University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: 3

Date: Jul 26, 2024

Group Number: 27

Name	Student Number	CS Alias	E-mail Address
Jeffrey Zhai	63347439	n3c7e	jeffreygxzhai@gmail.com
Yixian Cheng	94548492	b9k9n	chengyx@student.ubc.ca
Mark Zhu	25611807	j9k7w	markziyuzhu@gmail.com

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above.

In addition, we indicate that we are fully aware of the rules and the consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia.

Brief summary of project:

This project portrays a social media platform that supports both community-based communication online and commercial activities. We aim to create an application that allows users to greet others and form communities via posting contents and participate in activities; we also allow sponsor companies to engage in activities and offer merchandise, which users could exchange for with the point credits they earned from the activities.

Task Breakdown and Timeline:

Stage	Task	Detail	Task Assignment & Deadline
Setup	Dependency Installment	Install dependencies such as react, node, express, jest, cors and oracleDB.	Jeffrey July 30
	Establish Project Structure	Update README file	N/A
		Make frontend, backend and test directories	Jeffrey July 30
Frontend	Backend Interaction	Create frontend api to interact with Oracle Database; - E.g., via HTTP methods (get/post/put/delete) + fetch method, passing in user inputs in request;	Jeffrey August 1
	User Status	Design a user status that can be maintained on the user side, so that once a user logs in, they do not need to input user information for operations requiring user info (e.g., post content, join community);	(Optional; implement if time permits)
	Pages: displays visual components	Create a homepage (index) connecting users to all other pages via a	Jeffrey July 30

		navigation bar; - If needed, offer buttons to display statistics of the database;	
		Create posts + topics page; - Should display a list of posts, each with relevant info; - Should display a list of topics; users are allowed to view posts by selecting one or multiple topics; - Should offer buttons to add/remove a post/topic;	Jeffrey July 31
		Create posts + comments Detail Page; - Should display relevant info for one single topic, including user, post time, content, likes; - Should display relevant comments; - Should offer buttons to like, favorite or remove a comment; - Should offer buttons to add or remove comment;	Jeffrey July 31
		Create account profile page; - Should display relevant info for user; - Should display communities that the user is in, with hyperlinks to community detail pages; - Should display favorite lists;	Jeffrey August 2
		Create community + activities detail page; - Should display relevant info for the specific	Yixian August 2

		community; - Should display a list of community members; - Should display a list of activities, each involving its topic list; - Should offer buttons to let user join/quit a community;	
		Create merchandise exchange page; - Should display relevant info for each merchandise; - Should offer button to let user purchase an item; - Should allow user to filter/sort items;	Yixian August 2
	Components & Styling	Create css files for customized styling;	(Optional)
		Ensure functional tables, buttons, forms, interactable components are styled.	(Optional)
Backend	Configuration: connect backend server with database	Create .env file to store connection key	Mark July 28
		Create config file for database connection setup and create a database connection object to make queries on	Mark July 28
	Models: defines the SQL query schema for every relationship and entity table	Create a schema to get data	Yixian July 30
		Create a schema to insert data	Yixian July 30
		Create a schema to delete data	Yixian July 30

		Create a schema to update data	Yixian July 30
	Services: directly interacts with the database	Create an insert function that uses the query/model to add a tuple based on values provided by the user (inserted tuple will be returned)	Mark, Yixian July 30
		Create a delete function that uses the query/model to delete a tuple based on a the given primary key (deleted tuple will be returned)	Mark, Yixian July 30
		Create an update function that uses the query/model to find the correct tuple to update with the user inputs (updated tuple will be returned)	Mark, Yixian July 30
		Create two get functions that allows the user to either retrieve all data from a table or a specific tuple from the table (retrieved data will be returned)	Mark, Yixian July 30
	Controllers: handles incoming requests and calls the necessary services	Create function to extract values from the incoming api request body	Mark Aug 2
		Create function to pass the values to the service layer for processing	Mark Aug 2
		Send back a response message to be displayed to the frontend	Mark Aug 2
	Routers: define different	Create a router for each of the four main entity groups	Mark Aug 2

	endpoints for different requests	(posts + comments, account + communities, topics + activities, and sponsors + merchandises)	
	Server: entry point as well as the starting point for backend logic	Setup an express.js server to handle frontend requests	Mark Aug 2
		Setup middlewares that will be used to parse requests	Mark Aug 2
		Use router objects for the different endpoints	Mark Aug 2
Testing	API testing	Use Postman to test API endpoints and calls	(Optional; implement if time permits)
	Frontend testing	Use RTL to test frontend components and event handler functions	(Optional; implement if time permits)
	Backend testing	Use Jest to write unit tests for services and controllers	(Optional; implement if time permits)

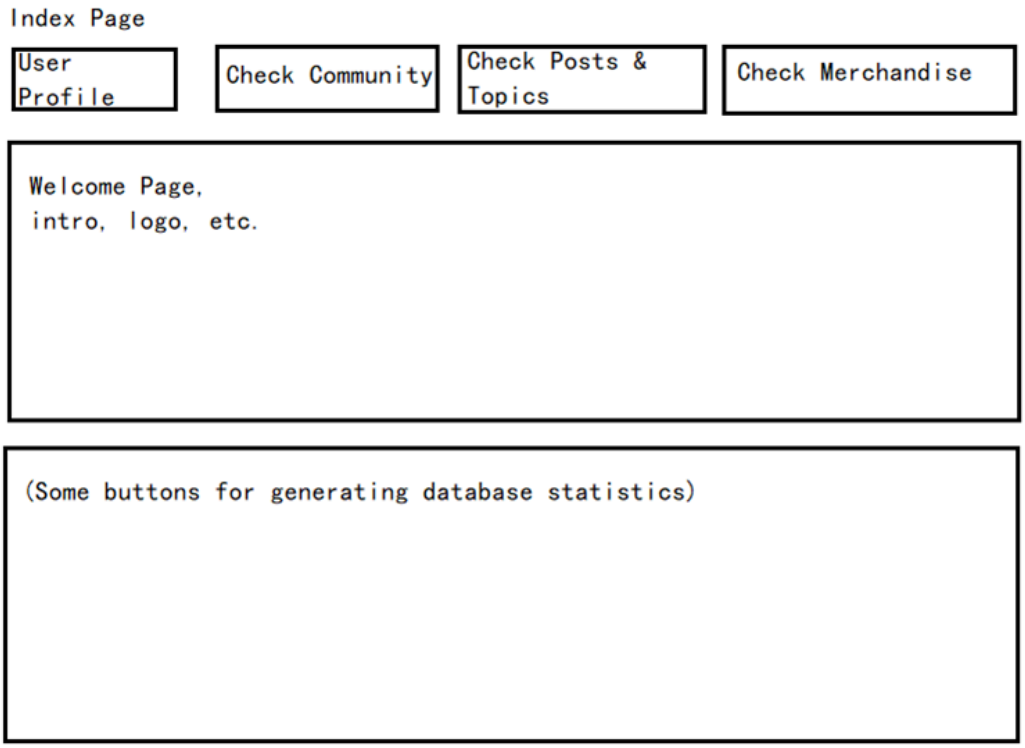
Challenges:

- Deciding the proper way to retrieve data to insert into a table from the user requires specific visual components and design knowledge.
- Making queries efficient will also be highly challenging and may require a lot of research.
- Interaction between front and backend (Compatibility Issues).
- Testing frontend components and api can be difficult and we might not have time to build a test suite with high code coverage.

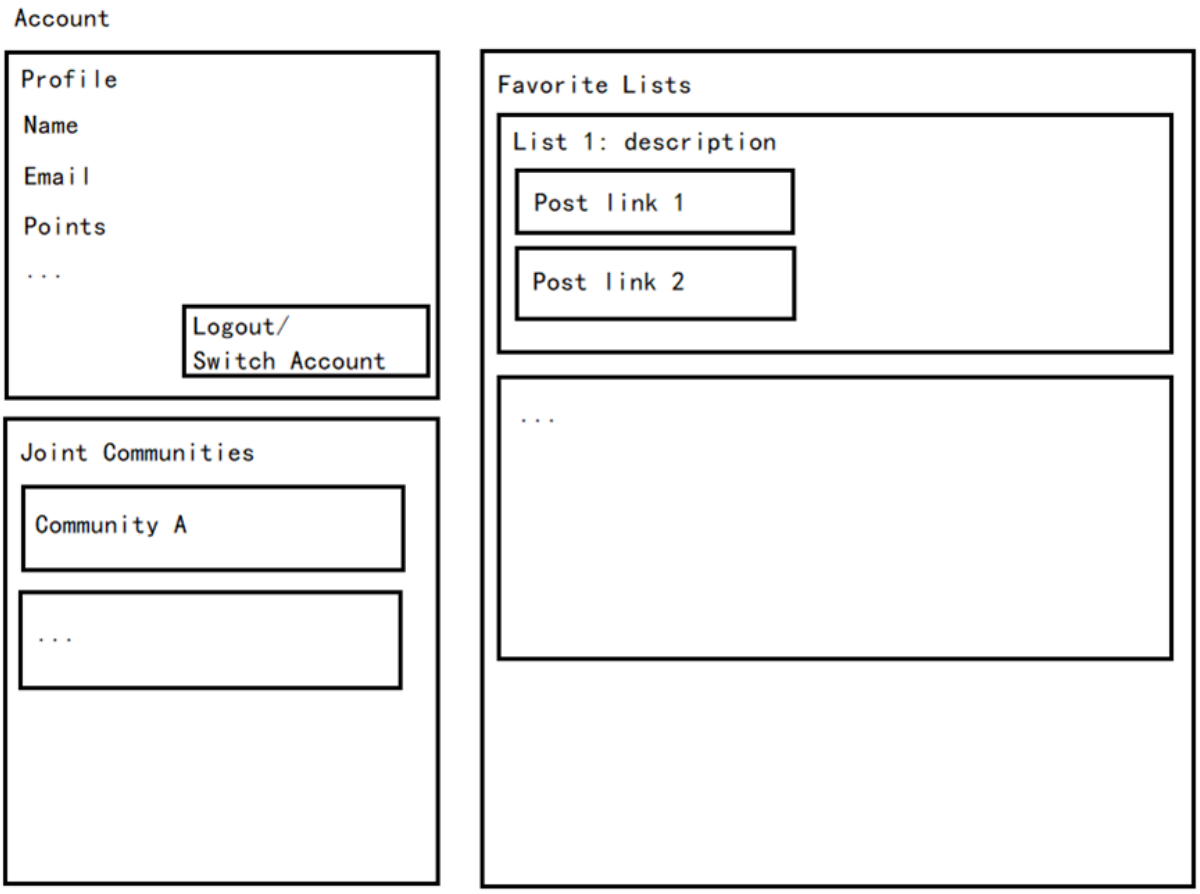
Front End Design

Index Page: home page

- Directs to different pages that display specific information from certain database;
- May involve buttons to allow users view different database statistics;
- Depending on page types, the top 4 buttons may directly lead to a page that displays the entire list of all relevant data (e.g., posts & topics), or may ask the user to select one instance and display its specific information (e.g., community);

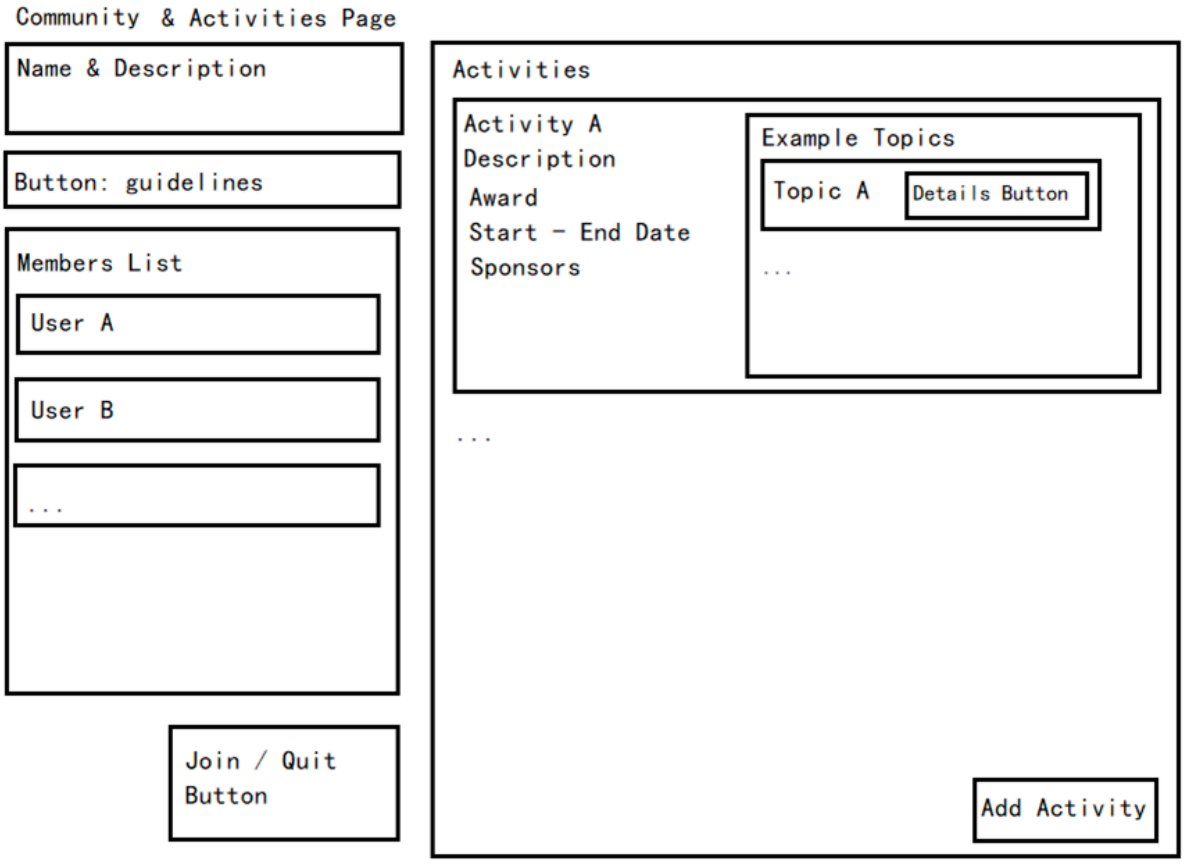


User Profile



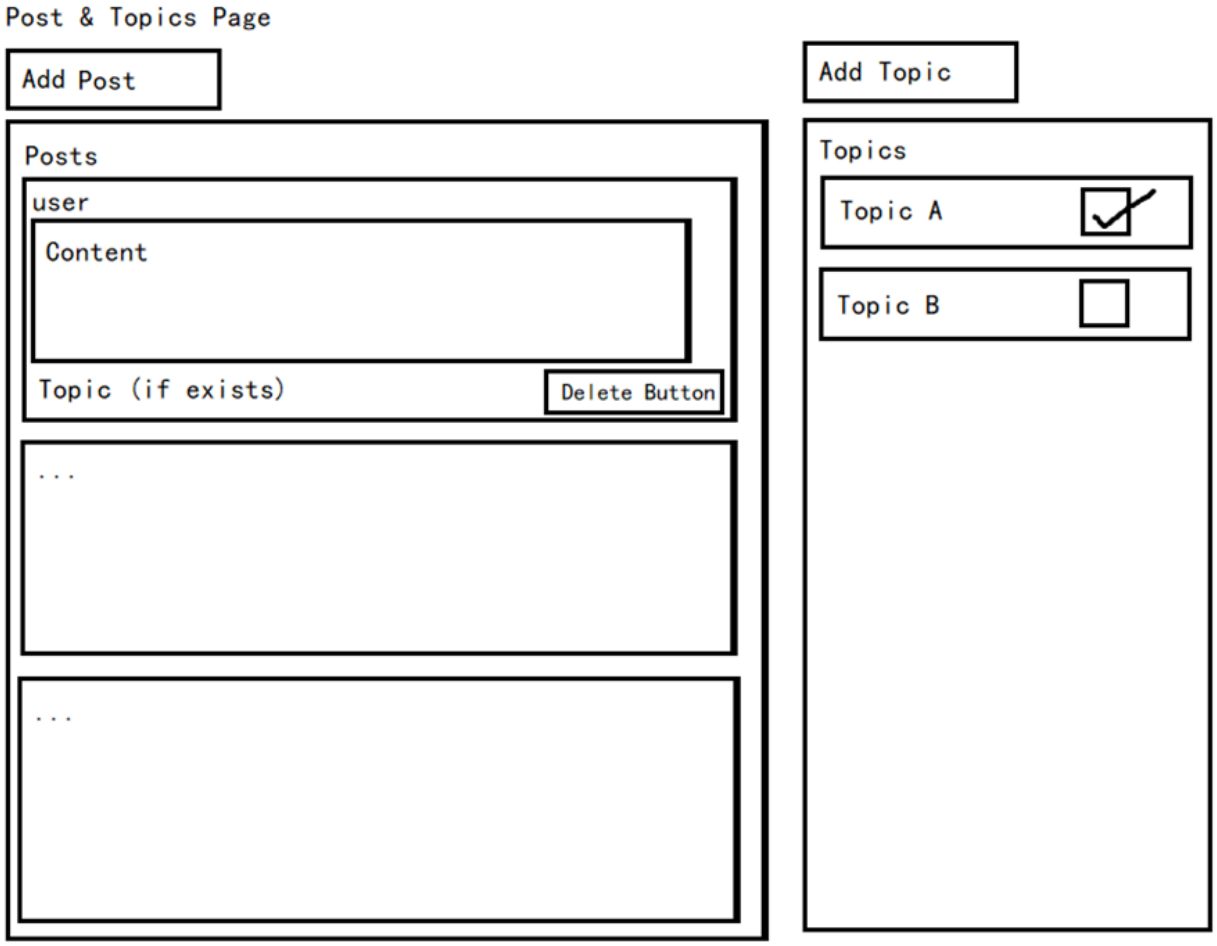
Community + Activities

- Clicking on “Details Button” for an activity will lead to the post + topics page, with the chosen topic being selected;

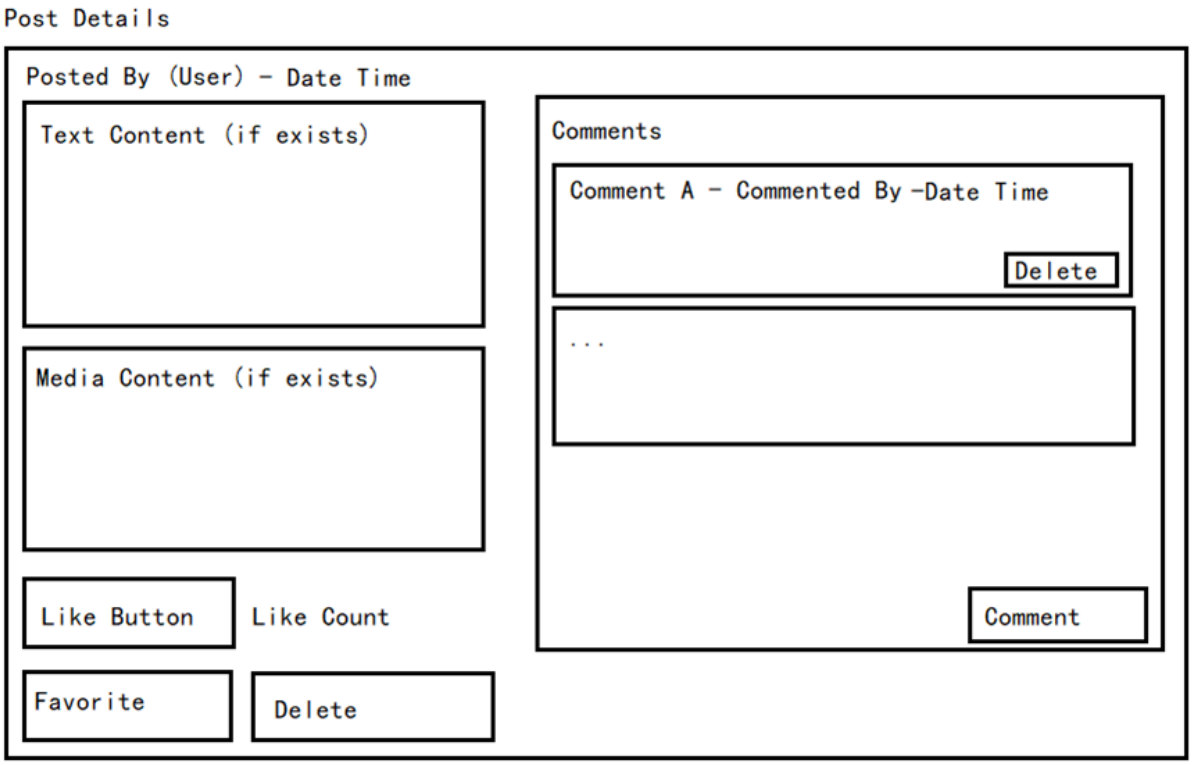


Posts + Topics

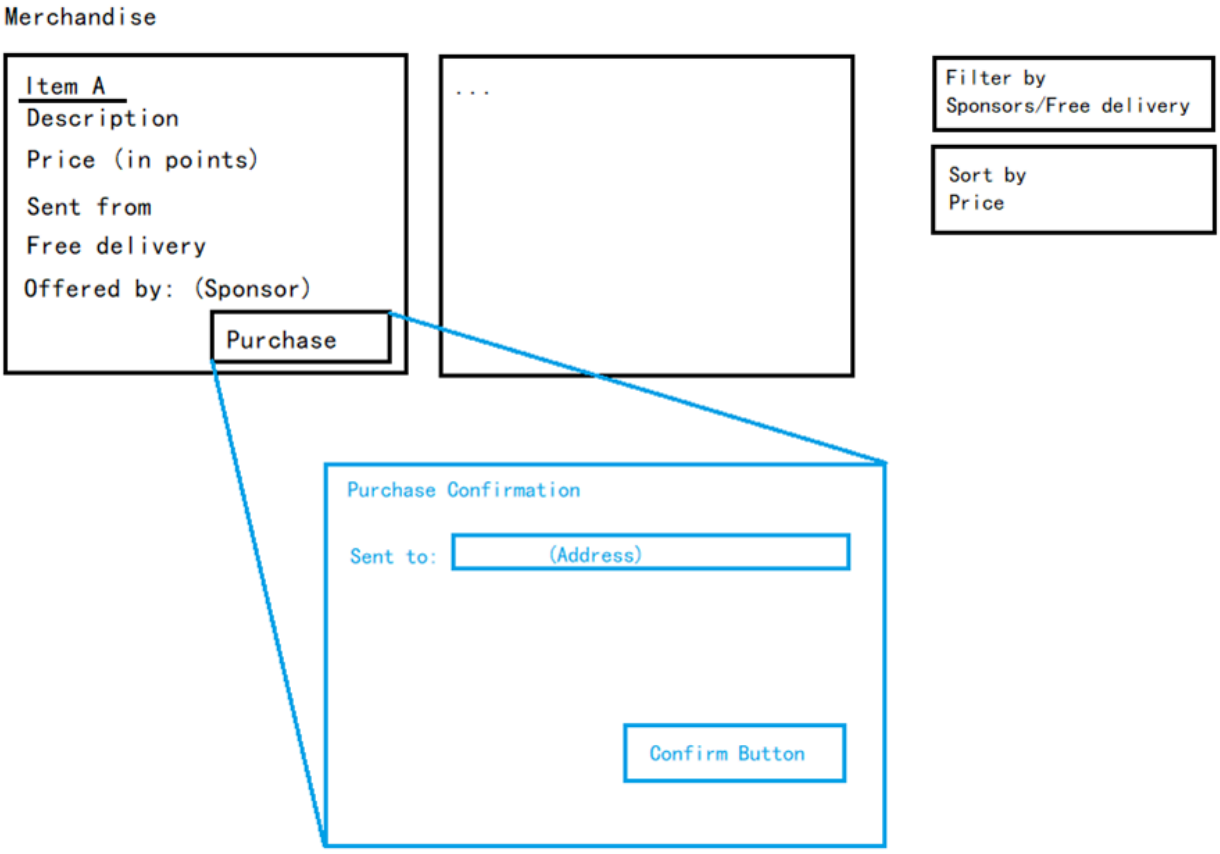
- Individual Posts are clickable; upon click, will lead to Post Detail page;
- Topics offer checkboxes to allow users view posts under specific topics;



Post Detail


Post Details
Posted By (User) - Date Time
Text Content (if exists)
Media Content (if exists)
Like Button
Like Count
Favorite
Delete
Comments
Comment A - Commented By -Date Time
Delete
...
Comment

Merchandise



Insertion Panels

There are multiple insertion panels involved in the aforementioned pages. Depending on whether the login system is implemented, the insertion panels may or may not require the user to input the credentials of the account involved in the action.