Coding Test for QA Automation Engineers

Goal:

Create a simple automation framework for testing the basic functionalities of the OrangeHRM demo web app. The test should cover login, user/candidate creation, created user/candidate details verification, view and verify project time report.

Tools/Technologies:

Cypress
Programming Language: JS

Test Steps:

Setup:

Create a new Cypress project
Include the necessary dependencies
Set up a basic project structure.

Create a Test Script for Login:

Navigate to the OrangeHRM demo web app (use the publicly available OrangeHRM demo instance: https://opensource-demo.orangehrmlive.com/).
Write a test script to login to the application.
Use a data-driven approach (e.g., store login credentials in a configuration file or constants).

Create a Test Script for Adding a New User/Candidate :

After successful login, navigate to the "recruitment" or “admin” section.
Write a test script to add a new candidate or user based on which section you navigate to with sample data (e.g., first name, last name).
Verify that the employee is added successfully.

Create a Test Script for Verifying User/Candidate Details:

Write a test script to navigate to the "recruitment" or “admin” section.
Verify that the details for the added user or candidate match the data provided during the creation of the user or candidate.

Create a Test Script to View and Verify Project Time Report:

After successful login, navigate to the "Time reports"" section.
Search query for “Apache Software Foundation - ASF - Phase 1”
Also include time frame of Jan 1 2021 to Dec 31 2023 in you query
Assert the following activities are listed:

- Bug fixes
- Feature Development
- Implementation
- QA Testing
- Requirement Gathering
- Support & Maintenance

Assertions and Reporting:

Include assertions to validate the expected outcomes at each step.
Implement reporting (e.g., using TestNG or any other reporting tool).
Ensure that the test reports provide clear information about the test results.

Test Data Management:

Use external files (e.g., Excel, JSON, or properties file) to manage test data.
Implement data-driven testing for different scenarios (e.g., test with multiple sets employee data).

Clean-Up:

Implement a clean-up mechanism to delete the test data created during the test (e.g., delete the employee created during the test).

Submission Guidelines:

Share the source code via a GitHub repository.
Please include all dependency files
Include a README.md file with instructions on how to run the tests.
Provide a brief explanation of the framework structure and design choices.

Evaluation Criteria:

Code organization and structure.
Effective use of assertions for validation.
Appropriate use of data-driven testing.
Clean and clear reporting.
Handling of exceptions and errors.