Testing Document and Specification

Test Plan

CS451R

Recur

Curtis Lemke, Logan George, Viger Romo, Brendan Clouston

Introduction

This document outlines the test plan for the Recur subscription tracking application. As outlined in the project Requirements Document, this system needs to provide a medium for users to connect financial data, identify and track recurring subscriptions, view overall spending, and manage subscription-related preferences and account settings.

This application will allow users to securely authenticate, complete an initial setup flow, and then monitor recurring subscription charges over time via dashboards and analytics views. It will also provide mechanisms for users to adjust preferences such as notification settings, theme (light/dark), and two-factor authentication options on their account.

The testing activities discussed in this document will verify that the Recur software meets the needs of Commerce Bank by ensuring that the requirements for this system, as outlined in the Requirements Document, are met.

The testing routine will evaluate Recur's behavior when a user signs up and logs in, completes setup, connects bank accounts or imports data, views analytics, manages subscriptions, and changes preferences or security settings. Additionally, the routine will test the system's handling of protected routes, redirections (e.g., from the dashboard to setup when no data exists), and integration with external services such as Supabase (authentication/database) and Plaid (bank connections). The results of this testing procedure will enable the creators of this system to gauge project success as outlined in the Macro Project Plan.

Terminology

Throughout this document, the terms user, subscription, transaction, system, site, and username/password will be used frequently; therefore, formal definitions will be given

System - the Recur web application, including the Next.js frontend, Supabase backend (authentication and database tables such as userlogin and transactions), and Plaid integration used to retrieve financial data.

User - any individual who creates an account in Recur (via email and password) and interacts with the application to track and manage their subscriptions.

Subscription - a recurring payment (e.g., streaming service, SaaS tool, mobile app, gym membership) identified either automatically from transactions or entered manually by the user; subscriptions are characterized by merchant name, amount, category, and renewal cadence.

Transaction - a financial record stored in the system's transactions table, typically imported from Plaid or file import, containing information such as date, amount, merchant, and whether it is considered recurring.

Site - the Recur front-end web site (e.g., /login, /signup, /setup, /homepage, /analytics, /preferences, /accountSettings) that the user interacts with in a browser.

Username/Password - unique identifiers (email and password) that authenticate and validate a user through Supabase Auth.

Items Tested

Items that will be tested during the testing phase, as laid out by the Project Plan, will include, but are not limited to:

- Ability for an unauthenticated user to be redirected appropriately and for an authenticated user to log in and access the dashboard

Test Case: 4.1.1.1 (Login with valid/invalid credentials), 4.1.1.2 (Redirect to /login from protected pages), 4.1.1.3 (Logout behavior)

- Ability for a new user with no transactions to complete the setup flow and then access the main dashboard

Test Case: 4.1.2.1 (Redirect to /setup for zero-transaction user), 4.1.2.2 (Complete bank/CSV/manual setup), 4.1.2.3 (Return to /homepage after setup)

- Ability for a user to connect a bank account via Plaid and have recurring subscription transactions imported and identified

Test Case: 4.1.3.1 (Successful Plaid Link flow), 4.1.3.2 (Handle Plaid errors gracefully), 4.1.3.3 (Recurring detection and grouping)

- Ability for a user to import or manually add subscriptions and see them reflected in subscription lists and analytics

Test Case: 4.1.4.1 (Import file and verify transactions), 4.1.4.2 (Manual subscription creation and display), 4.1.4.3 (Editing subscription details)

- Ability to view analytics, including category breakdowns and monthly totals for subscriptions, and to navigate from analytics to subscription management

Test Case: 4.1.5.1 (Category pie chart and tooltips), 4.1.5.2 (Monthly bar chart), 4.1.5.3 (Subscriptions list and “View All Subscriptions” navigation), 4.1.5.4 (Insight text correctness)

- Ability for a user to configure preferences such as email notifications, weekly summaries, and theme (light/dark), and have changes persist to Supabase

Test Case: 4.2.1.1 (Load prefs from userlogin and user_metadata), 4.2.1.2 (Toggle email notifications), 4.2.1.3 (Toggle weekly summary), 4.2.1.4 (Change theme)

- Ability for a user to manage account settings, including two-factor authentication settings and OTP delivery method, and to initiate account deletion

Test Case: 4.2.2.1 (Load account settings data), 4.2.2.2 (Enable/disable 2FA), 4.2.2.3 (Change OTP method email/SMS), 4.2.2.4 (Account deletion entry)

Items Not Tested

Some features will not be included in the current testing procedure. This does not mean that these features will not be implemented in the future, but at the current moment, they will not be tested.

- Historical archives or export of long-term subscription history (e.g., exporting multiple years of data to external formats)

- Detailed administrative tooling for managing users at a system level (e.g., admin dashboards, global reporting across users)

- Advanced alerting mechanisms beyond basic email/weekly summaries (e.g., SMS alerts for renewals, push notifications, complex budgeting rules)

- Deep cross-browser and device testing (older browsers, legacy mobile devices)

- Performance benchmarking at production scale with large datasets and high concurrency

Approach

- The overall method for this testing procedure is manual system testing, with test cases derived directly from the functional and non-functional requirements in the Requirements Document.

- Each test case created will have a direct link to one or more requirements and will focus on concrete user flows. Test cases that include similar feature methods will be tested together. Examples of these features include logging in to access the dashboard, completing setup, connecting Plaid, importing/manual entry of subscriptions, and accessing protected routes such as /homepage, /analytics, /preferences, and /accountSettings.

- Test cases such as login, logout, and access control test both usability and security features of the system, ensuring that only authenticated users can access sensitive pages. For these, both valid and invalid data (e.g., incorrect password, unknown email) will be used to confirm that Recur enforces constraints consistently and provides clear feedback messages.

- Features that involve financial data and analytics, such as importing transactions, identifying recurring subscriptions, and calculating monthly totals, will be tested together in coordinated scenarios. Testers will verify that transactions imported via Plaid or other means are correctly stored, grouped into subscriptions, and surfaced in the dashboard and analytics with correct amounts, categories, and renewal dates. Visual checks on charts and lists will validate that values rendered align with the underlying data.

- Manual system testing will continue throughout each iteration of the project. For each iteration, both previously implemented and newly implemented features will be tested. Adding new features or functionality can sometimes interfere with the functionality of existing features; to ensure product and project success, all features implemented should function as intended throughout the life of the software.

Pass/Fail Criteria

- The minimum requirements for this software system are laid out in the Requirements Document, and the Macro Project Plan outlines what the creators consider project success.

- Implemented features that meet the requirements as determined by the customer, meaning the feature does what the user expects it to do with very little difficulty, and passes the testing procedure. Difficulty, as used here, is determined by user comprehension and the user's ability to use the feature with little to no training. For Recur, this includes flows such as connecting to a bank, understanding analytics, and toggling preferences without confusion.

  Features that contain major defects will fail the testing procedure and will be documented via an incident report and reviewed by developers for investigation and revision. Examples of major defects include incorrect financial calculations, inability to log in or access protected pages, crashes, or data loss. Medium and minor defects (such as cosmetic UI issues or minor wording inconsistencies) will be recorded and prioritized according to severity and impact on user tasks.

Test Deliverables

In addition to this Test Plan, other test deliverables include the Test Specification, which outlines the specific test cases and expected results of each test, and Test Reports, which are comprised of incidents, defects, and changes.

Testing Tasks
The following lists the testing deliverables and the activities required to produce each deliverable.

Deliverables / Activities

Test Plan

- Analyze requirements for Recur system features (authentication, setup, analytics, subscription management, preferences, account settings)
- Determine testable and non-testable features for the current iteration
- Develop approach/method for testing (manual system testing, coverage by feature area)
- Determine tasks and estimate effort needed for each feature area
- Develop a schedule for testing aligned with development milestones

Test Specifications

- Analyze requirements in detail for each feature and user flow
- Define test cases for testable features as outlined by the Test Plan, including:
  - Login/signup/reset flows
  - Setup and data import/manual entry
  - Dashboard and analytics views
  - Subscription management operations
  - Preferences and theme switching
  - Account security and 2FA configuration
- Document preconditions, test steps, input data, and expected results for each test case

Test Reports

- Implement test cases as outlined by the Test Specifications
- Document incidents and defects observed during execution, with steps to reproduce and environment details
- Determine severity and priority of incidents and defects (e.g., blocking, critical, major, minor)
- Determine changes that need to be made to the system, including design clarifications where necessary
- Document and submit change requests to developers and track status through resolution