Mental Health App Milestone 1

—

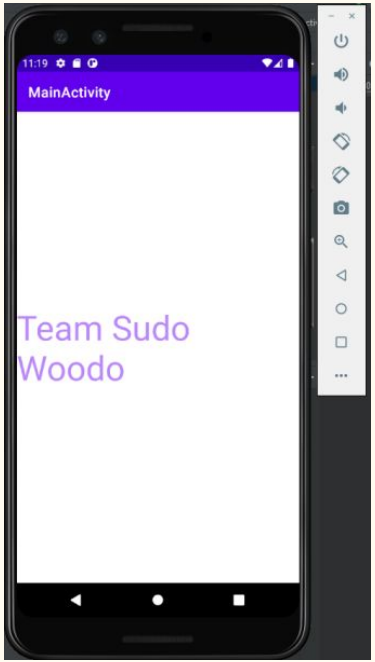
Team Members: Joshua Breininger, Daniel Bornemann, Phi Duong
Faculty Advisor: Dr. Bernhard

Overview of Milestone 1

Task	Progress	Daniel	Josh	Phi	To Do
1. ("hello world") demo(s)	100%	16%	16%	66%	None
2. Resolving technical challenges	100%	16%	16%	66%	None
3. Compare and select collaboration tools	100%	16%	66%	16%	None
4. Create a Requirements Document	100%	50%	25%	25%	None
5. Create a Design Document	100%	25%	50%	25%	None
6. Create a Test Plan	100%	50%	25%	25%	None

Task 1: “hello world” demo

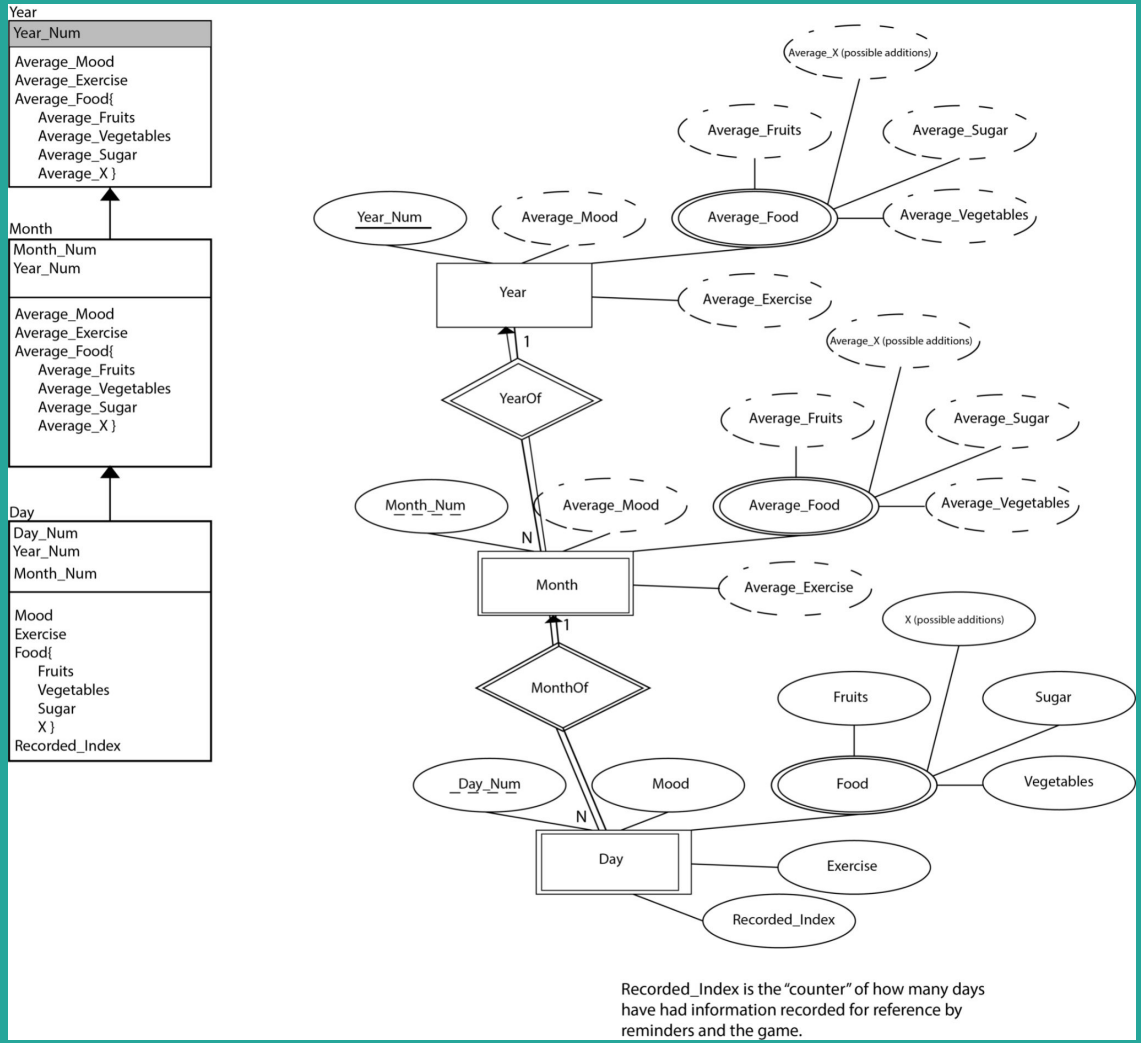
For the demos, we successfully installed and used Android Studio to emulate an Android Device, which we ran code on. We also successfully wrote a “hello world” program for Java to ensure our installations of the language were correct. A screenshot of the emulation running is contained below.



Task 2: Resolving Technical Challenges

All of us spent some time familiarizing ourselves with Java, although to an extent we will have to learn as we go, which will be a future challenge. We also came up with a database to use, being an SQLite relational database as Android has SQLite naturally installed and highly recommends it.

ER Diagram


Year
Year_Num
Average_Mood
Average_Exercise
Average_Food{
Average_Fruits
Average_Vegetables
Average_Sugar
Average_X }
Month
Month_Num
Year_Num
Average_Mood
Average_Exercise
Average_Food{
Average_Fruits
Average_Vegetables
Average_Sugar
Average_X }
Day
Day_Num
Year_Num
Month_Num
Mood
Exercise
Food{
Fruits
Vegetables
Sugar
X }
Recorded_Index
Average_X (possible additions)
Average_Fruits
Average_Sugar
Year_Num
Average_Mood
Average_Food
Average_Vegetables
Year
Average_Exercise
1
YearOf
Average_X (possible additions)
Average_Fruits
Average_Sugar
Month_Num
Average_Mood
Average_Food
Average_Vegetables
N
Month
Average_Exercise
1
X (possible additions)
MonthOf
Fruits
Sugar
_Day_Num
Mood
Food
Vegetables
N
Day
Exercise
Recorded_Index
Recorded_Index is the "counter" of how many days have had information recorded for reference by reminders and the game.

Task 3: Compare and Select Collaboration Tools

Visual Studio Code is professionally used and has a massive number of features, and one of our teammates has experience in using it. Codeshare.io had chat abilities which although useful is not a particularly important attribute to have as our communication has already been through Discord, a messaging and calling app we are used to. Codecollab.io like Codeshare.io did not have the same level of feature depth as VSC and required internet access to work on code, which is limited in usefulness. Furthermore, we ended up using Android Studio for our code writing uses as it has an inbuilt emulator for Android, which means we don't have to separately install it. Since the above tools had limited integration we could find with Android Studio, we decided to simply use GitHub as our collaboration tools to upload and view code, and continue using the video and messaging app Discord as we could share our screen to show our emulations running which is more useful for this project than simply sharing code.

Task 4: Create Requirements Documents

Our requirement document is meant to describe the requirements as specified by the client to fulfill the goal of our project. It contains descriptions of all features we need to have, and what those features have to do as well as what kind of behavior is correct for them.

The app will save user logged information for future use, and can access it at a later time in written and graphical format.

The app examines the database of stored logs to create notifications for the user.

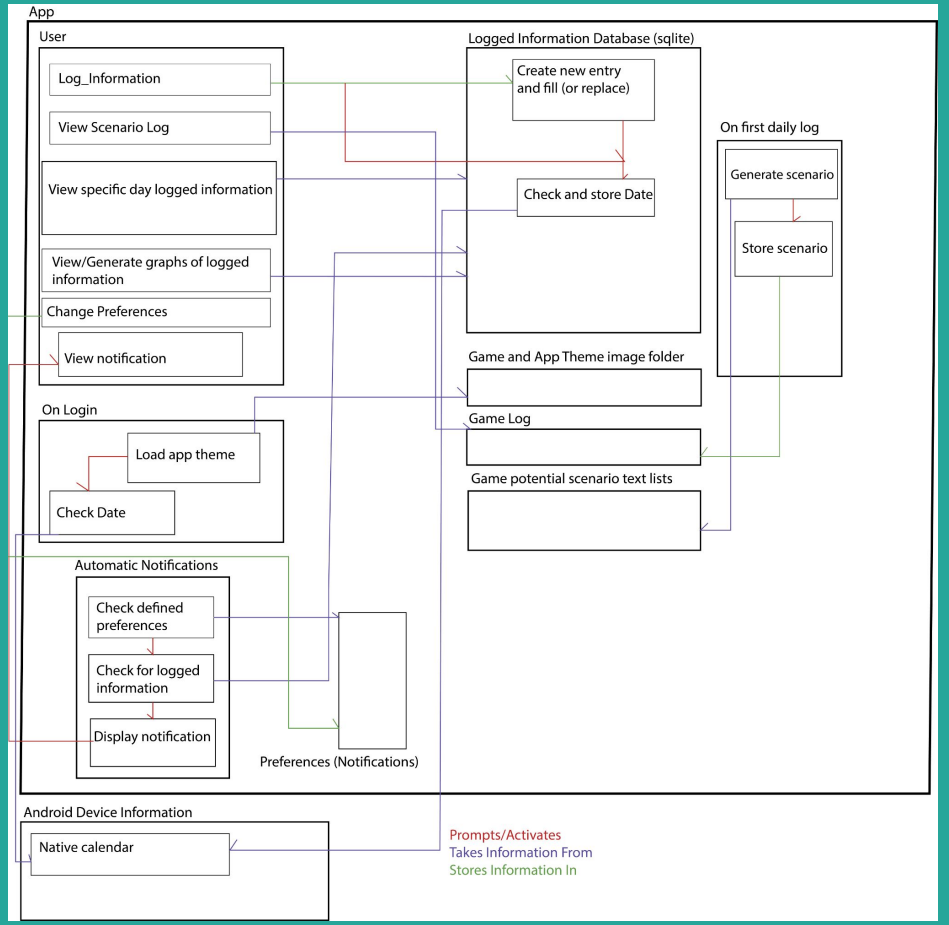
The app's game creates a randomly generated scenario and accesses a list of graphical elements with certain attributes to make visuals that fit.

The app changes its theme and visuals depending on what month it is, which changes the game scenarios.

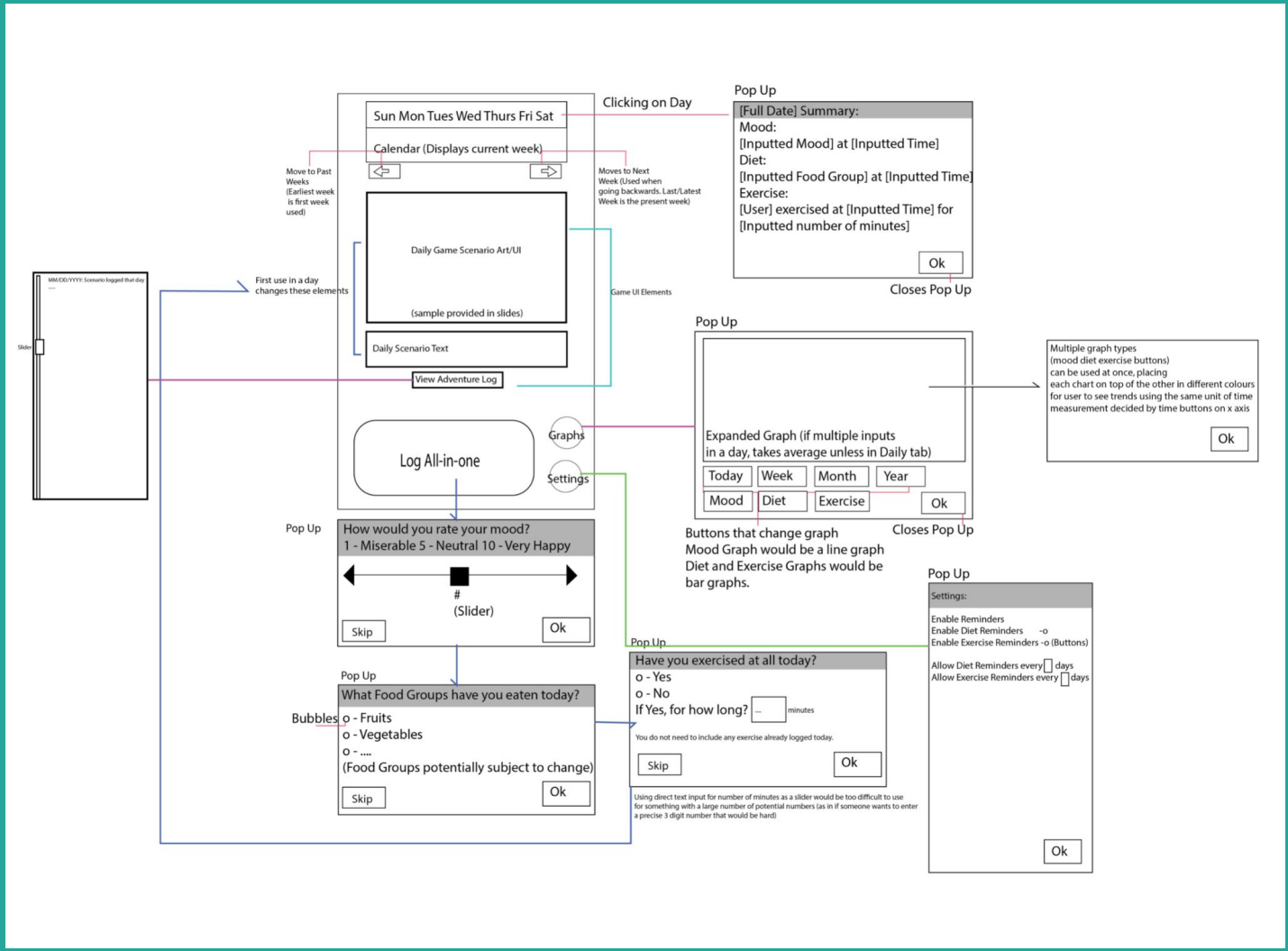
Task 5: Create a Design Document

The design document was made to show how our system designs worked and how the program interacts with the user and itself. Since the database user needs to be small and not take up much space, the effort was put in to keep entities and attributes to a minimum while also providing functionality in a relatively efficient way. The UI was designed to be easy to understand and make a quick and smooth “loop” through the features of the app.

System Architecture Diagram



UI


Sun Mon Tues Wed Thurs Fri Sat
Calendar (Displays current week)
Clicking on Day
Pop Up
[Full Date] Summary:
Mood:
[Inputted Mood] at [Inputted Time]
Diet:
[Inputted Food Group] at [Inputted Time]
Exercise:
[User] exercised at [Inputted Time] for [Inputted number of minutes]
Ok
Closes Pop Up
Move to Past Weeks (Earliest week is first week used)
Moves to Next Week (Used when going backwards. Last/Latest Week is the present week)
Daily Game Scenario Art/UI
(sample provided in slides)
Daily Scenario Text
Game UI Elements
First use in a day changes these elements
MM/DD/YYYY: Scenario logged that day
Slider
View Adventure Log
Log All-in-one
Graphs
Settings
Pop Up
Expanded Graph (if multiple inputs in a day, takes average unless in Daily tab)
Today
Week
Month
Year
Mood
Diet
Exercise
Ok
Closes Pop Up
Buttons that change graph
Mood Graph would be a line graph
Diet and Exercise Graphs would be bar graphs.
Multiple graph types
(mood diet exercise buttons)
can be used at once, placing
each chart on top of the other in different colours
for user to see trends using the same unit of time
measurement decided by time buttons on x axis
Ok
Pop Up
How would you rate your mood?
1 - Miserable 5 - Neutral 10 - Very Happy
#
(Slider)
Skip
Ok
Pop Up
What Food Groups have you eaten today?
Bubbles
o - Fruits
o - Vegetables
o - ....
(Food Groups potentially subject to change)
Skip
Ok
Pop Up
Have you exercised at all today?
o - Yes
o - No
If Yes, for how long? ... minutes
You do not need to include any exercise already logged today.
Skip
Ok
Using direct text input for number of minutes as a slider would be too difficult to use for something with a large number of potential numbers (as in if someone wants to enter a precise 3 digit number that would be hard)
Pop Up
Settings:
Enable Reminders
Enable Diet Reminders -o
Enable Exercise Reminders -o (Buttons)
Allow Diet Reminders every [ ] days
Allow Exercise Reminders every [ ] days
Ok

Notifications UI

Notifications

Free to Use Stock Image

You have not logged eating any fruit for [] days in a row

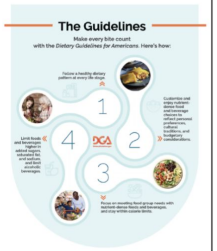
Ok

Free to Use Stock Image

You have not logged eating any vegetables for [] days in a row

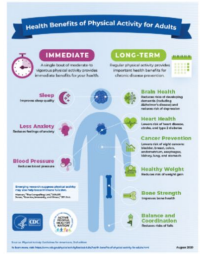
Ok

You have logged eating high sugar foods for [] days in a row



Ok

You have not logged any exercise for [] days in a row



Ok

graphic provided by dietaryguidelines.gov

https://www.dietaryguidelines.gov/resources/downloadable-graphics

graphic provided by cdc.gov

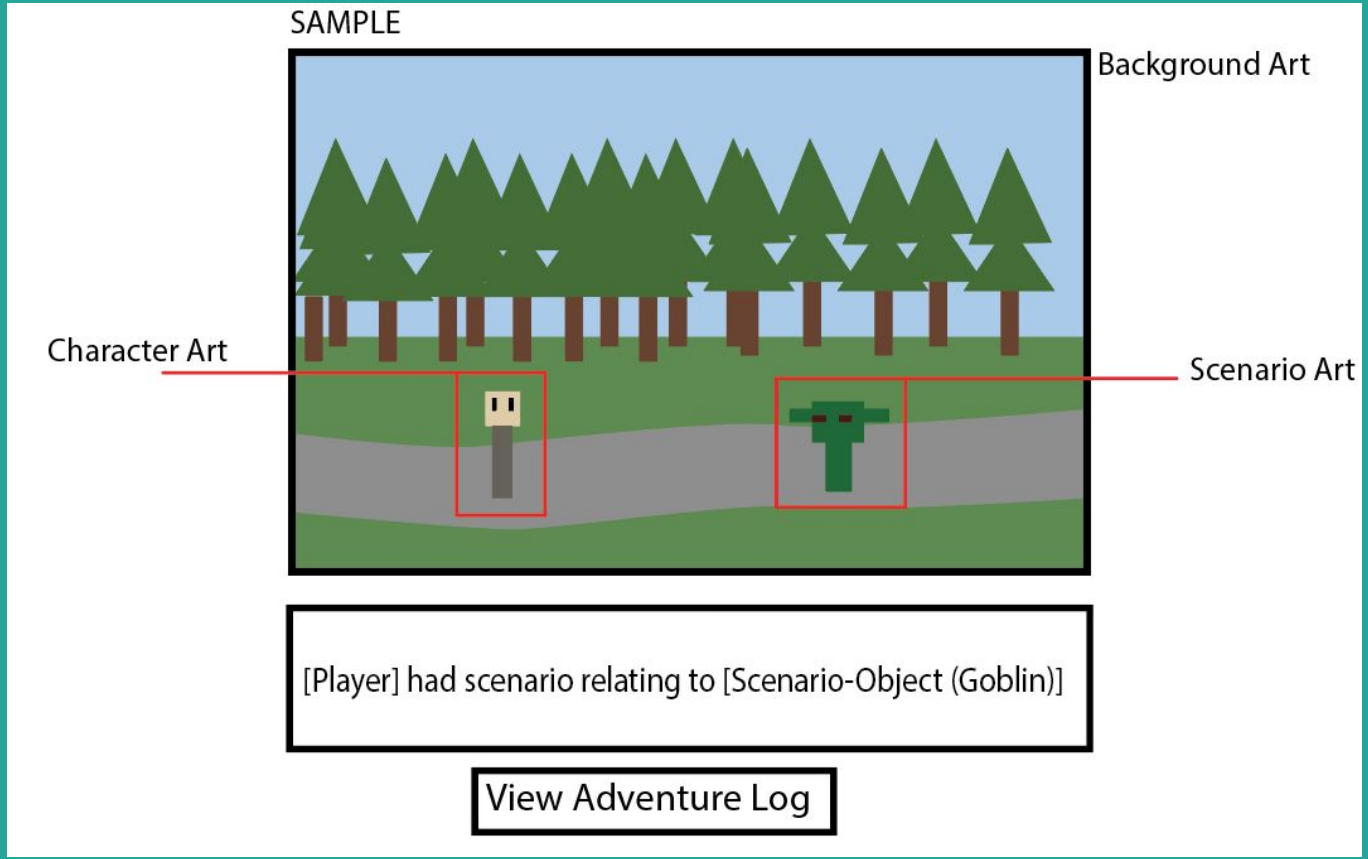
https://www.cdc.gov/physicalactivity/basics/adults/health-benefits-of-physical-activity-for-adults.html

Note: If copyright is an issue then can be replaced, although as they are obtained from .gov it should be fine as it appears that government documents/websites/etc are public domain.

We likely would add more notifications depending on the end food groups chosen, currently focusing on vegetables/fruit and sugary foods due to our Reference article.

Note: Our notifications don't focus on portions or specifics as that would require logging much more specific and lengthy information, and there are many apps that can do this as if someone was motivated to log their diet that extensively they likely would just use those. Increasing the specificity of the diet or exercising logging would increase the barrier of usage, and we want the reminders to still be useful/clear to someone not motivated enough to track and log the specifics of their diet.

Sample Game UI



Task 6: Create a Test Contribution

This will include our test cases that we should be able to implement and test. This will be our guide to show our progress with the app. The user's amount of input is minimized to prevent mistakes and make the process easier which reduces the number of tests we need to do. However, the primary function of the app relates to a database, and so a relatively large number of test cases need to be made to ensure none of the user's input could potentially mess up the database which could result in other functionalities being broken, such as the graphing functionality.

Discussion of Team Contribution

- Joshua Breininger focused mostly on working on descriptive and planning elements of the milestone. He was the primary communicator with the clients and participated heavily in writing the documents.
- Daniel Bornemann was in charge of the Software Test Plan and the Software Requirements Document, as well as communicating with the Faculty Advisor.
- Phi Duong took part in the document writing and checked and edited them for format and errors. Phi also focused more on getting the Android emulation up and running.

Plan For Milestone 2

Task	Josh	Phi	Daniel
1. Implement, test, and demo a basic Android UI.	50%	25%	25%
2. Implement, test, and demo an interactive pop-up with an input field.	25%	25%	50%
3. Implement, test, and demo saving inputted information into an in-app database.	33%	33%	33%
4. Implement, test, and demo an in-app database structured around a calendar.	25%	50%	25%
5. Implement, test, and demo retrieving information from the database.	33%	33%	33%

Client Feedback

- Add option for the user to decide a name for their avatar (added to documents.)
- The UI looks understandable but plain. (Addresses with themes)
- Themes should be made specifically to be appealing to look at and not use exhaustive or unpleasant imagery.
- Aside from that, it appears to fulfill the requested functionality.

Questions?