**CHAPTER 1**

**INTRODUCTION**

In today's fast-paced world, managing and accessing home services can often be a challenging and time-consuming task. The Enhanced Portal for Home Services aims to revolutionize this experience by providing a comprehensive and user-friendly web application that integrates various functionalities to streamline the process. This innovative platform is designed to cater to the diverse needs of homeowners by offering a wide range of essential services, including electricians, plumbers, mechanics, and more, all accessible through a single, cohesive portal.

The Enhanced Portal for Home Services leverages modern technologies to enhance the efficiency of recruiting, managing, and locating home service providers. By ensuring effective scheduling, transparent pricing, and consistent service delivery, the platform facilitates seamless communication between clients and service providers. Key features such as safe payment methods, real-time tracking, and user evaluations are incorporated to elevate the overall consumer experience and build trust within the home services industry.At its core, the project aims to assist homeowners with critical home repairs, accessibility modifications, and energy-efficient upgrades. By fostering community efforts to support those in need of general home repair and improvements, the Enhanced Portal for Home Services not only simplifies the process of seeking help but also strengthens community bonds.

In summary, the Enhanced Portal for Home Services is dedicated to transforming how homeowners manage and access essential services. By offering a streamlined, reliable, and community-driven platform, it aims to make domestic needs manageable and accessible for everyone.

**CHAPTER 2**

**ORGANIZATION PROFILE**

Fantasy solution as a leading IT solution and service provider, provides innovative information technology - enabled solutions and services to meet the demands arising from social transformation, shaping new life styles for individuals and creating values for the society. Focusing on software technology, Fantasy solution provides industry solutions and product engineering solutions, related software products & platforms, and services, through seamless integration of software and services, software and manufacturing, as well as technology and industrial management capacity. Fantasy solution helps industry customers establish best practices in business development and management. The fantasy solution serves include real time projects, web designing, web hosting, software development and training etc, in many of which, has a leading market share. Notably, Fantasy Solution has participated in the formulation of many national IT standards and specifications.

Fantasy solution has the world’s leading product engineering capabilities, ranging from consultation, design, R&D, and integration to testing of embedded software, in the fields of automotive electronics, smart devices, digital home products, and IT products. The software provided by fantasy solution runs in a number of globally renowned brands.

Particularly offering the services that include application development & maintenance, ERP implementation & consulting, testing, performance engineering, software localization & globalization, IT infrastructures, BPO, IT education & training, etc.Sticking to its business philosophy and brand commitment of “Beyond Technology”, fantasy solution is dedicated to providing innovative information technologies to drive the sustainable development of society, as well as becoming a company that is well recognized and respected by employees, shareholders, customers, and society.

**Our services:**

This ever-changing environment, keeping a competitive edge means being able to anticipate and respond quickly to changing business conditions. Fantasy solution is a global software development company providing IT solutions to enterprises worldwide. Combining proven expertise in technology, and an understanding of emerging business trends, Fantasy3 delivers a range of software development solutions that includes e-business solutions, computer telephony, enterprise applications, professional web site design and development, product engineering, Electronic Health Records, CSoftware’s, Payment solutions, Time and attendance tracking software’s, Debt collection software’s, Appointment Reminder Solutions, Medical Transcription Services etc. We study, design, develop, enhance, customize, implement, maintain and support various aspects of information technology.

**Mission**

Fantasy Solutions’ mission includes:

* + 1. Providing high quality software development services, professional consulting and development outsourcing that would improve our customers’ operations;
    2. Making access to information easier and securer (Enterprise Business);
    3. Improving communication and data exchange (Business to Business);
    4. Providing our customers with a Value for Money and Providing our employees with meaningful work and advancement opportunities.

**CHAPTER 3**

**SYSTEM CONFIGURATION**

* 1. **HARDWARE CONFIGURATION**

* + - Processor : Dual core processor 2.6.0 GHZ
    - RAM : 4GB
    - Hard disk : 320 GB
    - Compact Disk : 650 Mb
    - Keyboard : Standard keyboard
    - Monitor : 15-inch color monitor

## 3.2 SOFTWARE CONFIGURATION

* + - Operating system : Windows OS
    - Front End : Python
    - Back End : MySQL SERVER
    - IDLE :Python2.7

**CHAPTER 4**

**SOFTWARE FEATURE**

# 4.1 Frond End: Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales. In July 2018, Van Rossum stepped down as the leader in the language community. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library. Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of Python's other implementations. Python and CPython are managed by the non-profit Python Software Foundation. Rather than having all of its functionality built into its core, Python was designed to be highly extensible. Van Rossum's vision of a small core language with a large standard library and easily extensible interpreter stemmed from his frustrations with ABC.While offering choice in coding methodology, the Python philosophy rejects exuberant syntax (such as that of Perl) in favor of a simpler, less-cluttered grammar. As Alex Martelli put it: "To describe something as 'clever' is not considered a compliment in the Python culture."Python's philosophy rejects the Perl "there is more than one way to do it" approach to language design in favour of "there should be one—and preferably only one—obvious way to do it".

Python's developers strive to avoid premature optimization, and reject patches to non- critical parts of CPython that would offer marginal increases in speed at the cost of clarity.[ When speed is important, a Python programmer can move time-critical functions to extension modules written in languages such as C, a just-in-time compiler.An important goal of Python's developers is keeping.

occasionally playful approaches to tutorials and reference materials, such as examples that refer to spam and eggs (from a famous Monty Python sketch) instead of the standard for and bar.

A common neologism in` the Python community is pythonic, which can have a wide range of meanings related to program style. To say that code is pythonic is to say that it uses Python idioms well, that it is natural or shows fluency in the language, that it conforms with Python's minimalist philosophy and emphasis on readability. In contrast, code that is difficult to understand or reads like a rough transcription from another programming language is called un pythonic. Users and admirers of Python, especially those considered knowledgeable or experienced, are often referred to as Pythonists, Pythonistas, and Pythoneers. Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed. Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception. A source level debugger allows inspection of local and global variables, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

Python can be used in Web development, numeric programming, game development, serial port access and more.

There are two attributes that make development time in Python faster than in other programming languages:

* + 1. Python is an interpreted language, which precludes the need to compile code before executing a program because Python does the compilation in the background. Because Python is a high-level programming language, it abstracts many sophisticated details from the programming code. Python focuses so much on this abstraction that its code can be understood by most novice programmers.
    2. Python code tends to be shorter than comparable codes. Although Python offers fast development times, it lags slightly in terms of execution time. Compared to fully compiling languages like C and C++, Python programs execute slower. Of course, with the processing speeds of computers these days, the speed differences are usually only observed in benchmarking tests, not in real-world operations.

**4.2 Back End: My SQL**

MySQL is the world's most used open source [relational database management system](http://en.wikipedia.org/wiki/Relational_database_management_system) (RDBMS) as of 2008 that run as a server providing multi-user access to a number of databases. The MySQL development project has made its [source code](http://en.wikipedia.org/wiki/Source_code) available under the terms of the [GNU General Public License,](http://en.wikipedia.org/wiki/GNU_General_Public_License) as well as under a variety of [proprietary](http://en.wikipedia.org/wiki/Proprietary_software) agreements. MySQL was owned and sponsored by a single [for-profit](http://en.wikipedia.org/wiki/Business) firm, the [Swedish](http://en.wikipedia.org/wiki/Sweden) company [MySQL AB,](http://en.wikipedia.org/wiki/MySQL_AB) now owned by [Oracle Corporation.](http://en.wikipedia.org/wiki/Oracle_Corporation)

MySQL is a popular choice of database for use in web applications, and is a central component of the widely used LAMP open source web application software stack—LAMP is an acronym for "[Linux](http://en.wikipedia.org/wiki/Linux), [Apache](http://en.wikipedia.org/wiki/Apache_HTTP_Server), MySQL, [Perl](http://en.wikipedia.org/wiki/Perl)/[PHP](http://en.wikipedia.org/wiki/PHP)/[Python](http://en.wikipedia.org/wiki/Python_%28programming_language%29)." [Free-software](http://en.wikipedia.org/wiki/Free_software)-open source projects and also includes of the

Many high-profile, large-scale [World Wide Web](http://en.wikipedia.org/wiki/World_Wide_Web) products, including Wikipedia, Google(though not for searches), [ImagebookTwitter,](http://en.wikipedia.org/wiki/Facebook) [Flickr,](http://en.wikipedia.org/wiki/Flickr) [Nokia.com,](http://en.wikipedia.org/wiki/Nokia) and [YouTube](http://en.wikipedia.org/wiki/YouTube).

## Inter images

MySQL is primarily an RDBMS and ships with no GUI tools to administer MySQL databases or manage data contained within the databases. Users may use the included command line tools, or use MySQL "front-ends", desktop software and web applications that create and manage MySQL databases, build database structures, back up data, inspect status, and work with data records. The official set of MySQL front-end tools, MySQL Workbench is actively developed by Oracle, and is freely available for use.

## Graphical

The official MySQL Workbench is a free integrated environment developed by MySQL AB, that enables users to graphically administer MySQL databases and visually design database structures. MySQL Workbench replaces the previous package of software, [MySQL GUI Tools.](http://en.wikipedia.org/wiki/MySQL_GUI_Tools) Similar to other third-party packages, but still considered the authoritative MySQL frontend, MySQL Workbench lets users manage database design & modeling, SQL development (replacing MySQL Query Browser) and Database administration (replacing MySQL Administrator).MySQL Workbench is available in two editions, the regular free and open source Community Edition which may be downloaded from the MySQL website, and the proprietary Standard Edition which extends and improves the feature set of the Community Edition.

**CHAPTER 5**

**PROJECT DESCRIPTION**

**5.1 OVERVIEW OF THE PROJECT**

The goal of the Home Servicers Booking System project is to meet the increasing need for quick access to a range of household services by developing an effective, user-friendly platform for booking and maintaining home service appointments. With the help of this system, clients can get in touch with qualified service providers for jobs like appliance repairs, plumbing, electrical, and cleaning. Through a web application, users can conveniently browse offered services, examine provider profiles, verify availability, and schedule appointments. Features like automated reminders, secure payment processing, ratings and reviews of service providers, and real-time booking updates are all included in the platform. Tools that help service providers keep track of appointments, manage their calendars, and process payments are beneficial.

## PROBLEM STATEMENT

The difficulties that clients and service providers in the home services sector encounter are addressed via the Home Servicers Booking System. Consumers typically experience annoyance and inefficiency while trying to locate reputable service providers, evaluate costs, and arrange schedules. On the other side, service providers struggle to make appointments, connect with potential customers, and collect money on time. These problems are made worse by the absence of a consolidated platform, which leads to lost opportunities and lower customer satisfaction. By creating an all-inclusive, user-friendly system that expedites the booking procedure, improves communication, and offers a dependable platform for scheduling home care appointments, this project seeks to address these issues. The system aims to enhance the overall experience for both clients and service providers by incorporating features

## 5.3 MODULE DESCRIPTION

**MODULES**

## Admin

**Servicer**

## User

* Login
* View Service Details
* View User Details
* Register
* Login
* Update Service Details
* View User Booking Info
* Report
* Register
* Login
* View Service Details
* Book Service
* Make Payment

**MODULE DESCRIPTION**

1.Admin

* + - Login

In this module, the admin can login in the system using his/her username and password.

* + - View Service Details

In this module, the admin can view the service information posted by servicer.

* + - View User Details

In this module is used to view the registered user details. The user details are stored.

## 2.Servicer

* + Register

There is registration form available where new servicer can create their account by providing required information to the system. The registration form details are like servicer name, email, gender, mobile number, address, and etc. These details are stored in the database. And then can getting to the username and password in the system.

* + Login

In this module, servicer can login in the system using username and password.

* + Update Service Details

In this module is used to add/update the service details by servicer. Servicer can view the added service details. The service details contain service name, type, description etc.

* + View User Booking Info

This module is used to view the booking details by servicer. The details are stored in the database.

* + Report

In this module, the servicer can generate the report based on user requirements, booking details etc.

## 3.User

* + Register

There is registration form available where new user can create their account by providing required information to the system. The registration form details are like name, email, gender, mobile number, address, and etc. These details are stored in the database. And then can getting to the username and password in the system.

* + Login

In this module, user can login in the system using username and password.

* + View Service Details

In this module is used to user can view the service details updated by servicer.

* + Book Service

In this module, the user can book the particular service. The service booking details contain service type, location, date and address etc. These details are filled by user.

* + Make Payment

In this module used to make payment. This module contains user’s card details like name, card no, amount etc.

## CHAPTER 6

**SYSTEM ANALYSIS**

## EXISTING SYSTEM

As booking is having manual booking system, they are facing some problems issuing booking requests of users. All the necessary booking stuffs are being done in hard copy. So it become much difficult for servicers to keep the records updated all the time. As for example, if the user needs to change the check in date it become difficult for them to find out the customers booking details for updating as there are so many customers booking records. Difficult to search nearby booking service.

**DISADVANTAGES**

* + - It is difficult to maintain important information in booking.
    - Time Consuming.
    - Less Efficient.
    - More manual work required.

## PROPOSED SYSTEM

Thus by application the flaws in the existing system and proposing a new solution, a real time Web Application has been built using the technologies mentioned and named it as “Enhanced Portal for Home Services”. Thus this application seems to be more dynamic, effective and efficient than the existing system. In this proposed system the user can search the more than one servicer in particular category.

## ADVANTAGES

* Error free and menu driven interface.
* Saves lot of time.
* Quick preparation of reports.

# CHAPTER 7

# SYSTEM DESIGN

* 1. **UML DIAGRAM**

## 7.1.1 Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. In this context, a "system" is something being developed or operated, such as a web site. The "actors" are people or entities operating under defined roles within the system.

**Registration**

**Login**

**Add service info**

**view service info**

**book service**

**make payment**

**view booking**

**Accept/reject booking**

**View user info**

**view servicer info**

**Feedback**

**report**



**User**



**Admin**



**Servicer**

**Fig.No.7.1.1 UseCase Diagram for Enhanced Portal for Home Services.**

## 

## 7.1.2 Class Diagram

A class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classesh, their attributes, operations and the relationships among objects. The class diagram is the main building block of object-oriented modelling. It is used for general conceptual modelling of the systematic of the application, and for detailed modelling translating the models into programming code.

+Login()

+Registration()

+Login()

+add service()

+accept/reject booking()

+report()

+View booking

+View servicer

+View user

**servicer**

**Admin**

|  |
| --- |
| **user** |
| +view service info() |
| +Registration()  +Login()  +book service()  +make payment()  +Feedback() |

**Fig.No.7.1.2 Class Diagram for Enhanced Portal for Home Services.**

## Sequence Diagram

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

ification()

3 : Register()

4 : Login()

ification()

6 : add service info()

7 : Reg

8 : Lo

ification()

10 : view se

11 : book

12 : view booking()

: accept/reject bookin

14 : view user info()

dback()

17 : report()

16 : Fee

5 : view servicer info()

g()

service()

rvice info()

9 : Ver

gin()

ister()

5 : Ver

2 : Ver

1 : Login()

User

Servicer

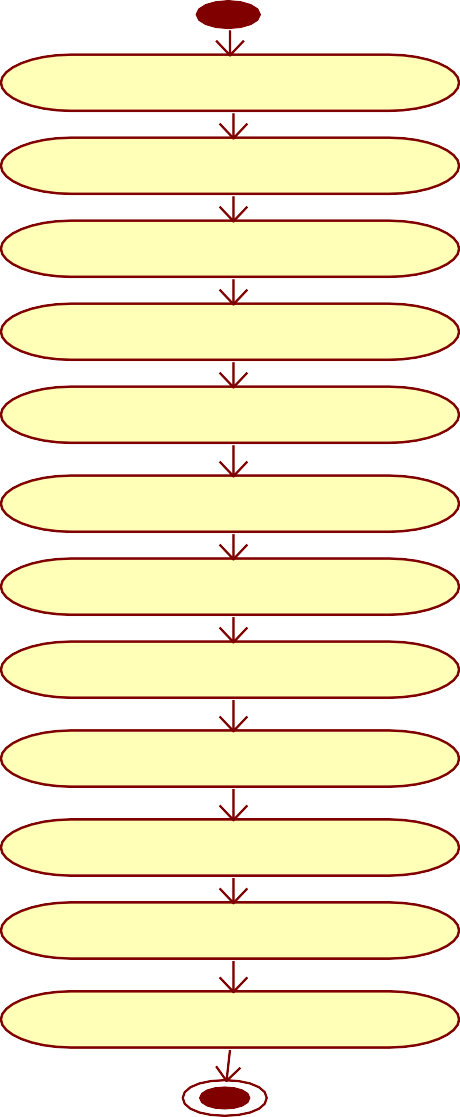
Server

Admin

**Fig.No.7.1.3 Sequence Diagram for Enhanced Portal for Home Services.**

## 7.1.4 Activity Diagram

Activity diagram displays a special state diagram, where most of the state are action states and most of the transitions are triggered by completion o0f the action in the source states. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent. Activity diagrams deals with all type of flow control by using different elements.



Registration

Login

Add service info

view service info

book service

make pay ment

view booking

Accept/reject booking

View user info

view servicer info

Feedback

report

**Fig.No.7.1.4 Activity Diagram for Enhanced Portal for Home Services.**

## 7.1.5 Component Diagram

A component diagram, sometimes referred to as a UML component diagram, shows how the actual components of a system are wired and arranged. Component diagrams are frequently used to help model implementation specifics and verify that planned development addresses all necessary functionalities of the system.

**report**

**View User Details**

**Feedback**

**Accept/ reject**

**Make Payment**

**Book Service**

**View Service Details**

**Update Service Details**

**Login**

**Register**

**Fig.No.7.1.5 Component Diagram for Enhanced Portal for Home Services.**

## 7.1.6 Deployment Diagram

Components and artifacts are displayed in a deployment diagram according to their respective uses in the deployed system. The arrangement of the system's artifacts and components is specified in a component diagram. Note: Deployment topologies are a different kind of model from deployment diagrams.

**report**

**View User Details**

**Feedback**

**Accept/ reject**

**Make Payment**

**Book Service**

**View Service Details**

**Update Service Details**

**Login**

**Register**

**Fig.No.7.1.6 Deployment Diagram for Enhanced Portal for Home Service**

## 7.2 DATABASE DESIGN

Information is arranged into rows and columns in a table, a type of data structure. It may be applied to the structured storing and displaying of data. Databases, for instance, keep information organized into tables so that certain rows may be easily retrieved. Tables are a common way for websites to show several rows of data on one page. Spreadsheets store and present data in an organized manner, combining the functions of a table.

Databases frequently include several tables, each intended for a particular function. For instance, distinct columns for clients, suppliers, and workers may be present in a firm database. Depending on what information each table needs to record, each table may have a unique collection of fields. Every field in database tables is seen as a column, and every entry, or record, is regarded as a row. You may retrieve a particular value from the table by requesting information from a single column and row.

**DATABASE DESIGN**

**7.2.1 Table Name: ADMIN TABLE**

**Description: This table contains the admin details**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD** | **DATA TYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| **Admin Name** | varchar(45) | Not Null | Admin name |
| **password** | varchar(25) | Not Null | Password of Admin |

**7.2.2 Table Name: REGISTRATION TABLE**

**Description: This table contains the user registration details**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD** | **DATA TYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| User\_id | Int(20) | Primary key | User id |
| Name | varchar(20) | Not NULL | Name of User |
| Email | varchar(14) | Unique Key | Email id |
| Mobile | Int(10) | Unique Key | Mobile |
| Address | varchar(50) | Not NULL | Address |
| Password | varchar(25) | Not NULL | Password |

**w**

**7.2.3 Table Name: SERVICER/Employee TABLE**

**Description: This table contains the employee details.**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD** | **DATA TYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| **Servicer\_id** | Int(10) | Primary Key | Id of Servicer |
| **ServicerName** | Varchar(20) | Not Null | Name of servicer |
| **Email** | varchar(20) | Unique KeY | Email id of Servicer |
| **MobileNumber** | Int(10) | Unique Key | Mobile Number of Servicer |
| **UserName** | varchar(25) | Not Null | User name |
| **Password** | varchar(15) | Unique key | Password |
| **ptype** | varchar(25) | Not Null | Type of service need |

**.**

**7.2.4 TABLE NAME: SERVICE TABLE**

**DESCRIPTION: This table contains the Service information.**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD** | **DATA TYPE** | **CONSTRAINSTS** | **DESCRIPTION** |
| **Id** | Primary key | Unique Key | Id |
| **ServiceName** | varchar(25) | Not Null | Name of the service |
| **Info** | varchar(50) | Not Null | Information of a service |
| **Image** | varchar(500) | Blob | Image of a Service |

**7.2.5 Table Name: BOOKING TABLE**

**Description: This table contains the user booking details.**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD** | **DATA TYPE** | **CONSTRAINS** | **DESCRIPTION** |
| **Book\_id** | int(10) | Primary Key | Booking id |
| **User Name** | Varchar(50) | Not Null | Name of User |
| **ServiceName** | varchar(50) | Foreign Key References Service(Servicename) | Name of the Service |
| **ServicerName** | Varchar(50) | Foreign Key References Servicer (Servicername) | Name of Servicer |
| **Date** | varchar(15) | Date | Date of Service Booking |
| **Address** | varchar(25) | Unique Key | Address of the User |
| **info** | Varchar(50) | Not Null | Information about service |
| **Status** | ENUM(booked,Notbooked) | Not Null | Booking Status |
| **Amount** | varchar(50) | Not null | Amount of Service to pay |

**7.2.6 TABLE NAME:REVIEW TABLE**

**Description:This table contains the Review information**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD** | **DATATYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| **Id** | Int(20) | Not Null | Id |
| **ServicerId** | varchar(25) | Not Null | Id of servicer |
| **Mobile** | varchar(25) | Not Null | Mobile number |
| **SType** | varchar(25) | Not Null | Type of a Service |
| **ServicerName** | varchar(25) | Primary Key | Name of the servicer |
| **Image** | varchar(50) | Blob | Image of service |
| **UserName** | varchar(25) | Not Null | Name of User |
| **Rate** | int(25) | Not Null | star Rating of servicer |
| **Review** | varchar(50) | Not Null | Review given by user |
| **Smile1** | varchar(25) | Not Null | service was excellent |
| **Smile2** | varchar( (25) | Not Null | service was very good |
| **Smile3** | varchar( (25) | Not Null | service was good |
| **Smile4** | varchar( (25) | Not Null | service was Average |
| **Smile5** | varchar( (25) | Not Null | service was not bad |
| **Smile6** | varchar( (25) | Not Null | service was bad |
| **Result** | varchar(25) | Not Null | Result of review |

**7.2.7 Table Name: FEEDBACK TABLE**

**Description: This table contains the feedback details updated by the user.**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD** | **DATA TYPE** | **CONSTRAINTS** | **DESCRIPTION** |
| **Id** | int(10) | Primary Key | Id |
| **User\_Name** | Varchar(50) | Not Null | Name of user |
| **Feedback** | varchar(50) | Not Null | Feedback of Service/Servicer |

**CHAPTER 8**

**SYSTEM TESTING**

* 1. **TEST PLAN**

Testing Level Specific Test Plans**:** Plans for each level of testing.

* + - Unit Test Plan
    - Integration Test Plan
    - System Test Plan

## 1.Unit Test Plan

* + - **Purpose**: Ensures individual units of code (such as functions or methods) operate correctly in isolation.
    - **Scope**: Focuses on the smallest testable components of the software.
    - **Key Components**: List of units to be tested, dependencies and mock objects required, specific features to be tested, and exclusions (like external systems).

## 2.Integration Test Plan

* **Purpose**: Validates the interactions and data flow between integrated units or components.
* **Scope**: Ensures combined units work together as expected.
* **Key Components**: List of integrated modules to be tested, interaction points, data exchange paths, and exclusions (e.g., individual unit functionality).

## 3.System Test Plan

* + - **Purpose**: Verifies the complete and integrated software system meets specified requirements.
    - **Scope**: Covers end-to-end testing of the entire system.

**TEST PLAN**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test plan Id** | **Test module** | **Type of testing** | **Test description** | **Duration** | **Remarks** |
| TP01 | Login Form | Validation testing | User name and password | 2 hours | Success |
| TP02 | Servicer Registration Form | Unit testing | Servicer basic details | 2 hours | Success |
| TP03 | Login Form | Validation testing | User name and password | 2 hours | Success |
| TP04 | User Registration Form | Unit testing | user basic details | 2 hours | Success |
| TP05 | User Login Form | Validation testing | User name and password | 2 hours | Success |
| TP06 | Booking Form | Unit testing | User details and Service details | 2 hours | Success |
| TP07 | Payment Form | Unit testing | User card details | 2 hours | Success |

## 8.2 TEST CASES

A test case has components that describe input, action and an expected response, in order to determine if a feature of an application is working correctly. A test case is a set of instructions on “HOW” to validate a particular test objective/target, which when followed will tell us if the expected behavior of the system is satisfied or not.

Characteristics of a good test case:

* + - Accurate: Exacts the purpose.
    - Economical: No unnecessary steps or words.
    - Repeatable: Can be used to perform the test over and over.

**8.2.1 Test Plan ID:** TP01

**Test Module:** Login Form

**Type of Testing:** Validation Testing

**Test Description:** User name and password

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Scenario** | **Test Steps** | **Expected Result** | **Actual Result** | **Status** | **Remarks** |
| TC01 | Valid Login | 1. Open login page.  2. Enter valid username and password.  3. Click login button. | User successfully l ogs in | User successfully logged in | Pass | None |
| TC02 | Invalid Username | 1. Open login page.  2. Enter invalid username and valid password..  3. Click login button. | Error message displayed | Error message displayed | Pass | None |
| TC03 | Invalid Password | 1. Open login page.  2. Enter valid username and invalid password.  3. Click login button. | Error message displayed | Error message displayed | Pass | None |
| TC04 | Blank Username and Password | 1. Open login page.  2. Leave username and password fields blank.  3. Click login button. | Error message displayed | Error message displayed | Pass | None |
| TC05 | SQL Injection Attempt | 1. Open login page.  2. Enter SQL injection code in username and password fields.  3. Click login button. | Error message displayed/ No login | Error message displayed/ No login | Pass | None |

**8.2.2 Test Plan ID:** TP02

**Test Module:** Servicer Registration Form

**Type of Testing:** Unit Testing

**Test Description:** Servicer basic details

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Scenario** | **Test Steps** | **Expected Result** | **Actual Result** | **Status** | **Remarks** |
| TC01 | Valid Registration | 1. Open registration page.  2. Enter valid servicer details.  3. Click submit button. | Servicer successfully registered | Servicer successfully registered | Pass | None |
| TC02 | Missing Required Fields | 1. Open registration page.  2. Leave required fields blank.  3. Click submit button. | Error message displayed | Error message displayed | Pass | None |
| TC03 | Invalid Email Format | 1. Open registration page.  2. Enter invalid email format.  3. Click submit button. | Error message displayed | Error message displayed | Pass | None |
| TC04 | Password Mismatch | 1. Open registration page.  2. Enter password and confirm password that do not match.  3. Click submit button. | Error message displayed | Error message displayed | Pass | None |
| TC05 | SQL Injection Attempt | 1. Open registration page.  2. Enter SQL injection code in any text field.  3. Click submit button. | Error message displayed / No registration | Error message displayed / No registration | Pass | None |

**8.2.3 Test Plan ID:** TP03

**Test Module:** Login Form

**Type of Testing:** Validation Testing

**Test Description:** User name and password

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Scenario** | **Test Steps** | **Expected Result** | **Actual Result** | **Status** | **Remarks** |
| TC01 | Valid Login | 1. Open login page.  2. Enter valid username and password.  3. Click login button. | User successfully logs in | User successfully logged in | Pass | None |
| TC02 | Invalid Username | 1. Open login page.  2. Enter invalid username and valid password.  3. Click login button. | Error message displayed | Error message displayed | Pass | None |
| TC03 | Invalid Password | 1. Open login page.  2. Enter valid username and invalid password.  3. Click login button. | Error message displayed | Error message displayed | Pass | None |
| TC04 | Blank Username and Password | 1. Open login page.  2.Leave username and password fields blank.  3. Click login button. | Error message displayed | Error message displayed | Pass | None |
| TC05 | SQL Injection Attempt | 1. Open login page.  2. Enter SQL injection code in username and password fields.  3. Click login button. | Error message displayed / No login | Error message displayed / No login | Pass | None |

**8.2.4 Test Plan ID:** TP04

**Test Module:** User Registration Form

**Type of Testing:** Unit Testing

**Test Description:** User basic details

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Scenario** | **Test Steps** | **Expected Result** | **Actual Result** | **Status** | **Remarks** |
| TC01 | Valid Registration | 1. Open registration page.  2. Enter valid user details.  3. Click submit button. | User successfully registered | User successfully registered | Pass | None |
| TC02 | Missing Required Fields | 1. Open registration page.  2. Leave required fields blank.  3. Click submit button. | Error message displayed | Error message displayed | Pass | None |
| TC03 | Invalid Email Format | 1. Open registration page.  2. Enter invalid email format.  3. Click submit button. | Error message displayed | Error message displayed | Pass | None |
| TC04 | Password Mismatch | 1. Open registration page.  2. Enter password and confirm password that do not match.  3. Click submit button | Error message displayed | Error message displayed | Pass | None |
| TC05 | SQL Injection Attempt | 1. Open registration page  2. Enter SQL injection code in any text field.  3. Click submit button. | Error message displayed / No registration | Error message displayed / No registration | Pass | None |

**8.2.5 Test Plan ID:** TP05

**Test Module:** User Login Form

**Type of Testing:** Validation Testing

**Test Description:** User name and password

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case**  **ID** | **Test Scenario** | **Test Steps** | **Expected**  **Result** | **Actual Result** | **Status** | **Remarks** |
| TC01 | Valid Login | 1. Open login page.  2. Enter valid username and password.  3. Click login button. | User successfully logs in | User successfully logged in | Pass | None |
| TC02 | Invalid Username | 1. Open login page.  2. Enter invalid username and valid password.  3. Click login button. | Error message displayed | Error message displayed | Pass | None |
| TC03 | Invalid Password | 1. Open login page.  2. Enter valid username and invalid password.  3. Click login button. | Error message displayed | Error message displayed | Pass | None |
| TC04 | Blank Username and Password | 1. Open login page.  2. Leave username and password fields blank.  3. Click login button. | Error message displayed | Error message displayed | Pass | None |
| TC05 | SQL Injection Attempt | 1. Open login page.  2. Enter SQL injection code in username and password fields.  3. Click login button. | Error message displayed / No login | Error message displayed / No login | Pass | None |

**8.2.6 Test Plan ID:** TP06

**Test Module:** Booking Form

**Type of Testing:** Unit Testing

**Test Description:** User details and Service detailsTest Case Details

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Scenario** | **Test Steps** | **Expected Result** | **Actual Result** | **Status** | **Remarks** |
| TC01 | Valid Booking | 1. Open booking form.  2. Enter valid user and service details.  3. Click submit button | Booking successfully recorded | Booking successfully recorded | Pass | None |
| TC02 | Missing Required Fields | 1. Open booking form.  2. Leave required fields blank.  3. Click submit button | Error message displayed | Error message displayed | Pass | None |
| TC03 | Invalid Date Format | 1. Open booking form.  2. Enter invalid date format  3. Click submit button | Error message displayed | Error message displayed | Pass | None |
| TC04 | Invalid Service ID | 1. Open booking form  2. Enter invalid service ID  3. Click submit button | Error message displayed | Error message displayed | Pass | None |
| TC05 | SQL  Injection Attempt | 1. Open booking form  2. Enter SQL injection code in any text field  3. Click submit button | Error message displayed / No booking | Error message displayed / No booking | Pass | None |

**8.2.7 Test Plan ID:** TP07

**Test Module:** Payment Form

**Type of Testing:** Unit Testing

**Test Description:** User card details

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Scenario** | **Test Steps** | **Expected Result** | **Actual Result** | **Status** | **Remarks** |
| TC01 | Valid Payment | 1. Open payment form  2. Enter valid card details.  3. Click submit button | Payment successfully processed | Payment successfully processed | Pass | None |
| TC02 | Missing Required Fields | 1. Open payment form  2. Leave required fields blank  3. Click submit button | Error message displayed | Error message displayed | Pass | None |
| TC03 | Invalid Card Number | 1. Open payment form  2. Enter invalid card number  3. Click submit button | Error message displayed | Error message displayed | Pass | None |
| TC04 | Expired Card | 1. Open payment form  2. Enter expired card details  3. Click submit button | Error message displayed | Error message displayed | Pass | None |
| TC05 | SQL  Injection Attempt | 1. Open payment form  2. Enter SQL injection code in any text field  3. Click submit button | Error message displayed / No payment | Error message displayed / No payment | Pass | None |

## 8.2 BUG REPORT

Bug report providing any important information uncovered by the tests accomplished and including assessments of the quality to the testing effort, the quality of the software under test and statistics derived from incident reports

|  |  |  |  |
| --- | --- | --- | --- |
| **Bug id** | **Test module** | **Details** | **Bug status** |
| B01 | User module | User can entered invalid username and password. It shows incorrect user name and password | Success |

## CHAPTER 9

## CONCLUSION

The project entitled as “**ENHANCED PORTAL FOR HOME SERVICES**” concluded that can be deduced from automation of the entire system improves the efficiency it provides a friendly graphical user interface which proves to be better when compared to the existing system. It gives appropriate access to the authorized users depending on their permissions. It effectively overcomes the delay in communications. Updating of information becomes so easier. This project can be providing good interaction & communication facilities between customer & Servicer. System security, data security and reliability are the striking features. The System has adequate scope for modification in future if it is necessary.

**CHAPTER 10**

**FUTURE ENHANCEMENT**

In future we can develop this project in android application with extra features like service provider location tracking and user ratings system.

In the future, this project can be further developed into an Android application with additional features such as service provider location tracking and a user ratings system. Additionally, the following enhancements can be made to the website:

* **Subscription Plans**: Introduce subscription plans for regular customers with added benefits such as discounts, priority service, and exclusive offers
* **Multiple Payment Gateways**: Integrate multiple payment gateways to offer users a variety of payment options, including digital wallets, credit/debit cards, and net banking.
* **Enhanced Security Measures**: Implement enhanced security measures such as two- factor authentication (2FA) and data encryption to further protect user information.
* **Real-Time Notifications**: Enable real-time notifications to keep users informed about the status of their service requests, special promotions, and important updates.
* **Integration with Social Media**: Allow users to sign in using their social media accounts and share their service experiences on social platforms.

**APPENDIX 1**

**APPENDIX 1: SAMPLE SOURCE CODE**

from flask import Flask, render\_template, flash, request, session  
  
import mysql.connector  
import sys  
  
app = Flask(\_\_name\_\_)  
app.config['DEBUG']  
app.config['SECRET\_KEY'] = '7d441f27d441f27567d441f2b6176a'  
  
  
@app.route("/")  
defhomepage():  
return render\_template('index.html')  
  
@app.route("/Home")  
defHome():  
return render\_template('index.html')  
  
@app.route("/AdminLogin")  
defDoctorLogin():  
return render\_template('AdminLogin.html')  
  
@app.route("/NewServicer")  
defNewServicer():  
return render\_template('NewServicer.html')  
  
@app.route("/ServicerLogin")  
defServicerLogin():  
return render\_template('ServicerLogin.html')

@app.route("/UserLogin")  
defUserLogin():  
return render\_template('UserLogin.html')  
  
  
@app.route("/NewUser")  
defNewUser():  
return render\_template('NewUser.html')  
  
  
@app.route("/NewProduct")  
defNewProduct():  
return render\_template('NewProduct.html')  
  
  
@app.route("/AdminHome")  
defAdminHome():  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM regtb ")  
 data = cur.fetchall()  
return render\_template('AdminHome.html', data=data)  
  
@app.route("/ABookingInfo")  
defABookingInfo():  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM booktb ")  
 data1 = cur.fetchall()  
return render\_template('ABookingInfo.html', data1=data1)

@app.route("/AFeedBack")  
defAFeedBack():  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM feedtb ")  
 data = cur.fetchall()  
return render\_template('AFeedBack.html', data=data)  
  
@app.route("/adminlogin", methods=['GET', 'POST'])  
defadminlogin():  
if request.method == 'POST':  
if request.form['uname'] == 'admin' or request.form['password'] == 'admin':  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM regtb ")  
 data = cur.fetchall()  
 flash("Login successfully")  
return render\_template('AdminHome.html', data=data)  
  
else:  
 flash("UserName Or Password Incorrect!")  
return render\_template('AdminLogin.html')  
  
@app.route("/AServicerInfo")  
defAServicerInfo():  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM employeetb ")  
 data = cur.fetchall()  
return render\_template('AServicerInfo.html', data=data)

@app.route("/newemp", methods=['GET', 'POST'])  
defnewemp():  
if request.method == 'POST':  
 name = request.form['name']  
 mobile = request.form['mobile']  
  
 email = request.form['email']  
  
 address = request.form['address']  
  
uname = request.form['uname']  
 password = request.form['password']  
ptype = request.form['ptype']  
  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cursor = conn.cursor()  
cursor.execute(  
"INSERT INTO employeetb VALUES ('" + name + "','" + email + "','" + mobile + "','" + address + "','" + uname + "','" + password + "','" + ptype + "')")  
conn.commit()  
conn.close()  
 flash('Employee Info Register Successfully')  
  
return render\_template('NewServicer.html')  
  
@app.route("/emplogin", methods=['GET', 'POST'])  
defemplogin():  
if request.method == 'POST':  
 username = request.form['uname']  
 password = request.form['password']  
 session['ename'] = request.form['uname']

conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cursor = conn.cursor()  
cursor.execute("SELECT \* from employeetb where username='" + username + "' and Password='" + password + "'")  
 data = cursor.fetchone()  
if data is None:  
  
 flash('Username or Password is wrong')  
return render\_template('ServicerLogin.html', data=data)  
else:  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM employeetb where username='" + username + "' and Password='" + password + "'")  
 data = cur.fetchall()  
 flash("Login successfully")  
return render\_template('ServicerHome.html', data=data)  
  
@app.route("/ServicerHome")  
defServicerHome():  
 username = session['ename']  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
# cursor = conn.cursor()  
cur = conn.cursor()  
cur.execute("SELECT \* FROM employeetb where username='" + username + "' ")  
 data = cur.fetchall()  
return render\_template('ServicerHome.html', data=data)  
  
@app.route("/Remove")  
defRemove():

id = request.args.get('id')  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cursor = conn.cursor()  
cursor.execute(  
"delete from protb where id='" + id + "'")  
conn.commit()  
conn.close()  
  
 flash('Product info Remove Successfully!')  
  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM protb ")  
 data = cur.fetchall()  
return render\_template('EProductInfo.html', data=data)  
  
@app.route("/SBookInfo")  
defSBookInfo():  
 username = session['ename']  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM booktb where ServicerName='" + username + "' and Status ='waiting' ")  
 data = cur.fetchall()  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM booktb where ServicerName='" + username + "' and Status !='waiting' ")  
 data1 = cur.fetchall()  
return render\_template('SBookInfo.html', data=data, data1=data1)  
  
  
@app.route("/SPaymentInfo")  
defSPaymentInfo():  
 username = session['ename']  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM booktb where ServicerName='" + username + "' and Status ='Paid' ")  
 data = cur.fetchall()  
return render\_template('SPaymentInfo.html', data1=data)  
  
@app.route("/Accept")  
defAccept():  
 id = request.args.get('id')  
uname = request.args.get('uname')  
 session['bid'] = id  
 session['uname'] = uname  
  
return render\_template('Amount.html')  
  
@app.route("/uamount", methods=['GET', 'POST'])  
defuamount():  
if request.method == 'POST':  
amt = request.form['amt']  
 id = session['bid']  
uname = session['uname']  
  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cursor = conn.cursor()  
cursor.execute(  
"Update booktb set Status='Accept' , Amount='" + amt + "' where id='" + id + "'")  
conn.commit()  
conn.close()

conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cursor = conn.cursor()  
cursor.execute("SELECT \* from regtb where username='" + uname + "' ")  
 data = cursor.fetchone()  
if data:  
sendmail(data[1], 'Booking Status Is Approved! Amount : ' + str(amt))  
  
 username = session['ename']  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM booktb where ServicerName='" + username + "' and Status ='waiting' ")  
 data = cur.fetchall()  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM booktb where ServicerName='" + username + "' and Status !='waiting' ")  
 data1 = cur.fetchall()  
  
return render\_template('SBookInfo.html', data=data, data1=data1)  
  
  
@app.route("/Reject")  
defReject():  
 id = request.args.get('id')  
uname = request.args.get('uname')  
  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cursor = conn.cursor()

cursor.execute(  
"Update booktb set Status='Reject' where id='" + id + "'")  
conn.commit()  
conn.close()  
  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cursor = conn.cursor()  
cursor.execute("SELECT \* from regtb where username='" + uname + "' ")  
 data = cursor.fetchone()  
if data:  
sendmail(data[1], 'Booking Status Is Reject!')  
  
 username = session['ename']  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM booktb where ServicerName='" + username + "' and Status ='waiting' ")  
 data = cur.fetchall()  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM booktb where ServicerName='" + username + "' and Status !='waiting' ")  
 data1 = cur.fetchall()  
  
return render\_template('SBookInfo.html', data=data, data1=data1)  
  
@app.route("/newuser", methods=['GET', 'POST'])  
defnewuser():  
if request.method == 'POST':  
 name = request.form['name']  
 mobile = request.form['mobile']

email = request.form['email']  
 address = request.form['address']  
  
uname = request.form['uname']  
 password = request.form['password']  
  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cursor = conn.cursor()  
cursor.execute(  
"INSERT INTO regtb VALUES ('" + name + "','" + email + "','" + mobile + "','" + address + "','" + uname + "','" + password + "')")  
conn.commit()  
conn.close()  
 flash('User Register successfully')  
  
return render\_template('UserLogin.html')  
  
@app.route("/userlogin", methods=['GET', 'POST'])  
defuserlogin():  
if request.method == 'POST':  
 username = request.form['uname']  
 password = request.form['password']  
 session['uname'] = request.form['uname']  
  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cursor = conn.cursor()  
cursor.execute("SELECT \* from regtb where username='" + username + "' and Password='" + password + "'")  
 data = cursor.fetchone()  
if data is None:  
 flash('Username or Password is wrong')

return render\_template('UserLogin.html')  
else:  
  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM regtb where username='" + username + "' and Password='" + password + "'")  
 data = cur.fetchall()  
 flash("Login successfully")  
  
return render\_template('UserHome.html', data=data)  
  
@app.route("/UserHome")  
defUserHome():  
uname = session['uname']  
  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM regtb where username='" + uname + "' ")  
 data = cur.fetchall()  
  
return render\_template('UserHome.html', data=data)  
@app.route("/Search")  
defSearch():  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM employeetb ")  
 data = cur.fetchall()  
return render\_template('Search.html', data=data)  
@app.route("/search", methods=['GET', 'POST'])  
defsearch():

if request.method == 'POST':  
ptype = request.form['ptype']  
  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM employeetb where ptype='" + ptype + "' ")  
 data = cur.fetchall()  
  
return render\_template('Search.html', data=data)  
  
@app.route("/Book")  
defBook():  
ename = request.args.get('ename')  
 session["ename"] = ename  
return render\_template('Book.html', ename=ename)  
  
@app.route("/book", methods=['GET', 'POST'])  
defbook():  
if request.method == 'POST':  
uname = session['uname']  
ename = request.form['ename']  
sname = request.form['sname']  
 date = request.form['date']  
 address = request.form['address']  
 info = request.form['info']  
  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cursor = conn.cursor()  
cursor.execute(  
"INSERT INTO Booktb VALUES ('','" + uname + "','" + ename + "','" + sname + "','" + date + "','" + address + "','" + info + "','waiting','')")

conn.commit()  
conn.close()  
  
 flash(' Service Book successfully')  
  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM Booktb where username='" + uname + "' ")  
 data = cur.fetchall()  
  
return render\_template('UserBookInfo.html', data=data)  
  
@app.route("/UserBookInfo")  
defUserBookInfo():  
uname = session['uname']  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM Booktb where username='" + uname + "' and Status='Accept' ")  
 data = cur.fetchall()  
  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM Booktb where username='" + uname + "' and Status !='Accept' ")  
 data1 = cur.fetchall()  
return render\_template('UserBookInfo.html', data=data, data1=data1)

@app.route("/pay")  
defpay():  
 id = request.args.get('id')

amt = request.args.get('amt')  
st = request.args.get('st')  
 session['bid'] = id  
  
if st == 'Accept':  
return render\_template('PayAmount.html', amt=amt)  
else:  
 flash("Booking Status " + str(st))  
return render\_template('UserBookInfo.html')  
  
  
@app.route("/payment", methods=['GET', 'POST'])  
defpayment():  
if request.method == 'POST':  
 id = session['bid']  
  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cursor = conn.cursor()  
cursor.execute(  
"update booktb set status='Paid' where id='" + id + "' ")  
conn.commit()  
conn.close()  
 flash('Payment Booked successfully')  
  
uname = session['uname']  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cur = conn.cursor()  
cur.execute("SELECT \* FROM Booktb where username='" + uname + "' and Status='Accept' ")  
 data = cur.fetchall()  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')

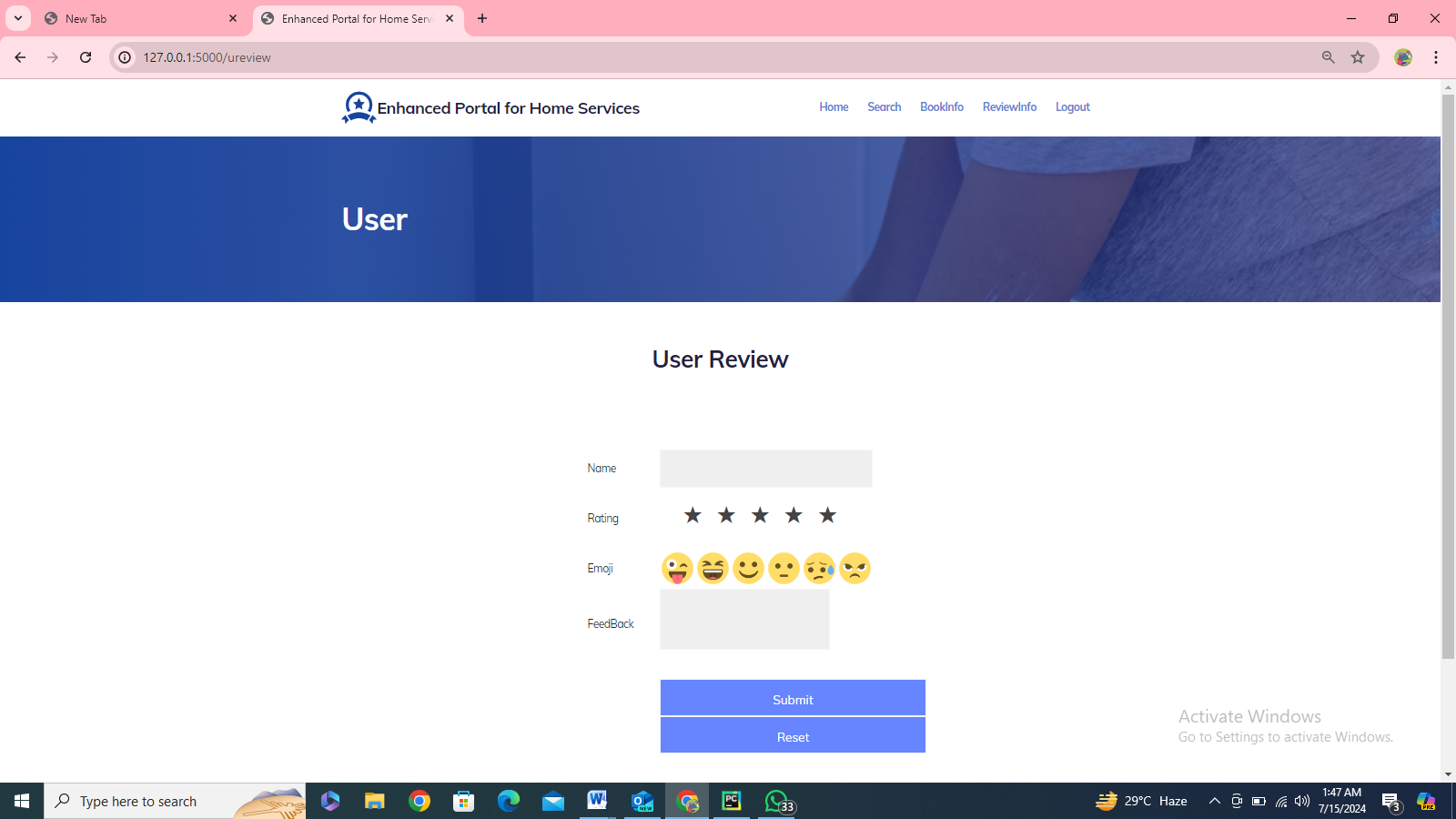
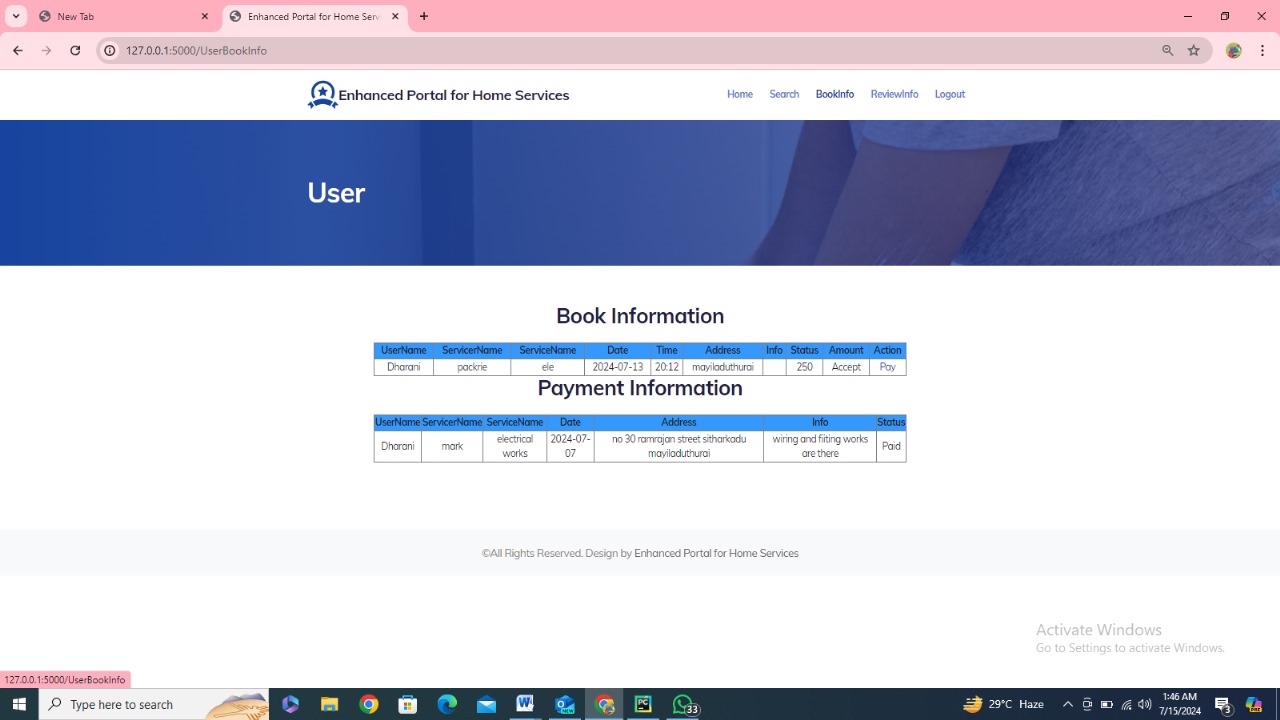
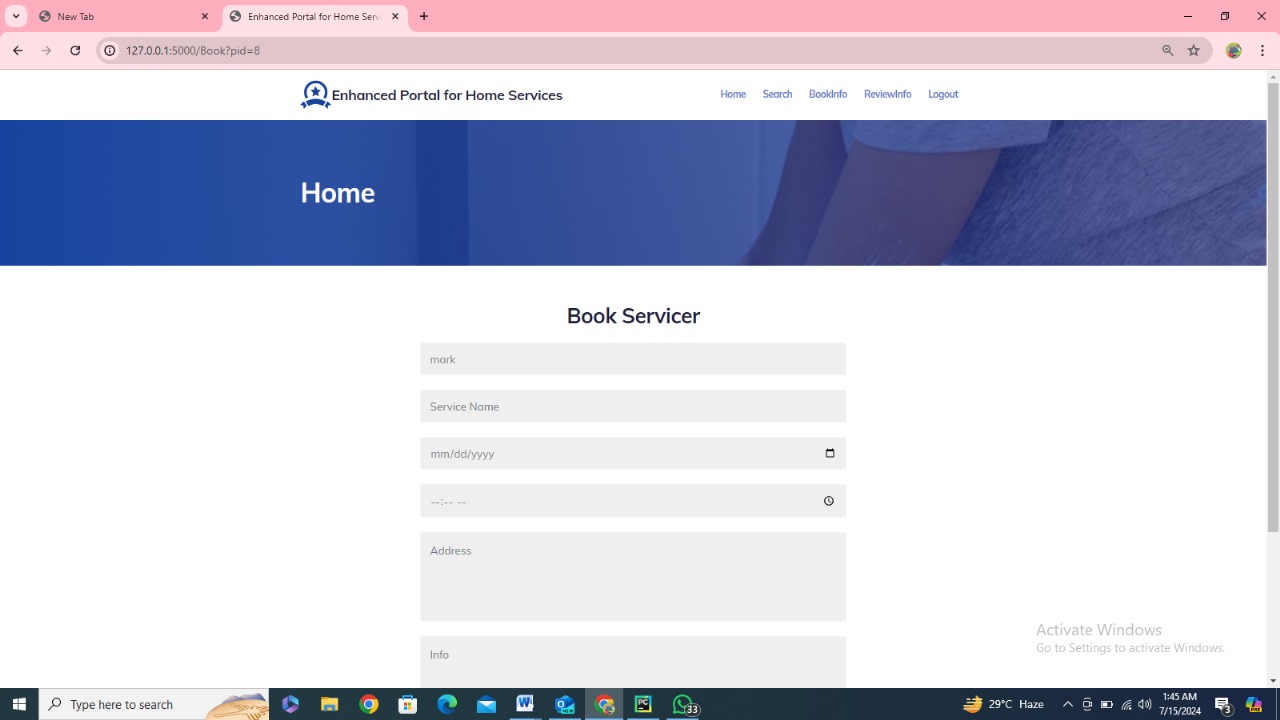
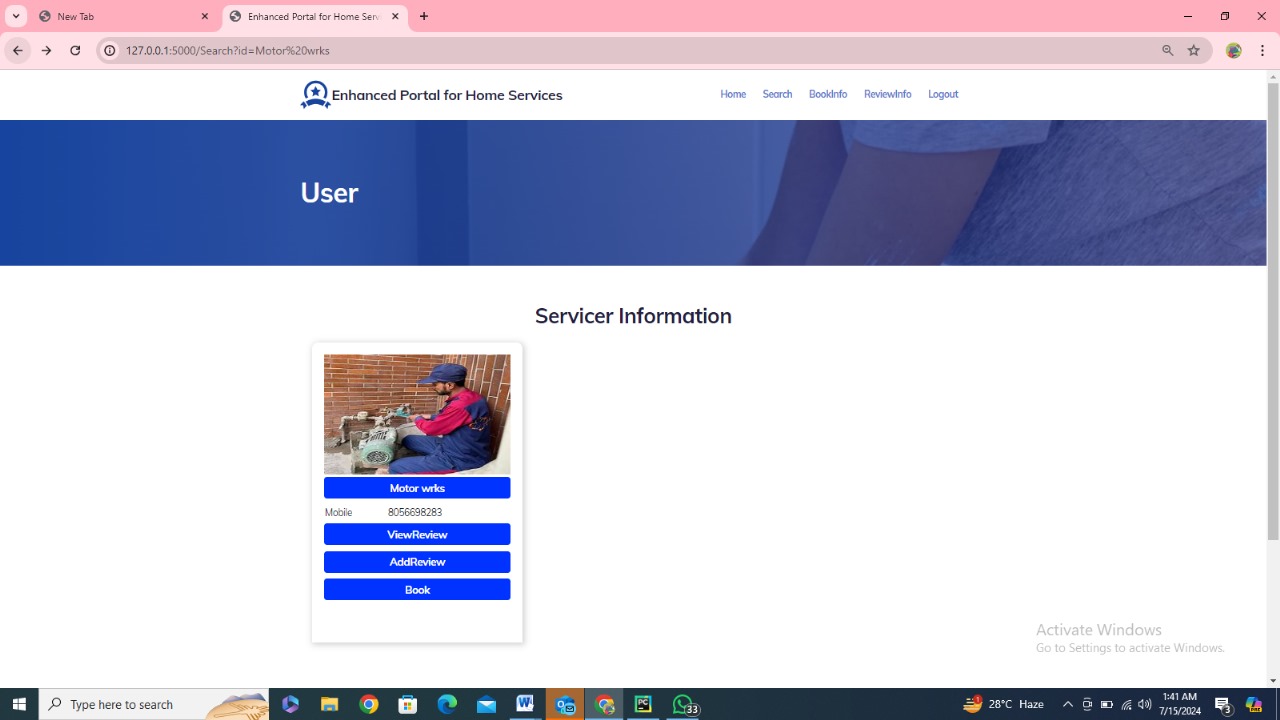
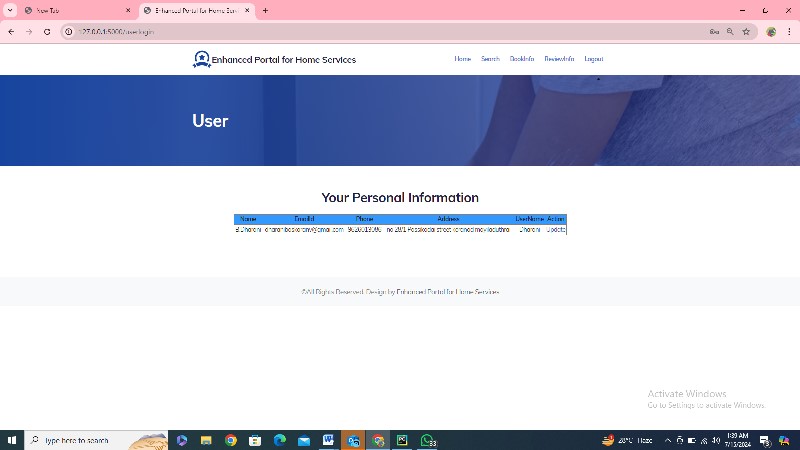
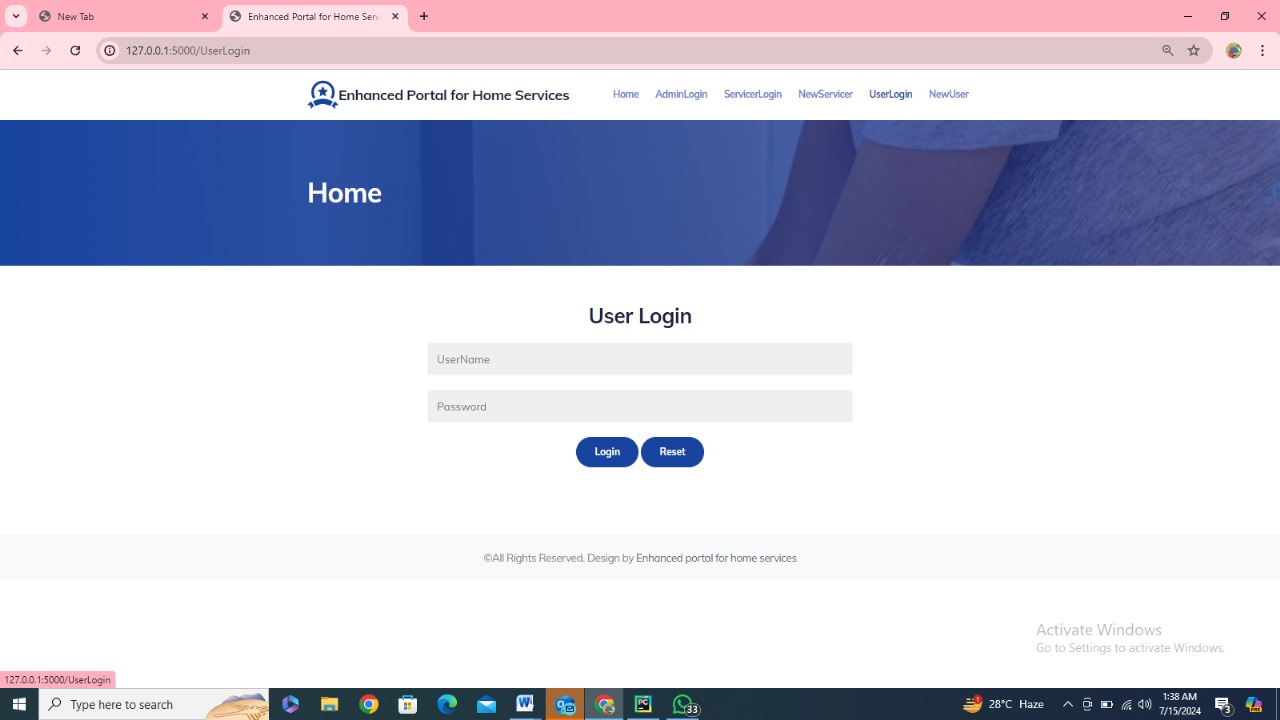
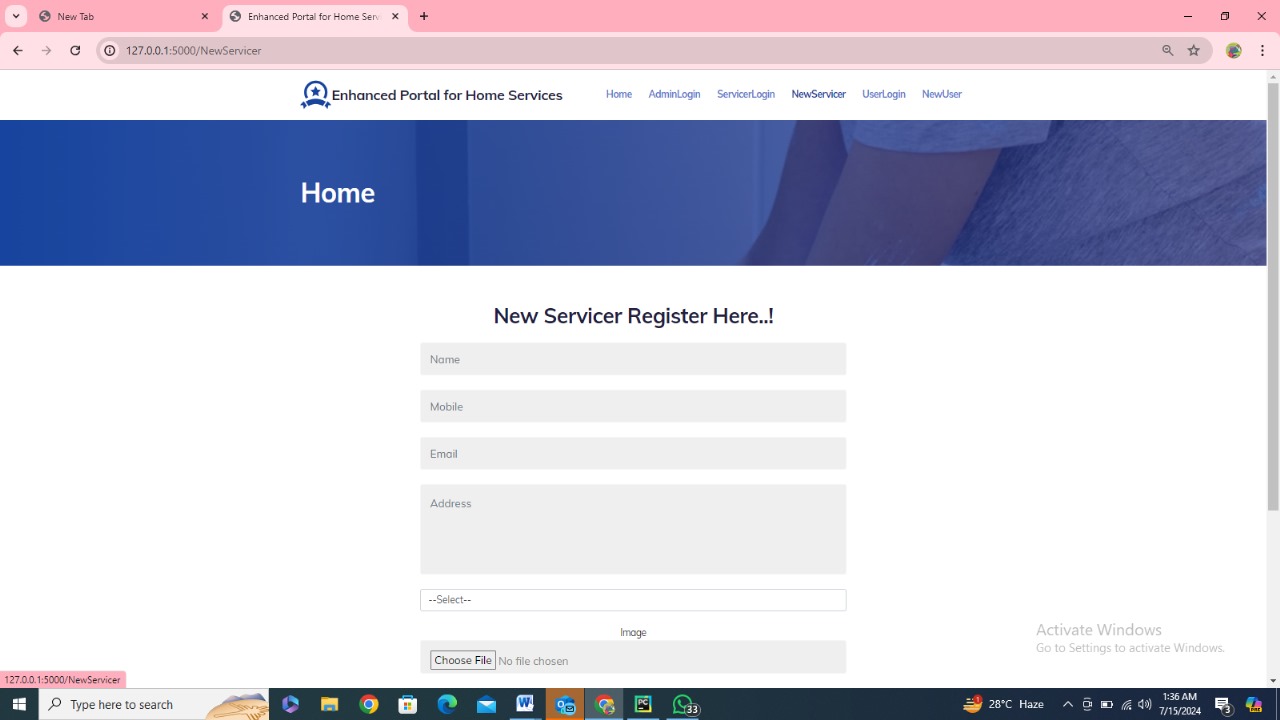
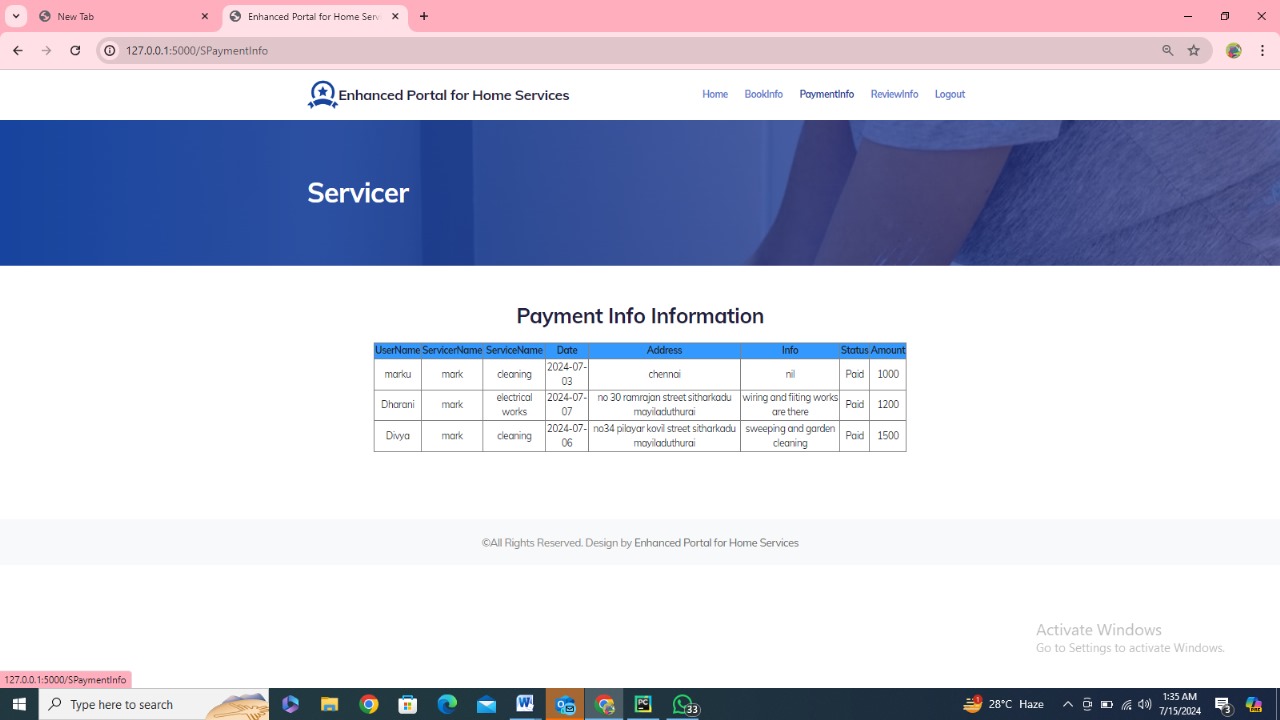
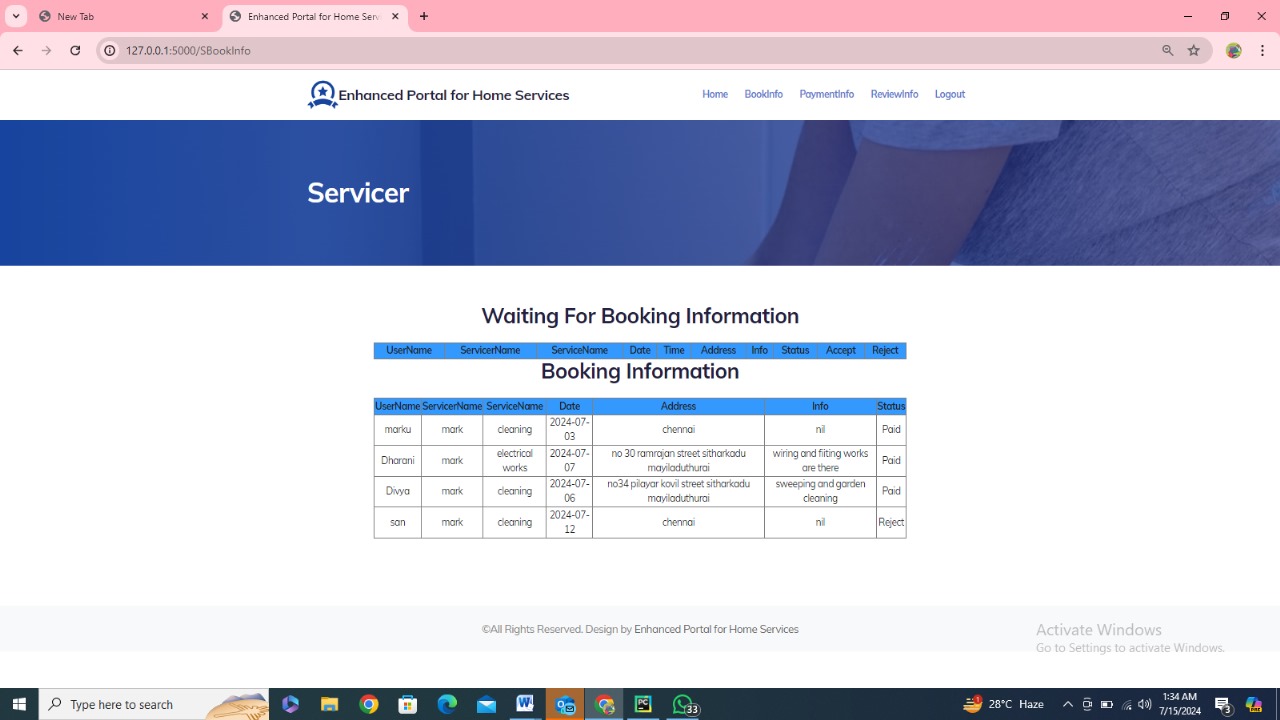
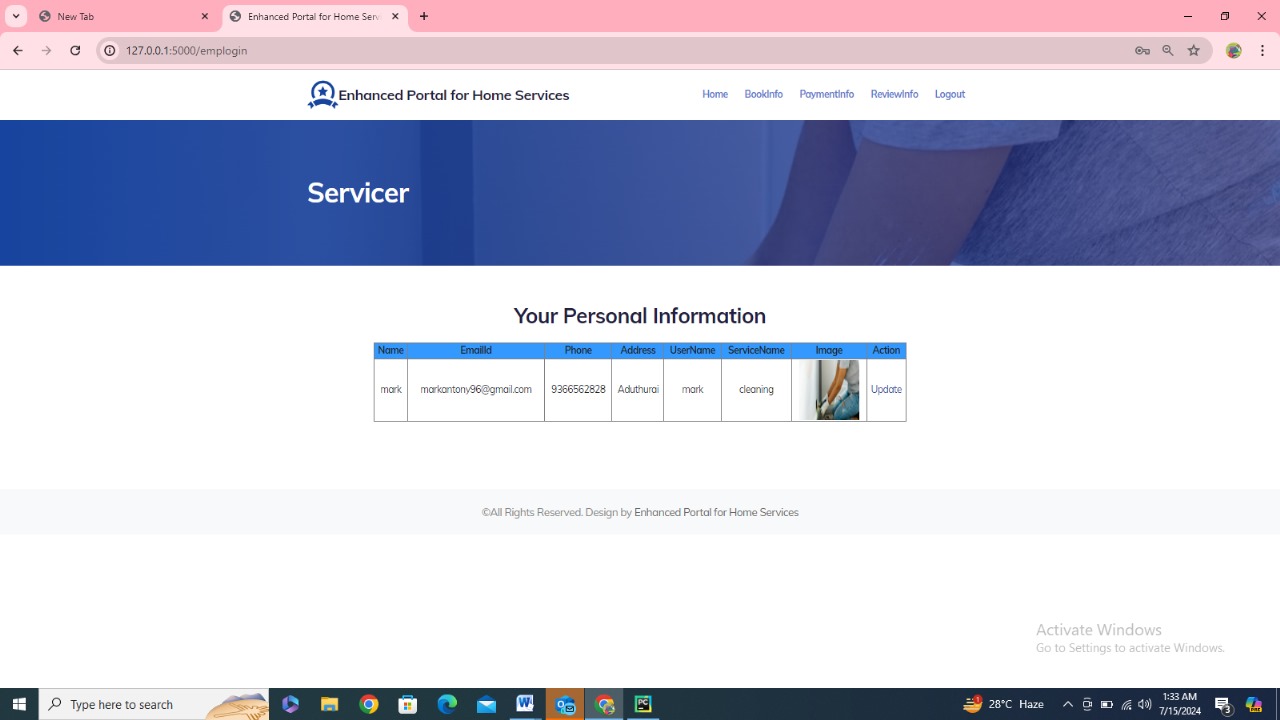
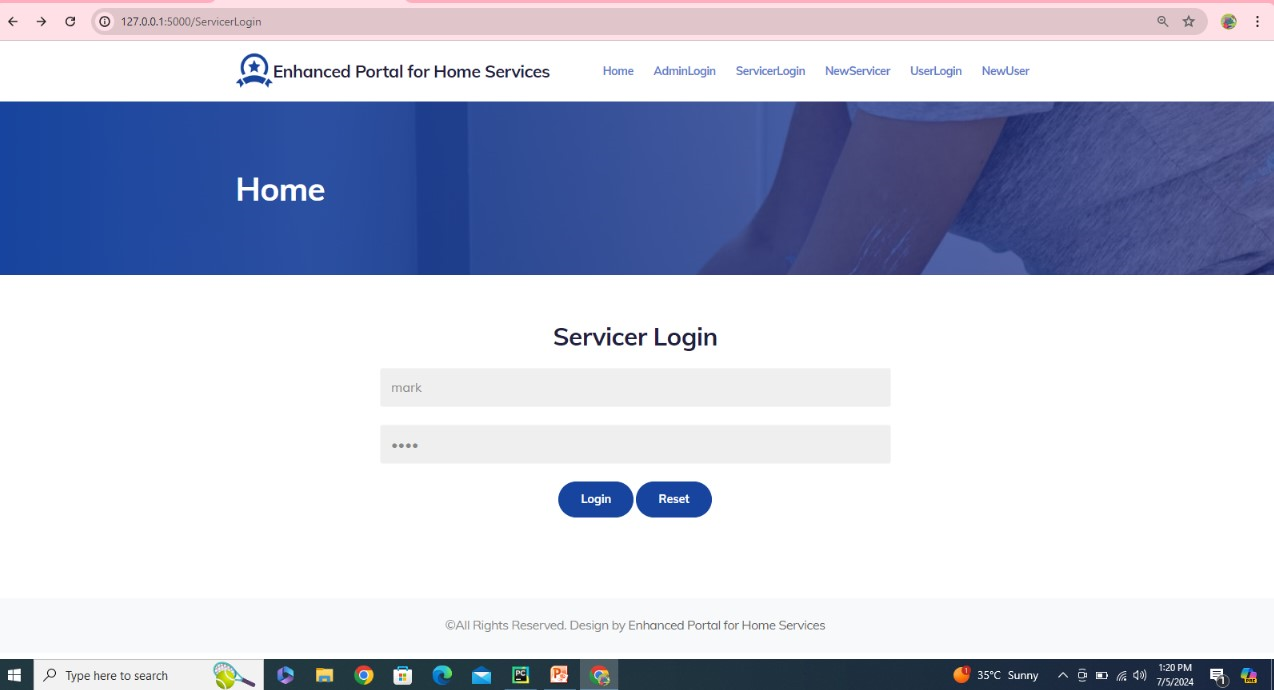
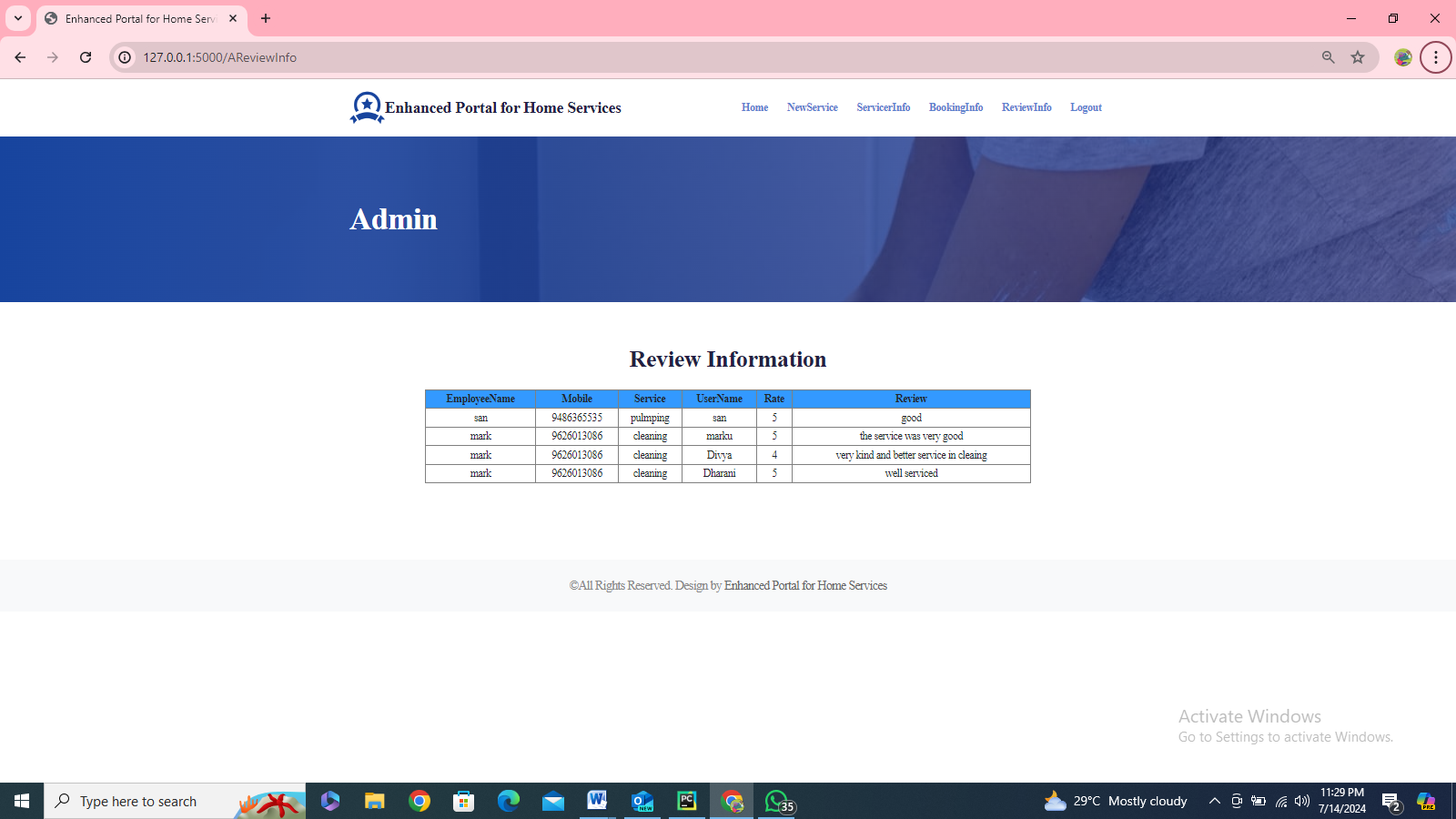
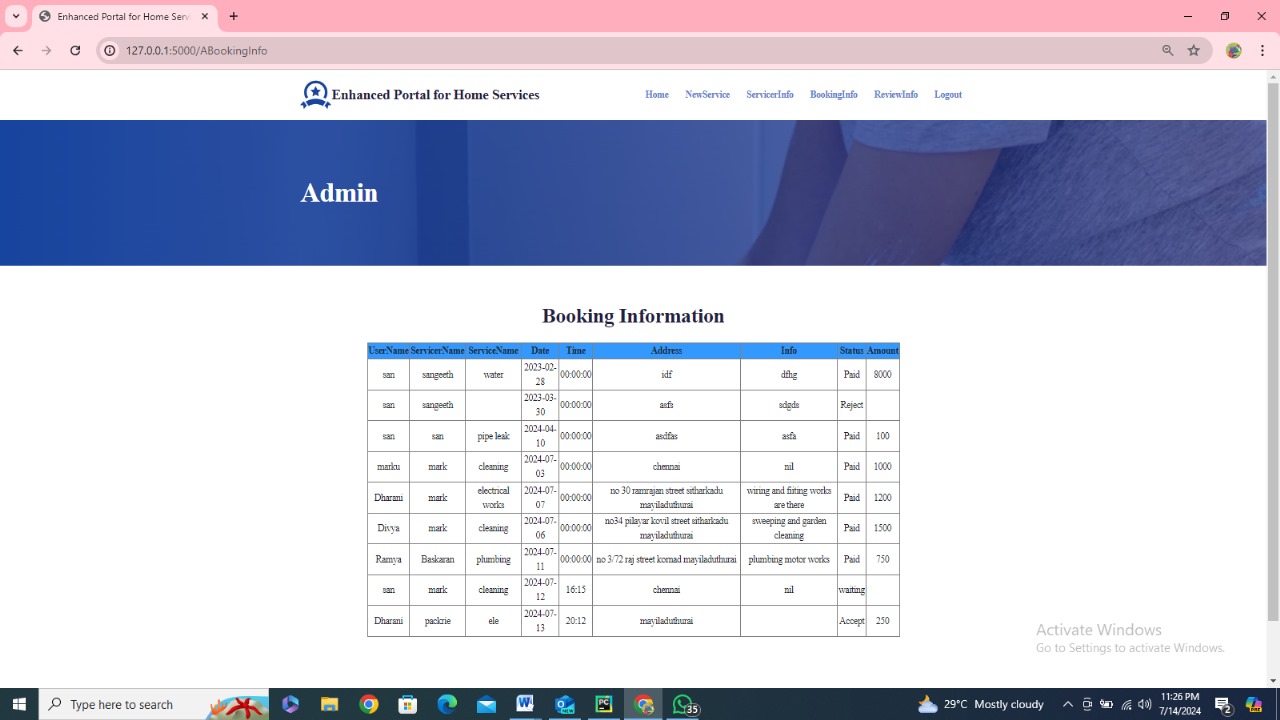
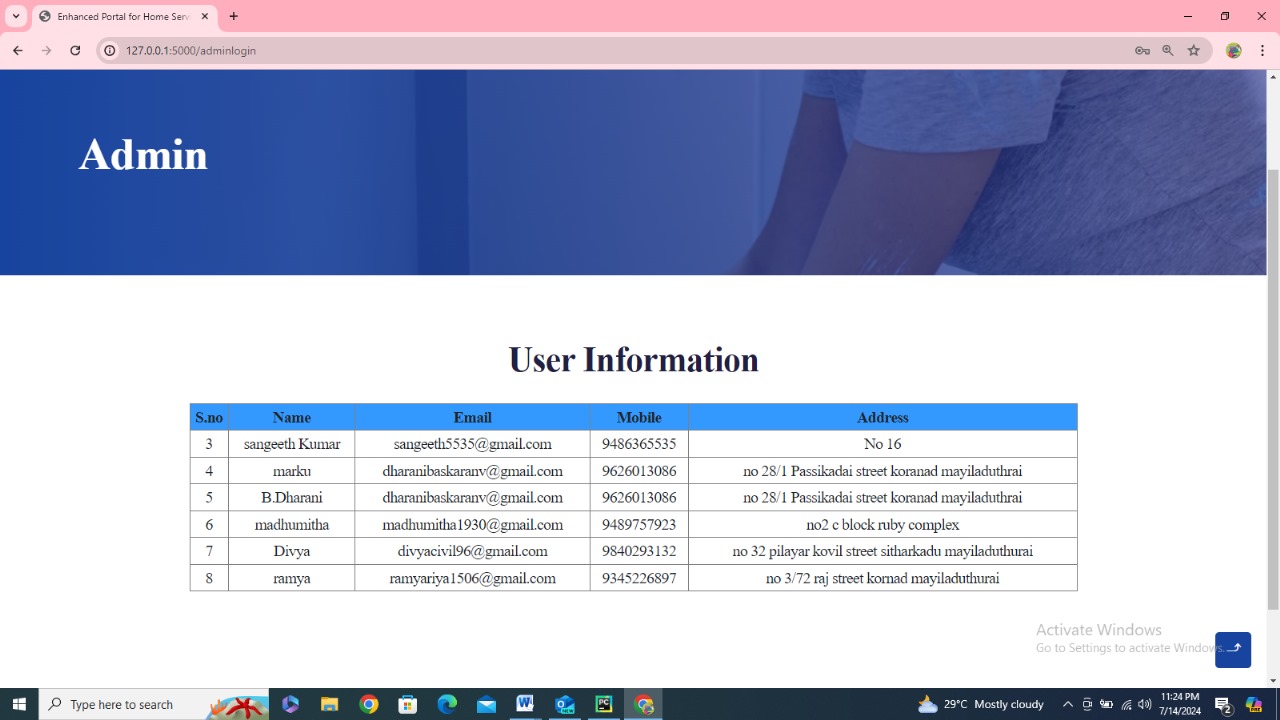
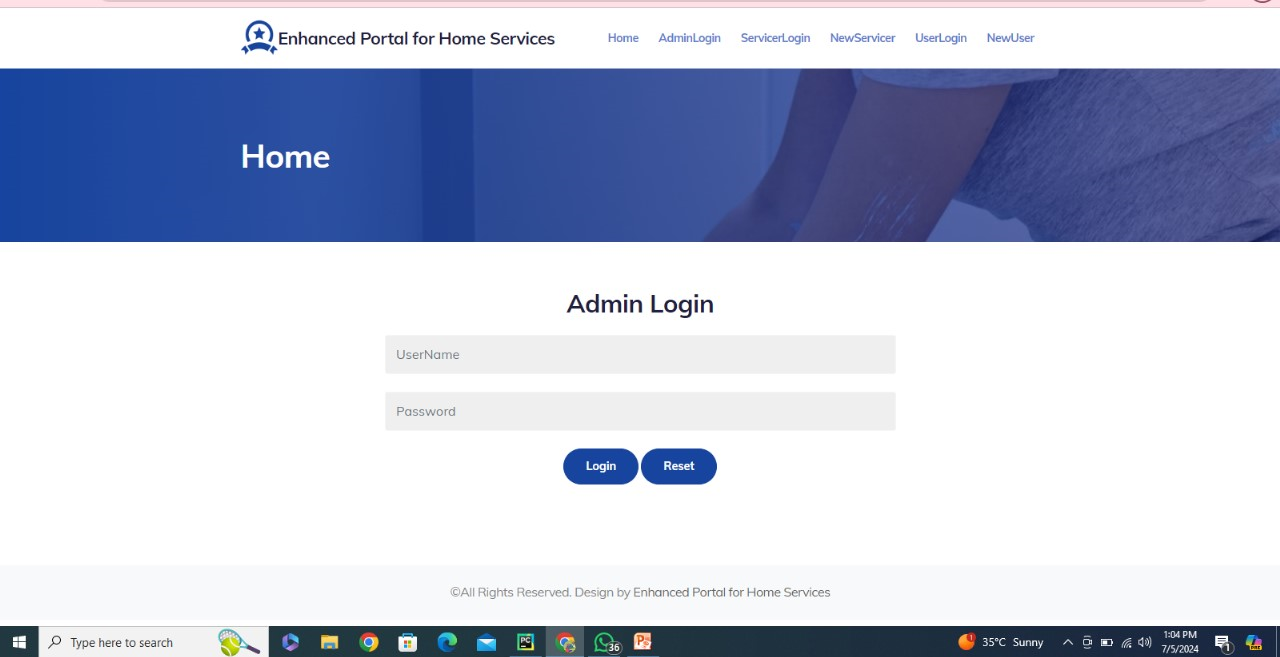
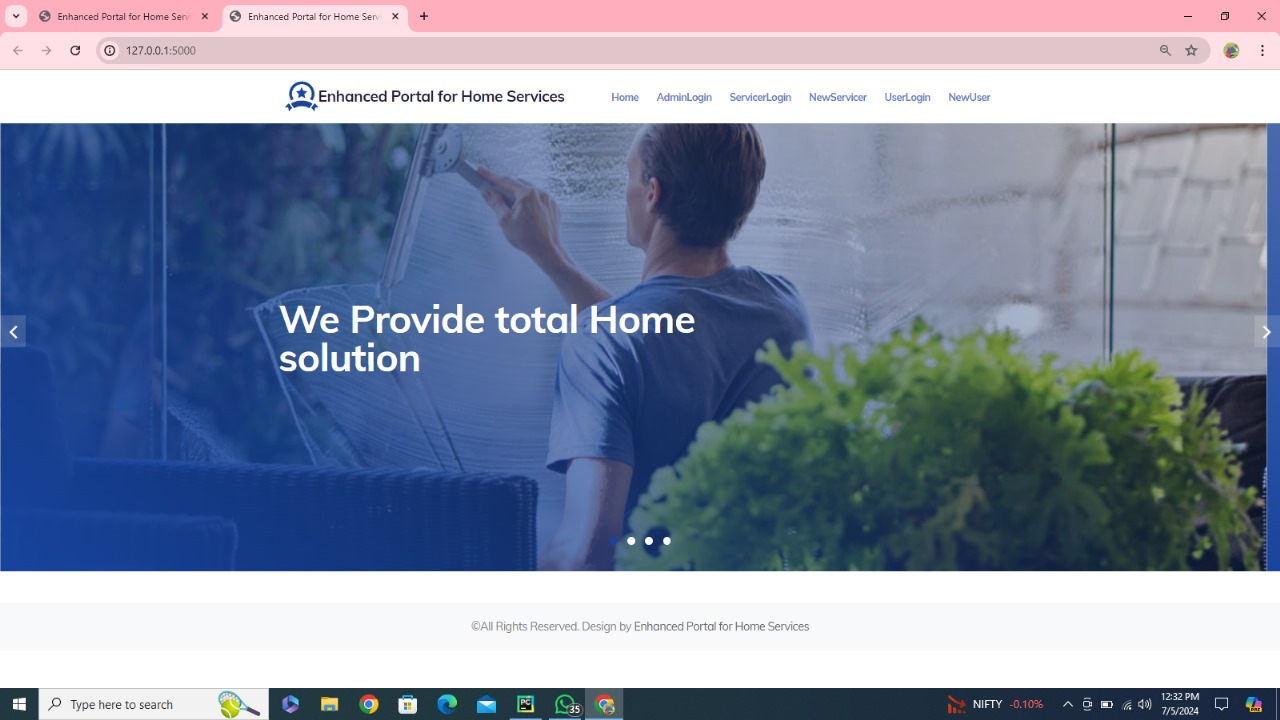
cur = conn.cursor()  
cur.execute("SELECT \* FROM Booktb where username='" + uname + "' and Status !='Accept' ")  
 data1 = cur.fetchall()  
  
return render\_template('UserBookInfo.html', data=data, data1=data1)  
  
@app.route("/FeedBack")  
defFeedBack():  
uname = session['uname']  
  
return render\_template('FeedBack.html', uname=uname)  
@app.route("/newfeed", methods=['GET', 'POST'])  
defnewfeed():  
if request.method == 'POST':  
 info = request.form['info']  
  
 conn = mysql.connector.connect(user='root', password='', host='localhost', database='1handydbpy')  
 cursor = conn.cursor()  
cursor.execute(  
"INSERT INTO feedtb VALUES ('','" + session['uname'] + "','" + info + "')")  
conn.commit()  
conn.close()  
 flash('New FeedBack Register Successfully')  
return render\_template('FeedBack.html')  
defsendmail(Mailid, message):  
import smtplib  
from email.mime.multipartimport MIMEMultipart  
from email.mime.textimport MIMEText  
from email.mime.baseimport MIMEBase  
from email import encoders  
fromaddr = "sampletest685@gmail.com"  
toaddr = Mailid

# instance of MIMEMultipart  
msg = MIMEMultipart()  
  
# storing the senders email address  
msg['From'] = fromaddr  
  
# storing the receivers email address  
msg['To'] = toaddr  
  
# storing the subject  
msg['Subject'] = "Alert"  
  
# string to store the body of the mail  
body = message  
  
# attach the body with the msg instance  
msg.attach(MIMEText(body, 'plain'))  
  
# creates SMTP session  
s = smtplib.SMTP('smtp.gmail.com', 587)  
  
# start TLS for security  
s.starttls()  
# Authentication  
s.login(fromaddr, "hneucvnontsuwgpj")  
# Converts the Multipart msg into a string  
text = msg.as\_string()  
# sending the mail  
s.sendmail(fromaddr, toaddr, text)  
  
# terminating the session  
s.quit()  
if \_\_name\_\_ == '\_\_main\_\_':  
app.run(debug=True, use\_reloader=

**APPENDIX 2**

**SAMPLE SCREENSHOTS**

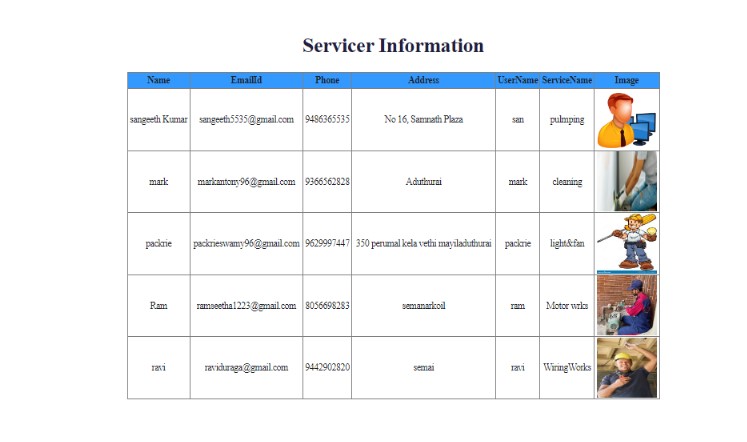
**APPENDIX 2: SAMPLE SCREENSHOTS**



**APPENDIX 3**

**SAMPLE REPORTS**

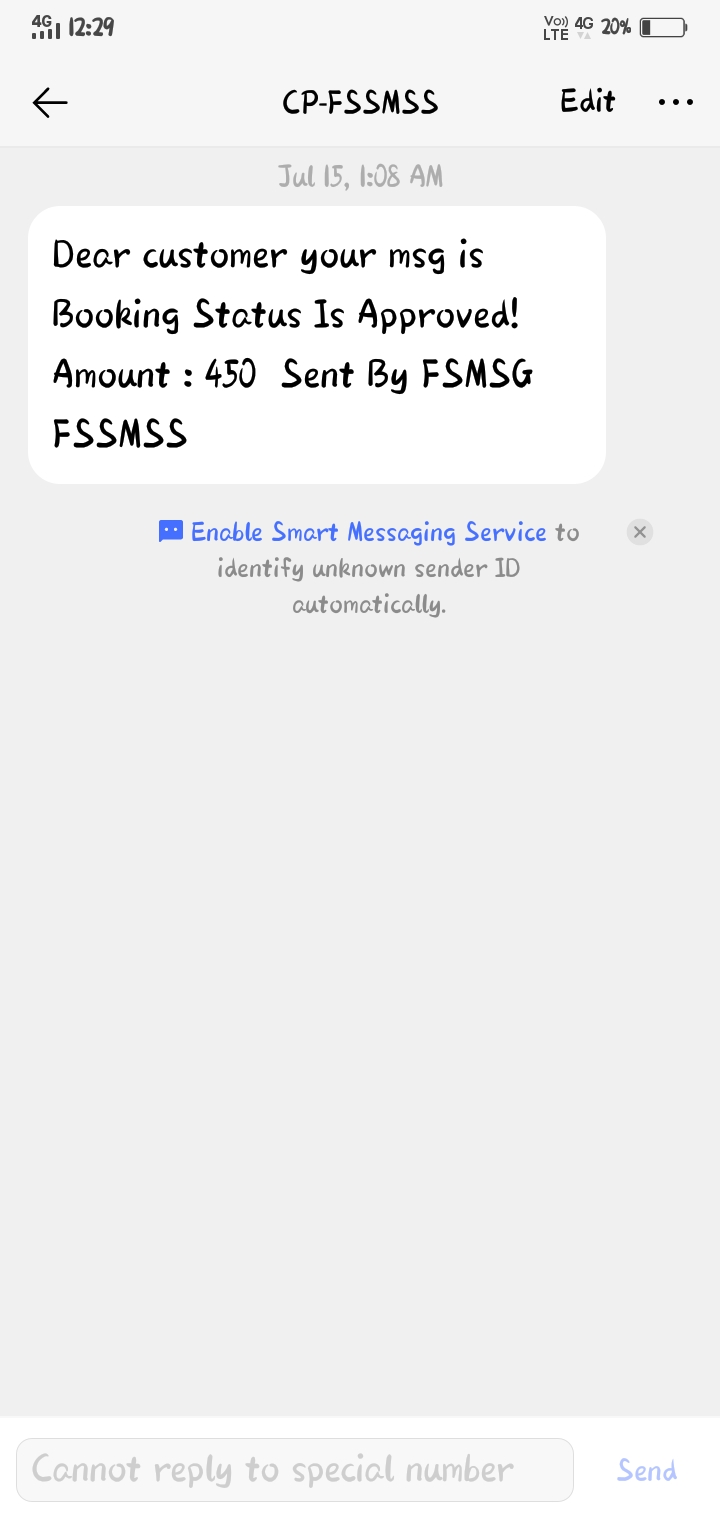
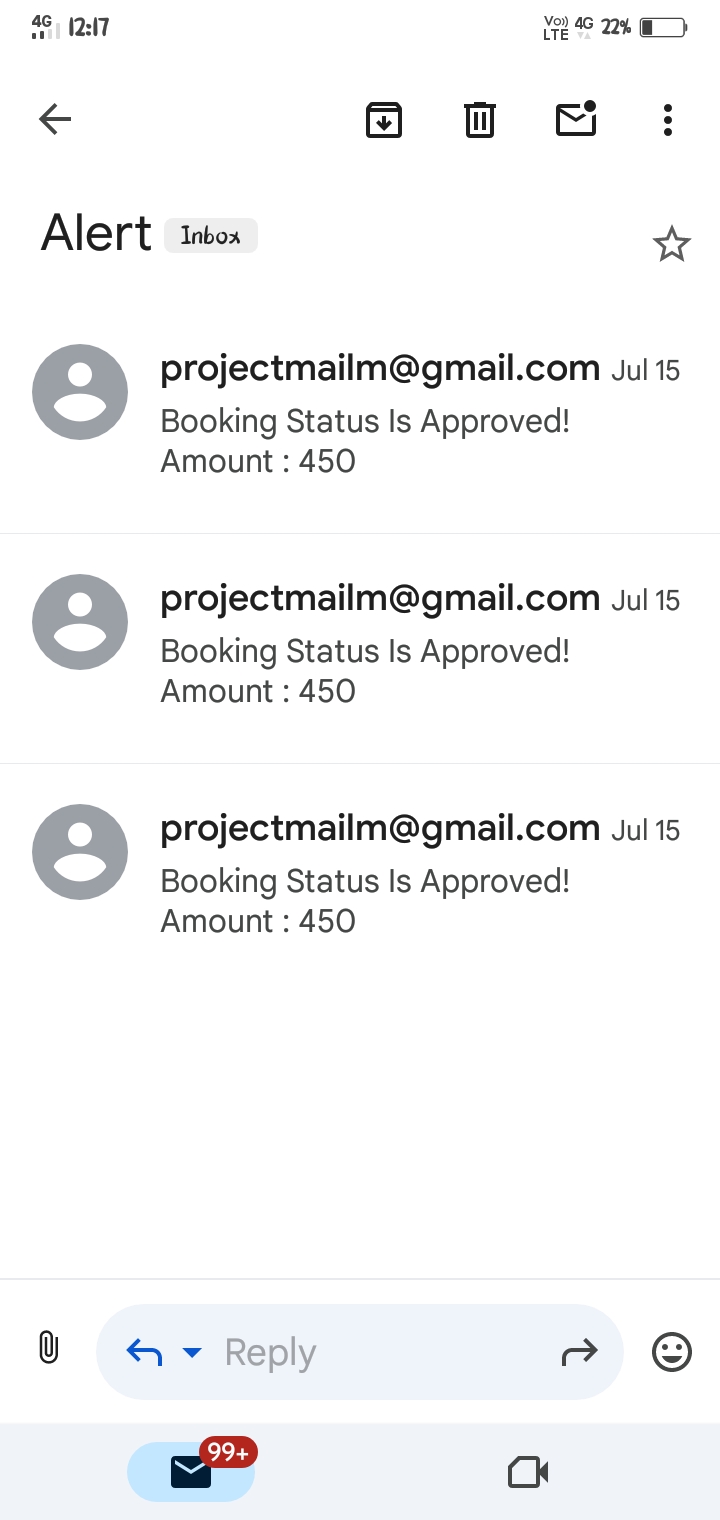
**SERVICER WISE REPORT**



**BOOKING WISE REPORT**



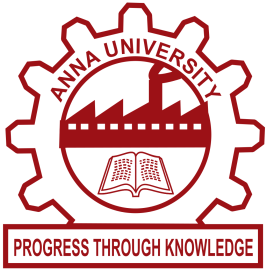
**NOTIFICATION**



## REFERENCES

**References**

1. Heinold, Brian (2021), "A practical introduction to Python programming" .
2. Kneusel, 2021, Ronald T. Practical deep learning: “A Python-based introduction” No Starch Press.
3. Dhruv, Akshit J., Reema Patel, and Nishant Doshi, (2021): 292-299 "Python: the most advanced programming language for computer science applications and Technology Publications”.
4. Sundnes, Joakim 2020,”Introduction to scientific programming with Python”. Springer Nature.
5. Hill, Christian 2020,“Learning scientific programming with Python”. Cambridge University Press.
6. .https://medium.com/javarevisited/10-free-python-tutorials-and-courses-from-google- microsoft-and-coursera-for-beginners.
7. https:/[/www.bestcoll](http://www.bestcolleges.com/bootcamps/guides/learn-python-free/)e[ges.com/bootcamps/guides/learn-python-free/](http://www.bestcolleges.com/bootcamps/guides/learn-python-free/)
8. https://[www.programiz.com/python-programming](http://www.programiz.com/python-programming)



**ENHANCED PORTAL FOR HOME SERVICES**

By

**B.DHARANI**

**Register No: 820322622011**

of

**A.V.C COLLEGE OF ENGINEERING**

Mayiladuthurai ,Mannampandal-609305

**A PROJECT REPORT**

Submitted to the

**FACULTY OF INFORMATION AND COMMUNICATION ENGINEERING**

*in partial fulfillment of the requirements for the award of the degree*

*of*

**MASTER OF COMPUTER APPLICATIONS**

**AUGUST 2024**

**CERTIFICATE**

 **A.V.C. COLLEGE OF ENGINEERING**

**MANNAMPANDAL, MAYILADUTHURAI (DT)**

**TAMILNADU - 609305.**

|Approved by AICTE | | Affiliated to Anna University, Chennai |

| Accredited by NAC (CSE,EEE,ECE & MECH) & NAAC with ‘B++’ Grade (2nd Cycle)|

| An ISO 9001:2015 Certified Institution |

**DEPARTMENT OF COMPUTER APPLICATIONS**

[A Recognized Research Centre of Anna University, Chennai]

**BONAFIDE CERTIFICATE**

Certified that this project report titled **“ENHANCED PORTAL FOR HOME SERVICES”** is the bonafide work of **Ms.** **B. DHARANI (Reg.No: 820322622011)** who carried out the project under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

**SUPERVISOR**  **HEAD OF THE DEPARTMENT**

**Place:**

**Date:**

Submitted for the Anna University Examination held on: ------------------------------------

**INTERNAL EXAMINER** **EXTERNAL EXAMINER**



**Date: 08.07.2024**

**To**

Head of Department,

Department of Computer Applications,

A.V.C. College of Engineering,

Mannampandal, Mayiladuthurai.

**Sir,**

**Sub: Project work Completion Certificate-Reg.**

This is to certify that **Ms.** **B. DHARANI** (**Reg. No.: 820322622011)** student of MCA from A.V.C. College Of Engineeringhas completed her final year project entitled, **“ENHANCED PORTAL FOR HOME SERVICES”** in our company during the period of **March 2024 to June 2024**.

Authorized Signature

(With Seal)

**DECLARATION**

**DECLARATION**

I, **B. DHARANI (Reg.No:820322622011),** the student of A.V.C College of Engineering, Department of Computer Applications, would like to declare that the project work entitled “**ENHANCED PORTAL FOR HOME SERVICES**” is the result of the original work done by me during the course of the study and is submitted on the partial fulfilment for the award of degree of “**MASTER OF COMPUTER APPLICATIONS”** of Anna University, Chennai.

**Place: Mannampandal Signature**

**Date: [B.DHARANI]**

**ABSTRACT**

**ABSTRACT**

The project entitled “ENHANCED PORTAL FOR HOME SERVICES”

This web Application portal being designed will help to integrate different functionalities of the organization by coordinating different users of the application through the single portal. Homeowners can access a range of essential services through a comprehensive platform known as the Home Service Provider Project. This digital system increases the efficiency of recruiting, managing, and locating various home services by utilizing modern technologies. Effective scheduling, transparent pricing, and consistent service delivery are ensured by the platform's ease of communication between clients and service providers.

The goals of the Home Service Provider Project to provide features like safe payment methods, real-time tracking, and user evaluations in order to enhance the overall consumer experience in the home services industry. Users will find it simpler and easier to access and take care of their domestic needs as a result. This management systems for different applications such as electrician, plumber, mechanic etc. but our main objective is to develop a web-based application for home owners with critical home repairs, accessibility modifications, and energy-efficient upgrades. This project helps to bring the communities together in efforts to assist those in need of general home repair and improvements. In this project, the Homeowners must submit a detail with a request. Every person needs helps from the electrician, plumber, mechanic etc. Thus, this web application will enable the user to seek and call them for help. This project will also let the user find them according to their locations, so that help can always come in time when needed.

**ACKNOWLEDGEMENT**

**ACKNOWLEDGEMENT**

This project work itself is an acknowledgement to the intensity drive and technical competence of many who have contributed to it. I express my thanks to everyone who contributed and guided me much for the successful completion of this simple effort.

I express my sincere thanks to **Dr. C.SUNDAR RAJ, M.E., Ph.D., Principal, A.V.C College of Engineering** who helped me in providing required facility in completing the project.

I express my gratitude thanks to **Dr. S. SELVAMUTHUKUMARAN, MCA., Ph.D., Vice-Principal, Professor cum HOD, Department of Computer Applications, A.V.C College of Engineering** who allowed me to do my final project work independently and effectively.

I am grateful to my internal guide Mr. S. SIVAKUMAR, MCA., M.Phil. Assistant Professor, Department of Computer Applications, A.V.C College of Engineering , for his valuable guidance, idea, advice and encouragement for the successful completion of this project to identify our mistakes and also achieving our goal.

I am thankful to my external guide **Ms. N.Rajiyabanu MCA., Manager, Fantasy Solution, Trichy** for her valuable guidance for successful completion of this project.

I would like to record our deepest gratitude to our parents for their constant encouragement and support which motivated us to complete our project on time.

My hearty thankful to my friends who were with me and gave me all the support I needed to complete my project successfully.

**DHARANI B**

TABLE OF CONTENT

**TABLE OF CONTENT**

| **CHAPTER NO** | **CONTENT** | **PAGE NO** |
| --- | --- | --- |
|  | **LIST OF TABLES** | **(i)** |
|  | **LIST OF FIGURES** | **(ii)** |
|  | **LIST OF ABBREVIATIONS** | **(iii)** |
| **1** | **INTRODUCTION** | **1** |
| **2** | **ORGANIZATION PROFILE** | **2** |
| **3** | **SYSTEM CONFIGURATION**  3.1 Hardware Configuration  3.2 Software Configuration | **4**  **4** |
| **4** | **SOFTWARE FEATURES**  4.1 PYTHON  4.2 MYSQL | **5**  **7** |
| **5** | **PROJECT DESCRIPTION**  5.1 Overview of the project  5.2 Problem Statement  5.3 Module Descriptions | **9**  **9**  **10** |
| **6** | **SYSTEM ANALYSIS**  6.1 Existing System  6.2 Proposed System | **13**  **13** |
| **7** | **SYSTEM DESIGN**  7.1 UML Diagram  7.2 Data Base Design | **14**  **20** |
| **8** | **SYSTEM TESTING**  8.1 Test Plan  8.2 Test Cases  8.3 Bug Report | **25**  **27**  **34** |
| **9** | **CONCLUSION** | **35** |
| **10** | **FUTURE ENHANCEMENT** | **36** |
|  | **APPENDICES**  1: Sample Source Code Appendix  2: Sample Screenshots Appendix  3: Sample Reports | **37**  **53**  **61** |
|  | **REFERENCES** | **62** |

**LIST OF TABLES (i)**

|  |  |  |
| --- | --- | --- |
| **TABLE NO.** | **TABLE NAME** | **PAGE NO.** |
| 7.2.1 | Admin | 20 |
| 7.2.2 | Regisration | 21 |
| 7.2.3 | Servicer | 21 |
| 7.2.4 | Service | 22 |
| 7.2.5 | Booking | 22 |
| 7.2.6 | Review | 23 |
| 7.2.7 | Feedback | 24 |
| 8.1 | Test plan | 25 |
| 8.2 | Test case | 27 |

**LIST OF FIGURES (ii)**

|  |  |  |
| --- | --- | --- |
| **FIG. NO.** | **FIGURE NAME** | **PAGE NO.** |
| 7.1.1 | Use Case Diagram | 14 |
| 7.1.2 | Class Diagram | 15 |
| 7.1.3 | Sequence Diagram | 16 |
| 7.1.4 | Activity Diagram | 17 |
| 7.1.5 | Component Diagram | 18 |
| 7.1.6 | Deployment Diagram | 19 |

|  |  |  |
| --- | --- | --- |
| **S.NO.** | **ABBREVIATION** | **MEANING** |
| 1 | SQL | Structured Query Language |
| 2 | HTML | Hyper Text Markup Language |
| 3 | CSS | Cascading Style Sheet |

**LIST OF ABBREVIATION (iii)**