



Trackify, Organize your finances

Sadiq Alhumood - Joseph Lee

Trackify - Smart Finance Tracker

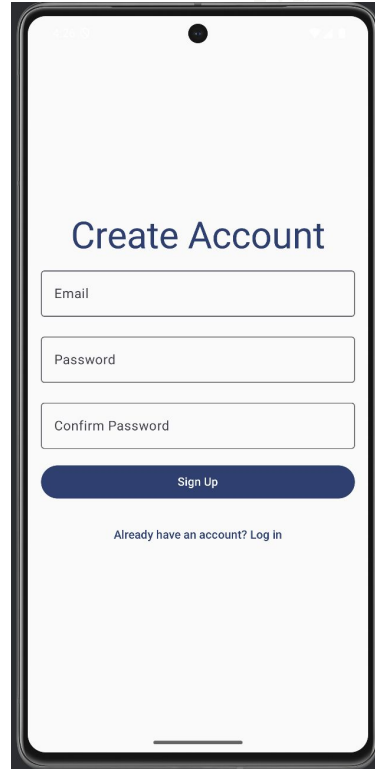
Trackify is a finance management app that helps users take control of their spending and saving habits. With features like transaction tracking and insightful charts, users can manage their finances effectively.

The Journey

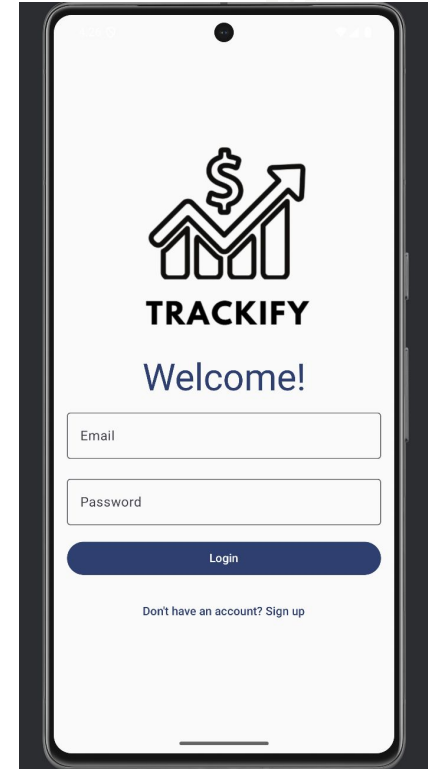
Login Page

Firestore

- Launches coroutine to perform Firestore Authentication
- On success, navigates to main screen
- Coroutines - asynchronous programming in Kotlin
- Manage background tasks

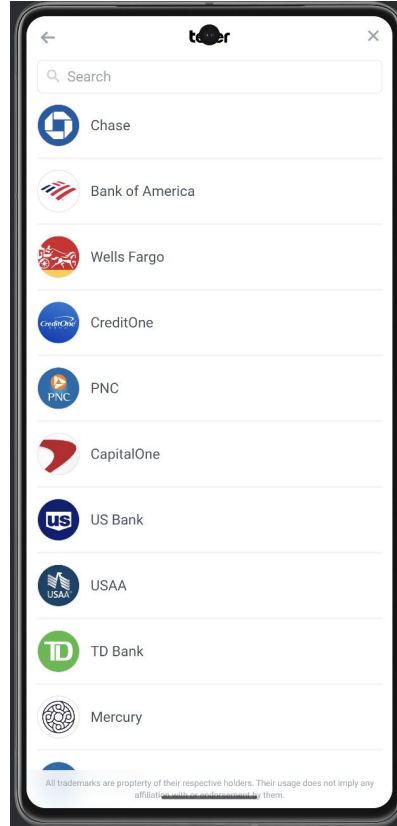
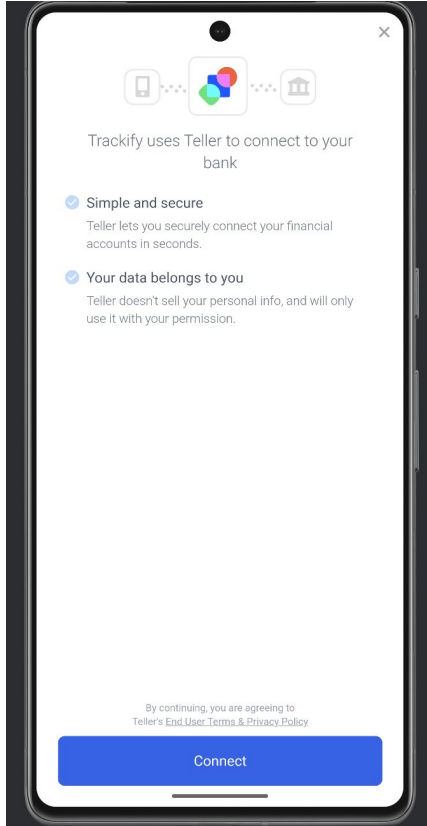


Mobile app screen showing the 'Create Account' form. The screen has a light gray background. At the top, the text 'Create Account' is displayed in a dark blue font. Below this, there are three input fields: 'Email', 'Password', and 'Confirm Password'. Each field has a light gray border and a small 'x' icon on the right. Below the input fields is a dark blue button with the text 'Sign Up' in white. At the bottom, there is a link that says 'Already have an account? Log in'.



Mobile app screen showing the 'Login' form. The screen has a light gray background. At the top, there is a logo consisting of a stylized bar chart with a dollar sign and an upward arrow. Below the logo, the text 'TRACKIFY' is displayed in a bold, dark blue font. Below this, the text 'Welcome!' is displayed in a dark blue font. Below the text, there are two input fields: 'Email' and 'Password'. Each field has a light gray border and a small 'x' icon on the right. Below the input fields is a dark blue button with the text 'Login' in white. At the bottom, there is a link that says 'Don't have an account? Sign up'.

Teller API Interface

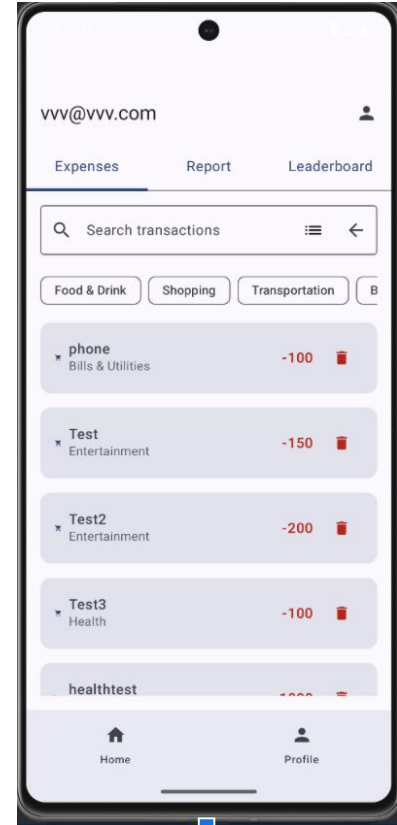


Teller Connect

- ConnectActivity class integrating Teller Connect
- Integrates Teller SDK for securely linking financial accounts
- Handles Teller Connect lifecycle events
- onSuccessRegistration - saved accessToken in shared Preferences and Firestore (with the currently authenticated user)

TransactionViewModel to Fetch Transactions

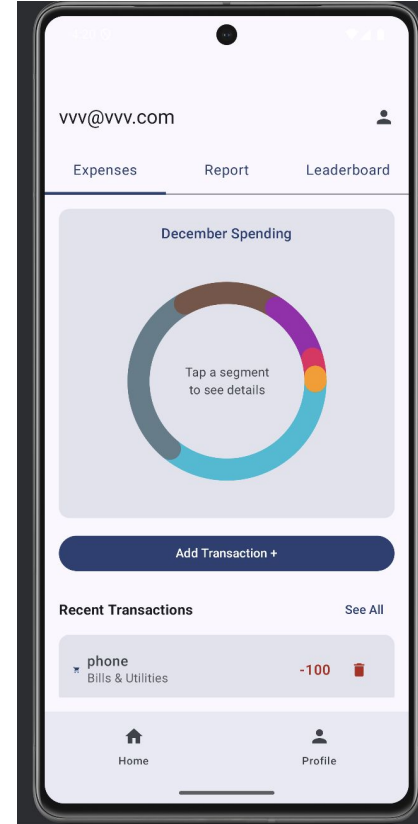
- Main feature - fetch transaction data from Teller API
- Stores locally in Room database for offline functionality
- TellerAPI Service - fetch account and transaction data
- State Management:
 - allTransactions - list of all transactions
 - recentTransactions - 3 most transactions
 - transactionTotals - income, expenses total
- Helper functions such as updateTransactionLists for latest sync to transaction data



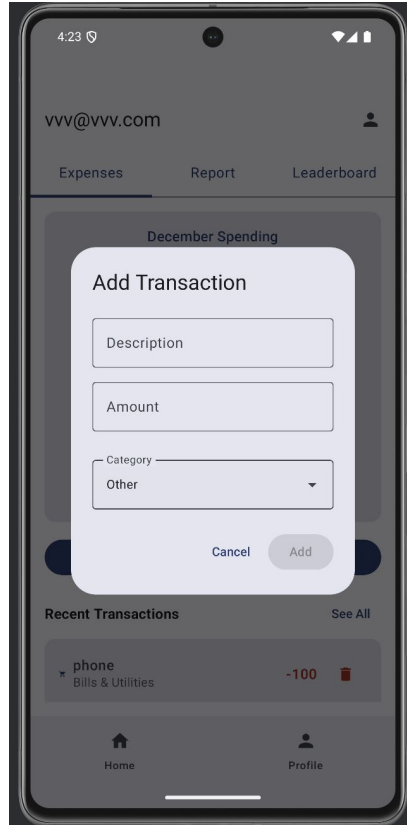
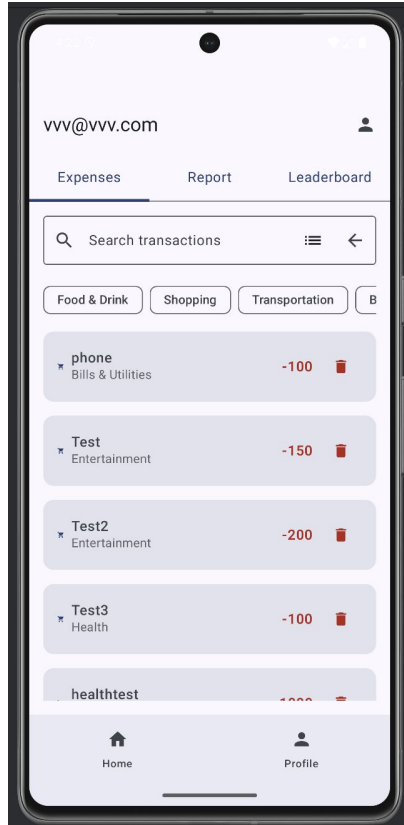
Pie Chart

Compose Canvas

- Filters transactions by the current month
- Calculates total amount spent and categorizes the spending
- Canvas iterates through categoryData to draw each segment as arc
- `pointerInput` to detect touches - upon touch, information on expenses, percentage spent on that category is displayed
- Called into the `ExpensesScreen` to display the chart



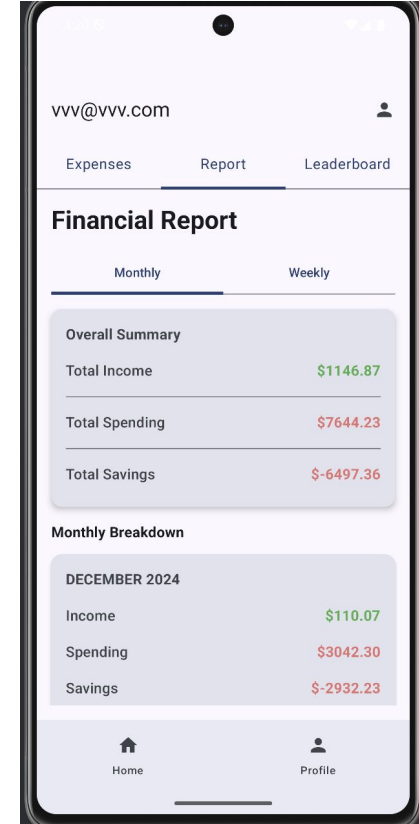
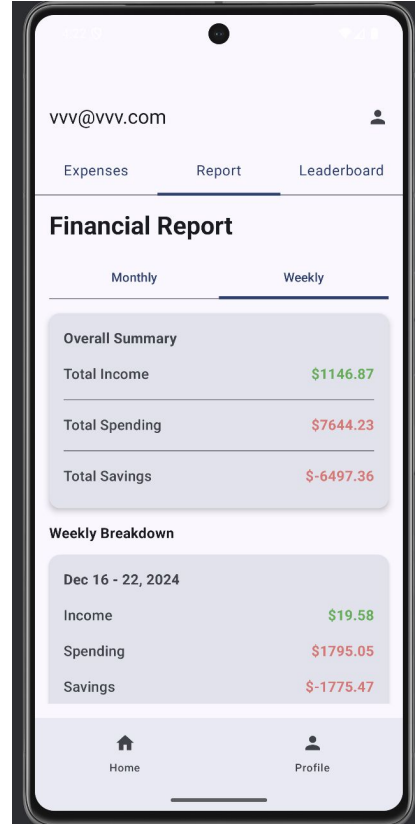
Transactions



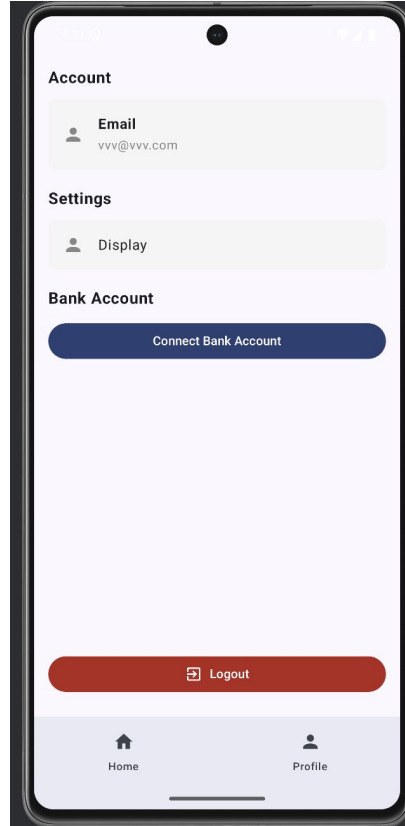
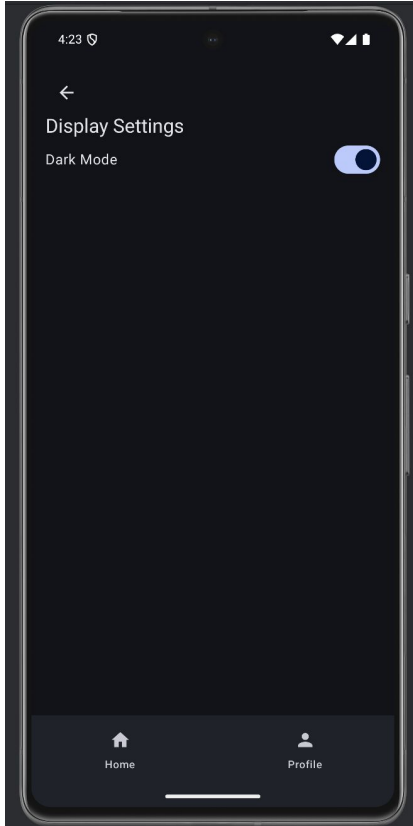
- Technologies Used: Compose, Room, ViewModel, Flow.
- All Transactions: Displays a scrollable list of transactions.
- Real-Time Updates: Merges and observes manual and backend transactions.
- Add Transaction Button: Opens a dialog to capture and save transactions.

Report Page

- Dynamic Period Selection: Users can toggle between weekly and monthly financial reports.
- Visual Financial Summary: Highlights total income, spending, and savings with color-coded UI.
- Data Source: Transactions are fetched from Room Database and computed live.
- Composable Components: Leverages LazyColumn, TabRow, and Card for interactive and modern UI.



Profile



- Profile Management:
- Display user email and account status.
- Connect bank account via external activity.
- logout with navigation back to the login page.
- Dark Mode Settings:
- Toggle between light and dark themes.
- Persistent theme preference stored using SharedPreferences.

Demo Time