

Group Project Log

Group Name:	Med+
--------------------	------

Group Members:	Ankit Varshney Malav Patel Karthik Rammoorthy Karthick Parameswaran
-----------------------	--

Deliverable:	Project Report
---------------------	----------------

Group Member Name	Work Done (%)
Ankit Varshney	25%
Malav Patel	25%
Karthik Rammoorthy	25%
Karthick Parameswaran	25%
Total:	100%

TECHNICAL REPORT

Med+

Members and Contributors

Ankit Varshney
B00784085
an225884@dal.ca

Malav Patel
B00790747
ml427209@dal.ca

Karthik Rammoorthy
B00790749
kr536074@dal.ca

**Karthick
Parameswaran**
B00791224
kr688977@dal.ca

Faculty of Computer Science
Dalhousie University

10, August, 2018

ABSTRACT

MedPlus is an electronic commerce website that is developed to serve general public; providing online marketplace for customers to purchase from a large collection of pharmaceutical drugs and delivering the purchased items at customers' door step. The application is a single-page, progressive web application developed with Angular 6 as frontend and Java Springboot REST service as backend. All functionalities are implemented as individual components, making the application modular and cohesive. The business logic of the application takes place at the backend. On the user action, a REST call is transmitted to the backend API to trigger the corresponding business logic. Database interaction is restricted to backend and the RESTful services are hosted in Heroku. Proper validations are implemented in user forms such as user registration, login and update profile components. User would be able to add multiple items to cart and place the order. Also, there is provision to view order history and edit user profile information such as contact and mailing addresses. Users would also be able to rate the purchased items and provide feedback for the same.

The objective is to develop a full stack web project with distinguished frontend and backend components that interact through HTTP methods in JSON data format.

KEYWORDS

Angular components, Async Validators, DAO classes, Heroku server, HTTP endpoints, JSON data format, MySQL database, RESTful service, Springboot API, Typescript Services.

CONTENTS

1. INTRODUCTION	6
2. BACKGROUND	6
2.1 Competitive Landscape	6
2.2 Problem and Approach	7
3. APPLICATION DETAILS	7
3.1 Target User Insights	7
3.2 User-Centered Design Approach	7
3.2.1 Information Architecture	8
3.2.2 Design and Layout	8
4. TECHNICAL SPECIFICATION	15
5. APPLICATION WORKFLOW	16
5.1 Interaction Design	16
5.2 Process and Service Workflow	24
6. CONCLUSION	36
7. REFERENCES	37

TABLE OF FIGURES

Figure 1 - canadadrugs.com - Website down.....	6
Figure 2 - canadadrugs.com - Website Online.....	7
Figure 3 - Sitemap.....	8
Figure 4 - Home Page(1).....	8
Figure 5 - Home Page(2).....	9
Figure 6 - Login Page	9
Figure 7 - Registration Page	10
Figure 8 - Search Results Page	10
Figure 9 - Product Details Page	11
Figure 10 - Cart Page	11

Figure 11 - Checkout Page.....	12
Figure 12 - Add Review Page.....	12
Figure 13 - Review section on Product Details Page.....	13
Figure 14 - Order History	13
Figure 15 - Update User Profile.....	14
Figure 16- Interaction design: Login flowchart.....	16
Figure 17 – Interaction design: User Registration	17
Figure 18- Interaction design: Search Functionality.....	18
Figure 19 - Interaction design: Update User Profile.....	19
Figure 20 - Interaction design: Update User Profile.....	20
Figure 21 – Interaction design: Checkout.....	21
Figure 22- Interaction design: Add to Cart.....	22
Figure 23 - Interaction design: Feedback.....	23
Figure 24 – Process Flow: Login	24
Figure 25 – Process Flow: Logout.....	25
Figure 26 – Process Flow: Product	25
Figure 27 – Process Flow: Add to Cart.....	26
Figure 28 – Process Flow: Checkout	27
Figure 29 – Process Flow: Delete Cart	28
Figure 30 – Process Flow: Place Order.....	29
Figure 31 – Process Flow: Get User Details	30
Figure 32 – Process Flow: User Registration	31
Figure 33 – Process Flow: Search Functionality	32
Figure 34 – Process Flow: Update User Profile.....	33
Figure 35 – Process Flow: Order History	34
Figure 36 – Process Flow: Feedback Functionality	35
Figure 37 – Process Flow: Retrieve Feedback based on Product ID	35

1. INTRODUCTION

Med+ is an on-demand online medicines oriented service that provides business to consumer sales service and operates on the website. Our vision is that people everywhere have access to the essential medicines and health products who need to strive for the better healthy life. With the advancement of technology and networking, Med+ has the wide range of medicines online, sourced from the trusted network of pharmacies and medical stores. We try to bring people, ease of access to the basic medicines they need.

2. BACKGROUND

2.1 Competitive Landscape

We are providing medicines in the Canadian origin. The only online pharmacy that provided medicine at doorstep were www.canadadrugs.com. The site is taken down as they were selling prescription medicines. Med+ is only focussed on selling non-prescription drugs which would not require permission from the government officials. We are the only online pharmacy that will help people find medicines easily at their doorstep. Also, our search filter is more powerful compared to canada drugs as we tend to search medicines based on symptoms and names so a user can find out equivalent medicines if they don't remember names of the medicines as well.

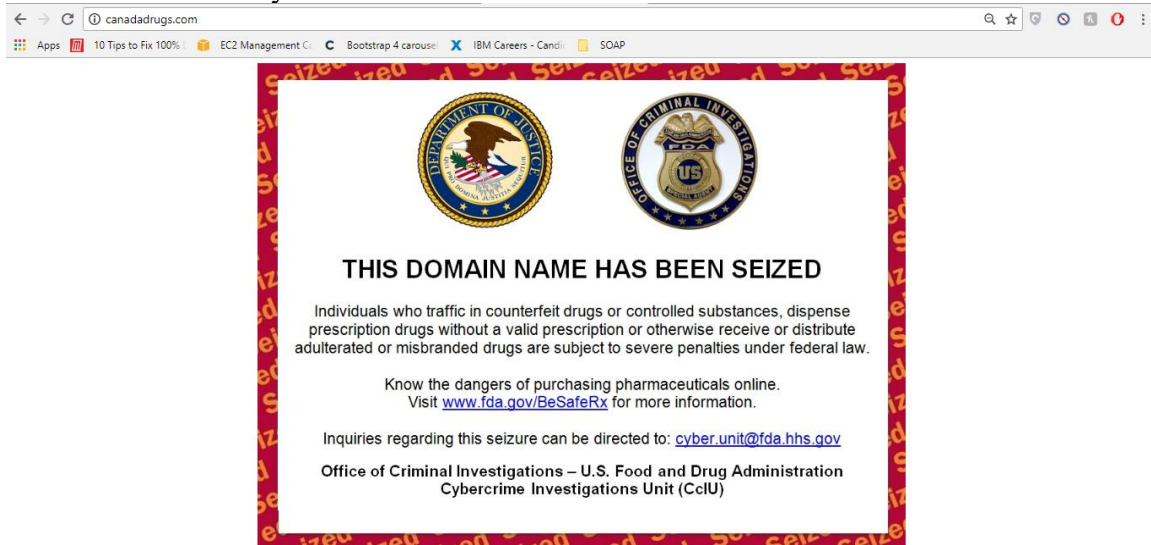


Figure 1 - canadadrugs.com - Website down

2.2 Problem and Approach

Before starting Med+, we analysed canadadrugs.com website and it was all filled with content. Below is the screenshot we had before they closed, as one can see it is filled with contents all over the place and the top part of the website has nothing related to medicines. We believe that the website updates are important but it should not be all over the place and website should serve the meaning of its existence. Med+ has a great design in terms of product placement and user can easily find out products on the homepage.



Figure 2 - canadadrugs.com - Website Online

3. APPLICATION DETAILS

3.1 Target User Insights

Anyone who is looking to buy medicines are our target users. MedPlus aims to make medicines available to everyone from anywhere. So, anyone who couldn't find medicines around them or live in remote place can use our website. what makes our website different from others is that there is no other website in canada providing platform to buy medicines online. As our website provides different medicines for same symptoms, user get different choices to select from and can compare them and buy the one best suitable for them. User just need to have a basic knowledge of medicines they are looking for to use our website.

3.2 User-Centered Design Approach

Keeping our users in mind, we tried to keep our website as simple as possible.

3.2.1 Information Architecture

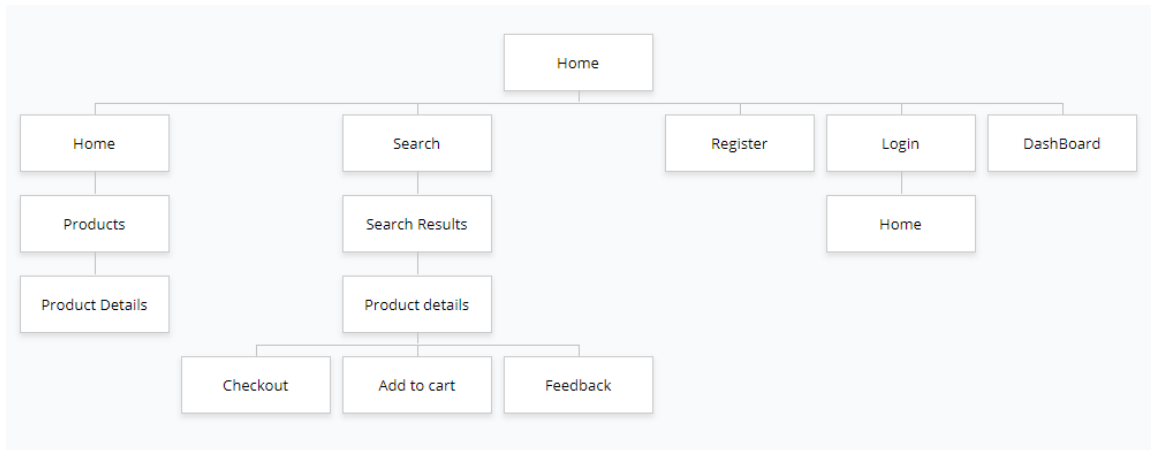


Figure 3 - Sitemap

As e commerce website is made up of many pages: Cart, Product details, Checkout, etc. This makes the website complex. We designed the website in simplest way possible. We tried to make things simple and quick for user by applying 3 click rule. So there are only 2 workflows after the home page, its either by search or by clicking on product.

3.2.2 Design and Layout

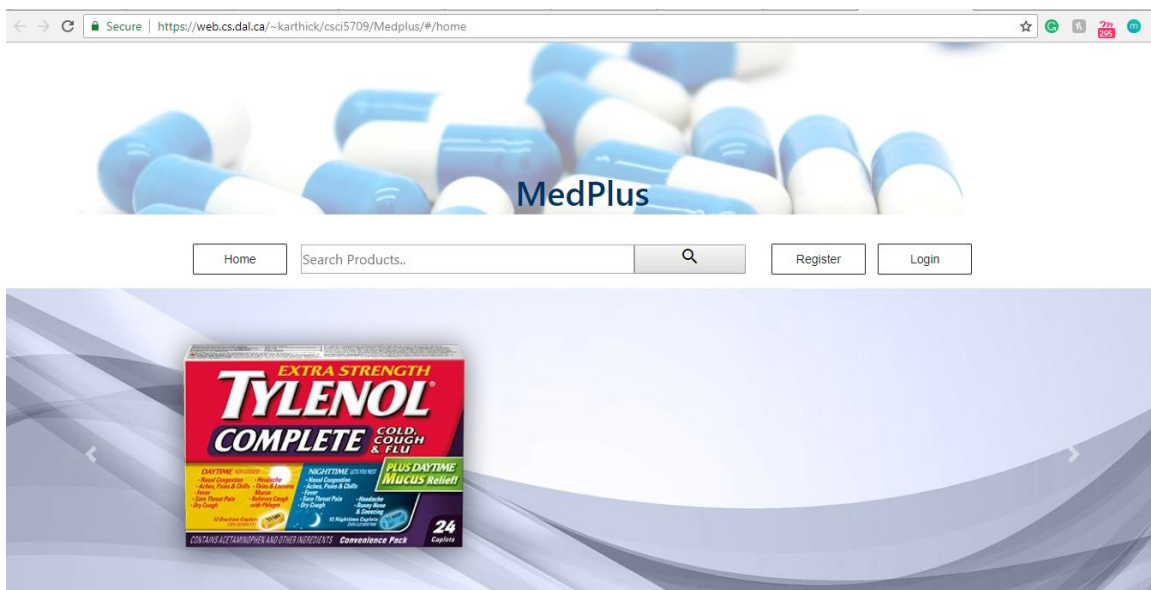


Figure 4 - Home Page(1)

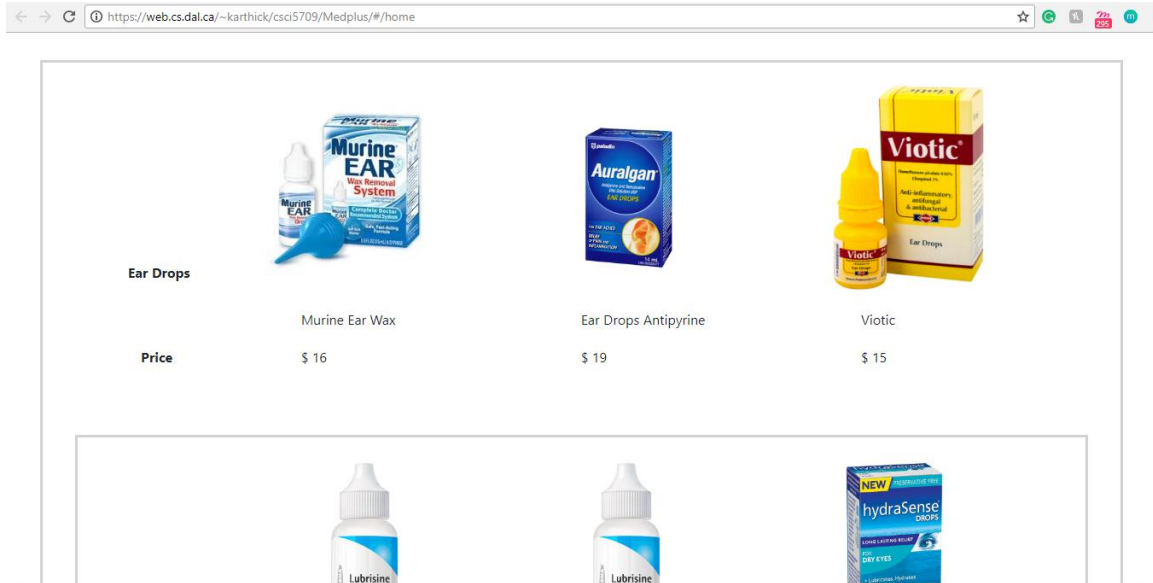


Figure 5 - Home Page(2)

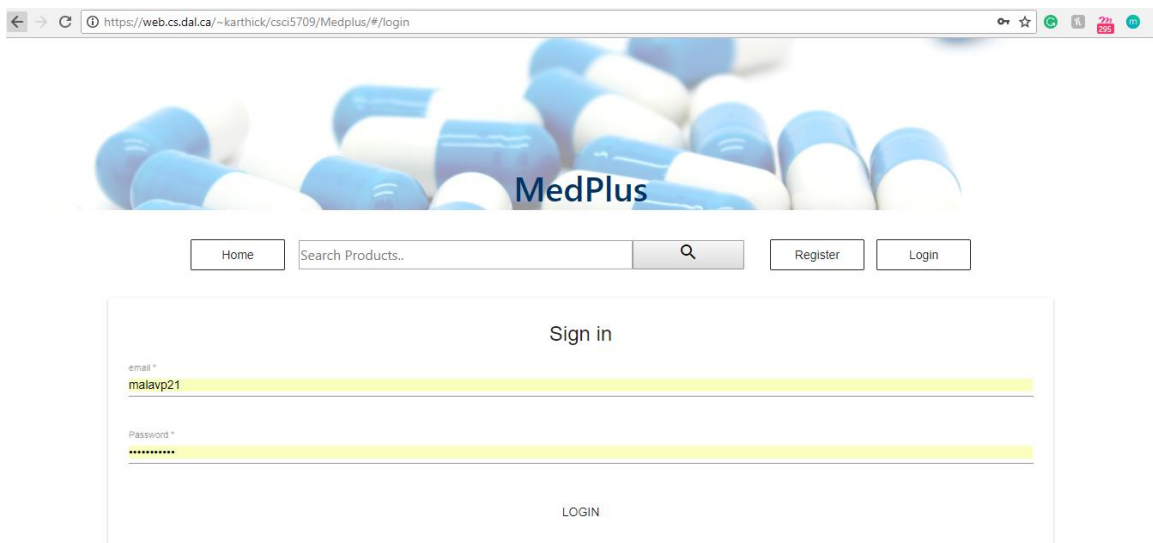



Figure 6 - Login Page

← → ↻ <https://web.cs.dal.ca/~karthick/csci5709/Medplus/#/register> ☆ 📄 📄 📄



MedPlus

Home Murine Ear Wax 🔍 Register Login

Sign up

Enter your first name *

Please enter valid first name

Enter your last name *


Enter your email id *

Enter your password *

Password must contain a minimum of 8 letters

Figure 7 - Registration Page

← → ↻ <https://web.cs.dal.ca/~karthick/csci5709/Medplus/#/searchresultsearchterm=Murine%20Ear%20Wax> ☆ 📄 📄 📄



MedPlus

Home Murine Ear Wax 🔍 Register Login



Product : Murine Ear Wax
Category : ear

Price Per Quantity : \$ 16 Available Quantity : 25

Product Info ⓘ

Figure 8 - Search Results Page

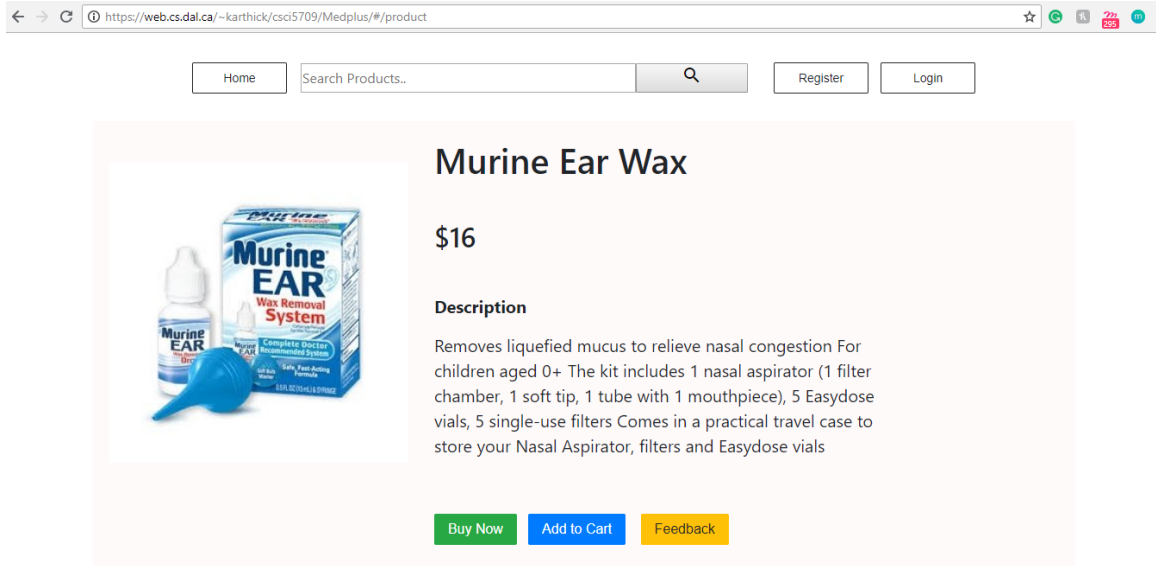


Figure 9 - Product Details Page

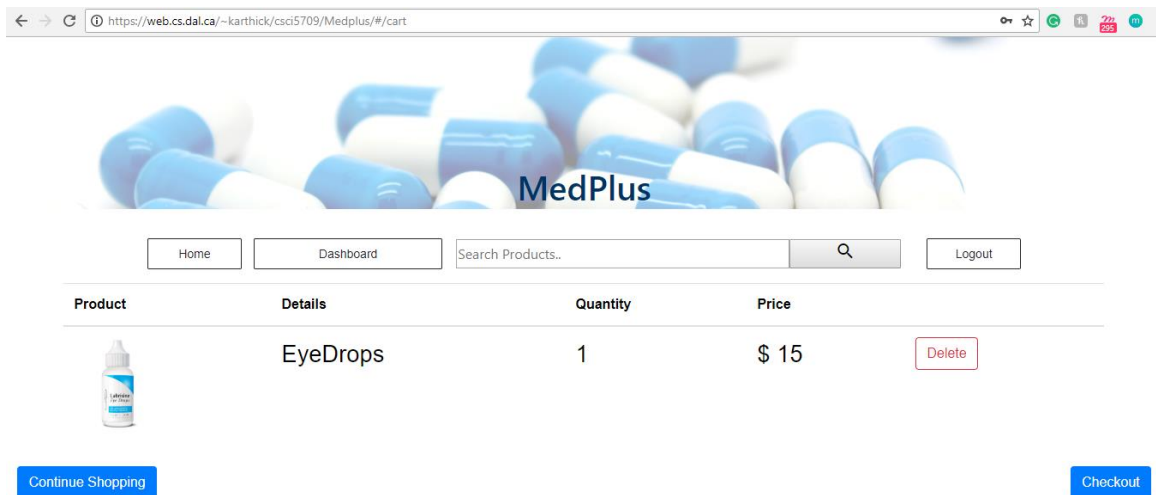



Figure 10 - Cart Page

Home Dashboard Search Products.. Logout

Product	Details	Quantity	Price	
	EyeDrops	1	\$ 15	Delete

Address

501 - Brunswick Street
Halifax
Nova Scotia
Canada
B3J 3J7

Payment Information

Enter 16 digit card number *

expiry date * mm/dd/yyyy security number *

CardHolder Name *

Place Order

Figure 11 - Checkout Page

Home Dashboard Search Products.. Logout

User Feedback

Enter your comments *

User Rating :
☆☆☆☆☆☆

Rating : 0

Submit Feedback

Figure 12 - Add Review Page

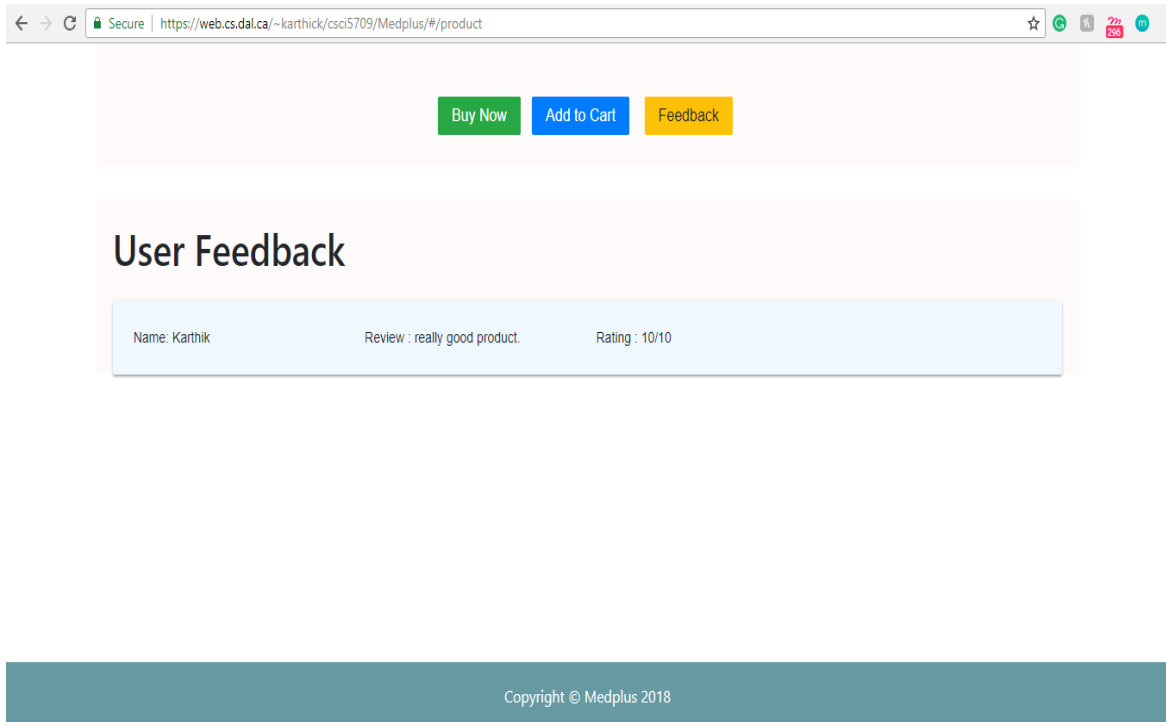


Figure 13 - Review section on Product Details Page



Figure 14 - Order History

Update Details

Enter your first name *

Karthik

Enter your last name *

Rammoorthy

Enter your email id *

kr536074@dal.ca

Enter your password *

Password must contain a minimum of 8 letters

Reenter password *

Contact Number *

9027894476

Suite Number *

503

Street Name *

Brunswick Street

City *

Halifax

State *

Nova Scotia

Country *

Canada

Zip Code *

B3J 3J7

Format ### ###

Update

Figure 15 - Update User Profile

4. TECHNICAL SPECIFICATION

Component	Software Specification
Front End Framework	Angular 6 with HTML and CSS
Bank End Framework	Spring Boot REST API
Version Control	GitHub
Collaboration Tool	Slack, Trello
Database	MySQL
Host Server	Heroku
Front End Libraries	Angular Material, FlexLayout API.

5. APPLICATION WORKFLOW

5.1 Interaction Design

Login and logout

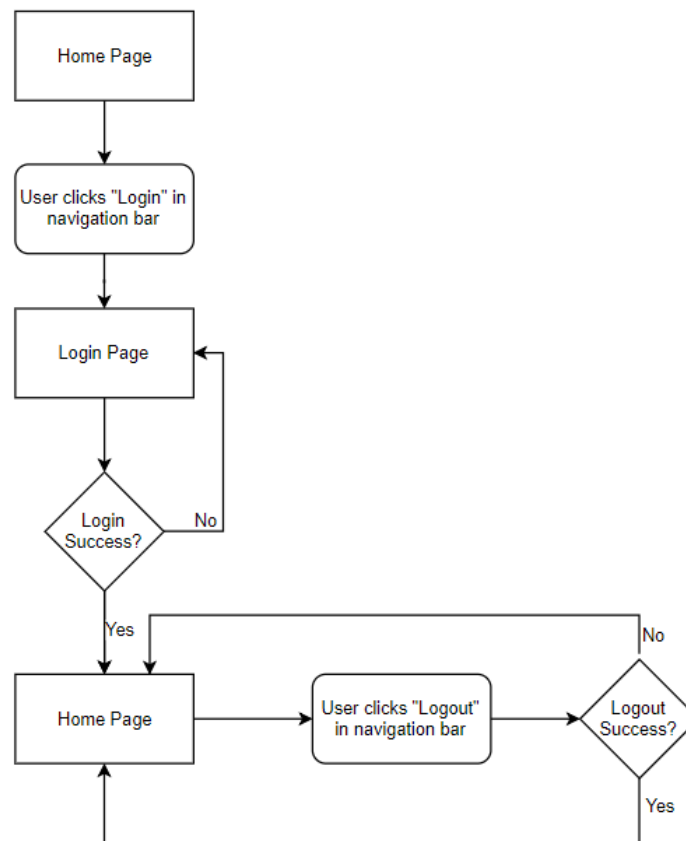


Figure 16- Interaction design: Login flowchart

- User login process is handled in the login component of the angular application.
- Existing users are required to login before placing any orders in the website.
- A click on the login button in the navigation bar will guide us to login page.
- After successful login, user can logout by a click on the logout button in the navigation bar.

User Registration

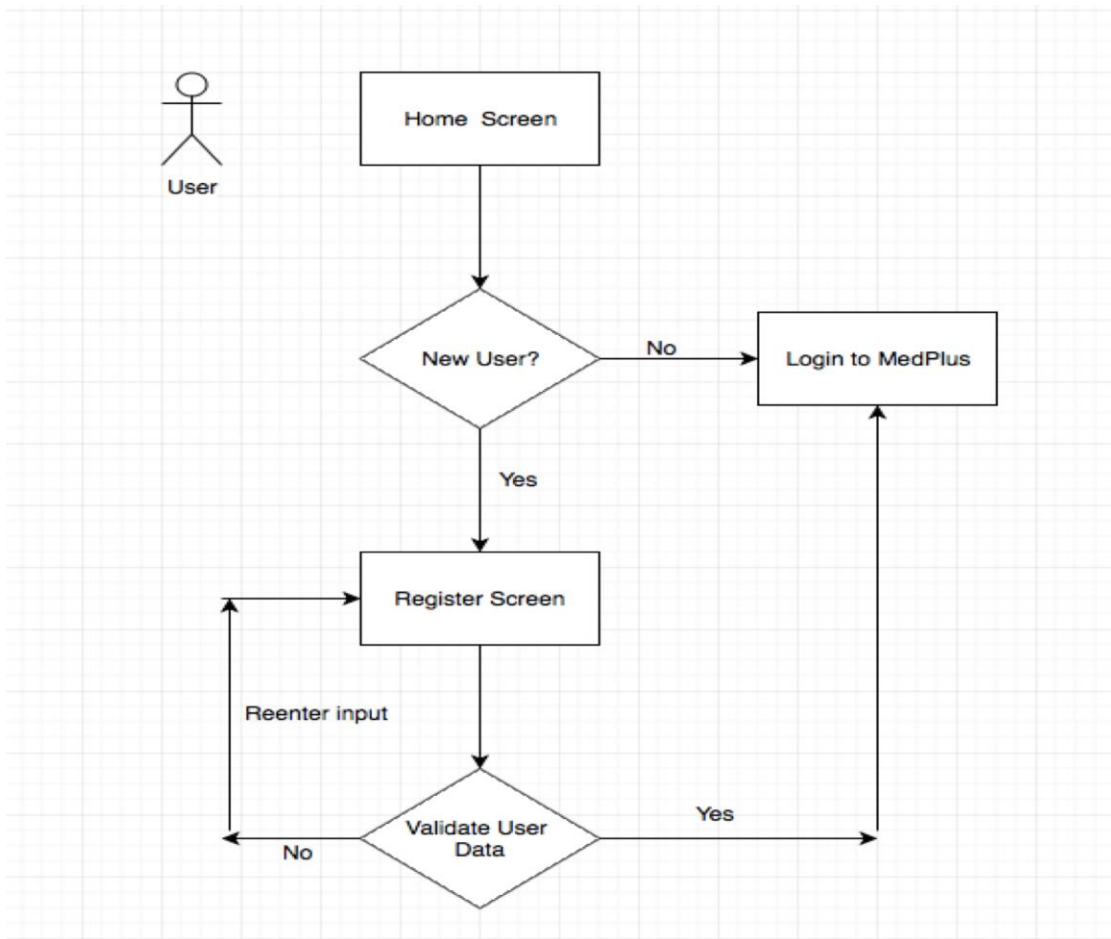


Figure 17 – Interaction design: User Registration

- User registration process is handled in the Register Component of the Angular application.
- New users are required to create an account in order to place an order in MedPlus.
- A click on the Register tab in the Navigation pane loads the Register Component.
- User details such as email address, phone, mailing address are captured along with password.
- Asynchronous validation, whereby user data on each input element is subjected to validation.
- After successful validation Signup button is enabled. Clicking on the Signup button sends a POST request on the CreateUser Endpoint of the Backend API.

Search Functionality

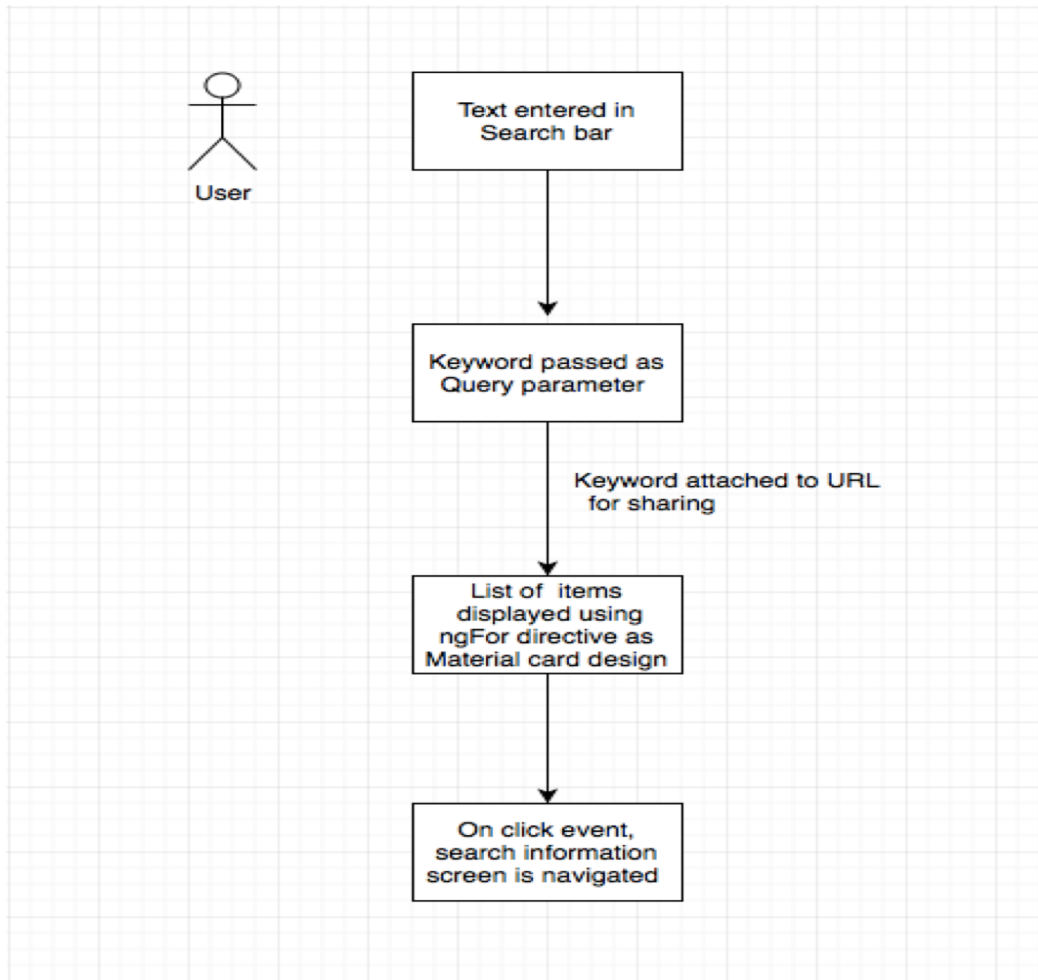


Figure 18- Interaction design: Search Functionality

- The required product to be searched is entered in the search bar component of the home screen.
- On click event, a GET request is sent to the backend RESTful API along with the search keyword.
- The response which is list of products are displayed in Search Result component.
- The product information screen is displayed for the selected product from the list.

Update User Profile

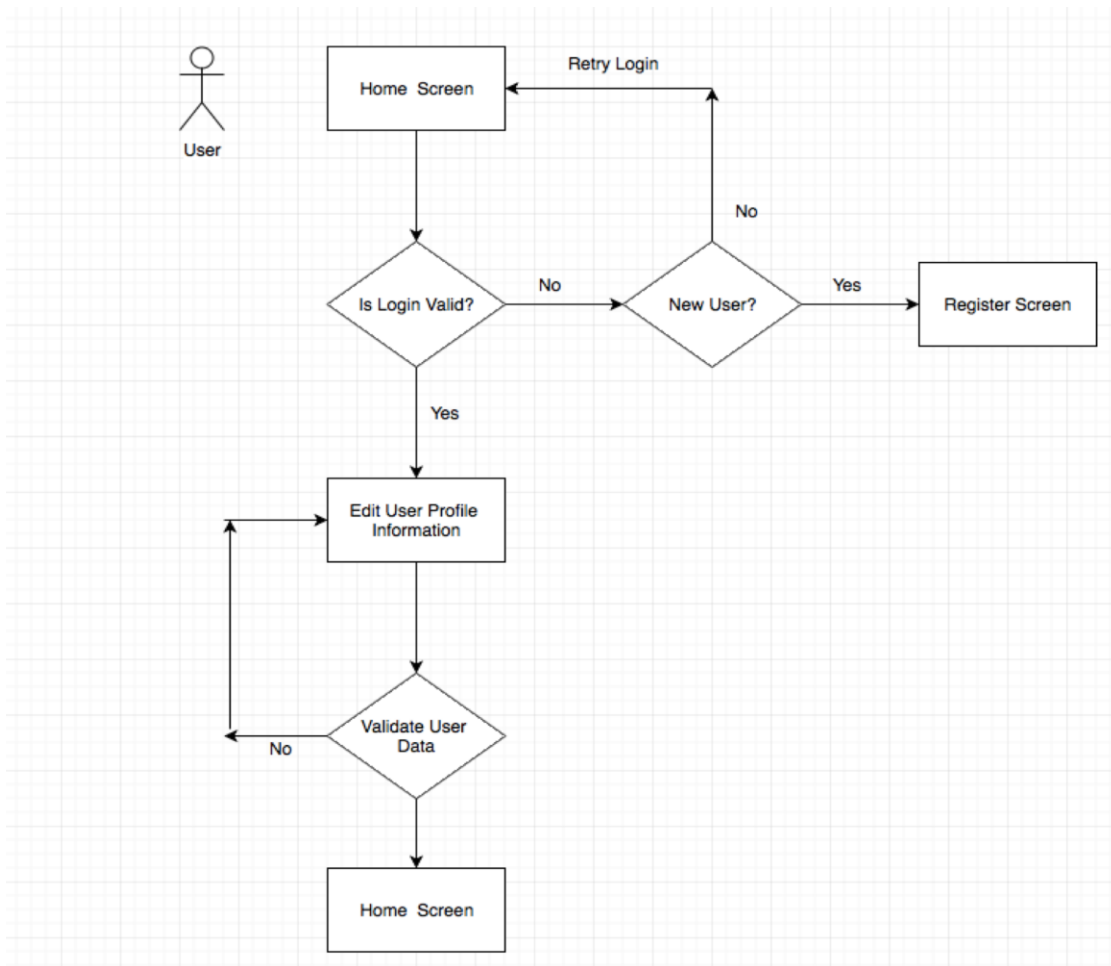


Figure 19 - Interaction design: Update User Profile

- The User profile information that was given during registration could be updated anytime.
- On successful login, the user could open Update User screen from Navigation menu.
- All the validations that were applied during registration would also be applied in Update screen as well.
- The updated information is reflected consistently throughout the system.
- User object stored in local storage is updated with the latest information. This is helpful for maintaining login information.

View Order History

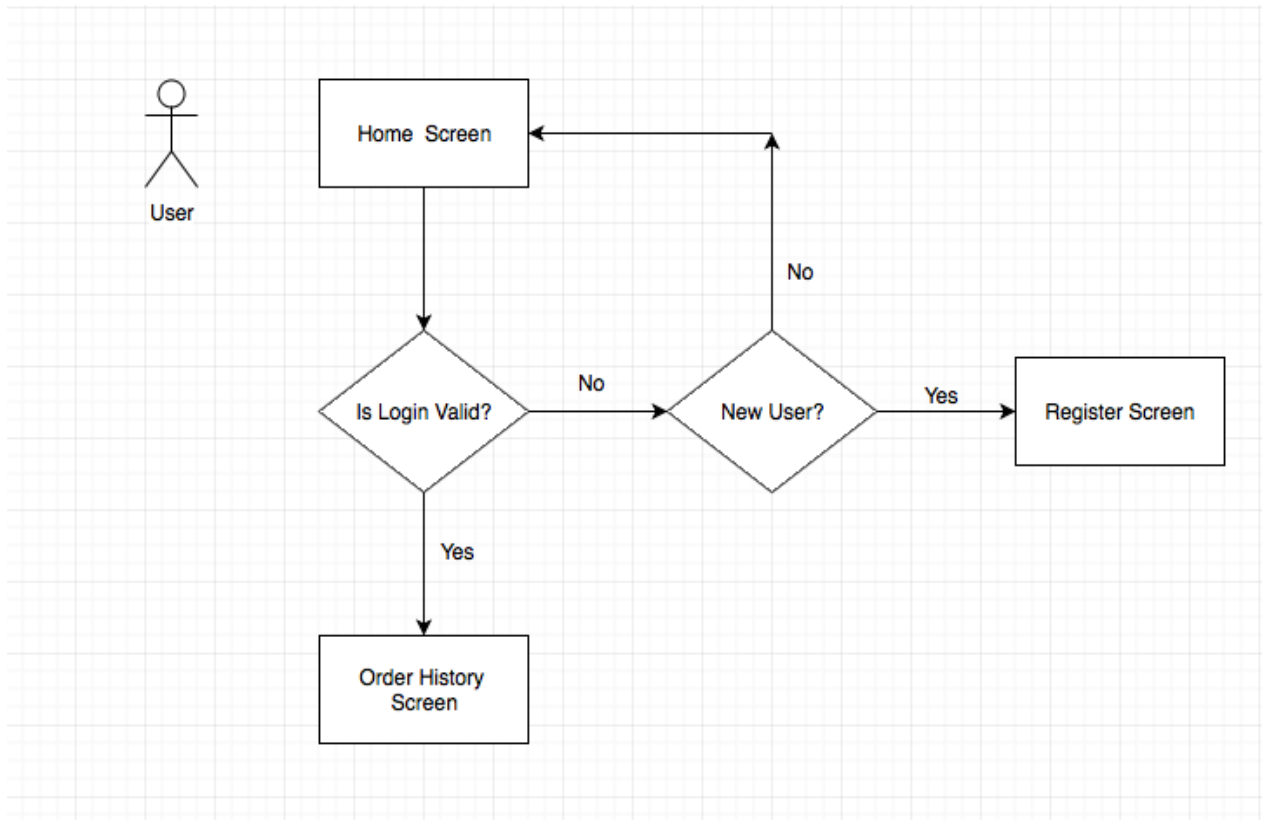


Figure 20 - Interaction design: Update User Profile

- Users have the options to place multiple orders at the same time.
- The list of orders for the specific user are displayed in the Order History screen.
- Order History screen could be navigated from Navigation menu after successful login.

Checkout

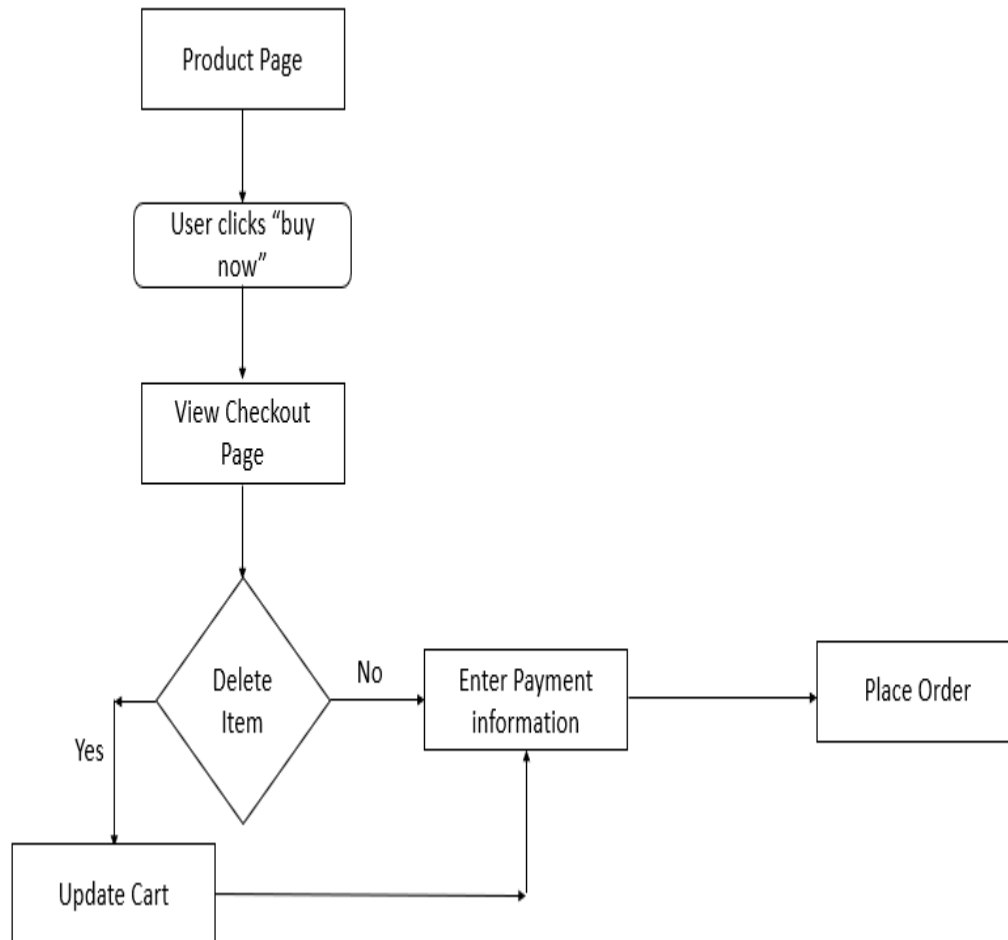


Figure 21 – Interaction design: Checkout

- The user can navigate to checkout page to update or delete items in the cart.
- Once user is on the checkout page, they'll be able to see cart items and can also delete them.
- Once everything is perfect, user can enter payment information and place order.

Add to Cart

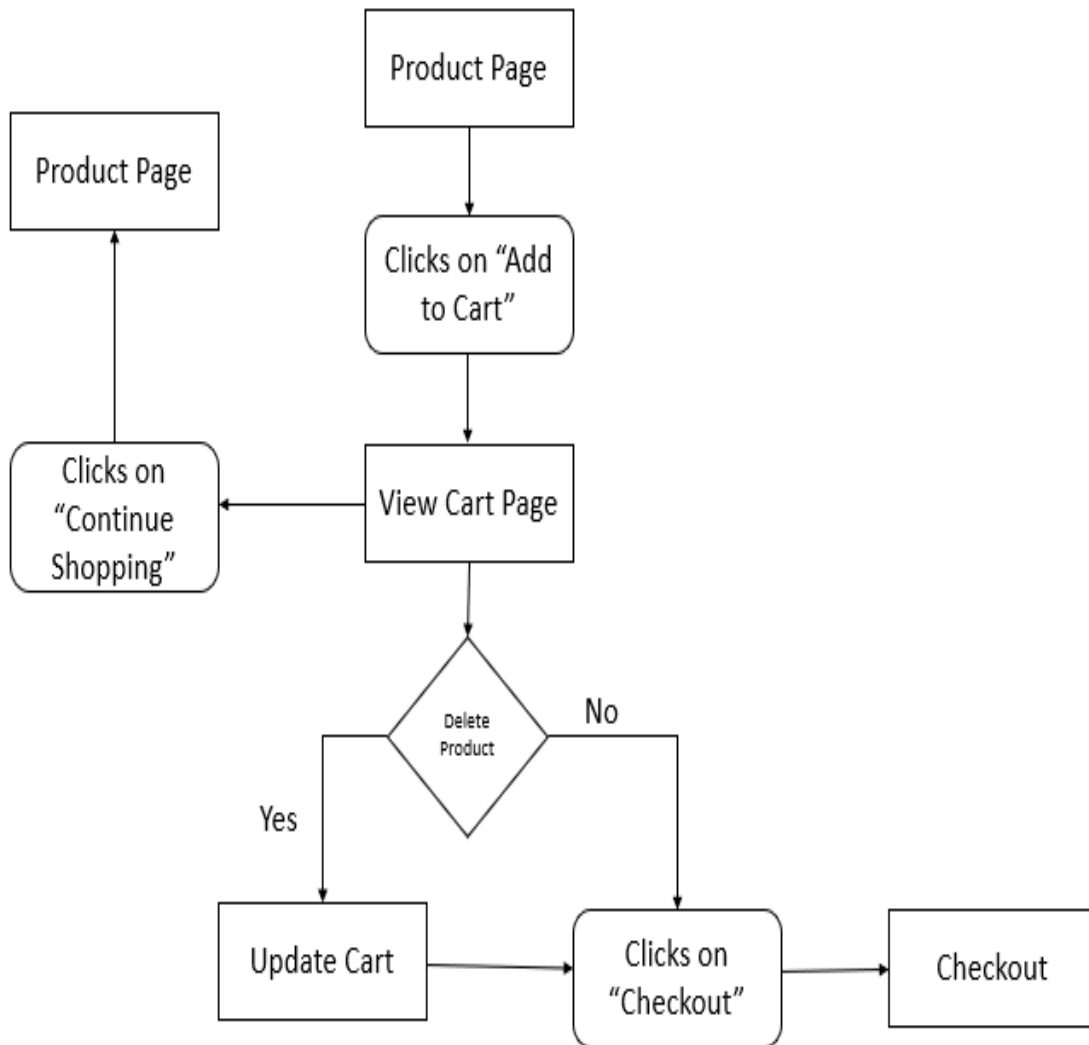


Figure 22- Interaction design: Add to Cart

- User can add items to the cart that they wish to purchase
- Once user is on Cart Page, they can see cart items. They can continue shopping for more products or can go to the Checkout Page.
- User can remove items from the Cart using Delete button.

Feedback

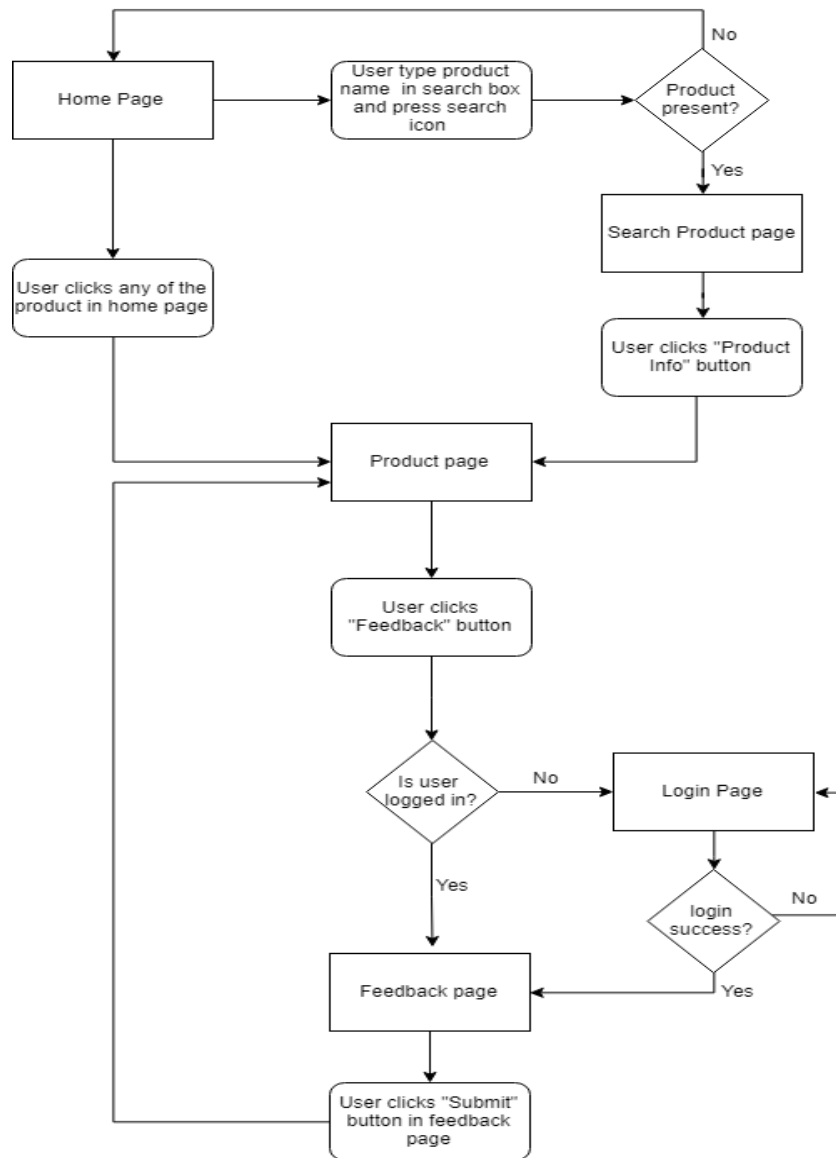


Figure 23 - Interaction design: Feedback

- User can add feedback to the particular product for the user experience.
- Once user is logged in, he/she can go to the respective product and add review which will get reflected immediately in the product page as well.
- If user did not login, he/she will be redirected to the login page to identify which user is giving the feedback.

5.2 Process and Service Workflow

Login page

1) Login endpoint

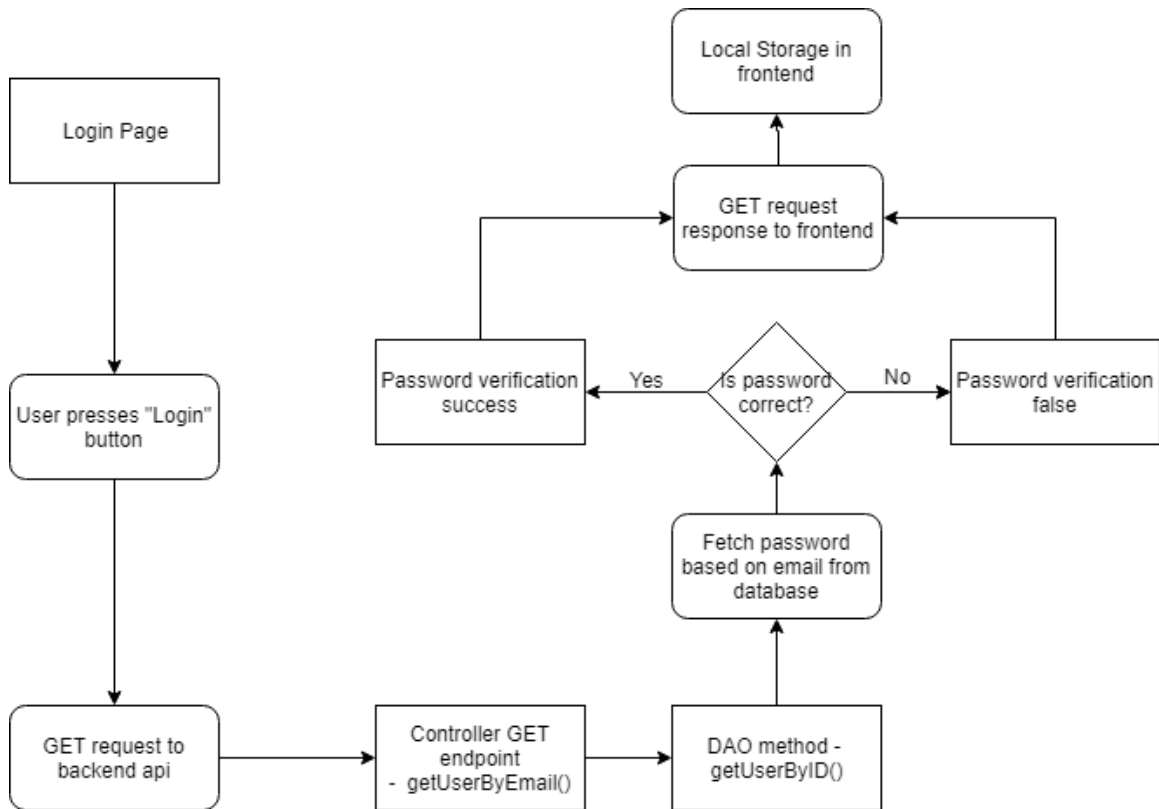


Figure 24 – Process Flow: Login

- User controller in the backend handles the GET request based on the email and password as form data in the request.
- User DAO handles the query to fetch the password based on the email and verification happens in the backend only.
- Based on verification, GET request will receive the responses.
- For the successful verification, localStorage is set.

2) Logout

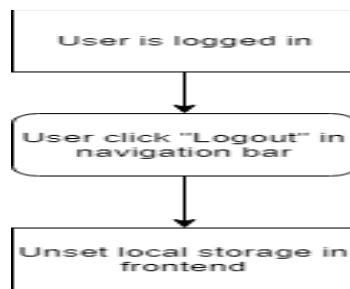


Figure 25 – Process Flow: Logout

On logout, local storage is unset and there would not be any interaction with the backend for this scenario.

Product Page

1) Get product endpoint

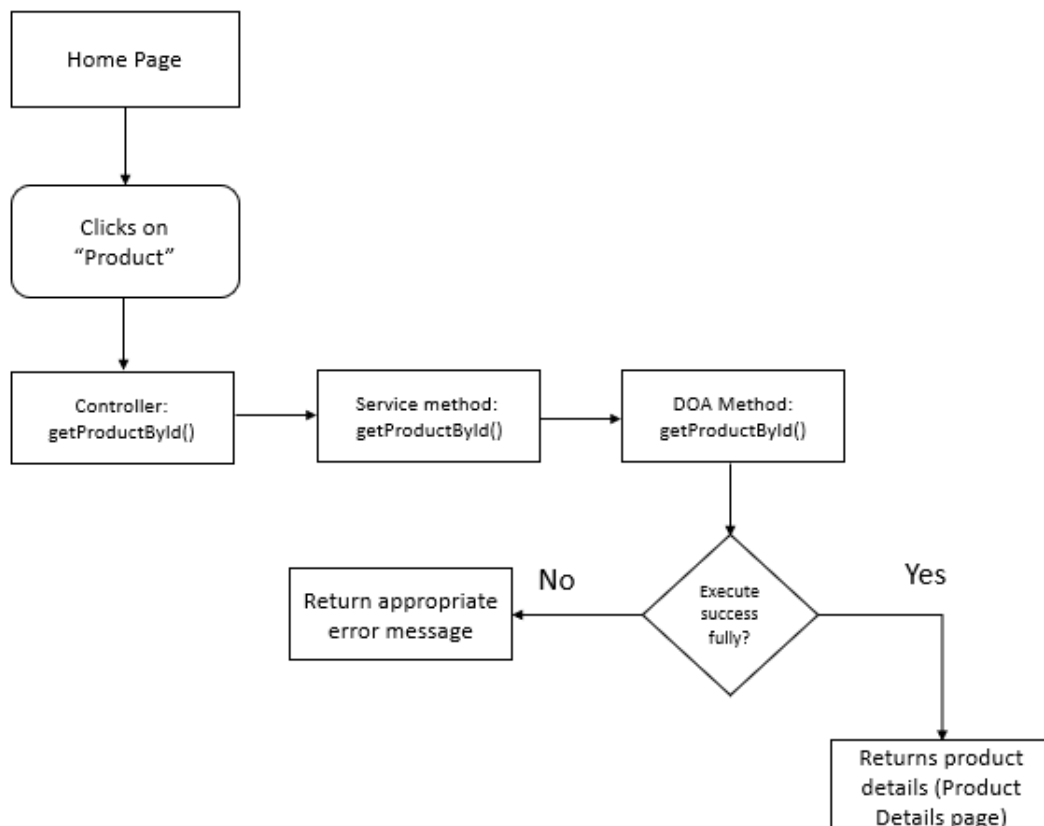


Figure 26 – Process Flow: Product

- Once the user clicks on any product, a GET request is generated.
- The product details is fetched based on it's productId.
- Product controller has getProductById() which takes care of the GET request, it sends the request to DAO's getProductById() which connects with database and fetches Product Details.

2) Add to cart endpoint

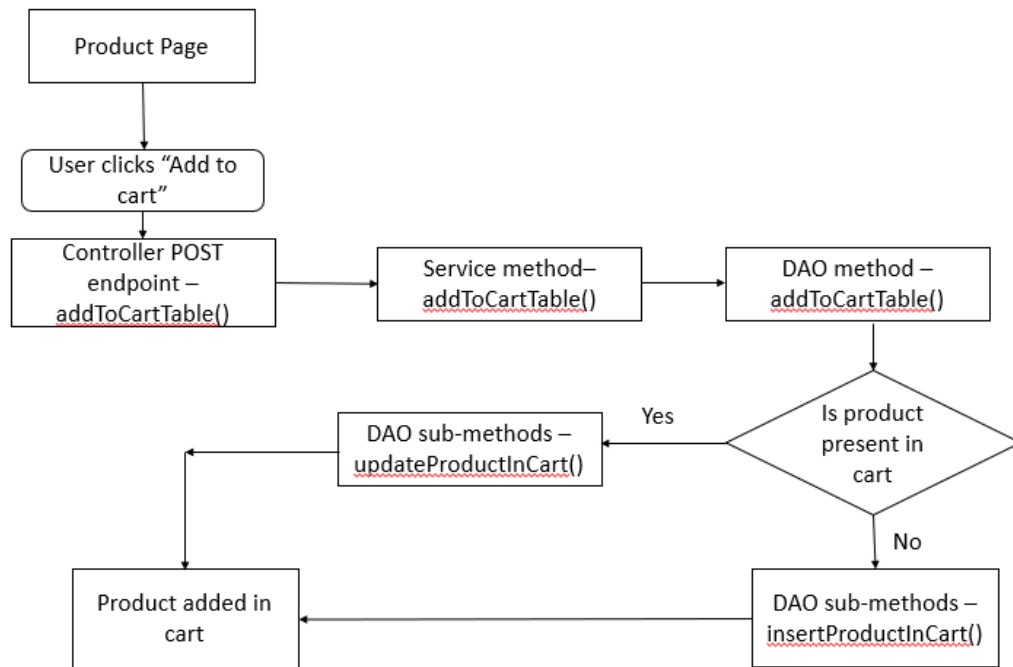


Figure 27 – Process Flow: Add to Cart

- POST request is generated once user clicks on Add to Cart Button.
- Product controller has addToCartTable() which handles POST request, it sends the request to DAO's addToCartTable().
- There are another sub-methods also insertProductInCart() which inserts the new product into the cart and updateProductInCart() which updates the quantity of product that is already present in the cart.

Checkout page

1) GET cart product endpoint

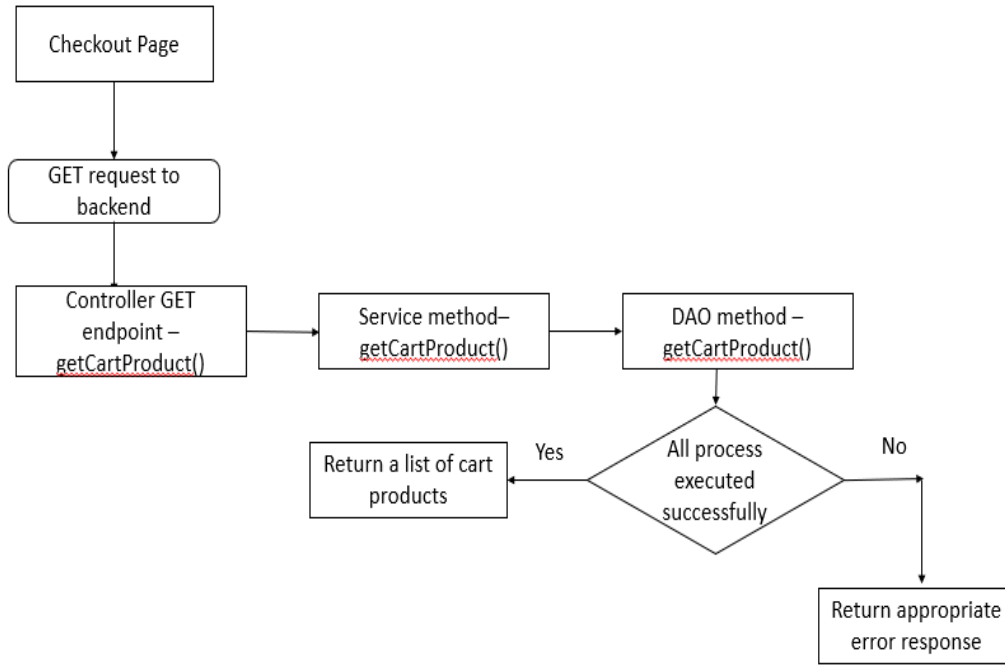


Figure 28 – Process Flow: Checkout

- Once the user navigates to checkout page, a GET request is generated to the backend services.
- The GET request will fetch all the items present in the cart based on user id.
- Product controller has `getCartProduct()` which takes care of the GET request, it sends the request to DAO's `getCartProduct()` which connects with database and fetches cart items.

2) Delete cart product endpoint

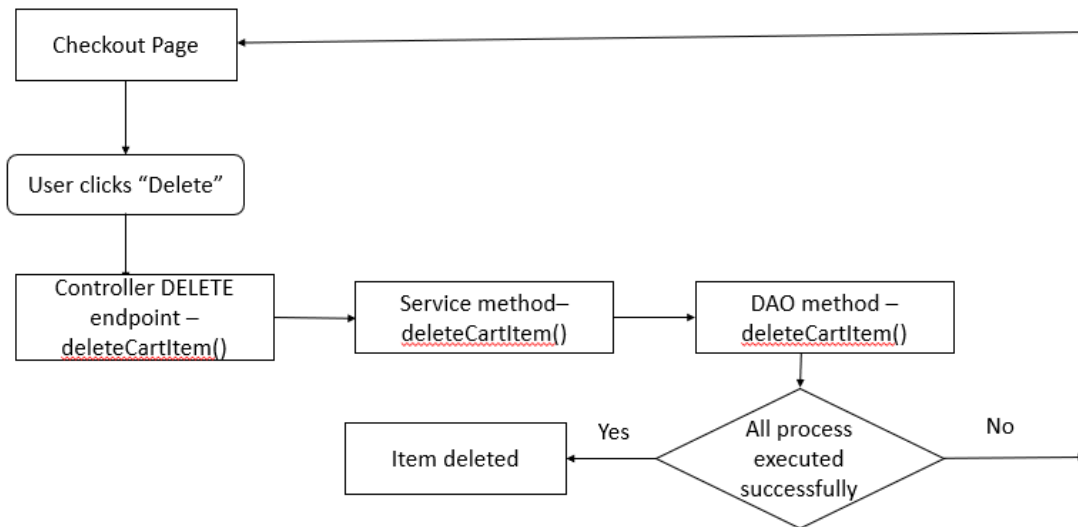


Figure 29 – Process Flow: Delete Cart

- User can delete items present in the cart.
- A DELETE request is generated which will delete the product from the cart for that user
- Product controller has deleteCartItem() which handles DELETE request, it sends the request to DAO's deleteCartItem() which connects with database and deletes items from the cart.

3) Place order

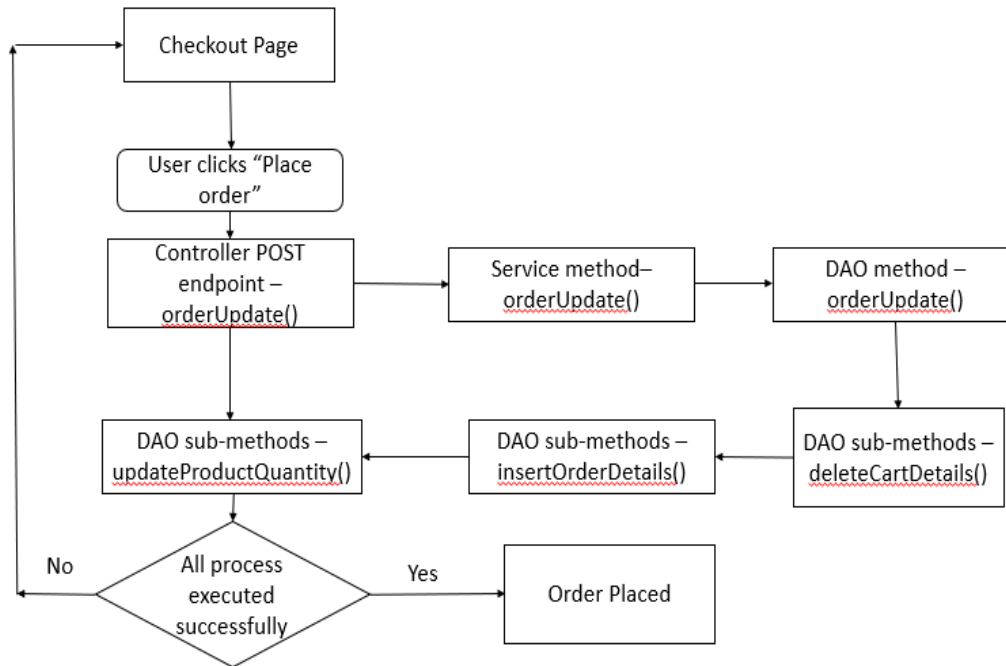


Figure 30 – Process Flow: Place Order

- Once the user is satisfied with the cart item, they can enter payment information and place order.
- After placing order, a POST request will be generated to the backend server
- Product controller has `orderupdate()` which handles POST request, it sends the request to DAO's `orderupdate()`.
- DAO has other sub-methods as well such as `deleteCartDetails()` which delete cart items for the user, `insertOrderDetails()` will insert the order details in user order history and `updateproductquantity()` will decrease the quantity of product from the database.
- Once all the processes are executed successfully then the order is placed successfully.

4) GET user details for address endpoint

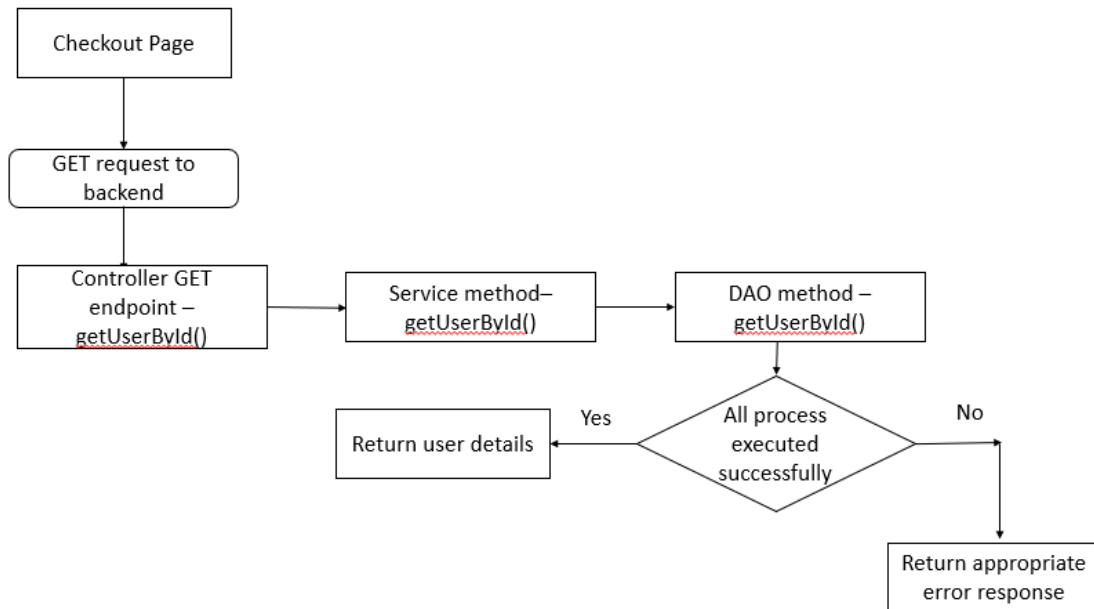


Figure 31 – Process Flow: Get User Details

- Before ordering the user would want to know which address is present in the Med+ database.
- The GET request will fetch the user address from the database.
- Usercontroller has `getUserById()` which takes care of the GET request, it sends the request to DAO's `getUserById()` which connects with database and fetches user details.

User Registration

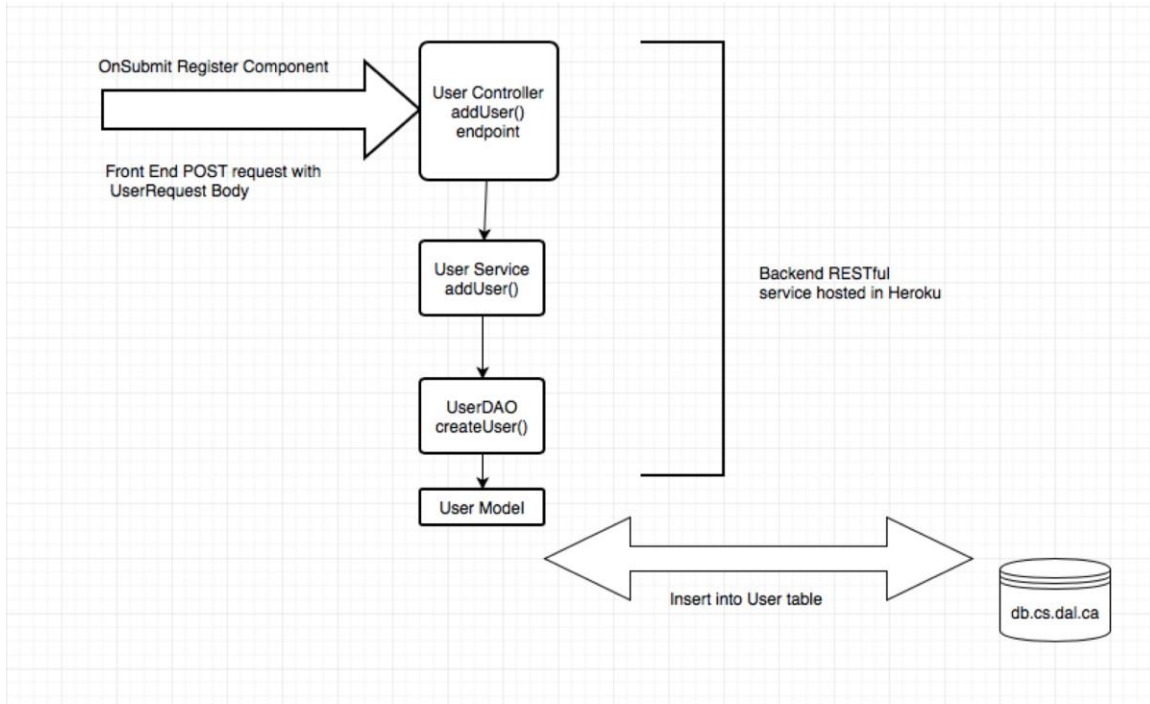


Figure 32 – Process Flow: User Registration

- Once the user data is submitted in Register form, a POST request is sent to backend API with user data.
- The incoming request is handled in the backend by User controller that sends the user data to UserDAO object.
- The business logic is applied in the UserDAO layer where the user data is inserted as a new record in User table.

Search Functionality

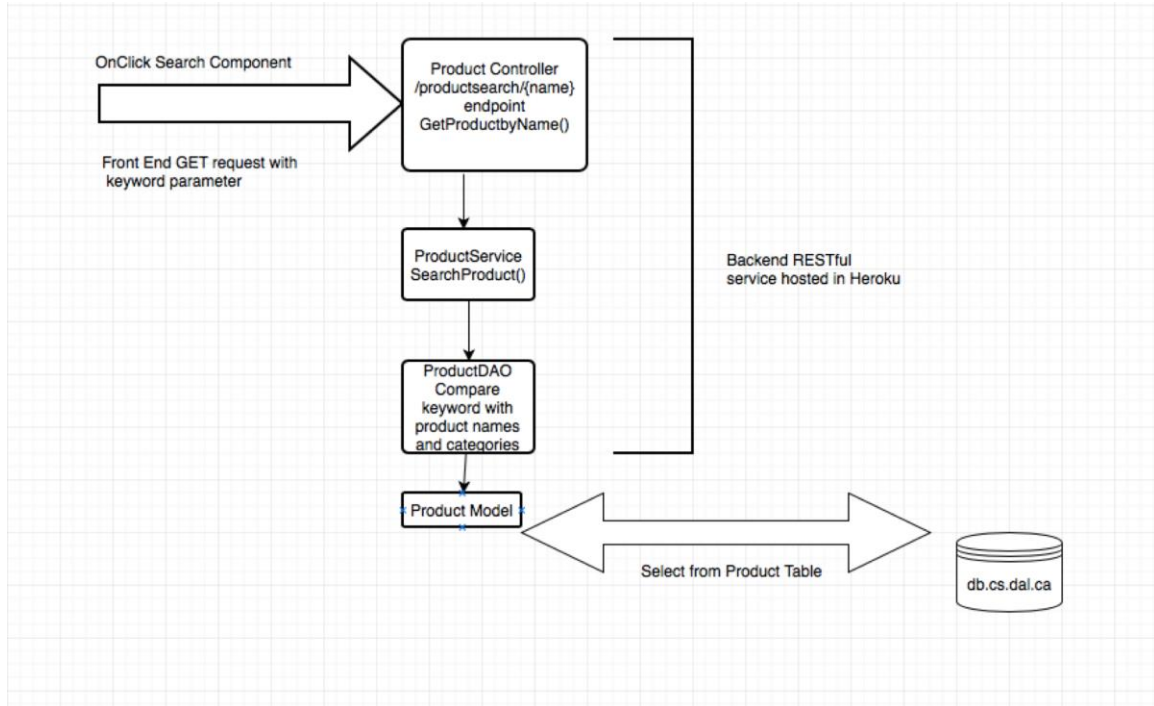


Figure 33 – Process Flow: Search Functionality

- Product Controller handles the incoming GET request from the front end search component
- Product DAO fetches the list of products whose names and categories match with the given keyword and sends it to front end in JSON format.

Update User Profile

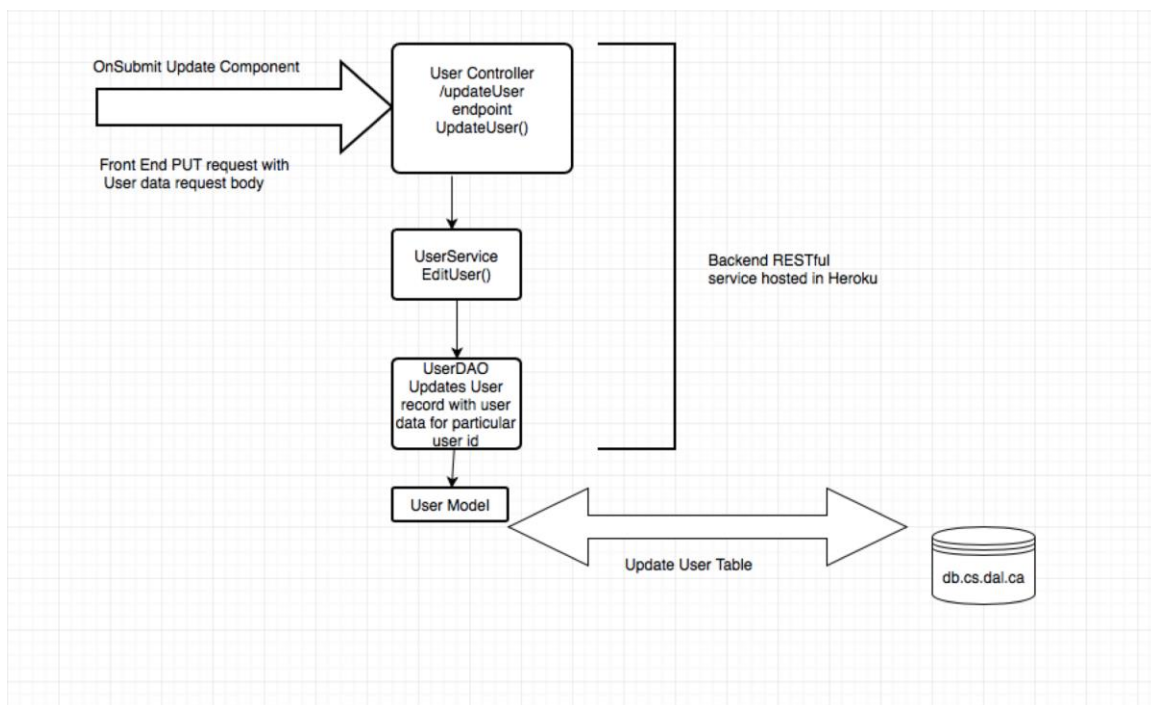


Figure 34 – Process Flow: Update User Profile

- User controller handles the Update request from the frontend.
- UserDAO updates the database with the user data for the specific user id.

View Order History

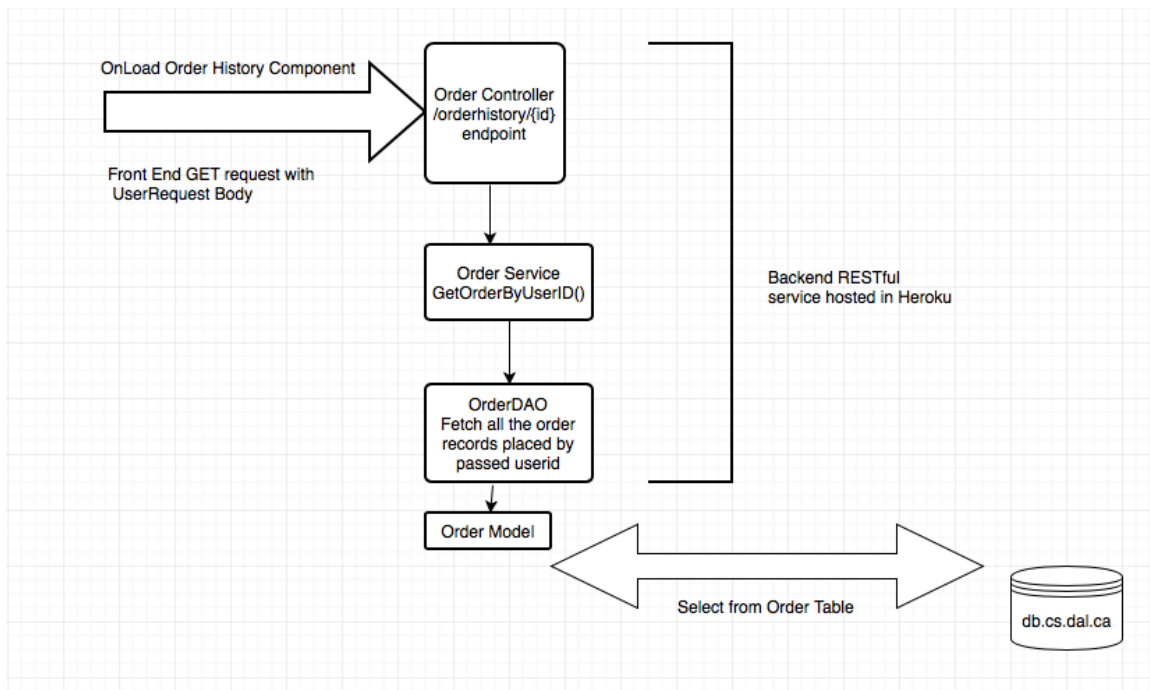


Figure 35 – Process Flow: Order History

- Order Controller handles the Order history request from the frontend.
- The corresponding GetOrderByUserID() method is triggered in OrderService Class.
- OrderDAO retrieves all the order records for the particular user and transmitted to the frontend in JSON format.

Feedback page

1) Write feedback endpoint

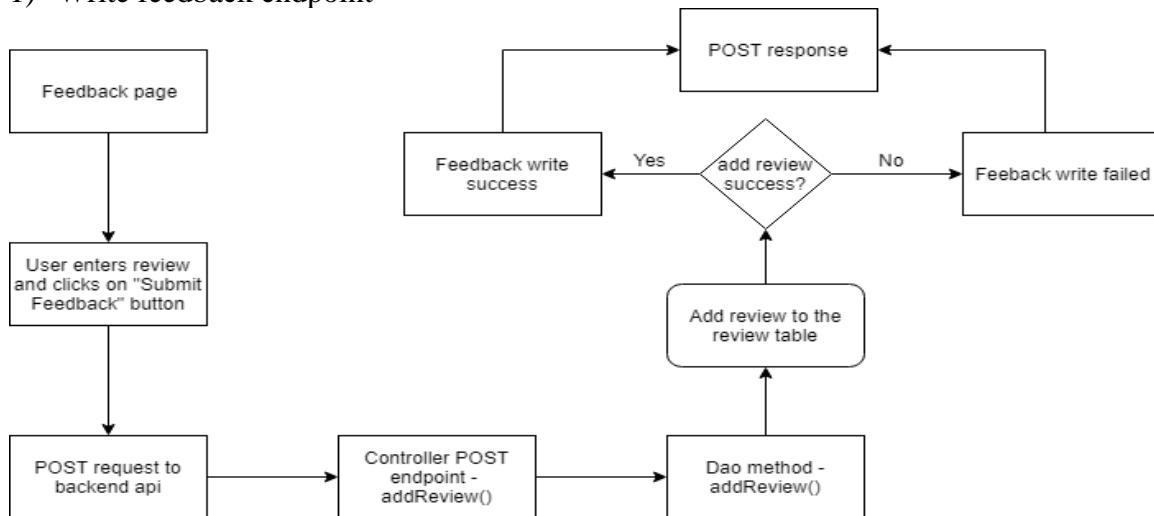


Figure 36 – Process Flow: Feedback Functionality

- Feedback controller in the backend handles the POST request of the writing the feedback to the database.
- Review DAO handles the query to write the data to the review table in the database

2) Get feedbacks based on product id

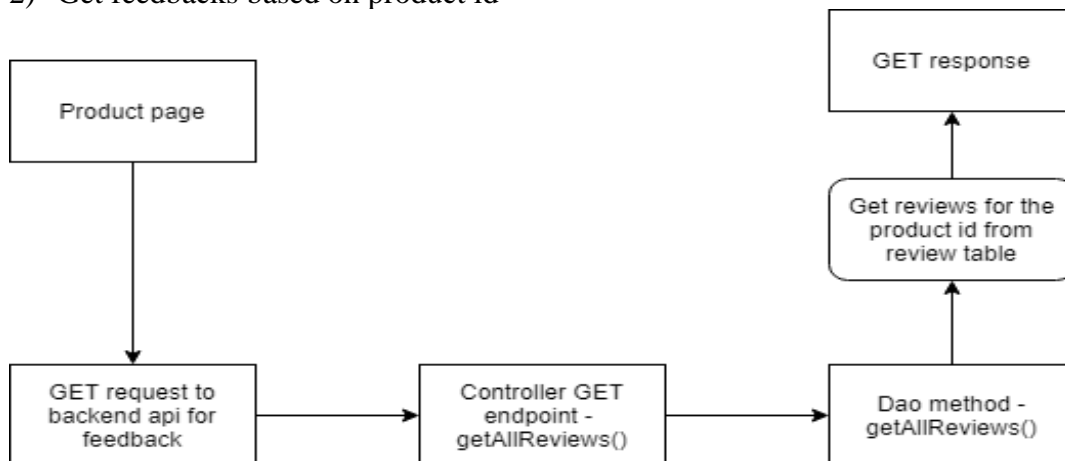


Figure 37 – Process Flow: Retrieve Feedback based on Product ID

- Feedback controller handles the GET request for reading all the reviews for the particular product id.
- Feedback DAO handles the query to read all the reviews from the review table in the database.

6. CONCLUSION

It has been a wonderful experience developing this project. MedPlus, an e-pharmacy service where pharmaceutical drugs could be purchased at the click of a button and delivery would be provided at customers' door step. Being developed with Angular 6 Frontend framework, all the functionalities are developed as individual components. This helped the application to be modular and assisted in parallel development. Similarly, business logic and data access layers of the project are developed as independent RESTful services deployed in Heroku. The data is transmitted in JSON format between frontend and backend format. Some of the main functionalities of the application is searching for the product, user registration and login, adding the pharmaceutical drugs to cart and placing order. There are also provisions to view order history and update user profile information.

There were many issues faced during the development of the project. Processes such as hosting backend in Heroku, JSON communication between Angular and Rest API posed challenges. Since the development was happening in an agile environment, there was an opportunity to retrospect our work and discuss the solutions for the difficulties faced.

7. References

Tools:

- [1] "gloomaps," [Online]. Available: <https://www.gloomaps.com/>
- [2] Microsoft, [Online]. Available: <https://www.office.com/launch/powerpoint?ui=en-US&rs=US&auth=2>
- [3] Draw.io, [Online]. Available: <http://draw.io>