DASS Assignment 1

Deadline: 22nd January, 11:55 PM

Concerned TAs - Gurkirat Singh, Kanish Anand, Kartik Garg, Shlok Pandey, Vikrant Dewangan

Post all doubts on Moodle or Teams (TA Discussions and Queries channel). Messenger requests WILL NOT BE entertained!

In this assignment, you will be learning how to build a web application following REST principles. You will be using the MERN stack to build the application:

- **React JS** for Frontend
- Backend Framework using **Express JS** implementing *REST* API.
- **Node JS** for Backend
- **MongoDB** for Database

Introduction

> 💡 After around two years of being online, students at IIIT finally got to see their campus. Some of them were disappointed to see BBC closed and were shouting *Bhaiya Cheese Maggi* outside the closed building to their utter nostalgia. Hence managers at BBC planned to restart the canteen as soon as possible so that their supremacy for Cheese Maggi can once again be established. But running a canteen can result in crowded places, which might increase the risk of an outbreak.
> Hence you have to build a web-app (**for all vendors inside IIIT**) where students can place the order in advance to any vendor, and as soon as the order is ready, they can come and take it away.

As part of the assignment, you will build a Food Ordering Portal following where users have the option to search and order various food items. At the same time, various vendors have the ability to list food items.

Requirements

Your Web Portal hosts two kinds of users - Vendors and Buyers.

Buyer Details

Each buyer should have the following attributes:

- Name
- Email [Unique]
- Contact Number
- Age
- Batch Name: [possible fields UG1, UG2, UG3, UG4, UG5]

Vendor Details

Each vendor should have the following attributes:

- Manager's Name
- Shop Name like JC, VC, BBC, etc. [Unique]
- Email Address [Unique]
- Contact Number
- Canteen Opening and Closing Time

Login and Registration [10 marks]

- Common registration page with a drop-down to choose user type (Vendor / Buyer) and have fields appear as per their profile. [5 marks]
- Common login portal redirecting to respective UI after login. [5 marks]
- Implement login using Google/Facebook API. [**Bonus 2 marks]**
- Implement CAS Login (the one which we use to login in each IIIT website using college email id) [Link Here] [**Bonus 3 marks]**

Food Item Details

Each Food Item should have the following attributes:

- Item Name
- Image upload **[Bonus 2 marks]**
- Price: integer-based pricing
- Rating [0-5]: By default, the rating should be 0 i.e. the item is unrated.
- Veg / Non-Veg
- Food add-on fields providing users an option to add taste-enhancing elements such as Cheese, Sauce, etc. Each food item has its own set of food add-ons, with their pricing <Addon, Price>. A buyer has the option to select a single quantity of each food add-on at max while buying a particular food item. e.g: (Extra Cheese, Rs. 50), (Sauce, Rs. 20), etc.

- Adding tags such as Drinks, Sweet, Cold, Hot etc. These tags that will help in finding an item of their interests. Each food item can have multiple tags.

Wallet Details

- A wallet will store the money required to buy items in a canteen.
- For an order to be placed, the wallet amount should be greater than or equal to the order amount. For e.g, If the user wants to buy an item worth 30 there should be a minimum balance of 30 in the wallet.
- To add some money to the wallet, implement a dummy money adding field, which just takes an integer amount and adds it into the wallet.
- As soon as the order is placed, the money is deducted from the wallet.
- If the order gets cancelled, the money is refunded back to the wallet.

Buyer Use Cases [45 marks]

- Profile page with buyer details and an option to edit them. [5 marks]
- The dashboard should have the following features - [10 marks]
  - Implement a Search Bar, where one can search for a food item based upon item name.
  - Implement fuzzy search based upon food item name **[Bonus 3 marks]**.
  - Add Filter section like in e-commerce websites, with following filters
    - Veg/Non-Veg: Checkbox
    - Shop Name: Multi-Select
    - Food Tags: multi-select **OR** selection
    - Price: Range filter from X to Y range. You can have two text boxes for X and Y or you can also have a Slider with 2 pins for filtering.
    - Sort in Ascending / Descending order based on:
      - Items price
      - Item Rating
- Favorites tab: A buyer can mark some food items as favorites and they should appear in a different section on the dashboard. [2 marks]
- In the food item list, for each food item, the buyer should be able to view all its properties such price, name, vendor, etc. [3 marks]
- For each food item, the buyer should be able to choose from:
  - its quantity (an integer value)
  - some addons available within a food item

  and place an order with a buy button. [5 marks]
- Wallet amount should be visible at the top, and the functionalities present in the Wallet Details section should be implemented. [5 marks]
- A buyer cannot place an order if the canteen is closed. Such items should be present at the end of the item list with an indication that identifies them as unavailable. [5 marks]

- My Orders page [10 marks]
  - This should display all the current orders, in which each order should have the following fields: **Placed Time, Vendor Name, Food Item, Quantity, Status, Cost** and **Rating**.
  - The *status field* should tell the current status of the order. It can be of 6 types -
    - PLACED
    - ACCEPTED
    - COOKING
    - READY FOR PICKUP - When the status is **Ready for Pickup**, a button name "**Picked up"** should come up. On clicking Picked up, the order status should become **Completed.**
    - COMPLETED - Every completed order should get an option to rate the food item in the **Rating** field. Once rated, the food items' average rating should get updated. [For orders not in the Completed stage, you can have the Rating field blank.]
    - REJECTED

Vendor Use Cases [45 marks]

- Profile page with vendor details and an option to edit them. [5 marks]
- A Food Menu dashboard displaying all food items of the vendor. There must be a button at the top to add a food item. For each added food item, there must be 2 options available - to Edit, and to Delete the food item. [10 marks]
- A dashboard to view orders issued to the vendor, each order should have details such as **placed time**, **food item, quantity,** along with a button [titled "MOVE TO NEXT STAGE"] to change the status of the order. This button should be dynamic in nature, i.e. the functionality of this button depends on the status of the order.

  Each order as discussed earlier, can have 6 stages - [15 marks]
  - PLACED: Initially, a placed order comes here. An additional REJECT button should be present along with the MOVE TO NEXT STAGE button.
  - ACCEPTED: Upon clicking the latter button, the order should move to the Accepted stage.
  - COOKING: The vendor can then change the state of order from the **Accepted** stage to the **Cooking** stage using the button.
  - READY FOR PICKUP: Once the order is in the **Cooking** Stage, the Vendor can use the button to change the state of the order from the **Cooking** stage to **Ready for pickup**.
  - COMPLETED: Orders that have been picked up by their respective buyers.
  - REJECTED: Orders rejected by the vendor
- A Statistics Page [10 marks]
  - Top 5 Items that have been sold
  - Counts for
    - Orders Placed
    - Pending Orders
    - Completed Orders

    - Graphs to visualise batch-wise, age-wise distribution of **completed** orders **[Bonus 5 marks]**
- Any vendor can have at-max **10** orders at ACCEPTED and COOKING stage combined. It can't move any more orders in the ACCEPTED stage till any of these orders reach READY FOR PICKUP stage. An error popup should appear in case the vendor tries to accept more. [5 marks]

Bonus [25 marks]

- You need to deploy your web app (both frontend and backend) on any platform (For eg, Heroku). Deployment is a very necessary skill to have in your pocket, please do try it. **[5 marks]**
- Email to the buyers on acceptance/rejection of their order. You can create a common email ID for mailing purposes stating that "<vendor-name>" accepted your order. **[5 marks]**
- ***Other bonuses are explained along with concerned points. Marks for every bonus are written along with them.***

Deliverables

Submission must be in the following format:

```
<roll_no>/
|--backend/
  |-- package.json
  ... other files
|--frontend/
  |-- package.json
  ... other files
|-- README.md
```

`backend` and `frontend` are directories with the `Express.js` and `React` app respectively.

Do **NOT** submit the `node_modules/` folder.

Zip this and submit it as a single **<roll_no>.zip** on moodle

There will be a **10% penalty** for not following the submission format and a **straight 0** for submitting corrupt zips

Additional Instructions

- You may use any npm package, as long as it's not off-the-shelf. (**Confirm it from TAs**)
- You are allowed to use Bootstrap/CSS templates, or Material UI.
- Follow good coding practices and good database designs.
- In the final submission, ensure that you have the correct `package.json` files. Make sure it includes all the packages you have worked with.
- Cases of plagiarism will be given a straight zero. **DO NOT COPY ANY PART FROM ANY SOURCE** including your friends, seniors, or the internet.
- Marks indicated for requirements assume proper working of frontend and handling of corresponding database operations.

- Please test your application thoroughly. Any errors occurring during the evaluation will be penalized.
- Error Handling should be well implemented such that all the edge cases are taken care of like the price of food items should accept only positive integers.
- **Please start early. The deadline is a hard one, and won’t be extended**.