**AMERICAN UNIVERSITY OF SCIENCE & TECHNOLOGY**

Faculty of Engineering and Computer Science

Department of Computer Science

CSI308 – Software Engineering and System Design

**Pine Hill Hotel Reservation System**

By

Zoulfikar Malak – 12220658

Submitted to

Dr. Aziz Barbar

Achrafieh, Lebanon

May, 27, 2025

# Case Description:

1. **Introduction*:***

The Pine Hotel Reservation System is designed to automate and modernize the reservation, registration, billing and checkout process for all hotel guests. This system enables both the guests and hotel staff to have a smooth and pleasant experience ensuring accuracy in reservations, secure payment gateways, and efficient service management. The goal of this Reservation System is to support the full hotel guest cycle from reservation to final checkout.

1. **Business Rules:**

This reservation system is regulated to specific rules:

* + Reservations and check in:
* Guests most reserve rooms using a credit card only.
* The same card used during reservations is assumed to be used upon check in.
* No alternatives payments methods are accepted during reservation or check in (e.g., Cash or Crypto).
  + Room Assignment and Registration:
* Once a guest arrives, they must complete a registration process.
* Our system assigns a room automatically upon registration confirmation.
  + Service Charge and Tipping:
* Guests are allowed to request services (e.g., room service, cleaning)
* A service charge is paid before during or after the service is completed.
* Tips are not added immediately but are added later in the evening to the service charge.
* To keep it simple, the tip is included as a part of the Service Charge instead of making it a separate transaction.
  + Billing and check out:
* A final bill is generated at the time of checkout.
* during checkout, guests are allowed to pay using credit card, check or cash.
* Inheritance can be used writing this code: the payment class can have subclasses like CreditCardPayment, CheckPayment and CashPayment.

* + Authorization and External Systems:
* All credit card transactions are processed via a Credit Authorization System.
* The hotel’s external system interacts with the external system for approval and authorization.

1. **System Actors:**

|  |  |
| --- | --- |
| **Actors** | **Role** |
| Guests | Reserve room, Register, Request Services, Tip Staff, Pay Bills, Checkout. |
| Employee | Adds service charges to guest accounts. |
| Credit  Authorization System | External system is responsible for processing and authenticating credit card transactions. |

1. ***Event Table Summary:***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Event Number** | **Event Description** | **System Input** | **Actor Initiating Input** | **System Output** | **Actor Receiving Output** |
| **1** | Guest Reserving Room | Reservation Request | Guest | Confirmation,  Authorization Request | Guest, Credit System |
| **2** | Guest Registers | Registration Request | Guest | Room Assignment | Guest |
| **3** | Guest Checking Service Charge | Service Request | Employee | Service Charge Receipt | Guest |
| **4** | Guest Adding a tip | Tip | Guest | \_\_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_\_\_\_\_\_\_\_\_ |
| **5** | Generate Final Bill | \_\_\_\_\_\_\_\_\_\_\_\_\_ | System | Final Bill | Guest |
| **6** | Guest Pays Bill | Payment | Guest | Authorization, Request Receipt | Credit System, Guest |
| **7** | Guest Checkout | Checkout Request | Guest | \_\_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_\_\_\_\_\_\_\_\_ |

1. **System Summary:**

* The Pine Hill Reservation System is designed to ensure a smooth interaction between hotel guests and the service providers (Employees) by the hotel. It features flexible components for handling reservations, service management, billing and multiple method payments at checkout. The key designs of this reservation priorities include automation of service charge tracking, integration with payment authorization systems, and adaptability for future enhancements and growth.

# Software Process Model:

1. **Introduction:**

* The pine hill reservation system was developed using a Reuse Oriented

approach. This process emphasizes assembling software systems from existing reusable components, frameworks and reducing the need to build everything from scratch.

1. **Why Reuse Oriented?**

* Many required functionalities, such as payment handling, from validation, and user authentication, are already available, reliable and reusable components.
* Reusing tested modules improves development speed, reliability, and system maintainability.

1. **Reuse Oriented phases:**

* Component Analysis
* Identified existing modules that handle payments, reservation and User Interface components.
* Requirements Modification
* Requirements were adjusted to fit available reusable components (restrict reservation to credit card only)
* System Design with reuse
* The architecture was designed to support integration of reusable and custom build modules.
* Diagrams such as Context, Use case and Sequence were used to model component interactions.
* Development and integration
* Custom logic (room assignment, tip handling) was developed.
* Testing and Validation :
* Reuse components were assumed tested, focus was integration testing.
* The system was validated as complete product through system and acceptance testing.

# Requirement Specification:

1. **Introduction:**

The requirement specifications define both functional and non functional behaviors expected from the hotel system. In this chapter we will lay out what the system should do (functional requirements), and how it should perform(non functional requirements).

These requirements are derived directly from the event table and business rules described earlier.

1. **Functional Requirements:**

* The system should allow a guest to make a room reservation using credit card.
* The system should confirm the reservation and send authorization requests to the credit card system.
* The system should allow the guest to register upon arrival
* The system should designate a room to the guest during registration.
* The system should allow employees to add service charge.
* The system should generate a final bill upon check out.
* The system should allow payments via credit card, cash, or check during checkout.
* The system should interact with the credit authorization system for payment processing through a secure gateway.
* The system should issue a receipt after successful payment.

1. **Non Functional Requirements*:***

* The system should be user friendly, suitable for all types of ages and usable by front desk staff with basic computer knowledge.
* The system should ensure a secure handling of sensitive data, especially payment information (credit card number, password, cvv)
* The system should respond to reservation and billing requests under 4 seconds.
* The system should be available 24/7 without downtime except for scheduled maintenance.
* The system should support logging for every financial transaction.
* The system should support scalability for handling multiple reservations at the same time.

1. **System Constraints:**

* Credit cards are the only payment method accepted for reservation and check in.
* Tips are only applied after a service charge is recorded.
* The same credit card id assumed to be used for reservations and check in.
* External communication is limited to the credit authorization system only.

1. **Assumptions:**

* Each room can only be reserved once per time slot.
* Guests do not share rooms unless specified.
* Services are billed per use and tracked under the guest’s room number.
* Employees have system accounts that control access to service charge entry functions.

# Architectural Design:

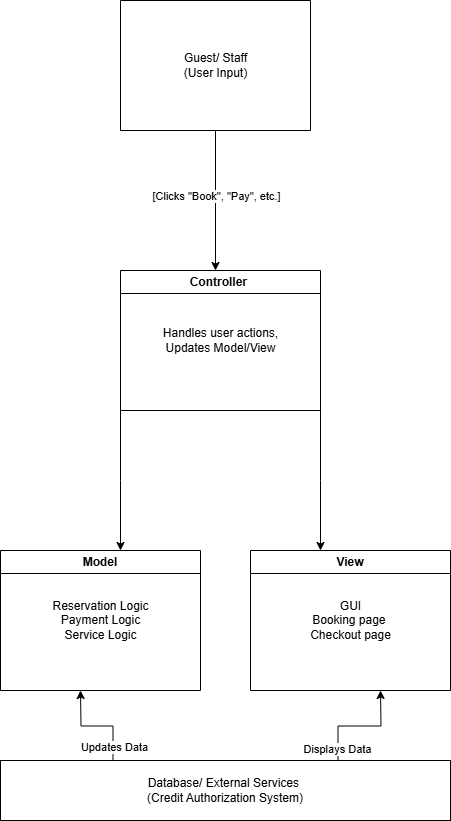
1. **Why MVC is suitable for our system:**

|  |  |
| --- | --- |
| Reason | Explanation |
| Separation | MVC separates the system into distinct parts, logic(model), interface(view), and control flow(controller), this fits our graphical user interface based hotel system with different user roles. |
| User interface based | The reservation system heavily involves user input(booking, tipping, payment), which MVC handles well by linking input events to specific controllers. |
| Maintainability | Each part of the system (booking, billing services ) can be modified independently, allowing easy updates. |

1. **How MVC applies to our system:**

|  |  |  |
| --- | --- | --- |
| Component | MVC Element | Example |
| Room booking logic | Model | Stores reservation information, prices, service details. |
| GUI panels | View | Interface for guests and staff to interact. |
| Button events | Controller | Handles user actions, communicates with model, updates the view |

1. **MVC Diagram:**



# Design and Implantation

1. **Introduction*:***

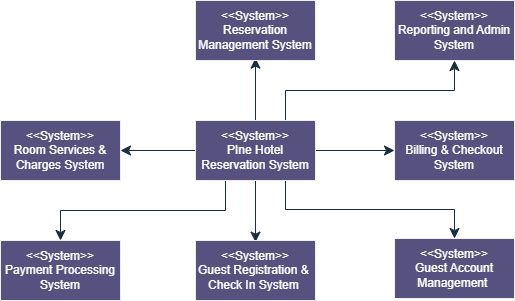
This chapter presents the design of the Pine Hill Hotel Reservation System. It includes various UML and structured diagrams that illustrates the internal design and interaction between system components and external actors. These diagrams help visualize system workflows, data movement, state transitions, and behavior sequence.

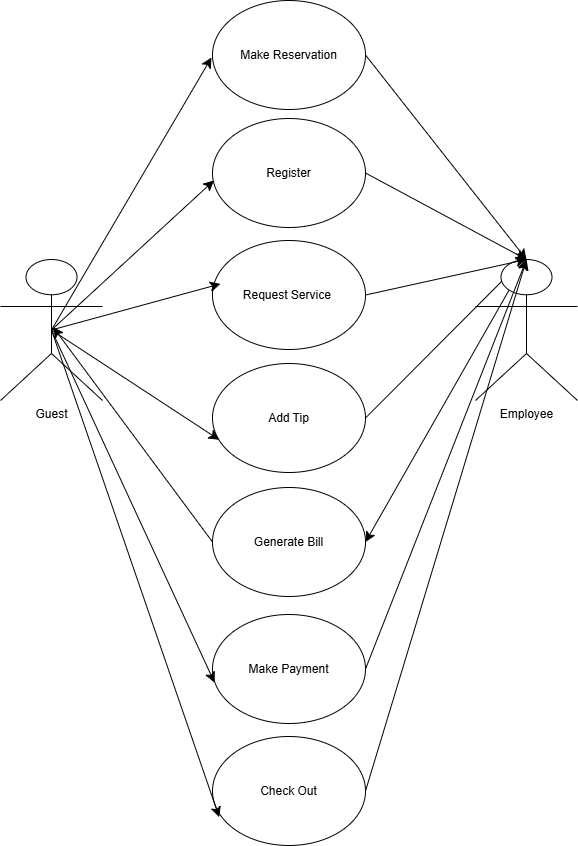
1. **Context Diagram:**

* Purpose:
* The context diagram provides a high level view of the systems interactions with external actors. It treats the hotel system as a single process and shows how guest, employees, and external service communicate with it.
* External Entities:
* Guest: Interacts with the system for reservation, registration, service usage, payment, and checkout.
* Employee: Submits service charge requests.
* Credit Authorization System: Authorizes credit card payments for reservations and final bills.
* Data Flow:

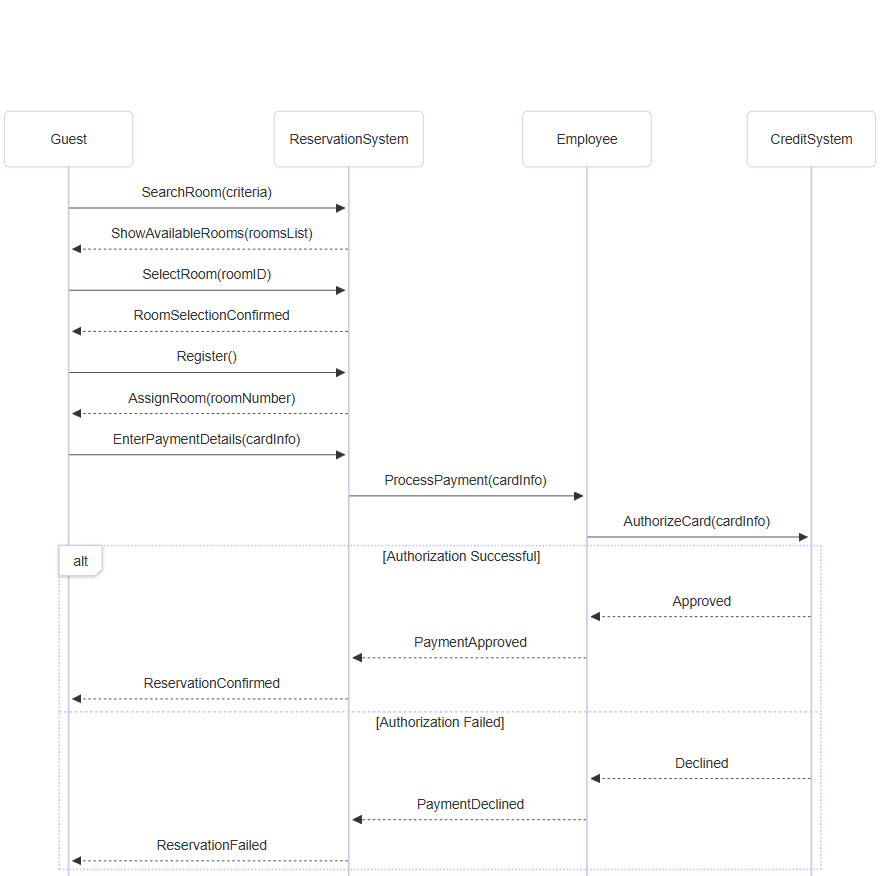
|  |  |  |
| --- | --- | --- |
| **Actors** | **Input** | **Output** |
| **Guest** | Reservation, Registration, Tip, Payment, Checkout Request | Confirmation, Room Assignment, Receipt |
| **Employee** | Service Request | Service Charge Receipt |
| **Credit Card Authorization System** | Authorization Response | Authorization Request |

* Diagram:



1. **Use Case Diagram: ****

|  |  |
| --- | --- |
| **Actors** | Guest, Employee |
| **Description** | The guest is mainly the actors that initiates all the action. These actions include making a reservation before arrival, registering upon check in, requesting additional services during their stay and optionally adding a tip for the services received.  Hotel employees may also perform these same actions on behalf of guests, especially for in person service requests, billing, or payment processing. |
| **Data** | Guest information, reservation details, Services, Billing and Payment records. |
| **Stimulus** | User command issued by guest or hotel employee |
| **Response** | Confirmation that the check-out process is complete. |
| **Comments** | The system verifies payment completion and updates reservation and room status before completing checkout. |

1. **Sequence Diagram**

# Software Testing

1. **Introduction:**

* Software testing is the most crucial and important part before deploying a software to ensure that the system operates correctly, meets user expectations, and is free of critical errors. For the Pine Hill Hotel Reservation System, testing validates the behavior of the core functionalities such as room booking, billing and payment processing.

1. **Testing Strategy:**

* Testing is organized into three main phases:
* Development Testing
* Release Testing
* User testing (Acceptance Testing)

Each of these phases include specific types of testing, carried out at different stags of the system’s development and deployment.

1. **Types of testing used:**

|  |  |
| --- | --- |
| **Testing Type** | **Purpose** |
| **Integration testing** | Integration testing verifies that different modules or components of the system work together correctly as a combined unit. |
| **Component testing** | Tests groups of related objects (Reservation +payment) to ensure they work together. |
| **System testing** | System testing tests the entire systems flow from start to end (booking -> checkout) |
| **Acceptance testing** | Acceptance testing validates whether the system meets business requirements and is ready for deployment. |

1. **Test plan:**

The test plan defines what is to be tested, the testing strategy, the schedule, and how the results will be recorded. For critical systems, it may also include a full set of defined tests that must be passed before release.

For this system, the test plan includes:

* Testing system modules during development.
* Integration testing after components were completed.
* Full System testing in a simulated environment.
* Final acceptance testing by user based on real case scenarios.

1. Each unit or component is tested through defined input partitions and expected outcomes:

|  |  |  |  |
| --- | --- | --- | --- |
| **Test case** | **Purpose** | **Input** | **Expected result** |
| **Test Case 1** | Room reservation executed successfully | Valid date, room type | Booking confirmed |
| **Test case 2** | Tip addition | $5 tip | The tip will appear in the guest bill |
| **Test case 3** | Payment failure | Invalid card info | Payment declined message shown |
| **Test case 4** | Check out | Guests requests checkout | Bill generated,  room marked as a vacant. |

# Conclusion and Future Work

1. **Conclusion**:

The Pine Hill Hotel Reservation System was developed to automate the key operations, including the room reservation system, guest registration, service charge handling, billing and checkout. The system architecture was modeled using a variety of different diagrams including context diagram, use case diagram, state machine, data flow and sequence diagram. The testing strategies used to test this specific software include unit testing, integration testing and acceptance testing were applied to ensure the system is reliable and ready for deployment.

This specific software only allows guests to reserve rooms using only credit cards, register upon arrival, request services, tipping and perform secure payments at checkout. It also supports hotel staff in managing these operations efficiently. The use of modular design principles and mock testing for external services like credit card authorization helped in building a robust and adaptable solution.

This project has demonstrated the application of software engineering principles from the very first step which is requirement gathering till the very last step which is testing.

1. **Future Work:**

Software change is mandatory due to new user requirements, evolving environments, and improvements in technology. The following improvements are proposed for future versions of the system:

* **Loyalty and reward system:**

A digital card where the user earns points each time, they book a room at any of the Pine Hill Hotel branches. Gathered points can be redeemed for discounts or services.

* **Security:**

Integrating facial recognition technology allowing users to securely access their account eliminating the traditional username/password methods.

* **Mobile app integration:**

Developing a mobile version of the Pine Hill Hotel Reservation System to make the reservation, billing and requesting services a smooth and a user friendly experience.