Software Requirements Specification

for

HouseMatchers

Version 1.4 approved

Prepared by GroupThree

Chryston Boo (U1840555B), Atrik Das (U1823950C), He Wanru (U1820328A), Chandrasekhar Aditya (U1923951A), Lim Jun Sheng Zachary (K2020205K)

Nanyang Technological University, Singapore

7 September 2020

Table of Contents

Revision History

	Date	Reason For Changes	Version
	07/09/2020	Functional & Non Functional Requirements Use case Diagram & Description Data Dictionary Initial UI Mockup	1.0
	22/09/2020	Class Diagram DIalog Map Sequence Diagram	1.1
	27/10/2020	Class Diagram edits System Architecture UI mockup edits Use Case edits (Non) functional requirements edits	1.2
	4/11/2020	Introduction edits Overall Description edits External Interface edits	1.3
	10/11/2020	Test Cases Appendix	1.4

1. Introduction

1.1 Purpose

HouseMatchers is a web-based resale HDB flat recommendation tool that provides customers with personalised recommendations based on their inputs, as well as offer data visualisation for resale HDB flat data in Singapore. This document outlines the plan for the development of HouseMatchers. The intended readers of this document are current and future developers of HouseMatchers. The plan consists, but is not limited to, a summary of the system functionality, the functional and non-functional requirements of the system, use case diagram and use case models, user interface, and data-flow diagrams (Class diagram, sequence diagram, dialog map, system architecture diagram).

1.2 Document Conventions

This document features the usage of the font Arial of size 12. It must be noted that each new header uses Arial of size 19 and is stylized to be bold. Each subheading uses Arial of size 15 and set to be stylized in bold. In numbering our detailed requirements and use cases we employ a nested (indented) numbering system whereby a title numbered 1 has subheadings 1.1, 1.2 and so on.

1.3 Intended Audience and Reading Suggestions

This document is intended for the following types of readers:

1. Project managers
2. Developers involved in the maintenance application or fixing existing bugs.
3. Quality Assurance (QA) Engineers who are designing test cases and performing tests on the application

Developers should start by reading the Introduction, the Overall Description and the rest of the document consecutively, including the diagrams in the appendix.

QA Engineers should focus on the System Features and Non-Functional Requirements for system testing and requirements verification.

1.4 Product Scope

HouseMatchers will be an application that would facilitate customers to choose a suitable resale HDB flat. The information provided by HouseMatchers would act as a supplementary guideline for customers to make an informed choice for their purchase.

2. Overall Description

2.1 Product Perspective

HouseMatchers aims to provide customers with additional information when making a HDB resale flat purchase decision. Currently, property websites such as propertyguru.com and 99.co provide customers with options for available resale HDB flats. However, the price of that said listing is dependent on the property agent and may not reflect the value of the HDB resale flat. With HouseMatchers, customers may look up on resale HDB flat data of similar properties before purchasing. Furthermore, HouseMatchers can also be used as a preliminary research for narrowing housing options for many customers.

2.2 Product Functions

HouseMatchers must contain the following functionality:

1. Customers must be able to create an account with their name, email address and password.
2. Customers must be able to login using their email address and password. The system must prompt the customer to register for an account if the customer does not possess an account.
3. Upon successful login, the system will redirect the user to the Home page. The customer is then able to navigate to the Home, Recommend, Visualisation and Contact tabs at any time.
4. When the customer navigates to the Recommend tab, the system must display a form containing price, flat type, storey, remaining lease and location. Upon successful submission of the form, the customer will be redirected to the results page where there are 5 recommendations provided by the system. The customer may then explore more details for the recommendations or return to resubmit the form again.
5. When the customer navigates to the Visualisation tab, the system will display the bar graph of HDB resale flat data. The customer may also view a heatmap of HDB prices based or a timeseries for prediction of HDB resale flat pricing over the next 1 year.
6. When the customer navigates to the Contact tab, the customer may submit feedback to the developers.

2.3 User Classes and Characteristics

HouseMatchers will be mainly used by customers who are looking to purchase a HDB resale flat. HouseMatchers may also be used by property agents to aid them in valuation of HDB resale flats. Lastly, real estate investors may use HouseMatchers to visualise market trends in HDB resale flat prices.

2.4 Operating Environment

The software will be designed to operate on Windows, Linux and Mac OS.

2.5 Design and Implementation Constraints

MongoDB has been selected to be used as the database of the website. HouseMatchers is not affiliated with MongoDB in any way. The developers of this website are not liable for any risks associated with storing user data on MongoDB. Any attempts on hacking the database or any data breach must not be traced back to the developers.

The website is hosted on the free tier provided by Heroku and hence any traffic load above the maximum capacity allowed on Heroku may result in lag or other issues.

The website is currently only available in English.

2.6 User Documentation

The website is designed to be intuitive and user friendly. No other documentations are currently available.

2.7 Assumptions and Dependencies

The recommendation system of HouseMatchers is based on the resale flat prices obtained from the API from https://data.gov.sg/dataset/resale-flat-prices. The reliability and accuracy of the recommendations depends on the called API. The data might also be outdated as the API only includes data until September 2020. Disclaimer is advised for customers using HouseMatchers.

The Google Map of HouseMatchers is based on the Google Maps API from https://cloud.google.com/maps-platform/maps. This is used to get the user input of location and is used as part of our recommendation system.

Users must be connected to the internet with a decent signal in order to access the website. Depending on the strength of the internet, some images may not decide to load due to bandwidth constraints.

3. External Interface Requirements

3.1 User Interfaces

The User Interface (UI) of HouseMatchers aims to follow good design principles. Initial UI mockups were constructed using Figma (88.1.0). The website aims to strive for consistency for terminology, fonts and other UI design elements. The color scheme of the website is also not too striking to reduce visual fatigue of the user. The website was also designed using guidelines of material design.

3.1.1 Login & Signup

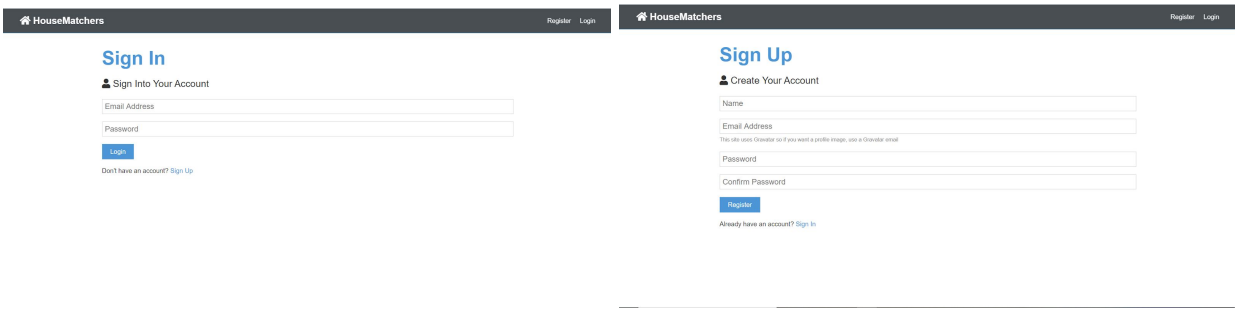

Figure 1. Login & Signup Page

Figure 1 shows the login and signup page. The system provides informative feedback to the user by displaying a corresponding error message for any invalid text field.

3.1.1 Home Tab

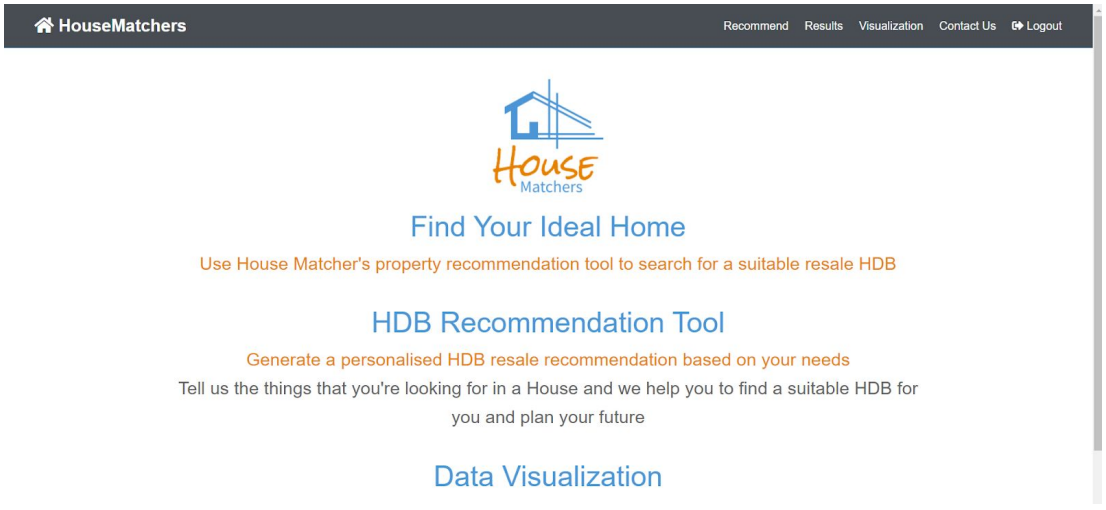

Figure 2. House Matchers Home Tab

Figure 2 shows the home tab of HouseMatchers. The layout of the website is kept simple with the navigation buttons located on the top.

3.1.2 Recommend Tab

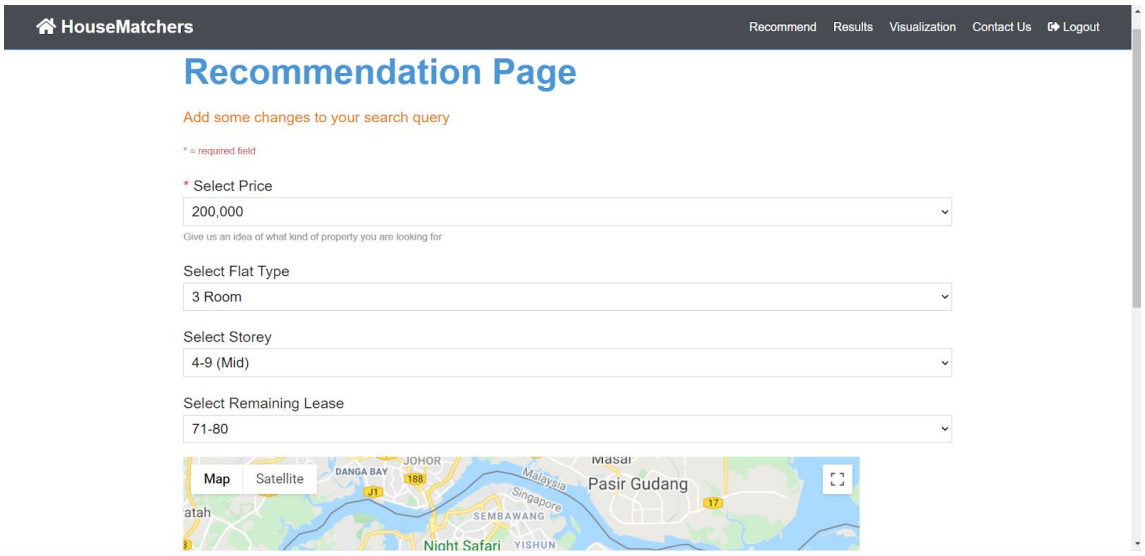

Figure 3. Parameter Input Page for Recommend Tab

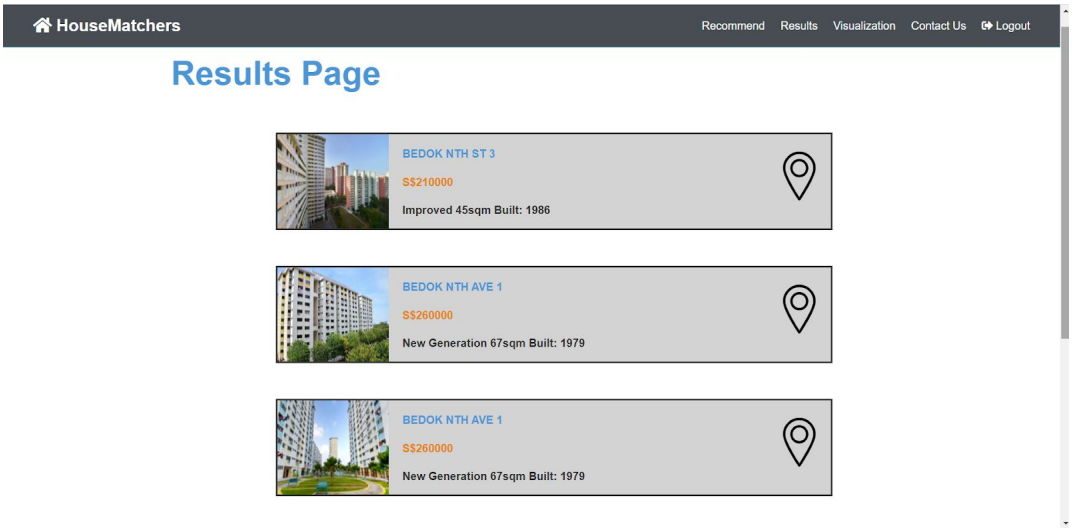

Figure 4. Results for Recommend Tab

Figure 3 shows the parameter input page which can be accessed by clicking on the recommend tab after successful login. Customers can indicate their desired price, flat type, storey, remaining list and location through the drop down menu. By having a drop down menu, it reduces input error by the customers as compared to a text input.

The entered parameters would then be used to provide recommendations as seen in Figure 4. Customers can update the results page by submitting the form again, permitting easy reversal of actions.

3.1.3 Visualisation Tab

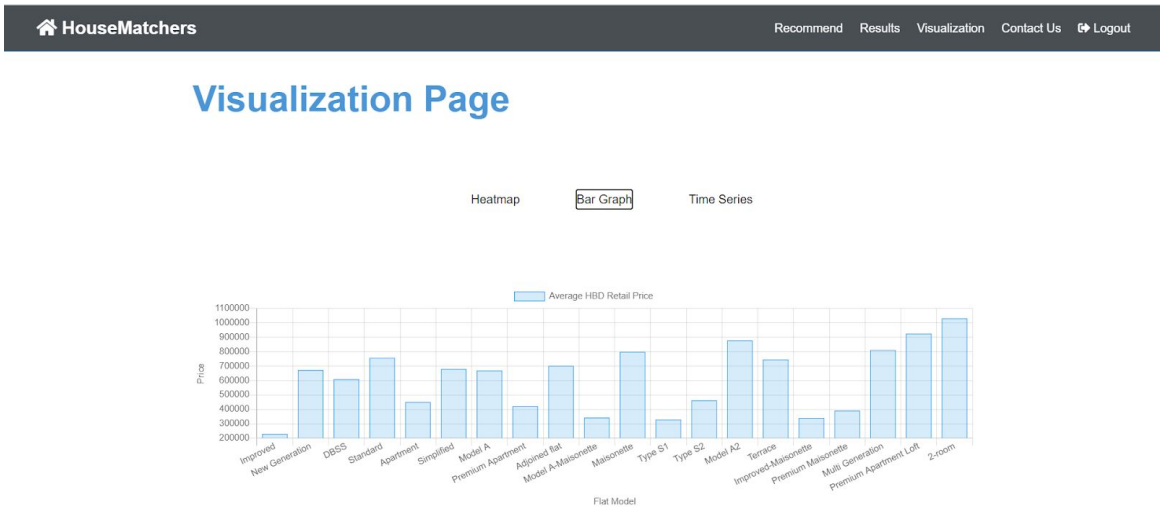

Figure 5. Visualisation Tab

Figure 5 shows the visualisation tab which can be accessed by clicking on the visualisation tab after successful login. Customers can select on the visualisation mode they want to view, supporting internal locus of control. Three types of visualisation available for display are: heatmap of HDB prices, bar graph of HDB data and time series of HDB prices.

3.1.4 Contact Tab

Figure 6. Contact Tab

Figure 6 shows the contact tab which can be accessed by clicking on the visualisation tab after successful login. Customers can send feedback regarding bugs to the developers through this tab.

3.2 Hardware Interfaces

The website is designed to be responsive and functional on most screen sizes, ranging from large television screens to smaller mobile phone screens.

3.3 Software Interfaces

This website has only been tested on Google Chrome. Running this website on other browsers such as Firefox or Safari may yield different results as javascript behaves differently based on the browser.

3.4 Communications Interfaces

The contact page uses the Nodemailer node module to send emails. It uses the Simple Mail Transfer Protocol (SMTP) communication protocol for mail transmission and it focuses heavily on security to minimize chances of interception during the transport.

4. Functional Requirements

4.1 Registration

4.1.1 The system must allow the user to create an account.

4.1.1.1 The system must allow the user to fill in his/her name, email address, password and confirmation password.

4.1.1.1.1 The system must validate that all the text fields are filled up.

4.1.1.1.2 The system must validate that the email address provided is valid.

4.1.1.1.3 The system must validate that the email address has not already been registered.

4.1.1.1.4 The system must validate if the password and confirmation password contains at least 6 characters respectively.

4.1.1.1.5 The system must validate the password matches the confirmation password.

4.1.1.2 The system must redirect the user to the home tab upon successful account creation.

4.2 Login

4.2.1 The system must provide a separate login entry for the customer and admin.

4.2.1.1 The system must allow the customer to login to the customer domain.

4.2.1.2 The system must allow the admin to login to the admin domain.

4.2.2 The system must contain a link to redirect the customer to the registration page.

4.2.3 The system must be able to validate the email and password text fields.

4.2.3.1 The system must be able to validate that the email address text field is filled.

4.2.3.2 The system must be able to validate that the password text field is filled.

4.2.3.4 The system must be able to validate that the email address is a valid email address.

4.2.3.3 The system must display the corresponding error message when any of the above mentioned validation fails.

4.2.4 The system must be able to validate the customer and admin's credentials with the database before letting them use the customer and admin domain respectively.

4.2.4.1 The system must display an error message when the email is not registered.

4.2.4.2 The system must display an error message when the password does not match with the customer and admin database.

4.2.5 The system must redirect customer/admin to the home tab after successful login.

4.3 Customer Domain

4.3.1 The customer domain must have the following components - View Data Analytics, User Recommendation System and Send Email.

4.3.1.1 View Data Analytics must be found in the Visualise tab

4.3.1.1.1 The system must be able to display the heatmap of regions in Singapore based on prices of the resale HDB flats.

4.3.1.1.2 The system must be able to display a bar graph with price on the y-axis and flat type on the x-axis.

4.3.1.1.3 The system must be able to display a time series graph to show the predicted value of a HDB resale flat in the next 1 year.

4.3.1.2 User Recommendation System must be found in the Recommend tab.

4.3.1.2.1 The system must allow the customer to select his/her desired price range, flat type, storey, remaining lease and location.

4.3.1.2.1.1 The system must allow the customer to select a price range for the price of HDB resale flat from $100,000 - $800,000 with every step increasing by $100,000.

4.3.1.2.1.2 The system must allow the customer to select a flat type from 1 room, 2 room, 3 room, 4 room, 5 room and Executive.

4.3.1.2.1.3 The system must allow the customer to select a storey range from Any, 1-3 (Low), 4-9 (Mid) and ≥10 (High).

4.3.1.2.1.4 The system must allow the customer to select a remaining lease range from Any, 40-50, 51-60, 61-70, 71-80, 81-90 and ≥91.

4.3.1.2.1.5 The system must allow the customer to select a location from the provided Singapore map.

4.3.1.2.2 The system must validate that all the parameters stated above are successfully filled up before providing a recommendation.

4.3.1.2.3 The system must display an error message if any of the above parameters are not filled up.

4.3.1.2.4 The system must save the data mentioned in 4.3.1.2.1 to the database.

4.3.1.2.4 The system must provide the customer with 5 recommendations using the parameters mentioned in 4.3.1.2.1 if available.

4.3.1.2.4 The system must provide the customer with an error message if there are no recommendations to be provided.

4.3.1.2.5 The system must allow the customer to edit the parameters mentioned in 4.3.1.2.1.

4.3.1.3 Send Email must be found in the Contact tab.

4.3.1.3.1 The system must provide the customer with a platform to contact the admin for suggestions, feedback and bug reporting.

4.3.1.3.1.1 The customer must write their own email address in the email box.

4.3.1.3.1.2 The customer must write their own message required in the message box.

4.3.1.2 The system must be able to validate that all the text fields are filled up before submitting.

4.3.1.3 The system must be able to validate the email address entered by the customer is valid.

4.3.1.4 The system must display the corresponding error message if the above-mentioned validation fails.

4.4 Admin Domain

4.4.1 The admin must have the following components - View Customer Information, Delete Customer

4.4.1.1 View Customer Information

4.4.1.1.1 The admin must be able to view a list of registered customers.

4.4.1.1.2 The admin must be able to view registered customers activity login in the admin panel.

4.4.1.2 Delete Customer

4.4.1.2.1 The admin must be able to delete a customer from the system.

4.4.1.2.2 The customer must not be able to login after account deletion.

4.5 Use Case Diagram

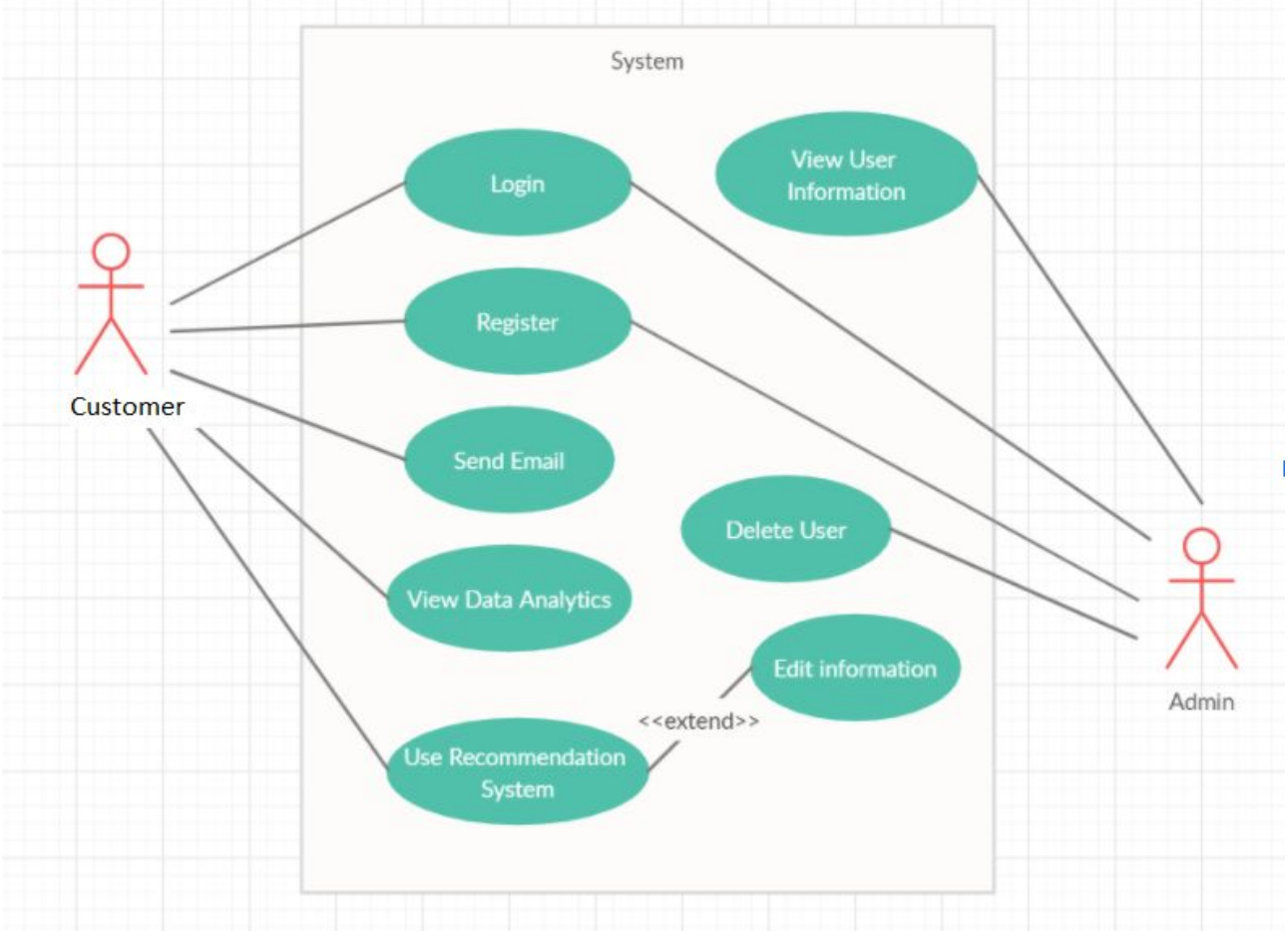

Figure 7. Use Case Diagram for HouseMatchers

4.6 Use Case Descriptions

primary actors	use cases
Customer	1.0 Register 2.0 Login 3.0 Use Recommendation System 3.1 Edit Information 4.0 View Data Analytics 5.0 Send Email
Admin	1.0 Register 2.0 Login 6.0 View Customer Information 7.0 Delete Customer

Use Case ID:	1.0		
Use Case Name:	Register		
Created By:	Atrik Das	Last Updated By:	
Date Created:	06/09/2020	Date Last Updated:	

Actor:	(User) Customer and Admin
Description:	Create account to the website for the first time
Preconditions:	Valid email address
Postconditions:	User account created
Priority:	
Frequency of Use:	
Flow of Events:	1. (User) User selects the 'Register Now' button on the 'Login' page and fills in the name, email address and password and password confirmation fields required on the 'Registration' page. 2. (System) System checks if all the fields are filled. 3. (System) System checks if the provided email address is valid. 4. (System) System checks if the password contains at least 6 characters. 5. (System) System checks if the confirmation password provided matches the password. 6. (System) System checks if the email address provided has not been registered before. 7. (System) System creates a user account and displays output that the account has been successful. 8. (System) System redirects the user to the Home tab.
Alternative Flows:	AF-1: If any of the text fields are not completed. 1. System highlights the empty fields and displays the message "X field is required", where X is the name of the empty textfield. AF-2: If the provided email address is not valid 1. System highlights the email field and displays the message "Email is invalid". AF-3: If the provided password is less than 6 characters long.

	1. System highlights the password field and displays the message “Password must be at least 6 characters”. AF-4: If the provided password does not match the confirmation password. 1. System highlights the email field and displays the message “Passwords must match”. AF-5: If the email address specified is already connected to an existing user. 1. System displays the message “Email already exists”.
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

Use Case ID:	2.0		
Use Case Name:	Login		
Created By:	Atrik Das	Last Updated By:	
Date Created:	06/09/2020	Date Last Updated:	

Actor:	(User) Customer and Admin
Description:	Login to the user's account
Preconditions:	Valid email address User account created
Postconditions:	User logs in
Priority:	
Frequency of Use:	
Flow of Events:	1. (User) User selects the "Login" button on the "Login" page 2. (User) User keys in his valid email address and password 3. (System) If all information is valid, system logs the user in.
Alternative Flows:	AF-1: If any of the text fields are not completed. 1. System highlights the empty fields and displays the message "X field is required", where X is the name of the empty textfield. AF-2: If the email address is not valid 1. System displays the message "Email not found". AF-3: If the password does not match with the database 1. System displays the message "Password is incorrect" AF-3: If the user is an Admin 1. System logs the user into the Admin panel
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

Use Case ID:	3.0		
Use Case Name:	Use Recommendation System		
Created By:	Atrik Das	Last Updated By:	
Date Created:	06/09/2020	Date Last Updated:	

Actor:	Customer
Description:	
Preconditions:	Customer logged in
Postconditions:	Customer's information gets saved in the database
Priority:	
Frequency of Use:	
Flow of Events:	1. (User) Customer navigates to the 'Recommend' tab 2. (User) Customer selects the dropdown for all the required fields of the form ('Price', ''Flat type', 'Storey' and 'Remaining Lease') 3. (User) Customer selects the 'Location' from the map. 4. (User) Customer clicks on the 'Submit' button. 5. (System) System saves the data then redirects to a new page with the top 5 recommendations based on the user's inputs.
Alternative Flows:	AF-1: If any of the fields are not completed 1. System highlights empty fields and displays the message "Please complete all required fields".
Exceptions:	EX-1: If there are no relevant properties based on the customer's inputs: 1. Display the message "There are no properties to show based on your inputs. Please try to expand your search criteria".
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

Use Case ID:	3.1		
Use Case Name:	Edit Information		
Created By:	Atrik Das	Last Updated By:	
Date Created:	06/09/2020	Date Last Updated:	

Actor:	Customer
Description:	Edit Customer's search criteria
Preconditions:	Customer logged in; Customer already has submitted their search criteria
Postconditions:	Customer's edited information gets saved in the database
Priority:	
Frequency of Use:	
Flow of Events:	1. (User) Customer clicks the 'Edit' button. 2. (System) System displays the parameter input page, auto-filling with the customer's previous input. 3. (User) Customer edits his/her desired parameter(s) and presses the 'Submit' button. 4. (System) System saves the data then redirects to a new page with the top 5 recommendations based on the user's new inputs
Alternative Flows:	AF-1: If any of the fields are not completed 1. System highlights empty fields and displays the message "Please complete all required fields".
Exceptions:	EX-1: If there are no relevant properties based on the customer's inputs: 1. Display the message "There are no properties to show based on your inputs. Please try to expand your search criteria".
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

Use Case ID:	4.0		
Use Case Name:	View Data Analytics		
Created By:	Atrik Das	Last Updated By:	
Date Created:	06/09/2020	Date Last Updated:	

Actor:	Customer
Description:	Display various graphs and charts
Preconditions:	User logged in
Postconditions:	
Priority:	
Frequency of Use:	
Flow of Events:	1. (User) Customer clicks on the “Visualisation” tab. 2. (System) System displays a bar graph to visualise the bar graph for price of resale HDB flats. 3. (User) Customer clicks on “Heatmap” to visualise a heatmap of resale HDB prices. 4. (User) Customer clicks on “Time Series” to visualise time series analysis on the prices of resale HDB flats and prediction in the next 1 year.
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

Use Case ID:	5.0		
Use Case Name:	Send Email		
Created By:	Atrik Das	Last Updated By:	
Date Created:	06/09/2020	Date Last Updated:	

Actor:	Customer
Description:	Contact the people maintaining this website
Preconditions:	
Postconditions:	An email is sent to the maintainers
Priority:	
Frequency of Use:	
Flow of Events:	1. (User) Customer clicks on the “Contact” tab. 2. (User) Customer fills in the ‘email’ and ‘message’ fields. 3. (System) System sends an email from the user’s email to the maintainer’s email containing the message. 4. (System) System displays a success message.
Alternative Flows:	AF-1: If the email address is incorrect. 1. System displays the message “Please input a valid email ID”. AF-2: If some fields are not completed. 1. System highlights empty fields and displays the message “Please complete all required fields”.
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

Use Case ID:	6.0		
Use Case Name:	View Customer Information		
Created By:	Atrik Das	Last Updated By:	
Date Created:	06/09/2020	Date Last Updated:	

Actor:	Admin
Description:	Ability to view Customers in admin panel
Preconditions:	Must login with admin credentials
Postconditions:	
Priority:	
Frequency of Use:	
Flow of Events:	1. (User) Admin can view the email and name of all the Customers currently registered in the website.
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

Use Case ID:	7.0		
Use Case Name:	Delete Customer		
Created By:	Atrik Das	Last Updated By:	
Date Created:	06/09/2020	Date Last Updated:	

Actor:	Admin
Description:	Ability to delete Customers from admin panel
Preconditions:	Must login with admin credentials
Postconditions:	Deleted Customer cannot use their registered email to login anymore
Priority:	
Frequency of Use:	
Flow of Events:	1. (User) Admin clicks on the red cross next to the user they want to delete. 2. (System) System displays a dialog box asking, “Are you sure you want to delete this Customer?” with 2 buttons ‘Yes’ and ‘No’. 3. (User) Admin clicks ‘Yes’. 4. (System) System sends an email to the deleted Customer that their account has been deleted. 5. (System) System removes the Customer credentials from the database and the admin panel.
Alternative Flows:	AF1: If the Customer selects no, the Customers profile remains in the database
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

4.7 Data Dictionary

Term	Definition
Resale Price	The cost of the resale HDB flat where the buyer and seller agree upon. There are many factors determining the resale price of a HDB flat, such as the purchase date, the town which is situated at, the type of flat will be staying and the remaining lease of the date.
Month	The year and the month that an applicant will buy the resale HDB flat on.
Town	The location of the town where the applicant will buy the HDB resale flat at. Town can either be labelled as “mature” or “non-matured” and even the proximity to the nearest MRT station or shopping areas which can overall affect the price.
Flat type	The type of flat that the buyer is focusing on. A HDB flat is categorised into: 1 room, 2 rooms, 3 rooms, 4 rooms, 5 rooms, executive.
Flat Model	Model of the flats that the buyer is focusing on. For older HDB flats, it is subdivided into 1 or more models listed in the following such as Standard, New Generation, Improved, Simplified, Apartment, Model A, Flexi, Maisonette and 3 Gen. These models are built based on the year as well as the location it was built on.
Storey range	The floor in which the resale flat is at. The storeys ranges are in groups of three. e.g. 01 to 03, 04 to 06 and so on. A higher floor flat gives a higher price in selling the flat. The storey is divided as follows: Storey 01 to 03 is low, Storey 04 to 09 is mid and Storey 10 and above is high.
Lease Commencement Date	The starting date of the HDB flat when it was first leased to the applicants the first time. The usual lifespan of the HDB is 99 years and can be used to calculate the remaining years left on the property in determining the resale price.
Flat	Flats are subsidised public housing for Singaporeans. They are constructed by HDB.
HDB	HDB stands for “Housing and Development Board” which is SIngapore’s public housing authority and a statutory board under the Ministry of National Development. HDB is often used as a term to refer to the HDB flats.

4.8 Class Diagram

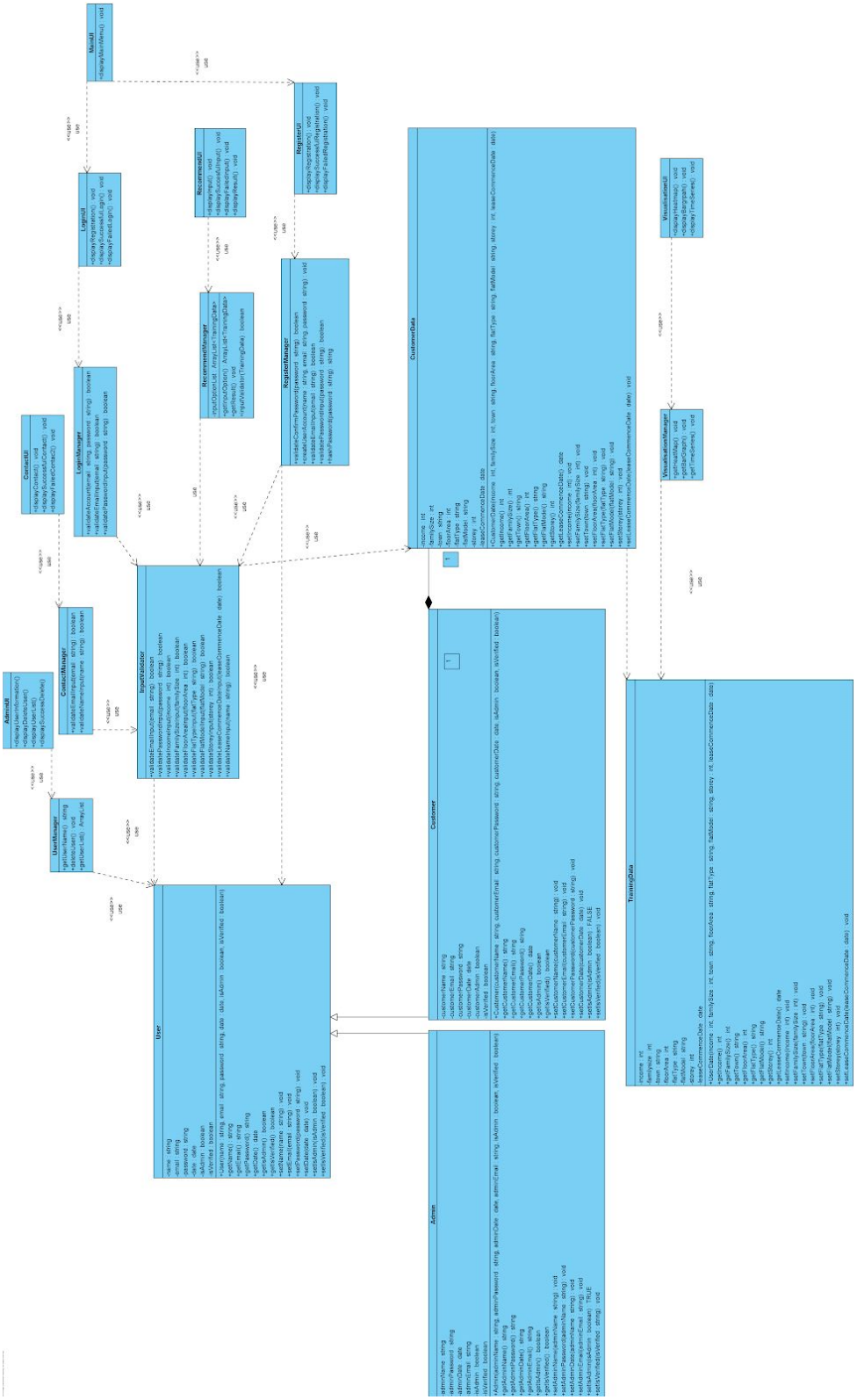

Fig 8. Class Diagram for HouseMatchers.

4.9 Dialog Map

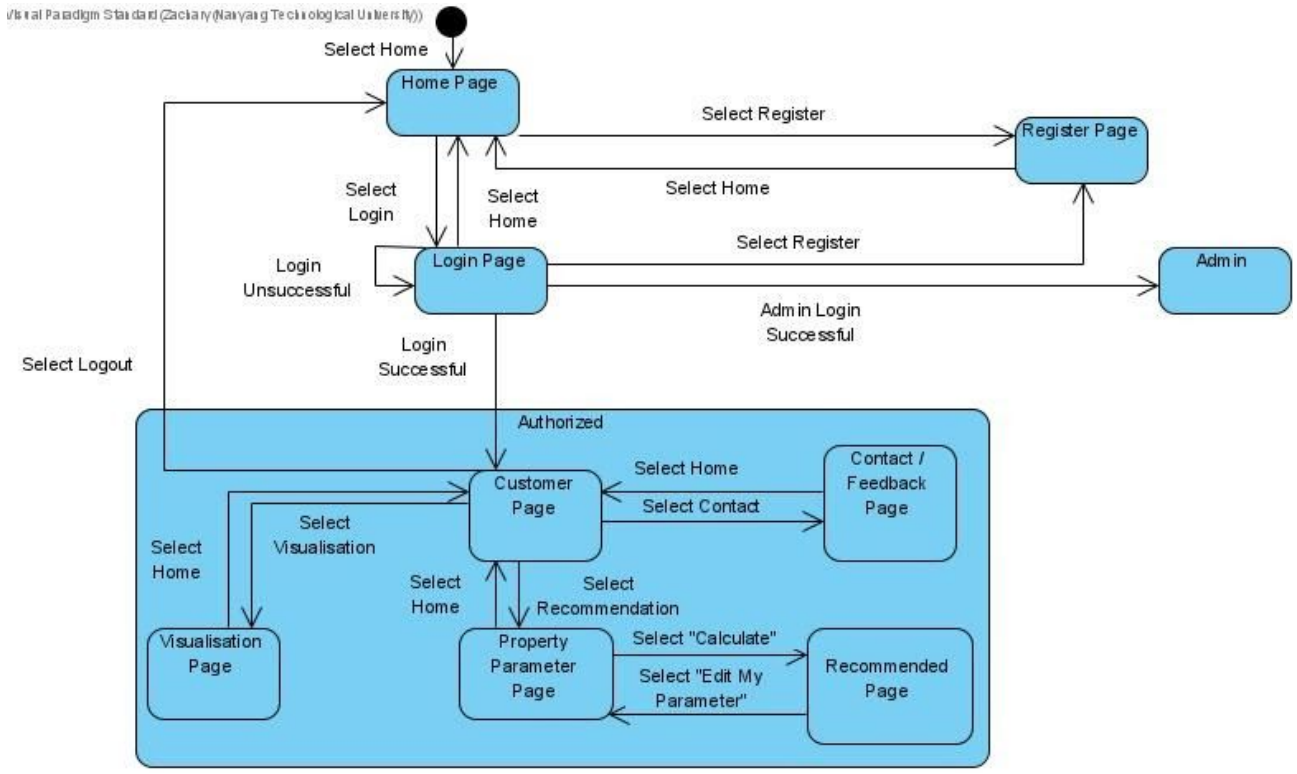

Fig 9. Dialog Map for HouseMatchers.

4.10 Sequence Diagrams

4.10.1 Register

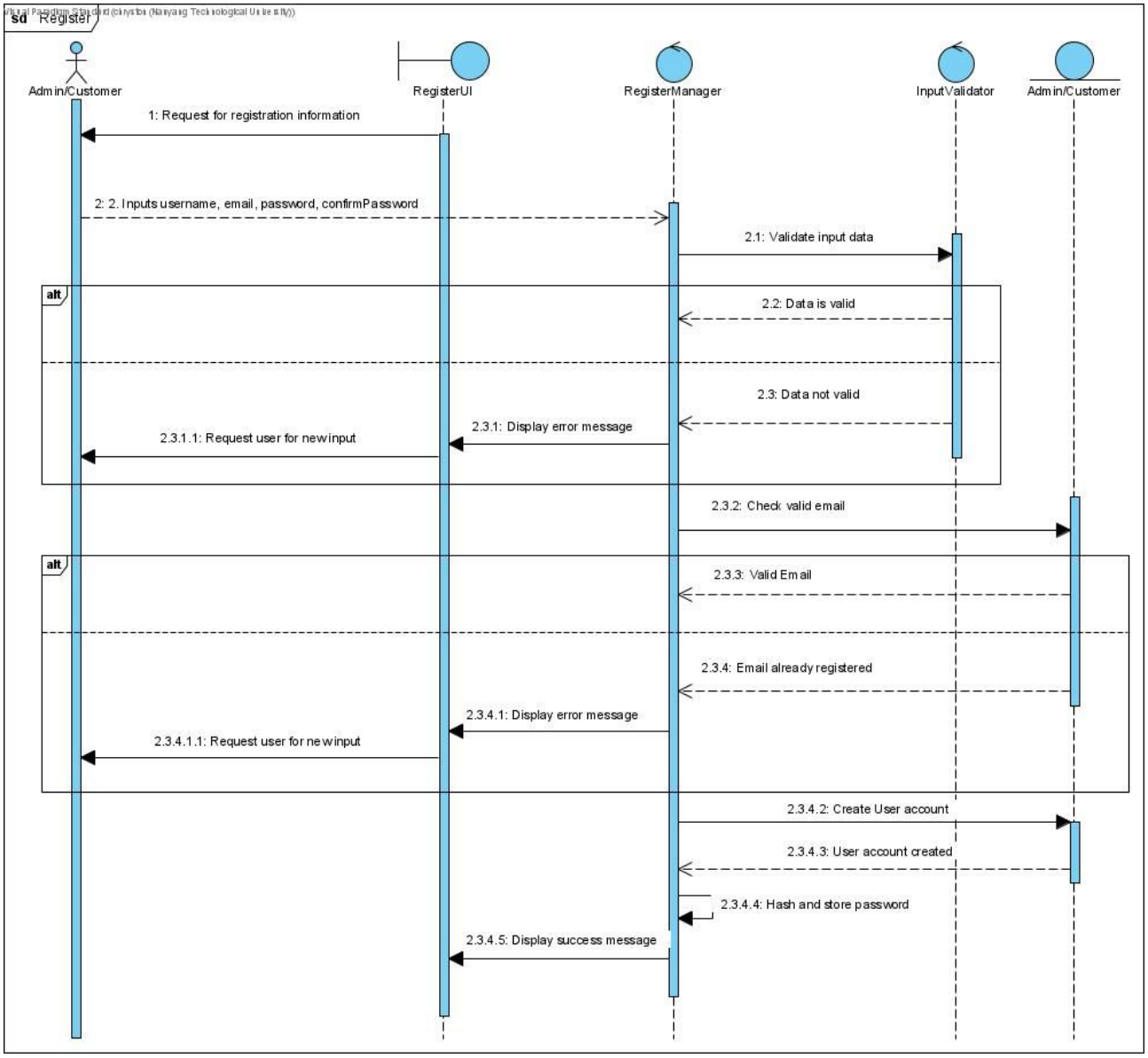

Fig 10. Sequence Diagram for HouseMatchers (Register).

4.10.2 Login

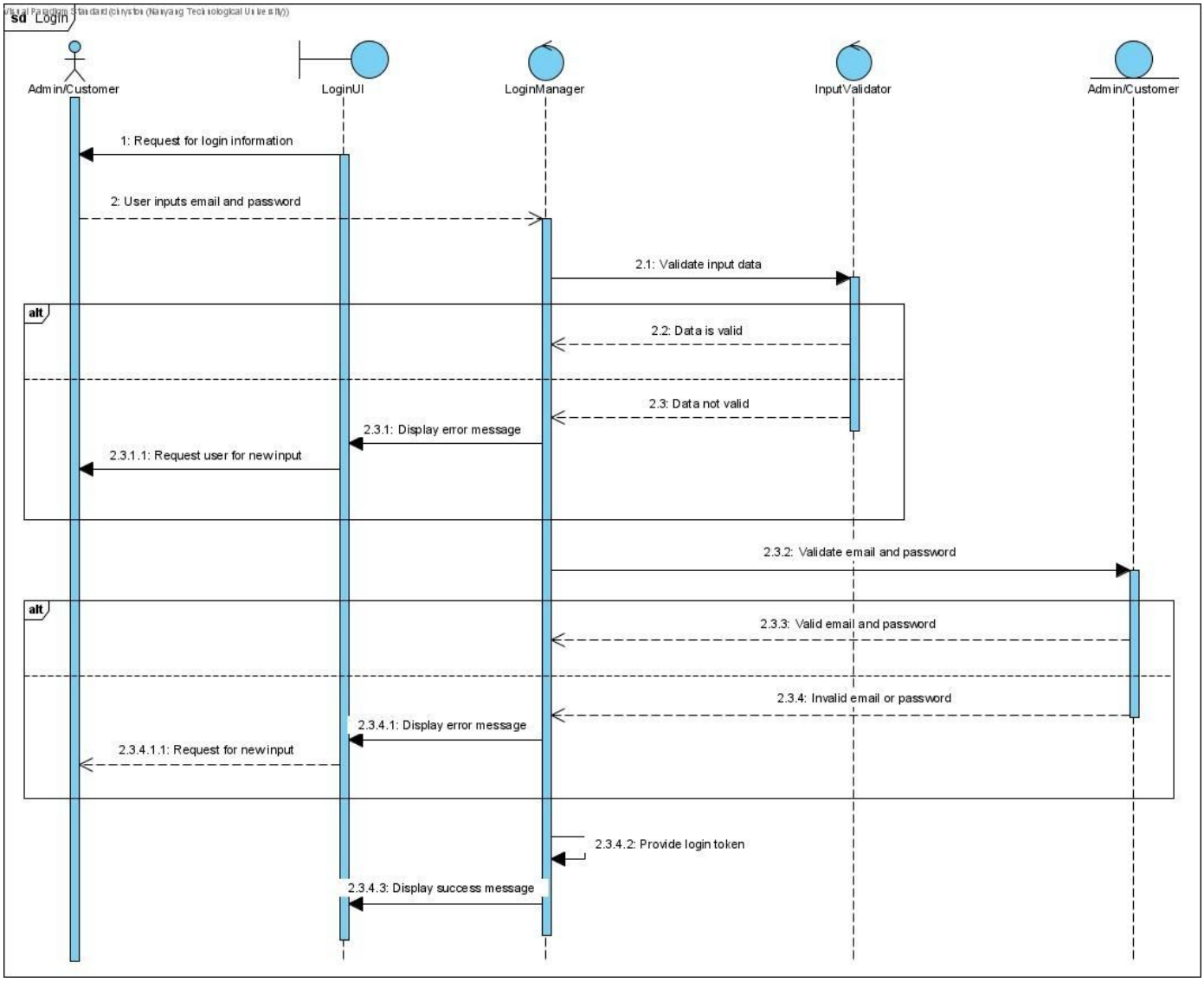

Fig 11. Sequence Diagram for HouseMatchers (Login).

4.10.3 View Data Analytics

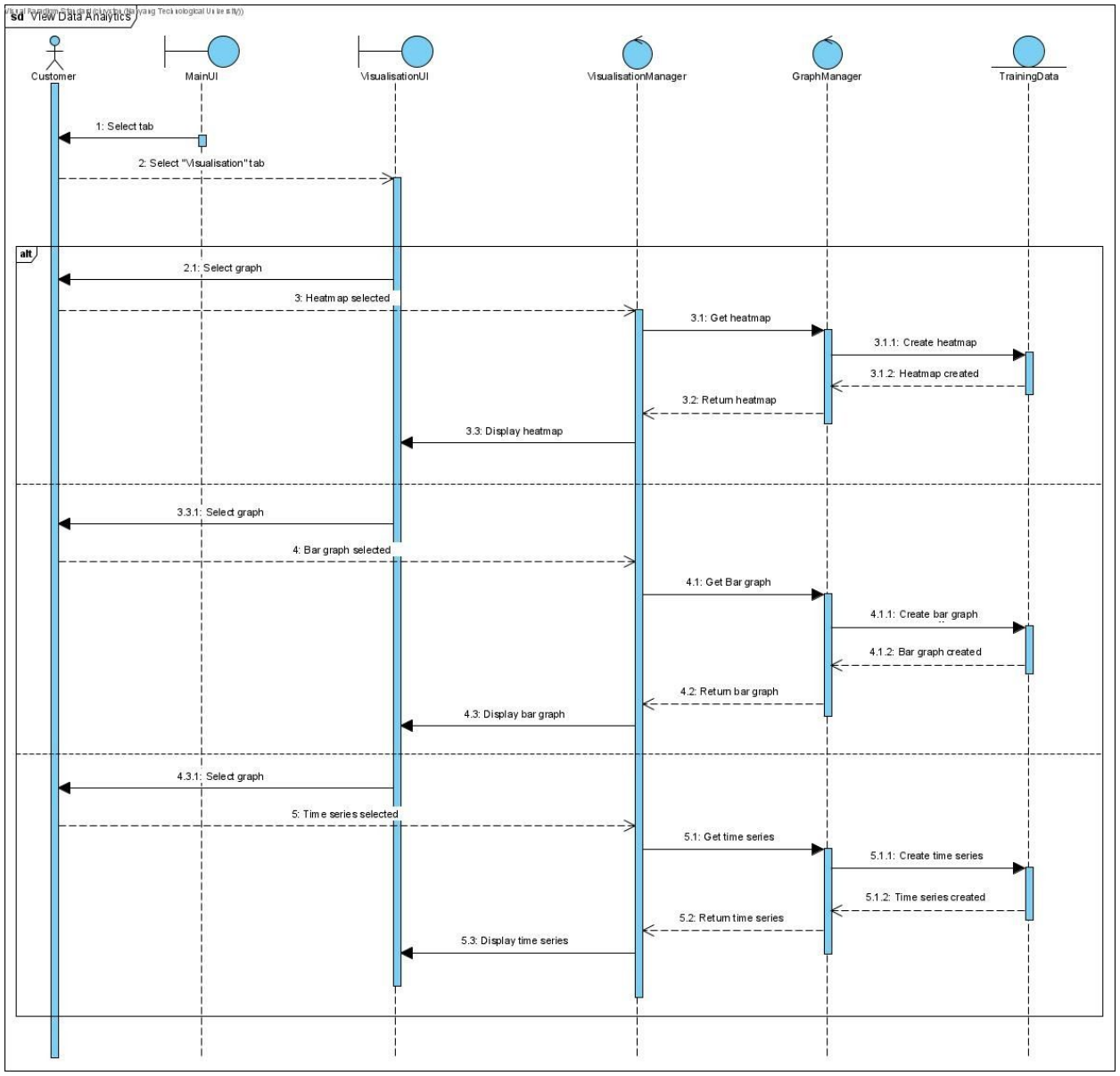

Fig 12. Sequence Diagram for HouseMatchers (View Data Analytics).

4.10.4 User Recommendation System

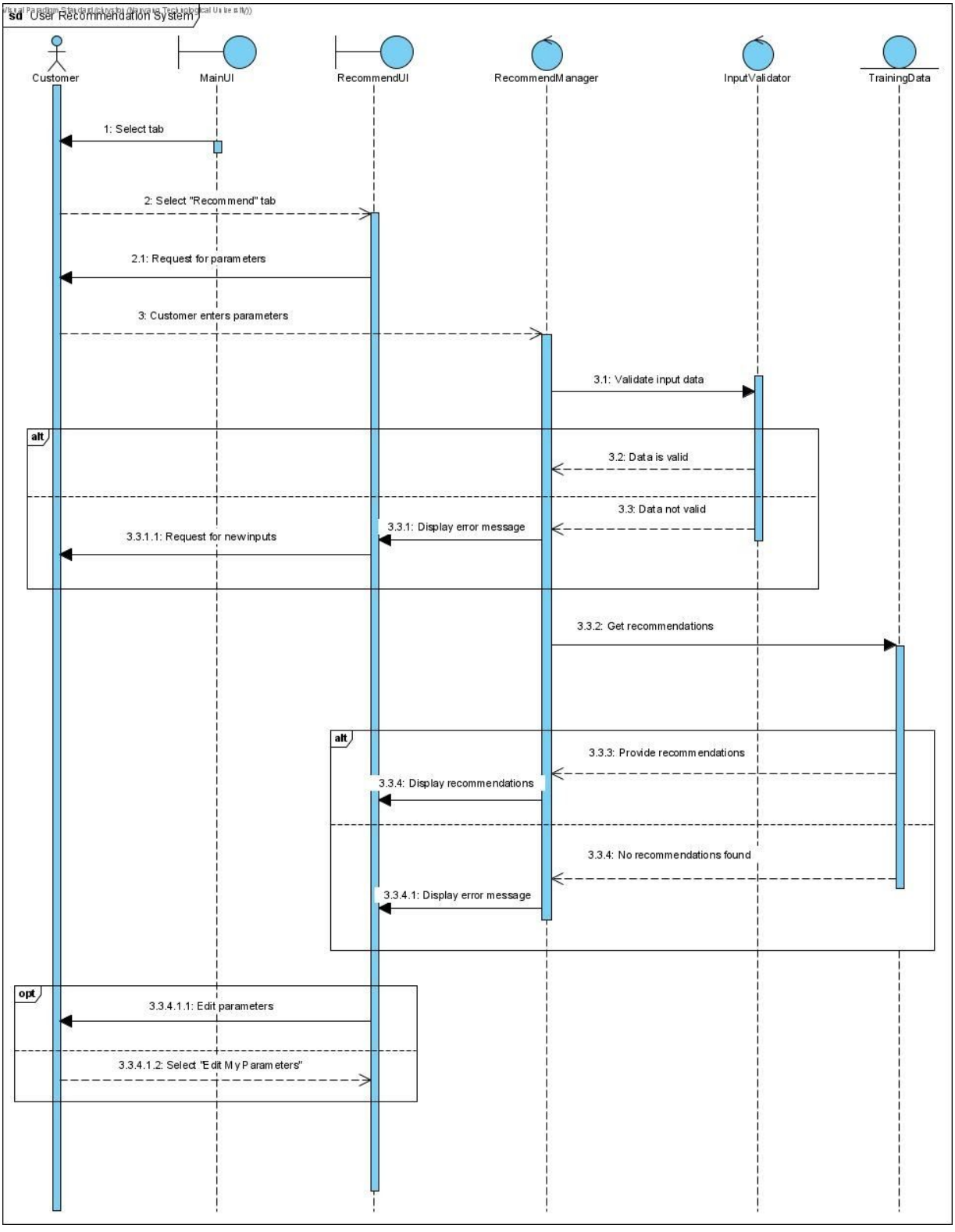

Fig 13. Sequence Diagram for HouseMatchers (User Recommendation System).

4.10.5 Send Email

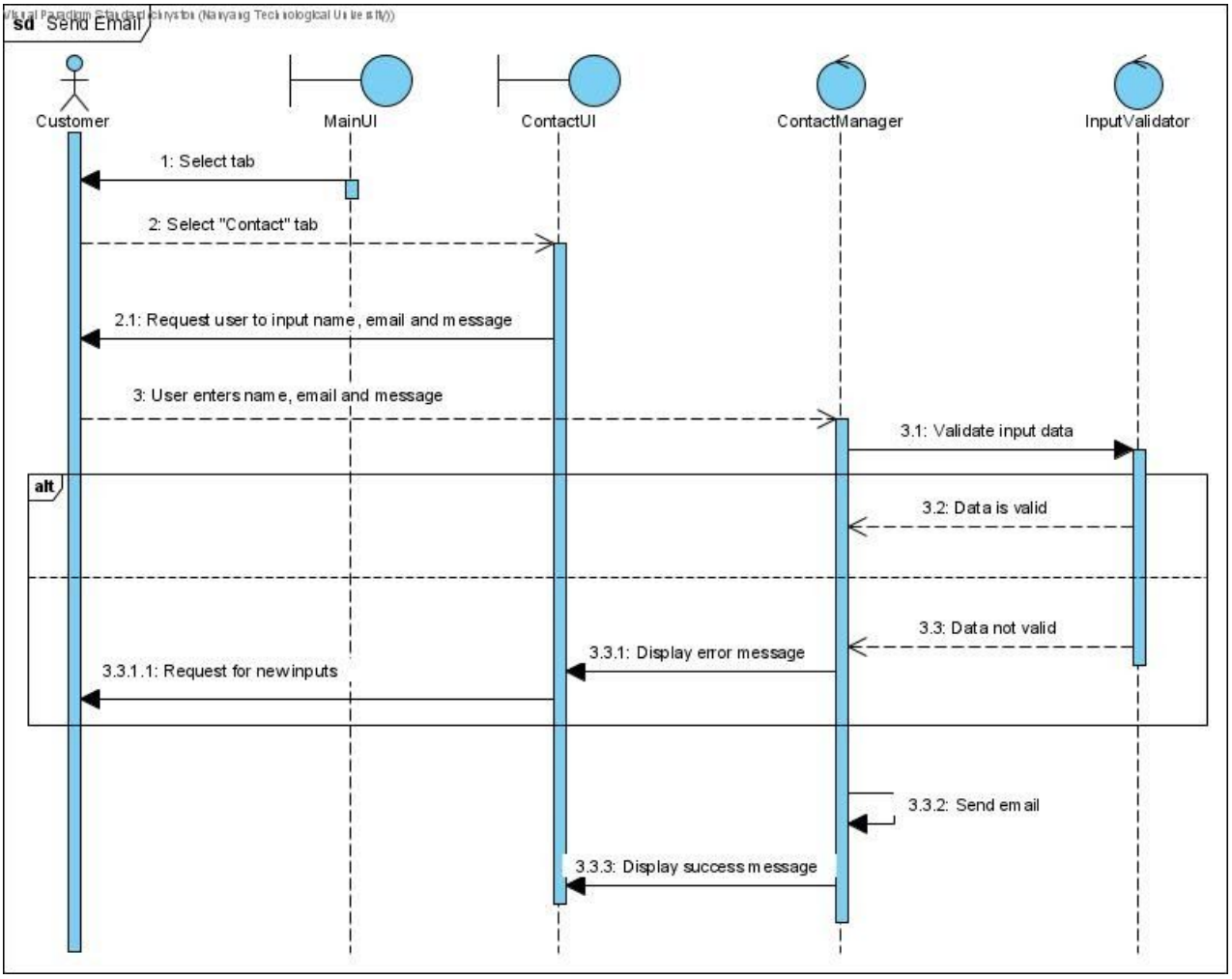

Fig 14. Sequence Diagram for HouseMatchers (Send Email).

4.10.6 Admin UI

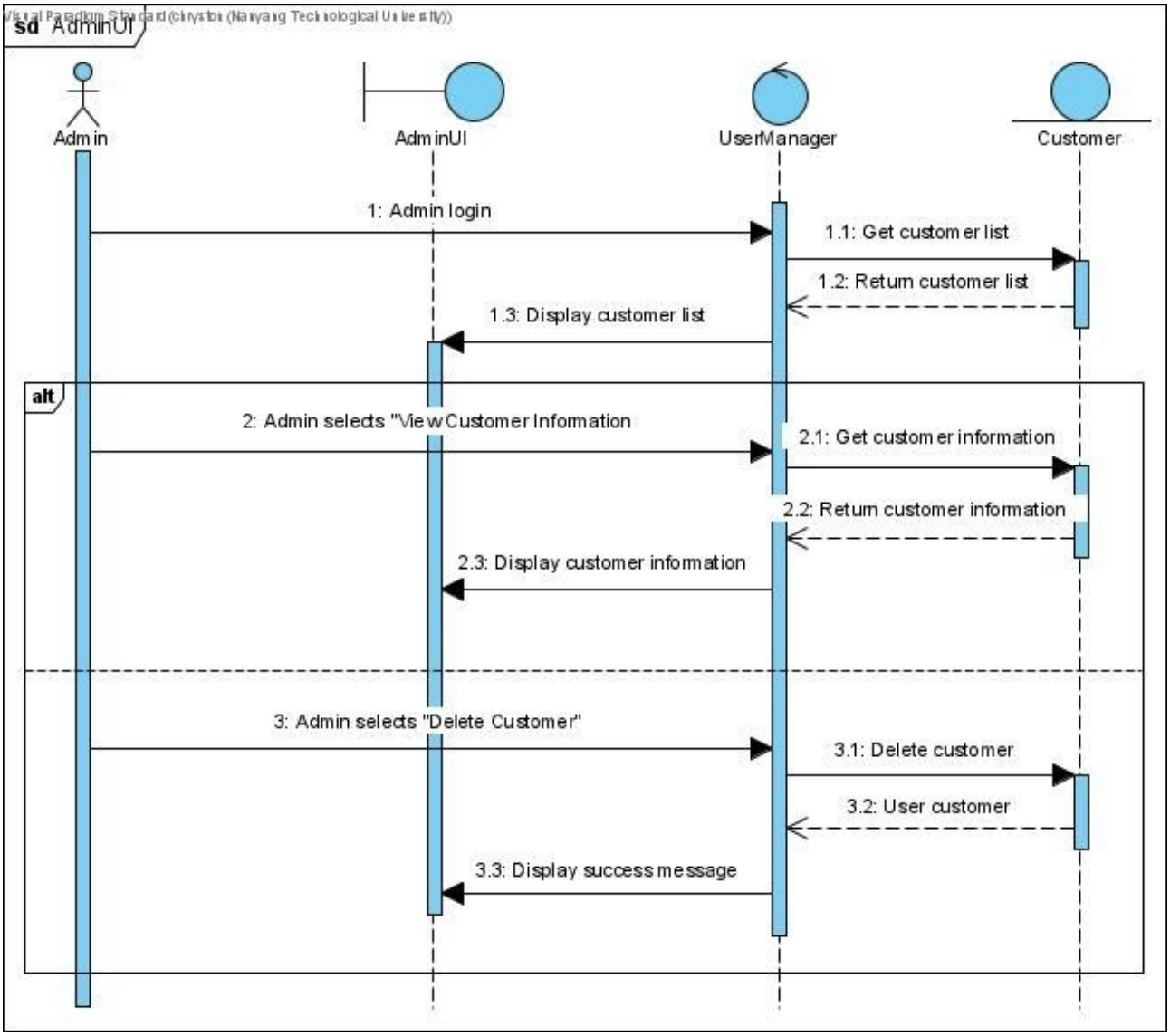

Fig 15. Sequence Diagram for HouseMatchers (AdminUI).

4.11 System Architecture Diagram

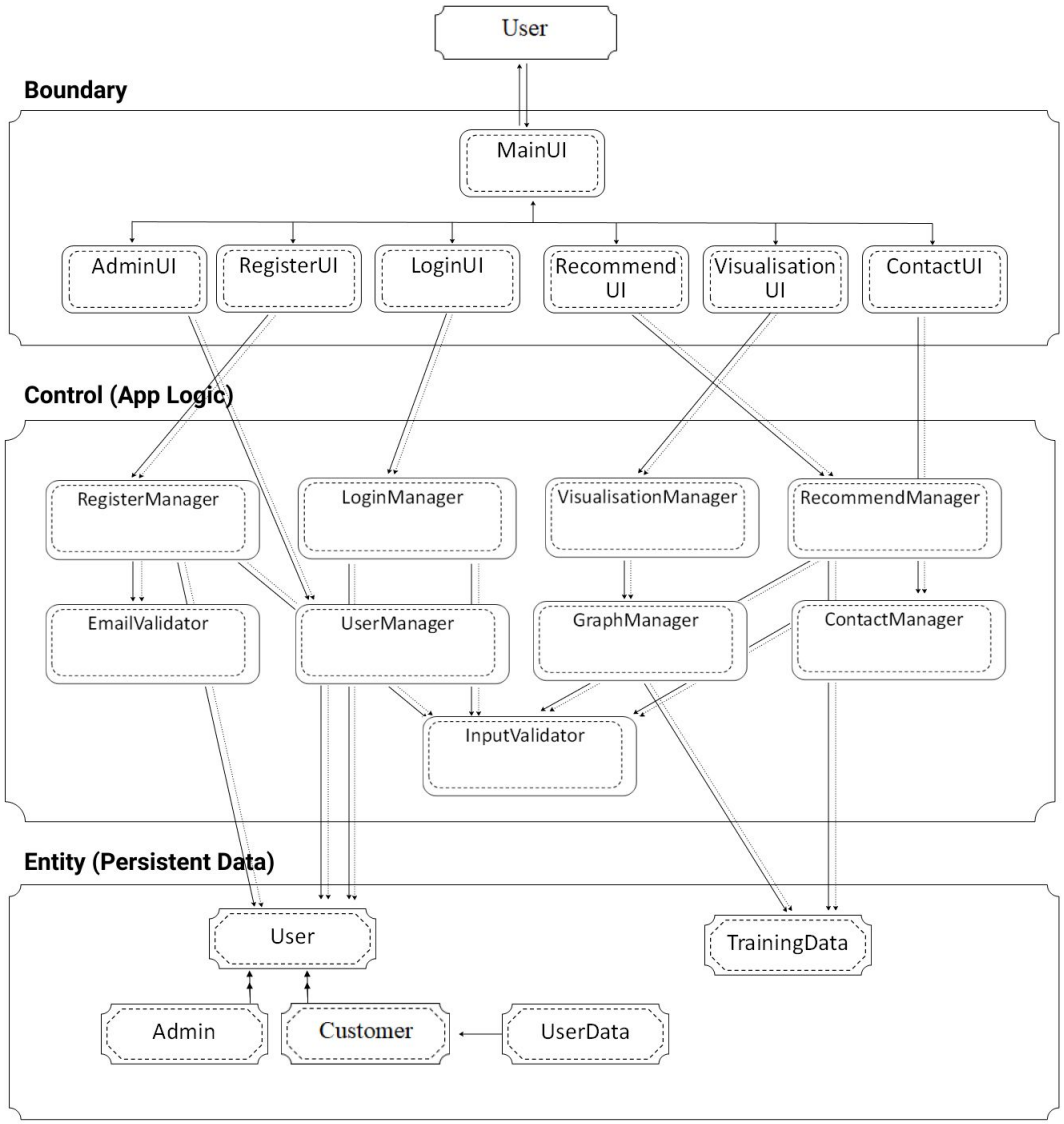

Fig 16. System Architecture Diagram for HouseMatchers.

4.12 Model-View-Controller Architecture

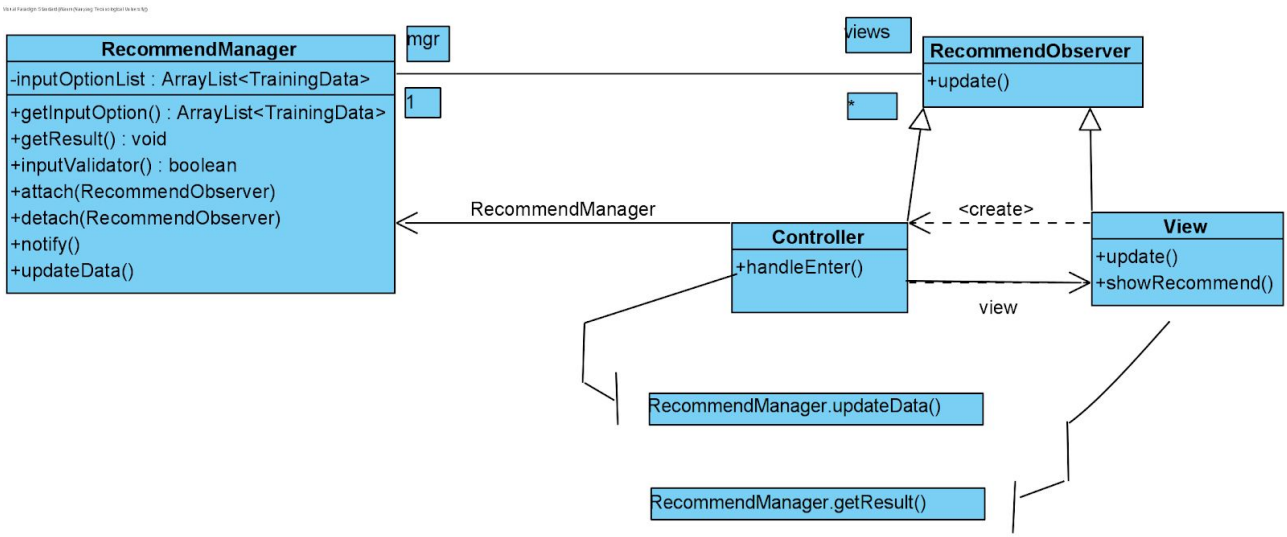

Fig 17. Class Diagram of MVC Architecture.

The design of the website follows the Model-View-Controller (MVC) architecture. For applications of the observer pattern in MVC, the view subscribes for changes to the model. The controller gathers input from the users and updates the model. The model updates the change and notifies all subscribers of the change. The view is then notified and updates its display.

The observer pattern is implemented into the system. When the customer enters his/her parameters in the recommend tab, the subscribed controller and view are updated respectively via notification mechanism. Therefore, the view can “pull’ the relevant data from the database to update their results page.

4.13 Design Patterns

The strategy pattern, factory pattern and dynamic loading pattern were implemented to the system. An interface <<Graph>> was created as a superclass for the different types of graphs. A static factory was created to instantiate <<Graph>> class was created to instantiate the creation of a graph. The static factory can be used to decouple class selection and instantiate new objects easily. Using the strategy pattern, heatmap, bar graph and time series all implements <<Graph>>. Through dynamic loading, the code can be easily extended to include other graphs and visualisation.

Facade pattern was implemented to reduce coupling and dependencies between classes. For example, a facade was created with the various parts to hide the implementation of the

code. By implementation of the facade pattern, the code and implementation of the dependencies between the boundary classes (i.e. UI classes) are decoupled with the control classes (i.e. Manager classes).

5. Other Nonfunctional Requirements

5.1 Usability Requirements

5.1.1. The system must allow easy reversal of customer's actions.

5.1.1.1. The customer must be able to change their search criteria by editing the text fields and resubmitting the new values.

5.1.2. The system must reduce short-term memory load.

5.1.2.1. The recommendation would be limited to 5 only because more would clutter the UI and have the customer scroll a lot.

5.1.3. The system must have consistency.

5.1.3.1. A consistent sequence of actions is required for similar situations.

5.1.3.2. A consistent visual layout matching the website's theme should be adopted (e.g. labels, fonts, and colour palette)

5.1.4. The system must offer informative feedback.

5.1.4.1. To provide necessary feedback to the user when invalid inputs are detected.

5.1.4.2. To display an appropriate error message when an exception occurs.

5.1.4.3. To display an appropriate warning message when the user is about to undertake an irreversible process.

5.2 Security Requirements

5.2.1. The system will only store Secure Hash Algorithm (SHA) hashed passwords into the database to prevent hacking attempts.

5.2.2. On login, the system will compare the salt-hashed version of the customer's password with the hashed password stored in the database and will allow entry only if they match.

5.2.3. The system will mask the password field with asterisks to prevent onlookers from stealing the password.

5.2.4 The system will store customer data. Only the customer himself/herself will be able to view and edit the data. The data will not be used for other purposes other than storage under PDPA.

5.3 Performance Requirements

5.3.1.The system must be able to display the results to the user within 3 seconds.

5.3.1.1. The machine learning algorithm in the backend should run for approximately 2 seconds maximum.

5.3.2.The system must not crash at any process during the website's lifetime.

5.3.3. The system must be able to run with little or no downtime.

5.3.3.1. Any maintenance work should be done on a different branch of the codebase to allow users to continue using the main branch.

5.3.4. The system must have fast response times (within 10 milliseconds) in order to prevent the user from experiencing any input lag or latency issues.

6. Test Cases

The following are test cases conducted following black box testing with equivalent classes. During testing, only one of the fields was set to be invalid error handling or behaviour for each type of invalid input.

Test ID	Test Name	Scenario	Oracle (Expected output)	Log (Actual output)
1	Incorrect Login Credentials (password)	The user inputs incorrect password	Error: Invalid Credentials	Pass
2	Empty Email Field for Login	The user does not type anything in the email field	Error: Please include a valid email	Pass
3	Empty Password for Login	The user types an email ID but password field is empty	Error: Invalid Credentials	Pass
4	Correct Login	The user types a correct Email ID and corresponding Password	Response 200	Pass
5	Correct Account Registration	The user inputs a valid name, email and password	Response 200	Pass

6	Existing Email Registration	The user inputs an already existing email for creating an account	Error: User Already Exists	Pass
7	Empty Confirm Password Field for account registration	The user leaves the confirm password field empty	Error: Password don't match	Pass
8	Empty Name field for Registration	The user leaves the Name field Empty	Error: Name is required	Pass
9	Check if Profile Exists (Recommend)	We check in the database whether a particular user exists or not	Error: User not Found	Pass
10	Renders the Header Component Correctly	All the Elements in the Header should exists in the App	To be truthy	Pass
11	Renders the Body Component	All the Elements in the Body Component should exist in the App	To be truthy	Pass

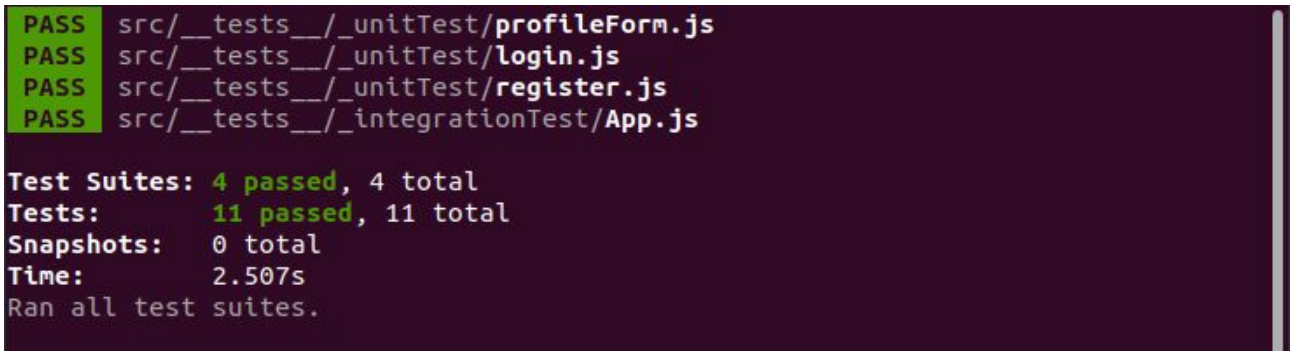

Fig 18. Results of Test Cases.

Appendix

1. High resolution versions of all the images and diagrams provided in this SRS (use case diagram, class diagram, sequence diagrams, architecture diagram, dialog map, UI mockups) can be found in the SVN submission folder. An NTU VPN is required to access the SVN outside of the NTUSECURE network.

2. Visual Paradigm UML Software (https://www.visual-paradigm.com/) was used to create data flow diagrams in this SRS. (ver.16.1)

3. Heroku (https://www.heroku.com/) is used to host the website.

4. MongoDB (https://www.mongodb.com/) is used as the database system for the website.

5. This Software Requirements Specification (SRS) is organised according to the template provided by the module coordinators of CE/CZ2006 Software Engineering from Nanyang Technological University (NTU).

6. Initial UI Mockups