Frontend Developer Test Case

Edixir

December 5, 2024

Objective

The purpose of this test case is to evaluate the candidate's ability to:

- Work with APIs and implement secure session management.
- Follow frontend security best practices.
- Demonstrate advanced knowledge of Object-Oriented Programming (OOP) and SOLID principles.
- Use Git effectively for version control.

Scenario

You are tasked with creating a **User Dashboard** for an admin system with the following features:

1. **Login and Session Management**:
   - Users log in using a JSON Web Token (JWT)-based system.
   - Tokens must be securely stored and refreshed when they expire.
2. **User List**:
   - Display a paginated list of users fetched from the API.
   - Allow filtering users by role (`Admin`, `Editor`, `Viewer`).
3. **User Management**:
   - Allow creating and editing users via a reusable modal component.

API Endpoints

The following API endpoints are available:

- `POST /api/login`: Authenticate and return a JWT and refresh token.
- `GET /api/users?page=<page>&role=<role>`: Fetch paginated user data filtered by role.

- `POST /api/users`: Create a new user.
- `PUT /api/users/:id`: Update an existing user's details.

Requirements

Frontend Architecture

- Use **OOP and SOLID principles** to design services:
  - **AuthService**: Handle login, token storage, and refreshing.
  - **UserService**: Abstract API calls for user data.
- Demonstrate **dependency injection** for service classes.

UI Components

- Create a reusable **PaginatedTable** to display the user list.
- Implement a **UserModal** component for creating and editing users.

Session and Security

- Implement secure token storage (e.g., in-memory or `httpOnly` cookies).
- Redirect users to the login page if their session expires.

Bonus Challenge

1. Write unit tests for service classes.
2. Add role-based access to hide certain user actions for non-admin roles.