Software Design Document

<Project Name>

Shahab Moheb (s5264189)

Jakob Karl (s5289718)

Vaibhav Jha (s5295470)

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 document provides a comprehensive overview of the Sydney Airbnb Data System Development project, encompassing its structure, workflow, and essential components. The initial 1.0 section provides insights into the project's background and scope, setting the stage for the document's contents. Moving into section 2.0, the Work Breakdown Structure, elaborates on every crucial task and sub-task integral to the project's completion. This segment is partnered with section 3.0, which provides the finer details of each WBS task, describing their characteristics and anticipated completion timelines as well as task predecessors and what phase of the project they are. Moreover, the document's 4.0 section focuses on the Gantt Chart, offering a visual representation of the project's timeline, interdependencies, and milestones. This document provides a comprehensive resource for understanding and navigating the project's evolution.

## System Overview

The document unfolds with a meticulous overview of a system envisioned to serve users harnessing insights from the Airbnb Sydney dataset. It is a blend of several modules: from a simplified search portal and data representation instruments to feedback structures and user account handling. Every module is sculpted keeping the end-user's conveniences at the forefront, ensuring an undisturbed engagement with the dataset.

* A system that can take user input.
* A system that can report listings of specified suburbs.
* Can take user input.
* Use user input to find data in suburb.
* Report all that relate to the user’s input.
* A system that can produce a chart to show the distribution of prices of properties.
* A system that retrieves all records that contain ‘keyword’.
* Can take user input.
* Use user input to find all record of ‘keyword’.
* Display all records containing ‘keyword’.
* A system that can analyse how many customers have commented on factors related to cleanliness.
* Use dictionary to determine words that relate to cleanliness.
* Display how many customers have commented on cleanliness.
* A system that can sort listings by date.

## Potential Benefits

* User-friendly Interaction: With intuitive navigation and search features, users can easily locate and analyse data.
* Informed Decisions: Through visual representations, users can gauge property price distributions, cleanliness standards, and more.
* Tailored Insights: With user accounts, the system remembers preferences, offering personalized data views and suggestions.
* Continuous Improvement: The feedback mechanism ensures the system evolves based on user needs.

# 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 user’s 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?**

Allow users to search and view a report of all listings in a specified area.

-          Allow users to view a chart of the distribution of prices of properties.

-          Allow users to retrieve all records that contain a searched ‘keyword’.

-          Allow users to see how many customers commented on factors of cleanliness.

-          Allow users to filter the properties by date listing was posted.

## Software Requirements

Functional Requirements:

R1.1: The system shall provide a user-friendly interface for searching Airbnb listings.

R1.2: The system shall display detailed information for each Airbnb listing, such as location, price, room type, and reviews.

R1.3: The system shall generate visual representations, like charts, for specific data distributions.

R1.4: The system shall facilitate users to leave feedback or reviews for listings.

R1.5: The system shall incorporate advanced filtering and sorting capabilities for enhanced user experiences.

Non-functional Requirements:

R2.1: The system shall ensure user data protection in compliance with data privacy regulations.

R2.2: The system shall offer a responsive user interface, ensuring usability across various devices.

R2.3: The system shall maintain a consistent performance even under heavy user loads.

## Use Cases & Use Case Diagrams

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

# 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

### 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

### Detailed Design

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

# User Interface Design

The foundation of our user interface design was driven by the intent to create an intuitive and efficient user experience. In our design journey, tools like Pycharm Community, Github, Gitbash, and wxFormBuilder emerged as invaluable assets. Their collective strengths in coding, version control, terminal operations, and interface design respectively, formed the backbone of our design process. As we transition into the design's specifics, we'll explore how the structural and visual elements were conceived and iterated upon, taking cues from preliminary user feedback and best practices in UI/UX.

These tools - Pycharm Community, Github, Gitbash, and wxFormBuilder - play a vital role in the project's design and coding endeavours. They collectively contribute to the system's development by facilitating coding, version control, terminal operations, and interface design. Moving forward, section 4.1 Structural Design will utilize diagrams to elucidate the architecture of the system's design and its utilization of Airbnb data.

## Structural Design

Our software adopts a user-centric hierarchical structure. The main screens encapsulate:

* Home Screen:
* The initial interaction point, offering diverse options like searching for listings, accessing profiles, or providing feedback.
* Search Results Screen:
* Presents listings resonating with the user's query.
* Listing Detail Screen:
* Delves into the nuances of a chosen Airbnb listing.
* Visualization Screen:
* Graphical representations based on user preferences emerge here.
* Profile Screen:
* A personal space, documenting search history, bookmarked listings, and user details.

Grouping Information:

Data grouping is both logical and user oriented. On the Search Results screen, for instance, listings cluster around location or price brackets, pivoting on the user's choice.

Navigating Through the Product:

Every navigation step is designed for simplicity. A consistent navigation bar sprawls across screens, ensuring smooth transitions throughout the software. Furthermore, breadcrumbs fortify user orientation, illustrating their position within the software's hierarchy.

Our design choices emanate from a user-centric philosophy. The emphasis on clear structuring, logical information clustering, and seamless navigation aims to craft an intuitive user experience. This reduces the learning curve and amplifies user engagement.

## Visual Design

As we delve into the visual realm, wireframes become our illustrative tool. These preliminary designs breathe life into the structural concepts discussed, offering a visual interpretation. Every wireframe accentuates key interface components, ensuring their strategic placement fosters a logical flow and an enriched user experience.

Our visual paradigm champions simplicity and clarity. A palette of neutral hues ensures the content remains the centrepiece, while the spacious layout keeps visual fatigue at bay. Icons, used judiciously, guide users without inducing clutter. Fonts, chosen for their legibility, employ size variations to distinguish between primary content and subsidiary details. champions simplicity and clarity. A palette of neutral hues ensures the content remains the centrepiece, while the spacious layout keeps visual fatigue at bay. Icons, used judiciously, guide users without inducing clutter. Fonts, chosen for their legibility, employ size variations to distinguish between primary content and subsidiary details.

***Image 1: User Profile***

A screenshot of a user profile

Description automatically generated

***Image 2: Home Page***

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated***Image 3: Data Visualization***

A screenshot of a search results

Description automatically generated***Image 4: Search***

***Image 5: Feedback***

A screenshot of a computer screen

Description automatically generated