Software Design Document

<Project Name>

s5291506 – Jamil Deris  
s5287914 – Tanish Dhir  
s5295636 – Arjan Dangol

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 purpose of the Sydney Airbnb Data Analysis Tool is to provide users with a user interface, for analysing and visualising data from the Sydney Airbnb dataset. This tool allows users to explore aspects such as listings, prices, amenities, and cleanliness factors in order to gain insights into the Sydney Airbnb market.

## System Overview

The system is a software application that enables users to interact with the Sydney Airbnb dataset using a user interface (GUI). Through this GUI users can perform functions for data analysis and visualisation. These functionalities include retrieving information about listings generating charts depicting price distribution searching for keywords analysing comments related to cleanliness and discovering insights.

## Potential Benefits

Real estate professionals can utilise this tool to understand pricing trends across suburbs. Travel enthusiasts can easily find properties based on amenities like pools, pet friendliness, furnished accommodations, locations etc.  
Airbnb hosts have the opportunity to analyse cleanliness related comments in order to enhance their listings. Researchers can extract insights, into customer preferences and behaviour within the Airbnb market.

# Requirements

## User Requirements

The client wants to provide good staying services to customers want this tool, to enable customers have various options to stay when they visit Sydney. This will be helpful for room owners as this can help them to improve their services. The software feature of producing a chart of pricing can help research and surveys to see trend of pricing of rooms as well as which month has more visitors in Sydney.

Client wants detailed listing of rooms, features, cost, feedback when customer searches for an area. The search should show rooms based on filter what customer enter by default it should present rooms with good feedback. There should be filter options like cost, feedback, list date.

## Software Requirements

R1.1 The program shall accept user input through a list of suburbs.

R1.2 The program shall show what rooms are available and show brief description about them.

R1.3 It shall have a date option to see for a particular date.

R1.4 It shall present a price for each room including details of payment.

R1.5 It shall display feedback chart of a room on basis of cleanliness, market, public transport, attractions. These will have values which indicate what is the review of the room. These are selected as tourists will look for these features in a room as it makes their visit a pleasant experience.

R1.6 The program shall present a price distribution chart when user selects a suburb and click on view price distribution chart.

R1.7 The program shall present how many times a property has been used for a user selected date.

## Use Cases & Use Case Diagrams

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

|  |  |
| --- | --- |
| Use Case ID | 1 |
| Use Case Name | View Listings |
| Actors | User |
| Description | Users will be able to view the rooms in one suburb at a given time. |
| Normal Flow | 1. User goes to the program website. 2. Selects time period. 3. Selects a suburb in the list. 4. Program shows all the rooms. |
| Alternate Flow | None |

A diagram of software

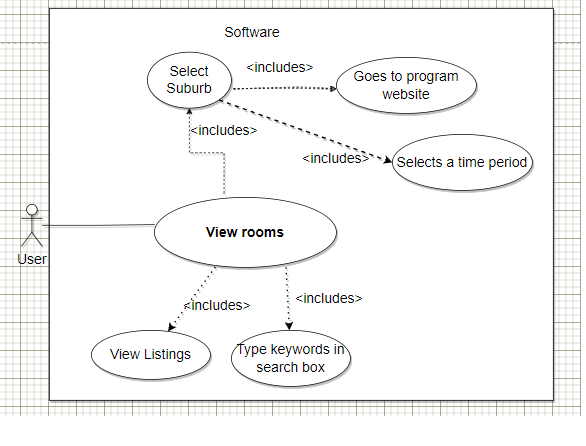
Description automatically generated

|  |  |
| --- | --- |
| Use Case ID | 2 |
| Use Case Name | View price distribution chart |
| Actors | User |
| Description | Users will be able to view the room prices in chart form in one suburb |
| Normal Flow | 1. User goes to the program website. 2. Selects time period. 3. Selects a suburb in the list. 4. Program lists all rooms. 5. User select show price chart button. 6. Program creates a chart form of prices of rooms. |
| Alternate Flow | None |

A diagram of a product

Description automatically generated

|  |  |
| --- | --- |
| Use Case ID | 3 |
| Use Case Name | View user specific rooms |
| Actors | User |
| Description | Users will be able to view the rooms with desired features |
| Normal Flow | 1. User goes to the program website. 2. Select time period 3. Selects a suburb in the list. 4. Program lists all the rooms. 5. User then type words like pool, spa and other words, then select search button. 6. Rooms with user entered word are presented. |
| Alternate Flow | None |



|  |  |
| --- | --- |
| Use Case ID | 4 |
| Use Case Name | View review chart |
| Actors | User |
| Description | Users will be able to see the feedback of rooms in a chart form. |
| Normal Flow | 1. User goes to the program website. 2. Selects a suburb in the list. 3. Program shows all the rooms. 4. User selects a room. 5. Clicks on show review chart. 6. Program shows chart of the room. |
| Alternate Flow | None |

A diagram of a software

Description automatically generated

|  |  |
| --- | --- |
| Use Case ID | 5 |
| Use Case Name | Show number of times a property has been used |
| Actors | User |
| Description | Users will be able to see how many times a property has been used. |
| Normal Flow | 1. User goes to the program website. 2. Selects a suburb in the list. 3. Program shows all the rooms. 4. User selects a room. 5. There is information about number of times that room has been used. |
| Alternate Flow | None |

A diagram of a software

Description automatically generated

# Software Design and System Components

## Software Design

A diagram of a data visualization software

Description automatically generated

Figure 1: Software design of the system

The figure demonstrated above represents a block diagram of the software design. It consists of all the essential components interconnected with each other showing the workflow within the system. Here user can extract the listing/visualisation of data from the Sydney Airbnb open dataset in the system. This system meets all the user requirements mentioned above. In the block diagram, the search engine represents a mechanism through which all the user’s action (select suburb, select period, search keyword, select room) is processed with the Sydney Airbnb dataset giving out the required responses. The responses are then represented as simple listings or visualisation.

## System Components

### Functions

1. **view\_listing(suburb\_name: string, date\_from: datetime, date\_to)**: Stores listings of all the rooms available in a specific suburb with the provided name of that suburb and a period (date range).
   * Input: suburb\_name (name of that suburb or area), date\_from (starting date), date\_to (end date)
   * Side effects: Updates the listings variable with the results in the system.
   * Return Value: Listings in a specified suburb with selected columns.
2. **create\_pricing\_chart(column\_name: string, date\_from: datetime, date\_to: datetime)**: Creates a visualisation chart showing the distribution of prices amongst all the properties with user specified date range.
   * Input: column\_name (here column\_name is “price”), date\_from, date\_to
   * Side effects: Creates a chart based on the distribution of the prices.
   * Return Value: A chart showing the distribution of prices.
3. **view\_user\_specific\_rooms(keyword: string, date\_from: datetime, date\_to: datetime)**: Creates listings of all the rooms that consists of the keyword and date range specified by the user.
   * Input: keyword, date\_from, date\_to
   * Side effects: Update the listings respective to the filtered data through selected keyword.
   * Return Value: All the records containing the specified keyword.
4. **view\_review\_chart(listing\_id: integer, keyword: string)**: Creates a visualisation chart displaying number of customers commented in a specific property (i.e., listing\_id) relevant to specified keyword.
   * Input: listing\_id, keyword
   * Side effects: Updates the listing reviews variable.
   * Return Value: A record of comments on selected property and keywords leading to visualisation chart.
5. **view\_property\_used\_times(listing\_id: integer, date\_from: datetime, date\_to: datetime)**: Displays the number of times that property has been used within user specified date range.
   * Input: listing\_id, date\_from, date\_to
   * Side effects: Updates the listing data with total number of the specific listing used.
   * Return Value: Number (total number indicating the use of that room).

### Data Structures / Data Sources

1. **listings:** 
   * It is an **array** of objects with different columns regarding the Sydney Airbnb dataset.
   * It consists of list of objects that has data members such as, **id**, **listing\_url**, **name**, **description**, **street**, **city**, **state**, **zipcode**, **country**, **price**, etc.
   * It is used in every function to filter or extract user specific data.
2. **listing\_data:** 
   * It is an **object** data structure that consists of different columns.
   * It consists of data members such as, **id, listing\_url, name, description, street, city, state, zipcode, country, price,** etc.
   * The functions that use this data are: **view\_property\_used\_times()** and **view\_review\_chart()**.
3. **listing\_reviews:** 
   * It is an **array** of objects with specified listing\_id and its reviews in each object.
   * It consists of list of objects that has data members such as, **listing\_id, id, date, reviewer\_id, reviewer\_name**, and **comments**.
   * This is the data that is retrieved by using **view\_review\_chart()** function.
4. **keyword:** 
   * It is a **string** that holds data specified by the user.
   * The functions that use this data are: **view\_user\_specific\_rooms(),** and **view\_review\_chart().**
5. **date\_from:** 
   * It is a **datetime** data structure which represents the start date of the period to extract the listings.
   * The functions that use this data are: **view\_listing (), create\_pricing\_chart(), view\_user\_specific\_rooms(), view\_review\_chart(),** and **view\_property\_used\_times().**
6. **date\_to:** 
   * It is a **datetime** data structure which represents the end date of the period to extract the listings.
   * The functions that use this data are: **view\_listing (), create\_pricing\_chart(), view\_user\_specific\_rooms(), view\_review\_chart(),** and **view\_property\_used\_times().**
7. **column\_name:** 
   * It is a **string** data structure that represents the specific column of the actual dataset which needs to be extracted.
   * The function that uses this data is **create\_pricing\_chart().**
8. **listing\_id:** 
   * It is an **integer** that represents the unique identifier for a specific property.
   * The functions that use this data are: **view\_review\_chart(),** and **view\_property\_used\_times().**
9. **suburb\_name:** 
   * It is a **string** type that represents the specific name of the suburb of which the listings are needed to be extracted.
   * The function that uses this data is **view\_listing ().**

### Detailed Design

1. Algorithm: UserSpecificRooms
   * Input: keyword, date\_from, date\_to
   * Output: Filtered listings (with user specified keywords in them)
   * Pseudocode:
     1. Initialize variables: listings = [], specific\_keyword\_listings = []
     2. Select suburb, start date, and end date to call view\_listing function (with listings CSV file).
     3. Store the data in listings variable.
     4. Filter the listings data with the specified keyword:
        + - Push the object containing specified keywords to the specific\_keyword\_listings array.
     5. If length of specific\_keyword\_listings is greater than 0:
        + Return specified\_keyword\_listings.
     6. Else:
        + Return 0 (i.e., there are no rooms relevant to the specified keyword)
     7. End.
2. Algorithm: CreatePricingChart
   * Input: ColumnName, DateFrom, DateTo
   * Output: Visualisation Chart (listing\_id, name, and price)
   * Pseudocode:
     1. Initialize variable: listings = []
     2. Select start date, and end date to call view\_listing function (with listings CSV file).
     3. Store the data in listings variable.
     4. Map listings data with column\_name as “price” to visualisation chart library in python.
     5. Display the chart with data formed in ascending format.
     6. End.
3. Algorithm: ViewReviewChart
   * Input: listing\_id, keyword
   * Output: Total number of comments
   * Pseudocode:
     1. Initialize variables: reviews\_listings = [], filtered\_listings = [], specific\_keyword\_listings = [], total\_reviews = 0
     2. Select start date, and end date to call view\_listing function (with reviews CSV file).
     3. Store the data in reviews\_listings variable.
     4. Filter the reviews\_listings data with specified listing\_id:
        + Push the object containing specified listing\_id to the filtered\_listings array.
     5. Filter the filtered\_listings data against comments column with the specified keyword:
        + - Push the object containing specified keywords to the specific\_keyword\_listings array.
     6. If length of specific\_keyword\_listings is greater than 0:
        + Calculate the length of specific\_keyword\_listings array.
        + Store it in total\_reviews variable.
        + Return total\_reviews.
     7. Else:
        + Return 0 (i.e., there are no comments relevant to the specified keyword)
     8. End.
4. Algorithm: ViewPropertyUsedTimes
   * Input: ListingId, DateFrom, DateTime
   * Output: TotalUsedTimes
   * Pseudocode:
5. Initialize variables: total\_used\_times = 0, listings = [], listing\_id = null
6. Select suburb, start date, and end date to call view\_listing function (with listings CSV file).
7. Store the data in listings variable.
8. Select listing\_id and store it in listing\_id variable.
9. Loop through each row in the listings data for the selected listing\_id:
   * + Increment total\_used\_times.
     + If total\_used\_times is greater than 0:
       - Return total\_used\_times.
     + Else:
       - Return 0 (i.e., that room hasn’t been used in the specified variables).
10. End.

# 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.