

PROJECT

# SIP SPLIT WEBSITE TEST PLAN

PREPARED BY

Felix O. Ugbeja

REVIEWED BY PM, DEV & PO

DATE 18TH SEPTEMBER, 2025

Signed off by	
Title	Sip Split Test Plan
Version	1.0
Date	2025/09/17

The test plan for the SIP SPLIT website provides a detailed roadmap for ensuring the application is functional, reliable, secure, and user-friendly. The test plan is structured and comprehensive, covering all in-scope aspects (and some identified future elements) for the SIP Split bill web application.

## **1. Introduction**

This section provides an overview of the test plan.

### **Purpose:**

To define the scope, objectives, schedule, and resources for testing the Sip Split web application, ensuring it meets all functional and non-functional requirements.

### **Target Audience:**

This document is intended for the QA team, developers, project manager, and product owner.

**System Under Test:** SIP Split Web Application

## **2. Test scope**

Clarify what is included and excluded from the test plan scope.

### **In scope**

- Adding, editing, and deleting expenses.
- Splitting the bill equally.
- Responsive design for web browsers.
- OCR-enabled phone to scan and convert a paper bill to text.
- Column to insert the total number of users.
- Column to insert total bill (optional in scenarios where OCR doesn't work).
- Split bill button: To equally divide the total bill by the number of users/participants.

### **Out of scope**

- User profile management.
- User registration and login.
- Group creation and management.
- Splitting the bill by percentages or other methods.
- Sending payment reminders.
- Tracking expense history within a group.
- Recording payments and settling debts.
- Third-party payment gateway functionality (beyond integration).
- Testing on unsupported browsers or operating systems.
- Specific mobile app testing (if it is a separate entity).
- Group analytics

### 3. Test strategy and approach

The testing approach and methodologies for this project shall be a combination of functional, regression, usability, and performance testing.

**Methodologies:** A combination of functional, regression, usability, performance, and testing will be used. Automated testing will be prioritized for critical and repetitive test cases, such as regression tests.

**Table 1: Testing Levels**

Testing Type	Description	Performed By
Unit Testing	To be performed by the DevOps team to test individual components.	DevOps team
Smoke Testing	To ensure functionalities can be iterated for future enhancements.	
Integration Testing	To verify communication between different parts of the application, such as the front-end and back-end APIs.	
System Testing	End-to-end testing of the complete application flow.	
Regression Testing	If applicable	
User Acceptance Testing (UAT)	To be conducted by PO, PM, BA, Developer, and UI/UX to ensure the website meets users' needs.	PO, PM, BA, Developer, UI/UX

### 4. Test environment and data

The required environment and data for testing shall be,

- **Environment:** TBD by Thea, PO, Developer, and QA
- **Hardware:** N/A
- **Software:**
  - Web browsers: Chrome, Firefox, Safari,
  - Operating systems: Windows, Android OS, Apple, macOS, and iOS.
- **Test Data:** Create and anonymize sample user data, groups, and expenses to cover various scenarios (e.g., even split, custom amounts, bill entry).

## 5. Roles and responsibilities

Assigned clear responsibilities to team members.

**Table 2: Roles & Responsibilities.**

Role	Responsibilities
QA Lead	Oversees the test plan, manages the QA team, and reports status to PM and PO
Developers	Perform unit testing and assist QA in reproducing and fixing bugs
Project Manager	Participates in ALL tests and provides sign-off
Product Owner	Provides sign-off for deliverables

## 6. Test deliverables

List of documents and artifacts that will be produced during the testing process.

- Test plan document.
- Test cases and scenarios.
- Test execution reports.
- Defect logs and reports.
- Summary report at the end of each test cycle.

## 7. Entry and Exit Criteria

Define the conditions for starting and stopping testing activities.

**Table 3: Entry and Exit Criteria**

Criteria Type	Criteria	Start of Testing	End of Testing
Entry Criteria	The test plan is reviewed and signed off.		
"	All features for the current release are code-complete.		
"	The test environment is set up and stable.		
Exit Criteria	All test cases have been executed.		
"	A pre-defined percentage of critical test cases have passed (% to be decided by PM, QA, and PO).		
"	No open critical or high-severity defects remain.		
"	The test summary report is signed off.		

## 8. Test schedule

**Table 4: Test Schedule** (Timeline for testing, including key milestones.)

Phase	Start Date	End Date
Planning	[Start Date]	[End Date]
Execution	[Start Date]	[End Date]
Regression	[Start Date]	[End Date] (If Applicable)
UAT	[Start Date]	[End Date]

## 9. Test cases and scenarios

A breakdown of test cases, organized by test areas.

**Table 5: Functional Test Scenarios**

Test Area	Scenario	Applicability
Bill Entry	Calculation with valid input (Happy path)	APPLICABLE
"	Validates system rejects '0' bill amount	"
"	Validates system rejects 'number of people' 0, <2	"
"	Verify the system rejects negative values in the total number of people field	"
Summary Display	Navigation via the 'Home' button	"
"	Verify the system auto-calculates when friends are added	"
"	Verify the system auto-calculates when friends are removed	"
"	Test the ability to edit and delete existing expenses.	APPLICABLE
Product Design	Verify the website adapts to a portrait orientation on a mobile phone	APPLICABLE
"	Verify functionality consistency on different devices	APPLICABLE
"	Verify expense calculations with different currencies.	Not APPLICABLE
Expense Management	Test adding a new expense with a simple, even split.	APPLICABLE
"	Verify that a user can add an expense and assign different amounts to each group member.	APPLICABLE
"	Verify that debt balances are accurately updated after a payment is recorded.	APPLICABLE
"	Test the "Reset" functionality to zero out all balances within a group.	APPLICABLE
Notifications and Reminders	Verify that a user can send a payment reminder to a group member.	NOT APPLICABLE

“	Test that notifications are correctly delivered to users.	NOT APPLICABLE
User Management	Verify that a new user can register with a unique email and a valid password.	NOT APPLICABLE
“	Test the login process with valid and invalid credentials.	NOT APPLICABLE
“	Verify the "Forgot Password" functionality.	NOT APPLICABLE
Group Management	Verify that a user can create a new group.	NOT APPLICABLE
“	Test adding and removing members from a group.	NOT APPLICABLE
“	Verify that group expenses and balances are updated correctly when members are added or removed.	NOT APPLICABLE

**Table 6: Non-functional Test Scenarios**

Testing Type	Details	Applicability
Usability Testing	Verify that the website is easy to navigate and understand. Test the intuitiveness of the bill splitting workflow. Ensure all interface elements are clear and logical.	APPLICABLE
Performance Testing	Measure the response time for adding a new expense under a simulated load of 15 concurrent users. Test the application's stability with a large number of users and groups.	APPLICABLE
Compatibility Testing	Verify that the website functions correctly across different web browsers (Chrome, Firefox, Safari) and screen sizes (desktop, tablet, mobile).	APPLICABLE
Security Testing	Test for common vulnerabilities like SQL injection and Cross-Site Scripting (XSS). Verify that sensitive user data, such as balances, is properly secured and not accessible to unauthorized users.	NOT APPLICABLE

## 10. Risk management

Identify potential risks and outline mitigation strategies.

**Table 7: Risk management grid.**

Risk	Mitigation
------	------------

The tight project schedule and low-level collaboration (except Thursday meetings) may lead to inadequate test coverage.	Prioritize testing of core functionalities and automate repetitive test cases where necessary.
Critical bugs are found late in the testing cycle.	Shift-left testing by integrating testing earlier in the development process.
The test environment is unstable or unavailable.	Automate environment provisioning and clearly define the escalation process for environment issues.