Richa Pahuja (s5353290), Dayyan Ahmed (s5280098), Zekun Liu (s5291882)

Griffith University  [Company address]

Software design document

Contents

[1. System Vision Document 2](#_Toc1479044422)

[1.1 Problem Description 3](#_Toc873897338)

[1.2 System Overview 3](#_Toc643353639)

[1.3 Potential Benefits 3](#_Toc1706357836)

[2. Requirements 4](#_Toc1437118682)

[2.2 Software Requirements 4](#_Toc759161042)

[2.2 User Requirements 6](#_Toc1005353733)

[2.3 Use Case & Use Case Diagram 8](#_Toc617308492)

[2.3.1 Brief Use Case Description 8](#_Toc188295688)

[2.3.2 Fully Developed Use Case Description 8](#_Toc1184254831)

[2.3.2 Use Case Diagram 10](#_Toc2017722328)

[3. Software Design and System Components 11](#_Toc894241160)

[3.1 Software Design 11](#_Toc515243266)

[3.2 System Components 11](#_Toc432683185)

[3.2.1 Functions 11](#_Toc558199820)

[3.2.2 Data Structures / Data Sources 12](#_Toc1062204178)

[3.2.3 Detailed Design 13](#_Toc2058259674)

[4. User Interface Design 15](#_Toc431595967)

[4.1 Structural Design 15](#_Toc907182720)

[4.2 Visual Design 16](#_Toc1689537413)

# 1. System Vision Document

## 1.1 Problem Description

There is a huge database of Airbnb activity in Sydney, however, the client is not able to navigate through it easily as it is cluttering, time-consuming, and unorganized according to clients’ requirements. Due to the clutter, clients’ interest doesn’t gravitate to go through every part of the dataset. Neither is the design interactive which should fascinate the client to spend more time researching about every Airbnb from their reviews to their ratings. Even the time duration of the reviews is not clearly stated which does make the entire method of researching quite frustrating for the client. Any information stated in a cluttering way becomes sore to the eye and instead of spending time on it and navigating through the information, the client directly closes that tab.

A website would be developed, that would provide a well-designed interface to the client with a search option, filtering his desired Airbnb according to his budget, location, rating, and reviews and sorting them further from best matches to the prices from low to high. The new system is effortless, timesaving and organized according to the client’s wants.

## 1.2 System Overview

The system capabilities proposed at high levels include:

* Digital interface for easy navigation of Data
  + Easily navigate with a search option, and interactive elements labelled.
  + Preference to narrow down the Airbnb proposed results.
  + The design is developed in an interactive and organized way.
  + Clean layout and visually appealing graphics.
* Search Functionality
  + Keyword-based searches across the website
  + Pricing, suburb-based searches are also accessible
  + Search results presented in an organized and structured way.
* Advanced Filtering
  + Users can apply filters according to their preference from pricing, ranking, date availability, along with keywords related to the property like pool or pets.
* Data Summarizing
  + The website provides a summarized overview of the data whilst highlighting trends & popular listings.
  + Clients can sort the data presented by best matches, low-high pricing, high-low ranking, rating, and most reviewed.
* Secure Data Handling
  + Secure data handling of client’s search history and any other sensitive information.
  + The website would comply with privacy regulations to ensure user data protection.

## 1.3 Potential Benefits

The expected benefits of the proposed system will include:

* The client can quickly and efficiently access data within minutes.
* The client can make better-informed decisions by looking at all trends and information in depth.
* Reduced need for extensive data search saving valuable time for the client.
* Creates a positive user experience with an interactive user-friendly design.
* Improved coordination among team members due to easy navigation of going through each dataset.
* Personalized data queries with the use of the filter.

# 2. Requirements

## 2.2 Software Requirements

This section examines the software requirements. These are the most essential properties which should be kept in mind while making a software.

|  |  |
| --- | --- |
| Requirements | |
| Software | Requirement Description |
| Usability | Website must have a well-designed and simplified user interface.  It must be compatible for both Windows and iOS.  It should be organised, simple and easy to navigate through.  The website must give a multilingual interface option as well.  It must be user-friendly and designed according to user’s preference.  The website must increase customer satisfaction. |
| Reliability | The website must be able to run flawlessly.  The website must have a 99.9 percent accuracy rate.  The website operational efficiency must not be compromised.  A backup mechanism must be present to for the website.  The standby team must be present to interact with any unforeseen outages. |
| Performance | The website servers must be compatible for high incoming traffic  The website must have a minimal average delay time.  The database must be maintained at high performance.  Real-time data adjustment must be handled by the management environment with minor delay. |
| Security | The settings for the database must be resistant to known attacks (XSS, SQL Injection, etc.).  The website must have high-security protocols.  The website must be using a secure interface.  High-risk data must be kept in the database in a non-readable and coded form. |
| Design Constraints | website and system databases can be installed using the AWS Standard M4 Server configuration.  Any modern laptop or PC  Acceptable RAM.  Dedicated EBS bandwidth of 750 Mbps.  The basic minimal requirements or newer must be opted for the website management environment to operate:  Mac iOS or Windows |
| Implementation | Adobe Photoshop must be used to design the user interfaces for the website.  For the website HTML5, CSS3, and JavaScript must be used to design the interfaces.  The website must be coded in Python and C++  MySQL must be used for database-related inquiries |
| Maintainability | Must have a very maintained architecture and design.  The website must be regularly updated. |

## 2.2 User Requirements

The new website contains a couple of subsystems with their own specification. Below are detailed explanations of user requirements. These services are provided on the website which will benefit the clients.

|  |  |  |
| --- | --- | --- |
| **Functional Requirements** | | |
| **Subsystem** | **Functions** | **Description** |
| **Listing Subsystem** | Displaying full description of the property.  Putting up the address and the contact details of the property on highlighting the suburb  Displaying an average review score of the property | The website will have the capability to list down all the properties present in the database.  The website would be able to display out full information of the property in the chosen suburb by the customer.  The website would also be able to display the result of the average review score of a specific property. |
| **Price Subsystem** | Will show the cost of living in the chosen time  Graph representation | The website will be able to manage the calendar data for listing.  The website will be giving a graphical representation for the prices in the selected suburb |
| **Key words subsystem** | Process the comments  Get the highlighted data comments | The subsystem may process client information after they have given a review.  The system would be able to find out the data using keywords |
| **Analysing cleanliness Subsystem** | Process the data with multiple keywords in respect to clean house | The website would be able to look for how clean the property was and would look for multiple words in which will give a graph score of cleanliness of the property |
| **Neighbourhood checking subsystem** | Check for the neighbourhood.  Check for the area.  Check for nearby facilities | The subsystem will be able to help people put filters and find houses that are near the beach entertainment area and grocery area.  The website will also be able to tell people about the neighbourhood of the listed property |

## 2.3 Use Case & Use Case Diagram

### 2.3.1 Brief Use Case Description

The table below shows the use case diagram

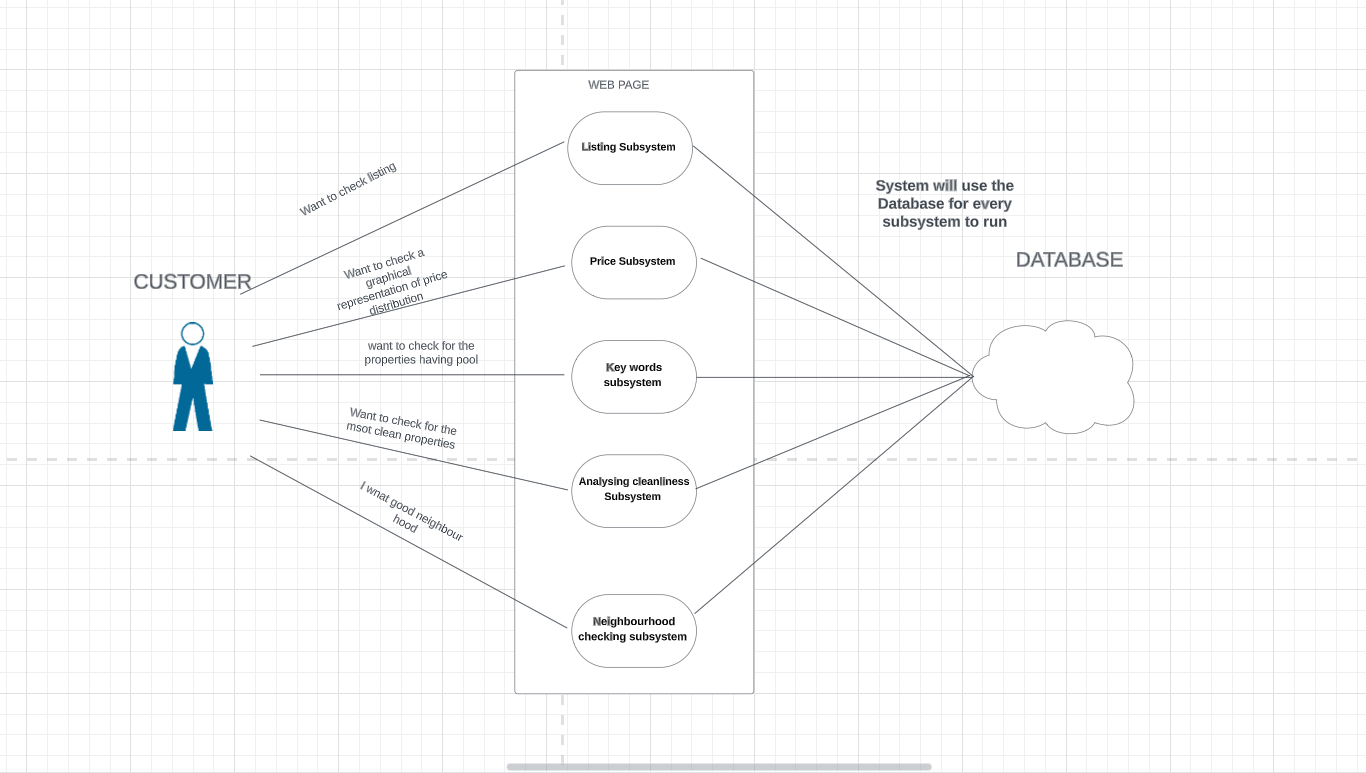
|  |  |  |
| --- | --- | --- |
| **Brief Use Case Descriptions** | | |
| **Use Case** | **Users/Actor** | **Brief Use Case Description** |
| Open up interface | Customers | The customers will be opening the interface |
| Putting up some filters like suburb | Customers | The customers will be putting some filters |
| Looking for prices | Customers | The customers will be putting the limits |
| Looking for cleanliness | Customers | The customers will be looking for how clean the house id |
| Looking at the booking history | Customers | The customers will be able to look at the graph that will help customers look for the property demands and how many days the property is booked |
| Looking at comments | Customers | The customers can look at the review submitted by the people and can look at the review score |
| Quitting the website | System, Customers | After deciding the customer would quit the website |

### 2.3.2 Fully Developed Use Case Description

The table below will be representing fully devoted use case descriptions

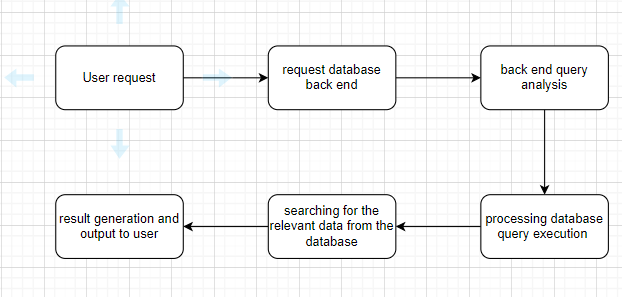
|  |  |  |  |
| --- | --- | --- | --- |
| **Fully Developed Use Case Description** | | | |
| **Use Case Name:** | Customer will be looking at the website for browsing property | | |
| **Scenario:** | A customer wants to book an Airbnb for his visit to Sydney. | | |
| **Triggering event:** | The customer wants to put some filter as per his demands | | |
| **Brief Description:** | The customer is going on Sydney for his personal reason, and he want to book an Airbnb to stay over there, and he want his Airbnb to be near beach and entertainment area so that why he visited the website, and he will use website to look for the options. | | |
| **Users/Actors:** | System, database, Customers. | | |
| **Related use cases:** | Checking the property | | |
| **Preconditions:** | Airbnb database must be available. And the system should be working properly, and the filter are placed properly | | |
| **Postconditions:** | Customers had looked around the optional made his mind on booking most suitable property | | |
| **Flow of Activities** | **Customers** | **System** | **Database** |
|  | 1 customer opened the website | 1.1 system would be able to put the customer to home interface |  |
|  | 2. Customer puts on filter and tap to search button | 2.1 system would be able to read the command and will be able to link to data base | Database would be able to provide the data to the system according to the quarries |
|  |  | 3 system displayed the list of the properties which are available according to the filters Provided by the customer |  |
|  | 4 customer browsers through the given data and would choose the most suitable property for him | 4.1 system still displaying the options |  |
|  | 5. After looking around the customer will go back to the home screen and quit the website | 5.1 System would close |  |
| **Exception Conditions:** | 2.1 System cannot get the data because of some error.  4 System will be showing wrong data because of the database being corrupted. | | |

### 2.3.2 Use Case Diagram



# 3. Software Design and System Components

## 3.1 Software Design



## 3.2 System Components

### 3.2.1 Functions

|  |  |
| --- | --- |
| loadData function | |
| Loaddata | Load data from a file or anywhere else |
| Description | Loads data from a file or external source |
| Input Parameters | file\_path (string) - Path to the data file |
| Side Effects | Loads data into memory or data structures |
| Return Value | Boolean indicating success or failure |
|  | |
| Display function | |
| Display | display the result before or after the queries |
| Description | Displays loaded data to the user |
| Input Parameters | data (list, array, etc.) - Data to be displayed |
| Side Effects | None |
| Return Value | None |
|  | |
| analysis function | |
| Selected columns: | Using the keywords to search the data from databases (single or multiple selections). Use the SQL queries to make the data more readable (user easier to understand) |
| Description | Applies algorithms to analyse the loaded data |
| Input Parameters | Data (list, array, etc.) - Data to be analysed |
| Side Effects | May modify data or generate analysis results |
| Return Value | Analysis results or data insights |

### 3.2.2 Data Structures / Data Sources

**Data Sources**

|  |  |
| --- | --- |
| DataProcessor Class | |
| Description | Manages data processing and analysis operations. |
| Methods | load\_data(file\_path: str): Loads data from the specified file.  display\_data(data): Displays data to the user.  analyze\_data(data): Applies analysis algorithms to data. |
|  | |
| DataStorage Class | |
| Description | Handles data storage and retrieval |
| Methods | store\_data(data): Stores data for future reference.  retrieve\_data(): Retrieves stored data. |

**Data Structure**

**Required Features:**

|  |  |
| --- | --- |
| For a user-selected period, report the information of all listings in a specified suburb | suburb |
| For a user-selected period, produce a chart to show the distribution of prices of properties | prices |
| For a user-selected period, retrieve all records that contain a keyword (user entered), e.g., pool, pet | (user entered keyword) |
| Analyzing how many customers commented on factors related to cleanliness (multiple key words may be associated with cleanliness – justify your selection) | cleanliness |
| One other ‘insight’ or analysis tool of your choice |  |

### 3.2.3 Detailed Design

**Function details:**

Mean Calculation Algorithm:

Description: Calculates the mean (average) of numerical data.

Algorithm: Sum all data points and divide by the number of points.

Input: Numeric data (list, array, etc.).

Output: Mean value (float).

**Pseudocode:**

MeanCalc( list):

Check input is correct format (not zero elements)

Sum all the data points from the input

Divide the sum of data points by number of points

Return results

**Function details:**

Data Clustering Algorithm:

Description: Groups similar data points together based on certain criteria.

Algorithm: Utilizes clustering techniques like k-means or hierarchical clustering.

Input: Multidimensional data (list, array, etc.).

Output: Cluster assignments or labels.

**Pseudocode:**

DataClustering(data):

Check if input data is valid if length of data is 0 or num\_clusters <= 0: return "Invalid input data or number of clusters."

Perform clustering using a chosen algorithm (e.g., k-means) clusters = PerformKMeans(data, num\_clusters)

Return the cluster assignments or labels return clusters

**Function Detail:**

Trend Analysis Algorithm:

Description: Identifies trends or patterns in time-series data.

Algorithm: Applies methods like moving averages or exponential smoothing.

Input: Time-series data (list, array, etc.).

Output: Detected trends or patterns.

**Pseudocode:**

TrendAnalysis(data):

Check if input data is valid if length of data is 0 or window\_size <= 0: return "Invalid input data or window size.

Return detected trends or patterns return trends

|  |  |
| --- | --- |
| Understand the Algorithm | Ensure that you have a deep understanding of the algorithm to be pseudo encoded. Break it down into high-level steps before delving into details |
| Choose Clear Names | Use meaningful names for variables and operations. This makes your pseudo code easier to understand |
| Start with Initialization | Variables and data structures required for initializing algorithms |
| Outline the Main Logic | Use indentation to represent different levels of control structures, such as loops and conditions. Gradually describe the main logic of the algorithm |
| Use Indentation | Appropriate indentation helps to visualize the nesting of loops, conditional statements, and other control structures |
| Describe Operations | Describe operations and calculations in simple English, avoiding programming specific syntax. For example, you can write increment x to 1 instead of writing x=x+1 |
| Use Loops and Conditionals | If the algorithm involves loops and conditions, please describe their conditions and the actions taken within them |
| Include Comments | Add comments as necessary to explain the complex steps or reasoning behind certain decisions |
| Keep It Clear and Concise | Focus on clarity and simplicity. Keep each step concise and understandable |

## 4. User Interface Design

The team will be using the draw.io to create my visual interface. The reason for choosing this software is because that’s a user friendly. I have look around for some user survey for this website which boosted my confidence for making my interface in this website. The actual design would be made using the <https://streamlit.io/> as this is a user-friendly website with a couple of tutorials which will help us in designing the real interface. We can also use PowerBI to make the interfaces.

## 4.1 Structural Design

For the structure design, I will be adopting a user friendly and easy to navigate approach. The interface will be simple and less complex. The main component for my interface would be as below

**Navigation**: I have placed the main navigation bar at the top which will consisting of the filters. This would be the main part of the interface which will ensure the user can get access to the demanded information.

**Content Grouping**: After the desired filter have been applied the user would be able to access to the analysed data of the centre of the main page where it would be displayed in different form.

**Visualisation**: User would be getting a graphical and the data representation of the data in according to the filter applied. The graph would be able to show some trend and the theoretical representation would be giving some reviews.

**Responsive Design**: My team will make sure that our interface will be able to work on wide range of screen size without being crashing down.

The justification for these choices is to make a good user experience. These choices will make our website easy to use as it would be single interface and not too complex. We can maximize the efficiency. This visual modification would help user gain a good and easy understanding of data.

## 4.2 Visual Design

Layout would be a single screen. The navigation bar is at the top. The 5 boxes at the top are for the input. The search button would be used to search. Tried to maintain everything simple and easy to use

Visual element the search button would be of dark colour and the graph would be used

Design and colour the font used all over the website will be calibre and at a font size of 14.

