GRADUATE CERTIFICATE
INTELLIGENT REASONING SYSTEM (IRS)

---------- **PROJECT REPORT** ----------

Rental Recommendation Systems in Singapore

GROUP 3

Liu Yifeng	A0285847J
Pan Shuyi	A0286033H
Cai Xichen	A0285786E
Li Linze	A0285963L
Li Jiaxuan	A0285821Y

Table of Contents

1 Executive Summary

Amid the rapid urbanization and globalization of Singapore, the housing market has been at the forefront of these changes. With the influx of professionals, both local and international, seeking the dynamic opportunities Singapore offers, the rental market has witnessed substantial growth and transformation. Residents and newcomers are on the hunt for residential spaces that not only fit their budget but also align with their preferences and lifestyle.

In today's digital era, where convenience and precision are paramount, potential tenants face the daunting task of sifting through an overwhelming number of property listings across various platforms. The sheer volume of options often results in decision fatigue, extended search durations, and potential mismatches between renters and their ideal homes.

To address this inefficiency, our team has meticulously crafted a Rental Recommendation System. Envisioned as a comprehensive platform, our system alleviates the cumbersome process of property hunting. By understanding a tenant's unique needs and preferences, it offers tailored property suggestions that resonate with individual requirements.

Our system is anchored on comprehensive data mining from property listing platforms Airbnb and github. Central to our system is the recommendation reasoning mechanism which harnesses a specialized content-based filtering approach, enriched by certain algorithmic strategies, to offer pinpointed property recommendations. On the frontend, our team has adeptly employed Bootstrap as the foundational framework, enhanced by Jinja2 integration, ensuring both adaptability and aesthetic coherence. Features like interactive drop-down menus powered by JavaScript ajax further elevate the user experience. Seamlessly uniting this frontend with our backend is the Python Flask-wtf application, which not only streamlines data interactions but also ensures swift and accurate response to user queries.

Our project team hopes that with our solution, people in need will be able to find the house that suits their specific requirements most.

2 Problem Description

2.1. Problem Statement

Singapore is a densely populated country with a multicultural society of about 5.64 million people. The society is divided into three main ethnic groups: Chinese (74%), Malays (13%) and Indians (9%). It is also one of the most religiously diverse societies in the world, with ten different religions co-existing.

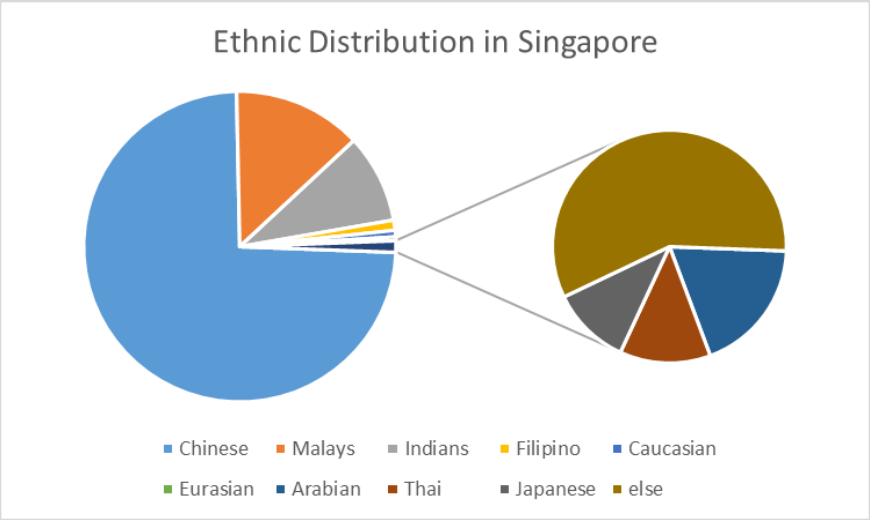


Figure 2.1. Ethnic distribution in Singapore

A large proportion of Singapore's population comes from immigrants. With the global economic downturn, saturated labour market and fierce competition in recent years, Singapore, as an important financial and business centre in Asia and the world, is now home to the regional headquarters of many global companies. According to the World Bank's Doing Business report, Singapore ranks second only to New Zealand among 190 economies. Coupled with the highly convenient business environment, many labourers choose to come to Singapore to look for job opportunities, and many entrepreneurs choose to expand their business in Singapore. This has resulted in Singapore's non-citizen population accounting for one-third of the country's total population in the 2022 Statistics Report, or 1.57 million non-citizens out of a total of 5.64 million people. This will increase to around 40% when permanent residents, students and other work permit holders are taken into account.

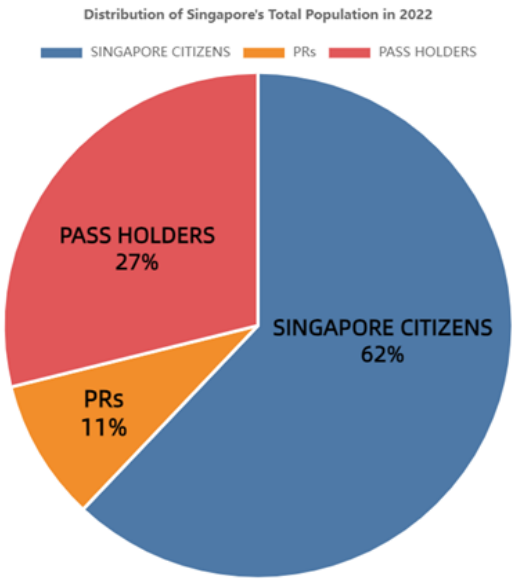


Figure 2.2. Distribution of Singapore's total population in 2022

While the demographic background exists, Singapore's rents fluctuate a lot with the regional distribution. City centre areas are usually more expensive, but are also closer to business districts and transport hubs. Suburban areas are relatively cheaper to rent while having a more tranquil living environment.

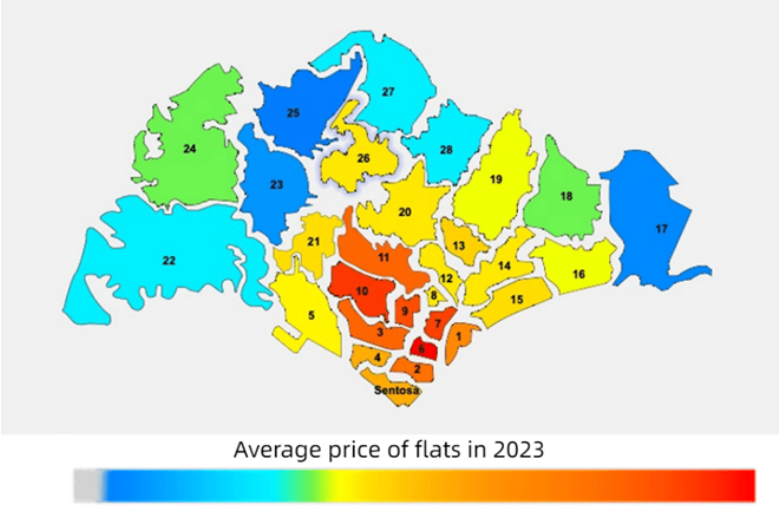


Figure 2.3. Average price of flats in 2023

Singapore's diverse cultural background has led to diverse housing needs, including the need for various housing types (e.g. flats, condominiums, townhouses and houses), various housing location amenities (e.g. proximity to universities, shopping malls, public transport), and various ways of renting (e.g. sharing a flat, renting on your own), among others.

The large number of immigrants, foreign workers and students has also greatly stimulated the rental market in Singapore, with the demand for housing far exceeding the supply and prices remaining consistently high throughout the year. At the same time, this group of people usually rent for longer periods and demand higher housing costs. Singapore is a high cost of living place and housing rents are relatively high, especially in the city centre areas. Therefore, renters usually need to carefully consider their budget and choose the right housing.

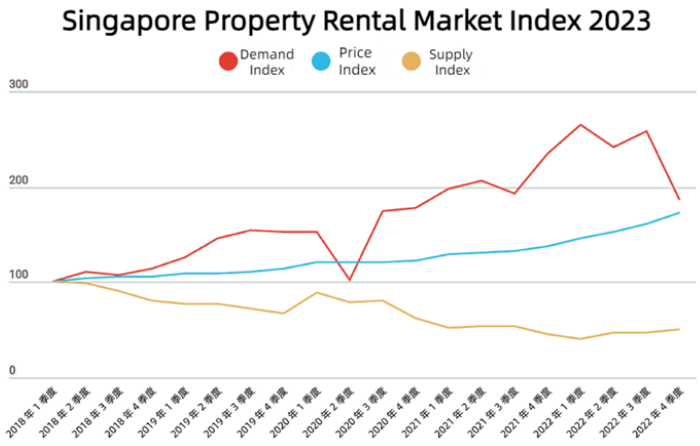


Figure 2.4. Singapore property rental market index 2023

2.2. Market research

Market Size: Singapore has a relatively large rental market, especially with the influx of international residents. Rental prices tend to be higher in central areas, but there is also significant demand in the suburbs.

Key Players: Key market players include real estate agencies, online rental platforms, property developers and individual landlords.

Competition: Singapore's online rental market is highly competitive, with platforms such as PropertyGuru, 99.co, Carousell and traditional real estate agencies all vying for market share.

Trends: The market is growing due to the influx of students and immigrants. Small and medium-sized families and single occupants make up a large proportion of the rental market. Meanwhile, current trends in the Singapore rental market include digital transformation, improved user experience, fluctuating rental prices and changes in rental policies. Sustainable and green housing are also emerging trends.

Market Trend: Singapore's rental market continues to grow steadily, driven mainly by demand from foreign professionals, students and immigrants. There is a market for both short-term and long-term rentals.

User demand: Renters typically seek housing that offers reasonable rents, a good location, convenient amenities and suits their lifestyles. Personalised rental options are becoming increasingly popular.

Potential Opportunities: Opportunities lie in improving the user experience, offering a wider range of housing options, using data analytics to better meet user needs, and providing specialised services to different segments of the rental market.

2.3. Project scope

We are going to design a rental recommendation system that uses machine learning and recommendation algorithms to provide accurate and personalised rental advice to users.

It will use content-based recommendation system, taking into account factors such as budget, location and other lifestyle preferences, to suggest matching properties for users.

Implement user feedback and rating system to make user-item based recommendation based on users' historical browsing and rating of the property.

3. Project Solution

3.1. Project Deliverables/ System architecture

Our group has developed the rental recommendation system architecture based on integrate several tools organically. The system consists of three foundation blocks: frontend, backend and knowledge base. For the frontend, we have built a clean, user-friendly light-weighted user interface for users to type in their needs and see their ideal listings in a straight and convenient way. To form the page more reasonable, we also add some self-define style using CSS. The backbone of our backend is developed using python and Flask framework. The Flask python file performs as a bridge among frontend, knowledge base and the recommendation algorithms. Specifically, the backend is responsible for managing the frontend page routing and interacting with the database and the recommendation algorithms to deal with the query requests.

The recommendation algorithms play the core role in the whole system. It utilizes content-based inference to perform the rental housing recommendation. Once a query request is triggered, the algorithms retrieve the information from both user input side and database side to perform the recommendation inference. The user input including the simple needs of every given field and users' thinking about how important each field is will be retrieved from the flask python file.

For the knowledge data base, we initially got the data from Airbnb website. However, we found that we cannot use the data straight, for some fields of data need to do knowledge discovering and data pre-processing, which will make the information more capable for the algorithms and the users. So, we use Jupyter notebook to do the knowledge discovering and data pre-processing tasks. In the database, we not only stored the information corresponding to the user input fields, but also the data related to the living convenience like the distance to the closest MRT and mall, and other information to enable the system to show more details of the listings.

Figure 3.1 shows an overview of our system architecture design.

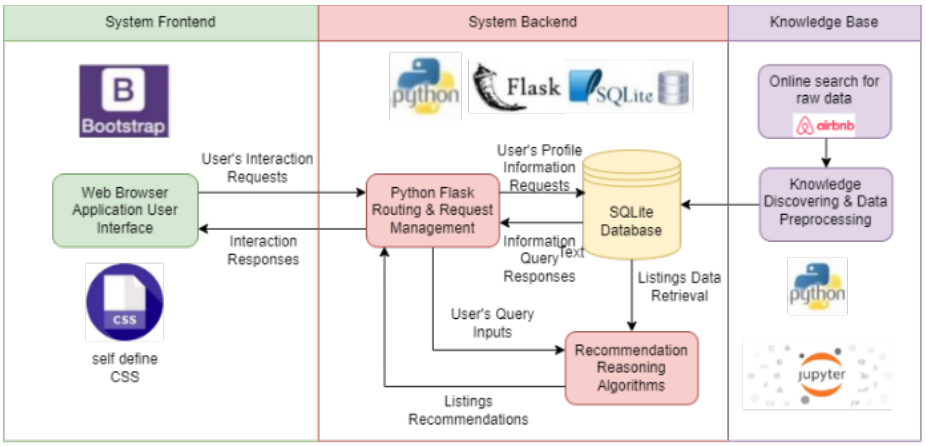


Figure 3.1. Overview of system architecture

3.2. Knowledge Representation/ Data

This system needs rental listings records to build a reliable recommendation system. The table below shows the list of web pages where data are to be extracted.

Website	Data
http://insideairbnb.com/get-the-data/	Detailed Listings data about rental
https://github.com/BlueSkyLT/siteselect_sg	The Land & Transport Singapore dataset

Data cleaning

To determine inaccurate and incomplete data and then improve the quality of data we got, data cleaning is required to be executed in the pre-processing stage. For useless columns, just drop them. For missing data, use the average value from the other houses to fill in the missing value. Data after cleaning are standardized and uniform to allow intelligence and data analytics tools to easily access.

Dataset merging

In the pursuit of refining our rental recommendation system, since Airbnb's original dataset does not have information about nearby MRT, shopping malls, etc., but contains latitude and longitude data, we went to the latitude and longitude information of these facilities and calculated the distance to the nearest MRT and shopping mall to each house as a new feature by utilizing the geopy library, a function was formulated to calculate the geographic distance of each residence from the closest target place, offering a holistic view for potential renters. This enrichment not only aids in refining rental recommendations but also facilitates users in comprehending the transportation and shopping conveniences around a property.

A preliminary audit of the dataset was conducted to ensure data integrity. This included addressing missing values, eliminating duplicate entries, and seamlessly merging various datasets.

We're confident in the robustness of our dataset. Encompassing a diverse range of rental information across Singapore, it stands as an invaluable asset for our rental recommendation system, combining both usability and practicality to serve the end user's needs.

4. System implementation

4.1. System Frontend

Our team chose Bootstrap as the main framework of frontend. Bootstrap not only has rich UI components and embedded a variety of nice CSS style, but also gives us high flexibility to freely define the themes and styles we need. It can meet our needs for the simplicity and the utility of the frontend user interface and the efficiency of the coding. To reduce the repetition of code snippets and allow frontend part and backend part to be coded in parallel, we also combined Jinja2 with Bootstrap. In addition, we use JavaScript ajax to equip our webpages with several on-page interactive functions such as drop-down menus and page redirection. The frontend is integrated with the backend request management module which is built based on the python Flask-wtf application.

4.2. User Interface

Our website application is divided into six pages, including 2 main functional pages and 4 basic pages [Fig x]. The first basic page is dedicated to introducing our personalized rental recommendation system. It serves as a warm welcome to our users, presented through concise phrases. Users are encouraged to initiate their experience by clicking the 'LOGIN AND GET STARTED!' button. Upon doing so, they will be prompted to enter their username and password, granting them access to our personalized services.

For individuals who are new to our platform, they can easily navigate to the registration page by selecting the 'Register Now' option. During the registration process, customers are required to confirm their password, ensuring data security. After successfully logging into the system, customers are directed to the welcome page, they can type in their personal needs of renting by clicking the “TAP IN HERE” button.

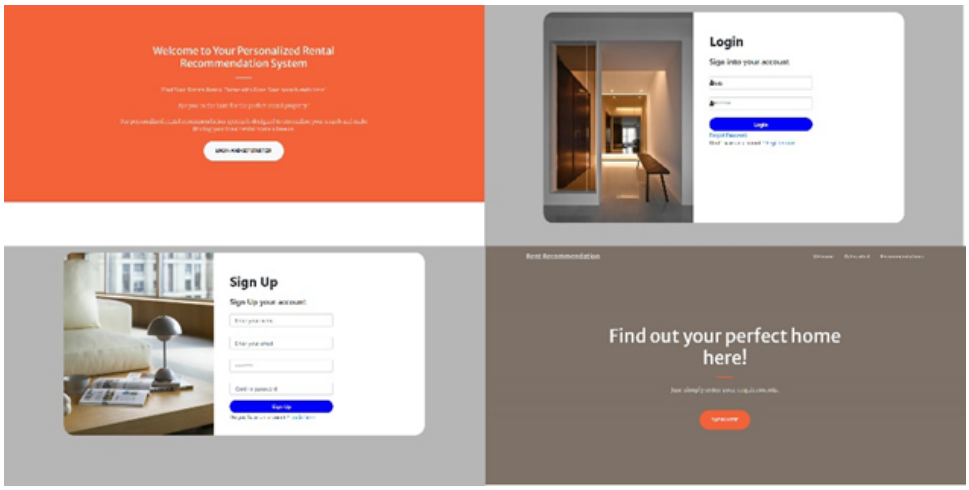


Figure 4.1.2 Rent recommendation website introduction, welcome, login and signup pages

The two main functional pages are user input page (Fig x) and recommendation page (Fig x). After being directed to the user input page, users are required to type in action to show their requirements. According to the market research, several factors in the context of renting are derived, specifically, the price, location, room type and the housing amenities. These factors are then combined with the check-in and check-out date as the input information. To establish a structured format for user input data and to maintain its integrity, the following restrictions on different input fields are placed:

1. All fields but amenities require input. It is a restriction that considers both reasonability and the needs of the backend-algorithms.
2. The rental length of the residence must be greater than 92 days (about 3 months). Based on the results of market research, it was found that there is no case where residence duration is less than 3 months.

With the aim of implementing a highly personalized rental recommendation system, the importance ratings for the required four inputs are set. These ratings indicate the importance of the factors when customers consider renting. For example, a customer may consider the location is the most important, so this customer may rate the importance of the location field as 5. The users are encouraged to give different rates for different fields, but we can also accept they give same rates for some fields, for the case when they may feel that one factor is just as important as another.

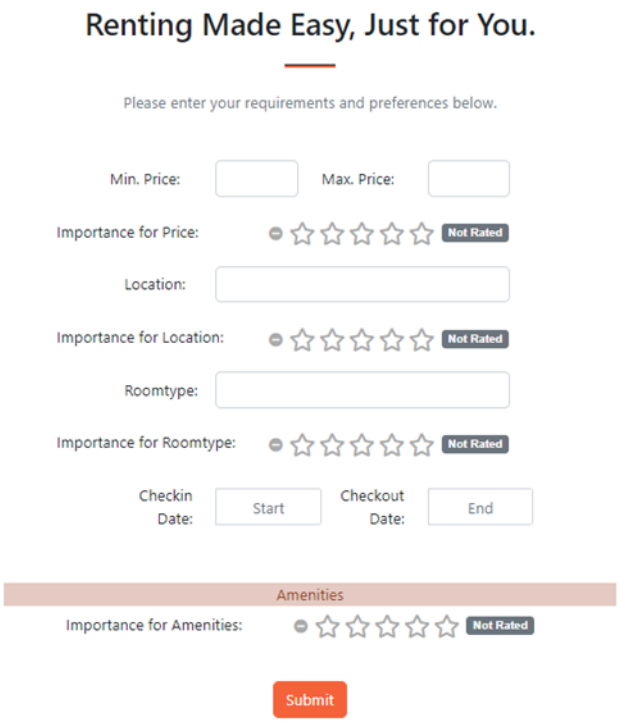


Figure 4.2.3 Rent recommendation websites user input page

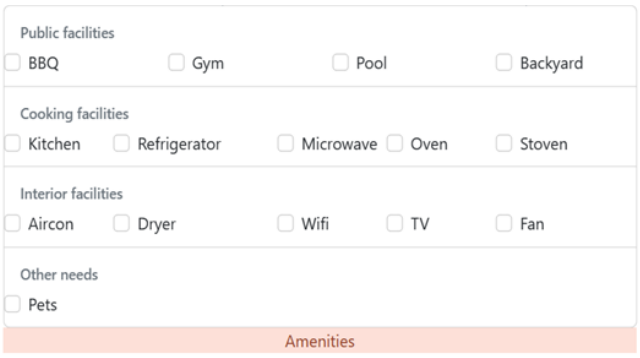


Figure 4.3.4 Rent recommendation websites user input page- Amenities

In response to a user clicking on 'amenities', the selected options will be displayed. After the users finish all the input they need to type, they can click on the submit button. Then the system will recommend top ten listings that match their needs the best.

The results will be shown on the recommendation page. Each card on that page represents a listing, and there is some detailed information of the listings, including specific property type and region, room type, and so on. Additionally, a link is provided to show the information about the location. The details of every kind of facility will be shown by clicking the corresponding field. This page enables the users to see more information about the recommended listings in addition to the information they request in the user input page.

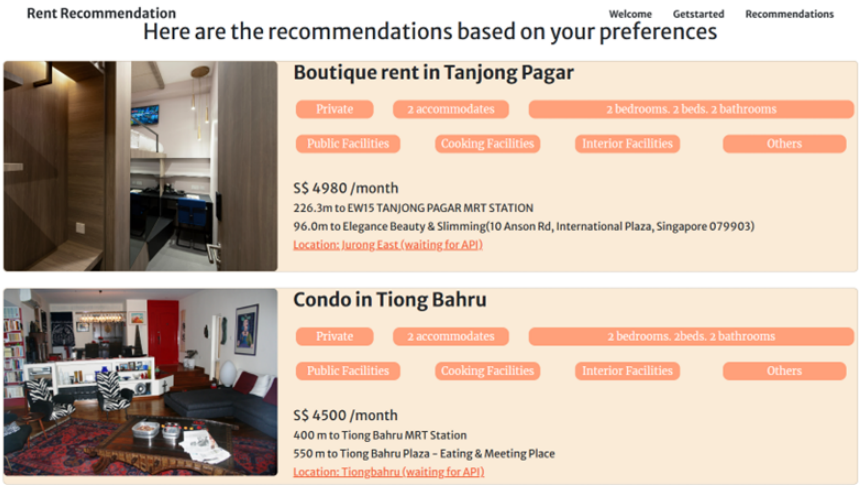


Figure 4.4.5 Rent recommendation websites recommend page

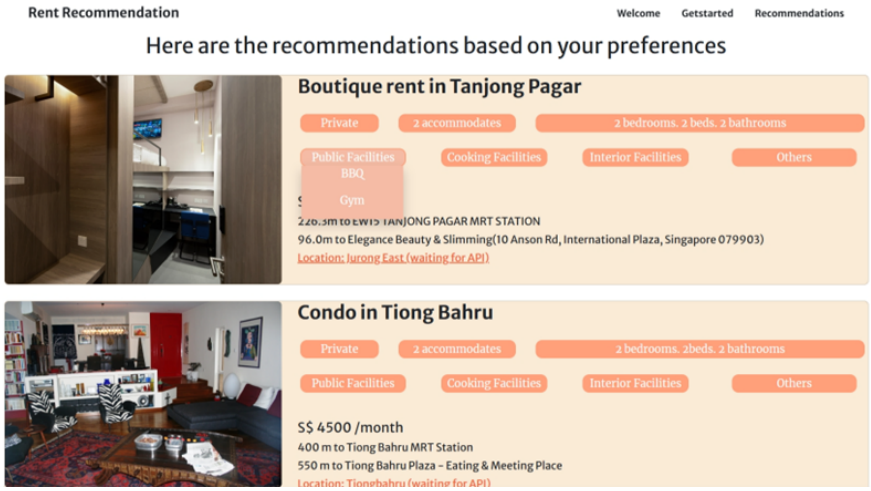


Figure 4.5.6 Rent recommendation websites recommend page-display of the facilities

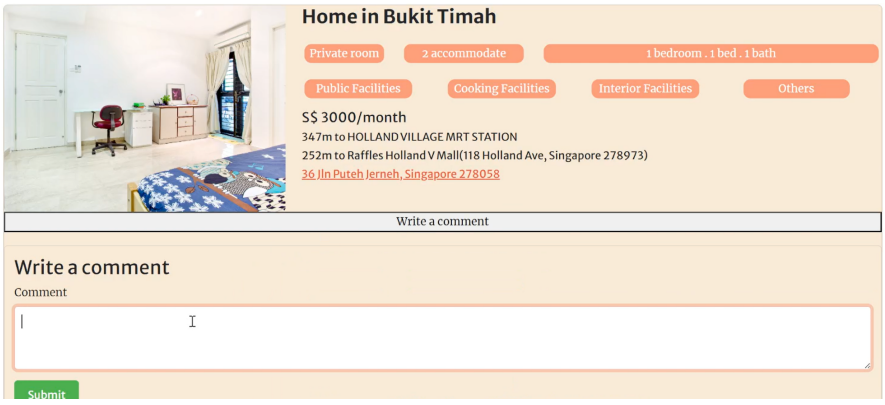


Figure 4.6. User Comment Input

4.3. System Backend

4.3.1. Backend System Architecture and Functional Modules

The backend of our system is built on Flask, enabling the power of various Flask extensions and custom-designed modules to establish a efficient environment [Fig x]. SQLAlchemy operates as the ORM for seamless database interactions. Secure password hashing and user session management are handled by Bcrypt and LoginManager, respectively. Flask-WTF is integral for form data processing, offering protection against CSRF attacks via CSRFProtect and easing the integration of web forms into Flask applications. Additional tools such as Flask Migrate assist in managing database schema changes over time, while Flask Minify plays a part in optimizing the delivery of static files.

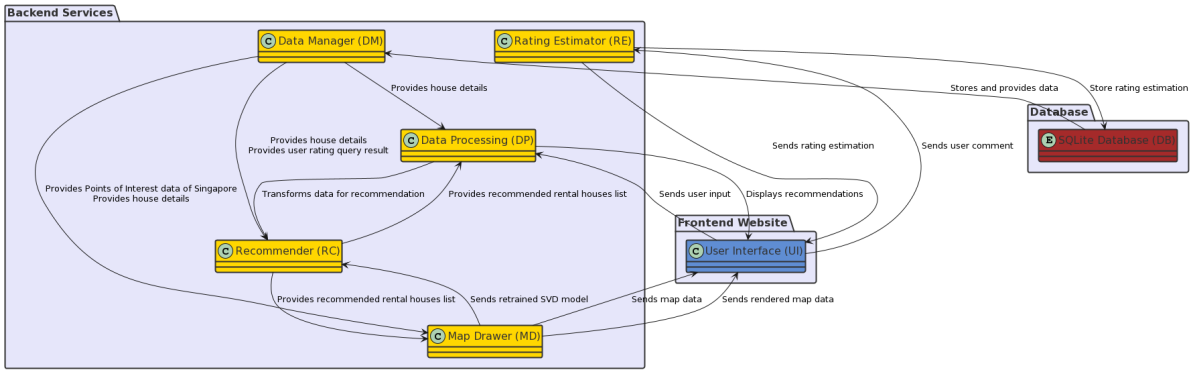


Figure 4.7. Backend Services and Data Flow

There are few custom modules designed in our backend, each fulfilling a specific role. DataManager is responsible for data retrieval and preprocessing, ensuring efficient extraction of necessary data from the database, preparing for recommendation or map visualization purposes. MapDrawer in charge of recommendations and user-specified desired locations on a map, enriching the user's spatial understanding of the available accommodations. The Recommender module is central to providing personalized recommendation, utilizing user input and historical data to implement content-based and hybrid recommendation algorithms, adapted to the user's comment history. RatingEstimator is tasked with deriving sentiment analysis-based rating predictions, analysing user reviews and estimating ratings.

The Recommender module stands at the heart of our system. It offers recommendations based on the user's input preferences and comment history. For users without a comment history, a content-based recommendation algorithm is employed, and calculating weighted cosine similarities. If a user has a history of comments, the module switches to a hybrid recommendation algorithm, blending content-based techniques with matrix factorization which using a pre-trained SVD model and updating the model upon the arrival of new comments.

4.3.2. Backend Database Tables and Relationships

The backend database for our recommendation system is structured to store and handle data queries to rental houses, users, ratings, and points of interest (POIs), facilitating efficient data management.

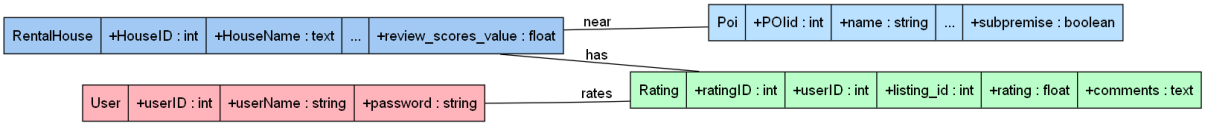


Figure 4.8. Database Schema for Backend Database

The database indexing on primary keys and vital fields ensures swift data retrieval, crucial for a responsive user experience. The inclusion of geographical coordinates in the tables enables map-based interactions and distance calculations, enriching the overall user experience.

4.3.3. User Authentication and Authorization

Our system utilizes the Flask-Login extension to efficiently manage user sessions and authenticate users. It ensures that a user's logged-in state is maintained across page reloads, storing the login state directly in the browser's cookies.

To protect certain view functions and resources, we employ Flask-Login's login_required() decorator, ensuring that only authenticated users can access protected areas of our platform. Unauthenticated users attempting to access these areas are swiftly redirected to the login page, enhancing the security of our system.

4.4. Recommendation Algorithm

Our system algorithm is designed to accommodate various user scenarios, including those without a history of interactions. In such cases, the algorithm first preprocesses the data through the user's inputs, transforming these into numerical vectors. Utilizing these vectors, it then calculates the similarity of each listing to the user's stated preferences, recommending the listings that best align with the user's needs. This approach ensures that even new or anonymous users receive tailored suggestions, making the platform accessible and immediately useful to everyone. By focusing on the explicit preferences provided during the current session, the system can generate relevant recommendations, fostering a positive user experience from the very first interaction.

For users with a history of estimated rating, we utilize a matrix factorization model, specifically Singular Value Decomposition (SVD), to predict and rank listings based on potential user interest. This model considers latent factors and patterns from all users' interactions, providing a more personalized and accurate set of recommendations. The results from both content-based similarity and matrix factorization are then synergistically combined in a hybrid recommendation approach. By adjusting the balance between these two methods through a weighting parameter, our system ensures that the final recommendations are not only like the user's explicit preferences but also resonate with their implicit interests, leading to a more comprehensive and satisfying user experience.

4.4.1. Feature Selection

Based on the database collected, we can see that not all data fields are necessary useful in the recommendation reasoning process. In addition, the rental data in the database are of variety of data types such as strings, integers and categorical values.

In order to develop the recommendation reasoning system, we have selected some data fields as features and are separated into two groups namely the text-based features and numerical features. A summary of selected data fields for the two groups are as follows:

Text features	Numerical Features
Reviews	Price
Location	Duration
Amenity	
Room type	

One-Hot Encoding Features

For the numerical features from both the course data and user inputs, we applied a simply one-hot encoding to convert them into binary matrix and vectors which prepare them for processing in the recommendation module.

For the `amenities` column, we tokenized the text to extract individual amenities. For each unique amenity identified, we created a new column. If a property had a particular amenity, we assigned a value of 1 to the corresponding column; otherwise, it was assigned a value of 0.

This transformation allowed us to represent the presence or absence of each amenity for every property in a binary format, which is easier for algorithms to process and analyse.

Standard Scaling

Before computing, it's essential to ensure that all our features are on the same scale. Given the diverse range of other numerical features like price, distance, and housing type, it becomes crucial to standardize these features.

For each of these features, we calculate the mean and standard deviation of the entire dataset. Using these values, we subtract the mean and then divide by the standard deviation for each data point in these features. The result ensures that our features, including the extracted amenities, have a mean of `0` and a standard deviation of `1`.

Now we ensure that no particular feature disproportionately affects our similarity calculations. It allows our recommendation algorithm to treat each feature with equal importance and offers more accurate and relevant housing suggestions to the users.

4.4.2. Recommendation Module

4.4.2.1. Content-based recommendation algorithm

When dealing with a new registered user, the architecture of the recommendation module is developed based on the content-based filtering recommendation approach to solve the cold-start problem.

User-Defined Weights

Since users are given the option to prioritize certain housing features over others. They can assign a weight or importance value to features such as location, price, amenities, etc., based on their current preferences. To ensure that the user-provided weights are comparable and don't disproportionately skew results, we normalize these weights so that they sum up to 1 or are within a consistent scale.

After preprocessing our features, we multiply each feature by its respective user-defined weight. This adjusts the value of each feature according to the user's importance ranking. Features with higher user-defined weights will have a more significant influence on the similarity scores.

By allowing users to define weights, our housing recommendation system offers a truly personalized experience. Users have the flexibility to adjust their criteria and see listings that best match their current priorities.

Similarity Score

The first process in the recommendation module is the computation of similarity score using cosine similarity function. The cosine similarity of two vectors of length n is defined as follows:

$$\text{Sim}_{\text{cosine}}(A, B) = \frac{A \cdot B}{|A||B|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$

The similarity score of the user input one-hot encoded vector against each of the course feature vector in the one-hot encoded matrix is computed. We choose cosine similarity mainly because：

Cosine similarity is a commonly used metric for comparing the similarity of two vectors in a high-dimensional space. The cosine similarity score ranges from 0 to 1, making it easy to interpret.

It is particularly effective when considering the direction of the vectors rather than their size. This is particularly important for our application because we are concerned with the degree of match between the user's preferences and the properties of the listings, rather than their absolute sizes.

Of course, let's delve into the penalty terms associated with distance and price in our housing recommendation project:

Penalty Terms

Penalty is introduced to ensure that houses too far away from the user's desired location are penalized in the similarity scoring and prioritize houses that are within the user's budget and penalize those that deviate significantly.

These penalty terms serve as crucial adjustment factors in our recommendation system. They guarantee that even if a house matches well in terms of amenities or other features, it won't be highly recommended if it's too far away or misaligned with the user's budget. This approach ensures a balanced and realistic set of top housing recommendations for our users.

The detailed content-based recommendation algorithm steps include:

1. **Pre-processin**g the data: converting features such as price, room type and amenities into a numerical format for subsequent calculations.
2. **Calculate the distance** of each listing from the user based on the user's location.
3. **One-Hot coding** of room types and extraction of relevant columns based on the user's selection.
4. **Extract amenities** in the listing with amenities of 1, which indicates that the listing has that amenity.
5. **Form feature vectors** for users and listings.
6. **Weight the features** according to the predefined weights.
7. Perform **feature scaling** on the weighted vectors to ensure that all features have the same importance in calculating similarity.
8. Calculate the **similarity** between the user vector and each listing vector using cosine similarity.
9. Introduce a **penalty term** to reduce the similarity score of listings that do not satisfy the user's preferences.
10. **Sort listing**s based on similarity score.

4.4.2.2. Matrix Factorization recommendation algorithm

Matrix factorization is a type of collaborative filtering algorithm that discovers latent factors underlying the interactions between users and items. When making recommendations to a user with comment history stored in our database, a recommendations using matrix factorization method is calculated for further combinations in our hybrid recommendation algorithm.

SVD Matrix Decomposition

The matrix factorization algorithm decomposes the user-item interaction matrix into two lower-dimensional matrices, representing latent user preferences and item attributes, respectively. User matrix is a matrix representing the latent factors of users. Each row corresponds to a user, and the columns represent different latent factors. Item matrix representing the latent factors of items. Each row corresponds to an item, and the columns represent different latent factors. In the case of our pretrained SVD model:

$$R = U \Sigma V^T$$

R is our user-item interaction matrix. Each row represents a user, and each column represents an item. The values are estimated rating score from the Airbnb review dataset. U represents users' preferences over latent factors. Σ contains singular values that signify the importance of corresponding latent factors, and V represents how items relate to the latent factors.

When we save the SVD model in Surprise package, we are essentially saving these matrices along with any additional configuration and hyperparameters used during training. These saved matrices can then be used to make predictions and generate recommendations. And the same hyperparameters are used to updating the model.

The detailed Matrix Factorization recommendation algorithm steps include:

1. Iterates through all the unique item IDs.
2. Estimates the rating for each item for the given user using the SVD model.
3. Ranks the items based on these estimated ratings.
4. Selects the top N items to recommend to the user.
5. When a new estimated rating is recorded, triggers a model retraining process.

4.4.2.3. Hybrid recommendation algorithm

The hybrid recommendation system comes into play for users who have a history of comments and estimated rating stored in our database. The recommendation result from Matrix Factorization algorithm are seamlessly integrated with content-based recommendations to generate a comprehensive and personalized set of recommendations.

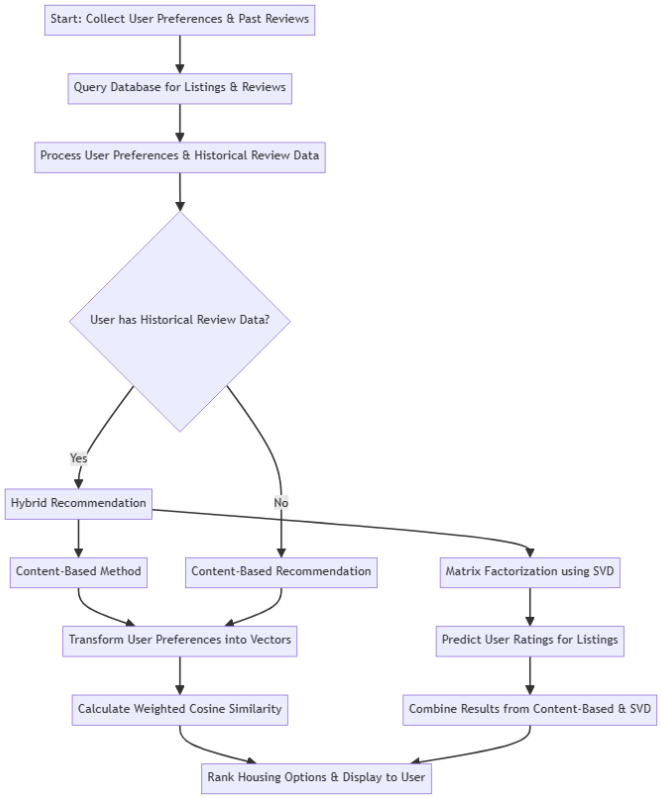


Figure 4.9. Hybrid recommendation algorithm

The detailed Hybrid recommendation algorithm steps include:

1. **Normalization** of Matrix Factorization Scores: Normalize the matrix factorization scores using Min-Max Scaler to align them onto a uniform scale ranging from -1 to 1, ensuring comparability with the content similarity scores.
2. **Weighted Combination** of Scores: Merge the normalized matrix factorization scores with the content similarity scores, calculating a weighted sum of both for each item. The parameter alpha determines the balance, facilitating a blend of personalized and content-based recommendations.
3. **Ranking** and Final Recommendation: Sort the items based on the combined scores in descending order, and select the top items as the final set of recommendations, achieving a balanced and tailored recommendation output.

4.4.3. Output

Finally, with the ranked and sorted rental info based on the user's intent text inputs and preference, we take the top 10 houses as the output of the recommendation module. The listings are then ranked based on their similarity scores. Houses that align more closely with the user's emphasized features will rank higher and will be among the top recommendations.

Overall, our system is versatile, catering to both new users without interaction history and existing users with interaction data, ensuring personalized and accurate recommendations in various scenarios. With the output, they are then sent to the frontend module for presenting the result to user.

4.5. User Rating Estimation

The Rating Estimation algorithm transforms user comments into estimated ratings through an intricate process. Upon reception of a user comment, it utilizes three sentiment analysis tools: Flair, Asent, and TextBlob. Flair performs a deep learning-based text analysis to predict a positivity score and a sentiment value. In parallel, Asent provides a rule-based sentiment analysis result, while TextBlob contributes additional insights with polarity and subjectivity scores.

The collected sentiment data is then synthesized, forming a comprehensive sentiment profile of the user review. By observing the predict result of different approaches, we found the Flair's prediction had highest accuracy. Therefore, a base estimated rating score is derived through an exponential mapping based on Flair's sentiment analysis result.

$$\text{exponential\_mapping}(\text{score}, \text{sentiment\_value}) = \begin{cases} 5 \cdot \exp\left(-2 \cdot \left|\frac{\text{score}-0.5}{0.5} - 0.99\right|\right), & \text{if sentiment\_value} = \text{'POSITIVE'} \\ 5 \cdot \exp\left(-2 \cdot \left|\frac{\text{score}-0.5}{0.5} - 0.01\right|\right), & \text{otherwise} \end{cases}$$

In scenarios where Flair's positivity score surpasses the 0.9 threshold, showcasing high confidence, the corresponding Flair-based rating is deemed reliable and directly adopted as the final estimated rating. Conversely, in situations marked by lower confidence or discrepancy across sentiment analysis results, the algorithm resorts to a pre-trained Random Forest Classifier. This classifier, trained with pseudo labels generated from ChatGPT's sentiment analysis on a curated Airbnb reviews, plays a crucial role in ensuring the precision and trustworthiness of the final rating estimation.

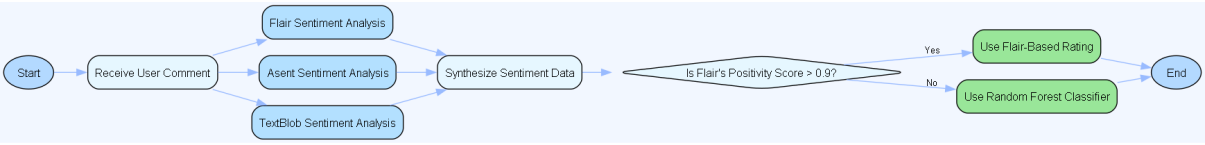


Figure 4.10. Rating Estimation algorithm

5. Challenge & Future Improvement

5.1. Challenge

Dynamic Housing Market Conditions

As mentioned in section 3.2 regarding Data Acquisition, our housing data is extracted from database provided by Airbnb and is up-to-date as of June 2023. While this method of data acquisition is relatively reliable and convenient, it faces the following challenges:

- The real estate market is ever-changing. Given that our dataset was obtained at a specific point in time and isn't updated in real-time, this could lead to a lack of timeliness in the data.
- Property information on Airbnb might undergo frequent updates, such as price changes, new property listings, or changes in the status of already listed properties. Not having real-time updates could affect the user's ability to access the most accurate property information.

User Preference Complexity

Unlike more static choices, selecting a house is often a multifaceted decision influenced by numerous factors:

- **Emotional considerations**: The feelings a property evokes can be just as crucial as its objective features.
- **Future projections**: Users may think about their future needs or the potential appreciation of the property's value.

This complexity can make it challenging for a recommendation system to consistently hit the mark, especially if relying solely on past behaviour or explicit inputs.

5.2. Future Improvement

Enhanced Data Collection and Verification

To address the challenges posed by the dynamic nature of the housing market and inconsistent data:

We could look into more direct integrations with real estate platforms or databases, allowing for real-time or near-real-time data updates.

Deploying mechanisms to verify the authenticity and accuracy of listings, such as user feedback loops or collaborating with property agents, can improve the quality of our dataset.

Adaptive Recommendation Algorithms

Given the unique nature of house hunting:

Instead of solely relying on traditional machine learning models, we might explore hybrid systems that combine collaborative and content-based filtering.

Incorporating user feedback actively and adjusting recommendations in real-time can provide a more responsive and personalized user experience.

Advanced Algorithmic Approaches

The evolving nature of machine learning and data analytics provides ample opportunities for refining our recommendation system. Future directions for improvement include:

- **Deep Learning Integration**: Transitioning from traditional machine learning models to neural networks can enhance recommendation accuracy. For instance, employing convolutional neural networks (CNNs) to analyse property images may yield insights into user preferences based on visual appeal.
- **Collaborative Filtering**: While our current model is content-based, integrating collaborative filtering can tap into collective user behaviours. By analysing interactions of similar users, we can offer suggestions that may not be immediately obvious from the property attributes alone.
- **Reinforcement Learning**: Implementing a reward system wherein the algorithm learns over time based on user interactions. Such a system can adapt and improve recommendations based on direct user feedback after viewing or booking a property.

APPENDIX A – User Guide

Installation Guide

First, download and Clone from Github Repository, then enter the folder XX.

In order to use the rental recommendation system, users need to first create a new Python environment. Our recommendation system runs within this new environment. If using anaconda, users can create an activation environment using the following command.

For Windows users

```
conda create --name pjrent python=3.11

conda activate pjrent
```

For Linux users

```
conda create --name pjrent python=3.11

conda activate pjrent
```

After successfully creating and activating the environment, install the Required Python Packages, running the following command.

For Windows users

```
pip install -r requirements.txt
```

For Linux users

```
pip install -r requirements.txt
```

After ensuring that the installation has been completed correctly, run the following code.

For Windows users

```
flask run
```

For Linux users

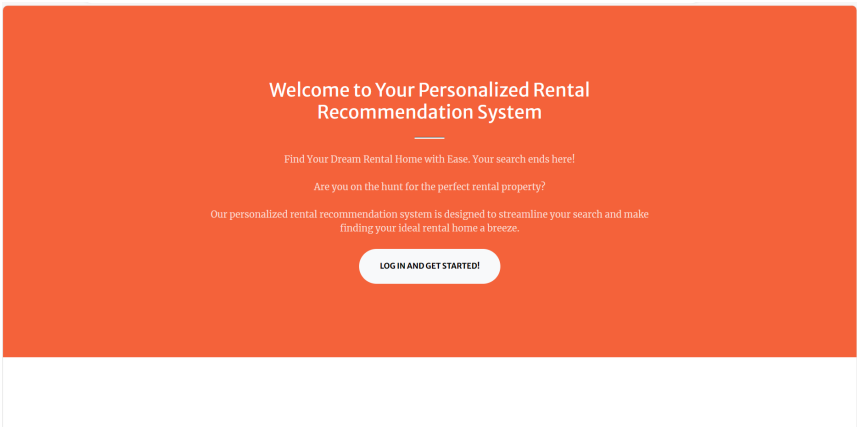
```
flask run
```

By default, Flask will start the application on port 5000 on localhost.

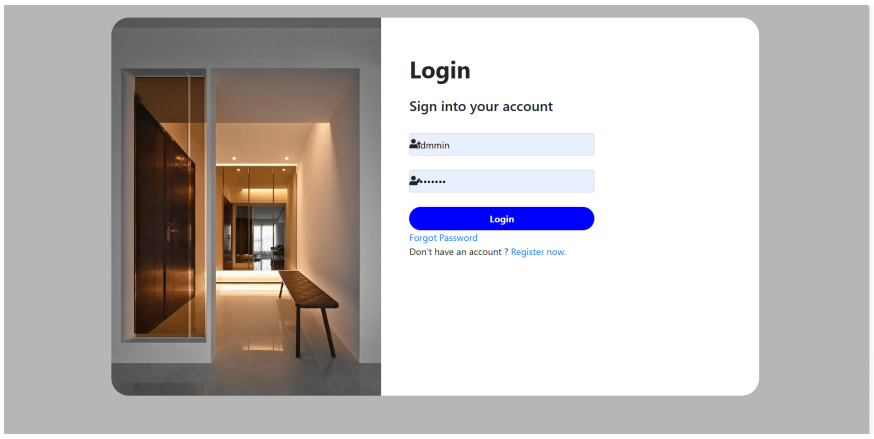
In the terminal, users will see that the application has started and the address and port it is listening on. Typically, users can view the Flask application in a browser by visiting http://localhost:5000.

User Manual

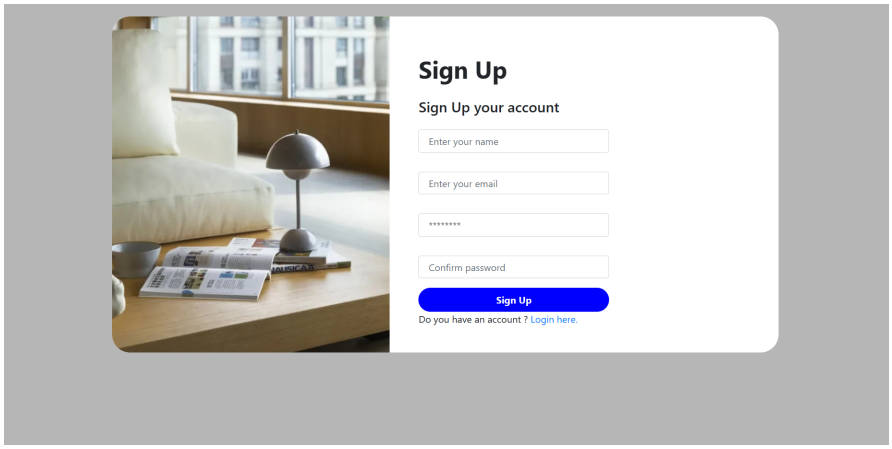
Below I will describe the specific process of using our rental recommendation.



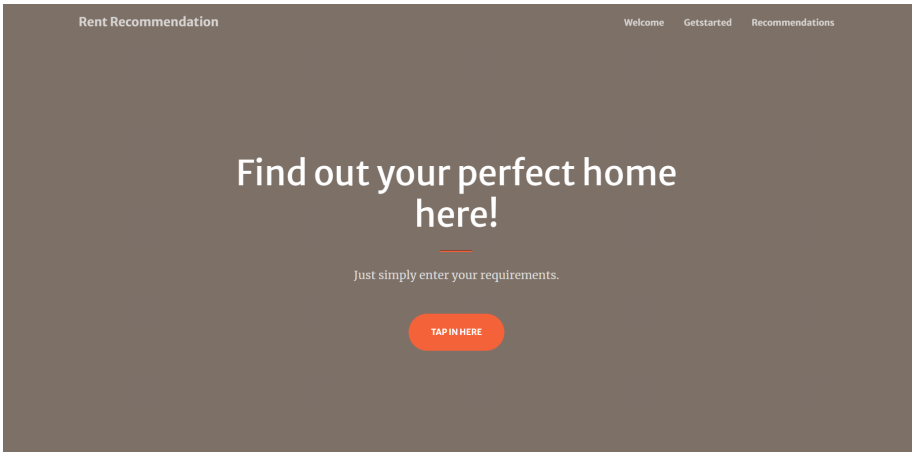
After all the above steps are completed successfully, the user will see the home page of our rental recommendation application. Click on “Log in and get started” button, users will be navigated to the log-in page.



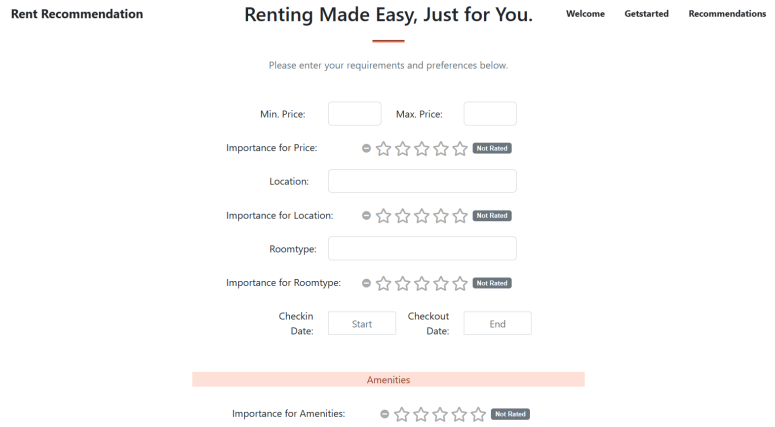
If new to this system, simply click on “Register now”. Then users can see the sign-up page. Enter the required information. After signing up, click the “login here” button.



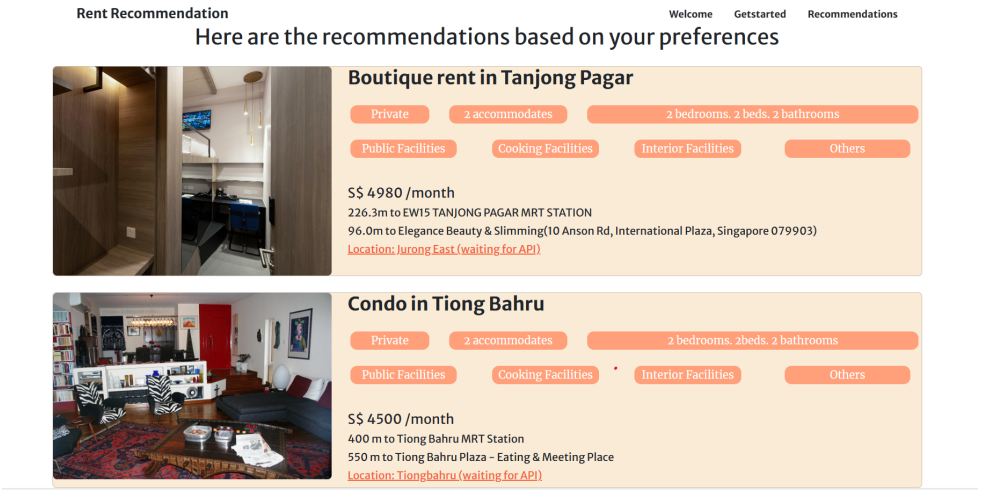
After completing the login, users will land on our welcome page. All users need to do here is to click the “tap in here” and wait to be navigated to the next page, where users are required to fill in there requirements on their rent.



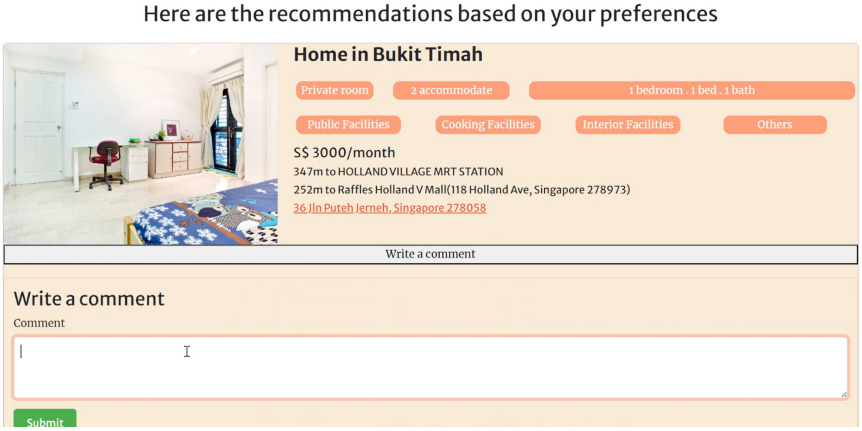
Please fill in the blanks one by one. All sections need to be populated, except for "amenities", which represents the user's amenity requirements for the rent, and which is not a must.



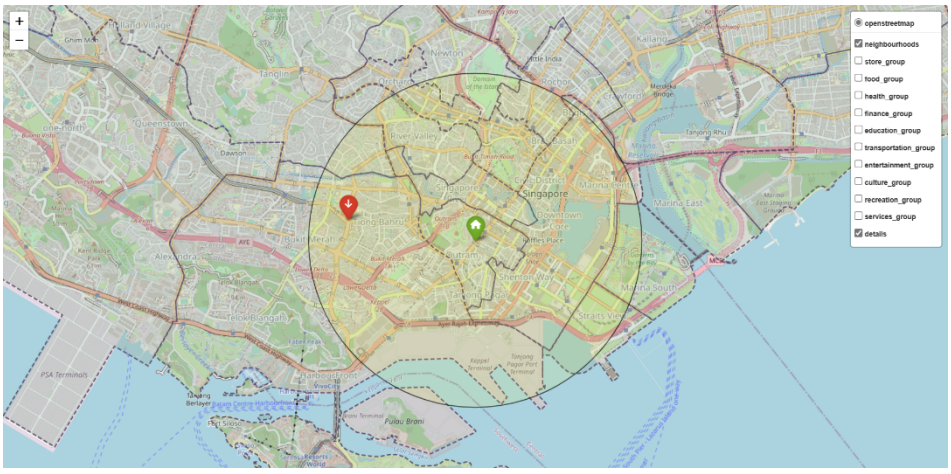
After the submit button is clicked, the recommendations page will be displayed, which includes ten items that are the best matches for the user inputs.



Hover the mouse over the button in the second row and a drop down box will appear to show the amenities that the room has. Click the “Write a comment” button, you will be able to describe your satisfaction of the recommended rental house. The page will refresh automatically and generate a new recommendation for you based on your feedback.



If you click on the "location" button, it will return a map with the location of the rent, which is convenient for the users to know the location information.



APPENDIX B – Proposal

Date of proposal:

1st Oct 2023

Project Title:

ISS Project – Rental recommendation system in Singapore

Sponsor/Client: *(Name, Address, Telephone No. and Contact Name)*

Background/Aims/Objectives:

Singapore is a densely populated country with a multicultural society of about 5.64 million people. The society is divided into three main ethnic groups: Chinese (74%), Malays (13%) and Indians (9%). It is also one of the most religiously diverse societies in the world, with ten different religions co-existing.

A large proportion of Singapore's population comes from immigrants. With the global economic downturn, saturated labour market and fierce competition in recent years, Singapore, as an important financial and business centre in Asia and the world, is now home to the regional headquarters of many global companies. According to the World Bank's Doing Business report, Singapore ranks second only to New Zealand among 190 economies. Coupled with the highly convenient business environment, many labourers choose to come to Singapore to look for job opportunities, and many entrepreneurs choose to expand their business in Singapore. This has resulted in Singapore's non-citizen population accounting for one-third of the country's total population in the 2022 Statistics Report, or 1.57 million non-citizens out of a total of 5.64 million people. This will increase to around 40% when permanent residents, students and other work permit holders are taken into account.

While the demographic background exists, Singapore's rents fluctuate a lot with the regional distribution. City centre areas are usually more expensive, but are also closer to business districts and transport hubs. Suburban areas are relatively cheaper to rent while having a more tranquil living environment.

Singapore's diverse cultural background has led to diverse housing needs, including the need for various housing types (e.g. flats, condominiums, townhouses and houses), various housing location amenities (e.g. proximity to universities, shopping malls, public transport), and various ways of renting (e.g. sharing a flat, renting on your own), among others.

The large number of immigrants, foreign workers and students has also greatly stimulated the rental market in Singapore, with the demand for housing far exceeding the supply and prices remaining consistently high throughout the year. At the same time, this group of people usually rent for longer periods and demand higher housing costs. Singapore is a high cost of living place and housing rents are relatively high, especially in the city centre areas. Therefore, renters usually need to carefully consider their budget and choose the right housing.

Singapore's rental market continues to grow steadily, driven mainly by demand from foreign professionals, students and immigrants. There is a market for both short-term and long-term rentals. From the tenants' perspective, in general, renters typically seek housing that offers reasonable rents, a good location, convenient amenities and suits their lifestyles. Personalised rental options are becoming increasingly popular. But users' demands vary from each other, so it is the meaningful to develop a rental recommendation system.

We are going to design a rental recommendation system that uses machine learning and recommendation algorithms to provide accurate and personalised rental advice to users. It will use content-based recommendation system, taking into account factors such as budget, location and other lifestyle preferences, to suggest matching properties for users. Implement user feedback and rating system to make user-item based recommendation based on users' historical browsing and rating of the property.

Requirements Overview:

- Data Processing skills
- Database skills
- Knowledge on recommendation systems
- Research ability
- Programming ability
- System integration ability

Resource Requirements (please list Hardware, Software and any other resources)

Hardware/System proposed for consideration:

- Laptop or PC with sufficient RAM, disk space and computing power to install and run the below software

Software proposed for consideration:

- Flask (Pythonic Web Framework)
- Bootstrap (Frontend)
- Python (Backend)
- SQLite (Database)
- Learning Models

Number of Learner Interns required: (Please specify their tasks if possible)

a team of four to six project members (or individual work upon lecturer approval)

1. Liu Yifeng A0285847J
2. Pan Shuyi A0286033H
3. Cai Xichen A0285786E
4. Li Linze A0285963L
5. Li Jiaxuan A0285821Y

Methods and Standards:

Procedures	Objective	Key Activities

		1. Gather & Analyze Requirements
Requirement Gathering and Analysis	The team should meet with ISS to scope the details of project and ensure the achievement of business objectives.	2. Define internal and External Design 3. Prioritize & Consolidate Requirements 4. Establish Functional Baseline
Technical Construction	· To develop the source code in accordance with the design. · To perform unit testing to ensure the quality before the components are integrated as a whole project	1. Setup Development Environment 2. Understand the System Context, Design 3. Perform Coding 4. Conduct Unit Testing
Integration Testing and acceptance testing	To ensure interface compatibility and confirm that the integrated system hardware and system software meets requirements and is ready for acceptance testing.	1. Prepare System Test Specifications 2. Prepare for Test Execution 3. Conduct System Integration Testing 4. Evaluate Testing 5. Establish Product Baseline
Acceptance Testing	To obtain ISS user acceptance that the system meets the requirements.	1. Plan for Acceptance Testing 2. Conduct Training for Acceptance Testing 3. Prepare for Acceptance Test Execution 4. ISS Evaluate Testing 5. Obtain Customer Acceptance Sign-off
Delivery	To deploy the system into production (ISS standalone server) environment.	1. Software must be packed by following ISS's standard 2. Deployment guideline must be provided in ISS production (ISS standalone server) format 3. Production (ISS standalone server) support and troubleshooting process must be defined.

Team Formation & Registration

Team Name: Group3
Project Title (repeated): Rental Recommendation in Singapore
System Name (if decided):
Team Member 1 Name: Li Linze
Team Member 1 Matriculation Number: A0285963L
Team Member 1 Contact (Mobile/Email): e1221775@u.nus.edu
Team Member 2 Name: Pan Shuyi
Team Member 2 Matriculation Number: A0286033H
Team Member 2 Contact (Mobile/Email): panshuyi08@u.nus.edu
Team Member 3 Name: Cai Xichen
Team Member 3 Matriculation Number: A0285786E
Team Member 3 Contact (Mobile/Email): e1221598@u.nus.edu
Team Member 4 Name: Liu Yifeng
Team Member 4 Matriculation Number: A0285847J
Team Member 4 Contact (Mobile/Email): e1221659@u.nus.edu
Team Member 5 Name: Li Jiaxuan
Team Member 5 Matriculation Number: A0285821Y
Team Member 5 Contact (Mobile/Email): e1221633@u.nus.edu

APPENDIX C – Mapping of System Functionalities to Courses

Courses	Knowledge and Techniques
Machine Reasoning (MR)	**Knowledge Elicitation and extraction**: Extract property data from multiple datasets, including but not limited to price, type of property, amenities, and location. Collect specific preferences from the user regarding price, location, and amenities based on their input. **Knowledge Representation**: Preprocess the data, such as converting price, property type, and amenities into numeric formats or One-Hot encoding, to form feature vectors. Construct a feature vector representing the user's expectations and preferences based on their input. **Rule Based System**: Establish a set of rules based on the user's location to calculate the distance between properties and the user, filtering suitable options. Determine filtering criteria based on the price range and other preferences provided by the user.
Reasoning Systems (RS)	**Search & Optimization**: Compute the match degree between user and property feature vectors using cosine similarity. Introduce a penalty mechanism to address properties that do not align with user preferences, such as those out of the desired price range or located too far away. Optimize and rank the property recommendation list based on similarity scores.
Cognitive Systems (CGS)	**Cognitive System**: Continuously optimize and adjust the recommendation algorithm based on past user feedback and behavior to align more closely with their actual needs. Return the top 10 most matching properties to the user, offering a more personalized recommendation experience.

APPENDIX D – Individual Member Report

Name:	Li Linze	**ID:**	A0285963L

Personal Contribution:

My personal contributions to this project can summarized as follows:

1. Conducted in-depth research on the Singapore rental market to provide strong market insights for the project.
2. Collaborated with team members to design and write a content-based reasoning rental recommendation system.
3. Ensured that the algorithms could be seamlessly integrated with the back-end code, enabling smooth access to information from the database and front-end user inputs.
4. Actively communicated and co-operated with team members, coordinating the progress and work of the project together.

Learning Journey and Outcome:

At the early stage of the project, I studied the Singapore rental market in depth and fully understood its characteristics such as huge potential, oversupply, high diversity of tenants' needs, and large fluctuations in geographical property prices. In response to a series of challenges in the market, I discussed with my team members and finalised our project concept and framework: a rental recommendation system based on users' needs.

During the implementation of the project, I worked with Shuyi and co-wrote a content-based reasoning algorithm for the rental recommendation system. This algorithm not only obtains feature information of rental demand and user's historical behaviour information from user input and database, but also outputs highly similar rental information through the content recommendation system.

Eventually, seamless integration with Yifeng's back-end code allows our algorithm to smoothly access user information in the database and front-end user input, thus successfully creating a comprehensive rental recommendation system.

Application of Knowledge:

Applying what I learnt in class to real-world projects is extremely exciting and challenging.

Through the rich and varied learning in the Reasoning System class, I learned that content-based recommender system is a very effective and commonly used recommendation method. It recommends items that are relevant to the user's interests based on the characteristics of the item and the user's historical behaviour. Also, as an international student coming to Singapore for the first time, I was deeply touched by the problem of finding a house for rent. The price of housing in Singapore fluctuates greatly depending on the location, and considering the convenience of schooling and living, I wanted to find a house that was close to the NUS, close to the business district, and affordable.

Therefore, we decided to apply our knowledge to a project that is highly relevant to our daily lives, and designed a content-based rental recommendation system that extracts feature information from user inputs and historical user behaviour data to efficiently output highly similar recommended houses.

I must admit that this was the first time I completed a system project without guidance, and we invested a lot of time and effort in data processing, algorithm construction, and integration with the back end, but the result was outstanding, which undoubtedly gave me great encouragement and confidence.

Name:	Pan Shuyi	**ID:**	A0286033H

Personal Contribution:

My personal contributions to this project can summarized as follows:

1. Contributed to the initial ideation and data collection
2. Accomplished data preprocessing, including data cleaning and merging
3. Established the whole recommendation algorithm and optimization
4. Participated in writing the project report (summary, data, recommendation algorithm and future development) and editing video

Learning Journey and Outcome:

Embarking on the development of our Rental Recommendation System was a transformative journey, unveiling the intricacies of data science and algorithmic design. I delved into the immense sea of data from Airbnb and was intrigued by its potential to be harnessed for smart recommendations. Just as I discovered various facets of AI in the Graduate Certificate, the nuances of data cleaning, preprocessing, and optimization became evident.

The Recommendation Algorithm design, particularly the content-based filtering approach, resonated with the concept of Reasoning Systems I learned earlier. Identifying user preferences and mapping them with property features required meticulous design and iterative optimization. The process underscored the significance of choosing the right recommendation technique tailored to specific scenarios.

It became evident how tailoring user preferences to property features demanded a meticulous algorithmic design and the significance of optimizing recommendation techniques specific to the data at hand.

Application of Knowledge:

The insights gleaned from this project will undoubtedly shape our future initiatives in the realm of real estate technology. An immediate application on our horizon is the deployment of intelligent chatbots to assist users with property-related inquiries, enhancing the overall user experience on our platform.

Moreover, there's a vision to refine our rental recommendations by analyzing user interactions, reviews, and search patterns. This approach, inspired by the concept of course recommendations based on academic results, can be mirrored in suggesting properties that align with a user's past preferences and behavior.

In essence, the synthesis of my learnings from this project and foundational knowledge has primed me to address challenges in the real estate tech sector innovatively and effectively. Furthermore, with the practical hands-on experience on collaborative programming, I am now more comfortable and competent in taking up or initiating projects which are bigger in scope, within my department.

Name:	Cai Xichen	**ID:**	A0285786E

Personal Contribution:

I contributed to the system design part at the beginning of the project by dividing it into three main parts. After that, I mainly focused on the development of the system front-end using Bootstrap to ensure our pages are displayed in a brief, neat and user-friendly way. Jiaxuan and I cooperate together with each other, after several times of discussion, we came up with ideas of managing user input page and recommendation page by combing the Bootstrap with our self-define CSS style. In order to make the whole process of programming as a parallel way, I contributed to the establishment of the demo of backend, including the routing of some pages, the redirect between pages and then ensure that the back end receives the data fetched from some of the front-end page.

Learning Journey and Outcome:

During the journey of deeply involved in the process of developing a Rental Recommendation System, I have learned the knowledge the basic constituent parts and the interactions between different parts. It is my first time for me to develop a user interface that enables users to interact with the whole system. To be honest, it is not easy for developing such a front-end, for there are so many start tags and end tags in grammar which need to be checked carefully. Sometimes even one incorrect indentation or the fact that the start tag is not aligned with the end tag can disable the intended functionality. The two who responsible for the front-end developing, Jiaxuan and I, usually get together for research and discussion in order to realize a certain display style or layout. It was truly a time-consuming work, but when we finally achieve these, I always felt a big sense of achievement.

In addition to the front-end development, I also participate brainstorm of the algorithms. I acquired a lot about the machine learning of recommendation, witnessed the changes of the algorithms, from the basic similarity calculation between one user input field and data of an attribute, to the calculation between weighted feature vectors. And I deeply understand the importance of data pre-processing (lots of works were needed before feeding data into the algorithms and get a good results)

Application of Knowledge:

I encountered some troubles of renting during the days after I arrived in Singapore . I found that my first landlord's HDB are not close to ISS building as I imagine, for most of the rental recommendation system usually show us whether the place is close to NUS or not. But NUS is too big, the place might be close to one building of NUS, but not close to customer's specific work or study point. The implementation of our system can ask users to enter specific location and will calculate the specific latitude or longitude of the location. This will help to avoid one of the renting problems I've encountered. Additionally, users' can self define the priorities of different input fields to influence the recommendation outputs they get. And we also show the content of transport and shopping convenience in the output page. All of above functions I mentioned will do contributions to the existing rental recommendation systems.

For me, this journey of system designing will help me to represent the outcome of things like machine learning algorithms in a brief and neat way. It will do great help for my future career.

Name:	Liu Yifeng	**ID:**	A0285847J

Personal Contribution:

I am responsible for the system's backend, including the integration with algorithms and frontend components. I design the database table structures and backend services, facilitate system integration, and actively participate in algorithm development. I maintain open communication with my teammates, discussing matters such as data types and definitions for seamless interconnection between different modules.

Learning Journey and Outcome:

In this project, I gained insights into the composition and construction of Flask applications. I also delved into the functionalities of various Flask extensions and acquired a comprehensive understanding of different aspects of web applications. I realized the crucial role of well-processed data in supporting algorithms effectively. Moreover, I experienced the transition of algorithms from Jupyter Notebook to actual project code. Through this project, I had the opportunity to review and reinforce my knowledge of recommendation systems related to the course, ultimately enhancing my engineering skills.

Application of Knowledge:

While examining the review data in the Airbnb dataset, I recalled the concepts of collaborative filtering and matrix factorization discussed in the course. However, I noticed that there were no actual ratings, only textual reviews. After some research, I came across projects that estimated ratings through sentiment analysis. I decided to experiment with this approach. When faced with conflicts between sentiment analysis results from different Python packages, I attempted to analyse if-else rules, realizing that decision tree training results could be translated into if-else structures.

In the absence of sentiment labels, I considered using Chat-GPT as a LLM for labelling, essentially adopting a self-supervised learning approach. Since decision trees performed poorly with limited data, I thought of using a random forest model. Finally, I analysed the sparsity of the rating matrix and opted for matrix factorization.

Name:	Li Jiaxuan	**ID:**	A0285821Y

Personal Contribution:

I was responsible for designing and developing the user interface to ensure it is both visually appealing and easy to navigate. I employed modern design principles and best practices with Xichen together to make sure users can easily browse recommended property listings, and view detailed information. This contributes to attracting more users and improving their satisfaction.

I worked closely with my team members, to ensure that the front-end design is in harmony with other components of the system. I also contributed to the report for installation guide, class diagram and knowledge base model.

I also actively involved through out project to provide ideas feedback.

Learning Journey and Outcome:

In this project, I gained hands-on experience in the field of machine learning. Learned how to perform data preprocessing and selecting different recommender system algorithms based on the type of data to get better results.

Designing the front-end interface for our housing recommendation system has been exciting and challenging for me, as it was my first experience with user interface and experience design. I gained proficiency in developing front-end applications by working with HTML and JavaScript. I learned how front-end applications facilitate communication with the backend, allowing for data exchange and user interactions. Additionally, I improved my skills in utilizing developer debugging tools and logging information to develop systems more flexibly and quickly.

Application of Knowledge:

Based on my experience with this project, I learned the basic components of an intelligent system and how the parts interact with each other. Armed with the knowledge I've gained from both the lessons and our group project, I'm now well-prepared to assess potential intelligent system projects using the framework introduced in prior courses. The first time for the frontend application design has laid the foundation for my future interdisciplinary projects, such as creating interactive front-end interfaces for machine learning models to monitor and control mechanical-electronic systems. This is highly valuable for the automation and intelligence of machines. Further developed my interdisciplinary skills and gave me the insight to carry out innovative projects in different fields.