Lab 3 Submission

for

<SwipeMyRoomie>

Version 1.0

Prepared by: Sankar Samiksha (U2021021D), Poon Yan Xin Melise(U2022504B), Bansal Arushi(U2023474E), Anagha Subramaniam Ani(U2023534K), Ng Jun Han(U2022688B), Ng Li Wang(U2021106D)

Nanyang Technological University (Singapore), SCSE

First Updated: 16 August 2021

Last Updated: 16 October 2021

Table of Contents

1. Introduction

1.1 Purpose of the document

This document aims to elicit and detail the software requirements (functional and non-functional) for the website. Additionally, a UML diagram and use cases will be provided in section 2.2 and 2.3 respectively.
A class diagram (section 2.5), sequence diagrams (section 2.4) for the respective use cases, and dialog map (section 2.6) would be provided. Furthermore, a UI mockup will be showcased in section 2.5. There is also a data dictionary given in Appendix A. A system architecture would also be included in Appendix B.

1.2 Purpose of the product

In Singapore, by law, it is not possible to own a HDB flat unless you are married or above the age of 35. Hence, university students and young adults need to find places to rent while studying or working in Singapore. This website aims to help students and young adults in their search for roommates. Students often would seek roommates to share the apartment with. Young adults who are immigrants may not have a family here in Singapore and may intend to stay with others. Hence, this website would enable students as well as young adults to find roommates and rent rooms.

1.3 Intended Audience and Reading Suggestions

Before diving further into the product, it is important to understand who the intended audience for this document are.

- Developers are people who will write the codes (**front-end** and **back-end**) for the **device**.
- Project Managers will be in-charge of the project designing the timeline, resources, and logistics for the project.
- Marketing staff will help to promote the system and encourage the public to use the website.
- Testers are employees that help to ensure that there are no bugs in the system and the system is running smoothly. If there are any **bugs**, they will report it to the development team and thereafter, the developers will make a patch for the bug.
- Users are defined as the people who will be utilizing the website. For SwipeMyRoomie, the target users are university students as well as young adults residing in Singapore.
- Documentation writers are those who report about the system requirements and its implementation in the form of a written report.

1.4 Product Scope

This product is intended to be used as an interactive website for students and young adults looking for roommates to communicate with each other. Consequently, they will be able to find roommates that they are compatible with.

Currently, the scope of the product will cover only **HDB rental services**. If the initial product release is successful, **condominium** and **apartment** rentals can be included. This will enable **private housing businesses** to find tenants who are students and young adults. Hence, increasing business opportunities for the private housing sector in Singapore.

Additionally, the product currently would allow students and young adults to create an account free of cost.

1.5 References

<List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.>

2. Overall Description

2.1 Product Perspective

<Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.>

This website is a new self-contained product. Hence, no further discussion will happen in section 2.1.

2.2 Product Functional Requirement

2.2.1 The system must allow the user to register for an account with their desired username and password.

2.2.2 The system must allow the user to login to their account.

2.2.3 The system must allow the user to view their profile and display their personal information.

2.2.4 The user must be able to update their profile based on their name, gender, age, occupation, ethnicity, and customise their roommate preferences.

2.2.5 The system must be able to search for appropriate roommates for the user based on the user's updated profile.

2.2.6 The system must allow the user to use the search function to input a location they prefer for their apartment and display the relevant apartment information to the user.

2.2.7 The system must allow the user to access the ChatBox to send and receive messages with other tenants.

UML

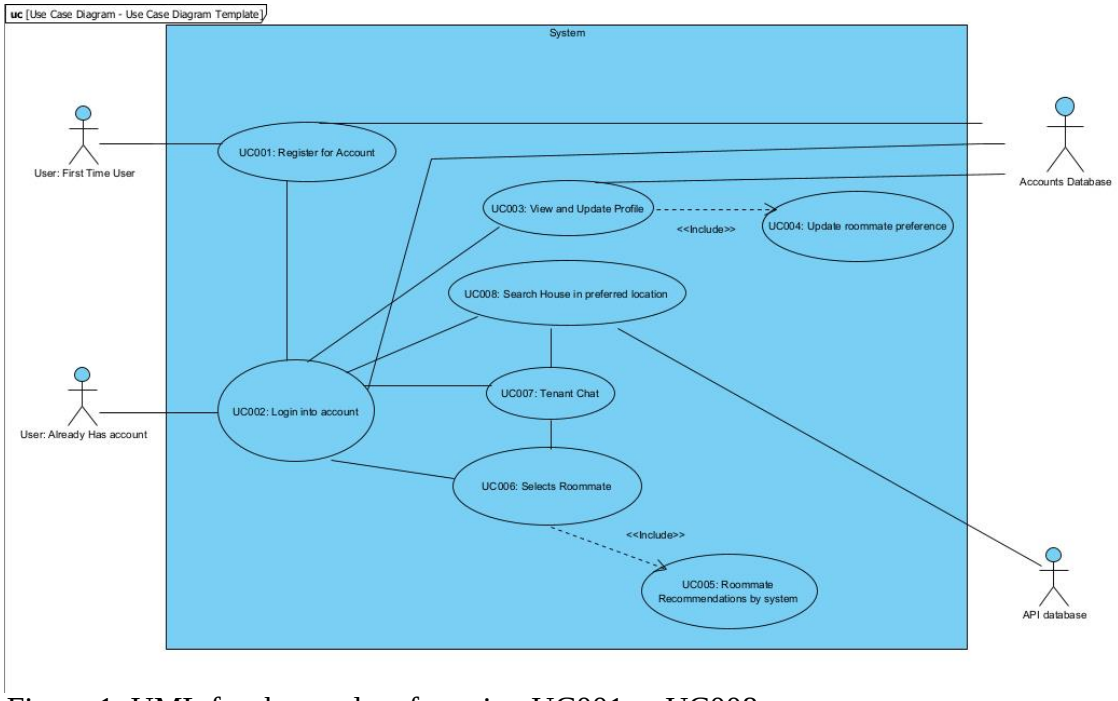


Figure 1: UML for the product featuring UC001 to UC008

2.3 Use Case

- UC001: Register for account
- UC002: Log in to the account
- UC003: View and Update Profile
- UC004: Update roommate preferences
- UC005: Roommate Recommendation
- UC006: Selects Roommate
- UC007: Tenants Chat
- UC008: Search for house in preferred area

Use Case ID:	UC001		
Use Case Name:	Register for new account		
Created By:	Ng Li Wang	Last Updated By:	Arushi
Date Created:	23 August 2021	Date Last Updated:	16 October 2021

Actor:	User, accounts Database
Description:	Register a new user account
Preconditions:	1) User has a valid email address to be used as login ID 2) User's email address has not been registered in accounts database
Postconditions:	1) New user account is created in the accounts database 2) The website redirects user to home page
Priority:	High
Frequency of Use:	1 per user in lifetime
Flow of Events:	Register: 1) The user enters an email and password. 2) The system checks with accounts database if there is an existing email address being used. 3) The system then checks if the password fulfils minimal criteria of having at least one upper case character, lower case character and a number. 4) The user then re-enters their desired password into the "Confirm Password" 5) The user then re-enters their desired password into the "Confirm Password" input box. 6) The user clicks "CREATE ACCOUNT" button. 7) The system confirms whether the password in the password and confirm password input box is the same (case sensitive). If it is, continue to the next step. 8) The system adds a new user record to the accounts database 9) The website prompts success message "Account Registered". 10) The website redirects users to home page
Alternative Flows:	If user inputs an email that has been used in the accounts database: 1) The system will display an error message "email already exists" below email input text field and return to step 1 If the user inputs a username of the wrong format (minimum criteria of example@mail.com): 1) The system will display an error message "invalid username (e.g. example@mail.com)" below username input text field. 2) It will clear the username entered and return to step 1.

	If the user inputs a password of the wrong format (minimum criteria of one uppercase, one lowercase, and one number): 1) The system will display an error message "invalid password (min. one uppercase, one lowercase and one number)" below password input text field. 2) It will clear the password entered and return to step 1. If the password and confirm password input fields are different: 1) The system will display an error message "password does not match" below the password input text field. 2) It will clear the password fields and return to step 1.
Exceptions:	If the user triggers alternative flow more than 3 times (entering existing email or invalid password), the user will be asked to try again later.
Includes:	-
Special Requirements:	-
Assumptions:	The user is a first-time user on SwipeMyRoomie. The server is working and there are no bugs in the system.
Notes and Issues:	-

Use Case ID:	UC002		
Use Case Name:	Login into account		
Created By:	Sankar Samiksha	Last Updated By:	Arushi
Date Created:	23 August 2021	Date Last Updated:	16 October 2021

Actor:	User, accounts Database
Description:	User will perform login procedure
Preconditions:	1) User has registered for an account
Postconditions:	1) User has logged into the account 2) Website redirects user to main page with user logged in
Priority:	High
Frequency of Use:	Every time the user uses the website.
Flow of Events:	1) The user clicks on Log In button on main page which redirects them to input their Email and Password. 2) The user enters their Email. 3) The user enters their Password. 4) The user clicks the log in button. 5) The system will search for the email in the accounts database. The email is case sensitive. 6) If the email is found, the system will check if the corresponding password saved in the accounts database matches with the entered password. 7) If the password matches, the user is logged into the website. 8) The website redirects users to home page
Alternative Flows:	Failed password/email: If user inputs an invalid password but valid email 1) The website displays 'Incorrect password' 2) The website prompts the user to reenter the password again. If the user inputs an email that is not found in the database, 1) The website displays 'Incorrect Email'
Exceptions:	If the user types in the wrong password 3 times, the user will be redirected to the main page.
Includes:	-
Special Requirements:	-
Assumptions:	The user has an existing account on SwipeMyRoomie. The database has successfully stored the user's account details. The server is working and there are no bugs in the system.
Notes and Issues:	-

Use Case ID:	UC003		
Use Case Name:	View and Update Profile		
Created By:	Bansal Arushi	Last Updated By:	Arushi
Date Created:	28 August 2021	Date Last Updated:	16 October 2021

Actor:	User, profile database, user_language database
Description:	User will be able to view his/her profile including his name, gender, age, occupation, ethnicity, and preferred spoken language. User will be able to update their profile.
Preconditions:	1) The user has an account on the website
Postconditions:	1) User can see their profile on one page and can scroll through the contents. 2) User can update their profile.
Priority:	High
Frequency of Use:	Whenever the user logs in, there is a high possibility of them viewing their profile. Hence, frequency can range from daily to once a month. Updating profile will not be as regular.
Flow of Events:	1) The user clicks their profile icon 2) The system gathers the user's information from the profile database and the user_language database. 3) The user is redirected to another page. 4) The system then displays the user's profile information on one page. 5) The user can now view their profile by scrolling through this page. 6) When the user clicks the "update self profile" button, they can update their profile by updating information about their name, gender, age, occupation, ethnicity and preferred spoken language. 7) The user clicks on "update profile". 8) The system updates the information in the profile database and the user_language database.
Alternative Flows:	If the person has no information about themselves 1) The page will show a "NIL" at the name, gender, age, occupation, ethnicity and preferred spoken language output text fields.
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	The user has an existing account on SwipeMyRoomie. The database has successfully stored the user's profile details. The server is working and there are no bugs in the system.
Notes and Issues:	-

Use Case ID:	UC004		
Use Case Name:	Update Roommate Preference		
Created By:	Sankar Samiksha	Last Updated By:	Arushi
Date Created:	28 August 2021	Date Last Updated:	16 October 2021

Actor:	User, profile database, roommate_language database
Description:	User will be able to update his/her roommate preference with regards to gender, age, occupation, ethnicity, and preferred spoken language.
Preconditions:	1) The user has an account on the website
Postconditions:	1) The user can set their roommate preference
Priority:	High
Frequency of Use:	The user may not frequently change their preferences for roommate. Hence, it could be once every few months.
Flow of Events:	1) The user clicks their profile icon 2) The user clicks on "update roommate preference" button. 3) The user can update the preference for their roommate with regards to gender, age, occupation, ethnicity and preferred spoken language. 4) The user clicks on "submit". 5) The preferences are updated into the profile database and roommate_language database. 6) This triggers UC005 automatically.
Alternative Flows:	If the person has no information about roommate preference. 1) The page will show a "NIL" at the name, gender, age, occupation, ethnicity and preferred spoken language output text fields.
Exceptions:	-
Includes:	UC005
Special Requirements:	-
Assumptions:	The user has an existing account on SwipeMyRoomie. The databases have successfully stored the user's roommate preference. The server is working and there are no bugs in the system.
Notes and Issues:	-

Use Case ID:	UC005		
Use Case Name:	Roommate Recommendation		
Created By:	Sankar Samiksha	Last Updated By:	Poon Yan Xin Melise
Date Created:	6 September 2021	Date Last Updated:	16 October 2021

Actor:	Accounts Database
Description:	User updating the roommate preference (UC004) would trigger this particular use case. In this use case, the system would match the user profile to other user profiles to give the user a recommended roommate.
Preconditions:	The User has gone through UC004 and updated their roommate preference.
Postconditions:	The user can now view their recommended roommates.
Priority:	High
Frequency of Use:	Daily recommendations will be given. Hence, UC005 is used daily.
Flow of Events:	1) Completion of UC004 will trigger the recommendation algorithm on the website. 2) Firstly, the system will filter users by gender preference. If there is no gender preference, the system moves on to the next step. 3) Secondly, the system will filter users by age preference within a range of +/- 5 years. If there is no age preference, the system moves on to the next step. 4) Thirdly, the system will filter users by occupational preference. If there is no occupational preference, the system moves on to the next step. 5) Then, the system will filter users by ethnicity. If there is no ethnicity preference, the system moves on to the next step. 6) Then, the system will filter users by language preferences. If there is no language preference move on to the next step. 7) The system will display the profiles of the filtered user on the roomie recommendation page.
Alternative Flows:	If roommate preferences have not been updated, The system will randomly pick users to display in the roommate recommendation page.
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	The user has an existing account on SwipeMyRoomie. The database has successfully stored the user's roommate preference. The server is working and there are no bugs in the system.
Notes and Issues:	-

Use Case ID:	UC006		
Use Case Name:	Select roommates		
Created By:	Anagha Ani	Last Updated By:	Poon Yan Xin Melise
Date Created:	28 August 2021	Date Last Updated:	16 October 2021

Actor:	User, Profile Database
Description:	The system will display the profiles of other users who best match their preferences on the home page. Users can then select which roommates they prefer.
Preconditions:	1) The user has logged into the system. 2) UC005 triggers successfully and can display users on the home page.
Postconditions:	1) The system displays the profiles of other users who best match the user's preferences. 2) The user performing the search can click on the profiles and thereafter through UC007 and UC008 contact the other users to find rental units that fit both of their preferences.
Priority:	High
Frequency of Use:	Daily. Users would use this function daily until they have managed to search for an appropriate roommate for themselves.
Flow of Events:	1) Completion of UC005 would display the profiles of the filtered users on the home page. 2) The user can click on the right arrow to view the profile of the next recommended user. 3) The user can click on the left arrow to view the profiles of the previous recommended user. 4) The user can also click on the blue "skip" button. 5) The system removes that filtered user from the user's recommendation page. 6) The user can click on the blue "chat" button to launch the Messages panel (UC007) if they wish to chat with other users.
Alternative Flows:	-
Exceptions:	-
Includes:	UC007
Assumptions:	The user has an existing account on SwipeMyRoomie. The server is working and there are no bugs in the system.
Notes and Issues:	-

Use Case ID:	UC007		
Use Case Name:	Tenant chat		
Created By:	Poon Yan Xin Melise	Last Updated By:	Poon Yan Xin Melise
Date Created:	23 August 2021	Date Last Updated:	16 October 2021

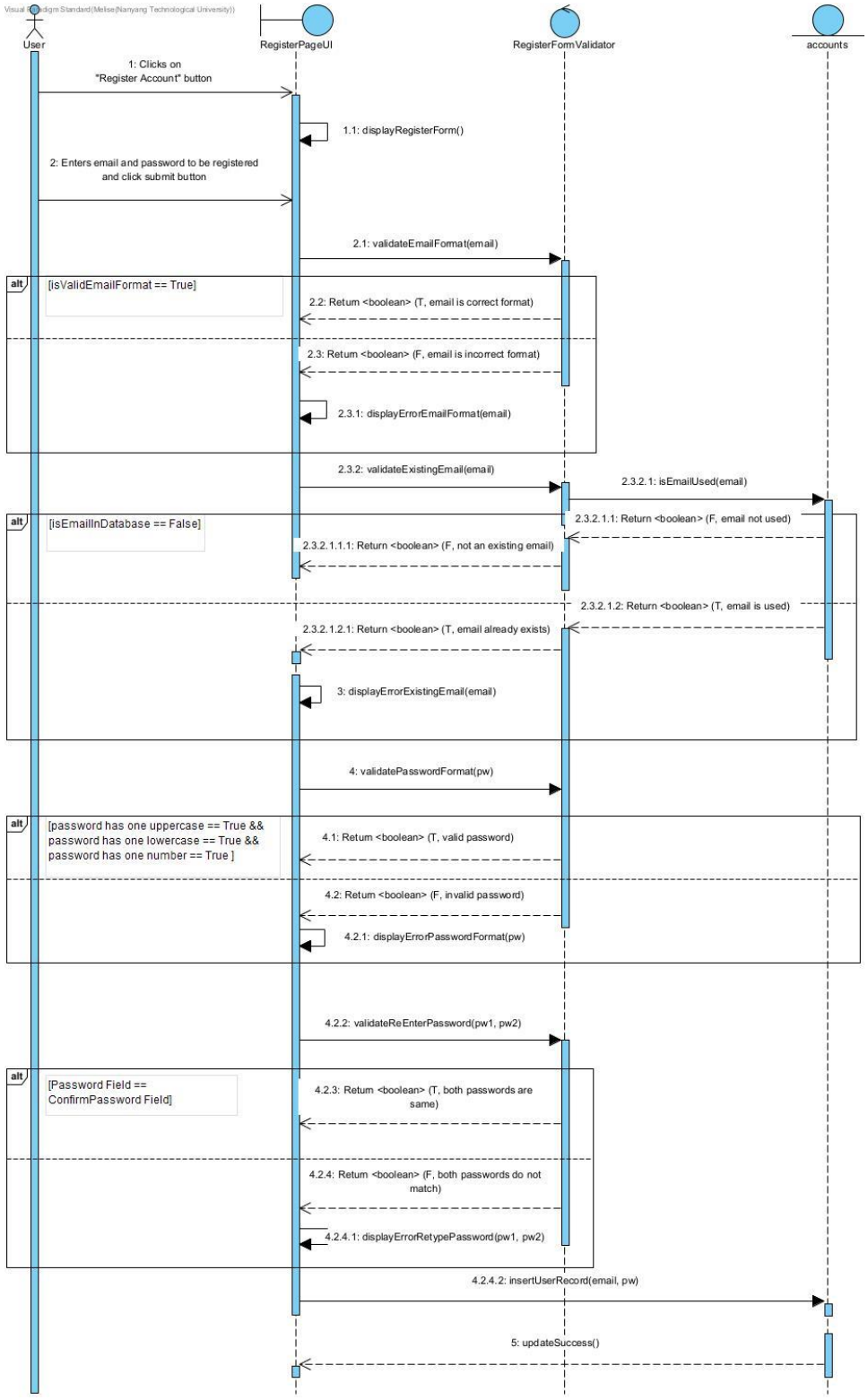
Actor:	User, Chat Database.
Description:	The User would be able to send and receive messages with another user on the website.
Preconditions:	1) The system database stores the data of existing chats. 2) The user has clicked on "chat" button from UC006.
Postconditions:	Users can exchange messages with another user to discuss personal preferences about being roommates.
Priority:	High
Frequency of Use:	From daily to weekly. This depends on the frequency of messages sent between the users.
Flow of Events:	1) The User clicks on the blue "chat" button from UC006. 2) The system launches a new chat with that filtered user on the panel at the left. 3) User types in a message. 4) User clicks on the send button. 5) System displays the message on the receiver's profile as a new message. 6) The receiver can now choose to respond, the flow of events is repeated. 7) Upon confirmation from both parties to be roommates, both users can click on red "Confirm Stay" button to confirm the pairing. 8) System displays a notification on both users' profile for the successful match. 9) This triggers UC008 automatically.
Alternative Flows:	From the main page, 1) User can click on "chat" button 2) User can click an existing chat. 3) Continue from Step 3 in main flow.
Exceptions:	-
Includes:	UC008
Assumptions:	The user has an existing account on SwipeMyRoomie. The server is working and there are no bugs in the system.
Notes and Issues:	-

Use Case ID:	UC008		
Use Case Name:	Search for house in preferred area		
Created By:	Sankar Samiksha	Last Updated By:	Poon Yan Xin Melise
Date Created:	23 August 2021	Date Last Updated:	16 October 2021

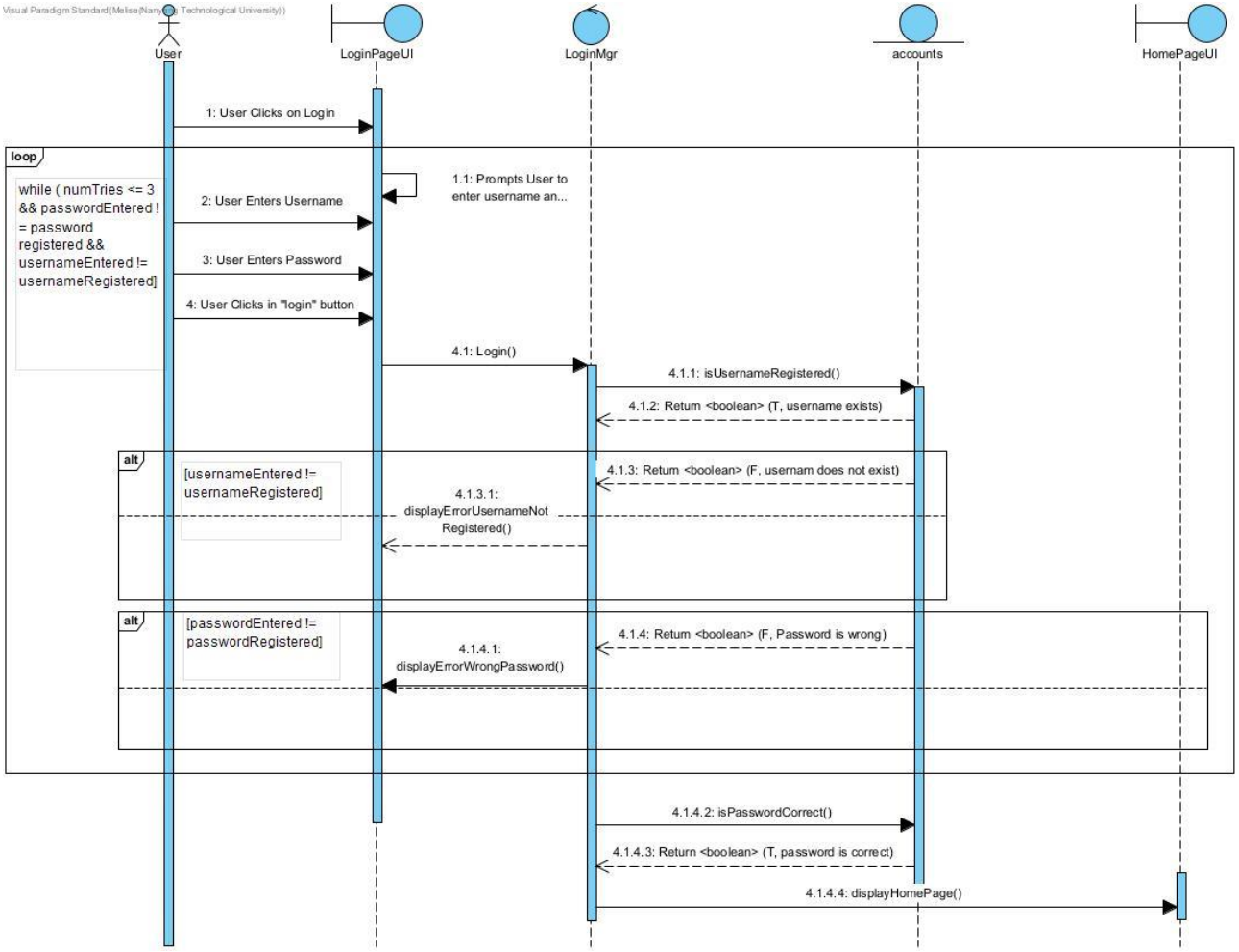
Actor:	User, API database
Description:	Searches for houses within 5km radius of the search location.
Preconditions:	The user has matched with a roommate in UC007.
Postconditions:	1) The houses within the 5km radius are shown. 2) The nearest facilities within the 5km radius are shown.
Priority:	High
Frequency of Use:	Daily while the user is using the website.
Flow of Events:	1) The user inputs their location (e.g., Punggol/Sengkang). 2) The system searches the API database to find all the houses that are available and within 5 km. 3) The system orders the houses based on distance (in ascending order). 4) The system searches the database for the nearest facilities. 5) The system prints the list of houses found with the name and the address of the houses. 6) The user can click on the house location to access more information about the house. 7) Upon clicking on the house location, the system displays the range of nearby facilities and a range of pictures of the house.
Alternative Flows:	If user inputs a location that has no valid address, 1) The system displays an error message. 2) The system returns to ask the user for another location. If user inputs an address with no houses available in the vicinity, 1) The system displays a “no houses available” message.
Exceptions:	-
Includes:	-
Assumptions:	The user has an existing account on SwipeMyRoomie. The server is working and there are no bugs in the system. Since the API for the house database is sourced externally, it is assumed that at the time of implementation the API is available.
Notes and Issues:	-

2.4 Sequence Diagram

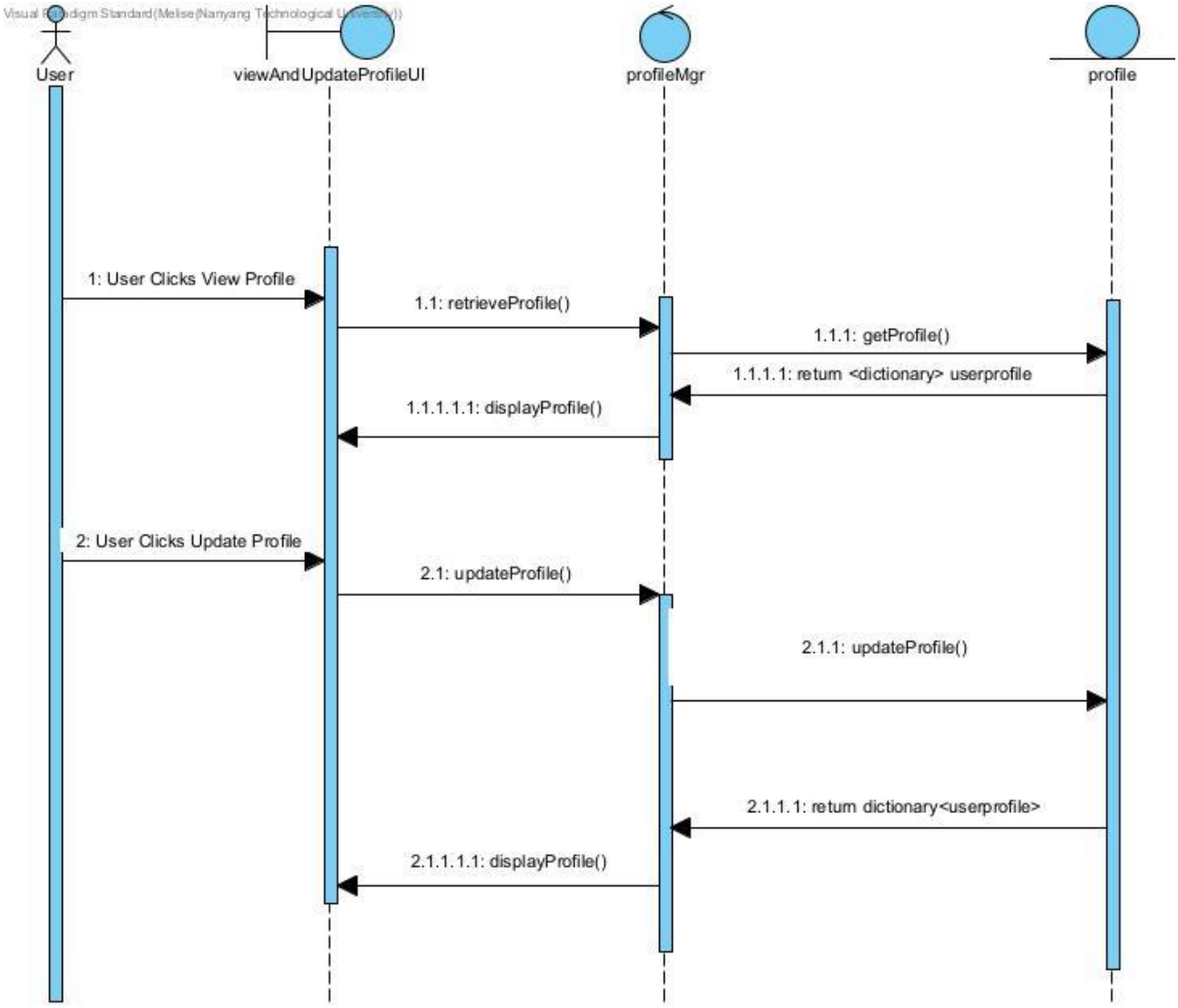
2.4.1 User Case 1



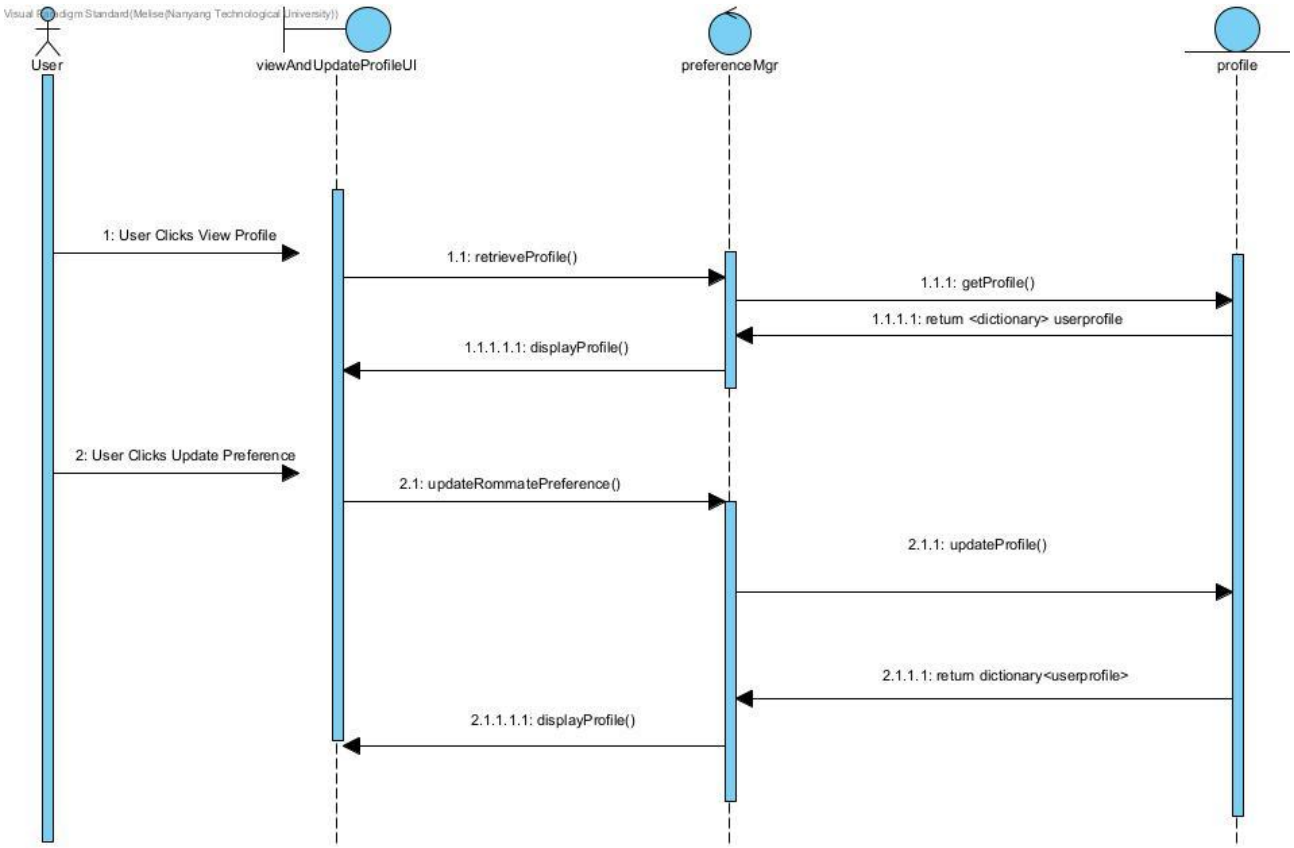
2.4.2 User Case 2



2.4.3 User Case 3

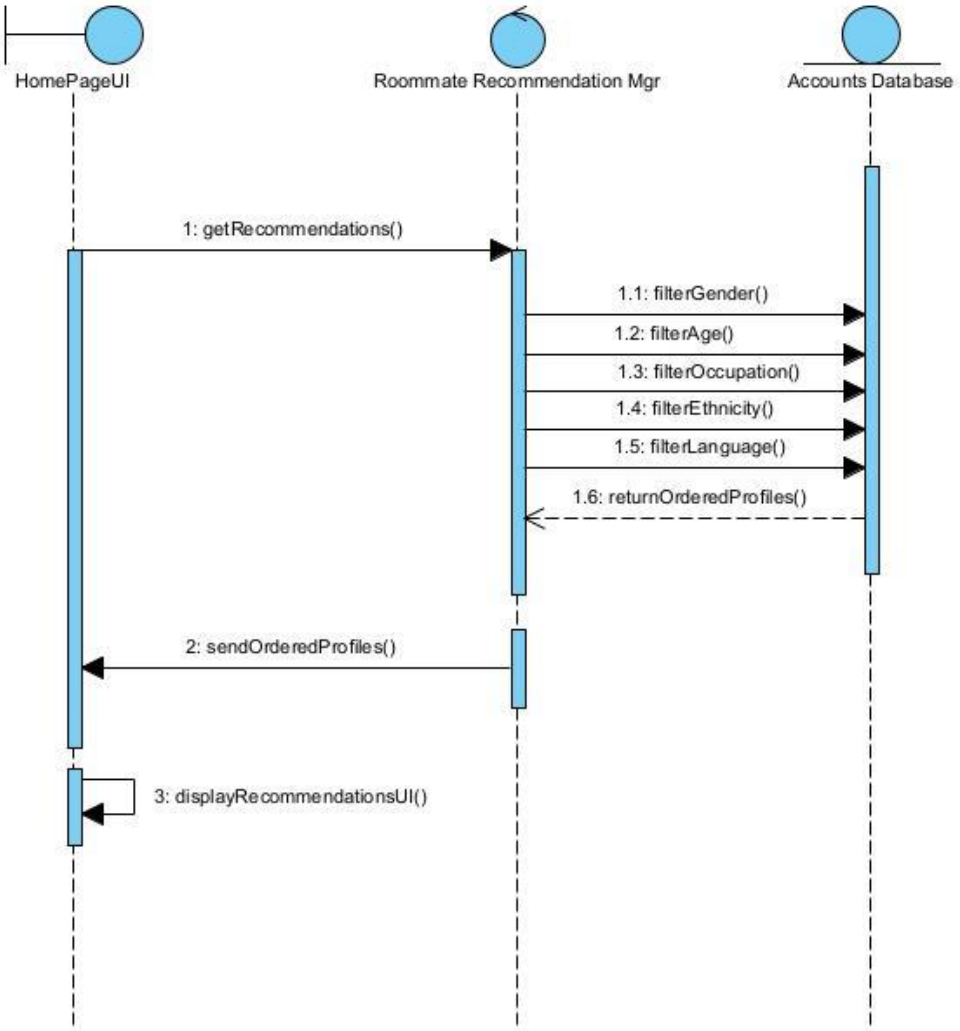


2.4.4 User Case 4

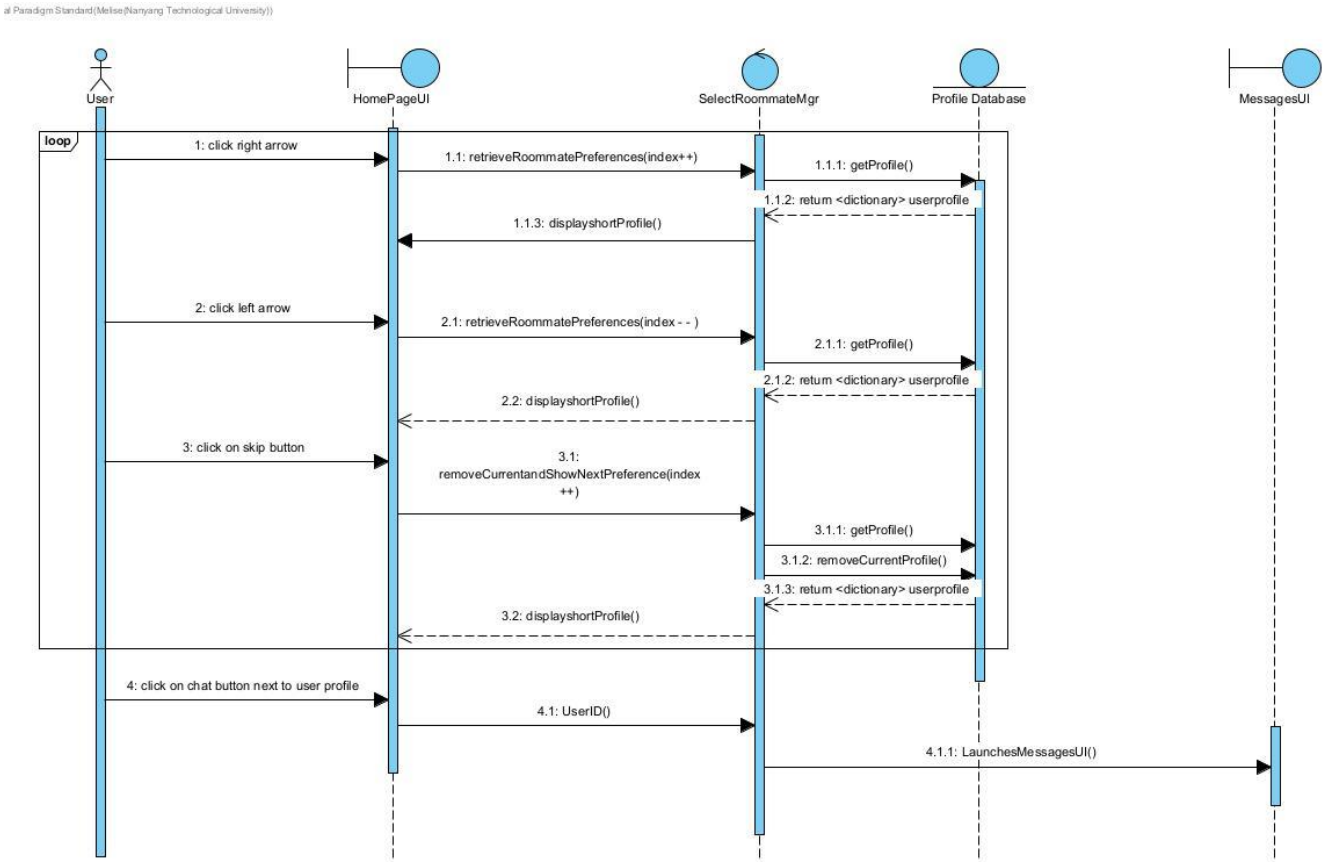


2.4.5 User Case 5

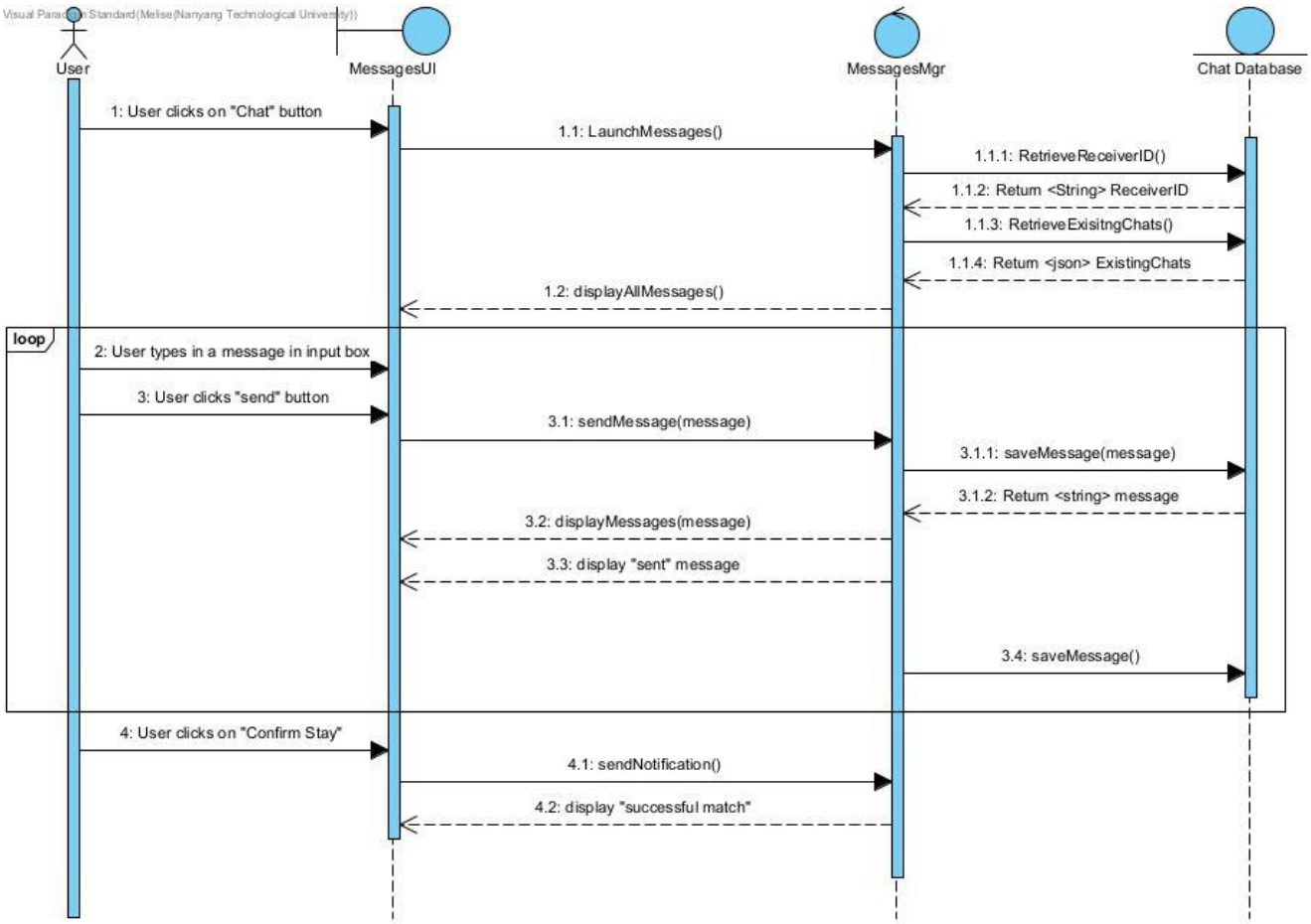
al Paradigm Standard (Melise (Nanyang Technological University))



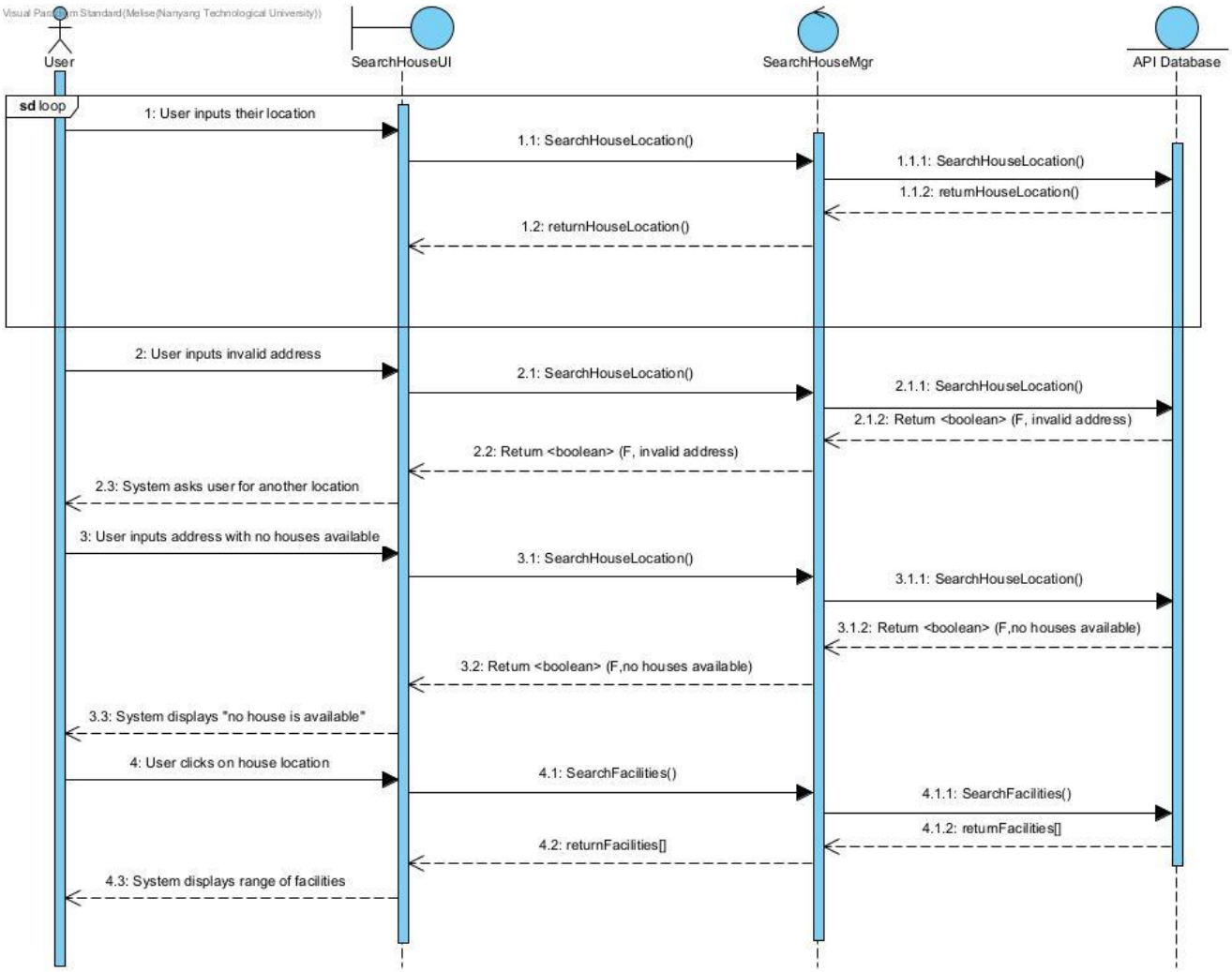
2.4.6 User Case 6



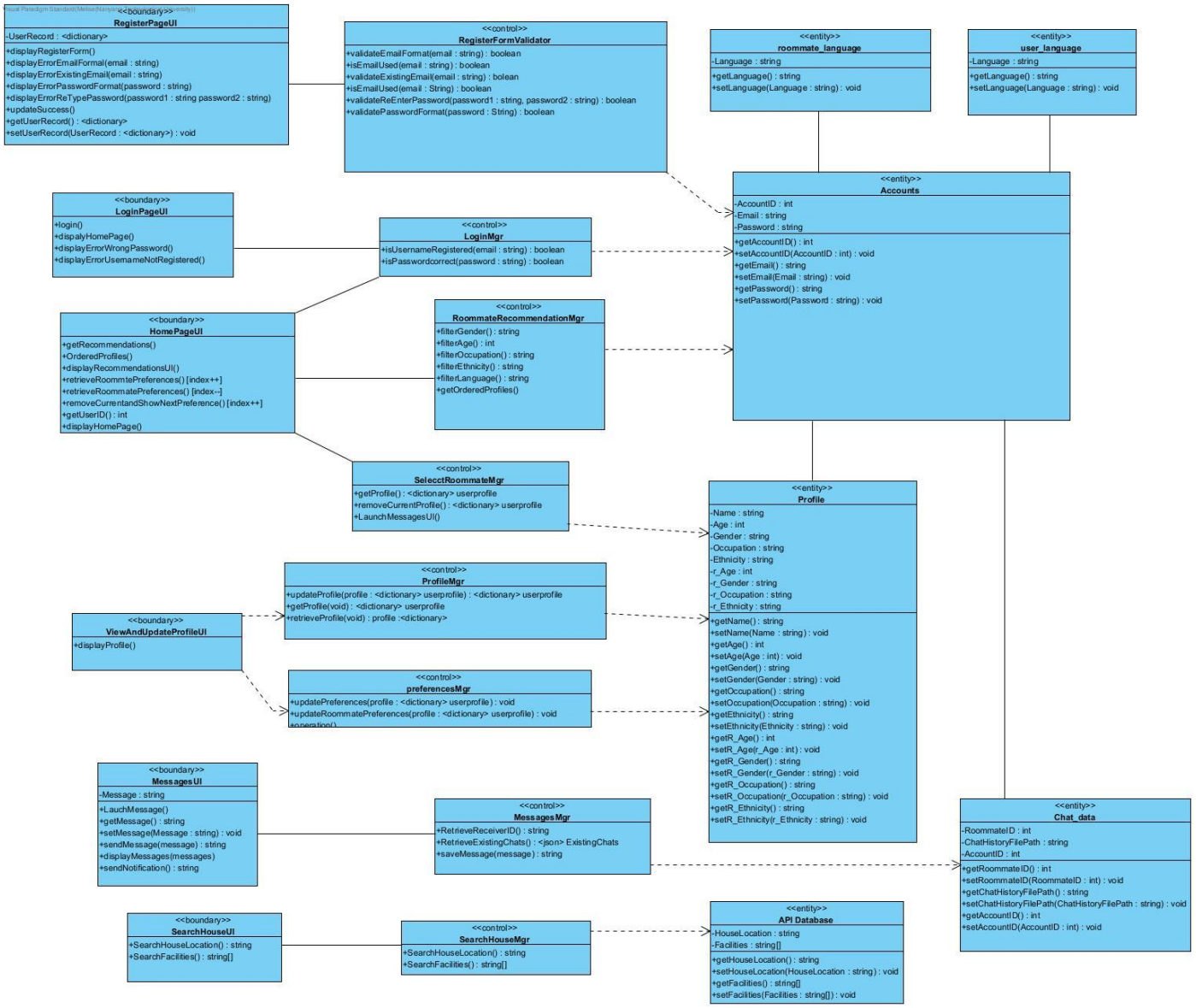
2.4.7 User Case 7



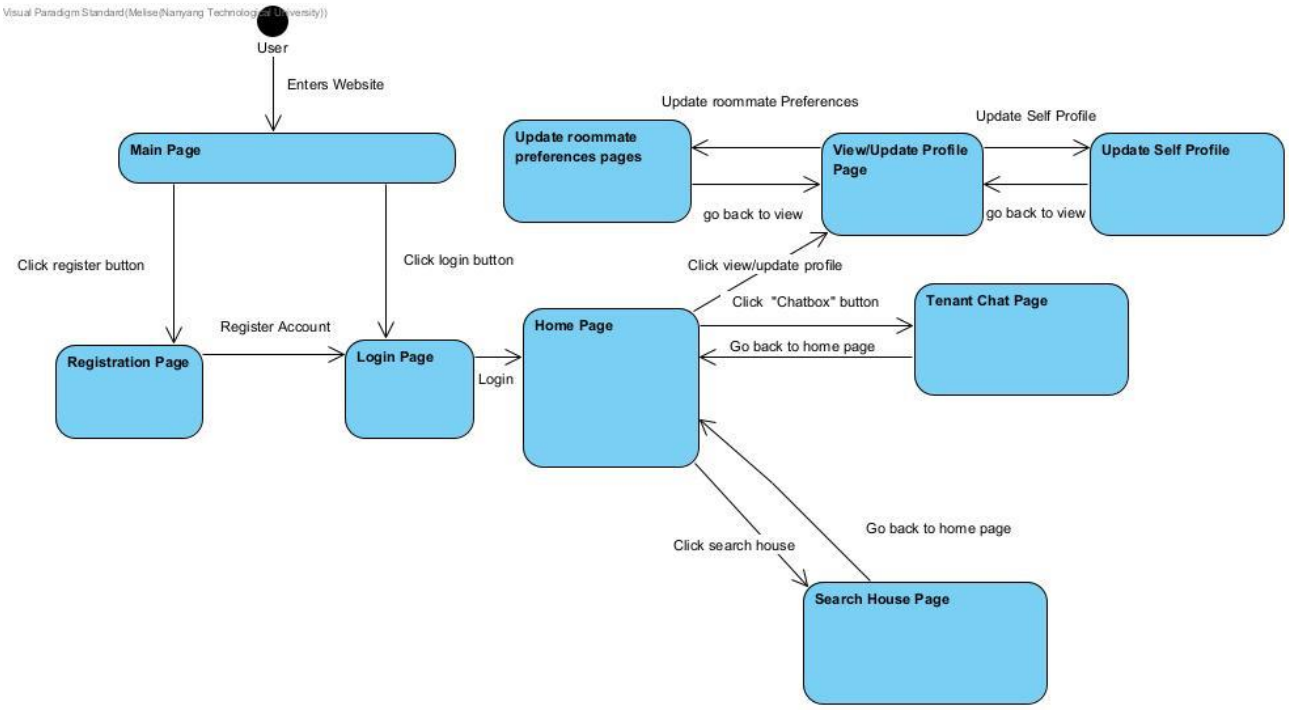
2.4.8 Use Case 8



2.5 Class Diagram



2.6 Initial Dialog Map



2.7 Operating Environment

This product, a dynamic website, will be able to work on any operating system (linux, unix, Windows).

2.8 Design and Implementation Constraints

<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer's organization will be responsible for maintaining the delivered software).>

2.9 User Documentation

<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>

2.10 Assumptions and Dependencies

<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>

3. External Interface Requirements

3.1 User Interfaces

In this section, the UI mockup will be explained in detail.

Figure 1: The main page

In the home page, two buttons are provided – “REGISTER” and “LOG IN”. These two buttons will be used by new users (those without an account) and those with an existing account respectively. There will also be a navigation bar to allow users to surf the website with ease.

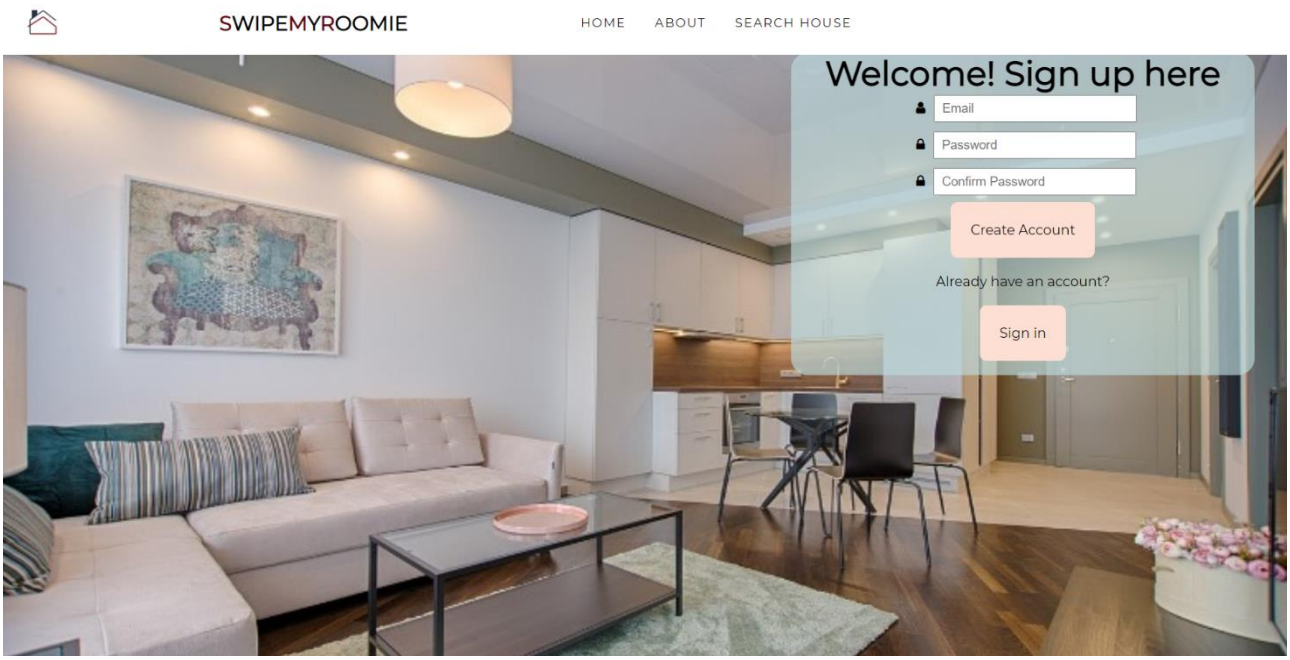


Figure 2: Registration page for new users

In the registration page, the new user can input an email and create a password. In this page, a “sign in” button is also provided should users want to be redirected to the login page.

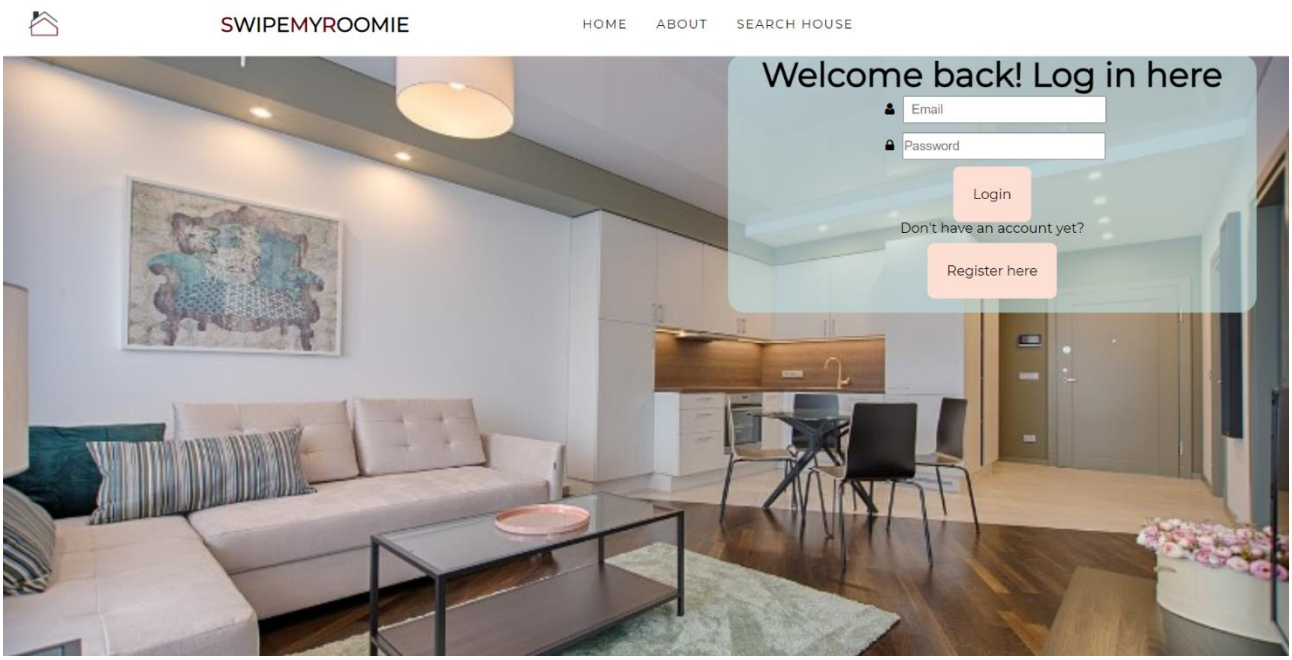


Figure 3: The login page

In this login page, the user can input their email and password to login into their account.

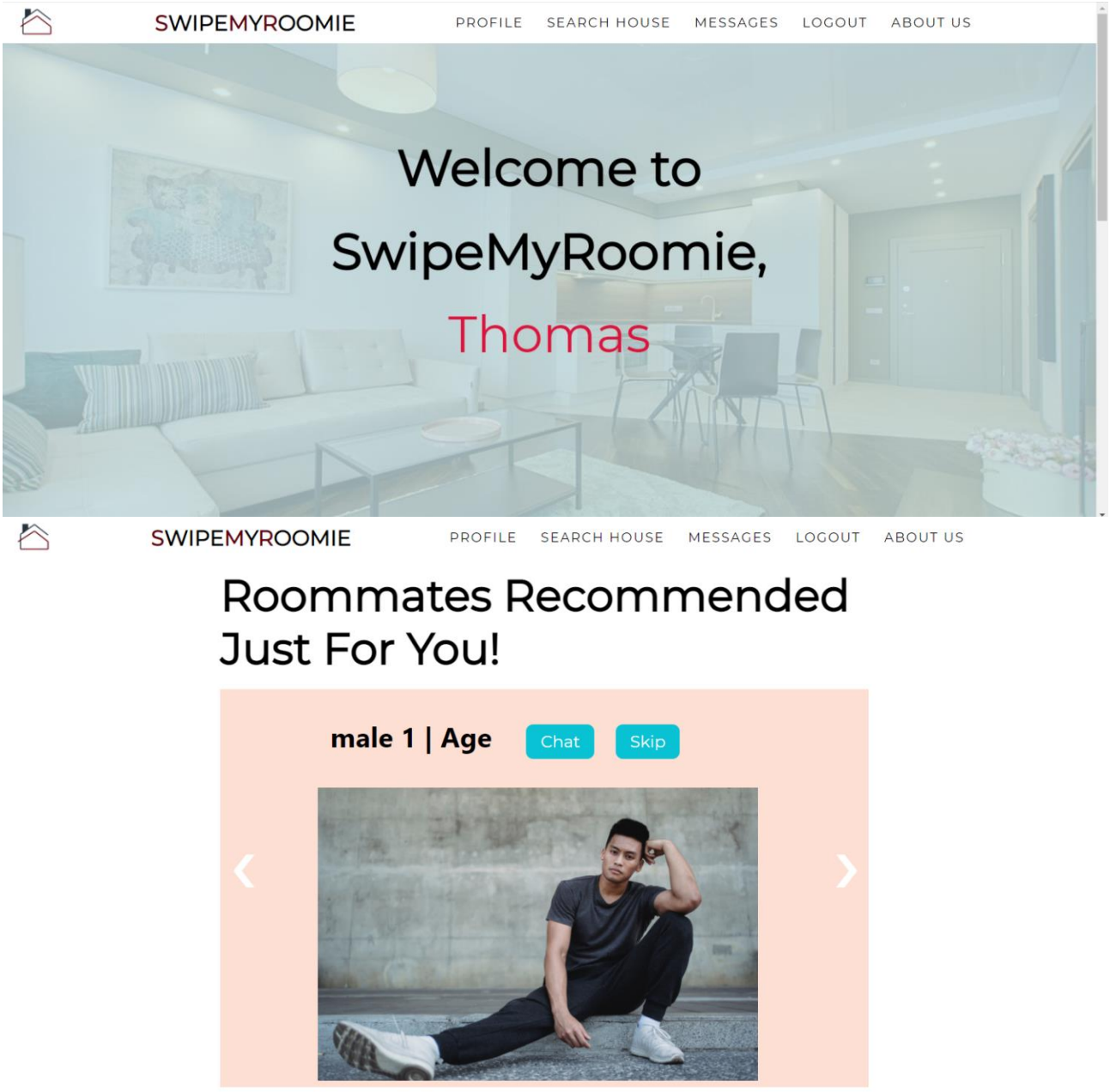


Figure 4: The home page

User can find their recommended roommates directly on the home page

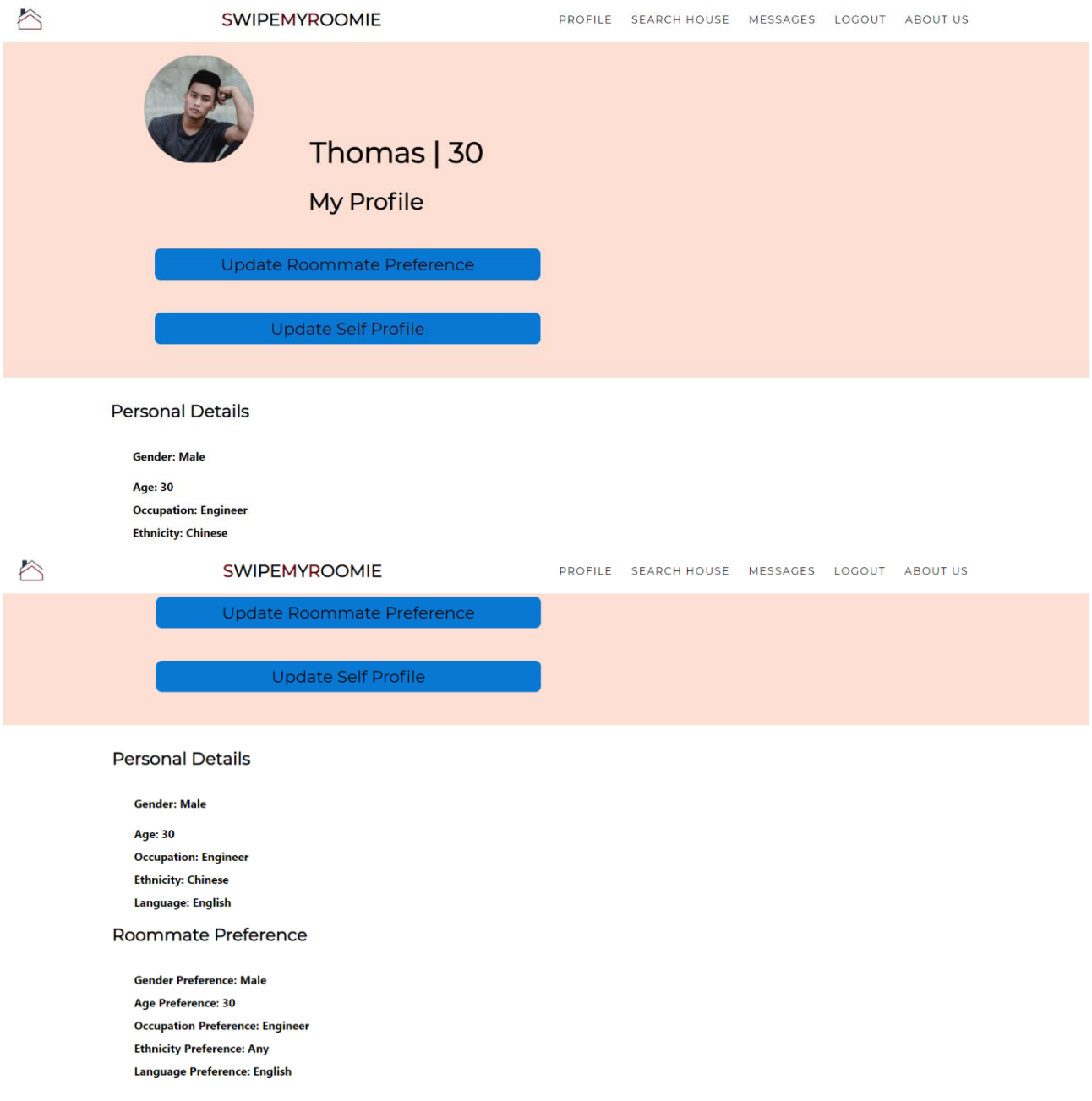


Figure 5: The profile page

Viewing further details about the user's profile and their preferred roommate preferences. The user can click on "Update Roommate Preference" which brings them to Figure 7 to update their roommate's preferences. The user can also click on "Update Self Profile" which brings them to Figure 6 to update their profile.

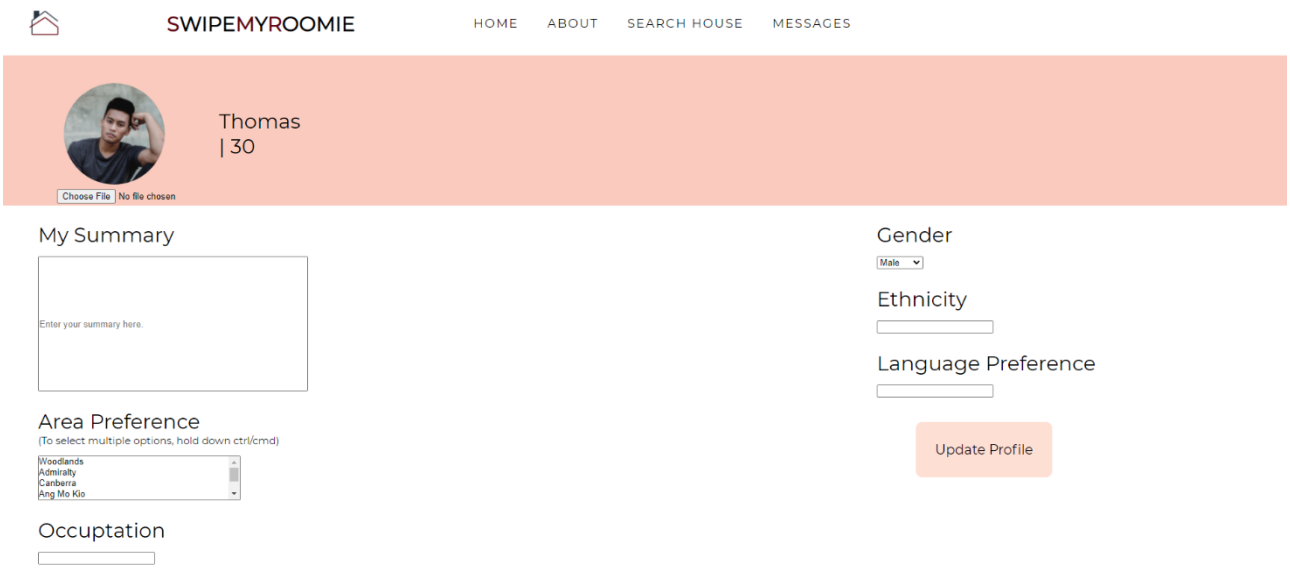


Figure 6: Updating Profile

The above page allows users to update their profile. The user can provide a short description of himself/herself, input their area preference for the apartment and their occupation.
The user can also input their Gender, Ethnicity and Language preference.
The user can then click on “Update Profile” to save the changes made.

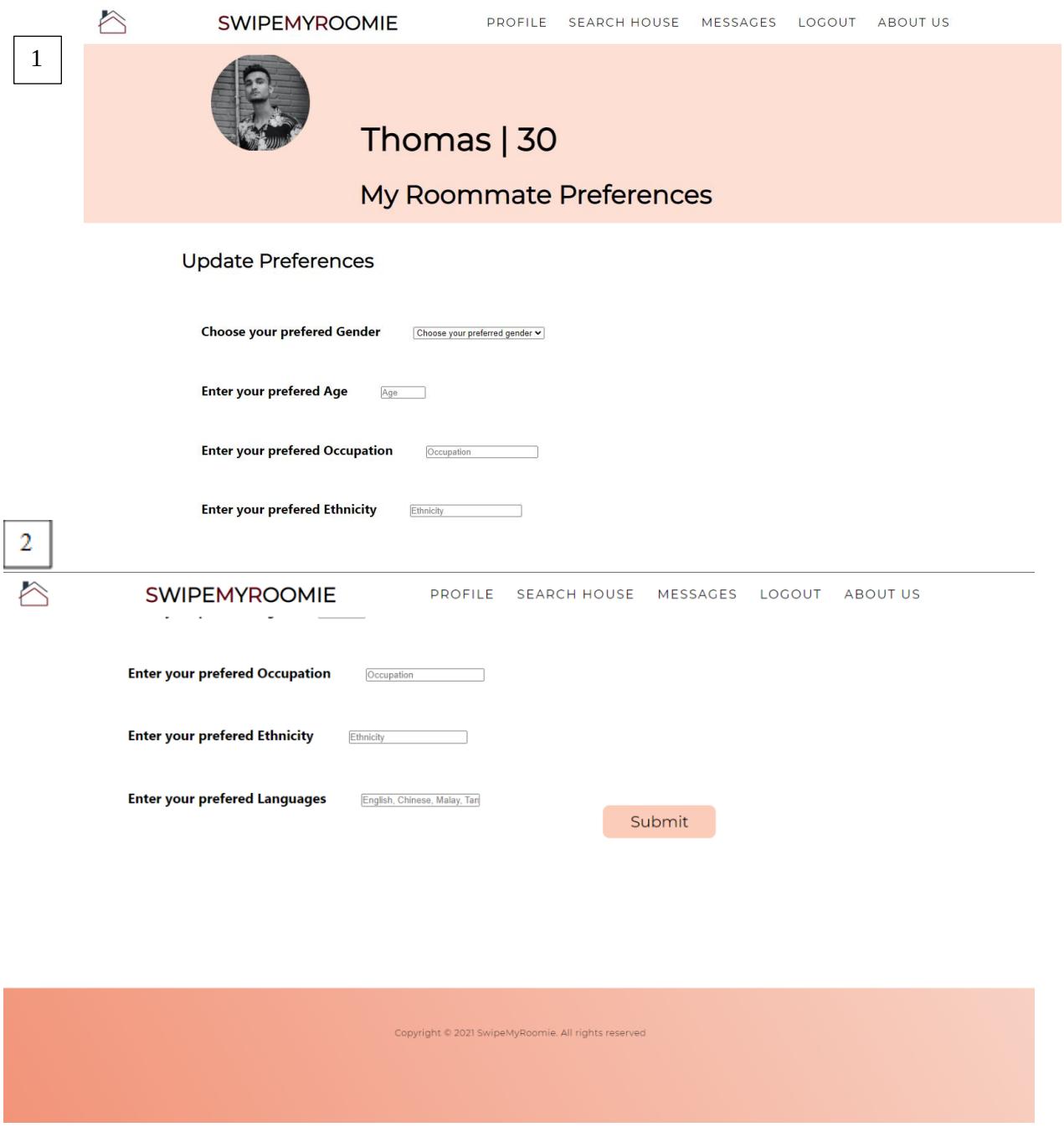


Figure 7: Updating roommate preference

The user can update their preferred roommate preference based on Gender, Age, Occupation, Ethnicity and Language.
They can click on “submit” when they are done with the inputs.

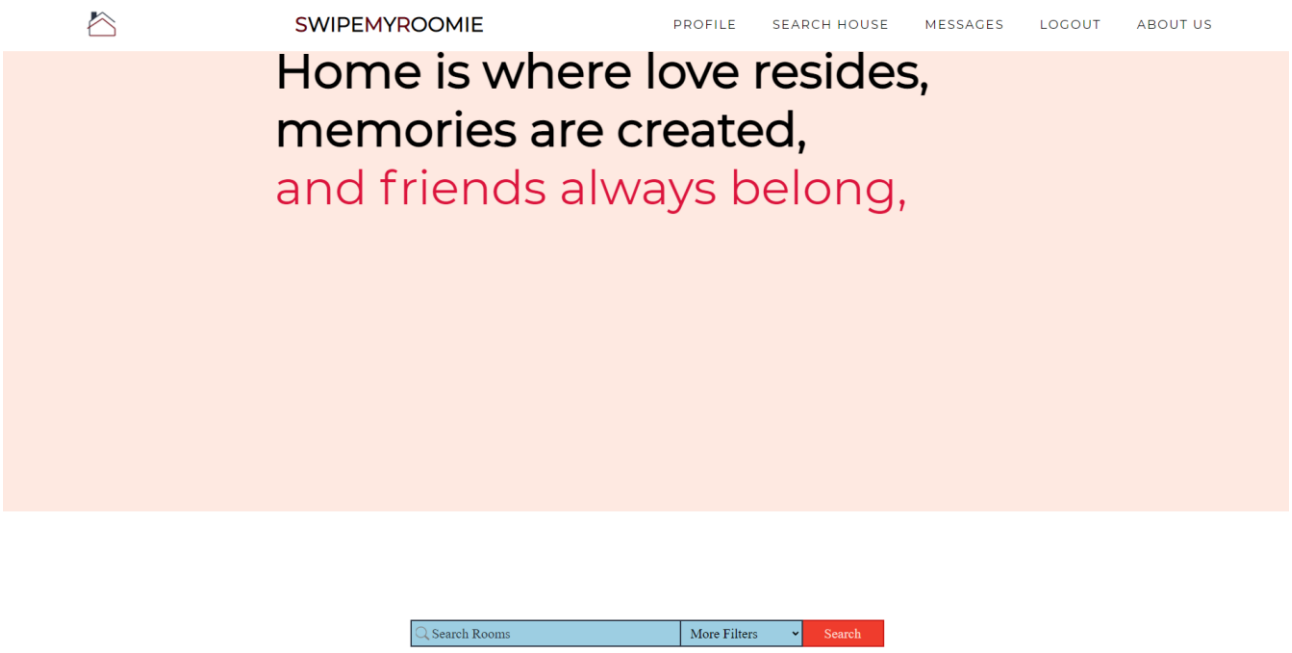


Figure 8: Search for houses

After the creation of accounts and finding their preferred roommate, the user and his roommate can search rooms based on their location preferences.

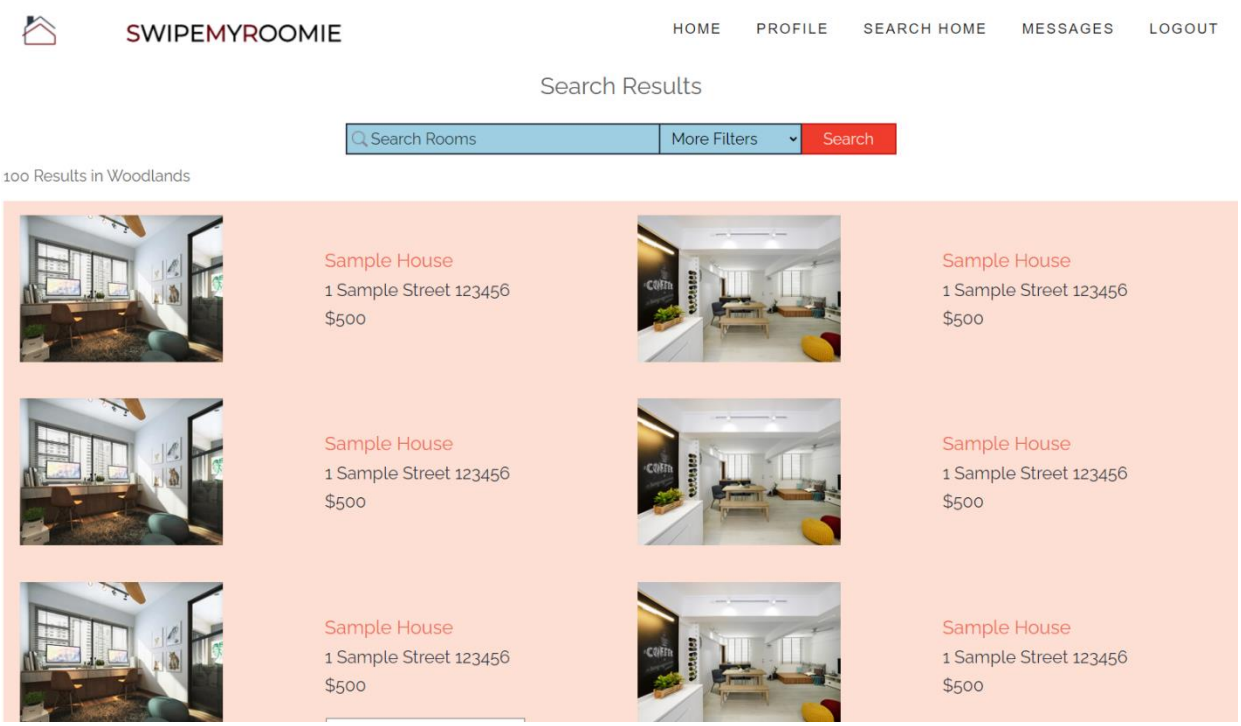


Figure 9: Search Results

If the user searches Woodlands for example, all the houses and their rental rates will be displayed on the screen.

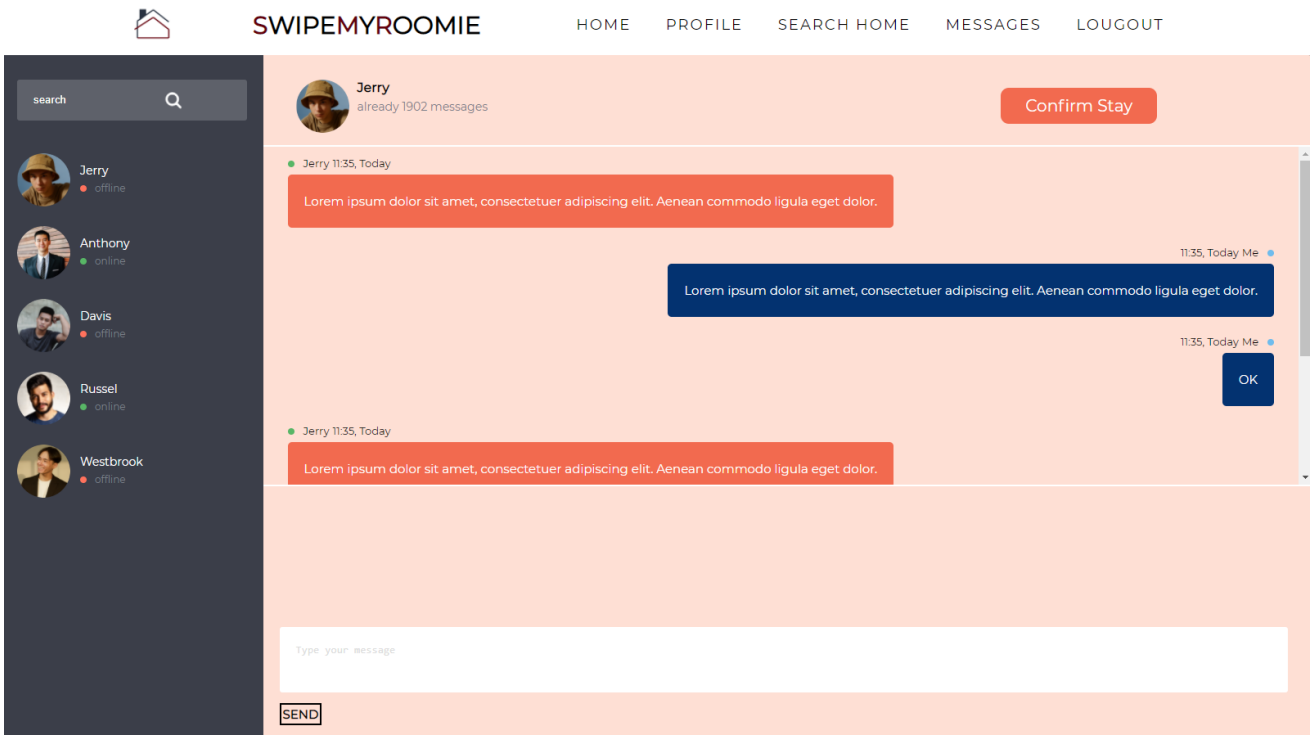
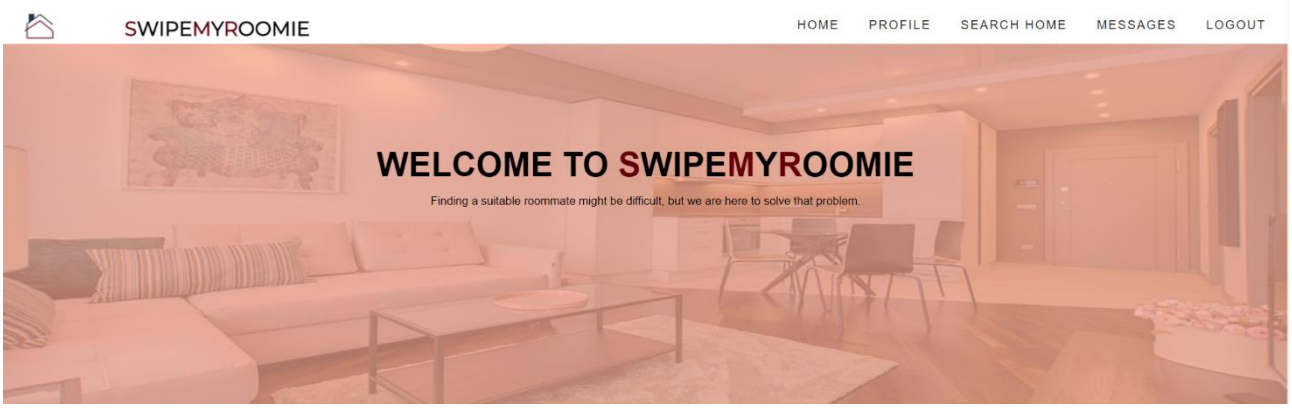


Figure 10: The chat feature

In this page, the user can chat with other users of similar preferences. They can type their message in the white box provided and click on “send” to send the message to the tenant they are messaging with.



Choose The Perfect **Roommate**

As university students and young adults need to find places to rent while studying or working in Singapore. Finding the right roommate might be difficult due to difference in lifestyle, personality. And SwipeMyRoomie aims to address that problem.

Our **Algorithm**

Simply answer some questions in the profile's page and the algorithm will handle the rest and recommend you suitable roomies!

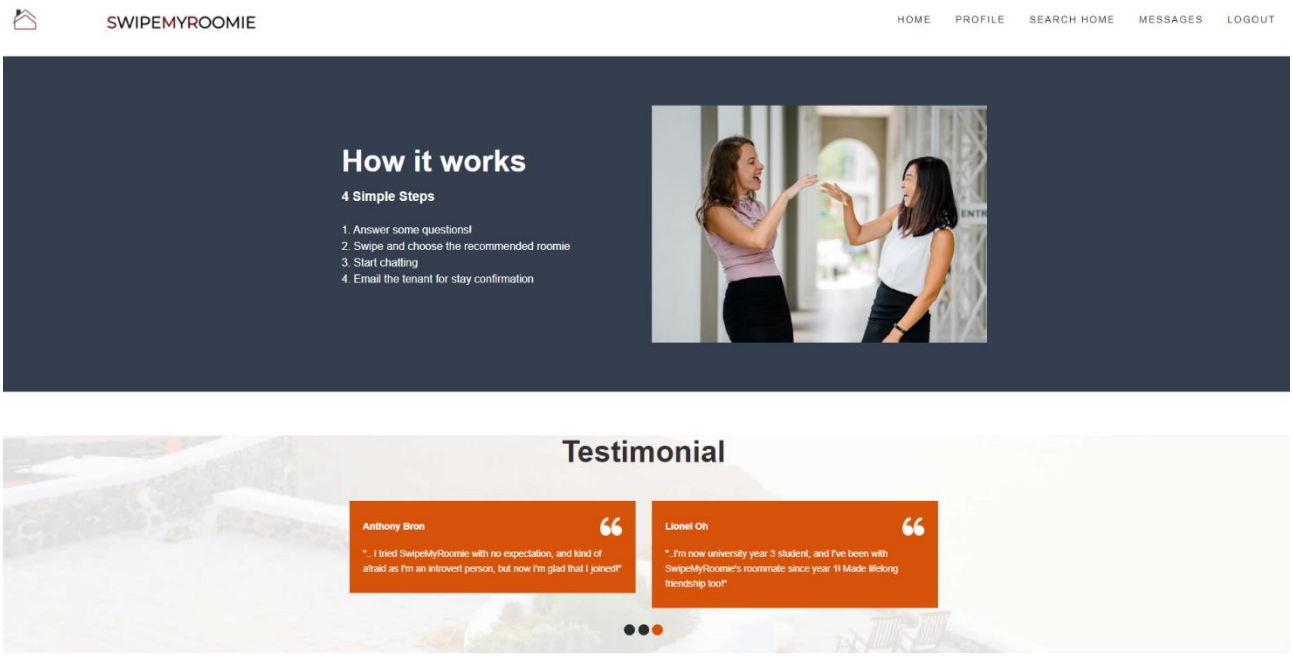


Figure 11: The about page

This page consists of the descriptions of SwipeMyRoomie, with details about how SwipeMyRoomie works, the algorithms used and also some testimonials for past and current users.

Implementation:

From the UI mockup, the working product will be built using the following softwares and programming languages(subjected to changes and will be adjusted according to the needs of the product and the user):

- Front end: HTML5, CSS3, Javascript
- Back-end: mySQL Workbench, Python Flask

3.2 Hardware Interfaces

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>

3.3 Software Interfaces

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>

3.4 Communications Interfaces

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>

4. System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

4.1 Register for Account

4.1.1 Description and Priority
Using this feature, the user should be able to create a new account with a password and an email address which is also used as a login ID. This feature is of high priority as without it, the user is unable to access any other feature of the website.

4.1.2 Stimulus/Response Sequences
1) The user enters an email and password.
2) The system checks with passwordUsername database if there is an existing email address being used.
3) The system then checks if the password fulfils minimal criteria of having at least one upper case character, lower case character and a number.
4) The user then re-enters their desired password into the “Confirm Password”
5) The user then re-enters their desired password into the “Confirm Password” input box.
6) The user clicks “CREATE ACCOUNT” button.
7) The system confirms whether the password in the password and confirm password input box is the same(case sensitive). If it is, continue to next step.
8) The system adds new user record to the passwordUsername database
9) The website prompts success message “Account Registered”.
10) The website redirects users to home page

4.1.3 Functional Requirements
REQ-1: The system must be able to check if the password fulfills the pre-defined criteria
REQ-2: The system must be able to save the registered email ID and password, and it must have the ability to retrieve the information later.
REQ-3: The system must be able to send out a success message
REQ-4: The system must be able to alert the user if any part of the sequence is incorrect, with an appropriate message
REQ-5: The system must be able to redirect the user to the home page after successful completion.

4.2 Login to Profile

4.2.1 Description and Priority
The user must be able to successfully login to the website with their pre-existing account, once it is created. This feature is of high priority as the other features are inaccessible until the user is successfully logged in.

4.2.2 Stimulus/Response Sequences

1) The user clicks on Log In button on main page which redirects them to input their Username and Password.
2) The user enters their Username
3) The user enters their Password.
4) The user clicks the log in button.
5) The system will search the username in the database. The username is case sensitive.
6) If the username is found, the system will check if the corresponding password saved in the database matches with the entered password.
7) If the password matches, the user is logged into the website.
8) The website redirects users to home page

4.2.3 Functional Requirements
REQ-1: The system must be able to alert the user appropriately if the account that they attempted to login with does not exist
REQ-2: The system must be able to alert the user if the password is not correct
REQ-3: The system must be able to recognise when the user ID and password exist, and are correctly matched, it must then login to the website successfully.

4.3 View and Update Profile

4.3.1 Description and Priority
User must be able to view his/her profile including his name, gender, age, occupation, ethnicity, and preferred spoken language. Additionally, the user must be able to update their profile information. This feature is of high priority.

4.3.2 Stimulus/Response Sequences

1) The user clicks their profile icon
2) The system gathers the user's information from the database
3) The user is redirected to another page.
4) The system then displays the user's information on one page for the user to scroll through.
5) The user can now view their profile by scrolling through the page.
6) When the user clicks the update button, they can update their name, gender, age, occupation, ethnicity and preferred spoken language.
7) The user can click on "save changes".
8) The system updates the database.

4.3.3 Functional Requirements
REQ-1: The system must be able to retrieve and display user information according to the profile being accessed
REQ-2: The system must be able to allow edits to the profile information, which can then be saved and retrieved when next accessed.

4.4 Find Roommates

4.4.1 Description and Priority
The system should be able to match user profiles to other user profiles to give each user a list of recommended roommates. They should be matched according to the roommate preferences on their own page and the user information on other users' pages. This feature is of high priority.

4.4.2 Stimulus/Response Sequences
1) The user fills in their information and saves it
2) The system finds profiles which match the criteria corresponding to the user's roommate preferences
3) The system displays these profiles
4) The user can then click through them and pick their preferred roommates

4.4.3 Functional Requirements
REQ-1: The system should be able to identify users which match the roommate preferences on each profile
REQ-2: The system should be able to display the most relevant users, and must provide an easy interface for users to identify potential roommates and contact them

4.5 Search for Apartment

4.5.1 Description and Priority
Users should be able to search for apartments by location. The system must output the available units within a 5km radius of the location searched for by the user. This feature is of high priority.

4.5.2 Stimulus/Response Sequences
1) The user inputs their location (e.g., Punggol/Sengkang)
2) The system searches the API database to find all the houses that are available and within 5 km.
3) The system orders the houses based on distance (in ascending order)
4) The system searches the database for the nearest facilities
5) The system will find a map for the 5km radius.
6) The system prints the houses on the map using pins.
7) The system prints the nearest facilities on the map using symbols.

4.5.3 Functional Requirements
REQ-1: The system must be able to reference with the existing database of available units according to the location input by the user and find the appropriate units.
REQ-2: The system must be able to display appropriate units in a format that allows users to find places easily according to location.
REQ-3: The system must be able to display the units on a map using pins, and display nearest facilities using symbols.

4.6 ChatBox Feature

4.6.1 Description and Priority
The user should be able to chat directly with other users on the website. This will enable users to personally determine if the profiles recommended to them qualify as potential roommate candidates. This feature is of high priority.

4.6.2 Stimulus/Response Sequences
1) The User clicks on the “ChatBox” button.
2) The system launches a new chat with a new tenant on the panel at the left.
3) User types in a message.
4) User clicks on the send button.
5) System displays the message on the sender’s screen as sent.
6) System displays the message on the receiver’s profile as a new message.
7) The receiver can now choose to respond, the flow of events is repeated.

4.6.3 Functional Requirements
REQ-1: The system should enable users to send and receive messages
REQ-2: The system should be able to save and retrieve sent messages and chat histories

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- The website must be available if there is internet connection.
- The website shall take within 3 seconds to load.
- The website must display error should there be a failure in internet connection.
- The website must be fully functional with no malfunctioning features
- When the website needs to be updated, the website should enable modular repair such that users are still able to use the website during downtime.
- The website should load every page within 5 seconds
- The system should be able to respond to potential internal or external changes affecting its delivery in a timely and cost-effective manner.
- The website must be able to be maintained easily after it is deployed- this includes easy repair in the event of bugs.
- All exception handling messages shall load within 0.75 seconds.
- Error messages must be displayed in English

5.2 Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product's design or use. Define any safety certifications that must be satisfied.>

- The users must be above the legal age of 18 years to avail use of the website to prevent children from interacting with strangers online.

5.3 Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

- The website must conceal the password while typing it.
- System shall prevent unauthorized access to its database by password protection.
- The website must ensure the integrity of the user account information.
- The server must not return a restricted web page to any browser that it cannot authenticate.

5.4 Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

- The database system must be robust to maintain a database of user passwords, profile, and usernames.
- Database query must be done with standard SQL queries
- The website must display accurate information. The displayed data must be correct.
- The website should be available to the users any time of the day so that they can use it
- according to their ease.

5.6 Business Rules

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>

- N/A

6. Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

- Database requirements (TBC)

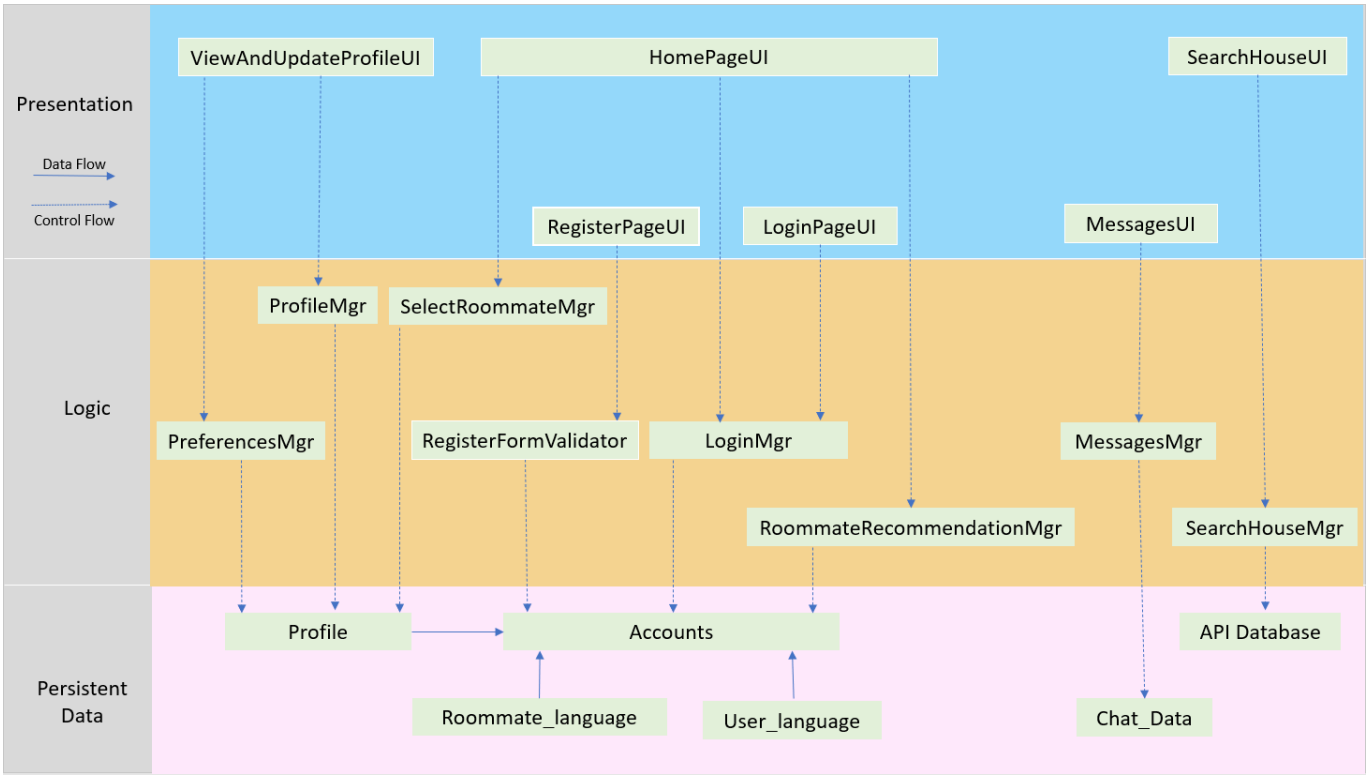
Appendix A: Glossary

Term	Definition
Product functions	The functional requirements of the product to be designed.
System functions	A system outcome which contributes to the goals and objectives of the program.
User Requirements	User requirements describe the business needs for what users require from the system. They are written early in the validation process, typically before the system is created.
Use Case	A written description of how users will perform tasks on the website.
Test Case	Specification of the inputs, execution conditions, testing procedures and expected results.
Front-End	Refers to the user interface, i.e., the graphical user interface of the system that makes it easier to use.
Back-End	The back end refers to parts of a computer application or a program's code that allow it to operate and that cannot be accessed by a user. This can include the server, application and the database.
Device	Device refers to something made or adapted for a particular purpose. In context, it refers to the website.
Bugs	An error, flaw or fault in a computer program or system that produces an incorrect or unexpected result.
HDB Flat	Public housing in Singapore managed by the Housing Development Board(HDB).
University Students	People who have tight budgets due to existing cost of education. Additionally, they may not be guaranteed stay at their university campuses through their years of education in that university. Hence, they would need a home close enough to their university.
Young Adults	People who are just entered the workforce and may not earn enough to purchase a house. Additionally, they are below 35 years of age making them ineligible to purchase a HDB.
Service Costs	The costs involved in finding and employing a property agent or searching for a rental unit without the help of the consolidated approach supported by this website.
Response time	The amount of time taken by the website to respond to user commands.
Speed	The amount of time taken by the system to execute instructions- this may include the time taken to fetch data, display results etc.
Runtime	Runtime or execution time is the final phase of a computer program's life cycle, in which the code is being executed. In other words, it is the running phase of a program.
Minimal downtime	Downtime refers to periods of time during which a computer system, server or network is shut off or unavailable for use.
Server redundancy	A redundant server is one that is kept offline. That is, it powers on with network/Internet connectivity but is not used as a live server. This server can be used to back up data, and can be used in the case of a fault or failure in the server that is normally used.
Backup	A backup is a copy of data or code taken and stored elsewhere so that it may be used to restore the original in the event of data loss.

HDB rental services	The service whereby residents of Singapore are able to rent (i.e., pay on a monthly or yearly basis to live in) housing units built by Singapore's Housing Development Board.
Roommates	A roommate is a person with whom one shares a living facility such as a room or flat.
Apartment	A suite of rooms forming one residence; a flat.
Immigrants	People who are new or temporary residents in Singapore, who do not have an established place of residence.
Condominium	A building or complex of buildings containing several individually owned apartments or houses- the property may have been built by the government or privately built by contractors. Condominiums in Singapore often include shared facilities such as gyms, swimming pools etc.
Private housing businesses	Businesses that are privately owned, which allow people to rent living spaces such as apartments, condos, and landed properties. Private Housing means neither the federal, state or local government subsidizes the rent.

Appendix B: Analysis Model

System Architecture



Appendix C: Reference

Source: http://www.frontiernet.net/~kwiegers/process_assets/srs_template.doc

Appendix D: Data Dictionary

No.	Name	Title	Type	Description
1	blk_no	Block Number	Text (general)	The block number is specified in this field
2	street	Street	Text (general)	The street no./name is specified in this field, which aids in identifying the location of the residential property.
3	max_floor_lvl	Maximum floor level	Numeric (general)	This field specifies the maximum number of floors in the building.
4	year_completed	Year completed	Datetime (Year) "YYYY"	This field specifies the year in which the property was built and completed. It aids in distinguishing between the older and newer models of residential units.
5	residential	Residential property tag	Text (general)	This tag is activated if the corresponding property is a residential property.
6	commercial	Commercial property tag	Text (general)	This tag is activated if the corresponding property is a commercial property.
7	market_hawker	Market and hawker tag	Text (general)	This tag details if there are markets and hawker centers in close proximity to the property.
8	miscellaneous	Miscellaneous	Text (general)	This field includes miscellaneous information about the property. Examples include admin office, childcare centre, education centre, Residents' Committees centre.
9	multistorey_carpark	Multi-Storey carpark tag	Text (general)	This tag is activated if the residents of the corresponding property have access to a multi-storey carpark.
10	precinct_pavilion	Precinct pavilion tag	Text (general)	This tag is activated if the property has an associated precinct pavilion.
11	bldg_contract_town	Town	Text (general)	This field specifies the town in which the property is located. Legend:

				AMK - ANG MO KIO BB - BUKIT BATOK BD - BEDOK BH - BISHAN BM - BUKIT MERAH BP - BUKIT PANJANG BT - BUKIT TIMAH CCK - CHOA CHU KANG CL - CLEMENTI CT - CENTRAL AREA GL - GEYLANG HG - HOUGANG JE - JURONG EAST JW - JURONG WEST KWN - KALLANG/WHAMPOA MP - MARINE PARADE PG - PUNGGOL PRC - PASIR RIS QT - QUEENSTOWN SB - SEMBAWANG SGN - SERANGOON SK - SENGKANG TAP - TAMPINES TG - TENGAH TP - TOA PAYOH WL - WOODLANDS YS - YISHUN
12	total_dwelling_units	Total dwelling units	Numeric (general)	This field specifies the total number of residential units available within the property.
13	1room_sold	Number of 1-room sold flats	Numeric (general)	This field specifies the total number of 1 room flats sold within the property.
14	2room_sold	Number of 2-room sold flats	Numeric (general)	This field specifies the total number of 2 room flats sold within the property. 2 room flat is a flat with 1 bedroom and attached bathroom. (~36-45 sq m).
15	3room_sold	Number of 3-room sold flats	Numeric (general)	This field specifies the total number of 3 room flat sold within the property. 3 room flat is a flat with 2 bedrooms, 1 of which is a

				master bedroom with attached bathroom. (~60-65 sq m).
16	4room_sold	Number of 4-room sold flats	Numeric (general)	This field specifies the total number of 4 room flats sold within the property. 4 room flat is a flat with 3 bedrooms, 1 of which is a master bedroom with attached bathroom. (~90 sq m).
17	5room_sold	Number of 5-room sold flats	Numeric (general)	This field specifies the total number of flats with 5 rooms sold within the property. 5 room flats have 3 bedrooms, one of which is a master bedroom with attached bathroom (~110 sq m).
18	exec_sold	Number of Executive sold flats	Numeric (general)	This field specifies the total number of Executive flats sold within the property. Executive flats feature an additional space for a study room. It consists of 1 master bedroom with attached bathroom and 2 additional bedrooms. (~130 sq m)
19	multigen_sold	Number of Multigeneration sold flats	Numeric (general)	This field specifies the total number of Multi-generation flats sold within the property. These flats are designed to meet the needs of multi-generation families It consists of 4 bedrooms, 2 of which have attached bathrooms. (~115 sq m). A multi-generation family is any of the following: 1.Married/ engaged couple and parents 2.Widowed/ divorced with a child and parents

20	studio_apartment_sold	Number of Studio apartment sold flats	Numeric (general)	This field specifies the total number of studio apartments sold within the property.
21	1room_rental	Number of 1-room rental flats	Numeric (general)	This field specifies the number of 1 room flats rented out within the property.
22	2room_rental	Number of 2-room rental flats	Numeric (general)	This field specifies the number of 2 room flats rented out within the property. 2 room flat is a flat with 1 bedroom and attached bathroom. (~36-45 sq m).
23	3room_rental	Number of 3-room rental flats	Numeric (general)	This field specifies the number of 3 room flats rented out within the property. A 3 room flat is a flat with 2 bedrooms, 1 of which is a master bedroom with attached bathroom. (~60-65 sq m).
24	other_room_rental	Number of Other room rental flats	Numeric (general)	This field specifies the number of rented flats within the property of types other than 1-room, 2-room, and 3-room.