Test Plan for Yoroi Wallet Extension testing:

### 1. Analyze the product or feature you’re testing

* Talk with the designer and developer to understand the scope, objectives, and functionality of the site.
* Review the project documentation (such as your SOW, project proposal, or even the tasks in your project management tool).
* Perform a product walkthrough to understand the functionality, user flow, and limitations.

This step is what gives you the context to write your test plan introduction and objectives and start to plan out the resources you’ll need to complete it.

### 2. Design the test strategies (and approach) you’re going to use

To decide the scope of your test plan. What’s included in the scope of your testing will depend on a number of factors beyond just the product or feature.

* Customer requirement: What are your users going to use most?
* Budget and timeline: How much time and resources do you have to complete testing?
* Product specs: What are the most important parts of this feature that need to be tested?
* Team abilities: Do you have the technical expertise you need to complete each test?

For our website redesign example, we might want to say that functionality, UX, and checkout flow are in scope. While stress, performance, and database testing are out of scope.

You might also want to think of this in terms of commonly used testing approaches, such as:

* Unit testing: Test the smallest piece of software or a specific feature.
* API testing: Test the API created for the application in multiple scenarios.
* Integration testing: Test multiple software modules or features as a group.
* System testing: Test the entire integrated system against its requirements.
* Install testing: Test the install/uninstall process your customers will go through.
* Compatibility testing: Test your software on different hardware, operating systems, and environments.
* Load and stress testing: Test your software performance as the workload increases (or goes beyond normal conditions).

Deciding what to test and documenting your test strategy are the most critical parts of your test plan. Don’t rush through it. Take the time to really understand your goals and needs and balance them against the resources you have for testing.

### Define the test objectives and Identify testing Types.

As you define each different test you’re going to run, you need to know when your test is “done.” This means defining the pass and fail criteria for each specific test, as well as some of the things we mentioned above, such as exit and suspension criteria.

To do this, you’ll want to identify individual system metrics that you’re checking and decide what success means for each one. For example, if you were doing a performance test you might look at metrics such as:

**Types of test cases to verify the application:**

* Add extension button should be displayed.
* Add extension button should be clickable.
* Extension should not be allowed to install on incognito mode.
* Added extension should be displayed on the extensions list.
* Select the language
* By default English language should be displayed.
* List fo Drop down for language should be displayed.
* Click on Continue button should redirect to Termas of Service agreement page.
* Check box for accepting the terms should be displayed and continue button should get enable when check box is checked.
* Payments page should be displayed with Skip and Allow button.
* Check Create Wallet option si dispayed and user selects the wallet.
* Check that 2 options for currency is displayed.
* Verify if user selects the cardono option from the list of currency.
* Verify to check 2 options for wallet creation should be dispayed.
* Verify if user selects create Wallet option from the list.
* Verify if user clicks on Learn More button.
* Verify if user click on Close icon.
* Verify if users wants to create the wallet.
* Verify The Wallet name should be mandatory.
* Verify to check for the Wallet name validation are applied(Blanks, Numbers should not be allowed,Special char, not more than 40 chars..etc)
* Verify to check password is in Encrypted format.
* Verify both password should be same.
* Verify if the data in password field is either visible as asterisk or bullet signs.
* Verify the create wallet page for both, when the field is blank and Create button is clicked.
* Verify that all the fields such as Walletname, Password has a valid placeholder
* Verify that the application’s user interface (UI) is responsive, so it will adapt to different screen resolutions and devices.
* Verify password should be more than 10 characters
* Verify right checkmark is displayed when both password are in same and in accepted format
* Verify to check for the recovery Phase
* Verify to check for correct recovery format.
* Verify invalid recovery format should not be accepted.
* Verify all the words in the recovery are displayed for reconfirming the recovery.
* Verify the confirmation checkbox are displayed after recovery.
* Verify the Dashboard is displayed for the User.
* Verify the Response time: Total time to send a request and get a response.
* Verify the Wait time: How long it takes to receive the first byte after a request is sent.
* Verify the Average load time: Average amount of time it takes to deliver every request.
* Verify the Peak response time: The longest amount of time it takes to fulfill a request.
* Verify the Requests per second: How many requests can be handled.
* Verify the Transactions passed/failed: The total number of successful or unsuccessful requests.
* Verify the Memory utilization: How much memory is needed to process the request.

### 4. Plan the test environment

The results of your test plan will depend as much on the feature you’re testing as the environment you’re testing it in. As part of the scope, you need to determine what hardware, software, operating system, and device combinations you’re going to test.

### 5.Execute your test plan and track progress in your project management tool.

Once your test plan is in place, there’s a specific process you need to follow. Think of this as the Software Testing Life Cycle (STLC). Similar to the Software Development Life Cycle, the STLC follows each phase of testing and usually looks something like this:

* Requirements/Design review
* Test planning
* Test designing
* Test environment setup
* Test execution
* Test reporting

### Identify Testing Type:

### Schedule & Estimation:

### Test Deliverables:

### Roles and Responsibilities:

### Test Execution Report: