Final Project GIK2F8

The aim of this project is to demonstrate your knowledge in applying front-end and back-end programming techniques used in the course: database, Node.js, JSON, Vue/JS with Ajax calls and Bootstrap for presentation of data.

For this project you will be building a **Q&A engine**. There are three types of roles in this web app: **consumer**, **contributor,** and **super admin**.

Group work:

This project will be carried out in groups of **two**. Both group members must submit the solution separately in Learn and mention, in a comment, who they worked with.

Plagiarism:

Plagiarism involves presenting the content of someone else's work (all or part of it) as one's own work. Work can be text, code, etc. To avoid this, mention the source of information, for example, mention the link to the source in your code.

G Task: (3p)

There are a limited number of contributors pre-selected for answering questions, hence the number of contributors must be pre-determined. This provides a clear distinction between consumers and contributors.

NOTE: The different users should login with different permissions. Follow this tutorial for tips on how to manage sessions with Node.js and Express: https://codeforgeek.com/manage-session-using-node-js-express-4/

Requirements:

As a consumer I can:

Ask questions and consume information (answers).

- I should be able to login as a consumer.
- I should be able to administer my questions i.e. add, delete and update them. The data that each question has is: question title, the question text, date/time when it was created and the category that the question

falls under.
(You have to think about database design, for example: qsID which is a primary key, unique and automatically enumerated).

- I should also be able to see a list of FAQ (frequently asked questions) returned as a result of a search query (linked to category).
- I should be able to view all my questions.
- Once a contributor has added an answer to my question, I should be able to see it below my question.

As a contributor I can:
Provide information (answers) to questions.

- I should be able to login as contributor.
- I should be able to see a list of questions classified by category.
- Under each question there should be a place for me to add my answer with my username and answer text. The date/time should be appended automatically.
- I should also be able to update my answer or delete it.
- The consumer should be able to see the answer I have provided to their question.

As a super admin I can:
See a list of all questions and administer the questions and the users.

- I should be able to login as super admin.
- Under each question I should be able to see all answers related to it. For both the questions and answers I should be able to see all related information: user, text and date/time. For answers, I should be able to view up-votes/down-votes (see further functionality requirements below).
- I should be able to block a user (i.e. consumer or contributor) for misconduct. In this scenario, when that user attempts to login they should not be able to login and should read somewhere that their account has been blocked.
- I should be able to manage contributors (i.e. add contributor, delete contributor and update contributor information).

Further functionality for **contributor:**

- I should be able to label a question as duplicate.

- More than one contributor can add answers to the same question.

Further functionality for **consumer:**

- I should be able to up-vote/down-vote an answer to my question.

Requirements for end-points and the server:

All end-points in the API should return JSON-formatted data.
HTML/CSS/Frontend Javascript should be the public folder in the project structure. Test that the end-points work in Postman. Error handling should be present in all end-points. If something goes wrong with the program, the server **should not crash** and need to be restarted.

Front-end requirements:

Make a user-friendly interface that opens immediately when the server is running and http://localhost:XXX is entered in the browser. We shouldn't have to look for the html page in your folder structure.
Each type of user; the consumer and the contributor should be directed to the layout that concerns their tasks based on their permissions. The super admin should be able to navigate to all layouts.

You have the freedom to choose the design you imagine for such a Q&A engine, but make sure it’s easy to navigate and that information/actions are where a user would “expect” them to be.

Other:

Be sure to name variables, objects, methods/functions so that they reflect this task of Q&A, questions, answers etc. We don't want to see anything about, for example, products, cars job ads or other irrelevant things.

Remember to enter the source (links or what lectures) from which you got inspiration for your code.

Try unzipping and running your zipped node.js project before submitting it. After the teacher unzips the submitted zipped node.js project, it must be possible to open and run through node server.js and/or through npm run server. If not, you will automatically fail.

Submission Instructions:

Write, in a comment, who you worked with.

1. Zipped node.js project without node_modules folder, a project submitted with node_modules folder will automatically be considered as fail.

2. After the teacher unzipped the submitted zipped node.js project, it must be possible to open and run through node server.js and/or through npm run server. If not, you will automatically fail.

Good luck,

Serena and Thomas