Client Side Foundation Assessment

Date: Friday Mar 1 2024
Assessment Time: 0900 - 1700 (including meal breaks)

Overview

There are **4 main tasks** in this assessment. Complete all tasks.

Passing mark is **65%** (**66 marks**). Total marks is **101**.

Read this entire document before attempting the assessment. There are 11 pages in this document.

IMPORTANT: Before You Start the Assessment

During the assessment, you may access any website except AI related ones like ChatGPT, Bard, etc. If you access any of these sites during your assessment, your assessment will be terminated immediately.

You must **uninstall all AI coding extensions from your IDE**. AI coding extensions are those that generate entire solutions like ChatGPT, Github Copilot, Tabnine, IntelliCode, etc. If you have installed any of the following AI extensions, please uninstall them from your IDE before you start the assessment.

Random checks will be performed during the assessment. If your IDE is found to have installed any of the above extensions or other AI coding tools, your assessment will be terminated. I forgot to uninstall is not an acceptable explanation.

Code completion extension is permissible; eg Angular Language Service, Angular Snippets, Java Language Support, Emmet, etc.

You cannot take this document out of the classroom during the assessment period. You cannot take pictures or scan this document during the assessment period. You cannot communicate with anyone

using any means when you are in the assessment classroom. If you are found to be doing this, your assessment will be terminated immediately.

Internet Access

This is an open book assessment.

You may go online to look up information. But you are **only limited to the following sites** listed below

- Java 21 - https://docs.oracle.com/en/java/javase/21/
- Spring Boot - https://docs.spring.io/spring-boot/docs/current/api/index.html
- Spring Framework - https://docs.spring.io/spring-framework/docs/current/javadoc-api/
- Thymeleaf - https://www.thymeleaf.org/documentation.html
- Jedis - https://javadoc.io/doc/redis.clients/jedis/latest/index.html
- MySQL - https://dev.mysql.com/doc
- Redis - https://redis.io/docs
- MongoDB - https://www.mongodb.com/docs/manual/
- Angular - https://angular.dev/overview
- NgRX Component Store - https://ngrx.io/guide/component-store#ngrxcomponent-store
- RxJS - https://rxjs.dev/api
- Dexie - https://dexie.org/docs/
- S3 - https://docs.aws.amazon.com/AWSJavaSDK/latest/javadoc//com/amazonaws/services/s3/package-summary.html
- Docker - https://docs.docker.com/reference/
- Railway - https://docs.railway.app
- StackOverflow - https://stackoverflow.com/
- Your GitHub repository - https://github.com/<your github user>
- Your Railway dashboard - https://railway.app/dashboard

You can access any subresources prefixed by the above URLs eg https://stackoverflow.com/questions/15182496/why-does-this-code-using-random-strings-print-hello-world is permissible.

You may use Google for searching but you can only open links listed above.

If you access any other sites that are not in the above list, your assessment will be terminated immediately. If you accidentally open a URL that is not in any of the above list, close the page IMMEDIATELY. If you linger and start to read its contents, your assessment will be terminated.

You may also reference any printed materials such as your 'cheatsheet', notes, slides, books, etc.

Assessment Setup

Assessment template

You will be given an assessment project template (ZIP file) to be used in this assessment. Unzip this project template; a directory called `vttp2023-batch4-csf-assessment-template` will be created. In the said directory, you will find the following 2 sub directories

- `data` - contains a CSV file of products
- `ecommerce` - contains a partially completed Spring Boot application running on JDK 21. The sub directory `client` contains a partially completed Angular application. All the necessary dependencies have been added to the Spring Boot and Angular projects You are free to add additional libraries if you wish to do so.

Initialise the assessment template directory as a Git repository. Create a Git repository in Github. The Github repository must initially be a **PRIVATE** repository. You should commit and push your assessment directory (`vttp2023-batch4-csf-assessment-template`). Do not wait until the end of the assessment; you should also commit and push your work as often as possible.

Make your repository **PUBLIC** after **1700 Friday March 1 2024** so that the instructors can access your work.

IMPORTANT: your assessment repository is PRIVATE and should only be accessible to yourself and nobody during the duration of the assessment. It should only be public **AFTER 1700 Friday March 1 2024**. If your work is plagiarised by others before the end of the assessment, you will be considered as a willing party in the aiding and abetting of the dishonest act.

Railway Project

Create a new Railway project. In your newly create Railway project, provision the following:

- MongoDB database
- MySQL database
- Service to deploy the Spring Boot application. The given Spring Boot should be running on JDK 21.

Assessment

In this assessment, you will be writing a shopping cart application. The frontend of the application is written in Angular and the backend is a Spring Boot application.

The application consist of 3 views (see Figure 1)

- View 0 - this is the 'landing page' of the application. View 0 displays a list of product categories. The top right corner shows the number of items currently in a customer's cart and a Checkout button. If a customer performs a checkout without any items in the cart, View 0 will display a message informing the customer that the cart is empty.
- View 1 - when a product category is selected, the application will transition to View 1. View 1 list products in the selected category. Customers can add one or more products into their cart by specifying the quantity and clicking Add. When a product is added into the cart, the 'Items in cart' (top right corner) should be updated to reflect the number of items in the cart
- View 2 - the application will transition to View 2 when the checkout button is pressed, provided the cart is not empty. In View 2, customers will enter the delivery address. View 2 will also show a list of all the products and the total order price of all the items in the cart.

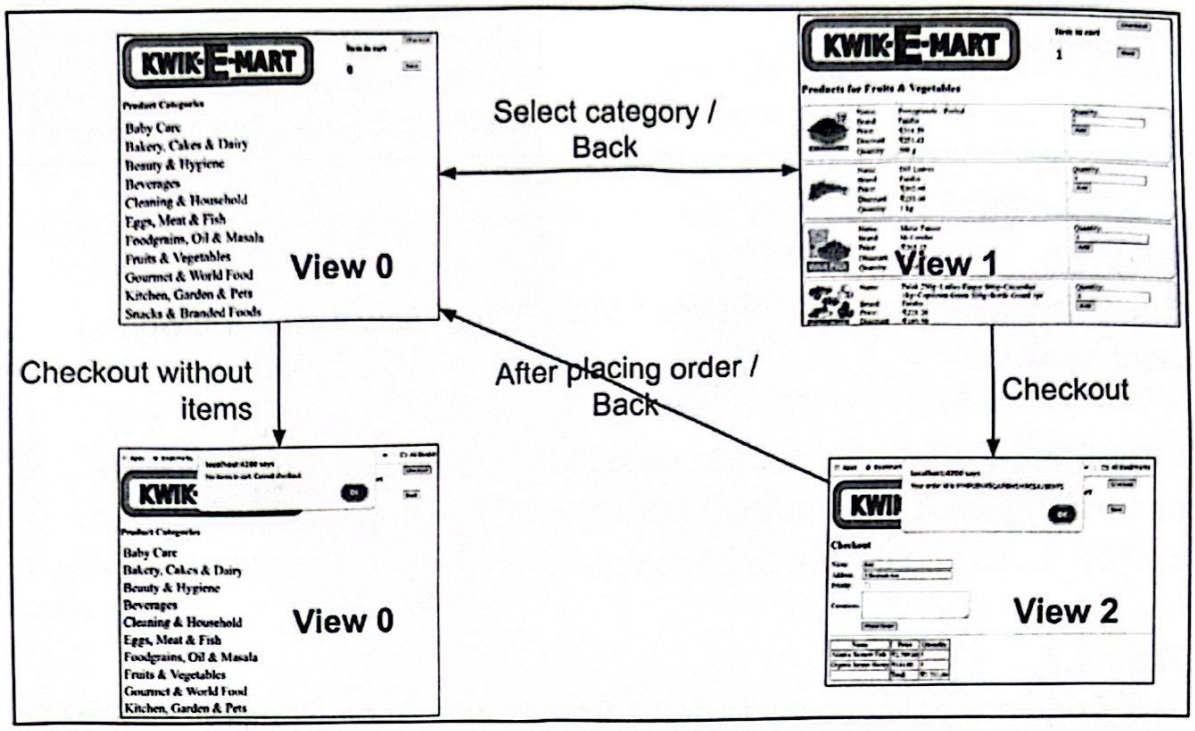

Figure 1 Application flow

Task 1 (12 Marks)

All the implementation of the interaction between View 0 and View1 and the Spring Boot application is complete. View 2 component have been generated but its functionality is incomplete.

Task 1.1

Configure the routes for View 0, View 1 and View 2 according to the following specification from Table 1

View	Component	Description
View 0	`MainComponent`	This is the first component that a customer sees when Angular starts. Displays a list of product categories
View 1	`CategoryComponent`	Transition from View 0. Displays a list of products of a selected category
View 2	`ConfirmCheckoutComponent`	Enter delivery details and perform checkout.

Wildcard		Shows View 0 when any application routes fail to match the router's configured routes

Table 1

Task 1.2
Study the data exchange between View 0 and View 1 with the Spring Boot backend.

Load the CSV file, `products.csv`, in the `data` directory. Write the command you use to import the file into Mongo in a text file called `task1_2.txt` in the `data` directory.

Task 1.3
The Angular application is to serve from Spring Boot. Configure a proxy to redirect all HTTP requests from Angular to Spring Boot during development.

Test to ensure that View 1 and View 2 are loading the categories and products from Spring Boot.

Task 2 (20 Marks)
When products are added to the cart in View 1, the products are held in a client side store. The contents of the store are only sent to the Spring Boot backend on checkout.

Task 2.1
Implement a client side store to temporarily persist the selected products. You may use any of the following techniques

- NgRX Component Store, or
- Service - implement your own store

Use the provided class `CartStore` in `cart.store.ts`.

Component Store package has been added to the Angular project.

Task 2.2

Integrate your store from Task 2.1 into View 1; whenever the Add button is pressed, add the selected product to the store.

You do not need to merge duplicate products; eg. if bread is added 2 times, then there will be 2 bread items in the store.

Task 2.3

Update the 'Items in cart' label to reflect the number of items in the store whenever a new product is added to the cart. This quantity is the number of products not its quantity; for example if you have added the following products: bread x 2, apple x 2, then the 'Items in cart' is 2 and not 4.

Task 2.4

Implement one of the following methods to prevent a customer from performing a checkout if there are no items in the store.

- Disable the 'Checkout' button when there are no items in the cart, or
- Use the Javascript function `alert()` to display a message informing the customer that the cart is empty

Task 3 (54 marks)

Task 3.1

Implement the checkout functionality for View 2. View 2 consists of 2 logical parts as shown in Figure 2 below.

The first part is the customer's delivery details and the second is the cart's contents.

Use the provided class and HTML (`ConfirmCheckoutComponent`) to create a form to capture the customer's details as described in Table 2

Field name	Type	Description
Name	string	Mandatory.
Address	string	Mandatory. Minimum of 3

		characters
Priority	boolean	Default to false
Comments	string	Not mandatory

Table 2

The 'Place Order' button should be disabled if the Table 2 requirements are not met.

Display the contents of the cart as shown in Figure 2. Calculate the total cost of the order and display the cost in the provided column. All prices are in Indian Rupee (INR). Display the Rupee currency symbol where appropriate. See Figure 2.

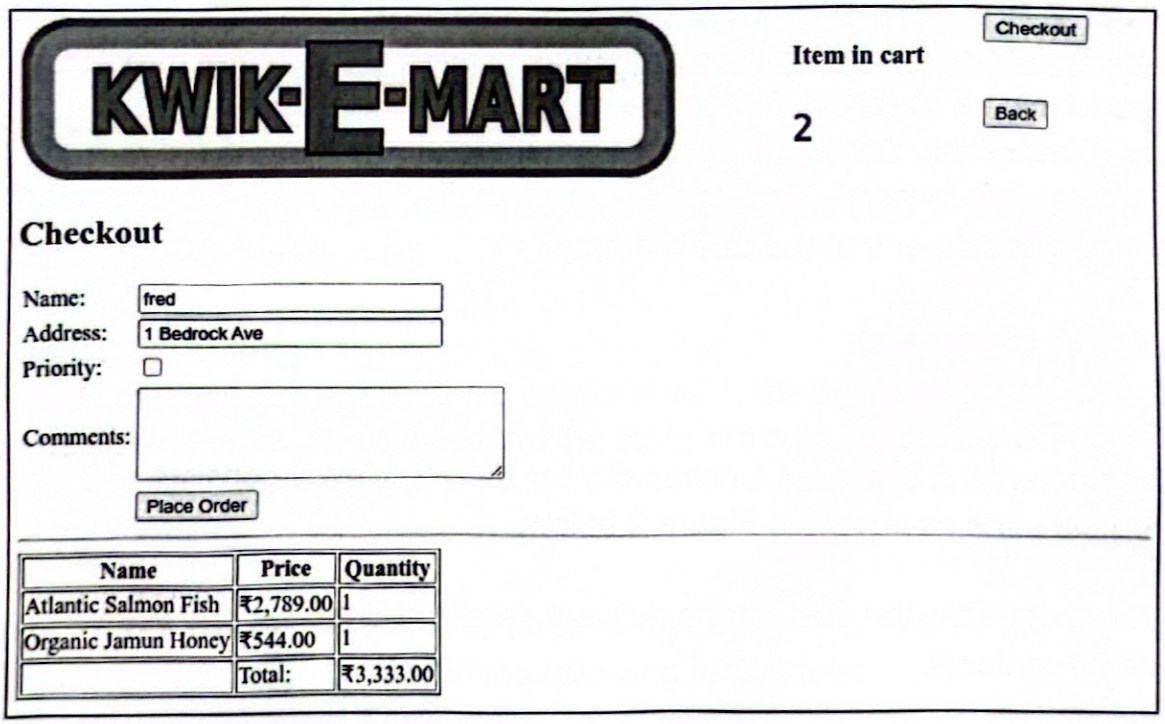

Figure 2 View 2 - Checkout

Task 3.2

When the 'Place Order' button is pressed, send the details of the order to the Spring Boot backend according to the following HTTP request

```
POST /api/order
Content-Type: application/json
Accept: application/json
```

Use the method `ProductService.checkout()` to send the order from Angular to Spring Boot.

Task 3.3

Create a database in MySQL to store the order. Write your schema is the file `task3_3.sql` in the `data` directory.

Execute `task3_3.sql` to create the database.

Hint: if you are not sure of what fields should be in the database, look at `Order` class.

Task 3.4

Write a Spring Boot controller to handle the checkout request from Angular in `OrderController.postOrder()`.

The controller should use `PurchaseOrderService.createNewPurchaseOrder()` to persist a new purchase order into the database that you have created in Task 3.3.

Implement the insert order in the `PurchaseOrderRepository.create()` method.

If the 'Place Order' is successful, return a 200 status with along with the following JSON payload

```
{ "orderId": "<the new order id>" }
```

The client should display the order id with an `alert()`. When the customer dismisses the alert box, navigate to View 0. Note: the order id is generated by the controller.

If the 'Place Order' is unsuccessful, return a 400 order status with the following JSON payload

```
{ "message": "<the error message>" }
```

Display the error message with an `alert()` and remain in View 2.

Task 4 (15 marks)
Create a Dockerfile to containerised the ecommerce application. The frontend Angular application should be served by Spring Boot.

The databases configurations eg. password, etc. should not be exposed either in `application.properties` or hard coded in the source code. Marks will be deducted if they are exposed.

Deploy your assessment to Railway. Remember that the Angular frontend should be served from the Spring Boot backend.

Do not undeploy your application from Railway until after **0900 Friday March 08 2024**.

Submission

You must submit your assessment by pushing it to your repository at GitHub.

Only commits on or before **1700 Friday March 1 2024** will be accepted. Any commits after **1700 Friday March 1 2024** will not be accepted. No other form of submission will be accepted (eg. ZIP file).

Your Railway deployment (**Task 4**) must be from a commit of your repo on or before **1700 Friday March 1 2024**. Remember to **make your repository public after 1700 Friday March 1 2024** so the instructors can review your submission.

After committing your work, post the following information to Slack channel `#04-csf-submission`

1. Your name (as it is shown in your NRIC)

2. Your email (as in Canvas)
3. Git repository URL. Remember to make your repository public after **1700 Friday March 1 2024**
4. Railway deployment URL. Do not undeploy your application and related data stores until after **0900 Friday March 08 2024**

It is your responsibility to ensure that all the above submission requirements are met. Your assessment submission will not be accepted if

1. Any of the 4 items mentioned above is missing from `#04-csf-submission` channel, and/or
2. Your information did not comply with the submission requirements eg. not providing your full name as per your NRIC, and/or
3. Your repository is not publicly accessible **after 1700 Friday March 1 2024**

You should post the submission information to the Slack channel `#04-csf-submission` **no later than 1715 Friday March 1 2024.**

Academic Integrity

This is an open book assessment. You may search the Internet for resources or use reference books during the assessment. The assessment must be your own work. You cannot ask a third party to write any part of this assessment or use AI tools such as ChatGPT to generate output and submit it as part of your assessment. This will result in an automatic disqualification from the assessment.

The NUS ISS takes a strict view of cheating in any form, deceptive fabrication, plagiarism and violation of intellectual property and copyright laws. Any student who is found to have engaged in such misconduct will be subject to disciplinary action by NUS ISS.

You are to ensure the integrity and working condition of your PC/notebooks (eg. wireless/internet connection, battery, screen, accidents like water spillage) during the assessment. NUS ISS will not accept any of these as a reason for deferring or retaking your assessment.