COSI 152A – Web Application Development

Final Project (CPA8)

Background:

The Brandeis University Student and Alumni Association (BUSAA) is a group dedicated to building a community between current students and alumni of Brandeis University. BUSAA is looking to develop a web application that allows its members to connect with each other, share information about job opportunities, post events and communicate through a chat feature. Additionally, BUSAA wants to display a list of upcoming events on the website and allow users to join an event through a modal.

Objectives:

- To gain hands-on experience in building a web application using the Express.js.
- To implement user authentication and authorization in an Express.js application.
- To implement CRUD (Create, Read, Update, Delete) operations.
- To implement RESTful API design principles in an Express.js application.
- To implement a real-time communication channel in an Express.js application.
- To implement database design and management in an Express.js application.
- To develop critical thinking and problem-solving skills.
- To gain practical experience working on a project that has real-world applications.
- To add a relevant project to the student's portfolio or resume.

Assignment details:

Your task is to combine all your previous Creative Project Assignments, make some necessary modifications and add some new features to build a complete BUSAA web application that meets the following requirements:

- User Authentication: The application should allow users to sign up for an account, log in, and log out. Only authenticated users should be able to access certain parts of the application, such as posting job opportunities or events.
- Job Opportunities: Authenticated users should be able to post job opportunities to the application, including information such as job title, company name, location, and a description. Limit the update operation to only the user who created the Job as well as to the Admin of the system. Limit the delete operation to the Admin only. Other users should be able to view these job postings.
- Events: Authenticated users should be able to post events to the application, including information such as event title, date and time, location, and a description. Limit the update operation to only the user who created the Event as well as to the Admin of the system. Limit the delete operation to the Admin only. Other users should be able to view these events and RSVP through the application.
- Chat Functionality: The application should have a chat feature that allows authenticated users to communicate with each other in real-time. Users should be able to join a chat, send messages to other users, and leave the chat.

- API for Events: The application should have an API that provides a list of upcoming events. The API should be secured with unique API tokens for each user and should be accessible via a URL endpoint and return the event information in JSON format.
- Event Modal: A modal should be placed on the application displaying the list of upcoming events, and a button to join an event. When a user clicks the "Join Event" button of an event in the event modal, they should be added to the list of RSVPs for that event.
- MVC Structure: The application should follow a Model-View-Controller (MVC) architecture. The model should represent the data, the view should be responsible for rendering the UI, and the controller should handle user request and interact with the model and view.
- Route Modules: The application should divide routes into different categories of route modules, such as userRoutes, jobRoutes, eventRoutes, and etc. Each module should have its own file and should export a router object.
- Flash Messages and Comments: The application should use proper flash messages on the potential success or failure of operations, and should also come with clear and explanatory comment for each code blocks.
- Other Requirements: This is a complete project and shall follow all the design requirements, code requirements and improvement guidelines given throughout the semester via different CPAs.

Note 1: Add a boolean property "isAdmin" to your User model to indicate if a user is an admin or not. In Mongoose campus, you should manually set one user as Admin of the system to test your project. `isAdmin: { type: Boolean, default: false }`

Note 2: These requirements are general guidelines and enough attention should be paid to details as well.

Grading rubric:

- User Authentication (10%)
- Job Opportunities (10%)
- Events (15%)
- Chat Functionality (10%)
- API for Events (10%)
- Event Modal and Join Functionality (15%)
- MVC Structure (10%)
- Route Modules (10%)
- Flash Messages and Comments (5)
- Other requirements (5%)

Bonus:

- Responsive Design: The application should be responsive and accessible on different screen sizes (2.5%)
- Error Handling: The application should handle errors gracefully and provide useful error messages to users (2.5%)

Assignment submission guideline:

1. Your project name is BUSAA
2. Submit it via Latte in a zip folder named yourFirstname_yourLastname_FINAL.zip

Submission deadline:

Please make sure to submit your assignment before **Thursday 12/07/2023 at 11:59 PM**