# Project Meeting 1 - Project 2 Case 2

Class 4 – Group 2: *Ong Jung Yi (1006655), Luong Hung (1007160), Luvyn Sequeira (1007081), Stavya Sharma (1006995), Zhuang Yang Kun (1006933), Yee Jia Zhen (1006969), Lee Jun Hui Ryan (1006652)*

# Introduction

In today’s digital age, online platforms for booking hotels and accommodations have revolutionized the way people travel. Travellers now prioritize convenience and efficiency to compare and find the best deals for hotels. With the increasing demand for a comprehensive, user-friendly hotel booking system, our project attempts to address this demand by creating a robust web application that helps users search, compare and book their accommodation.

## Purpose

* Enables users to search, compare and book accommodations.
* Contains aggregate listings from various hotels to display detailed information, user reviews and pricing.
* Provide users with a user-friendly platform to compare prices and book accommodation when travelling.
* Recommending hotels based on user preferences and past booking history
* Provide secure booking and payment processes to protect user personal data and financial transactional infromation.

## Target Audience

* Leisure tourists
* Business travelers
* Families
* Tech savvy individuals who prefer booking accommodations online & value the ability to compare prices
* Budget conscious travelers looking for a seamless booking experience

## Problem Statement

*To create a functional and interactive hotel booking web app.*

# Use Case Description

## Use Case 1: Login

|  |
| --- |
| **Description**  This use case enables customers to login to their Ascenda accounts before making a booking. While logging in is not required for booking, it provides additional convenience by auto filling in the user’s details and preferences when making a booking. |
| **Primary Actor**  Customer |
| **Secondary Actor**  Authentication System, Database |
| **Preconditions**  1. Authentication System and Database is online |
| **Postconditions**  1. The customer is authenticated and is logged in to their account |
| **Main Flow**  1. Customer navigates to login page - system provides displays login form with username/email and password fields.  2. Customer enters their username/email and password and then submits login form.  3. System processes login request (checks credentials against stored user data)  4. Valid credentials - system authenticates customer and grants access to account  5. Customer is redirected back to main page but now logged in with session token |
| **Alternative Flows**  2a. Customer clicks on “Forgot Password” link  1. Customer redirected to password recovery page  2. Customer submits email for password recovery  3. System sends password reset link to provided email  3a. Customer does not fill up both fields  1. System displays a message informing user to fill up both fields  2. Return back to step 2 of main flow  4a. Credentials are invalid  1. System displays an error message indicating incorrect username/email  2. Return back to step 2 of main flow |

## Use Case 2: Search

|  |
| --- |
| **Description**  This use case allows the customer to perform a search using specified search criteria. |
| **Primary Actor**  Customer |
| **Secondary Actor**  System |
| **Preconditions**  1. Customer must be logged into the hotel booking system. |
| **Postconditions**  1. The user may click on search to view the search results. |
| **Main Flow**  1. Customer logs into the hotel booking system.  2. Customer navigates to the search page.  3. Customer indicates relevant search criteria such as the dates of stay, country/region, number of guests etc.  4. Customer can click on the search button to display search results. |
| **Alternative Flow**  4a. Customer clicked on search button before entering mandatory search criteria such as country/region   1. System prompts the user to fill the relevant search criteria. 2. Customer fills in the relevant search criteria. 3. Customer clicks on search button. |

## Use Case 3: Display Search Results

|  |
| --- |
| **Description**  This use case describes the process of displaying search results for hotels based on user input such as location, duration of stay and the number of guests |
| **Primary Actor**  Customer |
| **Secondary Actor**  Authentication System, Database |
| **Preconditions**  1. Customer has accessed the hotel search page on the website  2. Customer has inputted the necessary search criteria  3. System is connected to a database of the hotel API which provides the hotel data |
| **Postconditions**  1. Customer sees a list of hotels that match the search criteria  2. Each hotel is displayed with a summarized version, including essential information such as hotel name, star rating etc…. Sorted by the prices  3. Customer can click on any hotel to view more detailed information and proceed with booking a room if interested |
| **Main Flow**  3. System processes search request  4. System calls an external API to retrieve hotel listing that match the search criteria  5. System sorts the search result by prices by default from lowest to highest  6. System displays the search results in a user-friendly format, showing essential information for each hotel  7. User interacts with the search result |
| **Alternative Flows**  1a. The user inputs invalid or incomplete search criteria the system will display an error message indicating the issue and prompt the user to correct it.  1b. If there is no hotels match the search criteria, the system will display a message indicating that no results are found and suggest modification of criteria  1c. System encounter a network issue; the system displays an error message and allows the user to retry the search or try again later |

## Use Case 4: Checkout Hotel

|  |
| --- |
| **Description**  This use case enables customers to complete the checkout process for a hotel stay. The system processes the checkout details, calculates the final bill based on the duration of stay and any additional charges, and handles payment. The customer receives an email confirmation of their checkout and payment. |
| **Primary Actor**  Customer |
| **Secondary Actor**  System |
| **Preconditions**  1. Customer must have an active booking. |
| **Postconditions**  1. The hotel checkout is completed.  2. The payment is processed.  3. An email confirmation is sent to the customer. |
| **Main Flow**  1. Customer navigates to the checkout page.  2. System displays the total bill, including stay duration and any additional charges.  3. Customer reviews the bill and selects a payment method.  4. Customer completes the payment.  5. System confirms the checkout and sends an email confirmation to the customer. |
| **Alternative Flows**  2a. If the customer chooses not to log in, the system allows the customer to proceed as a guest.  3a. If there are discrepancies in the bill or errors during payment, the system prompts the customer to resolve the issues or try a different payment method.  4a. Customer requests an itemized bill to review detailed charges before final payment. |

## Use Case 5: View Selected Hotel

|  |
| --- |
| **Description**  This use case describes the process of user viewing their selected hotel from hotel search results. |
| **Primary Actor**  Customer |
| **Secondary Actor**  System |
| **Preconditions**   1. Customer has searched for hotels. |
| **Postconditions**   1. Customer sees the details of the selected hotel such as hotel name, star rating, price etc 2. Customer can choose to proceed with booking the hotel. |
| **Main Flow**   1. Customer clicks on a hotel from the search results. 2. System calls an external API to retrieve the details of the selected hotel 3. System displays the hotel details. 4. User clicks book now to book the hotel. |
| **Alternative Flows**  4a. User click back to return to hotel search results.  1. Repeat steps 1 to 4 |

## Use Case 6: Payment Process

|  |
| --- |
| **Description**  This use case describes the process of a customer making a payment for a hotel booking. |
| **Primary Actor**  Customer |
| **Secondary Actor**  System |
| **Preconditions**  1. Customer has selected a hotel and room.  2. System is connected to a payment gateway API.  3. Customer has valid payment information. |
| **Postconditions**  1. Payment is processed, and the booking is confirmed.  2. Customer receives a booking confirmation.  3. In case of payment failure, the customer is informed and can retry. |
| **Main Flow**  1. Customer reviews booking and proceeds to payment.  2. System displays payment options.  3. Customer enters payment details.  4. System validates details and processes payment via gateway API.  5. Payment is confirmed; the system updates booking status and sends confirmation. |
| **Alternative Flows**  3a. If invalid payment details, system prompts for correct information, customer re-enters details.  3b. If payment failure, the system informs the customer of failure, customer retries or selects another method.  4a. If session timeout, system informs customer of timeout, customer restarts payment process. |

## Use Case 7: Notify Users

|  |
| --- |
| **Description**  This use case sends users an email confirmation to notify them of their successful checkout. |
| **Primary Actor**  Customer |
| **Secondary Actor**  System |
| **Preconditions**   1. Customer successfully completes checkout of their hotel and payment has been processed. |
| **Postconditions**   1. Customer sees the details of their booking such as the hotel name, duration of stay, type of room booked, etc. |
| **Main Flow**   1. Customer logs into the system. 2. Customer completes checkout of hotel. 3. Payment gateway API completes processing the payment. 4. System sends a notification email to the customer to provide them confirmation of their booking. 5. Customer may click on a link in the email that directs them back to the booking website. |
| **Alternative Flows**  4a. System does not successfully deliver email to the customer due to email server being down.   1. System logs this failure. 2. System tries to send the email again at regular intervals until the email is successfully delivered to the user. |

# Software Development Process – Waterfall Model

Our team has decided to follow this order in our software development process: Project Planning, Analysis, Design, Implementation, Support

* Project Planning where we understand the scope of the system and develop a timeline.
* Analysis where we focus on understanding the business needs in detail with the help of software models.
* Design where we design the solution system based on that.
* Implementation phase where we will divide the different implementation tasks among members and conduct integration and testing along the way.
* Finally, we will conduct more testing and make any minor changes required before our final submission.

# Risk and Constraints

**Inadequate knowledge** - team is new to JavaScript and web development

* Mitigation: stick to tools and frameworks being taught in 50.003 (Node.js, Express.js, etc.) and reduce overhead that comes with learning new and unfamiliar tools (time, suboptimal code, etc.). Regularly consult professors and TAs for technical assistance.

**Scalability** - software might struggle with high volume of users

* Mitigation: optimize database queries, scalable architecture design (services that can scale horizontally), regular load testing

**Security** - potential unauthorized access, data breaches, user personal data

* Mitigation: good security practices (don’t save passwords directly, use hashes), implement role-based access control, only store necessary user data

**Maintenance** - when system needs to be updated and maintained, might result in long periods of downtime

* Mitigation: comprehensive documentation, modular design, keep system design simple

**Time constraints** - team has extremely busy schedule, tight on time

* Mitigation: set out a project timeline (next slide) with sufficient buffer time. Implement only necessary features first. Optional features (like login system) can be developed if there is spare time, and software should be designed in a modular way such that optional features can be integrated easily (with minimal changes to existing code)

# Project Timeline & Distribution of Workload

|  |  |
| --- | --- |
| **Week** | **Tasks** |
| **7** | * Figma Prototype - *, Stavya Sharma, Zhuang Yang Kun* * Use Case Diagram - *Luong Hung* * App Structure - *Luvyn Sequeira, Yee Jia Zhen, Lee Jun Hui Ryan* * Setting up Git repository (version control) - *Ong Jung Yi* |
| **8** | * Begin Coding, Complete Basic Features - *Ong Jung Yi, Luvyn Sequeira, Yee Jia Zhen, Lee Jun Hui Ryan* * Finalise Use Case Diagram & Design - *Luong Hung, Stavya Sharma, Zhuang Yang Kun* * Demonstration of Basic Features in PM2 |
| **9** | * Completion of main features - *Ong Jung Yi, Luong Hung, Luvyn Sequeira, Stavya Sharma, Zhuang Yang Kun, Yee Jia Zhen, Lee Jun Hui Ryan* |
| **10** | * Completion of main features - *Ong Jung Yi, Luong Hung, Luvyn Sequeira, Stavya Sharma, Zhuang Yang Kun, Yee Jia Zhen, Lee Jun Hui Ryan* |
| **11** | * Finalize any last changes to sequence / use case diagrams - *Luong Hung, Stavya Sharma, Zhuang Yang Kun* * Completion of additional features - *Ong Jung Yi, Luvyn Sequeira, Yee Jia Zhen, Lee Jun Hui Ryan* * Start unity & system testing - *Ong Jung Yi, Luong Hung, Luvyn Sequeira, Stavya Sharma, Zhuang Yang Kun, Yee Jia Zhen, Lee Jun Hui Ryan* * Demonstration of all features & some tests |
| **12** | * Finalize features - *Ong Jung Yi, Luong Hung, Luvyn Sequeira, Stavya Sharma* * Finalize unit & system testing - *Zhuang Yang Kun, Yee Jia Zhen, Lee Jun Hui Ryan* |
| **13** | * Complete project & handover - *Ong Jung Yi, Luong Hung, Luvyn Sequeira, Stavya Sharma, Zhuang Yang Kun, Yee Jia Zhen, Lee Jun Hui Ryan* * Final Presentation |