# QA Automation Take-Home Exercise

## Automated Testing for Authentication and User Management

Welcome to the exercise! This task is designed to assess your skills in automated testing, test planning, and professional software development workflows from the ground up. We are looking for clean, professional code and adherence to best practices.

## App Details

- **Application URL:** http://cova-dev.vulnapp.io:8080
- **Credentials:** Will be provided separately via email/direct message.
    - **Email:** [Provided Separately]
    - **Password:** [Provided Separately]

## Initial Setup & Git Workflow

Since you received these files directly (via Google Drive, email, etc.), your first step is to establish your version control repository on GitHub.

1. **Create a New GitHub Repository:** Create a new **public** repository on your personal GitHub account for this exercise (e.g., snyk-qa-automation).
2. **Initialize Git Locally:** Navigate to the folder containing these files (the ones you received) and initialize a local Git repository.

```
None
git init
git branch -M main
```

3. **Initial Commit:** Add this README.md and any initial configuration files, and make your first commit.

```
None
git add .
```

```
git commit -m "Initial commit: Setup files and README"
```

4. **Push to GitHub:** Link your local repository to the new remote GitHub repository you created in Step 1.

```
None
git remote add origin <YOUR_GITHUB_REPO_URL>
git push -u origin main
```

# Task Requirements

You are required to develop automated End-to-End (E2E) tests for two primary functional areas: **Login** and **User Listing/Search**.

## Technology Stack

- **Mandatory Language:** JavaScript
- **Optional Language:** TypeScript *(encouraged, but not mandatory)*
- **Testing Framework:** Use any modern, industry-standard automated testing framework of your choice (e.g., Playwright, Cypress (*preferred*), WebdriverIO, etc.).

## Planning & Implementation

1. **Test Planning:**
    - Create a brief, well-structured **Test Plan** document as a `TEST_PLAN.md` file. This document must outline:
        - Your chosen framework and rationale.
        - High-level test strategy (e.g., Page Object Model, setup/teardown).
        - A breakdown of the specific test cases you plan to implement for each of the two tasks below.
2. **Implementation & Version Control:**
    - Create a separate branch for each task (e.g., `feature/login-tests` and `feature/user-tests`).
    - Commit your code.

## Submission

1. **Pull Requests (PRs):** For each task, **open a Pull Request (PR)** from its feature branch, targeting the **main** branch of your repository.
2. **Notification:** Once both PRs are open and ready for review, **notify your Snyk Talent Partner via email or the specified communication channel** to initiate the review.
3. **Final Step:** Do not merge your own PRs. We will review the code and discuss it with you.

## BONUS Challenge: Critical QA Eye

Please include a dedicated section in your TEST_PLAN.md (or a separate NOTES.md file) that addresses the following:

- **Identify at least three areas** within the provided application's front-end structure that could be improved to make building stable and maintainable automated tests easier for a QA team in the future.

## Task 1: Login Authentication Automated Tests

**Title:** [Automation] Develop Automated Tests for Login Functionality

**Focus:** Establish a reliable automation foundation by testing the application's authentication flow.

**Acceptance Criteria (ACs):**

1. **Successful Login:**
    - Test successful login using the provided valid credentials.
    - Verify that the user is redirected to the expected landing page (e.g., the user list page).
2. **Invalid Credential Handling:**
    - Test login attempts with an **invalid password** for a valid email.
    - Test login attempts with an **unregistered/invalid email** format.
    - Verify that an appropriate **error message** is displayed to the user for each invalid attempt.
3. **Form Validation:**
    - Test attempting to log in with **empty** email and password fields.
    - Verify that form validation messages or appropriate prompts are shown.

## Task 2: User List and Search Automated Tests

**Title:** [Automation] Develop Automated Tests for User Listing and Search Functionality

**Focus:** Test core user management features, including dynamic list interaction and data verification.

**Acceptance Criteria (ACs):**

1. **User List Display:**
    - Test that the user list page loads correctly after a successful login.
    - Verify that at least one user is present and displayed in the list.
    - Verify that the expected user details (Name, Email, Role) are displayed correctly for a sample user.
2. **Search Functionality - Valid Input:**
    - Test the search feature by searching for a **full, existing user name**.
    - Verify that the list is filtered to show **only** the matching user(s).
3. **Search Functionality - Partial Input:**
    - Test the search feature using a **partial name**.
    - Verify that all users whose names contain the partial input are displayed.
4. **Search Functionality - No Results:**
    - Test the search feature with a **non-existent/random name**.
    - Verify that an appropriate **"no results found"** message or state is displayed, and the user list is empty.