A close up of black text

Description automatically generated

**Centre Name: ACE-THUDUC-1-FPT.**

**Address: 62 Street 36, Van Phuc Residential Area, Hiep Binh Phuoc Ward,Thu Duc City**

**Hospitality Plus**

|  |  |  |
| --- | --- | --- |
| **Supervisor:** | *Mr. Pham Cong Danh* | |
| **Batch.** | *T5.2308.M0* | |
| **Group** | *Group 5* | |
| **Serial No** | **Enrollment Number** | **Student Name** |
| *1.* | *Student1501052* | *Nguyen Quang Thinh* |
| *2.* | *Student1501843* | *Tran The Anh* |
| *3.* | *Student1501200* | *Pham Minh Tri* |
| *4.* | *Student1501281* | *Nguyen Nhat Nam* |

***Month****: 01* ***Year****: 2024*

A black and gold frame

Description automatically generated

This is to certify that

|  |  |
| --- | --- |
| **Mr.** | **NGUYEN QUANG THINH** |
| **Mr.** | **TRAN THE ANH** |
| **Mr.** | **PHAM MINH TRI** |
| **Mr.** | **NGUYEN NHAT NAM** |

Have successfully Designed & Developed

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **Hospitality Plus** | | |

Submitted by:

|  |
| --- |
| **Mr. PHAM CONG DANH** |

Date Of Issue:

|  |  |
| --- | --- |
|  |  |

Authorized Signature:

|  |  |
| --- | --- |
|  |  |

|  |
| --- |
| **Content** |

Content

Acknowledge

Synposis

Review1 1

Review2 10

Review3 49

|  |
| --- |
| **Acknowledge** |

As we understand that the eProject is a step-by- step learning environment that closely simulates the class-room and Lab based learning environment into actual implementation. It is a projectimplementation at your fingertips!! An electronic, live juncture on the machine that allows you to.

* Practice step by step i.e. laddered approach.
* Build a larger more robust application.
* Usage of certain utilities in applications designed by user.
* Single program to unified code leading to a complete application.
* Learn implementation of concepts in a phased manner.
* Enhance skills and add value.
* Work on real life projects.
* Give a real life scenario and help to create applications more complicated and useful.
* Mentoring through email support.

We would like to send a great thank to our professor and others student for the addorable supports during the time in project.

Beside serveral mistakes we had made in the project, we hope to have more oppotunities to widen our knownledge in web developing world.

We also thank you so much for your infomation and the “demo” project from India APTECH to help us get more reality experience in this project. Since we have gained more experience from this project, we will not stop learning any thing as much as we can to be more successful in the future.

Your Sincerely,

|  |
| --- |
| **Synposis** |

The Objective of this program we aim is to give a sample project to work on real life projects. These applications help us build a larger more robust application.

The objective is not to teach us Javafx/Netbean but to provide us with a real life scenario and help us create basic applications using the tools.

Hence, we can revise the chapters before we start with the project.

This project is meant for students like us who have completed the module of Javafx. These programs should be done in the Lab sessions with assistance of the faculty if required.

The website brings everything in the world to you such as historic monuments, Hospitality Plus information, natural knowledge or skill in technology...

It is very essential for us to have a clear understanding of the subject. We think we should go through the project and solve the assignments as per requirements given.

And get back eprojects@aptech.ac.in as the assigned schedule.

**Review 1**

**Problem Defination**

**Hospitality Plus** Management Software is a comprehensive digital solution designed to streamline and enhance the operations of hotels in the modern age. It offers a wide range of features and basic functions aimed at optimizing hotel management and improving guest experience. Below is a general overview of the key features and functionalities provided by Hospitality Plus Management Software:

* **Reservation Management:** The software allows hotel staff to efficiently manage room reservations, including booking, modification, cancellation, and check-in/check-out processes. It enables easy organization, classification, and retrieval of reservation details, enhancing efficiency for hotel staff and satisfaction for guests.
* **Guest Management:** Hospitality Plus Management Software facilitates guest registration, profile management, and authentication processes. It enables hotel staff to maintain accurate records of guest information, track booking histories, and manage guest preferences effectively.
* **Room Inventory Management:** The software automates room inventory processes such as room allocation, availability tracking, and maintenance scheduling. It helps reduce manual workload for hotel staff, minimize errors, and ensure efficient utilization of room resources.
* **Billing and Invoicing:** With advanced billing and invoicing capabilities, Hospitality Plus Management Software enables hotel staff to easily generate and manage guest bills, invoices, and payment records. It supports multiple payment methods, tax calculations, and billing adjustments, enhancing accuracy and transparency in financial transactions.
* **Housekeeping Management:** The software facilitates the management of housekeeping tasks such as room cleaning, maintenance, and inventory replenishment. It provides scheduling tools, task assignment features, and real-time updates, ensuring timely and effective housekeeping operations.
* **Analytics and Reporting:** Hospitality Plus Management Software offers analytical tools and reporting features to help hotel managers gain insights into occupancy rates, revenue performance, and guest satisfaction metrics. It enables data-driven decision-making and strategic planning for hotel improvement.
* **Integration and Interoperability:** The software supports integration with other hotel systems, external booking platforms, and third-party services. It ensures interoperability with property management systems, channel managers, and online travel agencies, facilitating seamless information exchange and resource sharing.
* **Accessibility and User Experience:** Hospitality Plus Management Software prioritizes user-friendly interfaces, accessibility features, and mobile responsiveness. It provides intuitive navigation, personalized guest experiences, and multilingual support, catering to diverse guest needs and preferences.

Overall, Hospitality Plus Management Software serves as a vital tool for modern hotels seeking to enhance efficiency, guest satisfaction, and revenue performance. By leveraging advanced technologies and comprehensive features, it empowers hotel staff to manage hotel resources effectively and provide guests with seamless hospitality experiences.

|  |
| --- |
| **Customer Requirements Specifications** |

1. **Customer’s Specific Requirements**

The **Hospitality Smart** hotel management software caters to diverse needs with its array of features:

* **Registration/Login:** Users easily create accounts and securely log in.
* **Membership Benefits:** Exclusive perks incentivize loyalty.
* **Reservation:** Seamlessly make, modify, or cancel bookings.
* **Online Check-In:** Speed up arrivals with efficient online check-in.
* **Payment Integration:** Securely process payments through various gateways.
* **Room Service/Amenities:** Guests request services and amenities conveniently.
* **Review/Feedback:** Guests provide feedback for continuous improvement.
* **Reporting/Analytics:** Gain insights for informed decision-making.

This software optimizes hotel operations, enriches guest experiences, and fosters business growth.

1. **Functional Requirements**

The Hospitality Smart hotel system management software encompasses four main user roles:

1. **Guest**
   1. **View Information:** Guests can browse hotel details including available rooms, amenities, and services.
   2. **Description:** Guests can explore room options, amenities, and hotel services.
   3. **Functional Requirement:** Guests can view room types, facilities, and services offered by the hotel.
   4. **Register/Login:** Guests can register for an account to access additional features or log in if they already have an account.
   5. **Description:** Guests can create an account to access more features or log in if they have an existing account.
   6. **Functional Requirement:** Guests must provide necessary information like name, email, and password to register. Logging in requires a valid email and password.
   7. **Contact and Feedback:** Guests can contact the hotel and provide feedback.
   8. **Description:** Guests can communicate with the hotel and provide feedback.
   9. **Functional Requirement:** Guests can contact the hotel via phone, email, or feedback form on the website.
   10. **Search for Rooms:** Guests can search for available rooms based on their preferences.
   11. **Description:** Guests can search for rooms based on criteria like dates, room type, and price.
   12. **Functional Requirement:** Guests can search for rooms by specifying check-in and check-out dates, room type, and other preferences.
   13. **View Room Details:** Guests can view detailed information about available rooms.
   14. **Description:** Guests can see room descriptions, amenities, and pricing.
   15. **Functional Requirement:** Guests can view room details such as room type, description, amenities, and pricing.
2. **Registered User**
   1. **Login:** Registered users can log in to access their account.
   2. **Description:** Users with accounts can log in to access personalized features.
   3. **Functional Requirement:** Users must provide valid credentials (email and password) to log in.
   4. **Profile Management:** Users can manage their profile and password.
   5. **Description:** Users can update their personal information and change passwords.
   6. **Functional Requirement:** Users can edit their profile information like name, contact details, and password.
   7. **Book Rooms:** Users can search for and book rooms.
   8. **Description:** Users can find available rooms, select dates, and complete booking transactions.
   9. **Functional Requirement:** Users must log in to book rooms. They can specify booking details such as check-in/out dates and room preferences.
   10. **Review and Feedback:** Users can leave reviews and feedback.
   11. **Description:** Users can share their experiences and provide feedback.
   12. **Functional Requirement:** Users can write reviews for rooms or hotel services after their stay.
   13. **View Booking History:** Users can view their booking history.
   14. **Description:** Users can see past and current bookings.
   15. **Functional Requirement:** Users can access a history of their room bookings and check reservation statuses.
3. **Moderator**
   1. **Manage Bookings:** Moderators can manage room bookings.
   2. **Description:** Moderators oversee the booking process, including handling reservations and cancellations.
   3. **Functional Requirement:** Moderators can view, modify, or cancel room reservations.
   4. **Manage Users:** Moderators can manage user accounts.
   5. **Description:** Moderators oversee user accounts, including banning or deleting accounts as necessary.
   6. **Functional Requirement:** Moderators can view user accounts, search for users by username or email, and take actions like banning or deleting accounts.
   7. **Manage Feedback:** Moderators can manage user feedback.
   8. **Description:** Moderators can review and respond to user feedback.
   9. **Functional Requirement:** Moderators can hide inappropriate feedback or reply to user comments.
4. **Administrator**
   1. **Manage Website Content:** Administrators can manage website content.
   2. **Description:** Administrators oversee website content, including advertisements, blogs, and other information.
   3. **Functional Requirement:** Administrators can add, modify, or disable website content like advertisements and blogs.
   4. **Manage Rooms:** Administrators can manage room listings.
   5. **Description:** Administrators oversee room listings, including adding, modifying, or deleting rooms.
   6. **Functional Requirement:** Administrators can add new rooms, update room details, or remove rooms from listings.
   7. **Manage Room Types:** Administrators can manage room categories.
   8. **Description:** Administrators oversee room categories, including adding, modifying, or deleting categories.
   9. **Functional Requirement:** Administrators can add new room types, update type names, or remove empty categories.

## **System Requirements:**

**Hardware / Software requirement**

1. **Hardware**:

**Web Server**

* Processor Intel Core I3 or higher.
* Memory 6 GB RAM or greater.
* Modem/ADSL Internet access is required.

**Client**

* Processor Intel Core I3 or higher.
* Memory 6 GB RAM or greater.
* Monitor Super VGA (1024x768) or higher resolution.
* Modem/ADSL Internet access is required.

1. **Software**:

**Server:**

* Operation System: Window 7 or later.
* Databases: MySQL.
* Browser: Google Chrome version 35.

**Client:**

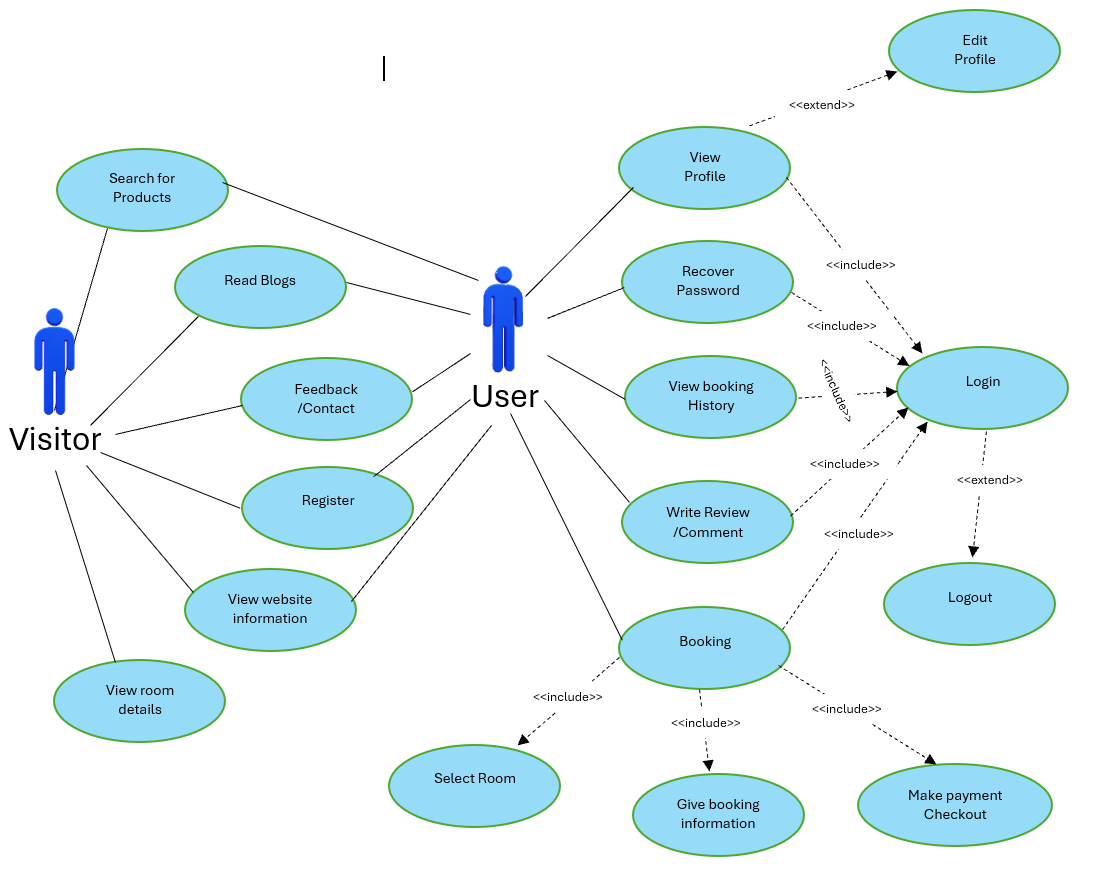
* Operation System Window 7 or higher.
* Browser Google Chrome , MS-Edge, Firefox.

REVIEW 1 – TASK SHEET

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Project Ref. | Project Title:  Smart Book Library | | Date of Preparation of Activity Plan:  27/5/2024 | | |
| No. | Task | Start Date | Actual Days | Member | Status |
| 01 | Acknowledgment &  Synopsis | 27/5/24 | 31/05/24 | * Tran The Anh | Completed |
| 02 | Problem Defination | * Pham Minh Tri | Completed |
| 03 | Customer Requirement Specification | * Nguyen Quang Thinh | Completed |
| 06 | Development Software | * Nguyen Nhat Nam | Completed |

REVIEW 2

## Use Case Diagram 1



**VISISTOR and USER**

1. **Vistor**
   1. **Search for Products**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Author** |  | | | |
| **Use Case Name** | Search roomtype | | | |
| **Actor** | Vistor/User | | |  |
| **Description** | Visitor want to search for available room types according to their schedule | | | |
| **Requirements** |  |  | | |
| **Status** | Success: Show all rooms that match input schedule.  Fail: No room are shown. | | | |
| **Basic flow** | Actor action:  1. Actor input ‘pick-up date’, ‘return date’ and ‘room type’ (optional) then click ‘Search’ button. | | System Response:  2. System redirects to Room list Page, show available rooms. | |
| **Alternative flow** |  | |  | |
| **Exception** |  | |  | |

* 1. **Read Blogs**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | Read Blogs | |
| **Actor** | Vistor/User |  |
| **Description** | Read blogs about room experiences, tips for booking,… | |
| **Requirements** |  |  |
| **Status** | Show blogs page | |
| **Basic flow** | Actor action:  1. Actor clicks “Blogs” on the Navigation Bar.  2. Actor clicks on the blog’s title to read. | System Response:  2. The system will redirect to the Blogs Page showing a list of blogs.  4. The system will redirect to the blog content page. |

* 1. **Feedback/Contact**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | Feedback/Contact | |
| **Actor** | Vistor/User |  |
| **Description** | Allow users to give feedback or contact the service for inquiries or issues. | |
| **Requirements** |  |  |
| **Status** | Show feedback/contact form | |
| **Basic flow** | Actor action:  1. Actor clicks "Feedback/Contact" on the Navigation Bar.  2. Actor fills in the necessary information (name, email, message).  3. Actor clicks the "Submit" button. | System Response:  2. The system will redirect to the Blogs Page showing a list of blogs.  3. The system will validate the input  4.The system will submit the feedback/contact information and show a confirmation message. |

* 1. **View website information**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | View website information | |
| **Actor** | Vistor/User |  |
| **Description** | View information about our company | |
| **Requirements** |  |  |
| **Status** | Show information page | |
| **Basic flow** | Actor action:  1. Actor click “About us” on the Navigation Bar | System Response:  2. The system will direct to the About Us Page. |
| **Alternative flow** |  |  |
| **Exception** |  |  |

* 1. **Register**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | Register an account | |
| **Actor** | Vistor/User |  |
| **Description** | Visitor want to register to become an user of website | |
| **Requirements** |  |  |
| **Status** | Success: The account will be createed  Fail: Refill information | |
| **Basic flow** | Actor action:  1. Actor click “Register” link.  3. Actor inputs register’s information and click ‘Register’ button. | System Response:  2. System redirects to Register page and displays Registration form with the following controls:  - ‘Full Name text field  - ‘Password’ text field  - ‘Confirm Password’ text field  - ‘Email’ text field  - ‘Phone’ text field  - ‘Address’ text field  - ‘Date of Birth' field  - ‘Photo’ field  - ‘Register’ button  - ‘Reset’ button  4. System validates the information.  5. System inserts the account into database  6. System |
| **Alternative flow** |  |  |
| **Exception** |  |  |

* 1. **View room details**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | View room type | |
| **Actor** | Vistor/User |  |
| **Description** | Actor want to view room details | |
| **Requirements** |  |  |
| **Status** | Show room detail information | |
| **Basic flow** | Actor action:  1. Actor click on ‘View Details’ button. | System Response:  2. System redirects to Room Details Page, showing Room detail information. |
| **Alternative flow** |  |  |
| **Exception** |  |  |

1. **User**
   1. **Login**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | Login | |
| **Actor** | User |  |
| **Description** | User who has register an account can login | |
| **Requirements** |  | |
| **Status** | Success: User is logged in to website  Fail: Refill information | |
| **Basic flow** | Actor action:  1. Actor clicks on ‘Login’ button.  3.Actor inputs Email and Password, then click the ‘Login’ button. | System Response:  2. System redirects to Login Page with the following controls:  - 'Email' text field  - 'Password' text field  - 'Login' button  4. System check the email and password.  5. System redirects to Homepage. |
| **Alternative flow** | 1. Actor click “Forget Password” button.  3. Actor provide email and click “Send mail”.  5. Actor use new password to login. | 2. System redirect to Forget password page.  4. System check user’s email in database. If correct, send new password to user’s email. Otherwise, return message “Email not found”. |
| **Exception** | 1. Actor inputs invalid email and password. | System redirects to Login page with message: “Invalid email or password’ |

* 1. **Edit profile**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | Edit profile | |
| **Actor** | User | User |
| **Description** | User who has logged in can edit profile | |
| **Requirements** |  | |
| **Status** | Success: Actor can see and change their info  Fail: Refill information | |
| **Basic flow** | Actor action:  1. Actor click on “My Account” button.  3. Actor click on “Edit profile’ button.  5. Actor edit their profile, then click Save | Actor action:  1. Actor click on “My Account” button.  3. Actor click on “Edit profile’ button.  5. Actor edit their profile, then click Save |

* 1. **Logout**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | Logout | |
| **Actor** | User |  |
| **Description** | User who has login can logout | |
| **Requirements** |  | |
| **Status** | User can log out website | |
| **Basic flow** | Actor action:  1. Actor clicks on ‘Log out’ button | System Response:  2. System delete user session data and redirect to Homepage. |
| **Alternative flow** |  |  |
| **Exception** |  |  |

* 1. **View History Booking**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | View rental history | |
| **Actor** | User |  |
| **Description** | User want to view rental history | |
| **Requirements** | User who has logged in | |
| **Status** |  | |
| **Basic flow** | Actor action:  1. User click on “Rental History” button | System Response:  2. System redirect to Rental history page, show rental history information. |
| **Alternative flow** |  |  |
| **Exception** |  |  |

* 1. **Recover password**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | Edit profile | |
| **Actor** | User |  |
| **Description** | User who has logged in can recover password | |
| **Requirements** |  | |
| **Status** | Success: Actor can see and change their info  Fail: Refill information | |
| **Basic flow** | Actor action:  1. Actor click on “My Account” button.  3. Actor click on “Change Password” button.  5. Actor give current password and new password. | Actor action:  1. Actor click on “My Account” button.  3. Actor click on “Change Password” button.  5. Actor give current password and new password. |
| **Alternative flow** |  |  |
| **Exception** | Actor action:  1. Input current password don’t match with the password of account.  2. ‘New password’ field left blank or incorrect format | Actor action:  1. Input current password don’t match with the password of account.  2. ‘New password’ field left blank or incorrect format |

* 1. **Book a room**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | Book a Room | |
| **Actor** | User |  |
| **Description** | User wants to book a room from the website | |
| **Requirements** | User who has logged in | |
| **Status** |  | |
| **Basic flow** | Actor action:  1. User clicks on "Book Room" button  3. User can select room type, check-in and check-out dates, and enter additional information (e.g., special requests)  5. User pays for the booking | System Response:  2. System redirects to the Book Room page with room information (availability, price, etc.)  4. System saves booking data to session, redirects to Checkout page, shows payment method, generates QR code for payment  6. System saves booking data to database, shows success message, and redirects to Home page |
| **Alternative flow** | Actor action:   |  |  | | --- | --- | | 1. User clicks on "View Rooms" to browse available rooms |  |  |  | | --- | | 3. User selects a room and clicks "Book Now" | | System Response:  2. System displays a list of available rooms with detailed information  4. System redirects to the Book Room page and follows the basic flow from step 3 |
| **Exception** | Actor action:  1. User attempts to book a room without logging in | System Response:  2. System prompts the user to log in or register before proceeding to the Book Room page |

**II. ADMIN and MODERATOR  
1.admin**

**1.1 admin login**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Login | |
| **Actors** | Admin | |
| **Description** | Admin can login to admin dashboard | |
| **Requirements** | Admin provide email and password | |
| **Status** | Success: show login message successfull  Fail: show message failed | |
| **Baisc flow** | Actor action:  1. Actor type email and password, then click “Login” button. | System response:  2. System validate information. If correct, redirect to system dashboard. Otherwise, show failed login message. |
| **Alternative flow** |  |  |
| **Exception** | - Actor type invalid email and password. | System show message:  - “Wrong email or password” |

**1.2 Admin logout**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Log out | |
| **Actors** | Admin | |
| **Description** | Admin can login to admin dashboard | |
| **Requirements** |  | |
| **Status** |  | |
| **Baisc flow** | Actor action:  1. Actor click “Logout” link. | System response:  2. System delete admin session data |
| **Alternative flow** |  |  |
| **Exception** |  |  |

**1.3 Admin manage all functions of the moderator**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Manage all functions | |
| **Actors** | Admin | |
| **Description** | Admin can reply comment and feedback | |
| **Requirements** | Admin is logged in | |
| **Status** |  | |
| **Baisc flow** | Actor action:  1. Add, edit, delete information for moderators and users, vehicle contracts, manage the process of vehicle booking contracts, and comments and feedback.  3. Update or create new and delete information as requested by the admin. | Actor action:  1. Add, edit, delete information for moderators and users, vehicle contracts, manage the process of vehicle booking contracts, and comments and feedback.  3. Update or create new and delete information as requested by the admin. |
| **Alternative flow** |  |  |
| **Exception** |  |  |

**1.4 Admin create category**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Manage all category | |
| **Actors** | Admin | |
| **Description** | Admin can create category | |
| **Requirements** | Admin is logged in | |
| **Status** |  | |
| **Baisc flow** | Actor action:  1. Click on the add category button.  3. Add information for the category , click on the add button. | System response:  2. Navigate to the page for adding category information.  4. System show add car form with following control:  - ‘Name’ text field  - ‘Description’ text field  - ‘Add’ button  - ‘Reset’ button  6. System validates the information.  7. System update account database and show update success message . |
| **Alternative flow** |  |  |
| **Exception** | Actor action:  1. ‘Name’ field left blank or incorrect format.  2. ‘Description’ field left blank or incorrect format. | System displays message to actor:  - ‘Name’ cannot be blank or incorrect format.  - ‘Description’ cannot be blank or incorrect format. |

**1.5 Admin delete category**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Manage all category | |
| **Actors** | Admin | |
| **Description** | Admin can delete category | |
| **Requirements** | Admin is logged in | |
| **Status** |  | |
| **Baisc flow** | Actor action create :  1. Actor click “delete” link. | Actor action create :  1. Actor click “delete” link. |
| **Alternative flow** |  |  |
| **Exception** |  |  |

**1.6 Admin can edit category**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Manage all category | |
| **Actors** | Admin | |
| **Description** | Admin can edit category | |
| **Requirements** | Admin is logged in | |
| **Status** |  | |
| **Baisc flow** | Actor action:  1. Click on the edit category button.  3. Add information for the category , click on the edit button. | System response:  2. Navigate to the page for editing category information.  4. System show edit car form with following control:  - ‘Name’ text field  - ‘Description’ text field  - ‘Edit button  - ‘Reset’ button  6. System validates the information.  7. System update account database and show update success message . |
| **Alternative flow** |  |  |
| **Exception** | Actor action:  1. ‘Name’ field left blank or incorrect format.  2. ‘Description’ field left blank or incorrect format. | System displays message to actor:  - ‘Name’ cannot be blank or incorrect format.  - ‘Description’ cannot be blank or incorrect format. |

**1.7 Admin add products**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Manage all category | |
| **Actors** | Admin | |
| **Description** | Admin can delete category | |
| **Requirements** | Admin is logged in | |
| **Status** |  | |
| **Baisc flow** | Actor action create :  1. Actor click “delete” link. | Actor action create :  1. Actor click “delete” link. |
| **Alternative flow** |  |  |
| **Exception** |  |  |

**1.8 Admin edit products creation**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Manage all products | |
| **Actors** | Admin | |
| **Description** | Admin can edit products | |
| **Requirements** | Admin is logged in | |
| **Status** |  | |
| **Baisc flow** | Actor action:  1. Click on the edit product button.  3. Edit information for the car , click on the save button. | System response:  2. Navigate to the page for page for editing product information.  4. System show edit car form with following control:  - ‘Name car’ text field  - ‘Brand’ text field  - ‘Color’ field  - ‘Type’ field  - ‘Year’ field  - ‘Capacity’ text field  - ‘Price’ text field  - ‘Overview’ text field  - ‘Images’ field  - ‘Thumbnail’ field  - ‘Save’ button  - ‘Reset’ button  6. System validates the information.  7. System update account database and show update success message . |
| **Alternative flow** |  |  |
| **Exception** | Actor action:  1. ‘Name car’ field left blank or incorrect format.  2. ‘Brand’ field left blank or incorrect format.  3. ‘Color’ field left blank or incorrect format.  4. ‘Capacity’ field left blank or incorrect format.  5. ‘Price’ field left blank or incorrect format.  6. ‘Overview’ field left blank or incorrect format.  7. ‘Images’ incorrect format.  8. ‘Thumbnail’ incorrect format. | System displays message to actor:  - ‘Name car’ cannot be blank or incorrect format.  - ‘Brand’ cannot be blank or incorrect format.  - ‘Color’ must choose.  - ‘Capacity’ cannot be blank or incorrect format.  - ‘Price’ cannot be blank or incorrect format.  - ‘Overview’ cannot be blank or incorrect format.  - ‘Images’ must not be empty and must be in the correct format.  - ‘Thumbnail’ must not be empty and must be in the correct format. |

**1.9 Admin delete products**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Manage all products | |
| **Actors** | Admin | |
| **Description** | Admin can delete products | |
| **Requirements** | Admin is logged in | |
| **Status** |  | |
| **Baisc flow** | Actor action create :  1. Actor click “delete” link. | Actor action create :  1. Actor click “delete” link. |
| **Alternative flow** |  |  |
| **Exception** |  |  |

**1.10 Admin delete banners and blogs**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Manage banner and blogs | |
| **Actors** | Admin | |
| **Description** | Admin can delete banners | |
| **Requirements** | Admin is logged in | |
| **Status** |  | |
| **Baisc flow** | Actor action create :  1. Actor click “delete” link. | System response:  2. System delete order in order database. |
| **Alternative flow** |  |  |
| **Exception** |  |  |

|  |  |  |
| --- | --- | --- |
| **Design Plan**:  EASYCAR | **Document Name**: REVIEW II | **SWD/Form** No.01/RV2/Ver1.0 |
| **Effective Date**:25/12/2023 | **Version** 1.0 | **Page No**: **29** of **1999** |

**1.11 Admin create banners and blogs**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Manage banner and blogs | |
| **Actors** | Admin | |
| **Description** | Admin can create banner and blogs | |
| **Requirements** | Admin is logged in | |
| **Status** |  | |
| **Baisc flow** | Actor action:  1. Actor click on the add Banner/Blogs button.  3. Add information for Banner/Blogs, click on the add button. | System response:  2. Navigate to the page for adding information of Banner/Blogs.  4. Save the added data to the database, and return to the management page |
| **Alternative flow** |  |  |
| **Exception** | - Actor enters incorrect information into the input fields. | System show message:  - error requesting re-entry of user account information |

**1.12 Admin edit banners and blogs**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Manage banner and blogs | |
| **Actors** | Admin | |
| **Description** | Admin can create banner and blogs | |
| **Requirements** | Admin is logged in | |
| **Status** |  | |
| **Baisc flow** | Actor action:  1. Actor click on the add Banner/Blogs button.  3. Add information for Banner/Blogs, click on the add button. | System response:  2. Navigate to the page for adding information of Banner/Blogs.  4. Save the added data to the database, and return to the management page |
| **Alternative flow** |  |  |
| **Exception** | - Actor enters incorrect information into the input fields. | System show message:  - error requesting re-entry of user account information |

**2.Moderator**

**2.1 Moderator login**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Log in | |
| **Actors** | Moderator | |
| **Description** | Moderator can login to admin dashboard | |
| **Requirements** | Moderator provide email and password | |
| **Status** | Success: show login message successful.  Fail: show message failed. | |
| **Baisc flow** | Actor action:  1. Actor type email and password, then click “Login” button. | System response:  2. System validate information. If correct, redirect to system dashboard. Otherwise, show failed login message. |
| **Alternative flow** |  |  |
| **Exception** | - Actor type invalid email and password. | System show message:  - “Wrong email or password” |

**2.2 Moderator logout**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Log out | |
| **Actors** | Moderator | |
| **Description** | Moderator can log out of the admin dashboard | |
| **Requirements** |  | |
| **Status** |  | |
| **Baisc flow** | Actor action:  1. Actor click “Logout” link | System response:  2. System delete admin session data. |
| **Alternative flow** |  |  |
| **Exception** |  |  |

**2.3 Moderator create car rental contract management**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Car rental contract management | |
| **Actors** | Moderator | |
| **Description** | Moderator can create car rental contracts | |
| **Requirements** | Moderator is logged in | |
| **Status** |  | |
| **Baisc flow** | Actor action create :  1. Actor click on the button to “create” a car rental contract.  3. Add contract information to button “create”. | System response:  2. Navigate to the car rental contracts page.  4. Save the added data to the database and navigate to the page management. |
| **Alternative flow** |  |  |
| **Exception** | - Actor enters incorrect information into the input fields. | System show message:  - error requesting re-entry of order information |

**2.4 Moderator edit car rental contract management**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Car rental contract management | |
| **Actors** | Moderator | |
| **Description** | Moderator can edit car rental contracts | |
| **Requirements** | Moderator is logged in | |
| **Status** |  | |
| **Basic flow** | Actor action create :  1. Actor click on the button to “edit” a car rental contract.  3. Edit the information on the contract, click on the “Save” button . | System response:  2. Navigate to the car rental contracts page.  4. Save the added data to the database and navigate to the page management. |
| **Alternative flow** |  |  |
| **Exception** | - Actor enters incorrect information into the input fields. | System show message:  - error requesting re-entry of order information |

**2.5 Moderator delete car rental contract management**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Car rental contract management | |
| **Actors** | Moderator | |
| **Description** | Moderator can delete car rental contracts | |
| **Requirements** | Moderator is logged in | |
| **Status** |  | |
| **Basic flow** | Actor action create :  1. Actor click “delete” link. | System response:  2. System delete order in order database. |
| **Alternative flow** |  |  |
| **Exception** |  |  |

**2.6 Moderator contract process management**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Confirm the contract with the customer | |
| **Actors** | Moderator | |
| **Description** | Moderator can confirm | |
| **Requirements** | Moderator is logged in | |
| **Status** |  | |
| **Basic flow** | Actor action create :  1. After confirming with the customer (calling), click on the ‘Details’ button  3. Click the ‘Confirmed’ button. | System response:  2. Navigate to the contract details page.  4. Change the status of the car rental contract to Confirmed.  5. Save the added data to the database and navigate to the page management. |
| **Alternative flow** |  |  |
| **Exception** |  |  |

**2.7 Moderator confirm handing over the vehicle**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Confirm handing over the vehicle | |
| **Actors** | Moderator | |
| **Description** | Moderator can confirm | |
| **Requirements** | Moderator is logged in | |
| **Status** |  | |
| **Basic flow** | Actor action create :  1. At the details page, click on the Delivered button. | System response:  2. Change the status of the car rental contract and the vehicle status to Delivered. |
| **Alternative flow** |  |  |
| **Exception** |  |  |

**2.8 Moderator confirm that the customer has returned the vehicle**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Confirm that the customer has returned the vehicle | |
| **Actors** | Moderator | |
| **Description** | Moderator can confirm | |
| **Requirements** | Moderator is logged in | |
| **Status** |  | |
| **Basic flow** | Actor action create :  1. At the details page, click on the Received the vehicle button. | System response:  2. Change the status of the car rental contract and the vehicle status to Received the vehicle back. |
| **Alternative flow** |  |  |
| **Exception** |  |  |

**2.9 Moderator create manage user accounts**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Manage user accounts | |
| **Actors** | Moderator | |
| **Description** | Moderator can create user account | |
| **Requirements** | Moderator is logged in | |
| **Status** |  | |
| **Basic flow** | Actor action create :  1. Actor click on the add customer button on the system management page.  3. Add contract information to button “create”. | System response:  2. Navigate to the customer's additional information page.  4. Save the added data to the database and navigate to the page management. |
| **Alternative flow** |  |  |
| **Exception** | - Actor enters incorrect information into the input fields. | System show message:  - error requesting re-entry of user account information |

**2.10. Moderator edit manage user accounts**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Manage user accounts | |
| **Actors** | Moderator | |
| **Description** | Moderator can edit user account | |
| **Requirements** | Moderator is logged in | |
| **Status** |  | |
| **Basic flow** | Actor action create :  1. Actor click on the edit button for customer information.  3. Edit the necessary information of the customer, and press the “save” button. | System response:  2. Navigate to the customer information editing page.  4. Save the added data to the database and navigate to the page management. |
| **Alternative flow** |  |  |
| **Exception** | - Actor enters incorrect information into the input fields. | System show message:  - error requesting re-entry of user account information |

**2.11. Moderator delete manage user accounts**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Manage user accounts | |
| **Actors** | Moderator | |
| **Description** | Moderator can delete user account | |
| **Requirements** | Moderator is logged in | |
| **Status** |  | |
| **Basic flow** | Actor action create :  1. Actor click “delete” link. | System response:  2. System delete user account in account database. |
| **Alternative flow** |  |  |
| **Exception** |  |  |

**2.12. Moderator manage customer feedback and comments**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use case name** | Manage comment, feedback | |
| **Actors** | Moderator | |
| **Description** | Moderator can reply comment and feedback | |
| **Requirements** | Moderator is logged in | |
| **Status** |  | |
| **Basic flow** | Actor action create :  1. Actor click the reply button for comments and feedback.  3. Click to select hide or show status. | System response:  2. Navigate to the comments, feedback page.  4. Save the added data to the database and navigate to the page management. |
| **Alternative flow** |  |  |
| **Exception** |  |  |

## **DATA FLOW DIAGRAM (DFD)**

A diagram of a user

Description automatically generated

## **ENTITY RELATIONSHIP DIAGRAM (ERD)**

A screenshot of a computer

Description automatically generated

## **Table Design:**

1. **Table accounts:**

A screenshot of a computer

Description automatically generated

1. **Table role\_account:**

A screenshot of a computer

Description automatically generated

1. **Table history\_login:**

A screenshot of a computer

Description automatically generated

1. **Table room\_type:**

A screenshot of a computer

Description automatically generated

1. **Table rooms:**

A screenshot of a computer

Description automatically generated

1. **Table\_Booking:**

A screenshot of a computer

Description automatically generated

1. **Table bill\_detail:**

A screenshot of a computer

Description automatically generated

1. **Table ratings:**

A screenshot of a computer

Description automatically generated

1. **Table comments\_car:**

A screenshot of a computer

Description automatically generated

1. **Table feedbacks:**

A screenshot of a computer

Description automatically generated

1. **Table ads\_Banners:**

A screenshot of a computer

Description automatically generated

REVIEW 2 – TASKSHEET

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Project Ref. | Project Title:  Game Store | Date of Preparation of Activity Plan:  27/5/2024 | | | |
| No. | Task | Start Date | Actual Days | Member | Status |
| 01 | Use Case Visitor | 31/5/24 | 7/6/24 | Tran The Anh | Completed |
| 02 | Use Case User | Pham Minh Tri | Completed |
| 03 | Use Case Moderation | Nguyen Quang Thinh | Completed |
| 04 | Use Case Admin | Nguyen Nhat Nam | Completed |
| 05 | Date Flow Diagram | Tran The Anh | Completed |
| 06 | ERD | Pham Minh Tri | Completed |
| 07 | Table Design | Nguyen Quang Thinh | Completed |

REVIEW 3

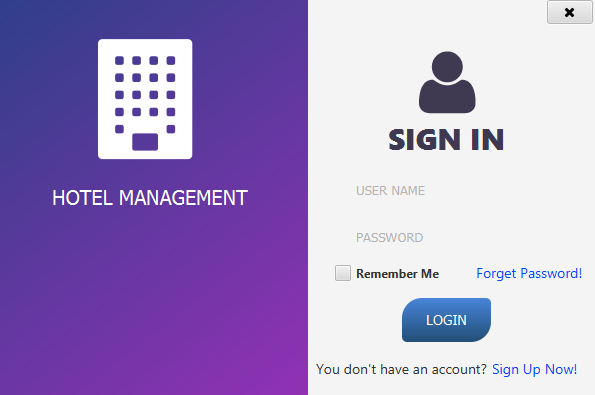
I.**Sitemap:**

A diagram of a computer system

Description automatically generated

**II.Screenshot:**

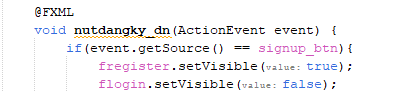
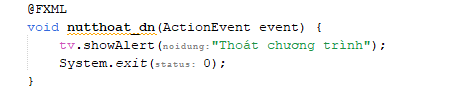
1. **Login:**



Users will fill in information log in: Text field Username

Validate: “Name is required!”, “User does not exist!”  
 Text field Password

Validate: “Password is required”, “Wrong password!” Icon   
Checkbox Remember: Remember password for the next sign in  
 Link “Forgot password": Click to go Reset password page   
Login Button: Complete fill in and start validation to sign in   
Sign Up Button: Click to go Register Account page





1. **Register:**

**A screenshot of a login form

Description automatically generated**

User fill full information:

Text fill Integer Customer Code  
 Validate: “Customer code is required”  
Text field String First Name

Validate: “First Name is required”

Text field String Last Name

Validate: “Last Name is required”

Text field String Email (Gmail is registered with the company)

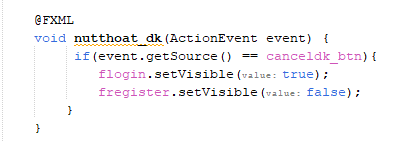
Validate: “Email is required”, validate with letter “@”

Text field String Address

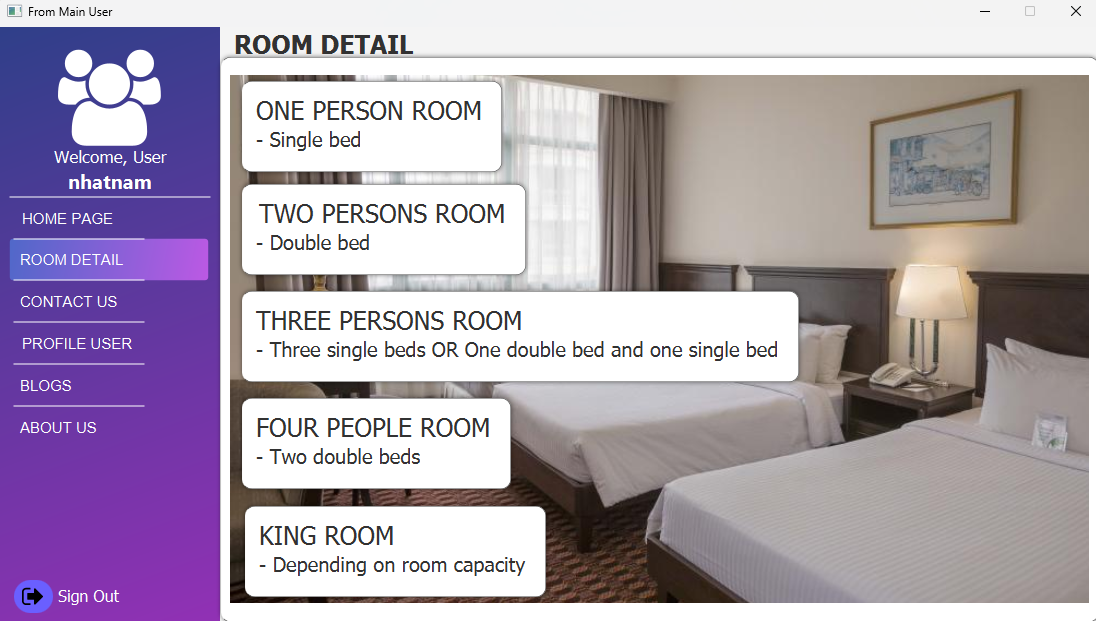
Text field Integer Phone

Validate: “Phone is required”, “Format error, enter number” Text field String Password  
 Validate: “Password is required”, “Wrong password!”  
 Role: Admin-Staff (Default Admin).  
 Then click the “Sign Up" button to complete the login process. If you have an account, you click the “Login Account" button to comeback sign up page.

A computer code with black and pink text

Description automatically generated

1. **Room Details:**



The User page displays:

-A tab bar in left side:

+Profile User with the title “Welcome \*Name of user\*”

+ Button: Available Room to let user see current room that is availble

+ Log Out to sign out the application  
  
 - Tab in middle:

+ Above: Let Customer search for room name and room base on the prices.

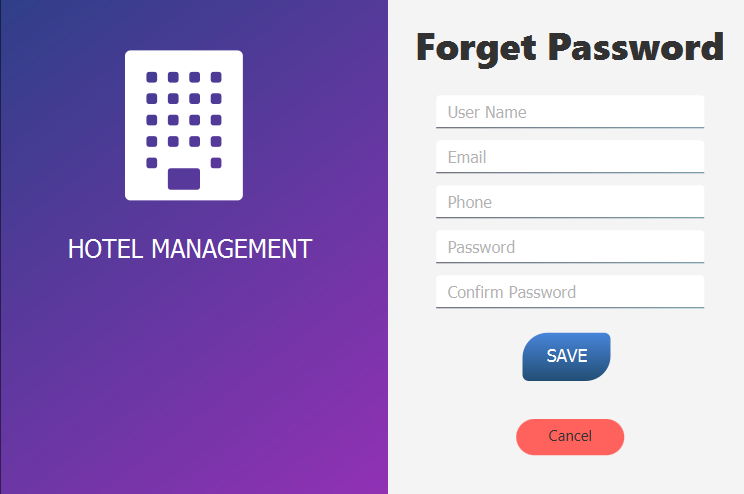
+Below: Button for check in which will bring them to contact base to ask about room information via our phone numbers.

- Table on right side:

+ Above: Search bar to search for room name

+Below: displays the result ( All room will be shown on default)

1. **Forget Password:**

****

The page forget password:

- Text fill Full Name

Validate: “Full name code is required”

- Text fill Email

Validate: “Email is required”, validate with letter “@”

- Text fill Phone

Validate: “Phone is required”, “Format error, enter number”

- Text fill Password

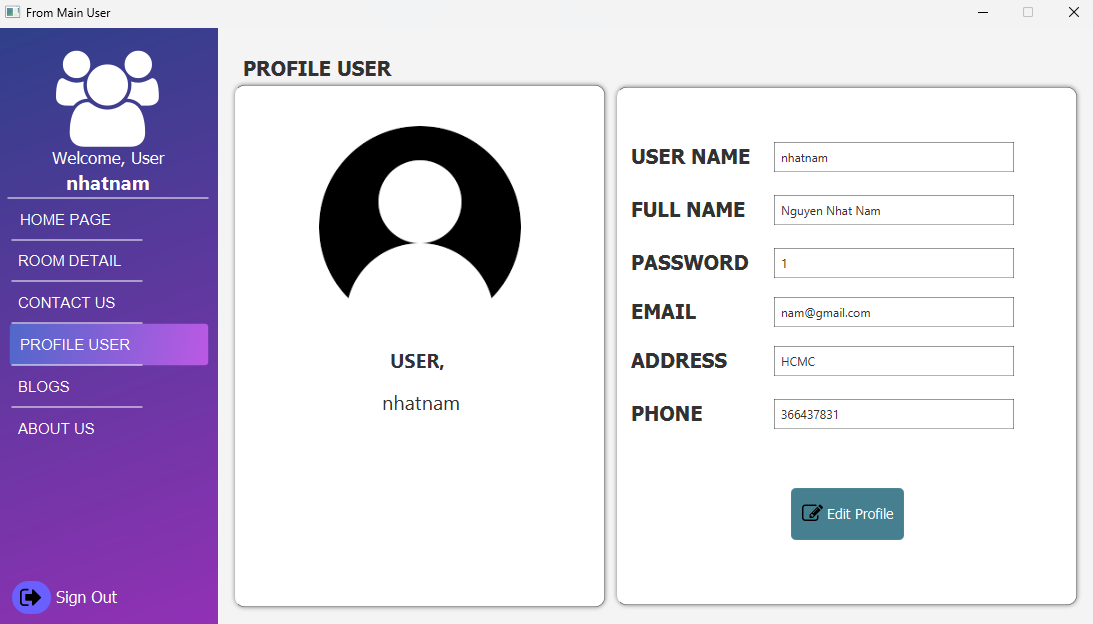
Validate: “Password is required”, “Wrong password!”

- Text fill Confirm Password

Validate: “Confirm Password is required”,“Please enter the correct password to confirm”

Then click the “Save” button to complete the change process.

1. **Profile User**

****

The page Profile User:

- Text fill Id User

Validate: “Show your Id User”

- Text fill Full Name

Validate: “Show your Full Name”

- Text fill Password

Validate: “Show your Password”

- Text fill Email

Validate: “Show your Email”

- Text fill Address

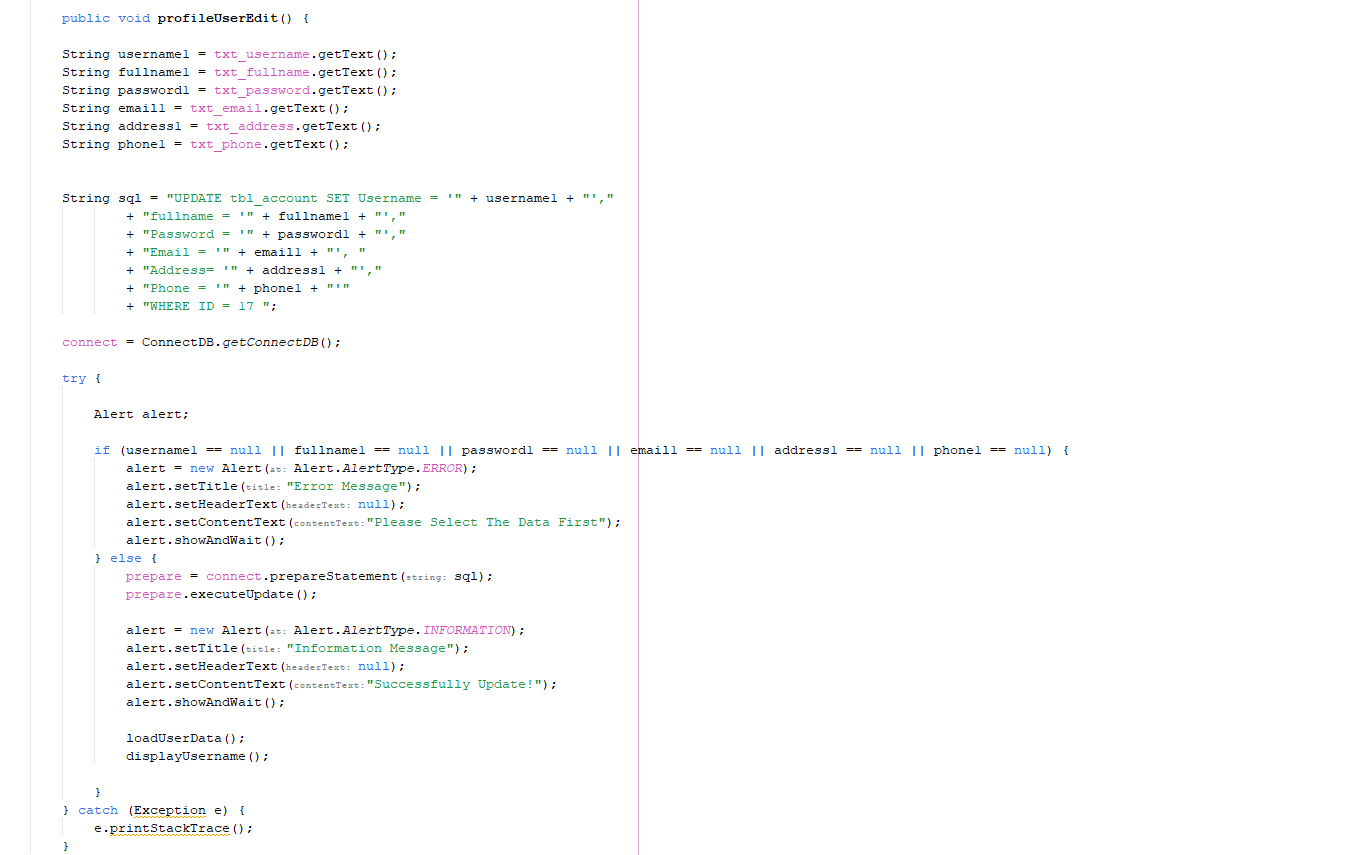
Validate: “Show your Address”

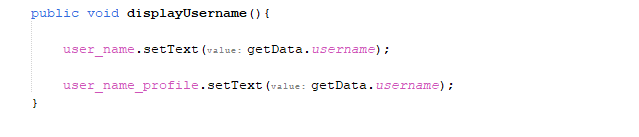
- Text fill Phone

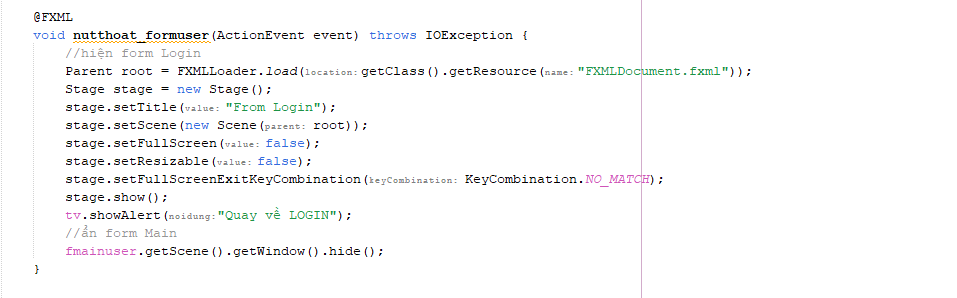
Validate: “Show your number phone”

- Text fill Role

Validate: “Show your Role”

**** Click "Edit Profile" when you want to edit your profile.

****

****

****

**A screenshot of a computer code

Description automatically generated**

1. **Booking History:**

**A screenshot of a website

Description automatically generated**

Booking history page:

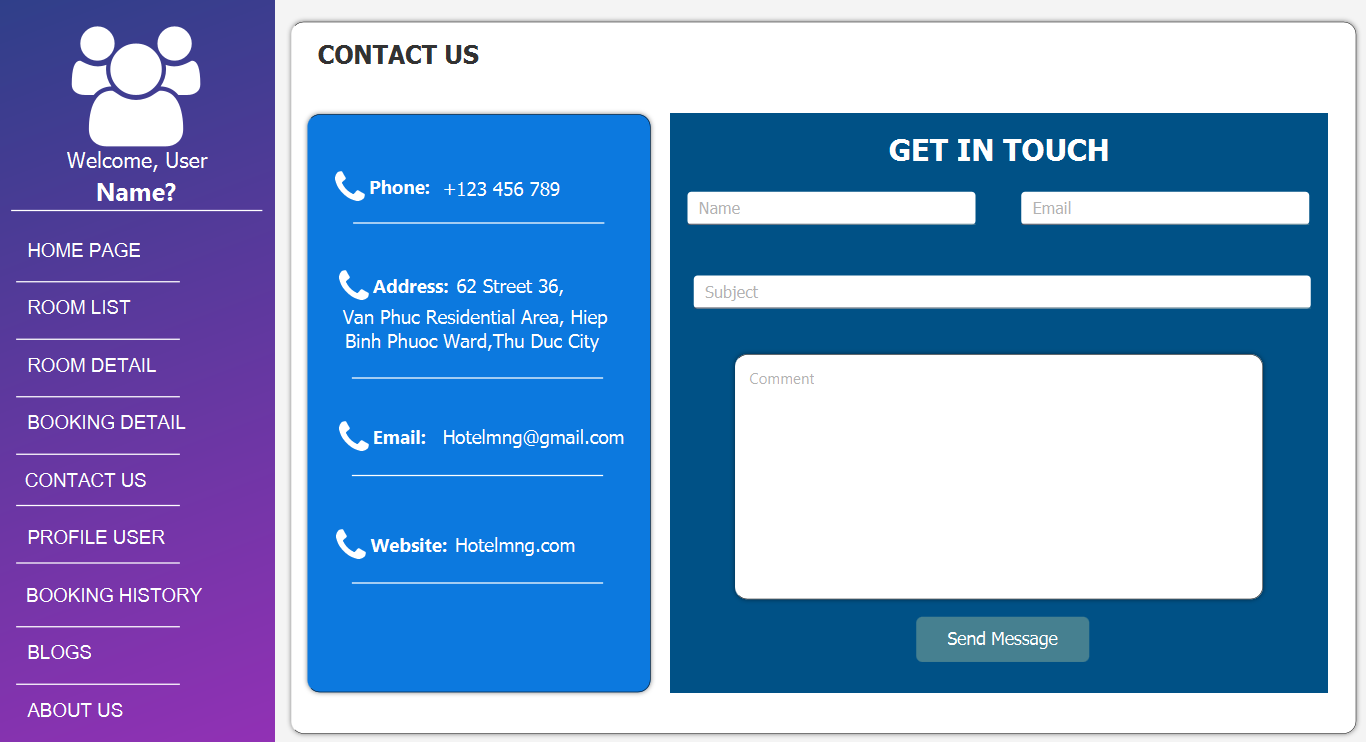
- A tab bar at the top:

+ Search your booking history

- Tabs at the bottom:

+You can review your entire booking history

1. **Contact Us:**



Contact Us:

* Name Field: Allows users to input their name when they are contacting the company.
* Email Field: Allows users to input their email address for contact purposes.
* Subjact Field: Allows users to input the subject of their message or inquiry.
* Comment Field : Allows users to input their message or inquiry in detail.
* **Send Message Button:** Submits the contact form, sending the user's input (Name, Email, Subject, Comment) to the company for processing.

1. **About Us :**

**A screenshot of a hotel management website

Description automatically generated**

Provides an overview of the hotel’s mission, services, and values. It highlights the hotel’s commitment to offering high-class accommodation services and modern amenities in a convenient and scenic location.

1. **Blogs:**

**A screenshot of a hotel management system

Description automatically generated**

The "Blogs" section on provides valuable insights and information on hotel management through well-structured and visually appealing blog post previews. Each post is designed to educate readers on different aspects of hotel management, emphasizing the importance of tools, technology, and effective management practices in the hospitality industry. The sidebar navigation enhances user experience by providing quick access to other sections of the application.

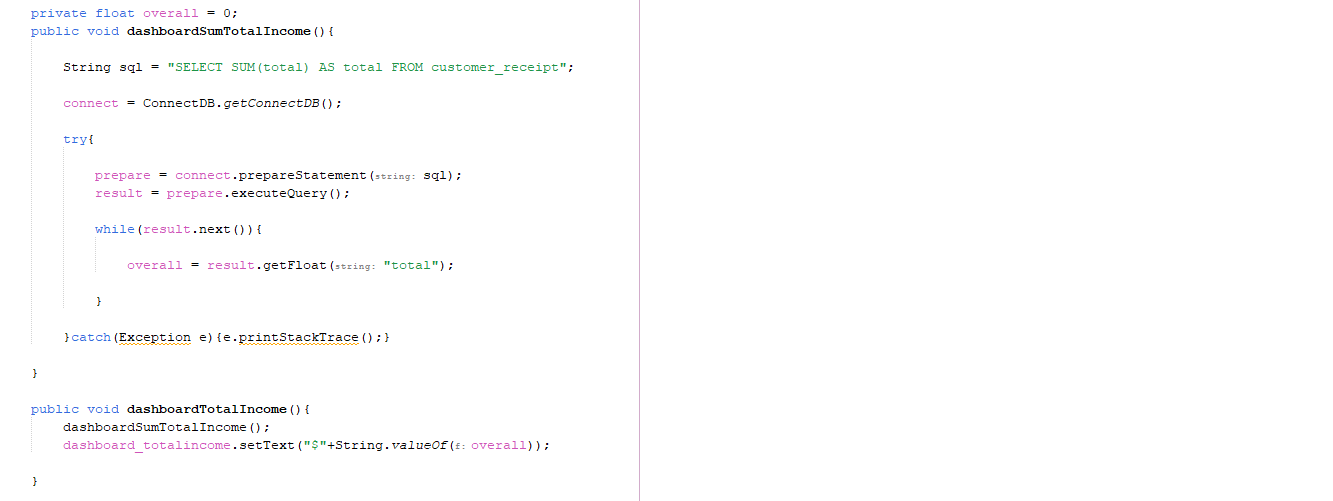
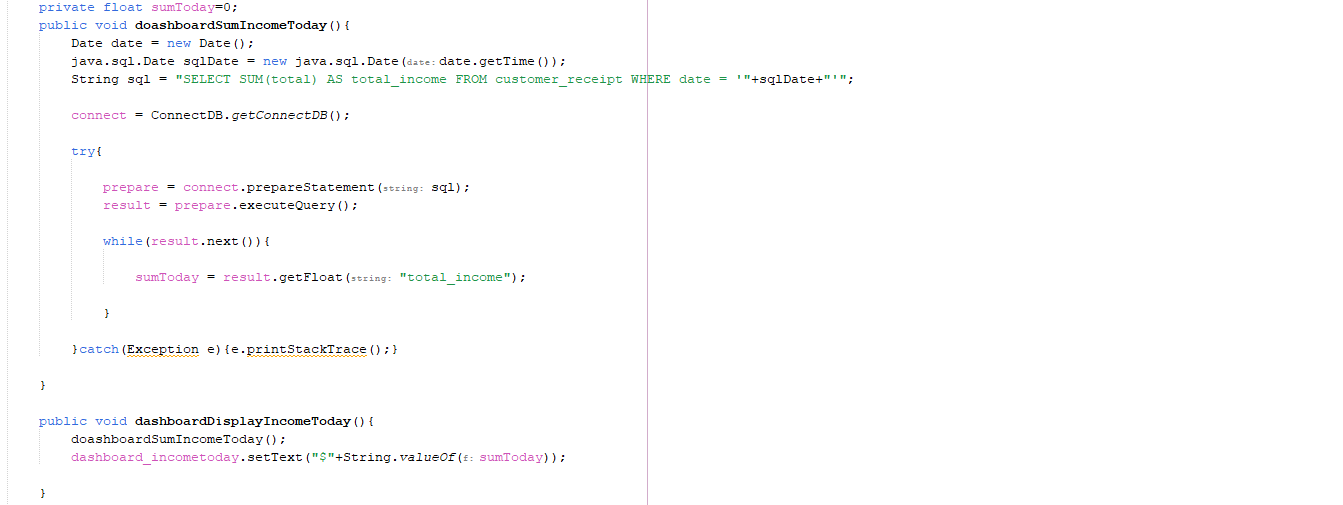
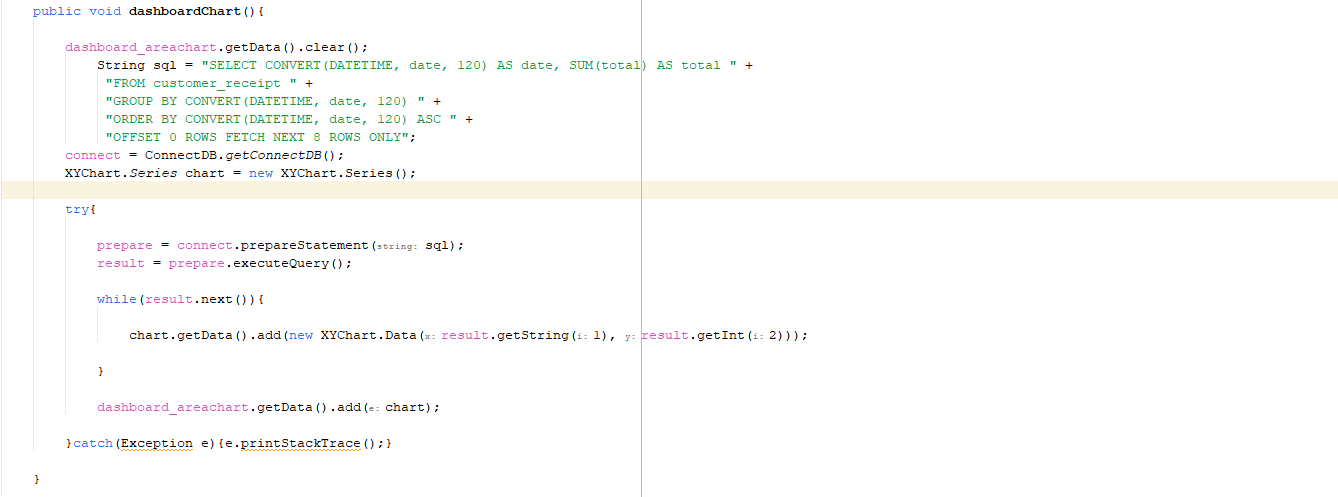
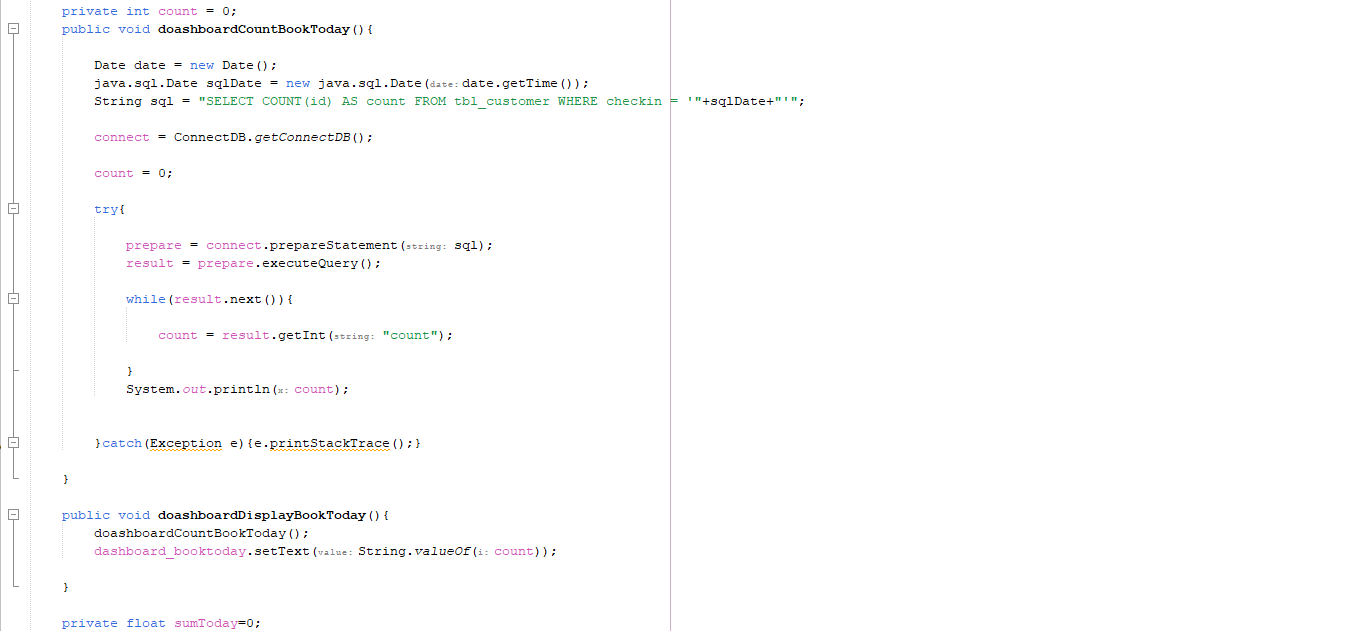
**Admin Dashboard:**

1. **Main Dashboard Page:**

**A screenshot of a computer

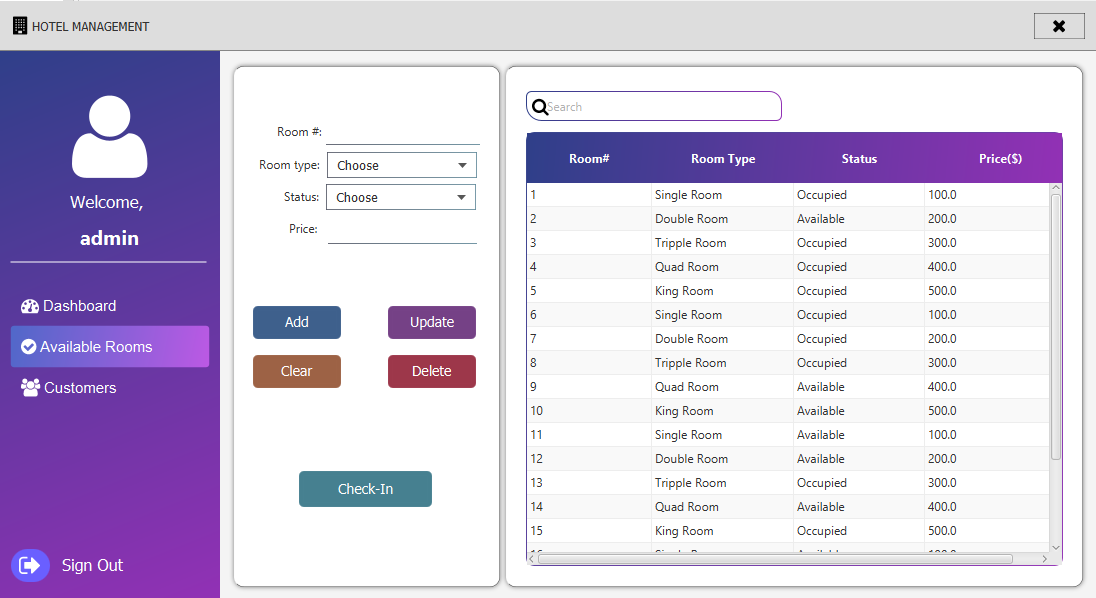
Description automatically generated**

* **Today's Book**: Shows the number of bookings made today. It provides a quick overview of the current day's activity.
* **Today's Income**: Displays the total income generated from today's bookings. This is a financial summary for the day.
* **Total Income**: Shows the total income generated over a specified period or since the system started tracking income. This provides a comprehensive financial overview.
* **Data Income Chart**: A graphical representation of income data over time. This chart helps visualize trends in income, showing changes over days, weeks, or months.
* **Dashboard**: Access the main dashboard for a summary of key metrics.
* **Available Rooms**: View and manage the list of rooms available for booking.
* **Customers**: Access customer information and details.
* **Sign Out**: Log out of the system securely.

****

****

1. **Available Room:**

****

**The "Available Rooms" page in a hotel management system serves as a comprehensive interface for managing the hotel’s room inventory. This page is designed for administrators to easily view, add, update, delete, and check the status of rooms.**

**On the left side of the screen, there is a navigation panel that includes links to the Dashboard, Available Rooms, and Customers sections. The active section is highlighted for easy identification.**

**The central section of the interface includes input fields for managing room details. Users can specify the room number, type (e.g., Single Room, Double Room, Triple Room, etc.), status (Occupied or Available), and price. There are action buttons for adding new rooms, updating existing room information, clearing the input fields, and deleting rooms. The "Check-In" button facilitates the process of checking in guests into available rooms.**

**The right side of the screen displays a searchable and scrollable table listing all rooms in the hotel. The table includes columns for room number, room type, status, and price, providing a quick overview of the current room availability and occupancy. The search functionality allows administrators to quickly find specific rooms by typing keywords.**

**Overall, the "Available Rooms" page is a crucial tool for hotel administrators to efficiently manage room availability and ensure smooth operations within the hotel.**

****

****

****

**A white background with black and red text

Description automatically generated with medium confidence**

1. **HomePage:**

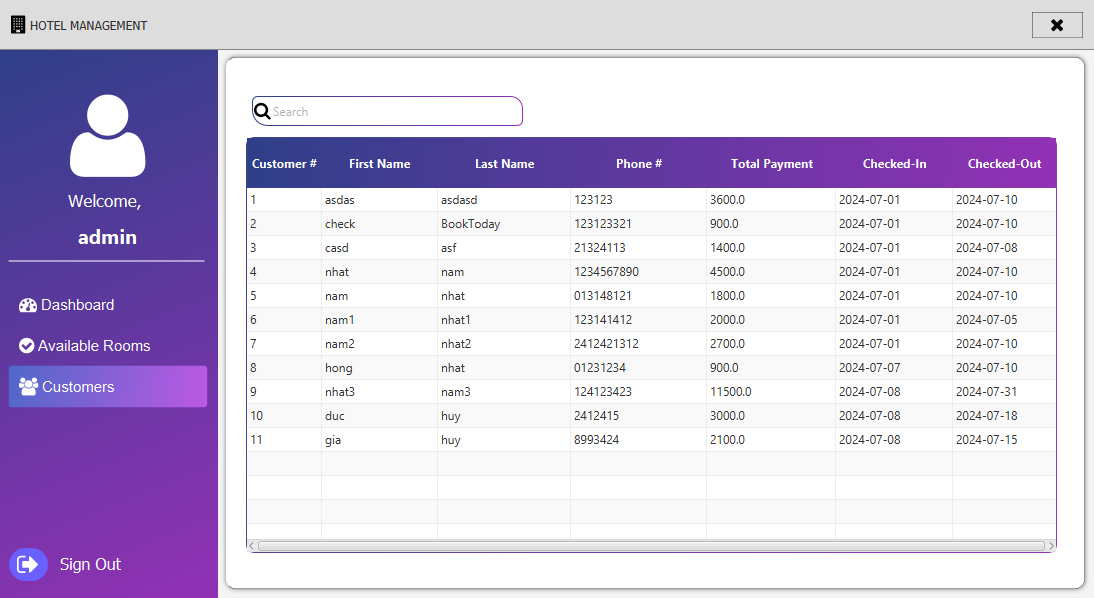
Welcome to the Home Page of our Hospitality Management System! This page serves as the central hub for navigating our platform. Designed with user convenience in mind, it features a modern and vibrant interface that highlights key aspects of hospitality management.

#### **Key Features:**

* **Welcome Message:** Personalizes the experience by greeting the user by name.
* **Navigation Menu:** Located on the left, the menu provides easy access to various sections, including Room Detail, Contact Us, Profile User, Blogs, and About Us. Each section is clearly labeled and accessible with a single click.
* **Main Banner:** The main banner showcases an illustration related to hospitality management, emphasizing the integration of technology in the industry.
* **Hospitality Description:** This section explains how our platform helps hospitality providers enhance both risk management and guest experiences. It highlights the benefits of situational awareness and automation in delivering superior service.
* **About Us Button:** Offers a quick link to learn more about our company's mission, values, and the team behind the platform.

Our Home Page is designed to provide a seamless and informative experience, guiding users effortlessly through the features and benefits of our hospitality management system.

1. **Customer:**

****

The Customer Page of our Hotel Management System is designed to provide administrators with a comprehensive view of all registered customers. This page offers an easy-to-navigate interface, enabling efficient customer management and streamlined operations.

#### **Key Features:**

* **Welcome Message:** Greets the admin user, enhancing the personalized experience.
* **Navigation Menu:** Located on the left side, it includes links to the Dashboard, Available Rooms, Customers, and Sign Out. Each section is easily accessible with a single click.
* **Search Functionality:** A search bar at the top allows admins to quickly find customers by their name, phone number, or other details.
* **Customer Table:** The main content area displays a detailed table with the following columns:
  + **Customer #:** Unique identifier for each customer.
  + **First Name:** The customer's first name.
  + **Last Name:** The customer's last name.
  + **Phone #:** The customer's contact number.
  + **Total Payment:** The total amount paid by the customer.
  + **Checked-In:** The date the customer checked in.
  + **Checked-Out:** The date the customer checked out or is scheduled to check out.

#### **Benefits:**

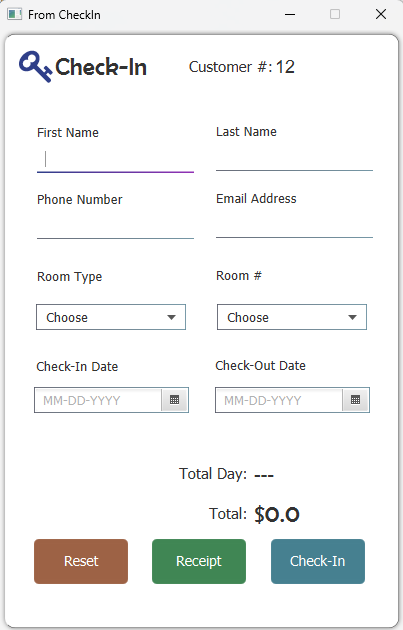
* **Efficiency:** Enables quick access to customer information, helping admins manage reservations and track payments effectively.
* **Organization:** Provides a structured and organized view of customer data, making it easier to handle multiple customers simultaneously.
* **Searchability:** The search function enhances the admin's ability to find specific customer details without scrolling through the entire list.

This Customer Page is a crucial component of our Hotel Management System, ensuring that administrators can manage customer information with ease and efficiency.

****

****

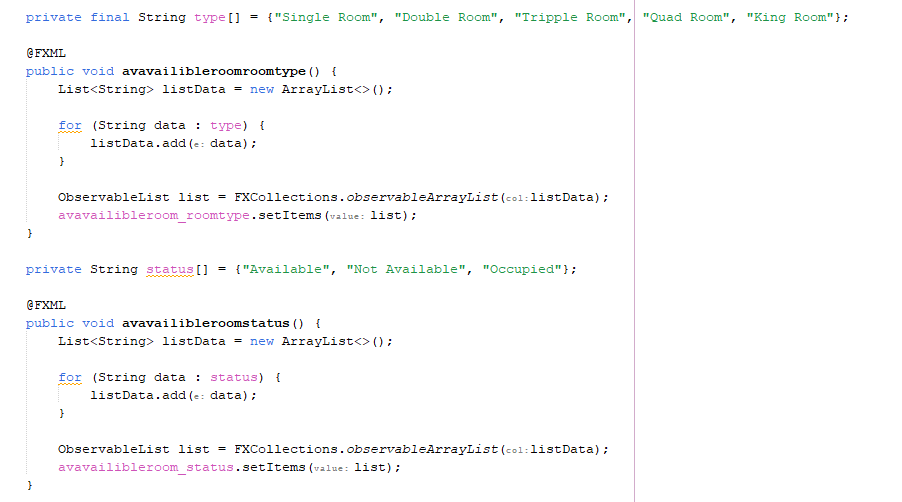
1. **Check In Form:**

****

The Check-In Form Page of our Hotel Management System allows administrators to efficiently register new customer check-ins. This page is designed to collect and organize essential customer information, ensuring a smooth check-in process.

* **Customer ID:** Unique identifier for each customer.
* **First Name:** The customer's first name.
* **Last Name:** The customer's last name.
* **Phone Number:** The customer's contact number.
* **Email Address:** The customer's email for correspondence.
* **Room Type (Dropdown):** Selection box to choose the type of room (e.g., Single, Double, Suite).
* **Room (Dropdown):** Selection box to choose the specific room assigned to the customer.
* **Check-in Date:** The date the customer checks in.
* **Check-out Date:** The date the customer checks out.
* **Total Day:** Automatically calculated based on the check-in and check-out dates.
* **Total Price:** Automatically calculated based on the room type and the total number of days.
* **Reset:** Clears all the input fields in the form, allowing the admin to start over.
* **Receipt:** Generates a receipt with the customer's check-in details and total payment.
* **Check-in:** Submits the form and completes the check-in process, registering the customer in the system.

This Check-In Form Page is essential for managing customer arrivals, ensuring accurate data collection, and facilitating a seamless check-in experience.

****

REVIEW 3 – TASKSHEET

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Project Ref. | Project Title:  Game Store | Date of Preparation of Activity Plan:  17/06/2024 | | | | |
| No. | GUI DESIGN | Start Date | Actual Days | | Member | Status |
| 1 | 7/6/24 | 17/6/24 | | Nguyen Quang Thinh  &  Pham Minh Tri  &  Nguyen Nhat Nam  &  Tran The Anh | completed |
| Review | | | | Signature of instructor | | |
|  | | | |  | | |
|  | | | | **Mr. Pham Cong Danh** | | |