Senior Automation QA Assessment

Overview

This assessment evaluates your technical proficiency and automation skills across mobile application testing, API testing, test automation design, and scripting. Please complete the following tasks, ensuring clear documentation and adherence to best practices. Try to avoid using AI tools. If used mark 'where'.

Task 1: Mobile Application Testing

Objective

Using the provided Android APK (a simple calculator app), develop end-to-end (E2E) tests that:

- Launch the application.
- Execute various mathematical operations (e.g., addition, subtraction, multiplication, division).
- Verify the accuracy of the computed results (the app has a bug).

Requirements

- **Tools:** Python, ADB, UIAutomator.
- **Implementation:**
  - Construct well-structured E2E tests.
  - Integrate relevant test design patterns.
  - Include comprehensive comments, docstrings, type annotations, and consistent formatting.

Task 2: API Testing

Objective

Using the provided Python Flask mock API, develop automated tests for the following functionalities:

- **GET /users:** Retrieve a list of users.
- **POST /users:** Add a new user. The request expects JSON in the format: `{"name": ..., "email": ...}`. Implement robust error handling and reporting for invalid input.

- **GET /users/<id>:** Retrieve a user by ID. Handle the following responses:
  - **404:** User not found.
  - **500:** Intentional error for testing purposes (when `ID = 999`).

Requirements

- **Tools:** Python, Requests library.
- **Tests:** Implement three distinct automated tests:
  1. **GET Test:** Validate user data retrieval.
  2. **POST Test:** Validate the creation of new users.
  3. **Error Handling Test:** Validate expected error responses (e.g., 400 or 500 status codes).
- Well structured code as before

Task 3: Gherkin Scenario Creation

Objective

Create a single test scenario in Gherkin + Behave format. This scenario should encapsulate either your mobile or API tests (ideally, both).**Example**

None

```
Feature: Calculator functionality

  Scenario: Validating mathematical operations

    Given the calculator app is opened

    When I multiply 3 and 4

    Then I should see the incorrect result
```

Task 4: Automation Test Runner Script

Objective

Create a `run_all_tests.sh` shell script to:

- Execute all tests developed in Tasks 1–3.

- Collect relevant logs for debugging (e.g., `adb logcat`, `adb shell dumpsys`).
- Save test outputs for subsequent analysis.

Requirements

- Ensure the script:
  - Executes smoothly.
  - Generates logs in a clear and organized manner.
  - Facilitates detailed analysis of test results.

Task 5: Repository Setup

Objective

Set up a GitHub repository containing all assignment deliverables. This should include:

1. **Code:** All source files for the completed tasks.
2. **Setup & Run Instructions:**
   - A guide for setting up the environment.
   - Instructions for running individual tests or all tests collectively.
3. **Test Descriptions:** Explain:
   - The purpose of each test validation.
   - The significance of each validation.
4. **Environment Details:** Document the Python version, tools used, and any dependencies. Using `pipenv` for environment management is encouraged but optional.
5. **Extras (Optional):**
   - Include screenshots for context or debugging as needed.
   - Add notes regarding potential improvements, identified risks, or insights.

Submission

- **Deadline:** 3–5 business days from receipt.
- **Deliverable:** Submit a link to your GitHub repository containing all required files.

Key Evaluation Criteria

- Code quality, organization, and readability (comments, docstrings, type annotations).
- Effectiveness of test design and coverage of core functionalities.
- Adherence to best practices in automation and documentation.
- Demonstrated thoughtfulness in identifying additional improvements or risks.