Software Design Document

Sydney Airbnb Data

Student Names

Table of Contents

[1.0 System Vision 3](#_Toc46748622)

[1.1 Problem Background 3](#_Toc46748623)

[1.2 System Overview 3](#_Toc46748624)

[1.3 Potential Benefits 3](#_Toc46748625)

[2.0 Requirements 4](#_Toc46748626)

[2.1 User Requirements 4](#_Toc46748627)

[2.2 Software Requirements 4](#_Toc46748628)

[2.3 Use Cases 4](#_Toc46748629)

[3.0 System Components and Software Design 5](#_Toc46748630)

[3.1 System Components 5](#_Toc46748631)

[3.2 Software Design 5](#_Toc46748632)

[4.0 User Interface Design 6](#_Toc46748633)

# System Vision

## Problem Background

The Sydney Airbnb rental market is a vibrant and varied environment with a wide selection of homes for short-term stays. Users looking to rent or invest in real estate may find it difficult to sort through the massive amount of data and make wise choices. Making decisions in this market can be strongly influenced by an understanding of price trends, property attributes, and customer feedback.

A big issue potential renters or investors may face with is Information Overload. Numerous characteristics are included in this information, including location, property type, amenities, and client feedback. Finding the best solutions after navigating through this information becomes a difficult undertaking.

In addition to Information Overload, the complexity of pricing in the Sydney Airbnb market adds to the decision-making challenge. The property's location in relation to important events and attractions as well as the provision of special amenities might have an impact on rental rates. Deep data research is necessary to understand pricing trends and determine the best times of year to book or list properties. Users could find it difficult to determine fair price without a thorough tool to analyse pricing distributions across time periods, which could result in missed opportunities or unsatisfactory rental plans. Dealing with this pricing complexity is therefore essential for the everyday customer.

## System Overview

We Propose a simple software system that will assist customers to narrow down their searches and requirements for Airbnb's, specifically in Sydney. Our system intends to simplify the decision-making process for users looking for short-term accommodations in the dynamic Sydney rental market by utilising the power of data analysis and visualisation. The system will behave as a simple program that analyses datasets from Sydney’s Airbnb environment and allow users to interact and extract specifications they require, of said datasets.

The system's fundamental feature allows users to choose specific suburbs, time periods, and keywords to analyse and visualise pertinent data. The software processes data on property listings, pricing trends, and customer reviews by interacting with the sizable Sydney Airbnb dataset. A graphical user interface will make it simple for consumers to obtain key information that influences their renting decisions.

## Potential Benefits

Potential benefits include a customer's Insight into Price trends, Transparent Property Attributes and Time Management.

Insight into Price Trends.

One of the key advantages of our proposed software system lies in its ability to provide users with insights into price trends within the Sydney Airbnb rental market. By allowing users to input specific time periods for analysis, the software generates comprehensive visualizations that outline how rental prices fluctuate over different seasons and events. This feature assists potential renters and property investors alike in making well-informed decisions about the optimal times to book or list properties. The clear visualization of price distributions empowers users to grasp the range of prices they can expect to encounter, thus preventing underestimation or overpayment. With these insights, users can strategically plan their rental experiences, avoiding the rush during peak seasons and seizing opportunities when prices are most favourable.

Transparent Property Attributes.

Another key benefit of our system is the transparency of property attributes. Having access to clear and thorough information about property features and services becomes crucial as customers browse through the wide variety of short-term accommodations available on the Sydney Airbnb market. By allowing customers to enter criteria, such as desired suburbs and keywords, to filter listings depending on their preferences, our programme streamlines the process. With the use of this tool, consumers can save time and effort by only seeing listings that match their specific needs. The system enables visitors to choose the accommodations that best meet their needs by showing property qualities including location, property type, and amenities that are offered.

Time Optimization.

A key benefit for users navigating the Sydney Airbnb market is time management. Making informed decisions takes much less time when a plethora of property-related data can be accessed and analysed quickly. Users may rapidly select listings based on their preferred suburbs, discover transparent property attributes, and gain insights on price trends instead of wading through mountains of information. This simplified procedure not only improves user experience but also makes it possible for users (such as investors) to quickly take advantage of rental opportunities. Users can concentrate their efforts on analysing qualities that most closely match their tastes and goals by cutting down on the time spent on data analysis.

# Requirements

## User Requirements

In this section you detail how a user is supposed to interact with or use your program. What do they ***need*** to be able to do? This should all be from the end users perspective. Can be a combination of narrative text and listing of needs.

**Assignment note: You have not been given a client/user, so you can make one up. Who do you think would be using your software?**

The expected user of the system will be an average traveller who makes their way into Sydney for a holiday stay.

The User must be able to do the following in order to fully utilize the program:

* Launch the Program.

This includes having up to date computers (phones, laptops, etc...) that can run modern software.

* Enter their preferences.
* Customize their search.
* Navigate results.
* Exit the program.

## Software Requirements

In this section you detail what the requirements for the software are. What functionality will it provide? This is usually a formal listing, with requirements often using the word ‘Shall’. IE:

R1.1 The program shall accept multiple file names as arguments from the command line.

R1.2 Each file name can be a simple file name or include the full path of the file with one or more levels.

etc …

Can be primarily functional requirements, though you may include other types if you think of them.

R1.1 The program shall accept multiple file names as arguments from the command line.

R1.2 Each file name can be a simple file name or include the full path of the file with one or more levels.

R1.3 The program shall provide a user-friendly interface for interactions.

R1.4 The program shall offer users their preferred suburb for accommodation searches.

R1.5 The program shall enable users to specify the time of their stay using start and end dates.

R1.6 The program shall offer a visualization feature to display the distribution of rental prices within user-defined time periods.

R1.7 The program shall allow users to conduct keyword-based searches by inputting amenities or preferences such as "pool" or "pet."

R1.8 The program shall process user inputs to filter property listings based on selected suburbs, time periods, and keywords.

R1.9 The program shall provide transparent property attributes including location, property type, amenities, and customer feedback.

R1.10 The program shall perform cleanliness analysis by identifying keywords related to cleanliness in customer reviews.

R1.11 The program shall display clear and understandable visualizations/listings of price distributions and cleanliness insights.

R1.12 The program shall be able to analyse the datasets and differentiate the property types respectfully and be able to generate listings of specific types respective of the user’s preference.

R1.13 The program shall be designed to handle various edge cases, such as invalid inputs and missing data, to ensure robustness.

R1.14 The program shall provide an option for users to exit the software when their interactions are complete.

## Use Cases & Use Case Diagrams

In this section you provide some use cases showing how people may use your software.

Below are 5 Use Cases and their respective diagrams for the use cases when using the system;

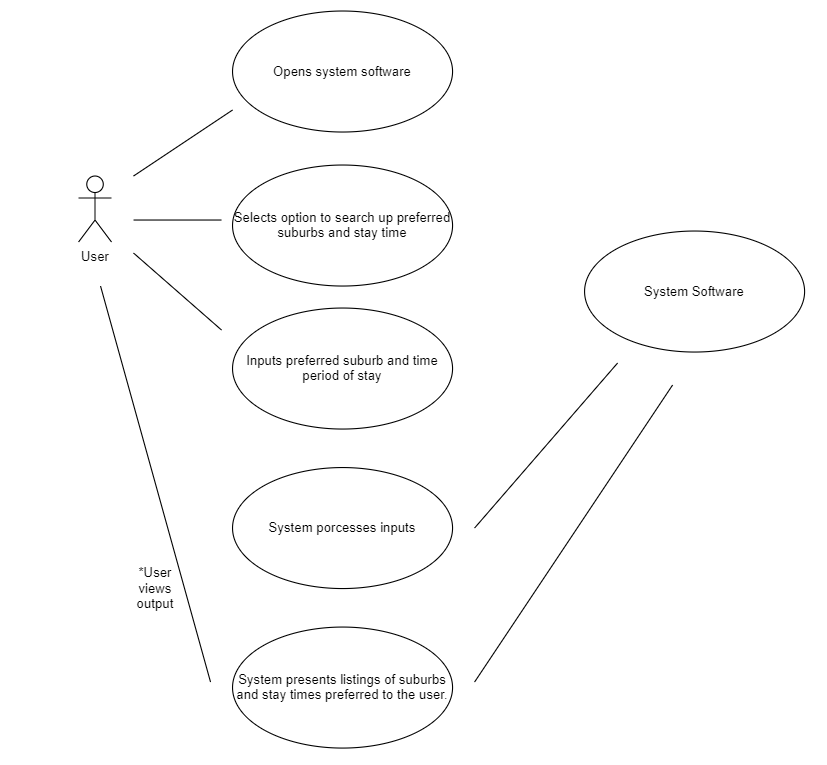
* Search by Suburb and Time Period
* Visualization of Price Distribution
* Searching by Keyword
* Analysing Cleanliness by Comments
* Property Type Preference

To note it will a paramount functional requirement that the System provide a simple, unclustered User Interface.

Use Case 1. User searching Suburb and Time Period on Software System

|  |  |
| --- | --- |
| **Use Case ID** | 1 |
| **Use Case Name** | Search by Suburb and Time Period |
| **Primary Actors** | User |
| **Description** | The software is used by the User to narrow down Airbnb accommodation options in Sydney specific to the User’s requirements of suburb and specific time frame. |
| **Pre-Conditions** | The user has launched the Airbnb data analysis and visualization program and is presented with the user interface. |
| **Flow of Events** | 1. User opens the software 2. User selects Option to search by suburb and time period. 3. User inputs suburb and stay dates. 4. User submits Input. 5. System processes input and retrieves listings. 6. System displays listing details to the user. |
| **Post-Conditions** | User receives an organised list of available accommodations in the chosen suburb for the specified time period. |
| **Alternate Flow** | Invalid inputs – System prompts user to re-enter information.  No listings found – System informs user and suggests adjusting parameters. |

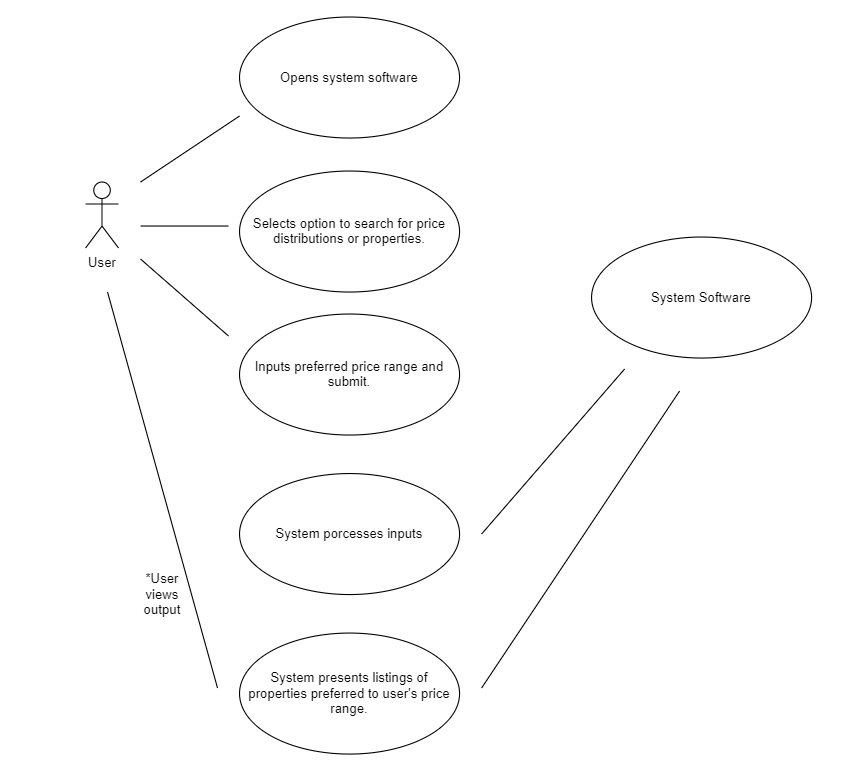
Use Case Diagram 1. Use Case ID 1.



Use Case 2. User Visualising Price Distribution on System Software

|  |  |
| --- | --- |
| **Use Case ID** | 2 |
| **Use Case Name** | Visualise Price Distribution |
| **Primary Actors** | User |
| **Description** | A user interacts with the system to produce a visual depiction of the distribution of Airbnb properties in Sydney. |
| **Pre-Conditions** | The user has launched the Airbnb data analysis and visualization program and is presented with the user interface. |
| **Flow of Events** | 1. User opens the software. 2. The user selects the option to search for price distributions of the properties. 3. User inputs their preferred price ranges. 4. User submits input. 5. System processes the user’s input and scans datasets accordingly against prices of properties. 6. The system generates and produces the user with a visualisation/listing of properties preferred to the user’s price ranges. |
| **Post-Conditions** | The user is provided with a visualisation that displays the distribution of property prices. |
| **Alternate Flow** | If there is an error during the processing of the user's input:  The system displays an error message and guides the user to retry generating the price distribution. |

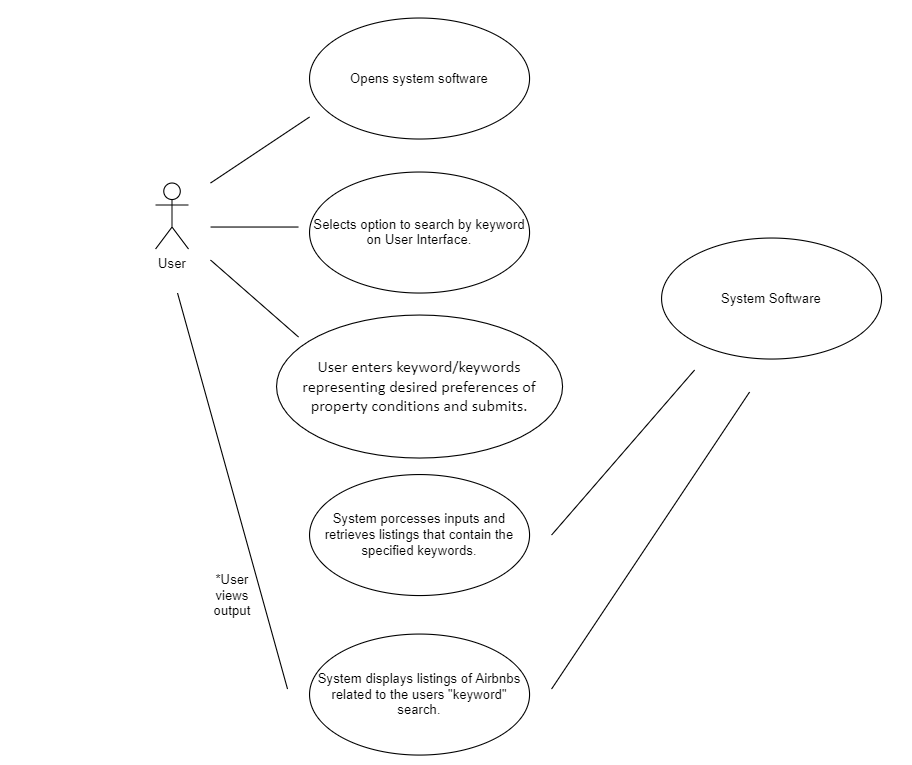
Use Case Diagram 2. Use Case ID 2.



Use Case 3. User searching for Airbnb's by keyword.

|  |  |
| --- | --- |
| **Use Case ID** | 3 |
| **Use Case Name** | Search by Keyword |
| **Primary Actors** | User |
| **Description** | The user interacts with the system to look up specific Airbnb listings using keywords they have entered. By using keywords like "pool," "pet," and other suitable terms, the user hopes to filter and find Airbnb descriptions or even ads linked to the Airbnb properties. |
| **Pre-Conditions** | The user has launched the Airbnb data analysis and visualization program and is presented with the user interface. |
| **Flow of Events** | 1. User opens Airbnb dataset system software. 2. User selects option to search by keyword on the UI. 3. User enters keyword/keywords representing desired preferences of property conditions. 4. User submits input. 5. The system processes the user's input and retrieves listings that contain the specified keywords. 6. The system displays listings of Airbnb’s that have descriptions/ads that contain terms related to the user’s search. |
| **Post-Conditions** | The user is presented with a list of Airbnb listings that contain the specified keywords, helping them find accommodations that cater to their preferences. |
| **Alternate Flow** | If no listings match the entered keywords:  The system informs the user that no listings match the criteria and suggests refining the search. |

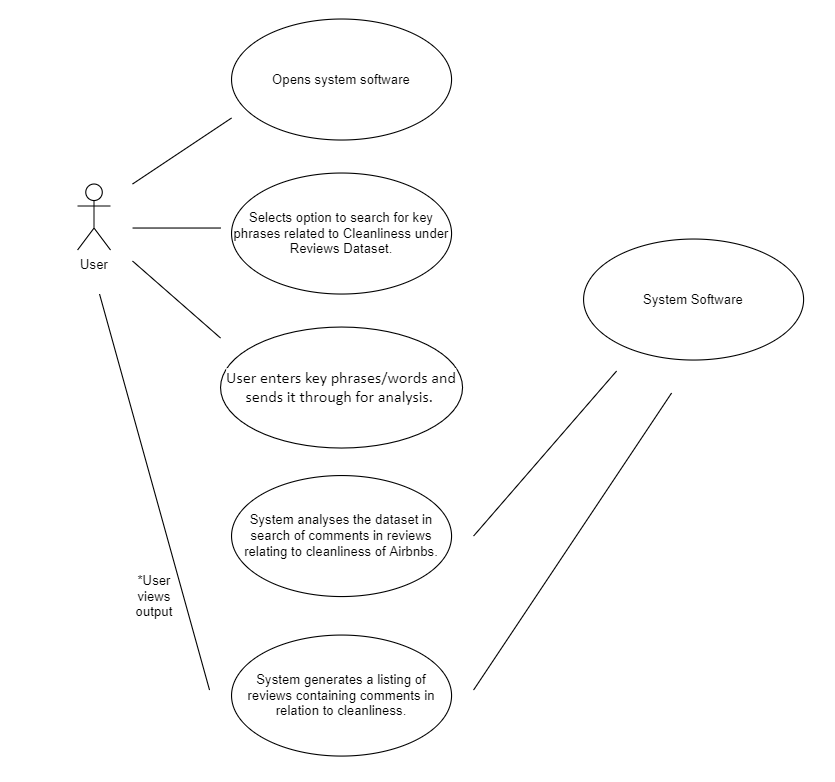
Use Case Diagram 3. Use Case ID 3.



Use Case 4. Analyse Cleanliness by Comments

|  |  |
| --- | --- |
| **Use Case ID** | 4 |
| **Use Case Name** | Analysing Cleanliness by searching Comments |
| **Primary Actors** | User |
| **Description** | In this use case, the system examines user reviews of Airbnb rentals to assess the standard of cleanliness and pinpoint elements that affect hygiene. The user searches in the system for reviews that highlight cleanliness aspects of Airbnb’s in Sydney. The user does this by inputting keywords or rather synonyms of words and phrases related to “clean”, such words as “tidy” and “spotless”. |
| **Pre-Conditions** | The system has access to customer reviews and associated data for Airbnb listings. |
| **Flow of Events** | 1. User opens System Software. 2. Under the “Reviews” selection of datasets, the user inputs key phrases or words in relation to cleanliness. 3. Once the input is sent through, the system analyses the dataset in search of comments related to the search. 4. The system generates a listing of comments and reviews in relation to cleanliness comments/words. |
| **Post-Conditions** | The system provides a report showcasing the number of cleanliness-related comments for each Airbnb listing. |
| **Alternate Flow** | If no cleanliness-related keywords are found in the reviews:  The system may display a message indicating that there are no comments specifically related to cleanliness. |

Use Case Diagram 4. Use Case ID 4.

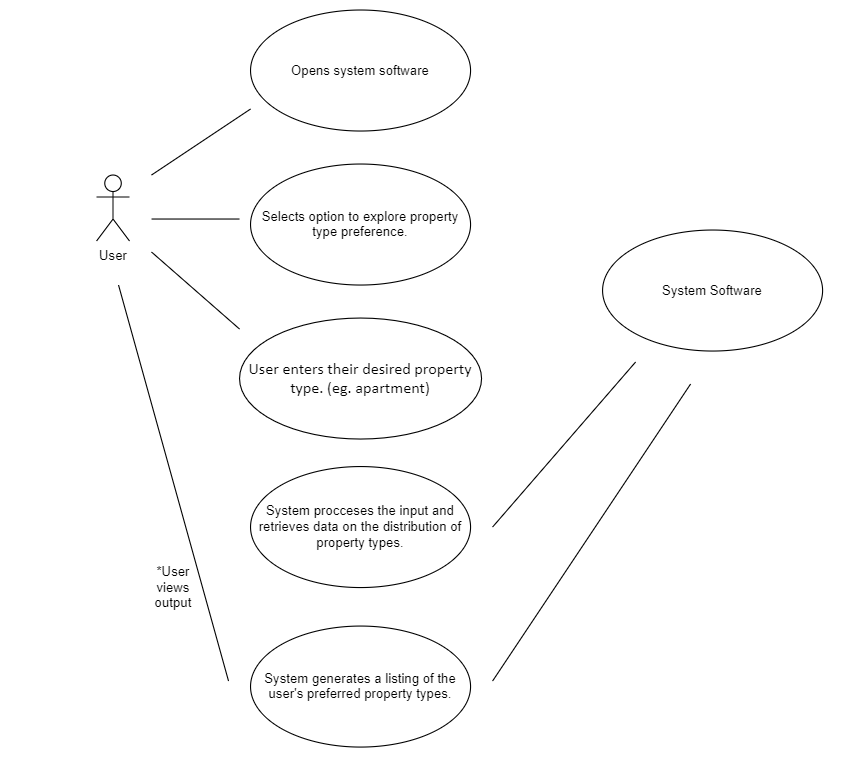


Use Case 5. Property Type Preference.

|  |  |
| --- | --- |
| **Use Case ID** | 5 |
| **Use Case Name** | Property Type Preference. |
| **Primary Actors** | User |
| **Description** | The user searches for their preferred Airbnb accommodation type. The user aims to understand the distribution and availability of various property types, such as apartments, houses, or shared rooms. |
| **Pre-Conditions** | The user has launched the Airbnb data analysis and visualization program and is presented with the user interface. |
| **Flow of Events** | 1. The user selects the option to explore property type preference. 2. The user inputs their preferred accommodation type (house, apartments, etc...). 3. The system processes the user's input and retrieves data on the distribution of property types in Sydney’s Airbnb scene across the dataset by analysing property descriptions and reviews. 4. The system generates a listing of the user’s desired property type as per their input has requested. |
| **Post-Conditions** | The user is presented with a visualization that showcases the distribution of property types, helping them identify their preferred property type for accommodations. |
| **Alternate Flow** | If there is an error during the processing of the user's input:  The system displays an error message and guides the user to retry inputting the property type preference. |

Notes: Only One property type can be requested at a time.

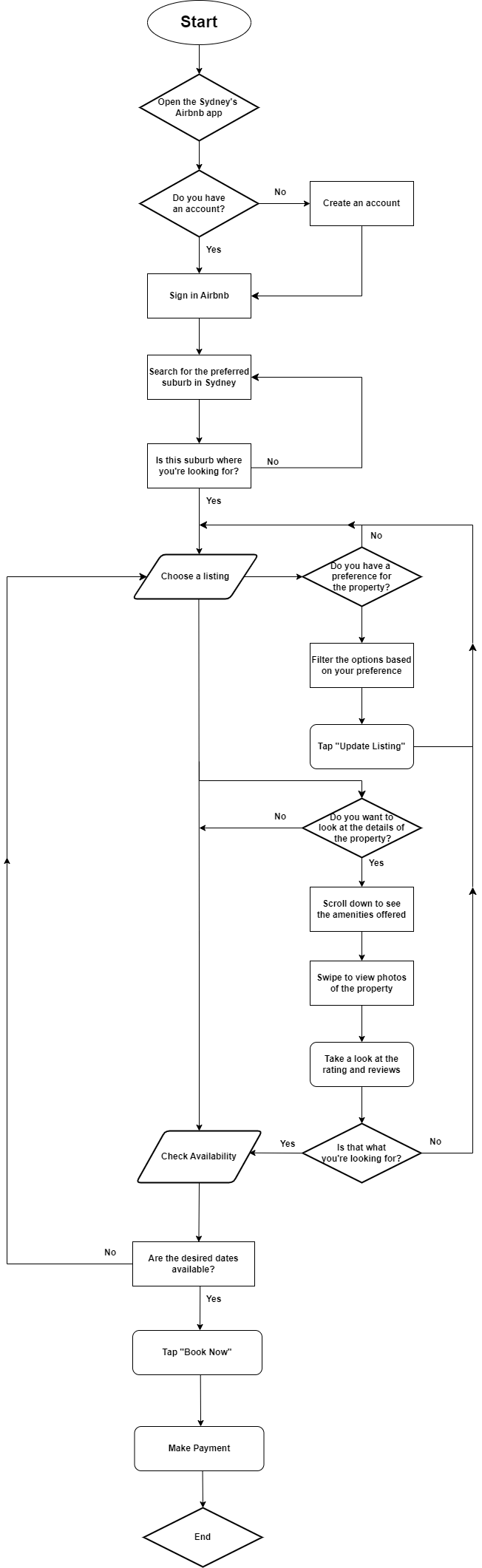
Use Case Diagram 5. Use Case ID 5.



# Software Design and System Components

## Software Design

A block diagram/flowchart of how your software might work



## System Components

### Functions

Preliminary list of all functions in the software. For each function in the list the following information is provided:

* a brief description of what it does (1 or 2 sentences);
* a list of the input parameters, and their data types, and what they are used for;
* a list of any side effects caused by the function (ie change global or member variables, changes data passed by reference from calling function etc)
* a description of the function’s return value

**1) User Registration:**

Description: Allows users to create new accounts.

Input Parameters: User details (name: string, email: string, password: string).

Side Effects: Creates a new user account in the database.

Return Value: None.

**2) User Login:**

Description: Allows users to log in to their accounts.

Input Parameters: User credentials (email: string, password: string).

Side Effects: Sets the user's login session.

Return Value: User ID or success indicator.

**3) Search Listings:**

Description: Enables users to search for properties based on various criteria.

Input Parameters: Location: string, dates: date range, preferences: object.

Side Effects: None.

Return Value: List of matching property IDs.

**4) View Listing Details:**

Description: Displays detailed information about a selected listing.

Input Parameters: Listing ID: string.

Side Effects: None.

Return Value: Listing details object.

**5) Submit Booking Request:**

Description: Allows users to submit booking requests for specific dates.

Input Parameters: Listing ID: string, requested dates: date range, guest count: integer.

Side Effects: Adds a booking request to the database.

Return Value: Booking request ID.

**6) Check Availability:**

Description: Checks if a property is available for the requested dates.

Input Parameters: Listing ID: string, requested dates: date range.

Side Effects: None.

Return Value: Availability status (boolean).

**7) Confirm Booking:**

Description: Confirms a booking after availability is confirmed.

Input Parameters: Booking details: object, payment information: object.

Side Effects: Confirms the booking in the database, updates property availability.

Return Value: Booking confirmation ID.

**8) Process Payment:**

Description: Handles payment processing for confirmed bookings.

Input Parameters: Payment details: object.

Side Effects: Processes the payment through a payment gateway.

Return Value: Payment success status (boolean).

**9) Host Interaction:**

Description: Facilitates communication between guests and hosts.

Input Parameters: Messages: string, requests: object, responses: object.

Side Effects: Updates message history.

Return Value: None.

**10) Send Check-In Instructions:**

Description: Sends check-in instructions to guests before arrival.

Input Parameters: Booking details: object.

Side Effects: Sends email or notification to the guest.

Return Value: None.

**11) Guest Check-Out:**

Description: Marks the end of a guest's stay.

Input Parameters: Check-out date: date.

Side Effects: Updates booking status and availability.

Return Value: None.

**12) Leave Reviews and Ratings:**

Description: Allows guests and hosts to leave reviews and ratings.

Input Parameters: Review text: string, rating: integer.

Side Effects: Updates review data in the database.

Return Value: Review submission status (boolean).

**13) Manage User Profile:**

Description: Lets users view and update their profile information.

Input Parameters: Profile details: object, updates: object.

Side Effects: Updates user profile data in the database.

Return Value: Updated profile data.

**14) Notifications:**

Description: Sends notifications to users for various events.

Input Parameters: Event triggers: object.

Side Effects: Sends notifications to users.

Return Value: None.

**15) Customer Support:**

Description: Provides customer support to users.

Input Parameters: Support request details: object.

Side Effects: None.

Return Value: Support response object.

### Data Structures / Data Sources

List of all data structures in the software (eg linked lists, trees, arrays etc) or eternal data sources. For each data structure in the list the following information is provided:

* Type of structure (tree, list etc),
* Description of where and how it is used
* List of data members, and what each one is for do
* List of functions that use it

**1) Database (Relational Database Management System):**

Type of Structure: Relational Database

Description: Stores all application data including user profiles, property listings, bookings, reviews, and more.

Data Members: Tables with various fields for different data types (e.g., Users, Listings, Bookings).

Functions Using It: Almost all functions interact with the database for data retrieval, updates, and storage.

**2) User Profile Object:**

Type of Structure: Object

Description: Represents user profile information.

Data Members: User ID, name, email, password, preferences, etc.

Functions Using It: User registration, user login, manage profile.

**3) Property Listing Object:**

Type of Structure: Object

Description: Represents a property listing with details.

Data Members: Listing ID, location, description, photos, amenities, availability, etc.

Functions Using It: Search listings, view listing details, submit booking request.

**4) Booking Object:**

Type of Structure: Object

Description: Represents a booking request or confirmed booking.

Data Members: Booking ID, guest ID, listing ID, dates, status, payment details, etc.

Functions Using It: Submit booking request, confirm booking, process payment.

**5) Message Thread (Linked List or Array of Objects):**

Type of Structure: Linked List or Array

Description: Manages communication between users (guests and hosts).

Data Members: Messages (text, sender, timestamp).

Functions Using It: Host interaction, communication between users.

**6) Review Object:**

Type of Structure: Object

Description: Represents a user review for a property or a guest/host.

Data Members: Review ID, reviewer ID, reviewed ID, text, rating, timestamp.

Functions Using It: Leave reviews and ratings.

**7) Notification Queue (Priority Queue or List of Objects):**

Type of Structure: Priority Queue or List

Description: Holds notifications for users to be delivered.

Data Members: Notifications (message, type, timestamp).

Functions Using It: Notifications for various events.

**8) Search Preferences Object:**

Type of Structure: Object

Description: Represents user preferences for searching listings.

Data Members: Location, dates, amenities, price range, etc.

Functions Using It: Search listings based on preferences.

**9) Support Request Object:**

Type of Structure: Object

Description: Represents a user's support request.

Data Members: Request ID, user ID, issue description, status, timestamp.

Functions Using It: Customer support interaction.

**10) External APIs (Geolocation, Payment Gateways, etc.):**

Type of Structure: External Data Source

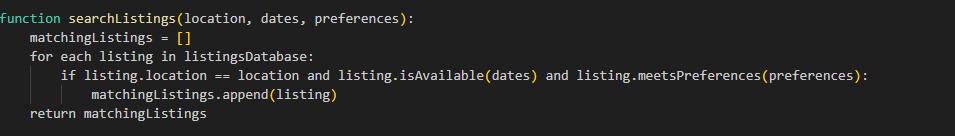
Description: Interfaces with external services like geolocation for maps, payment gateways for payments, etc.

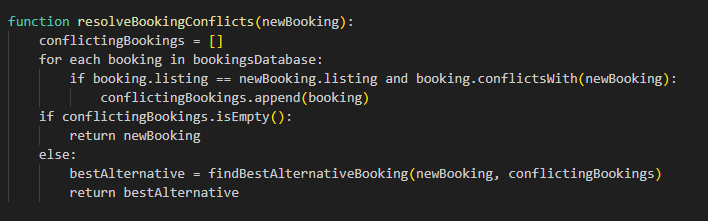
Data Members: API endpoints, authentication tokens, request/response formats.

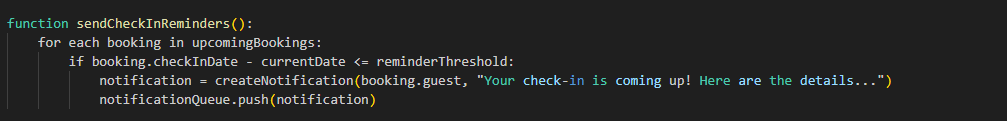
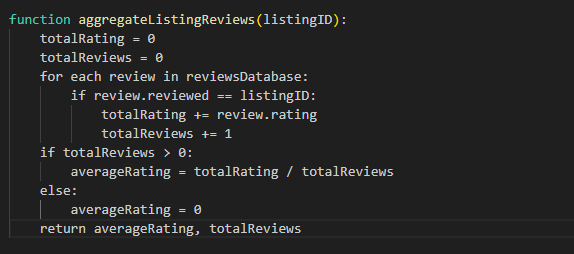
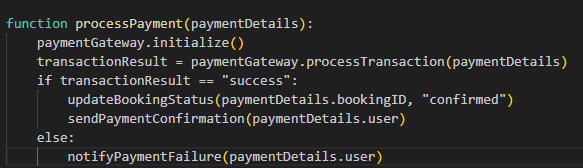
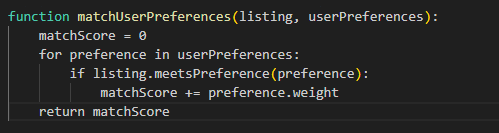
Functions Using It: Geolocation services, payment processing.

### Detailed Design

Pseudocode for all non-standard / non-trivial algorithms that operate on data structures

Search Listing:  
  


Booking Conflict:  
  


Check-in Notification:  
  
  
  
Review:   
  
  
  
Payment:   
  
  
User Preference Matching:  


# User Interface Design

This is your initial interface design. Describe the tools you used for this design stage and any key findings that informed your design. This introduction is descriptive and should explain what you have completed for the actual design work you will present in the sub-sections below.

## Structural Design

Structural design refers to the navigational and information structure of your product – the structure that supports the interface layout. How will you structure your product? How will you group your information? How will you navigate through your product? Why? This can take the form of a diagram showing structure and hierarchy, supported by a discussion and justification of your choices. Why have you made these design choices? Describe and outline the structure of your interface and of your information.

## Visual Design

Detail your visual design: Layout, visual elements, icons, graphics, style, colour, fonts general screen designs. This can be sketches, wireframes, mockups etc, supported by a discussion, explanation, and justification of your choices.