A white and blue cover with yellow text

Description automatically generated

Table of Contents

[1. Installation and Setup 1](#_Toc147698912)

[2. Directory setup: 1](#_Toc147698913)

[3. Data cleaning.py 3](#_Toc147698914)

[4. App.py 4](#_Toc147698915)

[5. Home Page: 6](#_Toc147698916)

[6. Function 1: Suburb Listing 7](#_Toc147698917)

[7. Function 2: **Price Charts** 8](#_Toc147698918)

[8. Function 3: **Keyword Search in Reviews** 9](#_Toc147698919)

[9. Function 4: Cleanliness Keyword Analysis 11](#_Toc147698920)

[10.Function 5: Room Type Analysis 12](#_Toc147698921)

[11. No data available: 14](#_Toc147698922)

# Installation and Setup

To run the Sydney Airbnb Data web application, you need to have a few essential software and libraries installed within your Python environment. Inside your Python Miniconda environment, you have installed the following:

* **Flask**: Flask is a Python web framework that enables the creation of web applications. It serves as the foundation for the Sydney Airbnb Data application.
* **pandas**: The pandas library is used for data manipulation and analysis. It helps in handling and processing data efficiently.
* **datetime**: The datetime module is a Python library for working with dates and times. It is essential for handling date-related operations in the application.
* **matplotlib**: Matplotlib is a data visualization library used to create charts and graphs. It's utilized for generating visual representations of data, such as price charts.
* **re**: The re module provides support for regular expressions, which are used for text searching and manipulation, particularly in the context of analyzing reviews.

These software and libraries are crucial for the functionality of the web application, enabling data analysis, visualization, and web serving capabilities.

A screenshot of a computer program

Description automatically generated

1. Directory setup:

* **static:** This directory typically contains static assets like CSS, JavaScript, and images. You have CSV files and image files here.
  + **CSVs:**
    - **listingtemp.csv**: This file is used for storing temporary listing data.
  + **images:**
    - **pie.png**: An image file, for the house analyses chart.
    - **plot.png**: An image file, for the price distribution chart.
* **templates:** This directory contains your HTML templates for rendering web pages.
  + **date\_form.html**: The HTML template for the home page where users can input date and location information.
  + **function1.html**: The HTML template for displaying Function 1 results (Suburb Listing).
  + **function2.html**: The HTML template for displaying Function 2 results (Price Charts).
  + **function3.html**: The HTML template for displaying Function 3 results (Keyword Search).
  + **function4.html**: The HTML template for displaying Function 4 results (Cleanliness number).
  + **function5.html**: The HTML template for displaying Function 5 results (Home Type Analysis).
  + **noData.html**: The HTML template for displaying a message when no data is available.
* **app.py:** This is your Flask application's main Python script, where you define your routes, application logic, and run the app.
* **calendar\_dec18.csv:** A CSV file containing calendar data.
* **data cleaning.py:** A Python script likely used for data preprocessing and cleaning.
* **listings\_dec18.csv:** A CSV file containing listing data.
* **listings\_summary\_dec18.csv:** A CSV file containing summarized listing data.
* **reviews\_dec18.csv:** A CSV file containing reviews data.

A screenshot of a computer program

Description automatically generated

# Data cleaning.py

In the initial steps of our data analysis process, we employ a Python script named "data cleaning.py" to ensure that the datasets are well-prepared and devoid of any inconsistencies. This script is vital for data quality assurance before further analysis.

Firstly, we utilize the pandas library to read the data from CSV files, including listings, calendar, reviews, and listings\_summary for December 2018. We focus on cleansing and refining these datasets to enhance usability.

For the "listings" dataset, we add a "Date" column by converting the "last\_scraped" column to a datetime format. We then select specific columns of interest, renaming them for clarity, and save the updated dataset back to "listings\_dec18.csv."

The "calendar" dataset is processed next. We set the "dayfirst" parameter to ensure proper date parsing and drop any rows with missing values. The cleaned data is then saved to "calendar\_dec18.csv."

Subsequently, we handle the "reviews" dataset by converting the "date" column to datetime and removing rows with missing comments. We select only the "date" and "comments" columns and save the cleaned data as "reviews\_dec18.csv."

Lastly, we clean the "listings\_summary" dataset by removing rows with missing values and selecting relevant columns, primarily focusing on "last\_review" and "room\_type." The resulting dataset is saved as "listings\_summary\_dec18.csv."

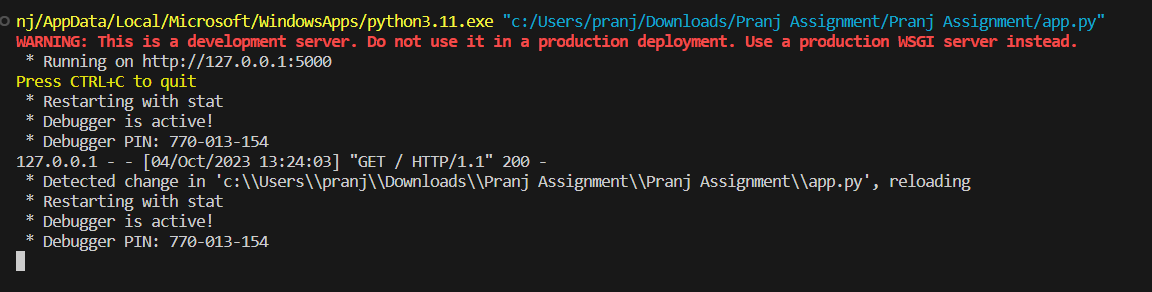
This data cleaning process sets the stage for subsequent data analysis, ensuring that our datasets are consistent, complete, and ready for further exploration.

A screen shot of a computer

Description automatically generated

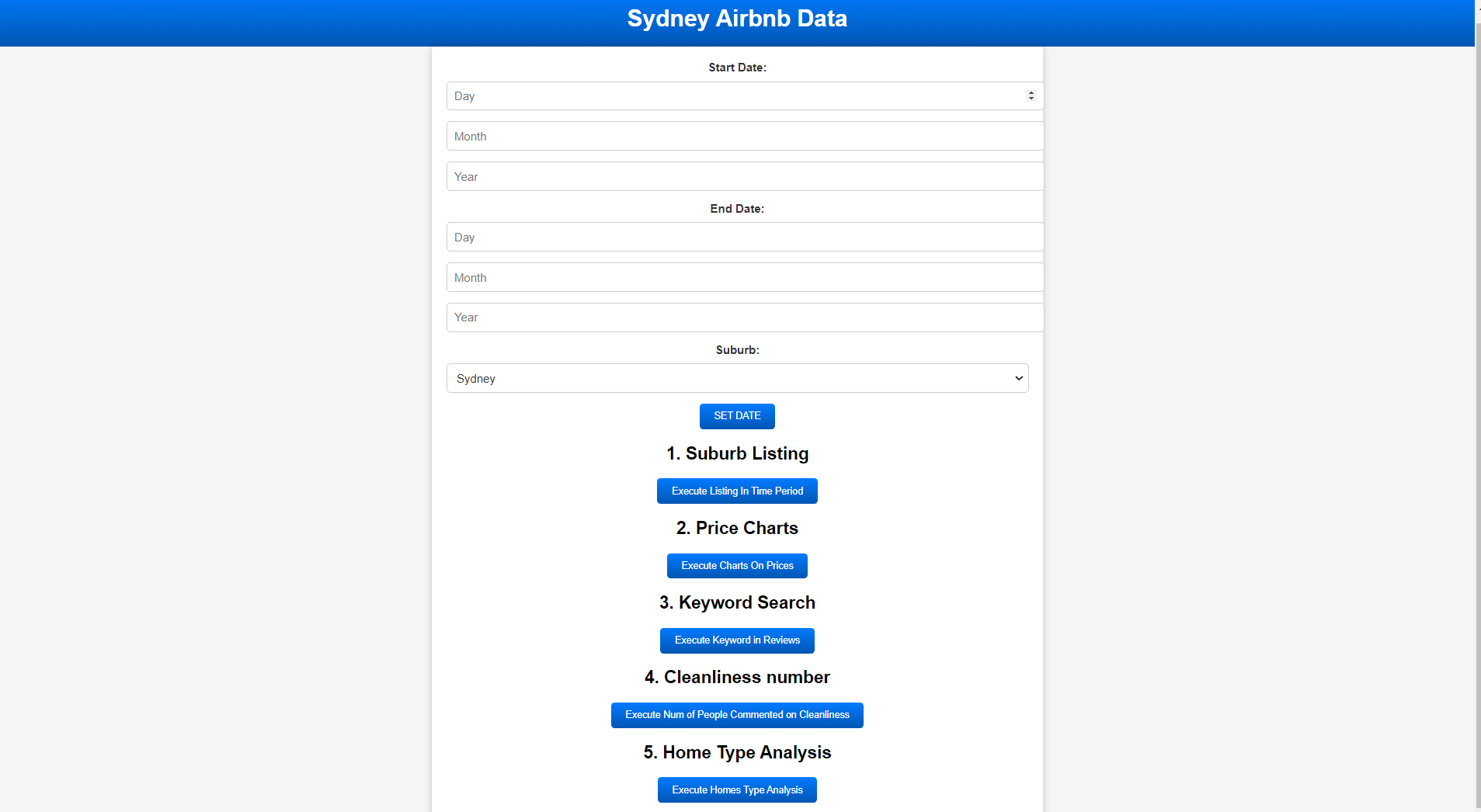
# App.py

1. **Importing Libraries**: This section imports the necessary Python libraries, including Flask for web functionality, pandas for data manipulation, datetime for date handling, matplotlib for creating charts, and re for regular expressions.
2. **Initialization**: It initializes **start\_date** and **end\_date** variables with default date values. These variables will later be used for filtering data within a selected time period.
3. **Functions**:
   * **listingFunc**: This function reads and filters listing data based on a selected suburb and count.
   * **chartPrices**: Reads and processes calendar data to create a price distribution chart within the specified date range.
   * **numCommented**: Analyzes the number of comments containing specific keywords within the selected time frame.
   * **areaAnalysis**: Analyzes the type of accommodations (shared, private, entire home) in a selected suburb.
4. **Flask Initialization**: The Flask app is initialized, and the **static** folder is specified for serving static assets like CSS, JavaScript, and images.
5. **Routes**:
   * **/**: The home page route, where users can input date and location information.
   * **/submit**: Handles form submission to update **start\_date**, **end\_date**, and perform data analysis functions.
   * **/function1**: Displays the results of Function 1 (Suburb Listing) as an HTML table.
   * **/function2**: Displays the price distribution chart (Function 2).
   * **/function3**: Handles keyword search for reviews and displays highlighted comments (Function 3).
   * **/function4**: Displays the count of comments containing specific words (Function 4).
   * **/function5**: Displays a pie chart of accommodation types (Function 5).
6. **Keyword Highlighting**: There's a function **highlight\_keyword** to highlight specific keywords in comments, which is used in Function 3.
7. **Run the Application**: Finally, the app is run with **app.run(debug=True)**.



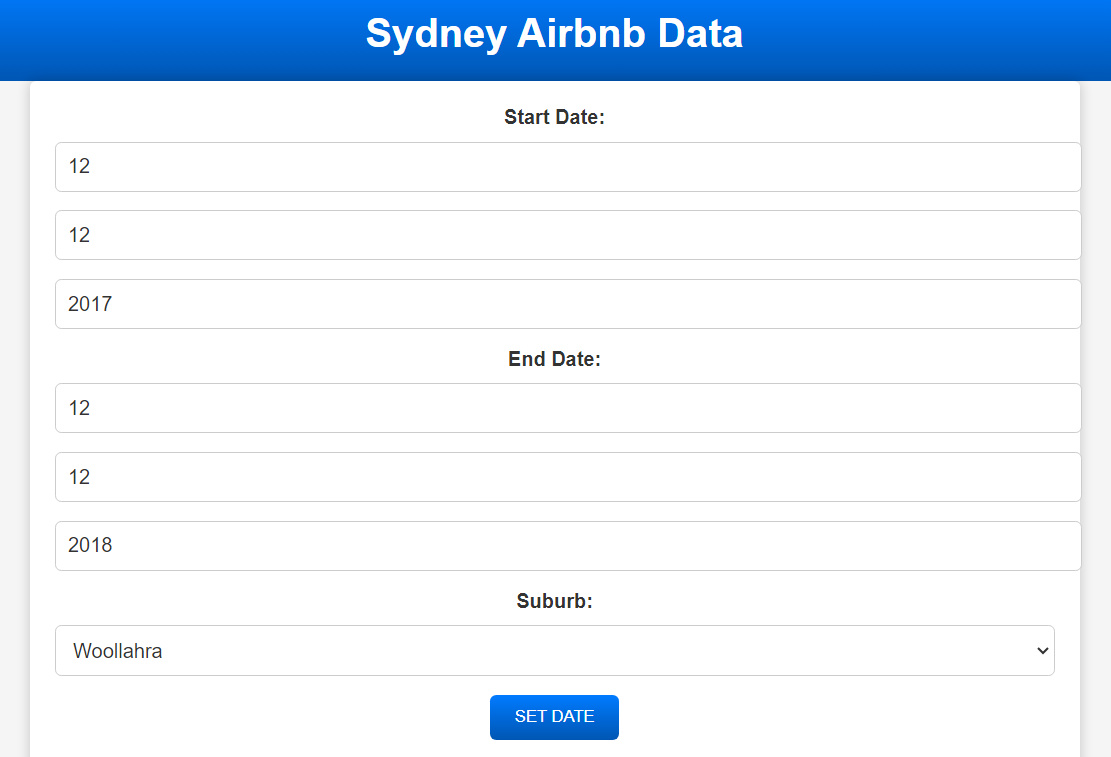
To access Flask web application, follow this link: [http://127.0.0.1:5000](http://127.0.0.1:5000/)

Click on the link, and it will take you to the home page of your web application.



# Home Page:

1. **Set the Date for Functions 1, 2, 4, and 5**:
   * On the home page, you will see the "Start Date" and "End Date" fields. These are used to set the date range for Functions 1, 2, 4, and 5.
   * Enter the desired start and end dates by specifying the day, month, and year in the corresponding input fields.
   * Note that Function 3 (Keyword Search) does not depend on the date, so you don't need to set the date for this function.
2. **Select the Suburb for Functions 1 and 5**:
   * On the home page, you will also find the "Suburb" dropdown menu. This menu allows you to select the suburb for Functions 1 and 5.
   * Choose the suburb you want to analyze from the dropdown list.
3. **Execute the Functions**:
   * Once you have set the date (for Functions 1, 2, 4, and 5) and selected the suburb (for Functions 1 and 5), you can execute the functions by clicking the corresponding buttons:
     + Function 1: "Execute Listing In Time Period"
     + Function 2: "Execute Charts On Prices"
     + Function 4: "Execute Num of People Commented on Cleanliness"
     + Function 5: "Execute Homes Type Analysis"
   * Clicking these buttons will perform the respective data analysis and display the results.



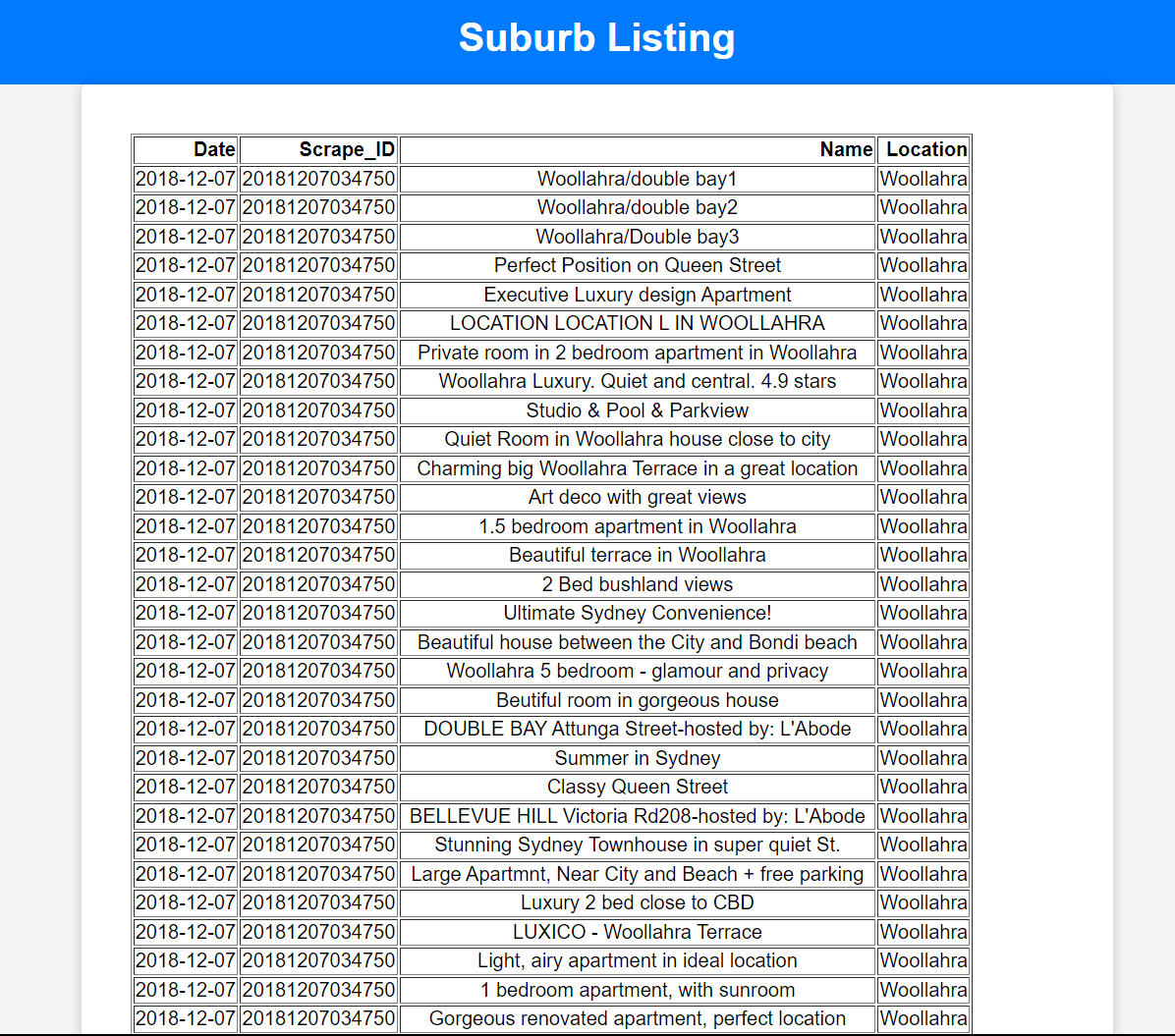
1. Function 1: Suburb Listing

Function 1, "Suburb Listing," allows you to retrieve a list of Airbnb listings in a specific suburb within a given time period. In this example, the selected suburb is "Woollahra," and the specified time period is from December 12, 2017, to December 12, 2018.

Here's what the function does:

* The function collects data from the "listings\_dec18.csv" file, which contains information about Airbnb listings.
* It filters the data based on the selected suburb, which is "Woollahra" in this case.
* The function also allows you to limit the number of results displayed (in this example, it displays the top 100 listings).
* After filtering the data, it displays a table with columns for Date, Scrape\_ID, Name, and Location. These columns provide information about the listing's date, scrape ID, name, and location (suburb).
* Each row in the table represents a unique Airbnb listing within the specified suburb and time period.

In the output, you can see a sample of the Airbnb listings in Woollahra, including details such as the listing's name and location. This information can be valuable for analyzing and comparing different Airbnb properties in the chosen suburb.



# Function 2: **Price Charts**

Function 2, "Price Charts," generates a histogram chart that visualizes the distribution of Airbnb property prices within a specified time period. In this example, the selected time period is from November 10, 2018, to December 22, 2018. The chart is designed to help users understand the range and frequency of prices for Airbnb listings during this time frame.

Here's how Function 2 works:

1. The function reads data from the "calendar\_dec18.csv" file, which likely contains information about Airbnb property prices and availability.
2. It processes the data, converting the "date" column to datetime format and removing the "$" sign from the "price" column to ensure it's a numeric value.
3. The data is then filtered to include only entries that fall within the specified time period (November 10, 2018, to December 22, 2018).
4. The function creates a histogram chart using Matplotlib. The histogram represents the distribution of Airbnb property prices, with values grouped into custom-defined value ranges (bins) such as 100-150, 150-200, 200-250, 250-300, etc.
5. The x-axis of the chart displays the value ranges (e.g., 100-150, 150-200), and the y-axis shows the frequency or count of properties falling into each range.
6. The chart is saved as an image file, typically named "plot.png," which can be displayed to the user.

When you click on "Function 2," you should see a histogram chart showing the distribution of Airbnb property prices within the specified time period. This chart helps users quickly grasp the price range most commonly found among the listings, providing valuable insights for planning a stay in the area. For example, users can see whether most listings fall within a certain price range or if there are outliers with significantly higher or lower prices.

A graph of a price distribution

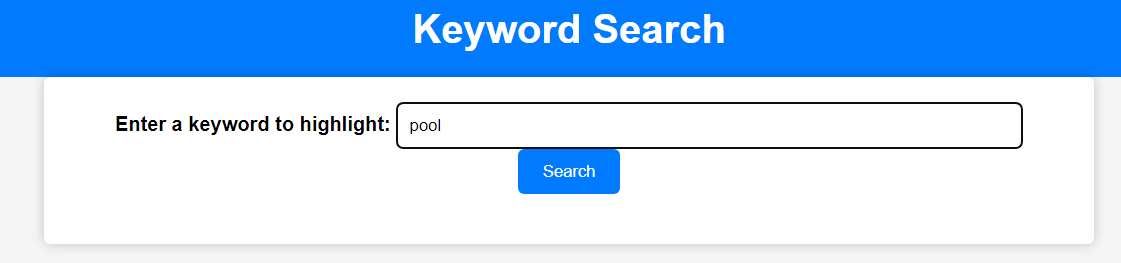
Description automatically generated

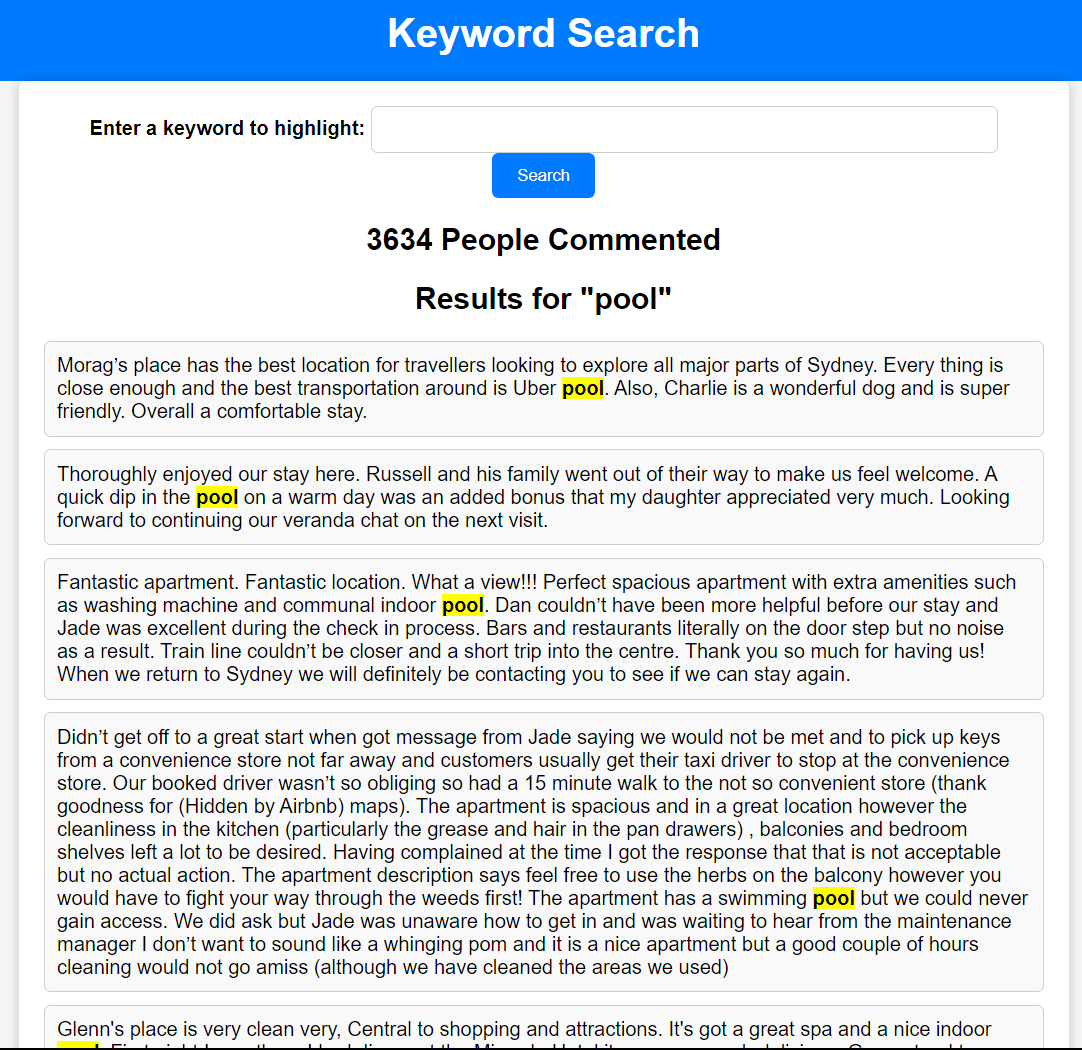
# Function 3: **Keyword Search in Reviews**

Function 3, "Keyword Search in Reviews," empowers users to search for specific keywords within Airbnb guest reviews. Here's a step-by-step explanation of how this function works:

1. Input Keyword: Users are presented with an input field where they can enter a keyword of their choice. This keyword represents the specific aspect they are interested in, such as "pool," "view," "clean," or any other term relevant to their Airbnb search.
2. Searching Process: Upon entering the keyword and clicking the "Search" button, the function begins its operation. It accesses and processes guest review data, typically stored in a CSV file named "reviews\_dec18.csv."
3. Date Filtering: The function takes into account the selected time period, which was defined as starting from November 10, 2018, and ending on December 22, 2018, earlier in the application. It filters the guest reviews to include only those posted within this specific date range.
4. Keyword Highlighting: The primary task of this function is to identify and highlight instances of the entered keyword within the guest reviews. When a review contains the keyword, it is displayed with HTML markup to make the keyword visually distinct. This highlighting helps users quickly spot the keyword within each review.
5. Total Comments: The function provides users with the total number of people who commented on Airbnb listings during the specified time frame. In the given example, the result shows that 3,634 people left comments within this period.
6. Keyword Search Results: Beneath the keyword input field and the total comments count, the function presents the search results. It displays snippets of guest reviews that mention the specified keyword. Each review snippet includes the highlighted keyword, making it easy for users to identify and understand how previous guests have described their experiences related to the keyword.

In the provided demonstration, the keyword "pool" was used to perform a search. As a result, the function returned excerpts from guest reviews that contain the word "pool." These snippets offer valuable insights into whether the Airbnb listings have a pool and how guests have perceived and commented on this particular feature.



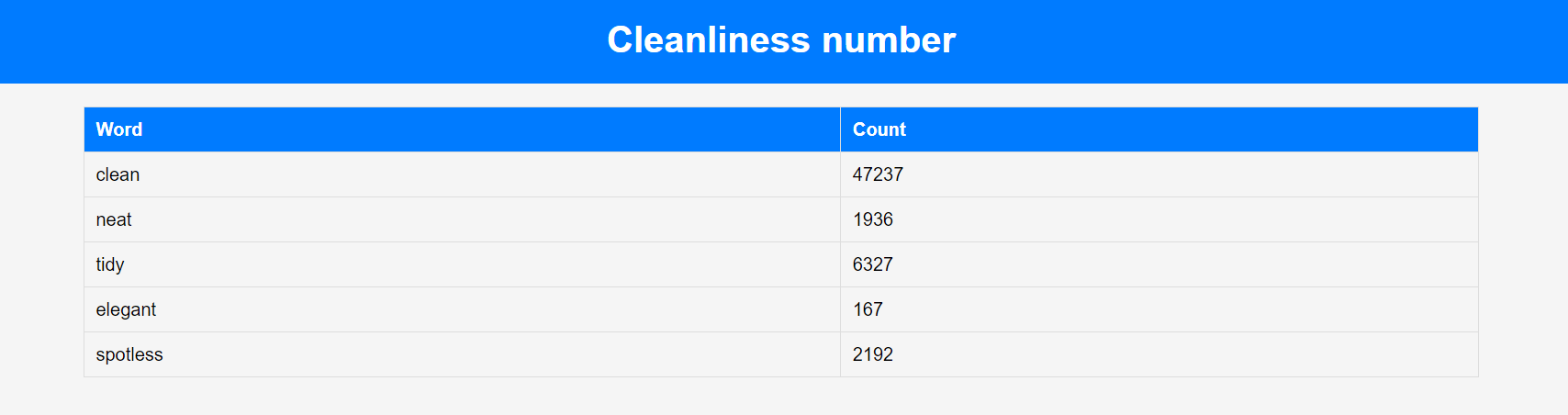


# Function 4: Cleanliness Keyword Analysis

Function 4, "Cleanliness Keyword Analysis," provides users with valuable insights into the cleanliness aspect of Airbnb accommodations based on guest reviews. Here's how Function 4 works:

1. **Data Analysis**: The function begins by accessing guest review data, likely sourced from a CSV file containing Airbnb guest reviews.
2. **Keyword Search**: It specifically focuses on five cleanliness-related keywords: "clean," "neat," "tidy," "elegant," and "spotless."
3. **Keyword Counting**: For each of these keywords, the function counts how many times they appear within the guest reviews. This counting process is conducted for reviews posted within the previously specified time period (November 10, 2018, to December 22, 2018).
4. **Results Presentation**: The function presents the results in a tabular format. It displays the cleanliness-related keywords and their respective counts. In this case, the results indicate the following:
   * "clean" appears 47,237 times in the reviews.
   * "neat" is mentioned 1,936 times.
   * "tidy" is found 6,327 times.
   * "elegant" is referenced 167 times.
   * "spotless" is used 2,192 times.

These counts offer valuable insights into how guests have described the cleanliness of Airbnb accommodations during the specified time frame. It provides potential guests with an understanding of the cleanliness standards maintained by hosts, helping them make informed booking decisions.



# 10.Function 5: Room Type Analysis

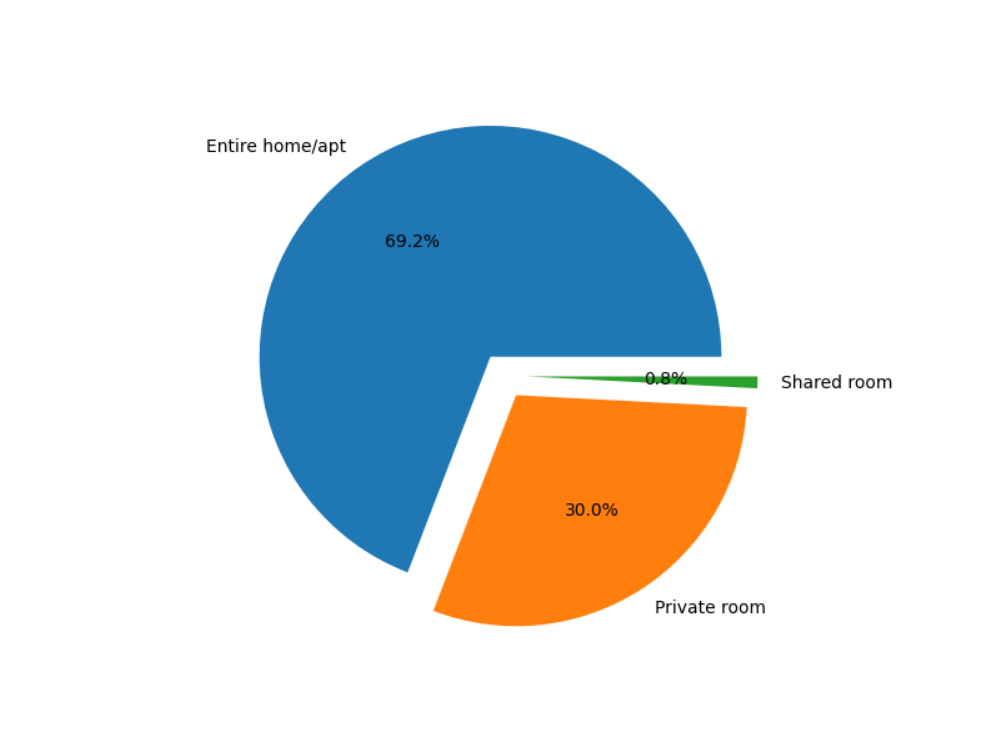
Function 5, "Room Type Analysis," provides insights into the distribution of Airbnb listing types (entire home/apartment, shared room, private room) within a specific neighborhood (suburb). Here's how Function 5 works:

1. **Data Analysis**: The function begins by accessing data from the Airbnb listings, likely sourced from a CSV file such as "listings\_summary\_dec18.csv."
2. **Filtering by Neighborhood**: It filters the data to include only listings within the selected neighborhood (suburb), as specified by the user.
3. **Room Type Distribution**: The function calculates the distribution of room types within the selected neighborhood. It determines the percentage of listings falling into each of the following categories:
   * Entire home/apt
   * Shared room
   * Private room
4. **Pie Chart Presentation**: To visualize the room type distribution, the function creates a pie chart. Each category is represented in the chart with a percentage and label. The chart provides a quick and clear overview of the types of accommodations available within the neighborhood.
5. **Result Presentation**: The pie chart is saved as an image file, which can be displayed to users when they access "Function 5" on the website.

In your provided example, the results indicate the following room type distribution within the selected neighborhood:

* Entire home/apt: 69.2%
* Shared room: 0.8%
* Private room: 30%

This information is valuable for users who want to understand the types of accommodations available in a specific neighborhood. It helps travelers choose the most suitable lodging option based on their preferences, whether they prefer an entire home, a private room, or a shared room.



# 11. No data available:

If no data is available for the selected time period, the application gracefully informs users by displaying a dedicated message. This message assures users that their query has been processed, but unfortunately, no relevant data was found within the specified date range. This user-friendly approach ensures transparency and prevents confusion, allowing users to refine their search or explore other available functions effectively.

