Panda Mall

High-Level and Low-Level Design

Project Advisor:

Mr. Razi-uddin

Group Members:

Asjad Iftikhar	18L-0951
Muhammad Zain	18L-1109
Tayyab Waseem	18L-1017

National University of Computer and Emerging Sciences
Department of Computer Science
Lahore, Pakistan

Table of Contents

List of Figures

1. Introduction

Panda Mall is a web-based recommendation system designed to recommend fashion and clothing products from hundreds of online shopping stores based on the preferences of the user. This would reduce the problem of choice overload [1], bring many stores under one roof and allow startups to find potential customers. Data of fashion products will either be scrapped from various websites or from a registration form on the website and through RS model those products will be recommended to the registered customers.

1.1 Purpose of this Document

The goal of this document is to provide a full overview of our project's high- and low-level design. For a better understanding of the project, it includes development methodologies and architectural strategies to be followed, as well as class and sequence diagrams. This document tries to demonstrate the interaction of system modules in order to better depict the system's flow and operating sequence.

1.2 Scope

The scope of Panda Mall includes:

- Manual registration of products/brands and automated web scraping
- Training data in ML model using standard RS algorithms.
- Developing a web-based GUI for the users, admin and brands.

1.3 Intended Audience

- Clothing brand owners
- Online shoppers

The aim is to facilitate both shoppers and brands to get the experience of a shopping mall online.

1.4 Version

This is the first version of the high- and low-level design document for Panda Mall.

1.5 Related Documents

This document is linked to the following previously submitted documents:

- Project Proposal
- Software Requirements Specifications
- Software Functional Specifications

1.6 Prerequisite Documents

This document expands on the ideas presented in the previous documents.:

- Project Proposal
- Software Requirements Specifications
- Software Functional Specifications

1.7 Background of this Document

This document divides the architectural design supplied in the FS documents into high- and low-level designs, and adds more to them, such as a class and sequence diagram, to better illustrate how the project works.

1.8 Definitions, Acronyms, and Abbreviations

FS:	Functional specification
RS:	Recommender System
ML:	Machine Learning
GUI:	Graphical User Interface
JS:	JavaScript
OS:	Operating System

1.9 Summary

This document primarily explains the system's high- and low-level structure. The introduction provides a general idea of the project by explaining the context, goals, and purpose of the document. Then there's a system overview, which depicts the system's flow from a broader scope while highlighting the start and end of sequence. The high- and low-level diagrams, as well as the component diagram of the system, are included in the system architecture, which also specifies the architectural styles that will be implemented and how the modules will interact with one another. Furthermore, class and sequence diagrams will aid in the comprehension of the system's flow or sequence of interactions. In addition, the document covers the document's goals and guidelines, as well as the development methodologies that will be employed, component dependencies, and general policies and tactics.

2. System Overview

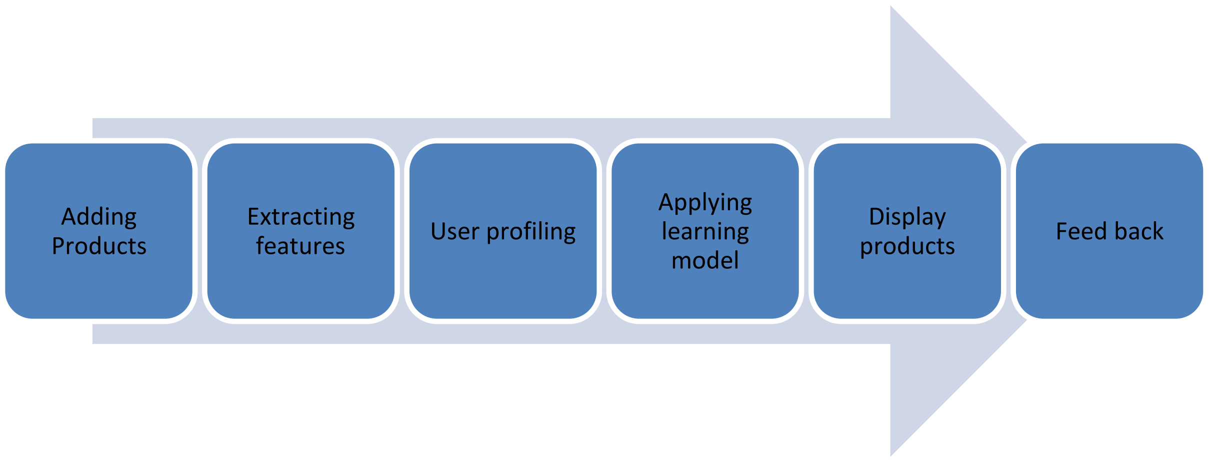

Figure 1: System Flow
Basic system flow diagram.

2.1 Adding Products

There are two ways for adding products on Panda Mall:

2.1.1 Manual

Brand accounts can fill out forms of the products they wish to add to Panda Mall and an admin must approve before publishing.

2.1.2 Automated Web Scrapping

Panda Mall System automatically scrapes product pages of popular e-commerce brands and add them to website.

2.2 Extracting Features

Once the products are added to the system. Their features like price, color is extracted for the model to learn from.

2.3 User Profiling

In the next step, customers signup and they are asked about their basic preferences as a starting point for the model to profile the features a user is most likely to appreciate or discard.

2.4 Learning Model

After the products and user data is collected, cleaned and analyzed the machine learning model uses it to create an RS system capable of rearranging products based on the user profiling.

2.5 Display Products

In the final product, when the customer logs in using valid credentials, the output of the RS system is utilized to organize the products for the particular customer preferences. They can buy the product by redirecting to the product page, add to favorites or apply filters to narrow down searches.

2.6 Feed back

Customers are reminded to give feedback of their purchases which is reused by the model to improve learning and recommendations over time.

3. Design Considerations

Before attempting to build a complete design solution, this section explains many of the challenges that must be addressed or overcome.

3.1 Assumptions and Dependencies

Following are the assumptions or dependencies regarding the software and its use. These may concern such issues as:

- User has a desktop with 2GB RAM and 64-bit Operating system.
- Users have JS enabled browsers installed.
- Users must have internet connection.
- User is familiar with understanding of basic e-commerce.

3.2 General Constraints

Following are the global limitations or constraints that have a significant impact on the design of the system's software:

3.2.1 Hardware or software environment

- We are designing a web application, so targeted systems should have JS supported browsers such as Google Chrome or Mozilla Firefox.
- In the case of hardware, web application requires 2 GB RAM, 64-bit OS.

3.2.2 End-user environment

- Users can utilize the online application with high-speed internet and a browser that supports the idea of web caching and cookies to save data for future use, as specified in the software requirements.

3.2.3 Availability or volatility of resources

- High speed and uninterrupted internet is required for better performance.

3.2.4 Interoperability requirements

- Client and server have to share and store data using common standards.

3.2.5 Interface/protocol requirements

- Http protocol will be used for communication between client and server.
- Interface will be implemented using react framework and running on browsers which are mentioned in software requirement.

3.2.6 Data repository and distribution requirements

- For web application data storage and retrieval, online cloud-based databases will be employed. It can only be changed by the administrator.

3.2.7 Security requirements (or other such regulations)

- System will be performing actions using secure protocols, secure the personal information of users by using the CSRF tokens for forms and data transfer.

- System will perform authentication and authorization.

3.2.8 Memory and other capacity limitations

- System requires CPU, Memory, I/O capacity, Bandwidth and cache space for better performance.

3.2.9 Verification and validation requirements (testing)

- The system will allow those users who have correct login credentials to conduct actions.

3.2.10 Language Constraints

- This system is only useful to those who are familiar with English language.

4. Goals and Guidelines

The following are the goals and guidelines that we are keeping in mind as we construct the software design for our system:

4.1 KISS principle

We are attempting to keep the design of our system as basic as possible in order to avoid any complexity that may lead to customer discontent. The user interface of our system must be simple to use. Furthermore, our backend implementation should be simple so that we can quickly handle any difficulties that arise.

4.2 Performance

Up to the set limit, our technology will ensure that no performance degradation occurs for users. To ensure scalability, we will be using an express server and a cloud database. Furthermore, our three-layer architecture ensures that screening products takes no longer than three seconds.

4.3 Accuracy

Our technology will ensure that the optimum algorithm is used to profile users. To guarantee that this is the case in our design, we will test as many profiling algorithms as feasible before selecting the best.

4.4 E-commerce Websites

The bulk of existing ecommerce websites, such as daraz, telemart, and elo, have a comparable user interface to ours. Although our system's backend is not the same as these websites', our frontend is similar.

5. Development Methods

We chose the agile model, specifically the scrum method [2]. When it comes to development, the main justification for using the scrum model is its productivity and quality. In addition, this paradigm aids in the rapid development of our software. In addition, we held daily meetings with each other to keep track of our development and performance. Another advantage that drew us to this approach over others was its flexibility to adapt to changing requirements as we went along. We used the Scrum paradigm to break tasks into sprints, which simplified our job. Following were the steps in each sprint:

1. Planning
2. Implementation
3. Review
4. Retrospect

We initially examined two methods: the agile model, which we ultimately chose, and the prototype model. The following are the main reasons for not implementing the prototype model:

1. Poor documentation because of changing system requirements.
2. Incomplete problem analysis.
3. Increases the complexity of the system.

6. Architectural Strategies

6.1 ReactJS, Node JS, Python, Express Server, Mongo DB

When it comes to machine learning there is no better option than Python. As it provides different types of libraries to manipulate and perform different functions on large amount of data. We will be using python with Jupiter Notebook.

Our system's front end will be designed using ReactJS. It provides us with JavaScript libraries that helps us to perform our work. Major benefit of using ReactJS is its reusable components that allows us to use same components for different pages wherever needed. Moreover, ReactJS enables significant data changes that result in automatic alteration in the selected parts of user interfaces.

We will be using NodeJS with python as our backend languages. Every other aspect expects for manipulating datasets will be handled by NodeJS. The reason for using NodeJS is its compatibility with ReactJS which we will be using as our frontend.

For our server we have decided to use Express Server. As we are using JavaScript for our frontend and backend so why not using it to implement the server. Major benefit of using this server is its ability to handle several requests efficiently. Moreover, it also has a highly supportive open-source community.

For our database we will be using MongoDB because of its scalability benefits. It has a very simple design with all the data stored as a single JSON object hence making key functions like saving and loading data relatively easier.

6.2 Future plans for enhancing software

For this project we are limiting our scope to a web application but in the future, we are planning to scale our project by:

- We are planning to implement a cart system that we will integrate with the payment procedure to help our users to buy product directly from our website.
- We plan to develop a mobile application with all the features of our web application in order to assist the user.

6.3 User interface paradigms

While making our UI we will implement the eight golden rules of user interaction in our Software Engineering course. Following are those rules:

1. Strive for consistency.
2. Seek universal usability.
3. Offer informative feedback.
4. Design dialogs.
5. Prevent errors.
6. Permit easy reversal of actions.
7. Keep users in control.
8. Reduce short term memory load.

This will help our UI to be able to yield better performance and efficiency. It will also make our UI user friendly.

6.4 Error detection and recovery

In our situation, the most typical error occurs during the authentication process, when a user enters incorrect credentials, which our system authenticates and, if they are invalid, displays an error message to the user.
Furthermore, if a user inputs incorrect information during registration or profile modification, our system will display an error message requiring the user to reenter the incorrect information.
If a user attempts to purchase a product that the vendor has rendered unavailable, our system will route the user to an error page informing them of the product's unavailability.
Furthermore, we will update the product list once a week to ensure that users have access to the most up-to-date products.

6.5 Concurrency and Synchronization

Hopefully a lot of users will be accessing our website and as result making lot of simultaneous requests. So, we need to make our system synchronized so that we can treat each request independently. In order to do that we have to implement threading into our business logic.

6.6 Memory management policies

As already stated above we will be using MongoDB which in an online database running in real-time on cloud. So, in order to access that DB our system users should be connected to the internet.

7. System Architecture

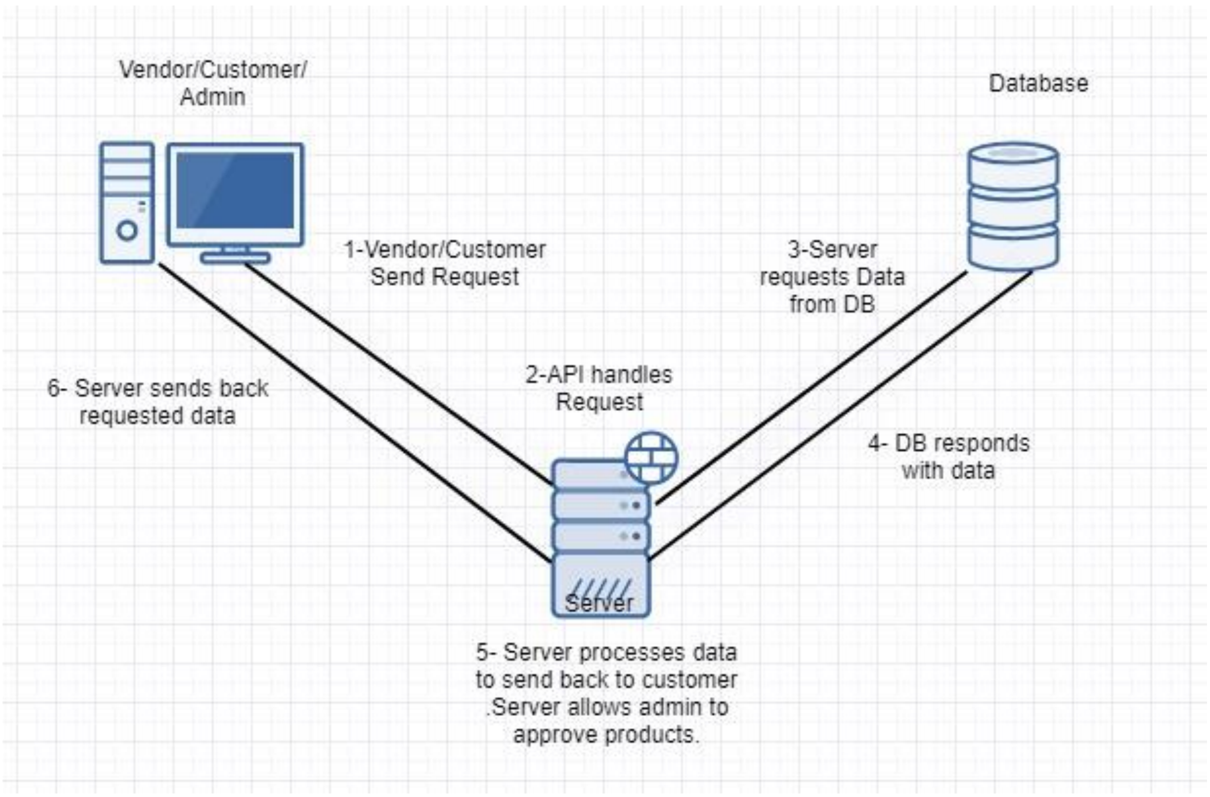

Figure 2: High Level System Architecture
Diagram for high level system architecture.

As show in the diagram above we will be using 3-layer architecture for our system. Our users (Customer, Vendor and Admin) will interact with the presentation layer that will forward user requests to the server which will be acting as the business layer. It will process all the user request and send them back to the presentation layer. If needed it will interact with our cloud-based database that corresponds to the database layer to retrieve any kind of information needed.

7.1 Front-end

Our front-end has the following components:

7.1.1 Customer

Customer is the user that logs in to our system using his account and interacts with the application to buy products. Each customer request goes from the presentation layer to the server where the request is processed and the required data is fetched from the database layer and then the result in forwarded back to the presentation layer where the results are shown to the customer. Customer can perform following requests/functions:

- Login
- Sign up
- Reset password
- Logout
- Edit Profile
- Buy product
- Add products to favorites

- Give feedback on products
- Filter products based on several filters

7.1.2 Vendor

Vendor is the user which creates a store and then adds his/her products to our website. Each vendor request goes from the presentation layer to the server where the request is processed and the given data is stored to the database layer. Vendor can perform following requests/functions

- Add products
- Update products
- View products
- Remove products

7.1.3 Admin

Admin is the user of the system which approves the newly added products. If he approved a product then the product is added to the database otherwise the product is discarded.

- Approve/Reject products
- View products

7.2 Back-end

Our Back-end has the following components:

7.2.1 Express Server

Our server connects our database layer to the presentation layer. It processes all the requests coming from the users (presentation layer) in HTML format. After processing the request, it fetches the required information from the database layer in the form of JSON objects. After that the server responds back to the presentation layer by sending the requested information.

7.2.2 MongoDB

This is our database where all the information about system users, products is stored in JSON format. Our database is hosted on the cloud so as a result it is scalable and efficient. It assists our server by providing required information needed to respond to user requests.

7.3 Subsystem Architecture

There is no such component in our system architecture section that merits a detailed discussion.

8. Class Diagram

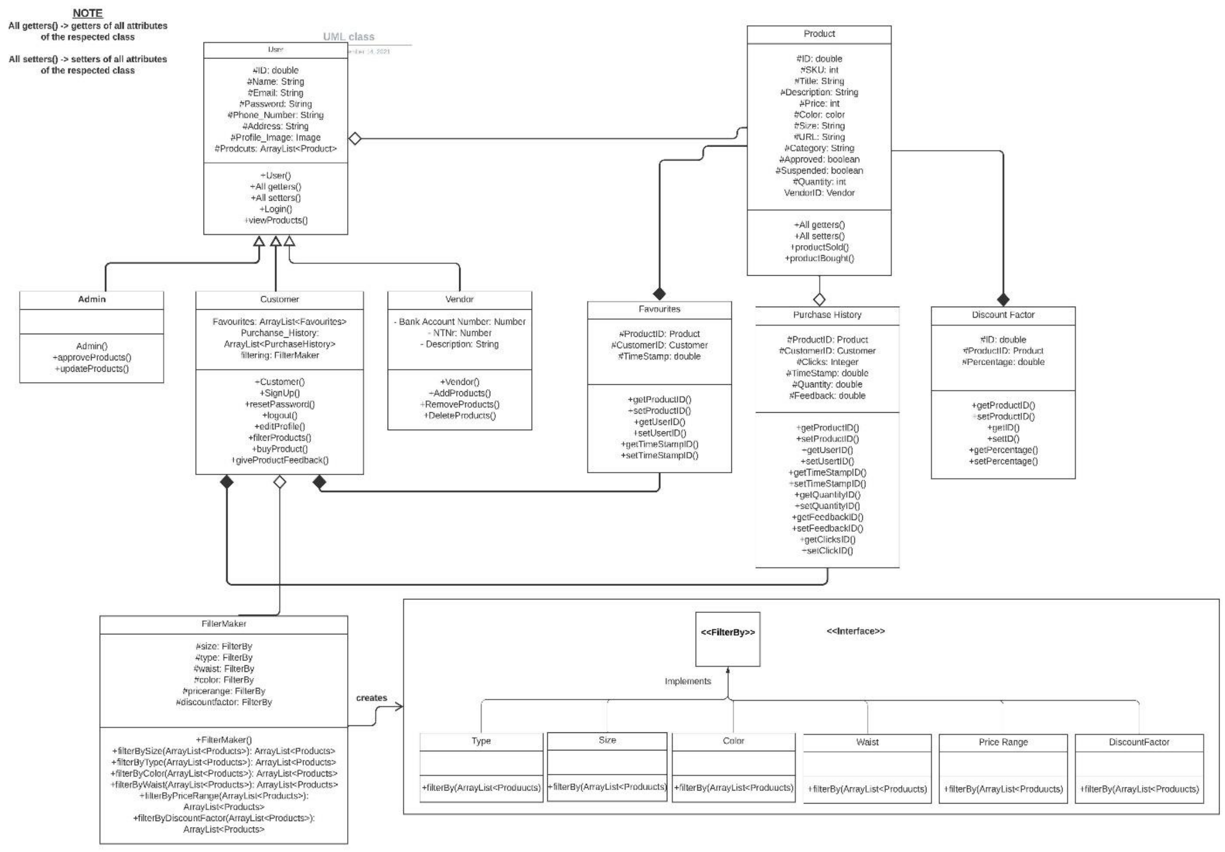

Figure 3: Class Diagram

Diagram for high level system architecture.

9. Sequence Diagrams

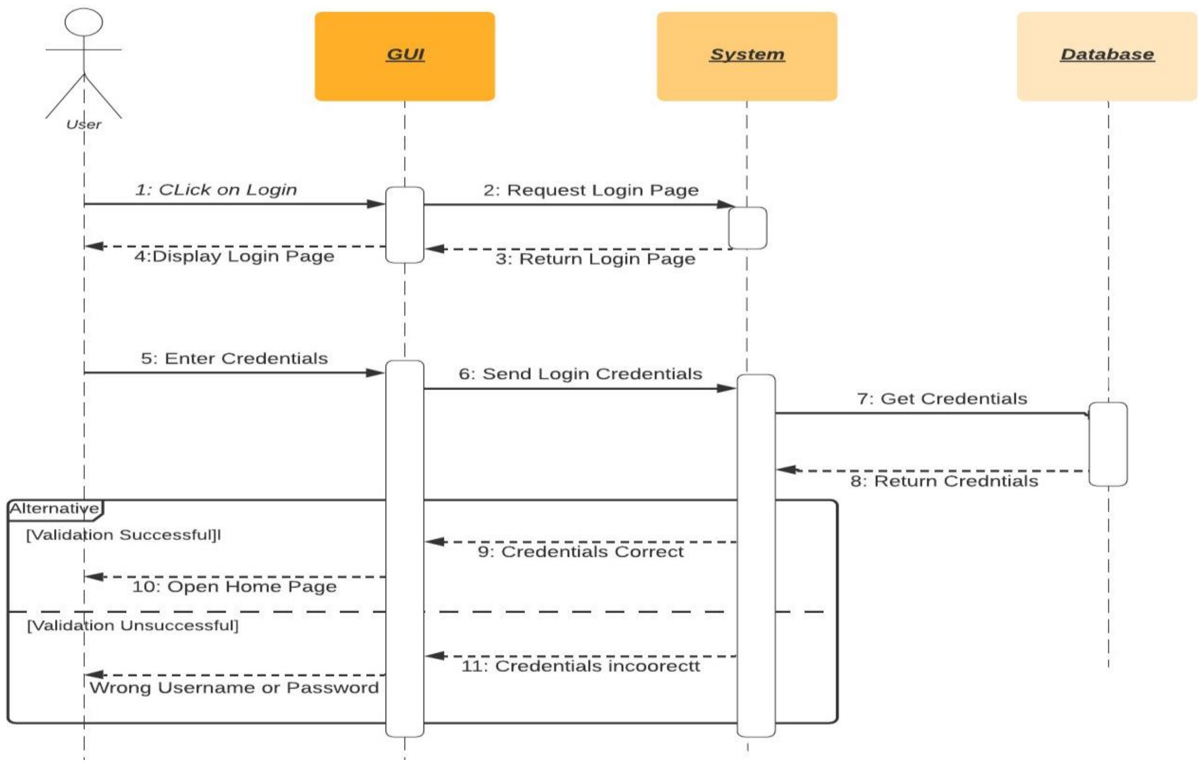

Figure 4: Login Account
This is the sequence diagram of Login Account for Users.

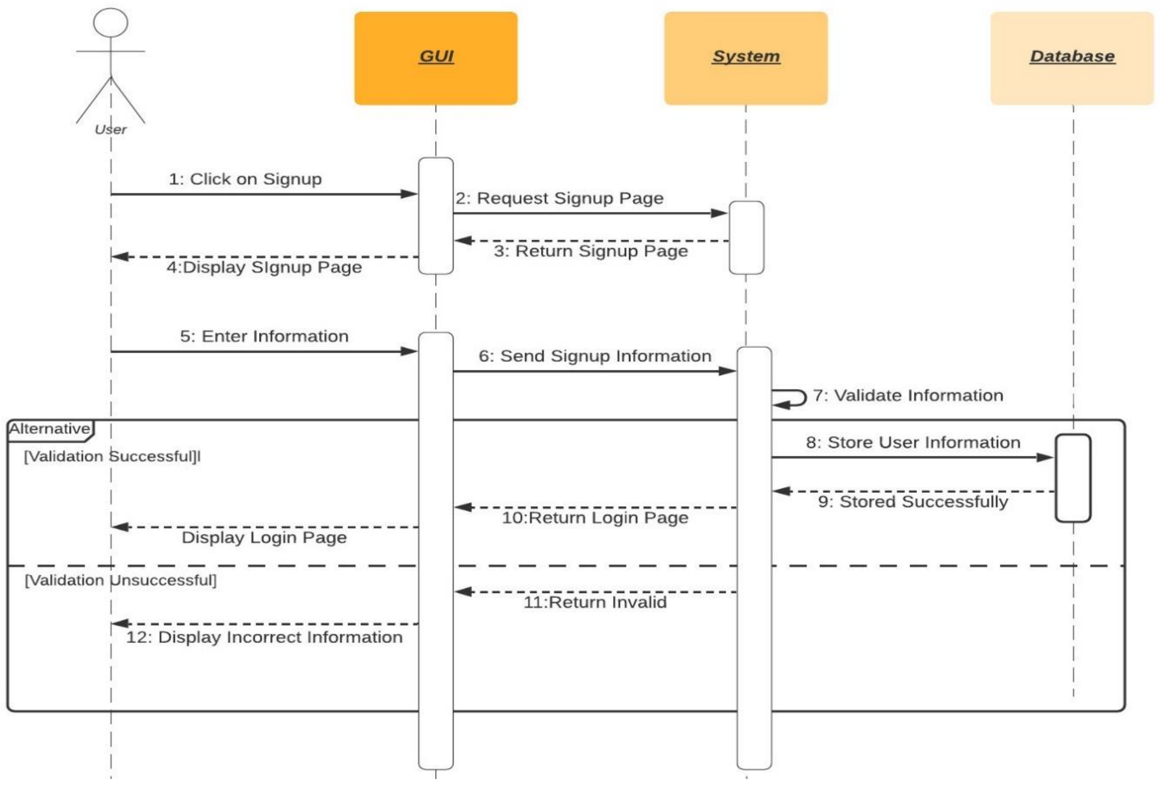

Figure 5: Create Account/Signup
This is the sequence diagram of Signup for Users.

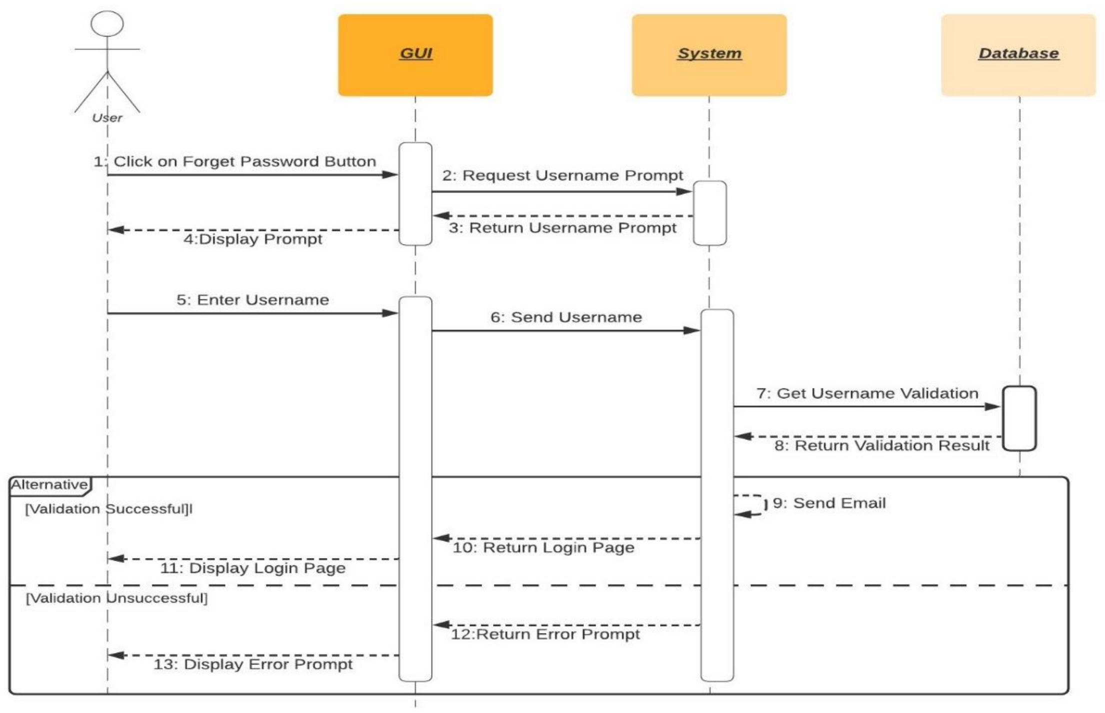

Figure 6: Reset/Forget Password
This is the sequence diagram of Reset/Forget Password for Users.

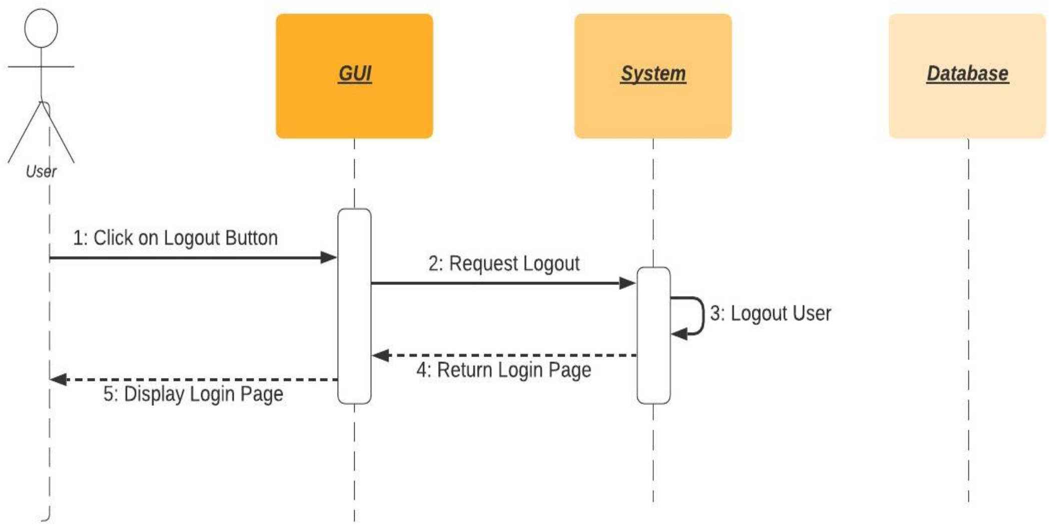

Figure 7: Logout
This is the sequence diagram of Logout Procedure for Users.

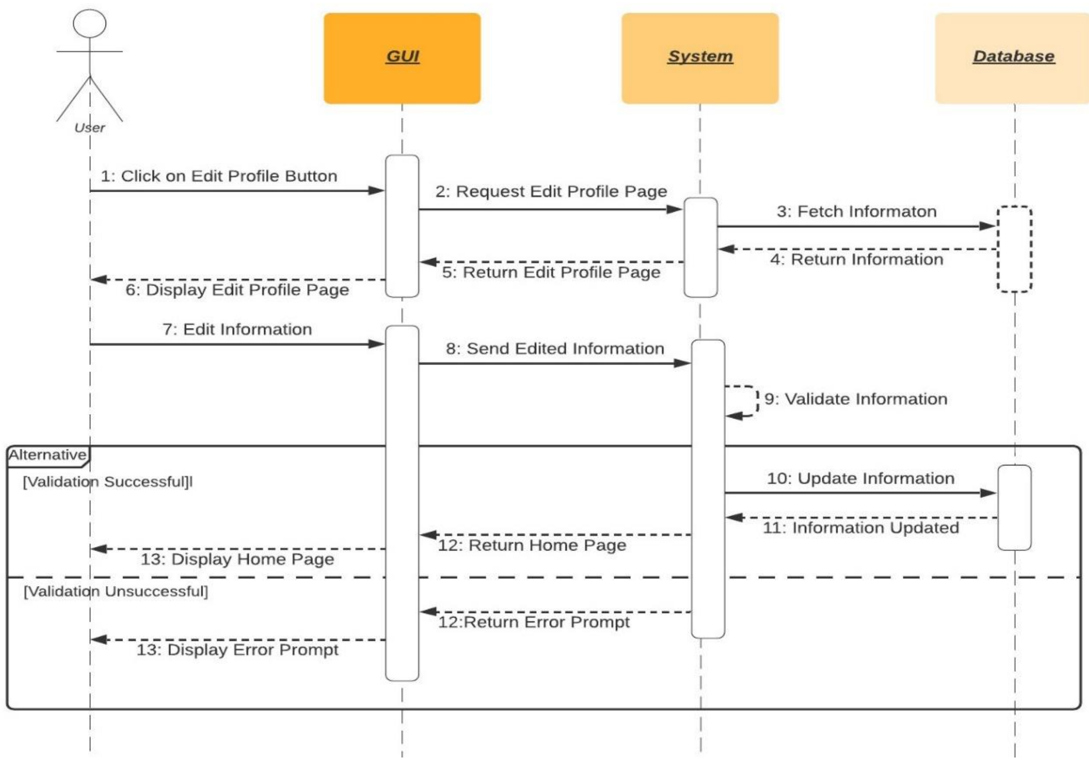

Figure 8: Edit Profile
This is the sequence diagram of Edit Profile for Users

.

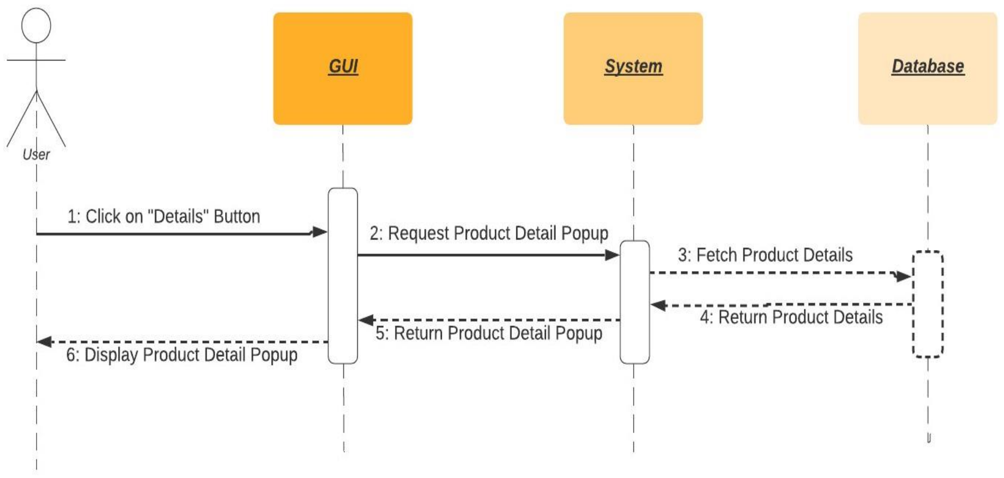
GUI
System
Database
User
1: Click on "Details" Button
2: Request Product Detail Popup
3: Fetch Product Details
4: Return Product Details
5: Return Product Detail Popup
6: Display Product Detail Popup

Figure 9: View Product Details
This is the sequence diagram of View Product Details.

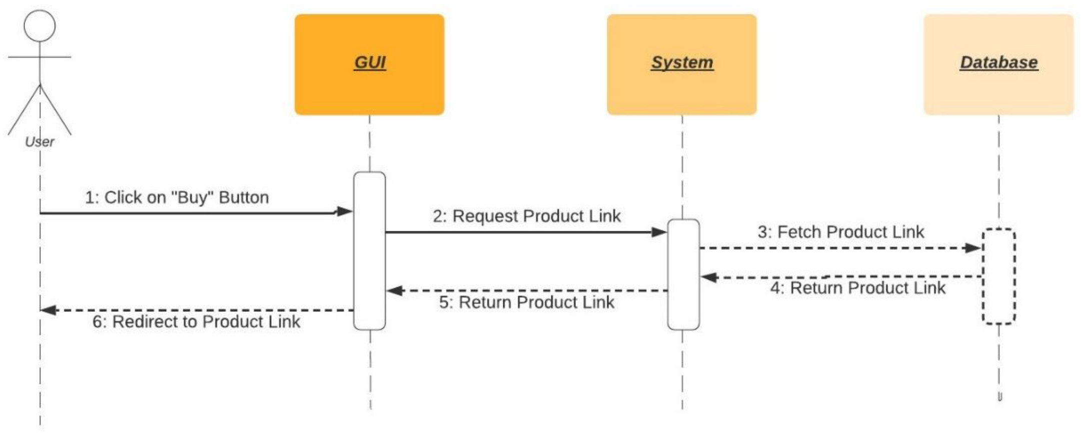

Figure 10: Buy Product
This is the sequence diagram of Buy a Product Procedure.

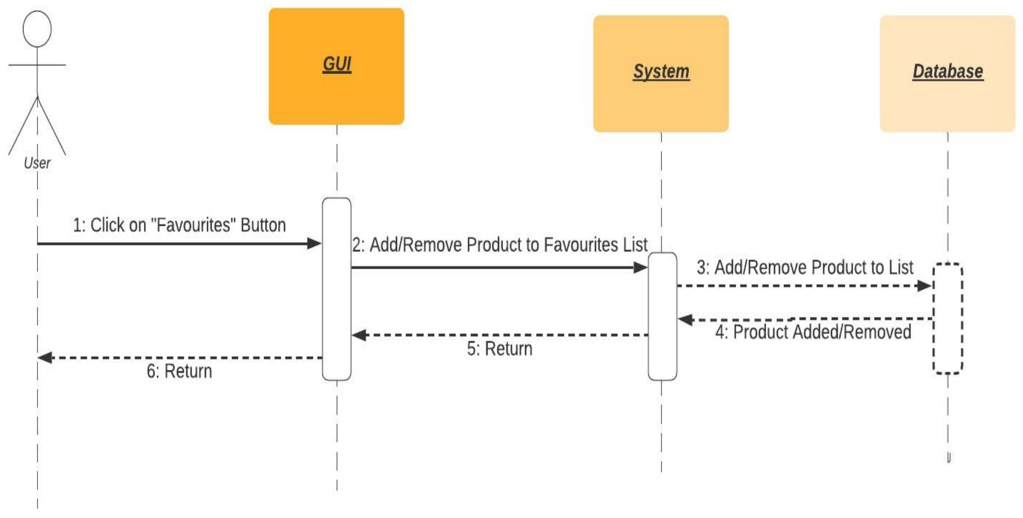
User
GUI
System
Database
1: Click on "Favourites" Button
2: Add/Remove Product to Favourites List
3: Add/Remove Product to List
4: Product Added/Removed
5: Return
6: Return

Figure 11: Add/Remove favorites
This is the sequence diagram of Removing and Adding Products from Favourites list.

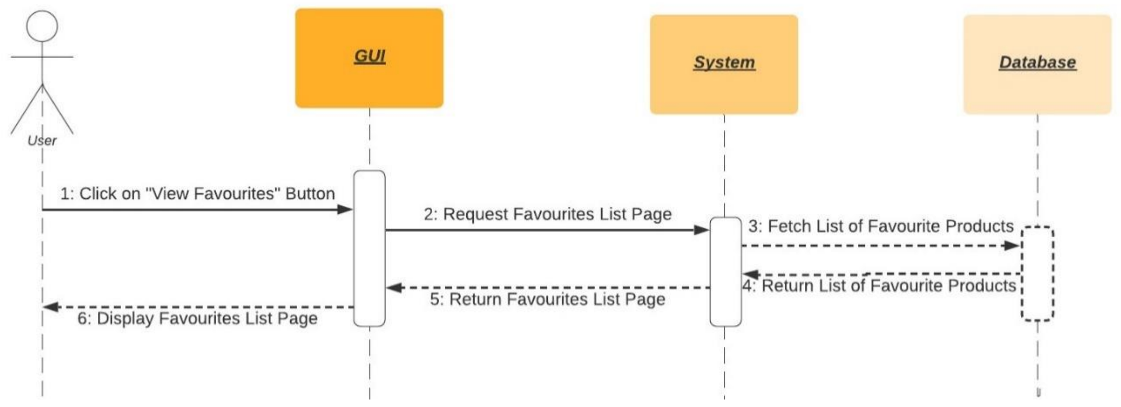
User
GUI
System
Database
1: Click on "View Favourites" Button
2: Request Favourites List Page
3: Fetch List of Favourite Products
4: Return List of Favourite Products
5: Return Favourites List Page
6: Display Favourites List Page

Figure 12: View Favorites List
This is the sequence diagram of Viewing Favourites List.

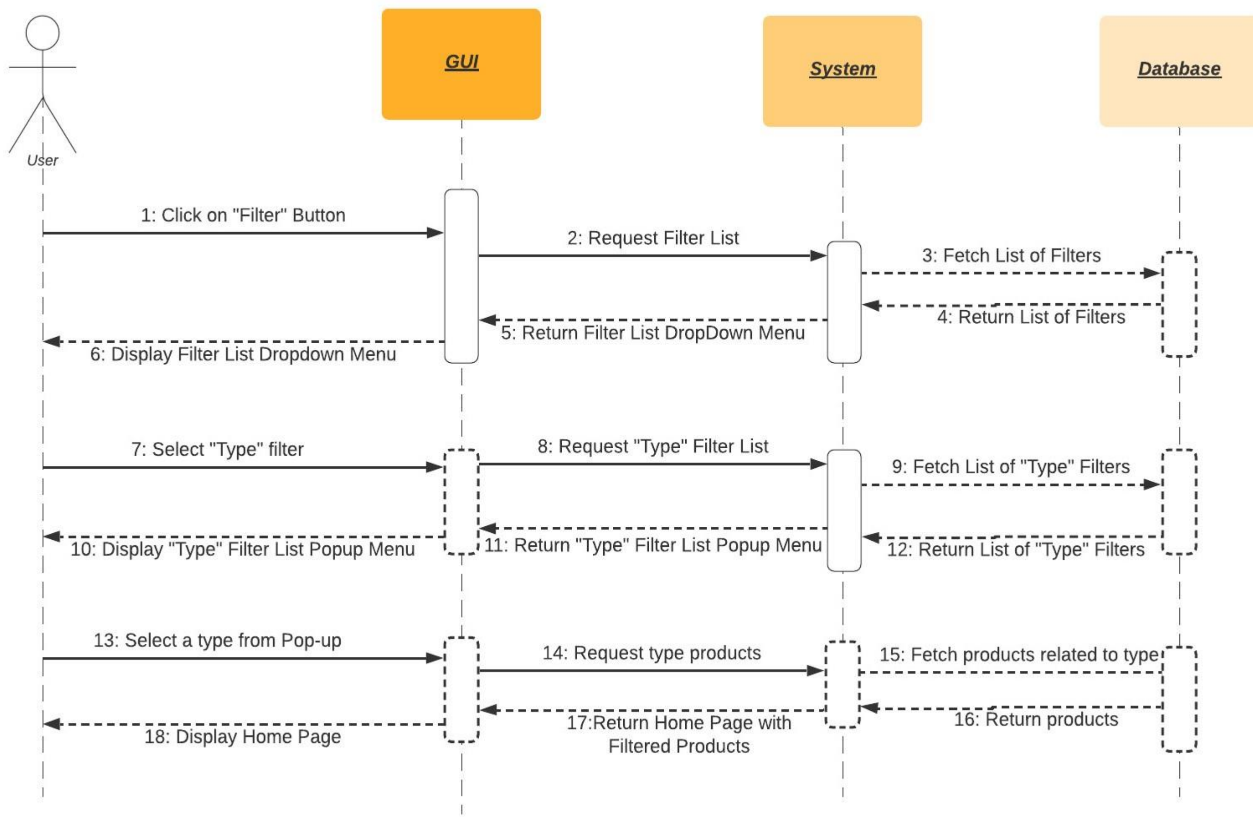
User
GUI
System
Database
1: Click on "Filter" Button
2: Request Filter List
3: Fetch List of Filters
4: Return List of Filters
5: Return Filter List DropDown Menu
6: Display Filter List Dropdown Menu
7: Select "Type" filter
8: Request "Type" Filter List
9: Fetch List of "Type" Filters
10: Display "Type" Filter List Popup Menu
11: Return "Type" Filter List Popup Menu
12: Return List of "Type" Filters
13: Select a type from Pop-up
14: Request type products
15: Fetch products related to type
16: Return products
17:Return Home Page with Filtered Products
18: Display Home Page

Figure 13: Filter by Type
This is the sequence diagram of filtering products by their "Type".

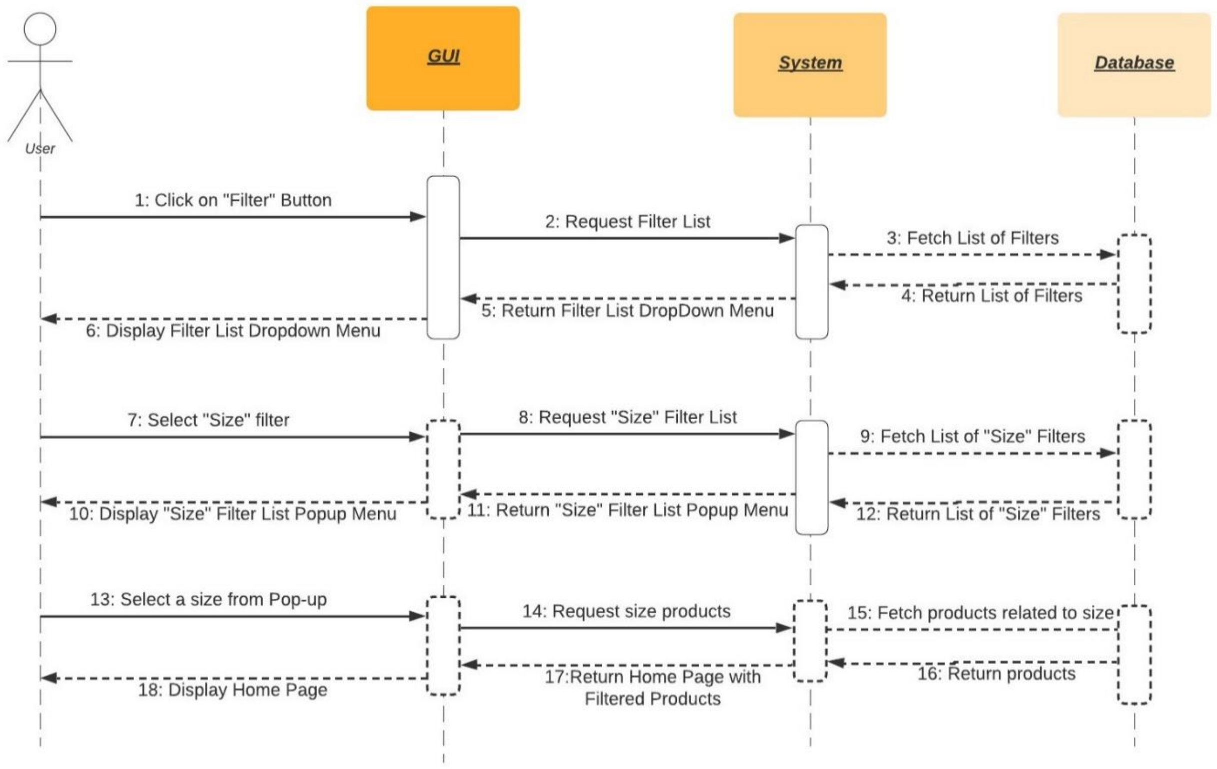
User
GUI
System
Database
1: Click on "Filter" Button
2: Request Filter List
3: Fetch List of Filters
4: Return List of Filters
5: Return Filter List DropDown Menu
6: Display Filter List Dropdown Menu
7: Select "Size" filter
8: Request "Size" Filter List
9: Fetch List of "Size" Filters
10: Display "Size" Filter List Popup Menu
11: Return "Size" Filter List Popup Menu
12: Return List of "Size" Filters
13: Select a size from Pop-up
14: Request size products
15: Fetch products related to size
16: Return products
17:Return Home Page with Filtered Products
18: Display Home Page

Figure 14: Filter by Size
This is the sequence diagram of filtering products by their "Size".

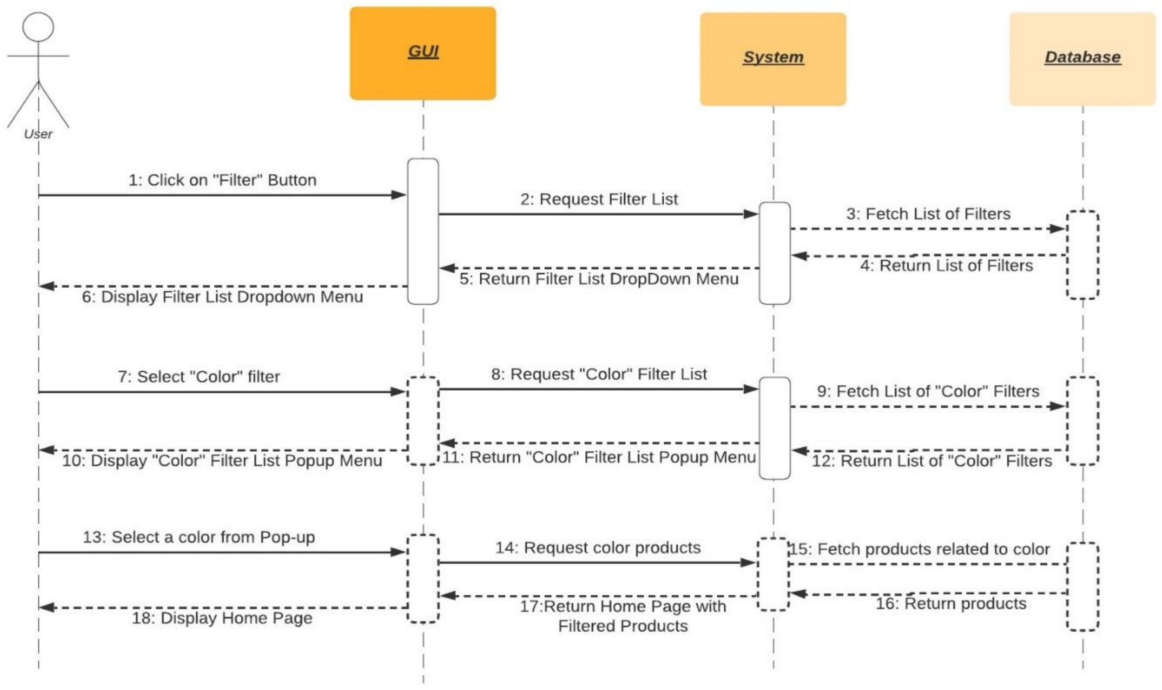
User
GUI
System
Database
1: Click on "Filter" Button
2: Request Filter List
3: Fetch List of Filters
4: Return List of Filters
5: Return Filter List DropDown Menu
6: Display Filter List Dropdown Menu
7: Select "Color" filter
8: Request "Color" Filter List
9: Fetch List of "Color" Filters
10: Display "Color" Filter List Popup Menu
11: Return "Color" Filter List Popup Menu
12: Return List of "Color" Filters
13: Select a color from Pop-up
14: Request color products
15: Fetch products related to color
16: Return products
17:Return Home Page with Filtered Products
18: Display Home Page

Figure 15: Filter by Color
This is the sequence diagram of filtering products by their "Color".

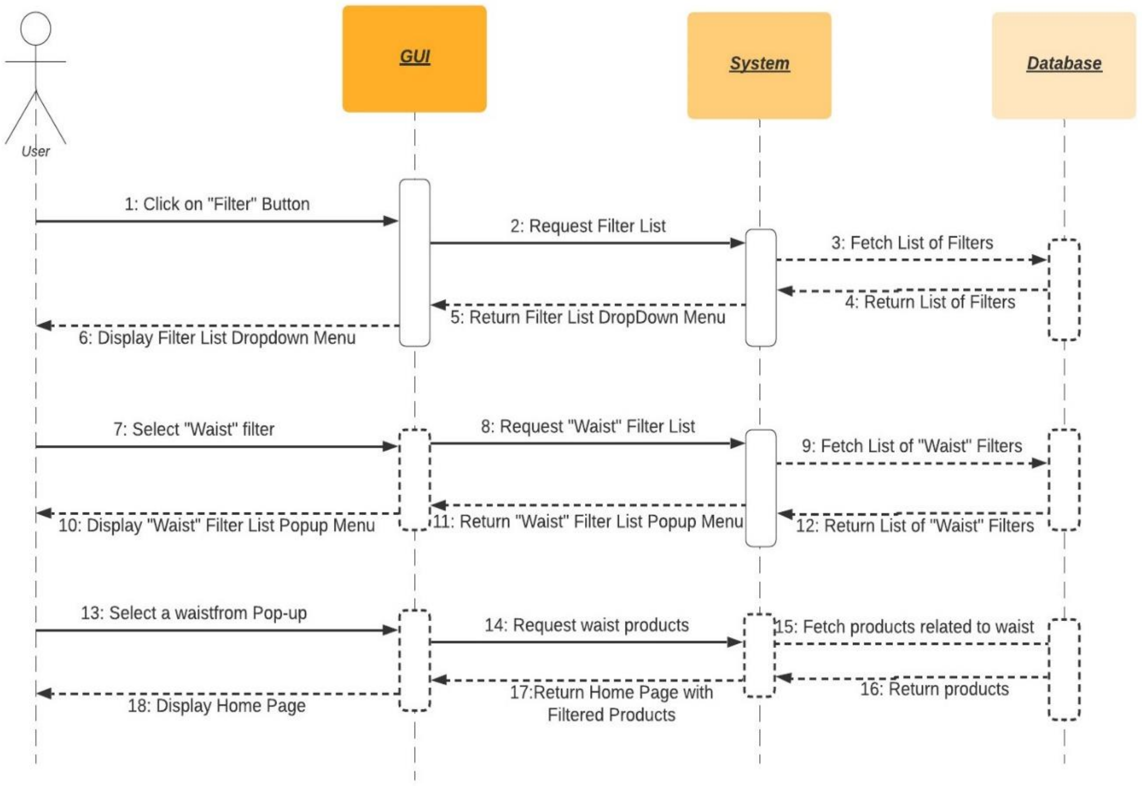
User
GUI
System
Database
1: Click on "Filter" Button
2: Request Filter List
3: Fetch List of Filters
4: Return List of Filters
5: Return Filter List DropDown Menu
6: Display Filter List Dropdown Menu
7: Select "Waist" filter
8: Request "Waist" Filter List
9: Fetch List of "Waist" Filters
10: Display "Waist" Filter List Popup Menu
11: Return "Waist" Filter List Popup Menu
12: Return List of "Waist" Filters
13: Select a waistfrom Pop-up
14: Request waist products
15: Fetch products related to waist
16: Return products
17:Return Home Page with Filtered Products
18: Display Home Page

Figure 16: Filter by Waist
This is the sequence diagram of filtering products by their "Waist".

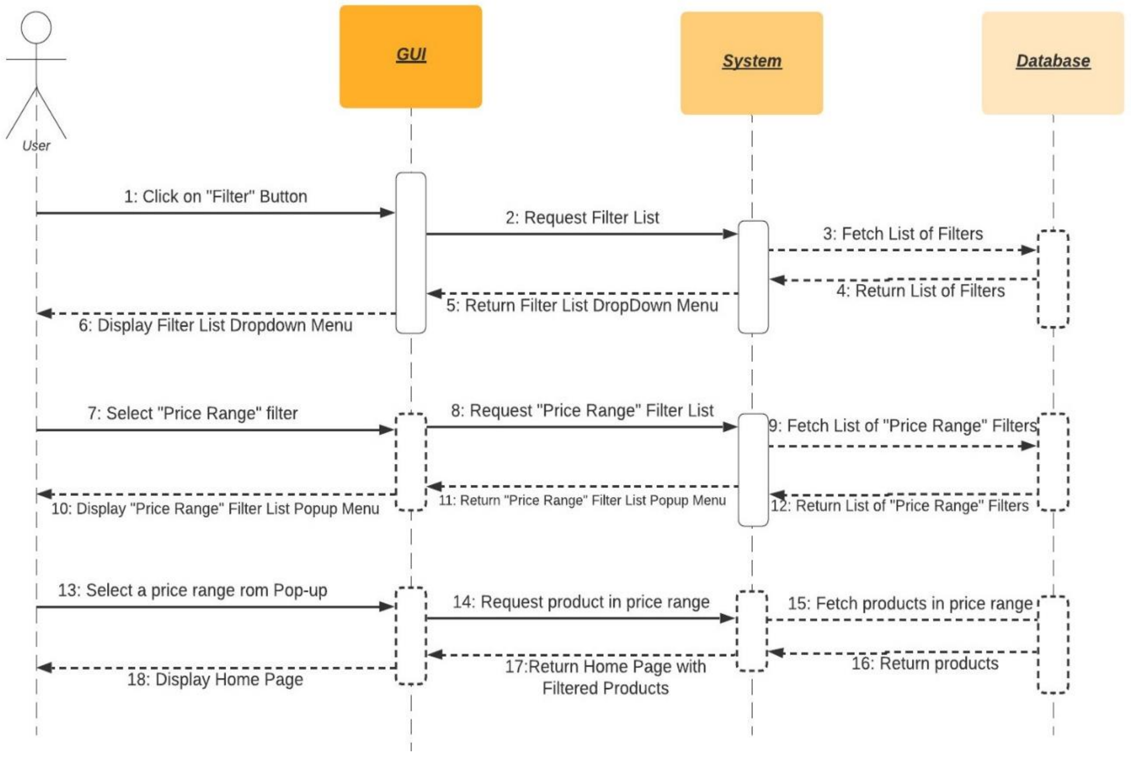
User
GUI
System
Database
1: Click on "Filter" Button
2: Request Filter List
3: Fetch List of Filters
4: Return List of Filters
5: Return Filter List DropDown Menu
6: Display Filter List Dropdown Menu
7: Select "Price Range" filter
8: Request "Price Range" Filter List
9: Fetch List of "Price Range" Filters
10: Display "Price Range" Filter List Popup Menu
11: Return "Price Range" Filter List Popup Menu
12: Return List of "Price Range" Filters
13: Select a price range rom Pop-up
14: Request product in price range
15: Fetch products in price range
16: Return products
17:Return Home Page with Filtered Products
18: Display Home Page

Figure 17: Filter by Price Range
This is the sequence diagram of filtering products by their "Price Range".

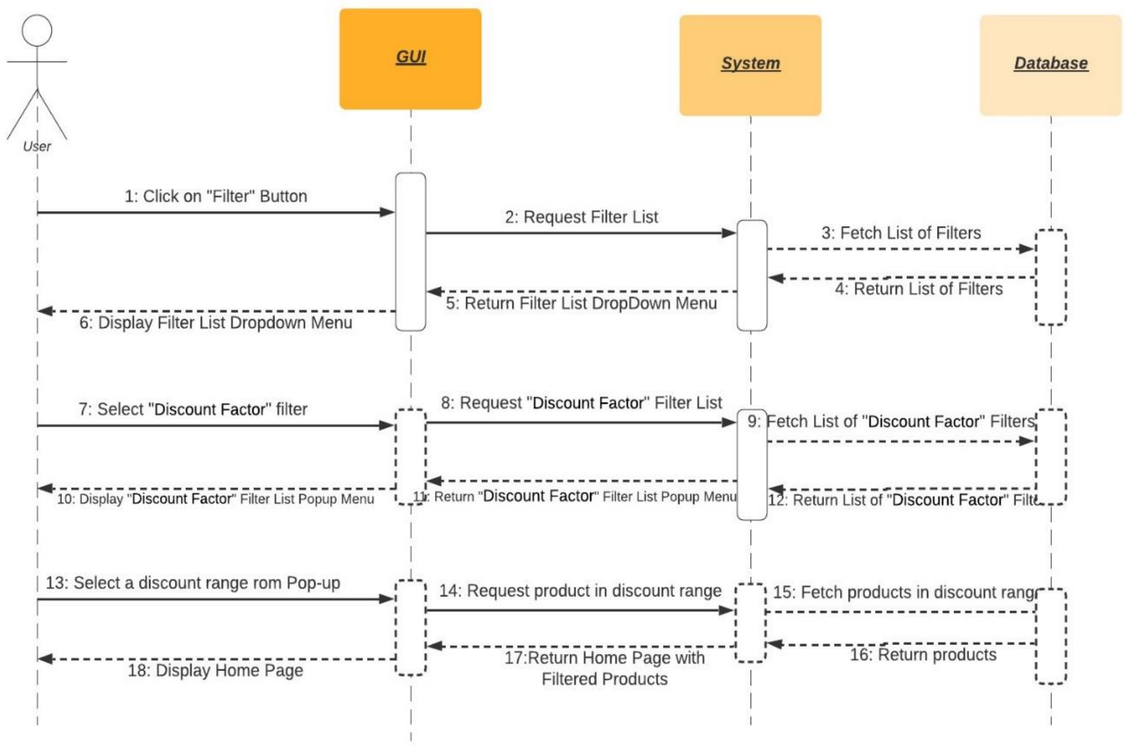
User
GUI
System
Database
1: Click on "Filter" Button
2: Request Filter List
3: Fetch List of Filters
4: Return List of Filters
5: Return Filter List DropDown Menu
6: Display Filter List Dropdown Menu
7: Select "Discount Factor" filter
8: Request "Discount Factor" Filter List
9: Fetch List of "Discount Factor" Filters
10: Display "Discount Factor" Filter List Popup Menu
11: Return "Discount Factor" Filter List Popup Menu
12: Return List of "Discount Factor" Filte
13: Select a discount range rom Pop-up
14: Request product in discount range
15: Fetch products in discount rang
16: Return products
17:Return Home Page with Filtered Products
18: Display Home Page

Figure 18: Filter by Discount Factor
This is the sequence diagram of filtering products by their "Discount Factor".

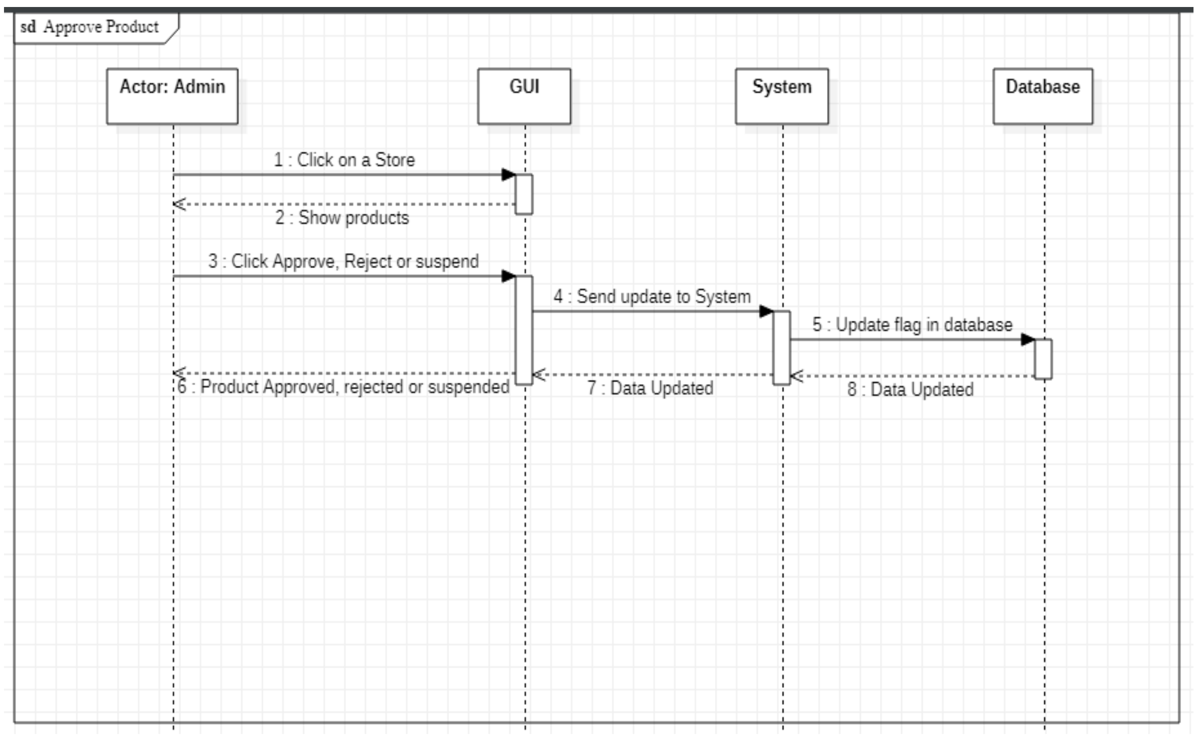
sd Approve Product
Actor: Admin
GUI
System
Database
1 : Click on a Store
2 : Show products
3 : Click Approve, Reject or suspend
4 : Send update to System
5 : Update flag in database
6 : Product Approved, rejected or suspended
7 : Data Updated
8 : Data Updated

Figure 19: Product Approval
This is the sequence diagram of product approval, rejection or suspension.

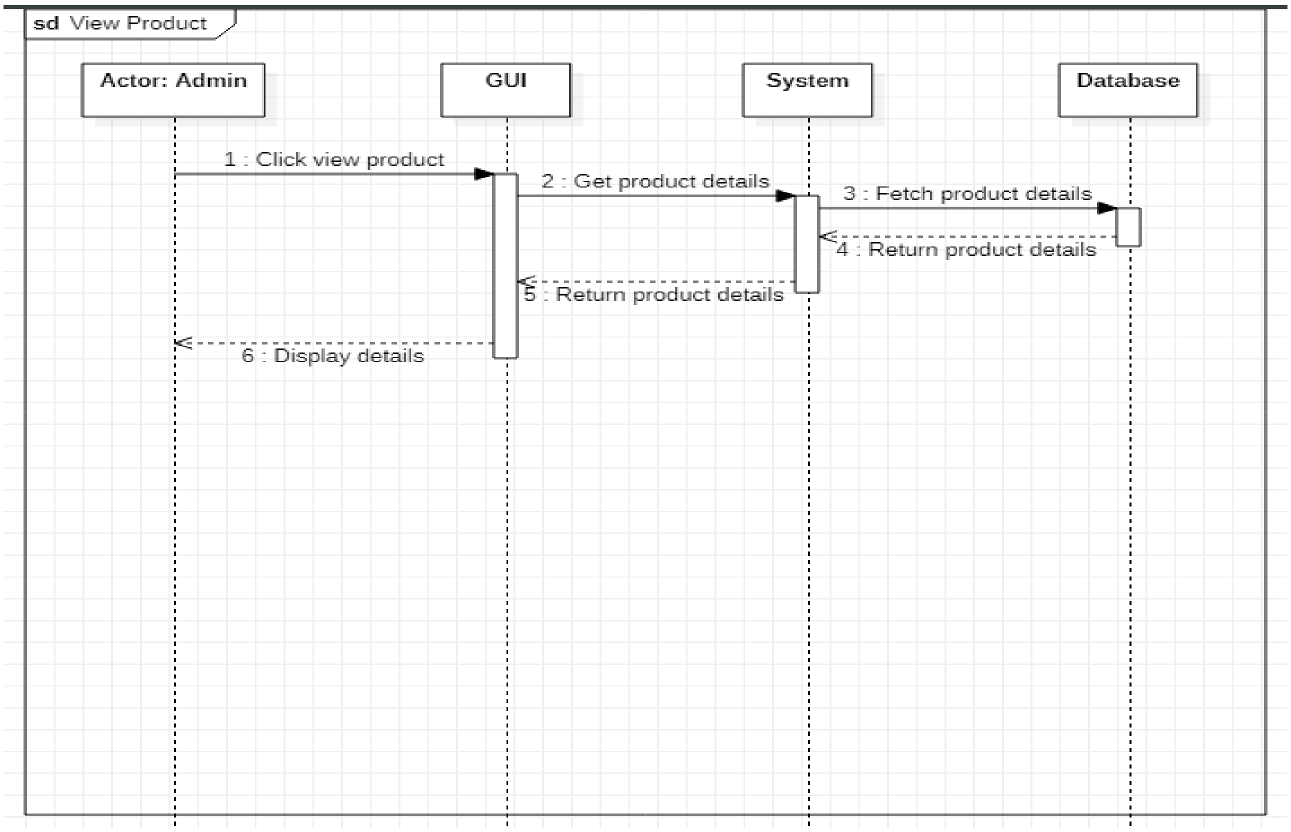

Figure 20: View Product
This is the sequence diagram of admin view product details.

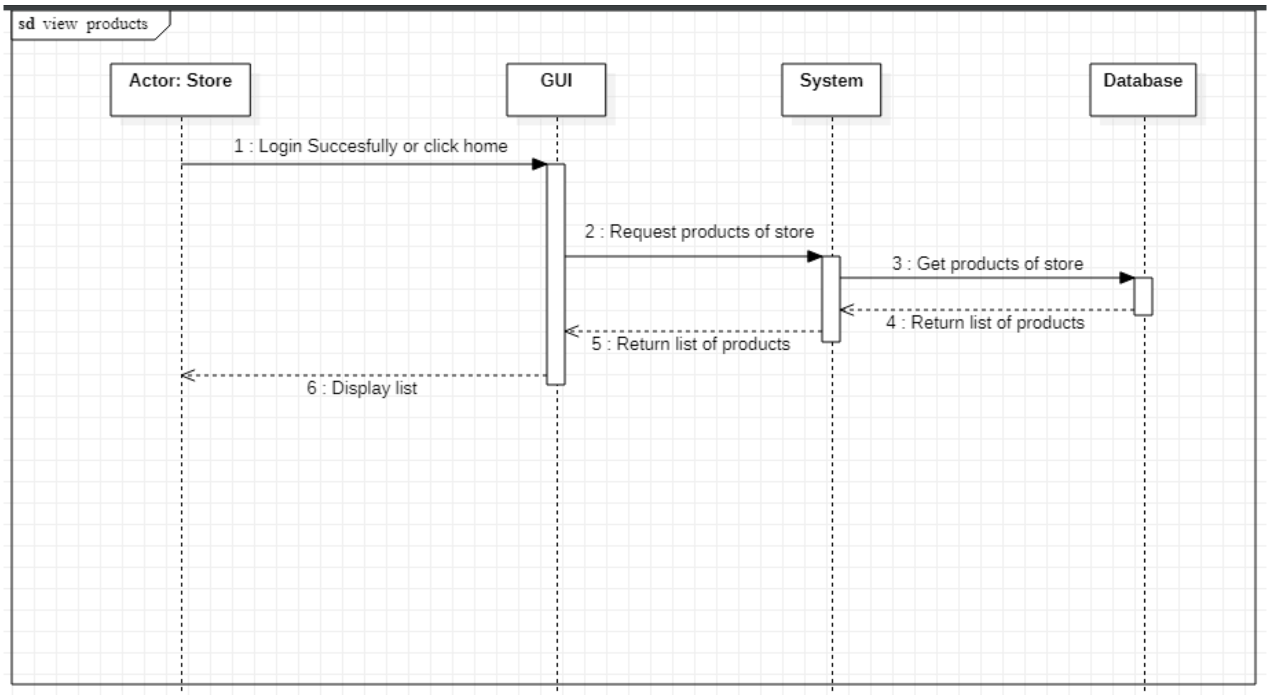
sd view products
Actor: Store
GUI
System
Database
1 : Login Succesfully or click home
2 : Request products of store
3 : Get products of store
4 : Return list of products
5 : Return list of products
6 : Display list

Figure 21: Home Page
This is the sequence diagram of store home page with product list.

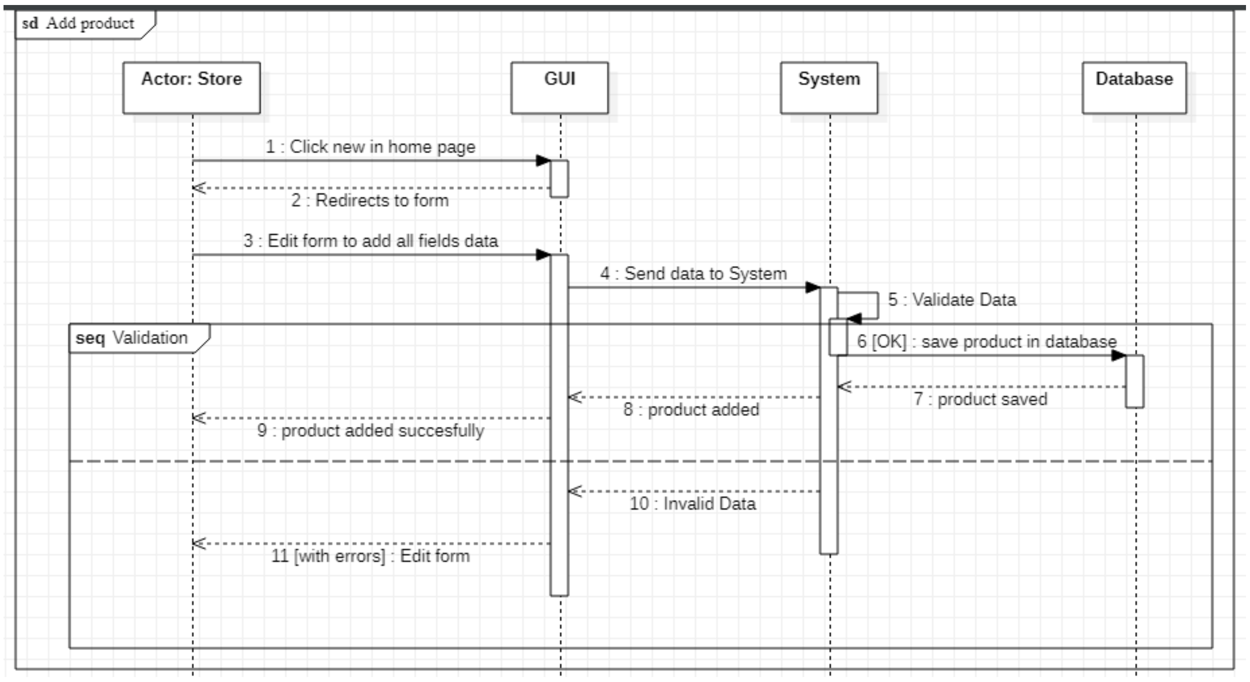

Figure 22: Add Product
This is the sequence diagram of store adding a product.

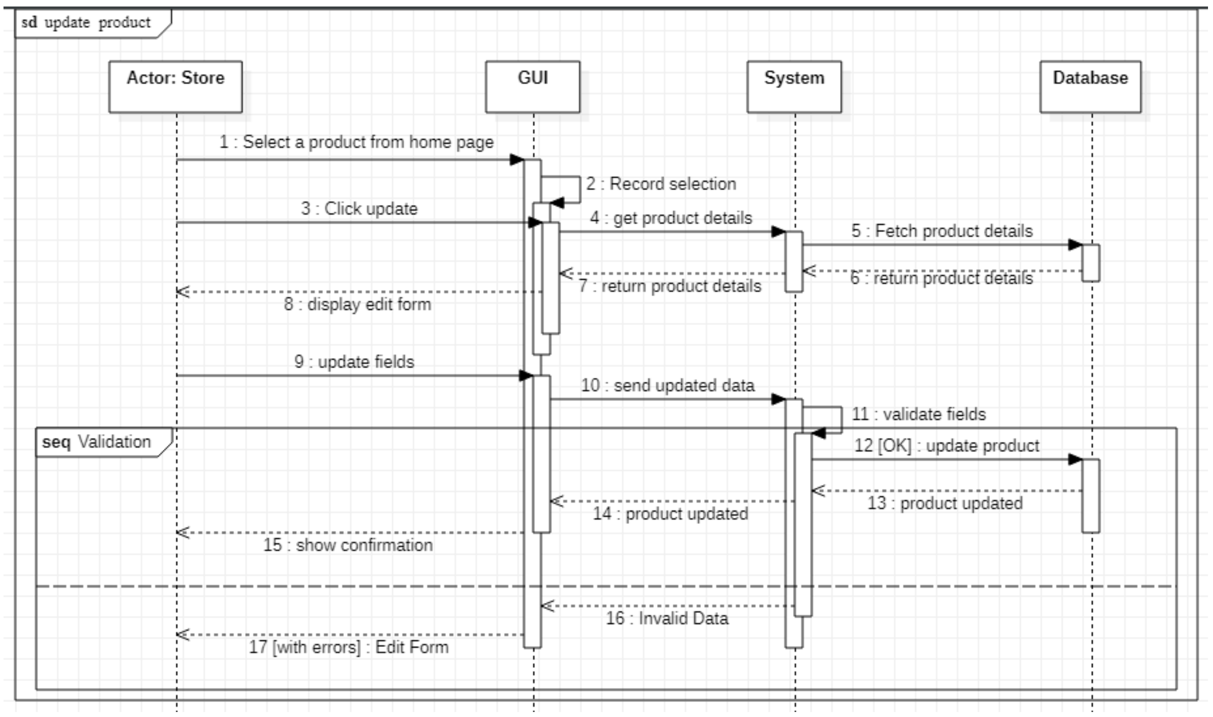

Figure 23: Update product
This is the sequence diagram of store updating a product.

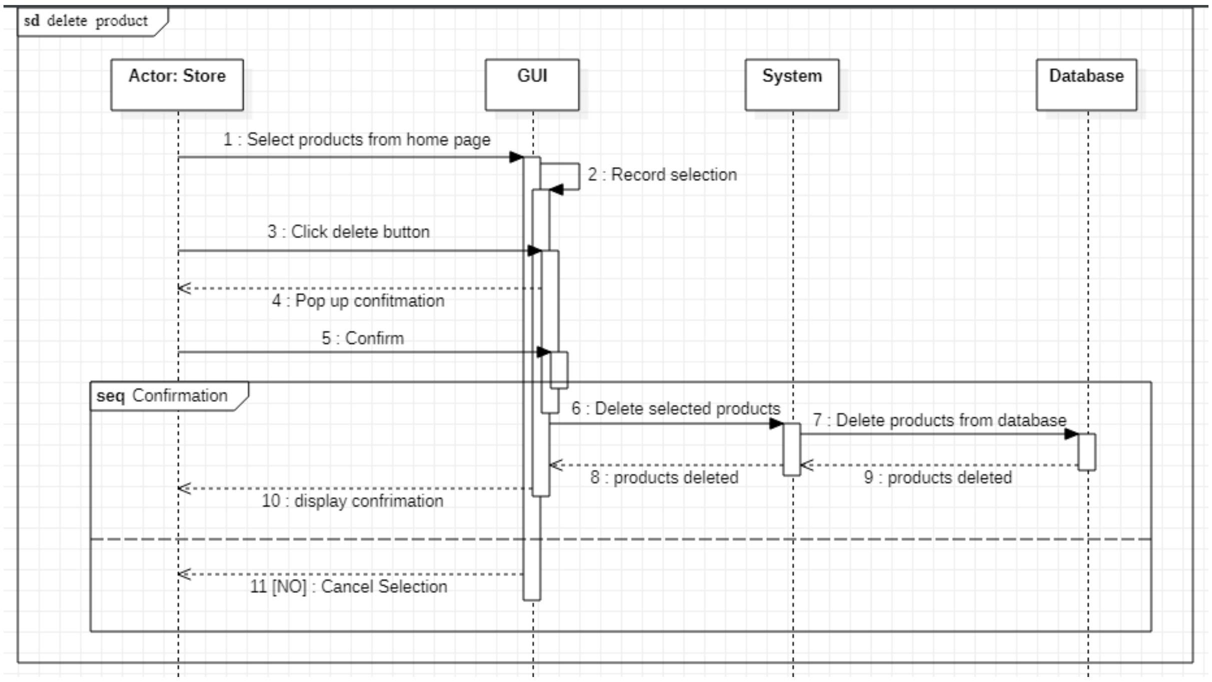

Figure 24: Delete Product
This is the sequence diagram of store deleting a product.

10. Policies and Tactics

10.1 Product to use

We will be using the latest version of python along with the Collaborative Filtering library. Moreover, we will use real-time cloud based databased named MongoDB. For running our Python code, we will use PyCharm.

10.2 Coding guidelines and conventions

To make our code more structural and legible, all standard coding rules will be followed, including properly commented code and the use of OOP ideas while coding. Furthermore, as previously indicated, we will use Ben Schneiderman's 8 golden standards of UI design when creating our UI.

10.3 Testing the software

Acceptance testing, unit testing, functional testing, performance testing, stress testing, and usability testing are some of the testing methodologies we studied in software engineering.

Furthermore, we will collect testing data from our university students and run it through our algorithm, after which we will compare the output of our algorithm to the testing data to ensure that our system is accurate enough.

10.4 Maintaining the software

After our fyp is finished we will release improved versions of our software with added feature to assist the users. Moreover, we will fix any possible future faults in our system.

10.5 Protocol

Http protocol will be used for communication between client and server. Interface will be implemented using react framework and running on browsers which are mentioned in software requirement. As we are using MongoDB, we will be storing data in JSON format hence data will flow over the network in JSON format.

10.6 Accessing the application

The web application would be hosted on cloud accessible with a URL link. Users having an account can access the application by going through the authentication procedure. Incase a user doesn’t have an account he can signup to make a new account. For vendors they must register their store on our website to add their products.

10.7 Choice of algorithm

We will use brute force approach along with different variants of collaborative filtering. We will choose the algorithm that gives us the best results.

10.8 Web Scrapping

We will scrap popular clothing brands website to retrieve their products by using a web scraping tool known as beautiful soup.

11. References

[1] Reutskaja, E., Lindner, A., Nagel, R. et al. “Choice overload reduces neural signatures of choice set value in dorsal striatum and anterior cingulate cortex.” *Nature Human Behavior*, vol 2, no. 925–935, Oct, 2018. [Online serial]. Available: https://doi.org/10.1038/s41562-018-0440-2 [Accessed Sept. 30, 2021]

[2] Cohn, M., 2021. *Scrum Methodology and Project Management*. [online] Mountain Goat Software. Available at: https://www.mountaingoatsoftware.com/agile/scrum [Accessed 15 December 2021]