Buy, Sell @ IIITH

Deadline: 8 PM on 2nd February, 2025

In today's world of late-stage capitalism, Whatscap has introduced a new policy: any organization running a Buy-Sell group will now be taxed 10% of the total profits generated across all transactions in the system. Naturally, we're not too keen on handing over a slice of our earnings!
To sidestep this, you've been tasked with developing a dedicated Buy-Sell website exclusively for the IIIT Community.

This website has to be built by you in MERN Stack:

1. **MongoDB** for Database
2. Backend Framework using **Express JS** implementing REST API
3. **Any React-based framework** for Frontend
4. **Node JS**

Requirements

Your Web Portal has users being of two kinds, one being Buyers others being Sellers. Any logged-in user can play the role of both a buyer and a seller.

Details Of Users

1. First Name
2. Last Name
3. Email [Unique, ONLY IIIT Mails are allowed!!]
4. Age
5. Contact Number
6. Password [should not be stored as plaintext in database]
7. Items added in the Cart
8. Seller Reviews (Reviews for that person)

Details of Item

1. Name
2. Price

3. Description
4. Category (example: clothing, grocery, etc.)
5. Seller ID

Details of Orders

1. Transaction ID
2. Buyer ID
3. Seller ID
4. Amount
5. Hashed OTP

You are free to add more parameters in each of the above mentioned details of different entities. You can also create new entities based on how you approach use cases and are free to change the above entities as per your implementation.

Login and Registration

1. A registration page for the users having appropriate fields as mentioned earlier [3 Marks]
2. Login portal redirecting to Profile Page for the User [3 Marks]
3. A user once logged in, should not be asked to login again even on website/browser/computer restart unless user explicitly logs out. [2 Marks]
4. Logout [1 Mark]
5. Authentication & Authorization [5 Marks]
   a. The password stored in the DB must be hashed (you can use bcrypt.js library) [1 Mark]
   b. All other pages must be accessible only after login/registration. Any attempt to go directly to those pages without logging in first should redirect the user to Auth page. [2 Marks]
   c. All the backend routes should be protected using token (you can use jsonwebtoken library) [2 Marks]
6. Implement Google Recaptcha/LibreCaptcha [Bonus 1 Mark]
7. Implement CASLogin [Bonus 3 Marks]

Use Cases

1. Dashboard

a. There must be a Navbar with links to all the pages mentioned in the following points (except 3). This navbar will be present regardless of the current page once logged in. [2 marks]
b. Profile Page: On this page show the Basic Details of the User with an Option to edit them [4 Marks]

---

2. Search Items page

This page consists of all the items that are being sold by any seller in IIIT. It should have:

a. A search bar which searches the item that the buyer is looking for (case insensitive). [2 Marks]

b. Implement a filter based on the categories of the items similar to what we have on e-commerce web sites (Multiple Categories can be selected, items having any of the the selected categories should come) [2 Marks]

c. If the search bar is empty or an empty search is given as input, all the items that are being sold by any seller in IIIT should be displayed, otherwise only the relevant results are retrieved. If any filter is applied and the search bar is empty, you need to diplay all yhe relevant items which match the filter's specifications. Relevant information includes: name, price, vendor selling this item, etc. These items can be displayed in any order. [2 Marks]

d. Clicking on any item should open the page of that particular item. [1 Mark]

---

3. Items page

Each item that is sold by some seller in the database has a unique items page. This page serves as a desciption of the item in consideration. It also serves the purpose of placing the order by the user. It should have:

a. The description of the item should be present along with it's name, price, the vendor selling the item, etc. [2 marks]

b. There should be an option to add the item in My Cart. If any item is added in the cart, relevant changes must be reflected in My Cart Page of the User. [2 Marks]

---

4. Orders History

This page keeps a track of all the orders placed/received by any user of the app. This page should have:
a. All the pending orders (placed by the user) which are yet not processed (more details have been explained below). A randomly generated OTP for the orders (received after

placing the order via My Cart page, details explained below) should also be present. [2 Marks]

b. All the items bought by the user earlier. [2 Marks]
c. All the items sold by the user earlier. [2 Marks]

Note that each of the above needs to be displayed in separate tabs for readability [1 Mark].

---

5. Deliver Items page

This page serves the purpose of display of orders received from different buyers so that sellers can keep track of the pending orders. This page should have:
a. All the items that need to be delivered to different buyers who placed the order. Basic information that needs to be shown is the name of the item, it's price along with the buyer who placed this order. [2 Marks]

b. If the seller wants to close the transaction, he/she needs the OTP from the buyer. If the OTP entered is correct, he can end the transaction, otherwise, an error should prompt that the entered OTP is not correct. (A button should be present for successful closure of the transaction). Once this button is pressed, this order would be removed from this page and the relevant changes should reflect in the orders' history page of the buyer. [4 Marks]

---

6. My Cart

a. In this page, you are required to show **ALL** the items which are added to cart by the user while shopping. Here, in this case, the user acts as a buyer who buys the items from different users (sellers). Note that a buyer cannot buy items/products from himself, i.e., the buyer and seller MUST be different. [2 Marks]

b. For each item added by the user, you are required to show the name and price, with an option to remove that item from the Cart. [1 Mark]

c. The total cost of all the items in the cart should be displayed to the user. [1 Mark]

d. **Final Order button at the bottom**: Performs the functionality of ordering all the items which are visible in this page. If this operation is successful, the My Cart page should be empty, otherwise not.[2 Marks]

e. In case the operation in part (e) is successful, this should reflect on the Deliver Items page of the respective sellers in the above and on Orders History page of the buyer. Also, the buyer receives a randomly generated OTP on the Orders History page(as mentioned

before) which must be stored in the database in Hashed Form and the transaction can be closed by the seller by entering the correct OTP. [2 Marks]

---

7. Support [Bonus 5 Marks]

To enhance user experience and foster engagement within the IIIT Community, implement an Automated ChatBot integrated into the Buy-Sell website. This chatbot, powered by any AI model (like Gemini) should be capable of responding intelligently to user queries or messages. You can use an API to access third-party models (like Gemini, ChatGPT, Llama, etc.) for the sake of implementing. You are NOT required to make the AI model. The interface should resemble a Chat UI, similar to the familiar chat support systems in apps like Flipkart, Swiggy, etc. Opening the support page would start a new session for chat and the response from the chatbot should consider the previous conversation of that particular session.

Miscellanous Details:

Wherever we say a different page should be created, there must be a corresponding URL change, i.e. the new page must not be dependent on the previous page, so even if the user copy pastes the URL and opens it in a new tab, the same result should come.

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/react-based framework respectively. **Do NOT submit node_modules folder.**

Zip this and submit it as a single <roll_no>.zip on moodle. 10% Penalty would be imposed for wrong format submissions and a straight 0 for submitting corrupt zips.

Additional Instructions

You are allowed to use any npm package unless it’s off the shelf (Confirm from TAs, on moodle). You can use any styling or component library (bootstrap, tailwind, styled-components, Material UI, etc.)

Final Submissions should have correct package.json files. DO NOT COPY, we would be checking plagiarism with everyone in your batch and also with your senior batches. Do thorough testing, any errors during evaluations will be penalized.