Software Design
Assignment 2

FoodPanda wants to make its way back into our lives :(
They are in desperate need of a Web application that only you can design and implement!

Now, the application need to accommodate only 2 types of users (lucky you):

- Regular User/Customer
- Restaurant Administrator

The ***Restaurant Administrator*** should be able to:

- [ ] log-in to the application
- [ ] add restaurant to the application
    - [ ] information needed: name, location, available delivery zones
- [ ] select category and add foods for each category
- [ ] view menu (with all existing foods, separated into categories)
- [ ] accept/decline order
- [ ] view orders (with status)
- [ ] filter orders by status
- [ ] change status of order (in order)

The ***Customer*** should be able to:

- [ ] register into the application
    - [ ] password should be encoded in the database
- [ ] log-in into the application
- [ ] view restaurants
- [ ] view menu for the selected restaurant
- [ ] add foods to cart
- [ ] place order
- [ ] see status of the order
- [ ] see history of all orders placed
- [ ] search the restaurants by name

Note: *The categories are predefined by you (eg. Breakfast, Lunch, Dessert, Beverages etc.). A menu will contain a number of categories and each category has a number of foods. A food should have: a name, a description, a price.*

Note2: *Order statuses:*

- *PENDING (just placed by Customer)*
- *ACCEPTED*
- *IN DELIVERY*
- *DELIVERED*
- *DECLINED*

An order can have two types of flows:

1. *PENDING -> DECLINED*
2. *PENDING -> ACCEPTED -> IN DELIVERY -> DELIVERED*

Note3: *When a customer adds a food to the cart, the total amount should be calculated and displayed. A customer can add food only from one restaurant in an order. The order will go the specific admin for that restaurant. Each restaurant has to have a different admin.*

Technical constraints:

- Client-Server application (Web), written in **Java (Spring)** + **React or Angular**
- Connectivity to **relational database** + storage of data
  - each table in the database should contain at least 5 entries
- Implement using **JPA Repository**
- Implement using **Maven**
- Implement the app using **Layered/MVC Architecture**
- **Documentation**
- **Validations** of the inputs on specific flows (valid dates, non-negative values, unique username etc.)

Deliverables:

Create a **github repository** with the followings: ***documentation, source code, SQL script.*** Provide the link of the repository.

Documentation should contain:

- Database diagram
- Use case diagram
- Class diagram (show how the classes are placed inside each layer and the relationship between them - to denote Layered Architecture)

Grading:

Grade	*Requirement*
5	Documentation + Backend&Frontend for: Restaurant Admin operations (login, add restaurant, create menu - select category, add food, view menu) + Customer (register, login, view restaurants, view menu)
+1	Backend&Frontend for: Customer (add food to cart, place order) Restaurant (view orders, accept/decline order)
+1	Backend&Frontend for: Restaurant (filter order, change status) Customer (search restaurant, see order history, see status of order)
+0.5	Usage of a Creational Design Pattern
+0.5	Usage of a Behavioral Design Pattern
+0.5	Usage of a Structural Design Pattern
+0.5	Encode password in database
+1	Clean code, clean architecture, validations