Software Design Document

<Project Name>

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 dataset includes the following Airbnb activity components:

- Listings: detailed listings data including full descriptions and average review score.

- Calendar: detailed calendar data for listings, including listing id and the price and availability for that day.

- Reviews: detailed review data for listings including unique id for each reviewer and detailed comments.

- Listings-Summary: summary information and metrics for listings (good for visualisations).

- Reviews-Summary: summary Review data and Listing ID (to facilitate time-based analytics and visualisations linked to a listing).

- Neighbourhoods: neighbourhood list for geo filter. Sourced from city or open-source GIS files.

Data input: users interact with the tool through a graphical user interface to select time periods, property type, property name, suburbs, cleanliness, price, and keyword for analysis

Data output: The tool will provide chart depicting the distribution of property prices and summaries based on user selections.

The tool solves sever problems:

- It helps users understand the Sydney Airbnb market, including pricing trends, listing details, and cleanliness-related insights.

- Hosts can optimize their listings, and travellers can make informed choices.

## System Overview

The system will cover these features:

- Data reporting: report the information of all listings in a specified suburb. This includes listing details such as property name, room types, price, review, etc.

- Keyword-based search: retrieve all records that contains a keyword (users entered). This feature enables users to find listings that match their preference.

- Price distribution chart: produce a chart to show the distribution of price of properties

- Cleanliness analysis: analysing how many customers commented on factors related to cleanliness.

- Time analysis: perform an innovation analysis, exploring booking trends over time

## Potential Benefits

**Suburb analysis**:

The geographical visualization of listing density across different suburbs helps users identify high-demand areas for the homestays, which is beneficial for travellers to seek convenient locations.

**Price distribution chart**:

Property owners can optimize their revenue by looking at price distribution chart, helping them increase their revenue.

**Cleanliness analysis**:

By analysing the cleanliness, property managers can align their homestay to meet their customer requirements, increasing the reputation

# 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 program will enable users to do:

1. View listings in a specified suburb:

When user access the application, they use search bar to input their preferred suburb and enter the date range they want to look at in the form DD/MM/YYYY to DD/MM/YYYY. Once they clicked the button ‘Search’, system then processes user’s request and retrieve listings based on their chosen time.

1. Price distribution chart:

Users click on the “Price distribution chart” and it will open a new window which displays a visual diagram of the price range for all previously selected filters by user.

1. Keyword-based search:

Users click on the “Keyword search” and it will open a new window, then users can search for specific terms. Once selected, user clicks “Go” button which displays data filtered by their keyword choice.

1. Analysing the cleanliness

Users click on the “Keyword search” and it will open a new window, then users can use predefined keyword. Once selected, the dataset will display all listings that contain the following words related to their choice.

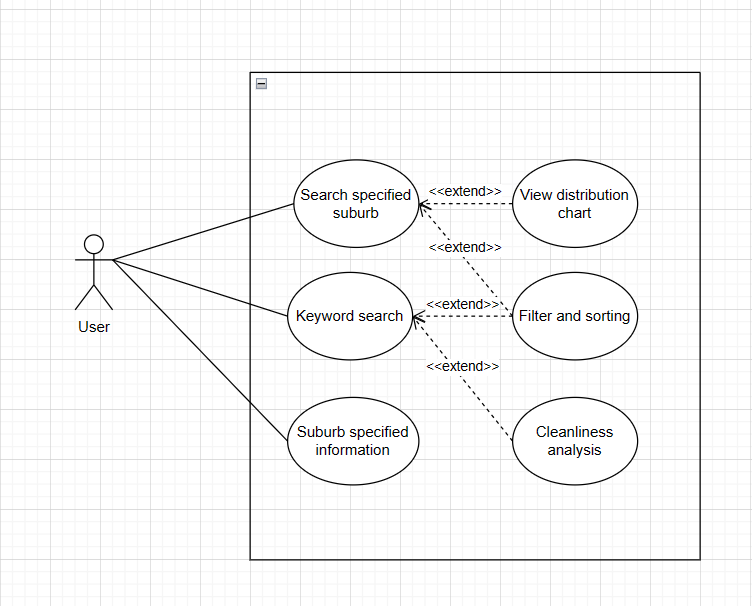
1. Filter and sorting:

When users retrieve listings based on their choice, sidebar accommodates filtering options for result refinement, including price range, rating, property type, room type. Once entered, user will click on the “Apply” button and the dataset will be updated.

## Software Requirements

* The system shall retrieve and store the Airbnb dataset, including Listings, Calendar, Reviews, Listings-Summary, Reviews-Summary, and Neighbourhoods.
* The dataset shall be updated monthly to ensure access to the latest data.
* Users shall have the option to select a specific time period (e.g., month or year) for data analysis.
* Users shall be able to input the name of a suburb in Sydney to narrow down data analysis to a specific area.
* Users shall be able to generate a price distribution chart for the selected time period, illustrating the distribution of property prices.
* Users shall have the ability to perform keyword searches (e.g., "pool" or "pet-friendly") to filter listings based on amenities and features.
* The system shall provide a tool for users to analyse cleanliness-related keywords in customer reviews.
* The system shall provide users with the ability to explore detailed information about Airbnb listings within the specified suburb and time period.
* Users shall have the option to filter listings based on various criteria, such as price range, number of bedrooms, or property type.
* The system shall calculate the number of customers who commented on cleanliness.

## Use Cases & Use Case Diagrams



|  |  |
| --- | --- |
| Use case ID | 1 |
| Use Case Name | Search specified suburb |
| Actors | User |
| Description | User should be able to view the specified suburb that they selected over the time |
| Flow of Events | 1. User accesses the application 2. User inputs their preferred suburb and enter the date range 3. System shows the listings user entered |
| Alternate Flow | None |

|  |  |
| --- | --- |
| Use case ID | 2 |
| Use Case Name | Keyword search |
| Actors | User |
| Description | User should be able to view the specified term that they search. |
| Flow of Events | 1. User access the application 2. User enters the keyword and click on ‘go’ button 3. A new window will pop out 4. System displays the listings that user entered |
| Alternate Flow | None |

|  |  |
| --- | --- |
| Use case ID | 3 |
| Use Case Name | Price distribution chart |
| Actors | User |
| Description | User should be able to create the price distribution chart from the previous research |
| Flow of Events | 1. User use their previous information search. 2. User clicks on the “Distribution chart” button 3. System produces the price distribution chart 4. New window will pop out 5. User views the price distribution chart |
| Alternate Flow | None |

|  |  |
| --- | --- |
| Use case ID | 4 |
| Use Case Name | Specified information suburb |
| Actors | User |
| Description | User should be able to view the specified information suburb |
| Flow of Events | 1. User access the application 2. User selects property type and room type then clicks “Search” button 3. Window displays a list containing all suburbs in original dataset sorted by average price for the user specified property type and room. |
| Alternate Flow | None |

|  |  |
| --- | --- |
| Use case ID | 5 |
| Use Case Name | Sorting and filtering |
| Actors | User |
| Description | User should be able to sort and filter the listings |
| Flow of Events | 1. User specify the information they want to sort 2. User click on “Apply” button 3. System will update the dataset and show it to user. |
| Alternate Flow | None |

# Software Design and System Components

## Software Design

A block diagram/flowchart of how your software might work

## System Components

### Functions

displayDashboard()

* Description: Displays the main dashboard with an overview of tool capabilities and market trends.
* Input Parameters: None.
* Side Effects: Renders the user interface component for the dashboard
* Return Value: None

Loaddata(filePath: str)

* Description: Loads the Airbnb dataset from a specified file path into memory
* Side effects: Loads the dataset into memory for analysis.
* Input Parameters: filePath (string) - The path to the dataset file.
* Return value:

selectTimePeriod(timePeriod: str)

* Description: Allows the user to select a specific time period (e.g., month or year) for data analysis.
* Input Parameters: timePeriod (string) - The selected time period.
* Side Effects: Sets the selected time period for analysis.
* Return Value: None.

SelectListingdata(timePeriod: str)

* Description: Once user select time period, system will find listing data matched the time period
* Input Parameters: timePeriod (string) - The selected time period.
* Side Effects: none
* Return Value:

SearchingInSuburb(suburb: str, Listings: arr)

* Description: Allows the user to specify the name of a suburb for data searching.
* Side effects: Sets the specified suburb for analysis
* Input Parameters: Suburb (string) - The suburb that users specify. Listings (array of dictionary)
* Return value:

cleanlinessAnalysis(cleanlinessKeywords: list, Listing: arr)

* Description: Analyses cleanliness-related keywords in customer reviews to assess the cleanliness of listings.
* Input Parameters: cleanlinessKeywords (list) - Keywords related to cleanliness. Listings (array of dictionary)
* Side Effects: Performs analysis and displays results, including the number of comments and sentiment related to cleanliness.
* Return Value: Analysis results such as the number of comments related to cleanliness, sentiment scores, or a summary of cleanliness-related data.

keywordBasedListingSearch(keywords: str, Listings: arr)

* Description: Allows the user to perform keyword searches to filter listings based on specific amenities or features.
* Input Parameters: keywords (string) - Keywords to search for, Listings (array of dictionary) .
* Side Effects: Displays listings matching the keywords and filters.
* Return Value:

generatePriceDistributionChart(listings: arr)

* Description: Creates a graphical chart illustrating the distribution of property prices for the selected time period.
* Input Parameters: listingsdata (array of dictionary)..
* Side Effects: Generates and displays the price distribution chart.
* Return Value: None.

### Data Structures / Data Sources

Data structure in the program will include:

1. Listings data structure:

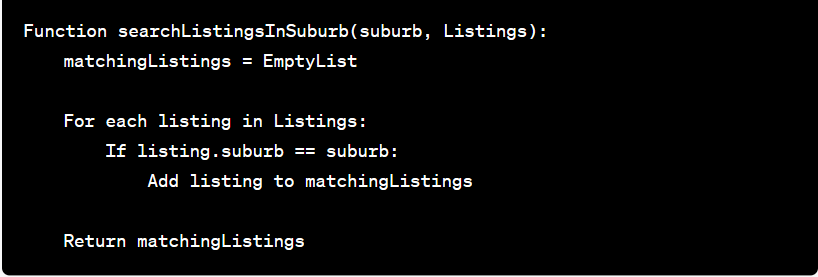
* Type of structure: List of Dictionaries
* Description: represents detailed information about Airbnb listings
* Data member: listing id, host location, property type, property name, number of bedrooms, number of bathrooms, occupancy capacity, rating and price.
* Functions that will use it: generatePriceDistributionChart, keywordBasedListingSearch, specifySuburb, loaddata, selectTimePeriod.

1. Review data structure:

* Type of structure: Dictionaries
* Description: detailed review data for listings including unique id for each reviewer and detailed comments
* Data member: listing id, id, date, reviewer id, reviewer name, comments
* Function that will use it: cleanlinessAnalysis, loaddata, selectTimePeriod.

### Detailed Design

There are pseudocodes for non-trivial algorithms that operate on data structures



# User Interface Design

For the overall design of our program, we utilised draw.io, a wireframing tool, to create a visual representation of the user interface. This enabled us to iteratively refine the layout and functionality before moving to the development phase. Our key findings that guided this design include:

* User-Centric Approach:

Our search functionality encourages users to filter their search based on their preferences. By allowing users to input their preferred suburb, desired date range, price range, minimum rating, property type and room type, we ensure that the search results are personalised to their requirements.

* Information Hierarchy:

We recognised the importance of presenting essential information upfront. The concise display of listing details, including bedrooms, bathrooms, ratings, prices, and property types, facilitates quick decision-making by providing users with the most pertinent information at a glance. Other information contained in the original datasets that we do not think is as important will not be displayed on the program to avoid overstimulating the user.

* Data Visualisation:

The inclusion of a price distribution chart enhances user understanding of the pricing landscape within their selected filters. This visual aid simplifies the interpretation of pricing ranges and supports users in making well-informed choices.

Key findings that influenced our visual design approach include:

* Colour Coding:

We incorporated colour coding to enhance user orientation. By using distinct blue buttons for data updates and orange buttons for opening new windows, users should find it easy to navigate through the interface. The strategic use of colour, helps users intuitively identify the purpose of each button and action, contributing to a smooth navigation experience.

* Readability and Typography:

Choosing simple and easy-to-read fonts like Georgia, Arial, or Times New Roman reflects our dedication to creating an interface that's visually appealing and user-friendly. This ensures that users can effortlessly grasp the information without any hassle.

* Clean Layout:

To maintain a clean and organised look, we chose to open new windows when creating a new dataset (keyword search, price distribution chart or suburb specific information). This decision enhances the user experience, preventing the interface from becoming cluttered with excessive data.

In the upcoming sub-sections, we delve into the finer details of the structural design, including the search functionality, result display, filters and sorting options, and additional features like the price distribution chart, keyword search, and suburb insights. This comprehensive design approach aims to provide users with an intuitive, informative, and visually pleasing interaction with the program.

## Structural Design

**Home Page:**

**Search Functionality:**

* Users use search bar to input their preferred suburb.
* Users can enter the date range they want to look at in the form DD/MM/YYYY to DD/MM/YYYY

No data will be displayed unless Suburb is entered. Users can initiate the search by clicking the "SEARCH" button.

**Result Display:**

A list of Airbnb listings matching the chosen suburb and date range is displayed.

Basic listing information is presented, including:

* Name
* Number of bedrooms
* Number of bathrooms
* Occupancy capacity
* Rating
* Price
* Property type (e.g., apartment, house, etc.)
* Superhost status

Users can access more details by clicking the "View More" button associated with each listing which will open the URL of the AirBnB listing in their preferred browser.

**Filters and Sorting:**

A sidebar accommodates filtering options for result refinement:

* Price Range: Users can set minimum and maximum price limits for result filtering.
* Rating: Users can exclude lower-rated listings by specifying a minimum rating.
* Property Type: Users can opt for specific accommodation types (e.g., apartments).
* Room Type: Users can opt for specific accommodation types (e.g., apartments).

Once entered, user will click on the “Apply” button and the dataset will be updated.

Sorting preferences can be tailored via dropdown menus when the title of a column in the dataset is clicked on:

* Name: A-Z or Z-A
* Type: A-Z or Z-A
* Bedrooms: Smallest to Largest or Largest to Smallest
* Bathrooms: Smallest to Largest or Largest to Smallest
* Occupants: Smallest to Largest or Largest to Smallest
* Rating: Smallest to Largest or Largest to Smallest
* Price: Smallest to Largest or Largest to Smallest

**Results Overview:**

An information snippet at the bottom right corner provides the count of listings that match the current filters.

**Open new window buttons (orange boxes on wireframes):**

* Price Distribution Chart
* Keyword Search
* Suburb Insights

**Price Distribution Chart:**

* Displays a visual diagram of the price range for all previously selected filters by user (suburb, date range, min rating, property type and room type)

**Keyword Search:**

* Users can search for specific terms or select predefined keywords (e.g., cleanliness, pool, parking, wi-fi, air conditioning, etc.).
* Once selected, user clicks “Go” button which displays data filtered by their keyword choice.

When the user clicks on the **“cleanliness”** checkbox, the dataset will display all listings that contain the following words:

* Clean
* Cleanliness
* Hygiene
* Tidy
* Tidiness
* Sanitary
* Neat
* Spotless
* Well-kept
* Dust
* Dirty
* Cleaned
* Unclean
* Dusty
* Unsanitary
* Messy
* Filthy
* Disgusting
* Shiny
* New

When the user clicks on the **“pool”** checkbox, the dataset will display all listings that contain the following words:

* Pool
* Swim
* Swimming
* Spa
* Jacuzzi

When the user clicks on the **“parking”** checkbox, the dataset will display all listings that contain the following words:

* Parking
* Garage
* Carport
* Vehicle
* Bike
* Automobile
* Parked
* Driveway
* Car
* Van
* Bus

When the user clicks on the **“wifi”** checkbox, the dataset will display all listings that contain the following words:

* WiFi
* Wi-Fi
* Internet
* WLAN
* Wireless
* Network
* Online
* Web
* NBN
* Broadband
* Hotspot

When the user clicks on the **“airconditioning”** checkbox, the dataset will display all listings that contain the following words:

* Airconditioning
* Aircon
* AC
* A/C
* Heating
* Cooling
* Fans
* Cold
* Hot
* Temperature
* Climate
* Air

**Suburb Insights:**

* User selects property type and room type then clicks Search button.
* Window displays a list containing all suburbs in original dataset sorted by average price for the user specified property type and room.
* Window displays a list containing all suburbs in original dataset sorted by average rating for the user specified property type and room.

**Justification:**

**Search Functionality:**

The search functionality is the core feature of the program. It allows users to find Airbnb listings based on their preferences. By including options to input the suburb and date range, it provides users with the flexibility to narrow down their search results.

**Results Display:**

The displayed information about each listing is concise and relevant, giving users a quick overview of what they can expect from the listings. Providing essential details like number of bedrooms, bathrooms, occupancy capacity, rating, price, and property type helps users make informed decisions without having to look through a gigantic dataset to find what they are looking for.

**Filters and Sorting:**

Filters and sorting options enhance the user experience by allowing users to refine their search results according to their specific criteria. This flexibility increases the likelihood of users finding listings that best match their preferences and requirements.

**Price Distribution Chart:**

The price distribution chart is an visual aid to help users understand the pricing landscape of the available listings. This feature can be especially helpful for users who want to see the range of prices within their chosen filters visually rather than in a dataset.

**Keyword Search:**

Including a keyword search feature helps users find listings with specific amenities or features that are important to them. This allows users to quickly narrow down their choices based on their preferences, saving time and effort.

**Suburb Insights:**

The suburb insights feature offers users valuable data for decision-making. Sorting suburbs by average price and average rating based on specific property and room types helps users identify potential suburbs that align with their budget and preferences.

**UI Design Choices:**

The use of checkboxes and dropdown menus for filtering and sorting offers a user-friendly interface. The layout of the filters and sorting options is designed to help users easily navigate and adjust their search criteria.

**View More Button:**

Providing a "View More" button that opens the Airbnb listing URL in the user's preferred browser ensures that users have easy access to detailed information about a listing. This button is designed to encourage users to explore the listing further and pull more traffic to the AirBnB website.

## Visual Design

This program is intentionally designed to be simple and user-friendly, ensuring even those unfamiliar with data can easily navigate through it. The user interface incorporates blue buttons for updating data within the current window, while orange buttons are used to open new windows. Each window is dedicated to specific content and features an appropriate title. To enhance readability, a basic and easily legible font like Georgia, Arial, or Times New Roman will be selected.

When creating a new dataset, it automatically opens in a separate window to maintain a clean and organized appearance, avoiding any visual clutter. The overall screen layout consists of a title, filters positioned either to the left or above the dataset, buttons to activate these filters, and a streamlined dataset displaying only relevant information rather than overwhelming users with extensive data.

