Test Plan / Test Strategy

This test plan describes the end-to-end strategy for automated UI and API testing using Selenium and REST-assured, intended for use with modern sample platforms such as SauceDemo and JSONPlaceholder. The document covers scope, coverage, tools, execution, and key project constraints.

1. What Is Being Tested

- **UI Functionalities (SauceDemo)**
    - User login (valid and invalid credentials)
    - Adding/removing products from the cart
    - Proceeding through checkout
    - Verification of expected UI state after key user actions
- **API Functionalities (JSONPlaceholder)**
    - Retrieval of data with `GET` (e.g., `/posts`)
    - Resource creation with `POST` (e.g., `/posts`)
    - Resource update with `PUT` (e.g., `/posts/1`)
    - Resource deletion with `DELETE` (e.g., `/posts/1`)
    - Input validation, 404 handling, and negative response scenarios

2. Test Coverage Areas

- **UI**
    - Positive and negative login scenarios
    - Cart add/remove flows
    - Checkout path and field validation
    - Visibility and correctness of UI elements after operations
- **API**
    - CRUD operations on `/posts`
    - Handling for invalid or non-existent resources
    - Response schema and status code validation

3. Tools Used and Why

Tool	Purpose	Reason for Use
Selenium (Java)	UI/browser automation	Widely supported, simulates user browser interactions
REST-assured	API test automation	Java-native, expressive and powerful for HTTP assertions
TestNG	Test framework	Supports structured tests, dependencies, reporting
Maven	Build/dep. management	Industry standard in Java for builds and dependencies
WebDriverManager	Driver management	Auto-downloads required browser drivers
GitHub Actions	CI / cloud test runs	Automates tests and feedback on every code push or PR

4. How to Run the Tests

- **Prerequisites:** Java 17+, Maven, Firefox browser (for Selenium UI tests)
- **Clone the repository:**

```
git clone https://github.com/Angro-Mustafa/UI-API-Test-Automation.git
cd UI-API-Test-Automation
```

- **Run UI Tests (SauceDemo):**

```
mvn test -Dtest=SauceDemoUITests
```

- **Run API Tests (JSONPlaceholder):**

```
mvn test -Dtest=JsonPlaceholderApiTests
```

- **Run All Tests:**

```
mvn clean test
```

- **Continuous Integration:**
  CI/CD setup (e.g., GitHub Actions) automatically triggers UI and API test suites and archives artifacts on every push and pull request.

5. Assumptions and Limitations

- *Test data is not persistent* on public demo platforms; UI or API state may reset or differ on each run.
- UI element IDs and selectors may change if the underlying demo site is updated; locator review is required if tests begin to fail.
- API endpoints and behavior (especially on JSONPlaceholder) are designed as mocks and may not represent transactional data storage.
- Public systems may rate-limit users or briefly block repeat feature usage.
- Robustness of test results relies on the stability of the public demo instances.
- Scope is intentionally limited to key business and data flows. Additional coverage (e.g., performance, accessibility) is considered out of scope unless otherwise specified.

This test plan ensures a maintainable, comprehensive, and repeatable QA baseline for both automated UI and API flows on standard demonstration platforms, covering positive and negative real-world scenarios with modern Java-based tooling.