Testing Document and Specification

Test Specifications
CS451R

Recur

Curtis Lemke, Logan George, Viger Romo, Brendan Clouston

Testing Specifications

Each test case should include ID, Title, Purpose, Initial Conditions, Test Data, Actions, Expected Results.

Test Case ID: 4.1.1.1

Title: Login with Valid Credentials
Feature/Subfeature: Authentication / Login
Purpose: To ensure that a user can log in to Recur using a valid email and password and be redirected to the homepage.
Initial Conditions: Application is running. User account exists in Supabase with confirmed email and known password. Browser is not currently authenticated. User is on the /login page.
Test Data:
Email: valid_user@example.com
Password: ValidPassword123!
Test Actions:
Open the /login page in a browser.
Enter valid_user@example.com into the Email field.
Enter ValidPassword123! into the Password field.
Click the “Login” button.
Expected Results:
After step 4, the login request succeeds, a Supabase session is created, and the user is redirected to /homepage. No error message is shown.

Test Case ID: 4.1.1.2

Title: Login with Invalid Credentials
Feature/Subfeature: Authentication / Login Validation
Purpose: To ensure that invalid credentials prevent login and display an appropriate error message.
Initial Conditions: Application is running. No session exists in the browser. User is on the /login page.
Test Data:
Email: valid_user@example.com
Password: WrongPassword999!
Test Actions:
Open the /login page.
Enter valid_user@example.com into the Email field.
Enter WrongPassword999! into the Password field.
Click the “Login” button.
Expected Results:

After step 4, login fails. The user remains on /login and an error message is displayed indicating invalid credentials. No session is created.

Test Case ID: 4.1.1.3

Title: Protected Route Redirect When Unauthenticated
Feature/Subfeature: Authentication / Route Protection
Purpose: To ensure that unauthenticated users are redirected to /login when attempting to access a protected route.
Initial Conditions: Application is running. No valid Supabase session exists in the browser.
Test Data: None.
Test Actions:
Navigate directly to /homepage in the browser.
Observe the resulting behavior.
Expected Results:
The system checks for a session, detects that no user is authenticated, and redirects the browser to /login.

Test Case ID: 4.1.1.4

Title: Logout from Sidebar
Feature/Subfeature: Authentication / Logout
Purpose: To ensure that users can log out and have their session cleared.
Initial Conditions: User is logged in with a valid session and currently on any authenticated page (for example, /homepage). Sidebar is visible with a logout control.
Test Data: None.
Test Actions:
While on /homepage, open the sidebar if not already visible.
Click the “Logout” or equivalent sign-out button.
Expected Results:
After step 2, Supabase signs the user out, the session is cleared, and the user is redirected to /login. Attempting to revisit /homepage without logging in again results in a redirect back to /login.

Test Case ID: 4.1.2.1

Title: Zero-Transaction User Redirected to Setup
Feature/Subfeature: Onboarding / Setup Redirect
Purpose: To ensure that a new user with no transactions is redirected from /homepage to /setup.
Initial Conditions: New user account exists and is logged in. Transactions table has zero rows for this user.
Test Data:
User ID: New user with no transactions.

Test Actions:
Log in as the new user.
Navigate to /homepage.
Expected Results:
The server checks the transactions table, finds no transactions for the user, and redirects the user from /homepage to /setup.

Test Case ID: 4.1.2.2

Title: Complete Setup via Manual Subscription Entry
Feature/Subfeature: Onboarding / Manual Setup
Purpose: To ensure that a user can complete setup by adding a manual subscription.
Initial Conditions: User is authenticated and has zero transactions. User has been redirected to /setup and is on the manual setup page (for example, /setup/manual).
Test Data:
Subscription Name: Netflix
Amount: 15.99
Category: Entertainment
Renewal: Monthly
Test Actions:
From /setup, navigate to the manual setup option.
Enter “Netflix” as the subscription name.
Enter 15.99 as the amount.
Select category “Entertainment”.
Select “Monthly” as the frequency or renewal.
Click the save or “Add Subscription” button.
Expected Results:
A new subscription record is created and at least one corresponding recurring transaction or subscription entry is stored. Setup completes successfully, and the user can proceed to /homepage without being redirected back to /setup on future visits.

Test Case ID: 4.1.2.3

Title: Homepage Accessible After Setup Completion
Feature/Subfeature: Onboarding / Setup Completion
Purpose: To ensure that once the user has transactions or subscriptions, they are no longer redirected to /setup.
Initial Conditions: User has completed setup and has at least one transaction or subscription in the database. User is logged in.
Test Data: Any existing subscription or transaction for the user.
Test Actions:
Log in as the user who completed setup.
Navigate to /homepage.
Expected Results:

The system detects that the user has transactions and renders the homepage dashboard instead of redirecting to /setup.

Test Case ID: 4.1.3.1

Title: Successful Plaid Connection and Import
Feature/Subfeature: Data Sources / Bank Connection (Plaid)
Purpose: To ensure that a user can connect a bank via Plaid and import transactions.
Initial Conditions: User is authenticated, has no or few existing transactions, and is on /setup/bank or /add-more/bank. Plaid sandbox credentials are correctly configured.
Test Data:
Plaid sandbox credentials and institution (for example, Plaid “First Platypus Bank”).
Test Actions:
Click the button to “Connect Bank” (launch Plaid Link).
Complete the Plaid sandbox flow by selecting a test institution and account.
Authorize connection and return to Recur.
Wait for transactions to be imported.
Expected Results:
After step 3, Plaid Link closes and Recur shows confirmation or progress. After step 4, the transactions table contains imported transactions for this user, and relevant recurring charges appear in the subscription and analytics views.

Test Case ID: 4.1.3.2

Title: Plaid Error Handling
Feature/Subfeature: Data Sources / Bank Connection Error Handling
Purpose: To ensure the system handles Plaid errors gracefully without crashing.
Initial Conditions: User is authenticated and on /setup/bank or /add-more/bank. Plaid is reachable but configured to simulate an error (for example, user cancels or sandbox error).
Test Data: None specific; use Plaid’s cancel or error scenarios.
Test Actions:
Click “Connect Bank” to open Plaid Link.
Intentionally cancel the flow or trigger a known sandbox error (for example, close the window without completing).
Observe the Recur UI.
Expected Results:
Recur displays a clear error or cancellation message and remains usable. No crash or blank screen occurs. No partial or invalid connection is saved.

Test Case ID: 4.1.3.3

Title: Recurring Subscription Detection from Transactions
Feature/Subfeature: Data Processing / Recurring Detection

Purpose: To ensure imported transactions are grouped into recurring subscriptions for analytics and listing.
Initial Conditions: User has multiple debit transactions imported from Plaid for the same merchant over several months.
Test Data:
Transactions: Three debit charges from “Spotify” over three consecutive months for the same amount.
Test Actions:
Log in as the user with the Spotify transactions.
Navigate to /analytics.
Navigate to the subscriptions list (from analytics or /homepage/subscriptions).
Expected Results:
The system groups the Spotify transactions into a single subscription entry with the latest amount and a calculated next renewal date (approximately one month after the last charge).
The subscription appears in analytics and the subscriptions list.

Test Case ID: 4.1.4.1

Title: Import Subscriptions from File
Feature/Subfeature: Data Sources / Import
Purpose: To ensure that importing a file creates transactions that are visible in analytics and lists.
Initial Conditions: User is authenticated and on /setup/import or /add-more/import. Import functionality is configured.
Test Data:
Import file containing several recurring charges, for example a CSV with monthly “Netflix” and “Hulu” entries.
Test Actions:
Navigate to the import page.
Select the prepared import file.
Confirm and submit the import.
After processing, navigate to /analytics.
Expected Results:
The system ingests the file, creates corresponding transactions, identifies recurring subscriptions, and shows them in analytics charts and subscription lists. No parsing errors occur for valid data.

Test Case ID: 4.1.4.2

Title: Add Manual Subscription and Display in Lists
Feature/Subfeature: Subscriptions / Manual Entry
Purpose: To ensure manual subscriptions appear in the subscription list and analytics.
Initial Conditions: User is authenticated and on a page that allows manual subscription entry (for example, /setup/manual).

Test Data:
Name: Gym Membership
Amount: 30.00
Category: Fitness
Frequency: Monthly
Test Actions:
Open the manual subscription form.
Fill in “Gym Membership”, amount 30.00, category “Fitness”, and monthly frequency.
Submit the form.
Navigate to the subscription list and /analytics.
Expected Results:
The new “Gym Membership” subscription appears in the subscription list with the correct amount and category and is included in analytics totals for its category and monthly spend.

Test Case ID: 4.1.4.3

Title: Edit Subscription Details and Propagate to Analytics
Feature/Subfeature: Subscriptions / Editing
Purpose: To ensure edits to subscription details persist and are reflected in analytics.
Initial Conditions: User is authenticated and has an existing subscription, for example “Netflix” with amount 15.99.
Test Data:
Updated Amount: 18.99
Updated Category: Entertainment – Streaming
Test Actions:
Navigate to the subscription list.
Select the “Netflix” subscription.
Edit the subscription to change the amount to 18.99 and category to “Entertainment – Streaming”.
Save the changes.
Navigate to /analytics and view category and monthly charts.
Expected Results:
The subscription now shows the updated amount and category in the list. Analytics reflect the new monthly cost and updated category totals; no stale values remain.

Test Case ID: 4.1.5.1

Title: Category Pie Chart Correctness
Feature/Subfeature: Analytics / Category Breakdown
Purpose: To ensure the category pie chart accurately reflects subscription totals by category.
Initial Conditions: User is authenticated and has multiple subscriptions across different categories.
Test Data:
Subscriptions:

Netflix – 15 (Entertainment)
Spotify – 10 (Entertainment)
Dropbox – 12 (Productivity)
Test Actions:
Log in as the user.
Navigate to /analytics.
Ensure chart mode is set to “category”.
Hover over each pie slice to view tooltips.
Expected Results:
The chart shows separate slices for “Entertainment” and “Productivity”. Entertainment total is 25, Productivity total is 12. Tooltips display values in the format “$X.XX/mo (Y.Y%)” with percentages matching the proportion of total spend.

Test Case ID: 4.1.5.2

Title: Monthly Bar Chart and Mode Toggle
Feature/Subfeature: Analytics / Monthly Totals
Purpose: To ensure the user can switch between category and monthly modes and see correct monthly totals.
Initial Conditions: User is authenticated and has subscriptions with charges spread over several months in the current year.
Test Data:
At least one subscription charge in three different months.
Test Actions:
Navigate to /analytics.
Confirm that the category pie chart is visible.
Use the UI control to switch chart mode to “monthly”.
Observe the bar chart.
Hover over one of the bars to view the tooltip.
Expected Results:
After step 3, the pie chart is replaced by a bar chart showing total subscription spend per month. Bars for months with charges display correct aggregated totals. Tooltips show amounts in the format “$X.XX”.

Test Case ID: 4.1.5.3

Title: Subscription List and “View All Subscriptions” Navigation
Feature/Subfeature: Analytics / Subscriptions List
Purpose: To ensure subscriptions are listed correctly and the “View All Subscriptions” button navigates to the full list.
Initial Conditions: User is authenticated and has at least one subscription.
Test Data: Any existing subscription set.
Test Actions:
Navigate to /analytics.

Scroll to the “Subscriptions (Cost)” list.
Verify that each subscription displays its logo, name, renewal date, monthly and yearly cost.
Click the “View All Subscriptions” button.
Expected Results:
Subscriptions in the list correspond to the user’s subscriptions with correct costs and logos.
Clicking “View All Subscriptions” navigates to the subscriptions overview page (for example, /homepage/subscriptions) showing the full set of subscriptions.

Test Case ID: 4.1.5.4

Title: Insight Text with and without Subscriptions
Feature/Subfeature: Analytics / Insight Messaging
Purpose: To ensure that the insight section displays a savings insight when subscriptions exist and a fallback message when none are present.
Initial Conditions: Application running; two scenarios:
Scenario A: User has at least one subscription.
Scenario B: User has zero subscriptions.
Test Data:
Scenario A: Multiple subscriptions including a “top” (most expensive) service.
Scenario B: No recurring subscriptions.
Test Actions:
Scenario A:
Log in as user with multiple subscriptions.
Navigate to /analytics.
Scroll to the insight section.
Scenario B:
Log in as a user with zero subscriptions.
Navigate to /analytics.
Scroll to the insight section.
Expected Results:
Scenario A: Insight text displays current monthly spend and identifies the most expensive service, including potential savings if paused.
Scenario B: Insight text displays the fallback message indicating that insights will appear once recurring subscriptions exist.

Test Case ID: 4.1.6.1

Title: Homepage Greeting Uses First Name
Feature/Subfeature: Homepage / Personalization
Purpose: To ensure the homepage greeting uses the user’s first name from userlogin.
Initial Conditions: User is authenticated, has transactions (so /homepage does not redirect), and userlogin.firstname is set (for example, “Logan”).
Test Data:
First name in userlogin: Logan

Test Actions:
Log in as the user with firstname set.
Navigate to /homepage.
Expected Results:
The dashboard loads and displays a greeting including “Logan” (or equivalent first name).

Test Case ID: 4.1.6.2

Title: Homepage Loads If Profile Query Fails
Feature/Subfeature: Homepage / Error Handling
Purpose: To ensure that the homepage still loads even if the profile query for first name fails.
Initial Conditions: User is authenticated and has transactions. The userlogin row is missing or the profile query intentionally fails in the test environment.
Test Data: None.
Test Actions:
Log in as a user whose profile query will fail (for example, user without a userlogin row).
Navigate to /homepage.
Expected Results:
The dashboard client loads successfully, possibly with a generic greeting, and no unhandled error or blank screen is shown. An error may be logged internally but not exposed as a crash to the user.

Test Case ID: 4.1.6.3

Title: Sidebar Navigation to Main Sections
Feature/Subfeature: Navigation / Sidebar
Purpose: To ensure sidebar navigation links correctly route to Overview, Subscriptions, Analytics, Account Settings, and Preferences.
Initial Conditions: User is authenticated, on /homepage, and the sidebar is visible.
Test Data: None.
Test Actions:
From /homepage, click the “Overview” link (if present).
Click “Subscriptions”.
Click “Analytics”.
Click “Account Settings”.
Click “Preferences”.
Expected Results:
Each click navigates to the expected route (for example, Overview /homepage, Subscriptions /homepage/subscriptions, Analytics /analytics, Account /accountSettings, Preferences /preferences) without errors.

Test Case ID: 4.1.7.1

Title: Subscription List from Recurring Transactions

Feature/Subfeature: Subscriptions / Listing
Purpose: To ensure recurring transactions are correctly represented in the subscription list.
Initial Conditions: User is authenticated and has multiple recurring transactions for several merchants.
Test Data:
Recurring transactions for at least three distinct merchants.
Test Actions:
Log in as the user with recurring transactions.
Navigate to the subscriptions list page.
Expected Results:
The list contains one entry per recurring subscription (per merchant or recurrence group) showing name, amount, category, and renewal date. No duplicate entries for the same recurrence group appear.

Test Case ID: 4.1.7.2

Title: Mark Subscription as Non-Recurring
Feature/Subfeature: Subscriptions / Recurrence Control
Purpose: To ensure a subscription can be marked as non-recurring and removed from lists and analytics.
Initial Conditions: User is authenticated and has at least one subscription that can be toggled to “not recurring”.
Test Data:
Subscription: Trial Service currently treated as recurring.
Test Actions:
Navigate to the subscription list.
Select the “Trial Service” subscription.
Use the UI control to mark the subscription or underlying transaction as “not recurring”.
Save changes.
Refresh the subscriptions list and /analytics.
Expected Results:
“Trial Service” no longer appears in the recurring subscriptions list and is excluded from recurring analytics totals, while original transactions remain in history if viewed elsewhere.

Test Case ID: 4.1.7.3

Title: Manage Reminders from Subscription Context
Feature/Subfeature: Subscriptions / Reminders Integration
Purpose: To ensure subscription-related reminder controls integrate with preferences.
Initial Conditions: User is authenticated and has a subscription with reminder options in the UI. Preferences page supports reminder-related settings.

Test Data:
Existing subscription with reminder toggle.

Test Actions:
Navigate to the subscription list and open a subscription with reminder options.
Enable a reminder toggle for renewal (if available).
Navigate to /preferences.
Verify that relevant reminder settings reflect the change or are consistent with subscription preferences.
Expected Results:
The reminder change made at the subscription level is stored and consistent with the user's preferences. No mismatch or lost setting occurs between the subscriptions and preferences views.

Test Case ID: 4.2.1.1

Title: Load Preferences from Userlogin and Metadata
Feature/Subfeature: Preferences / Initialization
Purpose: To ensure preferences are loaded from userlogin.prefs with fallback to user_metadata.
Initial Conditions: User is authenticated and has entries for email_notifications and weekly_summary either in userlogin.prefs or user_metadata.
Test Data:
Scenario A: userlogin.prefs contains both flags.
Scenario B: userlogin.prefs is empty; user_metadata contains flags.
Test Actions:
Scenario A:
Log in as user with prefs set in userlogin.
Navigate to /preferences.
Scenario B:
Log in as user with prefs only in user_metadata.
Navigate to /preferences.
Expected Results:
Scenario A: Toggle states reflect values from userlogin.prefs.
Scenario B: Toggle states fall back to user_metadata values when userlogin.prefs is missing those keys.

Test Case ID: 4.2.1.2

Title: Toggle Email Notifications
Feature/Subfeature: Preferences / Email Notifications
Purpose: To ensure email notification preference can be toggled and persists to auth metadata and userlogin.
Initial Conditions: User is authenticated, on /preferences, and emailNotifications is currently ON.
Test Data:
Current state: ON

New state: OFF
Test Actions:
Navigate to /preferences.
Click the control to toggle Email notifications from ON to OFF.
Save changes (if separate save action exists) or wait for confirmation message.
Refresh the page or navigate away and back to /preferences.
Expected Results:
The UI shows a success message (if implemented) and emailNotifications remains OFF after refresh. Supabase auth metadata and userlogin.prefs are updated with the new value.

Test Case ID: 4.2.1.3

Title: Toggle Weekly Summary
Feature/Subfeature: Preferences / Weekly Summary
Purpose: To ensure weekly summary preference can be toggled and stored correctly.
Initial Conditions: User is authenticated, on /preferences, and weeklySummary is currently OFF.
Test Data:
Current state: OFF
New state: ON
Test Actions:
On /preferences, locate the “Weekly summary” preference.
Toggle it from OFF to ON.
Save changes (if needed) and wait for success feedback.
Refresh /preferences.
Expected Results:
The preference remains ON after refresh, and any confirmation message indicates success. Supabase auth metadata and userlogin.prefs include weekly_summary: true.

Test Case ID: 4.2.1.4

Title: Change Theme (Light/Dark) and Persist Across Navigation
Feature/Subfeature: Preferences / Theme
Purpose: To ensure theme changes take effect immediately and persist during navigation.
Initial Conditions: User is authenticated and on /preferences. Theme is currently set to light.
Test Data:
New theme: dark
Test Actions:
On /preferences, locate appearance settings.
Click the “Dark” theme option.
Verify visual change on the current page.
Navigate to /analytics and /homepage.
Expected Results:

After step 2, the UI switches to dark mode (background and text colors changed). Theme remains dark on /analytics and /homepage, indicating that theme state persists across navigation.

Test Case ID: 4.2.1.5

Title: Reminders and Insights Buttons from Preferences
Feature/Subfeature: Preferences / Navigation Shortcuts
Purpose: To ensure the “Manage reminders” and “See insights” buttons navigate to the correct pages.
Initial Conditions: User is authenticated and on /preferences with the “Reminders & Insights” section visible.
Test Data: None.
Test Actions:
On /preferences, click “Manage reminders”.
Observe the resulting route.
Click “See insights”.
Observe the resulting route.
Expected Results:
“Manage reminders” navigates to the subscriptions management page (for example, /homepage/subscriptions). “See insights” navigates to /analytics. No errors occur.

Test Case ID: 4.2.2.1

Title: Load Account Settings Data
Feature/Subfeature: Account Settings / Initialization
Purpose: To ensure account settings load first name, email, phone, prefs, and 2FA status correctly.
Initial Conditions: User is authenticated and has a corresponding userlogin row with firstname, phone, prefs, and "2fa" fields.
Test Data:
firstname: Logan
phone: 555-555-1234
"2fa": true or false
Test Actions:
Log in as the user.
Navigate to /accountSettings.
Expected Results:
Account settings page shows correct first name, email, phone, and current 2FA status. Preferences and OTP method reflect underlying userlogin and metadata values.

Test Case ID: 4.2.2.2

Title: Enable Two-Factor Authentication with Email
Feature/Subfeature: Account Settings / 2FA Enable (Email)
Purpose: To ensure 2FA can be enabled with email codes and persisted.
Initial Conditions: User is authenticated, 2FA is currently disabled, and a valid email is present. User is on /accountSettings.
Test Data:
OTP method: email
Test Actions:
On /accountSettings, locate the 2FA section.
Ensure “Email” is selected or select it as the OTP method.
Click “Enable 2FA”.
Observe any confirmation message.
Refresh /accountSettings.
Expected Results:
2FA is shown as enabled after step 3 with a success message. After refresh, 2FA remains enabled, and auth metadata and userlogin["2fa"] reflect the enabled state with OTP method “email”.

Test Case ID: 4.2.2.3

Title: Prevent Enabling SMS 2FA Without Phone
Feature/Subfeature: Account Settings / 2FA Validation
Purpose: To ensure SMS 2FA cannot be enabled when no phone number is on file.
Initial Conditions: User is authenticated and on /accountSettings. User’s phone field is empty or null. 2FA is disabled.
Test Data:
OTP method attempt: sms
Test Actions:
On /accountSettings, select “SMS” as the OTP method.
Click “Enable 2FA”.
Expected Results:
The system does not enable 2FA via SMS and displays an error message instructing the user to add a phone number first. 2FA remains disabled.

Test Case ID: 4.2.2.4

Title: Change OTP Delivery Method
Feature/Subfeature: Account Settings / OTP Method
Purpose: To ensure the OTP delivery method can be changed between email and SMS when 2FA is enabled.
Initial Conditions: User is authenticated, 2FA is enabled, phone number is present, and an OTP method is already set. User is on /accountSettings.
Test Data:
Current method: email

New method: sms
Test Actions:
On the 2FA section, ensure 2FA is enabled.
Select “SMS” as the OTP delivery method.
Click “Save Method”.
Refresh /accountSettings.
Expected Results:
OTP method is saved as SMS, and a confirmation message appears. After refresh, SMS remains selected, and auth metadata and preferences reflect this method.

Test Case ID: 4.2.2.5

Title: Disable Two-Factor Authentication
Feature/Subfeature: Account Settings / 2FA Disable
Purpose: To ensure 2FA can be turned off cleanly.
Initial Conditions: User is authenticated, 2FA is enabled, and user is on /accountSettings.
Test Data: None.
Test Actions:
In the 2FA section, click “Disable 2FA”.
Confirm if any confirmation dialog appears.
Refresh /accountSettings.
Expected Results:
2FA status changes to disabled, and any confirmation message displays success. After refresh, 2FA remains disabled, with metadata and userlogin["2fa"] updated accordingly.

Test Case ID: 4.2.2.6

Title: Account Deletion Entry Point
Feature/Subfeature: Account Settings / Deletion
Purpose: To ensure the account deletion control is visible and behaves as designed in the current iteration.
Initial Conditions: User is authenticated and on /accountSettings. Deletion feature is present as a button.
Test Data: None.
Test Actions:
Scroll to the “Deletion” section.
Verify the presence of the “Delete account” button.
Click the “Delete account” button.
Expected Results:
The UI makes clear that deletion is permanent. Depending on implementation stage:
If fully implemented: user is logged out and account data is removed from Supabase.
If not implemented: button does not cause a crash and may show a message indicating upcoming or unsupported functionality.

Test Case ID: 4.3.1.1

Title: Privacy Page Accessible Without Authentication
Feature/Subfeature: Legal / Privacy
Purpose: To ensure the privacy policy page is publicly accessible.
Initial Conditions: Application is running. No user is logged in (no session).
Test Data: None.
Test Actions:
Open /privacy in a browser.
Expected Results:
The privacy page loads successfully, displaying privacy-related content. No redirect to /login occurs.

Test Case ID: 4.3.1.2

Title: Terms Page Accessible Without Authentication
Feature/Subfeature: Legal / Terms
Purpose: To ensure the terms of service page is publicly accessible.
Initial Conditions: Application is running with no authenticated user.
Test Data: None.
Test Actions:
Open /terms in a browser.
Expected Results:
The terms page loads successfully and displays terms content without redirecting to /login.

Test Case ID: 4.4.1.1

Title: Homepage Performance Sanity Check
Feature/Subfeature: Non-Functional / Performance
Purpose: To ensure the homepage loads within an acceptable time under normal test conditions.
Initial Conditions: Application is running in a typical test environment. User has a moderate number of subscriptions and transactions.
Test Data:
Environment with seeded data.
Test Actions:
Log in as a user with several subscriptions.
Navigate to /homepage.
Measure approximate time from navigation to visible, interactive dashboard.
Expected Results:
The dashboard loads and becomes interactive within an acceptable threshold (for example, under 3 seconds on local or staging). No obvious performance stalls or timeouts occur.

Test Case ID: 4.4.2.1

Title: Analytics Reliability on Transient Supabase Error
Feature/Subfeature: Non-Functional / Reliability
Purpose: To ensure the analytics page handles transient Supabase errors without crashing.
Initial Conditions: User is authenticated. Test environment can simulate a temporary Supabase error (for example, by using invalid query for one run or temporarily blocking one request).
Test Data:
Existing subscriptions and transactions.
Test Actions:
Simulate a transient error for the transactions query (using test tools or configuration).
Navigate to /analytics.
Expected Results:
The page does not crash. It may show an error message or empty state such as “No recurring subscriptions to display yet”, and logs the error in the console. Once the error is resolved and page is refreshed, analytics load normally.

Test Case ID: 4.4.3.1

Title: User Data Isolation Between Accounts
Feature/Subfeature: Non-Functional / Security – Data Isolation
Purpose: To ensure users cannot see other users’ subscriptions or transactions.
Initial Conditions: Two distinct users exist (User A and User B) with different subscriptions and transactions.
Test Data:
User A: Subscription Netflix
User B: Subscription Adobe Creative Cloud
Test Actions:
Log in as User A and navigate to /homepage and /analytics. Record visible subscriptions.
Log out.
Log in as User B and navigate to /homepage and /analytics.
Expected Results:
User A sees only their own subscriptions and transactions; User B sees only their own. No cross-account data appears in any lists or charts.

Test Case ID: 4.4.3.2

Title: Direct URL Access to Protected Page Without Session
Feature/Subfeature: Non-Functional / Security – Access Control
Purpose: To ensure direct URL access to protected pages is blocked for unauthenticated users.
Initial Conditions: Application running. Browser has no valid session (logged out).
Test Data: None.
Test Actions:
Attempt to load /analytics directly by typing the URL.
Attempt to load /preferences directly.

Attempt to load /accountSettings directly.
Expected Results:
Each attempt redirects the user to /login. Protected content is not visible until login is completed.

Test Case ID: 4.4.4.1

Title: Basic Accessibility – Keyboard Navigation on Login Page
Feature/Subfeature: Non-Functional / Accessibility
Purpose: To ensure primary controls on the login page are reachable via keyboard.
Initial Conditions: Application is running and no user is logged in.
Test Data: None.
Test Actions:
Navigate to /login.
Use the Tab key to move focus through the email field, password field, “Forgot Password?” button, “Login” button, and “Resend confirmation email” button.
Use Enter or Space to activate “Forgot Password?” and “Login”.
Expected Results:
All interactive controls can be focused via keyboard in a logical order, with visible focus indication. Activating “Login” submits the form; activating “Forgot Password?” triggers the reset flow.