



CSCI 5308
Advanced Topic in Software Development

Project Documentation

Group 23
Development Team of: Be The Donor

Prof: Tushar Sharma

Team Members

Manali Shah B00890746
Prachi Raval B00883324
Dharmik Soni B00867641
Dhairya Shah B00900984
Jayshree Ramasubramaniam B00894948

Project Documentation

Overview of Be The Donor

Be the Donor website is made aiming to assist covid sufferers in obtaining their necessities. They might find themselves in not a good financial situation or won't be able to go to buy essentials by themselves. To solve this problem, patients can pick which of their necessary things will be visible to order. Following that, anyone can help them by becoming a donor and making a financial contribution towards patients. Donors can assist many patients, depending on their needs. They also can donate anonymously towards the cause. Riders can assist patients by bringing essentials to their location of which the donations have been made.

Flow of Be The Donor

Users must register on the site by giving relevant information and specifying their user type, such as Patient, Donor, Rider, or Admin. The patient will be brought to a page where they can choose the item and other necessities that they require. The order would then be placed in the donor section, where it would be available for donations. The patient can check the status of their order, such as whether it is in "Order placed and pending payment," "pending delivery," "ready to deliver," or "delivered" status. The orders placed by the patients are made accessible for donation on the donor page. Donors can donate to any order or donate anonymously, of which the amount will be automatically assigned to orders based on the time and total amount of order. The orders that have been paid for will be displayed in the rider's section enabling riders to choose whether or not to deliver. The rider would also be given a tip to assist them.

Technologies Used:

Frontend: Html, CSS, JS, Thymeleaf
Backend: Java
Framework: springboot
Architecture: MVC layered architecture
Database: mysql
Hosted on: Heroku

CI/CD Stages:

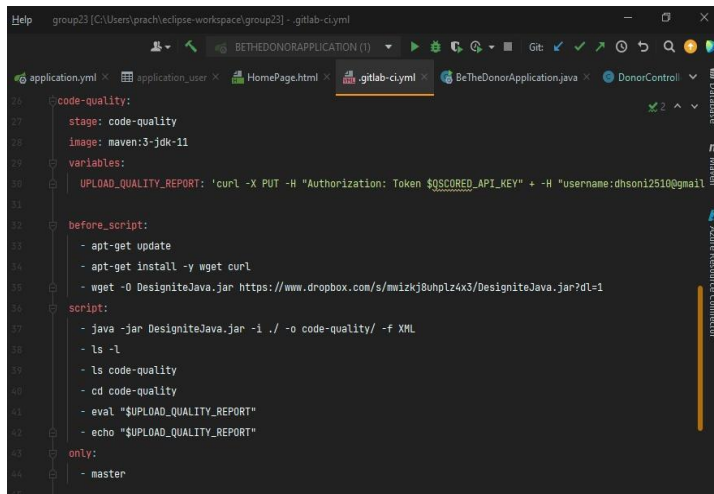
We have set the following stages for the CI/CD pipeline

Build – To run our application perform the following command

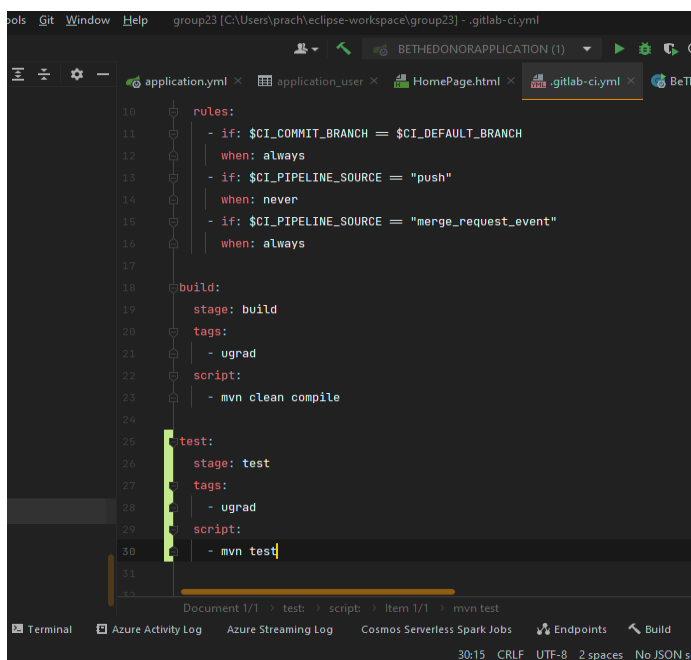
Test – To test our application we have setup a test pipeline

Code Quality – By running this code, we generate a Qscore document for code smells.

Deploy – Deployment on Heroku with this command.



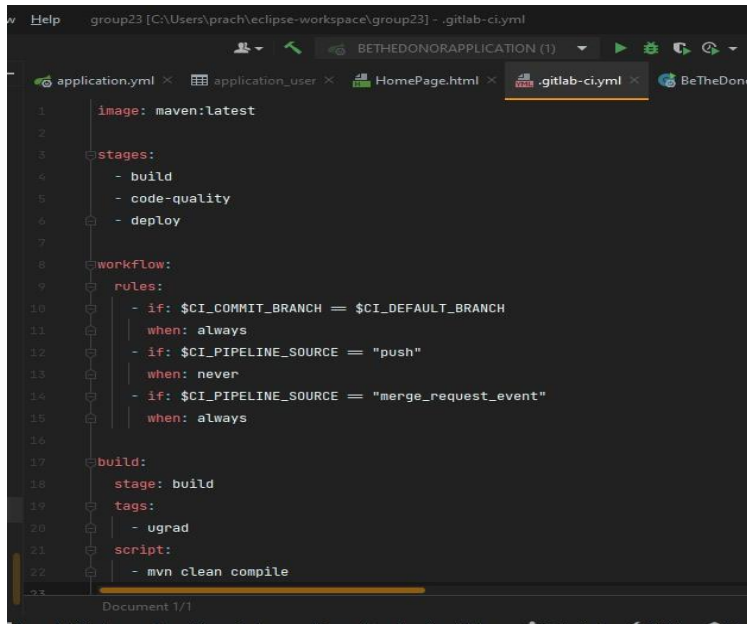
```
code-quality:
  stage: code-quality
  image: maven:3-jdk-11
  variables:
    UPLOAD_QUALITY_REPORT: 'curl -X PUT -H "Authorization: Token $QSCORED_API_KEY" + -H "username:dson12510@gmail'
  before_script:
    - apt-get update
    - apt-get install -y wget curl
    - wget -O DesigniteJava.jar https://www.dropbox.com/s/mwizkj8uhtz4x3/DesigniteJava.jar?dl=1
  script:
    - java -jar DesigniteJava.jar -i ./ -o code-quality/ -f XML
    - ls -l
    - ls code-quality
    - cd code-quality
    - eval "$UPLOAD_QUALITY_REPORT"
    - echo "$UPLOAD_QUALITY_REPORT"
  only:
    - master
```



```
rules:
  - if: $CI_COMMIT_BRANCH == $CI_DEFAULT_BRANCH
    when: always
  - if: $CI_PIPELINE_SOURCE == "push"
    when: never
  - if: $CI_PIPELINE_SOURCE == "merge_request_event"
    when: always

build:
  stage: build
  tags:
    - ugrad
  script:
    - mvn clean compile

test:
  stage: test
  tags:
    - ugrad
  script:
    - mvn test
```



Dependencies:

The following are the **Internal Dependencies** used for building this project:

- Spring Boot JPA
 - `<dependency>`
 `<groupid>org.springframework.boot</groupid>`
 `<artifactid>spring-boot-starter-data-jpa</artifactid>`
 `</dependency>`
- Spring Boot Mail
 - `<dependency>`
 `<groupid>org.springframework.boot</groupid>`
 `<artifactid>spring-boot-starter-mail</artifactid>`
 `</dependency>`
- Spring Boot Security
 - `<dependency>`
 `<groupid>org.springframework.boot</groupid>`
 `<artifactid>spring-boot-starter-security</artifactid>`
 `</dependency>`
- Gson
 - `<dependency>`
 `<groupid>com.google.code.gson</groupid>`
 `<artifactid>gson</artifactid>`
 `<version>2.9.0</version>`
 `</dependency>`
- Npm

- ```
<dependency>
 <groupid>org.webjars.npm</groupid>
 <artifactid>bootstrap-autocomplete</artifactid>
 <version>2.3.7</version>
</dependency>
```

- Persistence API

- ```
<dependency>
  <groupid>javax.persistence</groupid>
  <artifactid>persistence-api</artifactid>
  <version>1.0.2</version>
</dependency>
```

- Mysql Connector

- ```
<dependency>
 <groupid>mysql</groupid>
 <artifactid>mysql-connector-java</artifactid>
 <scope>runtime</scope>
</dependency>
```

- Project Lombok

- ```
<dependency>
  <groupid>org.projectlombok</groupid>
  <artifactid>lombok</artifactid>
  <optional>true</optional>
</dependency>
```

- Spring Boot Test

- ```
<dependency>
 <groupid>org.springframework.boot</groupid>
 <artifactid>spring-boot-starter-test</artifactid>
 <scope>test</scope>
</dependency>
```

- Spring Security Test

- ```
<dependency>
  <groupid>org.springframework.security</groupid>
  <artifactid>spring-security-test</artifactid>
  <scope>test</scope>
</dependency>
```

- Spring Boot Thymeleaf

- ```
<dependency>
 <groupid>org.springframework.boot</groupid>
 <artifactid>spring-boot-starter-thymeleaf</artifactid>
</dependency>
```

- Spring Boot Validation

- ```
<dependency>
  <groupid>org.springframework.boot</groupid>
```

```
<artifactid>spring-boot-starter-validation</artifactid>
</dependency>
```

- **Model Mapper**

- ```
<dependency>
 <groupid>org.modelmapper</groupid>
 <artifactid>modelmapper</artifactid>
 <version>2.4.5</version>
</dependency>
```

- **JAVAX Blind**

- ```
<dependency>
  <groupid>javax.xml.bind</groupid>
  <artifactid>jaxb-api</artifactid>
  <scope>runtime</scope>
</dependency>
```

- **Spring Framework**

- ```
<dependency>
 <groupid>org.springframework</groupid>
 <artifactid>spring-context</artifactid>
 <version>5.3.16</version>
</dependency>
```

- **Spring Framework Boot**

- ```
<dependency>
  <groupid>org.springframework.boot</groupid>
  <artifactid>spring-boot-devtools</artifactid>
</dependency>
```

- **Spring Framework Security Core**

- ```
<dependency>
 <groupid>org.springframework.security</groupid>
 <artifactid>spring-security-core</artifactid>
 <version>5.6.2</version>
</dependency>
```

- **Spring Framework Security Crypto**

- ```
<dependency>
  <groupid>org.springframework.security</groupid>
  <artifactid>spring-security-crypto</artifactid>
  <version>5.6.2</version>
</dependency>
```

- **Webjars**

- ```
<dependency>
 <groupid>org.webjars</groupid>
 <artifactid>webjars-locator</artifactid>
 <version>0.42</version>
</dependency>
```

- JSON

- ```
<dependency>
  <groupId>com.googlecode.json-simple</groupId>
  <artifactId>json-simple</artifactId>
  <version>1.1.1</version>
</dependency>
```

- JWT

- ```
<dependency>
 <groupId>io.jsonwebtoken</groupId>
 <artifactId>jjwt</artifactId>
 <version>0.2</version>
</dependency>
```

- Bootstrap

- ```
<dependency>
  <groupId>org.webjars</groupId>
  <artifactId>bootstrap</artifactId>
  <version>5.1.1</version>
</dependency>
```

- Jupiter API

- ```
<dependency>
 <groupId>org.junit.jupiter</groupId>
 <artifactId>junit-jupiter-api</artifactId>
 <scope>test</scope>
</dependency>
```

- Jupiter Engine

- ```
<dependency>
  <groupId>org.junit.jupiter</groupId>
  <artifactId>junit-jupiter-engine</artifactId>
  <scope>test</scope>
</dependency>
```

- Thymeleaf

- ```
<dependency>
 <groupId>org.thymeleaf</groupId>
 <artifactId>thymeleaf</artifactId>
 <version>3.0.11.RELEASE</version>
</dependency>
```

- Thymeleaf Spring

- ```
<dependency>
  <groupId>org.thymeleaf</groupId>
  <artifactId>thymeleaf-spring5</artifactId>
  <version>3.0.11.RELEASE</version>
</dependency>
```

- Junit Launcher

- ```
<dependency>
 <groupId>org.junit.platform</groupId>
 <artifactId>junit-platform-launcher</artifactId>
 <scope>test</scope>
</dependency>
```

- Junit Engine

- ```
<dependency>
  <groupId>org.junit.vintage</groupId>
  <artifactId>junit-vintage-engine</artifactId>
  <scope>test</scope>
</dependency>
```

- H2Database

- ```
<dependency>
 <groupId>com.h2database</groupId>
 <artifactId>h2</artifactId>
 <scope>test</scope>
</dependency>
```

- Stripe Java

- ```
<dependency>
  <groupId>com.stripe</groupId>
  <artifactId>stripe-java</artifactId>
  <version>20.112.0</version>
</dependency>
```

- Mockito

- ```
<dependency>
 <groupId>org.mockito</groupId>
 <artifactId>mockito-inline</artifactId>
 <version>3.8.0</version>
 <scope>test</scope>
</dependency>
```

- Spring Boot Web

- ```
<dependency>
  <groupId>org.mockito</groupId>
  <artifactId>mockito-inline</artifactId>
  <version>3.8.0</version>
  <scope>test</scope>
</dependency>
```

- Spring Boot JSON

- ```
<dependency>
 <groupId>org.springframework.boot</groupId>
 <artifactId>spring-boot-starter-web</artifactId>
 <!-- Exclude the default Jackson dependency -->
 <exclusions>
```



```
<exclusion>
 <groupid>org.springframework.boot</groupid>
 <artifactid>spring-boot-starter-json</artifactid>
</exclusion>
</exclusions>
</dependency>
```

### Plugins:

- Spring Boot Maven Plugin

### Build and Deployment:

- mvn clean install

### Use Case Diagram

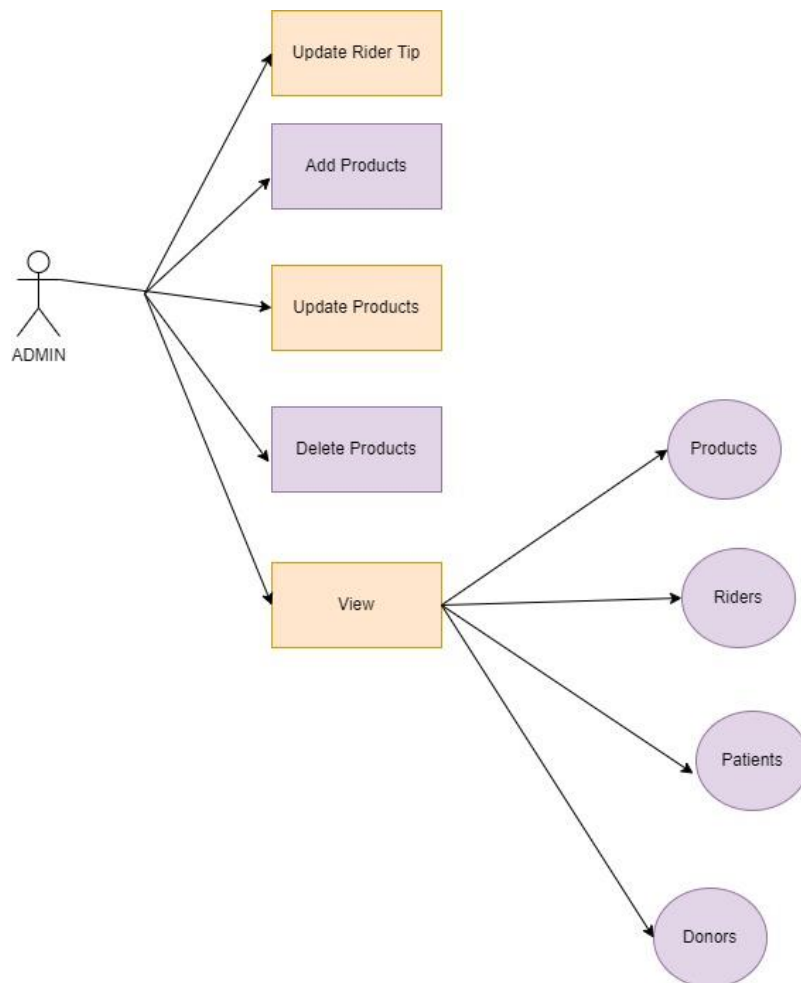


Figure 1

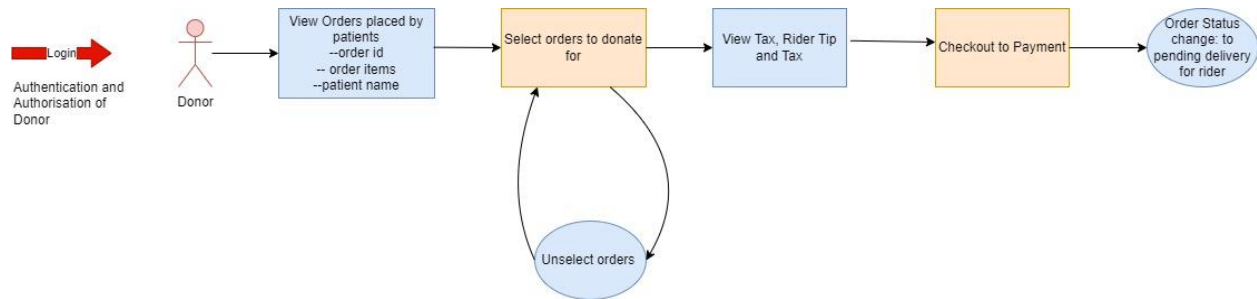


Figure 2

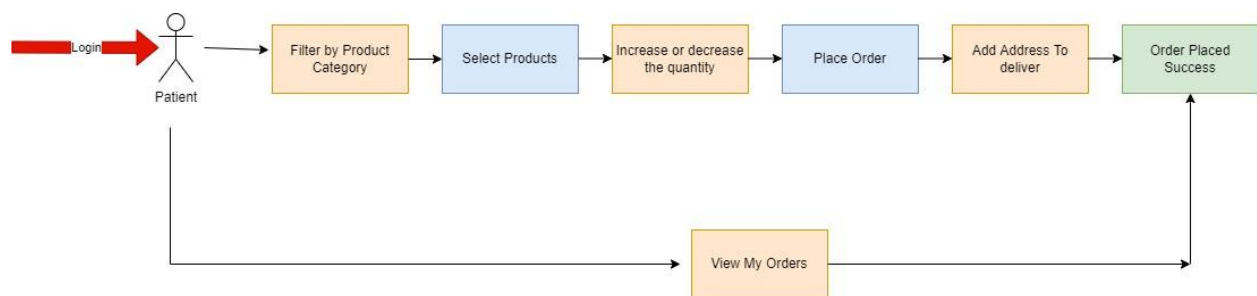


Figure 3

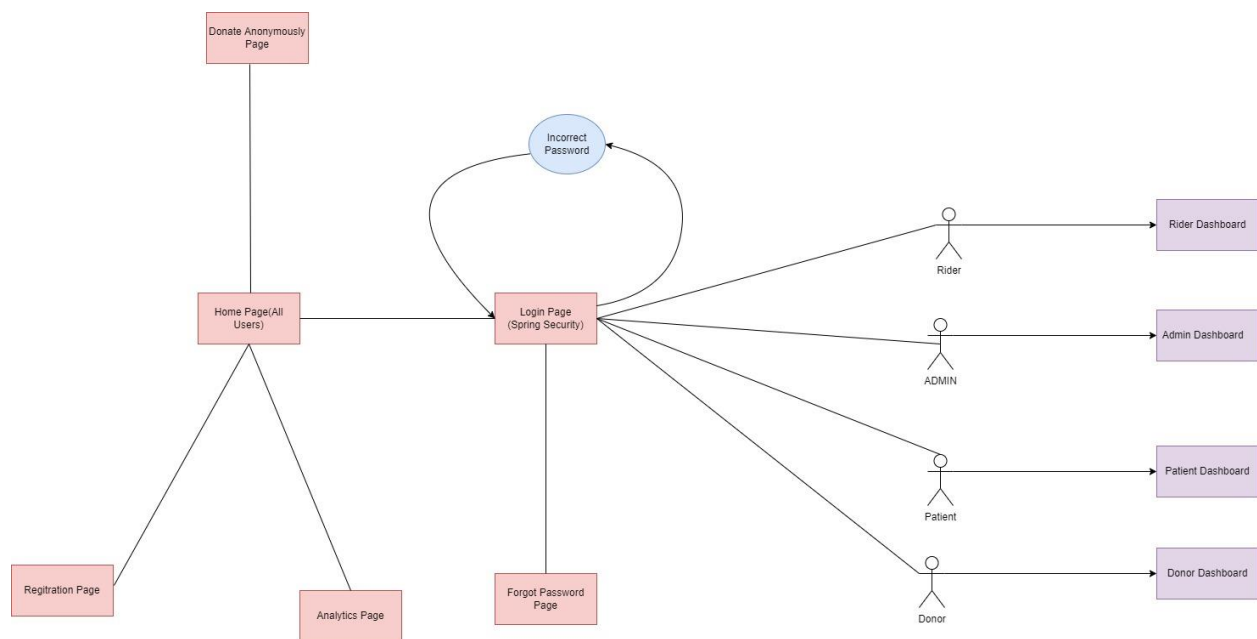


Figure 4

*User Scenarios:*

<b>Use Case</b>	1
<b>Scenario</b>	Registration
<b>Input</b>	First Name, Last Name, Email, Phone Number, Role, Password, Confirm Password
<b>Output</b>	User must receive a verification mail.
<b>Description</b>	As part of registering on the application, users must enter all valid details in-order to receive a verification mail to register successfully.

<b>Use Case</b>	2
<b>Scenario</b>	Verification Mail
<b>Input</b>	Receive an email
<b>Output</b>	Confirm by verifying on the email and get register successfully.
<b>Description</b>	After verification on email, the user gets register successfully and would be able to login into the application.

<b>Use Case</b>	3
<b>Scenario</b>	User Login Authorisation
<b>Input</b>	Email Id, Password
<b>Output</b>	User must log in successfully based on the role they registered as.
<b>Description</b>	A role base log in would be done with spring security. According to the user role, user would be directed to respective role pages and won't be able to access pages of different roles.

<b>Use Case</b>	4
<b>Scenario</b>	User Login Authentication
<b>Input</b>	Email Id, Password
<b>Output</b>	Successful login based on correct credentials
<b>Description</b>	Checks the password and username

<b>Use Case</b>	5
<b>Scenario</b>	Admin Page – Add Products
<b>Input</b>	Product Name, Quantity, Price, Category, Comment
<b>Output</b>	Product added in database.
<b>Description</b>	Admin can add products in the database as per the inventory available. Adding these products would make these products available to be displayed and selected on the patient module screen for ordering purpose.

<b>Use Case</b>	6
<b>Scenario</b>	Admin Page – Update Product
<b>Input</b>	Quantity, Price
<b>Output</b>	Quantity or price or both of product would be updated on the basis of input.
<b>Description</b>	Admin can update the details of product such as quantity, price of individual products according to the inventory he/she has.

<b>Use Case</b>	7
<b>Scenario</b>	Admin Page – Delete Product
<b>Input</b>	Click of delete product
<b>Output</b>	Product would be deleted
<b>Description</b>	Admin can delete the product according to the need. If a particular product is not present in the repository and if admin wants to delete that particular product, then admin can do so by clicking the delete button for particular product.

<b>Use Case</b>	8
<b>Scenario</b>	Admin Page – View Patients, Donors, Riders
<b>Input</b>	Null
<b>Output</b>	Would be able to view list and details of patients, donors and riders.
<b>Description</b>	Admin has the rights to view the details of users who have registered as patient, donor, or rider. Each of the 3 roles have different tables in which the details are stored and can be viewed by the admin.

<b>Use Case</b>	9
<b>Scenario</b>	Admin Page – Update Rider Tip
<b>Input</b>	Rider tip percentage
<b>Output</b>	The tip percentage of the rider would be updated.
<b>Description</b>	When a donor donates anonymously, the tip for the rider is not defined. So, this helps in giving a pre-defined tip percentage of the total amount to the rider.

<b>Use Case</b>	10
<b>Scenario</b>	Patient Page – Categories
<b>Input</b>	Select categories
<b>Output</b>	Products on display will be displayed on the basis of the category they belong to.
<b>Description</b>	Patients can look upon products based on the categories they belong to, so as to make the selection of products easier for the patients.

<b>Use Case</b>	11
<b>Scenario</b>	Patient Page – Select products
<b>Input</b>	Quantity of products
<b>Output</b>	Total amount of products chosen will be displayed.
<b>Description</b>	As patient, patient can select the items and essentials they want to order from the list of products on the display list.

<b>Use Case</b>	12
<b>Scenario</b>	Patient Page – Delivery Address
<b>Input</b>	First Name, Last Name, Address, Country, City, Province, Postal Code
<b>Output</b>	The delivery address for the patient would be set.
<b>Description</b>	After the patients select the items and essential to order, they are directed to a delivery page where details of address is taken so as to save the address of the patient where the order is to be delivered. This would be reflected in the rider module.

<b>Use Case</b>	13
<b>Scenario</b>	Patient Page – Order Request Success
<b>Input</b>	Click continue

<b>Output</b>	Order request placed successfully.
<b>Description</b>	After the patient sets the delivery address, the order request has been placed successfully and is made available on the donor section for donation.

<b>Use Case</b>	14
<b>Scenario</b>	Donor Page – Show Orders
<b>Input</b>	Select order to donate
<b>Output</b>	Grand total including subtotal, tax and rider tip will be displayed for the donor to donate.
<b>Description</b>	Donor can select single/multiple orders to donate to. This is auto populating the amount fields according to the order(s) amount. Also, total donations made previously by this particular donor person is displayed on the top section.

<b>Use Case</b>	15
<b>Scenario</b>	Donor Page – Payment Gateway
<b>Input</b>	Card Number, Expiry Date, CVV, Email Address
<b>Output</b>	The total amount will be paid from the card of the donor towards the donation amount.
<b>Description</b>	After selecting the order(s) to help, the donor can move to the payment gateway to make the payment towards the donation. By providing the card details and email, the donor can make the donation and these orders will be available on the riders section to deliver.

<b>Use Case</b>	16
<b>Scenario</b>	Rider Page – Search by location
<b>Input</b>	Location
<b>Output</b>	Orders placed for delivery at this particular location will be showed.
<b>Description</b>	Riders who have logged in can search orders to deliver based on the location they chose to deliver to.

<b>Use Case</b>	17
<b>Scenario</b>	Rider Page – Show items
<b>Input</b>	Click show items
<b>Output</b>	The total number of items, their name and quantity will be showed.
<b>Description</b>	Every shown order which has been placed has an option to see the items in the order details. Clicking the show items will open a side box to show all the items in that order.

# Project Flow and snippets

## Home

The project flow starts from the Home Page. Home page gives an overview of project structure and features. Using the home page all the user's would be able to view the details of the available features depending on the role. User's can also donate anonymously by clicking on the link on home page. User's can also donate anonymously by clicking on the link on home page.

User's can click on register and sign in from the home page.

Registration is done based on email id and user role.

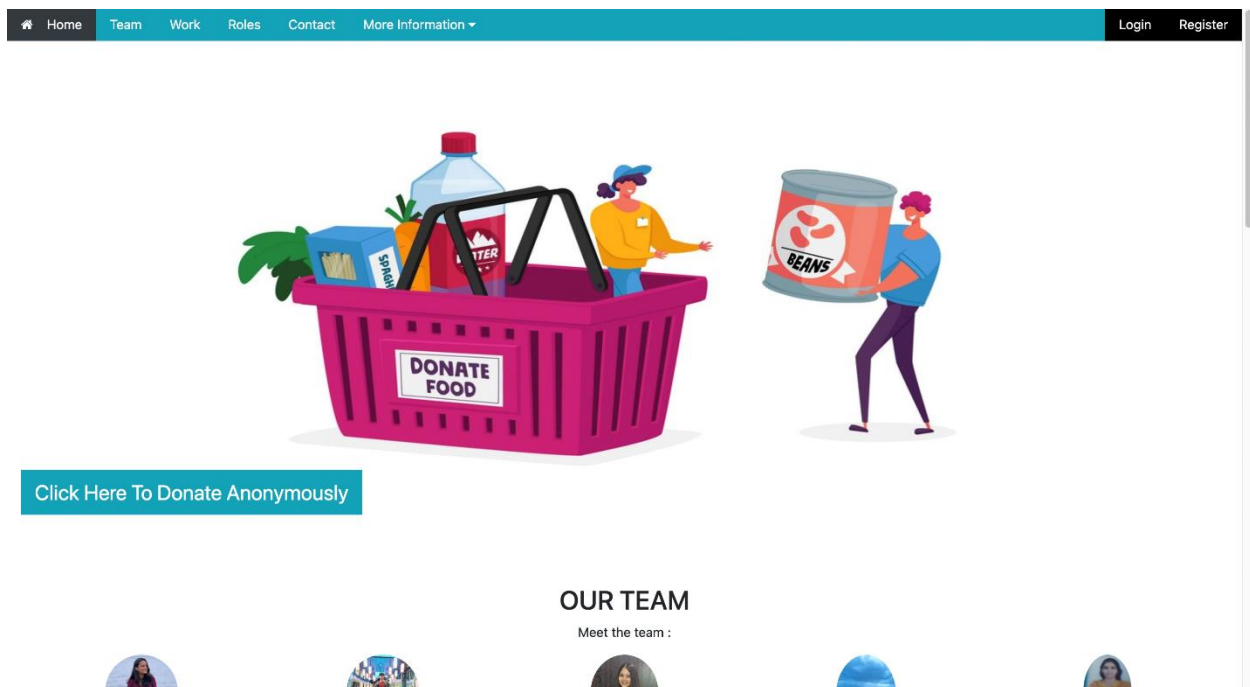


Figure 5

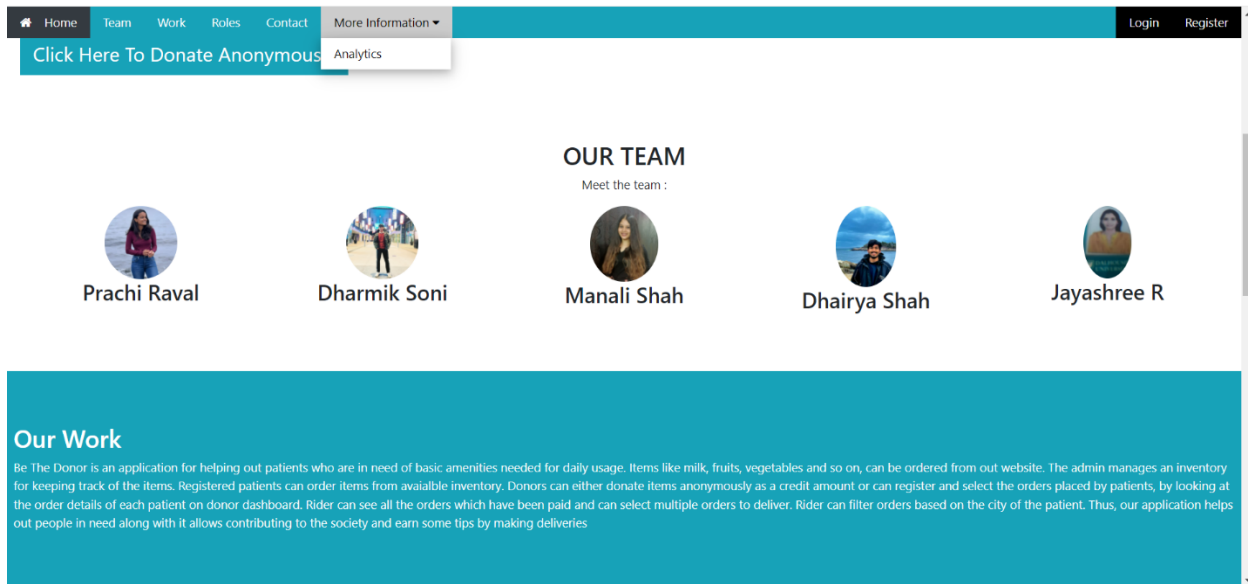


Figure 6

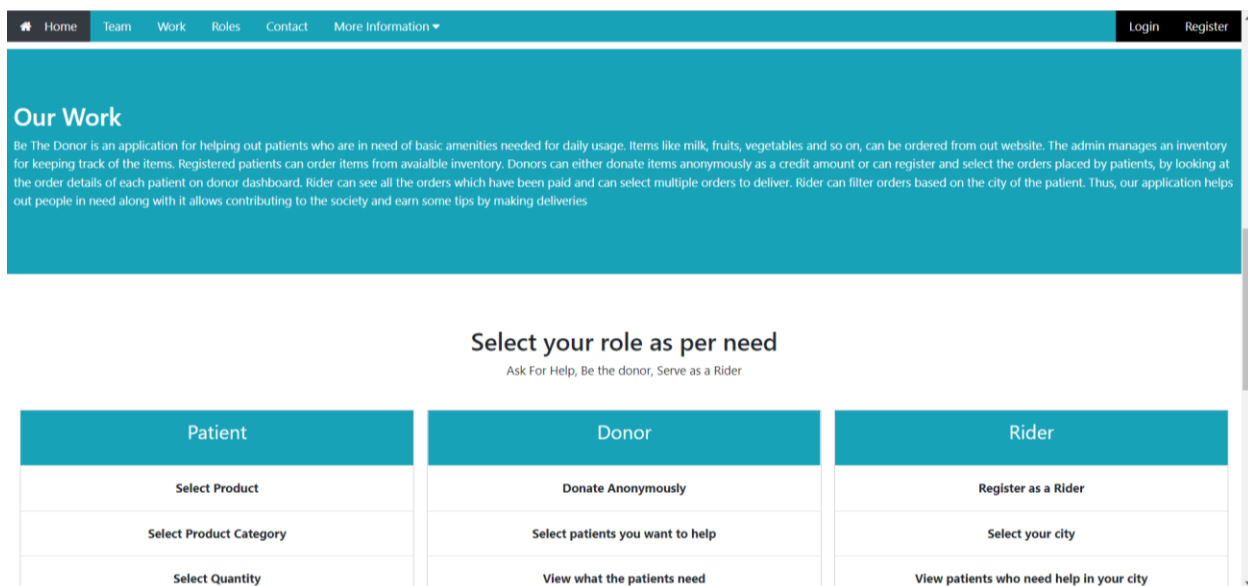


Figure 7

Home Team Work Roles Contact More Information Login Register

Register Now Register Now Register Now

### Contact Us

#### Address

Dalhousie University.

Halifax, Canada

+19029932608

bethedonor051@gmail.com

Name

Email

Message

☒ I Like it! Send

© 2022 Copyright: bethedonor.com

Figure 8

## Sign Up

After registering user will receive a confirmation email to activate their account. We have stored the password in encrypted form.

### Sign up

dhairya

shah

PATIENT

9029932608

dhairyashah051@gmail.com

\*\*\*\*\*

\*\*\*\*\*

Register

Be The Donor Go To Home Page Login Here

Figure 9



## Verification mail

The screenshot shows a web browser window with the 'Be The Donor' logo in the top left corner. In the top right corner, there are two buttons: 'Go To Home Page' and 'Login Here'. A notification banner at the top of the main content area reads 'Registration email sent successfully'. Below this, the heading 'Sign up' is displayed. To the left of the heading is a registration form with the following fields: 'FirstName', 'LastName', 'DONOR', 'PhoneNumber', 'Email', 'Password', and 'Confirm Password'. To the right of the form is an illustration of two hands holding a large heart, with a cloud-like shape behind it. At the bottom of the form is a 'Sign up' button.

Figure 10

## Verification Mail

After registering user will receive a confirmation email to activate their account.

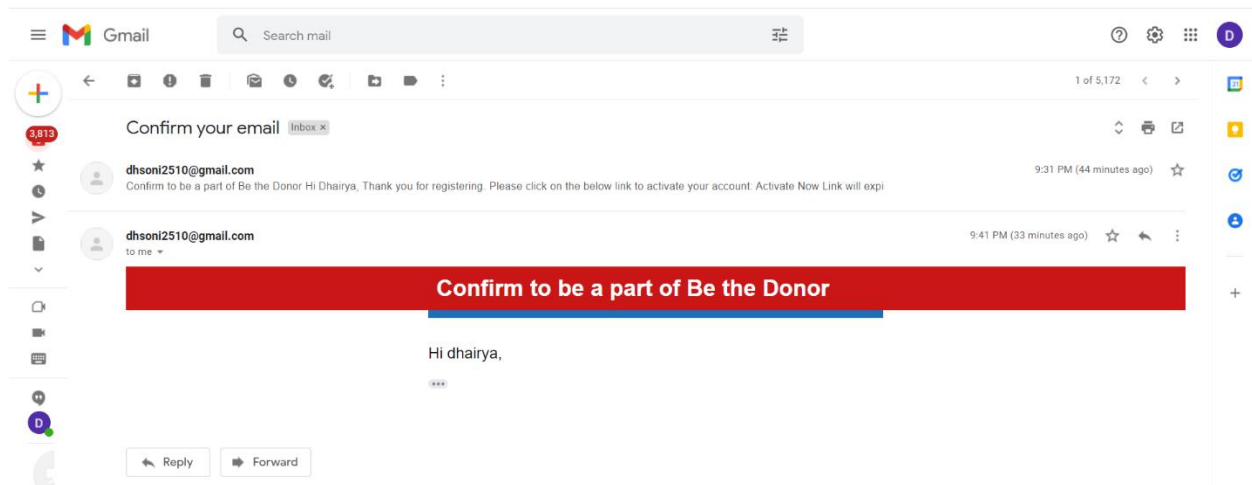


Figure 11

## Login

For login and session management we used jwt initially, but it could not support the token for multiple roles of user. Thus, we later used spring security which is used to login based on user role. There are 3 roles - rider , donor, and patient. Each type of user would only be able to login to their respective views and will not be able to access other views. The spring handles both user authorisation and authentication.

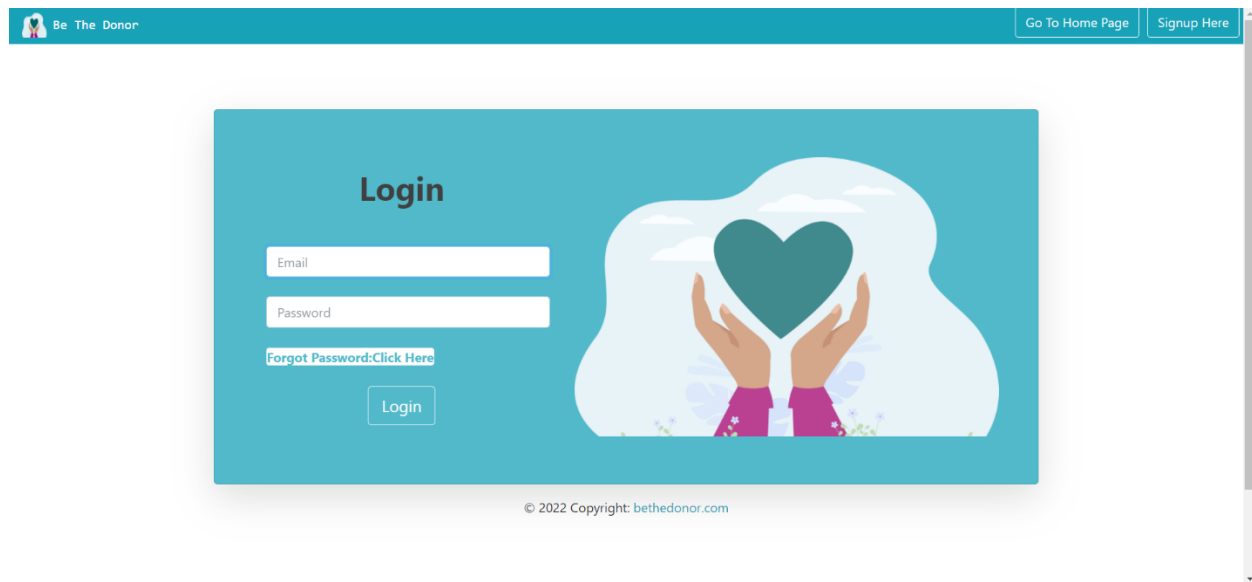


Figure 12

## Admin

Admin can add products, delete them and update the quantity based on the inventory available. Admin has the authority to view the available patients, riders, and donors registered.

**BE A DONOR - ADMIN**

**Dashboard**

Add Product

Name	Quantity	Price	Category	Measure	
<input type="text" value="Name"/>	<input type="text" value="0"/>	<input type="text" value="Price"/>	<input type="text" value="Category"/>	<input type="text" value="Measure"/>	<button>Add</button>

Products

Name	Quantity	Price	Category	Measure	Action
<input type="text" value="Santizer"/>	<input type="text" value="18"/>	<input type="text" value="250.0"/>	<input type="text" value="covid"/>	<input type="text" value="piece"/>	<button>Update</button> <button>Delete</button>
<input type="text" value="Mask"/>	<input type="text" value="898"/>	<input type="text" value="50.0"/>	<input type="text" value="covid"/>	<input type="text" value="piece"/>	<button>Update</button> <button>Delete</button>
<input type="text" value="Apples"/>	<input type="text" value="74"/>	<input type="text" value="800.0"/>	<input type="text" value="fruits"/>	<input type="text" value="piece"/>	<button>Update</button> <button>Delete</button>

Figure 13

**BE A DONOR - ADMIN**

Patients

First Name	Last Name	Email Id	Phone Number
Dhairya	Shah	dhairyashah051@gmail.com	9029831511

Donors

First Name	Last Name	Email Id	Phone Number
Manali	Shah	manali.shah127@gmail.com	9029831333

Add Rider Tip Percentage

Tip Percentage	Action
<input type="text" value="Update tip percentage"/>	<button>Update</button>

Figure 14

BE A DONOR - ADMIN

Manali

Shah

manali.shah127@gmail.com

9029831333

Add Rider Tip Percentage

Tip Percentage

Update tip percentage

Action

Update

Riders

First Name	Last Name	Email Id	Phone Number
prachi	raval	prachiraval0803@gmail.com	902996292
Manali	Shah	manali.s0106@gmail.com	9029831333
SONI	DHARMIK	dhsoni2510@gmail.com	9029892923

Figure 15

## Patient

After logging in the patient will be able to view the available orders and can order products by filtering the category. After that patient enters their address and then places the orders. Patients can also view their older orders on my orders page

Show all

fruits

Add required products to cart

Reset

Bananas

piece

- 0 +

\$50

Kiwi

kg

- 0 +

\$20

Apples

kg

- 0 +

\$50

Sub-Total

\$0

To Pay

\$0

Continue

Figure 16

## Delivery Address

Please enter your delivery address details.

FIRST NAME Dhairya	LAST NAME Shah	
ADDRESS Vesu Abhava Road,Near Nandini ,Vesu ,Surat, C-401,Nandini 2		
COUNTRY India		
CITY Surat	PROVINCE Gujarat	POSTAL CODE 395004
<button>PLACE ORDER</button>		

Figure 17

## Donor

After logging in donors can view the available orders, which have not been paid off. Donors can see the order id, patient name, order products and total amount of the orders. Donors can select and unselect the orders and view the total amount with tax and rider tip added. Donor will click on pay and will be redirected to the payment gateway. After that the order status will change to opening deliver for the rider to view.

After logging in the particular donor will be able to view total donations amount made by them in the past.

Be The Donor

Logout

### DashBoard

Total Donation Made By You:









	Orders	Amount	Select	Total
	Order id : 1 Patient Name :dhairya Products List: Bananas,Kiwi,Apples	\$170	<input type="checkbox"/> <a href="#">View</a>	\$0
	Order id : 2 Patient Name :dhairya Products List: Bananas,Kiwi	\$70	<input type="checkbox"/> <a href="#">View</a>	\$0
	Order id : 3 Patient Name :dhairya Products List: Bananas	\$50	<input type="checkbox"/> <a href="#">View</a>	\$0
	Order id : 4 Patient Name :dhairya Products List: Bananas	\$250	<input type="checkbox"/> <a href="#">View</a>	\$0

Figure 18

	Order id : 2 Patient Name :dhairya Products List: Bananas,Kiwi	\$70	<input type="checkbox"/>	\$0
	Order id : 3 Patient Name :dhairya Products List: Bananas	\$50	<input type="checkbox"/>	\$0
	Order id : 4 Patient Name :dhairya Products List: Bananas	\$250	<input checked="" type="checkbox"/>	\$250
	Order id : 5 Patient Name :dhairya Products List: Bananas,Kiwi,Apples	\$120	<input checked="" type="checkbox"/>	\$120
Subtotal				\$370.00
Tax (5%)				\$18.50
Rider Tip				\$15.00
Grand Total				\$403.50

© 2022 Copyright: bethedonor.com

[Checkout](#)

Figure 19

## Payment

### Payment Gateway

Please fill the form below to complete the order payment

Enter credit or debit card below

☐ Card number
  MM / YY CVC

Email Address

dhairyashah051@gmail.com  
 prachiraval2608@gmail.com  
 manali.s0106@gmail.com  
 9970182004  
 dhsoni2510@gmail.com  
 dh263020@dal.ca

[Pay With Your Card](#)

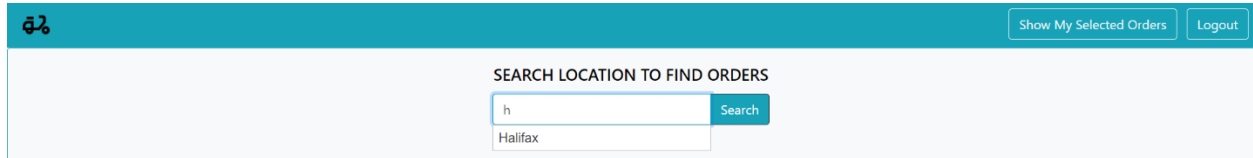
By clicking the button above, you agree to our [Terms of Service](#).

Figure 20

Donor will click on pay and will be redirected to the payment gateway. After that the order status will change to opening deliver for the rider to view.

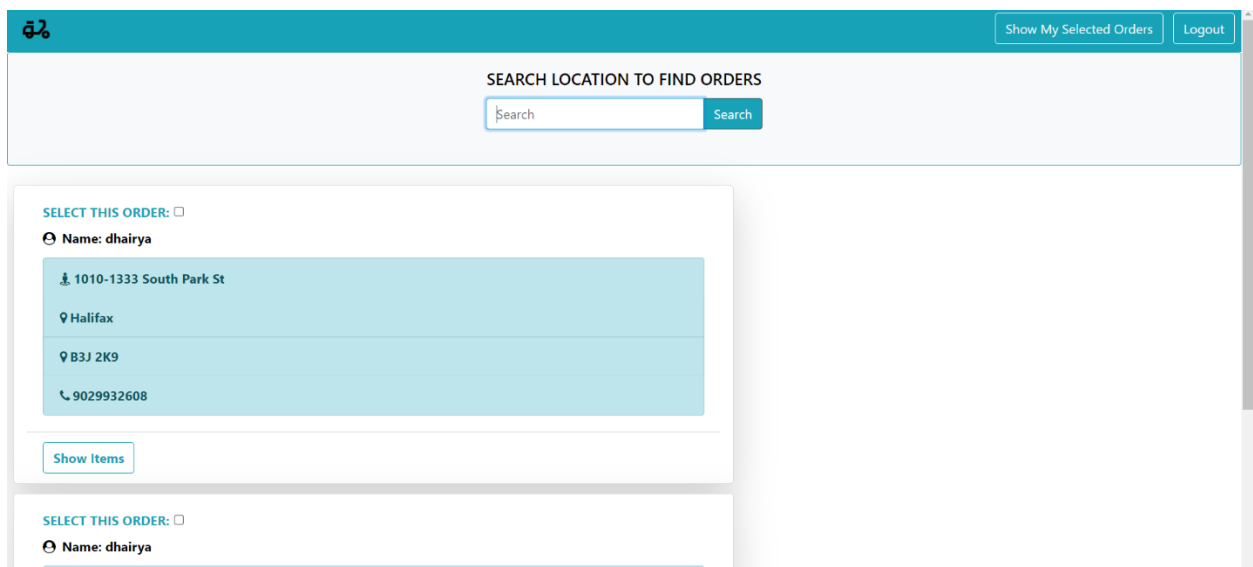
## Rider

After logging in rider can search the city and find the orders based on the city. Rider can select the orders he wants to deliver and add orders.



The screenshot shows the top section of the Rider dashboard. It features a teal header bar with a logo on the left and two buttons, "Show My Selected Orders" and "Logout", on the right. Below the header is a light gray search bar with the title "SEARCH LOCATION TO FIND ORDERS". Inside the search bar, there is a text input field containing the letter "h" and a dropdown menu showing "Halifax". A teal "Search" button is located to the right of the input field.

Figure 21



The screenshot shows the Rider dashboard with a search bar and a list of orders. The search bar is titled "SEARCH LOCATION TO FIND ORDERS" and contains a text input field with the placeholder "Search" and a teal "Search" button. Below the search bar is a list of orders. Each order is displayed in a white box with a teal border. The first order is titled "SELECT THIS ORDER: ☐ Name: dhairya". It contains a list of details: "1010-1333 South Park St", "Halifax", "B3J 2K9", and "9029932608". Below the details is a teal "Show Items" button. The second order is partially visible below the first one, showing the same title and name.

Figure 22