A picture containing text

Description automatically generated

**Final Project: Muhni, A Money Management App**

CSCI-380: Intro to Software Engineering

Fall 2022

Prof. Maherukh Akhtar

**Manhattan Software Engineers**

Jeffrey Aguilar - 1265967

Jason McHenry - 1265317

Alexander Pachnicki - 1230402

Omar Sheikh - 1296249

**TABLE OF CONTENTS**

1.0 Introduction

1.1 Existing Systems

1.2 Proposed System

1.3 Software Engineering Model

1.4 Purpose

1.5 Project Objective/Goals and Scope

1.6 Technologies & Tools

1.7 Users

1.8 Motivation

1.9 Timeline

2.0 Analysis of the System

2.1 Activity List

2.2 Data Flow/Context Diagram

3.0 Database Design

3.1 Table Schema

3.2 Entity Relationship Diagram

4.0 Functionality and Implementation

4.1 Features

4.2 Security/Privilege Levels

4.3 File Structure

4.3.1 Example 1

4.3.1a Code Snippet

4.3.1b Code Explanation

4.3.2 Example 2

4.3.2a Code Snippet

4.3.2b Code Explanation

4.3.3 Example 3

4.3.3a Code Snippet

4.3.3b Code Explanation

5.0 Testing

6.0 Conclusion

6.1 Limitations

6.2 Future Enhancements/Recommendations

6.3 Team Members

6.3.1 Learning Experience and Outcome

7.0 References

**TABLE OF FIGURES**

Figure 1: Activity List……………………………………………………………………………13

Figure 2: DFD……………………………………………………………………………………14

Figure 3: Database Schema………………………………………………………………………15

Figure 4: Parent Table……………………………………………………………………………16

Figure 5: Child Table.……………………………………………………………………………17

Figure 6: ERD……………………………………………………………………………………18

Figure 7: Child Account…………………………………………………………………………19

Figure 8: Parent Account……………………………………………………………………...…20

Figure 9: Code Snippet 1……………………………………………………………………...…22

Figure 10: Code Snippet 2..…………………………………………………………………...…23

Figure 11: Code Snippet 3..…………………………………………………………………...…25

**Abstract**

The goal of our application, Muhni, was to keep our users in mind: children between the grades of six to twelve. We created an app that assists these children in managing their finances at an early age, providing them with financial knowledge and independence. With the help of their parents, they will gain financial responsibility by visualizing the money they earn, spend and save on a daily basis. Our app conveys this information in an interactive, friendly, and simple manner that makes it easy to learn about. Muhni was created through Flutter/Dart while being paired with the Firebase database system and can be accessed through iOS or Android.

Keywords: Money, Budgeting, Parent, Child, Educational, Firebase, Flutter/Dart, GitHub, Scrum

1. Introduction

1.1 Existing Systems

The first existing system with similar functionality was an app called Greenlight. Greenlight’s mission is to provide financial stability and services to children, all while having their parents in the loop as well. Founded in 2014 by Tim Sheehan and Johnson Cook, around four million users have downloaded the app in the last eight years (Forbes). They allow children to set up a debit card under the MasterCard title that earns cash back, giving them the opportunity to make the most of their money. Through the use of algorithms in machine learning, Greenlight is able to survey how a user might spend their money and how the app can assist them in doing so. While Greenlight does an adequate job of providing users with financial management, it focuses too much on the parental aspect of it. The lack of a learning curve for children in Greenlight leaves room for improvement in the informative aspect of saving money.

The second system that proved to be similar was an app called FamZoo. Based in Palo Alto, Famzoo has similar functionality to Greenlight in terms of parent and child interaction (Famzoo). Parents can add money to a children’s account along with a prepaid debit card for use. Parents can track their children’s spending as well as the money going in and out of their accounts. The main focus of Famzoo is parental advising and supervision, helping their children find avenues for financial success. While this can be seen as a success, the responsibilities do not lie in the hands of children. It is an app that makes a parent’s life easier in regards to keeping track of one’s family expenses. Not only that, but the six dollar monthly subscription leaves FamZoos at a disadvantage with its competitors.

The third and final system that drew similarities to our application was the Mint app. Founded by Aaron Patzer and worth one hundred and seventy million dollars, Mint is a widely popular app amongst those in need of a money-managing application (Inc). Users are able to have access to specific bank accounts, credit cards, and investments in order to centralize all their finances. Mint puts everything one can imagine that deals with expenses under one umbrella, providing the user with an accessible hub. Despite Mint’s popularity, it is catered towards teens and adults with an abundance of money. A child has a long way to go before managing their investments or multiple bank accounts. Mint works for older individuals but fails to include a younger demographic.

1.2 Proposed System

Our app, Muhni, is designed to help children understand the importance of saving money through self-management. Unlike many other money-saving applications that are aimed at high-income individuals, our app is specifically tailored to children. To appeal to our younger demographic, we have a user-friendly interface that is simple and visually appealing. Given that our users are typically between the ages of 6 and 12, our straightforward design will provide them with a streamlined experience.

When the app is opened, users are presented with a login screen that asks whether they are a child or a parent. Depending on the user's response, different features will be made available. If the user is a child, they will be asked a series of questions about their income, expenses, and savings goals. After providing this information, the app will display an amount of their total money inflow and outflow in their virtual wallet. Users will be able to input their daily expenses and income manually. To keep the app as simple as possible, we plan to group all expenses and income into their respective categories. In addition, the app will include weekly saving goals based on the user's needs, as well as messages of encouragement and suggestions for saving money on days when spending is higher than usual. For parents, the app will provide features that promote supervision and assistance with income, but the child will have the ultimate responsibility for managing their budget. Parents will be able to create separate groups for each of their children, allowing them to provide individualized support and guidance. Through a messaging system, parents will be able to offer their children opportunities for earning money through chores and verify that these tasks have been completed before sending payment. This system encourages children to take responsibility for their own savings and spending habits.

1.3 Software Engineering Model

For our software engineering model, we used the iterative model to complete our project in the most efficient way possible. This model was well-suited to our project because it allowed us to develop a simple, basic version of the program that met our requirements and accomplished our goals. We set the blueprint for what our app wanted to do first, allowing us to get our feet wet. After our initial iteration, we then worked on revising and improving the design, testing, and implementation of the application, rather than starting from scratch. This allowed us to focus on refining and perfecting the application. The iterative model is particularly useful for our project because we had clear requirements and the possibility of future changes. This model enabled us to create a rough version of the application, which achieved the goals we have set, and then iteratively improve and refine that version until we were satisfied with the final product. Due to our experience in development, we didn’t expect ourselves to have it perfect from the beginning. Creating a base application and adjusting it along the way seemed best fit for our money management application. Overall, the iterative software engineering model was the ideal choice for our project because it helped us develop a functional application quickly and then focus on improving and refining it. Muhni was a work in progress over the course of our project timeline, however, we were able to perfect it as much as we could.

1.4 Purpose

Our application will bridge the gap between children and financial responsibility. It will do so by giving the user a simple, straightforward, and easy-to-navigate application to manage their finances and budget their money. Muhni gives full financial responsibility to the child, allowing them to keep track of their income. Coupling a parent account with a child account provides the user with positive reinforcement and motivation to achieve financial independence.

1.5 Objective and Scope

Our objective while creating this application was to create a functioning application for fiscally inclined child users in hopes of encouraging saving money. Our implementation of a parent account allows the child user to work hand in hand with the parent, detailing specific purchases and various methods of income. Users can have either an Android or iOS device in order to use the application, allowing the application to be catered to any user who is between the grades of six to twelve and has finances to manage.

1.6 Technologies and Tools

We used a plethora of technologies and tools in order to complete our project. Due to the requirements of the project, it was crucial that we spread our talents and used as many resources as possible. The creation of our database was through Firebase as opposed to SQL, as we found that Firebase was much easier to use and manage. In terms of coding the user interface and project itself, we resorted to Flutter/Dart. The open-source software allowed us to develop the framework and user interface through the use of object-oriented programming. While these two technologies carried a heavy load for our project, we could not have completed it without communication within the group. Discord served as the main avenue of communication through texting, voice calls, and video calls. Committing our code in order to have a centralized hub for our work was crucial as well. We used GitHub for this task given that it is the most popular tool for sharing code. Google Slides and Google Docs were the primary technologies used for documentation and presentation throughout the project.

1.7 Users

Our application is designed for kids in grades six through twelve and their guardians. By providing tools for learning about money management during these formative years, we aim to help users develop good habits and achieve financial independence early on in their lives. Our app allows users to plan and budget their expenses, and to see how much money is coming in and going out of their accounts. For older users who have jobs, the app can help them understand the value of their income and how to manage it effectively. Parents can also create their own accounts to monitor and support their children's financial well-being.

1.8 Motivation

The motivation behind this project is to provide those in need of guidance a tool to achieve their goals. Often, children are inexperienced with the money that they own. Whether it be through gifts or income, having an abundance of money at a young age can allow one to feel invincible. This common misconception can lead children down a path of irresponsibility soon. Creating an application that can steer them in the right direction seemed beneficial to the group. An application like this could have helped us at an early age as well.

1.9 Timeline

The timeline of our project was difficult to follow due to our lack of experience and other projects. We started off on the right foot with our diagrams given that we learned about them in class. Creating the entity-relationship and data flow diagrams gave us a clear vision between November 2nd and 7th, allowing us to stay on track. Our next task was the setup of our database. While we planned to have that done within a week from November 7th, we experienced our first setback due to the learning curve of Firebase. We finished this task two days late on November 16th. Our biggest hit was the development of the front end, as our goal was to have it completed by November 21st. Setting ourselves back by two days during our previous task put us on a time crunch for the front end. While we wanted to complete it early, it was near impossible to have it done in five days. We finished our front end by December 1st and had the back end completed five days later on December 6th. We expected our back end to be time-consuming, but it proved to be otherwise. We estimated that we would take sixteen days to complete our back end, however finishing it earlier put us back on track.

2. Analysis of the System

2.1 Activity List

Below is a representation of our activity list, visualizing our tasks as well as our timeline. Dividing the tasks in four major categories allowed us to focus during those blocked off time periods. Table

Description automatically generated

Figure 1: Activity List

2.2 Data Flow/Context Diagram

Displayed underneath is our context data flow diagram. The application is the central hub for data entering and leaving, our DFD helped plan out the data transfer. The user inputs their personal information while receiving back a step-by-step process of using the application. The app receives data such as account details, income per month, and income over a variable period of time.

Diagram

Description automatically generated

Figure 2: DFD

3. Database Design

The database was designed using Firebase in order to have direct access and connection between the code and the database. While we did consider using MySQL, we found that Firebase suited our needs given our experience and need for something simple and straightforward.

3.1 Table Schema

Displayed below is the database schema for our project



Figure 3: Database Schema

Displayed below is an explanation of the two tables

Graphical user interface, text, application

Description automatically generated

       Figure 4: Parent Table

As seen above, this table features the name of the user as well as their personal information. It includes the first name, last name, email address, and user identification number. The children for the user are listed as well.

Graphical user interface, application

Description automatically generated

Figure 5: Child Table

As seen above, this table features the name of the child user as well as their personal information. It includes the first name, last name, email address, and user identification number. Balance, chores, expenses, and income are displayed as well.

3.2 Entity Relationship Diagram

Displayed below is our entity relationship diagram that we created based on our application. It shows the correlation between the parent and child account as well as the income and expenses.

Diagram

Description automatically generated

Figure 6: ERD

4. Functionality and Implementation

The project was designed and coded using Flutter/Dart for the entirety of it. Flutter has a unique framework that allows us to use XCode, all while being smooth and organized. It promotes app development in the fastest way possible which catered to our needs for this project. Not only that, but it allows for both iOS and Android app development. Using Flutter gave us the versatility that we needed to create a fully functional application.

4.1

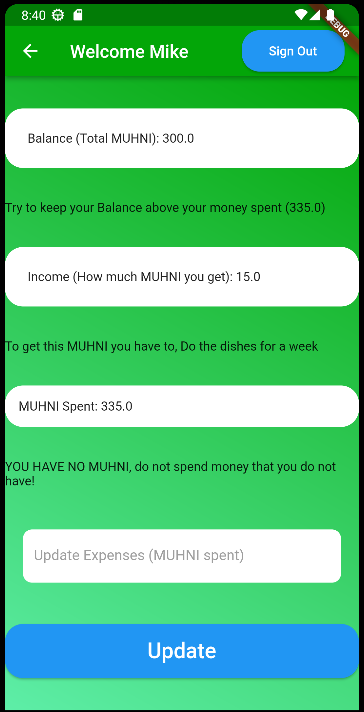
Features

Figure 7: Child Account

Due to our app being catered to children between the grades of 6 and 12, a straightforward user experience was key for us. As shown above, Muhni is simple and clean in order to convey a minimalistic layout. Having less distractions throughout the app allows the experience to be streamlined. The app portrays the balance of the child’s account, an option for adding income, and an option for money spent. Small messages of encouragement and suggestions throughout the day provide the user with a learning experience, as well as keeping one’s spirits up.

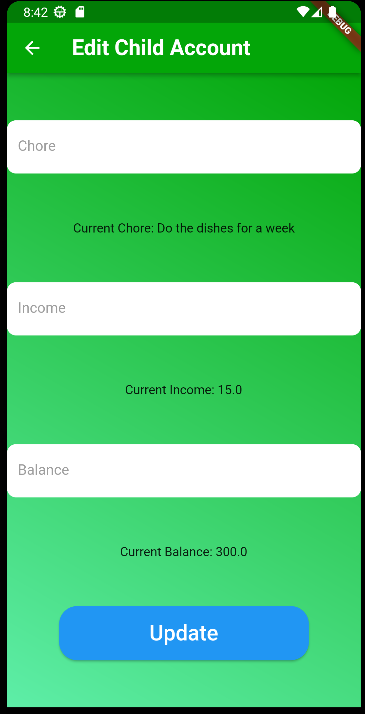
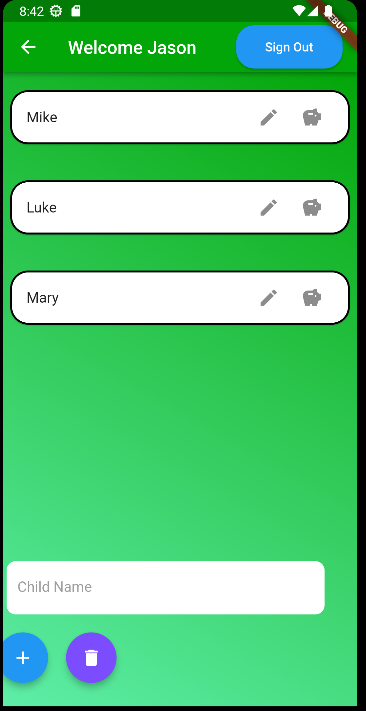


Figure 8: Parent Account

The parents will have features that promote supervision and assistance in income, however the budgeting of money is entirely up to the child. Parents will be able to create separate groups with each of their children in order to give them a one on one experience. They are able to view their child’s weekly saving progress and send them money through chores. Once the chores are completed, payment will be sent to ensure completion

4.2 Security/Privilege Levels

Due to time constraints, we were unable to add any other security features besides the ones Firebase included in their own library.

4.3 File Structure

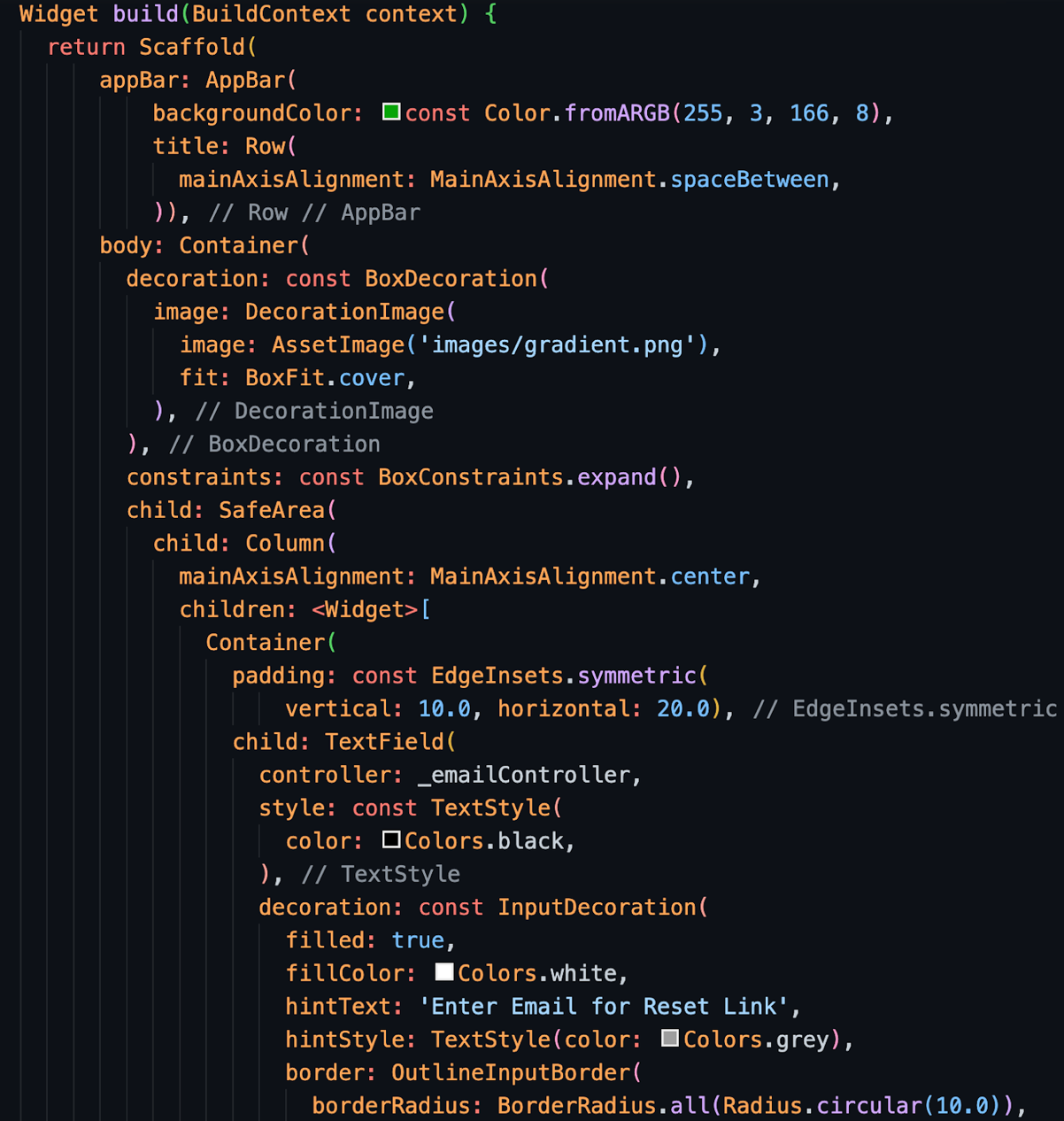
4.3.1 Example 1: Forgot Password Screen UI              

Figure 9: Code Snippet 1

4.3.1b Code Explanation: In the code above, the widget is built using a scaffold which is Flutter’s most basic page constructor that outlines what will go inside a UI page. The scaffold contains an app bar section and a body section. Inside the app bar, we simply just specified how we wanted the background of it to look like for aesthetic purposes. Inside the body, we wrapped everything in a container so we would be able to have our background gradient displayed. Next, we wrapped everything in a safe area widget so other future widget displays wouldn’t go off the screen when going between different platforms. Then, we wrapped everything else in a column so we would be able to display multiple child widgets inside a single page. Finally, you can see that we connected Firebase text controllers with our UI in order to take the email of our user and be able to check if they have an account connected and stored in our database in order to send them an email to reset their password.

4.3.2 Example 2: Home Screen UI

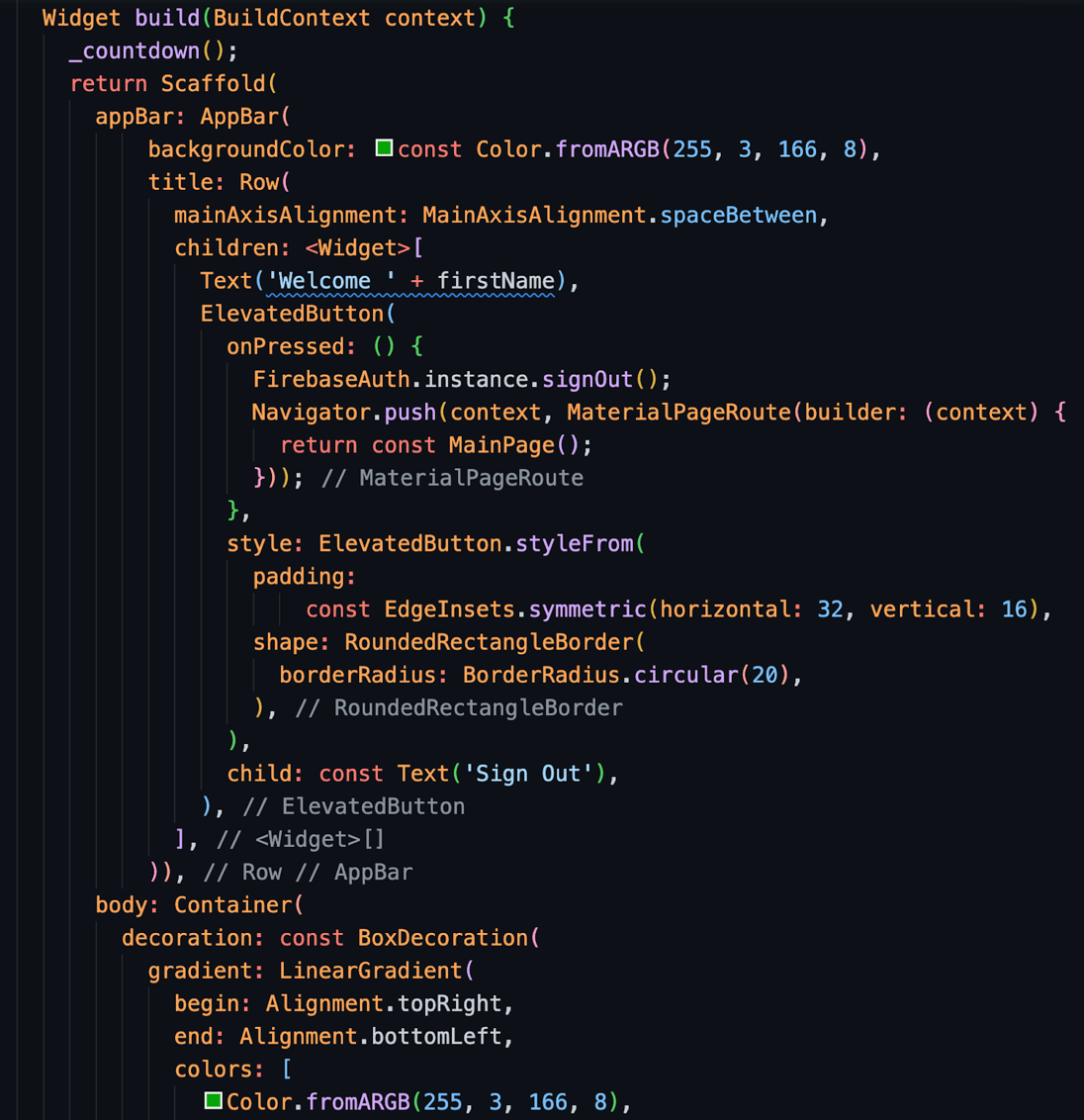


Figure 10: Code Snippet 2

4.3.2b Code Explanation: In the code above, the widget is built once again using a scaffold which is Flutter’s most basic page constructor that outlines what will go inside a UI page. Inside the app bar, we specified how we wanted the background color to look, added a greeting to our users, and added a sign out button for users. Inside the body, we wrapped everything in a container so we would be able to have our background gradient displayed. Next, we wrapped everything in a safe area widget so other future widget displays wouldn’t go off the screen. Finally, we included a ListView.builder() widget in order to display, add, remove, and edit the children a parent adds into their group.

4.3.3 Example 3: Sign Up Screen UI

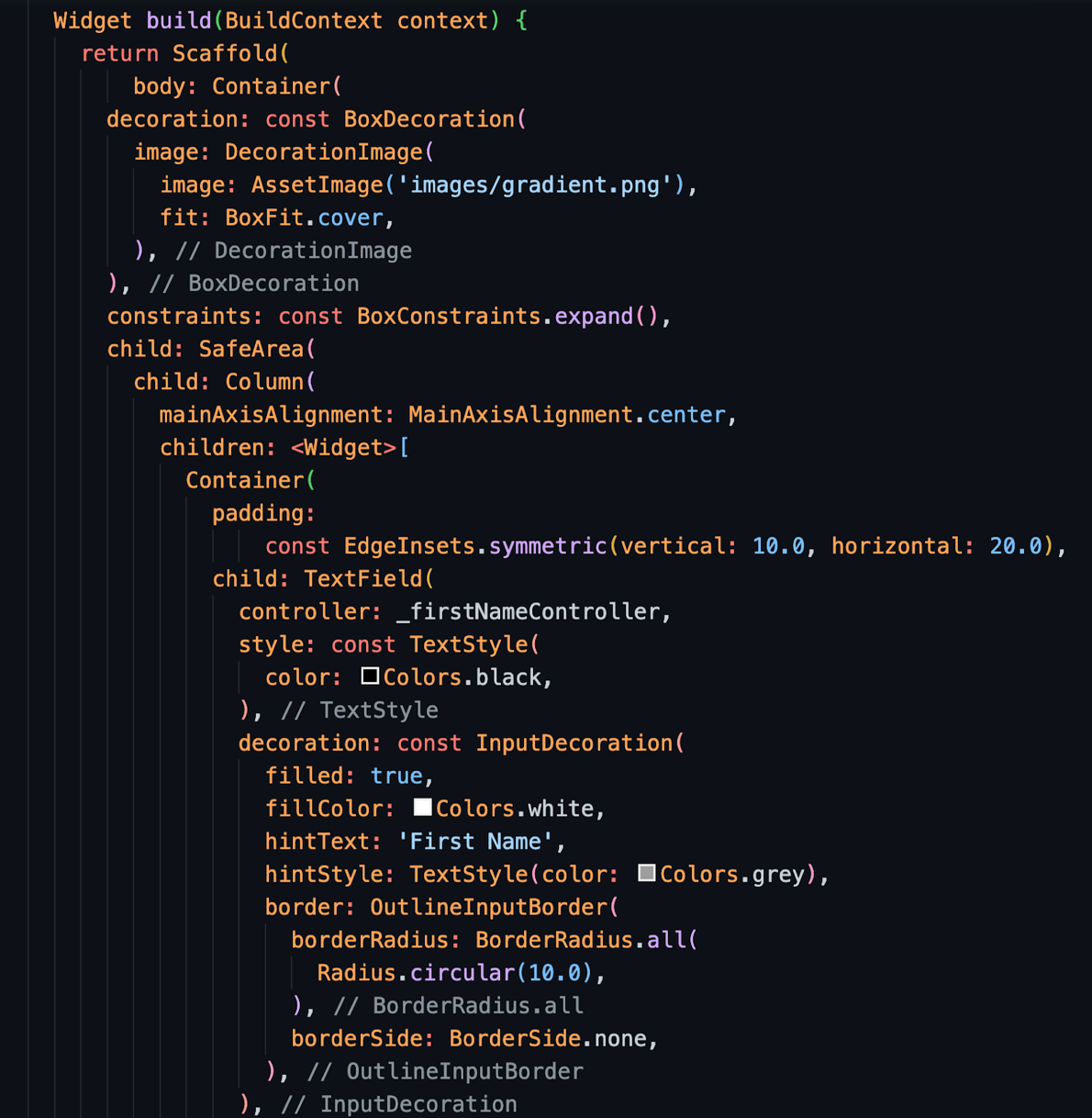


Figure 11: Code Snippet 3

4.3.3b Code Explanation: In the code above, the widget is built once again using a scaffold which is Flutter’s most basic page constructor that outlines what will go inside a UI page. However, for this UI page, we decided not to use an app bar. Inside the body, we wrapped everything in a container so we would be able to have our background gradient displayed. Next, we wrapped everything in a safe area widget so other future widget displays wouldn’t go off the screen. Then, we wrapped everything else in a column widget so we could fit our containers that contained text fields such as name, email, the forgot password button, and the sign up button.

5. Testing

Throughout the duration of the project’s development, we had to perform a lot of unit testing. This was necessary in order to debug the program and to overall check if it the app works.

6.0 Conclusion

6.1 Limitations

Despite finding success in our application, we came across a few mishaps and limitations along the way. The hardest part of finalizing Muhni was connecting the front-end to the back-end. Our lack of extensive development knowledge made it hard to make a full fledged application from scratch. The task was scary and became frustrating at several points during our timeline. However, we were able to teach ourselves and adapt to the new technologies (Firebase) that we were using in order to figure it out. Not only that, but reading and writing data from the database was a difficult task. Creating the user interface and having it run smoothly was something that was new for our group as well. Little nuances such as refreshing the page didn’t seem like tasks that were going to hinder us in the long run, however those were the things that were most important. Creating an application is not only about the back-end but also creating a fluid front-end for the user. We strived on creating a simple yet functional application and that meant addressing the smallest yet most significant aspects of the front-end. Learning how to use Firebase was also a challenge, as most of the tutorials online were outdated.

6.2 Future Enhancements/Recommendations

As far as future enhancements go, there was a lot that we could have done to improve. Given the opportunity, we wanted to create a clean and enhanced user interface. Our user interface was already simple and minimalistic, however we can always find things to improve if given the chance. Creating an option for a dark mode for aesthetics is always a fan favorite amongst users so that would be an option. Adding customizable profiles would be a way to take the app to the next level as well, whether it be profile pictures or bios. Connecting profiles to existing accounts such as Google or Apple could assist in that, elevating the user experience and making the sign in easier. With the expansion of an application comes risk so security would have to be included. Adding more security features would create a safer environment for the user, as the app functions but lacks protection at the moment.

6.3 Team Members

6.3.1 Jeffrey Aguilar (Team Leader 1)

For my current technical knowledge, I was very familiar with Java, HTML, and CSS as well as some knowledge in Python, SQL, Flutter/Dart, Xcode/Swift, and C++. I was responsible for constructing the UI pages for the app. A difficulty I faced while completing this project was when integrating Firebase. When Firebase was integrated into the app, my VSCode was unable to run the app since I have an Apple Mac M1 Chip and Firebase is really weird with the architecture of my system. Even after spending a few hours trying to fix the problem, at most I was able to run the app but it wouldn’t launch onto the iOS simulator. Another difficulty I faced was time management since I had four total projects due this semester and they were all due in a short time frame.

I was able to learn a lot more about Flutter/Dart during this project’s development as well as a better understanding on how front-end development is handled. I also discovered how much of a headache back-end development can be and I appreciate my teammates who were able to deal with that pain. Now I know I would rather stick to front-end development rather than the back-end. Personally, I think our app came out great due to all our collective efforts.

6.3.2 Jason McHenry (Team Leader 2)

For my current technical knowledge, I am experienced with Unity, C#, Java, HTML, and SQL. For this project, I was a team leader and was mainly responsible for the backend of the project. I was mainly responsible for creating the Firebase Firestore and allowing the user's information to be dynamically updated to the app. Allowing the user to have no need to refresh their app, as it is automatically done for them. A large amount of this project was spent around researching Flutter/Dart and Firebase. I previously have never used any of these softwares before and had to spend countless hours of research to understand how they work.

Overall, the project was pretty successful as we created an app that we all felt was usable and had many additional features we wanted to implement to actually make it to the project. However, a large struggling point of the project was centered around our inexperience with Flutter/Dart and not knowing how to access and display information from Firebase to our application. This causes several issues and countless hours of debugging and researching, mainly to be met with no solution in sight, to occur until we manage to get something functional to work. However, through excellent team communication and spending a large amount of time working together to finish the project, I felt that as a group we were quite successful and had an overall good experience working together.

6.3.3 Alexander Pachnicki

My current technical knowledge includes a deep understanding of Java programming language as well as SQL and I have worked with various IDEs and programs such as Android Studio. I had backend responsibilities in this project and I would say we all helped each other out in each section of the project. Sometimes we would all work on the backend or all on the front end.

I think I learned a lot from this project, not only from a professional standpoint but also as a team member and working in a group. I was able to learn from my teammates to accomplish goals and solve problems. I was very happy to be a part of my group and I think we all did a great job on this project. Some of the challenges we faced were connecting the front end and the back end and also dealing with firebase being difficult to understand. Being able to figure out a solution to a problem as a team was very satisfying. One example is when we tried to solve the issue of firebase not refreshing the data which we had displayed on the app. Being able to work together and solve this problem was a great and amazing experience. Overall, having to balance other classes made it much more stressful but I think if this was my only class it would have been a much more enjoyable experience. Given how hard we worked I am proud to say that I am part of the team that created Muhni.

6.3.4 Omar Sheikh

For my current technical knowledge, I am experienced with XCode, Visual Studio, and Eclipse. I also have knowledge in C++, Java, Python, SQL, C#, .NET. I initially had backend responsibilities, however, my main laptop broke and left me to use my outdated and old one. The lack of an updated operating system put me in a less helpful position in coding so I had to help elsewhere. I helped as much as I could and I completed a majority of the slides and written report.

I was able to learn a lot during this project from my fellow group members. Not having a lot of experience in Firebase or Flutter gave me an opportunity to look at some of the work they were doing and pick up on it. I thought the experience was a fun one overall, however, it was hard. There were many sleepless nights within the group, whether it was figuring out how to connect the front end to the back end or simply finishing up the report. Thankfully, we had a cohesive group that worked together well. Where someone needed help, another picked up where they left off, and vice versa. I thought our outcome as a group was amazing given our capabilities and experience. Creating a functioning app connected to a database was fulfilling, especially because it had a purpose. I would say that I am extremely proud of what we were able to accomplish and how Muhni turned out by the end of the semester.

7.0 References

“Famzoo Faqs.” *The FamZoo Blog*, https://blog.famzoo.com/p/famzoo-faqs.html#what.

*The Kid behind a $170 Million Website | Inc.com*. https://www.inc.com/articles/2009/09/mint-qa.html.

Lake, Rebecca. “Greenlight Card Review.” *Forbes*, Forbes Magazine, 17 Nov. 2022, https://www.forbes.com/advisor/banking/greenlight-card-review/.