BIOVERSE
Vibrant Living Backed by Science

Coding Exercise: Intake Questionnaire System

Objective: Create a web application that allows users to answer questions and administrators to view all the answers provided by users.

Given 3 csv files: *questionnaire_questions*, *questionnaire_junction*, *questionnaire_questionnaires*, load these csv files into a database and construct an intake system such that a user will be able to answer these intakes and record their answers as they complete it.

questionnaire_questionnaires: Holds the data for each questionnaire that exists
questionniare_junction: Provides the mapping between questionnaires and questions, with priority indicating the question order. Priority indicates the order of which questions to show in order of lowest priority -> highest.
questionnaire_questions: Contains the data for each question.

User Interface (Front-End):
Login Page:

- Function: Allows users to enter a username and password and navigate the user to the correct page accordingly.
- Requirements: Once logged in as a user, navigate the user to the questionnaire selection page. Once logged in as an admin, navigate the user to the admin panel page.
- Required Fields: Username, Password
- *You do not need to implement auth into the application. A simple username - password login may work*

Questionnaire Selection Page

- Function: Allow users to select which questionnaire they would like to complete
- Requirements: Navigate the user the respective questionnaire when clicked on.

Questionnaire Page:

- Function: Displays questions to users' and stores the user's input data into the database.
- Requirements:
    - Each question is rendered logically such that the user is able to interact and provide an answer which will be reviewed by an administrator later.
        - Upon completion of an intake, navigate the user back to the questionnaire selection page.
    - The only input validation you need to make is for inputs to not be empty & not accept whitespace.
    - If a question has been previously answered on another questionnaire, pre-populate the answer for the question.

    - Questions with **Select all that apply** must record all answers that the user selects.

Admin Panel:

- Function: Enables administrators to view a different users' answers organized
- Features: A table of usernames and how many questionnaires they have completed. Administrators can click into the row and a modal opens displaying all the answered questionnaires for the user.
- For displaying questions, show the username, questionnaire name, then followed by the questions/answers in a "Q: … A: …" format.

General Requirements

- How you go about implementing the architecture for saving answers for each user is entirely up to you.
- You can create more DB tables as you see fit.
- No specific design pattern must be used. You can utilize different UI libraries as necessary.

Technical Requirements:

- **Front-End:** Preferably implemented using React (NextJS).
- **Back-End:** Language/Framework of your choice. Preferred PostgreSQL.
- **Deployment:** App should be deployed online (e.g., Vercel or similar platform).
- **UI appearance:** We don't expect a completely designed application, but it should be neat and easy to use.
- **User Authentication:** You don't need to properly set up authentication. A login form with hardcoded logins is sufficient.

Time and Submission Guidelines:

- **Deadline:** Please submit your work within **five** days of receiving this prompt. We're looking for engineers who excel in rapid development and iterative problem-solving to help us quickly build our MVP and adapt based on user feedback before the full launch.
- **Submission Format:** Complete exercise to be submitted as a GitHub repository URL.
- **Submission Recipients:** Send the GitHub URL and the deployment links to Shanti (shantibraford@gmail.com) and Lara (Lara@gobioverse.) and the REVELO Team

If you have any questions, do **not** hesitate to reach out to us.