SP-26-Team Blue — Budgeting App

CS 4850 - Section 01 – Spring 2024

Apr 26, 2024

PROJECT INFO

- Professor: Sharon Perry
- Important Links:
  - github.com/SP-26-Blue
  - https://youtu.be/YrxO1qv_orI
  - https://sp-26-blue.github.io/Website/ksu_course_page
- Team members:
  - Sepehr Eshaghian
  - Branden Woods
  - Matthew Lambert
- Stats:
  - Number of UI components: 31
  - Number of code components: 28
  - Number of lines of code: 1335

Contents

INTRODUCTION

Majority of people between ages 18-25 struggle financially so we decided for Our application to aim and help those groups of people, but in general the application can be useful to everyone. The ideology behind the app is to help the user to develop financial thinking in their mind so they automatically stop wasting money with out the need of the application and have their finances in order.

The application connects to the user bank through API for a better insight as for they most recent financial activities, the application has a place for them to set their goal which then shows them the percent of their goal accomplished by a percentage bar shown below the goal. The application sends out a quote related to their goal with positive massage so the user knows we care and we are analyzing them, it helps the user to keep themselves in check if they know someone is always checking on their activity.

REQUIREMENTS

- **Functional Requirements:**
    - Internet connection.
    - User authentication (login/signup).
    - Integration with banking APIs to fetch transaction data.
    - Display recent transactions.
    - Ability to set and track saving goals.
    - Notifying the user for every transaction.
    - Sending positive message to user.
- **Non-Functional Requirements:**
    - Security measures for protecting user data (done through firestore).
    - Fast response time due to cloud storage but internet speed dependent.
    - Available to both Android and IOS.

ANALYSIS

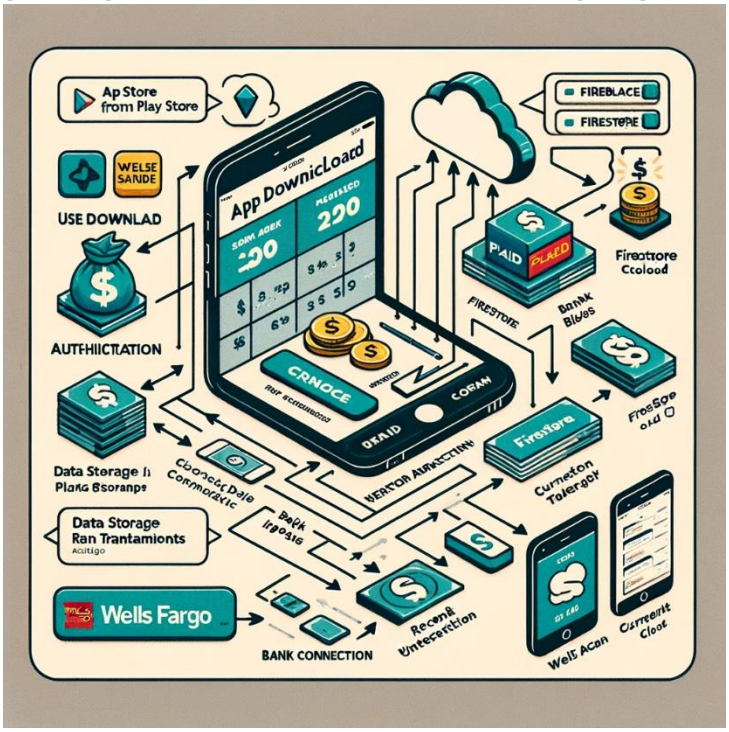
- **User Analysis:**
    - **+ User needs to be constantly updated through every transaction made to their account.**
    - **+ The user needs a reason to re-open the app**

- **+ The most important reason people check their budgeting app is to know how much they have left in their account**
- **-The application sends out notification for any update it gets from the bank API.**
- **- With everyday messaging, we make a personal connection to every user**
- **- The remaining balance in their account will be the first thing they see when they open the app with a huge font**

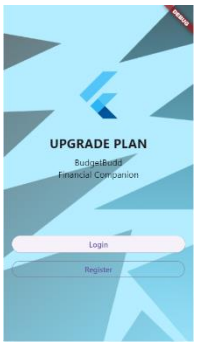
- **Competitor Analysis:**
  - **Budgeting app is not a brand-new idea for a mobile application even every bank has its own app now, so I compare our application to the other groups in our class our.**
  - **Our application sends out incentives every other day, it has instant notifier for any transaction made, it has a section for goals which shows the amount of goal accomplished, it gives insight of how the money is being spent in a piece chart such as "grocery, Bills, etc." and none of the teams have implemented all these features at the same time. Also our application theme goes with our teams color (jk).**

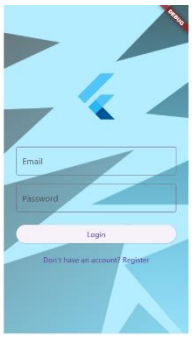
DESIGN

- **Architecture Design:**
    - **The user downloads the app either through app store or play store, then sign in/ sign up, the authentication process goes through firebase credited by google and the user data will be stored in the firestore cloud, the user uses plaid API to connect to their bank, for this example lets say it's a "wellsfargo", and through API we get access to the users current balance and recent transaction to show on the main page**
    - **The image is generated through AI and it didn't fully grasp what I was going for, hence some of the texts are smushed together but it gives a good visualization of what will be going on**

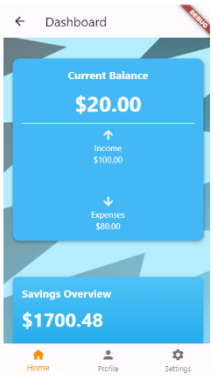


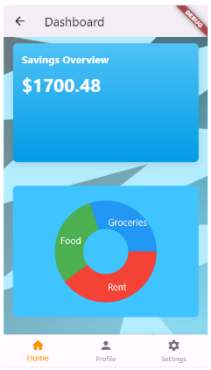
- **UI/UX Design:**

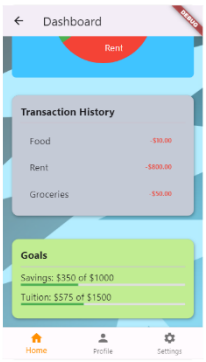




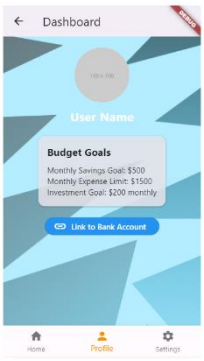
- **Intro** **Log-In** **Sign-Up**

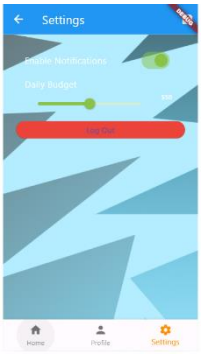






- **Main-Page** ------- -------





- **Profile** **Settings**

- **At Intro Page user can choose to log-in or sign-up, then they will be directed to the main page which majority of the functionality of the app lies, the user will also see these exact numbers since they are**

mock data but when they link their bank account form the profile page the numbers instantly change to the actual data. The user also can choose a profile picture for themselves in the profile page if they want to.

- **Based on the users income/allowance they can set a limit for themselves in the setting page, or if they want to disable the notification from app, its also done through settings.**

DEVELOPMENT

- **Technology Stack:**
  - **List the technologies used (e.g., programming languages, frameworks, databases).**
  - **The application is coded in Dart which is a language model for flutter applications.**
  - **For user authentication and data storage we used Firebase**
  - **The application runs on flutter 3.1.6, and Dart Software Development Kit of 3.3.0**
  - **All the Updates and version controls went through Github**
- **Implementation Highlights:**
  - **Fully functioning user authentication.**
  - **Notifying user of every transactions.**
  - **When the total balance of the user goes below a certain threshold they get notified as well.**
- **Challenges Faced:**
  - **The Firebase was super tricky to get working but after changing the google JSON file and flutter version and dart version we got it working.**
  - **Getting plaid API to work was a very hard thing to do, I think one of the team members is still trying to get it to work but he fails every time.**
  - **The hardest challenge was working as a team, and thank God it's over.**

TEST (PLAN AND REPORT)

- **Testing Plan:**
    - **We used unit testing for this project, since the job was divided, whom ever made smallest development, after double checking its functionalities, it then got uploaded for the rest to also test it.**
- **Testing Report:**
    - **Thanks to the resources available online and tutorials out there majority of the bugs and issues would resolve quickly aside from the API that we are still struggling against.**

VERSION CONTROL

- **Version Control Practices:**
    - **The IDE we used for this application is Android Studio which we connected directly to our Github, hence after we are adjustment to the code, we just uploaded the code to Github.**
    - **We had 4 branches, 1 "Master" which the finished product goes to, and every member got their own branch, so after every update, we send it to our branch and after it got reviewed by other, we pushed it to the Master.**

NARRATIVE DISCUSSION

- **We began our project with research for how we would like our app to look, the number of pages and what would be our greatest challenge so we could schedule in a way so we spend most of our time with that. After we came to a conclusion about the programming language and a theme along with how the skeleton of the app should look, we started with authentication page and dashboard, after everything was in place and the app looked good, we started to apply the functionalities such as fire base authentication, setting page and a function API to sandbox, unfortunately we were unable to fully connect to the sand box but the rest of the app turned out great.**

SUMMARY

- **We were trying to develop a budgeting app to help people with managing money, the skeleton and most of the functionalities of the app is ready the only thing preventing our app to be fully done is the plaid API which we still lack to connect with.**
- **Majority of the reviews we got for our presentation and product were positive so I think an average person would also enjoy using our app and approve it.**
- **As for future improvement, it would be great if we could get the hold of plaid API and then add AI API to it as well so it can make comment on the user transactions, it would act as both a companion and also a financial advisor.**

MOBILE APP REQUIERMENT

- Evidence of finishing the flutter tutorial