Scrap-Bot

Software Requirements Specification

Version 2.2

30 – April – 2020

Hasan Haider

Inara Chagani

Zainab AliVyajkora

Raza Naqvi  
Muhammad Shahzain

Prepared for

CS 353—Software Engineering

Instructor: UmairAzfar Khan

Spring 2020

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Description** | **Author** | **Comments** |
| 3/4/2020 | Version 1.0 | Muhammad Shahzain | Section 1-3 updated. |
| 14/4/2020 | Version 1.1 | Hasan Haider  Zainab Vyajkora | Updated Section 1.2, 2, 3.5 |
| 28/4/2020 | Version 2.0 | Hasan Haider | Added Section 4.1 |
| 30/4/2020 | Version 2.1 | Zainab Vyajkora | Updated sections 3.4 and 4.1 |
| 30/4/2020 | Version 2.2 | Hasan Haider | Updated sections 3.4 and 4.1 |

# Document Approval

The following Software Requirements Specification has been accepted and approved by the following:

|  |  |  |  |
| --- | --- | --- | --- |
| **Signature** | **Printed Name** | **Title** | **Date** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

**REVISION HISTORY II**

**DOCUMENT APPROVAL II**

**1. INTRODUCTION 1**

1.1 Purpose 1

1.2 Scope 1

1.3 Definitions, Acronyms, and Abbreviations 2

1.4 References 2

1.5 Overview 2

**2. GENERAL DESCRIPTION** 3

2.1 Product Perspective 3

2.2 Product Functions 3

2.3 User Characteristics 3

2.4 General Constraints 4

2.5 Assumptions and Dependencies 4

**3. SPECIFIC REQUIREMENTS** 4

3.1 External Interface Requirements 4

*3.1.1 User Interfaces* 4

*3.1.2 Hardware Interfaces* 4

*3.1.3 Software Interfaces* 4

*3.1.4 Communications Interfaces* 5

3.2 Functional Requirements 5

*3.2.1 Sign Up and Log In* 5

*3.2.2Set Up Preferences* 5

*3.2.3Fetch Advertisements* 6

*3.2.4Post Advertisements* 6

3.3 Use Cases 7

*3.3.1 Use Case #1 7*

*3.3.2 Use Case #2 9*

*3.3.3 Use Case #3 11*

*3.3.4 Use Case #4 13*

*3.3.5 Use Case #5 15*

*3.3.5 Use Case #6 17*

*3.3.5 Use Case #7 18*

*3.3.8 Use Case #8 19*

*3.3.9 Use Case #9 21*

*3.3.10 Use Case #10 23*

*3.3.11 Use Case #11 24*

*3.3.12 Use Case #12 25*

*3.3.13 Use Case #13 26*

3.4 Classes / Objects 28

*3.4.1 Class Diagram CD-1* 28

3.5 Non-Functional Requirements 31

*3.5.1 Performance* 31

*3.5.2 Reliability* 31

*3.5.3 Availability* 31

*3.5.4 Security* 31

*3.5.5 Maintainability* 31

*3.5.6 Portability* 31

3.6 Design Constraints 32

3.7 Logical Database Requirements 32

3.8 Other Requirements 32

**4. ANALYSIS MODELS** 32

4.1 Sequence Diagrams 32

4.3 Data Flow Diagrams (DFD) 45

4.2 State-Transition Diagrams (STD) 45

**5. CHANGE MANAGEMENT PROCESS 5**

**A. APPENDICES** 46

A.1 Appendix 1 46

A.2 Appendix 2 5

# Introduction

## 1.1 Purpose

The purpose of this piece of writing is to document all the functional and non-functional requirements that are to be expected in the Scrap-Bot end-product. The intended audience are the software developers, programmers, designers, testers and project managers.

## 1.2 Scope

The Scrap-Bot will scrap car ads from two websites namely OLX and PakWheels using ad scraping and ad crawling tools. It consists of three major modules, namely:

1. Ad Fetch Module: A typical user will visit the website to search for car ads according to his/her preferences. The system will return results from the host app as well as all relevant results scraped from OLX and PakWheels, etc. as per its last updated records.
2. Ad Post Module:A registered user may also post ads to the system and additionally to OLX and PakWheels, etc. as per his/her choice. He/she may update his/her posted advertisements at any point. This feature is to be availed by registered users only.
3. User Module: The system will also maintain a record of registered users. Users may create an account and maintain a profile including personal preferences and a wish list which contains his/her selected advertisements. A registered user may also post ads to the system and additionally to OLX and PakWheels, etc. as per his/her choice.

The intended goal of the Scrap-Bot is to provide a one-stop platform to those who are looking for a specific automobile but don’t want to search through multiple websites, possibly coming across identical ads. The Scrap-Bot minimizes redundancy, is time-efficient and is a common platform for buyers and sellers for increased outreach.

The Scrap-Bot is limited to scraping and crawling through a limited number of websites only. It is updated once every 24 hours. In case a user updates an advertisement posted on OLX or PakWheels, etc using those websites instead of the Scrap-Bot, the database will be updated at the time of the routine update.

## 1.3 Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| Project Manager | One who handles the planning and execution of a certain project. |
| Web Crawling | A technique mostly used by search engines to systematically browse the WWW(Internet). |
| WWW | WorldWideWeb |
| Web Scraping | A process in which large amounts of data are extracted from the web and is usually saved in a database. |
| Database | A systematic way of organizing data to ensure fast access, insertion and deletion of data. |

## 1.4 References

## 1.5 Overview

The SRS is divided into five components. Beginning with the introduction, it jumps onto the

General Description, then comes onto Requirements followed by Analysis Models and

Appendix.

The General Description has all the specifications of the product in a high level fashion while the

Requirements contains detailed description of what the product should be able to do, or contain.

Analysis Models tell us the description of certain requirements while the appendices contain all

thefigures, diagrams and additional information related to SRS.

# 2. General Description

## 2.1 Product Perspective

This product is being developed as an innovation to basic primary buy-sell websites like OLX or PakWheels. Several similar services exist but the Scrap-Bot is being developed as a means to combine them and provide a more coherent system. It will allow users to effectively search and post car advertisements at a common platformwith increased audience. Once users register on the website, they may maintain their preferences and wish list containing filtered ads that the user may have picked out. It allows the users to interact effectively through text messages and provides email notifications for relevant ads as well as new messages.

## 2.2 Product Functions

**Post Ads:** The User will be able to post advertisements on the selected websites. The system collects required information about the advertisement from the user and then posts it on the specified website.

**Fetch Ads:** The User will be able to search advertisements posted by people on the other websites. The user may look at featured websites on the Scrap-Bot's home page or apply specific filters to obtain results according to the his/her selected preferences. The system would use web scraping and crawling techniques to gather all ads that matches the user's search criteria from the host website, as well as other car ad websites.

**Set User Preferences:** Based on the previous choices of the user, the system will show relevant advertisements. Registered users may also save their preferences on their profile.

**Create User Wish-list:** The User will be allowed to maintain a wish-list for the products they like. A user may make multiple searches and add each ad they like to a personal wish list which he/she may be able to browse through at any time. The system will notify the user by means of a "sold" tag if a car in the wish list has been sold.

## 2.3 User Characteristics

2.3.1 A user may be an owner of a car that he wishes to sell. This could be a personal car owner or part of a car selling company or business.

2.3.2 A user may be a person looking to buy a car.

## 2.4 General Constraints

## 2.4.1 The accurateness of the ads being listed on the server depends on the time the user uses the website as the database is updated once at a fixed time every day.

2.4.2 The website will be available as a desktop version of a web application only. A mobile app does not exist.

2.4.3 The Scrap-Bot is entirely dependent upon the other websites from where it scrapes and crawls advertisements. Any change in the html tags of these websites will result in malfunction.

## 2.5 Assumptions and Dependencies

2.5.1 The user will be fluent in English language.

2.5.2 The user will have the hardware required to use the application, i.e. a machine, possibly keyboard/mouse/a tablet device.

2.5.3The user will have a working browser to access the web application.

2.5.4The system heavily depends on and assumes a stable network connection.

2.5.5 Each user is registered on both OLX and PakWheels with the same name.

2.5.6 The system is dependent upon OLX, PakWheels, sastigari and pkmotors for fetching and posting advertisements. It also assumes that the UI of any of these websites does not change.

# 3. Requirements

## 3.1 External Interface Requirements

### 3.1.1 User Interfaces

It will have a web-application based interface. It can be used on a web browser.

### 3.1.2 Hardware Interfaces

Hardware specifics are not related to standalone devices. The main system will have a hosted server while the user system can be a simple personal computer or any portable device with a screen and a browser available.

### 3.1.3 Software Interfaces

The web browser is essential to run the website. The websites mentioned in section 2.5.6 of this document must exist in the same way (in terms of html tags etc) for the system to work correctly.

### 3.1.4 Communications Interfaces

It’s mostly web dependent that means it must be hosted on a fast-response server for efficient retrieval of data.

## 3.2 Functional Requirements

*This section describes specific features of the software project. If desired, some requirements may be specified in the use-case format and listed in the Use Cases Section.*

### 3.2.1 Sign Up and Log In

3.2.1.1 Introduction

The user should be able to sign up on the platform to use it.

3.2.1.2 Inputs

The user will connect to the website and click on the Sign Up button.

After signing up, the user will click the log in button where the user inputs his/her credentials.

3.2.1.3 Processing

The user credentials will be saved in the credential database when signing up.

And these credentials will be verified upon logging in.

3.2.1.4 Outputs

The user will be signed up OR logged in, and will be able to use the system.

3.2.1.5 Error Handling

If user is already registered, signing up will return an error.

If user inputs wrong credentials, he will not be able to log in.

**3.2.2Set Preferences**

3.2.2.1 Introduction

The user should be able to set preferences about the make and type of automobile they want to see.

3.2.2.2 Input

The user will select “Preferences” button where the user will have the option to type and save the preferences.

3.2.2.3 Processing

The specified preferences will be saved in the Scrap-Bot Database.

3.2.2.4 Output

The ad-fetching mechanism will work based on these preferences now.

### 3.2.3 Fetch Advertisements

3.2.3.1 Introduction

The user should be able to fetch ads upon the basis of the preferences they have set.

3.2.3.2 Inputs

There will be no user inputs in this case as the Scrap-Bot will fetch ads on the basis of preferences that are set.

3.2.3.3 Processing

The Scrap-Bot will fetch user preferences from the database and will start the process of web-scraping to get relevant advertisements.

3.2.3.4 Outputs

The user will start receiving relative ads on the Scrap-Bot platform.

### 3.2.4Post Advertisements

3.2.4.1 Introduction

The user should be able to post ads by filling out relevant car specifications.

3.2.4.2 Inputs

The user must fill in all the required specifications i.e., Make, Model, Year, Color, Engine in a form, select the websites he/she wishes to post the advertisement on and then press "Post Ad" button.

3.2.4.3 Processing

The Scrap-Bot will fetch verify whether all required information has been filled in the relevant fields. Upon successful verification, it will then post ads to the specified websites.

3.2.4.4 Outputs

The ads will be posted on the specified websites.

## 3.3 Use Cases

# 3.3.1 Use Case #1 Create Advertisement

# Brief Description

This use case describes how the member uses the website to post an advertisement from his/her account.

**Actors**

* **Member**
* **Scrap-Bot Database**

**Preconditions**

There is an active network connection to the website.

The User already has an account on the website.

The User has already logged-in through a valid id and password.

**Basic Flow of Events**

1. The use case begins when User clicks on the “Post your Ad” button.
2. Use Case: Create Advertisement is performed.
3. The system prompts for car details.
4. The system then asks the user to select the websites he/she wants to post the advertisement on.
5. The user may then update previous credentials for the websites.
6. The advertisement is posted.
7. The use case ends successfully.

**Alternative Flows**

* **Incomplete Car Details**

If in step 3 of the basic flow Member the use case: take Credentials does not complete successfully, then

1. The user is prompted to fill in complete information.
2. The use case resumes at step 3.

* **Websites not selected**

If in step 4 of the basic flow the member does not select any website, then

1. The website will select Scrap-Bot to post the advertisement on.
2. The use case resumes at step 5.

* **Logout**

If at point prior to step 7 in the basic flow the Member selects to log out, then

1. The system prompts the user that all activities have been cancelled.
2. The use case ends

**Key Scenarios**

* **No Response from Website**

**Post-conditions**

* **Successful Completion**

The Member has successfully posted the advertisement and the database has been updated.

* **Failure Condition**

The logs have been updated accordingly.

**Special Requirements**

All steps are performed in binary. That is they are either performed successfully or they fail entirely.  
The Database is not updated until the advertisement is posted.

# 3.3.2 Use Case #2 – Delete Advertisement

**Brief Description**

This use case describes how the member uses the website to delete an advertisement from his/her account.

**Actors**

* **Member**
* **Scrap-Bot Database**

**Preconditions**

There is an active network connection to the website.

The User already has an account on the website.

The User has already logged-in through a valid id and password.

The User has already created and posted an advertisement.

**Basic Flow of Events**

1. The use case begins when User clicks on the “Delete your Ad” button.
2. Use Case: Delete Advertisement is performed.
3. The system removes advertisement details from the database.
4. The use case ends successfully

**Alternative Flows**

* **Logout**

If at point prior to step 4 in the basic flow the Member selects to log out, then

1. The system prompts the user that all activities have been cancelled.
2. The use case ends

**Key Scenarios**

* **No Response from Website**

**Post-conditions**

* **Successful Completion**

The Member has successfully deleted the advertisement and the database has been updated.

* **Failure Condition**

The logs have been updated accordingly.

**Special Requirements**

All steps are performed in binary. That is they are either performed successfully or they fail entirely.  
The Database is not updated until the advertisement is deleted.

# 3.3.3 Use Case #3-Update Advertisement

# Brief Description

This use case describes how the member uses the website to update an advertisement from his/her account.

# Actors

## Member

## Scrap-Bot Database

# Preconditions

There is an active network connection to the website.

The User already has an account on the website.

The User has already logged-in through a valid id and password.

The User has already created and posted the advertisement.

# Basic Flow of Events

# The use case begins when User clicks on the “Update your Ad” button.

# Use Case: Update Advertisement is performed.

# The system prompts for car details.

# The system then asks the user to select the websites he/she wants to post the advertisement on.

# The user may then update previous credentials for the websites.

# The advertisement is posted.

# The use case ends successfully

# Alternative Flows

## Incomplete Car Details

1. If in step 3 of the basic flow Member the use case: take Credentials does not complete successfully, then the user is prompted to fill in complete information.
2. The use case resumes at step 3.

## Websites not selected

If in step 4 of the basic flow the member does not select any website, then

1. The website will select Scrap-Bot to post the advertisement on.
2. The use case resumes at step 5.

## Logout

If at any point prior to step 7 in the basic flow the Member selects logs out, then

1. The system prompts the user that all activities have been cancelled.
2. The use case ends.

## No new changes

If the User does not make any new changes, then

1. The database is not updated.
2. Use case resumes at step 5.

# Key Scenarios

## No Response from Website

# Post-conditions

## Successful Completion

The Member has successfully posted the advertisement and the database has been updated.

## Failure Condition

The logs have been updated accordingly.

# Special Requirements

All steps are performed in binary. That is they are either performed successfully or they fail entirely.  
The Database is not updated until the advertisement is posted.

# 3.3.4 Use Case #4-Search Car Ads

# Brief Description

This use case describes how the member uses the website to search for a car advertisement using his/her account.

# Actors

## Member

## Scrapbot Database

# Preconditions

There is an active network connection to the website.

The User already has an account on the website.

The User has already logged-in through a valid ID and password.

# Basic Flow of Events

1. The use case begins when Member clicks on "Search for Car Ads" button.
2. The System displays the different fields required to fill in the car specifications to search.
3. The User fills in the desired search criteria. [See reference document]
4. The Scrapbot returns matching results and the system displays all results based on the specified search criteria.
5. The User browses through the results.
6. The User selects a Car Ad.
7. The user adds the selected Car Ad to Wishlist.
8. The use case ends successfully.

# Alternative Flows

## No Search Criteria Specified

If in step 3 of the basic flow the User has not filled in any search criteria, then

1. The System shall display the message “Please specify Search criteria”.
2. The use case resumes at step 2.

## No Results Returned

If there are no results that match the search criteria specified by user in step 4, then

1. The System will display a message "No results found. Please enter new search criteria."
2. The use case resumes at step 2.

## No Car Selected

If in step 5 of the basic flow, Member does not select any of the car ads displayed, then

1. The User clicks on "New Search".
2. The use case resumes at step 2.

## Logout

If at point prior to step 8 in the basic flow the Member logs out, then

1. The system prompts the user that all activities have been cancelled.
2. The use case ends.

# Key Scenarios

## No Response from Website

# Post-conditions

## Successful Completion

## The user has selected a Car Ad and added it to Wishlist.

## Failure Condition

## The logs have been updated accordingly.

# Special Requirements

All steps are performed in a strict flow.

The database is actively updated in order to display most accurate information.

The user profile is maintained according to his/her search history and preferences.

# 3.3.5 Use Case #5-Send Message

# Brief Description

This use case describes how the member uses the website to send a message to another user using his/her account.

# Actors

## Member

## Scrapbot Database

# Preconditions

There is an active network connection to the website.

Both the User and the message recipient already have an account on the website.

The User has already logged-in through a valid ID and password.

The Recipient of the message is added as a contact in the User's messenger account or reached directly through an Ad.

The User must have completed one of the Use cases "Show Result" or "Browse Wishlist".

# Basic Flow of Events

# The use case begins when Member clicks on "Contact Seller" button on an Ad or selects an already added User in his/her messenger contacts.

# The System prompts the User to type in a message.

# The User types in his/her message.

# The User clicks on the "Send" button.

# The System prompts the user to confirm whether he/she wants to send the message.

# The User selects "Confirm".

# The System displays a message "Message successfully sent".

# The use case ends successfully.

# Alternative Flows

## Message field empty

If in step 3 of the basic flow the User has left the Message field empty, then

1. the System shall display the message “Please type a message. The message cannot be empty.”.
2. The use case resumes at step 2.

## No Confirmation

If in step 6 the user selects "Go Back", then

1. The System will display the message the user has typed and he may edit it.
2. The use case resumes at step 3.

## Logout

If at any point prior to step 8 in the basic flow the Member selects to log out, then

1. The system prompts the user that all activities have been cancelled.
2. The use case ends.

# Key Scenarios

## No Response from Website

# Post-conditions

## Successful Completion

The user has sent a message to the seller/other user.

## Failure Condition

The logs have been updated accordingly.

# Special Requirements

All steps are performed in a strict flow.

The user profile and messenger history is updated and new contacts are added to the list of active conversations.

# 3.3.6 Use Case #6-Browse Wishlist

# Brief Description

This use case describes how member browses wishlist and performs different actions on advertisements in wishlist .

# Actors

# Member

# Scrapbot database

# Preconditions

There is an active network connection to the website.

# Basic Flow of Events

1. The use case begins when the member clicks “Browse Wishlist” button.
2. Advertisements in the wishlist are displayed.
3. The User may delete any advertisement from the wishlist.
4. The User may send message to any seller whose advertisements are displayed.
5. The use case ends successfully.

# Alternative Flows

## No advertisements in the wishlist

In step 2 of basic flow of events, if wishlist is empty then

1. No Advertisements are displayed and step 3 and 4 of basic flow of events are no longer valid.

# Key Scenarios

## No Response from Website

# Post-conditions

## Successful Completion

The user has successfully viewed advertisements from the wishlist. If user has removed any advertisement from wishlist, it has been removed. If user has contact any seller, his message has been sent to the seller.

## Failure Condition

The logs have been updated accordingly.

# Special Requirements

The database is updated only if any advertisement is removed from the wishlist or a seller has been sent message to by the user.

# 3.3.7 Use Case #7-Scrap Car from websites

# Brief Description

This use case describes how the system scraps car advertisements from different advertisement websites like PakWheels, olx etc.

# Actors

## Scrapbot Database

# Preconditions

The system has the URLs of all the car advertisement websites.

The system has sufficient storage capacity.

System has an active network connection.

# Basic Flow of Events

1. The use case begins automatically at a fixed time everyday.
2. The system extracts the URLs of all car advertisement websites.
3. The system accesses all the advertisement websites and scraps all car advertisements posted on these websites. Scrapped advertisements are stored on the database.
4. All duplicate advertisements are removed from the database.
5. Finally, the database is updated and use case ends.

# Alternative Flows

## Website Error

In step 3 of basic flow of events, if any advertisement website cannot be accessed then

* 1. Advertisements will not be scrapped from websites with access issues.
  2. The use case will end with failure condition.

# Key Scenarios

## System failure

# Post-conditions

## Successful Completion

Recently posted car advertisements are available on scrapbot’s website.

## Failure Condition

The logs have been updated accordingly.

# Special Requirements

In case of layout modification of an advertisement website, new code must be written to scrap advertisements from that website.

# 3.3.8 Use Case #8 Login

# Brief Description

This use case describes how the Bank Customer uses the website to login to his/her account.

# Actors

## Member

## Scrap-bot Database

# Preconditions

There is an active network connection to the website.

The User already has an account

# Basic Flow of Events

1. The use case begins when the User clicks the “Log In” button.
2. Use Case: “Login” is performed.
3. The system prompts the user to fill in his/her username and password.
4. The user will then press the “login” button to verify his/her credentials.
5. The user has successfully logged in.
6. The use case ends successfully

# Alternative Flows

## Invalid Login

If in step 4 of the basic flow Member the use case: Verify Credentials does not complete successfully, then

1. the web page shall display the message “Invalid Username/Password – please try again”
2. The use case resumes at step 3

## Logout

If at point prior to step 6 in the basic flow the User logs out, then

1. The use case ends

# Key Scenarios

## No Response from Website

# Post-conditions

## Successful Completion

The Member has successfully logged in.

## Failure Condition

The logs have been updated accordingly.

# Special Requirements

All steps are performed in binary. That is they are either performed successfully or they fail entirely.

# 3.3.9 Use Case #9 Sign Up

# Brief Description

This use case describes how the Member uses the website to create an account.

# Actors

## Member

## Scrap-Bot Database

# Preconditions

There is an active network connection to the website.

# Basic Flow of Events

1. The use case begins when the User clicks on the “Sign Up” button.
2. Use Case: “Signup” is performed.
3. The system prompts the User to fill in the details.
4. The user will then press the “Sign Up” button to validate user information.
5. The user successfully signs up.
6. The use case ends successfully.

# Alternative Flows

## User already exists.

If in step 4 of the basic flow, the Member account already exists, then

1. The web page shall display the message, “This Member already exists”.
2. The use case ends.

## Invalid Value

If in step 4 of the basic flow, the Member enters an invalid value in any of the data fields, then

1. The web page will display the message “Invalid Value in this field!”, and ask the user to enter a valid value.
2. The use case resumes at step 3

## Logout

If at a point prior to step 6 in the basic flow the Member logs out, then

1. The system prompts the user that the account has not been created.
2. The use case ends.

# Key Scenarios

## No Response from Website

# Post-conditions

## Successful Completion

The member account has been created and the database has been updated.

## Failure Condition

The logs have been updated accordingly.

# Special Requirements

All steps are performed in binary. That is they are either performed successfully or they fail entirely.  
The Database is not updated until the member account has been created.

# 3.3.10 Use Case #10 View Messages

# Brief Description

This use case describes how the member uses the website to view all his/her messages.

# Actors

## Member

## Scrap-Bot Database

# Preconditions

There is an active network connection to the website.

The user is already logged in.

# Basic Flow of Events

1. The use case begins when User clicks on the “Chat” tab.
2. Use Case: “View Messages” is performed.
3. The system displays all the messages, from various sellers, to the user.
4. The user has the liberty to search specific messages.
5. The use case ends successfully

# Alternative Flows

## Logout

If at a point prior to step 4 in the basic flow the User logs out, then

1. The use case ends.

# Key Scenarios

## No Response from Website

# Post-conditions

## Successful Completion

The system displays all the messages.

## Failure Condition

The logs have been updated accordingly.

# Special Requirements

All steps are performed in binary. That is they are either performed successfully or they fail entirely.

# 3.3.11 Use Case #11 View Notifications

# Brief Description

This use case describes how the member uses the website to view all his/her notifications.

# Actors

## Member

## Scrap-Bot Database

# Preconditions

There is an active network connection to the website.

The user is already logged in.

# Basic Flow of Events

1. The use case begins when User clicks on the “View Notifications” tab.
2. Use Case: “View Notifications” is performed.
3. The system displays all the notifications to the user.
4. The use case ends successfully

# Alternative Flows

## Logout

If at a point prior to step 4 in the basic flow the User logs out, then

1. The use case ends.

# Key Scenarios

## No Response from Website

# Post-conditions

## Successful Completion

The system displays all the notifications.

## Failure Condition

The logs have been updated accordingly.

# Special Requirements

All steps are performed in binary. That is they are either performed successfully or they fail entirely.

# 3.3.12 Use Case #12 Set Preferences

# Brief Description

This use case describes how the member uses the website to set his/her car preferences.

# Actors

## Member

## Scrap-Bot Database

# Preconditions

There is an active network connection to the website.

The user is already logged in.

# Basic Flow of Events

1. The use case begins when User clicks on the “Set Preferences” button.
2. Use Case: “Set Preferences” is performed.
3. The system prompts the User to fill in his/her desired car preferences and features.
4. The user will then press the “Save” button to store the preferences.
5. The data is saved in the database.
6. The use case ends successfully

# Alternative Flows

## Log out

If at a point prior to step 6 in the basic flow the User logs out, then

1. The system prompts the user that any preferences selected were not saved.
2. The use case ends.

# Key Scenarios

## No Response from Website

# Post-conditions

## Successful Completion

The database has been successfully updated with the preferences selected by the user.

## Failure Condition

The logs have been updated accordingly.

# Special Requirements

All steps are performed in binary. That is they are either performed successfully or they fail entirely.  
The Database is not updated until the user saves the preferences.

# 3.3.13 Use Case #13 Edit User Profile

# Brief Description

This use case describes how the member uses the website to edit his/her profile.

# Actors

## Member

## Scrap-Bot Database

# Preconditions

There is an active network connection to the website.

The user is already logged in.

# Basic Flow of Events

1. The use case begins when User clicks on the “Edit Profile” button.
2. Use Case: “Edit User Profile” is performed.
3. The system prompts the User to update some details.
4. The user will then press “Save changes” button to validate entered information.
5. The updates are saved in the database.
6. The use case ends successfully

# Alternative Flows

## Invalid Value

If in step 4 of the basic flow, the Member enters an invalid value in any of the data fields, then

1. The web page will display the message “Invalid Value in this field!”, and ask the user to enter a valid value.

2. The use case resumes at step 3

## Logout

If at a point prior to step 6 in the basic flow the User chooses to log out, then

1. The system prompts the user that any updates made were not saved.
2. The use case ends

# Key Scenarios

## No Response from Website

# Post-conditions

## Successful Completion

The database has been successfully updated with the relevant changes made by the user.

## Failure Condition

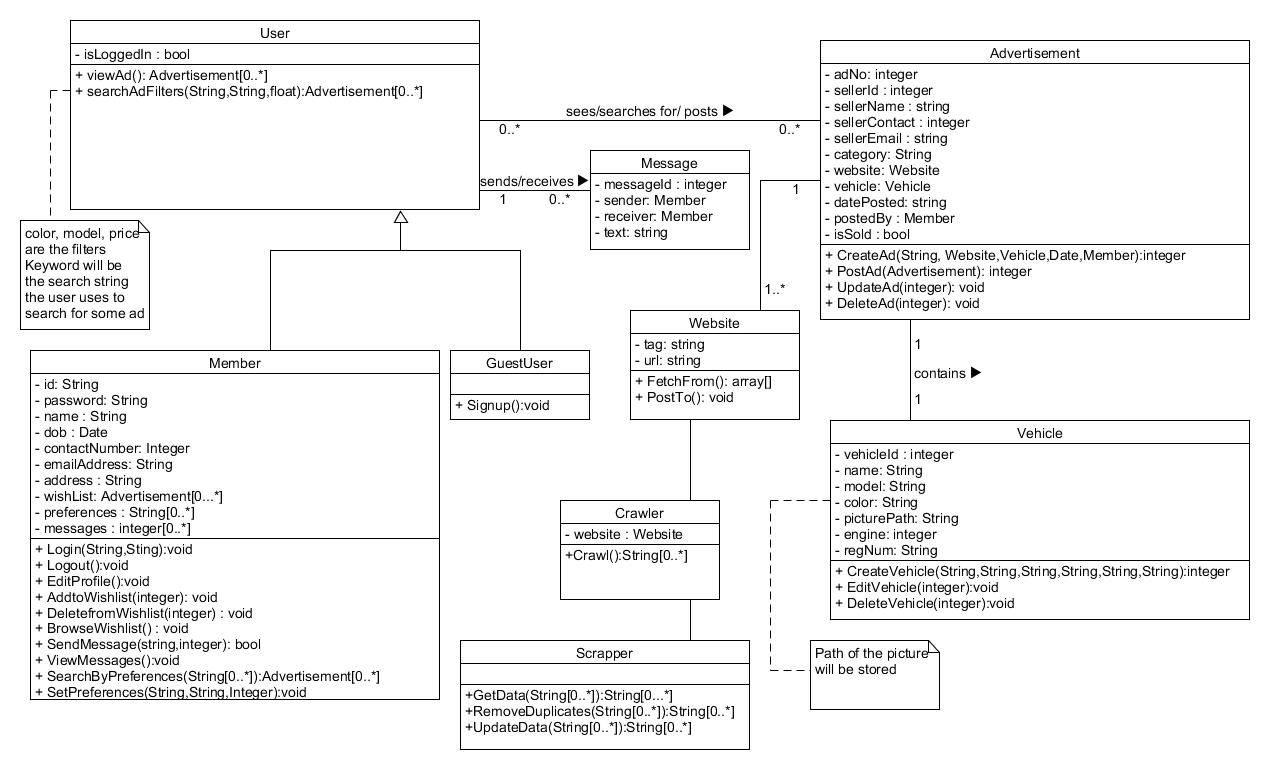
The logs have been updated accordingly.

# Special Requirements

All steps are performed in binary. That is they are either performed successfully or they fail entirely.  
The Database is not updated until the user saves the changes.

## 3.4 Classes / Objects

### 3.4.1 Class Diagram 1 (CD-1)



**3.4.1.1 Attributes**

**CD-1.1 Class: User**

1. - isLoggedIn : bool

**CD-1.2 Class: Member**

1. - id: String
2. - password: String
3. - name : String
4. - dob : Date
5. - contactNumber: Integer
6. - emailAddress: String
7. - address : String
8. - wishList: Advertisement[0...\*]
9. - preferences : String[0..\*]
10. - messages : integer[0..\*]

**CD-1.3 Class: GuestUser**

**CD-1.4 Class: Message**

1. - messageId : integer
2. - sender: Member
3. - receiver: Member
4. - text: string

**CD-1.5 Class: Advertisement**

1. - adNo: integer
2. - sellerId : integer
3. - sellerName : string
4. - sellerContact : integer
5. - sellerEmail : string
6. - category: String
7. - website: Website
8. - vehicle: Vehicle
9. - datePosted: string
10. - postedBy : Member
11. - isSold : bool

**CD-1.6 Class: Website**

1. - tag: string
2. - url: string

**CD-1.7 Class: Vehicle**

1. - vehicleId : integer
2. - name: String
3. - model: String
4. - color: String
5. - picturePath: String
6. - engine: integer
7. - regNum: String

**CD-1.8 Class: Crawler**

1. - website : Website

**CD-1.9 Class: Scrapper**

**3.4.1.2 Functions**

**CD-1.1 Class: User**

1. + viewAd(): Advertisement[0..\*]- UC#4
2. + searchAdFilters(String,String,float):Advertisement[0..\*] – UC#4

**CD-1.2 Class: Member**

1. + Login(String,Sting):void – UC#8 and FR:3.2.1
2. + Logout():void– FR:3.2.1
3. + EditProfile():void - UC#13
4. + AddtoWishlist(integer): void - UC#4
5. + DeletefromWishlist(integer) : void – UC#6
6. + BrowseWishlist() : void – UC#6
7. + SendMessage(string,integer): bool – UC#5
8. + ViewMessages():void – UC#10
9. + SearchByPreferences(String[0..\*]):Advertisement[0..\*]–UC#4
10. +SetPreferences(String,String,Integer):void – UC#12 and FR:3.2.2

**CD-1.3 Class: GuestUser**

1. + Signup():void – UC:# and FR:3.2.1

**CD-1.4 Class: Message**

**CD-1.5 Class: Advertisement**

1. + CreateAd(String, Website,Vehicle,Date,Member):integer– UC#1 and FR:3.2.4
2. + PostAd(Advertisement): integer – UC#1 and FR:3.2.4
3. + UpdateAd(integer): void – UC#3 and FR:3.2.4
4. + DeleteAd(integer): void – UC#2

**CD-1.6 Class: Website**

1. + FetchFrom(): array[]– UC#7 and FR:3.2.3
2. + PostTo(): void – UC#1

**CD-1.7 Class: Vehicle**

1. + CreateVehicle(String,String,String,String,String,String):integer – UC#1
2. + EditVehicle(integer):void – UC#3
3. + DeleteVehicle(integer):void – UC#2

**CD-1.8 Class: Crawler**

1. +Crawl():String[0..\*] – UC#7

**CD-1.9 Class: Scrapper**

1. +GetData(String[0..\*]):String[0...\*] – UC#7 and FR:3.2.3
2. +RemoveDuplicates(String[0..\*]):String[0..\*] – UC#7 and FR:3.2.3
3. +UpdateData(String[0..\*]):String[0..\*] – UC#7 and FR:3.2.3

## 3.5 Non-Functional Requirements

### Performance:

Depends on the quality of Internet. Ideally, any transaction should not take more than a second. 90% of the transactions shall be done in less than a second. The system shall not crash unless the browser crashes. The server shall be able to handle around 99 requests per second.

### Reliability:

If the website crashes, the user will know in less than a second. If a transaction cannot be completed, the system will inform in less than a second. The probability of system failure is 1/100 approximately.

### Availability:

Since it is a website, it will be available 24/7. The website (or parts of website) may be unavailable during maintenance.

### Security:

The User has an account and their data cannot be accessed without the credentials.

### Maintainability:

The website can be maintained while people are still using it. It will only go under maintenance if bugs are found.

### Portability:

With the availability of internet and a browser, the website can be access on laptops, desktop systems, mobile phones, tablets and any other devices anywhere in the world.

## 3.6 Design Constraints

Not Applicable.

## 3.7 Logical Database Requirements

SQL Server database will be used.

## 3.8 Other Requirements

Not Applicable.

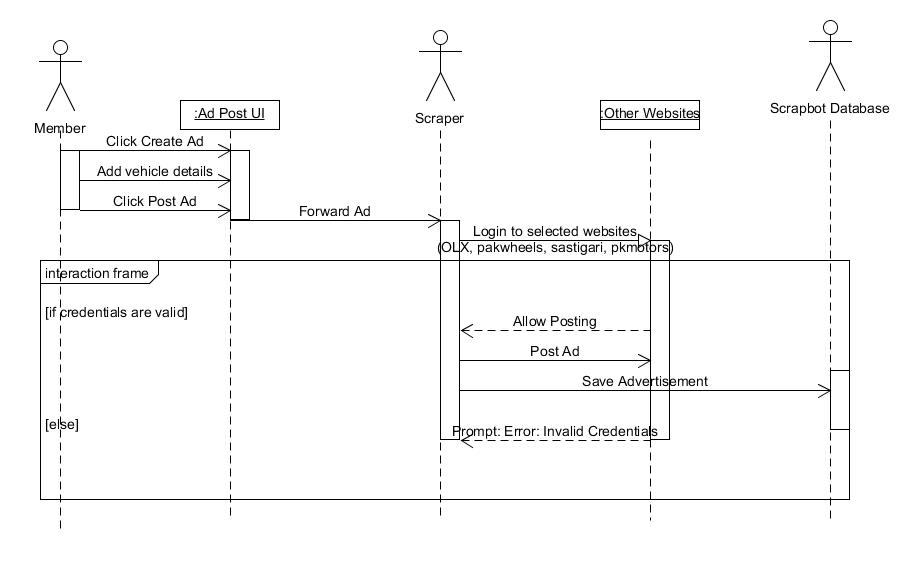
# 4. Analysis Models

## Sequence Diagrams

**4.1.1 Ad Post Module**

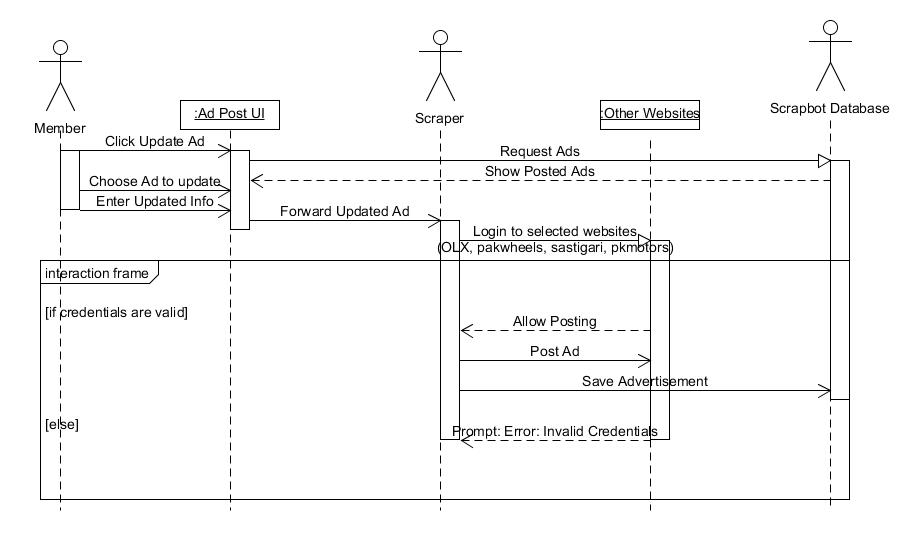
**(a) Create Advertisement**

The member interacts with the system UI to post and advertisement. After clicking on the create advertisement button, the user enters vehicle details and clicks on post advertisement button. The UI then forwards the advertisement to the scrapper which first attempts to login to the selected websites. After successful selection, the advertisement is posted and the database is updated.



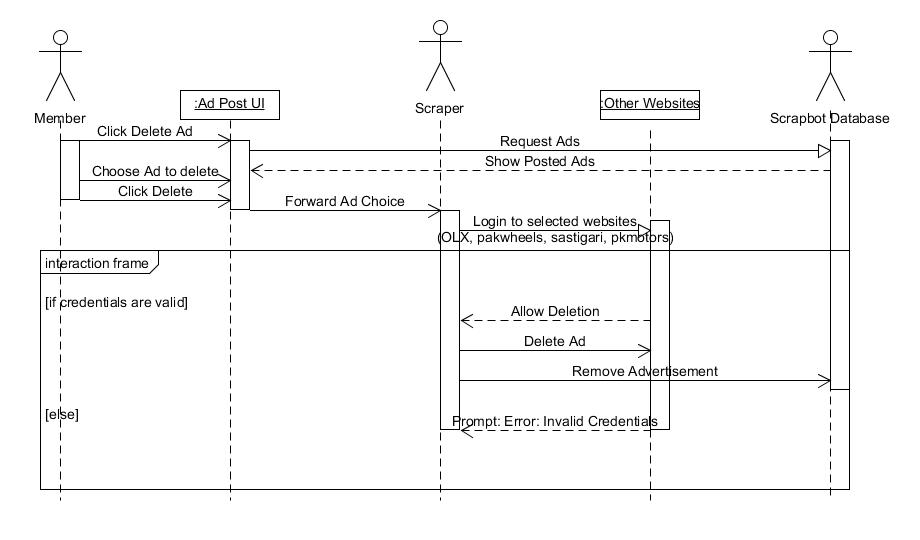
## (b) Update Advertisement

When the member clicks the update advertisement button, he or she then chooses which advertisement to update. Then the user updates the advertisement details. The UI forwards the updated advertisement to the scraper. After this, the process is same as in the “create advertisement” as described in 4.1.1(a).



## (c) Delete Advertisement

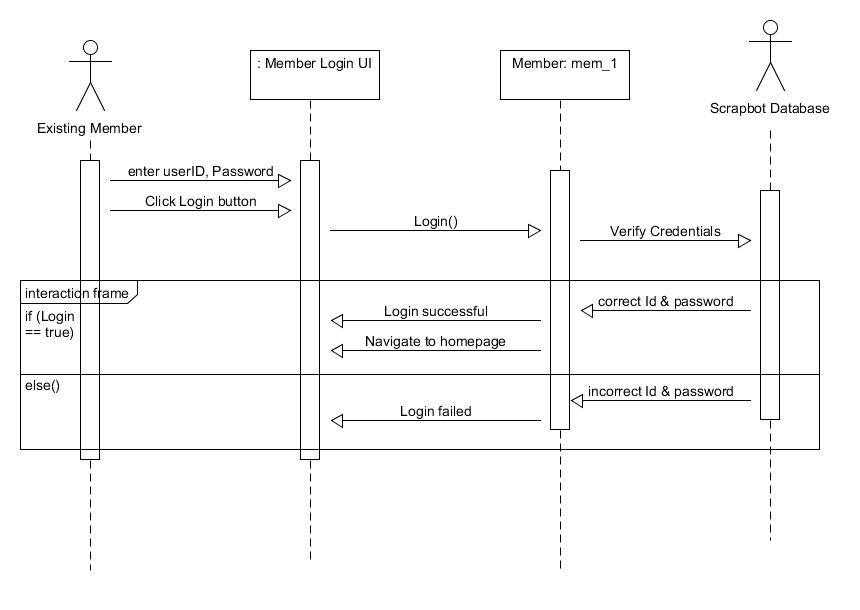
When the user clicks the delete advertisement button, he or she then has to select the advertisement to be deleted. After the advertisement is selected, the user clicks on the delete button. The choice is forwarded by UI to scraper, which logs in to the websites where this advertisement is posted. Upon successful login, the advertisement is deleted and the database is updated.



* + 1. **Member Module**

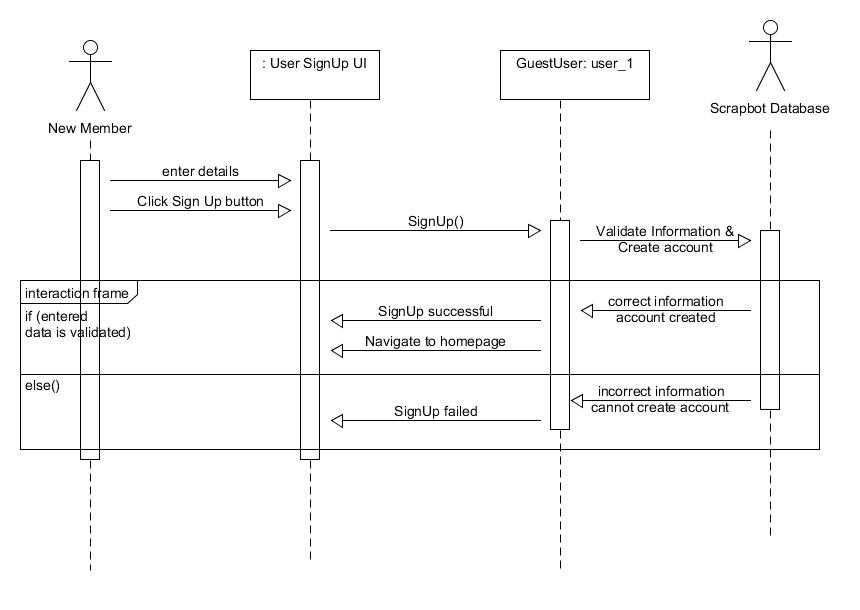
1. **Member Login**

The Member enters their user ID and password and clicks the login button, at which the Login function verifies the user credentials. In case of invalid User Id or password, the Login process is unsuccessful and the UI layer displays an “Invalid Login” message. If the entered User ID and password are correct the login is successful and the home page UI is loaded.

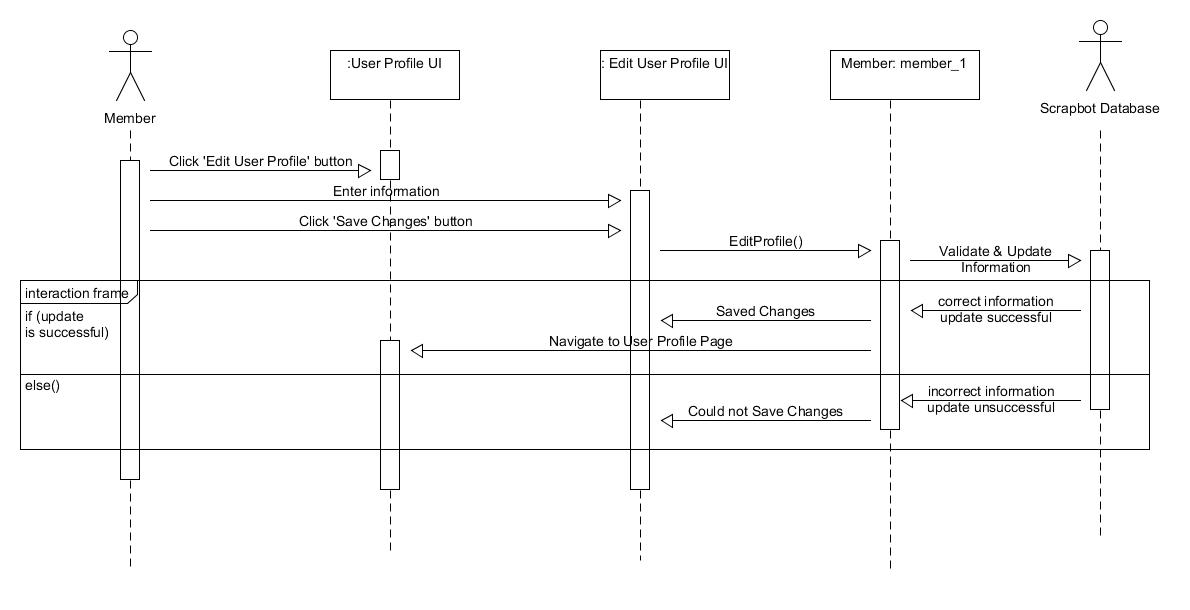


1. **User Sign Up**

The user enters his details ( Name, Date of Birth, Address, Emailaddress, etc) and clicks the Sign Up button on which the SignUp function validates the information entered by the user. If the information entered is correct and contains no invalid data, the user account is created and the user is notified by navigating to the homepage and their user name appearing on the navigation bar. In case of invalidate information, the user account is not created. The user is notified by showing them a “Sign Up failed” message on the Sign UpUI.

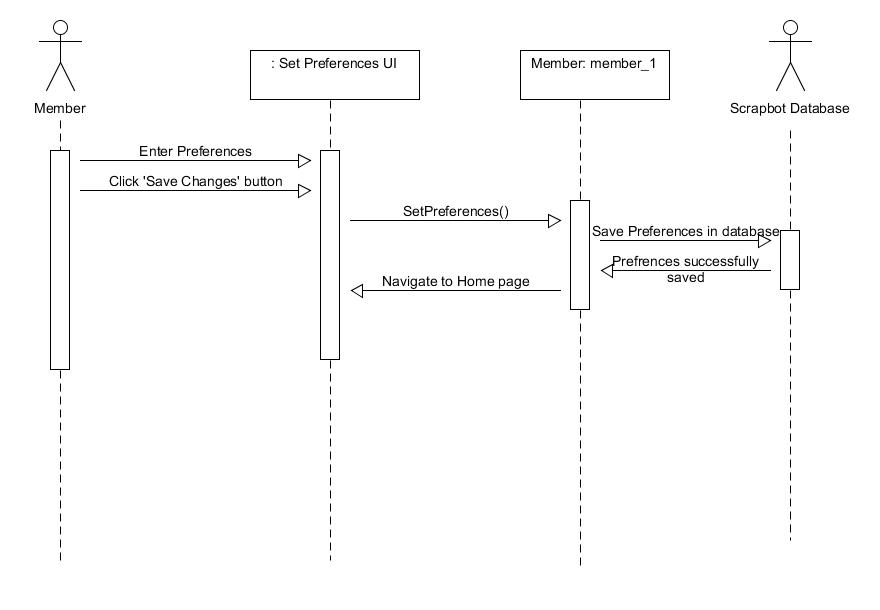
1. **Edit User Profile**

The User will click the Edit user Profile button on the User Profile UI, at which the Edit User Profile UI will be loaded. The User will enter whatever information they wish to edit and click the Save Changes button on which the Edit Profile function will be called which will validate the data entered by the user. If the validation is successful, the user account will be updated in the database and the user will be notified by a “Changes saved successfully” message after which the user will be navigated back to the User Profile UI. In case the validation fails, the user will be notified by a “could not save changes” message. No changes in the database will made in this condition.



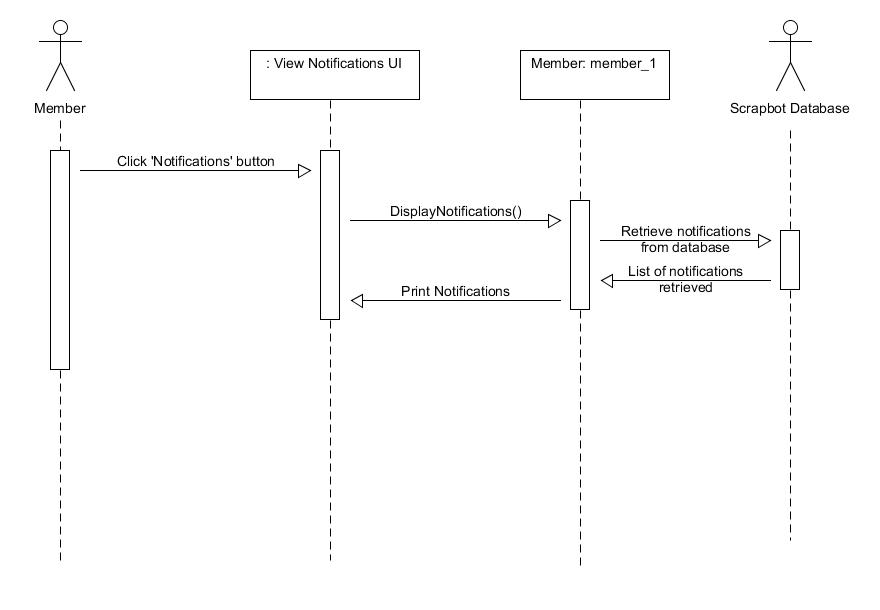
1. **Set Preferences**

The member will enter his preferences and click the Save button at which the SetPreferences function will save the data entered by the user in the database. After saving the preferences, the user will be navigated back to the Set Preferences UI with the updated changes.



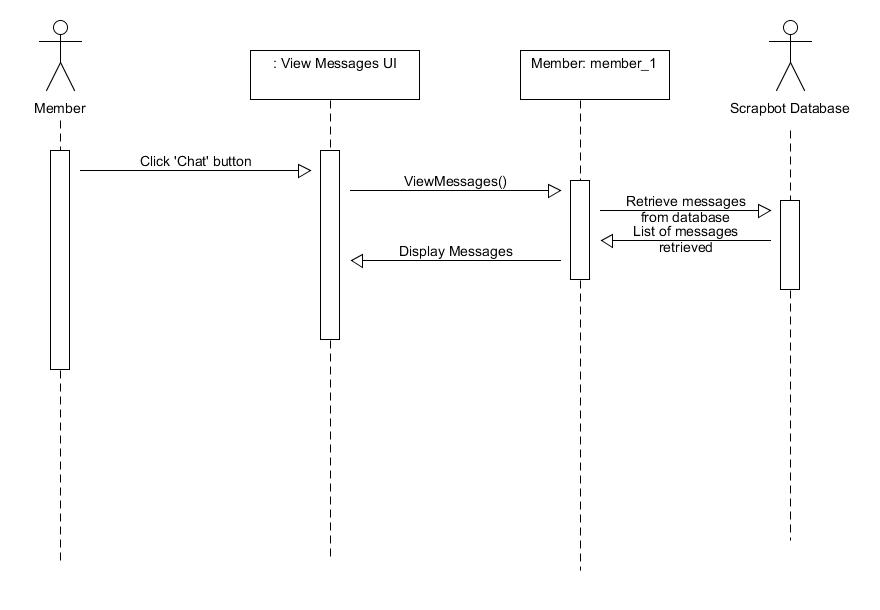
1. **View Notifications**

The User will click the Notifications button at which the DisplayNotifications function will retrieve the notifications from the database and display it all on the Notifications UI.



1. **View Messages**

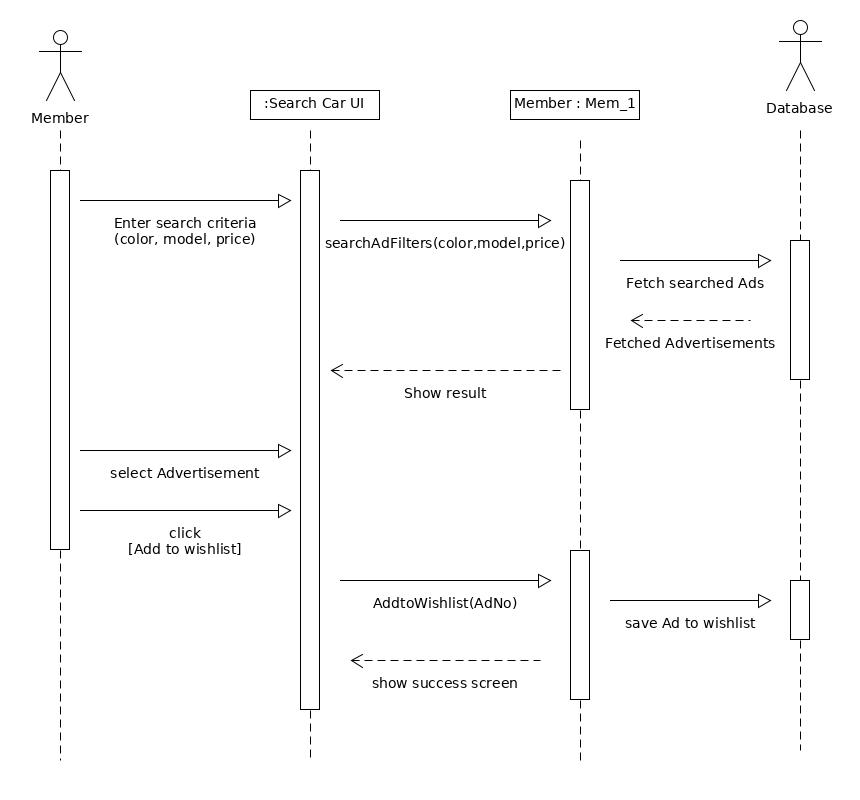
The User will click the Chat button at which the ViewMessages function will retrieve all the chats of the user and display it in a list on the View Messages UI.



* + 1. **Ad Fetch Module**

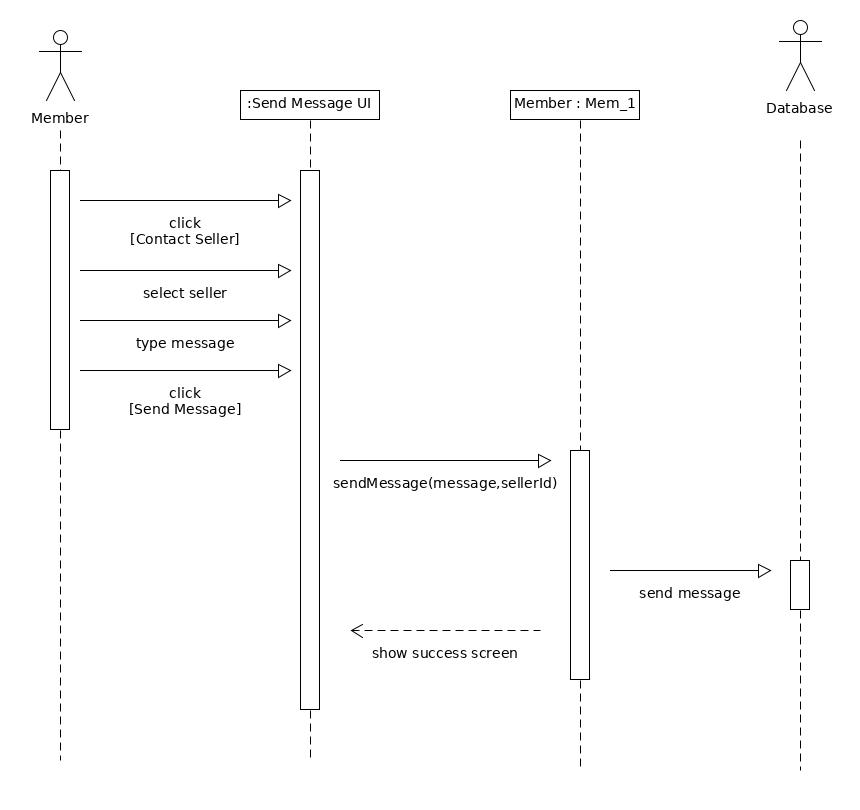
**(a) Search Car Ads**

The user enters a desired search criteria on the Search Car UI which is forwarded to the server. The server scans the database for all ads matching the selected criteria. The fetched advertisements are returned to the server which are sent to the Search Car UI. The UI displays the results to the user. He/she may select an advertisement and click the "Add to Wish list" button. The Ad is then sent to the database through the server and saved in the database maintaining the User's profile. A success screen is displayed to the user on the interface upon successful completion.



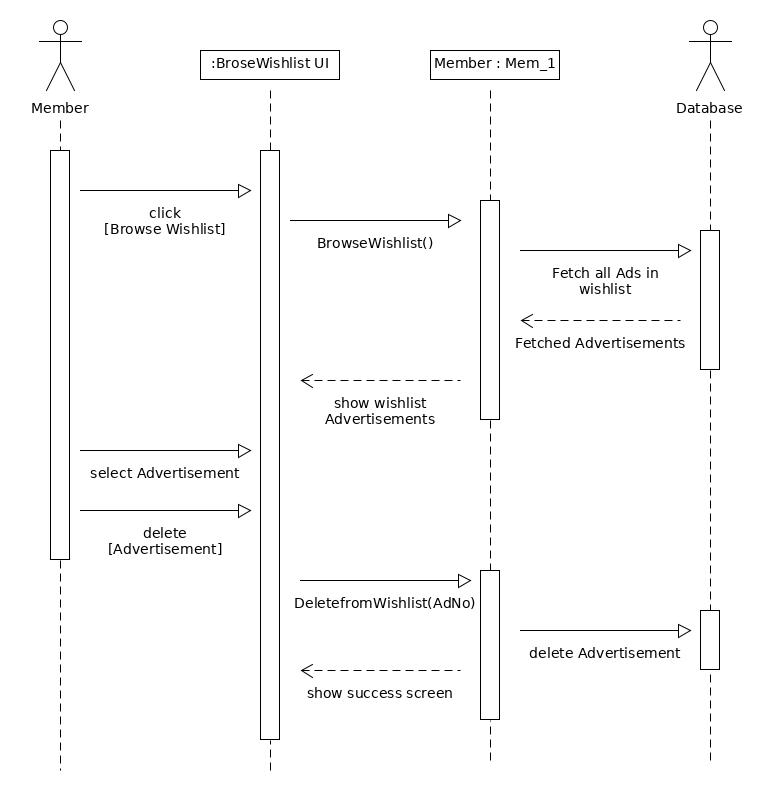
**(b) Send Message**

The user selects an ad and clicks on "Contact Seller". An interface is shown on which user selects message recipient, types in message text, and clicks "Send Message". The message with recipient's information is sent to the server. Upon success, The message is saved in the database and a success screen is shown to the user on the UI.

****

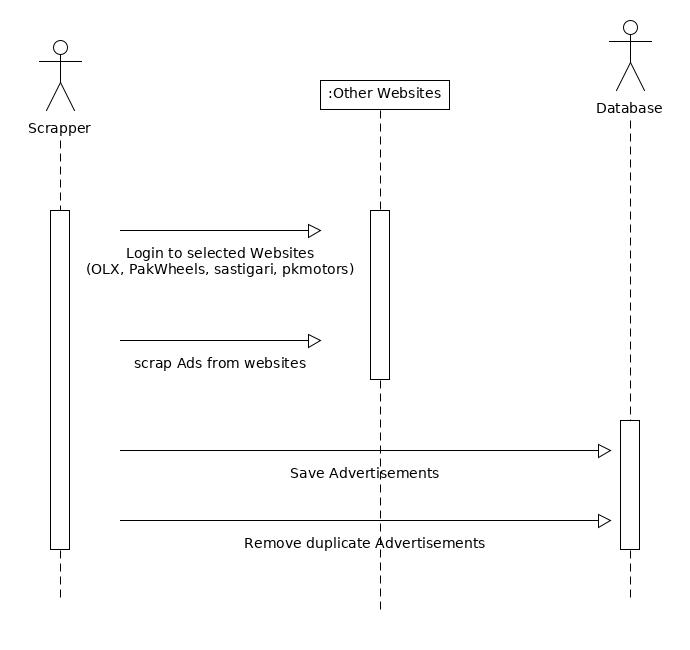
**(c) Browse Wish List**

On the dashboard, the user clicks on "Browse Wishlist". A request is sent to the server and the server fetches the list of all ads in the current user's wish list from the database. The list of ads is displayed on the UI. The user may then select an ad and press the "Delete" button which would send a request for deletion to the server. The selected ad would be removed from the database and a success screen would be shown to the user.

****

**(d) Scrap Ads From Websites**

The scrap bot accesses ads from the different websites with the ad scraping tool. It then saves all the websites to the database, once it is done, it scans the database and removes duplicate ads.

****

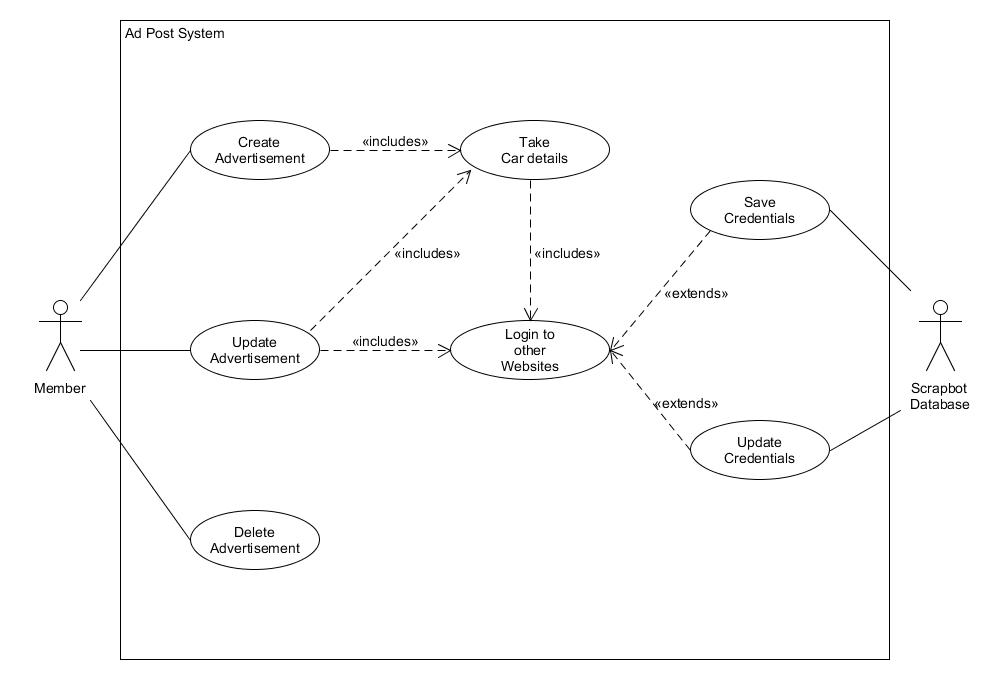
## 4.2 Data Flow Diagrams (DFD)

## 4.3 State-Transition Diagrams (STD)

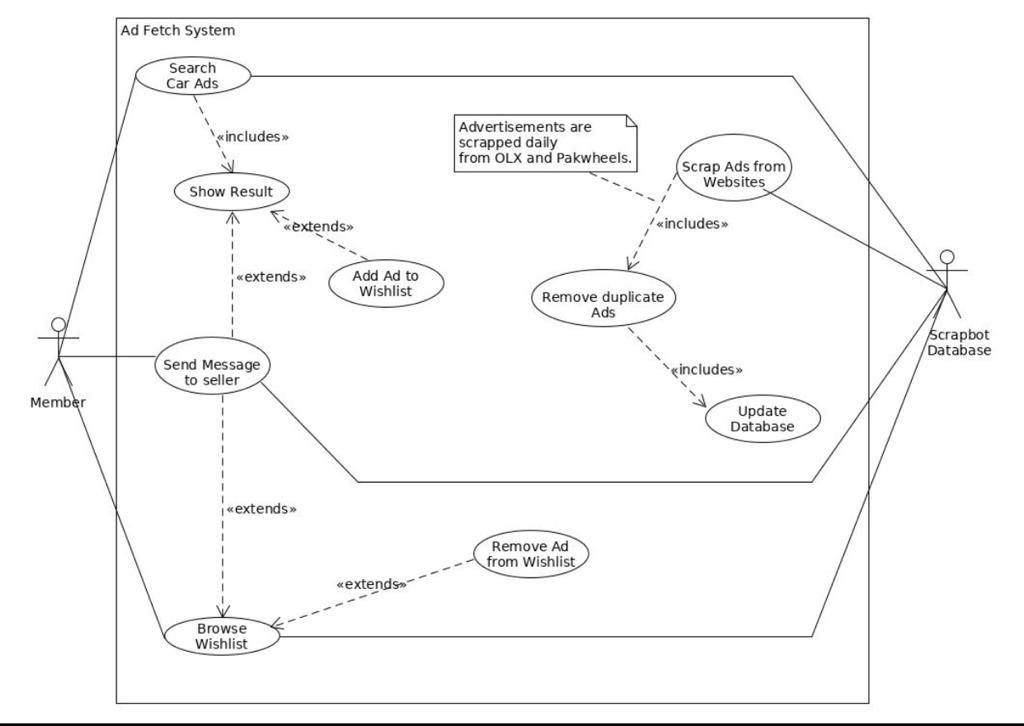
# A. Appendices

## A.1 Appendix 1

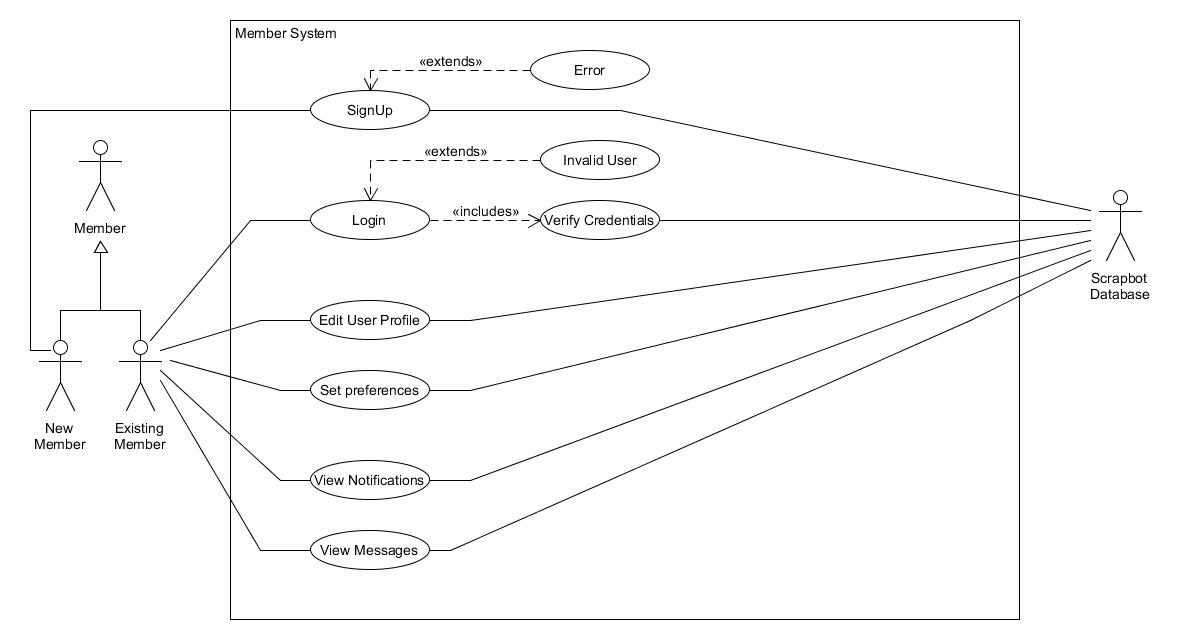
**A.1.1 Use Case Diagram - Ad Post Module**



**A.1.2 Use Case Diagram - Ad Fetch Module**



**A.1.3 Use Case Diagram - Member Module**



## 