WEB-BASED HOTEL RESERVATION SYSTEM FOR NECTAR MOUNT RESORT

P. D. B. D. SIRIMEVAN

June 2023

WEB-BASED HOTEL RESERVATION SYSTEM FOR NECTAR MOUNT RESORT

P.D.B.D. Sirimevan

R181043

1810431

Name of the Supervisor: Mr. M.P.S. Wijesinghe

June 2023

This dissertation is submitted in partial fulfillment of the requirement of the Degree of Bachelor of Information Technology (External) of the University of Colombo School of Computing

Declaration

Declaration

I certify that this dissertation does not incorporate, without acknowledgment, any material previously submitted for a degree or diploma in any University/Institute, and to the best of my knowledge and belief, it does not contain any material previously published or written by another person or myself except where due reference is made in the text.

Signature of Candidate: . . . . . Date: 02/06/2023

Name of Candidate: P.D.B.D. Sirimevan

Countersigned by:

Signature of Supervisor(s)/Advisor(s): . . . . . Date: 02/06/2023

Name(s) of Supervisor(s)/Advisor(s): Mr. M.P.S. Wijesinghe

Abstract

The Nectar Mount Resort has been recognized as a premier hotel in the Rathnapura area since 2017. It is situated in a picturesque location, offering stunning views and a serene environment. Two reception halls are possessed by them, which are considered perfect venues for celebrating various events such as birthday parties, engagements, wedding receptions, business Parties, Private Parties, etc. They provide a unique and memorable experience for their guests with the most affordable and particularly flexible packages.

Currently, a manual file-based process is used for their management, which leads to various issues throughout the business process. The current system at the hotel requires customers to be visited in person for reservations and payments. Manual processes for tasks such as payment processing, event scheduling, and customer communication are considered inefficient. Effective communication of important information to staff and customers can be found to be challenging, leading to confusion and errors. The task of generating reports from previous years is also found to be time-consuming.

To address the limitations of the manual system, a web-based hotel reservation system was developed. It was built using PHP as the programming language, following the modern waterfall model. MySQL was selected to design the database structure. To improve the user experience, various client-side technologies like HTML, CSS, JavaScript, and Bootstrap were integrated. XAMPP served as the web server, and NetBeans was the chosen Integrated Development Environment (IDE). Extensive testing was carried out using Google Chrome, and the UML diagrams in the dissertation were created with draw.io.

The client's requirements are intended to be met by the proposed system, and it is expected that Nectar Mount Resort's business activities will be enhanced in a more productive and well-organized manner. The hotel will be enabled to automate its management process, providing efficient handling of bookings and streamlining customer service through the implementation of this system. By utilizing this system, the hotel will be able to operate more effectively and offer a better experience to its guests.

Acknowledgments

My heartfelt gratitude is wanted to be expressed to all those who supported and guided me during this project. In particular, I am especially thankful to my supervisor, Mr. M.P.S. Wijesinghe, for his invaluable advice and knowledge, which played a pivotal role in the success of this endeavor.

I am also thankful to express my gratitude to Nectar Mount Resort for their cooperation and insights into the hotel's operations. I extend my appreciation to the staff at Nectar Mount Resort for their assistance during the data-gathering phase. My friends and family deserve recognition for their unwavering support and belief in me.

The University of Colombo School of Computing's BIT Coordinator and the Project Examination Board of Bachelors of Information Technology are also to be thanked for providing the necessary resources for the BIT Degree Program.

To all those involved, big or small, I sincerely appreciate your contributions. Thank you for being an integral part of this project and enhancing my learning experience.

Table of Contents

List of Figures

List of Tables

List of Acronyms

PHP - PHP Hypertext Preprocessor

SQL – Structured Query Language

UCSC – University of Colombo School of Computing

BIT – Bachelor of Information Technology

XAMPP – Cross-Platform, Apache, MySQL, PHP, Perl

IDE – Integrated Development Environment

HTML – Hyper Text Markup Language

CSS - Cascading Style Sheets

UML - Unified Modeling Language

SDLC - Software Development Life Cycle

Chapter 01 – Introduction

Nectar Mount Resort has been known as a leading hotel in the Rathnapura area since 2017. Two reception halls are possessed by them, which are considered perfect venues for celebrating various events such as birthday parties, engagements, wedding receptions, business Parties, Private Parties, etc. The most affordable and especially flexible packages around the area are offered to customers by them.

1.1. Project Motivations

Nectar Mount Resort Currently their management is done manually using a file-based system. which leads to various issues throughout the business process. The motivation factors for the project are as follows.

- The customers must need to visit the hotel to place their reservations and make their payments which could be a highly time-consuming process and it is considered inefficient in today's digital age.
- Manual processes such as processing payments, scheduling events, and communicating with customers can be time-consuming, leading to inefficiencies and decreased productivity.
- Sometimes, the manual system may result in scheduling overlapping and being duplicated.
- Difficult to communicate important information to staff and customers with a manual system. For example, confusion and errors may be caused by ineffective communication of changes to event details.
- Difficult to keep track of all the relevant information with manual processes, which can lead to errors in recording data and difficulty in finding the required information when needed.
- The process of generating reports of previous years is made highly time-consuming by the manual system.
- The security of the data is considered an issue as the files can be easily accessed by anyone.

- Issues with business documents can be caused by the loss of information and difficulty in recovering it in the event of a disaster.

1.2. Project Objectives

As a Solutions for the above-mentioned problems were developed the web-based hotel reservation system. By addressing these motivations and implementing an automated system, operational efficiency can be achieved by the hotel, customer satisfaction can be enhanced, decision-making can be improved, and competitiveness can be maintained in the dynamic hospitality industry. Below is listed the objectives of this project.

- Reservation and Payment Process Streamlining: A digital reservation and payment system will be developed, allowing reservations and payments to be made online by customers, thus eliminating the need for physical visits to the hotel.
- Automation of Manual Processes: Automation will be implemented for payment processing, event scheduling, and customer communication to improve efficiency and productivity.
- Scheduling Conflict Prevention: A system will be created to ensure that scheduling conflicts and duplications are minimized or entirely eliminated.
- Communication Enhancement: Communication between staff and customers will be improved by providing effective tools for conveying important information, including changes to event details, in a clear and timely manner.
- Efficient Data Management: A system will be developed to enable efficient data management, reducing errors in data recording and enhancing the ability to promptly retrieve necessary information.
- Report Generation Simplification: The process of generating reports from previous years will be simplified, making it less time-consuming and more accessible for business analysis and planning.
- Data Security Enhancement: Robust data security measures will be implemented to protect sensitive information, addressing concerns about unauthorized access.

1.3. Project Scope

The scope of the desired system is the primary factor that needs to be taken into account the most while developing a system. Below are some details on the system's scope.

1. Login - An essential component of the system is the login module, as it is provided a secure and controlled environment for users to access the system and manage their tasks effectively. User authentication and authorization, Password reset and recovery and Account lockout for security are including in this module and System administration tasks are excluded.

2. Customer Management - A streamlined and convenient way is provided by the customer management module for new customers to register and create an account through online. Additionally, hotel staff is helped by it to keep track of their customer information.

3. Reservation Management - Availability checking, Bookings made by customers can be managed by hotel staff, including reviewing and accepting or rejecting reservations, editing reservations, Reservation history tracking with the help of this module.

4. Customer Payment Management - Secure and efficient payment processing for customers, the streamlined billing process, Accepting and processing customer payments, managing payment methods and transactions and revenue and financial data tracking are ensured by this module, making it a critical component of a system. Online payment gateway integration is not including in this module.

5. Service Management - The services offered at the reception hall can be effectively managed, service booking and availability can be streamlined, and the overall customer experience can be improved by the service management module. Service provider management is not including in this module.

6. Menu Item Management – Creating and managing individual menu items, specifying item details, prices and Menu item availability and updates are including in this module. This module helps to create new menu packages for using various types menu item.

7. Menu Package Management - Effective management of the menu package options offered to customers, streamlining the booking process, and improvement of the overall customer experience are facilitated by the menu package management module. Creating and managing pre-defined menu packages, specifying package details, prices, and inclusions, Package availability and updates are including in this module. Customizing menu packages is not including in this module.

8. Wedding Hall Management - The effective management of physical spaces within the reception hall, streamlined space booking and availability, setting up hall configurations and layouts, Handling event-specific requirements are facilitated by this module.

9. Employee Management - The process of managing employee's information and records can be streamlined and the overall efficiency of reception hall operations can be improved with the help of the employee management module. Scheduling and shift management, tracking employee attendance and performance, Payroll and benefits administration, Training and development programs are not including in this module.

10. User/Security Management - The user management module of a web-based hotel management system is considered a critical component, as it allows user accounts to be created and managed by administrators for staff members, and customers, who need access to the system. Control of user permissions of the system based on user roles can be set up by administrators using this module. The security of the system can be improved, and proper access to the correct information can be ensured through user account management. Network security and infrastructure management, Data backup and disaster recovery are not including in this module.

11. Report Generation - The reporting module of a web-based hotel management system allows various reports to be generated by users, providing insights into different aspects of the business. With this module, managers can monitor the performance of their business and informed decisions can be made to improve the overall profitability of the business.

12. Customer Reviews and feedback- With this module, feedback on their experiences with the reception hall's services and facilities can be provided by customers in the form of

ratings, comments, and suggestions. This feedback can be used by hotel staff to identify areas for improvement and make necessary changes to their operations.

13. Refund Payment Management – This module allows refunds for canceled events or overpayments to be managed and processed by businesses. The refund process is streamlined by the system through the automation of calculations, generation of refund requests, updating payment records and issuing refunds, and tracking of refund statuses.

14. Hall Arrangement - In a web-based hotel reservation system, the process of organizing events is simplified by the hall arrangement feature. Their event type and date can be easily selected by users, and tables, chairs, and other items can then be visually arranged with a user-friendly drag-and-drop interface. The hall layout can be adapted to their specific needs by users, accommodating guest numbers, seating preferences, and additional event requirements with customization options. A visual preview is provided, and reports are generated for reference and printing by the system once the arrangement is finalized. Event planning is streamlined by this feature, ensuring a seamless and organized experience for both hosts and guests.

The novelty of web-based hotel reservation system lies in the ability to digitize and streamline management processes associated with reception halls. Tasks are automated, online booking and payment are enabled, reporting and analytics are provided, customer feedback is collected, and streamlined communication is facilitated. Overall, traditional manual processes are transformed into a more convenient and efficient approach, and customer experience is enhanced while venue owners and staff are empowered with valuable insights.

1.4. Outline of the dissertation

The chapters listed below are part of the SDLC phases, and their purpose is to show how the work was done to construct the suggested system during each step.

Chapter 02 Analysis - In this chapter, a description of the requirement-gathering method and the analyzing techniques used in the project is provided. References to existing similar systems consulted during the development of the proposed system are mentioned, and both functional and non-functional requirements of the system are presented.

Chapter 03 Design - A presentation of a few user-interface designs and a depiction of the system design through diagrams can be found within this chapter.

Chapter 04 Implementation - In this chapter, the technologies employed in the implementation, the hardware and software requirements, and an outline of the major code module structure are described.

Chapter 5 Evaluation - The testing of the system with test cases and an explanation of the obtained results are encompassed by this chapter. A detailed description of the test cases and scenarios employed is also provided.

Chapter 06 Conclusion - In this chapter, insights into the lessons learned during the project are found, along with recommendations for future improvements to the system.

Appendices - Additional details about the developed system that are not included in the preceding chapters but are relevant to the development of the system are contained within the appendices.

Chapter 02 –Analysis

In this chapter, the first and most critical stage of the software development process, requirement analysis, will be addressed. An overview of the existing system and an outline of an existing similar solution will be presented. Functional requirements and quality attributes will be discussed, and the suitable process model will be identified with a justification. By focusing on these elements, a strong foundation for the project can be established, ensuring its success.

2.1. The Current System

In the analysis phase of a software development project, the creation of high-level use case diagrams and flow charts to explain the current system or process is one of the activities. The functionality and flow of the existing system are understood by stakeholders through these visual representations before any changes or improvements are made.

2.1.1. High-level use case diagram for existing system

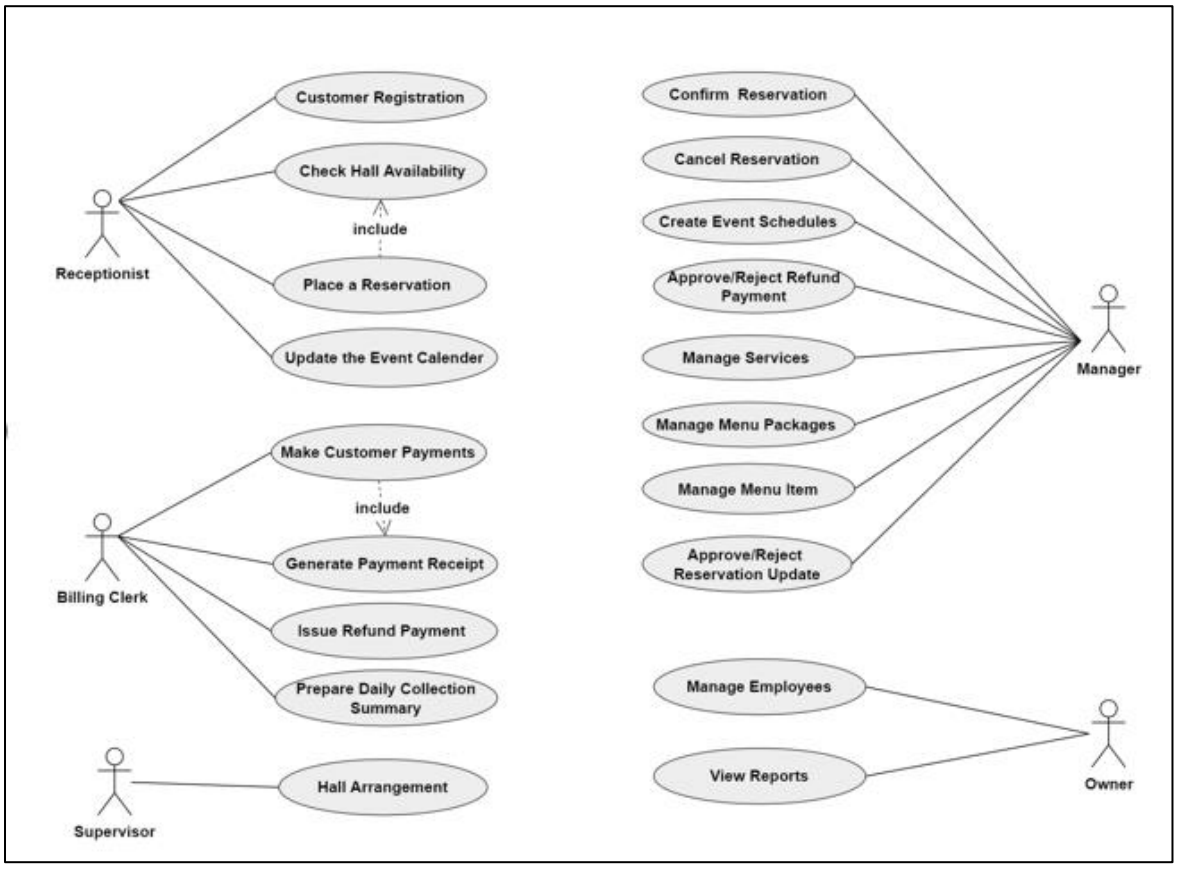


Figure 2. 1 High level Use Case Diagram for existing system

2.1.2. Flow chart for existing system

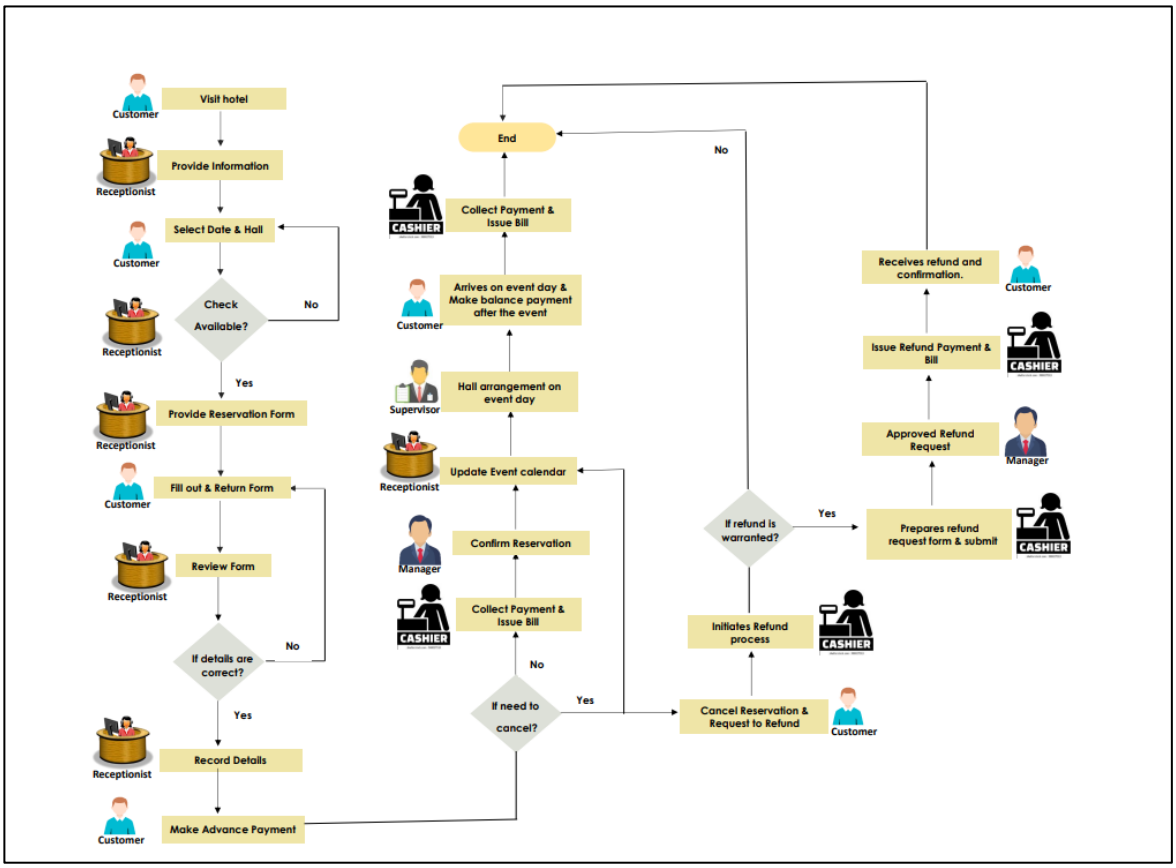


Figure 2. 2 Flow Chart for an existing system

2.2. Review Similar Systems

During the analysis phase, a crucial role is played by the Similar Systems review in evaluating and analyzing different systems that exhibit similarities in functionality, features, or purpose. In reviewing similar systems, the strengths, weaknesses, and unique aspects of each system are examined, facilitating the identification of the most suitable solution for a given context.

2.2.1. Weddings & Events Booking System of the Hilton

The Hilton Colombo weddings and events booking system has been designed as an efficient and convenient online platform to enable clients to book halls for various events at the hotel. A user-friendly interface is provided for clients to browse halls, view their capacity, and book them online. The availability of halls can be viewed and a suitable date and time for the event can be selected by clients. Furthermore, a range of amenities and services offered by the hotel, including catering and audiovisual equipment, can be chosen by clients. Halls can be booked from anywhere with an internet connection using the system, and any concerns or queries can be discussed with the hotel staff via the online platform [1].

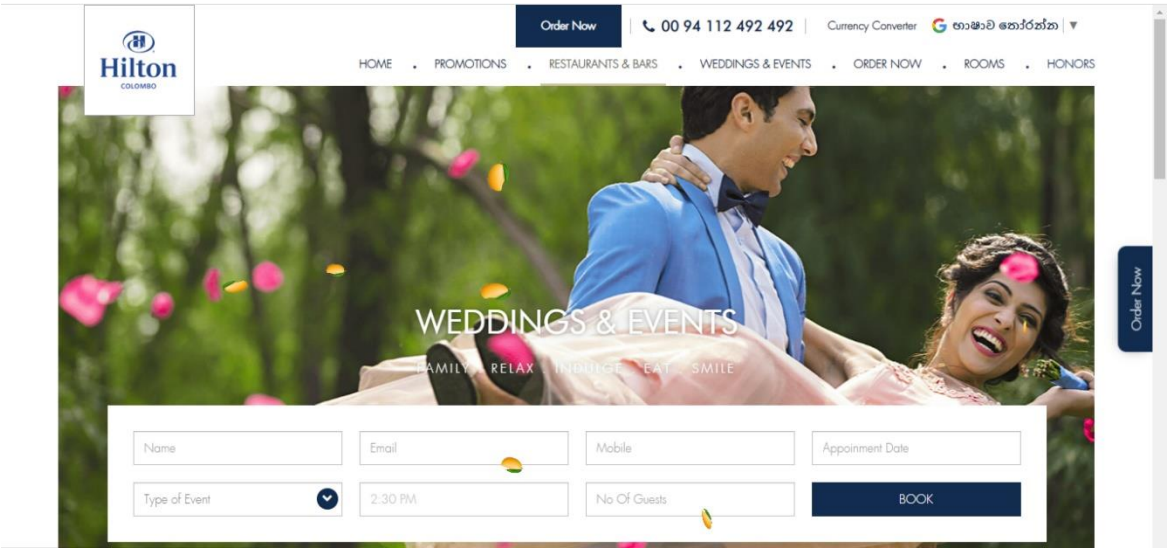


Figure 2. 3 Website Interface of the Hilton

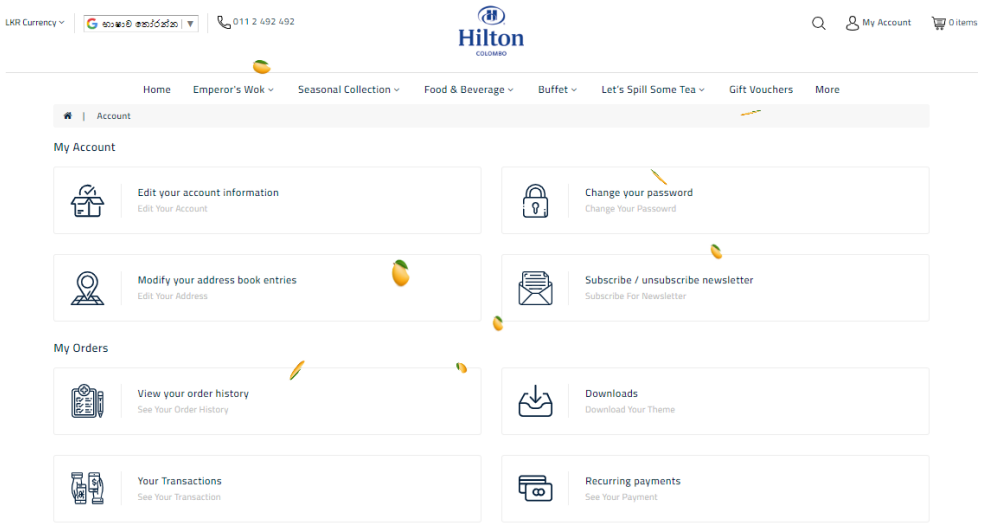


Figure 2. 4 Customer Dashboard of the Hilton

2.2.2. Online Banquet Booking System of OBBS

The primary aim of this application's development is for a system to be created that can efficiently oversee the management of data connected with the numerous banquets booking events held at our venue. The core objective is for a centralized database encompassing all pertinent information related to these banquet events to be established. Our ultimate goal is for the comprehensive support of various functions and processes essential for efficient data management to be ensured by the system. The website and information pertaining to the "Online Banquet Booking System" can be accessed by users, and communication with the administration can also be facilitated through message submissions by them [2].

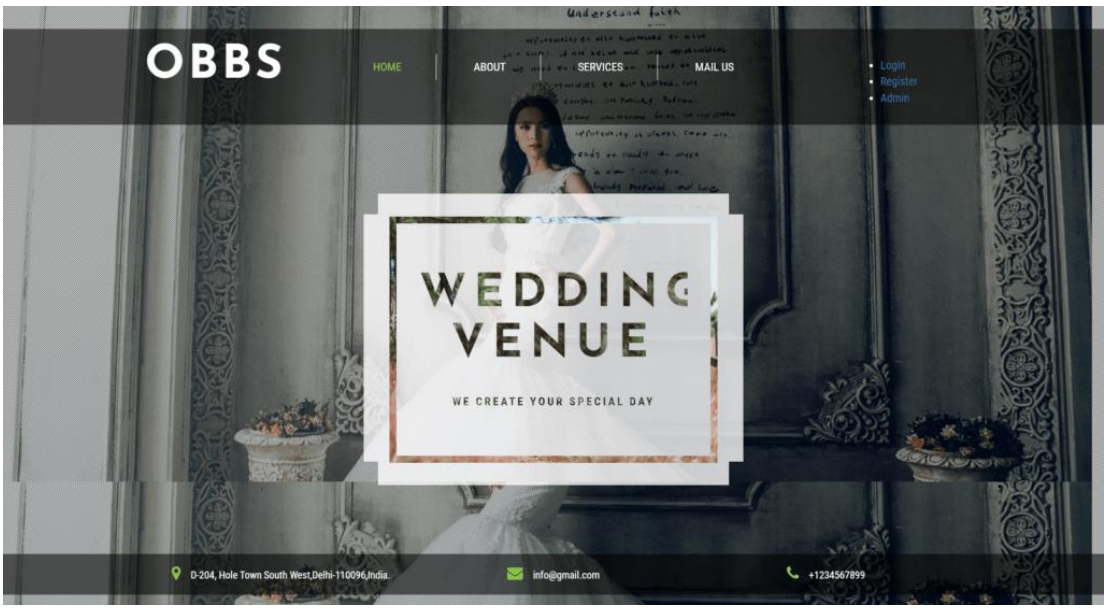


Figure 2. 5 Website Interface of the OBBS

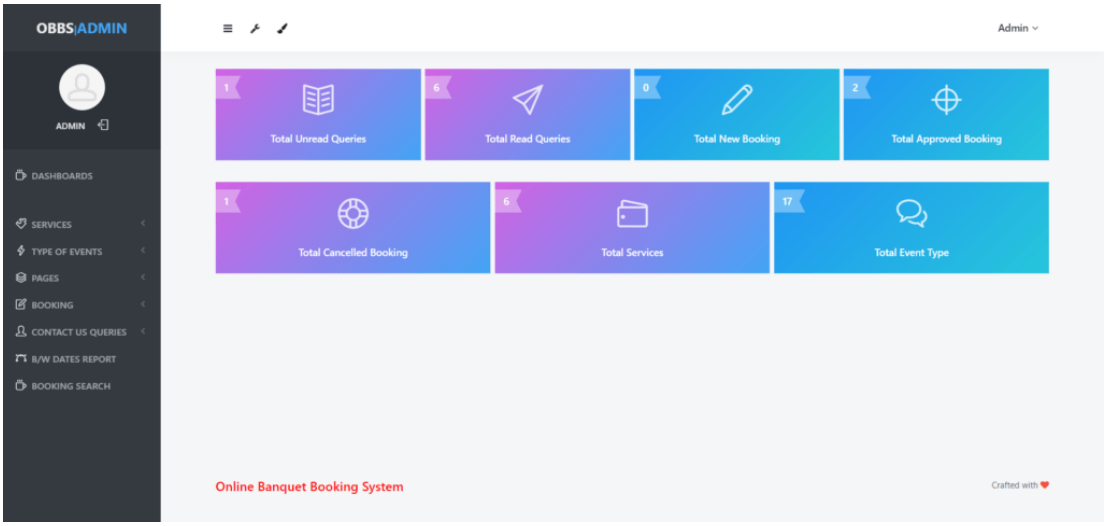


Figure 2. 6 Admin Dashboard of the OBBS

2.2.3. Comparison between Proposed System and Similar Systems

Features	**Proposed System**	**Weddings and Event Booking System of Hilton**	**Online Banquet Booking System of OBBS**
Customer Registration	√	√	√
Reservation Management	√	√	√
Customer Payment Management	√	√	X
Hall Management	√	√	√
Service Management	√	√	√
Menu Package Management	√	√	X
Menu Item Management	√	√	X
Employee Management	√	X	X
User Management	√	√	√
Refund Payment Management	√	X	X
Customer Review and Feedback	√	√	√
Hall Arrangement	√	X	X
Generate Reports	√	√	√

Table 2. 1 Comparison between proposed system and similar systems

2.3. Analysis of the Requirements

The identification and prioritization of functional requirements and quality attributes are considered crucial steps in the software development process. During this phase, to determine the essential features and functionalities that must be possessed by the software. This process assists in the establishment of clear objectives and provides guidance to the develop and focus on the key requirements and attributes that will ensure the success and effectiveness of the software.

2.3.1. Functional Requirements of the System

The mandatory part of a system is constituted by functional requirements. A web-based hotel reservation system should have different functional requirements based on user roles including Admin, Owner, Manager, Receptionist, Billing Clerk and Supervisor. The following functional requirements of the proposed system are as follows.

1. Customer Management

- Shall be able to register as a new customer.
- Shall be able to view customer details.
- Shall be able to update customer details.

2. User Management

- Shall be able to login to the system
- Shall be able to add new user accounts.
- Shall be able to view user accounts.
- Shall be able to update user accounts.

3. Employee Management

- Shall be able to add new employees.
- Shall be able to view employees.
- Shall be able to update employees.

4. Menu Item Management

- Shall be able to add new menu item.
- Shall be able to view menu item.
- Shall be able to update menu item.

5. Menu Package Management

- Shall be able to add new menu package.
- Shall be able to view menu package.
- Shall be able to update menu package.

6. Service Management

- Shall be able to add new service.
- Shall be able to view service.
- Shall be able to update service.

7. Hall Management

- Shall be able to add new hall.
- Shall be able to view hall.
- Shall be able to update hall.

8. Reservation Management

- Shall be able to check the Hall availability.
- Shall be able to make a new reservation.
- Shall be able to view reservations.
- Shall be able to update reservations.
- Shall be able to cancel reservations.

9. Customer Payment Management

- Shall be able to add new customer payment.
- Shall be able to view customer payments.
- Shall be able cancel customer payments.
- Shall be able to generate payment receipt.

10. Refund Payment Management

- Shall be able to request refund payments.
- Shall be able to view refund payments.
- Shall be able to update refund payments.
- Shall be able to generate refund payment receipt.

11. Customer Review Management

- Shall be able to add customer reviews.
- Shall be able to view customer reviews.
- Shall be able to reply customer reviews.

12. Hall Arrangement

- Shall be able to add new hall arrangement.
- Shall be able to view hall arrangement.
- Shall be able to update hall arrangement.

13. Generate Reports

- Shall be able to generate management reports.

2.3.2. Quality attributes of the System

The attributes that describe the reliability, portability, response and processing times and other characteristics of a system are non-functional requirements. Here are some reasons why a web-based hotel reservation system should pay close attention to Reliability, portability, response, and processing times.

- **Reliability** - The availability and functionality of the system at all times are expected by users, and therefore, reliability is considered important in any web-based system. In the case of a hall management system, loss of revenue and customer dissatisfaction could be the result of double bookings or cancellations caused by an unreliable system. A reliable system that can handle high volumes of traffic and user requests without crashing or experiencing downtime is essential.
- **Portability** - It refers to the capacity of the system to be easily transferred to different environments, such as different web browsers or operating systems. The ability of a portable system to ensure that users can access the system from any device, anywhere, without encountering compatibility issues is crucial. In a hall management system, this is especially important, as customers may need to access the system from different devices, such as laptops or smartphones.

- **Response time and processing times** - In a web-based hotel management system, it's crucial to have quick and efficient response times to meet users' expectations. Slow response times could lead to dissatisfaction among users, resulting in lost business. Additionally, if the processing time is slow, it could affect the system's overall efficiency and lead to scheduling conflicts and delays in bookings. To ensure the system responds efficiently, efficient coding practices and performance optimization are vital.
- **Usability** - A system that is easy to use and navigate by both customers and staff is essential, making usability a critical consideration. An intuitive and user-friendly interface improves system efficiency, reduces training time for new users, and minimizes errors. Difficult-to-use systems could lead to frustration and lost business. Therefore, it's important to design a clear and easy-to-navigate user interface.
- **Security** - Dealing with sensitive customer data such as personal and financial information makes security critical for any web-based system. In the case of a reception hall management system, this includes booking and payment details. Security breaches could result in data loss, identity theft, and legal and financial consequences for the business. Therefore, it's necessary to implement robust security measures, including encryption, firewalls, and regular security updates, to protect the system and its users' data.
- **Availability** - A reception hall management system needs to be accessible and functional 24/7, making availability a crucial consideration. If the system experiences downtime, it could lead to lost revenue and damage to the business's reputation. Therefore, it's essential to ensure the system is reliable and can handle high volumes of traffic and user requests without crashing or experiencing downtime. This includes utilizing load balancing, redundancy, and failover mechanisms to guarantee continuous system availability.

2.4. Process Model

A sequential approach is followed by the web-based hotel management system's modern waterfall process model, a development process where each stage is finished before going on to the next. The system architecture is designed and a detailed design specification is created, followed by gathering and analyzing requirements.

The development team begins the implementation phase once the design phase is over, during which the system is constructed and tested. The system is thoroughly tested after that for faults, mistakes, and usability.

After successful testing, the system is deployed to a production environment and made ready for use. Ongoing maintenance and support, including updates, bug fixes, and new feature implementations, are involved in the final phase. Throughout the process, documentation is maintained and updated to ensure a comprehensive understanding of the system's functionality and design [3].

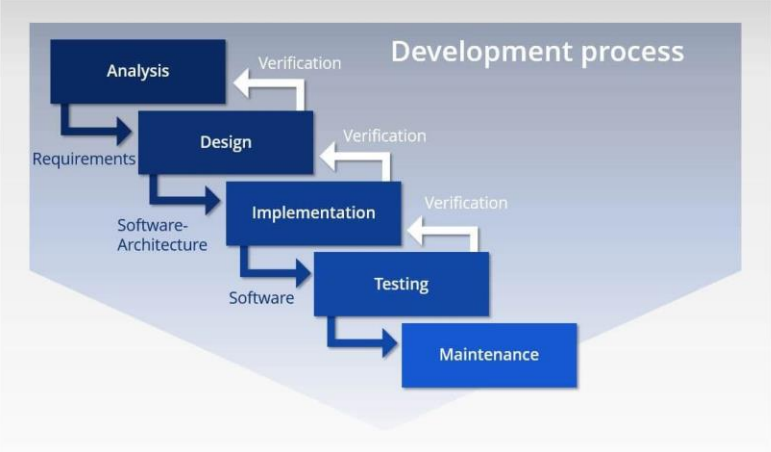


Figure 2. 7 Modern Waterfall Process Model

Chapter 03 – Design

A vital role is played by the design phase in every software development project, as it serves as a critical bridge between the initial analysis and the actual implementation.

3.1. Relevant Design Diagrams

3.1.1. Different competing design strategies

For the Nectar Mount Resort System, two competing design strategies are being considered: the development of a stand-alone system or a web-based system. The stand-alone system would be designed as a self-contained application that runs locally, providing offline functionality and enhanced data security. Conversely, the web-based system would be designed to be accessed through a web browser, offering everywhere to access, real-time updates, and simplified maintenance. The decision between the two strategies is influenced by factors such as specific requirements, infrastructure, and user needs.

3.1.2. Architectural Design of The System

In this case, the system is planned to be developed using a non-object-oriented approach, specifically procedural programming, following the concept. Emphasis is placed on the use of procedures or functions to manipulate data and execute tasks, while focusing on step-by-step execution and the logical flow of instructions. Simplicity, efficiency, and direct control over program flow are offered by this approach, making it suitable for systems where data manipulation and algorithms are central. To ensure a well-structured and efficient system, the architectural design will incorporate the principles and best practices of procedural programming [4].

3.1.3. Use Case Diagrams

The use case diagram gives a high-level picture of the functioning of the system and interactions between various users. It is served as a starting point for system design and development, offering insights into the system's behavior and the roles of various actors. The use case diagram for the suggested system is displayed below, illuminating the relevant actors and their associated interactions.

Use Case Diagram of the Customer

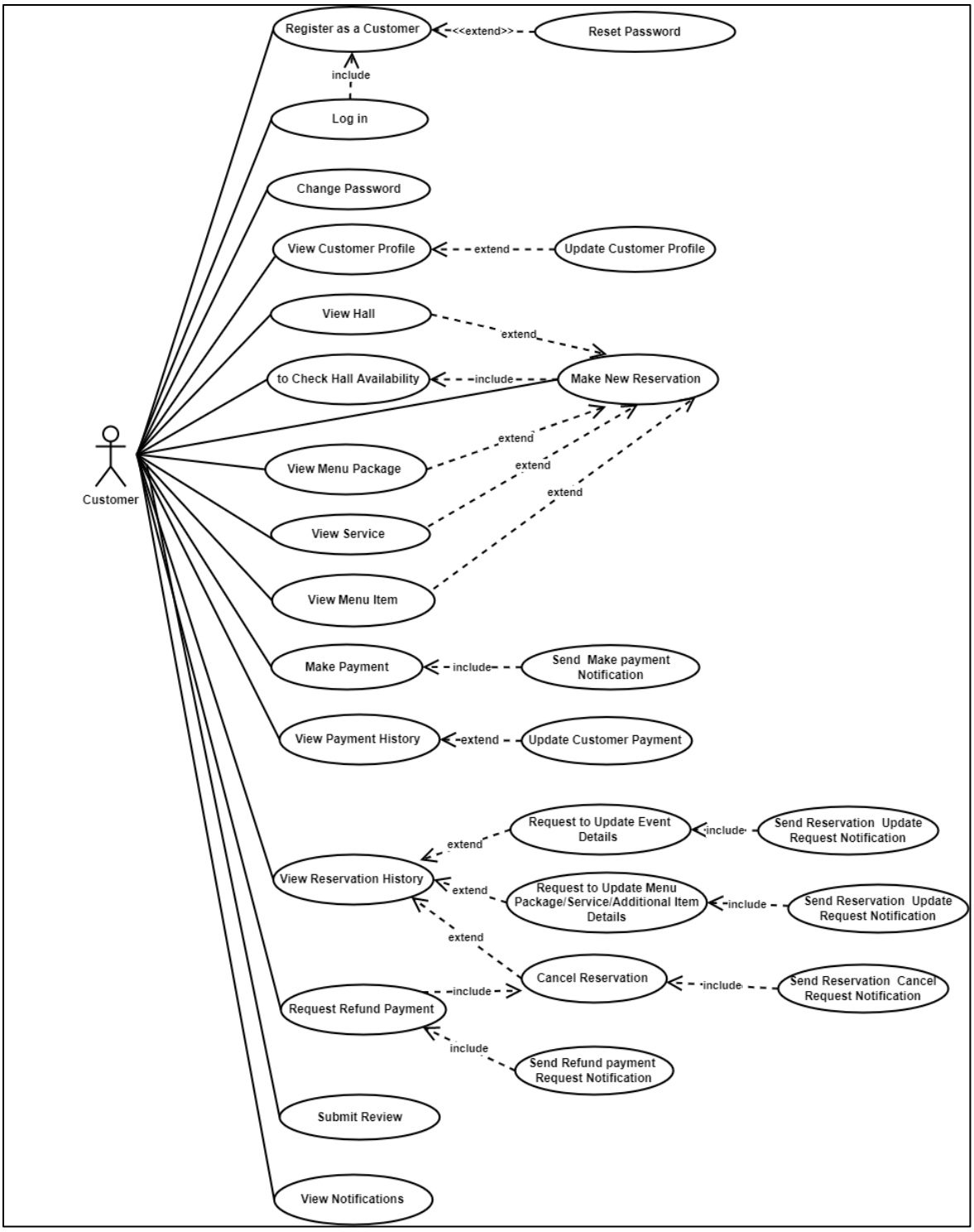


Figure 3. 1 Use Case Diagram of the Customer

Use Case Diagram of the Manager

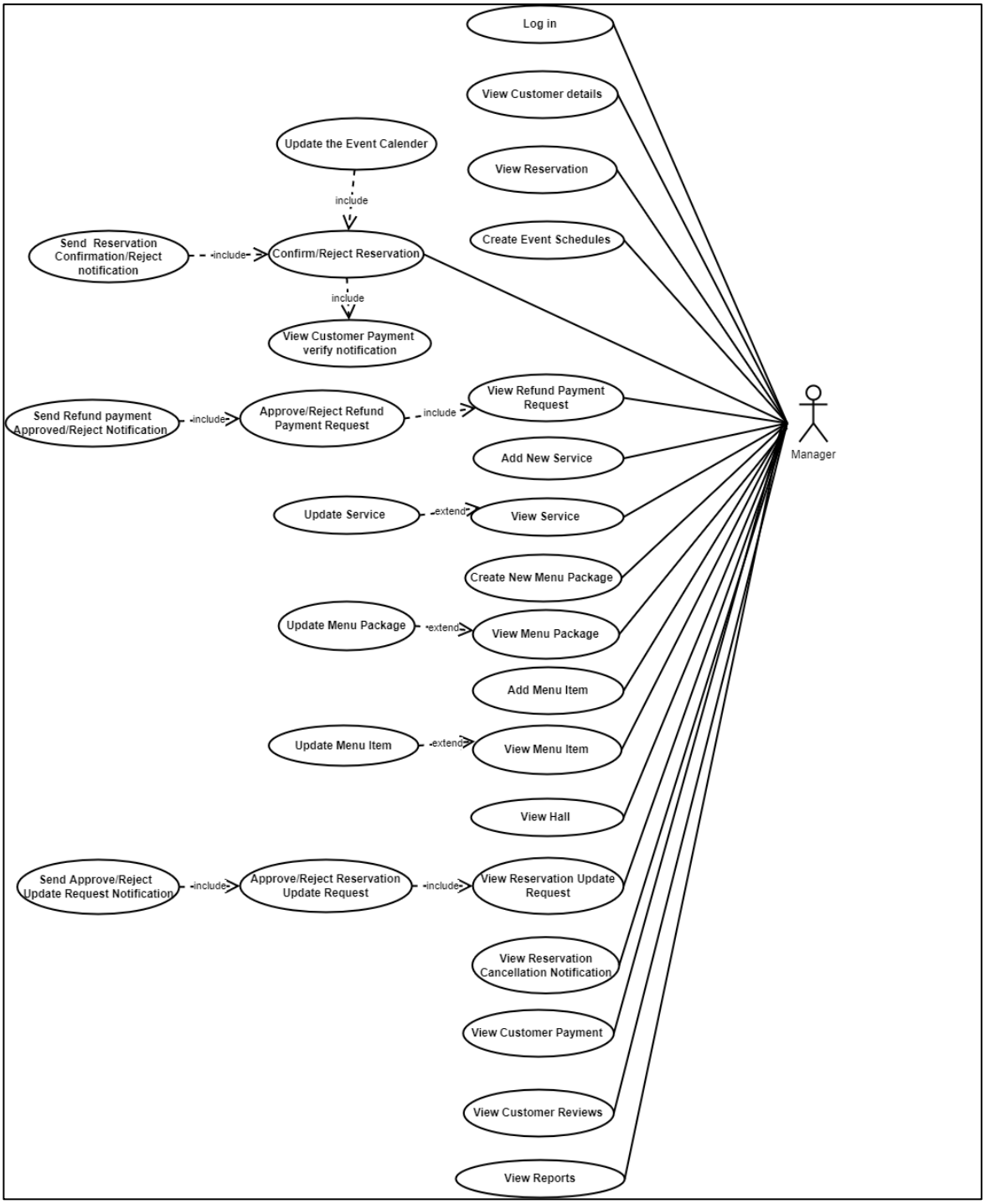


Figure 3. 2 Use Case Diagram of the Manager

Use Case Diagram of the Receptionist

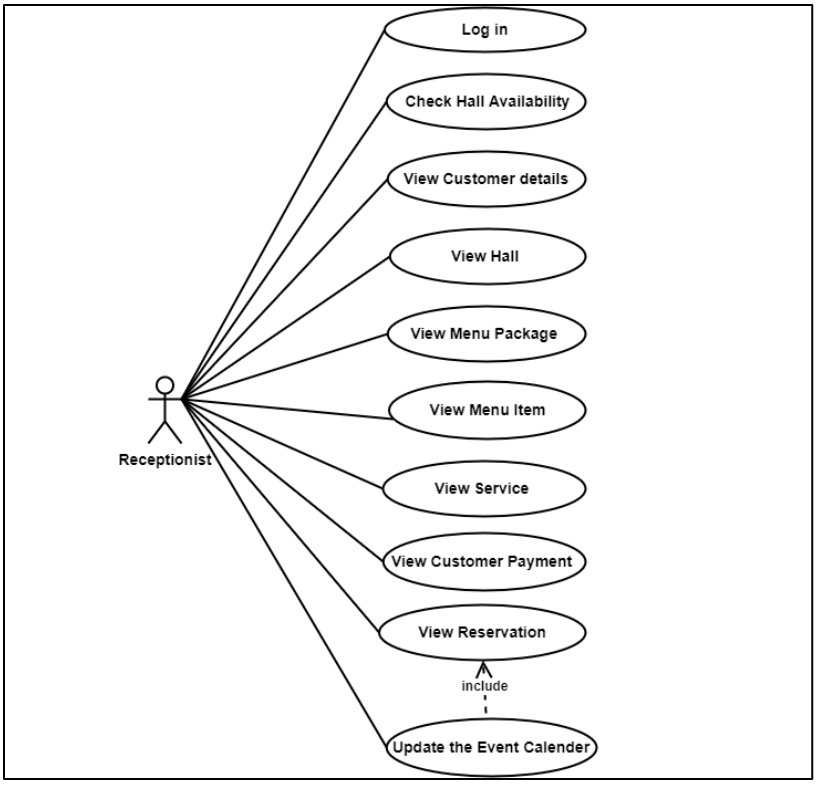


Figure 3. 3 Use Case Diagram of the Receptionist

Use Case Diagram of the Supervisor

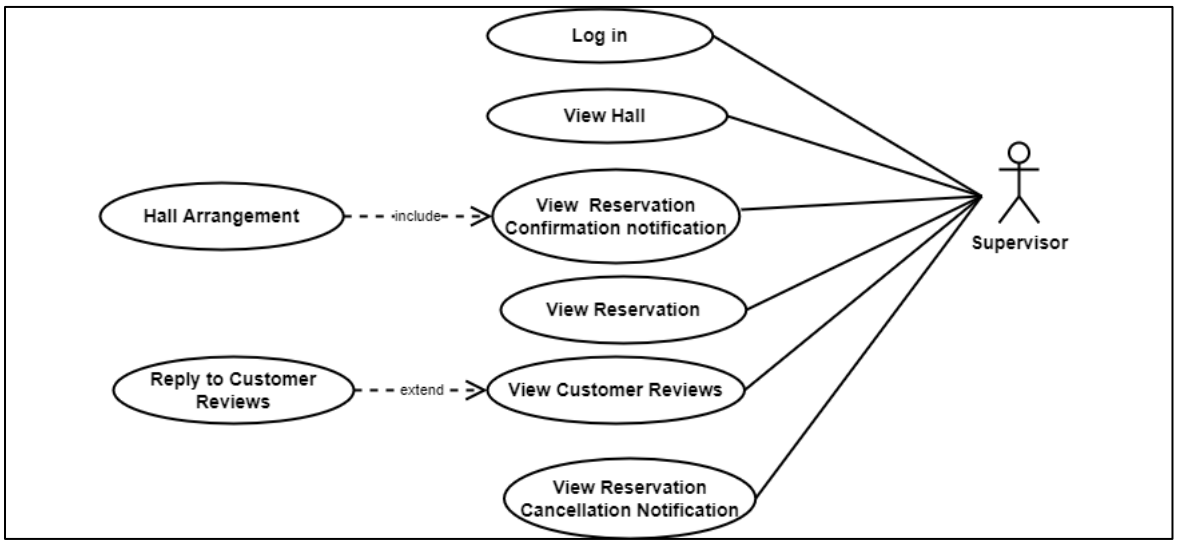


Figure 3. 4 Use Case Diagram of the Supervisor

Use Case Diagram of the Owner

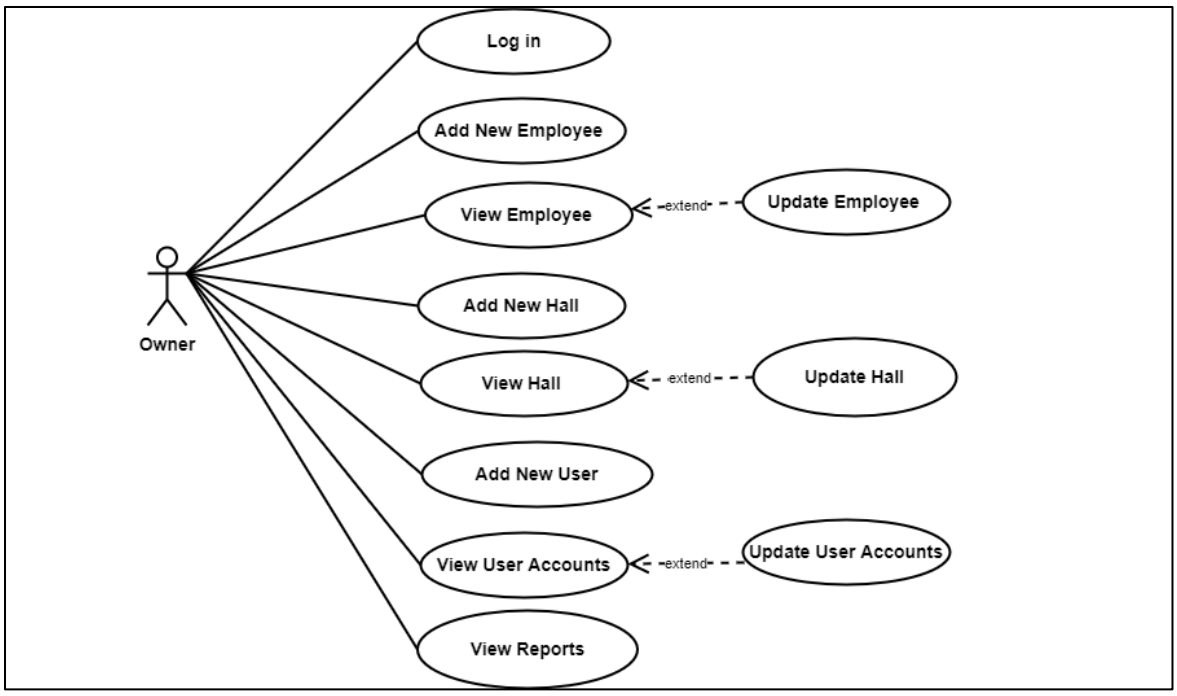


Figure 3. 5 Use Case Diagram of the Owner

Use Case Diagram of the Billing Clerk

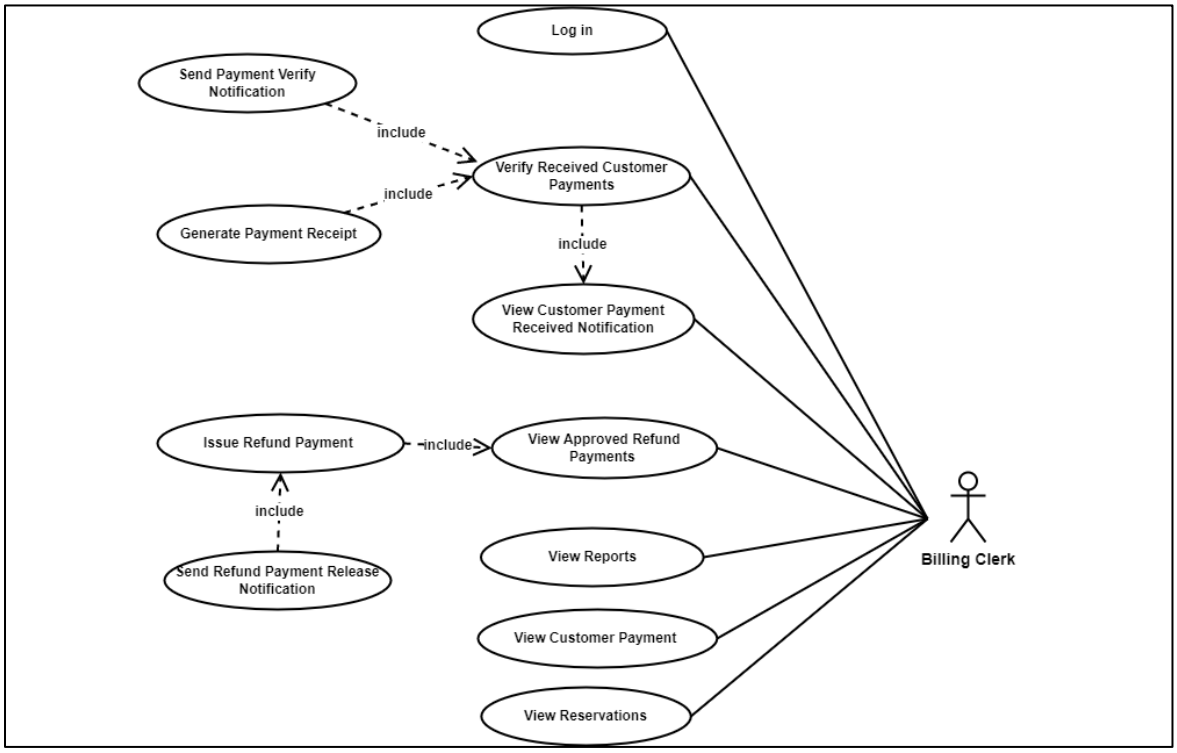


Figure 3. 6 Use Case Diagram of the Billing Clerk

3.1.4. Use Case Narratives

A detailed account of how a particular use case is executed within a system is provided by a use case narrative. The specific steps, actions, and interactions between actors and the system to accomplish a specific goal or task are outlined. Within the narrative, a description of the preconditions, main flow of events, alternative or exceptional flows, and the postconditions or expected outcomes is typically included. All the use case narratives of the system are shown below.

Use-case Register as a Customer

Use-case Number	UC-01	
Use-Case Name	Register as a Customer	
Priority	High	
Actor	Customer	
Description	This use case indicates how to register customers into the system.	
Precondition	None.	
Post-condition	Display a Success message after the successful registration. The customer can log in to the System using the new username and password.	
Basic Course of Action	**User Action**	**System Response**
	1. The customer clicks on the Register Now button. 3. The customer fills up the following relevant data into the fields in the form and clicks on the Register button. (Title, First Name, Last Name, NIC, Gender, District Address, Contact No, Contact No 2, Email, User Name, Password, Confirm Password)	2. The system provides a register as a new customer form to fill up. 4. The system checks all required fields have been filled up by a user. 5. The system validates the Contact number, NIC, Email, and Password format. 6. The system validates that the NIC, Email, User Name, and Password already exist. 7. The system checks the Password and Confirm Password is matching. 8. The system encrypts the Password.

	10. The customer Checks the entered details and clicks on the confirm button.	9. The system displays the message of Confirmation. 11. The system saves the details in the database. 12. The system generates the Registration number. 13. The system updates the Registration number in the database. 14. The system shows a "You have Successfully Registered!" message. 15. Use case ends.
An alternate course of Action	4.1 If all required fields of the form weren't filled up, the system notifies the "required field missing" message and redirects to step 3 of the basic course of action to enter again. 5.1 If the user fills up an invalid Contact No, NIC, Email, and Password format. The system notifies the "Invalid format" and redirects to step 3 of the basic course of action to enter again. 6.1 If the user entered the existing NIC, Email, User Name, and Password. The system shows the "You entered existing" message and redirects to step 3 of the basic course of action to enter again or proceed with login. 7.1 If the user enters the Confirm Password is not matching Password, then the system notifies the "The Password and Confirm Password Not Matching" and redirects to step 3 of the basic course of action to enter again.	

Table 3. 1 Use Case Narrative for Register as Customer

Use-case Login

Use-case Number	UC-02	
Use-Case Name	Log in	
Priority	High	
Actor/s	Customer, Receptionist, Billing Clerk, Supervisor, Manager, Owner	
Description	This use case indicates how users are login into the system.	
Precondition	The user must have a user account. The user must have a valid username and password.	
Post-condition	The user is now logged into the system if the use case was successful. If not, the system state is unchanged.	
Basic Course of Action	**User Action**	**System Response**
	1. The User is on the Login page to login into the system. 3. The User enters the username and password and clicks on a login button.	2. The system provides the User to enter a Username and password. 4. Those two fields are verified by the system. 5. If it is correct, the user successfully logged into the system. 6. Use-case Ends.
An alternate course of Action	4.1 If all fields are not filled out and not matched to the username or password the system shows the user a message “Invalid user name or password” and returns to step 3 of the basic course of action to enter again.	

Table 3. 2 Use Case Narrative for Login

Use-case Reset Password

Use-case Number	UC-03	
Use-Case Name	Reset Password	
Priority	High	
Actor/s	Customer, Receptionist, Billing Clerk, Supervisor, Manager, Owner	
Description	This use case indicates how users reset their passwords in the system.	
Precondition	The user must have a user account in the system. The user has forgotten their password.	
Post-condition	The user reset their password if the use case was successful and they can log in to the System using the new password.	
Basic Course of Action	**User Action**	**System Response**
	1. The user clicks on the Log In link.	2. The system provides the user login page of the system.
	3. The User clicks on the "Forgot Password?" link.	4. The System redirects the password reset page.
	5. The User enters their registered email address and clicks on a submit button.	6. The System validates the email address provided by the User.
		7. The System sends an email to the User's registered email address containing a password reset link.
	8. The User receives the email and clicks on the password reset link.	9. The System redirects the User to a password reset form.
	10. The User enters a new password and confirms it.	11. The System validates the new password for compliance with any password requirements (e.g., length, complexity).
		12. The system checks the Password and Confirm Password is matching.
		13. The system encrypts the Password.
		14. The system displays the message of Confirmation.
	15. Check the entered details and click on the confirm button.	

		16. The System saves the updated password for the User's account. 17. The System displays a success message to the User, confirming that their password has been successfully reset. 18. Use-case Ends.
An alternate course of Action	6.1 If the email address is not found in the System's database, the System displays an error message “Invalid email address” and returns to step 5 of the basic course of action to enter again. 6.1 If the required fields of the form weren’t filled up, the system notifies the “required field missing” message and redirects to step 5 of the basic course of action to enter again. 11.1 If the new password is not compliant, the System displays an error message “Incorrect Format” and returns to step 10 of the basic course of action to enter again. 12.1 If the user enters the Confirm Password is not matching Password, then the system notifies the “The Password and Confirm Password Not Matching” and redirects to step 10 of the basic course of action to enter again.	

Table 3. 3 Use Case Narrative for Reset Password

Use Case - Change Password

Use-case Number	UC-04	
Use-Case Name	Change Password	
Priority	High	
Actor/s	Customer, Receptionist, Billing Clerk, Supervisor, Manager, Owner	
Description	This use case indicates how users change their passwords in the system.	
Precondition	The user has an existing account in the system. The user must have logged into the system.	
Post-condition	The user changes their password if the use case was successful and they can log in to the System using the new password.	
Basic Course of Action	**User Action**	**System Response**
	1. The user clicks on the My Profile link.	2. The system loads the My Profile menu.
	3. The User clicks on the "Change Password" link.	4. The system provides a Change Password form to fill up.
	5. The user fills up the following relevant data into the fields in the form and clicks on the Change button. (Current Password, New Password, Confirm Password)	6. The system checks all required fields have been filled up by a user. 7. The System validates the new password for compliance with any password requirements (e.g., length, complexity). 8. The system checks the Password and Confirm Password is matching. 9. The system encrypts the Password. 10. The system displays the message of Confirmation.
	11. Check the entered details and click on the confirm button.	12. The System saves the updated password for the User's account. 13. The System displays a success message. 14. Use-case Ends.

An alternate course of Action	6.1 If the required fields of the form weren't filled up, the system notifies the "required field missing" message and redirects to step 5 of the basic course of action to enter again. 7.1 If the new password is not compliant, the System displays an error message "Incorrect Format" and returns to step 5 of the basic course of action to enter again. 8.1 If the user enters the Confirm Password is not matching Password, then the system notifies the "The Password and Confirm Password Not Matching" and redirects to step 5 of the basic course of action to enter again.

Table 3. 4 Use Case Narrative for Change Password

Use-case View Customer Profile

Use-case Number	UC-05	
Use-Case Name	View Customer Profile	
Priority	Medium	
Actor/s	Customer	
Description	This use case indicates how customers can view their profiles.	
Precondition	The user must have logged into the system. The user has an existing account in the system.	
Post-condition	Customers can retrieve entered their personal and account details in the system.	
Basic Course of Action	**User Action**	**System Response**
	1. The customer clicks on the My Profile link. 3. The customer should click on the view profile link.	2. The system loads the My Profile menu. 4. The system loads the user profile page of the customer. 5. Use-case Ends.
An alternate course of Action	None	

Table 3. 5 Use Case Narrative for View Customer Profile

Use-case Update Customer Profile

Use-case Number	UC-06	
Use-Case Name	Update Customer Profile	
Priority	Medium	
Actor/s	Customer	
Description	This use case indicates how customers can update their profiles.	
Precondition	The user must have logged into the system.	
Post-condition	Customers can update their profile details successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The customer clicks on the My Profile link. 3. The customer clicks on the "Edit Profile" link. 5. Change the required details and click on the "Update" button. 9. The customer checks the updated information and clicks the "Confirm Changes" button.	2. The system loads the My Profile menu. 4. Load the update form with already entered data. 6. Check if all required fields are filled. 7. Validate the updated data according to the format. 8. The system shows a "Confirm Changes" dialog box. 10. The system saves the updated information in the database. 11. The system shows a "successfully Updated." message. 12. Use-case Ends.
An alternate course of Action	6.1 If all required fields of the form weren't filled up, the system notifies the "required field missing" message and redirects to step 5 of the basic course of action to enter again. 7.1 If the values entered are in the incorrect format. The system notifies the "Invalid format" and returns to step 5 of the basic course of action to enter again.	

Table 3. 6 Use Case Narrative for Update Customer Profile

Use-case View Customer Details

Use-case Number	UC-07	
Use-Case Name	View Customer details	
Priority	Medium	
Actor/s	Receptionist, Manager	
Description	This use case allows to users view or display existing customer details in the system.	
Precondition	The user must have logged into the system. The customer's account exists in the system.	
Post-condition	Users can view existing customer details successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should select the "Customer" option from the main menu.	2. The system shows all customer lists on the dashboard of Customer Management. (Reg No, Customer Name, NIC, Contact No, Email Address, Status) with a view button for each row.
	3. The user should click the "View" button for the selected row.	4. The system displays all the details about the selected customer. 5. Use-case Ends.
An alternate course of Action	2.1 If no matched customer is available in the database, go to the Basic course of action of 5.	

Table 3. 7 Use Case Narrative for View Customer Details

Use-case Add New Hall

<table>
<tr><td>Use-case Number</td><td colspan="2">UC-08</td></tr>
<tr><td>Use-Case Name</td><td colspan="2">Add New Hall</td></tr>
<tr><td>Priority</td><td colspan="2">High</td></tr>
<tr><td>Actor/s</td><td colspan="2">Owner</td></tr>
<tr><td>Description</td><td colspan="2">This use case indicates how to owner adds a new hall into the system.</td></tr>
<tr><td>Precondition</td><td colspan="2">The user must have logged into the system.
The user must have the necessary privilege to add a new hall to the system.</td></tr>
<tr><td>Post-condition</td><td colspan="2">The user can add a new hall to the system successfully.</td></tr>
<tr><td rowspan="2">Basic Course of Action</td><td>User Action</td><td>System Response</td></tr>
<tr><td>1. The user should click on the “Hall” option from the main menu.
3. The user should click on the “New Hall” button.
5. The user fills up the following relevant data into the fields in the form and clicks on the submit button.
(Hall name, Minimum guest count, Maximum guest count, Available Features, Hall Image, Status)
9. Check the entered details and click on confirm button.</td><td>2. The system loads the dashboard of hall management.
4. The system loads the add new hall form to fill up.
6. Check if all the required fields are filled.
7. The system validates the hall name is already existing.
8. The system displays the message of confirmation.
10. If confirmed, save details in the database.
11. The system shows a “New Hall Added Successfully!” message.
12. Use-case Ends.</td></tr>
<tr><td>An alternate course of Action</td><td colspan="2">6.1 If all required fields of the form weren’t filled up, the system notifies the “required field missing” message and redirects to step 5 of the basic course of action to enter again.
7.1 If the hall name exists in the System's database, the System displays an error message “This Hall Name is Already Exists” and returns to step 5 of the basic course of action to enter again.</td></tr>
</table>

Table 3. 8 Use Case Narrative for Add new Hall

Use-case View Hall

Use-case Number	UC-09	
Use-Case Name	View Hall	
Priority	Medium	
Actor/s	Receptionist, Supervisor, Manager, Owner	
Description	This use case allows users to view or display hall details in the system.	
Precondition	The user must have logged into the system.	
Post-condition	Users can view all hall details.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should select the "Hall" option from the main menu. 3. The user should click the "View" button for the selected row.	2. The system shows all Hall lists on the dashboard of Hall Management. (Hall name, Minimum guest count, Maximum guest count, Available Features, Hall status, Hall Image) with a view button for each row. 4. The system displays all the details about the selected hall. 5. Use-case Ends.
An alternate course of Action	2.1 If no matched hall is available in the database, go to the Basic course of action of 5.	

Table 3. 9 Use Case Narrative for View Hall

Use-case Update Hall

Use-case Number	UC-10	
Use-Case Name	Update Hall	
Priority	Low	
Actor/s	Owner	
Description	This use case indicates how to owner can update new features for the hall.	
Precondition	The user must have logged into the system. The hall is already entered into the system. The user must have the necessary privilege to update hall details.	
Post-condition	The Owner can update the halls successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user is on the Hall Management dashboard of the system. 3. The user clicks on the "Edit" button of the selected row. 5. Change the required details and click on the "submit" button. 9. The user checks the updated information and clicks the "Confirm Changes" button.	2. The system loads the already entered hall list in the table with the search bar and edit button for each row. 4. Load the update form with already entered data. 6. Check if all required fields are filled. 7. Validate the updated data according to the format. 8. The system shows a "Confirm Changes" dialog box. 10. The system saves the updated information in the database. 11. The system shows a "successfully Updated." message. 12. Use-case Ends.
An alternate course of Action	6.1 If the Required fields are empty, display the message "Required fields are Empty" and return to step 5 of the basic course of action to enter again. 7.1 If the values entered are in the incorrect format, display the message and correct format with an example and return to step 5 of the basic course of action.	

Table 3. 10 Use Case Narrative for Update Hall

Use-case View Hall

Use-case Number	UC-11	
Use-Case Name	View Hall	
Priority	Medium	
Actor/s	Customer	
Description	This use case allows users to view or display hall details on the website.	
Precondition	None	
Post-condition	Users can view all hall details.	
Basic Course of Action	**User Action**	**System Response**
	1. The user navigates the website and clicks on the "Halls" option from the top bar menu.	2. The system shows all Hall Details on the website. (Hall name, Minimum guest count, Maximum guest count, Available Features, Hall status, Hall Image) 3. Use-case Ends.
An alternate course of Action	None	

Table 3. 11 Use Case Narrative for View Hall

Use-case Add New Menu Item

Use-case Number	UC-12	
Use-Case Name	Add New Menu Item	
Priority	High	
Actor/s	Manager	
Description	This use case indicates how to manager adds a new menu item to the system.	
Precondition	The user must have logged into the system. The user must have the necessary privilege to add menu items to the system.	
Post-condition	The user can add a new menu item successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should click on the "Menu Item" option from the main menu.	2. The system loads the dashboard of menu item management.
	3. The user should click on the "New Menu Item" button.	4. The system loads the add new menu item form to fill up.
	5. The user fills up the following relevant data into the fields in the form. (Menu Item Category, Menu Item Name, Menu Item Cost, Profit Ratio, Menu Item Image, Status)	6. The system auto-fills up the following relevant data into the fields in the form (Portion Price)
	7. The user clicks on the submit button.	8. Check if all the required fields are filled. 9. The system validates the menu item name is already existing. 10. Check if the Menu Item Cost and Profit Ratio are in the correct format.
	12. Check the entered details and click on confirm button.	11. The system displays the message of confirmation. 13. If confirmed, save details in the database. 14. The system shows a "New Menu Item Added Successfully" message. 15. Use-case Ends.

An alternate course of Action	8.1 If all required fields of the form weren't filled up, the system notifies the "required field missing" message and redirects to step 5 of the basic course of action to enter again. 9.1 If the Menu item name exists in the System's database, the System displays an error message "This Menu Item Name Already Exists" and returns to step 5 of the basic course of action to enter again. 10.1 If the user fills up an invalid Menu Item Cost and Profit Ratio format. The system notifies the "Invalid Price format" and redirects to step 5 of the basic course of action to enter again.

Table 3. 12 Use Case Narrative for Add New Menu Item

Use-case View Menu Item

Use-case Number	UC-13	
Use-Case Name	View Menu Item	
Priority	Medium	
Actor/s	Receptionist, Manager	
Description	This use case allows to users view or display available menu items in the system.	
Precondition	The user must have logged into the system.	
Post-condition	Users can view all menu item details.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should select the "Menu Item" option from the main menu. 3. The user should click the "View" button for the selected row.	2. The system shows all Menu item lists on the dashboard of Menu Item Management. (Item Category, Menu Item Name, Item Cost, Profit Ratio, Portion Price, Item Image, Status) with a view button for each row. 4. The system displays all the details about the selected menu item. 5. Use-case Ends.
An alternate course of Action	2.1 If no matched menu item is available in the database, go to the Basic course of action of 5.	

Table 3. 13 Use Case Narrative for View Menu Item

Use-case Update Menu Item

Use-case Number	UC-14	
Use-Case Name	Update Menu Item	
Priority	Low	
Actor/s	Manager	
Description	This use case indicates how to manager can update menu item details.	
Precondition	The user must have logged into the system. The menu item is already entered into the system. The user must have the necessary privilege to update menu items.	
Post-condition	The Manager can update menu item details successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user is on the Menu Item Management dashboard of the system. 3. The user clicks on the "Edit" button of the selected row. 5. Change the required details and click on the "submit" button. 9. The user checks the updated information and clicks the "Confirm Changes" button.	2. The system loads the already entered menu item list in the table with the search bar and edit button for each row. 4. Load the update form with already entered data. 6. Check if all required fields are filled. 7. Validate the updated data according to the format. 8. The system shows a "Confirm Changes" dialog box. 10. The system saves the updated information in the database. 11. The system shows a "successfully Updated." message. 12. Use-case Ends.
An alternate course of Action	6.1 If the Required fields are empty, display the message "Required fields are Empty" and return to step 5 of the basic course of action. 7.1 If the values entered are in the incorrect format, display the message and correct format with an example and return to step 5 of the basic course of action.	

Table 3. 14 Use Case Narrative for Update Menu Item

Use-case View Menu Item

Use-case Number	UC-15	
Use-Case Name	View Menu Item	
Priority	Medium	
Actor/s	Customer	
Description	This use case allows to users view or display available menu items in the website.	
Precondition	None	
Post-condition	Users can view all menu item details.	
Basic Course of Action	**User Action**	**System Response**
	1. The user navigates the website and clicks on the "Menu Item" option from the top bar menu.	2. The system shows all Menu items on the website. (Item Category, Menu Item Name, Portion Price, Item Image, Status) 3. Use-case Ends.
An alternate course of Action	None	

Table 3. 15 Use Case Narrative for View Menu Item

Use-case Create New Menu Package

Use-case Number	UC-16	
Use-Case Name	Create New Menu Package	
Priority	High	
Actor/s	Manager	
Description	This use case indicates how to manager is creating a new menu package for reservations.	
Precondition	The user must have logged into the system. The user must have the necessary privilege to add a new menu package to the system.	
Post-condition	The user can create a new menu package successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should click on the "Menu Package" option from the main menu.	2. The system loads the dashboard of menu package management.
	3. The user should click on the "New Menu Package" button.	4. The system loads the add new menu package form.
	5. The user fills up the following relevant data into the fields in the form. (Menu Package Type, Menu Package name, Menu Item List, Plate price, Service Charge, Status)	6. The system auto-fills up the following relevant data into the fields in the form (Last Price) 8. Check if all the required fields are filled.
	7. Clicks on the submit button.	9. The system validates the menu package name is already existing. 10. Check if the Plate Price and Last Price are in the correct format. 11. The system displays the message of confirmation.
	12. Check the entered details and click on confirm button.	13. If confirmed, save details in the database. 14. The system shows a "New Menu Package Added Successfully" message.

		15. Use-case Ends.
An alternate course of Action	8.1 If all required fields of the form weren't filled up, the system notifies the "required field missing" message and redirects to step 5 of the basic course of action to enter again. 9.1 If the Menu package name exists in the System's database, the System displays an error message "This Menu Package Name Already Exists" and returns to step 5 of the basic course of action to enter again. 10.1 If the user fills up an invalid Plate Price and Last Price format. The system notifies the "Invalid format" and redirects to step 5 of the basic course of action to enter again.	

Table 3. 16 Use Case Narrative for Create New Menu Package

Use-case View Menu Package

Use-case Number	UC-17	
Use-Case Name	View Menu Packages	
Priority	Medium	
Actor/s	Receptionist, Manager	
Description	This use case allows users to view or display menu package details in the system.	
Precondition	The user must have logged into the system.	
Post-condition	Users can view all menu package details successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should select the "Menu Package" option from the main menu.	2. The system shows all Menu package lists on the dashboard of Menu Package Management. (Menu Package Type, Menu Package name, Plate Price, Service Charge, Plate Last price, Status) with a view button for each row.
	3. The user should click the "View" button for the selected row.	4. The system displays all the details about the selected menu package. 5. Use-case Ends.
An alternate course of Action	2.1 If no matched menu package is available in the database, go to the Basic course of action of 5.	

Table 3. 17 Use Case Narrative for View Menu Package

Use-case Update Menu Package

Use-case Number	UC-18	
Use-Case Name	Update Menu Package	
Priority	Low	
Actor/s	Manager	
Description	This use case indicates how to manager can update menu package details.	
Precondition	The user must have logged into the system. The menu package is already entered into the system. The user must have the necessary privilege to update menu packages.	
Post-condition	The Manager can update menu package details successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user is on the Menu Package Management dashboard of the system. 3. The user clicks on the "Edit" button of the selected row. 5. Change the required details and click on the "submit" button. 9. The user checks the updated information and clicks the "Confirm Changes" button.	2. The system loads the already entered menu package list in the table with the search bar and edit button for each row. 4. Load the update form with already entered data. 6. Check if all required fields are filled. 7. Validate the updated data according to the format. 8. The system shows a "Confirm Changes" dialog box. 10. The system saves the updated information in the database. 11. The system shows a "successfully Updated." message. 12. Use-case Ends.
An alternate course of Action	6.1 If the Required fields are empty, display the message "Required fields are Empty" and return to step 5 of the basic course of action. 7.1 If the values entered are in the incorrect format, display the message and correct format with an example and return to step 5 of the basic course of action.	

Table 3. 18 Use Case Narrative for Update Menu Package

Use-case View Menu Package

Use-case Number	UC-19	
Use-Case Name	View Menu Package	
Priority	Medium	
Actor/s	Customer	
Description	This use case allows users to view or display menu package details on the website	
Precondition	The user must have logged into the system.	
Post-condition	Users can view all menu package details successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user navigates the website and clicks on the "Menu Package" option from the top bar menu.	2. The system shows all Menu packages on the website. (Menu Package Type, Menu Package name, Item List, Status) 3. Use-case Ends.
An alternate course of Action	None	

Table 3. 19 Use Case Narrative for View Menu Package

Use-case Add New Service

Use-case Number	UC-20	
Use-Case Name	Add New Service	
Priority	High	
Actor/s	Manager	
Description	This use case indicates how to Manager adds a new service for a reservation.	
Precondition	The user must have logged into the system. The user must have the necessary privilege to add a new service to the system.	
Post-condition	The user can add a new service successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should click on the "Service" option from the main menu. 3. The user should click on the "New Service" button. 5. The user fills up the following relevant data into the fields in the form. (Service name, Service Price, Profit Ratio, Status) 7. The user clicks on the submit button. 12. Check the entered details and click on confirm button.	2. The system loads the dashboard of service management. 4. The system loads the add new service form to fill up. 6. The system auto-fills up the following relevant data into the fields in the form. (Service Last price) 8. Check if all the required fields are filled. 9. The system validates the service name is already existing. 10. Check if the Service Price and Service Last price are in the correct format. 11. The system displays the message of confirmation. 13. If confirmed, save details in the database.

		14. The system shows a “New Service Added Successfully” message. 15. Use-case Ends.
An alternate course of Action	8.1 If all required fields of the form weren’t filled up, the system notifies the “required field missing” message and redirects to step 5 of the basic course of action to enter again. 9.1 If the service name exists in the System's database, the System displays an error message “This service Name Already Exists” and returns to step 5 of the basic course of action to enter again. 10.1 If the user fills up an invalid Service Price, Profit Ratio format. The system notifies the “Invalid format” and redirects to step 5 of the basic course of action to enter again.	

Table 3. 20 Use Case Narrative for Add New Service

Use-case View Services

Use-case Number	UC-21	
Use-Case Name	View Services	
Priority	Medium	
Actor/s	Receptionist, Manager	
Description	This use case allows users to view or display all service details in the system.	
Precondition	The user must have logged into the system.	
Post-condition	Users can view all service details successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should select the "Service" option from the main menu. 3. The user should click the "View" button for the selected row.	2. The system shows all service lists on the dashboard of Service Management. (Service name, Service Price, Profit Ratio, Service Last Price, Status) with a view button for each row. 4. The system displays all the details about the selected service. 5. Use-case Ends.
An alternate course of Action	2.1 If no matched Service is available in the database, go to the Basic course of action of 5.	

Table 3. 21 Use Case Narrative for View Services

Use-case Update Service

Use-case Number	UC-22	
Use-Case Name	Update Service	
Priority	Low	
Actor/s	Manager	
Description	This use case indicates how to manager can update service details.	
Precondition	The user must have logged into the system. The service is already entered into the system. The user must have the necessary privilege to update services.	
Post-condition	The Manager can update service details successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user is on the Service Management dashboard of the system. 3. The user should click on the "Edit" button of the selected row. 5. Change the required details and click on the "submit" button. 9. The user checks the updated information and clicks the "Confirm Changes" button.	2. The system loads the already entered service details list in the table with the search bar and edit button for each row. 4. Load the update form with already entered data. 6. Check if all required fields are filled. 7. Validate the updated data according to the format. 8. The system shows a "Confirm Changes" dialog box. 10. The system saves the updated information in the database. 11. The system shows a "successfully Updated." message. 12. Use-case Ends.
An alternate course of Action	6.1 If the Required fields are empty, display the message "Required fields are Empty" and return to step 5 of the basic course of action. 7.1 If the values entered are in the incorrect format, display the message and correct format with an example and return to step 5 of the basic course of action.	

Table 3. 22 Use Case Narrative for Update Service

Use-case View Services

Use-case Number	UC-23	
Use-Case Name	View Services	
Priority	Medium	
Actor/s	Customer	
Description	This use case allows users to view or display all service details on the website.	
Precondition	None	
Post-condition	Users can view all service details successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user navigates the website and clicks on the “Service” option from the top bar menu.	2. The system shows all services on the website. (Service name, Service Last Price, Status) 3. Use-case Ends.
An alternate course of Action	None	

Table 3. 23 Use Case Narrative for View Services

Use-case Check Hall Availability

Use-case Number	UC-24	
Use-Case Name	Check Hall Availability	
Priority	High	
Actor	Customer, Receptionist	
Description	This use case indicates how to check the availability of the hall for a reservation on a specific date and time.	
Precondition	None	
Post-condition	The users can view the specific date, time, and hall availability when the use case is successful.	
Basic Course of Action	**User Action**	**System Response**
	1. The user navigates the website and clicks on the "Check Availability" button. 3. The user inserts data into the fields (Event, Reservation Date, Function Start Time, Function End Time, Guest Count) and clicks on the "Check Availability" button.	2. The system loads the form to check availability. 4. Check if all required fields are filled. 5. Check if the Guest Count is in the correct format. 6. The system displays the available hall list. 7. Use case Ends.
An alternate course of Action	4.1 If all required fields of the form weren't filled up, the system notifies the "required field missing" message and redirects to step 3 of the basic course of action to enter again. 5.1 If the user fills up an invalid format of Guest Count. The system notifies the "Invalid format" and redirects to step 3 of the basic course of action to enter again. 6.1 If the hall is unavailable display the message "Not Available" and return to step 3 of the basic course of action to enter again.	

Table 3. 24 Use Case Narrative for Check Hall Availability

Use-case Make New Reservation

Use-case Number	UC-25	
Use-Case Name	Make New Reservation	
Priority	High	
Actor/s	Customer	
Description	This use case indicates how customers make a reservation based on the availability of the hall for an upcoming event.	
Precondition	The customer must check the availability of the hall. The customer must have logged into the system.	
Post-condition	The customer can make a new reservation through the online reservation system.	
Basic Course of Action	**User Action**	**System Response**
	1. The customer clicks on the "Reservation" Link in the sidebar menu.	2. The system loads the Reservation Menu.
	3. The customer clicks on the "New Reservation" link.	4. The system loads the Check Hall availability form to fill up.
	5. The customer inserts data into the fields (Event, Reservation Date, Function Start Time, Function End Time, Guest Count) and clicks on the "Check Availability" button.	6. Check if all required fields are filled. 7. Validate all inserted data according to the format. 8. The system displays the available hall list with a "Book Now" button for each hall.
	9. The customer should click on the "Book Now" button of the selected hall.	10. The system loads the Make New Reservation form to fill up (Event Details/Menu Package Details/Service Details/Additional Item Details) and auto-fill the event details data according to the customer entered in the check availability form.
	11. The customer checks the Event Details data and if needs to change	

	the guest count and then changes it and clicks on the next button. 16. The customer fills up the following relevant data into the fields in the form. (Menu Package Name) 18. The customer clicks on the next button. 22. The customer Selects Services if needed. 24. The customer clicks on the next button. 27. The customer Selects Additional Menu Items if needed. Then enter the qty of the selected row. 30. The customer clicks on the submit button.	12. Check if all required fields are filled. 13. Check if the Guest Count is in the correct format. 14. Check if the Guest Count is in the Minimum and Maximum Range. 15. The system loads the Menu Package Details form to fill up. 17. The system auto-fill the following relevant data into the fields in the form (Menu Package Price, Total Menu Package Price) and displays the Menu Item List of selected menu Package with Category vise. 19. Check if all required fields are filled. 20. The system loads the Service Details form to fill up. 21. The system displays the Service List with Service Name, Service Price, and Select Check Box for each row. 23. The system auto-fill the following relevant data into the fields in the form. (Total Service Price) 25. The system loads the Additional Item Details form to fill up. 26. The system displays the Additional Item List with Item Category, Menu Item Name, Portion Price, Select Check box, Enter Qty, and Total Price.

	33. Check the entered details and click on confirm button.	28. The system auto-fills the total price of selected each row in the additional item list. 29. The system auto-fills the following relevant data into the fields in the form. (Total Additional Item Price/Total Reservation Price/Tax/Discount/Last Reservation Price) 31. Check if all the required fields are filled. 32. The system displays the message of Confirmation. 34. The system saves the details in the database. 35. The system generates the Reservation number. 36. The system updates the Reservation number in the database. 37. The system shows a "Your Reservation Added Successfully...!" message. 38. Use case ends.
An alternate course of Action	6.1 If all required fields of the form weren't filled up, the system notifies the "required field missing" message and redirects to step 5 of the basic course of action to enter again. 7.1 If the values entered are in the incorrect format, display the "Incorrect Format" message and correct format with an example and return to step 5 of the basic course of action to enter again. 8.1 If the hall is unavailable display the message "Not Available" and return to step 5 of the basic course of action to enter again. 12.1 If all required fields of the form weren't filled up, the system notifies the "required field missing" message and redirects to step 11 of the basic course of action to enter again.	

	13.1 If the Guest Count entered is in the incorrect format, display the "Invalid Format" message and correct format with an example and return to step 11 of the basic course of action to enter again. 14.1 If the Enters Guest Count is less than of the Minimum Guest Count, the system displays the error message and correct format with an example and returns to step 11 of the basic course of action to enter again. 14.2 If the Enters Guest Count is more than of the Maximum Guest Count, the system displays the error message and correct format with an example and returns to step 11 of the basic course of action to enter again. 19.1 If all required fields of the form weren't filled up, the system notifies the "required field missing" message and redirects to step 16 of the basic course of action to enter again. 31.1 If all required fields of the form weren't filled up, the system notifies the "required field missing" message and redirects to step 27 of the basic course of action to enter again.

Table 3. 25 Use Case Narrative for Make New Reservation

Use-case View Reservation History

Use-case Number	UC-26	
Use-Case Name	View Reservation History	
Priority	Medium	
Actor/s	Customer	
Description	This use case indicates how customers can view their previous reservations and get a confirmation of an upcoming event.	
Precondition	The user must have logged into the system. The user has made at least one previous reservation in the system.	
Post-condition	Customers can view their previous reservations and get a confirmation of an upcoming event in the system.	
Basic Course of Action	**User Action**	**System Response**
	1. The customer is on the Dashboard of the customer. 3. The customer should click the "View" button in the selected row.	2. The system displays in the table the booking history of the customer. (Reservation No, Event, Reservation Date, Start Time, End Time, Guest Count, Selected Hall, Status) with a view button for each row. 4. The system displays all the information about the selected reservation. 5. Use-case Ends.
An alternate course of Action	None	

Table 3. 26 Use Case Narrative for View Reservation History

Use-case View Reservation

Use-case Number	UC-27	
Use-Case Name	View Reservation	
Priority	High	
Actor/s	Receptionist, Billing Clerk, Supervisor, Manager	
Description	This use case indicates how users can view all previous reservations and view pending reservations for making arrangements in the hall and updating the event calendar.	
Precondition	The user must have logged into the system.	
Post-condition	The users can view all reservations successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should select the "Reservation" option from the main menu. 3. The customer should click the "View" button in the selected row.	2. The system shows all reservation lists on the dashboard of Reservation. Management. (Reservation No, Customer Name, Event, Reservation Date, Start Time, End Time, Guest Count, Hall, and Status) with a view button for each row. 4. The system displays all the information about the selected reservation. 5. Use-case Ends.
An alternate course of Action	2.1 If no matched reservation is available in the database, go to the Basic course of action of 5.	

Table 3. 27 Use Case Narrative for View Reservation

Use-case Request to Update Menu Package/Service/Additional Item Details

Use-case Number	UC-28	
Use-Case Name	Request to Update Menu Package/Service/Additional Item Details of the Reservation	
Priority	Medium	
Actor/s	Customer	
Description	This use case indicates how customers can request an update on their pending reservation.	
Precondition	The customer must have logged into the system. The customer already makes a reservation in the system. The customer must have the necessary privilege to request the update Menu Package/Service/Additional Item Details of the Reservation.	
Post-condition	Customers can request to update their pending Menu Package /Service/Additional Item Details of the Reservation successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The customer is on the Customer dashboard of the system. 3. The customer clicks on the "Edit" button of the selected row. 5. The customer clicks on the next button. 7. The customer Changes the required details of the following relevant fields in the form. (Menu Package Name)	2. The system loads the reservation history in the table with the search bar and edit button for each row. 4. Load the update form with already entered data (Event Details/Menu Package Details/Service Details/Additional Item Details) and event details fields are already disabled by the system. 6. The system loads the Menu Package Details with already entered data. 8. The system auto-fills the following relevant data into the fields in the form (Menu Package Price, Total Menu Package Price) and displays the Menu Item List of selected menu Package with Category vise. 10. Check if all required fields are filled. 11. The system loads the Service Details with already entered data.

	9. The customer clicks on the next button. 12. The customer Changes Services if selected. 14. The customer clicks on the next button. 16. The customer makes changes to Select Additional Menu Items and Then enters the qty of the selected row. 19. The customer clicks on the submit button. 23. The customer checks the updated information and clicks the "Confirm Changes" button.	13. The system auto-fills the following relevant data into the fields in the form. (Total Service Price) 15. The system loads the Additional Item Details with already entered data. 17. The system auto-fills the total price of selected each row in the additional item list. 18. The system auto-fills the following relevant data into the fields in the form. (Total Additional Item Price/Total Reservation Price/Tax/Discount/Last Reservation Price) 20. Check if all required fields are filled. 21. Validate the updated data according to the format. 22. The system shows a "Confirm Changes" dialog box. 24. The system saves the updated information in the database. 25. The system shows a "Your Update Request Was Sent. Please Wait for Confirmation" message. 26. The system sends an Update Request notification to the Manager. 27. Use-case Ends.
An alternate course of Action	10.1 If the Required fields are empty, display the message "Required fields are Empty" and return to step 7 of the basic course of action to fill again. 20.1 If the Required fields are empty, display the message "Required fields are Empty" and return to step 16 of the basic course of action to fill again. 21.1 If the values entered are in the incorrect format, display the message and correct format with an example and return to step 16 of the basic course of action to fill again.	

Table 3. 28 Use Case Narrative for Request to Update Menu Package/Service/Additional Item

Use-case Request to Update Event Details

Use-case Number	UC-29	
Use-Case Name	Request to Update Event Details of the Reservation.	
Priority	Medium	
Actor/s	Customer	
Description	This use case indicates how customers can request an update on their pending reservation.	
Precondition	The customer must have logged into the system. The customer already makes a reservation in the system. The customer must check the availability of the hall. The customer must have the necessary privilege to request to update the Event Details of the Reservation.	
Post-condition	Customers can request to update their pending Event details of the Reservation successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The customer clicks on the "Reservation" Link in the sidebar menu. 3. The customer clicks on the "Update Event Details" link. 5. The customer inserts data into the fields (Event, Reservation Date, Function Start Time, Function End Time, Guest Count) and clicks on the "Check Availability" button. 9. The customer should select the relevant reservation no and clicks on the "Request Update" button of the selected hall. 11. The customer checks the entered information and clicks the "Confirm" button.	2. The system loads the Reservation Menu. 4. The system loads the Check Hall availability form to fill up. 6. Check if all required fields are filled. 7. Validate all inserted data according to the format. 8. The system displays the available hall list with a Select Combo box of Reservation No and a "Request Update" button for each hall. 10. The system shows a "Confirm" dialog box.

		12. The system saves the entered information in the database. 13. The system shows a “Your Update Request Was Sent. Please Wait for Confirmation” message. 14. The system sends an Update Request notification to the Manager. 15. Use-case Ends.
An alternate course of Action	6.1 If the Required fields are empty, display the message “Required fields are Empty” and return to step 5 of the basic course of action to fill again. 7.1 If the values entered are in the incorrect format, display the message and correct format with an example and return to step 5 of the basic course of action to fill again. 8.1 If the hall is unavailable display the message “Not Available” and return to step 5 of the basic course of action to enter again.	

Table 3. 29 Use Case Narrative for Request to Update Event Details

Use-case Cancel Reservation

Use-case Number	UC-30	
Use-Case Name	Cancel Reservation	
Priority	High	
Actor/s	Customer	
Description	This use case indicates how customers cancel their pending reservations.	
Precondition	The customer must have logged into the system. The customer already makes a reservation in the system.	
Post-condition	The reservation status is changed to cancel. The availability of the hall for the canceled reservation date is updated.	
Basic Course of Action	**User Action**	**System Response**
	1. The customer is on the Customer dashboard of the system. 3. The customer clicks on the "Cancel" button of the selected row. 5. The customer clicks the "Ok" button.	2. The system loads the reservation history in the table with the search bar and cancel button for each row. 4. The system shows a "Are you sure to Cancel" dialog box. 6. The system saves the cancel information in the database. 7. The system shows a "Successfully Cancelled" message. 8. The system sends a Reservation Cancellation notification to the Staff. 9. Use-case Ends.
An alternate course of Action	None	

Table 3. 30 Use Case Narrative for Cancel Reservation

Use-case Make Payment

Use-case Number	UC-31	
Use-Case Name	Make Payment	
Priority	High	
Actor/s	Customer	
Description	This use case indicates how customers make payments for their pending reservations.	
Precondition	The customer must have logged into the system. The customer has an existing reservation in the system.	
Post-condition	The Customer successfully makes a payment for their reservation in the reception hall management system.	
Basic Course of Action	**User Action**	**System Response**
	1. The customer clicks on the Payment link. 3. The customer should click on the make payment link. 5. The customer fills up the following relevant data into the fields in the form. (Reservation No, Payment Category, Payment Method, Paid Date, Payment Slip Image) 7. The customer clicks on the submit button. 11. Check the entered details and click on confirm button.	2. The system loads the payment menu. 4. The system loads the Make Payment form to fill up. 6. The system auto-fills the following relevant data into the fields in the form. (Total Reservation Price, Paid Amount, Balance Amount) 8. Check if all required fields are filled. 9. Validate the entered data according to the format. 10. The system displays the message of Confirmation. 12. If confirmed, save details in the database. 13. The system shows a “Your Payment Added Successfully!” message.

		14. Use-case Ends.
An alternate course of Action	8.1 If all required fields of the form weren't filled up, the system notifies a message "required field missing" and redirects to step 5 of the basic courses of action to fill again. 9.1 If the values entered are in the incorrect format, display the message and correct format with an example and return to step 5 of the basic course of action to fill again.	

Table 3. 31 Use Case Narrative for Make Payment

Use-case View Payment History

Use-case Number	UC-32	
Use-Case Name	View Payment History	
Priority	High	
Actor/s	Customer	
Description	This use case indicates how customers can view their payment history.	
Precondition	The customer must have logged into the system. The customer has made at least one previous payment in the system.	
Post-condition	The Customer can view their previous payments to get a confirmation.	
Basic Course of Action	**User Action**	**System Response**
	1. The customer clicks on the Payment link. 3. The customer should click on the view payment link. 5. The customer should click the "View" button in the selected row.	2. The system loads the payment menu. 4. The system displays in the table the payment history of the customer. (Reservation No, Payment Category, Payment Method, Paid Amount, Balance Amount Paid Date, Payment Slip Image, Status) with a view button for each row. 6. The system displays all the information about the selected row. 7. Use-case Ends.
An alternate course of Action	None	

Table 3. 32 Use Case Narrative for View Payment History

Use-case Update Customer Payment

Use-case Number	UC-33	
Use-Case Name	Update Customer Payment	
Priority	Medium	
Actor/s	Customer	
Description	This use case indicates how the customer can update their pending payments successfully.	
Precondition	The customer must have logged into the system. The customer already makes at least one reservation in the system. The customer already makes at least one payment in the system.	
Post-condition	The user can update payment details successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The customer clicks on the Payment link. 3. The customer should click on the view payment link. 5. The customer should click the "Edit" button in the selected row. 7. Change the required details and click on the "submit" button. 11. The user checks the updated information and clicks the "Confirm Changes" button.	2. The system loads the payment menu. 4. The system displays in the table the payment history of the customer. (Reservation No, Payment Category, Payment Method, Paid Amount, Balance Amount Paid Date, Payment Slip Image, Status) with an edit button for each row. 6. Load the update form with already entered data. 8. Check if all required fields are filled. 9. Validate the updated data according to the format. 10. The system shows a "Confirm Changes" dialog box. 12. The system saves the updated information in the database.

		13. The system shows a "successfully Updated." message. 14. Use-case Ends.
An alternate course of Action	8.1 If the Required fields are empty, display the message "Required fields are Empty" and return to step 7 of the basic course of action to enter again. 9.1 If the values entered are in the incorrect format, display the message and correct format with an example and return to step 7 of the basic course of action.	

Table 3. 33 Use Case Narrative for Update Customer Payment

Use-case View Customer Payment

Use-case Number	UC-34	
Use-Case Name	View Customer Payment	
Priority	High	
Actor/s	Receptionist, Billing Clerk, Manager	
Description	This use case indicates how users can view the payment details of the customers.	
Precondition	The user must have logged into the system.	
Post-condition	The users can view customers' payment details successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should select the "Customer Payment" option from the main menu.	2. The system shows all customer payment lists on the dashboard of Customer Payment Management. (Reservation No, Payment Category, Payment Method, Paid Amount, Balance Amount Paid Date, Payment Slip Image, Status) with a view button for each row.
	3. The customer should click the "View" button in the selected row.	4. The system displays all the information about the selected payment. 5. Use-case Ends.
An alternate course of Action	2.1 If no matched Service is available in the database, go to the Basic course of action of 5.	

Table 3. 34 Use Case Narrative for View Customer Payment

Use-case Verify Received Customer Payments

Use-case Number	UC-35	
Use-Case Name	Verify Received Customer Payments	
Priority	High	
Actor/s	Billing Clerk	
Description	This use case indicates how to verify received payments from the customer.	
Precondition	The user must have logged into the system. The user views the customer payment received notification. The user must be able to view the account associated with the hotel setup to receive customer payments.	
Post-condition	The user verifies a customer payment into the system successfully. The customer's outstanding balance is updated accordingly. Payment Receipts are provided to the customer. Send payment verification notifications to the manager and customer.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should select the "Customer Payment" option from the main menu.	2. The system shows all customer payment lists on the dashboard of Customer Payment Management. (Reservation No, Payment Category, Payment Method, Paid Amount, Balance Amount Paid Date, Payment Slip Image, Status) with an edit button for each row.
	3. The user clicks on the "Edit" button for the selected row. 5. The user checks the entered customer details with the bank account details of the hotel for payment verification.	4. The system displays in the form the already entered payment details of the customer.
	6. After the verification user is Change the Status to "Verified" in the form and clicks on the submit button.	7. Check if all required fields are filled. 8. The system displays the message of Confirmation.

	9. Check the entered details and click on confirm button.	10. If confirmed, save details in the database. 11. The system generates the Bill number. 12. The system updates the Bill number in the database. 13. The system updates the customer's outstanding balance accordingly. 14. The system shows a “Payment Verification was Successful...!” message. 15. The system enables to download the payment receipt of the customer. 16. The system sends payment verification notifications to the manager and customer. 17. Use-case Ends.
An alternate course of Action	6.1 If the payment verification was unsuccessful then the user changes the status to “Invalid” in the form and the system sends payment invalid notifications to the customer to correct again their payment. 7.1 If all required fields of the form weren’t filled up, the system notifies the “required field missing” message and redirects to step 6 of the basic course of action to fill it again.	

Table 3. 35 Use Case Narrative for Verify Received Customer Payments

Use-case Confirm/Reject Reservation

Use-case Number	UC-36	
Use-Case Name	Confirm/Reject Reservation	
Priority	High	
Actor/s	Manager	
Description	This use case indicates how to manager confirms or rejects pending reservations of the customers.	
Precondition	The user must have logged into the system. The user views the customer payment verification notification.	
Post-condition	The reservation status is changed. Send reservation confirmation or reject notifications to the customer and supervisor. Update the event calendar.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should click the "Reservation" option from the main menu.	2. The system shows all reservation lists on the dashboard of Reservation Management. (Reservation No, Customer Name, Event, Reservation Date, Start Time, End Time, Guest Count, Hall, and Status) with the edit button for each row.
	3. The user clicks on the "Edit" button of the selected row.	4. The system displays in the form the already entered reservation details of the customer.
	5. The user checks the already entered data and changed the reservation status and clicked the submit button.	6. Check if all required fields are filled. 7. The system displays the message of Confirmation.
	8. The user checks the details and clicks the "Confirm" button.	9. If confirmed, the system updates the details in the database. 10. The system shows a Reservation Confirmation or Reject message.

		11. The system sends a notification to the customer and supervisor. 12. Update the event calendar. 13. Use-case Ends.
An alternate course of Action	6.1 If all required fields of the form weren't filled up, the system notifies the "required field missing" message and redirects to step 5 of the basic course of action to fill it again.	

Table 3. 36 Use Case Narrative for Confirm/Reject Reservation

Use-case View Reservation Update Request

Use-case Number	UC-37	
Use-Case Name	View Reservation Update Request	
Priority	High	
Actor/s	Manager	
Description	This use case indicates how to manager can view reservation update requests for pending reservations.	
Precondition	The user must have logged into the system.	
Post-condition	The users can view all reservation update requests successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should select the "Reservation" option from the main menu. 3. The user should click the "View Update Requests" button. 5. The user should click the "View" button in the selected request.	2. The system redirects the dashboard of Reservation. 4. The system displays all reservation update request lists (Reservation No, Customer Name, Event Type, Reservation Date, Request Date, and Status) with a view button for each row. 6. The system displays all the information about the selected reservation request. 7. Use-case Ends.
An alternate course of Action	None	

Table 3. 37 Use Case Narrative for View Reservation Update Request

Use-case Approve/Reject Reservation Update Request

Use-case Number	UC-38	
Use-Case Name	Approve or Reject Reservation Update Request	
Priority	High	
Actor/s	Manager	
Description	This use case indicates how to manager can be approved or rejected reservation update requests for pending customer reservations.	
Precondition	The user must have logged into the system. The system has at least one pending update request. The user must refer to the company's updated policy.	
Post-condition	The user can approve or reject reservation update requests successfully. The reservation update request status is changed. The system sends a notification to the customer that the update request has been approved or rejected.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should click the "Reservation" option from the main menu. 3. The user should click the "View Update Requests" button. 5. The user should click the "View" button in the selected update request. 7. The user checks the already entered data with the company update policy and then changed the reservation update status and clicks on the submit button. 10. The user checks the details and clicks the "Confirm" button.	2. The system redirects the dashboard of Reservation Management. 4. The system displays all reservation update request lists (Reservation No, Event Type, Reservation Date, Request Date, and Status) with a view button. 6. The system displays all the details about the selected reservation update request. 8. Check if all required fields are filled. 9. The system displays the message of Confirmation. 11. If confirmed, the system updates the details in the database.

		12. The system shows a Reservation Update Request Confirmation or Reject message. 13. The system sends a notification to the customer that the update request has been approved or rejected. 14. Use-case Ends.
An alternate course of Action	8.1 If all required fields of the form weren't filled up, the system notifies the "required field missing" message and redirects to step 7 of the basic course of action to fill it again.	

Table 3. 38 Use Case Narrative for Approve/Reject Reservation Update Request

Use-case Request Refund Payment

Use-case Number	UC-39	
Use-Case Name	Request Refund Payment	
Priority	Medium	
Actor/s	Customer	
Description	This use case indicates how customers can request refund payment for their canceled reservation.	
Precondition	The user must have logged into the system. The user must have canceled the reservation before requesting a refund payment. The user must refer to the company's refund payment policy.	
Post-condition	Customers can request to issue their refund payment for canceled reservations.	
Basic Course of Action	**User Action**	**System Response**
	1. The customer should select the "Refund Payment Request" option from their account page.	2. The system displays the customer's canceled reservation list in the table (Reservation No, Event Type, Reservation Date, Reservation Start Time, Reservation End Time, Hall, and Status) with a "Refund Request" button.
	3. The customer clicks on the "Refund Request" button of the selected event. 5. The customer enters their request (Reservation No, Refund Amount, Refund Request, Request Date) and clicks on the "Submit" button. 9. The customer checks the entered details and clicks the "Confirm" button.	4. The system prompts the customer to provide enter their refund payment request. 6. Check if all required fields are filled. 7. Validate the updated data according to the format. 8. The system shows a "Confirm" dialog box. 10. The system saves the details in the database.

		11. The system shows a “Your Refund Payment Request Was Sent” message. 12. The system sends a Refund Payment Request notification to the Manager. 13. Use-case Ends.
An alternate course of Action	6.1 If the Required fields are empty, display the message “Required fields are Empty” and return to step 5 of the basic course of action to fill again. 7.1 If the values entered are in the incorrect format, display the message and correct format with an example and return to step 5 of the basic course of action to fill again.	

Table 3. 39 Use Case Narrative for Request Refund Payment

Use-case View Refund Payment Request

Use-case Number	UC-40	
Use-Case Name	View Refund Payment Request	
Priority	High	
Actor/s	Manager	
Description	This use case indicates how to manager can view refund payment requests for canceled customer reservations.	
Precondition	The user must have logged into the system.	
Post-condition	The users can view all refund payment requests successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should click the "Refund Payment" option from the main menu. 3. The user should click the "View" button in the selected request.	2. The system shows all refund request lists on the dashboard of Refund Payment Management (Reservation No, Customer Name, Refund Amount, Refund Request, Request Date, Status) with a view button. 4. The system displays all the information about the selected refund payment request. 5. Use-case Ends.
An alternate course of Action	None	

Table 3. 40 Use Case Narrative for View Refund Payment Request

Use-case Approve/Reject Refund Payment Request

Use-case Number	UC-41	
Use-Case Name	Approve or Reject Refund Payment Request	
Priority	High	
Actor/s	Manager	
Description	This use case indicates how to manager can be approved or rejected refund payment requests for canceled customer reservations.	
Precondition	The user must have logged into the system. The system has at least one request for refund payment. The user must refer to the company's refund policy.	
Post-condition	The user can approve or reject refund payment requests successfully. The Refund Payments status is changed. The system sends a notification to the customer that the request has been approved or rejected. The system sends an Approved Refund Payment Request notification to the Billing Clerk.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should click the "Refund Payment" option from the main menu. 3. The user should click the "View Refund Request" button. 5. The user should click the "View" button in the selected refund request. 7. The user checks the already entered data with the company refund policy and then changed the refund payment status and clicks on the submit button. 10. The user checks the details and clicks the "Confirm" button.	2. The system redirects the dashboard of Refund Payment Management. 4. The system displays all refund payment request lists (Reservation No, Customer Name, Refund Amount, Refund Request, Request Date, and Status) with a view button. 6. The system displays all the details about the selected reservation refund request. 8. Check if all required fields are filled. 9. The system displays the message of Confirmation. 11. If confirmed, the system saves the details in the database.

		12. The system shows an “Approved” or “Reject” message. 13. The system sends a notification to the customer that the request has been approved or rejected. 14. The system sends an Approved Refund Payment Request notification to the Billing Clerk. 15. Use-case Ends.
An alternate course of Action	8.1 If all required fields of the form weren’t filled up, the system notifies the “required field missing” message and redirects to step 7 of the basic course of action to fill it again.	

Table 3. 41 Approve/Reject Refund Payment Request

Use-case View Approved Refund Payments

Use-case Number	UC-42	
Use-Case Name	View Approved Refund Payments	
Priority	High	
Actor/s	Billing Clerk	
Description	This use case indicates how to billing clerk can view approved refund payment requests to process refunds.	
Precondition	The user must have logged into the system.	
Post-condition	The user can view all approved refund payment requests successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should select the "Refund Payment" option from the main menu. 3. The user should click the "View" button in the selected request.	2. The system displays all approved refund payment request lists on a dashboard of Refund Payment Management. (Reservation No, Customer Name, Refund Amount, Refund Request, Request Date, and Status) with a view button for each row. 4. The system displays all the information about the selected approved refund payment request. 5. Use-case Ends.
An alternate course of Action	None	

Table 3. 42 Use Case Narrative for View Approved Refund Payments

Use-case Issue Refund Payment

Use-case Number	UC-43	
Use-Case Name	Issue Refund Payment	
Priority	High	
Actor/s	Billing Clerk	
Description	This use case indicates how to billing clerk issues refund payments for approved refund payment requests.	
Precondition	The user must have logged into the system. There are approved refund requests in the system. The user must have the necessary privilege to process refunds.	
Post-condition	The user can issue a refund payment successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user is on the Refund Payment Management dashboard of the system. 3. The user should click the "Issue Refund" button in the selected request. 5. The user initiates the refund payment using the appropriate payment method, such as a credit card refund, bank transfer, or issuing check. 6. The user changes the status is "Released" in the refund request form and clicks on the submit button. 9. The user checks the details and clicks the "Confirm" button.	2. The system displays all approved refund payment request lists with an issue refund button for each row. 4. The system displays all the details about the selected reservation refund request. 7. Check if all required fields are filled. 8. The system displays the message of Confirmation. 10. The system updates the customer payment status "Refunded" in the database. 11. The system shows a "Successfully Refunded" message.

		12. The system generates a refund receipt. 13. The system sends a notification to the customer that the refund payment is issued. 14. Use-case Ends.
An alternate course of Action	7.1 If all required fields of the form weren't filled up, the system notifies the "required field missing" message and redirects to step 6 of the basic course of action to fill it again.	

Table 3. 43 Use Case Narrative for Issue Refund Payment

Use-case Submit Reviews

Use-case Number	UC-44	
Use-Case Name	Submit Review	
Priority	Low	
Actor/s	Customer	
Description	This use case indicates how customers submit reviews to provide feedback and rate their experience of their previous reservations.	
Precondition	The user must have logged into the system. The user has recently attended an event at a hall.	
Post-condition	None.	
Basic Course of Action	**User Action**	**System Response**
	1. The customer should select the "Submit Review" option from their account page.	2. The system displays the customer's previous reservation list in the table (Reservation No, Event Type, Reservation Date, Reservation Time, Hall, and Status) with a Review button for each row.
	3. The customer should click on the "Review" button of the selected event.	4. The system provides the customer to enter their review and rating.
	5. The customer enters their review and selects a rating scale of 1-5 and clicks on the submit button.	6. The system displays the message of Confirmation.
	7. The customer checks the entered details and clicks on confirm button.	8. The system saves the details in the database.
		9. The system shows a "Thanks for Rating" message.
		10. Use-case Ends.
An alternate course of Action	None	

Table 3. 44 Use Case Narrative for Submit Reviews

Use-case View Customer Reviews

Use-case Number	UC-45	
Use-Case Name	View Customer reviews	
Priority	Medium	
Actor/s	Manager/ Supervisor	
Description	This use case indicates how users can view customer reviews in the system.	
Precondition	The user must have logged into the system. There are customer reviews in the system.	
Post-condition	The users can view customer reviews in the system to evaluate the performance of the venue and get feedback from customers.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should click the "Customer Review" option from the main menu. 3. The user should click the "View" button in the selected row.	2. The system displays all customer review lists (Customer Name, Reservation Details, and Rating) with a view button for each row. 4. The system displays all the details about the selected review of the customer. (Customer Review, Specific Comments) 5. Use-case Ends.
An alternate course of Action	None	

Table 3. 45 Use Case Narrative for View Customer Reviews

Use-case Reply to Customer Reviews

Use-case Number	UC-46	
Use-Case Name	Reply to Customer Reviews	
Priority	Low	
Actor/s	Supervisor	
Description	This use case indicates how to supervisor replies to reviews of the customer.	
Precondition	The user must have logged into the system. The Customer has already submitted reviews.	
Post-condition	The supervisor can reply to customer reviews successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should select the "Customer Review" option in the main menu. 3. The user should click the "View" button in the selected row. 5. The user replies to the review and clicks the "Reply" button.	2. The system displays all customer review lists (Customer Name, Reservation Details, and Rating) with a view button for each row. 4. The system displays all the details about the selected review of the customer. (Customer Review, Specific Comments). 6. The system saves the details in the database. 7. Use-case Ends.
An alternate course of Action	None	

Table 3. 46 Use Case Narrative for Reply to Customer Reviews

Use-case Hall Arrangement

Use-case Number	UC-47	
Use-Case Name	Hall Arrangement	
Priority	High	
Actor/s	Supervisor	
Description	This use case allows the supervisor to make arrangements for the hall before upcoming events.	
Precondition	The user must have logged into the system. Received the reservation confirmation notification.	
Post-condition	The supervisor makes arrangements for the hall successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should select the "Hall Arrangement" option from the main menu. 3. The user selects the reservation needed to make arrangements. 5. The user views the reservation details and arranges the hall. (Decorations/Setting up Tables and Chairs)	2. The system shows a calendar view of upcoming events. 4. The system displays the selected reservation details. 6. Use-case Ends.
An alternate course of Action	None	

Table 3. 47 Use Case Narrative for Hall Arrangement

Use-case View Reports

Use-case Number	UC-48	
Use-Case Name	View Reports	
Priority	High	
Actor/s	Owner, Manager, Billing Clerk	
Description	This use case allows the users to view or display various types of reports in the system.	
Precondition	The user must have logged into the system. The system has relevant data such as reservations, payments, and other information.	
Post-condition	The users can view various types of reports successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should select the "View Report" option from the main menu. 3. The user selects the report needs to view. 5. The user fills up the relevant data and clicks on the generate button.	2. The system shows a list of available reports according to the user privilege. (Reservation Report / Payment Report / Employee Report) 4. The system displays the user to select the date range and other criteria for the report. 6. The system generates the report. 7. Use-case Ends.
An alternate course of Action	None	

Table 3. 48 Use Case Narrative for View Reports

Use-case Add New Employee

Use-case Number	UC-49	
Use-Case Name	Add New Employee	
Priority	High	
Actor	Owner	
Description	This use case indicates how to add new employees into the system.	
Precondition	The user must have logged into the system.	
Post-condition	Display a Success message after the successful registration.	
Basic Course of Action	**User Action**	**System Response**
	1. The user clicks on the "Employee" option from the main menu. 3. The user clicks on the Add New Employee button. 5. The user fills up the following relevant data into the fields in the form and clicks on the submit button. (Title, First Name, Last Name, Calling Name, Full Name, Designation, Employee Photo, NIC, Gender, Date of Birth, Civil Status, Contact No-Mobile, Contact No-Land, Email, Assignment Date, House No, Street Name, City, District, Description, Status) 10. Check the entered details and click on the confirm button.	2. The system loads the Employee Management Dashboard of the system. 4. The system provides an add new employee form to fill up. 6. The system checks all required fields have been filled up by a user. 7. The system validates the Contact number, NIC, and Email format. 8. The system validates that the NIC, Email already exists. 9. The system displays the message of Confirmation. 11. The system saves the details in the database. 12. The system generates the Employee Registration number. 13. The system updates the Employee Registration number in the database.

		14. The system shows a "successfully Registered." message. 15. Use case ends.
An alternate course of Action	6.1 If all required fields of the form weren't filled up, the system notifies a message "required field missing" and redirects to step 5 of the basic course of action to enter again. 7.1 If the user fills up an invalid format of Contact No, NIC, Email. The system notifies the "Invalid format" and redirects to step 5 of the basic course of action to enter again. 8.1 If the user entered the existing NIC or Email. The system shows the "You entered existing NIC or Email" message and redirects to step 5 of the basic course of action to enter again or proceed with login.	

Table 3. 49 Use Case Narrative for Add New Employee

Use-case View Employee Details

Use-case Number	UC-50	
Use-Case Name	View Employee details	
Priority	Medium	
Actor/s	Owner	
Description	This use case allows to users view or display existing employee details in the system.	
Precondition	The user must have logged into the system.	
Post-condition	Users can view existing employee details successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should select the "Employee" option from the main menu. 3. The user should click the "View" button in the selected row.	2. The system shows all employee lists in the system. (Reg No, Employee Name, NIC, Contact No, Designation, Assignment Date, Status) with a view button for each row. 4. The system displays all the information about the selected employee. 5. Use-case Ends.
An alternate course of Action	2.1 If no matched employee is available in the database, go to the Basic course of action of 5.	

Table 3. 50 Use Case Narrative for View Employee Details

Use-case Update Employee

Use-case Number	UC-51	
Use-Case Name	Update Employee	
Priority	Low	
Actor/s	Owner	
Description	This use case indicates how to owner can update employee details.	
Precondition	The user must have logged into the system. The system must have at least one employee in the system.	
Post-condition	The owner can update employee details successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should select the "Employee" option from the main menu.	2. The system shows all employee details (Reg No, Employee Name, NIC, Contact No, Designation, Assignment Date, Status) with an edit button for each row.
	3. The user clicks on the "Edit" button.	4. Load the update form with already entered data.
	5. Change the required details and click on the "Save Changes" button.	6. Check if all required fields are filled. 7. Validate the updated data according to the format. 8. The system shows a "Confirm Changes" dialog box.
	9. The customer checks the updated information and clicks the "Confirm Changes" button.	10. The system saves the updated information in the database. 11. The system shows a "successfully Updated." message. 12. Use-case Ends.
An alternate course of Action	6.1 If the Required fields are empty, display the message "Required fields are Empty" and return to step 5 of the basic course of action. 7.1 If the values entered are in the incorrect format, display the message and correct format with an example and return to step 5 of the basic course of action.	

Table 3. 51 Use Case Narrative for Update Employee

Use-case Add New User

Use-case Number	UC-52	
Use-Case Name	Add New User	
Priority	High	
Actor	Owner	
Description	This use case indicates how to create a new user account in the system.	
Precondition	The user must have logged into the system. The system has existing employees in the system.	
Post-condition	Display a Success message after the successfully created account.	
Basic Course of Action	**User Action**	**System Response**
	1. The user clicks on the "User" option from the main menu. 3. The user clicks the new user button. 5. The user fills up the following relevant data into the fields in the form. (Employee Registration No, User Name, Password, Confirm Password, User Role, Status) 7. The user clicks on the submit button. 14. Check the entered details and click on the confirm button.	2. The system loads the User Management Dashboard of the system. 4. The system provides a add new user form to fill up. 6. The system auto-fills the following relevent data into the fields in the form. (Employee Full Name, Designation) 8. The system checks all required fields have been filled up by a user. 9. The system validates that the User Name already exists. 10. The system validates the Password format. 11. The system checks the Password and Confirm Password is matching. 12. The system encrypts the Password. 13. The system displays the message of Confirmation. 15. The system saves the details in the database.

		16. The system sends notifications to users including their login details. 17. The system shows a “successfully Registered.” message. 18. Use case ends.
An alternate course of Action	8.1 If all required fields of the form weren’t filled up, the system notifies a message “required field missing” and redirects to step 5 of the basic course of action to enter again. 9.1 If the user entered the existing User Name. The system shows the “You entered existing User Name” message and redirects to step 5 of the basic course of action to enter again or proceed with login. 10.1 If the user fills up an invalid format password. The system notifies the “Invalid password format” and redirects to step 5 of the basic course of action to enter again. 11.1 If the user enters the Confirm Password is not matching Password, then the system notifies the “The Password and Confirm Password Not Matching” and redirects to step 5 of the basic course of action to enter again.	

Table 3. 52 Use Case Narrative for Add New User

Use-case View User Account

Use-case Number	UC-53	
Use-Case Name	View User Account	
Priority	Medium	
Actor/s	Owner	
Description	This use case allows the owner to view or display existing user account details in the system.	
Precondition	The user must have logged into the system.	
Post-condition	The Owner can view existing user account details successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should select the "User" option from the main menu. 3. The user should click the "View" button in the selected row.	2. The system shows all user account details. (Employee Registration No, Employee Full Name, Designation, User Role, Status) with a view button for each row. 4. The system displays all the information about the selected user. 5. Use-case Ends.
An alternate course of Action	2.1 If no matched user account is available in the database, go to the Basic course of action of 5.	

Table 3. 53 Use Case Narrative for View User Account

Use-case Update User Account

Use-case Number	UC-54	
Use-Case Name	Update User Account	
Priority	Low	
Actor/s	Owner	
Description	This use case indicates how to owner can update user account details.	
Precondition	The user must have logged into the system. The system must have at least one user in the system.	
Post-condition	The owner can update user account details successfully.	
Basic Course of Action	**User Action**	**System Response**
	1. The user should select the "User" option from the main menu.	2. The system shows all user account details. (Employee Registration No, Employee Full Name, Designation, User Role, Status) with an edit button for each row.
	3. The user clicks on the "Edit" button.	4. Load the update form with already entered data.
	5. Change the required details and click on the "Save Changes" button.	6. Check if all required fields are filled. 7. Validate the updated data according to the format. 8. The system checks the Password and Confirm Password is matching. 9. The system encrypts the Password.
	11. The customer checks the updated information and clicks the "Confirm Changes" button.	10. The system shows a "Confirm Changes" dialog box. 12. The system saves the updated information in the database. 13. The system shows a "successfully Updated." message. 14. Use-case Ends.
An alternate course of Action	6.1 If the Required fields are empty, display the message "Required fields are Empty" and return to step 5 of the basic course of action.	

	7.1 If the values entered are in the incorrect format, display the message and correct format with an example and return to step 5 of the basic course of action. 8.1 If the user enters the Confirm Password is not matching Password, then the system notifies the “The Password and Confirm Password Not Matching” and redirects to step 5 of the basic course of action to enter again.

Table 3. 54 Use Case Narrative for Update User Account

3.1.5 Activity Diagrams

The flow of actions or activities in a system or process is modeled by an activity diagram, which is a type of UML diagram. The sequence of steps, decision points, and interactions between different components are visually represented by it. All the Activity Diagrams are shown below.

Activity Diagram of Register as a Customer

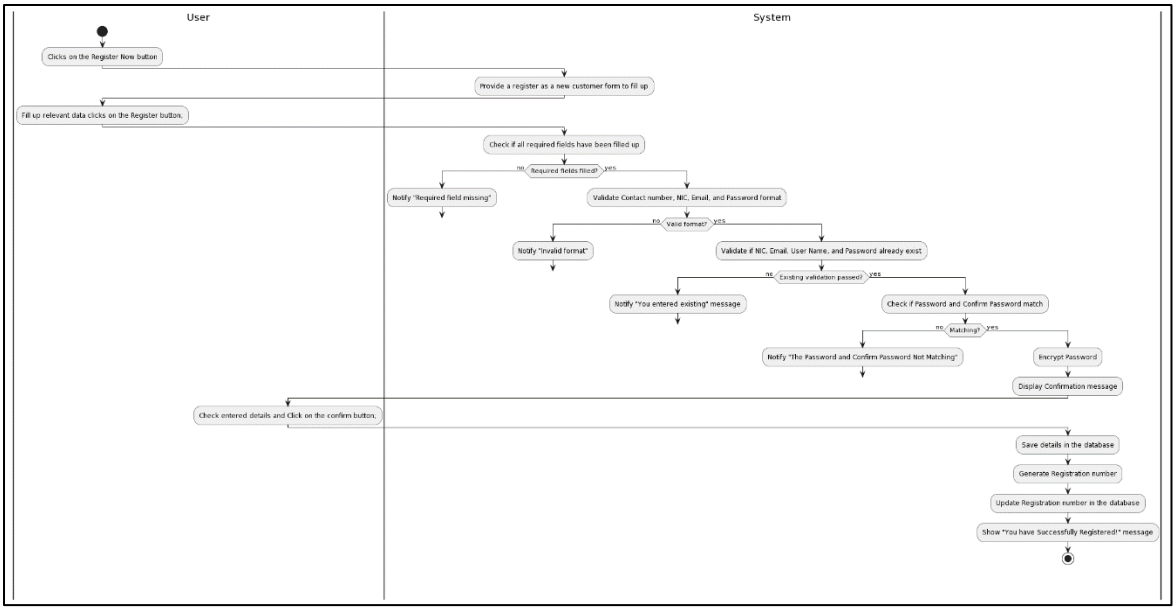


Figure 3. 7 Activity Diagram of Register as Customer

Activity Diagram of Login

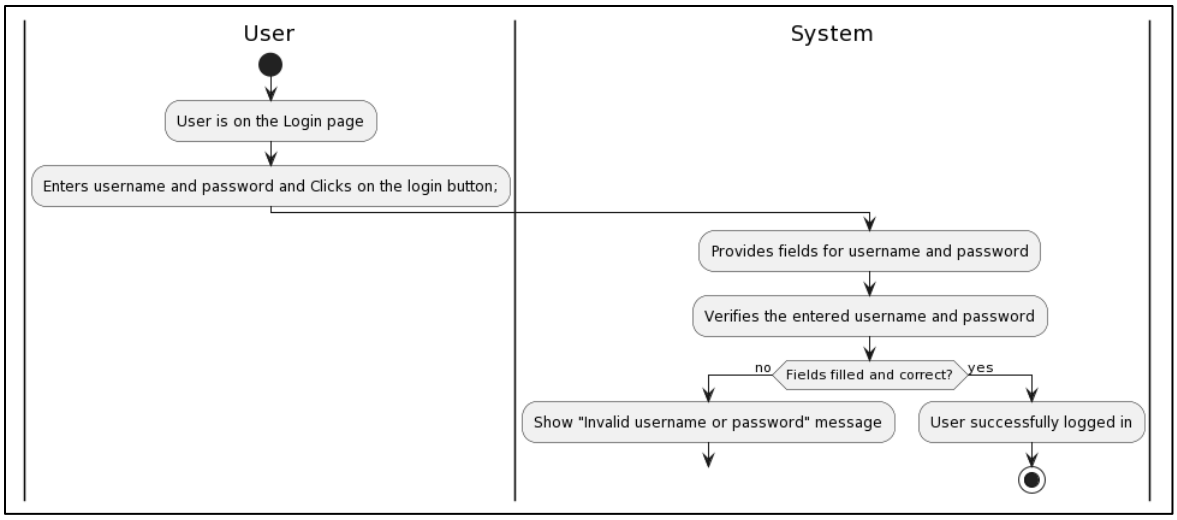


Figure 3. 8 Activity Diagram of Login

Activity Diagram of Reset Password

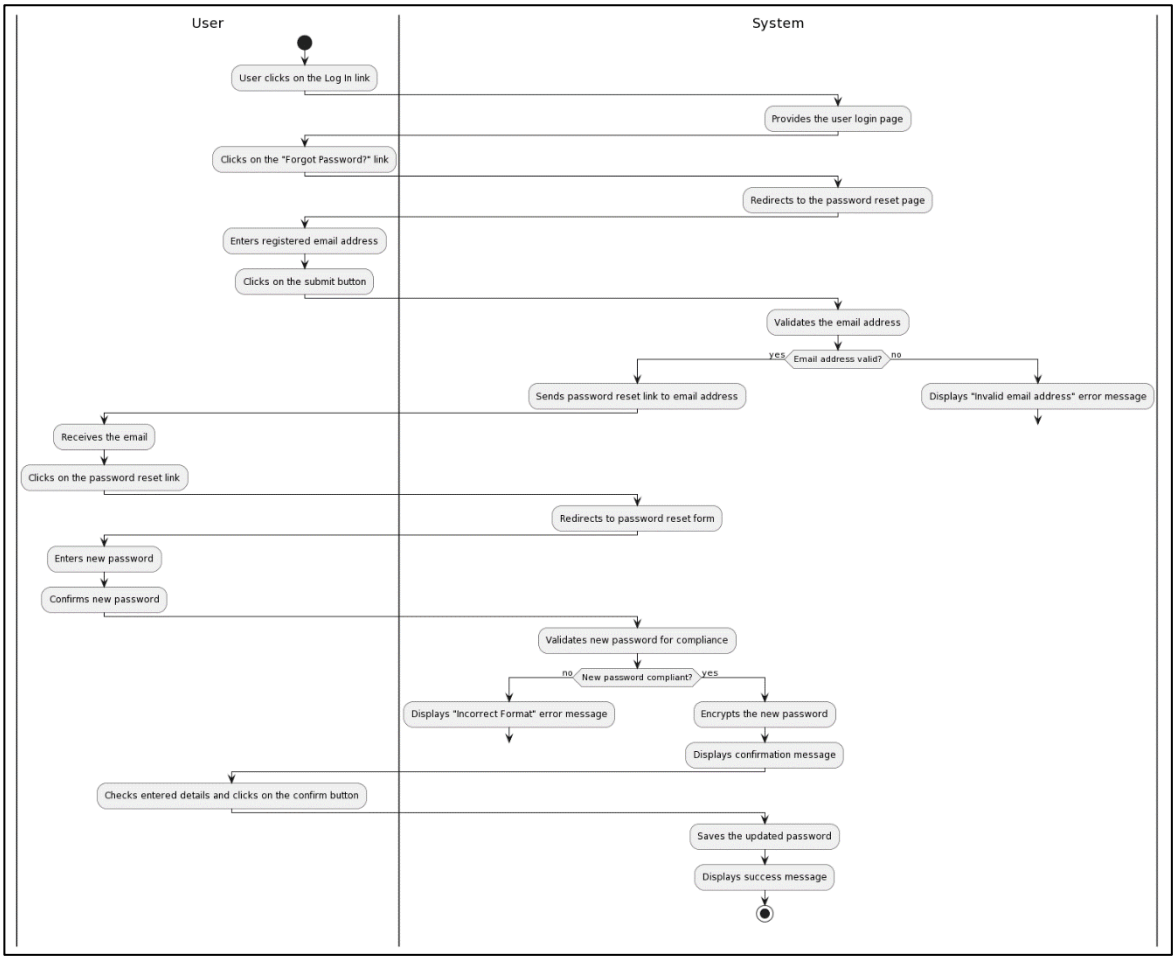


Figure 3. 9 Activity Diagram of Reset Password

Activity Diagram of Change Password

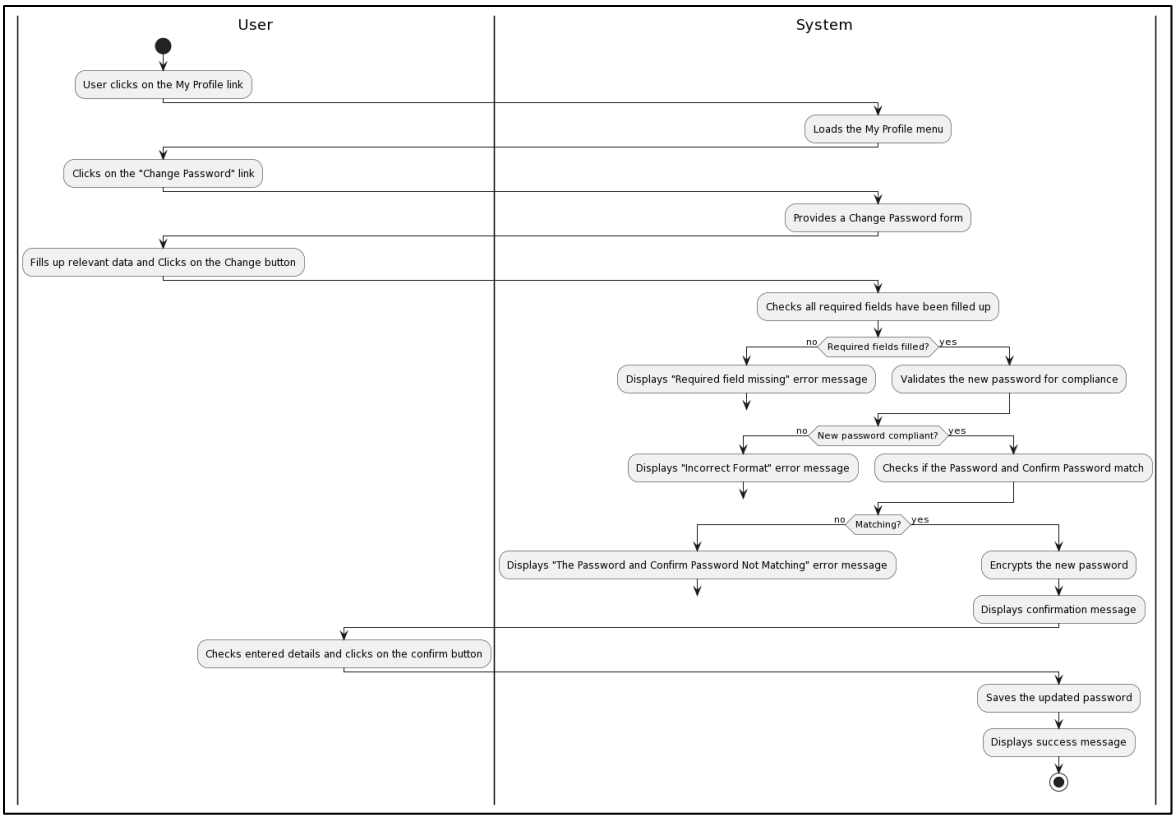


Figure 3. 10 Activity Diagram of Change Password

Activity Diagram of View Customer Profile

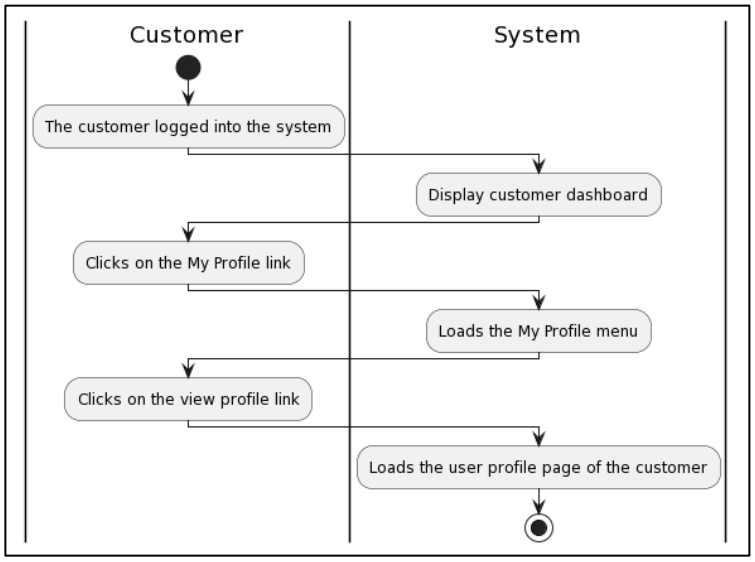


Figure 3. 11 Activity Diagram of View Customer Profile

Activity Diagram of Update Customer Profile

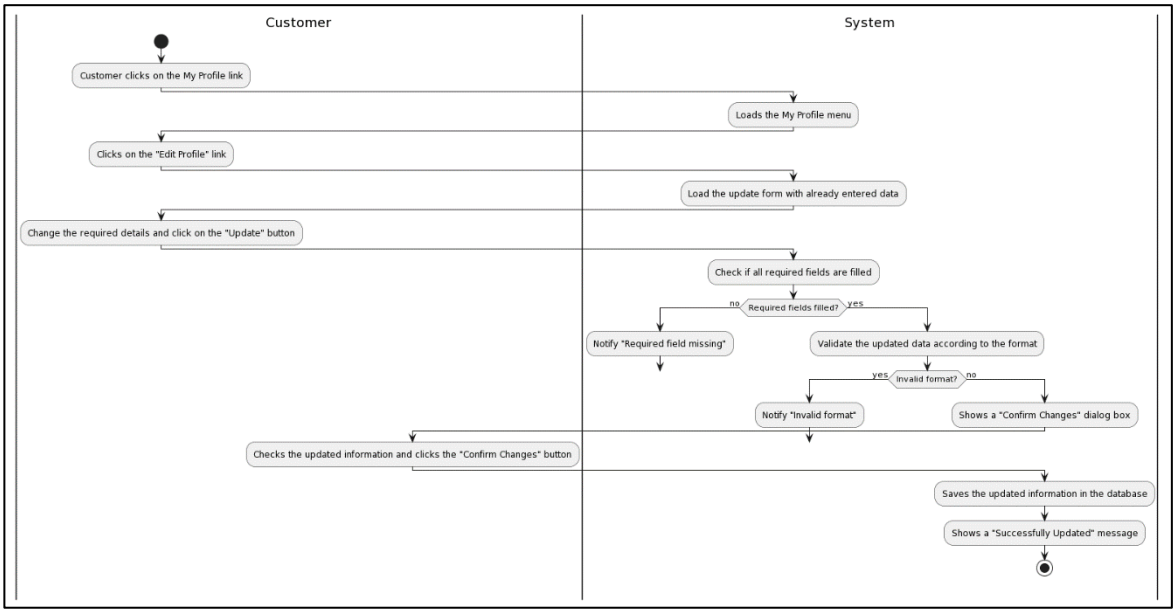


Figure 3. 12 Activity Diagram of Update Customer Profile

Activity Diagram of View Customer Details

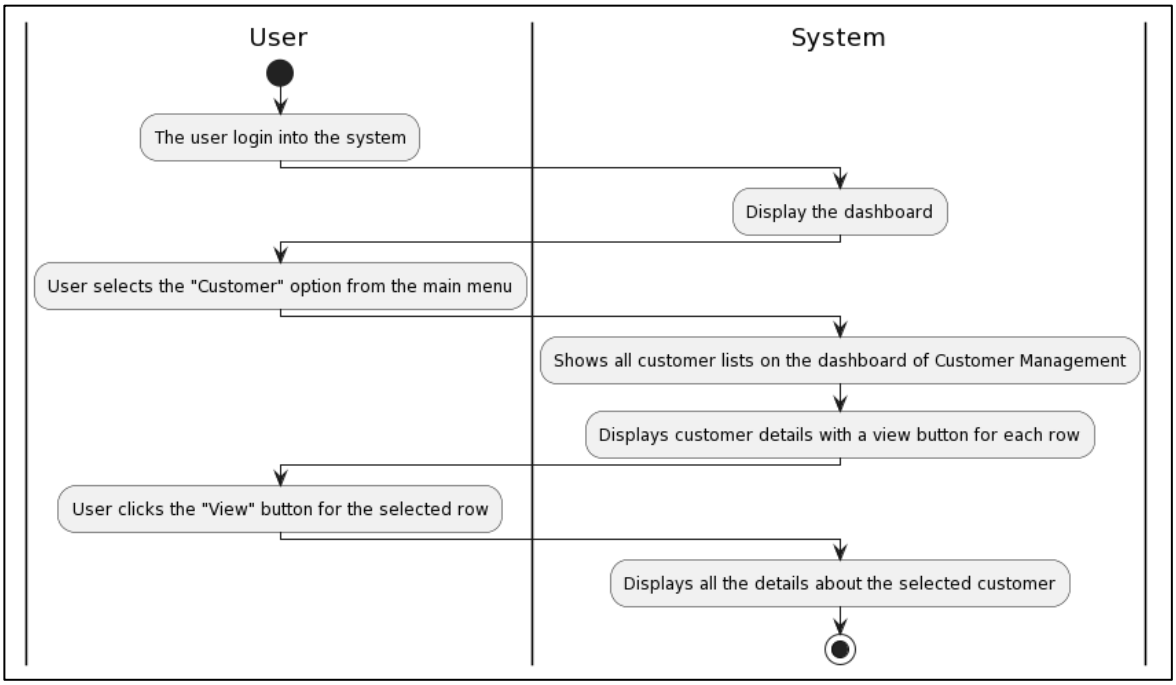


Figure 3. 13 Activity Diagram of View Customer Details

Activity Diagram of Add New Hall

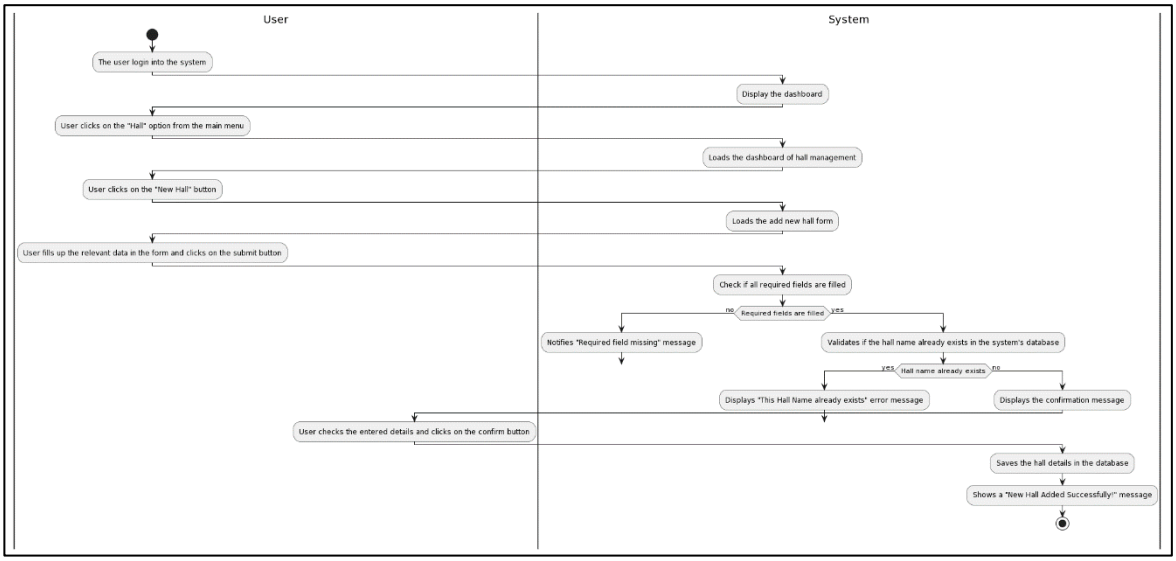


Figure 3. 14 Activity Diagram of Add New Hall

Activity Diagram of View Hall

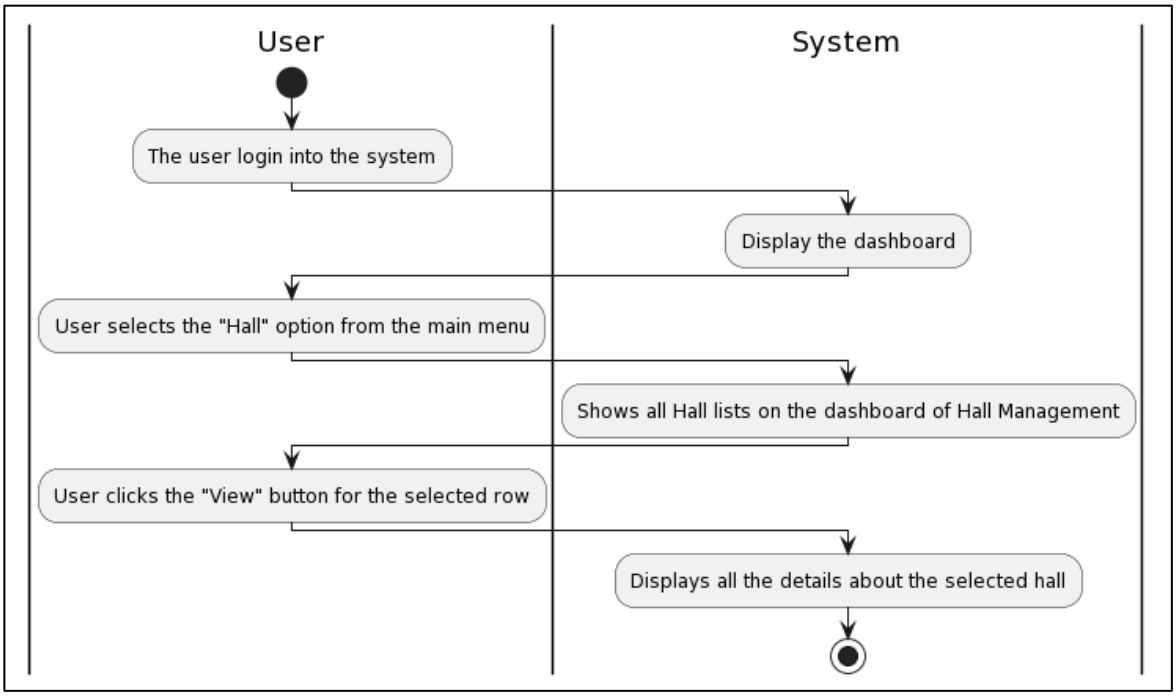


Figure 3. 15 Activity Diagram of View Hall

Activity Diagram of Update Hall

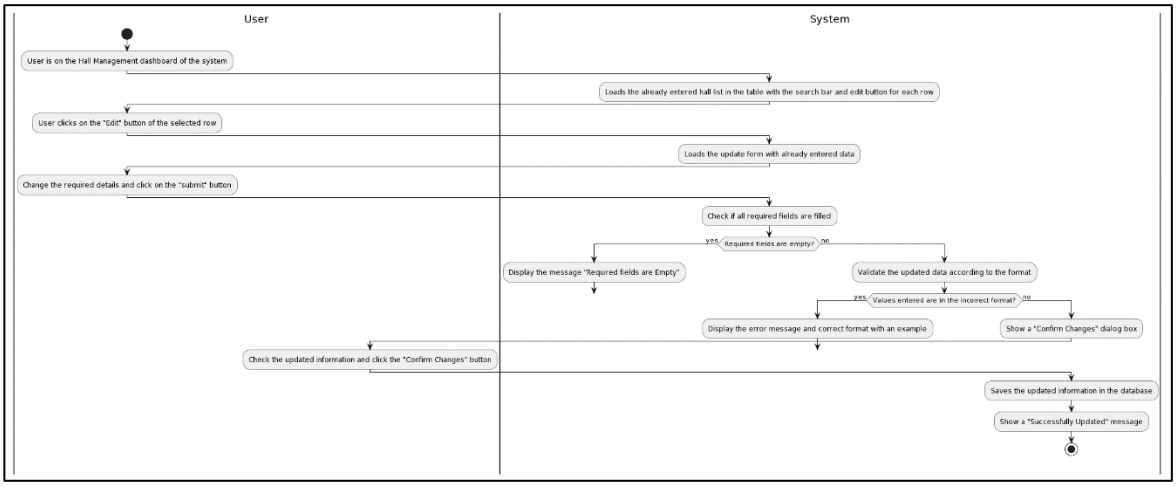


Figure 3. 16 Activity Diagram of Update Hall

Activity Diagram of View Hall

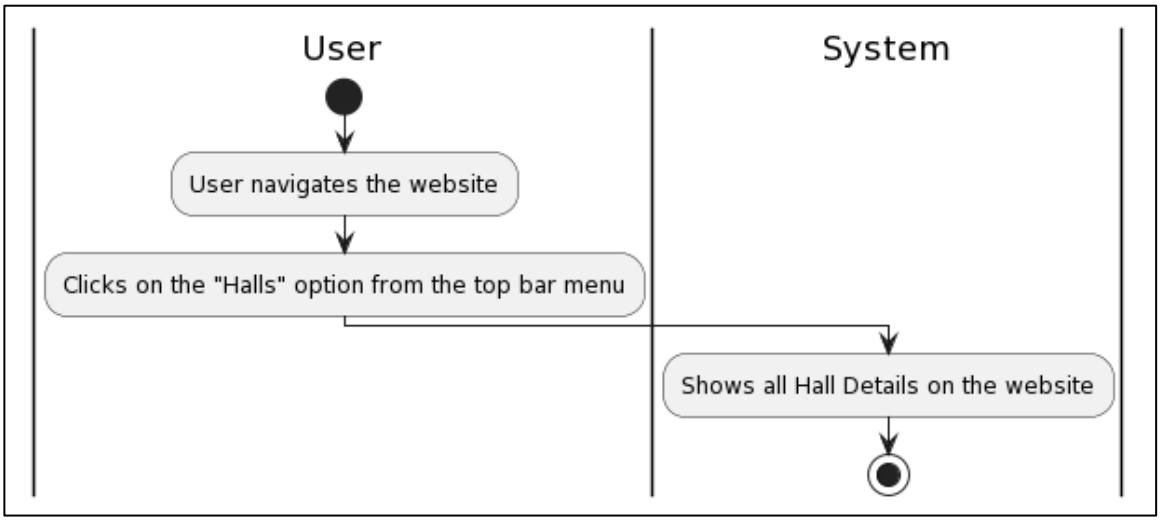


Figure 3. 17 Activity Diagram of View Hall

Activity Diagram of Add New Menu Item

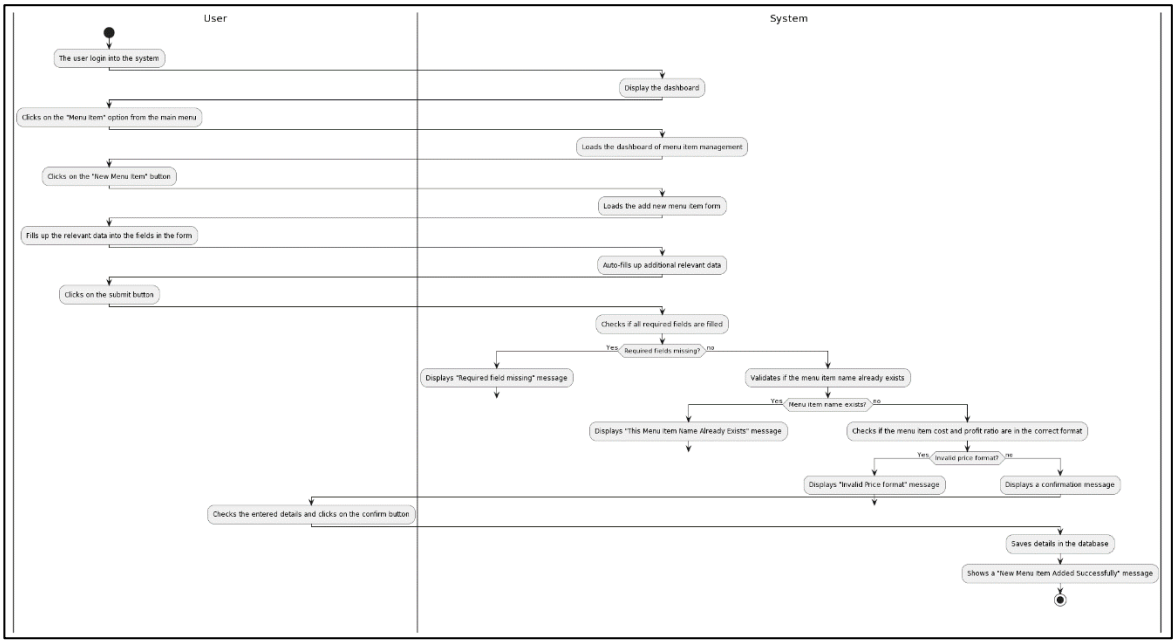


Figure 3. 18 Activity Diagram of Add New Menu Item

Activity Diagram of View Menu Item

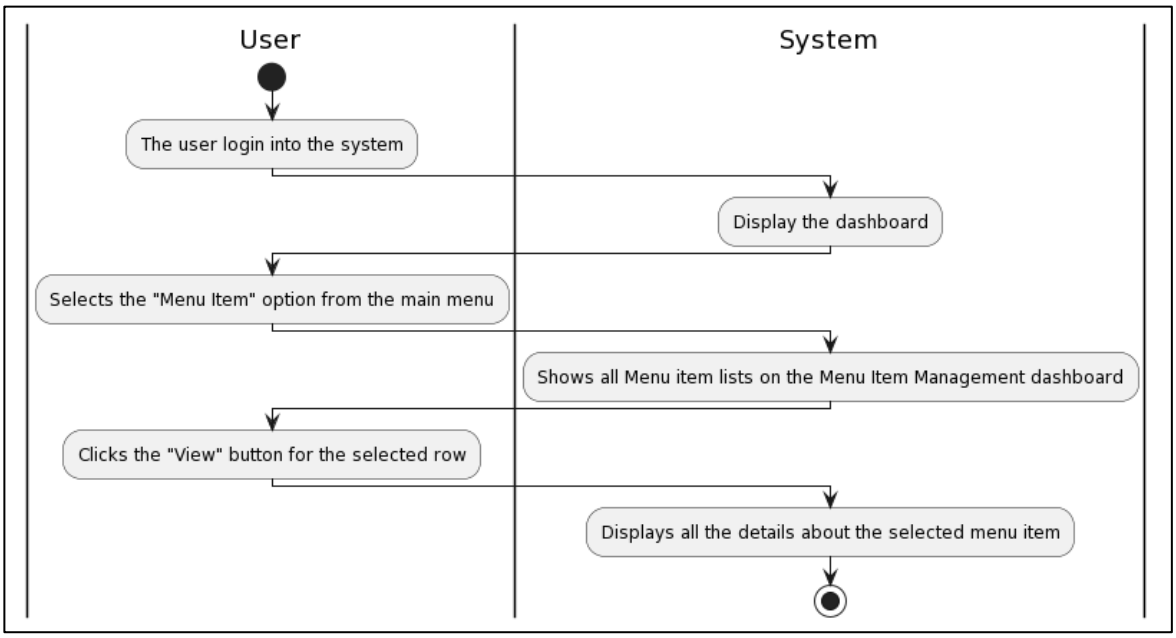


Figure 3. 19 Activity Diagram of View Menu Item

Activity Diagram of Update Menu Item

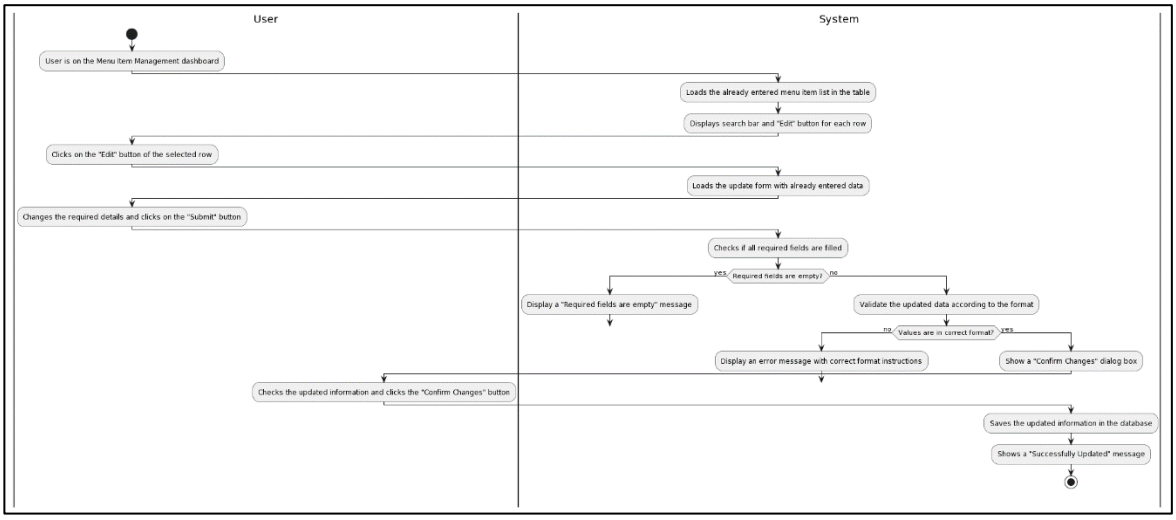


Figure 3. 20 Activity Diagram of Update Menu Item

Activity Diagram of View Menu Item

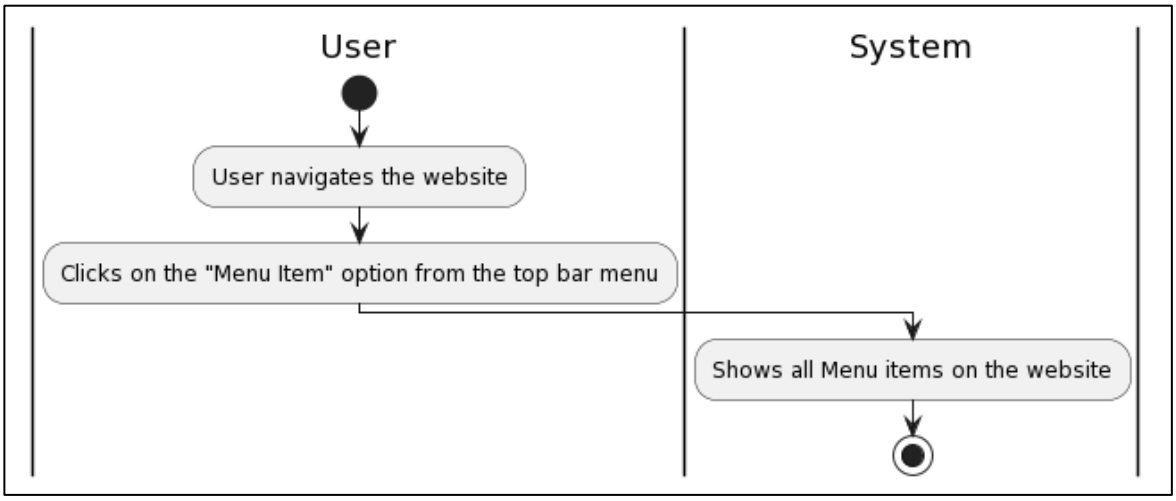


Figure 3. 21 Activity Diagram of View Menu Item

Activity Diagram of Create New Menu Package

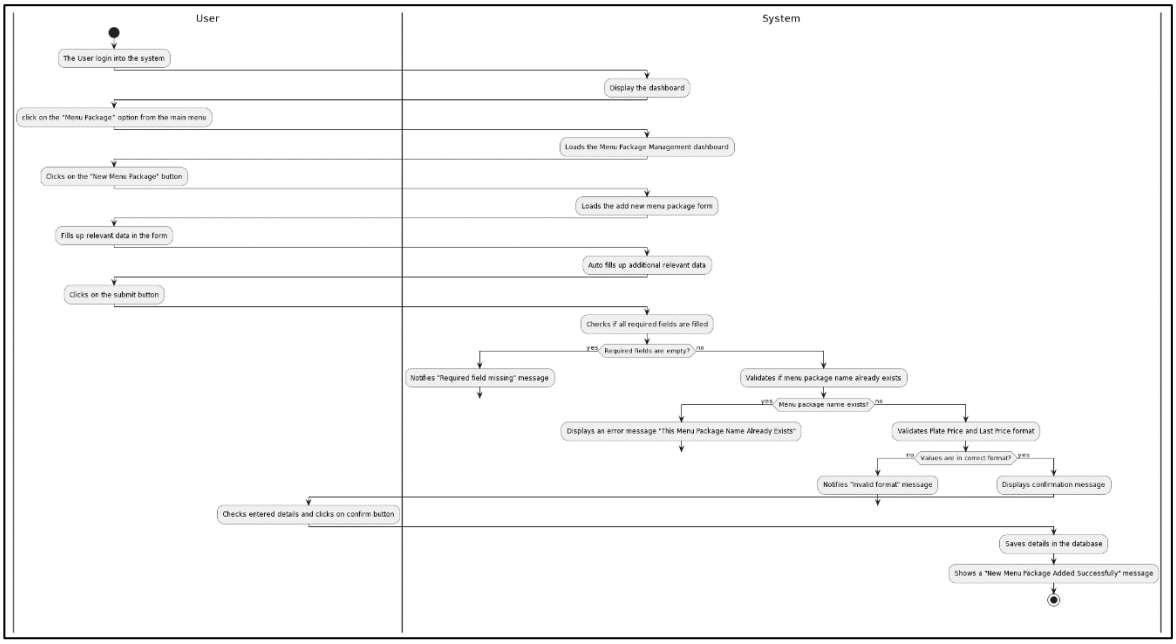


Figure 3. 22 Activity Diagram of Create New Menu Package

Activity Diagram of View Menu Package

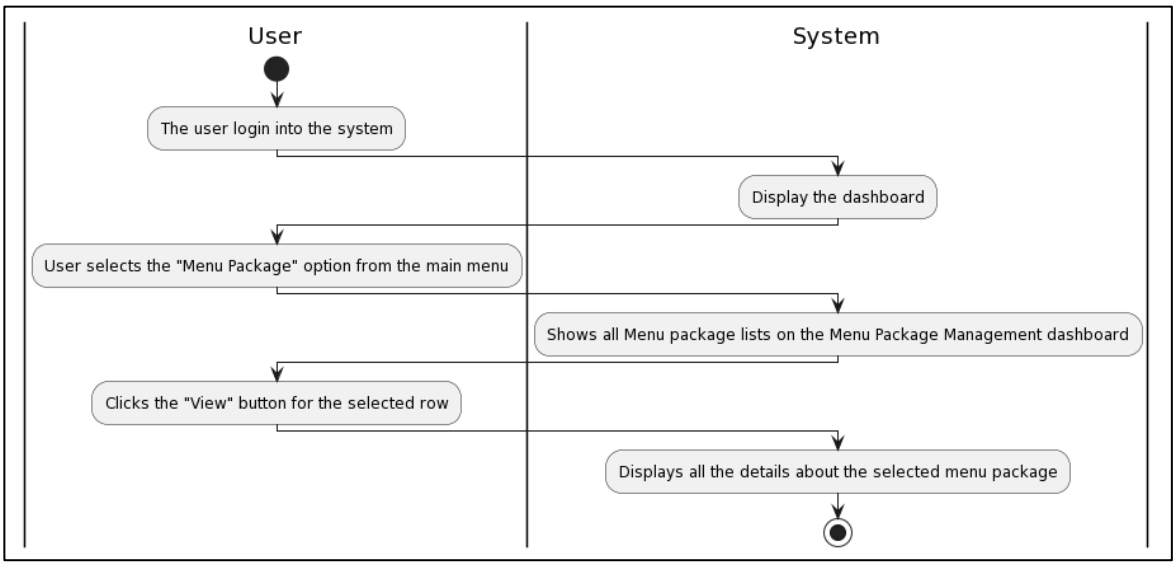


Figure 3. 23 Activity Diagram of View Menu Package

Activity Diagram of Update Menu Package

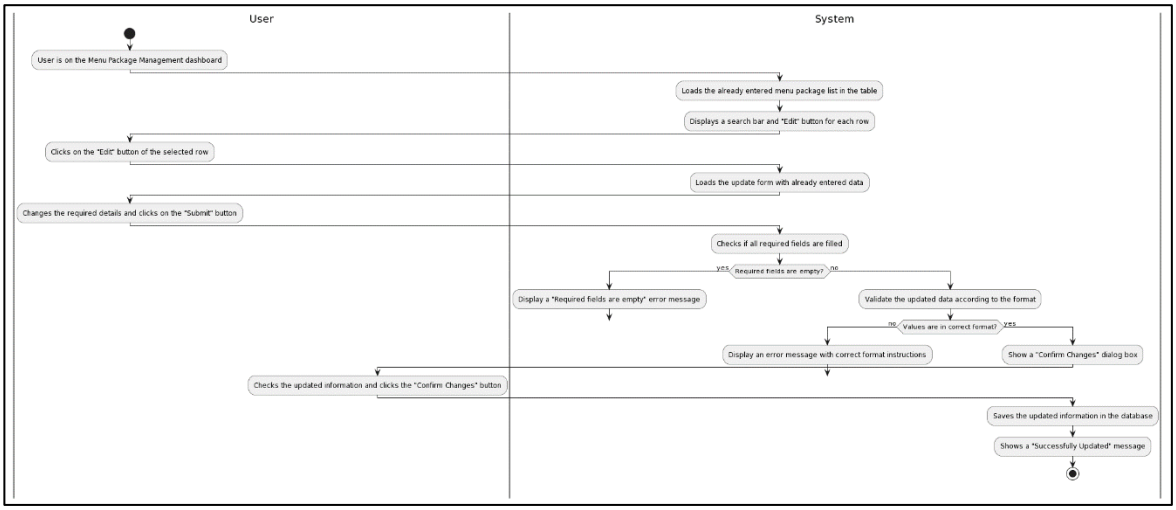


Figure 3. 24 Activity Diagram of Update Menu Package

Activity Diagram of View Menu Package

Figure 3. 25 Activity Diagram of View Menu Package

Activity Diagram of Add New Service

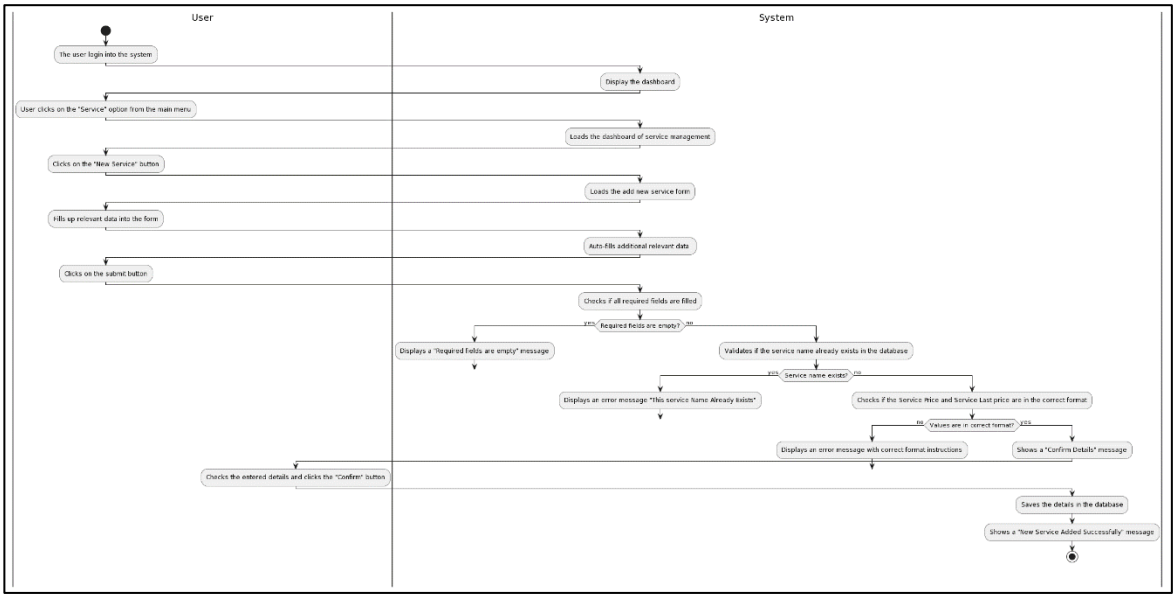


Figure 3. 26 Activity Diagram of Add New Service

Activity Diagram of View Services

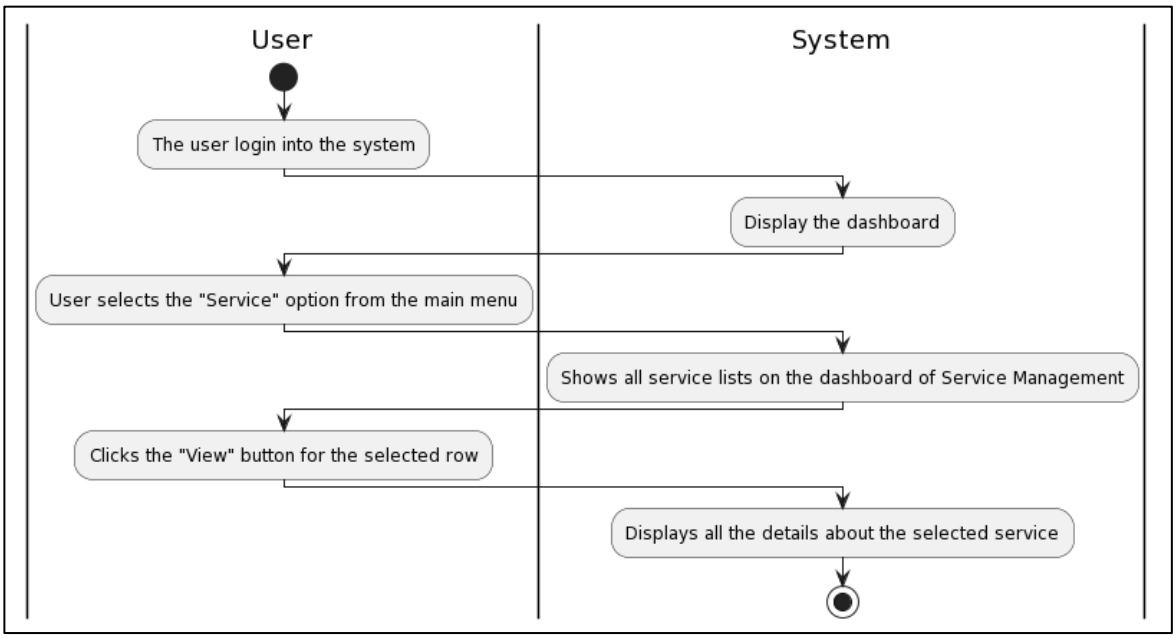


Figure 3. 27 Activity Diagram of View Services

Activity Diagram of Update Service

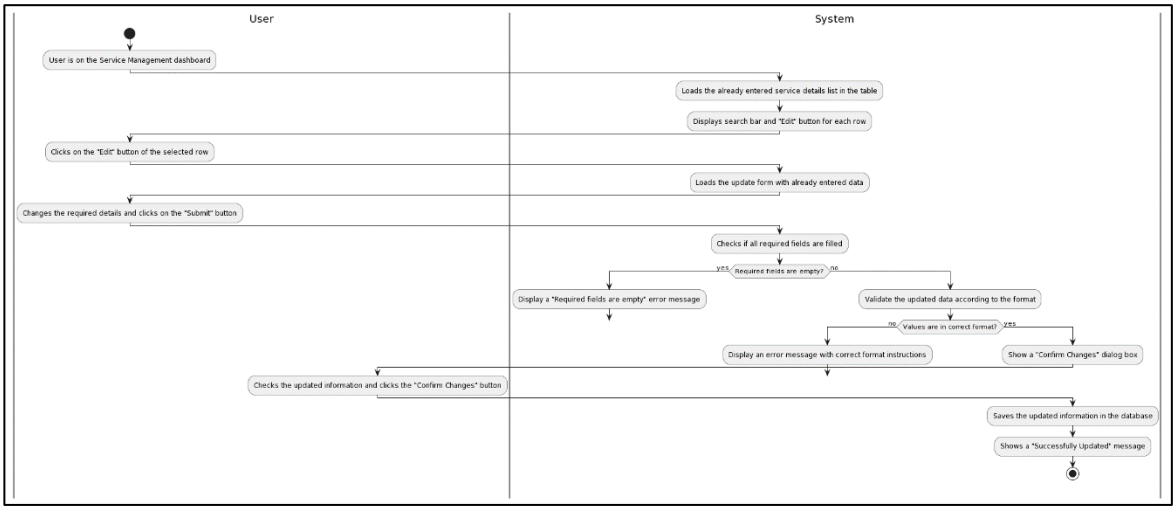


Figure 3. 28 Activity Diagram of Update Service

Activity Diagram of View Services

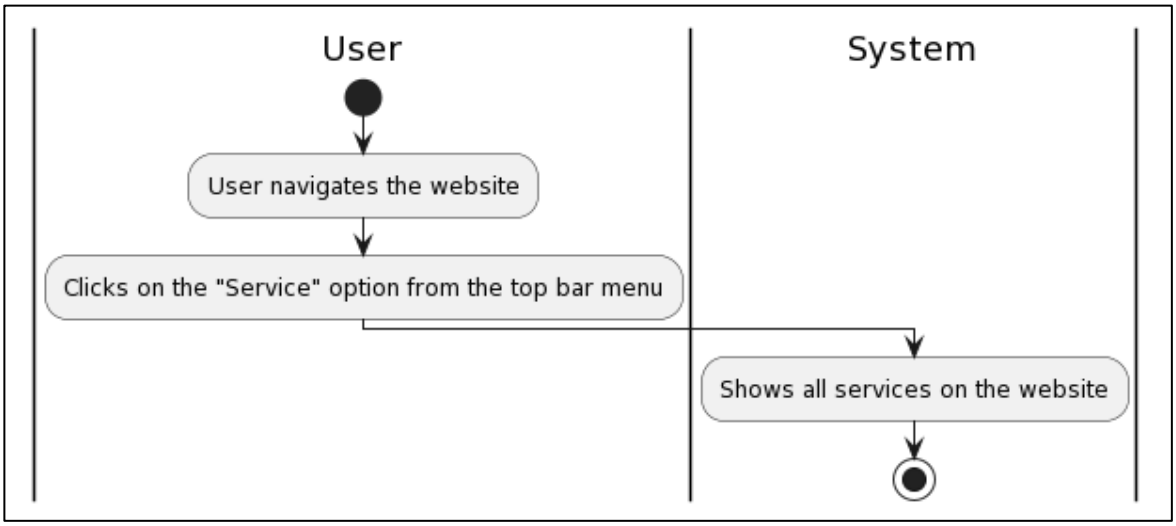


Figure 3. 29 Activity Diagram of View Services

Activity Diagram of Check Hall Availability

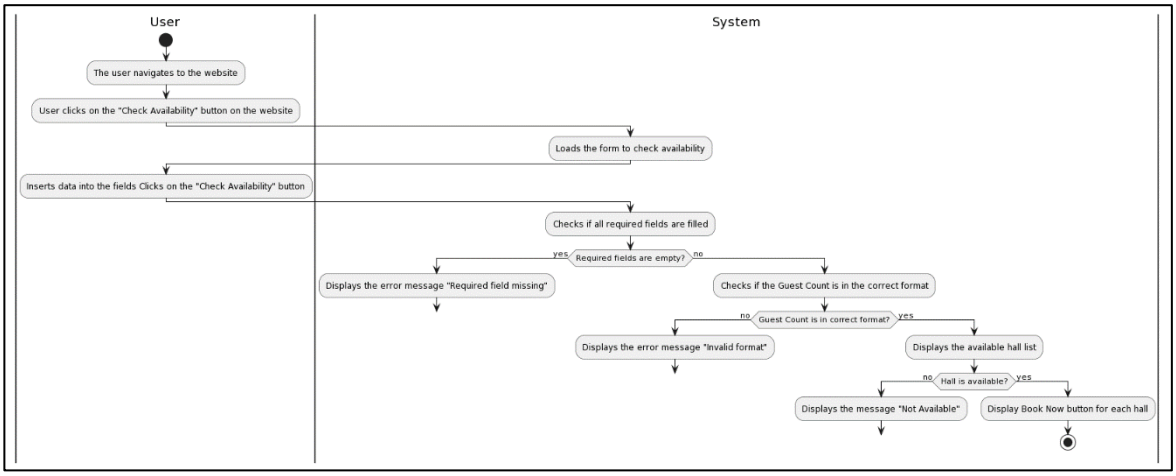


Figure 3. 30 Activity Diagram of Check Hall Availability

Activity Diagram of Make New Reservation (Step 1)

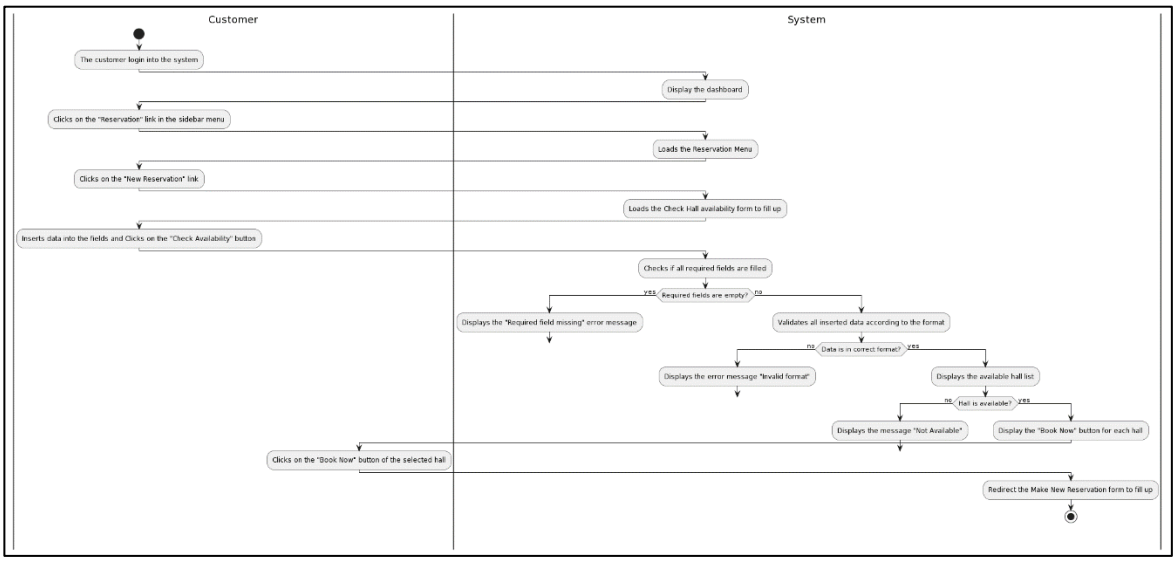


Figure 3. 31 Activity Diagram of Make New Reservation (Step 1)

Activity Diagram of Make New Reservation (Step 2)

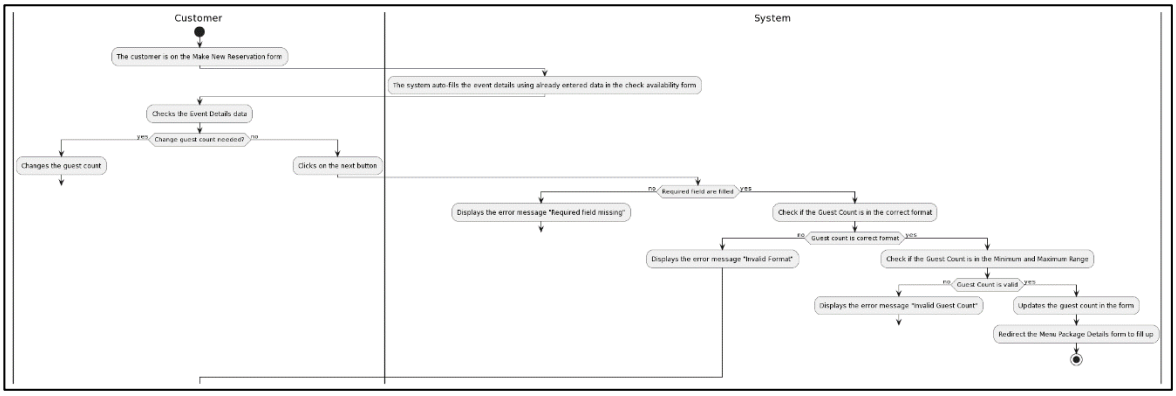


Figure 3. 32 Activity Diagram of Make New Reservation (Step 2)

Activity Diagram of Make New Reservation (Step 3)

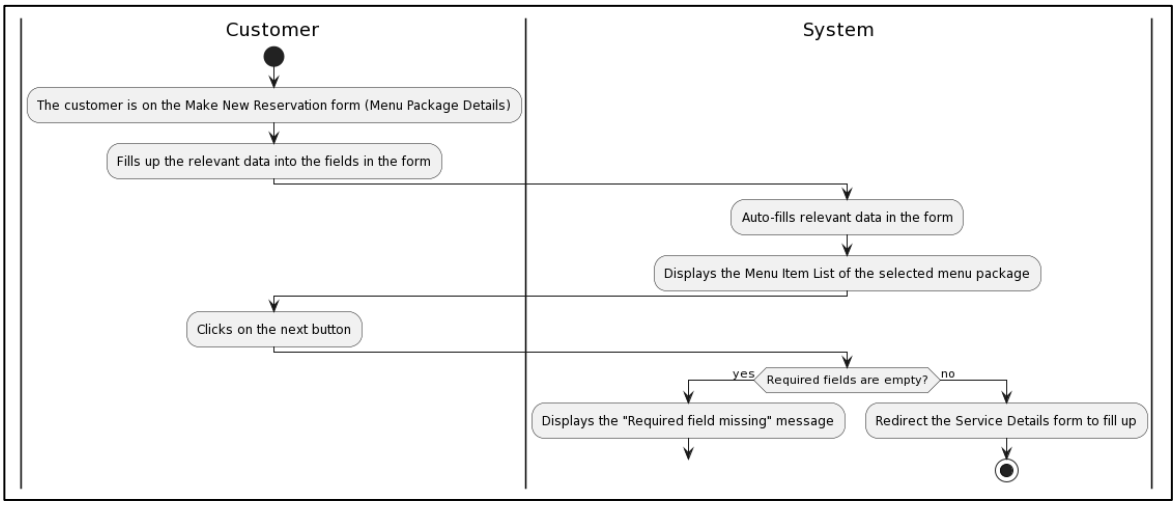


Figure 3. 33 Activity Diagram of Make New Reservation (Step 3)

Activity Diagram of Make New Reservation (Step 4)

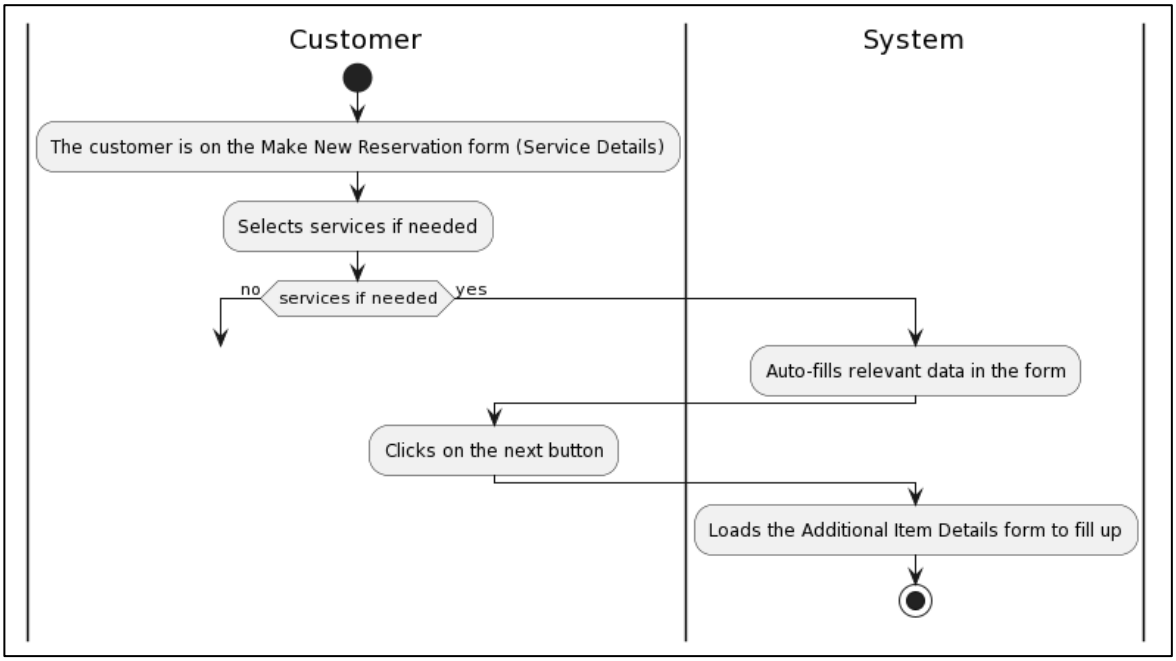


Figure 3. 34 Activity Diagram of Make New Reservation (Step 4)

Activity Diagram of Make New Reservation (Step 5)

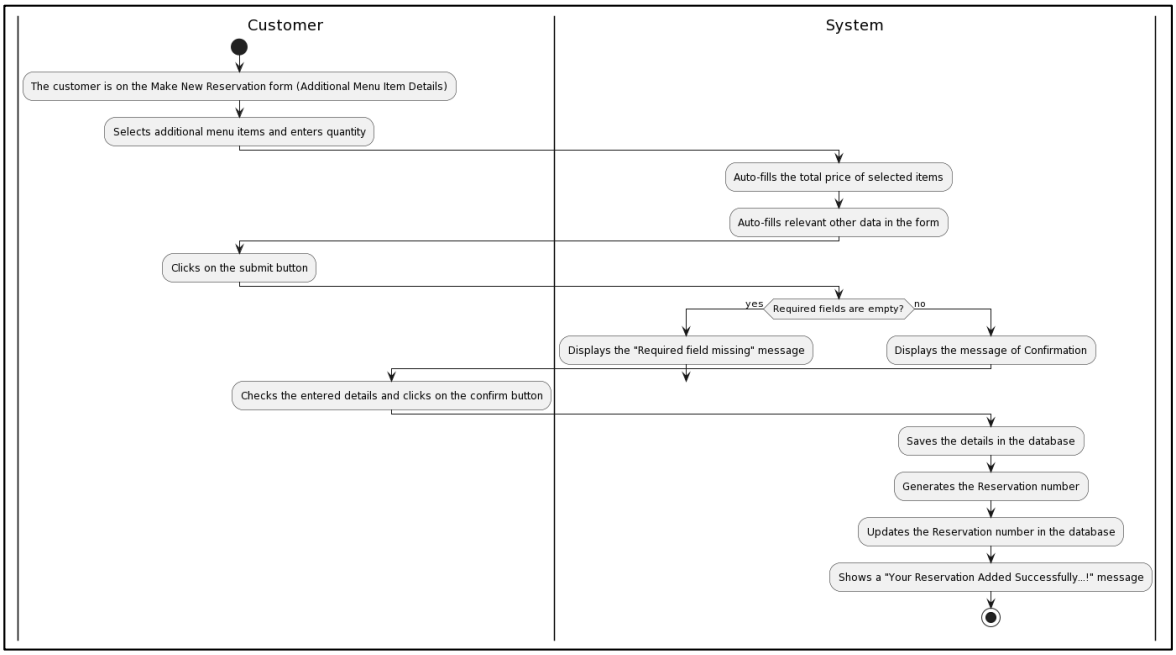


Figure 3. 35 Activity Diagram of Make New Reservation (Step 5)

Activity Diagram of View Reservation History

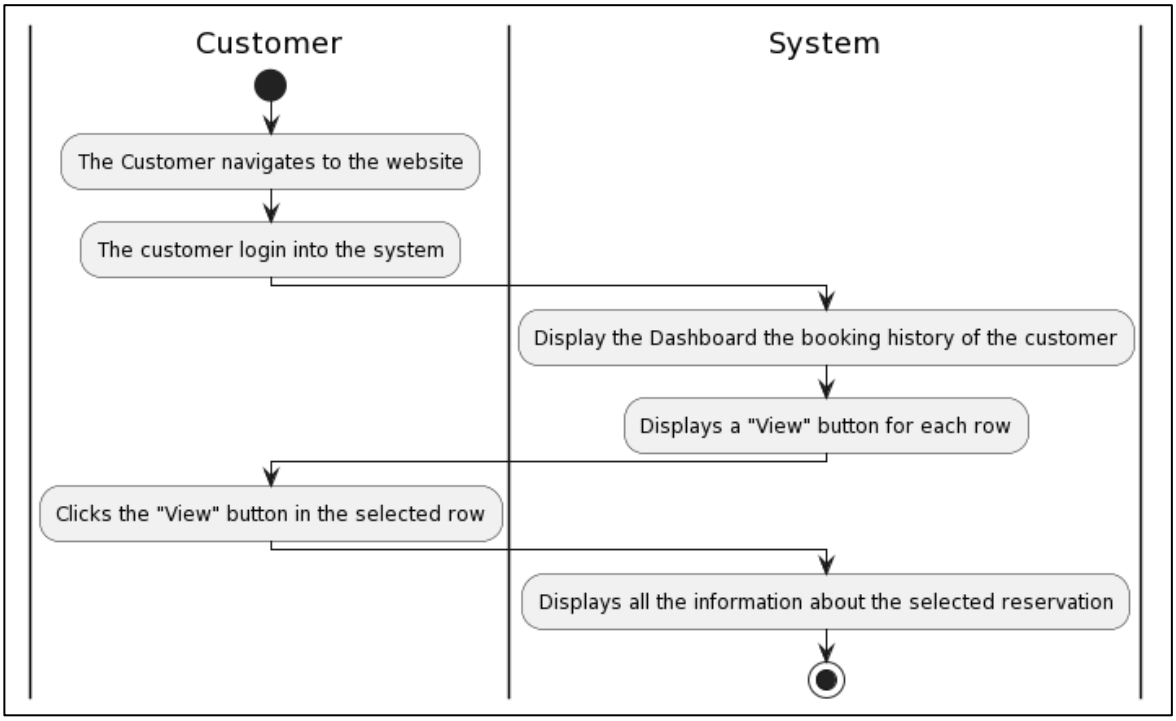


Figure 3. 36 Activity Diagram of View Reservation History

Activity Diagram of View Reservation

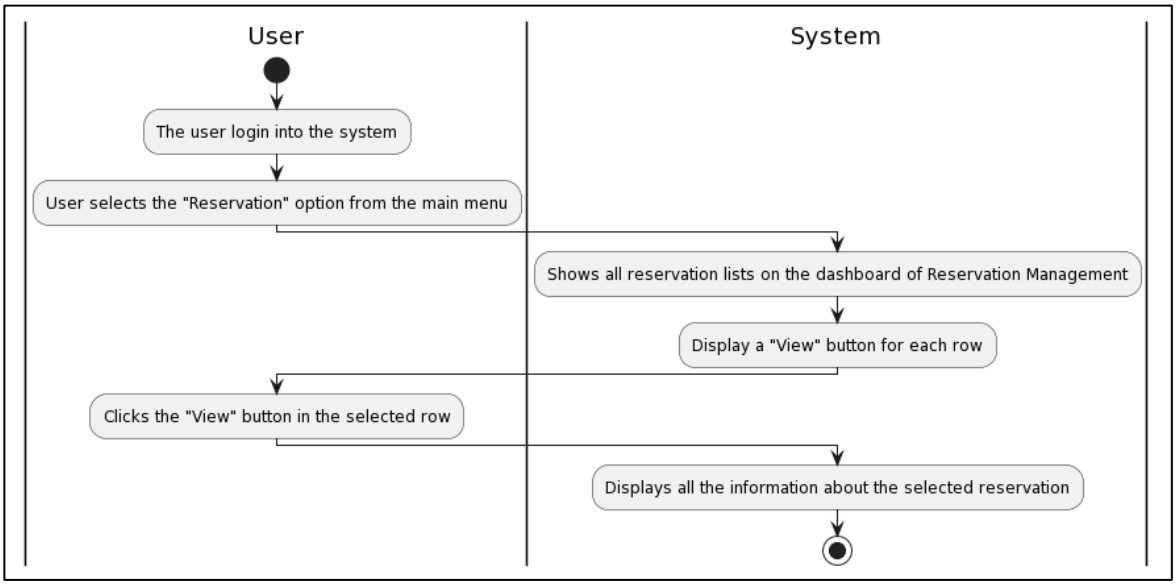


Figure 3. 37 Activity Diagram of View Reservation

Activity Diagram of Request to Update Menu Package/Service/Additional Item

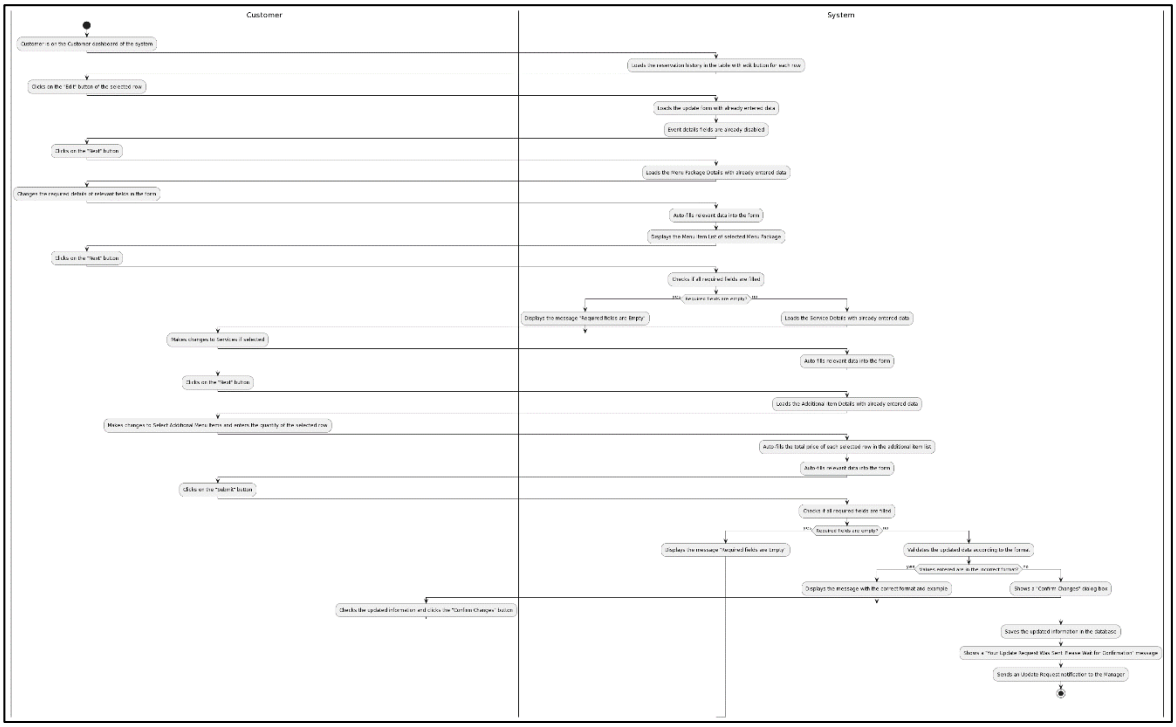


Figure 3. 38 Activity Diagram of Request to Update Menu Package/Service/Additional Item

Activity Diagram of Request to Update Event Details

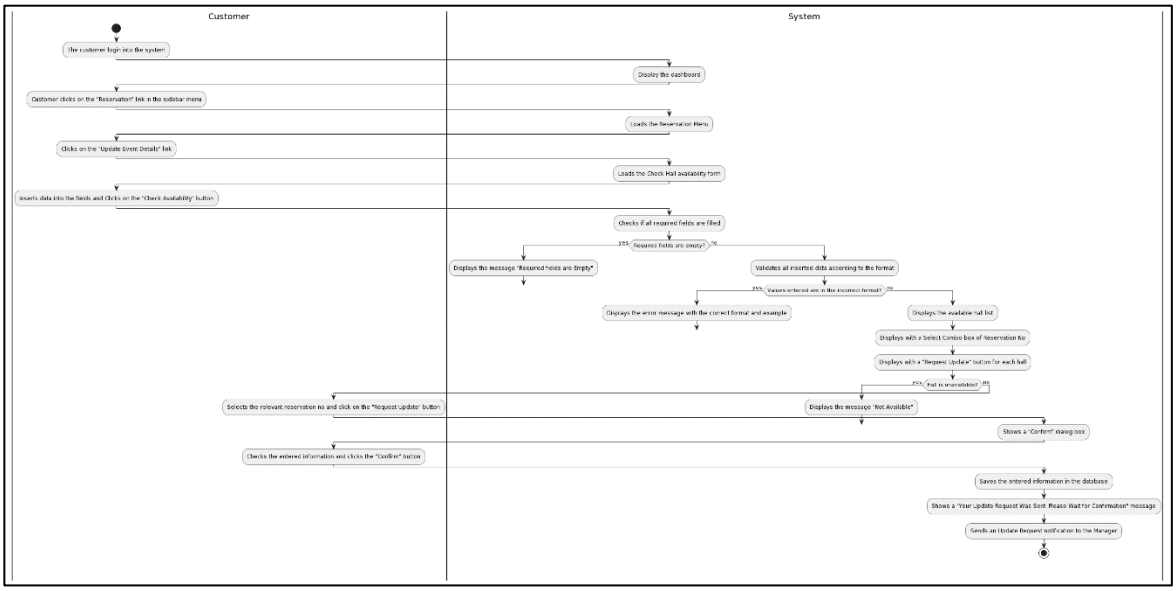


Figure 3. 39 Activity Diagram of Request to Update Event Details

Activity Diagram of Cancel Reservation

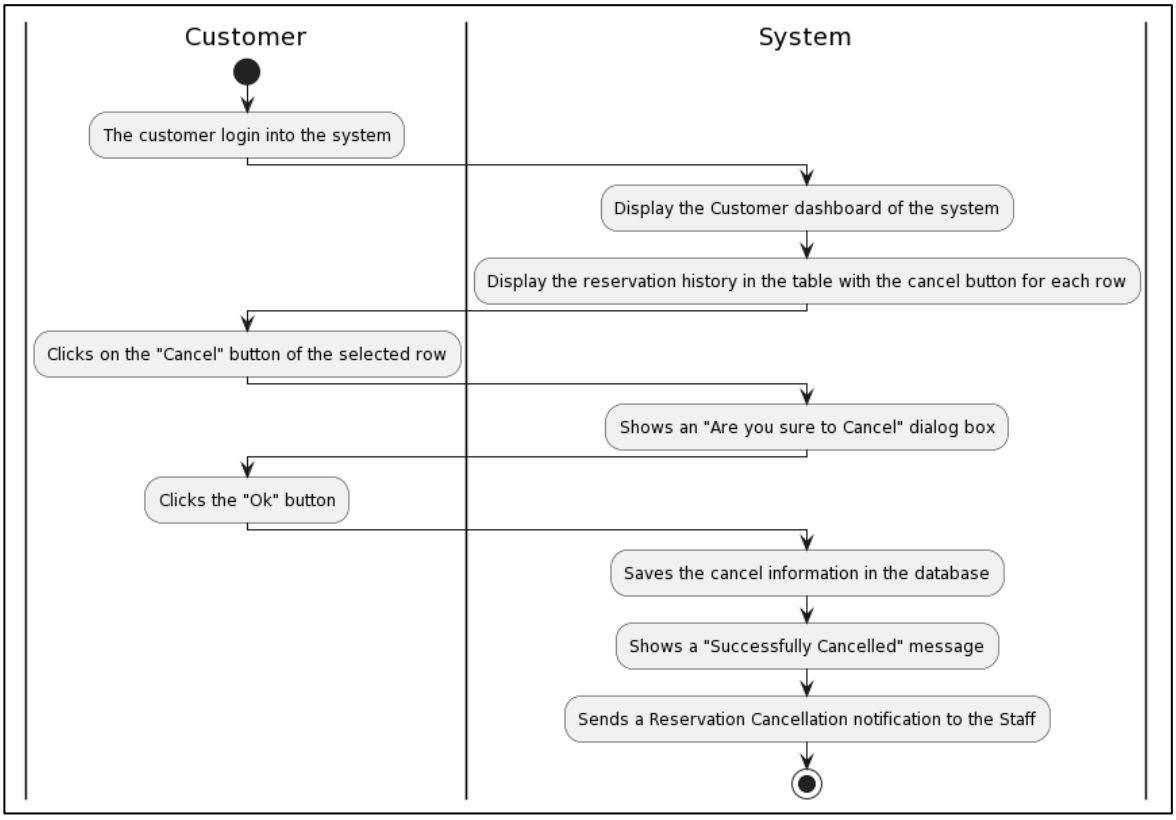


Figure 3. 40 Activity Diagram of Cancel Reservation

Activity Diagram of Make Payment

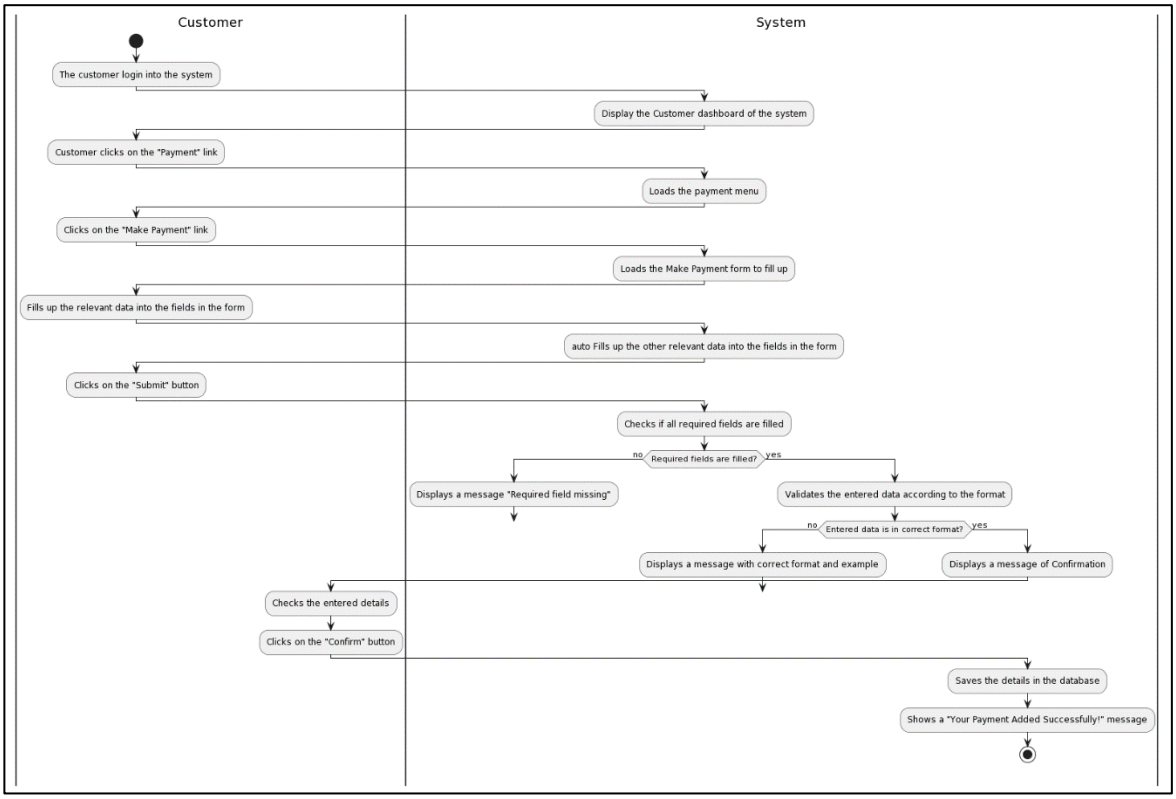


Figure 3. 41 Activity Diagram of Make Payment

Activity Diagram of View Payment History

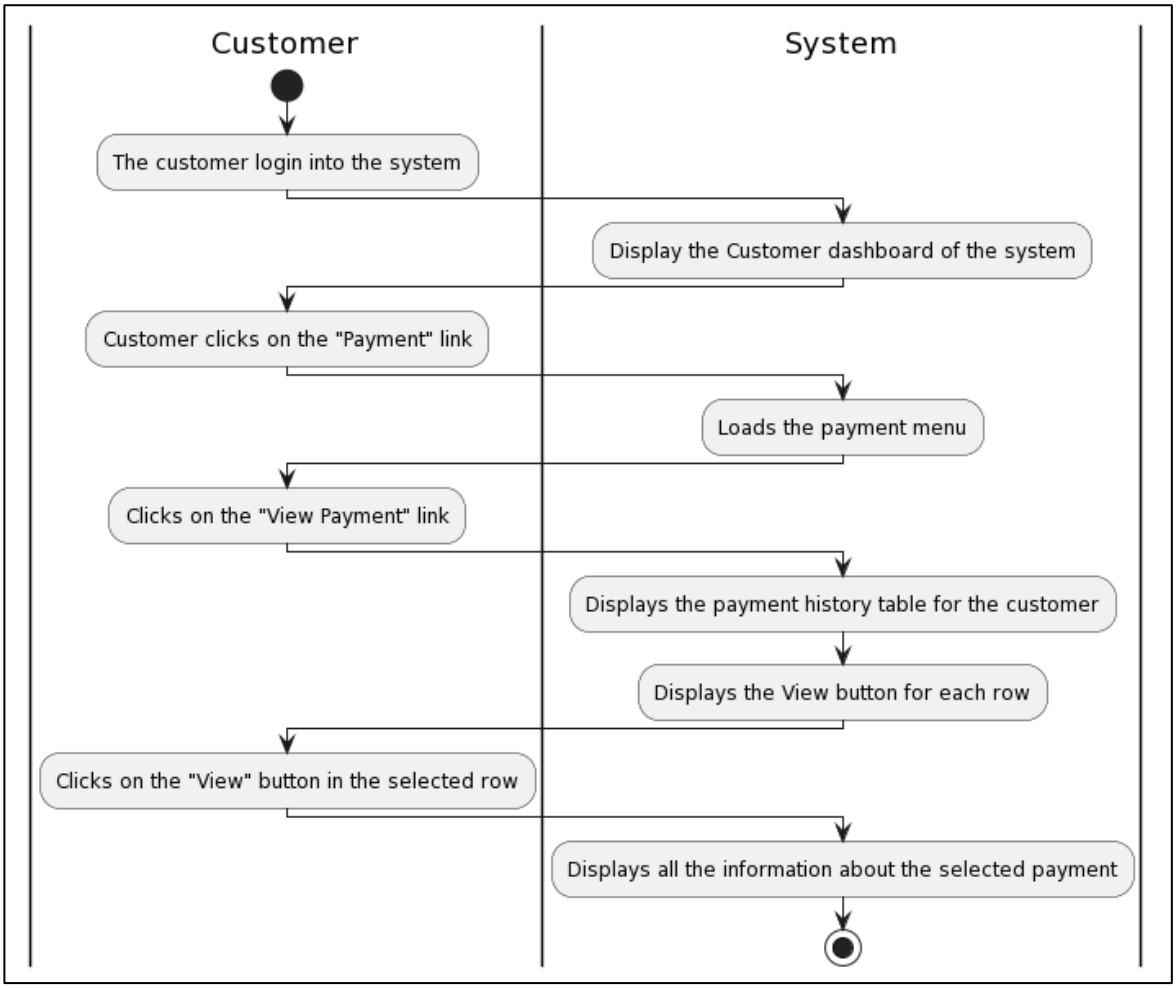


Figure 3. 42 Activity Diagram of View Payment History

Activity Diagram of Update Customer Payment

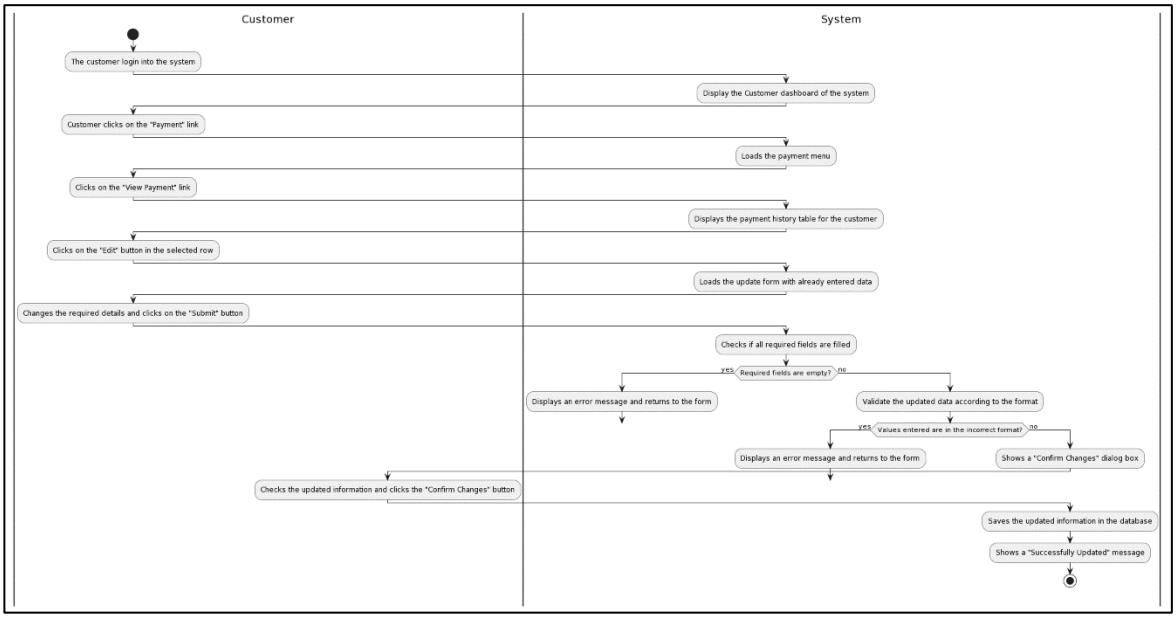


Figure 3. 43 Activity Diagram of Update Customer Payment

Activity Diagram of View Customer Payment

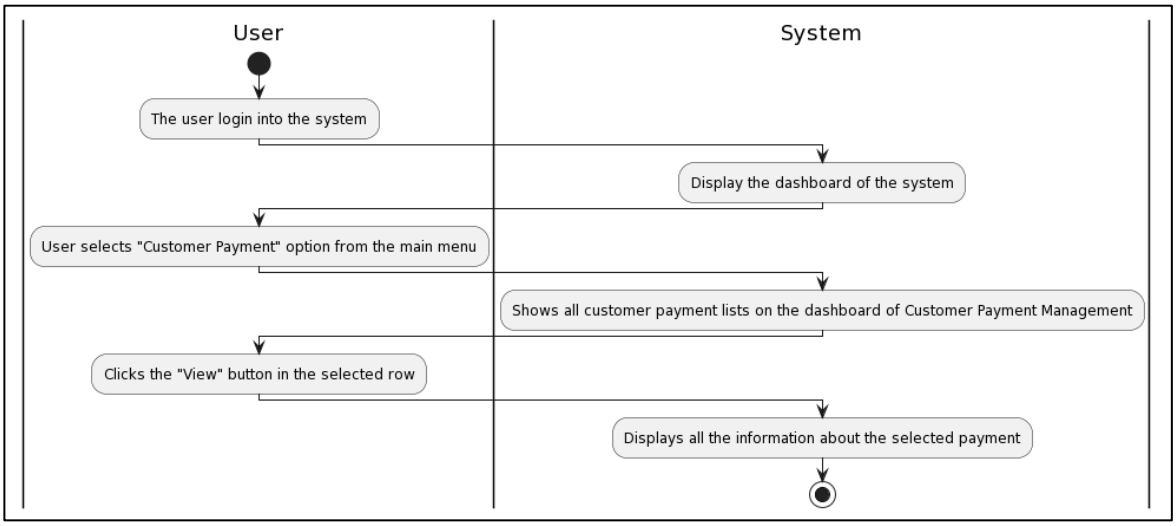


Figure 3. 44 Activity Diagram of View Customer Payment

Activity Diagram of Verify Received Customer Payments

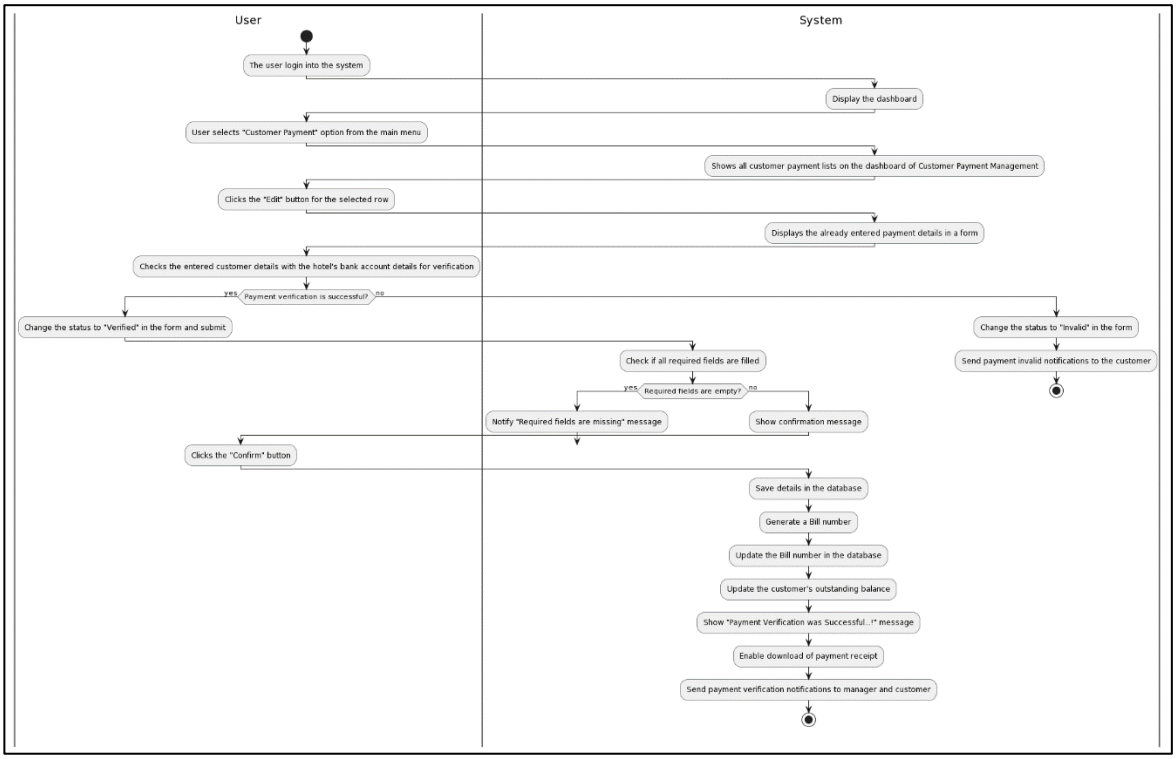


Figure 3. 45 Activity Diagram of Verify Received Customer Payments

Activity Diagram of Confirm/Reject Reservation

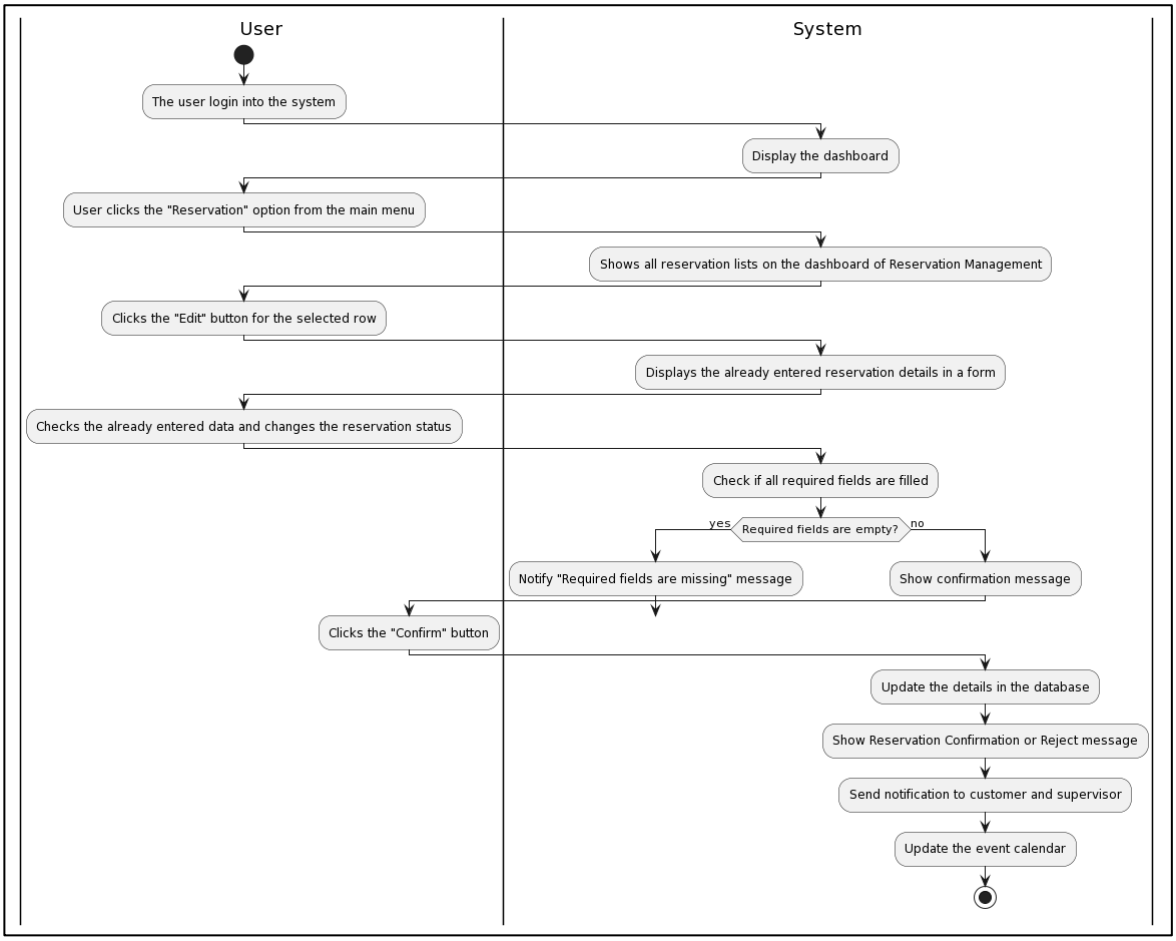


Figure 3. 46 Activity Diagram of Confirm/Reject Reservation

Activity Diagram of View Reservation Update Request

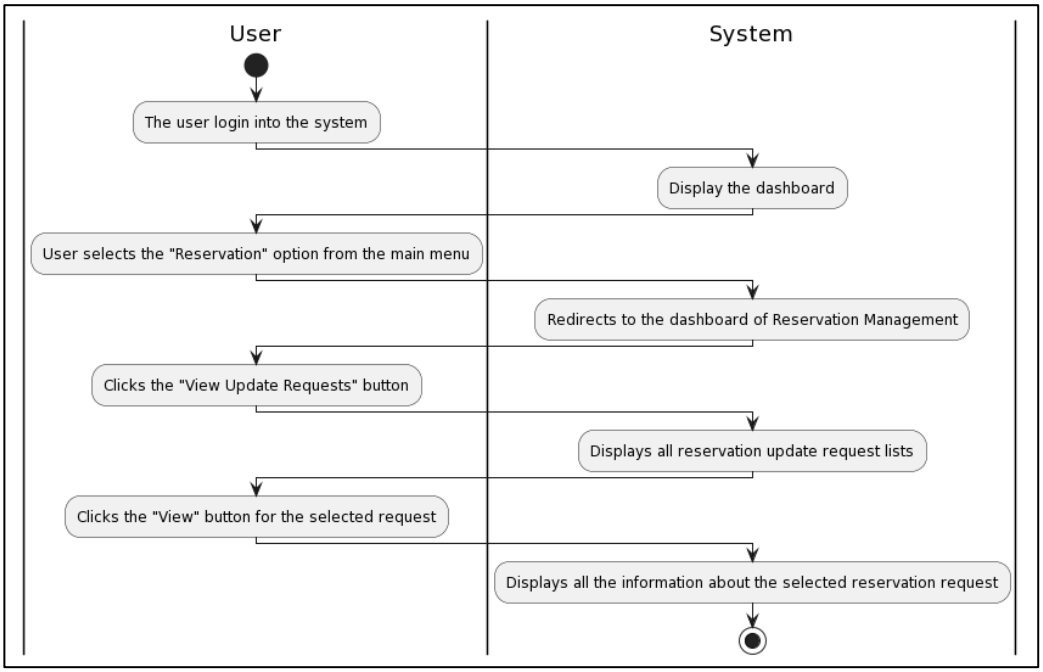


Figure 3. 47 Activity Diagram of View Reservation Update Request

Activity Diagram of Approve/Reject Reservation Update Request

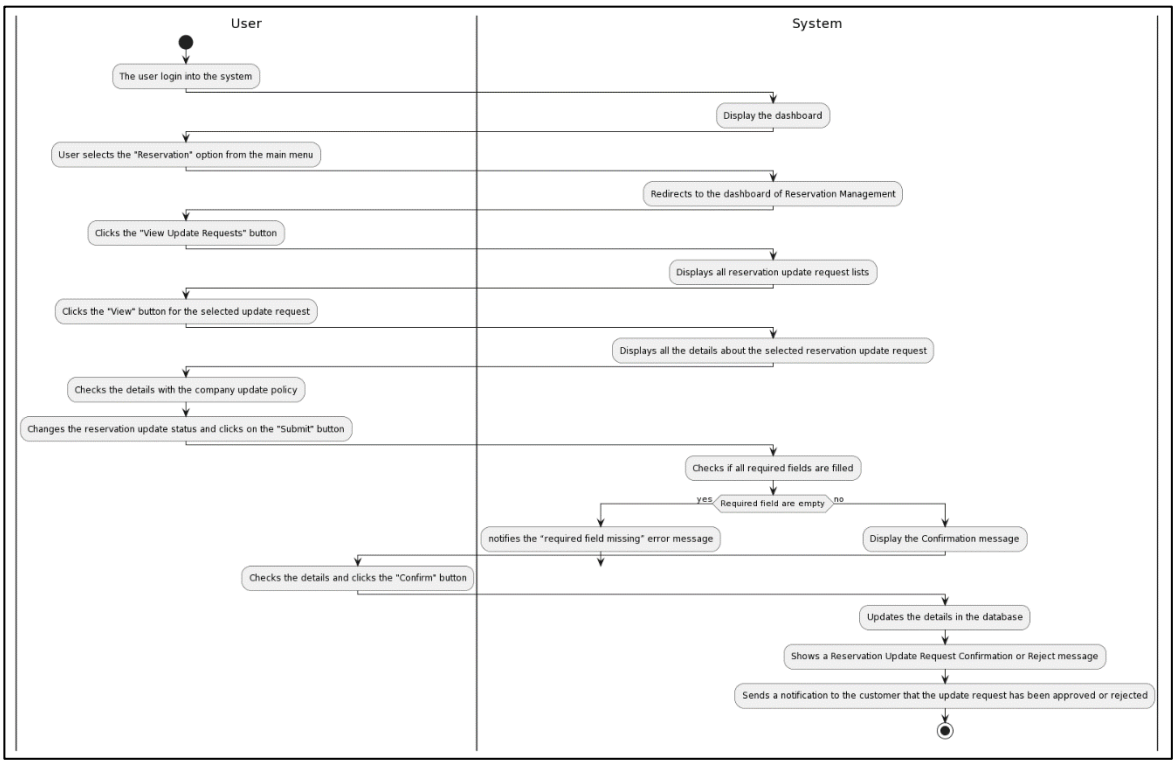


Figure 3. 48 Activity Diagram of Approve/Reject Reservation Update Request

Activity Diagram of Request Refund Payment

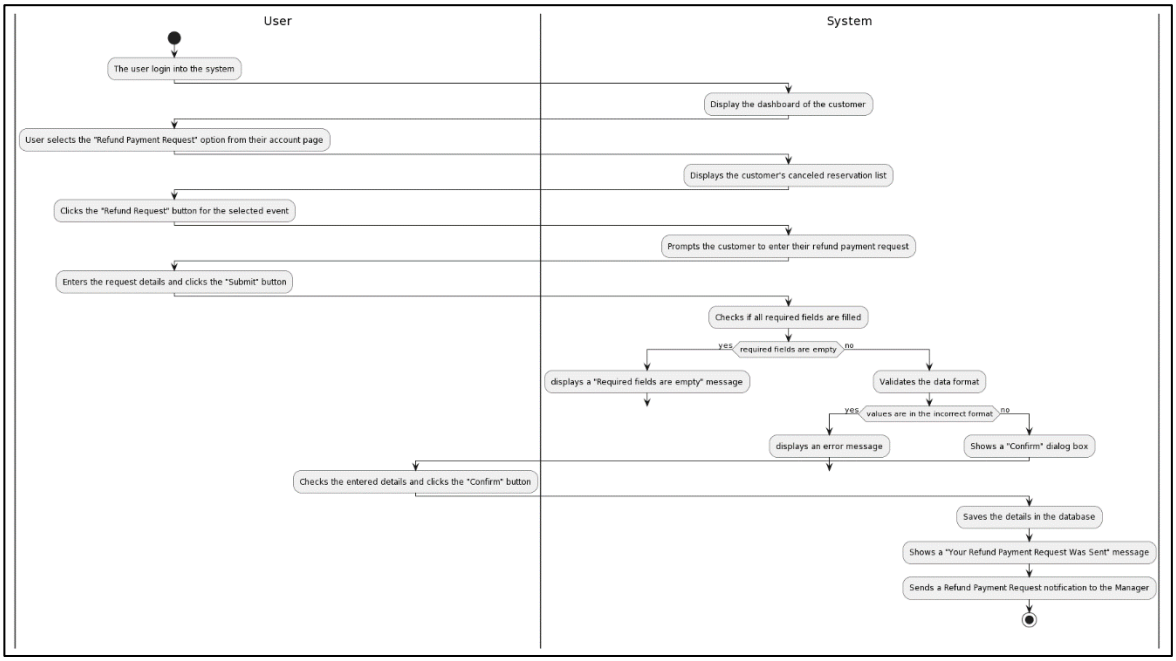


Figure 3. 49 Activity Diagram of Request Refund Payment

Activity Diagram of View Refund Payment Request

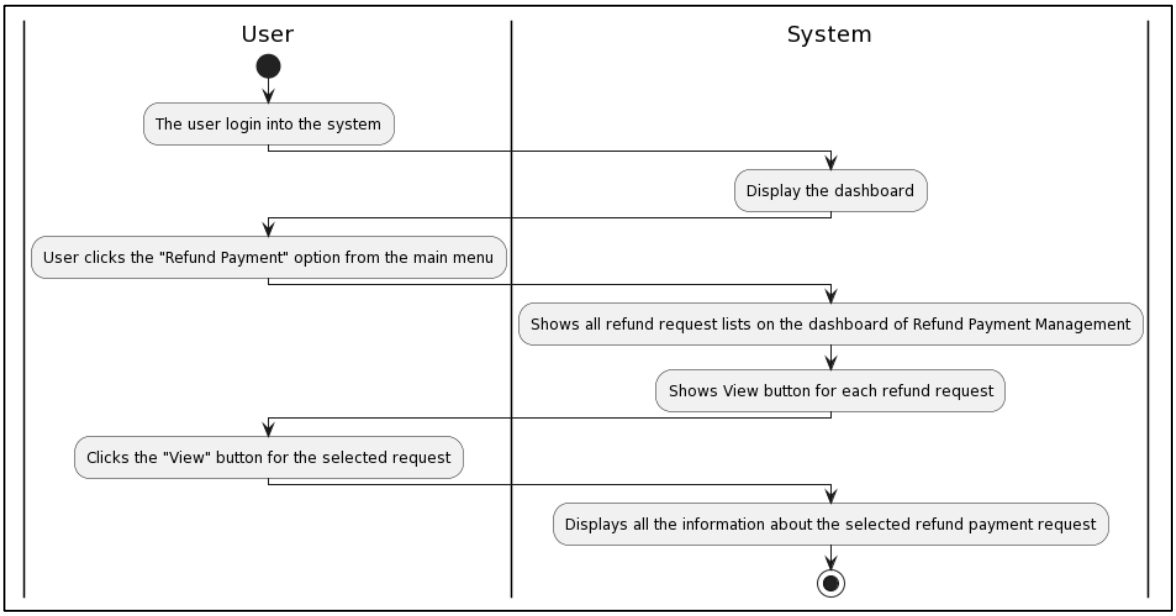


Figure 3. 50 Activity Diagram of View Refund Payment Request

Activity Diagram of Approve/Reject Refund Payment Request

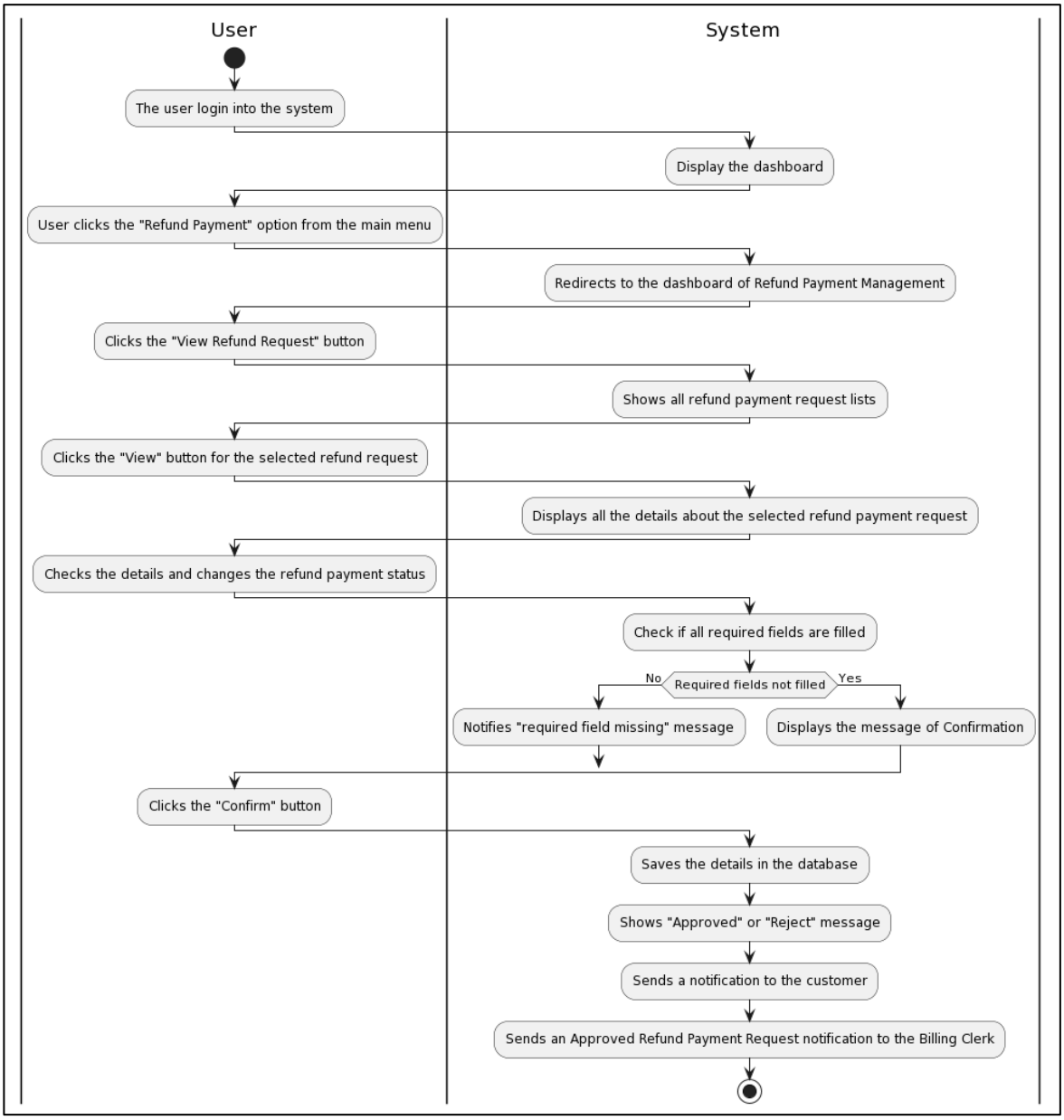


Figure 3. 51 Activity Diagram of Approve/Reject Refund Payment Request

Activity Diagram of View Approved Refund Payments

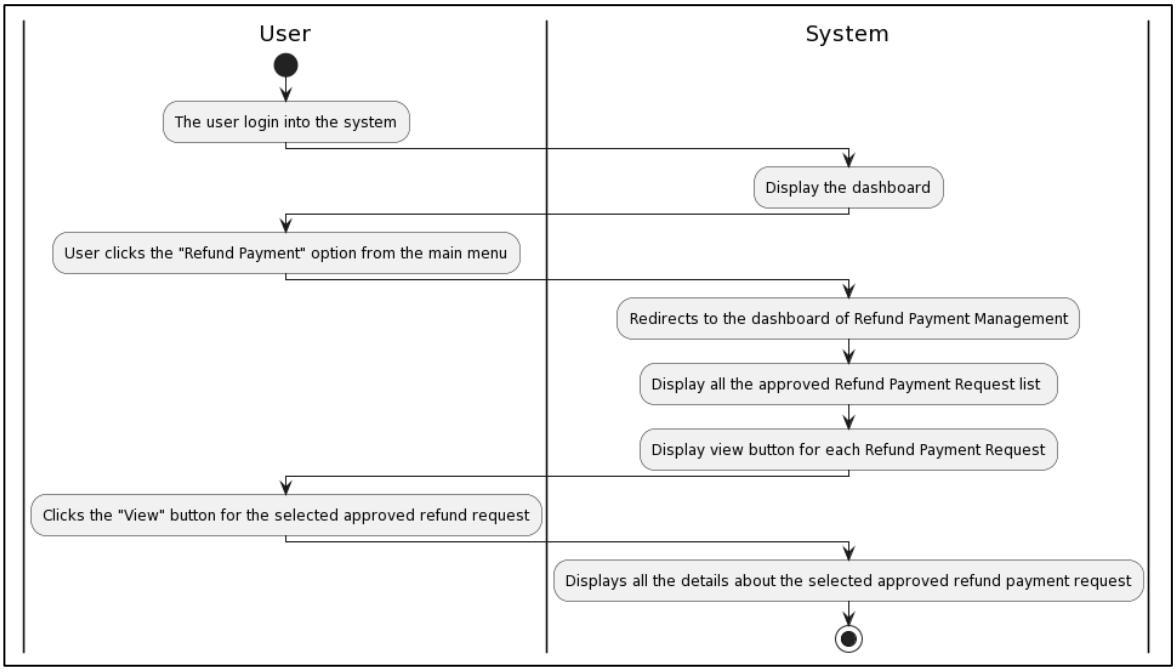


Figure 3. 52 Activity Diagram of View Approved Refund Payments

Activity Diagram of Issue Refund Payment

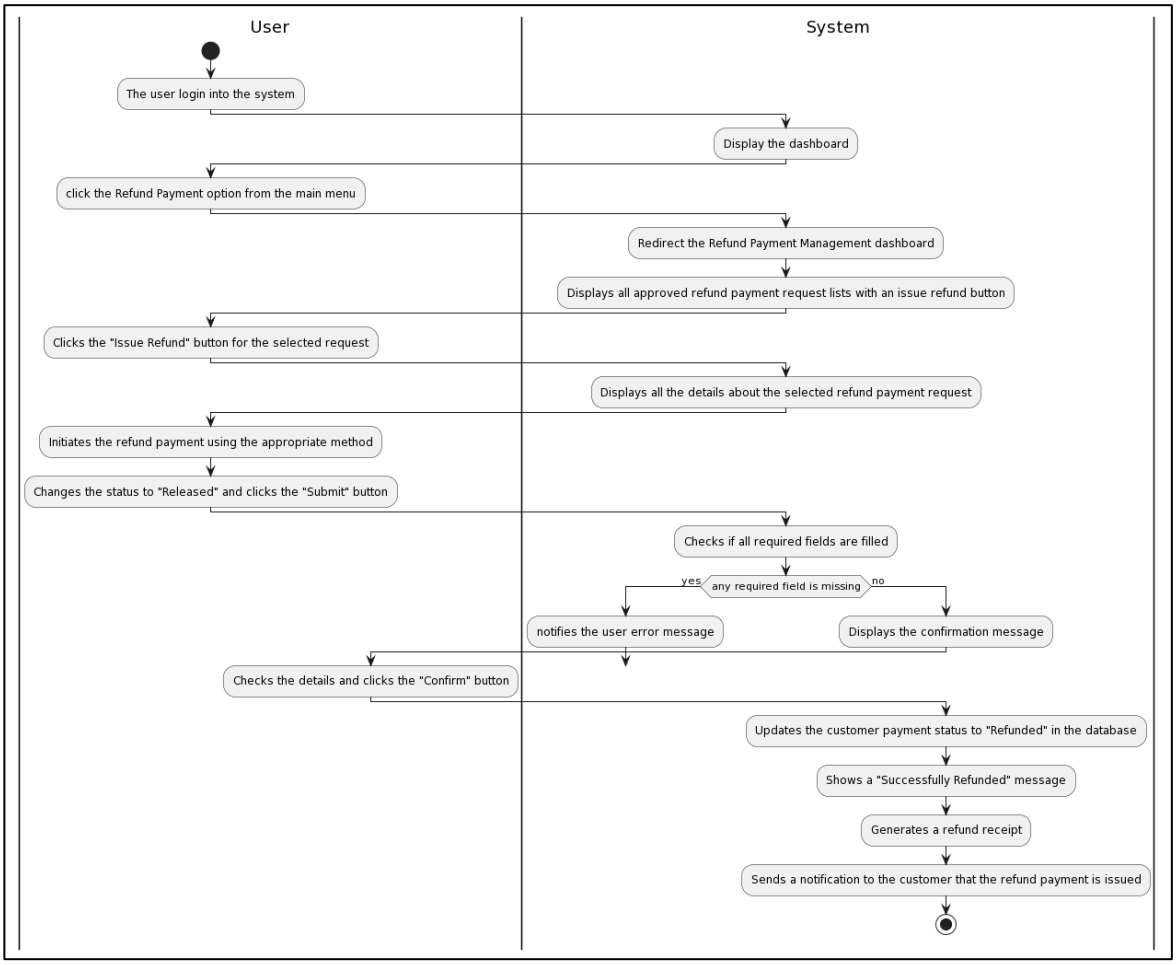


Figure 3. 53 Activity Diagram of Issue Refund Payment

Activity Diagram of Submit Reviews

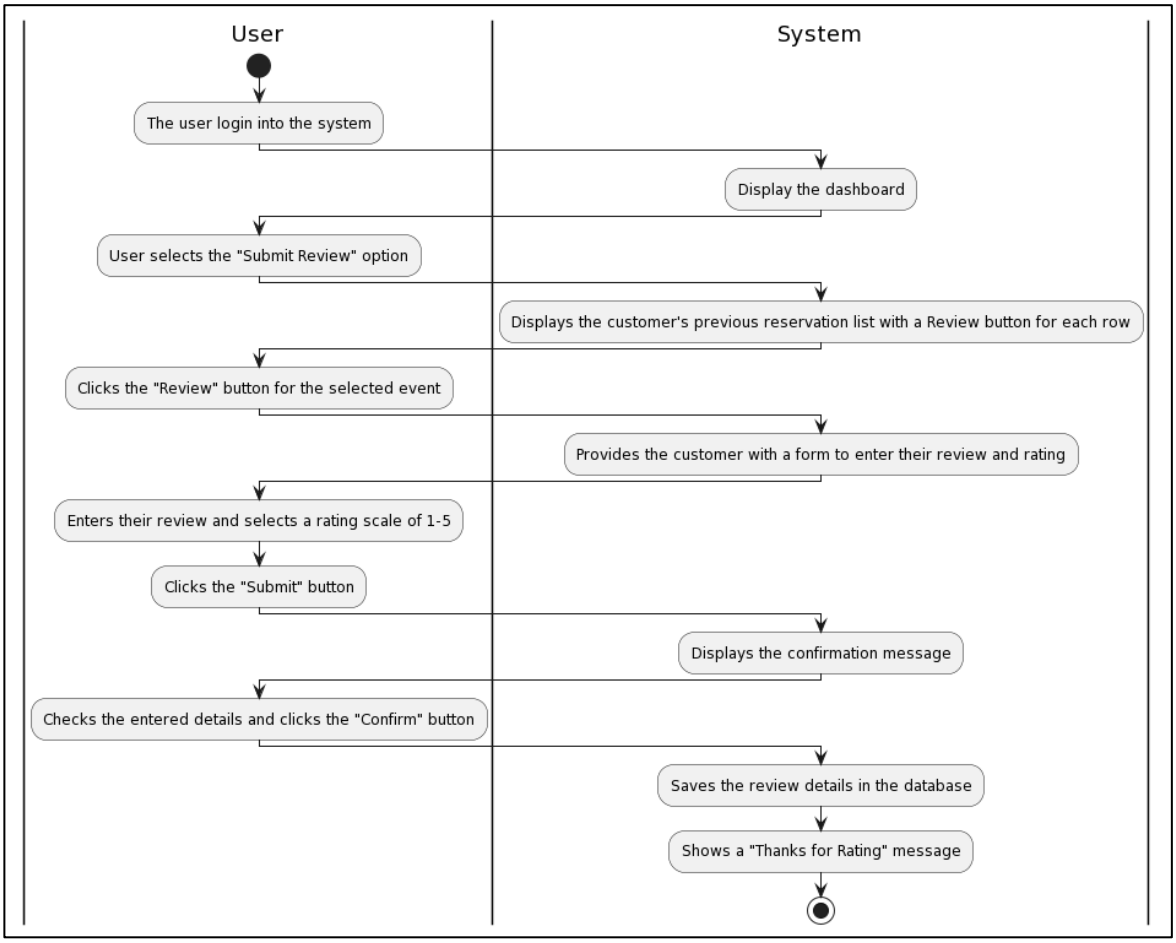


Figure 3. 54 Activity Diagram of Submit Reviews

Activity Diagram of View Customer Reviews

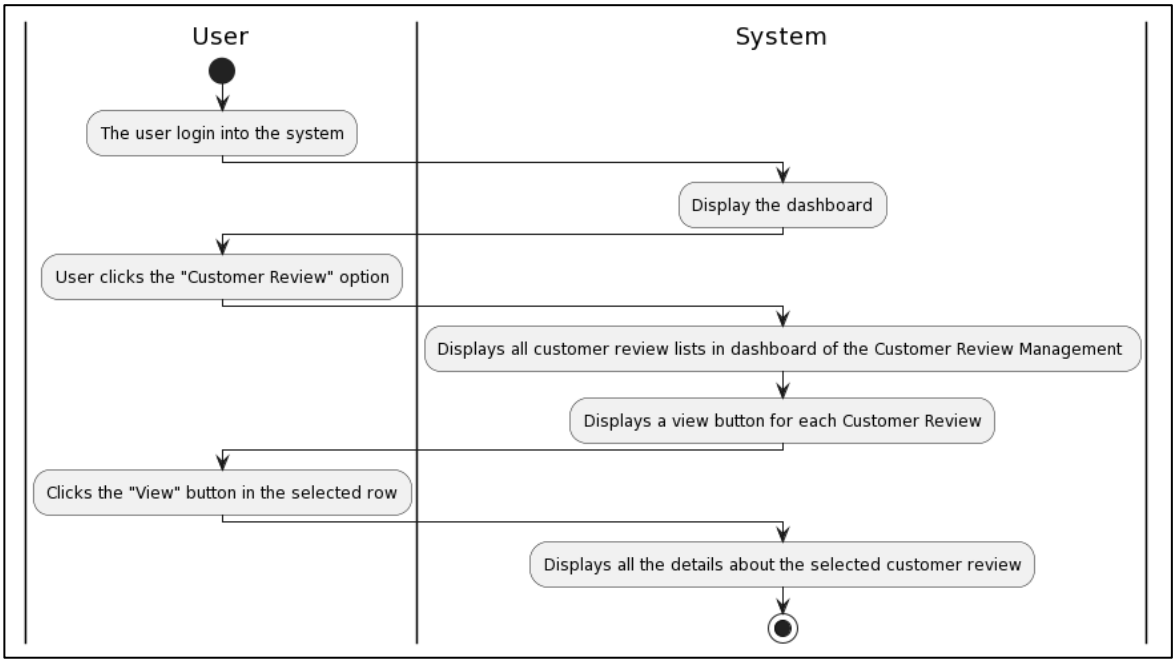


Figure 3. 55 Activity Diagram of View Customer Reviews

Activity Diagram of Reply to Customer Reviews

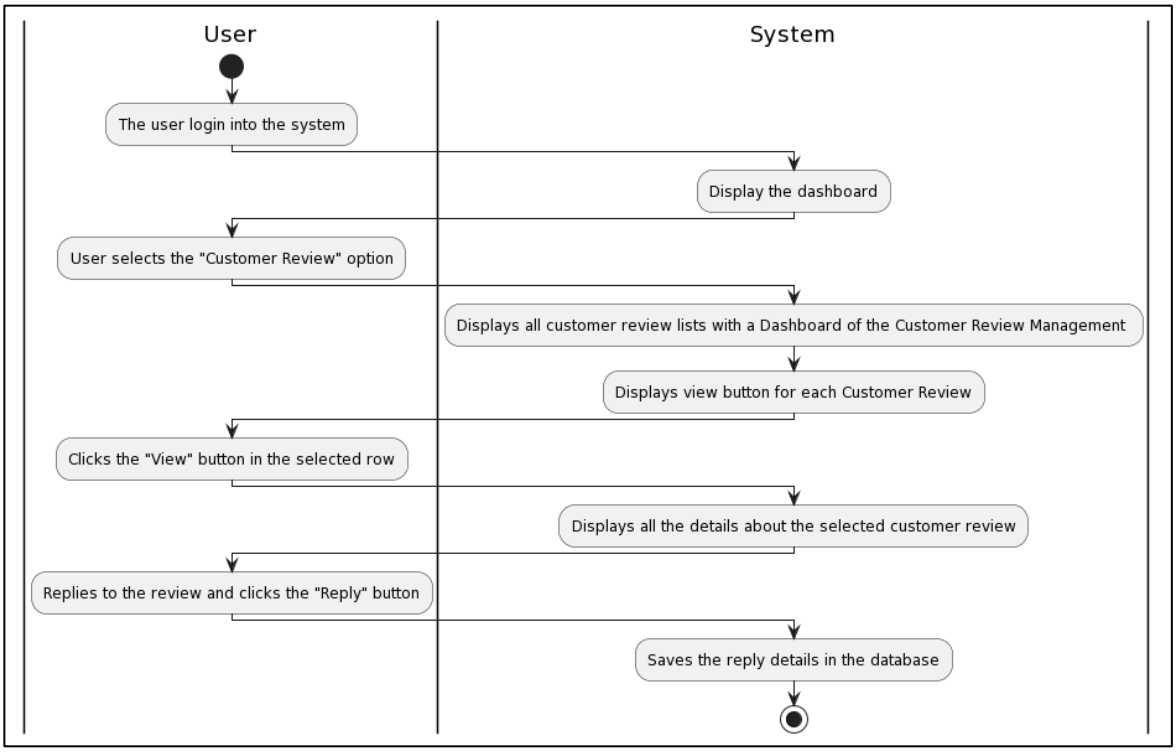


Figure 3. 56 Activity Diagram of Reply to Customer Reviews

Activity Diagram of Hall Arrangement

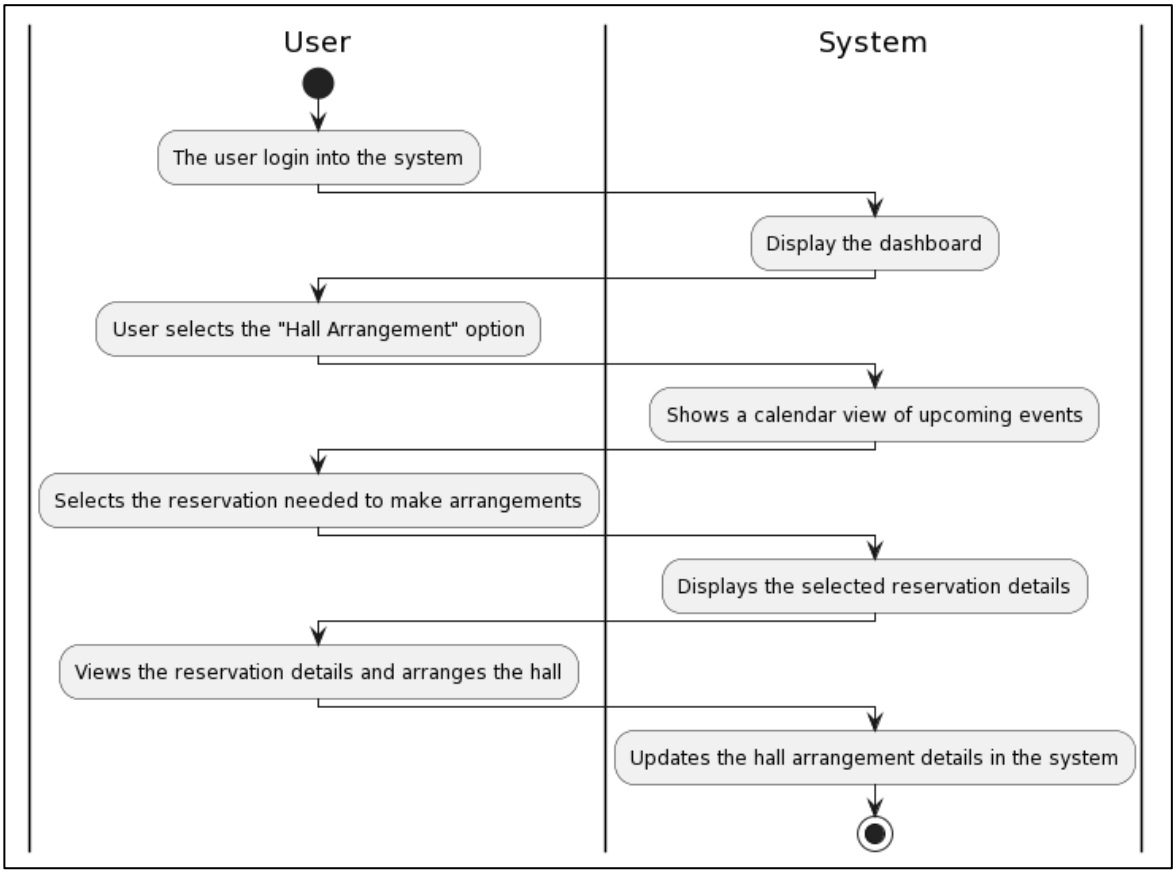


Figure 3. 57 Activity Diagram of Hall Arrangement

Activity Diagram of View Reports

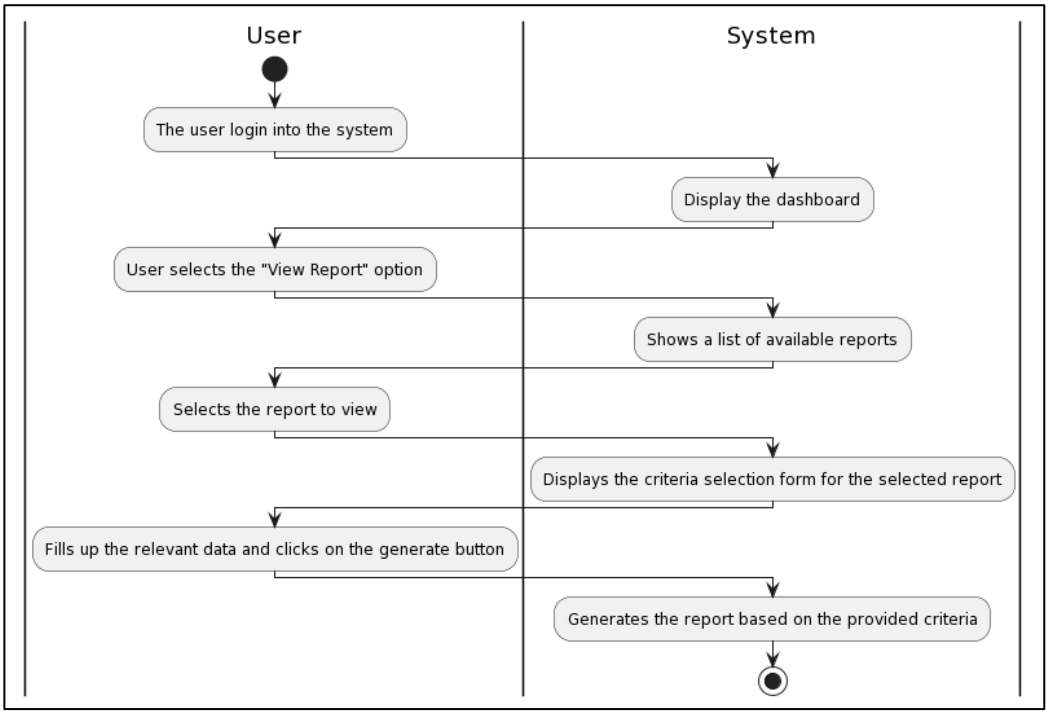


Figure 3. 58 Activity Diagram of View Reports

Activity Diagram of Add New Employee

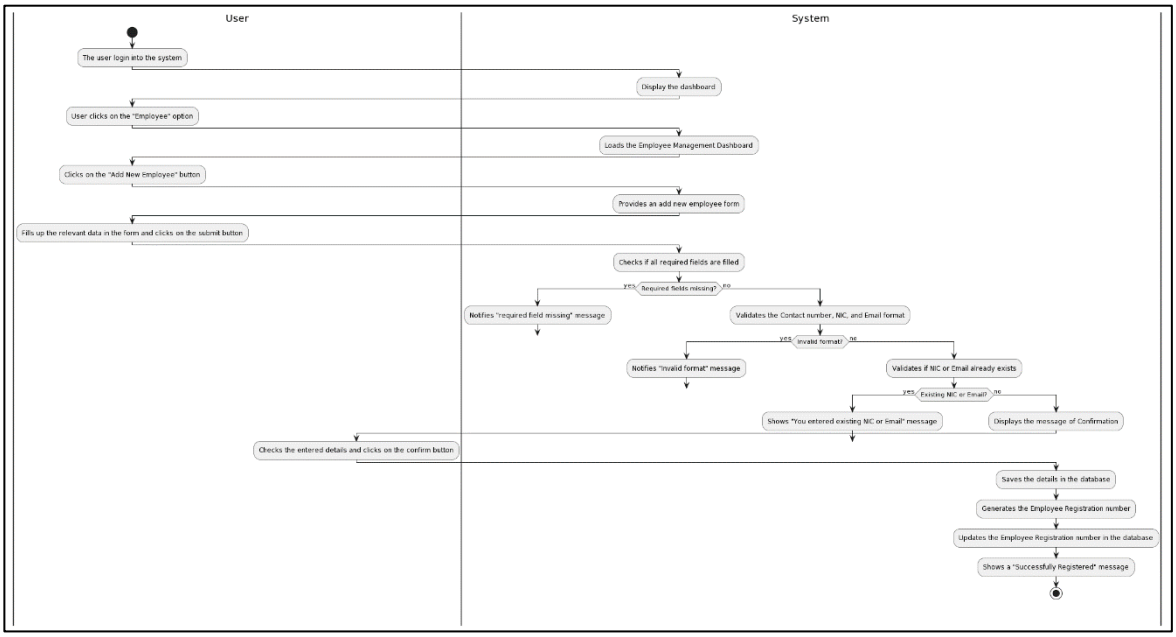


Figure 3. 59 Activity Diagram of Add New Employee

Activity Diagram of View Employee Details

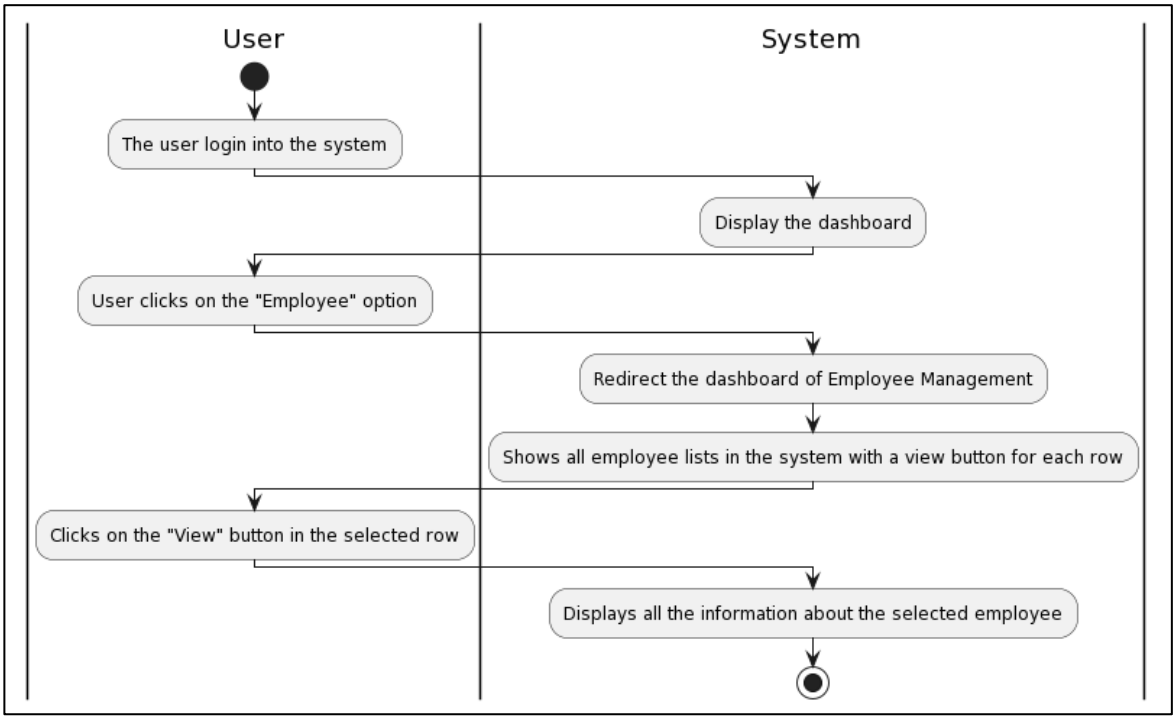


Figure 3. 60 Activity Diagram of View Employee Details

Activity Diagram of Update Employee

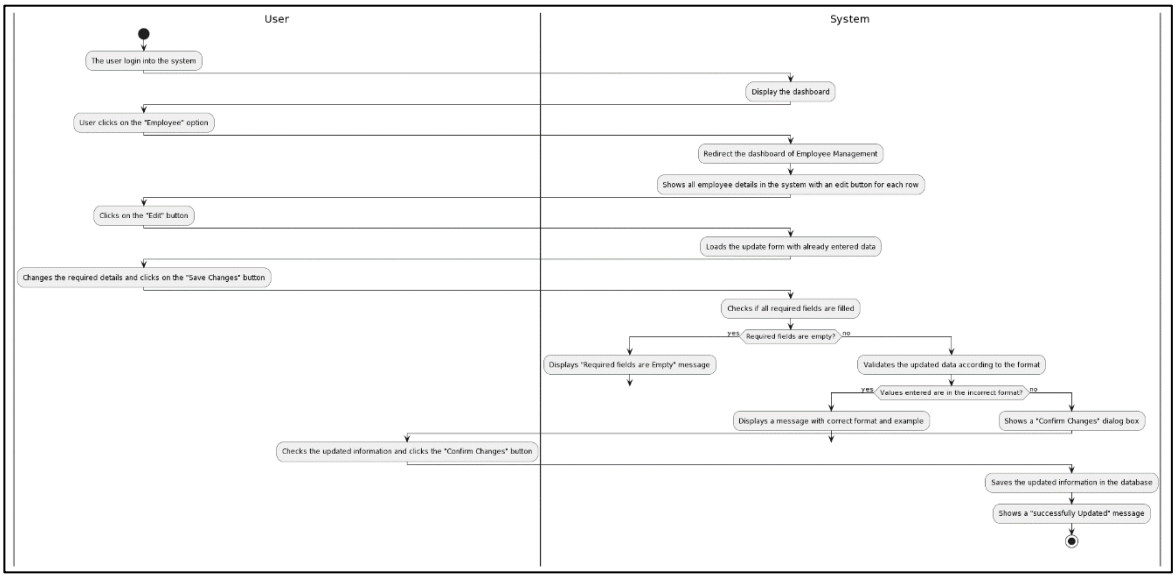


Figure 3. 61 Activity Diagram of Update Employee

Activity Diagram of Add New User

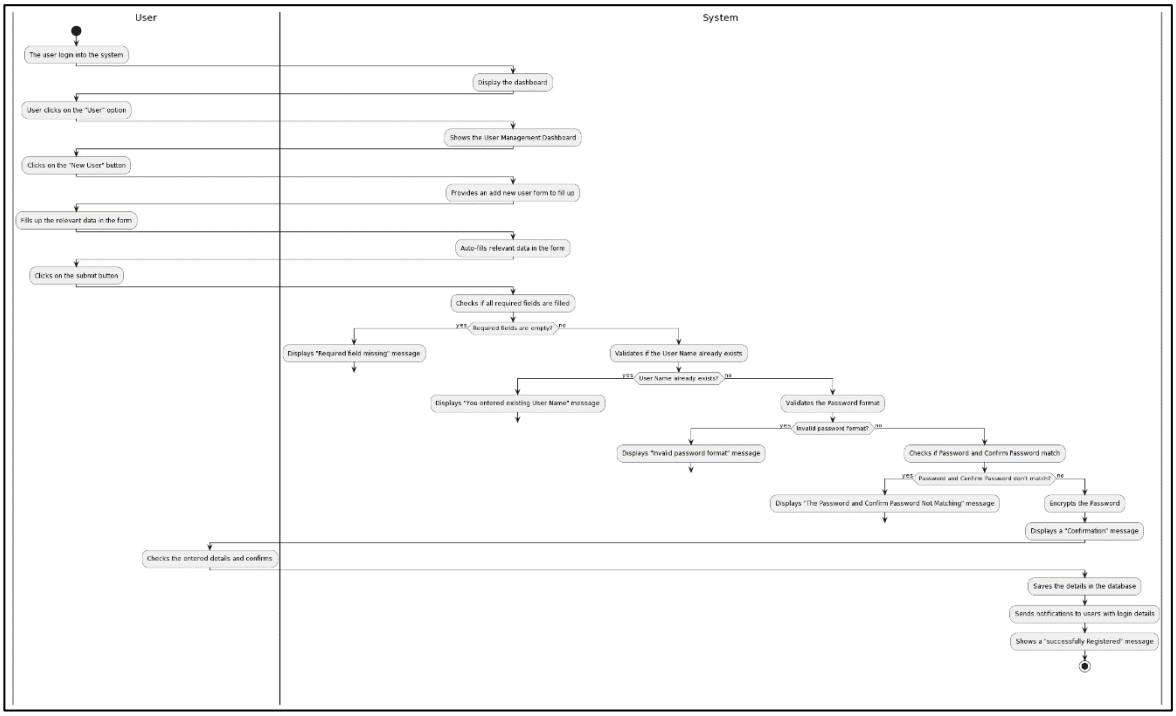


Figure 3. 62 Activity Diagram of Add New User

Activity Diagram of View User Account

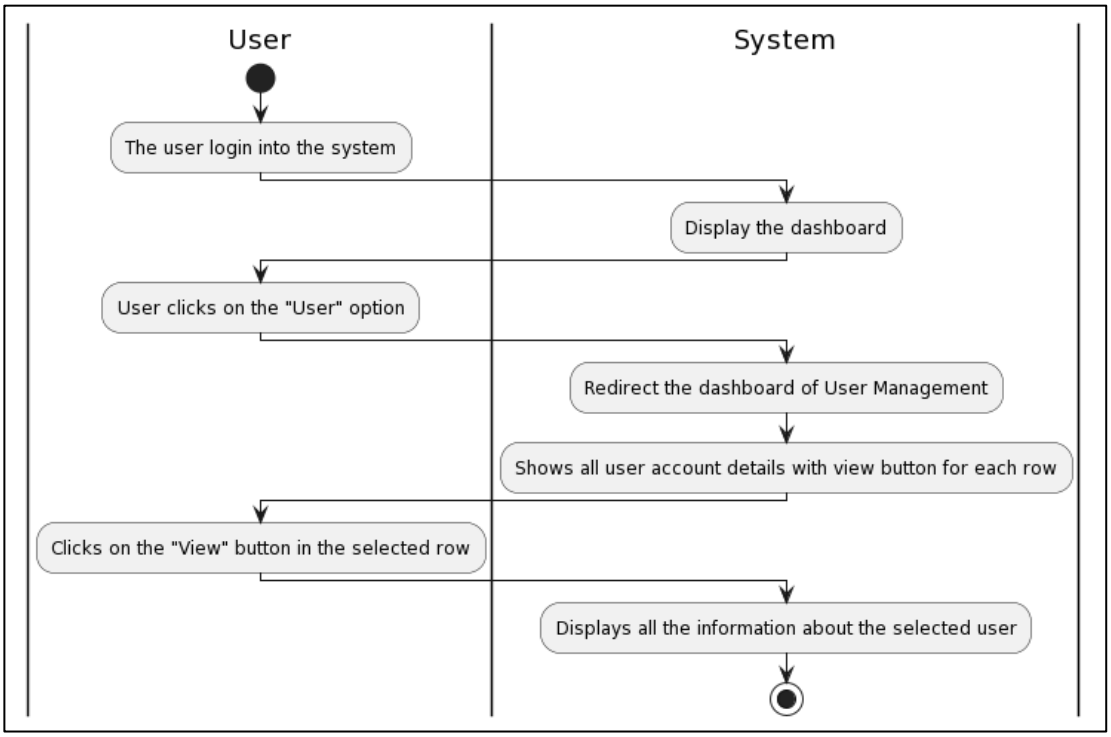


Figure 3. 63 Activity Diagram of View User Account

Activity Diagram of Update User Account

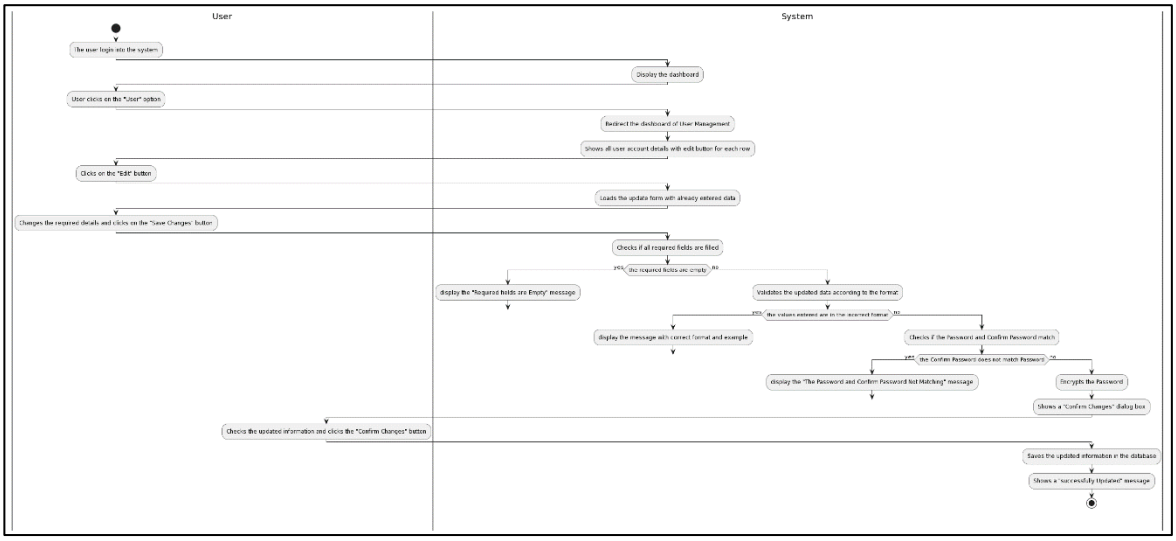


Figure 3. 64 Activity Diagram of Update User Account

3.1.6 Entity Relationship Diagram (ER)

The different entities involved in the system would be shown in an ER (Entity-Relationship) diagram. The relationships between these entities would be illustrated by the diagram. The data flow and relationships in the system would be visually represented by the ER diagram, which is useful for designing and understanding the system's functionality. The proposed system's overall ER diagram is displayed below.

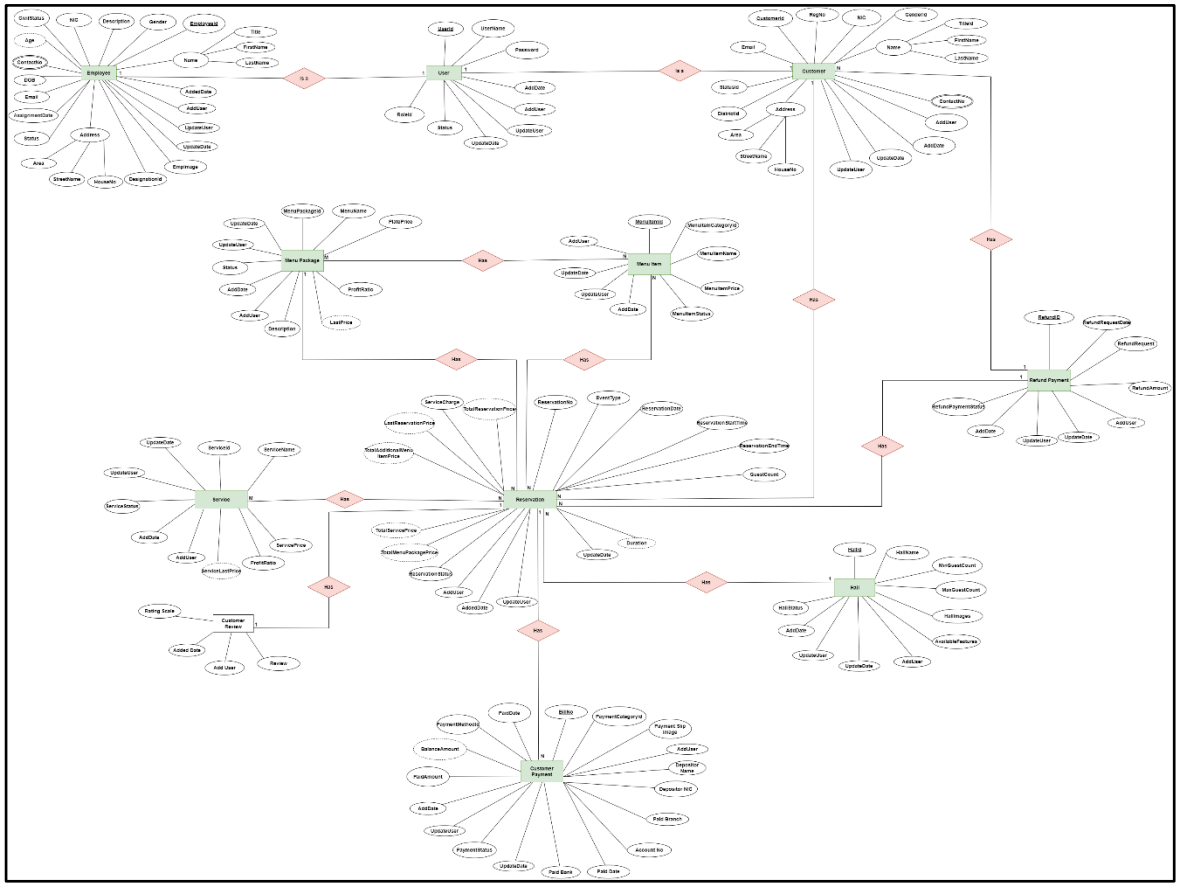


Figure 3. 65 Overall ER Diagram

3.2. User interface design with Wireframes

Creating wireframes or layouts to test the interface with users and gather feedback for improvement might be involved in the user interface design. The User interface design's objective ultimately to create an interface that is easy and enjoyable for users to use, which can help to increase user adoption and satisfaction with the system. Some of the wireframes of proposed system is shown below.

A basic visual representation or sketch of the layout and elements of a web page or application screen where customers can sign up or create an account is provided by a customer registration form wireframe. The placement and design of key components such as input fields, labels, buttons, and any additional instructions are outlined by it. The actual registration form is created by designers and developers using the wireframe as a blueprint, demonstrating how it will be organized and what information will be collected from users during the registration process.

Figure 3. 66 Register as New Customer UI

The check hall availability form wireframe is a basic visual representation or sketch that displays the layout and elements of an online form where inquiries about the availability of a hall or venue for an event can be made by users. The placement and design of fields where details such as event date, time, and any specific requirements can be input by users are outlined. The actual form will be created by designers and developers using this wireframe as a guide, demonstrating how it will be structured and what information can be submitted by users to check the availability of the hall.

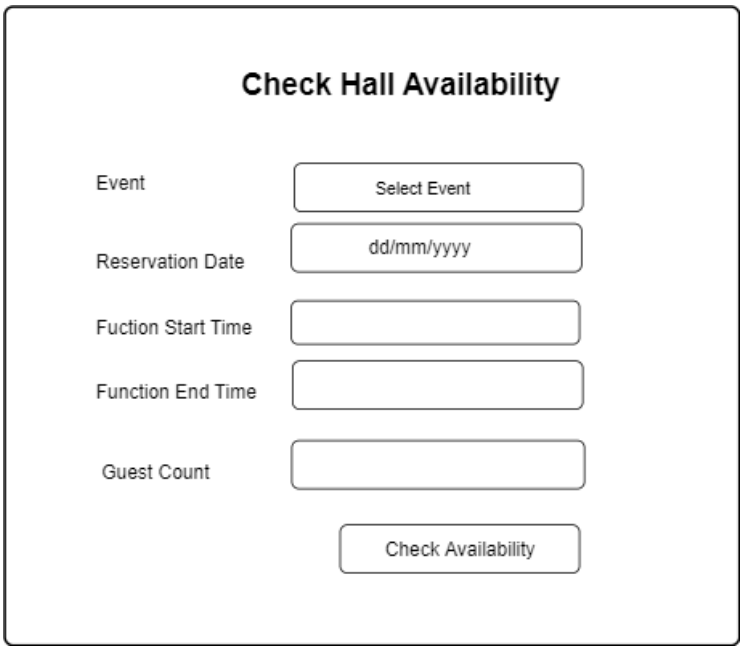


Figure 3. 67 Check Hall Availability UI

A wireframe for a new reservation form is a basic visual blueprint or sketch outlining the layout and elements of an online form where a fresh reservation or booking can be created by users. The placement of fields where information such as date, time, preferences, and contact details for the reservation can be input by users is illustrated. The actual reservation form will be created by designers and developers using this wireframe as a guide, demonstrating how it will be organized and what information can be provided by users to make a new booking.

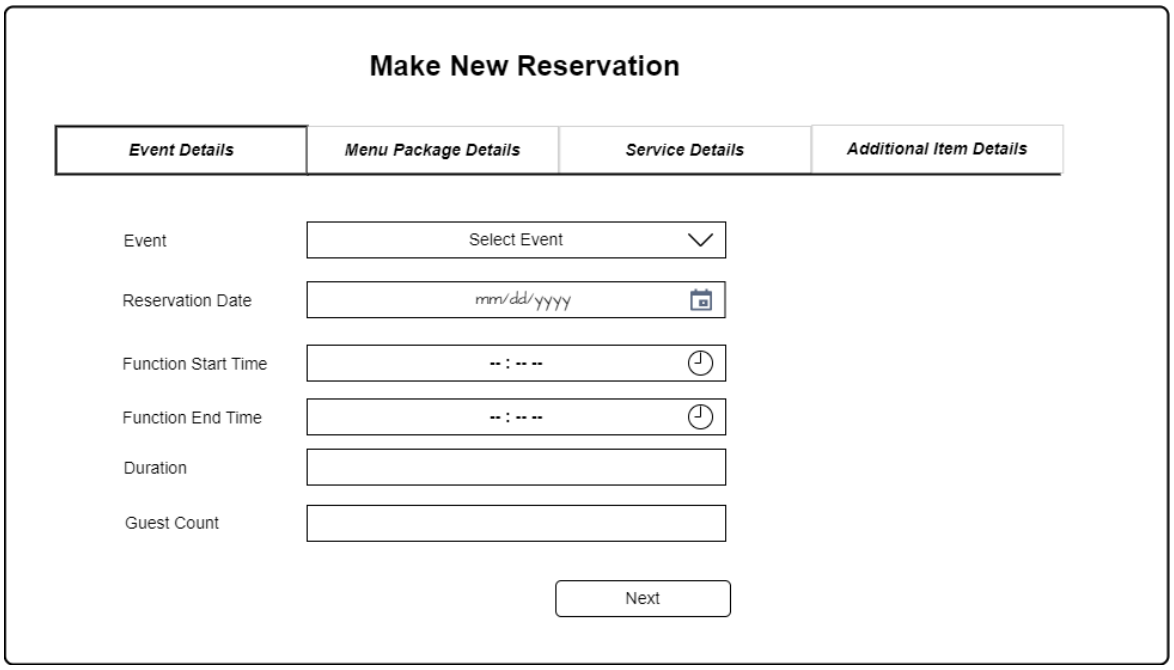


Figure 3. 68 Event Details of Make New Reservation UI

Figure 3. 69 Menu Package Details of Make New Reservation UI

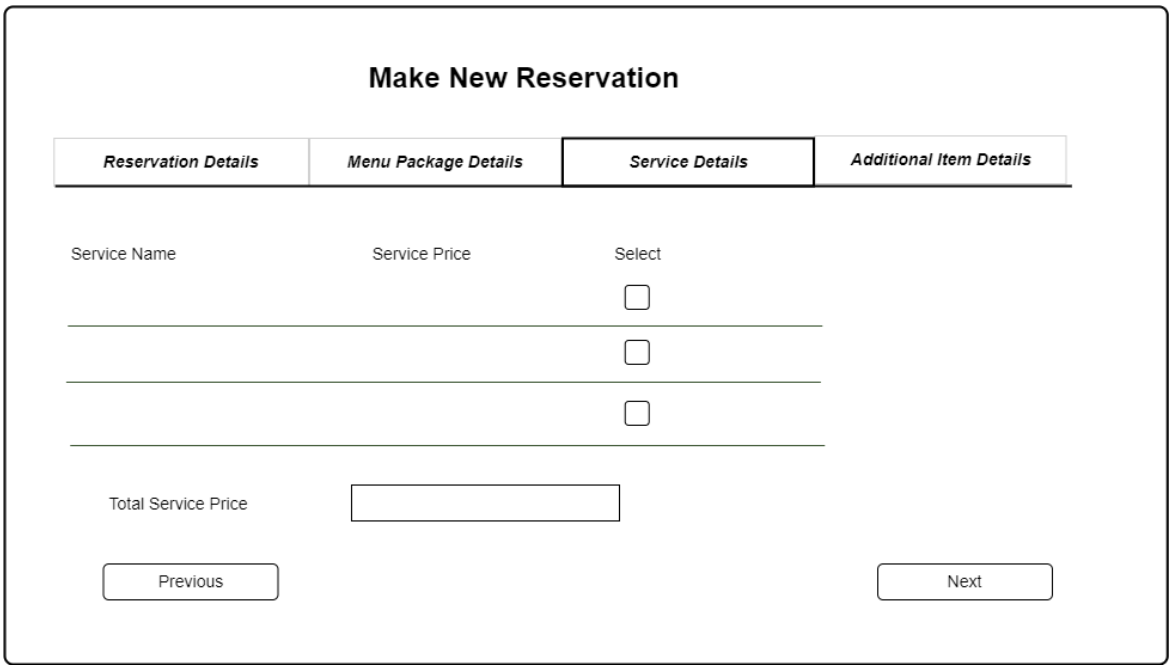


Figure 3. 70 Service Details of Make New Reservation UI

Figure 3. 71 Additional Menu Item Details of Make New Reservation UI

A make payment form wireframe' is a basic visual representation or sketch where the layout and elements of an online form for entering information to make a new payment are outlined. The arrangement of fields for inputting details such as payment amount, payment method, card information, and billing address is shown. This wireframe is used by designers and developers as a guide to create the actual payment form, demonstrating how it will be structured and what information needs to be provided by users in order to complete a payment transaction.

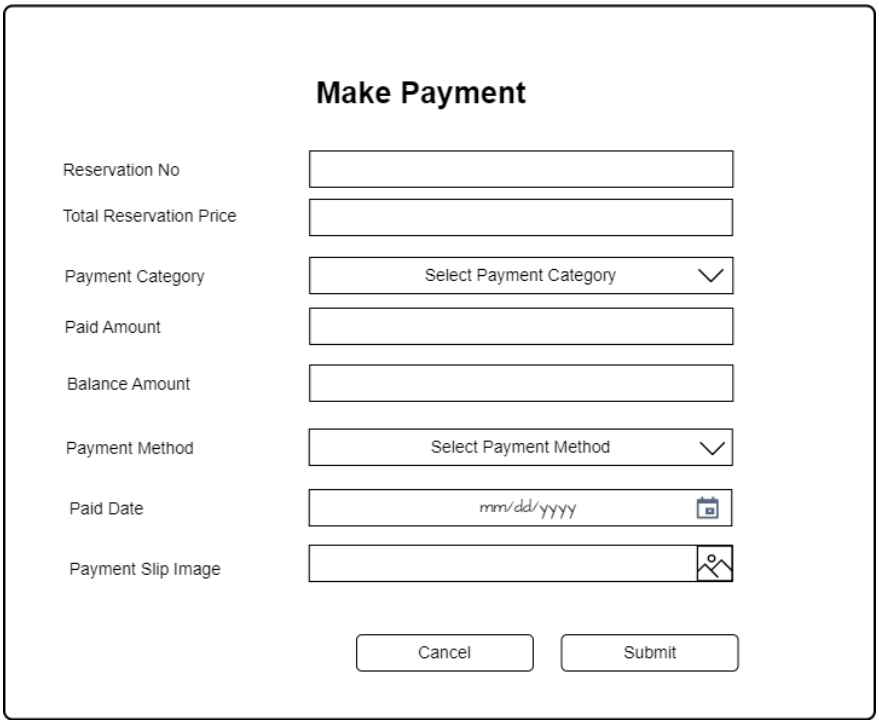


Figure 3. 72 Make Payment UI

A new employee addition form wireframe' is a basic visual sketch or blueprint where the layout and elements of an online form for adding a new employee to a company's records can be outlined by administrators or HR personnel. Fields for entering details like the employee's name, contact information, job title, and other relevant data are typically displayed. This wireframe is used by designers and developers as a guide to create the actual form, showing how it will be organized and what information must be collected to onboard a new employee into the company's system.

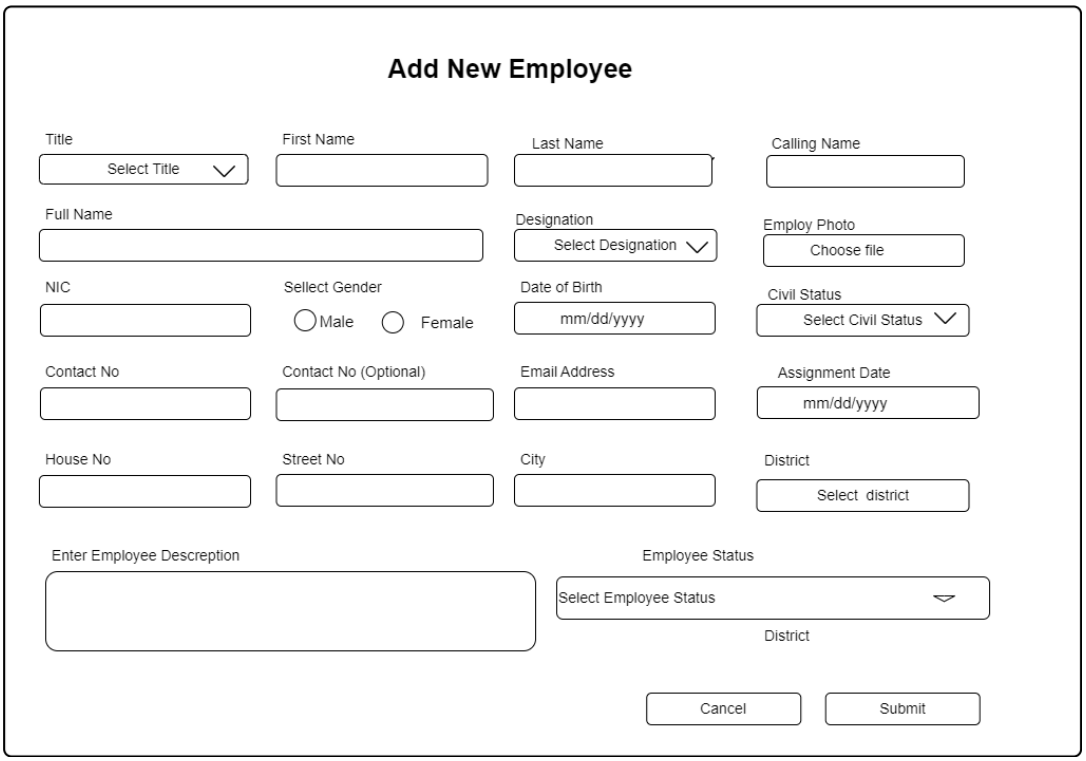


Figure 3. 73 Add New Employee UI

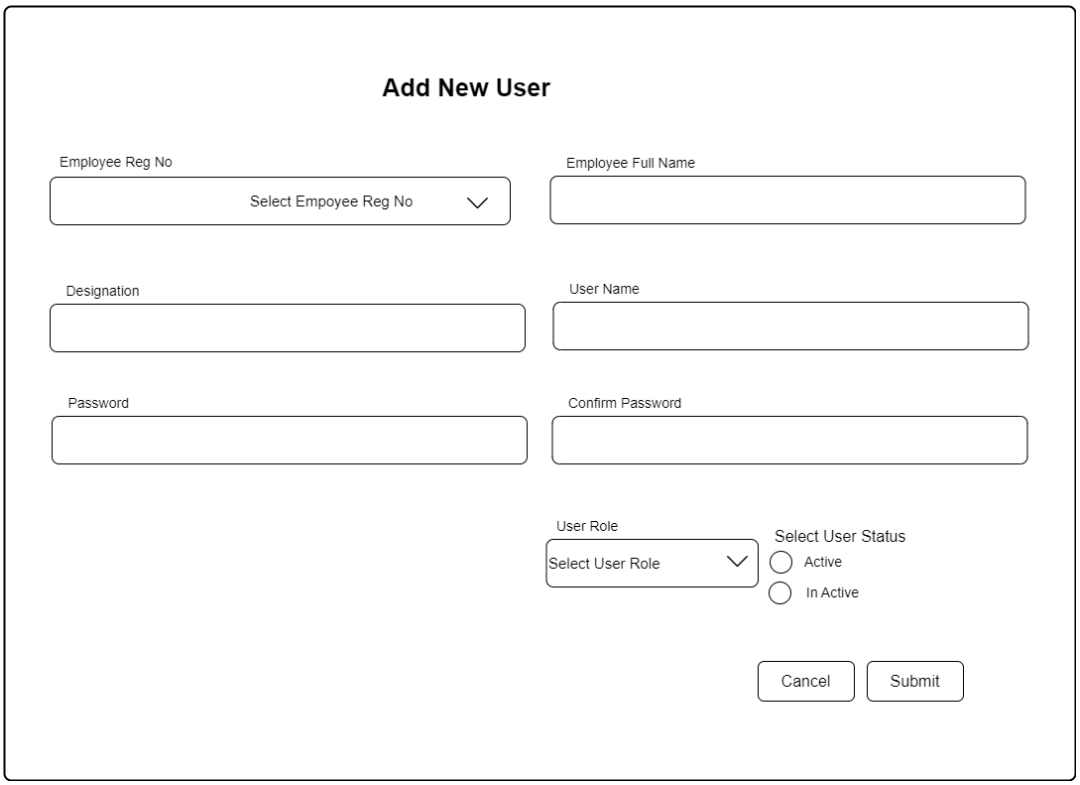


Figure 3. 74 Add New User

Chapter 4 – Implementation

In the implementation phase, the needs that were determined in the analysis chapter are realized utilizing the designs from the previous chapter and the appropriate software and hardware.

4.1. Implementation Environment

Software and hardware are the two key components that make up the implementation environment.

4.1.1. Software Environment

The Nectar Mount Resort's web-based hotel reservation system was created utilizing a mix of technologies, including HTML, CSS, and Bootstrap 5 for the system's design and styling, PHP for server-side scripting, JS for dynamic and responsive elements, and NetBeans as the integrated development environment (IDE). The system was locally hosted using the XAMPP server throughout the development period, and MySQL was used as the DBMS.

1. XAMPP Server - A popular and user-friendly web server solution known as XAMPP, which is represented by "X" (cross-platform), "Apache," "MySQL," "PHP," and "Perl," is employed. Essential web development tools, such as an Apache web server, MySQL database management system, PHP scripting language, and Perl programming language, are bundled together by it. This results in the creation of an environment for setting up a local web development environment on their computer by developers, which is made easy. The process of creating, testing, and debugging web applications is simplified by XAMPP, rendering it an invaluable tool for both beginners and experienced developers.

2. PHP - "Hypertext Preprocessor," abbreviated as PHP, is a server-side scripting language widely used for web development. Within HTML code, it is embedded and executed on the web server, enabling the creation of dynamic and interactive web applications by developers. Its popularity for building websites, web applications, and content management systems is attributed to its flexibility and ease of use. Tasks such as database connectivity, form processing, and session management can be handled by PHP, rendering it an essential tool for the creation of dynamic and data-driven websites by web developers.

3. MySQL - With A widely-used and robust open-source relational database management system (RDBMS) called MySQL is utilized. It is renowned for its efficiency, scalability, and versatility, which have resulted in its popularity as a choice for managing data in various applications. Users are allowed by MySQL to create, retrieve, update, and manipulate data that is stored in structured tables. This makes it a fundamental tool for storing and retrieving information in web applications, business systems, and more. Its flexibility, speed, and reliability have caused it to become a preferred option for efficient data management and storage solutions, both by developers and organizations.

4. HTML - HTML5, which is short for HyperText Markup Language 5, is the latest iteration of the standard language used for creating and structuring content on the World Wide Web. A range of new features and enhancements is introduced by it, resulting in its recognition as a powerful tool for web developers. With HTML5, multimedia elements like audio and video can be embedded directly into web pages without the need for plugins. Improved support for creating interactive web applications is also provided by it, including features such as canvas for drawing graphics, local storage for saving data on the user's device, and enhanced form controls for enhancing user experiences. HTML5's modern and efficient design has led to its establishment as a cornerstone of contemporary web development, enabling the creation of dynamic and responsive websites across various devices and platforms.

5. CSS - A fundamental technology for web design is CSS, also known as Cascading Style Sheets. The styling language for HTML documents is served as by it, allowing the control of the layout, colors, fonts, and overall presentation of web pages by web developers. CSS operates by defining rules that specify how different elements on a web page should be displayed. This separation of content (HTML) and presentation (CSS) results in web development being made more efficient and flexible. With CSS, visually appealing and responsive websites that adapt to different screen sizes and devices can be created by designers, providing a better user experience across the web. It's an essential tool for modern web design, enabling the customization and beautification of online content.

6. Bootstrap - A popular front-end framework for web development known as Bootstrap is utilized, simplifying the process of creating responsive and visually appealing websites and web applications. It was developed by Twitter and provides a collection of pre-designed,

reusable components, including buttons, navigation bars, forms, and grids, which can be easily integrated into web projects. Bootstrap's grid system, in particular, is utilized by developers to create flexible and responsive layouts that adapt seamlessly to various screen sizes and devices, making it a preferred choice for building mobile-friendly websites. With its extensive documentation and large community of users, Bootstrap is considered a powerful tool for both beginners and experienced developers.

7. NetBeans IDE - A versatile and user-friendly integrated development environment known as NetBeans IDE is utilized for simplifying software development across various programming languages. A comprehensive set of tools for creating, editing, testing, and debugging code is provided by it, thereby rendering it an invaluable resource for developers. Various languages like Java, PHP, C++, and more are supported by NetBeans, resulting in a seamless development experience being offered. Productivity and flexibility for programmers are enhanced by its features, including code templates, version control, and a robust plugin ecosystem. Whether a beginner or an experienced developer, a convenient and efficient platform for software development is offered by NetBeans IDE.

Apart from the above-mentioned Technologies and environment, the following additional third-party plugins were used in the project,

- **PHP Mailer** - This plugin allowed the system to send emails while processing reservations, sending invoices, etc.
- **PDF Generator** - In order to export reports, invoices, and bills as PDF documents, the PDF Generator plugin was used.

4.1.2. Hardware Environment

The following describes the hardware setup used to create the web-based hotel reservation system.

- Intel(R) Core (TM) i5-7200U CPU @ 2.50GHz 2.70 GHz
- 12 GB RAM
- 1366 × 768 resolution display
- 1 TB Hard Disk Drive

4.2. Module Structure

The system is made up of a number of modules, including those for managing customers, reservations, menu packages, payments, services, menu items, halls, employees, and users. The component diagram below offers a thorough breakdown of the system's architecture by illuminating the interactions between the major parts.

The following figure 4.1 shows the component diagram of the proposed system.

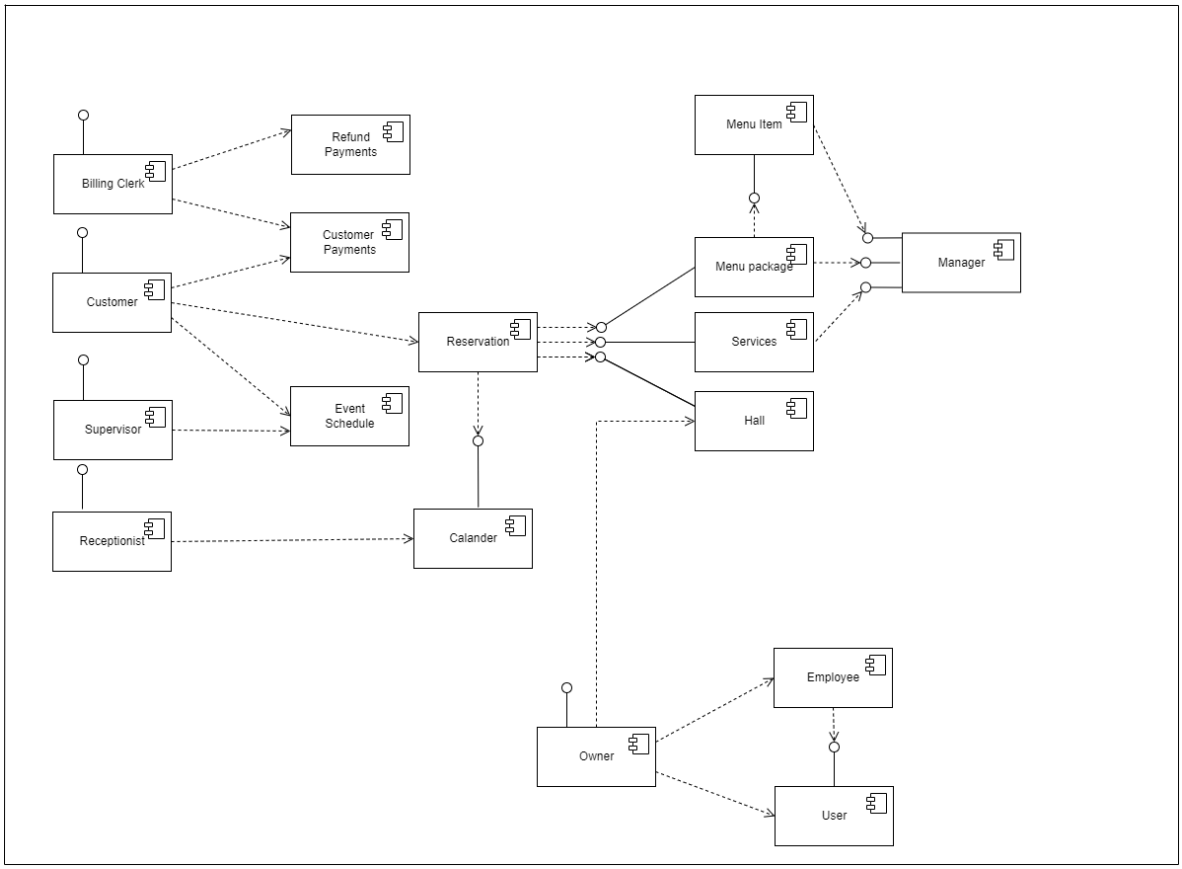


Figure 4. 1 Component Diagram for the proposed system

4.3. Important Code Segments

Important code segments of the web based hotel reservation system will be included here.

- **Login of System Users**

  The following figure 4.2. shows the login of the system.

```php
<?php
session_start();
include 'function.php';

//Check the Request Method
if ($_SERVER['REQUEST_METHOD'] == "POST") {

    extract($_POST);
    //var_dump($_POST);
    //Data Cleaning
    //Clean the username inputted and assign it to the UserName Variable
    $username = cleanInput($username);

    $message = array();
    //Required Field Validation
    if (empty($username)) {
        $message['error_username'] = "The Username Should not be Empty";
    }
    if (empty($password)) {
        $message['error_password'] = "The Password Should not be Empty";
    }
    //Authenticating the Credentials
    if (empty($message)) {
        //Convert password into sha1 encryption
        $password = sha1($password);
        //Query for retreiving data record with given credentials
        $sql = "SELECT * FROM users u"
                . " LEFT JOIN user_role r ON r.role_id = u.role_id"
                . " LEFT JOIN employee e ON e.user_id = u.user_id"
                . " WHERE u.username ='$username' AND u.password ='$password'";
        //creating the database connectivity
        $db = dbConn();
        //Executing the query
        $result = $db->query($sql);
        //Check for having matching records
        if ($result->num_rows <= 0) {
            //Displaying the error message if not any record matched
            $message['error_login'] = "Invalid Username or Password";
        } else {
            $row = $result->fetch_assoc();
            //Storing the relevant data into the session
            $_SESSION['userid'] = $row['user_id'];
            $_SESSION['employee_id'] = $row['employee_id'];
            $_SESSION['title'] = $row['title'];
            $_SESSION['first_name'] = $row['first_name'];
            $_SESSION['last_name'] = $row['last_name'];
            $_SESSION['user_role'] = $row['role_name'];

            //Redireting to the Dashboard of Authenticated User
            header("Location:index.php");
        }
    }
}
?>
```

Figure 4. 2 Login of the System

- **Customer Registration**

The following figure 4.3. shows the NIC validation of the Customer Registration form.

```
        if (empty($nic)) {
            $message['error_nic'] = "NIC should not be blank..!";
        } elseif (nicValidation($nic)) {
            $message['error_nic'] = "Invalid Nic Format";
        } else {
            $db = dbConn();
            $sql = "SELECT * FROM tbl_customers WHERE NIC='$nic'";
            $result = $db->query($sql);
            if ($result->num_rows > 0) {
                $message['error_nic'] = "This Nic is Already Exist";
            }
        }
```

Figure 4. 3 Code Segment of the Customer Registration of the system

- **Check Hall Availability**

The following figure 4.4. shows the Check Availability of the system.

```
<?php
if (!empty($resdate) && !empty($guest)) {
    $db = dbConn();
    $sql = "SELECT StartTime,EndTime FROM tbl_event_timeslot WHERE TimeSlotId=$eventmode";
    $result = $db->query($sql);

    if ($result->num_rows > 0) {
        while ($row = mysqli_fetch_assoc($result)) {
            $stime = $row['StartTime'];
            $endtime = $row['EndTime'];
        }
    }
    $sql = "SELECT * FROM tbl_hall WHERE HallStatus='Available' AND MinGuestCount<='$guest' AND MaxGuestCount>='$guest' "
            . "AND HallId NOT IN (SELECT HallId FROM tbl_reservation "
            . "WHERE ReservationStatusId=2 AND ReservationDate='$resdate' AND (FunctionStartTime='$stime' OR "
            . "FunctionEndTime='$stime' OR FunctionStartTime='$endtime' "
            . "OR FunctionEndTime='$endtime' OR (FunctionStartTime BETWEEN '$stime' AND '$endtime') "
            . "OR (FunctionEndTime BETWEEN '$stime' AND '$endtime') "
            . "OR (FunctionStartTime<'$stime' AND FunctionEndTime>'$endtime')) )";
    //   print_r($sql);
    $result = $db->query($sql);
    if ($result->num_rows > 0) {
        ?>
        <?php
        while ($row = mysqli_fetch_assoc($result)) {
            ?>
            <div class="col-md-4">
                <div class='alert bg-white' role='alert'>
                    <h5 class="text-danger"> Hall <?= $row['HallName'] ?> - Available</h5>
```

Figure 4. 4 Code Segment of the Check Availability

- **Make New Reservation**

The following figure 4.5. shows the Make New Reservation of the system.

```php
<?php ob_start(); ?>
<?php
include '../dashboardheader.php';
include '../dashboardsidebar.php';
?>
<main id="main" class="col-md-10 ms-sm-auto col-lg-11 px-md-4">
    <section>
    <div class="row">
    <div class="col-md-2"></div>
        <div class="col-md-10 d-flex justify-content-between flex-wrap flex-md-nowrap align-items-center pt-3 pb-2 mb-3 border-bottom">
                <h1 class="h2">Make New Reservation</h1>
         </div>
    </div>
        <br>
        <?php
        // Set input field values from check availability
        if($_SERVER['REQUEST_METHOD'] == "GET"){

        if (!empty($_SESSION['checkavailability'])) {
            $event = $_SESSION['checkavailability']['event'];
            $resdate = $_SESSION['checkavailability']['resdate'];
            $eventmode = $_SESSION['checkavailability']['eventmode'];
            $hall = $_SESSION['checkavailability']['hall'];
            $guest = $_SESSION['checkavailability']['guest'];
        }
        }
```

Figure 4. 5 Code Segment of the Make New Reservation

4.4. Reuse of the Existing Codes

Reusing existing code is a key component of software development projects because it enables programmers to use pre-existing components and solutions to expedite development and improve overall effectiveness.

Here are some examples of existing codes being reused in this project.

1. Function for Clean Inputs

This function is intended to remove extra spaces and characters that the user unintentionally entered into the database. The initial step in data processing once a user submits a form is to clean up the input for each module. Therefore, the "cleanInput(String)" function can be invoked by supplying the input value as a string parameter rather than having to write it each time.

The Clean Input Function of the system is depicted in figure 4.6 below.

```php
//My 1st Function - Clean the Input Data Before Saving into the Database
function cleanInput($input=null){
    return htmlspecialchars(stripslashes(trim($input)));
}
```

Figure 4. 6 Code Segment of the Clean Input Function

2. Function for Create Database Connectivity

The connection to the database is established using this function. We must establish a database connection before retrieving data from the database, putting it in the interfaces, changing it, or even loading it dynamically. In this manner, creating a database connection is simple and only requires invoking "dbConn ()".

The system's database connectivity is shown in figure 4.7 below.

```php
function dbConn() {
    $servername = "localhost";
    $username = "root";
    $password = "";
    $dbname = "nectar_mount_resort";

    $conn = new mysqli($servername, $username, $password, $dbname);

    if ($conn->connect_error) {
        die("Database Coneection Error " . $conn->connect_error);
    } else {
        return $conn;
    }
}
```

Figure 4. 7 Code Segment of the Database Connectivity

3. Function for Upload Images

The system is required to provide the feature to upload images of employees, menu items, halls, and events. When there is an image upload, it is convenient to utilize the following code as a reusable function rather than writing it out every time. This might be changed and improved further to allow uploading of additional file formats in addition to photos. For the time being, this satisfies the requirements.

The system's function for uploading images is shown in figure 4.8 below.

```php
if (empty($message)) {
    $MenuItemImage = $_FILES['itemimage'];
    $filename = $MenuItemImage['name'];
    $filetmpname = $MenuItemImage['tmp_name'];
    $filesize = $MenuItemImage['size'];
    $fileerror = $MenuItemImage['error'];
    $fileext = explode(".", $filename);
    $fileext = strtolower(end($fileext));
    $allowedext = array("jpg", "jpeg", "png", "gif");

    if (in_array($fileext, $allowedext)) {

        if ($fileerror === 0) {
            if ($filesize <= 2097152) {
                $file_name_new = uniqid("", true) . "." . $fileext;
                $file_destination = "../assets/images/menuitem/" . $file_name_new;
                if (move_uploaded_file($filetmpname, $file_destination)) {
                    echo "The file was uploaded successfully.";
                } else {
                    $message['error_file'] = "There was an error uploading the file.";
                }
            } else {
                $message['error_file'] = "This File is Invalid ...!";
            }
        } else {
            $message['error_file'] = "This File has Error ...!";
        }
    } else {
        $message['error_file'] = "This File Type not Allowed...!";
    }
}
```

Figure 4. 8 Code Segment of the Upload Image

4. Function for Validate a Password

This function's purpose is to confirm that the password is sufficient to protect the data. This function should be called to determine the password strength whenever a user registers with the system, attempts to reset a forgotten password, or changes a password through their profile.

The function of the system's password validation is shown in figure 4.9 below.

```php
if (empty($pwd)) {
    $message['error_pwd'] = "Password should not be blank..!";
} elseif (strlen($pwd) < 8) {
    $message['error_pwd'] = "Password must be at least 8 characters long..!";
} elseif (!preg_match('/[A-Z]/', $pwd) || !preg_match('/[a-z]/', $pwd) || !preg_match('/[0-9]/', $pwd) || !preg_match('/[^A-Za-z0-9]/',$pwd)){
    $message['error_pwd'] = "Password must contain at least one uppercase letter,one lowercase letter, one digit, and one special character..!";
} elseif ($pwd == $username) {
    $message['error_pwd'] = "Password must not be the same as the username..!";
} elseif (strpos($pwd, ' ') != false) {
    $message['error_pwd'] = "Password should not contain spaces";
}
```

Figure 4. 9 Code Segment of the Validate Password

5. Function for Validate a Text Input

This function will be used to ensure that users don't accidentally insert meaningless data into the database by entering special characters or numerical values when only text values are allowed.

The system's function to validate a text input is depicted in figure 4.10 below.

```
function textFieldValidation($text) {

    $pattern = '/^[A-Za-z\s]+$/'; // Pattern to validate 10 digit number starting with 0

    if (!preg_match($pattern, $text)) {
        return true;
    }
}
```

Figure 4. 10 Code Segment of the Validate a Text Input

6. Function for Validating a Contact Number

This function will check that the contact numbers users enter through forms are identical and have the proper formatting.

The System's Function for Validating a Contact Number is depicted in the following figure 4.11.

```
function contactNoValidation($contact) {

    $pattern = '/^0[0-9]{9}$/'; // Pattern to validate 10 digit number starting with 0

    if (!preg_match($pattern, $contact)) {
        return true;
    }
}
```

Figure 4. 11 Code Segment of the Contact Number Validation

7. Function for Validate an Email

This function verifies that an email entered by a user is formatted correctly.

The system's function to validate an email is depicted in figure 4.12 below.

.

```
function emailValidation($email) {

    if (filter_var($email, FILTER_VALIDATE_EMAIL) == false) {
        return true;
    }
}
```

Figure 4. 12 Code Segment of the Validate an Email

8. Function for Identifying the Updated Fields

This was primarily created to produce the success message while updating the system's existing data. The fields that are updated during the procedure will be checked.

The Function for Identifying the Updated Fields of the System is shown in figure 4.13 below.

```
// get updated field names
function updatedFields($olddata ,$newdata){
    // $olddata = array with existing record details in the database
    // $newdata = array with submitted details in the form after updat

    // create an array for store updated field names
    $updated = array();

    // ex: $olddata as title=>Mr
    foreach ($olddata as $oldfield=>$oldvalue){
        // ex: $newdata as $title=>Miss
        foreach ($newdata as $newfield=>$newvalue){
            // ex: $title==$title && Mr!=Miss
            if($oldfield==$newfield && $oldvalue!=$newvalue){

                // assign updated field name ex: $updated[0]="title";
                $updated[]=$oldfield;
                break;
            }
        }
    }
   return $updated;
}
```

Figure 4. 13 Code Segment of the Get Update fields Function

4.5. Implementation of the Design

System was implemented into place based on the concepts created in the previous chapter. The procedure is illustrated in the examples that follow.

The system's implemented user interface is depicted in figure 4.14 in accordance with the system's wireframe shown in figure 3.74 above.

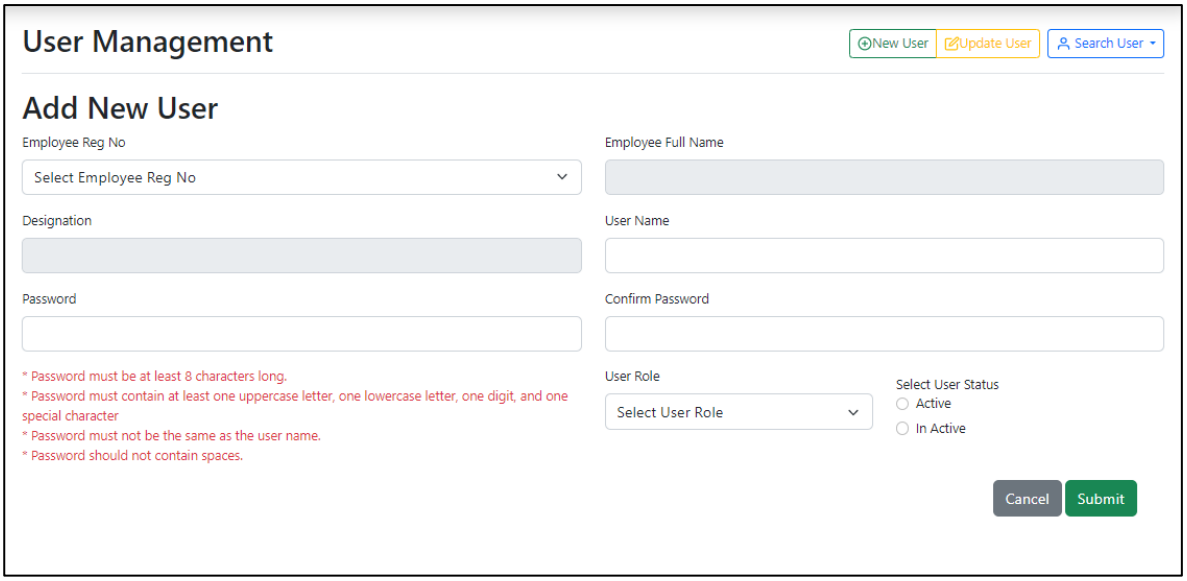


Figure 4. 14 User Interface of the Add New User Form

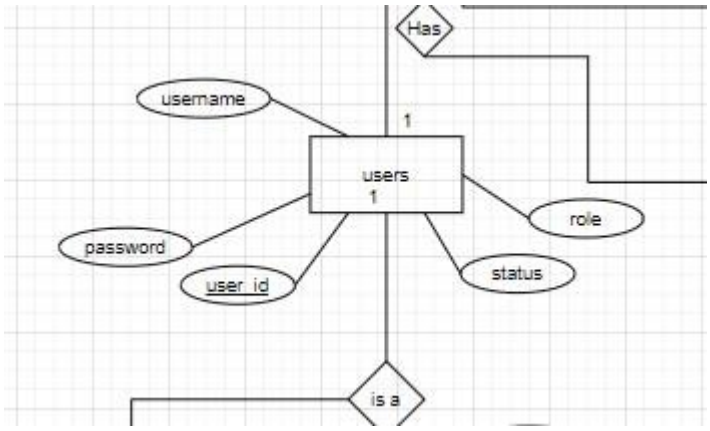
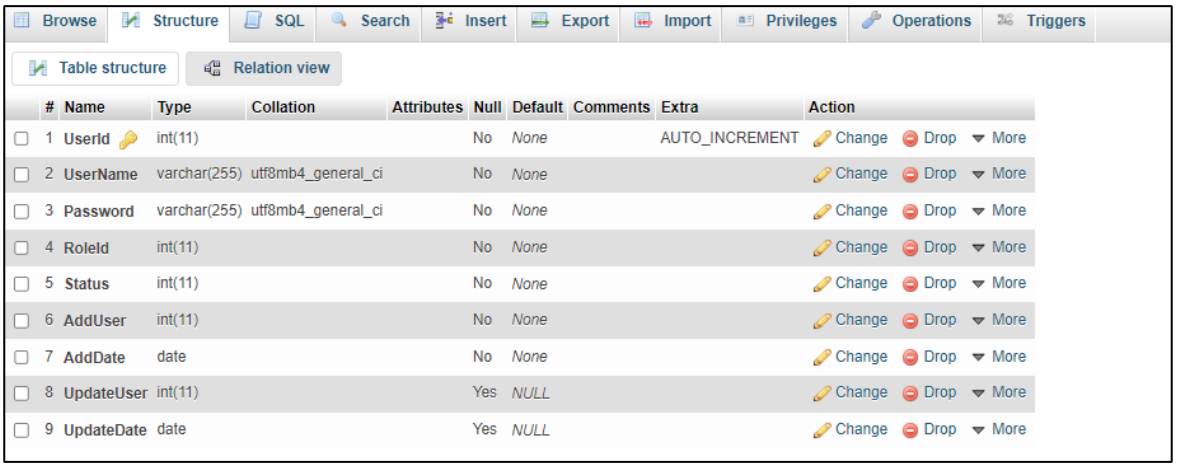


Figure 4. 15 User Entity in the ER



Browse | Structure | SQL | Search | Insert | Export | Import | Privileges | Operations | Triggers

Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	UserId	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	UserName	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
3	Password	varchar(255)	utf8mb4_general_ci		No	None			Change Drop More
4	RoleId	int(11)			No	None			Change Drop More
5	Status	int(11)			No	None			Change Drop More
6	AddUser	int(11)			No	None			Change Drop More
7	AddDate	date			No	None			Change Drop More
8	UpdateUser	int(11)			Yes	NULL			Change Drop More
9	UpdateDate	date			Yes	NULL			Change Drop More

Figure 4. 16 Database table of the User

4.6. Use of Best Practices

Here are a few of the coding best practices applied during the system's implementation.

1. Maintain Proper Line Indentations

The following code has every line appropriately indented. It is possible to do while inputting the codes, and the IDE also offers the option of a format to perform it automatically.

```
<form method="post" action="<?php echo htmlspecialchars($_SERVER['PHP_SELF']); ?>" >
    <div class="row">
        <div class="mb-3 col-md-3">
            <input type="text" class="form-control" placeholder="Name" name="cat_name" value="<?= @$cat_name ?>">
        </div>

        <div class="mb-3 col-md-3">
            <select name="availability" class="form-control form-select">
                <option value="">-Status-</option>
                <option value="1" <?php if( @$availability == '1'){ ?> selected <?php }?>>Available</option>
                <option value="0" <?php if( @$availability == '0'){ ?> selected <?php }?>>Unavailable</option>
            </select>
        </div>
        <div class="mb-3 col-md-3">
            <button type="submit" class="btn btn-warning">Search</button>
        </div>
        <div class="mb-3 col-md-3">
            <button type="reset" class="btn btn-info">Clear</button>
        </div>
    </div>
</form>
```

Figure 4. 17 Line Indentations in the Code

2. Consistent and Meaningful Naming Conventions

To improve the readability of the code, descriptive and meaningful names for variables and functions were chosen. The example that follows will show that it was adhered to throughout the coding process.

```php
<?php

// Data clean customized function
function cleanInput($input = null) {
    return htmlspecialchars(stripcslashes(trim($input)));
}
```

Figure 4. 18 Meaningful Naming Convention Practice Used in a Function

```php
// 3rd step- clean input
$title = cleanInput($title);
$fname = cleanInput($fname);
$lname = cleanInput($lname);
$nic = cleanInput($nic);
$district = cleanInput($district);
$houseno = cleanInput($houseno);
$streetname = cleanInput($streetname);
$area = cleanInput($area);
$contact = cleanInput($contact);
$contact2 = cleanInput($contact2);
$email = cleanInput($email);
```

Figure 4. 19 Naming Convention best practice in Defining Variable Names

3. Commenting

When appropriate, it has written brief and understandable comments that describe the function and goal of the code.

```php
<?php
session_start();
include 'function.php';

//Check the Request Method
if ($_SERVER['REQUEST_METHOD'] == "POST") {

    extract($_POST);
    //var_dump($_POST);
    //Data Cleaning
    //Clean the username inputted and assign it to the UserName Variable
    $username = cleanInput($username);

    $message = array();
    //Required Field Validation
    if (empty($username)) {
        $message['error_username'] = "The Username Should not be Empty";
    }
    if (empty($password)) {
        $message['error_password'] = "The Password Should not be Empty";
    }
    //Authenticating the Credentials
    if (empty($message)) {
        //Convert password into sha1 encryption
        $password = sha1($password);
        //Query for retreiving data record with given credentials
        $sql = "SELECT * FROM users u"
                . " LEFT JOIN user_role r ON r.role_id = u.role_id"
                . " LEFT JOIN employee e ON e.user_id = u.user_id"
                . " WHERE u.username ='$username' AND u.password ='$password'";
        //creating the database connectivity
        $db = dbConn();
        //Executing the query
        $result = $db->query($sql);
        //Check for having matching records
        if ($result->num_rows <= 0) {
            //Displaying the error message if not any record matched
            $message['error_login'] = "Invalid Username or Password";
        } else {
            $row = $result->fetch_assoc();
            //Storing the relevant data into the session
            $_SESSION['userid'] = $row['user_id'];
            $_SESSION['employee_id'] = $row['employee_id'];
            $_SESSION['title'] = $row['title'];
            $_SESSION['first_name'] = $row['first_name'];
            $_SESSION['last_name'] = $row['last_name'];
            $_SESSION['user_role'] = $row['role_name'];

            //Redireting to the Dashboard of Authenticated User
            header("Location:index.php");
        }
    }
}
?>
```

Figure 4. 20 Code Segment of Comment in login page

Chapter 5 - Evaluation

The produced system is tested as part of the assessment process to see whether it complies with customer requirements and to guarantee error-free operation. Tests are run using a variety of test cases created especially for distinct scenarios. It is crucial to make sure that the entire software is tested and that every component has been carefully checked.

5.1. Testing approaches

The Unit testing, integration testing, system testing, and acceptance testing are the four test stages that have been used to test the web-based hotel reservation system. Below is a brief explanation of each of the four levels [5].

1. Unit Testing - In the context of a web-based hotel reservation system, individual components or modules are tested in isolation as part of unit testing. For example, specific functions like customer registration, reservation, or payment processing can be tested. During unit testing, each component's correctness is validated by developers according to its design and requirements. This ensures that the fundamental building blocks of the system are functioning correctly as expected before integration.

2. Integration Testing - In the context of a web-based hotel reservation system, the focus of integration testing is on how different modules or components interact when combined. For instance, the interaction between the reservation module and the payment processing module might be tested to ensure that booked events are processed correctly for payment. Issues that may arise when different parts of the system work together, such as data flow problems or communication errors, are identified through integration testing.

3. System Testing - Testing the entire web-based hotel reservation system as a unified whole is the essence of system testing. This testing phase examines how all the integrated components collaborate to fulfill the system's functions. In this context, the end-to-end process of a user booking an event, including event scheduling, payment processing, and sending confirmation notifications, is assessed. System testing ensures that the specified requirements of the system are met, and it verifies that the system functions correctly from start to finish.

4. User Acceptance Testing - Confirmation that the web-based hotel reservation system aligns with the expectations and requirements of system users is the objective of acceptance testing. From the perspective of event organizers, customers, and administrators, the system is evaluated during this testing phase. Users validate that the system meets their needs, which may include booking event reception halls, managing events, and generating invoices. Successful completion of acceptance testing indicates that the system is prepared for deployment and actual use.

5.2. Test Plan

In a test plan, a group of circumstances or variables are specified. A tester then uses these conditions or variables to determine whether the system under test's requirements is satisfied or if it is functioning properly. It is possible to identify problems with an application's requirements or design by creating test cases. This all-inclusive strategy allows for a full examination of the system, the identification of any potential issues, and the implementation of any changes that are required to improve functioning and guarantee compliance with the standards.

User Login

Test No	Test Description	Expected Result	Status
1	Login form submission with blank fields	An error message will appear by displaying the error	Done
2	Enter an invalid username or password	An error message will appear by displaying the error	Done
3	Enter a valid username and password	Users can Log into the system	Done
4	Log in to the system with a valid username	Display logged username on the top status bar	Done
5	Click on “Logout” in the header section.	Redirect to the login page	Done

Table 5. 1 Test Plan for User Login

Register as Customer

Test No	Test Description	Expected Result	Status
1	Submit the form with empty fields.	An error message will display in every empty field.	Done
2	Enter an invalid Contact No	An error message will display entered invalid field.	Done
3	Enter an invalid NIC	An error message will display entered invalid field.	Done
4	Enter an invalid email	An error message will display entered invalid field.	Done
5	Enter an invalid Password	An error message will display entered invalid field.	Done
6	Enter the valid NIC	An error message will display entered fields already exist.	Done
7	Enter the valid email	An error message will display entered fields already exist.	Done
8	Enter the valid Username and Password	An error message will display entered fields already exist.	Done
9	Enter the Confirm Password.	An error message will display Confirm Password is not matching Password	Done
10	Click on the “Register” button.	Display success message.	Done

Table 5. 2 Test Plan for Register as Customer

Check Hall Availability

Test No	Test Description	Expected Result	Status
1	Submit the form with empty fields.	An error message will display in every empty field.	Done
2	Enter an invalid Guest count	An error message will display entered invalid field.	Done
3	Enter a valid Guest count	An error message will display guest count not in hall capacity.	Done
4	Submit the form with valid fields.	An error message will display hall is not available.	Done
5	Submit the form with valid fields.	Display available halls.	Done

Table 5. 3 Test Plan for Check Hall Availability

Make New Reservation

Test No	Test Description	Expected Result	Status
1	Submit the form with empty fields.	An error message will display in every empty field.	Done
2	Enter an invalid Qty of additional item details	An error message will display entered invalid field.	Done
3	Click on the “Save” button.	Display success message.	Done

Table 5. 4 Test Plan for Make New Reservation

5.3. User Evaluation

A document or online questionnaire, known as a user evaluation form, is designed for the collection of feedback and opinions from users regarding a product, service, or experience.

Typically, it incorporates questions about usability, functionality, design, and overall satisfaction.

The purpose of a user evaluation form is to amass valuable insights that can be utilized to enhance the quality and effectiveness of the product or service, relying on the perspectives and experiences of its users.

Here is a sample user evaluation form I used to gather overall feedback on the system.

User Evaluation of the System

Mark (√) inside the box, please provide a better answer with your truthful option.

Web-based Hotel Reservation System for Nectar Mount Resort					
Name of the User:		Role of the User:			
		Excellent	Good	Normal	Bad
01	User-friendliness of the system				
02	Overall attractiveness of the system				
03	Ability to understand error messages				
04	Ability to understand symbols of the system				
05	Easiness of data entry				
06	Responsiveness of the system				
07	Effectiveness of the functionalities				
08	Ease of Navigation throughout the system				
09	The usefulness of the reports				
10	Overall sense about the system				

.....

Signature **Thank you...!**

Figure 5. 1 User Evaluation form

I have provided the user evaluation form mentioned above to the staff at Nectar Mount Resort and gathered valuable feedback from our users. Subsequently, I meticulously analyzed and summarized the feedback, presenting the key insights in a visually engaging pie chart, as depicted below.

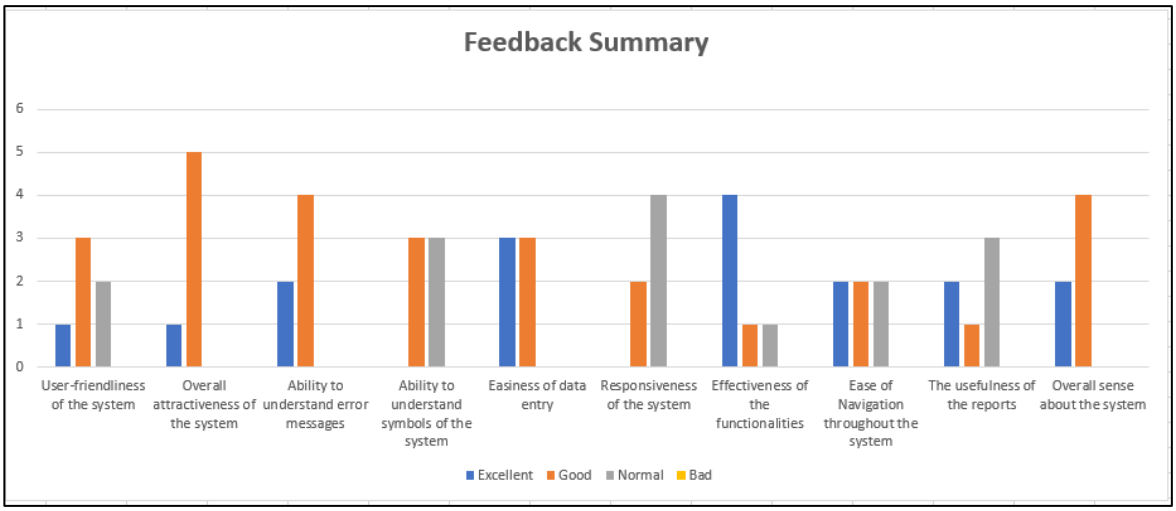


Figure 5. 2 User Feedback Summary

Chapter 6 - Conclusion

System reviews are important because they can help identify both problems with the system and its beneficial elements. Additionally, system reviews enable future developments with better understanding. This chapter mostly discusses the system's critical evaluation and the lessons discovered along the course of the project's development.

6.1. Critical evaluation of the system

The majority of the staff's needs have been successfully satisfied by the Nectar Mount Resort's web-based hotel reservation system. They have found the system's features to be very user-friendly, and they especially like how simple they are to use and how well they can present and summarize information. However, there have been several situations where specific system components, which depended on performing earlier tasks, caused handling complications. Despite these difficulties, it can be said that, when looking at the system as a whole, it has effectively met most of the requirements set forth by the staff. The web-based hotel reservation system has been judged to have performed satisfactorily

6.2 Future Work

Any development project can be made more valuable by consistently enhancing and updating its features. Several aspects have been chosen for future improvements after discussion with the client. Online payment capability is one significant enhancement. At the moment, uploads are the only way to accept payment receipts; there is no option for making payments online. In order to solve this, it was decided to include a payment gateway-based online payment facility. Customers would especially benefit from this improvement because it will give them a simple and safe option to make payments.

The addition of SMS reminders has also been voted upon as an improvement. At the moment, emails are used to send notifications and reminders. However, an SMS gateway will be incorporated into the system as part of future expansion. Customers will particularly benefit from the convenience as SMS-based reminders and notifications will be sent to hotel employees. They will be able to quickly and easily stay informed of crucial information as a result.

The functionality and user experience of the system will be further improved by these future additions, making it more effective and accommodating to the demands of both online and offline learners.

6.3. Lessons learnt

The skills and information acquired throughout the assignment were priceless. I had the chance to put the knowledge I had learned over the years in the BIT degree program into practice, which helped me learn more about the software development life cycle.

In terms of soft skills, the several meetings held to collect and fine-tune the criteria helped me communicate more effectively. Additionally, I was able to develop my managerial and reporting abilities.

Additionally, the business side of things was also explored. I had the opportunity to learn about development methodologies that I had never used before while creating the proposed system.

Appendices

Appendix A - System Manual

The system manual, a technical document for anyone interested in trying out the web-based hotel reservation system or desiring to enhance its code, provides comprehensive installation instructions. Step-by-step instructions on how to install the system are provided in this manual. Certain conditions must be satisfied in order for the software to be installed successfully.

Hardware requirements

Hardware	Minimum requirements
Processor	Intel® Core ™ i3
Memory	2GB or above
Hard Disk	250GB or above
Display	1024*768 resolution monitor
Printer	Inkjet printer or Laser printer
Internet	ADSL connection (Minimum speed 512kbps)

Table A. 1 Hardware Requirement

Software requirements

Hardware	Minimum requirements
Operating System	Microsoft Windows 7 or Above Operating System
Web Server	XAMPP v3.2.0 or above
Web Browser	Google Chrome / Firefox
Code Editor	Apache NetBeans IDE 12.0

Table A. 2 Software Requirement

Step 1 - Install the web browser

- From Google.com or Mozilla.com, download and install one of the recommended web browsers mentioned above.

Step 2 – Install the web server

- From www.apachefriends.org, download and install the Windows version of XAMPP.
- Give the computer's C: drive as the installation path. Before installing XAMPP on your computer, please refer to the installation guide posted on the previously mentioned page.

Step 3 - File extraction

- Open the folder named nectar_mount_resort and paste it to the following directory path, “C:\Xampp\www”.

Step 4 – Install the Database

- Open a web browser and enter the URL "http://localhost/phpmyadmin" (and, if you've set a username and password, those as well).
- Create an empty database by giving it the name "nectar_mount_resort" and then selecting "choose file" from the "import" menu. The "nectar_mount_resort.sql" file can then be found by opening the Database folder on the CD and selecting it.
  Next, click the "Go" button at the bottom of the page.

Step 5 - Launch the System

- Click the start buttons for Apache and MySQL in the control panel by going to "C: xampp" and opening it.
- To see the system, open a web browser and enter the following URL: "http://localhost/ nectar_mount_resort".

Appendix B - User Manual

The created system's user manual is provided in this page.

Home Page

I have used attractive carousel with navigation bar for web page of web-based hotel reservation system for Nectar Mount Resort.

Everyone in the world can be accessed the system by using WWW.

The figure B.1 shows the web page for Web based Hotel Reservation System for Nectar Mount Resort.

Figure B. 1 Home Page of the Website

User Login

Open the web browser that was set up during installation.

Navigate after entering http://localhost/nectar_mount_resort/web/.

The user can access the login page after navigating.

Figure B.2 illustrates it. A user can access the system by providing a login and password.

You won't be able to log in if your user's name or password are incorrect.

Figure B. 2 User Login of the system

Dashboard

When user log to the system, customer shows his dashboard.

The customer's dashboard is shown in figure B.3 below.

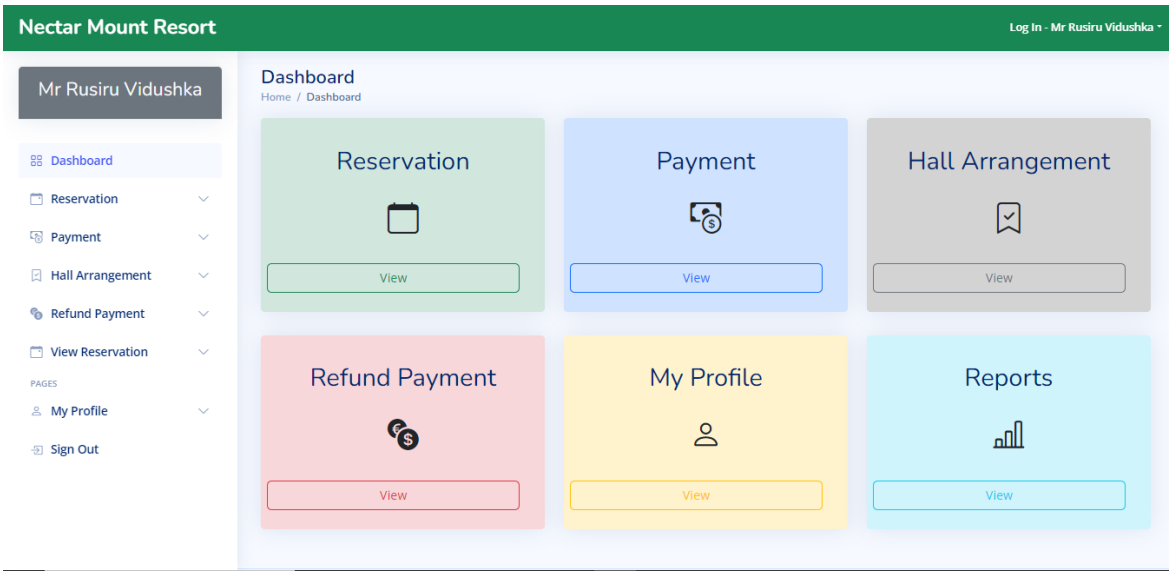


Figure B. 3 Dashboard of the System

Check Hall Availability

After logging into the system can be checked booking of the hall.

Following figure B.4 interface is shown checking hall.

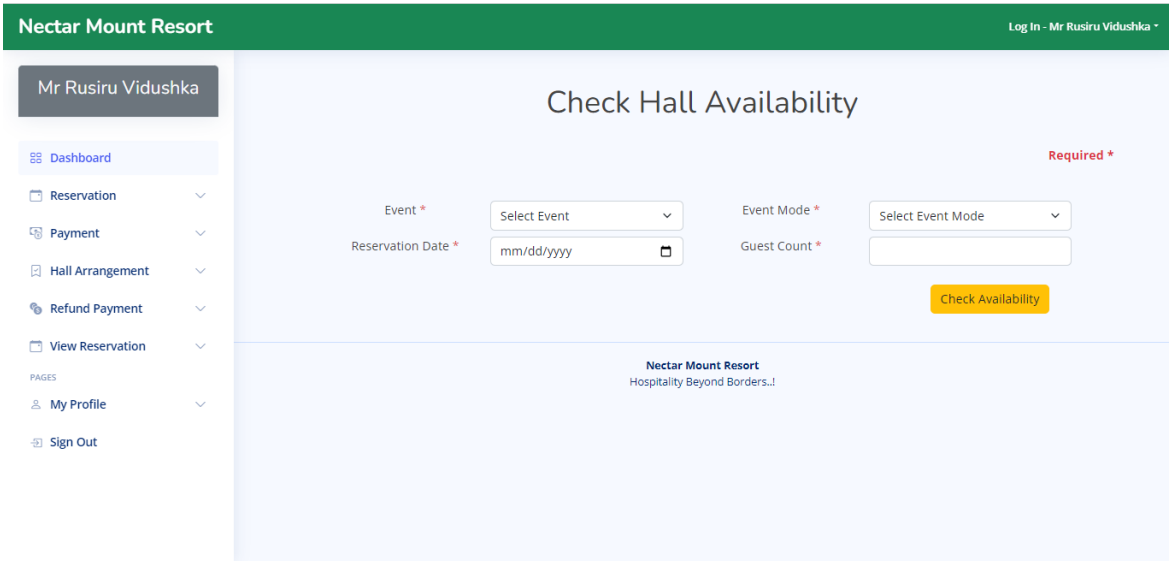
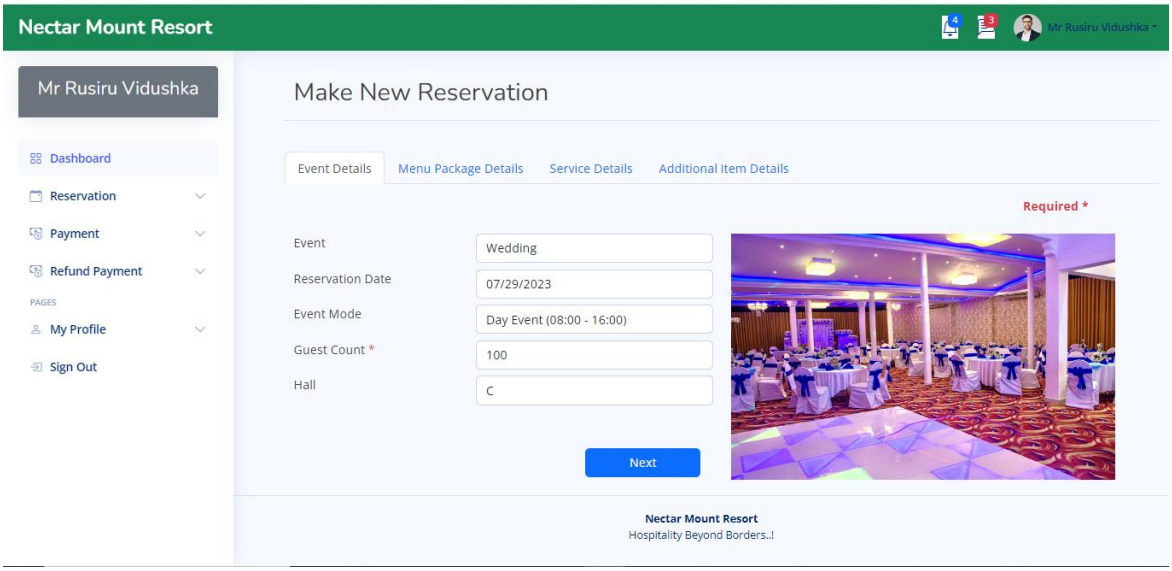


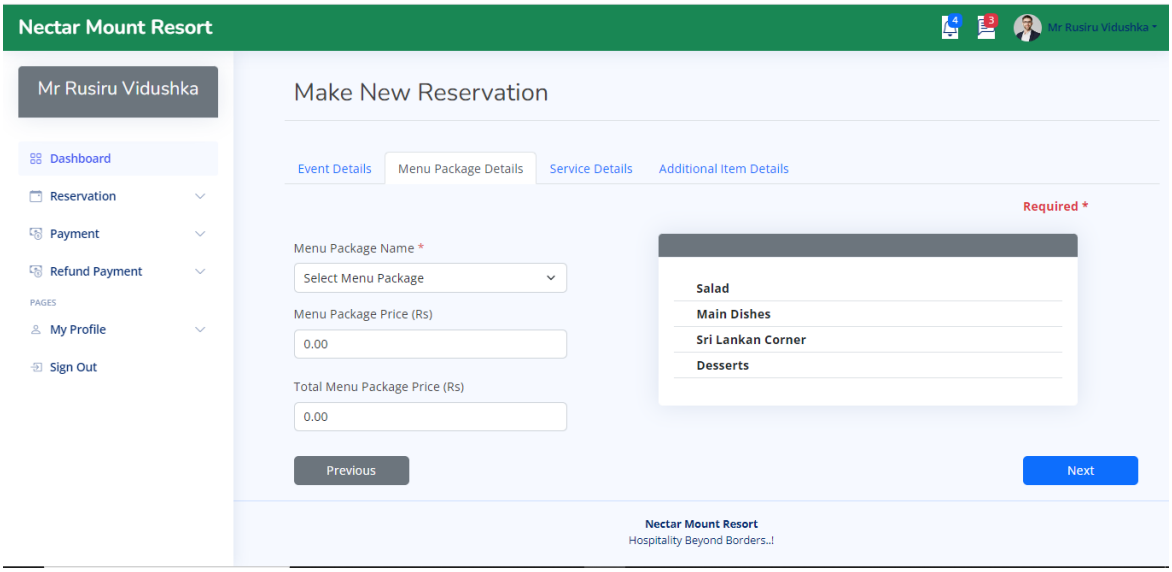
Figure B. 4 Check Hall Availability form of the system

Make New Reservation

When customer log to the system the customer can be make new reservations.

It shows figure B.5 interface to make new reservations.





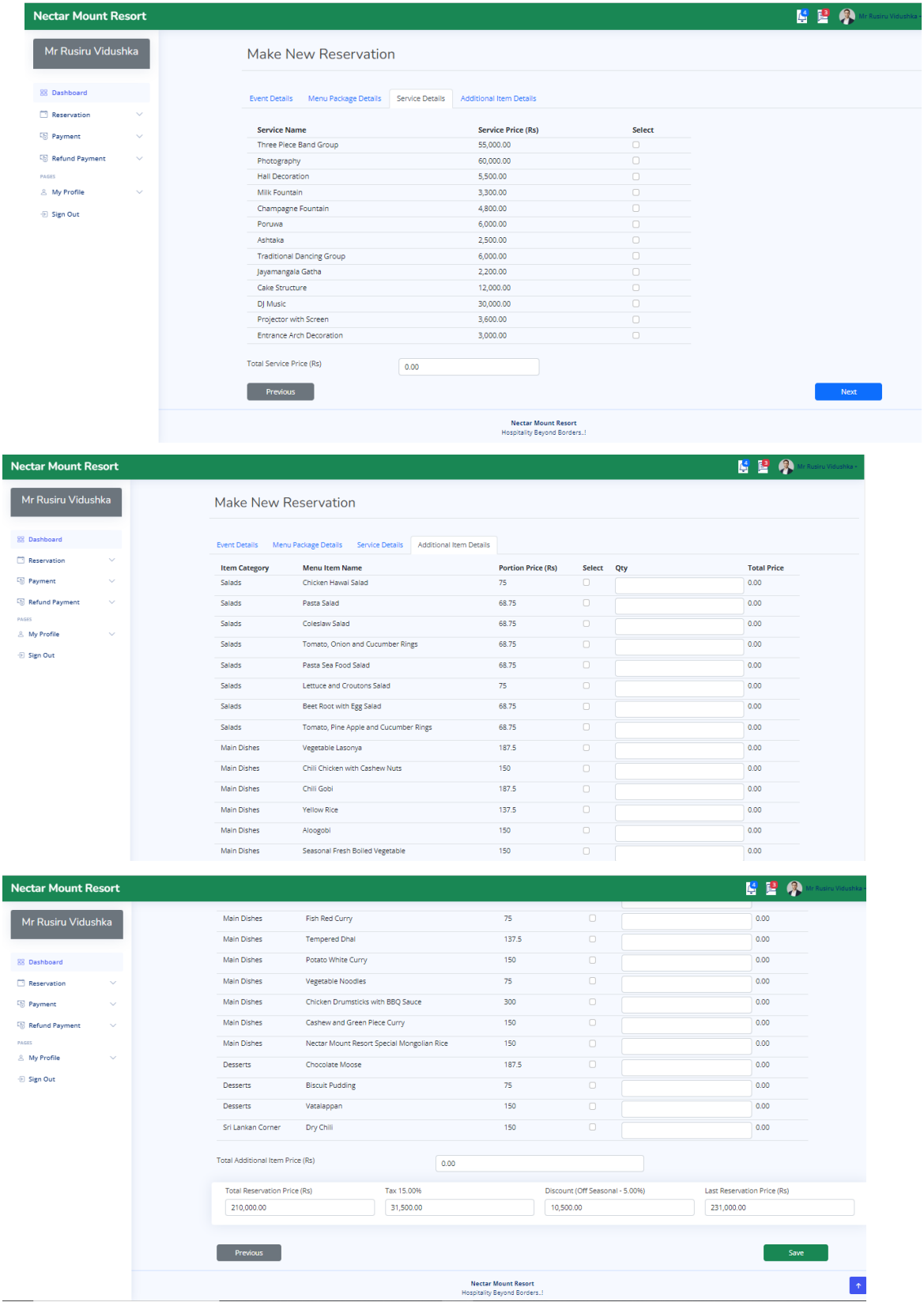


Figure B. 5 Reservation Form of the system

Customer Payment

Customer payments can be done with the following shown interface.

The figure B.6 shows the interface for customer payments

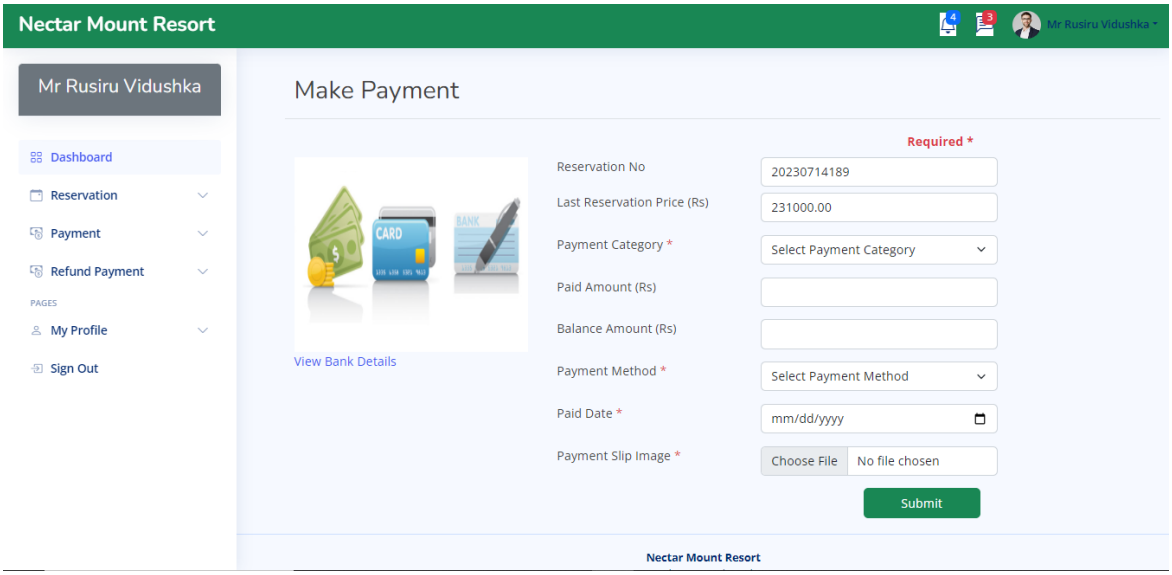


Figure B. 6 Customer Payment form of the system

Appendix C - Management Reports

The management reports produced by the system have a significant impact on its effectiveness.

Web based Hotel Reservation System for Nectar Mount Resort provides many different reports in several purposes. In here it tries to discuss about following reports.

Following tabular report is generated the amount of the customer who has registered through the system from 2023/07/01 to 2023/07/31

The figure C.1 interface to customer registration from 2023/07/01 to 2023/07/31

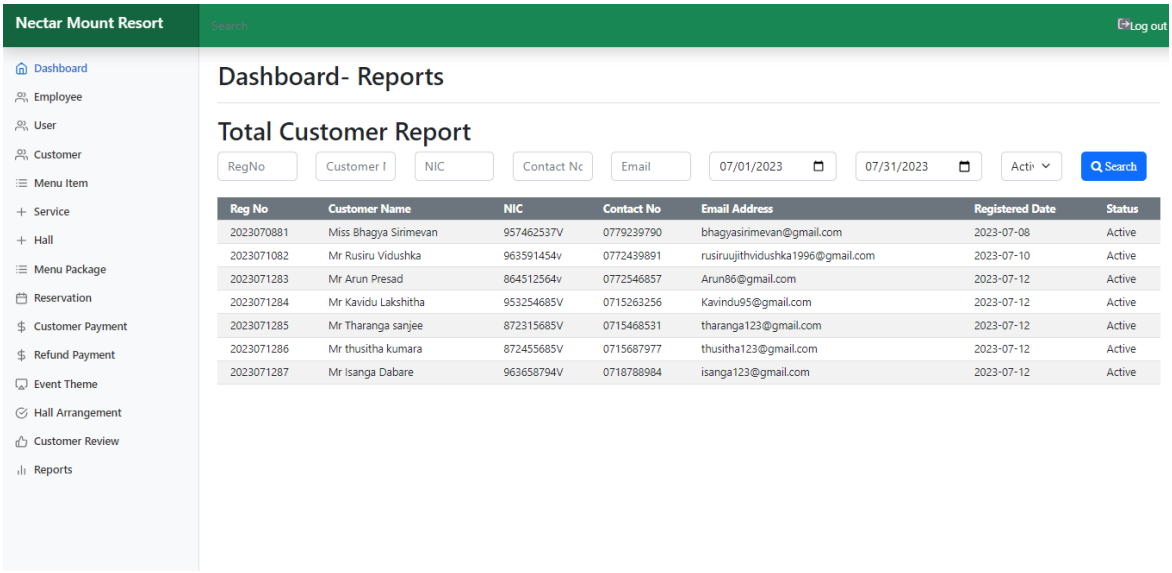


Figure C. 1 Customer Registration Report

The following tabular report is mentioned the amount of total income of the Nectar Mount Resort from 2023/07/01 to 2023/07/14.

The figure C.2 interface to total income of the Nectar Mount Resort from 2023/07/01 to 2023/07/14.

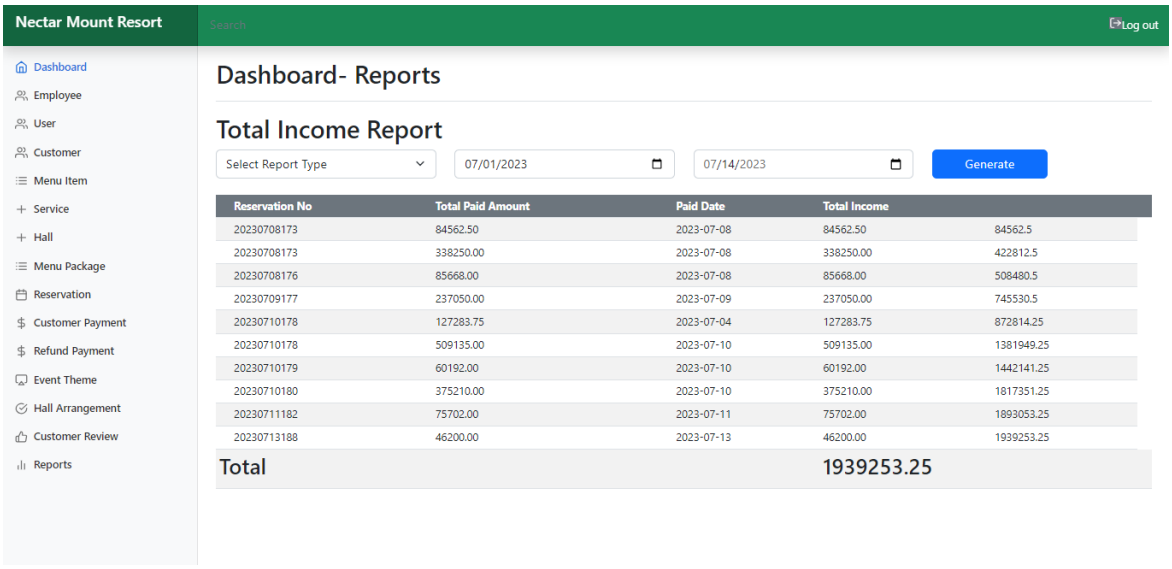


Figure C. 2 Total Income Report

Following tabular report has mentioned about the total reservation of the Nectar Mount Resort from 2023/06/01 to 2023/09/30

The figure C.3 interface to total reservation of the Nectar Mount Resort from 2023/06/01 to 2023/09/30

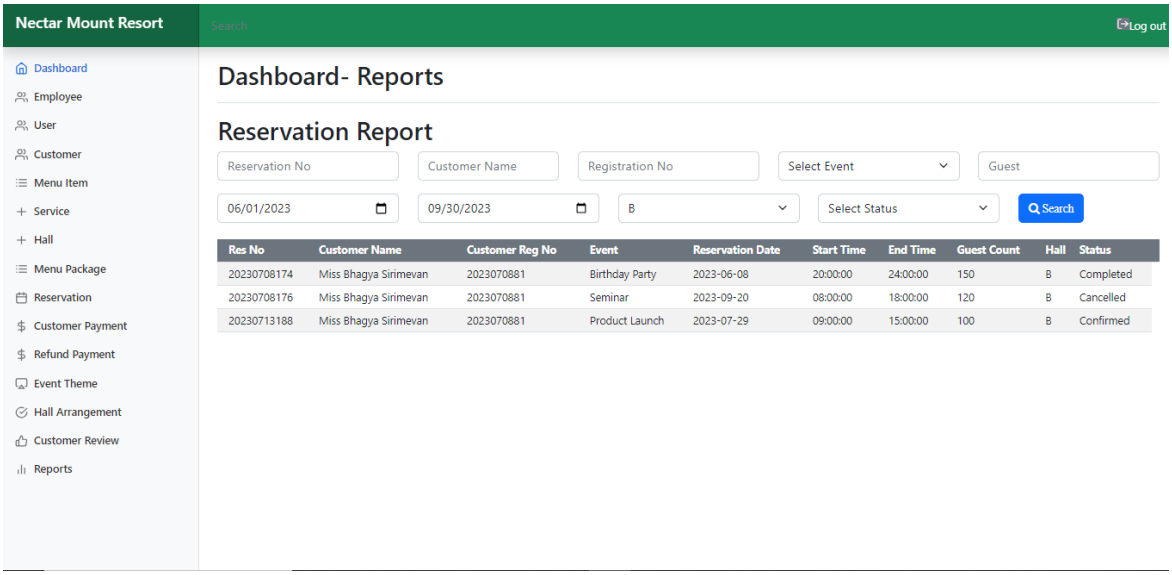
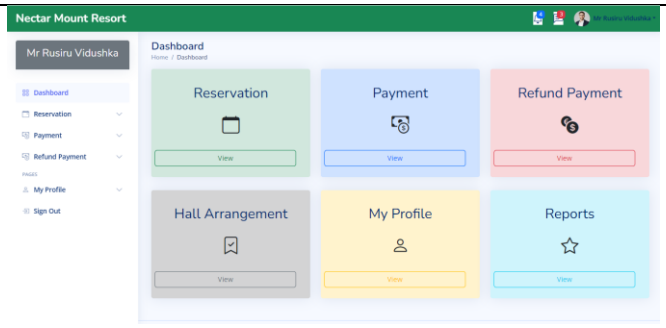


Figure C. 3 Total Reservation Report

Appendix D – Test Result

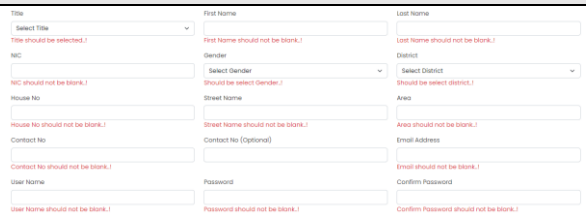
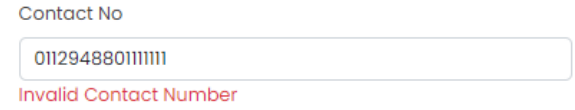
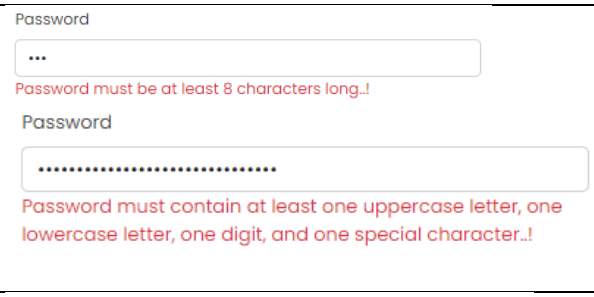
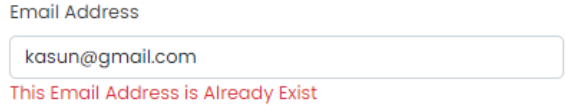
Login

Test No	Test Description	Expected Result	Actual Result
1	Submit login form with empty fields	An error message will appear by displaying the error	
2	Enter an invalid username or password	An error message will appear by displaying the error	
3	Enter a valid username and password	Users can Log into the system	
4	Log in to the system with a valid username	Display logged username on the top status bar	

5	Click on “Logout” in the header section.	Redirect to the login page	

Table D. 1 Test Result of the Login

Register as Customer

Test No	Test Description	Expected Result	Actual Result
1	Submit the form with empty fields.	An error message will display in every empty field.	
2	Enter an invalid Contact No	An error message will display entered invalid field.	
3	Enter an invalid NIC	An error message will display entered invalid field.	
4	Enter an invalid email	An error message will display entered invalid field.	
5	Enter an invalid Password	An error message will display entered invalid field.	
6	Enter the valid NIC	An error message will display entered fields already exist.	
7	Enter the valid email	An error message will display entered fields already exist.	

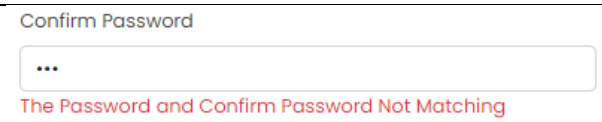
8	Enter the Confirm Password.	An error message will display Confirm Password is not matching Password	
9	Click on the "Register" button.	Display success message.	

Table D. 2 Test Result of the Register as Customer

Check Hall Availability

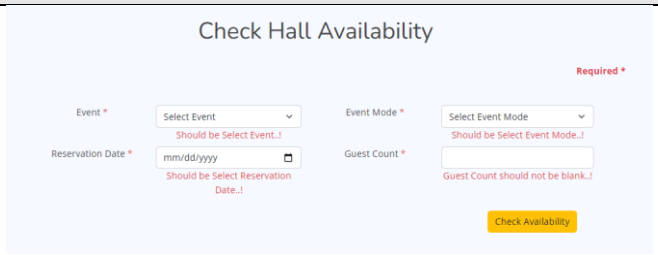
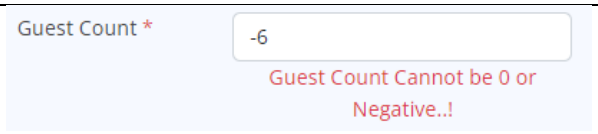
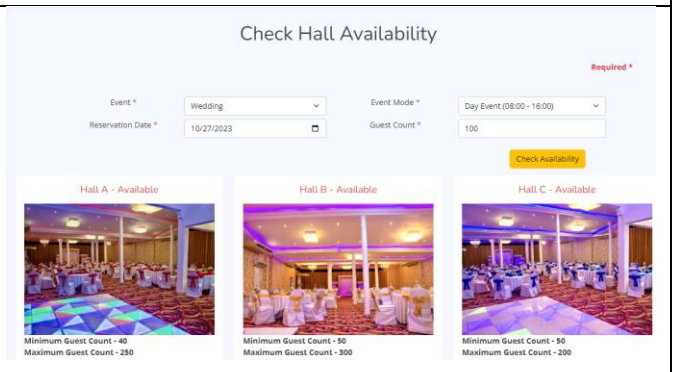
Test No	Test Description	Expected Result	Actual Result
1	Submit the form with empty fields.	An error message will display in every empty field.	
2	Enter an invalid Guest count.	An error message will display entered invalid field.	
3	Enter a valid Guest count	An error message will display guest count not in hall capacity.	
4	Submit the form with valid fields.	An error message will display hall is not available.	
5	Submit the form with valid fields.	Display available halls.	

Table D. 3 Test Result of Check Hall Availability

Make New Reservation

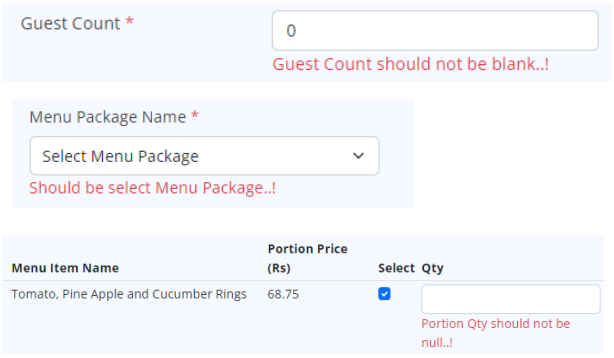
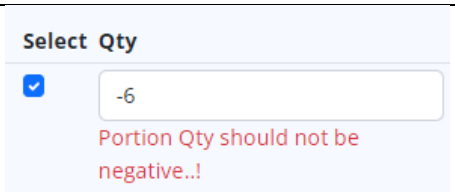
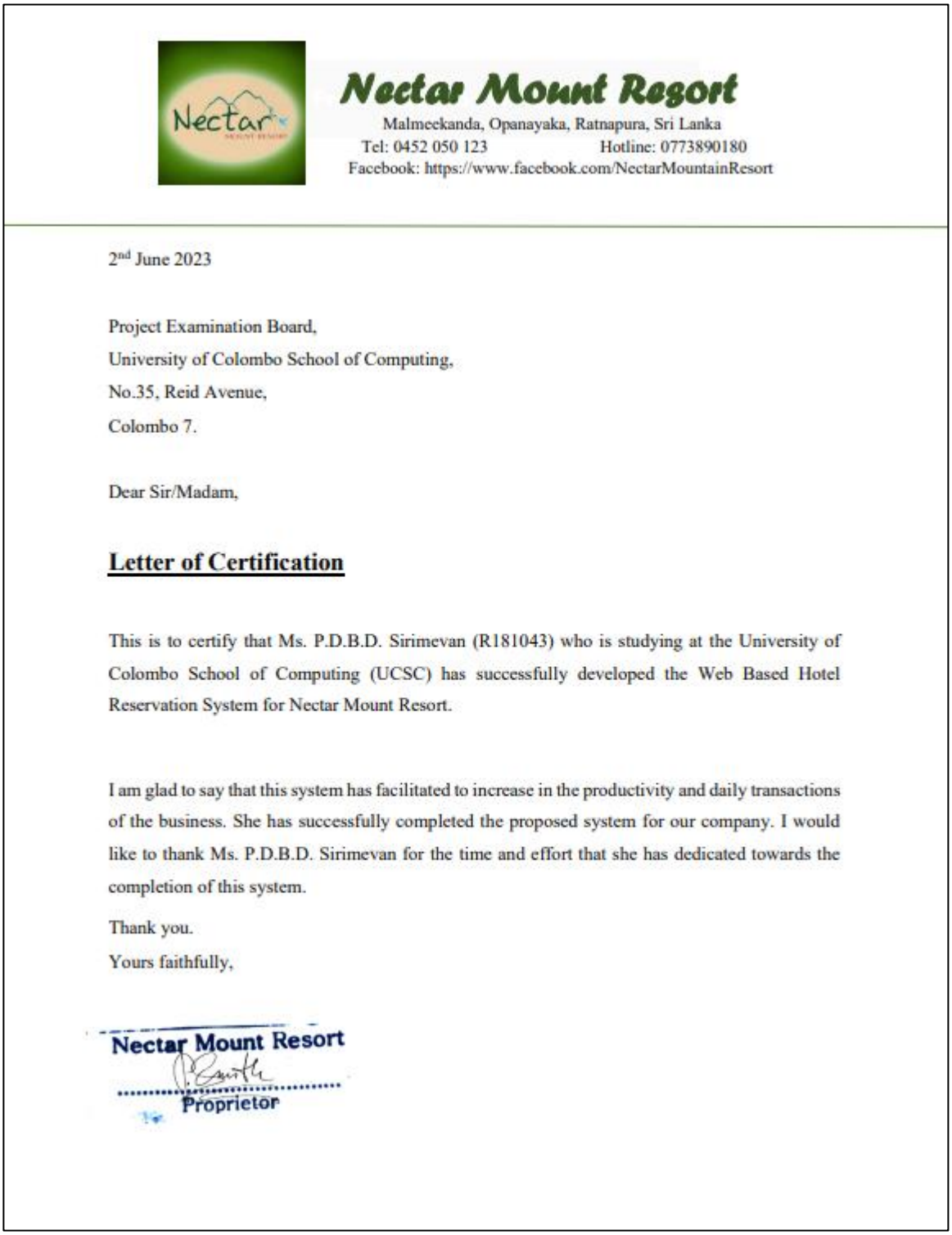
Test No	Test Description	Expected Result	Actual Result
1	Submit the form with empty fields.	An error message will display in every empty field.	
2	Enter an invalid Qty of additional item details	An error message will display entered invalid field.	
3	Click on the “Save” button.	Display success message.	

Table D. 4 Test Result of Make New Reservation

Appendix E- Client Certificate



Nectar Mount Resort

Malmeekanda, Opanayaka, Ratnapura, Sri Lanka
Tel: 0452 050 123 Hotline: 0773890180
Facebook: https://www.facebook.com/NectarMountainResort

2nd June 2023

Project Examination Board,
University of Colombo School of Computing,
No.35, Reid Avenue,
Colombo 7.

Dear Sir/Madam,

Letter of Certification

This is to certify that Ms. P.D.B.D. Sirimevan (R181043) who is studying at the University of Colombo School of Computing (UCSC) has successfully developed the Web Based Hotel Reservation System for Nectar Mount Resort.

I am glad to say that this system has facilitated to increase in the productivity and daily transactions of the business. She has successfully completed the proposed system for our company. I would like to thank Ms. P.D.B.D. Sirimevan for the time and effort that she has dedicated towards the completion of this system.

Thank you.
Yours faithfully,

Nectar Mount Resort
Proprietor

Figure E. 1 Client Certificate

References

[1] "Hilton," [Online]. Available: https://www.hiltoncolombo1.com/.

[2] "OBBS," [Online]. Available: https://phpgurukul.com/online-banquet-booking-system-using-php-and-mysql/.

[3] "IONOS," [Online]. Available: https://www.ionos.com/digitalguide/websites/web-development/waterfall-methodology/.

[4] "geeks," [Online]. Available: https://www.geeksforgeeks.org/differences-between-procedural-and-object-oriented-programming/.

[5] "Art of Testing," [Online]. Available: https://artoftesting.com/what-is-software-testing.