### **Test Plan**

#### **1. Objectives**

To validate the application for its intended functionalities, ensure its robustness under different conditions, and verify all features work seamlessly with no critical defects.

#### **2. Scope**

Covers user actions such as:

* **Sign Up**: Create an account.
* **Sign In**: Log in with valid credentials.
* **Onboarding Nodes to an Existing Blockchain**: Add nodes and wallets, review and submit.
* **Create New Private Blockchain**: Provide network details, add nodes, review, and submit.
* **Sign Out**: Ensure restricted access post-logout.

### **Testing Types**

#### **1. Functional Testing**

**Purpose**: Ensures each feature operates as per the functional requirements.

* **Manual Testing**:
  + Verify user actions like Sign Up, Sign In, and node onboarding work correctly.
  + Validate workflows for edge cases and input errors.
* **Automated Testing**:
  + Write test scripts for repetitive tasks like user authentication (Sign Up, Sign In, Sign Out).

#### **2. Boundary Testing**

**Purpose**: Tests input fields for constraints like valid formats, lengths, and boundary values.

* Example: Test valid and invalid Node ID, Wallet Address, and IP Address.

#### **3. Negative Testing**

**Purpose**: Checks how the application handles invalid, incomplete, or unexpected inputs.

* Example: Attempt submissions with empty fields or duplicate entries.

#### **4. Performance Testing**

**Purpose**: Measures app responsiveness under high-load conditions like adding multiple nodes and wallets.

* Use tools like JMeter to simulate heavy usage.

#### **5. Security Testing**

**Purpose**: Verifies sensitive data protection and system vulnerability against attacks.

* Example: Ensure proper encryption for passwords and wallet addresses.

#### **6. UI/UX Testing**

**Purpose**: Evaluates the usability, design consistency, and responsiveness of the application across devices and browsers.

* Ensure error messages are user-friendly and inputs are properly validated.

#### **7. Regression Testing**

**Purpose**: Confirms that new updates haven’t introduced bugs or broken existing features.

* Automated tools can be used here for quick validation after updates.

### **Test Cases**

#### **A. Sign Up**

|  |  |  |
| --- | --- | --- |
| **Test Case** | **Input** | **Expected Result** |
| Sign Up with valid data | Valid email, valid passwords | Account created successfully |
| Invalid email format | "test@" or "email@test" | Error: "Invalid email format" |
| Password mismatch | "password123" and "password456" | Error: "Passwords do not match" |
| Already registered email | Existing email | Error: "Email already exists" |

#### **B. Sign In**

|  |  |  |
| --- | --- | --- |
| **Test Case** | **Input** | **Expected Result** |
| Sign In with valid credentials | Registered email and valid password | Logged in successfully |
| Invalid email format | "user@" or "test" | Error: "Invalid email format" |
| Incorrect password | Correct email, wrong password | Error: "Invalid credentials" |
| Unregistered email | Email not in database | Error: "Account not found" |

#### **C. Onboard Nodes to Blockchain**

|  |  |  |
| --- | --- | --- |
| **Test Case** | **Input** | **Expected Result** |
| Add valid node details | NodeID-123, 192.168.0.1 | Node added successfully |
| Invalid Node ID format | "123-Node", "node123" | Error: "Invalid Node ID format" |
| Invalid IP format | "999.999.999.999" | Error: "Invalid IP Address" |
| Add duplicate node | Same Node ID repeated | Error: "Duplicate node detected" |

#### **D. Wallet Address**

|  |  |  |
| --- | --- | --- |
| **Test Case** | **Input** | **Expected Result** |
| Add valid wallet address | Valid 0x{checksum hexadecimal} | Wallet added successfully |
| Invalid wallet format | "wallet123" or "12345" | Error: "Invalid wallet address" |

#### **E. Create Private Blockchain**

|  |  |  |
| --- | --- | --- |
| **Test Case** | **Input** | **Expected Result** |
| Enter valid network details | Network Name and valid wallet | Details saved successfully |
| Enter duplicate network name | Same name as an existing network | Error: "Network name already exists" |

#### **F. Sign Out**

|  |  |  |
| --- | --- | --- |
| **Test Case** | **Input** | **Expected Result** |
| Successful Sign Out | Logged in user | Redirected to Sign In page |
| Access restricted resources | Try accessing dashboard post-logout | Error: "Access denied, please log in" |

### **Automated Testing Plan**

Automate repetitive and critical scenarios using **TestNG** and **Selenium WebDriver**.

* Sign Up: Validate form submission and error handling.
* Sign In: Test positive and negative login scenarios.
* Sign Out: Ensure restricted resource access after logout.

### **Performance Testing Plan**

Using **JMeter**:

* Simulate 100+ concurrent users onboarding nodes and wallets.
* Measure the application’s response time for submission.

### **Security Testing Plan**

1. Verify encrypted data transmission for sensitive information (e.g., passwords).
2. Perform SQL injection tests to validate server security.