
Software Requirements Specification

for

SimplyStay

Version 1.0 approved

Prepared by JENTS

SC2006 Software Engineering

19/04/2024

Table of Contents

1. Introduction	1
1.1 Purpose	1
1.2 Document Conventions	1
1.3 Intended Audience and Reading Suggestions	1
1.4 Product Scope	2
1.5 References	2
2. Overall Description	3
2.1 Product Perspective.....	3
2.2 Product Functions.....	3
2.2.1 Login/Signup Page	4
2.2.2 Search Page	4
2.2.3 Eligibility Calculator Page	4
2.2.4 Housing Loan Calculator Page	4
2.2.5 Favourites Page.....	4
2.2.6 Comparison Page	5
2.3 User Classes and Characteristics	5
2.3.1 Users eligible to purchase a BTO flat.....	5
2.3.2 Users ineligible to purchase a BTO flat	5
2.4 Operating Environment.....	6
2.4.1 Production Environment.....	6
2.4.2 Development Environment.....	7
2.5 Design and Implementation Constraints	7
2.6 User Documentation	9
2.7 Assumptions and Dependencies	9
3. External Interface Requirements	11
3.1 User Interfaces	11
3.1.1 Main Page	11
3.1.2 Login and Signup Page.....	12
3.1.3 Search Page	13
3.1.4 Eligibility Calculator Page	14
3.1.5 Housing Loan Calculator Page	15
3.1.6 Comparison Page	16
3.1.7 Favourites Page.....	17
3.2 Hardware Interfaces	18
3.3 Software Interfaces.....	18
3.3.1 Software Versions and Components.....	18
3.3.2 Software Architecture.....	18
4. System Features	19
4.1 Registration	19
4.1.1 Description and Priority	19
4.1.2 Stimulus/Response Sequences	19
4.1.3 Functional Requirements	22
4.2 Login	24
4.2.1 Description and Priority	24
4.2.2 Stimulus/Response Sequences	24
4.2.3 Functional Requirements	26
4.3 Searching by Price	28
4.3.1 Description and Priority	28
4.3.2 Stimulus/Response Sequences	28
4.3.3 Functional Requirements	30
4.4 Searching by Geographical Area	31

4.4.1	Description and Priority	31
4.4.2	Stimulus/Response Sequences	31
4.4.3	Functional Requirements	33
4.5	Searching by Housing Type	34
4.5.1	Description and Priority	34
4.5.2	Stimulus/Response Sequences	34
4.5.3	Functional Requirements	36
4.6	Searching by Multiple Filters	38
4.6.1	Description and Priority	38
4.6.2	Stimulus/Response Sequences	38
4.6.3	Functional Requirements	40
4.7	Favourite Houses	42
4.7.1	Description and Priority	42
4.7.2	Stimulus/Response Sequences	42
4.7.3	Functional Requirements	44
4.8	Eligibility Calculator	45
4.8.1	Description and Priority	45
4.8.2	Stimulus/Response Sequences	45
4.8.3	Functional Requirements	47
4.9	Housing Loan Calculator	49
4.9.1	Description and Priority	49
4.9.2	Stimulus/Response Sequences	49
4.9.3	Functional Requirements	51
4.10	Comparison	53
4.10.1	Description and Priority	53
4.10.2	Stimulus/Response Sequences	53
4.10.3	Functional Requirements	55
4.11	Maps	56
4.11.1	Description and Priority	56
4.11.2	Stimulus/Response Sequences	56
4.11.3	Functional Requirements	57
4.12	Logout	58
4.12.1	Description and Priority	58
4.12.2	Stimulus/Response Sequences	58
4.12.3	Functional Requirements	59
5.	Other Nonfunctional Requirements	60
5.1	Performance Requirements	60
5.2	Security Requirements	60
5.3	Software Quality Attributes	60
5.3.1	Maintainability	60
5.3.2	Scalability	60
5.3.3	Usability	60
5.3.4	Availability	61
5.3.5	Reliability	61
Appendix A: Data Dictionary		62
Appendix B: Analysis Models		64
B1:	Use Case Diagram	64
B2:	Class Diagram	65
B3:	Dialog Map	66
B4:	Sequence Diagrams	67
B4.1:	Eligibility Calculator	67
B4.2:	Compare Houses	68
B4.3:	Favourites	69
B4.4:	Housing Loan Calculator	70

B4.5: Login	71
B4.6: Search Houses	72
B5: Key Boundary and Entity Classes	73
B6: Architecture Diagram.....	74
B7: Test Cases and Test Results.....	75
B7.1: Black Box Testing	75
B7.2: White Box Testing	77
Appendix C: Design Patterns and Principles Used.....	79
C1: Design Patterns.....	79
C1.1: Observer Pattern	79
C1.2: Façade Pattern	79
C2: Design Principles	79
C2.1: Single Responsibility Principle	79
C2.2: Low Coupling and High Cohesion	80
Appendix D: Software Engineering Practices	81
D1: Clear Requirement Elicitation	81
D2: Reuse-oriented Software Engineering.....	82
D3: Agile Manifesto.....	82
D4: Version Control	83

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to outline the requirements specifications of the SimplyStay web application system. It will include the functional and non-functional requirements, its user interfaces, use case descriptions, as well as its constraints and dependencies.

1.2 Document Conventions

Font	Arial
Line Spacing	1.5
Text Alignment	Full Justification
Heading 1	Bold, size 18
Heading 2	Bold, size 16
Heading 3	Bold, size 14
Content	Size 12
Technical Standards	IEEE 830-1998

Please refer to Appendix A (Data Dictionary) for a list of definitions and special terms used in this document.

1.3 Intended Audience and Reading Suggestions

This document is intended to be read by the SimplyStay development team and SimplyStay users. However, external audiences interested in the SimplyStay website are welcome to read it as well.

This document starts with **Section 1: Introduction**, outlining the purpose and scope of the website. Next, **Section 2: Overall Description** gives an overall description of the website, explaining its functions and constraints, as well as its assumptions and dependencies. After that, **Section 3: External Interface Requirements** elucidates the user interfaces of the website. This is followed by **Section 4: System Features** and **Section 5: Other Nonfunctional Requirements** highlighting the functional and non-functional requirements and its corresponding use case descriptions. To conclude, **Appendix A** details the Data Dictionary, **Appendix B** presents all the analysis models used such as sequence and class diagrams, **Appendix C** covers the design patterns and principles used, and **Appendix D** describes the software engineering practices used.

All audiences are encouraged to read Section 1 to gain a brief overview of the website and the SRS document, as well as Appendix A to familiarise themselves with the terminology used in the document. The SimplyStay development team are highly encouraged to read the entirety of this document in sequential order. SimplyStay users may read Section 3 to gain a better understanding of the website's user interface.

1.4 Product Scope

In Singapore, buying public housing for the first time can be a daunting task for young adults. As such, we have elected to create a convenient and intuitive website, called SimplyStay, to help young Singaporeans ease the process of finding their dream home.

Our target demographic are young Singaporeans looking to purchase public housing in Singapore. While we are specifically targeting age groups of 20-35, Singaporeans of any age are welcome to use the website as well.

1.5 References

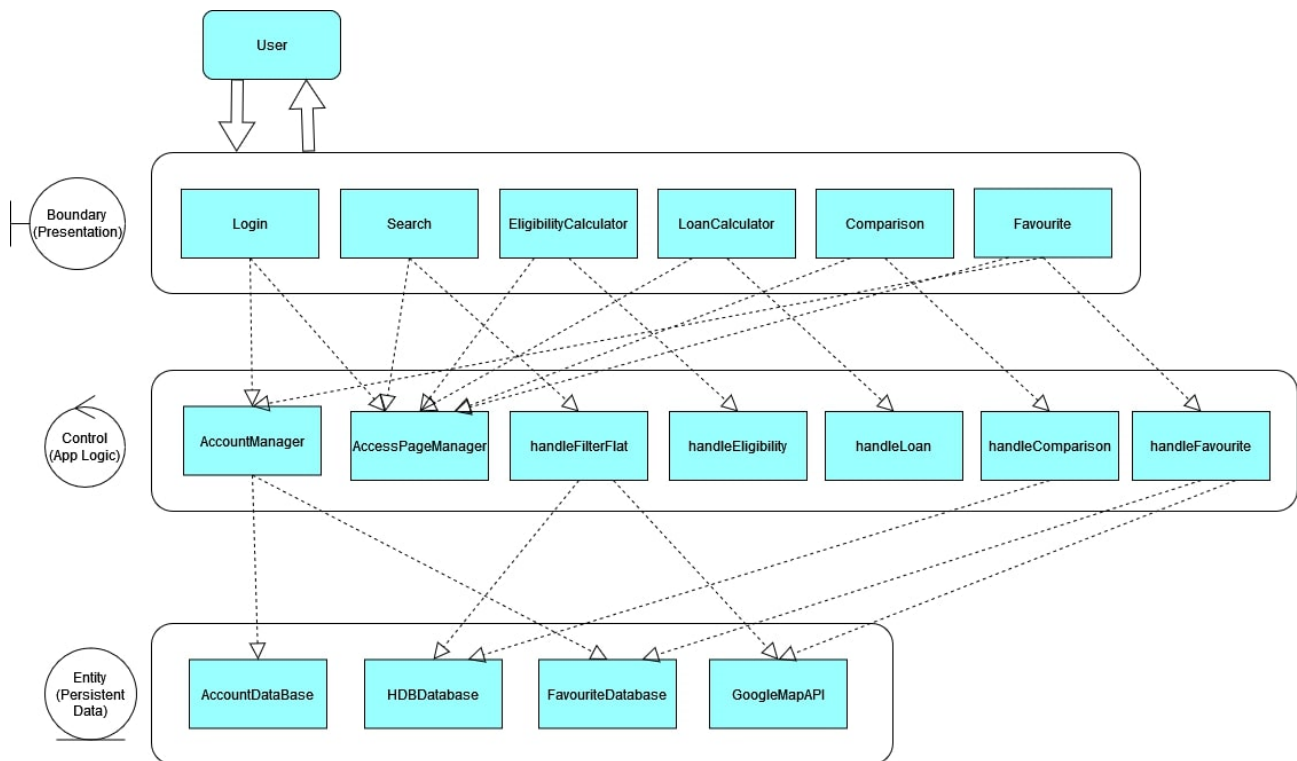
- "IEEE Guide for Software Requirements Specifications," in IEEE Std 830-1984, vol., no., pp.1-26, 10 Feb. 1984, doi: 10.1109/IEEESTD.1984.119205.
- Software Requirements by Karl Wieggers and Joy Beatty

2. Overall Description

2.1 Product Perspective

The SimplyStay website is a new, standalone, and self-contained web application, and is not part of a larger product family or product line. It is inspired by other websites that aggregate car prices for Singaporeans, and we aspire to not only achieve that functionality for BTO flats, but add value by aggregating other types of information pertinent to housing, such as the distance to public transport infrastructure.

An overall system diagram depicting the operation of the application is shown below:



2.2 Product Functions

The functions are summarised with respect to the website's different pages.

2.2.1 Login/Signup Page

- Users can register for an account using their email or Google account and a password of their choice.
- Users can login using their login credentials

2.2.2 Search Page

- Users can search for houses by different filters such as price, location, and housing type
- The page will return all the houses that satisfy the user's chosen filters
- Logged in users can choose to favourite the resulting houses

2.2.3 Eligibility Calculator Page

- Users can input their age, marital status, gross monthly income, monthly expenditure, CPF, and savings into the eligibility calculator.
- The page will recommend eligibility for a BTO, and payment plans based on the user's input

2.2.4 Housing Loan Calculator Page

- Users can input their desired house price, loan amount, interest rate, and loan tenure into the housing loan calculator.
- The page will calculate the monthly installments and the total interest paid based on the user's input

2.2.5 Favourites Page

- Logged in users can bookmark and remove their favourite houses
- Guest users will not be able to access this page, and will be notified by the website accordingly

2.2.6 Comparison Page

- Users can select two houses from the list of houses available
- The page will display the statistics of the selected houses and highlight superior and inferior statistics in green and red respectively

2.3 User Classes and Characteristics

The user classes are ranked in order of importance.

2.3.1 Users eligible to purchase a BTO flat

Aspect	Description
Frequency of Use	High
Subset of Product Functions Used	All
Technical Expertise	High
Characteristics	<p>These will be people who are aged 18 to 35 and are married, or unmarried people aged 35 and above.</p> <p>As the primary target group, they will use the website regularly and register an account on the website.</p> <p>Their technical expertise is expected to be high, that they can comfortably navigate modern websites on their desktops and know what details to look out for in purchasing a BTO flat.</p>

2.3.2 Users ineligible to purchase a BTO flat

Aspect	Description
Frequency of Use	Low
Subset of Product Functions Used	Search Page Comparison Page
Technical Expertise	Low
Characteristics	<p>These will be people who do not fit the requirements to purchase a BTO flat or are current homeowners curious about the housing market.</p> <p>They may visit the website occasionally from time to time and will likely be guest users.</p> <p>Their technical expertise may vary, although it is expected that they can comfortably navigate modern websites on their desktops.</p>

2.4 Operating Environment

2.4.1 Production Environment

Operating System	Microsoft Windows 7.0 and above macOS Sierra and above
Web Browsers	Google Chrome, Mozilla Firefox, Microsoft Edge, browsers that support HTML5, CSS3, and Javascript

2.4.2 Development Environment

Development environment	Description
Front-end: React.js	React is a free and open-source front-end JavaScript library for building user interfaces based on components.
User Interface Library: Ant Design	An enterprise-class UI design language and React UI library with a set of high-quality React components.
Database: Firebase	Firebase is a set of backend cloud computing services and application development platforms provided by Google. It hosts databases, services, authentication, and integration for a variety of applications, including Android, iOS, JavaScript, Node.js, Java, Unity, PHP, and C++.
Hosting service: Vercel	Vercel is a cloud platform for static sites and serverless functions. It allows developers to build and deploy web projects with ease.

2.5 Design and Implementation Constraints

Regulatory Policies	Details
General Data Protection Regulation (GDPR)	The website should adhere to GDPR and other relevant data protection regulations for user data
API Limitations	Details

Google Maps	<p>The developers must have a usable API key for the Google Maps API to function properly.</p> <p>As the website depends on the Google Maps API to display the location of the houses, it is subject to the availability of the API.</p>
Firebase (Firestore database)	<p>The developers must have a usable API key for the Firestore database to function properly.</p> <p>As the website depends on the Firestore database to store the information of the flats and authentication of its users, it is subject to the availability of the database.</p> <p>Firebase adopts a “pay-as-you-go” payment plan. In the scenario that the database receives more than 20K writes or 50K reads per day, the developers will be required to pay \$0.0369 per 100K documents and \$0.1107 per 100K documents respectively, according to “Singapore (asia-southeast1)” pricing.</p>

Design Conventions	Details
User Interface Standards	The website’s UI is implemented using Ant Design. As such, developers should adhere to Ant Design’s design patterns and values

	to maintain consistency in the website's interface.
Programming Standards	<p>Each React component must have its respective folder of the same name.</p> <p>Code must be properly indented with an indentation size of 4 spaces.</p> <p>Nested code must be avoided wherever possible. However, if it is unavoidable, a maximum of 3 nested statements must be met.</p> <p>Each component must have proper documentation and comments explaining the rationale and functions of the code.</p>

2.6 User Documentation

For the SimplyStay development team or any developers interested in maintaining and continuing the project, there will be a README.md file to guide them in setting up the development environment for the website.

<https://github.com/softwarelab3/2006-MACS2-JENTS/blob/main/README.md>

2.7 Assumptions and Dependencies

Assumptions:

- The user is connected to the Internet, as an Internet connection is required to access Firebase.

- The user is viewing the website on a desktop. The website is not optimised for mobile devices.
- Future versions of React, Ant Design, and Firebase will have backwards compatibility.

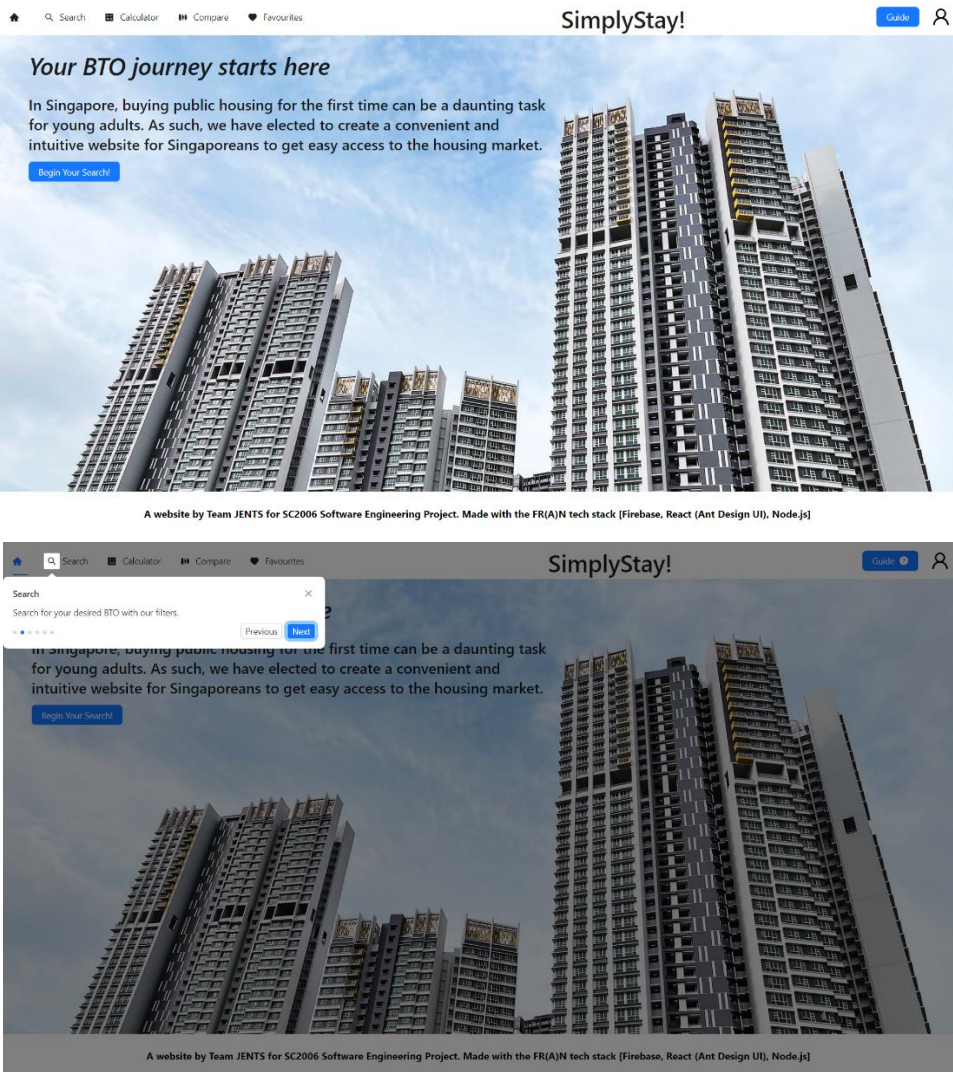
Dependencies:

- The website relies on the Google Maps API to display the location of the houses. Any changes to the API may result in abnormal behaviour of the website. Any downtime will cause the website to lose some functionality.
- The website depends on the Firebase database to store the information of the flats and authenticate its users. Any changes to the API will result in abnormal behaviour or a potential runtime error of the website. Any downtime will adversely affect the website and cause it to lose most of its functionality.

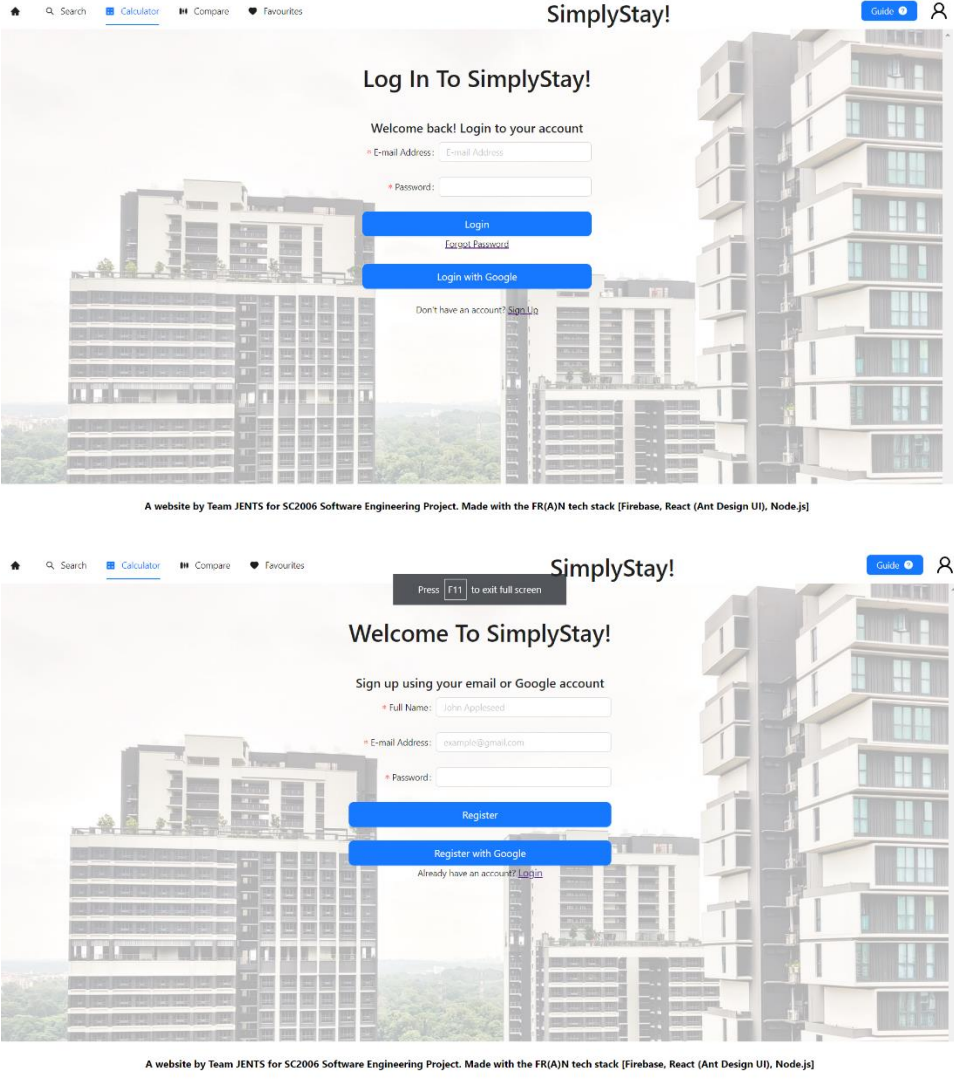
3. External Interface Requirements

3.1 User Interfaces

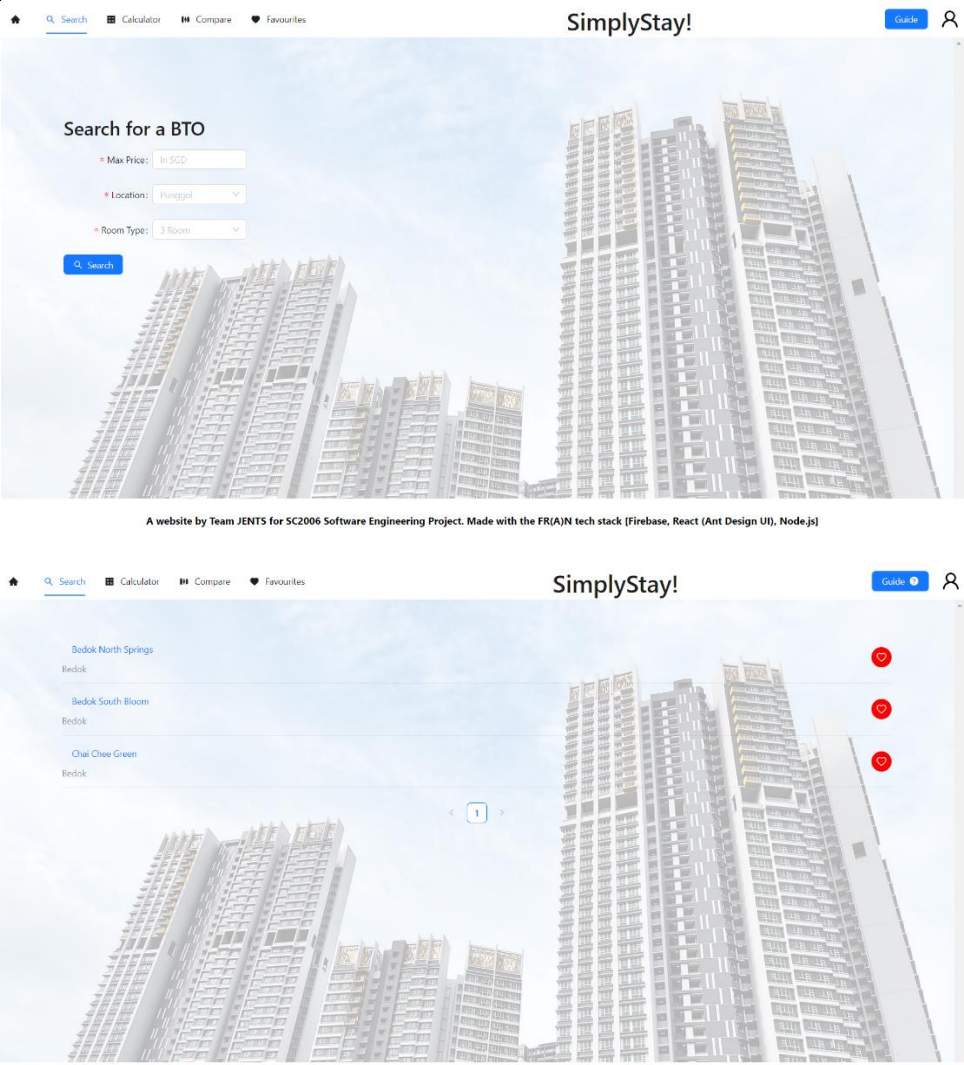
3.1.1 Main Page

Image	Description
	<p>This will be the page that greets users when entering the website for the first time. They will be able to navigate to the various pages on the website. A guide button will give the user a tour of the website and its functionalities.</p>

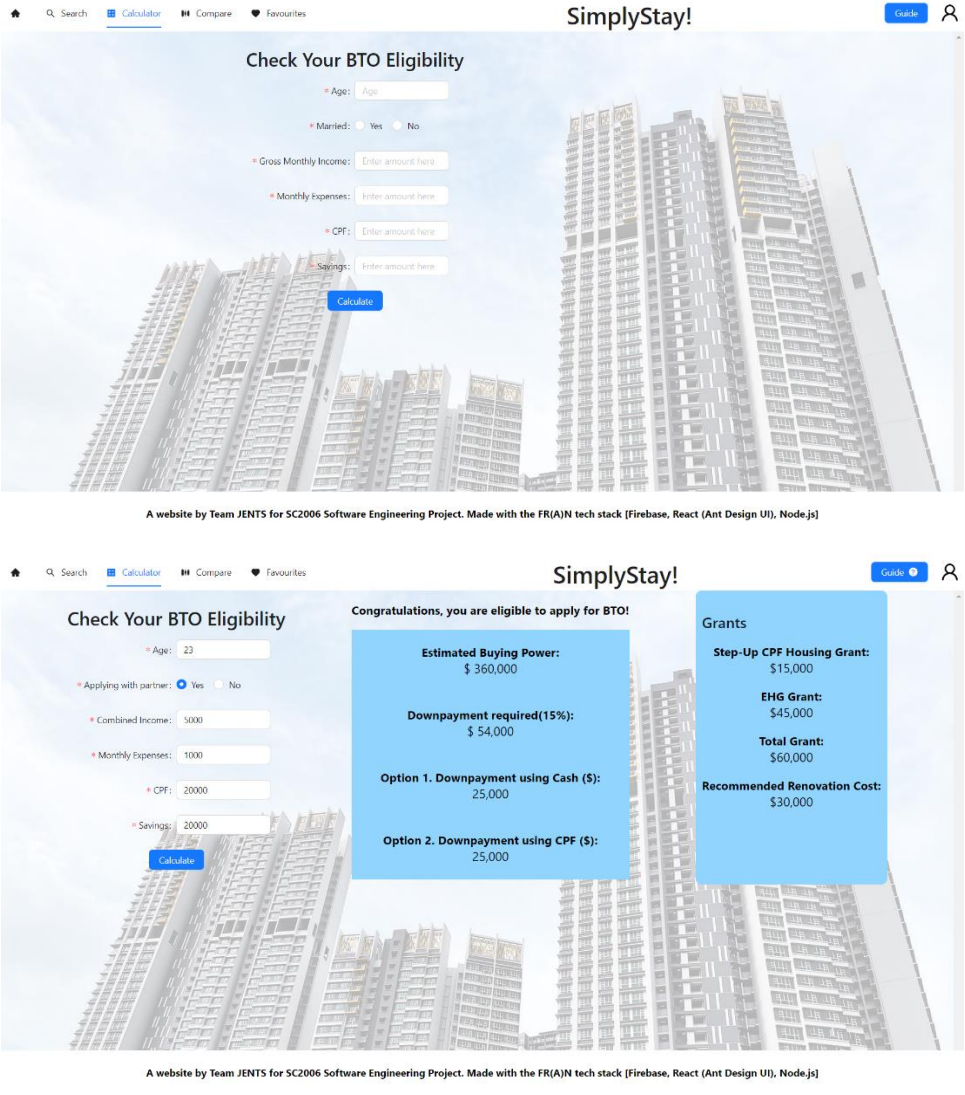
3.1.2 Login and Signup Page

Image	Description
 <p>The image displays two screenshots of the SimplyStay! web application. The top screenshot shows the login page with the heading "Log In To SimplyStay!". It includes a welcome message "Welcome back! Login to your account", input fields for "E-mail Address" and "Password", and buttons for "Login", "Forgot Password", and "Login with Google". A link "Don't have an account? Sign Up" is also present. The bottom screenshot shows the signup page with the heading "Welcome To SimplyStay!". It includes a message "Sign up using your email or Google account", input fields for "Full Name", "E-mail Address", and "Password", and buttons for "Register" and "Register with Google". A link "Already have an account? Login" is also present. Both pages feature a background image of modern apartment buildings and a footer stating "A website by Team JENTS for SC2006 Software Engineering Project. Made with the FR(A)N tech stack [Firebase, React (Ant Design UI), Node.js]".</p>	<p>Users can log in using their email or their Google account. If they do not have an account, they can choose to register for one.</p>

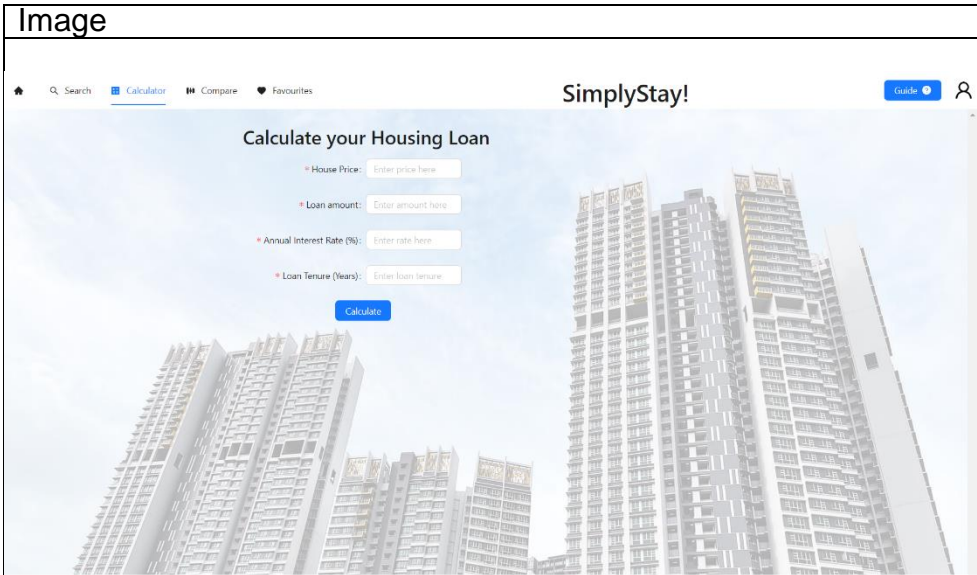
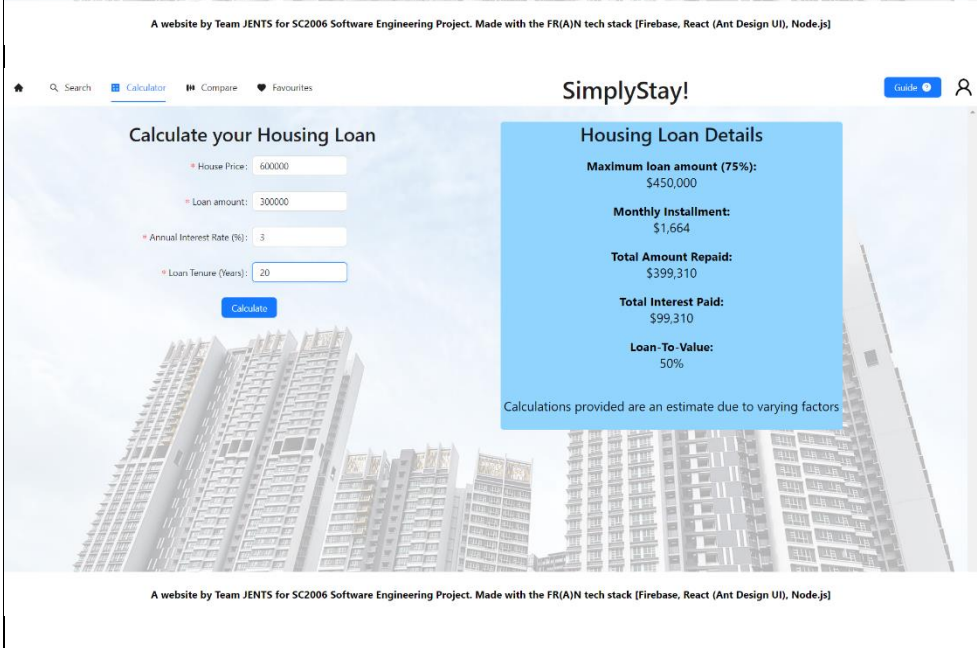
3.1.3 Search Page

Image	Description
 <p>The screenshot displays the 'SimplyStay!' web application interface. At the top, there is a navigation bar with links for 'Search', 'Calculator', 'Compare', and 'Favourites', along with a 'Guide' button and a user profile icon. The main section is titled 'Search for a BTO' and features three input fields: 'Max Price' (set to 'In SGD'), 'Location' (set to 'Punggol'), and 'Room Type' (set to '3 Room'). A blue 'Search' button is positioned below these fields. The background of the page shows a 3D rendering of modern high-rise apartment buildings. At the bottom, a footer note states: 'A website by Team JENTS for SC2006 Software Engineering Project. Made with the FR(A)N tech stack (Firebase, React (Ant Design UI), Node.js)'.</p>	<p>Users can search for BTO flats using the following filters: Maximum Price, Minimum Price, Location, and Housing Type. If the search is successful, it will return a list of flats that match their chosen filters.</p>

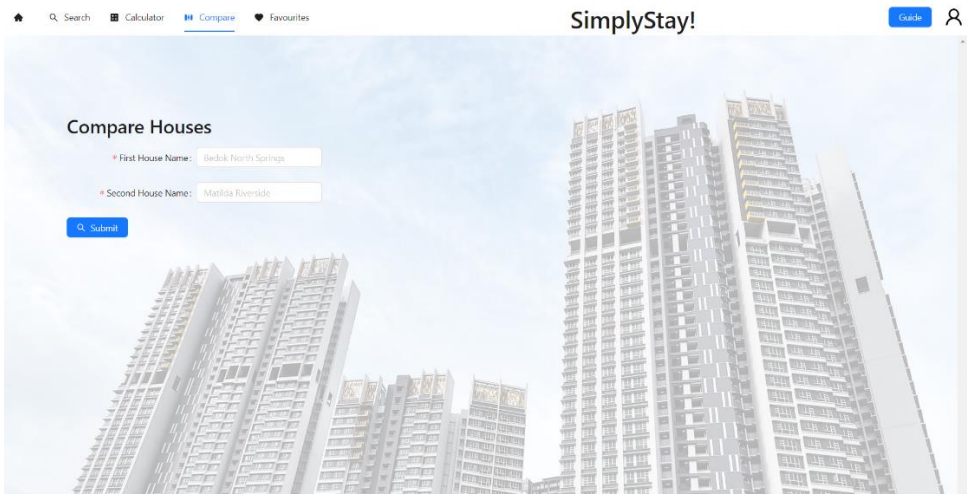
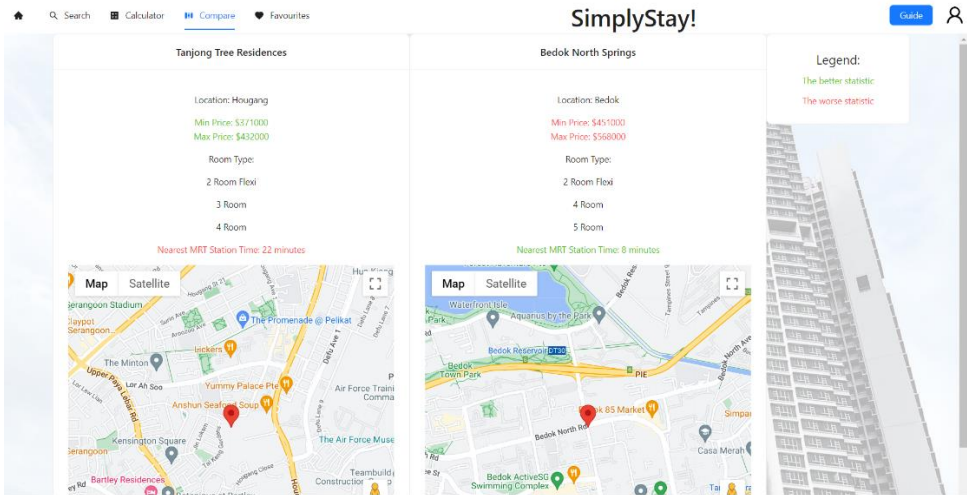
3.1.4 Eligibility Calculator Page

Image	Description
 <p>The image displays two screenshots of the 'SimplyStay!' website's Eligibility Calculator page. The top screenshot shows the input form with fields for Age, Married status, Gross Monthly Income, Monthly Expenses, CPF, and Savings. The bottom screenshot shows the results after calculation, including Estimated Buying Power, Downpayment required, and Grants.</p> <p>Check Your BTO Eligibility</p> <p>Age: <input type="text" value="23"/></p> <p>Married: <input checked="" type="radio"/> Yes <input type="radio"/> No</p> <p>Gross Monthly Income: <input type="text" value="5000"/></p> <p>Monthly Expenses: <input type="text" value="1000"/></p> <p>CPF: <input type="text" value="20000"/></p> <p>Savings: <input type="text" value="20000"/></p> <p>Calculate</p> <p>Congratulations, you are eligible to apply for BTO!</p> <p>Estimated Buying Power: \$ 360,000</p> <p>Downpayment required(15%): \$ 54,000</p> <p>Option 1. Downpayment using Cash (\$): 25,000</p> <p>Option 2. Downpayment using CPF (\$): 25,000</p> <p>Grants</p> <p>Step-Up CPF Housing Grant: \$15,000</p> <p>EHG Grant: \$45,000</p> <p>Total Grant: \$60,000</p> <p>Recommended Renovation Cost: \$30,000</p> <p>A website by Team JENTS for SC2006 Software Engineering Project. Made with the FR(A)N tech stack [Firebase, React (Ant Design UI), Node.js]</p>	<p>Users can input their age, marital status, gross monthly income, monthly expenses, CPF, and savings to calculate their eligibility for a BTO, and if so, the payment plans and grants awarded.</p>

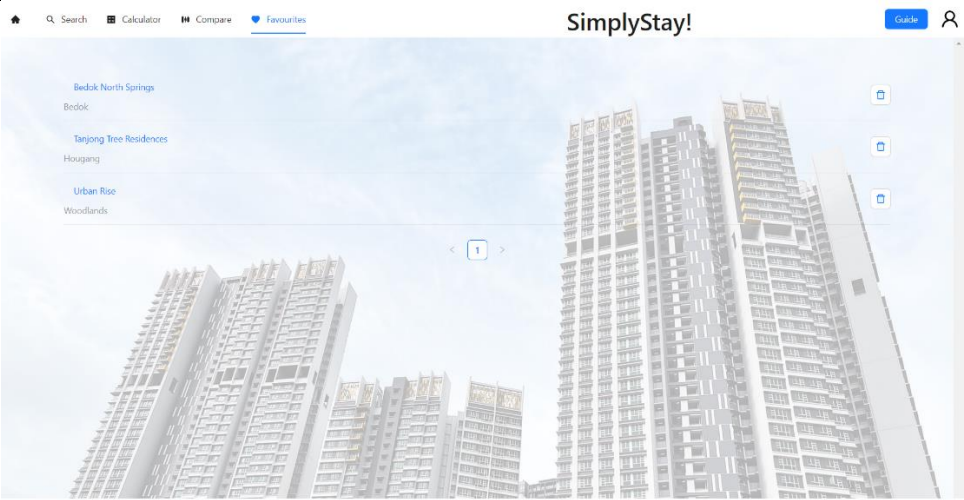
3.1.5 Housing Loan Calculator Page

Image	Description
 <p>The screenshot shows the 'Calculate your Housing Loan' page on the SimplyStay! website. The page features a background image of modern high-rise apartment buildings. At the top, there is a navigation bar with a home icon, a search bar, and links for 'Calculator', 'Compare', and 'Favourites'. The main heading is 'Calculate your Housing Loan'. Below this, there are four input fields: 'House Price' (placeholder: 'Enter price here'), 'Loan amount' (placeholder: 'Enter amount here'), 'Annual Interest Rate (%)' (placeholder: 'Enter rate here'), and 'Loan Tenure (Years)' (placeholder: 'Enter loan tenure'). A blue 'Calculate' button is positioned below the input fields. At the bottom of the page, a footer note reads: 'A website by Team JENTS for SC2006 Software Engineering Project. Made with the FR(A)N tech stack [Firebase, React (Ant Design UI), Node.js]'.</p>	<p>Users can input their desired house's price, their desired loan amount, the interest rate, and the loan tenure. The page will return and show the user the monthly installments to be paid, and the total interest paid.</p>
 <p>This screenshot shows the same 'Calculate your Housing Loan' page, but with the calculated results displayed in a blue box on the right side. The input fields now contain numerical values: House Price: 600000, Loan amount: 300000, Annual Interest Rate (%): 3, and Loan Tenure (Years): 20. The 'Calculate' button is still present. The blue box, titled 'Housing Loan Details', contains the following information: 'Maximum loan amount (75%): \$450,000', 'Monthly Installment: \$1,664', 'Total Amount Repaid: \$399,310', 'Total Interest Paid: \$99,310', and 'Loan-To-Value: 50%'. Below the blue box, a note states: 'Calculations provided are an estimate due to varying factors'. The footer note remains the same: 'A website by Team JENTS for SC2006 Software Engineering Project. Made with the FR(A)N tech stack [Firebase, React (Ant Design UI), Node.js]'.</p>	

3.1.6 Comparison Page

Image	Description
 <p>SimplyStay!</p> <p>Compare Houses</p> <p>First House Name: Bedok North Springs</p> <p>Second House Name: Mallisa Riverside</p> <p>Submit</p> <p>A website by Team JENTS for SC2006 Software Engineering Project. Made with the FR(A)N tech stack (Firebase, React (Ant Design UI), Node.js)</p>	Users can select two houses and compare their statistics. The superior statistics will be in green, and the inferior statistics will be in red.
 <p>SimplyStay!</p> <p>Tanjong Tree Residences</p> <p>Location: Hougang</p> <p>Min Price: \$371000 Max Price: \$432000</p> <p>Room Type:</p> <ul style="list-style-type: none">2 Room Flexi3 Room4 Room <p>Nearest MRT Station Time: 22 minutes</p> <p>Bedok North Springs</p> <p>Location: Bedok</p> <p>Min Price: \$451000 Max Price: \$568000</p> <p>Room Type:</p> <ul style="list-style-type: none">2 Room Flexi4 Room5 Room <p>Nearest MRT Station Time: 8 minutes</p> <p>Legend:</p> <ul style="list-style-type: none">The better statisticThe worse statistic <p>A website by Team JENTS for SC2006 Software Engineering Project. Made with the FR(A)N tech stack (Firebase, React (Ant Design UI), Node.js)</p>	

3.1.7 Favourites Page

Image	Description
 <p>The screenshot displays the 'SimplyStay!' web application's 'Favourites' page. The top navigation bar includes links for 'Search', 'Calculator', 'Compare', and 'Favourites' (which is active). A 'Guide' button and a user profile icon are also present. On the left, a list of bookmarked properties is shown: 'Bedok North Springs', 'Bedok', 'Tanjong Tree Residences', 'Hougang', 'Urban Rise', and 'Woodlands'. The main area features a large 3D architectural rendering of a modern high-rise apartment complex. A small pagination control at the bottom of the image shows '< 1 >'. At the very bottom of the image, a footer note reads: 'A website by Team JENTS for SC2006 Software Engineering Project. Made with the FR(A)N tech stack (Firebase, React (Ant Design UI), Node.js)'.</p>	Logged in users can view their bookmarked houses. They can choose to remove them.

3.2 Hardware Interfaces

All devices supporting browsers specified in **Section 2.4.1: Production Environment** are compatible with the website.

3.3 Software Interfaces

3.3.1 Software Versions and Components

The system's frontend was developed using React v18.2.0.

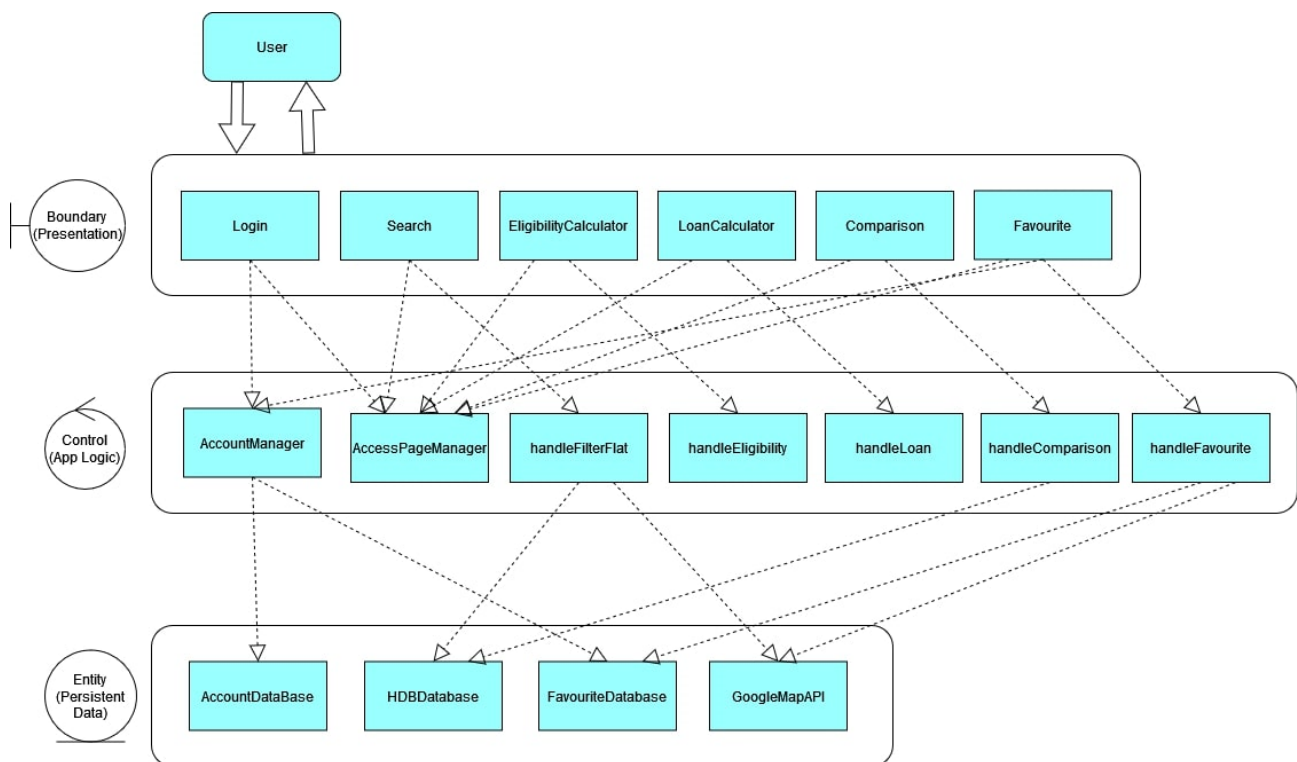
The system's user interface was implemented using Ant Design v5.15.2.

The system uses Firebase v10.10.0 for data storage and user authentication.

The system uses Google Maps API v2.19.3 for display of map data.

3.3.2 Software Architecture

The system uses a three-layered architecture design.



4. System Features

4.1 Registration

4.1.1 Description and Priority

Description	First time users can register for an account
Priority	High

4.1.2 Stimulus/Response Sequences

Use Case ID:	001		
Use Case Name:	Registration		
Created By:	Nigel Paliath	Last Updated By:	Nigel Paliath
Date Created:	5 February 2024	Date Last Updated:	18 April 2024

Actor:	User (Initiating Actor), SimplyStay Website, Database
Preconditions:	<ol style="list-style-type: none">1. The user must be connected to the Internet.2. The user does not have an account prior to registration.3. The user has navigated to the login interface

Postconditions:	<ol style="list-style-type: none">1. The user successfully registered an account with their email and password and their account is added into the system database. <p>OR</p> <ol style="list-style-type: none">2. The user is notified of the reason(s) why the registration was unsuccessful.
Frequency of Use:	Low
Flow of Events:	<ol style="list-style-type: none">1. At the homepage of the website, the user can click onto the “Login” button and the system will redirect the user to the login page.2. Since the user is a new user, he would need to register for a new account.3. The user can click on “Sign Up” and he will be redirected to the registration page.4. The user will need to input their full name.5. The user would then need to input a valid email address.6. The user will need to input a valid password that contains at least 8 characters.7. The user will click on the “Register” button to register their account.

	<p>8. The system validates if the email is valid and if there is an identical email existing in the system.</p> <p>9. The system will verify if the password satisfies all requirements</p> <p>10. Upon verification, the system will store all information in the database securely.</p> <p>11. Once registration is successful, the system will automatically redirect the user to the main page.</p>
Alternative Flows:	<p><u>AF-S5: The user inputs an invalid email address.</u></p> <p>1. The system displays the notification “Invalid email address!”.</p> <p>2. The system returns to Step 3 and waits for inputs from the user.</p> <p><u>AF-S6: The user inputs an invalid password</u></p> <p>1. The system displays the notification “Password must be at least 8 characters long!”.</p> <p>2. The system returns to Step 3 and waits for inputs from the user.</p> <p><u>AF-S8: The user did not complete all the input fields.</u></p> <p>1. Upon clicking on the “Sign up” button, the system will display the notification “Please fill in all the fields!”.</p>

	<p>2. The system will return to Step 3 and wait for user to complete all inputs.</p> <p><u>AF-S9: The user inputs an email address that had already been registered.</u></p> <p>1. The system redirects the user to the main page and displays the message “Registration failed. Please try again.”</p> <p>2. The system returns to Step 1 and waits for input from the user.</p>
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

4.1.3 Functional Requirements

1. User must be able to register for an account in the system

1.1. The system must provide three input fields for the user

1.1.1. One of the input fields must be the user's full name

1.1.2. One of the input fields must be the user's email address

- 1.1.3. One of the input fields must be the user's password
- 1.2. The user must fill in all the input fields before registering
- 1.3. The system must verify that the inputted information is correct
 - 1.3.1. The system must verify that the email address is valid
 - 1.3.2. The system must verify that the email address has not been registered
 - 1.3.3. The password must contain at least 8 characters
 - 1.3.4. The system must notify the user if any of the above is not met
- 1.4. The system must create an account for the user once verification is completed
- 1.5. The system must direct the user to the main page upon completion

4.2 Login

4.2.1 Description and Priority

Description	The user can login to their account with their correct credentials that they inputted
Priority	High

4.2.2 Stimulus/Response Sequences

Use Case ID:	002		
Use Case Name:	Login		
Created By:	Leow Zheng Jie	Last Updated By:	Leow Zheng Jie
Date Created:	22 February 2024	Date Last Updated:	

Actor:	User (Initiating Actor), SimplyStay Website, Database
Preconditions:	1. The user must be connected to the Internet. 2. The user has a registered account.

Postconditions:	<p>1. The User has successfully logged into his/her own account.</p> <p>OR</p> <p>2. The user is notified of the reason(s) why they are unable to login into his account</p>
Frequency of Use:	High
Flow of Events:	<p>1. At the main page, the user can click onto the “Login” button and the system will redirect the user to the login page.</p> <p>2. At the login page, the system requests the input of both the email address and password.</p> <p>3. The user inputs their registered email address and his password.</p> <p>4. The user clicks on the “Login” or “Login with Google” button.</p> <p>5. The system verifies the credentials provided with the database.</p> <p>6. If the Email and Password are correct and verified, the user will be directed into his account.</p>
Alternative Flows:	<p><u>AF-S4: The user left the email or password input field blank.</u></p> <p>1. Upon clicking onto the “Login” button, the system displays the following notification: “Login failed. Please try again.”</p>

	<p>2. The system will prompt the user to fill up all the fields.</p> <p>3. The system returns to Step 3 and waits for the user to fill in all the fields.</p> <p><u>AF-S5: The user inputs an incorrect email address or password.</u></p> <p>1. Upon clicking onto the “Login” button, the system displays the following notification: “Login failed. Please try again.”</p> <p>2. The system returns to Step 3 and waits for the user to fill in the fields again.</p>
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

4.2.3 Functional Requirements

1. The user must be able to login using their credentials
 - 1.1. The system must provide two input fields for the user

- 1.1.1. One of the input fields must be the user's email address
- 1.1.2. One of the input fields must be the user's password
- 1.2. The user must fill in all the input fields before logging in
- 1.3. The system must verify that the inputted information is correct
 - 1.3.1. The system must verify that the email address is valid
 - 1.3.3. The password must be tied to the email address
 - 1.3.8. The system must notify the user if any of the above is not met
- 1.4. The system must log in the user once verification is completed
- 1.5. The system must direct the user to the user account page upon completion

4.3 Searching by Price

4.3.1 Description and Priority

Description	Users are able to search for flats within a specified price range.
Priority	High

4.3.2 Stimulus/Response Sequences

Use Case ID:	003a		
Use Case Name:	Searching by price		
Created By:	Enric Tan	Last Updated By:	Enric Tan
Date Created:	6 February 2024	Date Last Updated:	27 February 2024

Actor:	User (Initiating Actor), SimplyStay Website, Database
Preconditions:	<ol style="list-style-type: none">1. The user must be connected to the Internet.2. The user has navigated to the search page
Postconditions:	<ol style="list-style-type: none">1. The user successfully receives details on the list of available flats within the specified price.

	<p>OR</p> <p>2. No flats meet the specified price required by the user and therefore result in an empty search result.</p>
Frequency of Use:	Frequent
Flow of Events:	<p>1. At the Search page of the website, multiple filters will be displayed. Filters include maximum price, minimum price, geographic area, and housing type.</p> <p>2. The user can select the filters they want to apply on the results of their search for potential flats that they are interested in.</p> <p>3. Once the user is satisfied with the filters applied, the user will click on the search button.</p> <p>4. Results of the filtered search will be displayed to the user in a list.</p> <p>5. Detailed and concise information of the flats will pop up when the user clicks on the flat's name. Information includes room type and their price range, nearest MRT station and the time taken, as well as a Google Maps box that shows its exact location. There will also be a favourite button to bookmark the selected flat.</p>
Alternative Flows:	<u>AF-S4: The user applied a specified price range where no available flats satisfy.</u>

	<ol style="list-style-type: none">1. The message “No available flats!” will be displayed.2. The user returns to Step 2 and reapply filters.
Exceptions:	
Includes:	Favourite Houses, Maps
Special Requirements:	
Assumptions:	
Notes and Issues:	

4.3.3 Functional Requirements

1. User must be able to search houses by price.
 - 1.1. Web page must display all available houses with a given price range.
 - 1.2. Price must be in SGD.
 - 1.3. Price entered must be a number above 0 and below 1000000.

4.4 Searching by Geographical Area

4.4.1 Description and Priority

Description	Users are able to search for flats within a specified geographical area.
Priority	High

4.4.2 Stimulus/Response Sequences

Use Case ID:	003b		
Use Case Name:	Searching by geographical area		
Created By:	Enric Tan	Last Updated By:	Enric Tan
Date Created:	6 February 2024	Date Last Updated:	27 February 2024

Actor:	User (Initiating Actor), SimplyStay Website, Database
Description:	Users are able to search for flats within a specified geographical area.
Preconditions:	<ol style="list-style-type: none">1. The user must be connected to the Internet.2. The user has navigated to the search page

Postconditions:	<ol style="list-style-type: none">1. The user successfully receives details on the list of available flats within the specified geographical area. <p>OR</p> <ol style="list-style-type: none">2. No flats are located in the geographical area specified by the user and therefore result in an empty search result.
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none">1. At the Search page of the website, multiple filters will be displayed. Filters include maximum price, minimum price, geographic area, and housing type.2. The user can select the filters they want to apply on the results of their search for potential flats that they are interested in.3. Once the user is satisfied with the filters applied, the user will click on the search button.4. Results of the filtered search will be displayed to the user in a list.5. Detailed and concise information of the flats will pop up when the user clicks on the flat's name. Information includes room type and their price range, nearest MRT station and the time taken, as well as a Google Maps box that shows its exact location. There will also be a favourite button to bookmark the selected flat.

Alternative Flows:	<p>AF-S4: <u>The user applied a geographic area filter where no available flats satisfy.</u></p> <p>1. The message “No available flats!” will be displayed.</p> <p>2. The user returns to Step 2 and reapply filters.</p>
Exceptions:	
Includes:	Favourite Houses, Maps
Special Requirements:	
Assumptions:	
Notes and Issues:	

4.4.3 Functional Requirements

1. User must be able to search houses by location.
 - 1.1. Webpage must display all available houses within the selected area.
 - 1.1.1. Examples include Bedok, Redhill, Lakeside, etc.

4.5 Searching by Housing Type

4.5.1 Description and Priority

Description	Users are able to search for flats within a specified housing type.
Priority	High

4.5.2 Stimulus/Response Sequences

Use Case ID:	003c		
Use Case Name:	Searching by housing type		
Created By:	Enric Tan	Last Updated By:	Enric Tan
Date Created:	6 February 2024	Date Last Updated:	27 February 2024

Actor:	User (Initiating Actor), SimplyStay Website, Database
Description:	Users are able to search for flats within a specified housing type.
Preconditions:	<ol style="list-style-type: none">1. The user must be connected to the Internet.2. The user has navigated to the search page

Postconditions:	<ol style="list-style-type: none">1. The user successfully receives details on the list of available flats within the specified housing type. <p>OR</p> <ol style="list-style-type: none">2. No flats meet the specified housing type required by the user and therefore result in an empty search result.
Priority:	
Frequency of Use:	
Flow of Events:	<ol style="list-style-type: none">1. At the Search page of the website, multiple filters will be displayed. Filters include maximum price, minimum price, geographic area, and housing type.2. The user can select the filters they want to apply on the results of their search for potential flats that they are interested in.3. Once the user is satisfied with the filters applied, the user will click on the search button.4. Results of the filtered search will be displayed to the user in a list.5. Detailed and concise information of the flats will pop up when the user clicks on the flat's name. Information includes room type and their price range, nearest MRT station and the time taken, as well as a

	Google Maps box that shows its exact location. There will also be a favourite button to bookmark the selected flat.
Alternative Flows:	<p><u>AF-S4: The user applied a housing type filter where no available flats satisfy.</u></p> <p>1. The message “No available flats!” will be displayed.</p> <p>2. The user returns to Step 2 and reapply filters.</p>
Exceptions:	
Includes:	Favourite Houses, Maps
Special Requirements:	
Assumptions:	
Notes and Issues:	

4.5.3 Functional Requirements

1. User must be able to search houses by housing type.
 - 1.1. Webpage must be able to display housing type requirements within the Search Page
 - 1.1.1. Webpage must be able to display all two room flexi houses within Singapore

- 1.1.2. Webpage must be able to display all three room houses within Singapore
- 1.1.3. Webpage must be able to display all four room houses within Singapore
- 1.1.4. Webpage must be able to display all five room houses within Singapore

4.6 Searching by Multiple Filters

4.6.1 Description and Priority

Description	Users can search for flats within a specified requirement. Requirements include maximum price, minimum price, geographic area, and housing type. Users should also be able to filter flats based on the fields mentioned.
Priority	High

4.6.2 Stimulus/Response Sequences

Use Case ID:	003d		
Use Case Name:	Searching		
Created By:	Enric Tan	Last Updated By:	Enric Tan
Date Created:	6 February 2024	Date Last Updated:	

Actor:	User (Initiating Actor), SimplyStay Website, Database
Preconditions:	1. The user must be connected to the Internet.

	2. The user has navigated to the search page
Postconditions:	<p>1. The user successfully receives details on the list of available flats within the specified requirements.</p> <p>OR</p> <p>2. No flats meet the specified requirements of the user and therefore result in an empty search result.</p>
Frequency of Use:	High
Flow of Events:	<p>1. At the Search page of the website, multiple filters will be displayed. Filters include maximum price, minimum price, geographic area, and housing type.</p> <p>2. The user can select the filters they want to apply on the results of their search for potential flats that they are interested in.</p> <p>3. Once the user is satisfied with the filters applied, the user will click on the search button.</p> <p>4. Results of the filtered search will be displayed to the user in a list.</p> <p>5. Detailed and concise information of the flats will pop up when the user clicks on the flat's name. Information includes room type and their price range, nearest MRT station and the time taken, as well as a</p>

	Google Maps box that shows its exact location. There will also be a favourite button to bookmark the selected flat.
Alternative Flows:	<p><u>AF-S4: The user applied a filter where no available flats satisfy.</u></p> <ol style="list-style-type: none"> 1. The notification “No available flats!” will be displayed. 2. The user returns to Step 2 and reapply filters.
Exceptions:	
Includes:	Favourite Houses, Maps
Special Requirements:	
Assumptions:	
Notes and Issues:	

4.6.3 Functional Requirements

1. User must be able to search for houses with the filters provided
 - 1.1. The system must provide four filters for the user
 - 1.1.1. One of the filters must be by maximum price in SGD
 - 1.1.2. One of the filters must be by minimum price in SGD
 - 1.1.3. One of the filters must be by location
 - 1.1.4. One of the filters must be by housing type

- 1.2. The user must be able to select any of the filters
- 1.3. The system must display all houses that satisfy the user's chosen filters
 - 1.3.1 The system must display the information in text form
- 1.4. The user must be able to bookmark the filtered houses

4.7 Favourite Houses

4.7.1 Description and Priority

Description	Users can add flats that they have searched for into a page called Favourites. The user will also be able to view the list of Favourite Houses bookmarked
Priority	High

4.7.2 Stimulus/Response Sequences

Use Case ID:	004		
Use Case Name:	Favourite Houses		
Created By:	Enric Tan	Last Updated By:	Enric Tan
Date Created:	6 February 2024	Date Last Updated:	

Actor:	User (Initiating Actor), SimplyStay Website, Database
Preconditions:	<ol style="list-style-type: none">1. The user must be connected to the Internet.2. The user has a valid account.

Postconditions:	1. The system successfully saves the bookmarked flats into the Favorite Houses folder.
Frequency of Use:	High
Flow of Events:	<p>1. At the Favourite page of the website, the user will be able to see all the flats that he has bookmarked under his account.</p> <p>2. If there are currently no bookmarked flats, the page will contain an empty list.</p> <p>3. The user will be able to click into the bookmarked flats to view more information on the house.</p> <p>4. Upon clicking on the bookmarked flats, the user will be brought to a page where detailed information of the house is displayed.</p> <p>5. The user will be able to navigate back to the Favourite Houses page by clicking on a back button available at the bottom of the detailed information page.</p> <p>6. The user is able to remove flats from the bookmarked folder by clicking on the delete button corresponding to the selected flat.</p> <p>7. To add flats to the favourite page, the user will follow the steps as described in use case descriptions 003a, 003b, 003c, 003d, under step 5.</p>

Alternative Flows:	
Exceptions:	
Includes:	Maps
Special Requirements:	
Assumptions:	
Notes and Issues:	

4.7.3 Functional Requirements

1. Logged in user must be able to view their bookmarked houses
 - 1.1. The system must display all the bookmarked houses of the user
 - 1.2. Logged in user must be able to remove their favourite houses
 - 1.2. Guest users must not be able to view the favourites page
 - 1.2.1. The system must notify the guest user of the restriction

4.8 Eligibility Calculator

4.8.1 Description and Priority

Description	User will be able to determine if they are eligible to BTO, and if so, the downpayment required and the total grants given to them, based on their monthly income, monthly expenses, relationship status, CPF, and savings.
Priority	High

4.8.2 Stimulus/Response Sequences

Use Case ID:	005a		
Use Case Name:	Eligibility Calculator		
Created By:	Nigel Paliath	Last Updated By:	Nigel Paliath
Date Created:	6 February 2024	Date Last Updated:	18 April

Actor:	User (Initiating Actor), SimplyStay Website
Preconditions:	1. User has valid internet connection

	2. User has valid inputs
Postconditions:	1. The webpage will display information on how the user would finance the target house (eg. downpayment, mortgage)
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. User will hover over the Calculator tab at the top of the webpage. 2. User will select "Eligibility Calculator" from the drop-down menu 3. User will key in their age, relationship status, monthly income, monthly expenses, CPF, and cash savings details in their respective fields. 4. User will click on the "Calculate" button. 5. Webpage will display financing information after calculations are done.
Alternative Flows:	<p><u>AF-4: The user has invalid inputs for any of the input boxes</u></p> <ol style="list-style-type: none"> 1. A prompt indicating the error would appear underneath the input box 2. The user would not be able to calculate anything until the errors have been corrected
Exceptions:	
Includes:	

Special Requirements:	
Assumptions:	
Notes and Issues:	

4.8.3 Functional Requirements

1. The user must be able to check if they are eligible to BTO

1.1. The system must provide six input fields for the user

1.1.1. One of the input fields must be the user's age

1.1.2. One of the input fields must be to indicate if the user is applying with a partner

1.1.3. One of the input fields must be the user's combined/single income

1.1.4. One of the input fields must be the user's monthly expenses

1.1.5. One of the input fields must be the user's CPF in SGD

1.1.6. One of the input fields must be the user's savings in SGD

1.1.7. The user must enter all six input fields

- 1.1.8. The system must notify the user if any of the inputs are unsuccessful
- 1.2. The system must display the buying power of the user in SGD
- 1.3. The system must display the down payment required in SGD
- 1.4. The system must display the grants available to the user in SGD

4.9 Housing Loan Calculator

4.9.1 Description and Priority

Description	User will be able to calculate how they would finance his/her target house based on the house price, loan amount, annual interest rate, and loan tenure
Priority	High

4.9.2 Stimulus/Response Sequences

Use Case ID:	005b		
Use Case Name:	Housing Loan Calculator		
Created By:	Nigel Paliath	Last Updated By:	Nigel Paliath
Date Created:	1 April 2024	Date Last Updated:	14 April 2024

Actor:	User (Initiating Actor), SimplyStay Website
Preconditions:	1. User has valid internet connection

	2. User has valid inputs
Postconditions:	1. The webpage will display information on the monthly instalments and the total interest to be paid
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. User will hover over the Calculator tab at the top of the webpage. 2. User will select "Loan Calculator" from the drop-down menu 3. User will key in their target house price, loan amount, interest rate and loan tenure. 4. User will click on the "Calculate" button. 5. Webpage will display financing information after calculations are done.
Alternative Flows:	<u>AF-4: The user has invalid inputs for the in the input boxes</u> <ol style="list-style-type: none"> 1. A prompt indicating the error would appear underneath the input box 2. The user would not be able to calculate anything until the errors have been corrected
Exceptions:	
Includes:	
Special Requirements:	

Assumptions:	
Notes and Issues:	

4.9.3 Functional Requirements

1. The user must be able to check the payment plans of their BTO

1.1. The system must provide four input fields for the user

1.1.1. One of the input fields must be the house price in SGD

1.1.2. One of the input fields must be the loan amount in SGD

1.1.3. One of the input fields must be the interest rate for the loan in percentage

1.1.4. One of the input fields must be the loan tenure in years

1.1.5. The system must notify the user if any of the inputs are unsuccessful

1.2. The system must display the maximum amount the user is able to loan based on the house price

1.3. The system must display the monthly instalment for the loan in SGD

1.4. The system must display the total amount repaid for the loan in SGD

1.5. The system must display the total interest paid by the user

1.6. The system must display the Loan-To-Value ratio in percent

4.10 Comparison

4.10.1 Description and Priority

Description	User will be able to compare 2 different houses based on their respective specifications
Priority	High

4.10.2 Stimulus/Response Sequences

Use Case ID:	006		
Use Case Name:	Comparison		
Created By:	Nigel Paliath	Last Updated By:	Nigel Paliath
Date Created:	6 February 2024	Date Last Updated:	

Actor:	User (Initiating Actor), SimplyStay Website, Database
Preconditions:	1. User has valid internet connection
Postconditions:	1. The webpage will display a table of comparisons between the two target houses' specifications

Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none">1. User will click on the Compare tab at the top of the webpage.1. User will need to select 2 different houses as their target houses for comparison from a drop-down list.2. User will click on the “Compare” button.3. Webpage will display a table comparing the 2 houses’ price, location, housing type and nearest MRT station time.4. User will be able to make a quick comparison from a glance as the more favourable statistic will be highlighted in green (e.g. cheaper price) while the less favourable statistic (e.g. longer time to MRT station) will be highlighted in red.
Alternative Flows:	
Exceptions:	
Includes:	Maps
Special Requirements:	
Assumptions:	
Notes and Issues:	

4.10.3 Functional Requirements

1. User must be able to compare two houses of their own selection
 - 1.1. The system must be able to display all the houses available
 - 1.2. The user must be able to select two houses by name before comparing
 - 1.3. The system must inform the user to select two houses before comparing
 - 1.4. The system must display the two houses' information side by side
 - 1.5. The system must indicate the superior and inferior statistics by colour
 - 1.5.1 The system must display the superior statistics in green
 - 1.5.2. The system must display the inferior statistics in red

4.11 Maps

4.11.1 Description and Priority

Description	Google Maps API will use the coordinates of the users' target house and the API would display the location of the house in a box.
Priority	High

4.11.2 Stimulus/Response Sequences

Use Case ID:	007		
Use Case Name:	Maps		
Created By:	Nigel Paliath	Last Updated By:	Nigel Paliath
Date Created:	17 February 2024	Date Last Updated:	

Actor:	User (Initiating Actor), SimplyStay Website, Google Maps
Preconditions:	1. User has valid internet connection 1. Google Maps API needs to be set up
Postconditions:	The webpage will display the the location of the house in a box.

Frequency of Use:	High
Flow of Events:	1. After the user selects the target house, the box will display the location of the box in Google Maps.
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

4.11.3 Functional Requirements

1. The system must display the location of the selected house
 - 1.1. The user must select a house
 - 1.2. The system must use the Google Maps API on the selected house
 - 1.3. The system must display nearby points of interest
 - 1.3.1 The system must display nearby amenities
 - 1.3.2. The system must display nearby public transport infrastructure

4.12 Logout

4.12.1 Description and Priority

Description	Users can log out of their accounts
Priority	High

4.12.2 Stimulus/Response Sequences

Use Case ID:	008		
Use Case Name:	Logout		
Created By:	Enric Tan	Last Updated By:	Enric Tan
Date Created:	17 February 2024	Date Last Updated:	

Actor:	User (Initiating Actor), SimplyStay Account System, Database
Preconditions:	<ol style="list-style-type: none">1. User has an account.2. User is already logged in to his account.
Postconditions:	<ol style="list-style-type: none">1. The user will not be logged in to any account on the website
Frequency of Use:	Low

Flow of Events:	<ol style="list-style-type: none"> 1. At the top right-hand corner of every page on the website, there will be a Profile icon that users can click on. 2. Upon clicking on the Profile icon, if the user is logged in, the user's profile will be displayed. 3. In the profile box, there will be a "Logout" button. 4. The user will be able to click on the "Logout" button to log out of their account. 5. The user will be returned to the website's main page upon logging out, there will be no active logged in account at this point.
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

4.12.3 Functional Requirements

1. The user must be able to log out of their account
 - 1.1. The system must be able to save the user's bookmarked houses
 - 1.2. The user must be able to log back in later

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- The system must be able to return the results within 5 seconds of querying.
- The website must be fully functional within 3 seconds of opening it.
- The user must be able to successfully register an account within 1 minute of entering the website.
- The user must be able to login within 3 seconds of inputting their login credentials

5.2 Security Requirements

- Users must not be able to view other users' email addresses and passwords on the website
- Users must not be able to change other users' passwords on the website
- The system must not reveal the user's password when the user inputs it

5.3 Software Quality Attributes

5.3.1 Maintainability

- The code must be modular and reusable where applicable
- Dependencies must be minimized to prevent coupling
- Documentation must be provided where necessary

5.3.2 Scalability

- The website must support more than 20 concurrent users

5.3.3 Usability

- The user interface must adhere to standard UI/UX principles

5.3.4 Availability

- The downtime of the website must not exceed 12 hours per year.

5.3.5 Reliability

- The system must display the correct information to the user 99.9% of the time

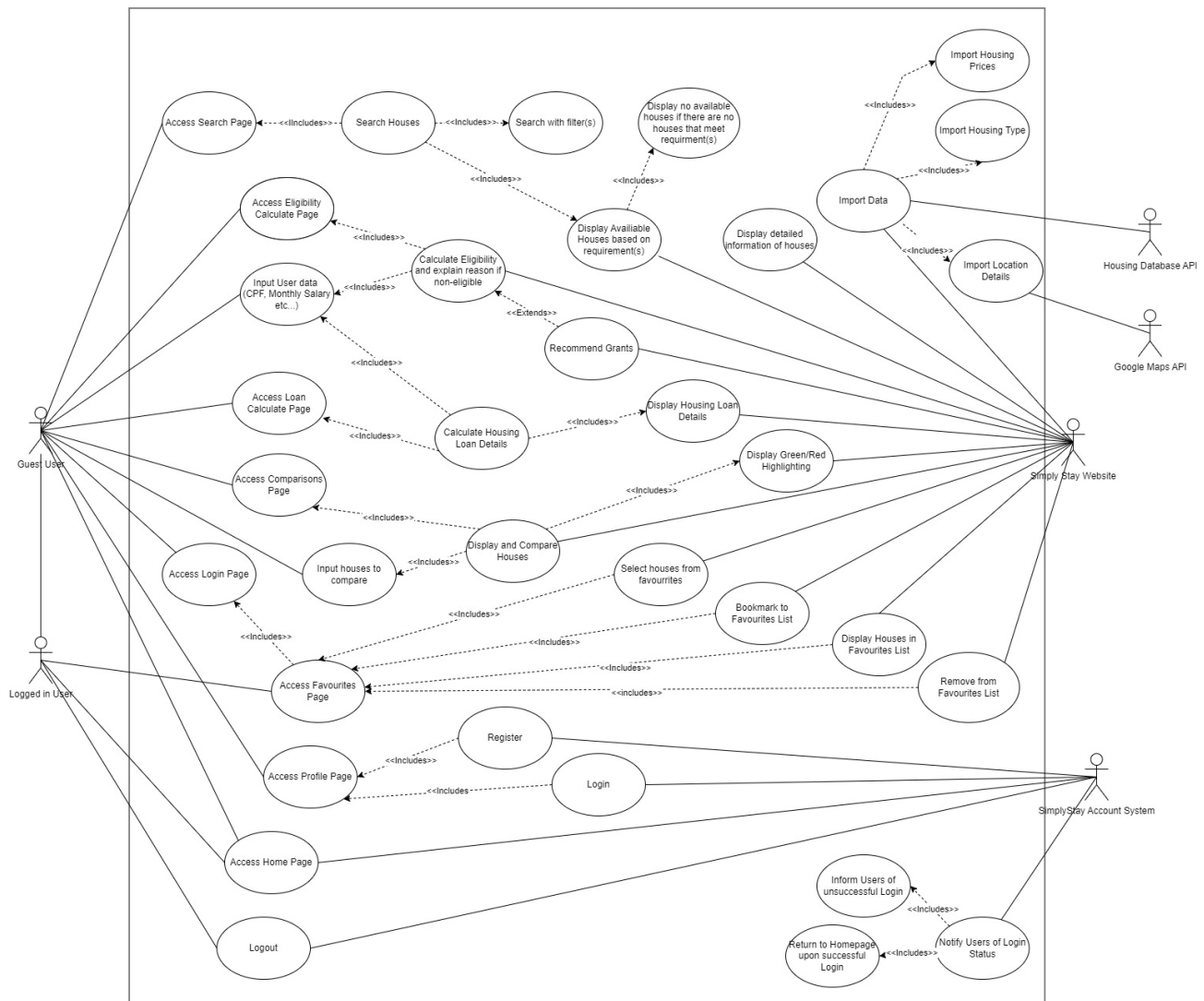
Appendix A: Data Dictionary

Created by	Jared and Nigel
Date created	30/1/2024
API	Application Programming Interface, which is a type of software interface used to facilitate to communication between two or more computer programs
User Account	A means by which users can access our website, and also save previous information and data within their account. Users will have to input a unique username as unique identification, and their registered email address.
Guest	A means of which users can make use of the website without having to sign up for an account. Take note that by using the website as a guest, users' data inputs and information will not be saved.
HDBs	Housing Development Board and is colloquially used as short-hand to describe any government housing in Singapore.
BTO	Built-to-Order. It is one of the modes of sale of new HDB flats in Singapore. An HDB BTO sales launch happens every three months and typically offers a range of HDB flat types across a few estates.
Housing Type	As defined by HDB: 2-room Flexi, 3-,4-,5-room, to 3Gen flats
Area	General housing regions in Singapore which users can inquire about. They are namely: North, South, East, West, Central
Price Range	Users input their price ranges for their housing units.

Comparison	A page on the website where users will have the ability to compare two housing units with regards to different factors, such as affordability, and location.
Database	A database containing all users' accounts, and their details, such as username, password, email/phone number, along with all other saved data with regards to their housing requirements, etc
Amenities and Infrastructure	Refers to nearby facilities and infrastructure (MRTs, Schools, Shopping Malls, etc)

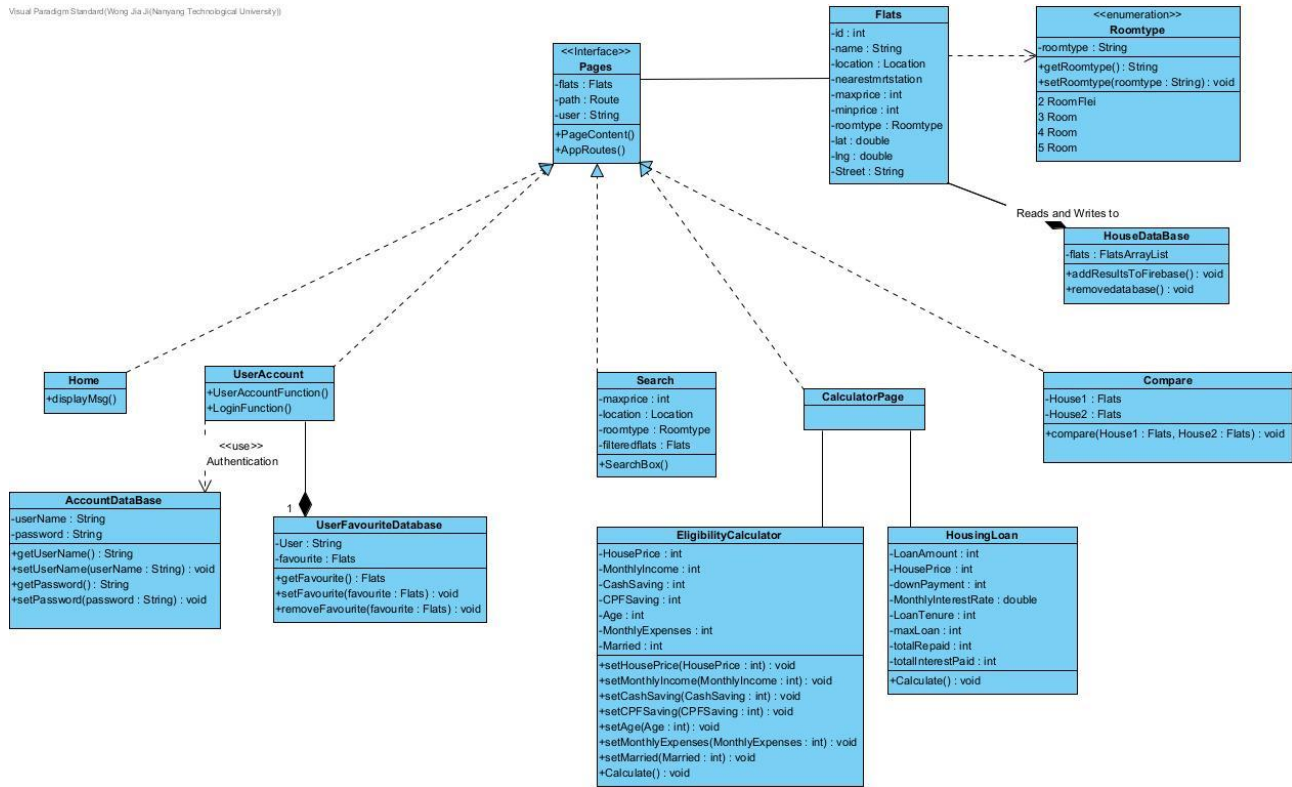
Appendix B: Analysis Models

B1: Use Case Diagram



B2: Class Diagram

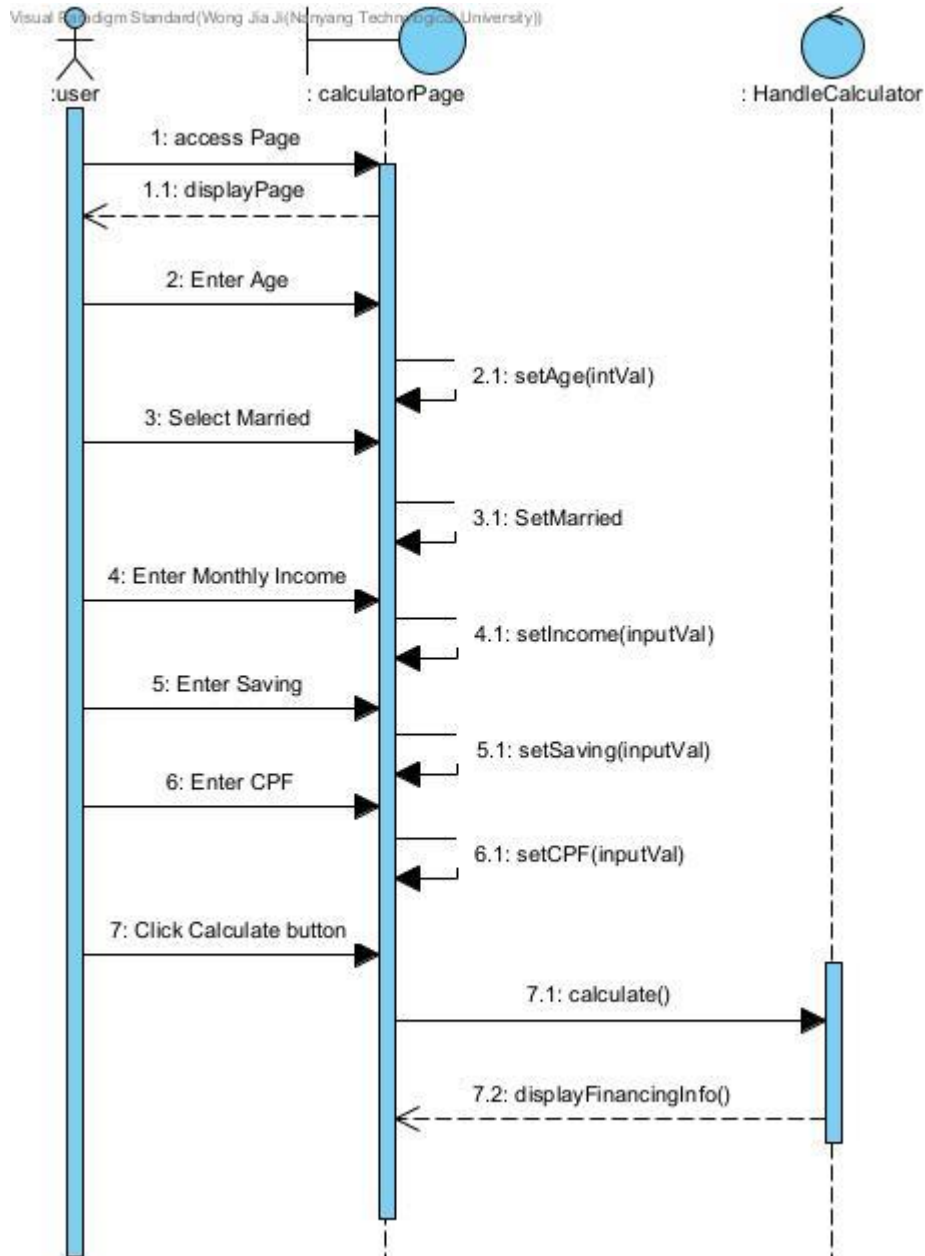
Visual Paradigm Standard (Wang Jia Jie/Nanjing Technological University)



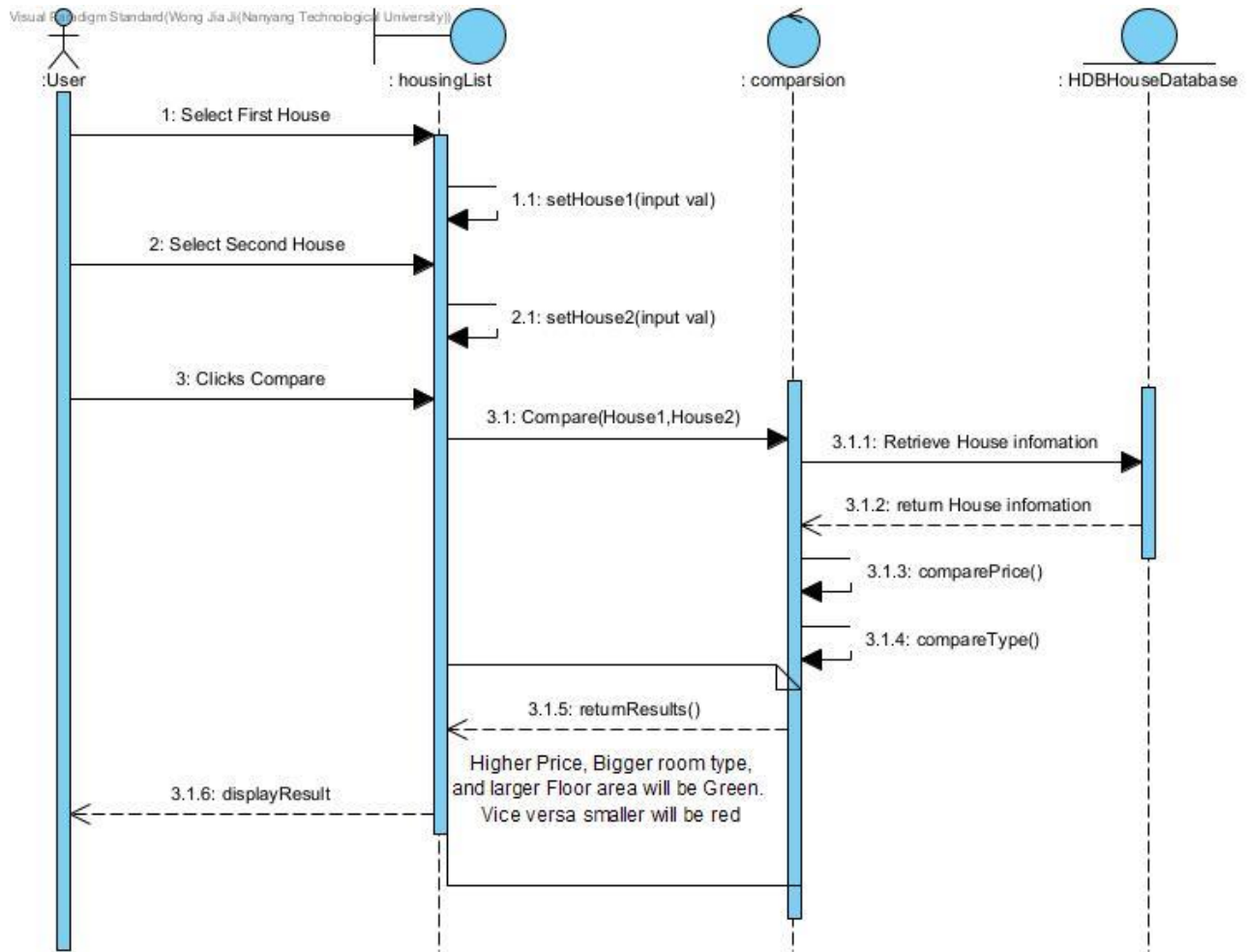
[illegible]

B4: Sequence Diagrams

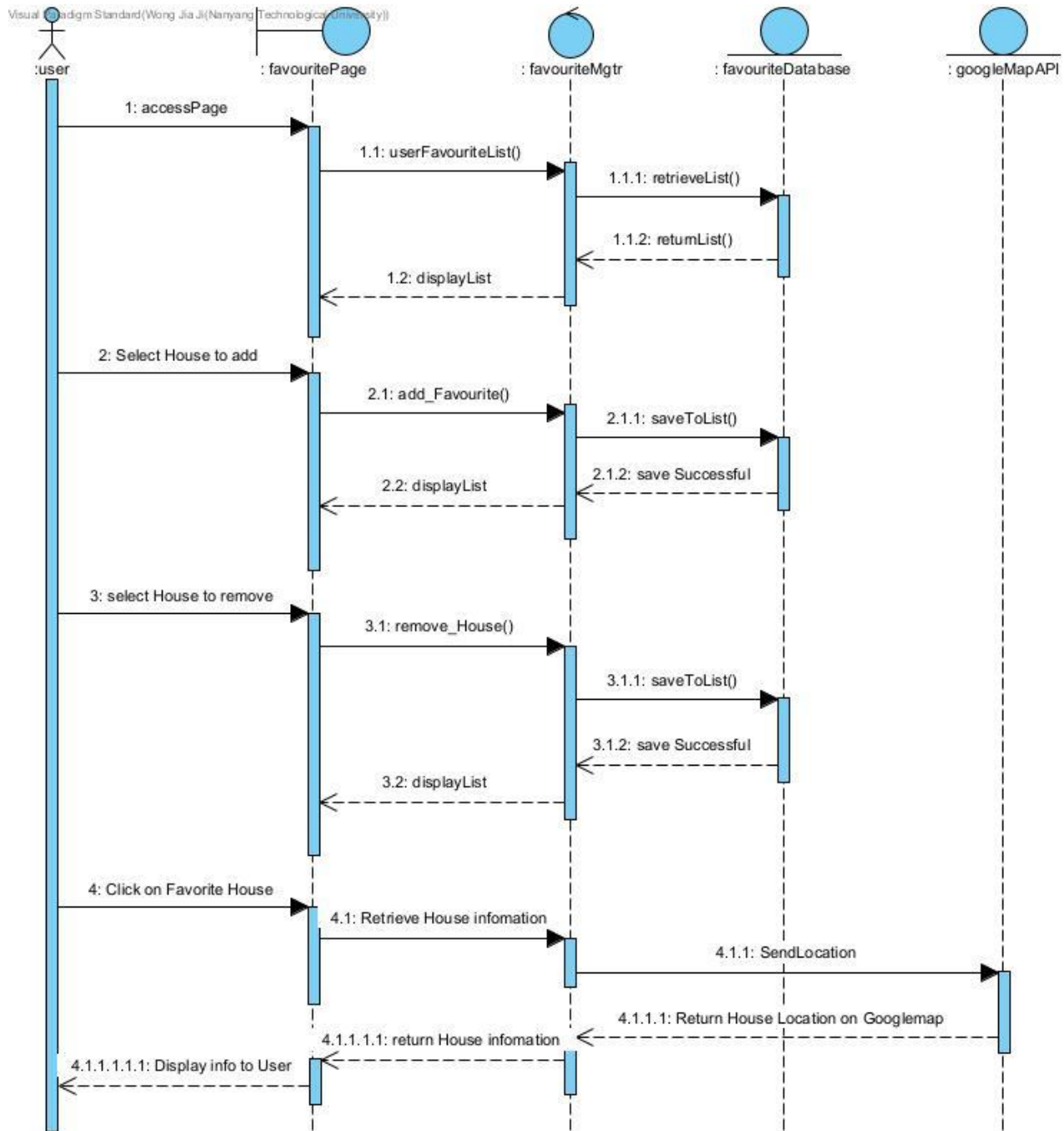
B4.1: Eligibility Calculator



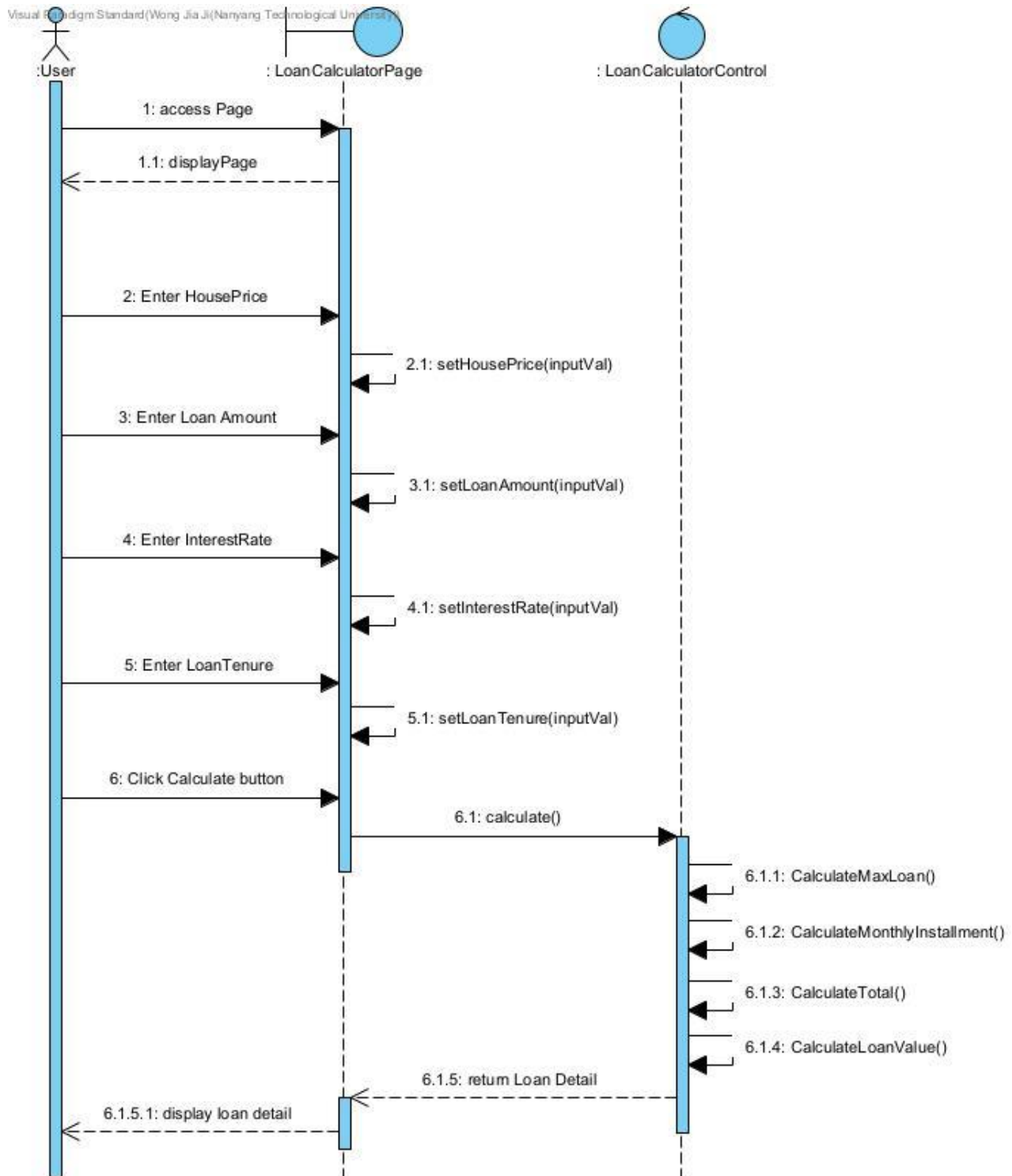
B4.2: Compare Houses



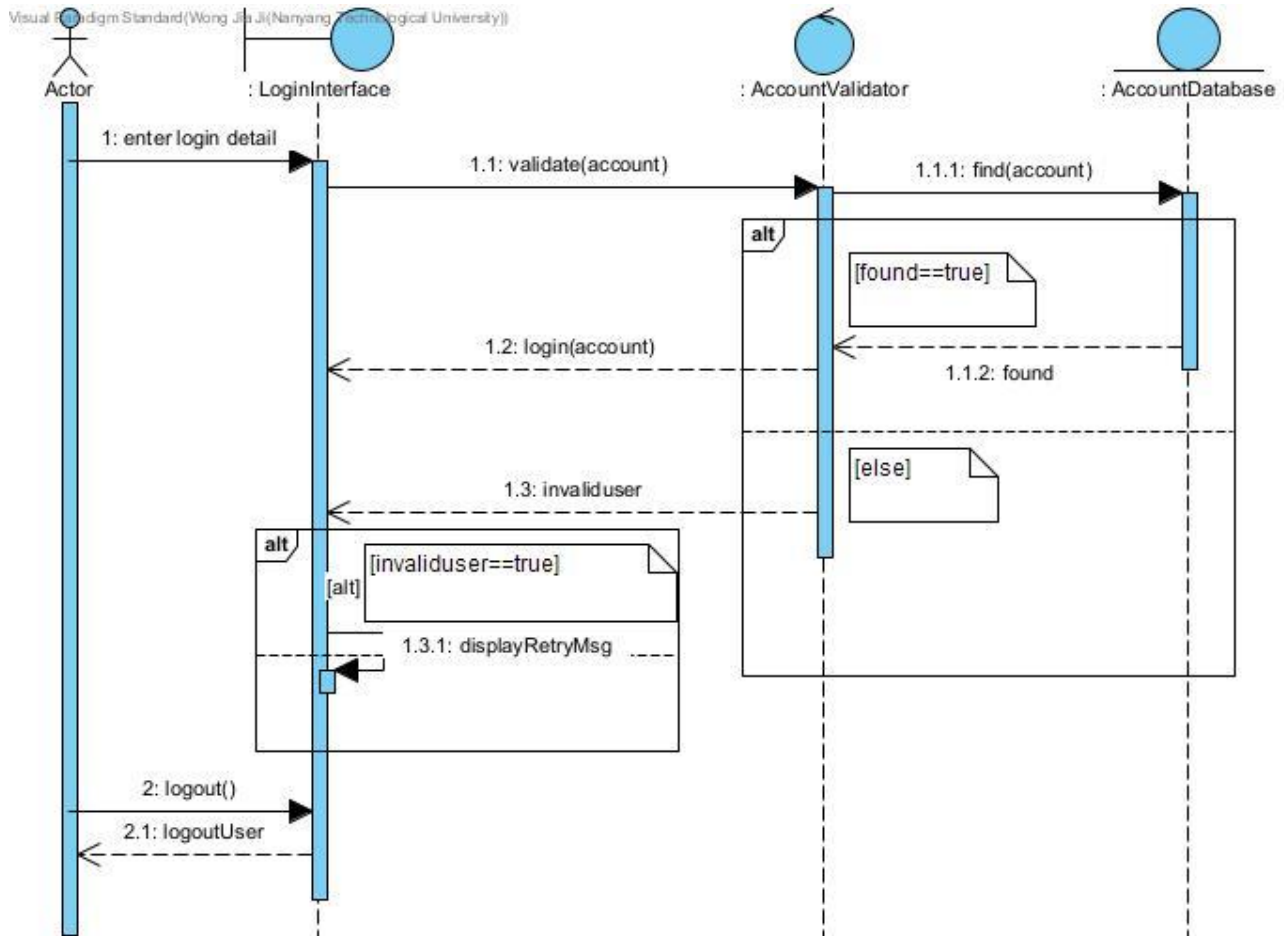
B4.3: Favourites



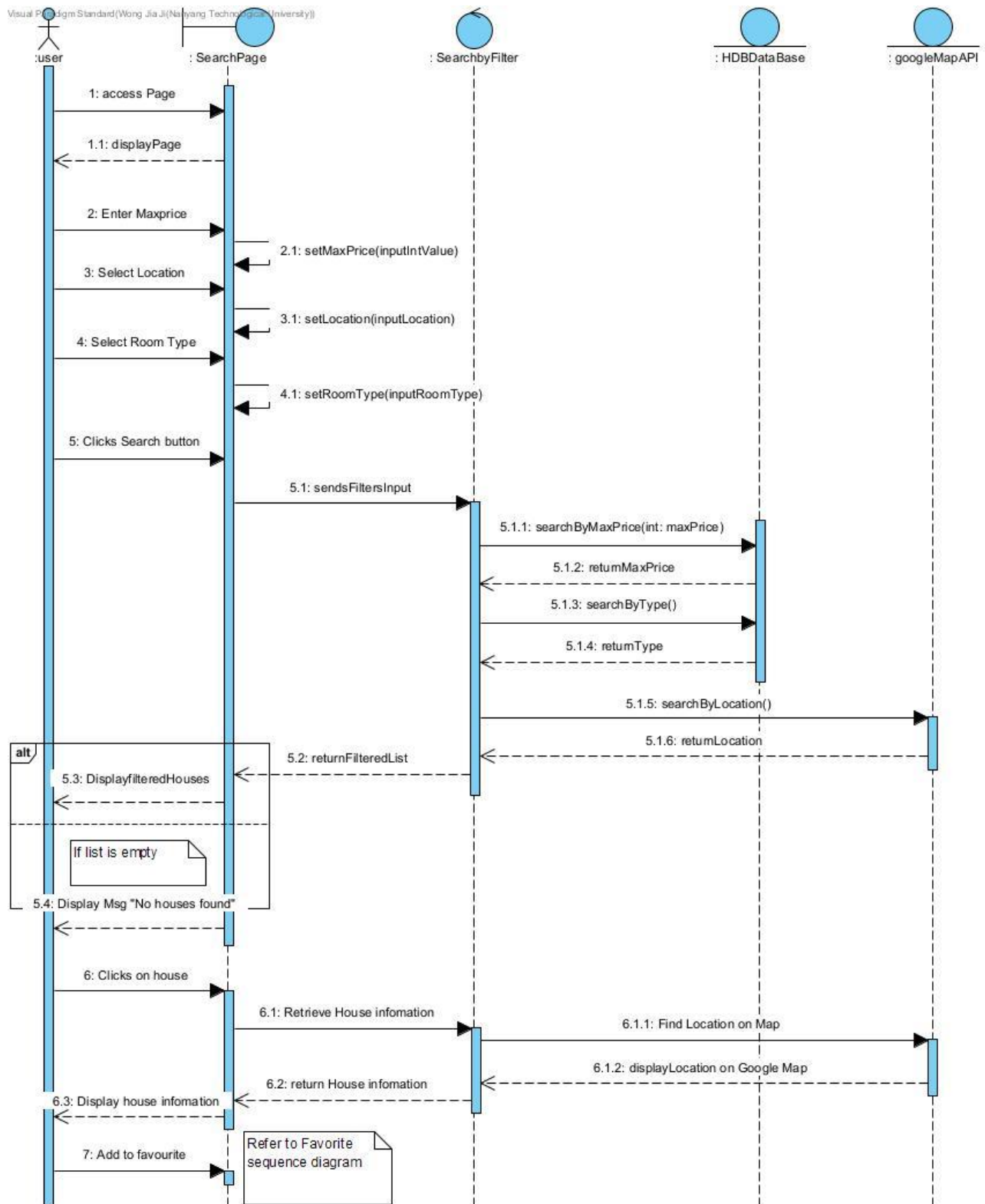
B4.4: Housing Loan Calculator



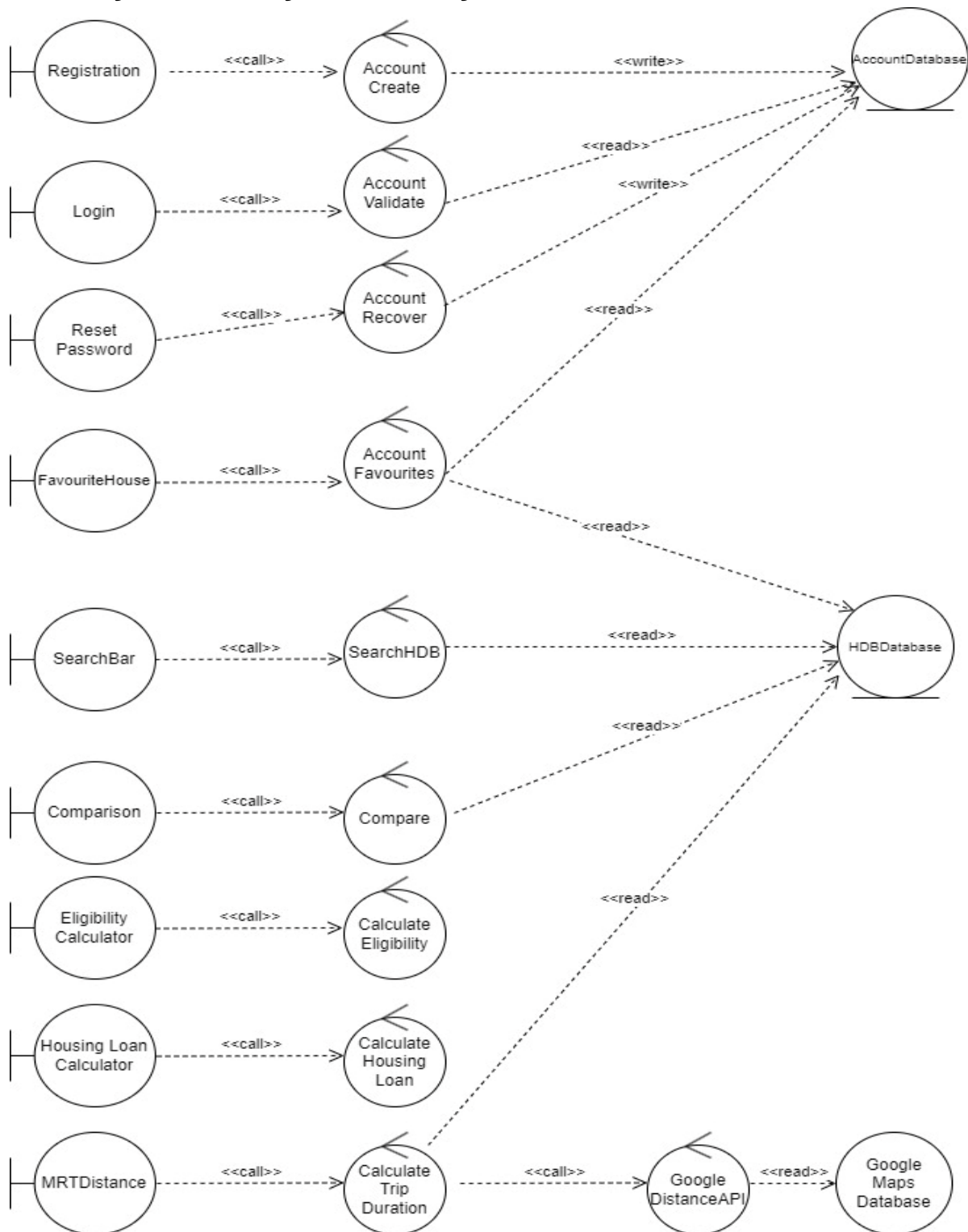
B4.5: Login



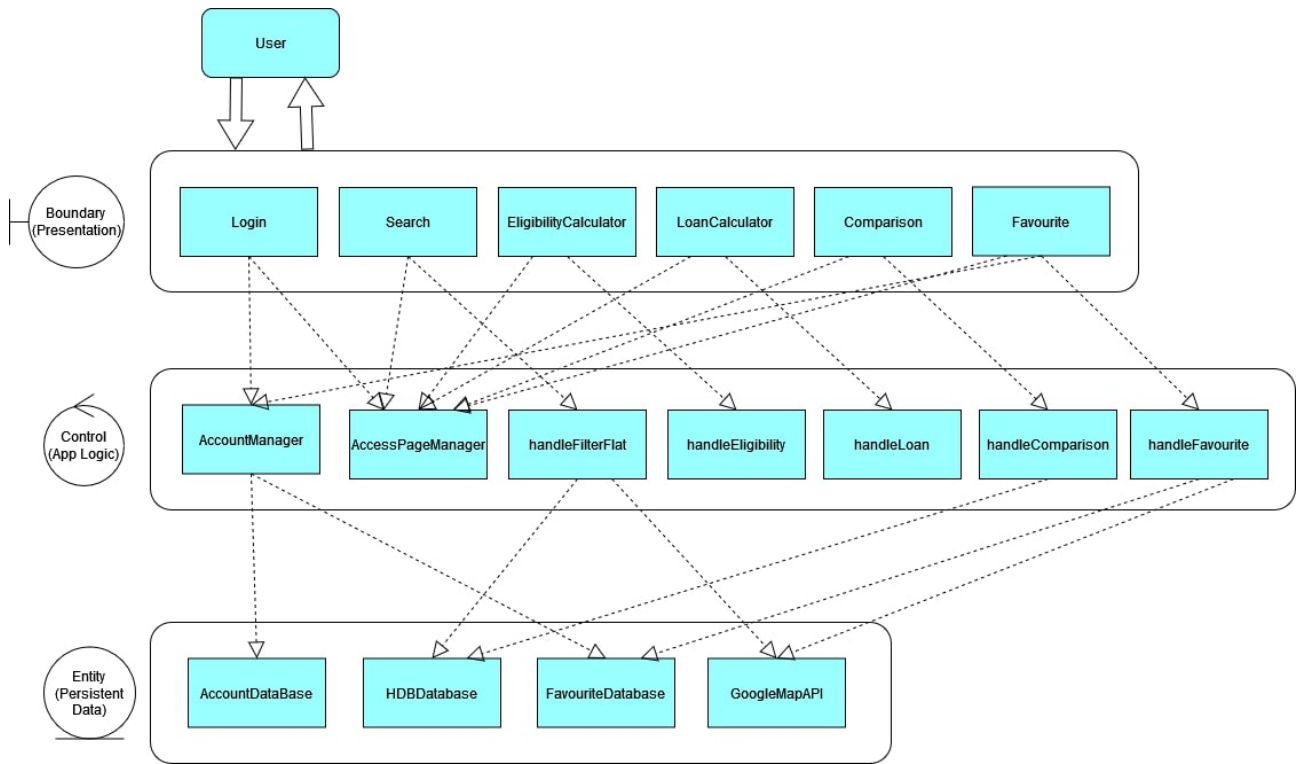
B4.6: Search Houses



B5: Key Boundary and Entity Classes



B6: Architecture Diagram



B7: Test Cases and Test Results

B7.1: Black Box Testing

1. Calculator for BTO eligibility
 - a. Generic cases

Test ID	Scenario	Expected Result	Actual Result
1	Valid inputs are entered for required fields	The system does not display any "please input a valid amount" message Display BTO Calculation info when Calculate is pressed	The system does not display any "please input a valid amount" message Display BTO Calculation info when Calculate is pressed
2	Non-integer and invalid input are entered for required fields	System display "Please input a valid amount" Nothing happens when "Calculate" is pressed	System display "Please input a valid amount" Nothing happens when "Calculate" is pressed
3	String input are entered into required fields	System display "Please input a valid amount" and fields displays default message.	System display "Please input a valid amount" and fields displays default message.

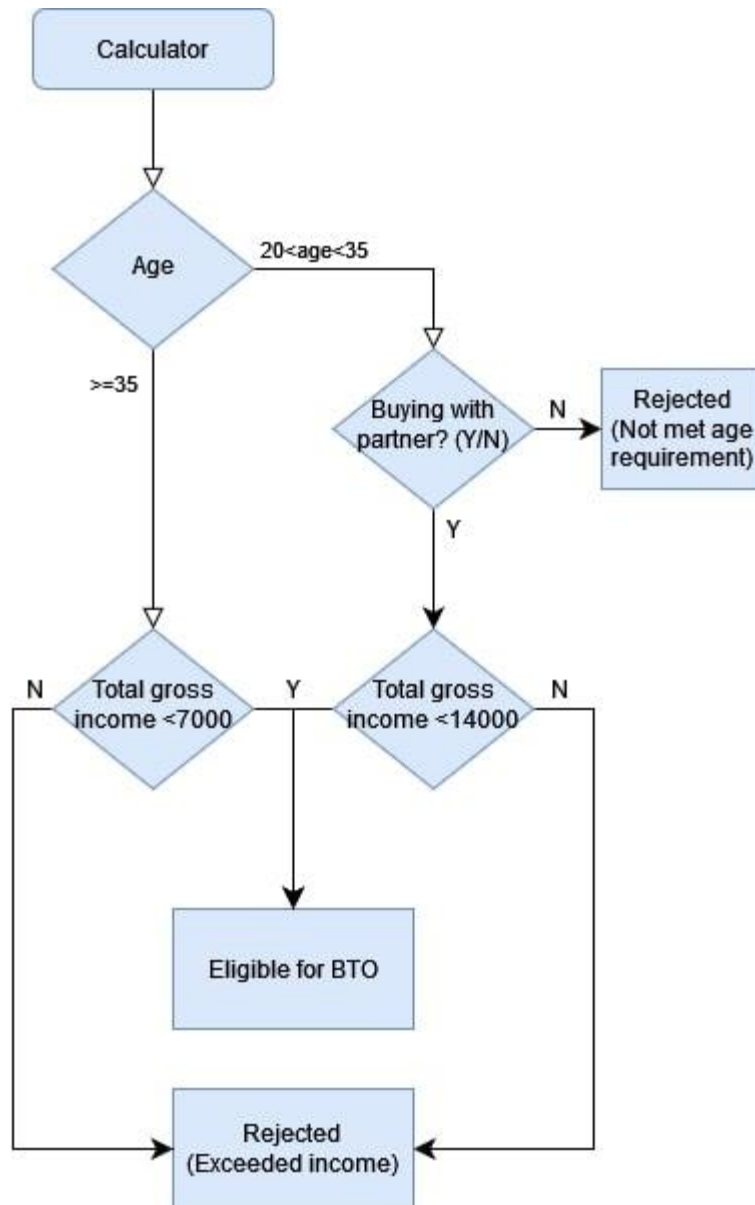
- b. Specific cases (BTO eligibility)

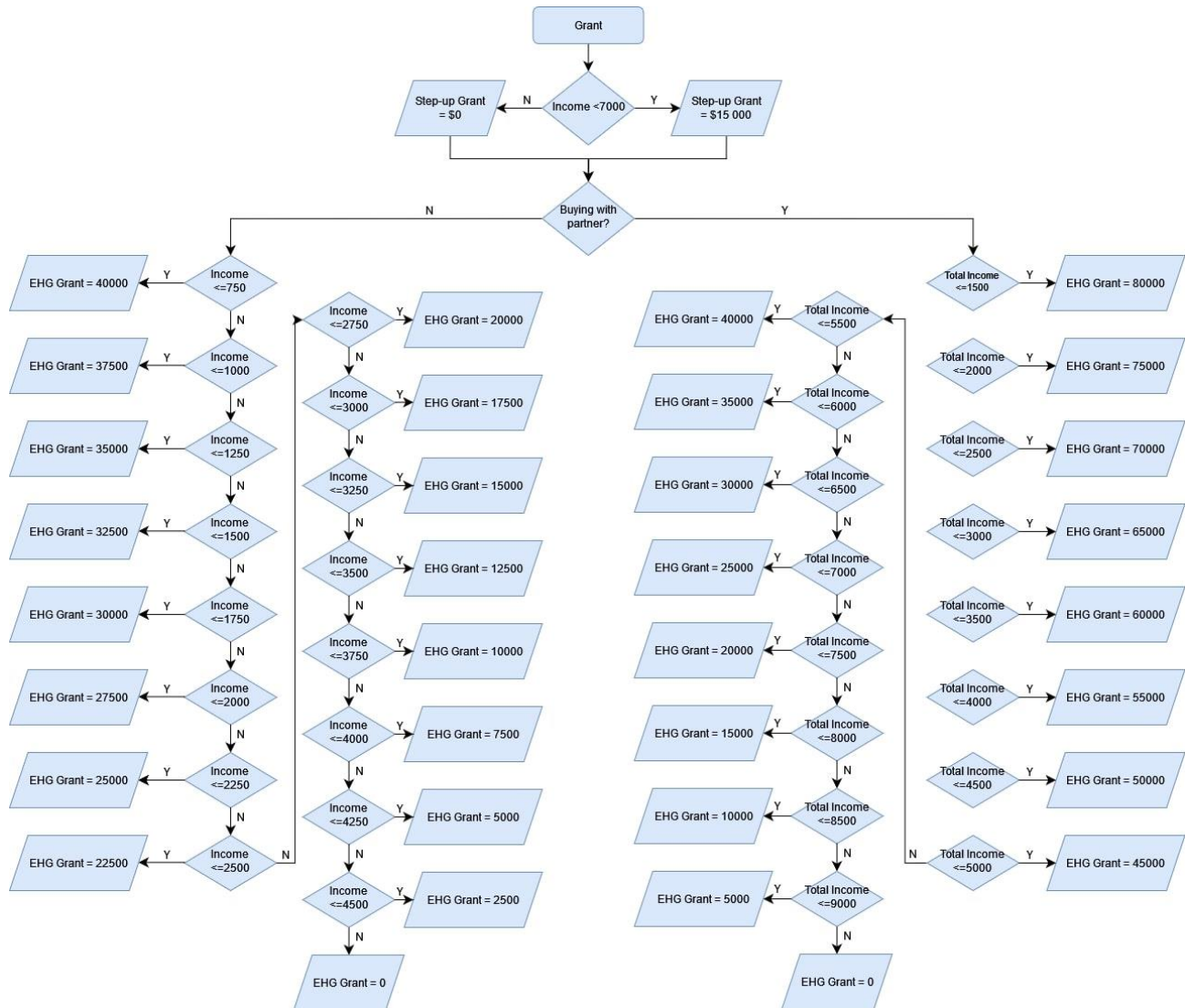
Age	Applying with partner (Y/N)	Combine income	Expected Result	Actual Result
21	N	7000	Ineligible (single below 35 years old)	Ineligible (single below 35 years old)
21	Y	14000	Eligible	Eligible
21	Y	14001	Ineligible(above income limit)	Ineligible(above income limit)
>35 & <100	N	7000	Eligible	Eligible
>35 & <100	N	7001	Ineligible(above income limit)	Ineligible(above income limit)
Empty ("")	Y	7000	Please input a valid age	Please input a valid age
35	Empty ("")	7000	'married is required'	'married is required'
35	Y	Empty ("")	Please input a valid amount	Please input a valid amount

c. Specific cases for Grant display (For eligible only)

Age	Applying with partner (Y/N)	Combine Income	Expected Result (Step-up Grant)	Expected Result (EHG Grant)	Actual Result (Step-up Grant)	Actual Result (EHG Grant)
35	N	750	15000	40000	15000	40000
		1000	15000	37500	15000	37500
		1250	15000	35000	15000	35000
		1500	15000	32500	15000	32500
		1750	15000	30000	15000	30000
		2000	15000	27500	15000	27500
		2250	15000	25000	15000	25000
		2500	15000	22500	15000	22500
		2750	15000	20000	15000	20000
		3000	15000	17500	15000	17500
		3250	15000	15000	15000	15000
		3500	15000	12500	15000	12500
		3750	15000	10000	15000	10000
		4000	15000	7500	15000	7500
		4250	15000	5000	15000	5000
		4500	15000	2500	15000	2500
21	Y	4501	15000	0	15000	0
		1500	15000	80000	15000	80000
		2000	15000	75000	15000	75000
		2500	15000	70000	15000	70000
		3000	15000	65000	15000	65000
		3500	15000	60000	15000	60000
		4000	15000	55000	15000	55000
		4500	15000	50000	15000	50000
		5000	15000	45000	15000	45000
		5500	15000	40000	15000	40000
		6000	15000	35000	15000	35000
		6500	15000	30000	15000	30000
		7000	15000	25000	15000	25000
		7001	0	20000	0	20000
		7500	0	20000	0	20000
		8000	0	15000	0	15000
		8500	0	10000	0	10000
		9000	0	5000	0	5000
		9001	0	0	0	0

B7.2: White Box Testing





Appendix C: Design Patterns and Principles Used

C1: Design Patterns

C1.1: Observer Pattern

We implemented the Observer design pattern in multiple aspects of our website, namely our SearchResults, CompareResults, and Favourites component. These components will only be updated once the user enters input and is implemented via React Hooks such as useEffect and useState.

This results in loose coupling between components and easily maintainable code.

C1.2: Façade Pattern

We also implemented the Façade design pattern in our user interfaces. The PageContent component abstracts and masks the complexities of our website and is the primary interface in which our user interacts with, rather than interacting with each component directly.

This enables us to change and adapt our codebase quickly in the event of new implementations and features. One example was the addition of a housing loan calculator, in addition to our eligibility calculator. With the façade pattern, we could easily add that feature to our page with minimal changes to the existing code.

C2: Design Principles

C2.1: Single Responsibility Principle

We used the Single Responsibility Principle when coding our application. Each component should only be responsible for one function, and one function only. Every one of our classes has its own focused responsibility. We have multiple unique classes that performs specific functions rather than similar classes with overlapping responsibility.

C2.2: Low Coupling and High Cohesion

Each of our classes are independent of one another as much as possible, there are little to no dependencies between each class and they work together closely to drive our application.

Appendix D: Software Engineering Practices

D1: Clear Requirement Elicitation

By clearly identifying our target group, it gave us a demographic that we can work with for our project. This is important as client involvement is crucial in requirement elicitation.

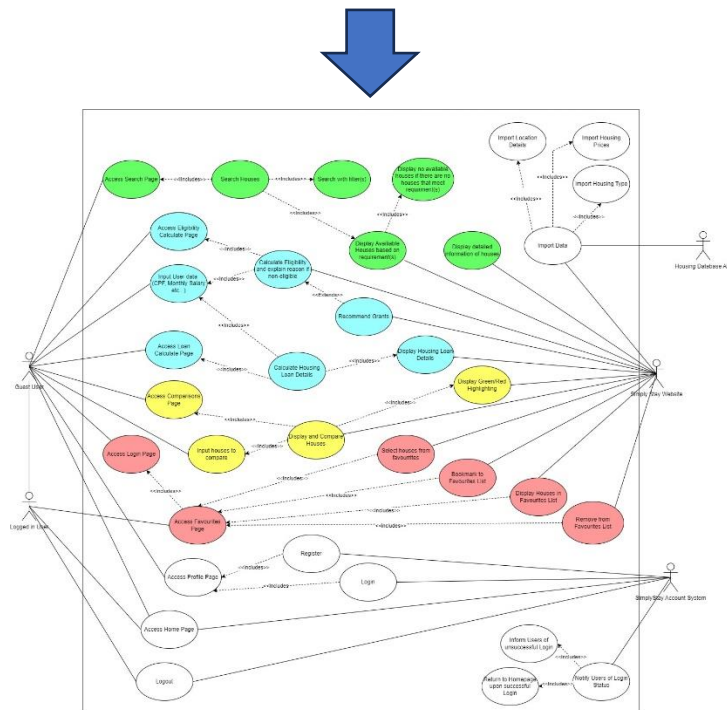
From there, we created the required functionalities of our app and translated them into use case diagrams and description which clearly lays out the requirements that we need to fulfill.

2.2.2 Search Page

- Users can search for houses by different filters such as price, location, and housing type
- The page will return all the houses that satisfy the user's chosen filters
- Logged in users can choose to favourite the resulting houses

2.2.3 Eligibility Calculator Page

- Users can input their age, marital status, gross monthly income, monthly expenditure, CPF, and savings into the eligibility calculator.
- The page will recommend eligibility for a BTO, and payment plans based on the user's input

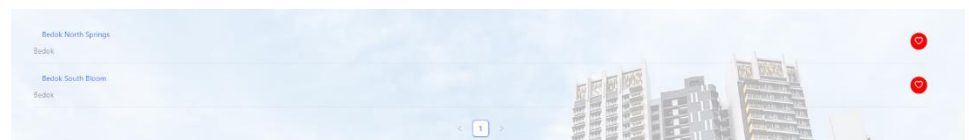
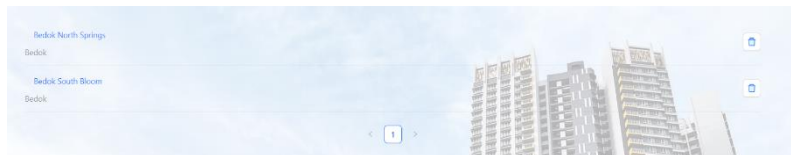


D2: Reuse-oriented Software Engineering

By using components of previously existing software such as Google Maps and Ant Design, we were able to save a lot of time and effort coding our application, enabling us to focus on the useful features rather than creating a website from scratch. This boosted our efficiency by a significant margin than if we started the website from a blank template.

```
import { List, Modal, Button, Empty, Result, Typography, notification } from 'antd';
```

```
<List
  pagination={{
    position,
    align,
    defaultPageSize: 5
  }}
  dataSource={flat}
  renderItem={flat => (
    <List.Item>
      <List.Item.Meta
        title=<Button type='link' onClick={()=>showModal(flat)}>{flat.name}</Button>
        description={flat.location}
      />
      <Button
        icon=<DeleteTwoTone />
        onClick={()=>clickdeletelast(flat)} />
    </List.Item>
  )}
/>
<Modal
  open={open}
  title={currentflat.name}
  onOk={handleOk}
  onCancel={handleOk}
  footer={
    <Button
      key="OK"
      onClick={handleOk}
    >
    OK
  </Button>
  </Modal>
```



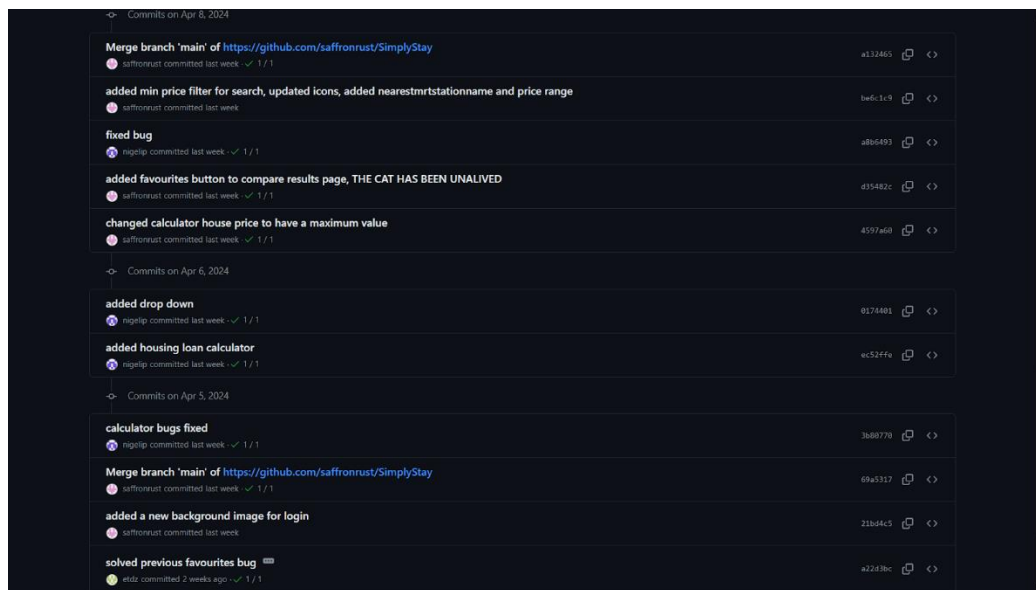
D3: Agile Manifesto

We constantly updated and refined our class diagrams and sequence diagrams throughout our project, all while working on our source code. This methodology is highly suitable for us

as we are a small project team with a tight deadline to meet. This allows for rapid software development that reduces process overheads by minimizing documentation and producing high-quality code.

D4: Version Control

Version control was practiced using Github, which provides us source control and a collaboration environment that prevents code conflicts.



Whenever we wished to implement new features, we would create a separate branch from the main branch to test it out. This isolated any bugs and issues that could arise that could potentially cripple the main code. When testing of the new feature was completed, a pull request would be made to merge it back into the main branch. This ensured that implementation of new code was seamless and efficient.

