CSE305 Software Engineering – Sprint 1 Report

Student: Başar Orhanbulucu
Student ID: 221805031
Project: My Budget Flow (Budget & Cash Flow Manager)
Sprint: #1 (Foundation & App Structure)

1. Plan for Sprint 1 #1

This section summarizes the Sprint 1 goals and their current status, as defined in the "Agile Process Model" document.

- Sprint Goal: To create a functional foundation where users can securely register, login, reset passwords, and navigate through the application skeleton (navigation, localization).
- Date Range: Nov 24 - Dec 14
- Status: ✅ Completed

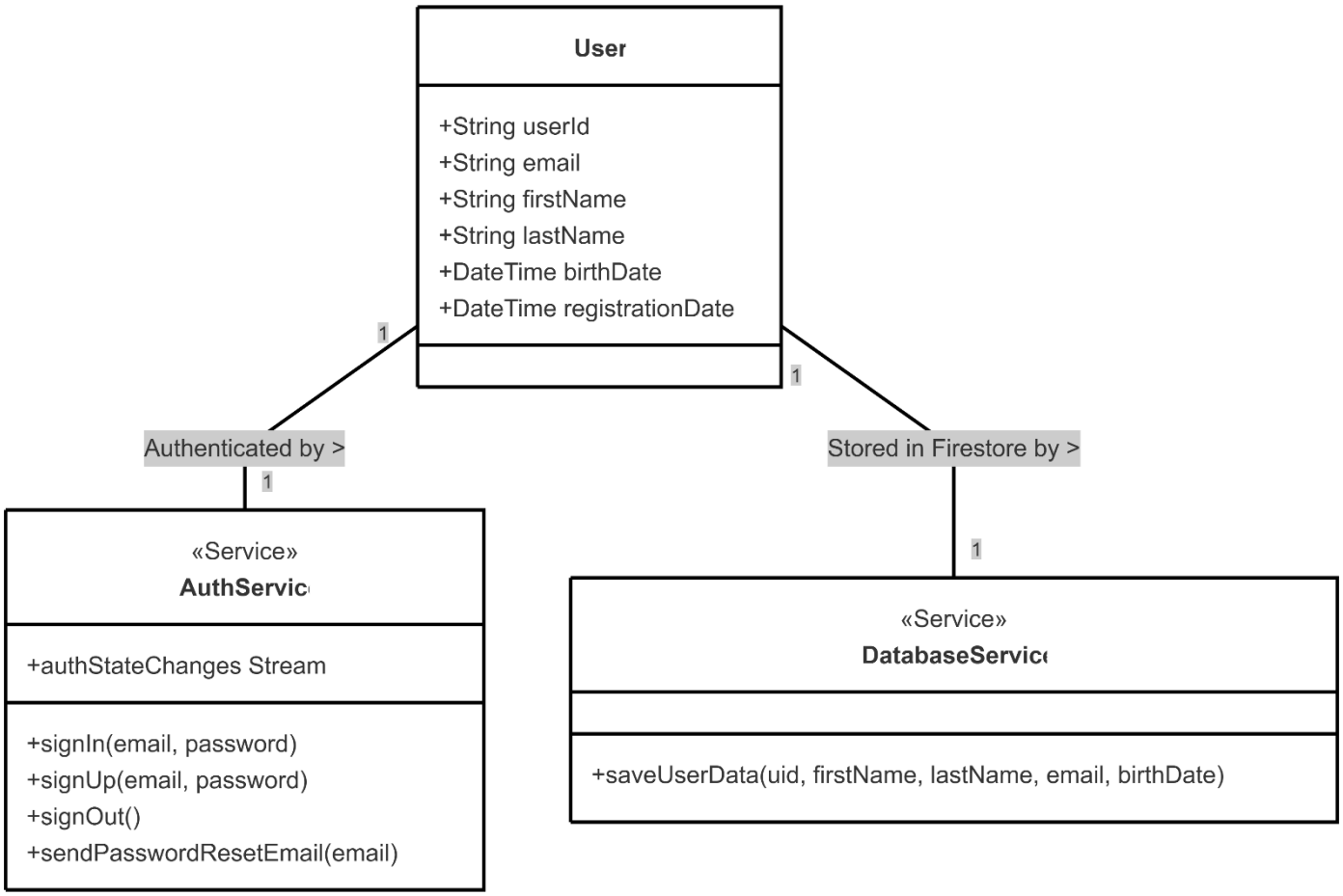
Sprint Backlog Scope:

User Story	Task	Description	Status
US 1.1	User Authentication	Firebase Auth setup, Login/Register/Forgot Password screens.	Done
US 2.1	App Architecture	Implementation of BottomNavigationBar and page routing.	Done
US 2.3	Localization	Infrastructure for TR/EN support using AppLocalizations and Riverpod.	Done
Tech	Database Setup	Coding the DatabaseService for the Firestore 'users' collection structure.	Done
UI	Auxiliary Screens	Preparation of placeholder screens for Profile, Settings, and Notifications.	Done

2. Domain Class Diagram (Sprint 1 Coverage)

The primary data entity focused on in Sprint 1 is the User class. Since entities like Transaction and Goal are planned for Sprint 2, only the structures currently coded and active are shown in this diagram.

This diagram is based on the logic implemented in auth_service.dart and database_service.dart.

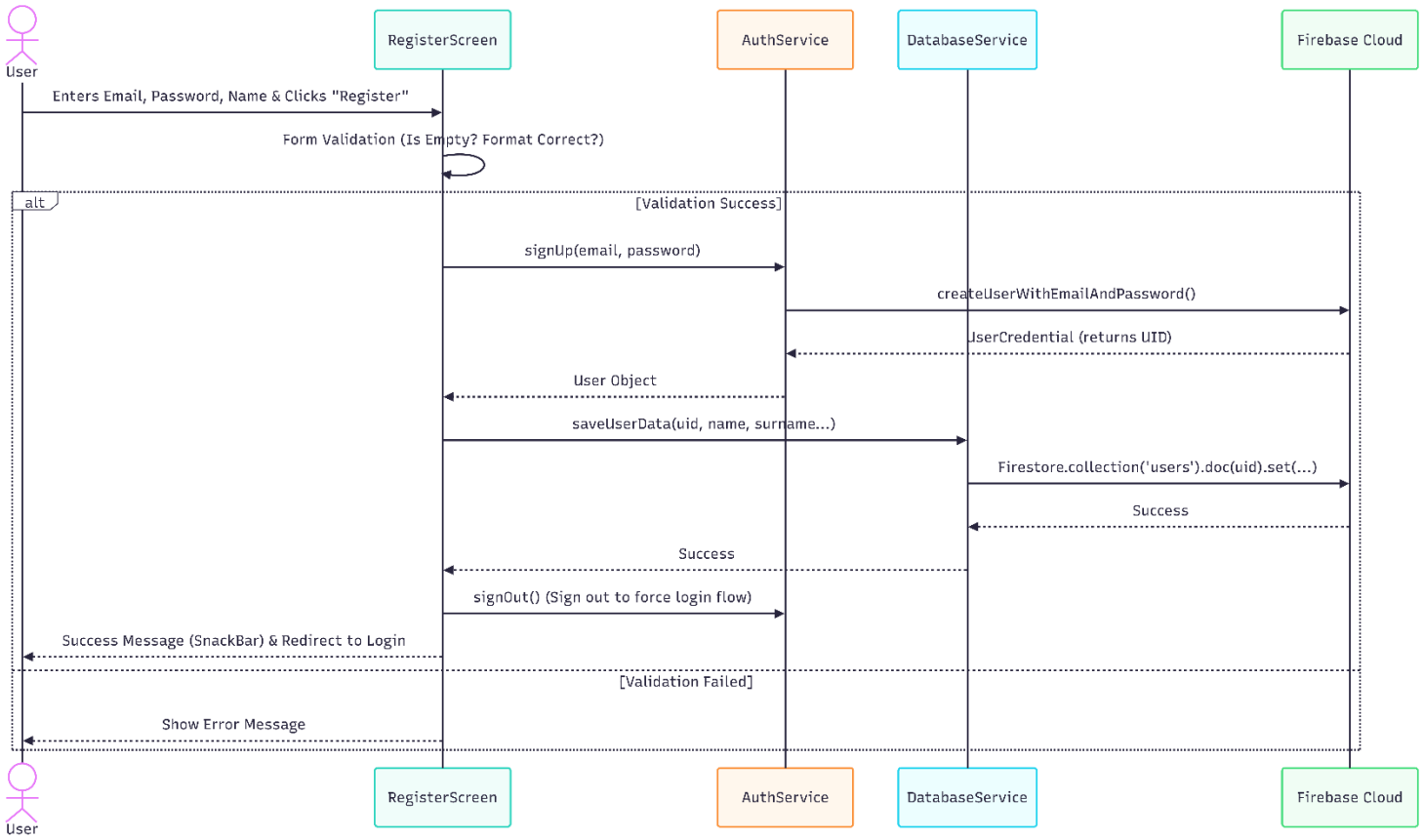


3. Sequence Diagrams

Below are the technical sequence diagrams for the three most critical operations in Sprint 1: Registration, Login, and Language Switching.

Scenario 1: User Registration Flow

The process of a user registering via RegisterScreen and writing data to both Auth and Firestore.



Actors and Components:

- User: The person interacting with the application.
- RegisterScreen: The UI component where the user inputs data.
- AuthService: The middleware handling authentication logic.
- DatabaseService: The middleware handling database operations.
- Firebase Cloud: The backend service (Auth and Firestore).

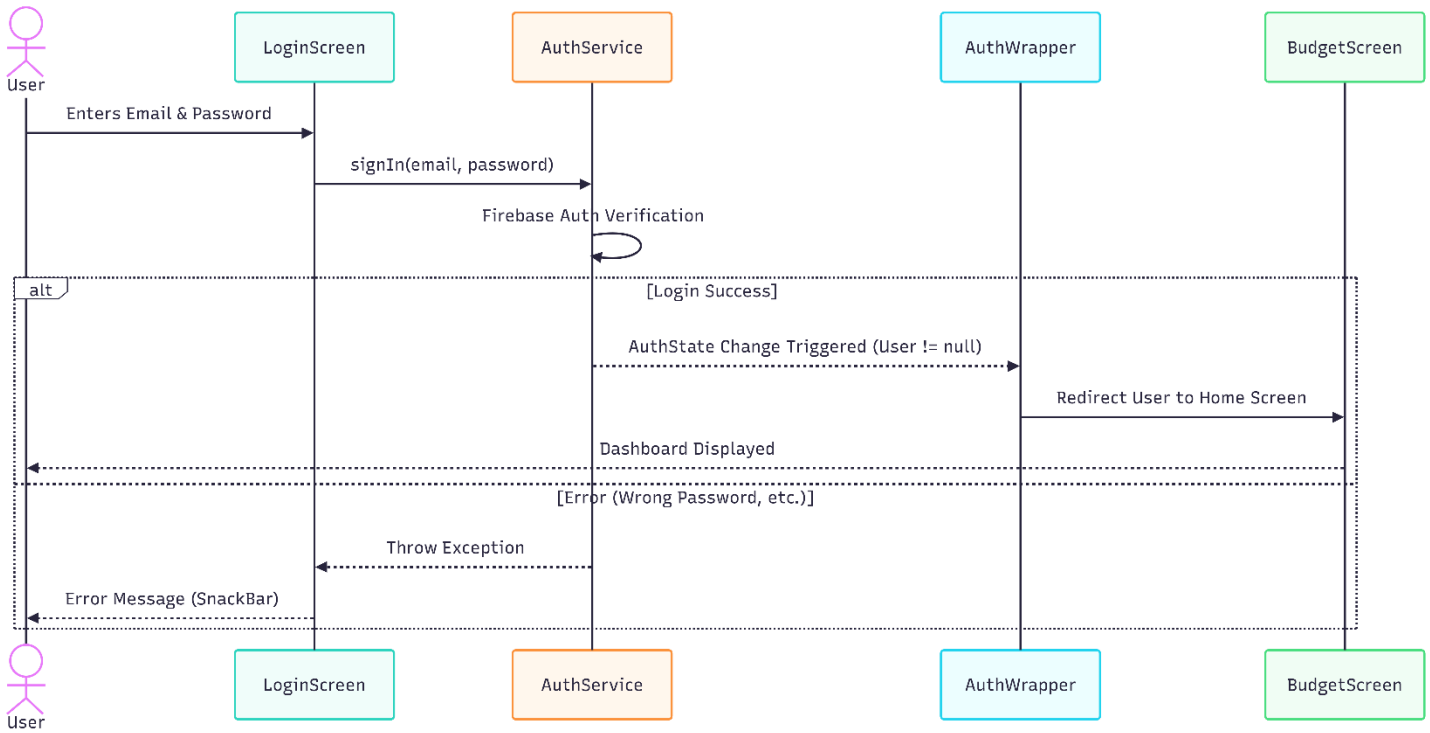
Step-by-Step Process:

1. Input: The User enters Email, Password, and Name, then clicks "Register".
2. Validation: The RegisterScreen performs a local check to see if the fields are empty or if the format is correct.

3. Decision Block (Alt):
    - If Validation Fails: An error message is shown directly to the User, and the process stops.
    - If Validation Succeeds: The following steps proceed:
4. Create Identity: RegisterScreen calls the signUp function in AuthService.
5. Firebase Request: AuthService calls createUserWithEmailAndPassword on the Firebase Cloud.
6. Return UID: Firebase confirms the creation and returns a UserCredential object containing the User ID (UID).
7. Initiate Data Save: RegisterScreen receives the user object and calls saveUserData in DatabaseService.
8. Firestore Write: DatabaseService saves the user details (name, surname, etc.) to the 'users' collection in Firestore using the specific UID (set(...)).
9. Success Response: Firebase returns a success signal, which propagates back to DatabaseService and then to RegisterScreen.
10. Sign Out (Force Flow): RegisterScreen calls signOut() on AuthService to ensure the user goes through the formal login flow initiated from the login screen.
11. Conclusion: A success message (SnackBar) is displayed to the User, and they are redirected to the Login screen.

Scenario 2: User Login Flow

The process of a user logging in via LoginScreen and being redirected to the main page via AuthWrapper.



Actors and Components:

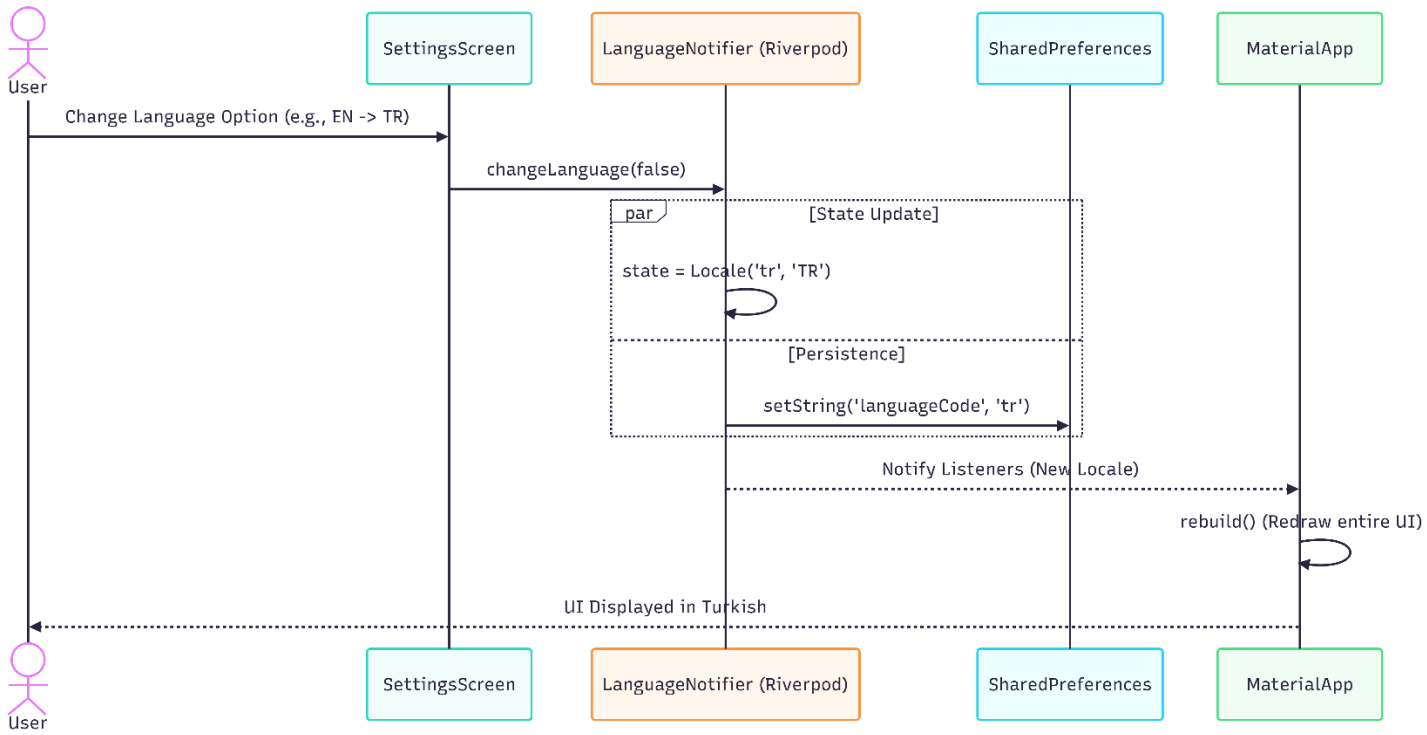
- User: The person attempting to log in.
- LoginScreen: The UI interface where the login form is presented.
- AuthService: The service managing login logic and Firebase communication.
- AuthWrapper: A wrapper component that listens to the authentication state (logged in vs. logged out) and handles navigation.
- BudgetScreen: The main screen (Dashboard) displayed upon successful login.

Step-by-Step Process:

1. Data Entry: The User enters their Email and Password.
2. Login Request: LoginScreen triggers the signIn function in AuthService using the provided credentials.
3. Verification: AuthService performs an internal Firebase Auth Verification.
4. Decision Block (Alt):
    - If Login is Successful [Login Success]:
        - AuthService triggers an AuthState Change, indicating that the user object is no longer null (User != null).
        - The AuthWrapper, listening for this change, detects that the user has logged in.
        - The AuthWrapper automatically redirects the User to the BudgetScreen (Home Screen).
        - The Dashboard is displayed to the User.
    - If an Error Occurs [Error - Wrong Password, etc.]:
        - AuthService throws an exception back to the caller.
        - LoginScreen catches this exception.
        - An error message (SnackBar) is displayed to the User informing them of the failure.

Scenario 3: Language Switch Flow

Dynamic language switching of the application using LanguageProvider (Riverpod).



Actors and Components:

- User: The person changing the settings.
- SettingsScreen: The UI screen where the language option is located.
- LanguageNotifier (Riverpod): The logic layer that holds and manages the application's language state.
- SharedPreferences: A simple storage mechanism for saving data persistently on the device.
- MaterialApp: The root widget of the application that encompasses the entire UI.

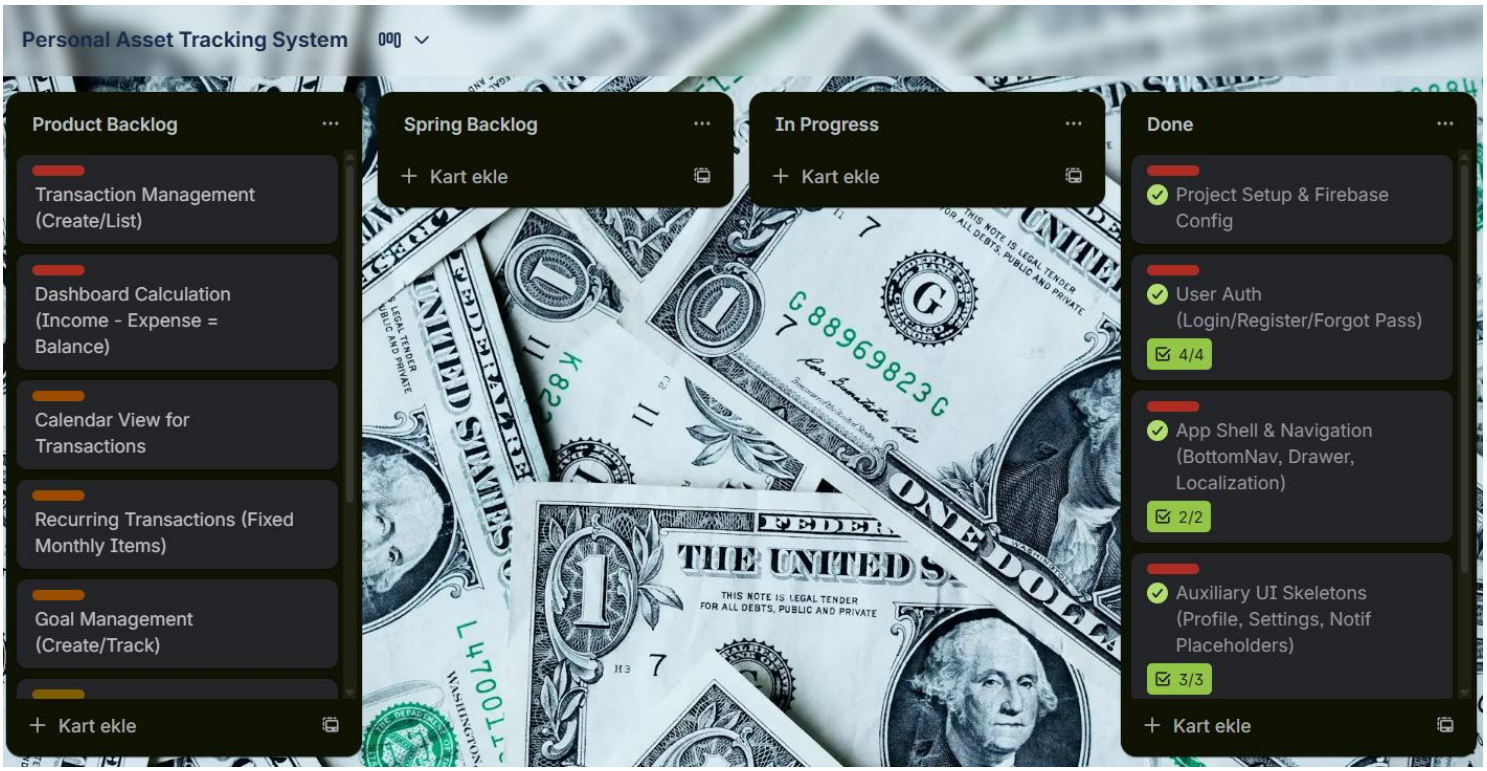
Step-by-Step Process:

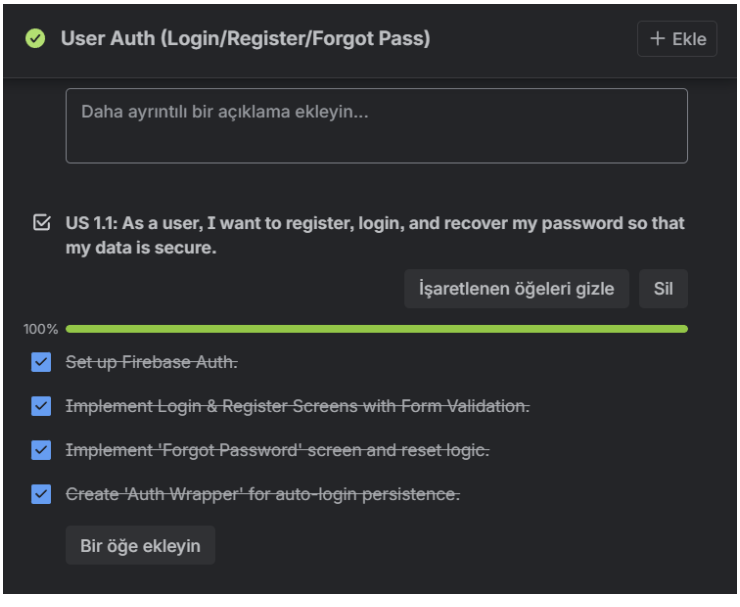
1. Trigger: The User changes the language option (e.g., from EN to TR) on the SettingsScreen.
2. Function Call: The SettingsScreen calls the changeLanguage function within the LanguageNotifier logic layer.
3. Parallel Execution (par): The par block in the diagram indicates that the following two operations happen concurrently:
    - State Update: The LanguageNotifier updates its internal state to the Turkish locale (Locale('tr', 'TR')). This is the in-memory change.

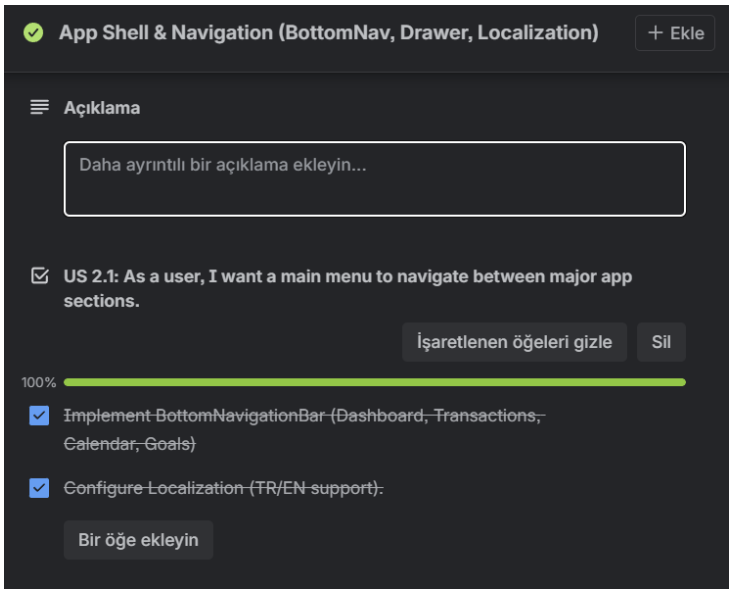
- Persistence: Simultaneously, to remember the preference after a restart, it saves the value 'tr' to SharedPreferences under the key 'languageCode' (setString).

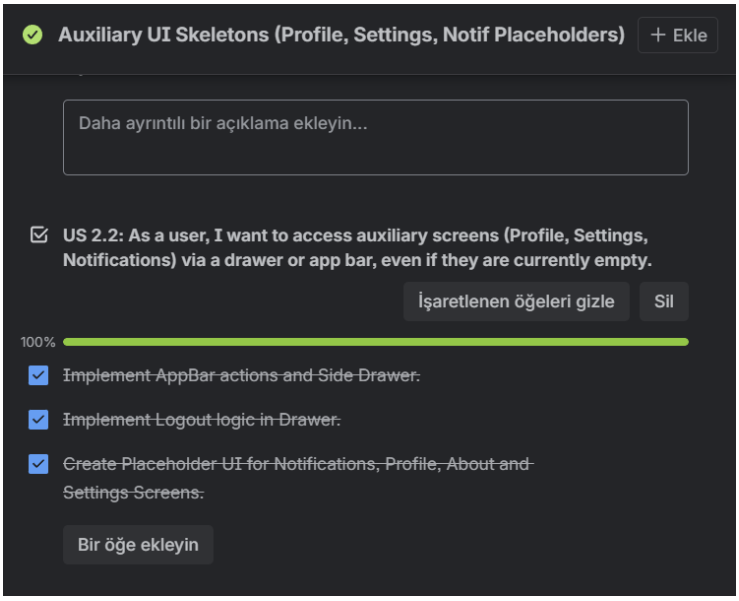
4. Notification: After the state change, LanguageNotifier sends a signal to its listeners (Notify Listeners) with the new Locale information.
5. Rebuild: Receiving this notification, the root MaterialApp triggers a rebuild(), redrawing the entire User Interface (UI) from scratch.
6. Result: All text elements in the UI are updated and displayed to the User in Turkish.

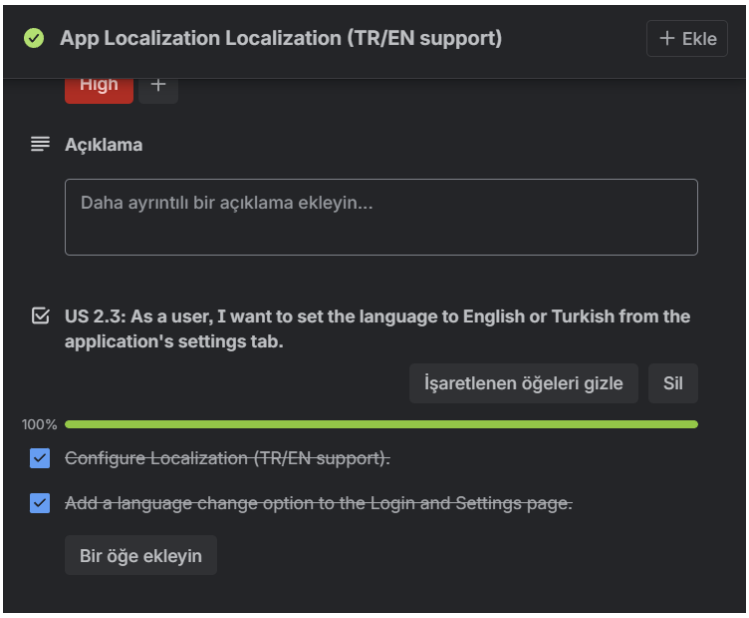
Trello Udpates:





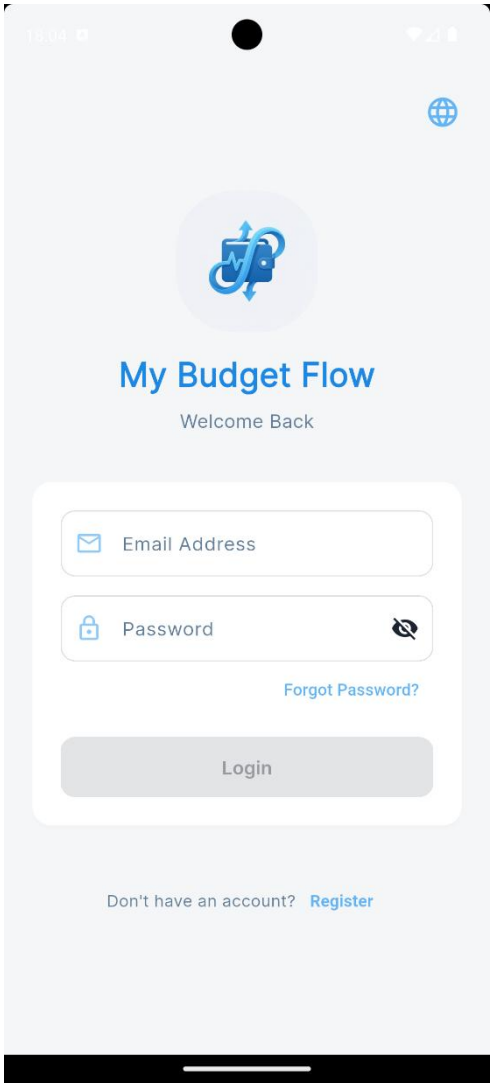


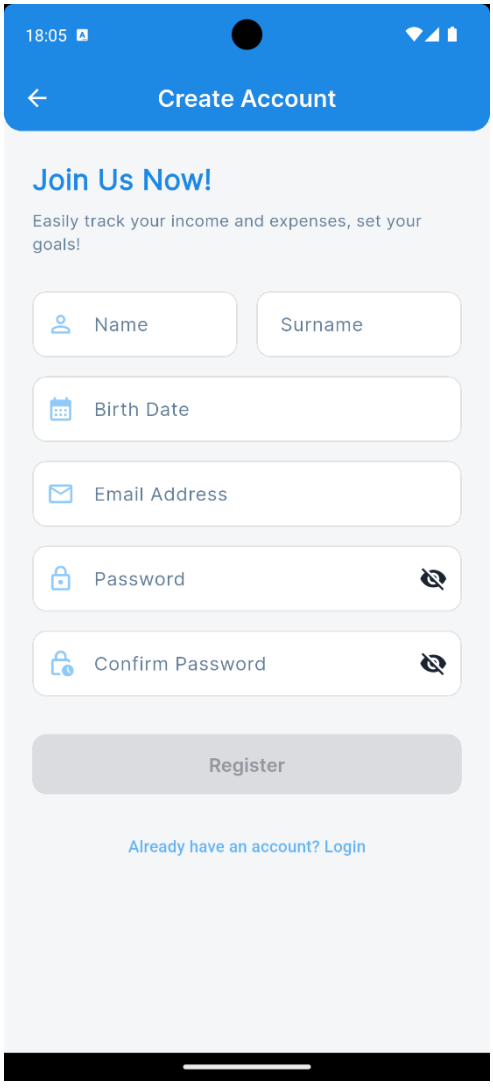


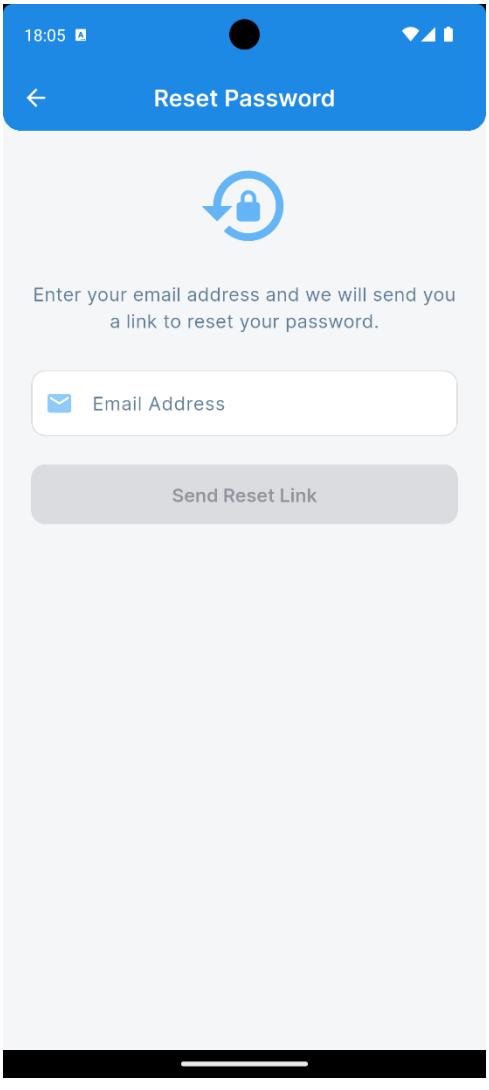


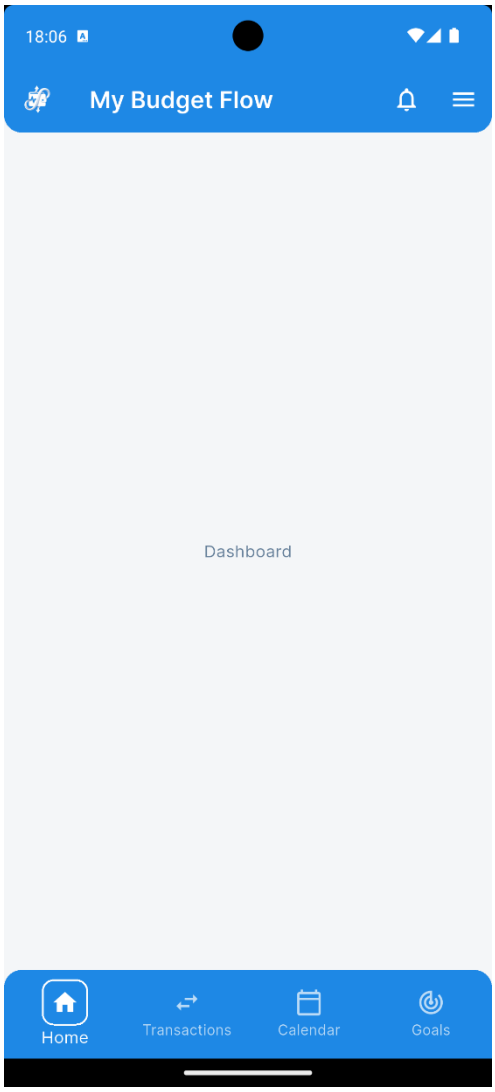
4. PSI (Potentially Shippable Increment)

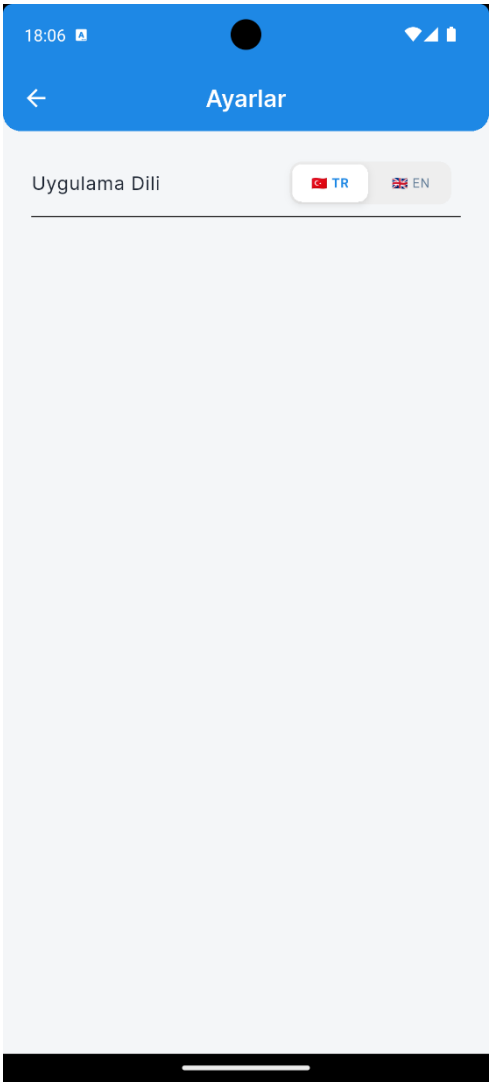
- Target for Sprint 1: A functional "App Shell". Users can securely login/register, reset passwords, and navigate through the complete UI skeleton (Dashboard, Settings, Profile, etc.) without crashes. The auxiliary pages (Profile, Notifications) are visible but static.

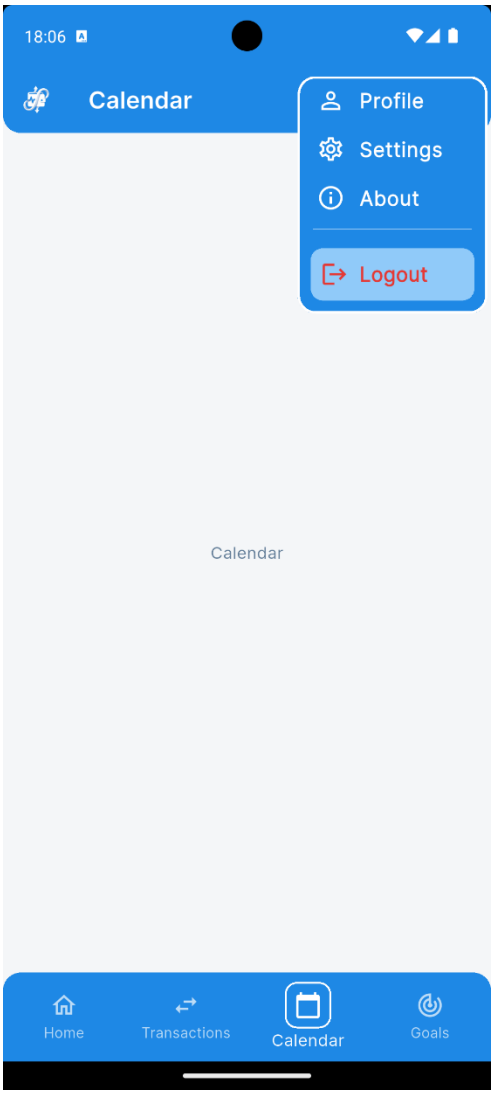







18:06
My Budget Flow
Dashboard
Home
Transactions
Calendar
Goals


18:06
Ayarlar
Uygulama Dili
TR
EN


18:06
Calendar
Profile
Settings
About
Logout
Calendar
Home
Transactions
Calendar
Goals