation, inheritance, abstraction, polymorphism, etc. The Object-oriented programming feature was added in PHP5. This feature helps in building complex reusable web pages and makes PHP comparable to powerful object-oriented languages like Java and Python.

* Interpretedlanguage

PHP is an interpreted language, which means there is no need for compilation. Interpreters run through a program line by line and execute the code. Since interpreters execute source code themselves, the code becomes platform-independent. Some other benefits of interpreted language include dynamic typing and short executable program size.

* Efficient

PHP is a versatile, reliable, and efficient programming language. The memory management of PHP is very efficient. Great session management, eliminating unnecessary memory allocation, are some of the features that make PHP efficient.

* FastPerformance

PHP scrips are usually faster than other scripting languages. Users can load their web pages faster, and they love it. PHP code runs faster than most of programming languages because it runs in its own memory space. Not only that, but its connection with databases is also fast and efficient.

* Freeandopen**-**source

PHP is open-source, which means it can be downloaded and used freely. There is absolutely no hassle to acquire a license to use it and no payment is required to use it, so it is kind to your pocket too!

* Case-sensitive

PHP is a partially case-sensitive language. Although functions names are not case-sensitive, other things in PHP are case-sensitive. The following things in PHP are case-sensitive:

* Variable names
* Constructs (if, if-else, if-elseif, while, do-while)
* Keywords (such as true and false)
* User-defined functions and class names
* Security

PHP has many pre-defined functions for data encryption. Users can also use third-party applications for security. Security and flexibility are often contrasting features, but PHP somehow manages to offer them both, and that’s great. PHP is designed specifically to be a more secure language for writing CGI (Computer-generated Imagery) programs. Security algorithms such as Sha1 (secure Hash algorithm 1) and MD5(Message digest 5) are used to encrypt the strings in PHP. Filter Var and strip tags functions help to keep the environment more secure and safe for users.

* Platformindependent

We can run PHP on any device and operating system (Microsoft Windows, macOS, Linux, RISC OS, or Unix). We can easily connect it with various databases and is also compatible with almost all web servers used today (Apache, IIS, and others). It supports a wide range of databases as well. Its cross-platform compatibility makes really popular among its users as it saves a lot of time and energy.

**1.8.1.3 HTML & CSS**

HTML is the standard markup language for Web pages. With HTML you can create your own websites. HTML describes the structure of a Web page. It consists of a series of elements HTML elements tell the browser how to display the content. HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc. Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes, and other items. HTML elements are delineated by *tags*, written using angle brackets. Tags such as <img/> and <input/> directly introduce content into the page. Other tags such as <p> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags but use them to interpret the content of the page.

HTML can embed programs written in a scripting language such as JavaScript, which affects the behaviour and content of web pages. The inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), former maintainer of the HTML and current maintainer of the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997. A form of HTML, known as HTML5, is used to display video and audio, primarily using the <canvas> element, in collaboration with JavaScript.

**1.8.1.4 JavaScript:**

**JavaScript** (often shortened to **JS**) is a lightweight, interpreted, object-oriented language with first-class functions, and is best known as the scripting language for Web pages, but it's used in many non-browser environments as well. It is a prototype-based, multi-paradigm scripting language that is dynamic, and supports object-oriented, imperative, and functional programming styles. JavaScript runs on the client side of the web, which can be used to design / program how the web pages behave on the occurrence of an event. JavaScript is an easy to learn and also powerful scripting language, widely used for controlling web page behaviour.

Contrary to popular misconception, **JavaScript is** not **"Interpreted Java"**. In a nutshell, JavaScript is a dynamic scripting language supporting prototype based object construction. The basic syntax is intentionally similar to both Java and C++ to reduce the number of new concepts required to learn the language. Language constructs, such as if statements, for and while loops, and switch and try ... catch blocks function the same as in these languages (or nearly so). JavaScript can function as both a procedural and an object oriented language. Objects are created programmatically in JavaScript, by attaching methods and properties to otherwise empty objects **at run time**, as opposed to the syntactic class definitions common in compiled languages like C++ and Java. Once an object has been constructed it can be used as a blueprint (or prototype) for creating similar objects.

**1.8.1.5 XAMPP:**

XAMPP is an abbreviation where X stands for Cross-Platform, A stands for Apache, M stands for MYSQL, and the P stands for PHP and Perl*,* respectively. It is an open-source package of web solutions that includes Apache distribution for many servers and command-line executables along with modules such as Apache server, MariaDB, PHP, and Perl. XAMPP helps a local host or server to test its website and clients via computers and laptops before releasing it to the main server. It is a platform that furnishes a suitable environment to test and verify the working of projects based on Apache, Perl, MySQL database, and PHP through the system of the host itself. Among these technologies, Perl is a programming language used for web development, PHP is a backend scripting language, and MariaDB is the most vividly used database developed by MySQL

# CHAPTER-2

SOFTWARE REQUIREMENT SPECIFICATION

## 2. Software Requirement Specification

### Introduction

Requirement is defined as a condition needed by a user to solve a problem or achieve an objective; a condition or capability that must be met or possessed a system to satisfy a contract, standard, specification, or other formally imposed document. The software requirements specification (SRS) describes what the proposed software should do without describing how the software will do it. It has a document which completely describes the external behavior of the software. It is the first job of the software developer to study the system that needs to be developed and specifies the user requirement before going for the designing part.

### 2.2 Need for SRS

The origin of most software system is in need of the client, who either wants to automate an existing manual system or describes a new software system. The software system itself is created by the developer. Finally, the complete system will be used by the end users. There are three major parties interested in new system; the client, the user and the developers. The requirements for the system, that will satisfy the needs of the client, the users and the developers.

* SRS is the medium through which the client and the user needs are specified.
* SRS establishes the basis for agreement between the client and the supplier on what the software product will do.
* An SRS provides reference for validation of the final product.
* A high-quality SRS is prerequisite to high quality software.
* A high-quality SRS reduces the development cost.

### Purpose

As the world more competitive, you have to sharpen your tools. Knowing what is on the customer mind is the more important thing we can do. What you want and what your customers want may not be the same, but unless you give them what they want, you won't get what you want.

The basic purpose of SRS is to bridge the communication gap between client and developer. It provides user friendly environment. This document contains information required for the developer of the software, clients and users of the software. Another important purpose is helping the client to understand their own needs.

* 1. **Scope of the Project**

The scope of the Kamath Residency booking system project encompasses the development and deployment of a dynamic, user-friendly web application to facilitate room bookings at Kamath Residency. The project aims to replace the existing static website with an interactive platform that enhances user experience and operational efficiency. By transitioning to a more responsive and feature-rich website, the project seeks to improve customer engagement and streamline the booking process for both users and hotel staff.

A key objective of the project is to implement a secure user registration and authentication system. This feature will allow users to create accounts, log in, and manage their bookings with ease. By providing a personalized user experience, the system will encourage repeat visits and foster customer loyalty. Additionally, the project will focus on developing a real-time room availability checker and booking system. Users will be able to view available rooms, select their desired dates, and make reservations seamlessly, ensuring a smooth and hassle-free booking experience.

* 1. **Overall Description**

In this section, background information about what type of requirement system should have will be provided briefly.

### 2.5.1ProductPerspective

* This is the special application according to the requirement of users.
* This software is used to book rooms online.

### 2.5.2ProductFunction

* It provides users with a solid platform to book rooms in Kamath residency.
* It allows online payment as well.

**2.5.3 System Interface**:

This project helps users to book rooms.The project employs Laravel PHP for the front end, and data will be stored in a MYSQL database.

### General Constraints

This application is not hosted in browser.

### 2.6.1Assumptions and Dependencies

### Information stored in the database can be retrieved at any time.User receives sms when he/she books a room or cancels a booking.

### Specific Requirements

* + 1. **External Interface Requirement**

This application has striking features that contains graphical user interface. So, any person who is authorized to access this system can easily understand what to do next when he is operating with.

### 2.7.1.1UserInterface

User Interface is designed in user friendly manner. The labels are used on the view controller to direct the user of what are the controls are used for textboxes, text area, dropdown list and buttons are used for input into the pages. Date picker, auto complete textboxes are also used for inputting the information. The system is using swift for the user benefit.

### Hardware Interface and Software Interface

Minimum Software Requirements

|  |  |
| --- | --- |
| Operating System | Windows 11 |
| Platform | Laravel PHP |
| Programming Language | HTML,CSS,JQUERY,JAVASCRIPT |
| Connectivity | HTTP |
| Processor | I5 or Ryzen5 Processor or above |

### Functional Requirement

Functional Requirements essentially are the input/output behavior specification for the software. Functional requirements are those that refer to the functionality of the system i.e., what services it will provide to the user. Non-functional requirement pertains to other information needed to produce the correct system and are detailed separately. The software must provide user interface to accept the valid details and display the information.

### Module Description

1. Administrator module
2. User module

**1. Administrator Module**

**Add Room:** Administrators can add new rooms to the system, specifying details such as room type, description, price, and availability. This ensures that the latest offerings are always available for users to book.

**View and Manage Bookings**: Admins have access to a comprehensive list of all bookings. They can view details of each booking, manage reservations, and update statuses as needed, ensuring efficient oversight of the booking process.

**Delete Bookings**: Administrators have the authority to delete bookings that may be invalid or canceled, maintaining an accurate and up-to-date booking system.

**Delete Rooms:** Admins can remove rooms from the system if they are no longer available for booking. This feature helps in keeping the room inventory current and relevant.

**Booking Graphs**: The admin panel includes a graphical representation of room bookings, allowing admins to view and analyze booking trends for specific months of a specific year. This feature provides valuable insights for strategic decision-making and resource allocation.

**2.User Module**

**View Available Rooms**: Users can search and view available rooms for specific dates, providing them with real-time information on room availability to make informed booking decisions.

**Book Room**: Users can book rooms by selecting their desired room type, dates of stay, and providing necessary personal details. This streamlined process ensures a user-friendly booking experience.

**Make Payments**: The system integrates a secure payment gateway, allowing users to complete their bookings with online payments. This ensures a seamless and convenient payment process.

**Receive SMS Notifications**: Upon successful booking, users receive SMS notifications confirming their reservation details. This feature enhances communication and provides users with immediate confirmation.

**Cancel Bookings**: Users have the option to cancel their bookings if needed. The system handles cancellations based on the defined cancellation policy, providing transparency and flexibility to users.

### Performance Requirement

* The performance of the overall system should be fast.
* The system has to be user friendly.
* Backup System: To store and protect the database we should have backup system.
* Security: Authorized user who knows the valid id and password can use the admin software.

### System Attribute

* + 1. **Adaptability**

A real time use of the Application is possible.

### Correctness

All activities of this project are particular method.

### Maintainability

Evolve to meet the changing needs of users.

### Dependability and Security:

Includes security and safety. Should not cause any physical or economic damage incase of system failure.

### Efficiency:

Includes responsiveness, processing time and memory utilization.

### Acceptability:

Acceptable to the type of users for which it is designed.

### Portability:

This software works in any browser.

### Testability:

Checking for the working of the application for the required output is possible.

### Designed Constraints

Software constraint:The software runs in IOS.

CHAPTER-3

DESIGN OF THE PROJECT

## 3.Design of the Project

### 3.1System Design

#### 3.1.1 Introduction

System design is the process or art of defining the architecture, components, and modules. Interface and the data for a system to satisfy specified requirements. System design is also called as top-level design, in which the focus is on deciding which modules is need for the system.

#### 3.1.2 Overview

System design is therefore the process of defining and developing a system to satisfy specified requirements of the user. Until the 1990’s standardization of hardware and software resulted in the ability to build modular systems. The increasing the importance of the software running on the generic platform has enhanced the discipline of software engineering. Object oriented analysis and design methods are becoming the most widely standard language used in the object-oriented analysis and design. It is widely used for modeling software system and it is increasingly used for big designing non software system and organizations.

#### Logical Design

The logical design of the system pertains to an abstract representation of data flows, inputs, and outputs of the system. This is often conducted via modeling, which involves a simplistic representation of an actual system.

### Physical Design

The physical design relates to actual input and output process of the system. This is laid down in terms of how data is imputed into a system, how it is verified physical design in this context does not refer to the tangible physical design of an information system. To use an analog, a personal computer’s physical design involves input via keyboard, processing within the CPU, and output via monitor, printer etc.

### Scope and Overview

**3.1.5.1 Scope**

The scope of the project is from giving the constraint and accessing the data and represents it in graphical form. It helps the user to understand the data. To do this we find the solution using collected requirements and design this solution. Design is done in this phase. This phase is done in correct way because it effects the next phases also.

### 3.1.5.2 Overview

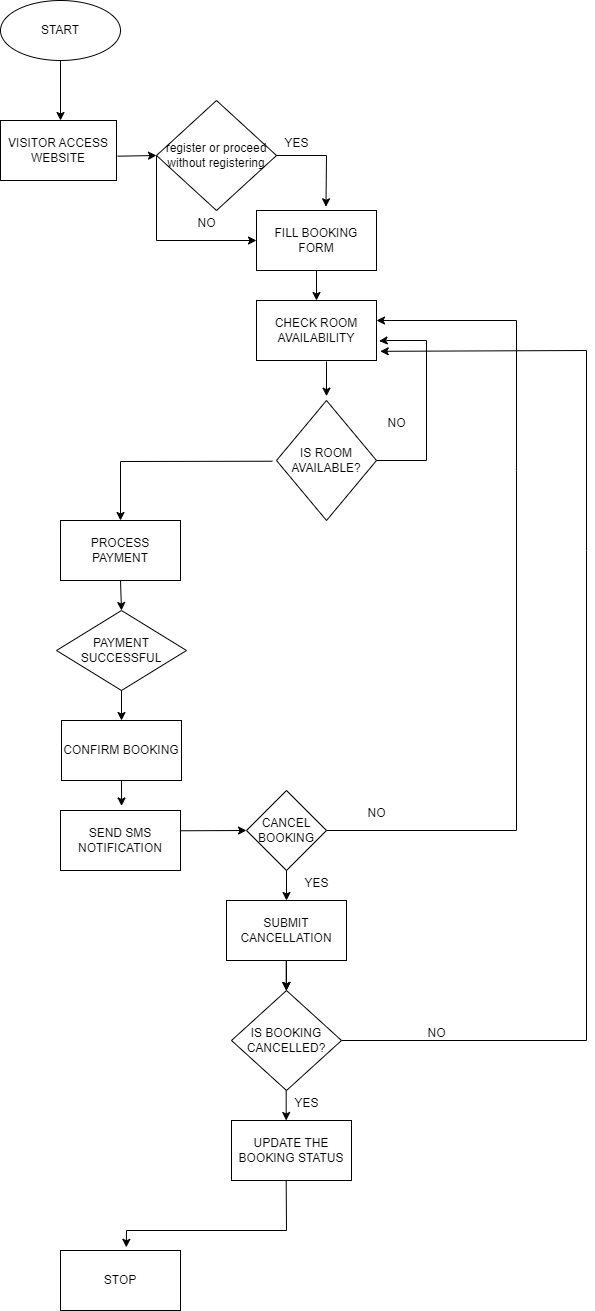
The software requirement specification step of a software development process yields specifications that are used in software engineering. If the system is semi-automated or user centered, software design may involve user experience design yielding a story board to help determine those specifications. If the software is completely automated, a software design may be as a flowchart or text describing a planned sequence of events. There is also semi standard method like Unified Modeling Language and Fundamental modeling concepts. In either case some documentation of the plan is usually the product of the design. A software design may be platform independent or platform specific, depending on the availability of the technology called for by the design.

### System Overview

Mining the user information and represent it in the graphical form is the main functionality of this software. We develop an application which working on the concept of data mining and satisfies the user constraints.

A flowchart is a type of diagram that represents a workflow or process. A flowchart can also be defined as a diagrammatic representation of an algorithm, a step-by-step approach to solving a task. A simple flowchart representing a process for dealing with a non-functioning lamp. The flowchart shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows. This diagrammatic representation illustrates a solution model to a given problem. Flowcharts are used in analysing, designing, documenting or managing a process or program in various fields.

|  |  |  |
| --- | --- | --- |
| SYMBOL | PURPOSE | DESCRIPTION |
|  | Flow line | Used to indicate flow of logic by connecting symbols. |
|  | Terminal(start/stop) | Used to represent start and end of flowchart. |
|  | Processing | Used for athematic operations and data manipulation. |
|  | Decision | Used to represent the operation in which there are two alternatives, true and false.  **Table 3.2.1Notification for Flowchart** |

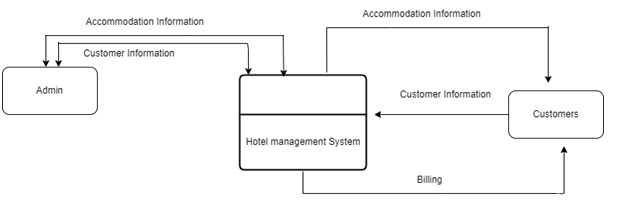


**Fig 3.2.1 Flow Chart**

### Context Flow Diagram(CFD)

A context flow diagram defines the boundary between the system, or part of the system, and its environment, showing the entities that react with it. It represents all the external entities that may interact with the system. In context flow diagram, the entire system is treated as a single process and all input, output, sinks and source of the process are shown. Using CFD Design methodology DFD of the system is represented. The environment in which (the context) the software is used is depicted in CFD.

The CFD shows the external entity acting on the software. The CFD shows the external entity on the software. The software is shown here in CFD as a single process. The best CFD’s are used to display how a system interoperates at a very high level, or how systems operate and interact logically.



**Fig 3.2.1.1Context Flow Diagram**

### Data Flow Diagram(DFD)

Data flow diagram are also called Data Flow Graph or Bubble chart which is commonly used during problem analysis. A DFD shows the flow of data through a system as a function that transforms the inputs into desired outputs.

The process in DFD is represented by named circle and data flow are represented by named arrows entering or living the bubbles. A source or sink is typically outside the main system of study.

The two main purpose of DFD are:

* To provide an indication of flow of data are transformed as they move through the system.
* To depict the function that transforms the flow.

The DFD provides additional information that is used during the analysis of the problem domain and servers as a basic for modelling functions. The DFD may be used to represent a system or software at any level of abstraction. In fact, DFD provides a mechanism for functional modelling as well as information flow modelling.

A level 0 DFD also called fundamental system model or a context model representing the entire software elements as a single bubble with inputs and output data indicated incoming and outgoing arrows respectively. Additional process and information flows are represented as level 0 DFD, which is partitioned to reveal more details. For example: a level 0 DFD might contain four or five bubble with interconnecting arrows. Each of the processes represented level 1 is sub function of overall system depicted in the context model.

The basic notation used to create a DFD makes is easy to analyze and understand. The DFD is graphical tool that can be very valuable during software requirement analysis.

**3.2.2.1 Notation for DFD-diagram**

**Table 3.2.2.1 Notification for DFD-diagram**

|  |  |
| --- | --- |
| **ENTITY:** | External entities are outside the system, but they either supply input data into the system or use system output. These represented by rectangle. It is used for specifying from data where data comes and where it reaches. |
| **PROCESS**: | A process shows a transformation or manipulation of data flows within the system. A process transforms incoming data flow into outgoing data flow. |
| **DATAFLOW**: | A data flow shows flow of information from source to destination. A data flow is represented by a line, with arrow head showing the direction of flow. |
| **DATABASE**: | Databases are outside the system, is used to store the data in backend. |
| **TABLES**: | Tables are the part of databases. Used to store the data. |

**Fig 3.2.2.1 DFD For Admin**

### Database Design

* + 1. **Data Object Design**

Database design is required to manage the large bodies of information. The management of data involves both the definition of structure of the storage of information and provisions of mechanism for the manipulation of information. In addition to the dB system must provide for the safety of information handled, despite the system crashes due to attempts at unauthorized access. For developing an efficient dB, we will have to fulfil certain condition such as:

* Control Redundancy
* Ease of use
* Data independence
* Accuracy and integrity

There are 5 major steps in designing process.

* Identify the table and relationship
* Identify the data that is need for each table and relationship.
* Resolve the relationship
* Verify the design
* Implement the design

**3.3.2 Table Structure**

The Utility and Maintenance System for Merchant Success Web Application uses following tables as fields.

**User**

|  |  |  |  |
| --- | --- | --- | --- |
| **Slno** | **Column Name** | **Data Type** | **Constraints** |
| 1 | id | bigint(20) | Primary Key |
| 2 | room\_id | varchar(255) | Unique |
| 3 | name | varchar(255) | Not Null |
| 4 | email | varchar(255) | Not Null |
| 5 | phone | varchar(255) | Not Null |
| 6 | payment\_status | varchar(255) | Not Null |
| 7 | status | varchar(255) | Not Null |
| 8 | start\_date | varchar(255) | Not Null |
| 9 | end\_date | varchar(255) | Not Null |
| 10 | amount | decimal(10,2) | Not Null |
| 11 | refund\_amount | decimal(8,2) | Not Null |
| 12 | created\_at | timestamp | Not Null |
| 13 | updated\_at | timestamp | Not Null |

**Table 3.3.2.1User Table**

**Room**

|  |  |  |  |
| --- | --- | --- | --- |
| **Slno** | **Column Name** | **Data Type** | **Constraints** |
| 1 | id | bigint(20) | Primary Key |
| 2 | room\_title | varchar(255) | Not Null |
| 3 | description | text | Not Null |
| 4 | price | int(11) | Not Null |
| 5 | wifi | tinyint(1) | Null |
| 6 | room\_type | varchar(255) | Null |
| 7 | created\_at | timestamp | Null |
| 8 | updated\_at | timestamp | Null |

**Table 3.3.2.2Room Table**

**Room image**

|  |  |  |  |
| --- | --- | --- | --- |
| **Slno** | **Column Name** | **Data Type** | **Constraints** |
| 1 | id | bigint(20) | Primary Key |
| 2 | created\_at | timestamp | Not Null |
| 3 | updated\_at | timestamp | Not Null |
| 4 | room\_id | bigint(20) | Foreign Key |
| 5 | image\_path | varchar(255) | Null |

**Table 3.3.2.3Room image Table**

**User Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **Slno** | **Column Name** | **Data Type** | **Constraints** |
| 1 | id | bigint(20) | Primary Key |
| 2 | name | varchar(255) | Not Null |
| 3 | email | varchar(255) | Not Null |
| 4 | phone | varchar(255) | Not Null |
| 5 | usertype | varchar(255) | Not Null |
| 6 | email\_verified\_at | timestamp | Not Null |
| 7 | password | varchar(255) | Not Null |
| 8 | two\_factor\_reset | text | Not Null |
| 9 | two\_factor\_recovery\_code | text | Not Null |
| 10 | two\_factor\_verified\_at | timestamp | Not Null |
| 11 | remember\_token | varchar(100) | Not Null |
| 12 | current\_team\_id | bigint(20) | Not Null |
| 13 | profile\_photo\_path | varchar(2048) | Not Null |
| 14 | created\_at | timestamp | Not Null |
| 15 | updated\_at | timestamp | Not Null |

**Table 3.3.2.4User Table**

**Entity Relationship (ER) Model**

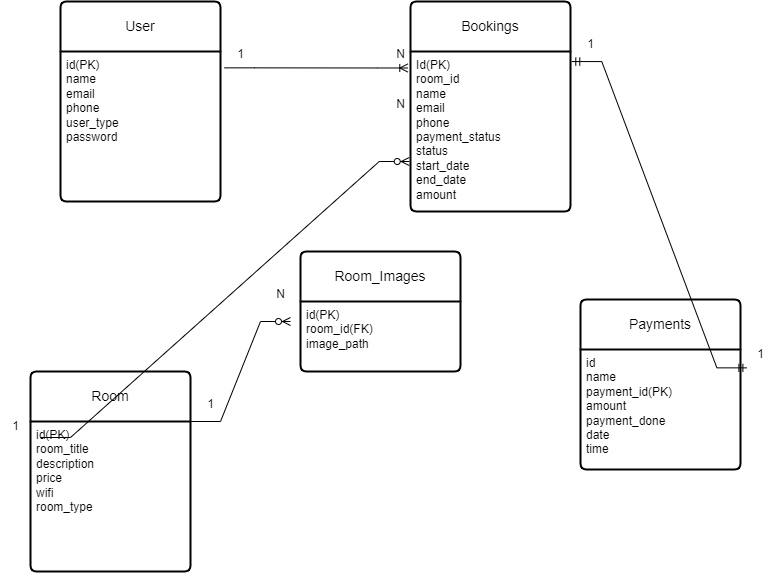
Entity Relationship model is popular high-level conceptual data model. This model d its variations are frequently used for the conceptual design of the database applications d many databases design tools employee its concepts. We described the basic data structuring concepts and constraints of the model and discuss their use in the design of conceptual; schemas for database application.

The basic units in an ER model are entities, their attributes, entity types and relationships between entity types.

* Entities: An entity represents an object defined within the information system about which you want to store information. An entity may be defined as a thing which is recognized as being capable of independent existence and which can be uniquely identified. An entity is an abstraction from the complexities of domain.
* Entity Attributes: Attributes are elementary pieces of information attached to entity. It is the entities where the number of attributes that can be used to describe them potentially vast, but the number that will actually apply to a given entity is relatively modest.
* Relationships: A relationship is a named connection or association between entities. A relationship captures how entities are related to one another.
* Weak Entity: Weak entity is an entity that cannot be uniquely identified by its attributes alone; therefore, it must use a foreign key in conjunction with its attributes to create a primary key.
* Key Attribute: An entity usually has an attribute whose values are distinct for each individual entity. This attributes uniquely identifies the individual entity. Such an attribute called a key attribute.

|  |  |
| --- | --- |
| **Components** | **Description** |
|  | A rectangle or process symbol is generally used to represent entity. |
|  | The diamond symbol is used to show relationships between entities. |
|  | The terminator or oval symbol is used to define the attributes of the entity. |
|  | The double oval symbol is used to define the multi valued attribute of the entity. |
|  | Weak entity |

**Table 3.3.2.11Notification for ER Model**

****

**Fig 3.3.2.3ER Diagram**

CHAPTER-4

TESTING

## System Testing

### Introduction

Testing is the major quality control measure used during software development. Its basic function is to detect errors in the software. Thus, the goal of testing is to uncover requirement, design and coding errors in the program. Testing is the phase where the errors remaining from all the previous phase must be detected. Hence testing performs the critical role for the quality assurance and ensuring the reliability of software. During testing, the program to be tested is executed with a set of test cases and the output of program for the test case is evaluated to determine if the program is performing as expected.

### 4.1.1TestStrategy

The testing strategy used will focus on reliability and performance of the product. Thisincludesthe reportingclient hardware, software,management.

The foundation for the success of this product is the Good Bandwidth. No matter how reliablethesoftwareif thebandwidth is not good the websitewill not work.

The next key to success is the performance of the website both the reporting user and remoteserver. Most operations must perform at a sub second performance level. This does not meanthat the operation is completed in a sub second although every effort should be done actuallyhit the task assign metric rather that the user sees the system respond in some fashion within asecond.

The final key is that the internal and scheduled processes operator reliable over long stretchesof time with limited user intervention. In order to grade the quality of the implementation, thetesting will focus on grading the reliability and performance characteristics of all the aspectsof the product. The system will have to pass multiple repeated testing and hit specific metricagreed upon.Failure to meet the metric will mean that the product is not implementedcorrectlyand must be corrected.

### Typesoftesting

“Program testing can be used to show the presence of bugs, but never to show their absence! “Clearly, the success of testing is revealing errors in programs depend critically on the test cases.

The 2 basic approaches are

1. Blackboxorfunctionaltesting.
2. Whiteboxorstructuraltesting.

### Blackbox orfunctional testing

Black box testing is also known as functional testing. A software testing technique wherebythe internal workings of the item being tested are not known by the tester. For example, in ablack box test on software design the tester only known’s the inputs and what the expectedoutcomes should be and not how the program arrives at those outputs. The testers everexamine the programming code and do not need any further knowledge of the program otherthanits specifications.

### Whiteboxorstructural testing:

Whiteboxtestingincludesanalyzingdataflow,controlflow,informationflow,codingpracticesandexceptionanderrorhandlingwithinthesystem,totesttheintendedandunintended software behavior. White box testing can be performed to validate whether codeimplementation follows intended design, to validate implemented security functionality, andto uncover exploitable vulnerabilities. White boxtesting requiresaccess to source code.Though White box testing can be performed any time in the life cycle after the code isdeveloped;it is agood practiceto perform Whiteboxtestingduringtheunit testingphase.

### DifferentLevelsoftesting

During testing process different levels of testing are used. Each level of testing aiming to test differentaspects of thesystem. Thethreedifferent levels of testingare

### Unittesting:

Unit testing focuses verification efforts on the smallest unit of software i.e., the module. Usingdetailed design and the process specification testing is done to uncover errors within theboundaryof themodule.

All modules must be successful in the unit test before the start of the integration testingbegins. Here different modules are tested against the specification produced during design forthemodule.It is necessaryto verifythecodewritten during thecodingphase.

### Integrationtesting

After the unit testing, we have to perform the integration testing. The goal here is to see ifmodulescanbeintegratedproperly,theemphasisbeingontestinginterfacesbetweenmodules. This testing activity can be considered as testing the design and hence the emphasison testing module interaction. During this testing many units tested modules are combined intosubsystems which arethen tested.

ThefollowingarethetypesofIntegrationTesting:

* + - * Top-DownTesting
      * Bottom-UpTesting

### 4.3.2.1 Top-downIntegration

This method is an incremental approach to the construction of programstructure. Modulesare integrated by moving downward through the control hierarchy, beginning with the mainprogram module. The module subordinated to the main program module is incorporated intothestructurein either adepth first or breadth firstmanner.

### 4.3.2.2 Bottom-UpIntegration

Thismethodbeginstheconstructionandtesting the modulesatthe lowestlevelintheprogram structure. Since the modules are integrated from the bottom-up processing requiredfor modules subordinated to a given level is always available and the need for stubs iseliminated.

Thebottom-upintegration strategiesmaybeimplementedwiththefollowingsteps:

* + - * Thelow-levelmodulesarecombinedintoclustersthatperformaspecificsoftwaresub-function.
      * Adriver(i.e.)thecontrolprogramfortestingis writtento coordinatetest case
      * Inputand output.
      * Thecluster istested.
      * Driversare modulatedandclustersarecombinedmoving upwardintheprogramstructure.

### Systemtesting

Systemtestingensuresthattheentireintegratedsoftwaresystemmeetsrequirements.Ittestsa configuration to ensure known and predictable results. An example of system testing is theconfiguration-orientedsystem integration test.

### Acceptancetesting

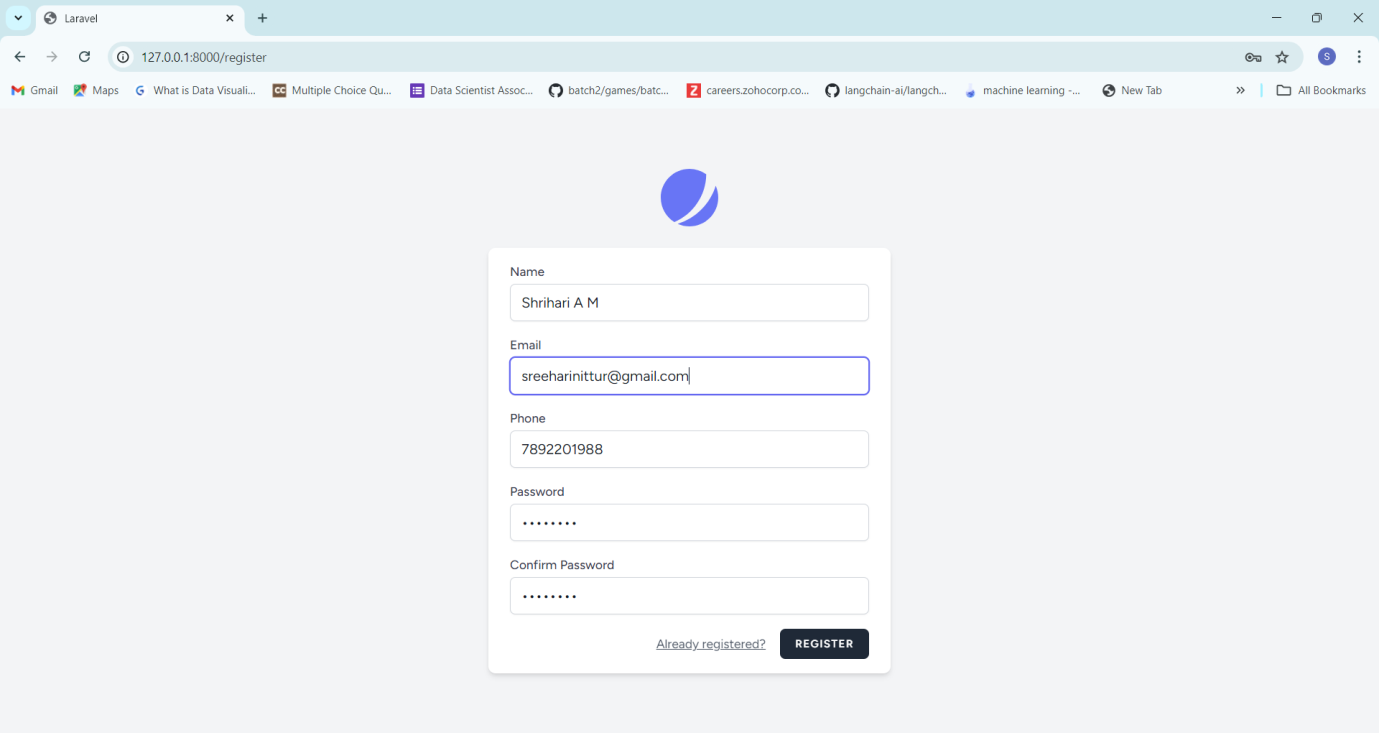
UserAcceptanceTestingisacriticalphaseofanyprojectandrequiressignificantparticipationbytheenduser.Italsoensuresthatthesystemmeetsthefunctionalrequirements.

### Software Test Report

### Register for User

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl.No** | **TestCase** | **ExpectedResult** | **ObtainedResult** |
| 1 | Email validation | Valid email. | Successful |

**Table 4.4.1 Register for User**

****

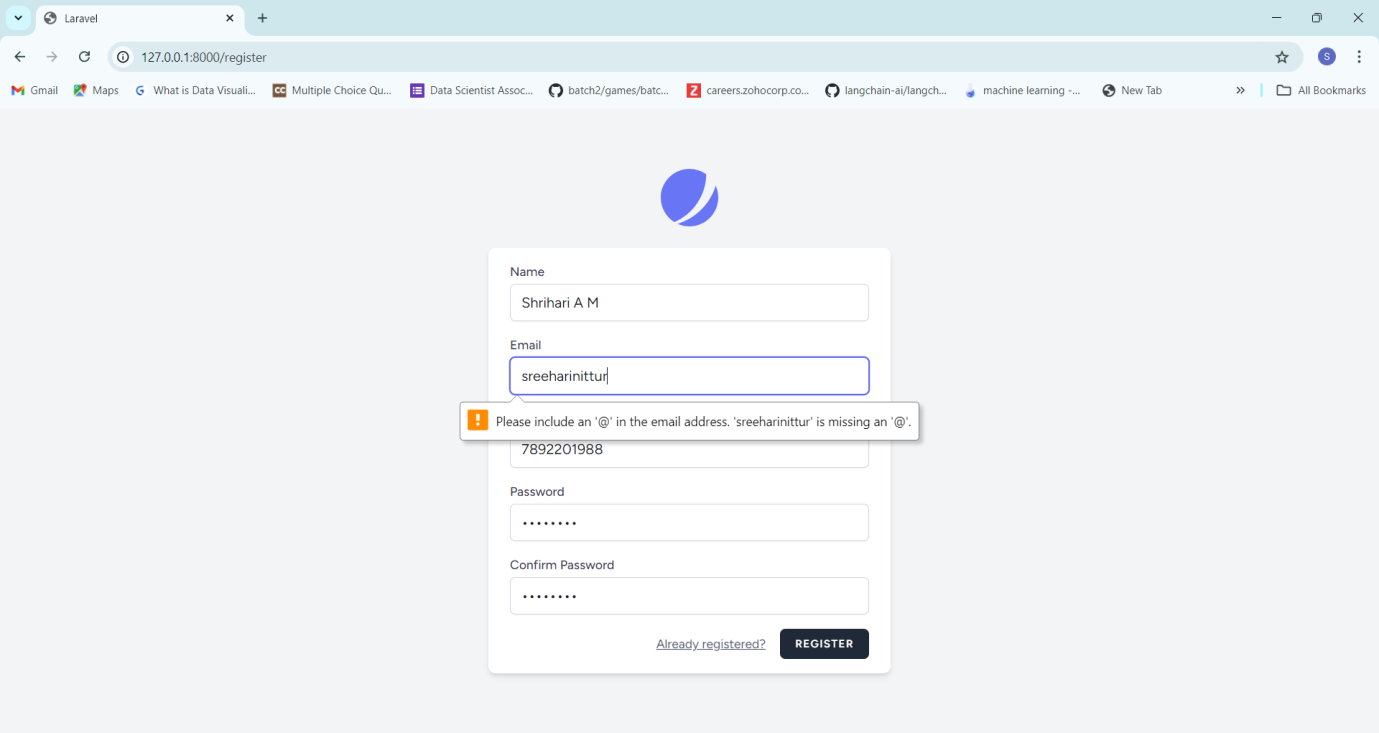
**Fig 4.4.1Register for User**

**Table 4.4.1 Register for User**

### Register for User

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl.No** | **TestCase** | **ExpectedResult** | **ObtainedResult** |
| 2 | Email validation | Please include an ‘@’ in the email address. | Successful |

**Table 4.4.2 Register for User**

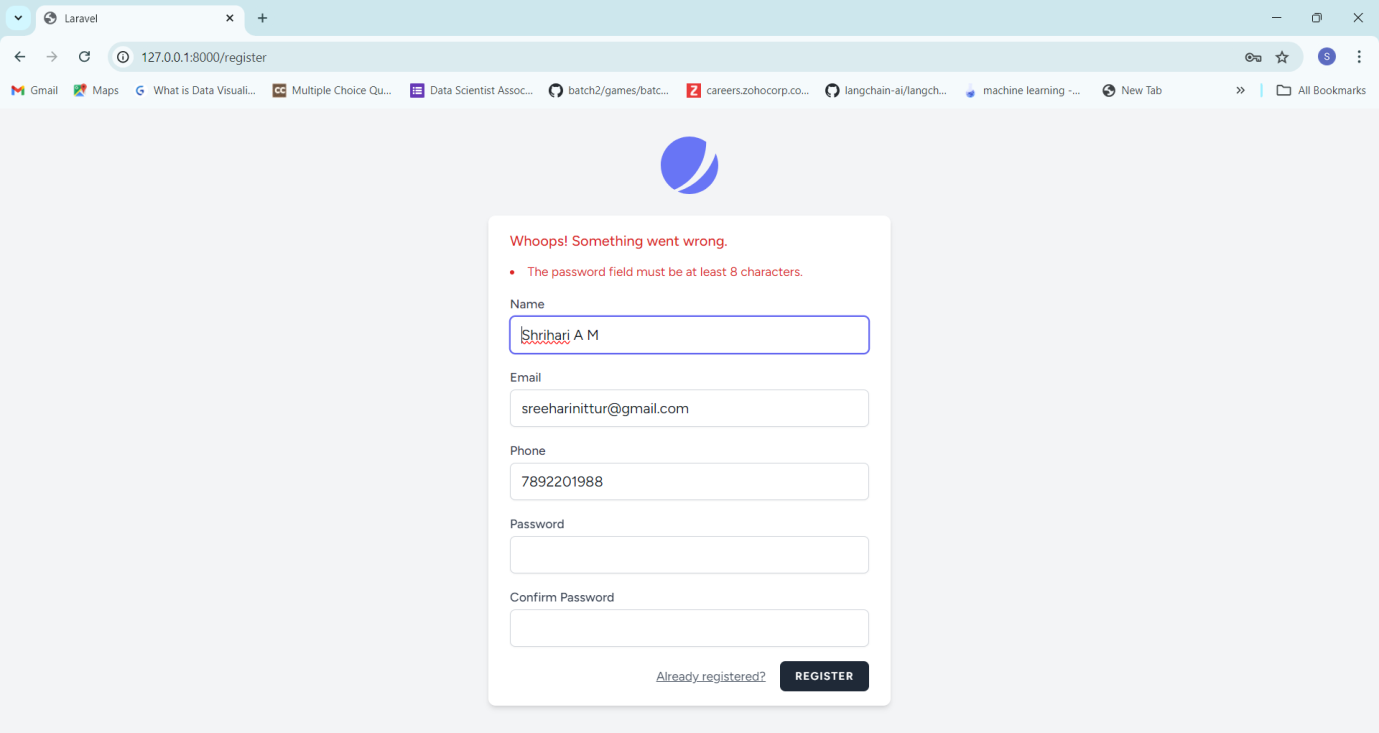
****

**Fig 4.4.2Register for User**

### Register for User

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl.No** | **TestCase** | **ExpectedResult** | **ObtainedResult** |
| 3 | Password validation | The password field must be at least 8 characters. | Successful |

**Table 4.4.3 Register for User**

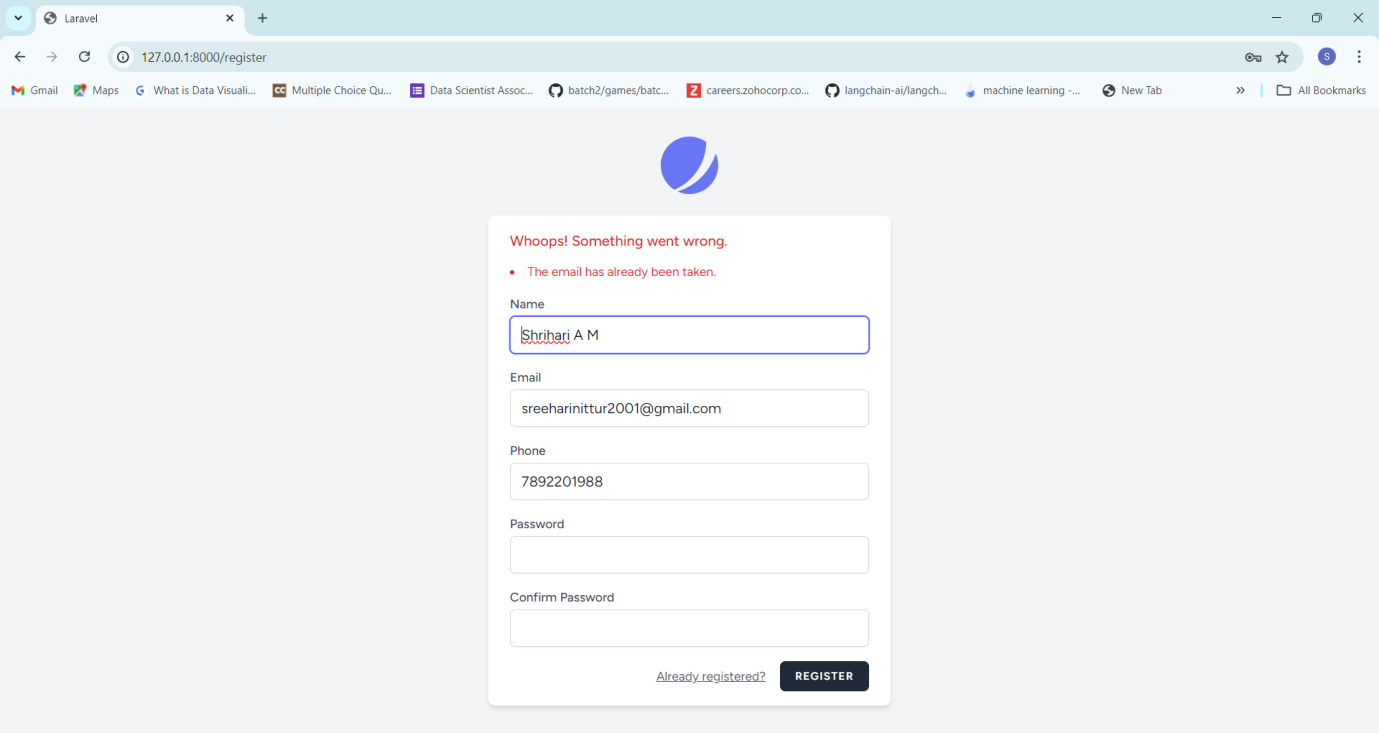
****

**Fig 4.4.3Register for User**

### Register for User

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl.No** | **TestCase** | **ExpectedResult** | **ObtainedResult** |
| 4 | Email validation | The email has already been taken. | Successful |

**Table 4.4.4 Register for User**

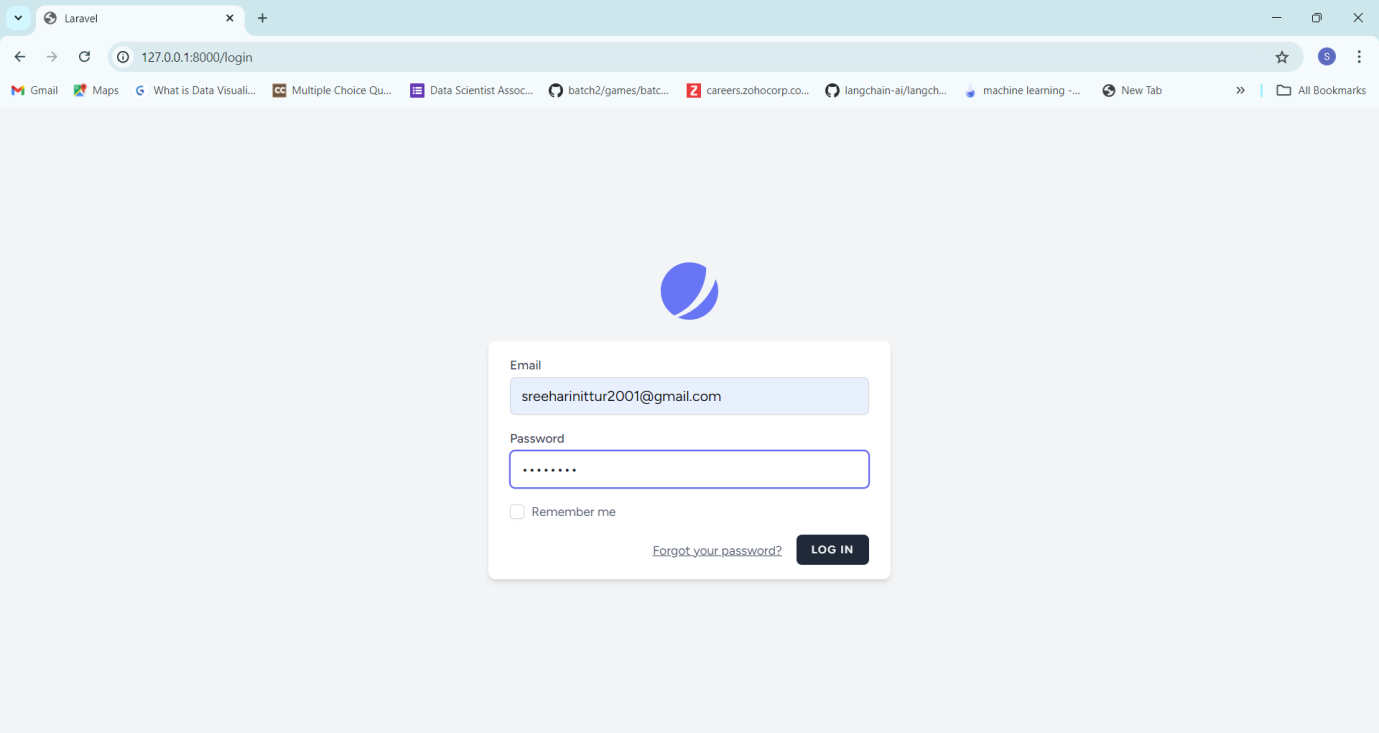
****

**Fig 4.4.4Register for User**

### Login for User

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl.No** | **TestCase** | **ExpectedResult** | **ObtainedResult** |
| 5 | Email validation | Valid email. | Successful |

**Table 4.4.5 Login for User**

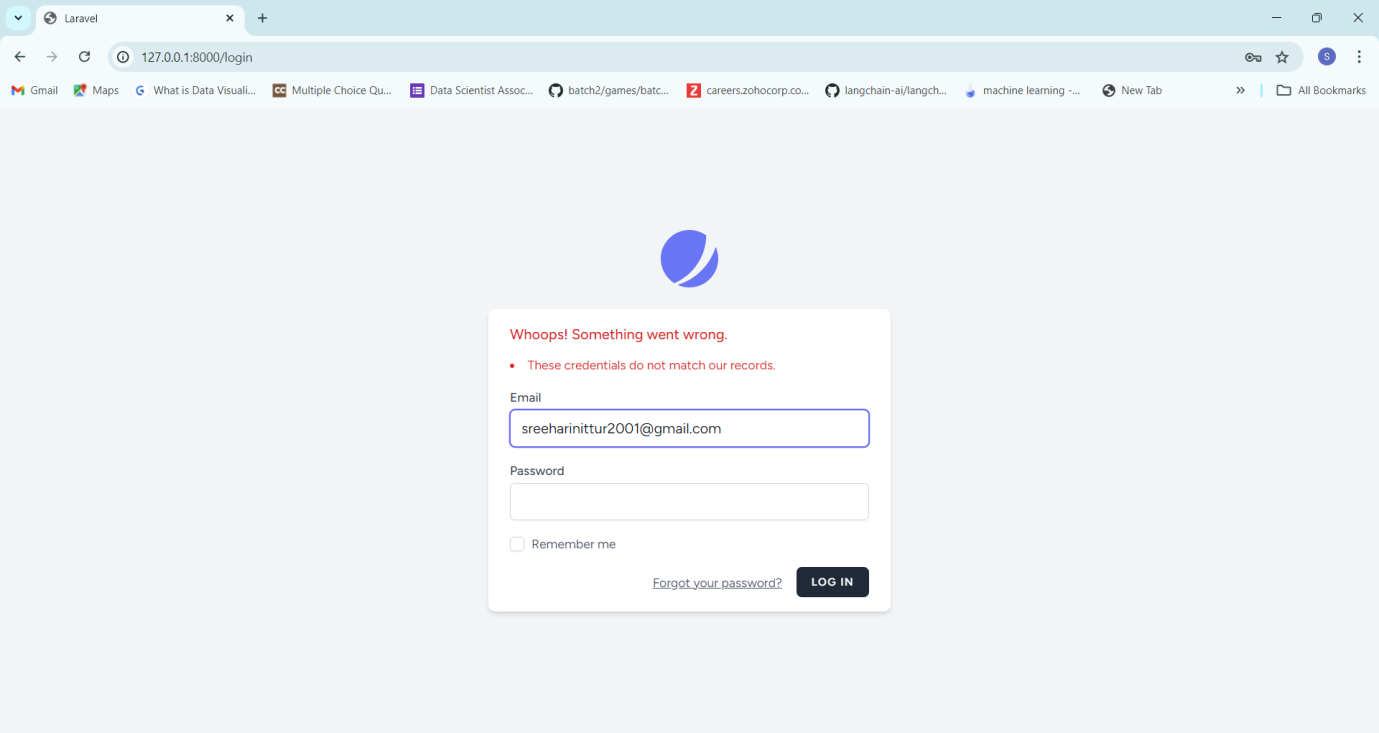
****

**Fig 4.4.5Login for User**

### Login for User

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl.No** | **TestCase** | **ExpectedResult** | **ObtainedResult** |
| 6 | Password validation | The provided credentials do not match our records. | Successful |

**Table 4.4.6 Login for User**

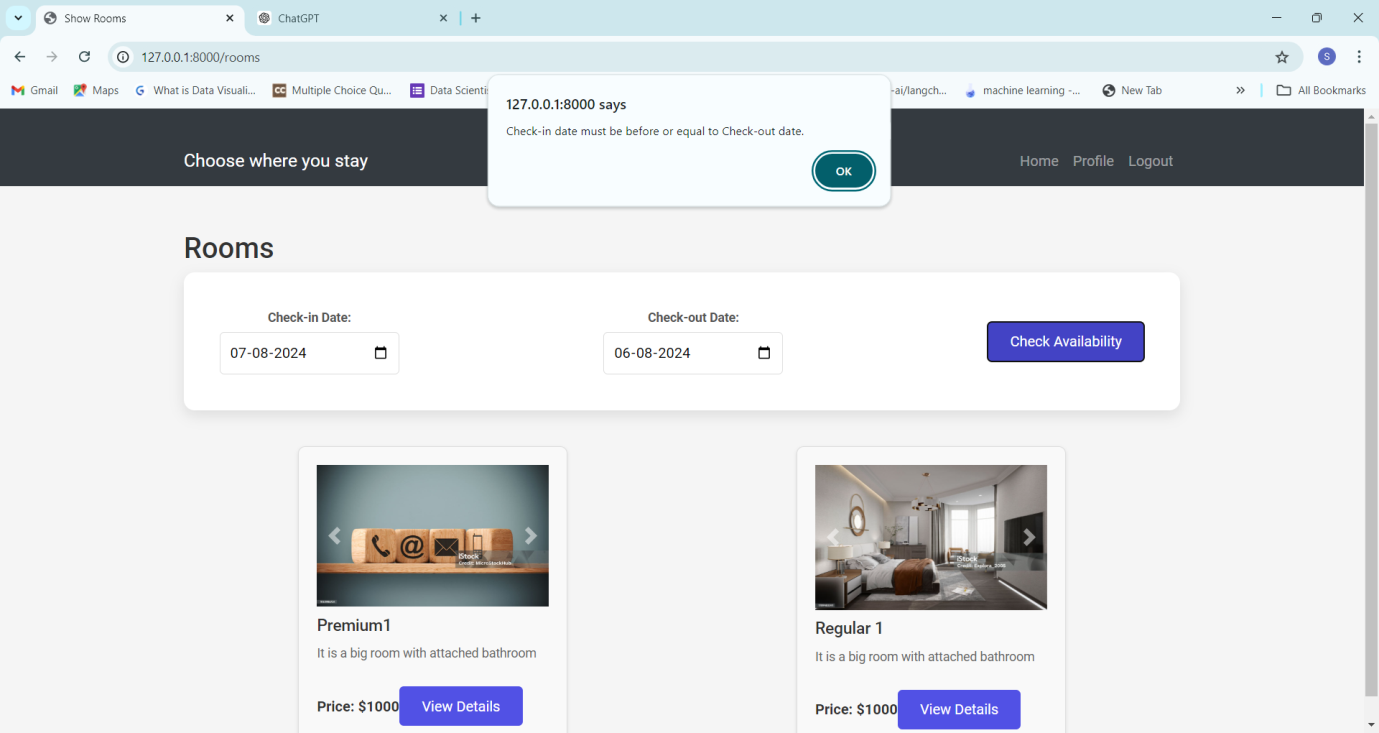
****

**Fig 4.4.6Login for User**

### 4.4.7Check room availability

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl.No** | **TestCase** | **ExpectedResult** | **ObtainedResult** |
| 7 | Check in after checkout date | The checkin date must be before or equal to checkout date. | Successful |

**Table 4.4.7.7 Check room availability**

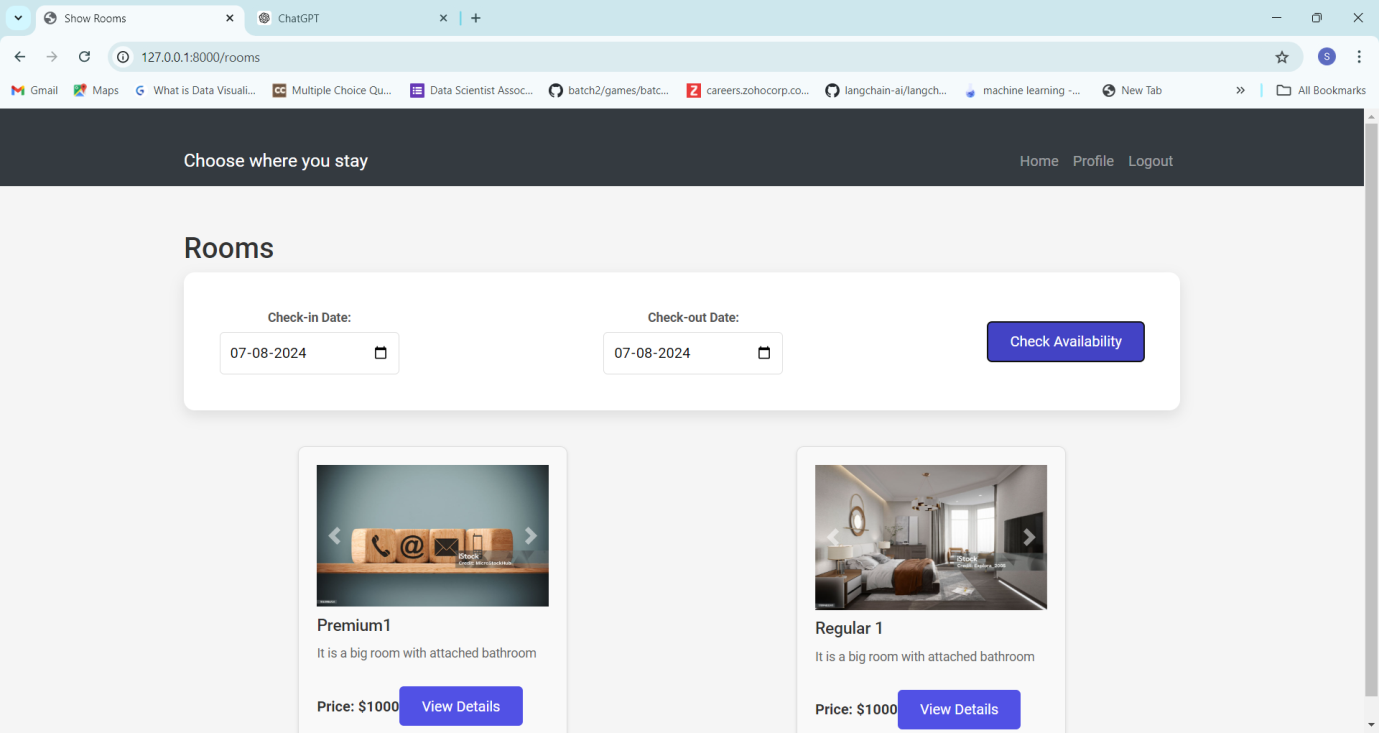
****

**Fig 4.4.7 Check room availability**

### 4.4.8 Check room availability

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl.No** | **TestCase** | **ExpectedResult** | **ObtainedResult** |
| 8 | Check in before checkout date | Showing available rooms | Successful |

**Table 4.4.8 Check room availability**

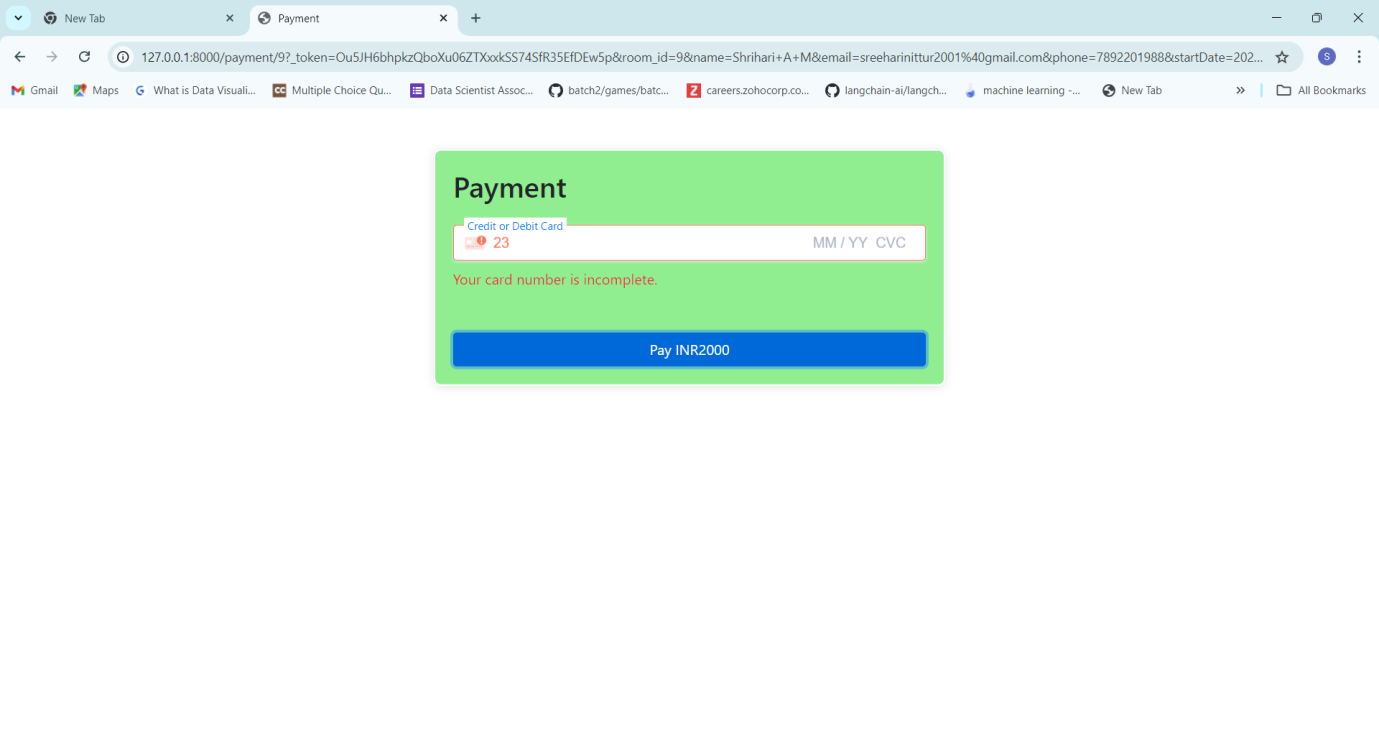
****

**Fig 4.4.8 Check room availability**

### 4.4.9Incomplete card number

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl.No** | **TestCase** | **ExpectedResult** | **ObtainedResult** |
| 9 | Incomplete card number | Your card number is incomplete | Successful |

**Table 4.4.9** Incomplete card number

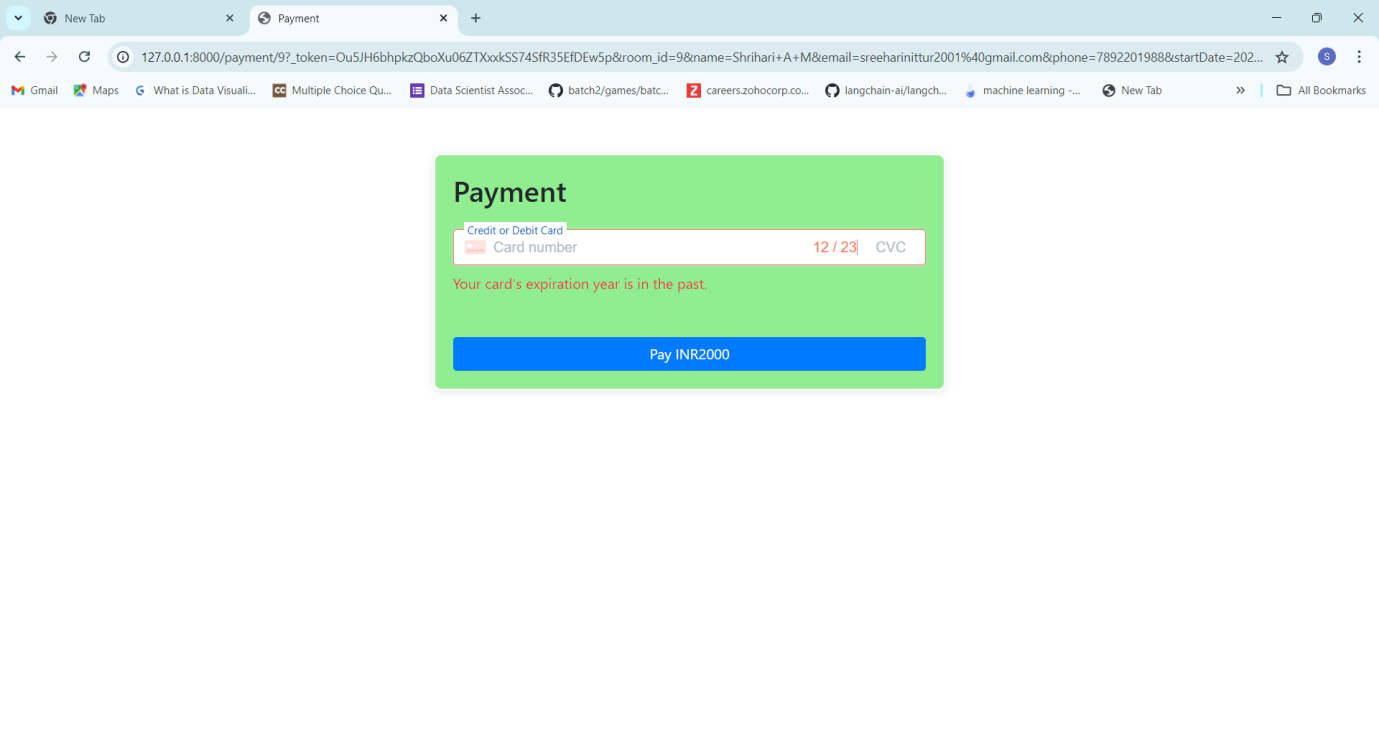
****

**Fig 4.4.9** **Incomplete card number**

### 4.4.10Expired card

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl.No** | **TestCase** | **ExpectedResult** | **ObtainedResult** |
| 9 | Expired Validity | Your card’s expiration year is in the past | Successful |

**Table 4.4.10 Assessment for User**

****

**Fig 4.4.10Assessment for User**

# CHAPTER-5 CONCLUSION &FUTURESCOPE

## Conclusion&Future Enhancement

### Conclusion

The Kamath Residency hotel website represents a significant step forward in enhancing the booking experience for our valued guests. By transitioning from a static website to a dynamic and interactive platform, we have successfully streamlined the booking process, ensuring a seamless and user-friendly experience. With real-time room availability, secure payment integration, and instant booking confirmations via SMS, our guests can now enjoy the convenience and reliability of modern technology.

Our commitment to excellence is reflected in every aspect of this new system, from the intuitive user interface to the robust backend infrastructure powered by Laravel 11. The incorporation of HTML, CSS, and JavaScript ensures that the website is both visually appealing and highly functional. The admin module empowers our management team with powerful tools to efficiently oversee room availability, bookings, and cancellations, all while providing insightful analytics to drive informed decision-making.

Looking ahead, we are excited to continue refining and expanding our website's capabilities, further enhancing the guest experience and maintaining our reputation as a premier destination. Kamath Residency is dedicated to leveraging technology to provide exceptional service, ensuring that every guest's stay is memorable and enjoyable.

We invite you to explore our website, experience the ease of online booking, and join us in embracing the future of hospitality at Kamath Residency.

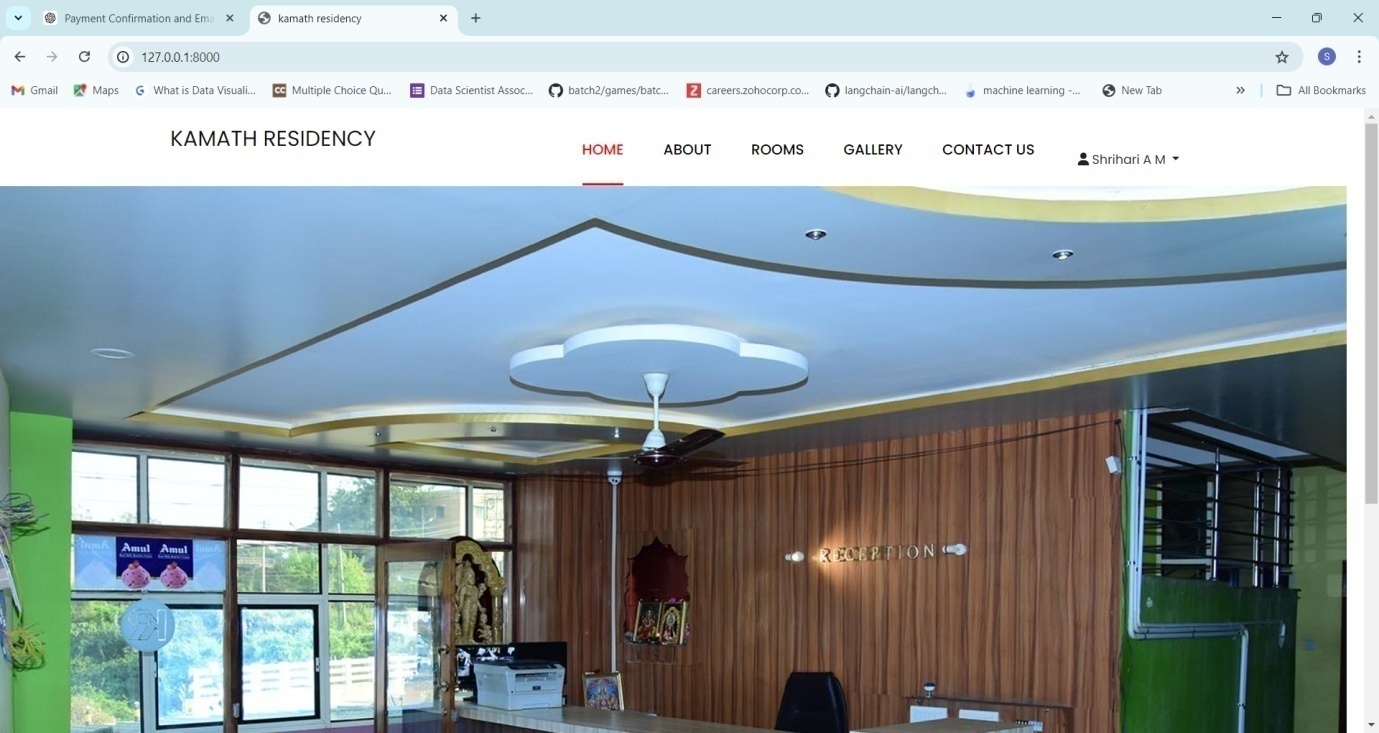
### FutureEnhancement

Looking ahead, Kamath Residency is dedicated to enhancing our website to better serve our guests. Planned future improvements include implementing a comprehensive loyalty program to reward returning guests with exclusive offers and discounts. We also aim to integrate advanced AI-powered chatbots for instant customer support and personalized recommendations. Additionally, the introduction of multilingual support will cater to our diverse clientele, ensuring a more inclusive and accessible user experience. Enhanced security features and improved data analytics will further optimize our operations, allowing us to anticipate guest needs and provide exceptional service. These enhancements reflect our commitment to leveraging technology to elevate the guest experience and maintain our position as a leader in the hospitality industry.

APPENDIX

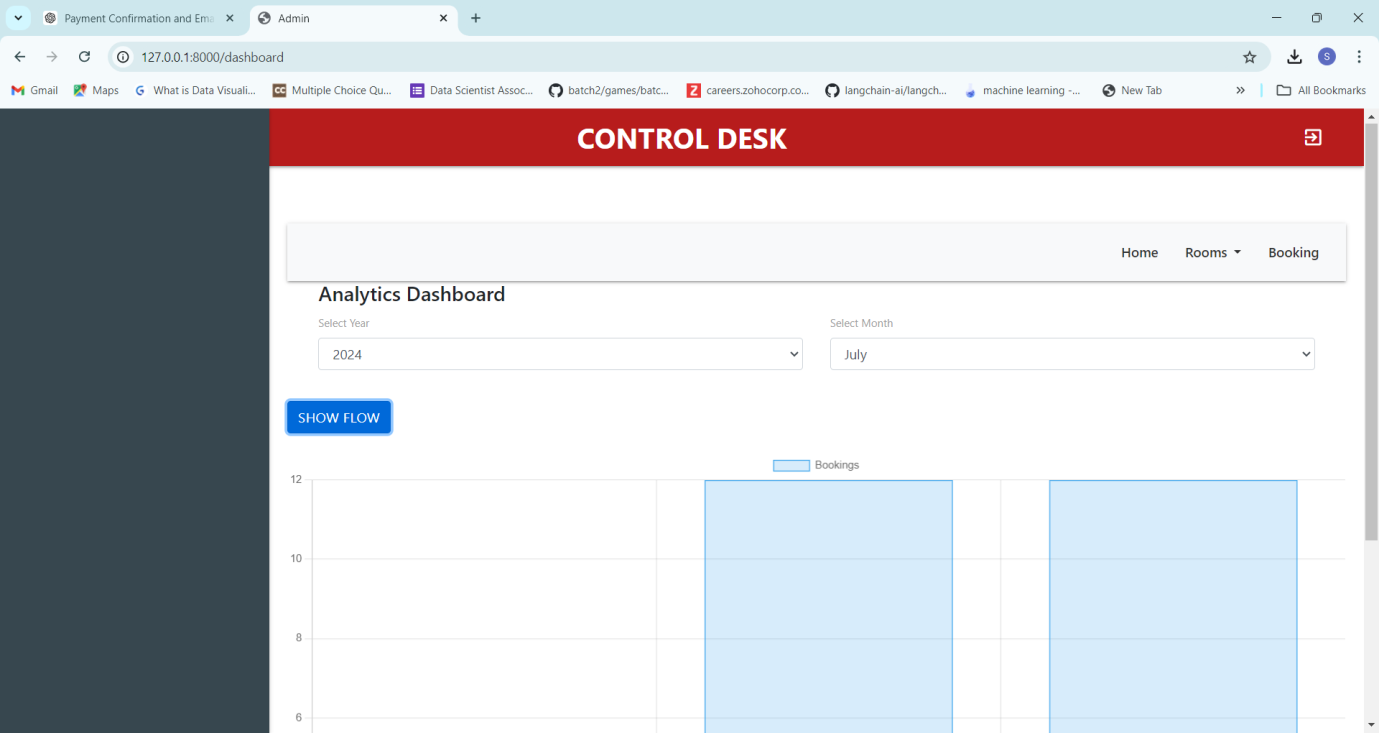
## Screenshots

**Home Page**

****

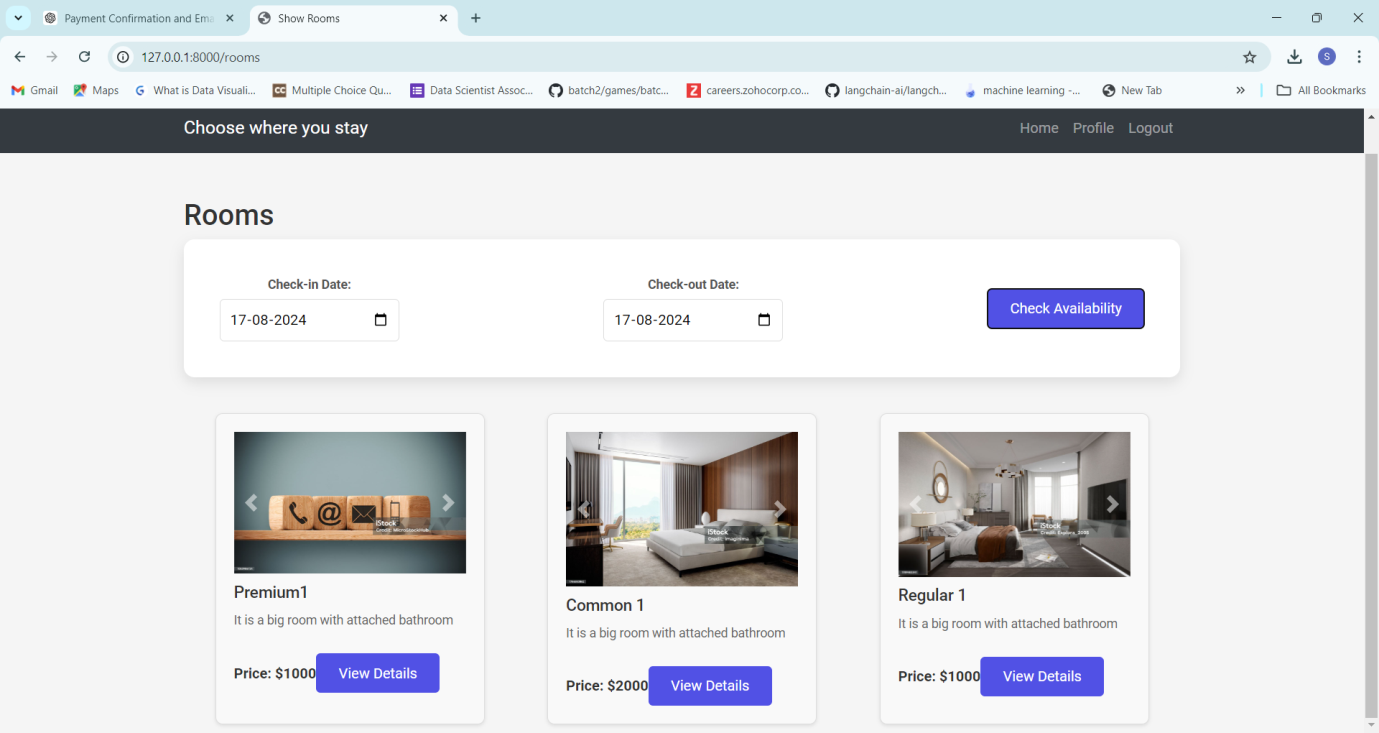
**Fig 1Home Page**

**Admin Dashboard**

****

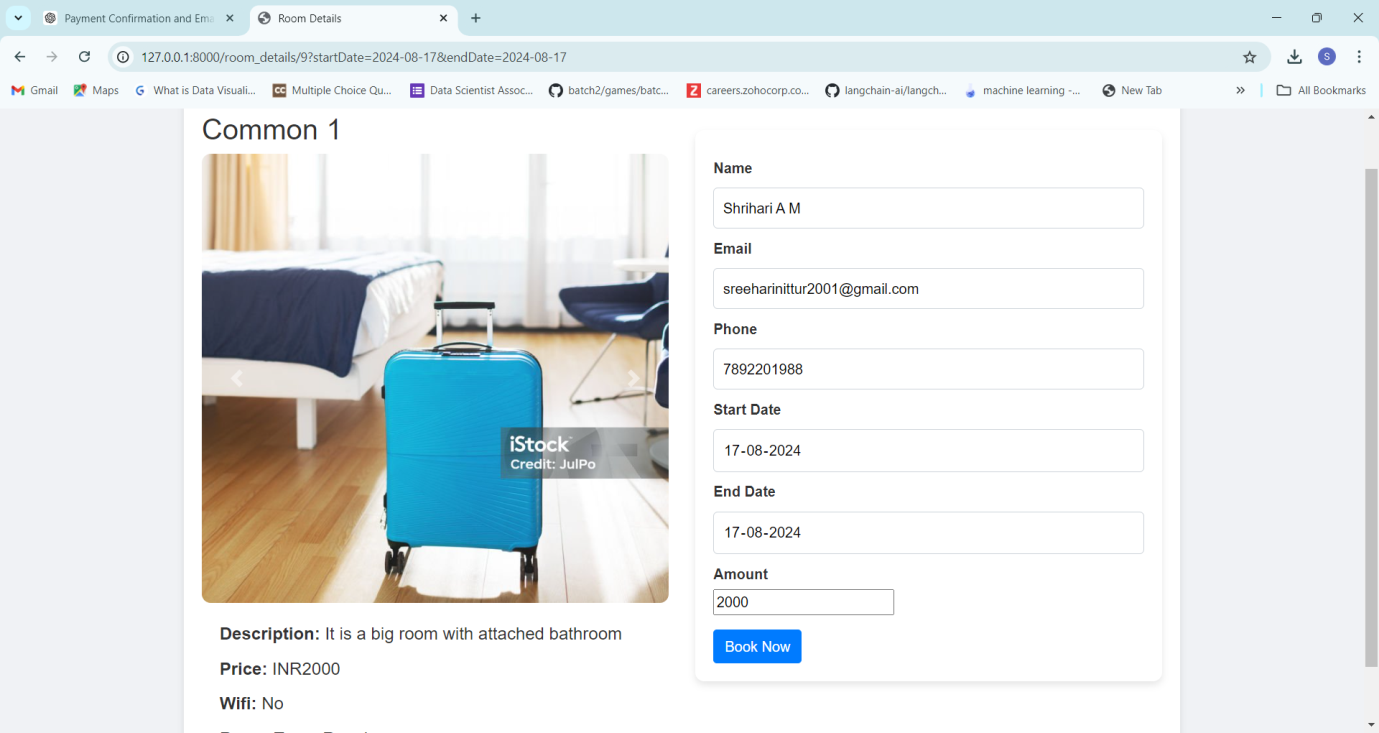
**Fig 2 Admin Dashboard page**

**Room Availability:**

****

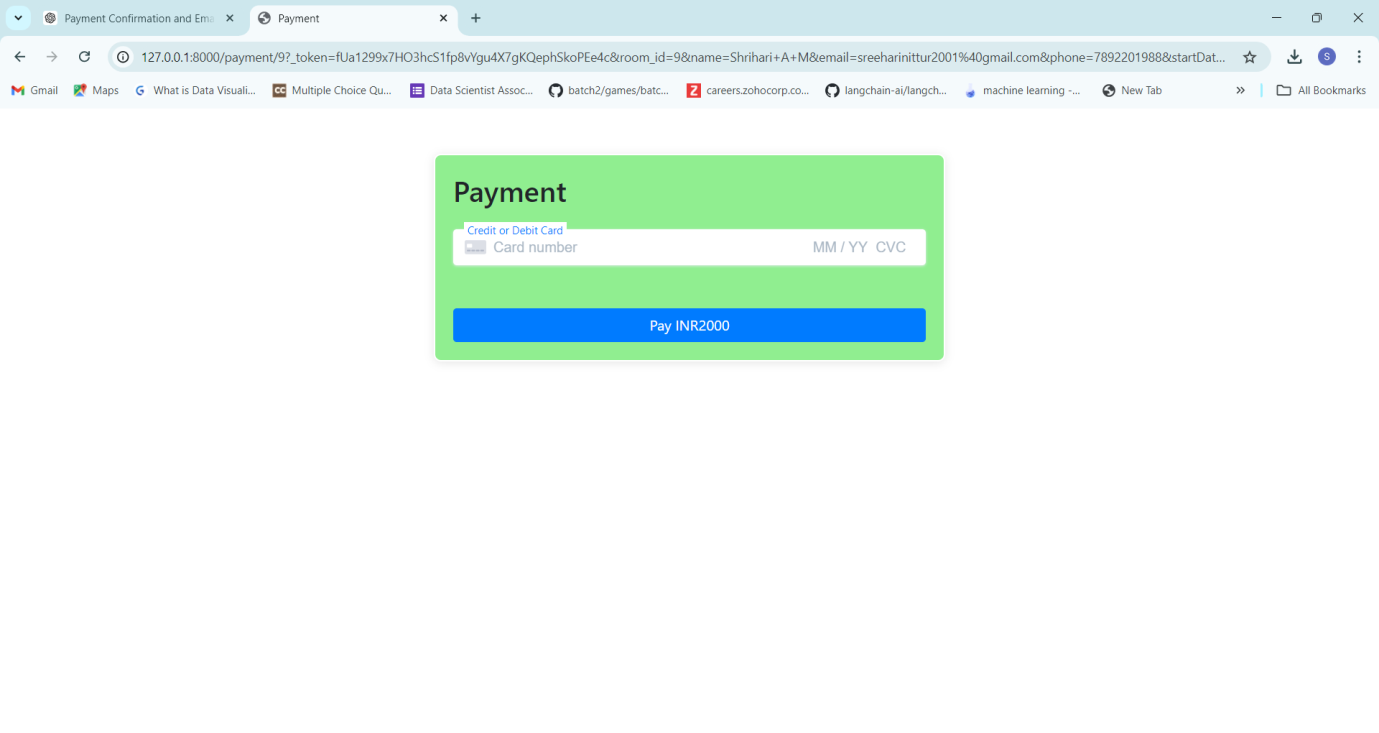
**Fig 4Room Availability**

**Room Booking:**

****

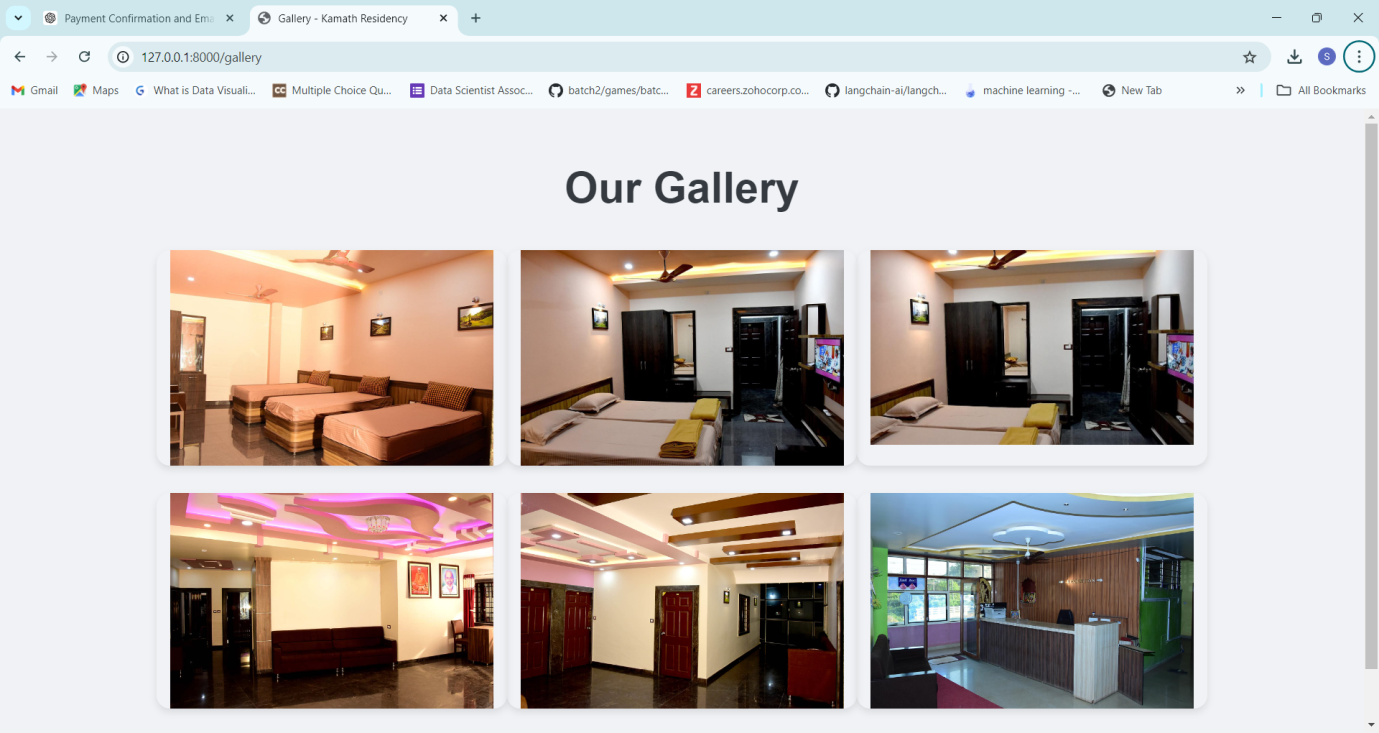
**Fig 5Room Booking**

**Payment:**

****

**Fig 6 Payment**

**Gallery:**

****

**Fig 7 Gallery**

REFERENCE

### 6.References

[1]. Software Engineering-A Practitioners approach 7th edition by **Roger S.**

[2]. Learning PHP, MySQL & JavaScript 4th Edition with jQuery by **Robin Nixon.**

[3]. Database Management System (DBMS)- A Practical Approach 5th edition by **Dr.Rajiv Chopra.**

[4]. <https://laravel.com/docs>

[5]. <https://getbootstrap.com/docs/5.0/components/accordion/>