Heliverse

Full Stack Development
Assignment Description:

Your task is to create a full-stack web application that allows users to view and interact with a list of users. The mock data for the users can be found at this link

https://drive.google.com/file/d/1ibmr3WD7Jw6oLL6O_W390WojCLfCHw-k/view?usp=sharing

The application should have the following functionalities:

Frontend:

1. **Display users in cards format with pagination**: The users should be displayed in a visually appealing card format. Implement pagination to display 20 users per page.

2. **Search by Name**: Users should be able to search for users by their names. As the user types in the search input, the list of displayed users should be dynamically updated to match the search query.

3. **Add 3 filters:** Implement three filters - Domain, Gender, and Availability. Users should be able to select multiple filters simultaneously, and the displayed user list should be updated accordingly.

4. **Create a team**: Users should be able to create a team by selecting users from the list. Only users with unique domains and availability should be selectable for the team (similar to adding items to a cart in e-commerce websites).

5. **Show team details**: Once the team is created, display the details of the team, including the selected users' information.

6. **Make it responsive**: Ensure that the application is responsive and displays properly on different screen sizes.

Backend:

Create CRUD API: Implement CRUD (Create, Read, Update, Delete) operations for managing the user data.

GET /api/users: Retrieve all users with pagination support.
GET /api/users/:id: Retrieve a specific user by ID.
POST /api/users: Create a new user.
PUT /api/users/:id: Update an existing user.

DELETE /api/users/:id: Delete a user.
Implement filtering, searching, and pagination on the backend:

Implement filtering logic to filter users based on the provided filter criteria (Domain, Gender, and Availability).

- Implement searching logic to search for users by their names.
- Implement pagination logic to retrieve a specific number of users per page.
- Create API endpoints for team-related operations:

POST /api/team: Create a new team by selecting users from the list with unique domains and availability.
GET /api/team/:id: Retrieve the details of a specific team by ID.

Tech Stack:

Frontend:

React.js for the UI components
Redux Toolkit for state management

For Styling use of libraries such as Tailwind, Material UI or Bootstrap is not restricted.

Backend:

Node.js and Express.js for the server or Nest js
MongoDB for the database
Mongoose for object modeling with MongoDB

Deliverables:

Share the GitHub repository containing the code for both the frontend and backend. (**Repo Must be Private: Share Access on shreays@heliverse.com or anmol@heliverse.com**)
Provide the **deployed links** for both the frontend and backend. (**IMPORTANT)**

Contact Information:

For any concerns or questions, you can reach out via email at
anmol@heliverse.com or hr@heliverse.com

Note: This is a test assignment, and the purpose is to assess your ability to create a full-stack web application using the specified tech stack and requirements.