Kuripot Chronicles

Version 1.0

Date: May 3, 2025

Name	Roles	Contact Information
James Rio L. Abaquita	Lead Programmer	2201103767@student.buksu.edu.ph
Vivian J. Bangcoyo	Documentary/QA	2201103762@student.buksu.edu.ph
Sedrick James P. Camiguing	Project Leader	2201103327@student.buksu.edu.ph

Table of Contents

1. Document Overview

1.1 Scope

This document explains the different parts of the KURIPOT CHRONICLES, a mobile budgeting android application tailored for students to help them track expenses, set saving goals, and manage finances efficiently. The mobile app will make their budgeting experience clear with visualize spending patterns.

Functional Scope:

- Tailored Budgeting for students (School expenses, rent, food, etc.)
- Visualize spending patterns with charts
- Savings goals with automatic progress tracking

Technical Scope:

- Usage of secure authentication methods such as Google OAuth
- Integration of API for functionality and flexibility
- Implementation of robust database solutions for online data management

Design Scope:

- Dynamic and intuitive user-centered design for effective budgeting experience.
- Responsive layout to be compatible for different mobile devices.

1. 2 Audience

This document is intended for everyone involved in the creation and development of the KURIPOT CHRONICLES mobile app.

- **Developers –** android app developers, backend developers, database managers that are involved in creating, developing, and maintaining the application.
- **Designers -** UX/UI designers responsible for creating a dynamic, intuitive, and user-centered interface.
- **Testers -** Quality assurance professionals tasked with verifying the app's performance, security, and functionality.
- **Stakeholders -** Project sponsors, potential investors, academic supervisors, and future collaborators interested in project goals, progress, and outcomes.

2. Project Overview

2.1 Executive Summary

This project aims to create an android mobile app for students by providing a personalized, efficient, and intuitive budgeting experience. By integrating multiple APIs and robust database solutions, Kuripot Chronicles will allow students to manage and track their money efficiently.

Business Goals:

- Financial Literacy: Empower students to manage their finances proactively.

Technical Goals:

- Real-time Sync
- Secure user authentication using Google OAuth API
- Ensure app and data security by using best security practices
- Build robust and reliable online data management

2.2 Problem Statement

Student struggle with financial management due to their irregular incomes, impulsive spending, and a lack of user-friendly budgeting tools tailored for their needs, leading to stress and poor savings habits; Kuripot Chronicles solves this by providing a simple, student-focused mobile app with expenses tracking, saving goals, real-time spending visuals to promote financial literacy and discipline.

User needs:

1. Simple expenses logging to avoid complexity in tracking expenses
2. Visual spending analytics (charts/graphs) to present the data in a more understandable method
3. Savings goal with progress tracking

Market Analysis Summary:

1. **Target Users**: College/university students (ages 18-24) with limited budgets
2. **Competitive Gap**: Existing apps lack student-specific features like school expenses categories.

3. Functional Specification

The Kuripot Chronicles Mobile app is tailored to help students handle, manage, and track their expenses efficiently by providing real-time expenses tracking and visualize spending patterns. The main functions, user interfaces, and use cases are described below to guarantee effective resource management and student convenience.

Feature List:

Feature Name	Description
Visualized Spending Patterns	Spending patters will be presented as charts/graphs to better identify the spending habits.
Log expenses by categories	Expenses will be logged by different categories including student-specific categories like tuition and dorm rent.
Set savings goal	Set savings for a time period and see it you reached it, then you can compare it to your previous goals.

User Stories and Requirements

User Story:

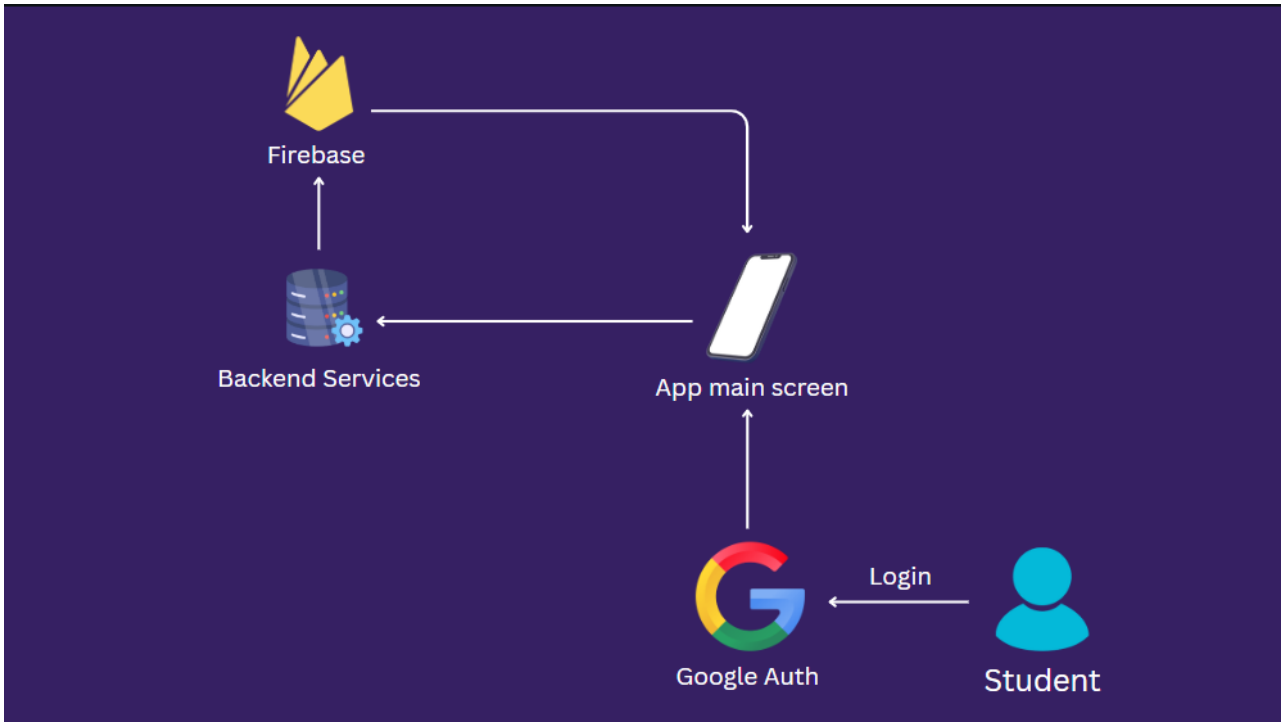
As a student, I want to see and track my weekly spending trends.

Requirements

- The allowance value should be deducted automatically after logging the expenses.
- The values of spending patters should be display clearly in the charts.

4. Technical Specifications

4.1 Architecture Overview



4.2 Platform-Specific Considerations

Android Guidelines

- Navigation: Botton navigation bar for easy navigation of main sections (Home, Budget page, Expenses Page, Profile)
- Back Button: regular back button with proper redirection
- Permissions:
    - Internet access for fetching data from online database (Firebase).
- Material Design: The app follows Material Design principles for consistent UI/UX

Android Requirements

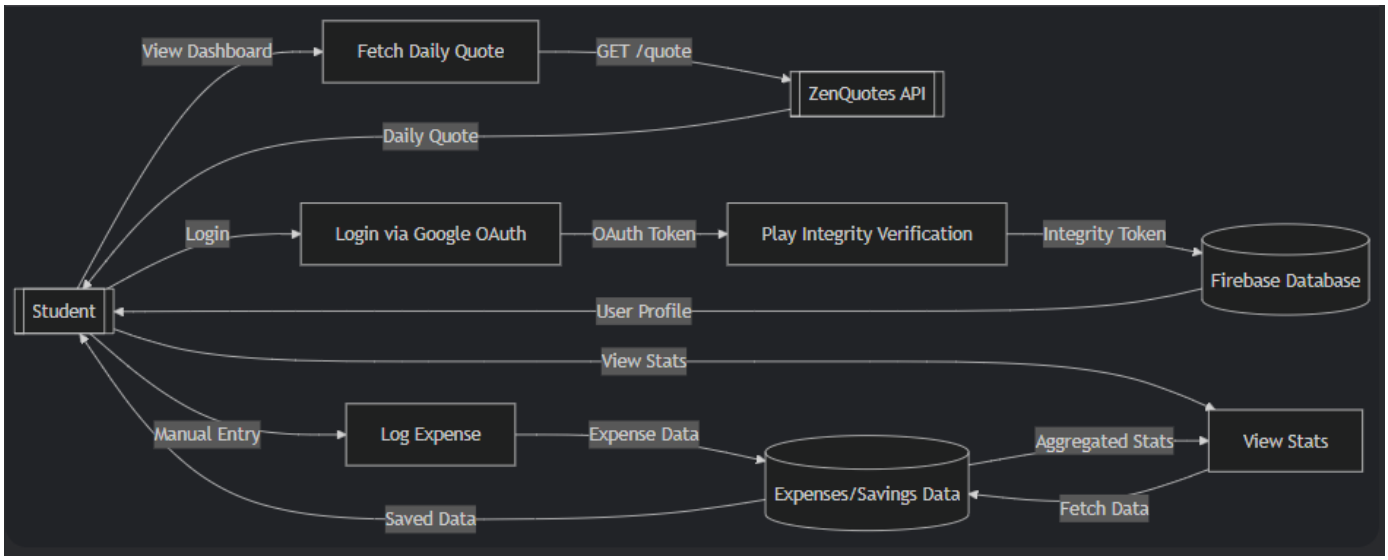
- Minimum Android Version: Android 6.0 (API level 23) or higher
- Target Android Version: Android 14 or higher
- Device Requirements:
    - 2GB RAM minimum

- 30 MB free storage space
- Internet connectivity (WiFi or cellular data)

- Screen Support:
  - Responsive to various screen sizes (Phone, Tablet, Pad)

4.3 Data Management

Dataflow Diagram:



Dataflow diagrams illustrate how data moves through the system:

- External Entities: The student interacts with the app (e.g., logging expenses).
- Processes: Includes actions like *Login via Google OAuth*, *Verify Device Integrity (Play Integrity)*, and *Save to Firebase*.
- Data Stores: Firebase Database stores expenses, user profiles, and aggregated statistics.
- Flows: Data travels from the student to Firebase via secure APIs and returns as visual charts or transaction lists.

The app uses Firebase Realtime Database, a NoSQL cloud database, to store and manage all user data. It is structured into collections for scalability and efficient querying:

- /users: Stores user profiles, including Google OAuth ID, email, currency preferences, and account creation timestamps.

- /expenses: Records transaction details such as amount, category (e.g., Food, Transport), payment method, and timestamps.

- /goals: Tracks savings goals with fields for target amount, current progress, deadlines, and status (Active/Completed).
- /quotes: Caches daily motivational quotes fetched from ZenQuotes to reduce API calls.

Key Features:

- Real-Time Syncing: Automatically updates data across devices (e.g., expense additions reflect instantly).
- Encryption: Data encrypted at rest (AES-256) and in transit (TLS 1.3).
- Scalability: Handles high read/write loads for student users globally.

API endpoints

The app interacts with external services and its own backend via RESTful APIs:

- **Authentication**:
    - POST /auth/google: Authenticates users via Google OAuth and returns a session token.
    - POST /integrity-check: Validates device integrity using Play Integrity API.
- **Expenses**:
    - POST /expenses: Logs a new expense (requires user ID, amount, category).
    - GET /expenses/{userId}?month=2025-04: Fetches expenses for a specific month.
- **Goals**:
    - PUT /goals/{goalId}: Updates savings goal progress.
- **Quotes**:
    - GET /quotes/daily: Fetches a daily budgeting quote from ZenQuotes.

Security and Privacy

- Sign-in Systems

The app employs Google OAuth 2.0 for secure, passwordless authentication, allowing users to sign in directly with their Google accounts. This eliminates risks associated with password storage and phishing, as Google handles credential validation. Additionally, the Play Integrity API ensures the app runs only on trusted devices by checking for root access, tampering, or malicious software. This dual-layer verification blocks unauthorized users and safeguards against account takeover attempts, ensuring that only legitimate students access the platform.

- Secure Connection

All data exchanges between the app and backend services are encrypted using HTTPS with TLS 1.3, protecting against eavesdropping and man-in-the-middle attacks. Firebase App Check further secures the connection by verifying requests originate from the genuine app instance, preventing API abuse from bots or counterfeit apps. This ensures that even if credentials are compromised, attackers cannot intercept or manipulate data during transmission.

- Data Protection

Sensitive data, such as transaction details and user profiles, is encrypted at rest using AES-256 encryption in Firebase. Field-level security rules restrict database access: for example, users can only read/write their own expense records, and admins cannot access personal data. Data retention policies automatically purge inactive accounts after 24 months, minimizing exposure of stale information. Backups are encrypted and stored in geographically redundant locations to prevent data loss.

- Privacy Policy

The app adheres to GDPR and CCPA regulations, granting users full control over their data. A transparent privacy policy outlines what information is collected (e.g., email for sign-in, expenses for tracking) and how it is used. Users can request data exports, corrections, or deletions via in-app tools, and the app avoids collecting unnecessary metadata (e.g., location is optional). Third-party services like ZenQuotes are vetted to ensure they comply with

privacy standards, and data-sharing agreements prohibit selling user information.

Third-Party Integration

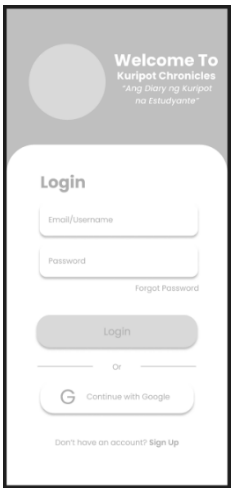
The app integrates several third-party services to enhance functionality, security, and user engagement while minimizing development overhead:

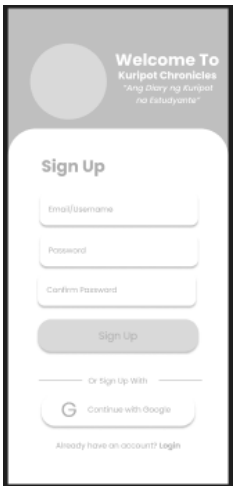
- Google OAuth 2.0:
    - Streamlines user authentication by allowing students to sign in securely with their Google accounts, eliminating password management risks.
    - Reduces sign-up friction, increasing user adoption rates.
- Play Integrity API:
    - Validates device authenticity to block root/jailbroken devices or emulators, preventing fraud and unauthorized access.
    - Ensures the app operates in a secure environment, complementing OAuth for robust security.
- Firebase:
    - Provides backend infrastructure for real-time database storage (expenses, goals), authentication, and serverless functions.
    - Enables seamless data syncing across devices and offline access support.
- ZenQuotes API:
    - Delivers daily motivational quotes (e.g., "*Save pennies for rainy days*") to encourage financial discipline.
    - Quotes are cached in Firebase to reduce API calls and ensure offline availability.

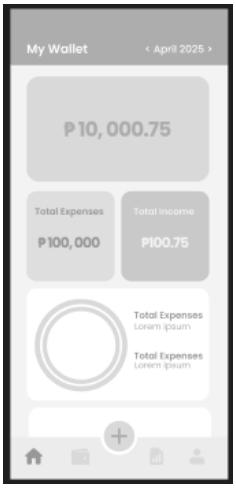
5. UI/UX Design Specifications

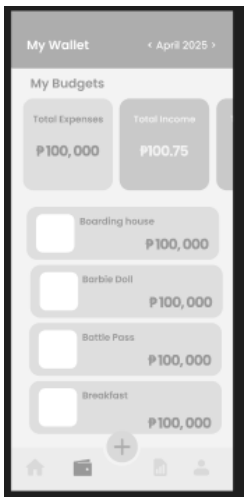
5.1 Wireframes & Mockups

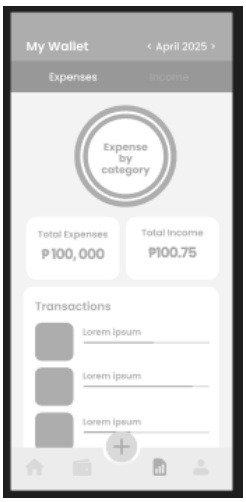
Screen Overview

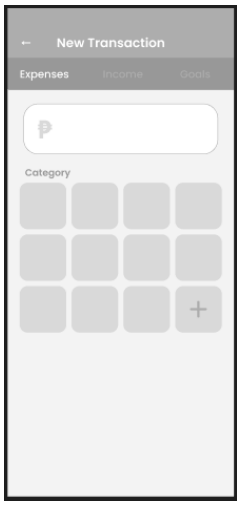


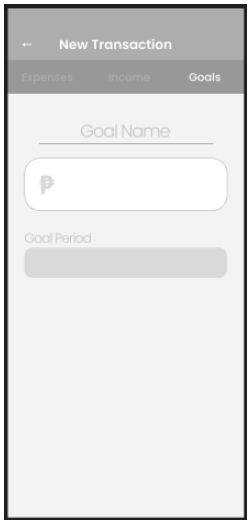




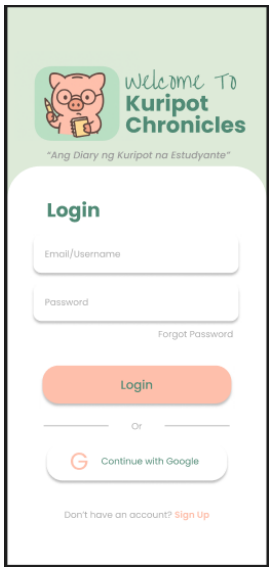


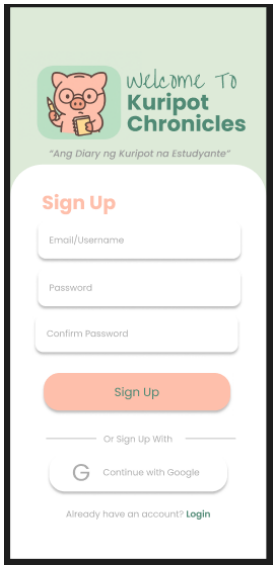


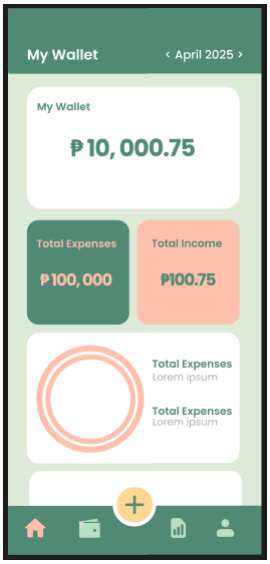


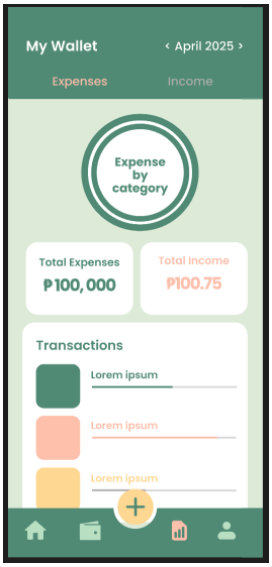


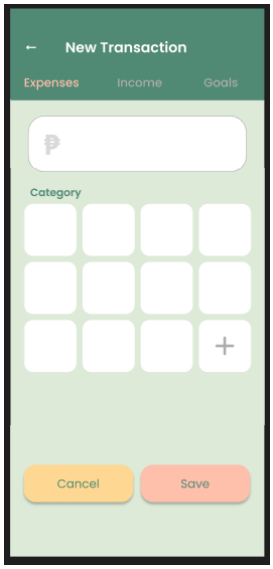
High Fidelity Designs

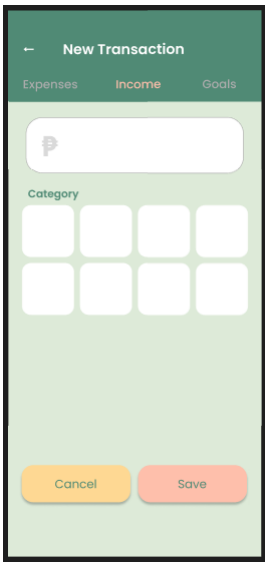


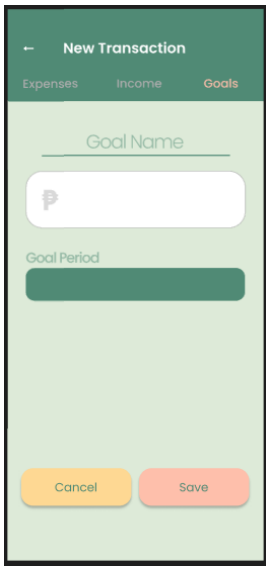










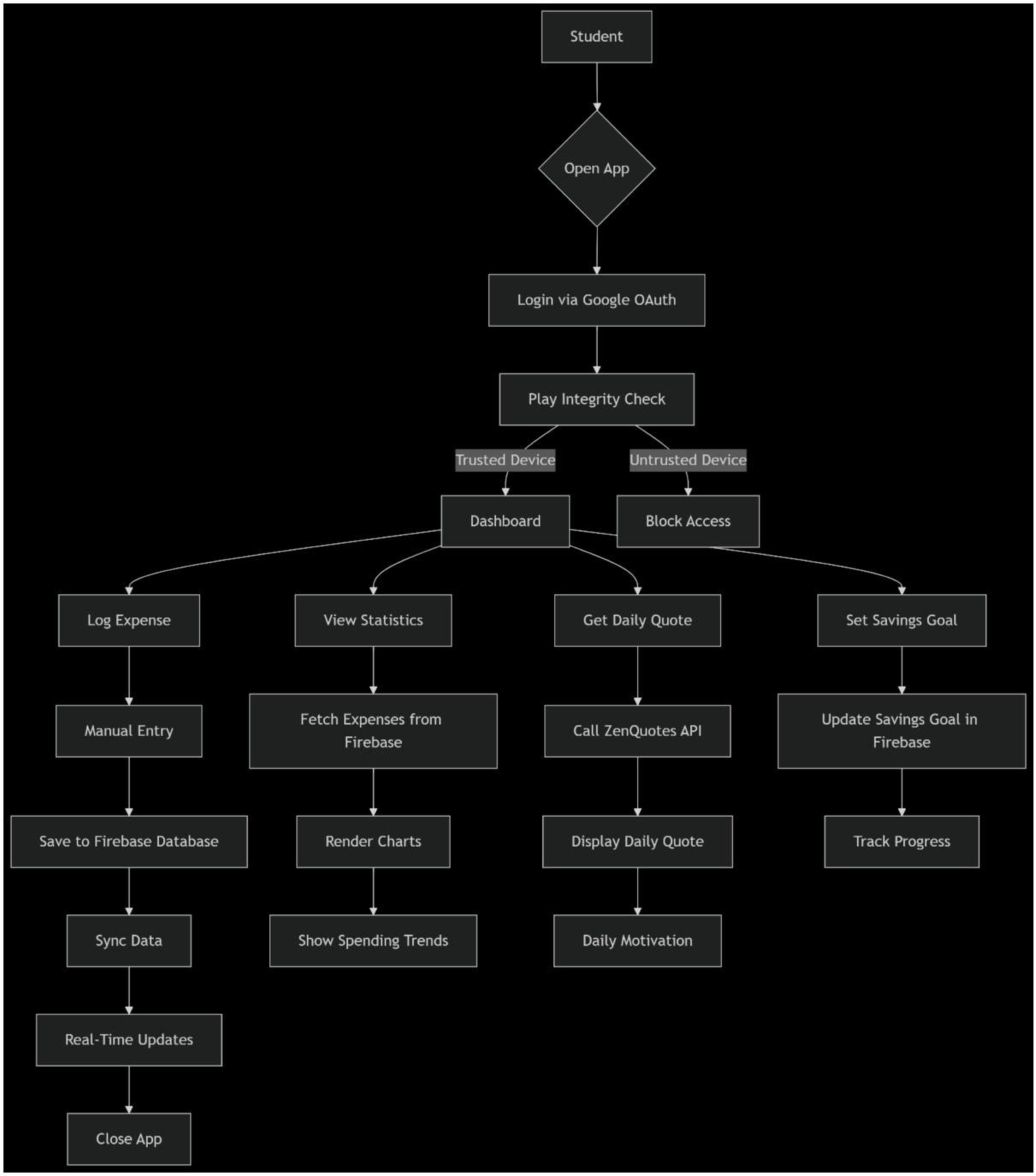


The presented designs and frames consists of the pages of Kuripot Chronicles. The wireframes for Kuripot Chronicles depict a streamlined budgeting app designed for students, featuring a dashboard-centric interface. The design prioritizes simplicity with bold financial metrics and collapsible sections with visual aids like charts or icons, which could enhance usability for student-specific needs such as tracking tuition or part-time income.

5.2 Navigation & Flow

Flowcharts map step-by-step user interactions and system logic:

- Start: Student opens the app and logs in via Google OAuth.
- Authentication: Play Integrity checks device trustworthiness.
- Core Actions: After authentication, the student can log expenses, view statistics (via Firebase data), or fetch daily quotes (ZenQuotes API).
- End: Actions loop until the user exits the app.


Student
Open App
Login via Google OAuth
Play Integrity Check
Trusted Device
Untrusted Device
Dashboard
Block Access
Log Expense
View Statistics
Get Daily Quote
Set Savings Goal
Manual Entry
Fetch Expenses from Firebase
Call ZenQuotes API
Update Savings Goal in Firebase
Save to Firebase Database
Render Charts
Display Daily Quote
Track Progress
Sync Data
Show Spending Trends
Daily Motivation
Real-Time Updates
Close App

Accessibility Features:

The app incorporates bold headers (e.g., “Total Expenses: ₱1,763.00”) and visually distinct emojis to emphasize financial metrics, aiding users in quickly identifying important data. Buttons like “SAVE” and “CANCEL” are prominently labeled with high-contrast text, ensuring clarity during transaction input. Interactive elements, such as checkbox categories (e.g., “Transport [x]”), provide immediate feedback for selections, while placeholder text (e.g., “Breakfast at McDonald’s”) guides users through optional fields. Layouts are organized hierarchically—grouping data under headers like “Expense by Category” and using tabular formats for recent transactions—to simplify navigation and reduce cognitive strain. Consistent icons and structured lists (e.g., bullet points for expense breakdowns) further enhance readability and scannability.

6. Deployment & Maintenance

6.1 Deployment Plan

Release Process

The Android app will be distributed via a GitHub-hosted landing page. A signed APK file will be uploaded to the project repository, and a direct download link will be embedded on the page. Users can visit the page via a browser, click the download button, and manually install the APK. Before release, the APK will undergo rigorous testing on Android 7.0+ devices to ensure stability and security.

Rollout Strategy

The app is designed for academic and project evaluation purposes. The APK will be shared exclusively with instructors, project evaluators, and a small group of beta testers for feedback. No public release or staged rollout is planned.

Maintenance Plan

- Monthly Updates: Bug fixes and minor feature enhancements.
- User Support: A GitHub Issues tab will track feedback and technical queries.
- Monitoring: Firebase Crashlytics and Performance Monitoring will log errors and usage metrics.

7. Glossary

Table 1. Technical Terms and Acronyms

Term	Description
API (Application Programming Interface)	Tools and protocols for integrating external services (e.g., Google OAuth, ZenQuotes)
Firebase	A Backend-as-a-Service (BaaS) platform for authentication, databases, and analytics
Google OAuth 2.0	Authentication protocol for secure sign-in via Google accounts.
Play Integrity API	Validates device authenticity to block tampered devices.
ZenQuotes API	Provides daily quotes for the app dashboard and other pages.
MVVM (Model-View-ViewModel)	Design pattern separating UI, logic, and data layers
TLS (Transport Layer Security)	Encrypts data during transmission (HTTPS)
GDPR (General Data Protection Regulation)	Compliance framework for user data privacy

8. References & Resources

Android Developers. (n.d.). *Guide to app architecture*. Android Developers. https://developer.android.com/topic/architecture

Android Developers. (n.d.). *Play Integrity API*. Android Developers. https://developer.android.com/google/play/integrity

Google. (n.d.). *Firebase documentation*. Firebase. https://firebase.google.com/docs

Google. (n.d.). *Material design guidelines*. Material Design. https://m3.material.io

Google Developers. (n.d.). *Using OAuth 2.0 to access Google APIs*. Google for Developers. https://developers.google.com/identity/protocols/oauth2

LinkedIn. (2023, September 26). *What is the best way to document architecture for mobile applications?* LinkedIn. https://www.linkedin.com/advice/0/what-best-way-document-architecture-mobile

React Native. (n.d.). *Accessibility*. React Native Documentation. https://reactnative.dev/docs/accessibility

ZenQuotes.io. (n.d.). *ZenQuotes API documentation*. ZenQuotes. https://zenquotes.io