Test Plan
For Scopex_Money version 0.1

#	Date	Description	Version	Created/ Changed By	Reviewed By	Approved By
1	04-Feb-25	Initial Draft	0.1	Shivraj Kamate	Manager/Team lead	QA Manager
2						
3						

Table of Contents

1. Purpose of Document

To validate the functionality and usability of the Scopex Money website, ensuring it meets specified requirements and provides a seamless user experience.

2. Project Background

Scopex Money is a modern money transfer platform that leverages blockchain technology to provide faster, more transparent, and secure transfers compared to traditional systems. Currently, it operates in the Eurozone and India, with plans for future expansion.

3. References

Reference documents used while defining test plans.

- Scopex Official Website
- Scopex on Google Play Store
- Scopex on the App Store
- Scopex Facebook Page
- Scopex Instagram Profile

4. Definitions of Terms / Abbreviations

#	Terms / Abbreviations	Description
	QA	Quality Assurance
	UAT	User Acceptance Testing
	CR	Change Request
	ER	Enhancement Request

5. Testing Objectives

The primary objectives of this testing initiative for Scopex Money are:

1. **User Registration**: Ensure that new users can successfully create accounts by providing accurate personal information, leading to a seamless transition to the dashboard.
2. **Adding a Recipient**: Verify that users can add recipients by entering the necessary details, with the new recipient appearing correctly in the recipient list.
3. **Logout Functionality**: Confirm that users can log out of their accounts, ensuring they are redirected to the login screen and that their session is securely terminated.

6. Scope of Testing

- **In Scope -** Scope defines the features, Functional and non-functional requirements of the application that will be tested.
  Following features will be developed as part of functional requirement.

    - **User Registration**:
        - Test the registration process with valid and invalid inputs.
        - Verify that the system enforces password strength requirements.
        - Ensure that the system handles duplicate registrations appropriately.
    - **Adding a Recipient:**
        - Test adding recipients with valid and invalid details.
        - Verify that the system handles duplicate recipient entries correctly.
    - **Logout Functionality:**
        - Test the logout process to ensure users are redirected to the login screen.
        - Verify that session data is cleared upon logout.

- **Usability Testing**:

    - Assess the user interface for ease of navigation.
    - Ensure that error messages are clear and helpful.

7. Test Scenarios:

1. User Registration

- **Scenario 1: Mandatory Fields Validation**
    - Test: Attempt to submit the registration form with required fields left blank.
    - Expected Outcome: The system should prevent submission and display error messages indicating the missing fields.
- **Scenario 2: Input Validation**
    - Test: Enter invalid data, such as an incorrectly formatted email address or a weak password.
    - Expected Outcome: xThe system should display error messages specifying the invalid inputs and prompt for correction.
- **Scenario 3: Unique Username/Email**
    - Test: Try to register with a username or email that already exists in the system.
    - Expected Outcome: The system should prevent registration and display a message indicating the username or email is already taken.
- **Scenario 4: Password Confirmation**
    - Test: Enter mismatched passwords in the 'Password' and 'Confirm Password' fields.
    - Expected Outcome: The system should display an error message indicating that the passwords do not match.
- **Scenario 5: Successful Registration**
    - Test: Provide valid and unique information for all fields and submit the form.
    - Expected Outcome: The system should register the user successfully and display a confirmation message or send a verification email.

- **Scenario 6: Error Messaging**
    - Test: Submit the registration form with various invalid inputs.
    - Expected Outcome: The system should display clear and specific error messages for each invalid input or missing field.
- **Scenario 8: Email Verification**
    - Test: Complete the registration process and check the provided email for a verification link.
    - Expected Outcome: The system should send a verification email with a link that, when clicked, activates the user's account.
- **Scenario 9: Successful Registration with Valid Email**
    - Test: Register with a valid email address and complete all required fields.
    - Expected Outcome: The system should accept the registration and send a "Verify Your Email to Get Started with ScopeX" email to the provided address.
- **Scenario 10: Immediate Password Reset Request**
    - Test: After successful registrationmake sure user receives "Forgot Password" option.
    - Expected Outcome: The system should send a password reset email promptly to the registered email address.
- **Scenario 11: Registration with Invalid Email**
    - Test: Register with an invalid or incorrectly formatted email address.
    - Expected Outcome: The system should display an error message indicating the email format is invalid and prevent registration.
- **Scenario 12: Duplicate Email Registration**
    - Test: Attempt to register with an email address that is already associated with an existing account.
    - Expected Outcome: The system should display an error message indicating the email is already in use and prevent registration.
- **Scenario 13: Missing Required Fields**
    - Test: Submit the registration form with one or more required fields left blank.
    - Expected Outcome: The system should display error messages for each missing field and prevent registration.
- **Scenario 14: Password Strength Validation**
    - Test: Enter a password that does not meet the system's strength requirements (e.g., too short, lacks special characters).
    - Expected Outcome: The system should display an error message indicating the password does not meet the required criteria and prevent registration.
- **Scenario 15: Successful Registration and Email Verification**
    - Test: Complete the registration process with valid information and verify the email address by clicking the verification link.
    - Expected Outcome: The system should activate the account and allow the user to log in.

2. Adding a Recipient

- **Scenario 1: Access to Add Recipient Feature**
  - Test: Try to access the 'Add Recipient' feature without logging in.
  - Expected Outcome: The system should redirect the user to the login page, preventing access to unauthenticated users.
- **Scenario 2: Mandatory Fields Validation**
  - Test: Attempt to add a recipient with required fields left blank.
  - Expected Outcome: The system should prevent submission and display error messages indicating the missing fields.
- **Scenario 3: Input Validation**
  - Test: Enter invalid recipient information, such as an incorrectly formatted account number.
  - Expected Outcome: The system should display error messages specifying the invalid inputs and prompt for correction.
- **Scenario 4: Duplicate Recipient Prevention**
  - Test: Try to add a recipient with details that already exist in the system.
  - Expected Outcome: The system should prevent duplication and display a message indicating the recipient already exists.
- **Scenario 5: Successful Addition**
  - Test: Provide valid and unique recipient information and submit the form.
  - Expected Outcome: The system should add the recipient successfully and display them in the recipient list.
- **Scenario 6: Error Messaging**
  - Test: Submit the 'Add Recipient' form with various invalid inputs.
  - Expected Outcome: The system should display clear and specific error messages for each invalid input or missing field.
- **Scenario 7: Editing/Deleting Recipient**
  - Test: Edit or delete an existing recipient's information.
  - Expected Outcome: The system should update or remove the recipient's information accordingly and reflect the changes in the recipient list.

3. Logout

- **Scenario 1: Logout Functionality**
  - Test: Click the 'Logout' button.
  - Expected Outcome: The system should log the user out and redirect them to the login page.
- **Scenario 2: Session Termination**
  - Test: Attempt to access a restricted page after logging out.
  - Expected Outcome: The system should prevent access and redirect the user to the login page.
- **Scenario 3: Cache and Cookie Clearance**
  - Test: Check the browser's cache and cookies after logging out.
  - Expected Outcome: All user-specific data should be cleared from the browser's cache and cookies.

- **Scenario 4: Security**
    - Test: Use the browser's back button to navigate to a previously accessed restricted page after logging out.
    - Expected Outcome: The system should prevent unauthorized access and redirect the user to the login page.
- **Scenario 5: Multiple Device Logout**
    - Test: Log out from one device and attempt to access the application from another device.
    - Expected Outcome: The user should be able to access the application on the other device without issues.

8. Test Strategy

8.1:Testing Levels

The testing process will be structured across the following levels:

- **Unit Testing:**
    - Objective: Check that each part of the application works correctly on its own.
    - Scope: Test individual functions, like those that handle user input, add recipients, or manage logout.
    - Approach: Write tests for each function to ensure they work as intended when tested separately.
- **Integration Testing:**
    - Objective: Confirm that different parts of the application work well together.
    - Scope: Test how modules interact, such as the connection between user registration and the database, or between recipient management and the user interface.
    - Approach: Run tests to ensure data flows correctly between connected parts and that the system responds properly to various inputs.
- **System Testing:**
    - Objective: Ensure the entire application meets the specified requirements.
    - Scope: Test the whole application, including features like user registration, adding recipients, and logout, in the intended environment.
    - Approach: Perform end-to-end tests to check the application's behavior in different scenarios, making sure all parts work together smoothly.
- **Acceptance Testing:**
    - **Objective**: Determine if the application meets business requirements and is ready for release.
    - **Scope**: Validate that the application meets the acceptance criteria set by stakeholders.
    - **Approach**: Execute test cases that mimic real-world use to confirm the system meets user expectations and requirements.
- **Cross-Browser Testing:**
    - **Objective**: Ensure the application provides a consistent experience across different web browsers.

- **Scope**: Test the application's features on various browsers, like Chrome, Firefox, Safari, and Edge, to find and fix compatibility issues.
- **Approach**: Use automated tools to test that features like user registration, adding recipients, and logout work correctly across all supported browsers.

8.2: Testing Types:

To meet the stated objectives, the following test types will be performed by the QA team during the software testing life cycle:

- Functional Testing
- Usability Testing
- Compatibility Testing
- Cross-Browser Testing
- Exploratory Testing
- Regression Testing
- Smoke Testing
- Sanity Testing
- Acceptance Testing

8.3: Test Deliverables:

- Test Cases
- Test execution reports
- Defect reports
- Sign-Off report
- Test Summary Report

9. Testing Environment

- **Hardware**
    - **Servers**: We'll use high-performance servers to host the application and its database, ensuring they can handle multiple users simultaneously.
    - **Client Devices**: Testing will be conducted on various devices to ensure compatibility:
        - **Desktop Computers**: For testing the web version of the application.
        - **Mobile Devices**: Smartphones and tablets for testing the mobile application.

- **Software**
    - **Operating Systems**: We'll test on multiple operating systems to ensure the application works across different platforms:
        - **Windows**: For desktop testing.
        - **macOS**: For desktop testing.
        - **iOS**: For mobile testing.
        - **Android**: For mobile testing.

    - **Browsers**: To ensure the web application works well across different browsers, we'll test on:
        - **Google Chrome**
        - **Mozilla Firefox**
        - **Safari**
        - **Microsoft Edge**
    - **Testing Tools**: We'll use specific tools to help manage and automate our testing process.

- **Network**

    - **Internet Connection**: A stable and fast internet connection to simulate typical user conditions.
    - **Security Settings**: Configured to match the live environment, ensuring our security tests are accurate.

- **Test Data**

    - **User Accounts**: Sample accounts to test user registration and login features.
    - **Recipient Information**: Sample data to test adding and managing recipients.
    - **Transactions**: Sample transaction data to test related functionalities.

- **Environment Setup**

    - **Staging Environment**: A setup that mirrors the live environment for initial testing phases.
    - **Production Environment**: The actual live environment for final testing phases, ensuring the application works as intended in real-world conditions.

10. Assumptions

Following assumptions are taken into consideration.

- **Stable Requirements:** The application's requirements are well-defined and will remain consistent throughout the testing phase.
- **Environment Availability**: All necessary testing environments, including hardware, software, and network configurations, will be available and properly configured when needed.
- **Resource Allocation**: Adequate personnel and tools are allocated for both manual and automated testing activities.
- **Responsive Design:** The website is designed to be responsive and adapt to various screen sizes, including desktops, tablets, and mobile devices.
- **Browser Compatibility**: The application is designed to support the latest versions of major browsers, including Chrome, Firefox, Safari, and Edge.
- **Mobile Platforms**: The mobile application supports the specified versions of iOS and Android operating systems.
- **Timely Issue Resolution**: Any defects identified during testing will be addressed promptly to prevent delays in the testing schedule.

- **User Authentication:** User authentication mechanisms, such as login and registration, are functioning correctly.
- **Data Integrity:** Data entered into forms is accurate and follows the specified format.
- **Development and Test team**: deadlines Testing team should be involved in initial project discussions and should have a working knowledge of the proposed system prior to integration and system testing
- It's important to note that these assumptions may vary based on the specific requirements and context of the website being tested. Regular updates and adjustments to assumptions should be made as the website evolves or as new information becomes available.
- Test data to be provided by the client.
- Test data contains positive as well as negative inputs.

11. Success factors

Listed below are the critical success factors to allow the execution of testing for the project.

- Clear requirements guide precise testing.
- Thorough test cases cover all functionalities.
- Iterative testing in sprints addresses issues progressively.
- Stable devices ensure reliable testing.
- Latest build availability validates new features.
- Security testing identifies and mitigates vulnerabilities.
- Regression testing maintains system stability.
- Continuous monitoring tracks progress and highlights improvements.
- UAT alignment validates against end-user expectations.
- Quality documentation supports knowledge sharing in testing.
- Adaptability in testing processes ensures agility amid project changes.

12. Test Inputs-STLC

- Requirement Document
- Test plan
- Identification of Test Scenarios
- Test-Case Design
- Test-Case Review
- Unit tested build
- Change to be tested
- Test data
- Test Case Execution
- Documentation

13. Test Case Design Techniques

Error Guessing:

- Enter blank spaces into text fields.
- Trigger null pointer exceptions.
- Input invalid parameters.
- Attempt division by zero.
- Upload files with the maximum limit.
- Check button functionality without entering values.
- Increment test cases based on tester experience.

Boundary Value Analysis:

- For a range of 10-100:
- Test with values at the minimum (10), maximum (100), and on either side of boundaries.
- Test with values just above (MAX+1), just below (MAX-1), just above minimum (MIN+1), and just below minimum (MIN-1).

Equivalent Class Partition:

- Validate input data types by dividing them into equivalent partitions.
- Example for values 1 to 60:
- Partition 1-20
- Partition 21-40
- Partition 41-60
- A-Z

Decision Table Technique:

- Test with different combinations of input values using decision tables.

14. Risks

- **Tight Schedules:** Limited time for testing might lead to missing some bugs.
- **Changing Requirements:** If the application's features or requirements change frequently, it can disrupt our testing process.
- **Limited Resources:** Not having enough team members or tools could slow down our testing efforts.
- **Technical Challenges:** Complex features or integration issues might cause unexpected problems during testing.
- **Data Issues:** Problems with test data, like inaccuracies or missing information, can affect test results.
- **Security Concerns:** Potential vulnerabilities might expose the application to security threats.
- **Compatibility Problems:** The application might not work properly across all intended devices or browsers.
- **Performance Bottlenecks:** The application could respond slowly under heavy user load.
- **User Experience Issues:** The interface might not be intuitive, leading to user dissatisfaction.

15. Constraints

- **Time Limitations:** We have a set schedule to complete our testing. This means we need to prioritize our tasks to ensure we cover the most critical areas first.
- **Resource Availability:** Our team has a specific number of members and tools at our disposal. We'll need to manage these resources carefully to maximize our testing effectiveness.
- **Access to Test Environments:** We might face challenges in accessing environments that closely resemble the live application setting. This could impact the accuracy of our test results.
- **Documentation Gaps:** If there are missing or incomplete documents about the application's features or requirements, it could lead to misunderstandings during testing.
- **Changing Requirements:** If the application's features or requirements change during the testing phase, it might require us to adjust our testing approach, which could affect our schedule.
- **Technical Constraints:** Certain technical limitations, such as dependencies on third-party services or specific hardware requirements, may restrict our testing capabilities.
- **Budget Constraints:** Limited financial resources may affect our ability to acquire necessary tools, technologies, or additional personnel for comprehensive testing.
- **Regulatory Compliance:** Adhering to industry regulations and standards may impose specific constraints on our testing procedures and data handling practices.

16. Tools to be used

- Test Case management Tool – Qtest Manager
- Defect Tracking Tool – Jira
- Cross browser testing- Browserstack

17. Defect Classification and Defect Life Cycle

All software bugs/defects/anomalies will be prioritized using the following levels:
Types:

- **Enhancement Request**– System function or process identified as required in the future.
- **Problem –** Defect/anomaly in code was encountered.
- **Business Issue / Change Request –** Requires decision by a Product Management team – possible change in the requirements document.

Priority:

- **Level 1 - High** – i)Testing for a major function cannot continue until the error is repaired and requires immediate action. This is a “show stopper” error.
  ii)Serious errors where the system cannot go live until it is fixed. Testing is usually able to continue.
- **Level 2 - Medium** – An adverse effect has been identified , however, a workaround can be implemented, enabling the system to go live. A risk assessment of the workaround needs to be performed and documented by the Project Management Team.
- **Level 3 - Low** – No major adverse effects to the business environment. This is generally a text or graphical error that can be repaired with a future build or release.

17.1 Defect reporting and Tracking Mechanism

Following state describe the way to carry out defect tracking in Orange Scrum.

1. **New -** QA is expected to apply all his testing skills and test the application to cover all the test cases, from positive as well as negative approaches. Once a encounters unexpected behavior which is nothing but defect then that defect should be entered into the OrangeScrum with the status New Bug.

   QA should provide the following information which describe the bug with more clarity.

   - **Assigned To -** Select the name From drop-down to assign the task.
   - **Due Date -** Provide a due date.
   - **Priority -** Provide priority to the task i.e. Low, Medium, High.
   - **Task Type -** Select the task type as bug.
   - **Milestone -** Provide the Milestone.
   - **Description -** Describe the Task details along with Build Name , Username, and Password.
   - **Attachment -** Provide the attachment if needed.
   - **Notify via Email -** Select the user to get notified by email.

2. **InProgress -** Once the bug task is assigned to the Developer , he/she will start working on it and he/she will change the status as Inprogress.
3. **Resolved -** Once the defect is resolved by the development team - they will mark it as Resolved then QA will test it.
4. **Close -** Once the tested defect is fixed and working fine then it will be marked as closed.

18. Roles & Responsibilities

Sr. No.	Roles	Responsibilities
1.	Test Engineer Shivraj Kamate	Preparing test cases, and providing test execution in Staging, Production environments.

19. Test Schedule

Record schedule here or give reference to Project Schedule.

Task	Milestone	Duration	Start	Finish

20. Test Output/ Deliverables

Sr. No.	Deliverable Name	Planned End Date	Responsibility	Delivery Mechanism
1.				
2.				
3.				
4.				
5.				