Team Members:
Tyler Dionne (tdionne2021@my.fit.edu), Kendall Kelly (kelly2021@my.fit.edu), Braden Corkum (corkumb2013@my.fit.edu)

Project Advisor:
Philip Chan, pkc@fit.edu

Project Title:
Tasteful Panthers: Food Recommendation at Dining Halls
Client: Philip Chan

Milestone 1 Progress Evaluation

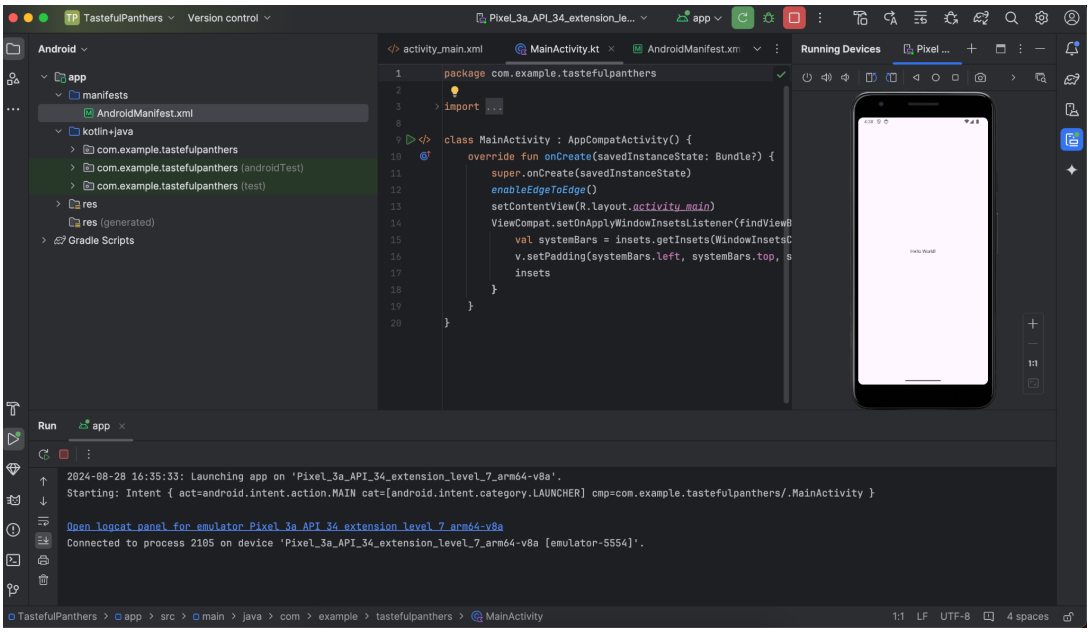
1. Progress of Current Milestone:

| Task | Completion % | Tyler | Kendall | Braden | To Do |
|---|---|---|---|---|---|
| Investigate Tools | 100% | 50% | 50% | 0% | N/A |
| Hello World Demos | 100% | 50% | 50% | 0% | N/A |
| Resolve Tech Challenges | 100% | 50% | 50% | 0% | Become a little more familiar with Android SDK and ADB, as well as Kotlin. |
| Select Collaboration Tools | 100% | 33.3% | 33.3% | 33.3% | N/A |
| Requirement Doc | 100% | 50% | 50% | 0% | N/A |
| Design Doc | 100% | 100% | 0% | 0% | N/A |
| Test Plan | 100% | 0% | 100% | 0% | N/A |

2. Discussion of each completed task:
    - Investigate Tools - To investigate tools we looked at our different options for android application development taking into consideration environment setup as well as emulation features. Our initial and current choice for our application development environment and emulation tool is Android Studio. We found an alternative application for android application development which is IntelliJ IDEA and an alternative application for

Android emulation which is genymotion. After looking into both of these tools we found that neither option is more appealing than Android Studio because of the fact that IntelliJ IDEA is essentially Android Studio with a different name and less features (no emulator) and genymotion is simply an entire application which does the same thing that a piece of Android studio does for us (the emulation). We also looked into our choice of programming language Kotlin and made sure it had capabilities for gps support in the Android application so we do not run into any problems later on in the project. We also ensured that our Kotlin application will be able to interact with a mySQL database. Therefore by investigating our tools we made sure that we will not have any major issues in the future for our application,

- Hello World Demos - We created a hello world demo android application in Kotlin using Android Studio as both a development environment and an emulator to test and debug. The demo is shown below:




We planned on also creating a hello world demo on an alternative application to Android Studio but unfortunately the alternatives we found did not have both emulation and a development environment built in. To create a hello world demo on these alternative tools we would have to essentially export the code we used to make this hello world demo into IntelliJ IDEA show the code is in this application then export the code out once again then find a way to export the code into a emulated Android device on genymotion then launch it. This felt redundant because it would be essentially the same application running on a different emulator that we had already decided not to use. This is an example of why using the other

tools would make the development process much more slow and complicated when it does not need to be. We also prepared some sample code of what it will look like when we implement GPS features into our application as well as sample code to demonstrate how we will interact with a mySQL database using our Kotlin android application. Given that much of the other code needed to bring all of the functionality of the application together is missing, these hello world demos for the GPS and mySQL database interactions at this point are infeasible due to the difficulty it would take to integrate these features into some sort of basic, demo environment. We attempted to implement a demo application that uses GPS features. This helped to understand the general procedure of how we will get this to work in the future. The steps are shown below:

1. Add the following lines under their respective categories in `libs.versions.toml` file in the `gradle` folder:
   [versions]
   play-services-location = "21.0.1"

   [libraries]
   google-play-services-location = { group = "com.google.android.gms", name = "play-services-location", version.ref = "play-services-location" }

2. Then in the applications build.gradle file add the following line under dependencies:
   dependencies {
   implementation(libs.google.play.services.location)
   …
3. Put necessary contents in main_activity.xml and MainActivity.kt
   }
4. Put the necessary permissions in the AndroidManifest.xml

   <?xml version="1.0" encoding="utf-8"?>
   <manifest xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools">

   <uses-permission
   android:name="android.permission.ACCESS_FINE_LOCATION" />
   <uses-permission
   android:name="android.permission.ACCESS_COARSE_LOCATION" />

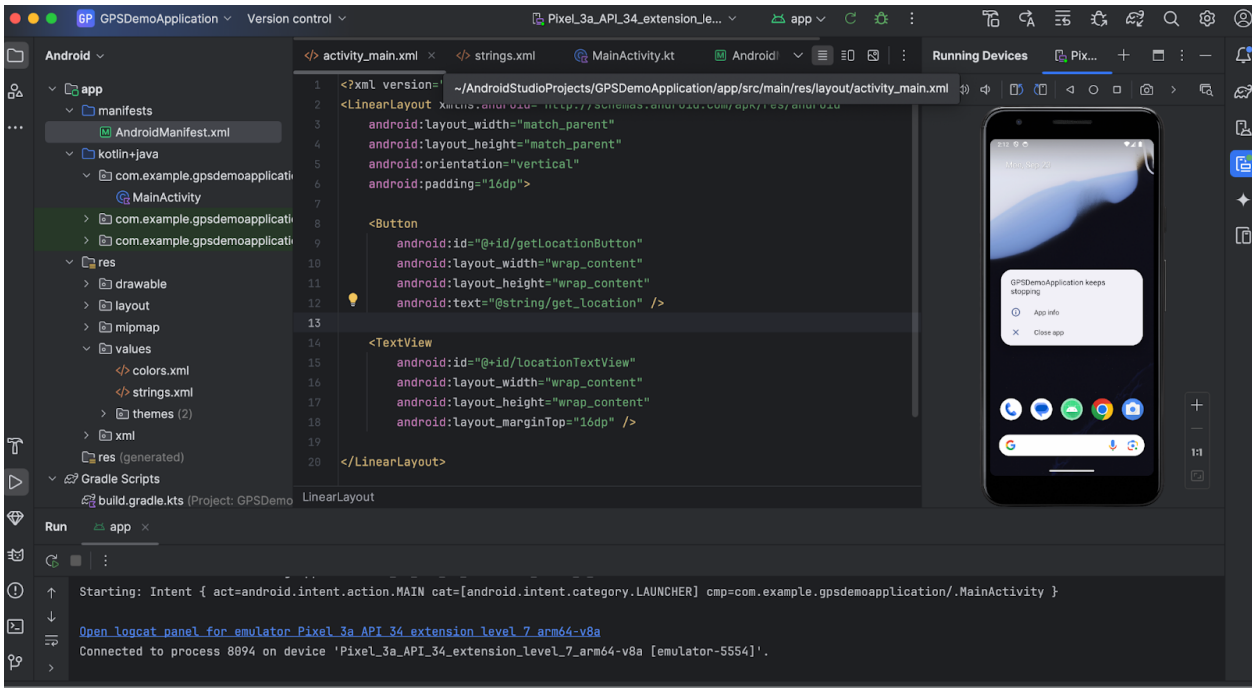
   <application

…

5. Add the following line to /res/values/strings.xml:
   <string name="get_location">Get Location</string>

6. If get issues with unresolved reference R add this in MainActivity.kt
   import com.example.gpsdemoapplication.R

7. Go to build > clean build and then build > rebuild project




As shown above this still needs more testing debugging in order to be fully functional.

- Resolve Technical Challenges - The first technical challenge was to become more familiar with our chosen Android environment, Android Studio. We have installed the application and become familiar with how to start projects, navigate through files, and work the emulator. Obviously we will end up with much larger files than simple demos, however, we currently have a good base knowledge of how the environment works. The second challenge was to become more familiar with Android SDK and ADB. Due to the fact that our demos are very simple we will have to spend more time becoming familiar with the tools we will be using as the code gets larger and more complex. The same can be said for becoming more familiar with Kotlin which is the third technical challenge. We were able to

write small “Hello World” demos and understand how other simple commands work but as for more complicated commands, this will have to be something we learn as we go. It’s hard to fully understand aspects of the language when not actively using it to write larger programs. Overall, we are fairly comfortable with Android Studio and have a basic understanding of Android SDK and ADB as well as Kotlin, however, we will need to build our knowledge as we continue on this project.

- Select Collaboration Tools - As a group we selected the following collaboration tools. For general communication within the group regarding tasks, due dates etc. we chose the mobile application “discord” which allows users to message, call, and video chat one another. Another collaboration tool we chose for the purpose of sharing code and version control is “Github” which is a platform for hosting and managing code repositories using Git and it facilitates version control, collaboration, and code sharing among developers. Another collaboration tool we chose as a group is a shared google calendar to keep track of due dates and other tasks.
- Requirement Document - We created a requirement document to specify what the system should do. This document contains functional requirements (personalized recommendations, review system, gps-based notifications, contests and leaderboard, meal suggestions, staff interaction with reviews). Non-Functional Requirements (performance requirements, usability requirements, reliability, security, compatibility). Interface Requirements (user interface, system interface). Lastly the document contains specific requirements which define specific system requirements our team came up with. An example of one we came up with was “The system should send a notification to the user when he/she is within 100 feet of the panther dining hall.” This document helped us to understand what our system needs to do so we can be sure to implement every desired feature by defining the requirements before starting development.
- Design Document - We created a design document to define how we will design a system to satisfy all requirements and implement all features. This includes system overview, system architecture diagram, modules (classes) functionalities & interface (methods), sketch of gui & screens, database (ER diagram, tables, keys) diagram.
- Test Plan - We created a test document to give a general overview of each key requirement, their possible test cases, and possible outputs. The document starts with a brief introduction to what the application is, what it entails, and the scope of the design document. We chose to outline the testing of all the key features, as well as a few more. The features

included in this document are as follows; application GUI, login system, personalized recommendations, review system, gps notifications, contests/leaderboards, user suggestions for meals, and the interactions staff have with reviews. Each testing overview for each requirement includes test case identifiers, objectives, inputs, expected and unexpected outcomes, and any environmental needs or dependencies, as per the IEEE Standard for Software Test Documentation.

3. Team Members Contributions to Milestone 1:
    - Tyler Dionne - Completed the design document. Completed 50% of the requirement document. Assisted in finding an alternative to Android Studio (IntelliJ IDEA + Genymotion). Worked to create hello world demos and the GPS location services demo. Worked to test the tools and languages we chose. Revised the requirement and design document after the M1 meeting.
    - Kendall Kelly - Revised parts A and B of the test document as well as writing and completing the introduction and most of the details section. Completed 50% of the requirements document. Assisted in researching IntelliJ IDEA and looking into Kotlin/Java. Revised the requirement as well as the test case document after the M1 meeting.
    - Braden Corkum - Completed parts A-B in the first draft of the test document.

4. Task Matrix for Milestone 2:

| Task | Tyler | Kendall | Braden |
|---|---|---|---|
| 1. Implement, test & demo diners entering/viewing reviews | Needs to implement entering and viewing reviews. | Needs to demo the logic for entering and viewing reviews. | Needs to test the logic for entering and viewing reviews. |
| 2. Implement, test & demo diners searching reviews | Needs to implement diners searching reviews. | Needs to demo diners searching reviews. | Needs to test diners searching reviews. |
| 3. Implement, test & demo kitchen staff search/view/ comment on reviews | Needs to implement kitchen staff search/view/ comment on reviews. | Needs to demo kitchen staff search/view/ comment on reviews | Needs to test kitchen staff search/view/ comment on reviews |

5. Discussion of each planned task for the next Milestone
    - Task 1 - The first task is to allow users to enter reviews and view reviews that have been submitted.
    - Task 2 - The second task is to allow users to be able to search for reviews by tags.
    - Task 3 - The third task is to allow kitchen staff to be able to search, view, and comment on reviews.

6. Date(s) of meeting(s) with Client during the current milestone:
    - September 11, 2024
    - September 25, 2024
7. Client feedback on the current milestone
    - See Faculty Advisor Feedback below

8. Date(s) of meeting(s) with Faculty Advisor during the current milestone:
    - September 11, 2024
    - September 25, 2024

Faculty Advisor feedback on each task for the current Milestone

Wed September 25 2024

2 input to personalized recommendation

1. User account preferences tags
2. User past reviews

Using reviews to differentiate profiles

User A User B

Rated: 5 Rated: 5

Rated: 5 Rated: 5

? Rated: 5

Recommend user A the third meal

User can leave these things on review

User can:
Enter Review

- Entering the 5 possibile actions listed below
- Need a submit button

Search reviews

- Search for tags (specific tags)
- Search for reviews higher than a specific star
- Optionally search on words contained in text comment reviews

View Review

- After search we can see the review
- Use personalized recommendation to allow user to view reviews for that recommendation
- Be able to click on items from the leaderboard and be able to look at those reviews

Recommendation
2 inputs to personalized recommendation

3. User account preferences tags
4. User past reviews

Want to output recommendation, explanation of why we pick and Provide a link to the reviews of that food item for that day

- Explain the recommendation via the algorithm
- (you matched with user a which liked f2 and f3 and he/she also likes f1 so we recommend you try f1)

- Stars
    - Can be paws or burger icons (any emoji)
- Text comment
    - Come up with a max character count so DB dont have to store a bunch
    - Ex. 100 characters
- Image/picture (google review link to image or some other host platform)
- Link to video (youtube)
- Tags
    - Tags: good for taste buds, health, restricted diets, playing sports, studying, paying attention in class

Notification

- Entering PDH (here's you recommendation of the day) (this notification include the outputs for the personalized recommendation)
- Student is in PDH about halfway through meal

- Use GPS location data to look for two timestamp to enter and leave (this will give a duration) Need to determine half way through meal. Want them to still have meal when we send the notification
- We will try to get them to leave a review (ex. Was your food horrible?)
- Want them to easily get to a “leave a review screen” from the notification (refer to section 2 on leaving reviews)

Kitchen Staff Suggestion

- Users can make suggestions to kitchen staff on meals for the upcoming week.
- In meal suggestion
    1. Name of dish
    2. Have section for reasoning
    3. Optionally for user is to enter link for recipe

(Competition between dishes)
Meal Suggestion Leaderboard:

- Users can vote on meal suggestions
- Users can see the rankings
- Ranks are based on votes by other students on meal suggestions
- Kitchen staff can view the suggestions and can cook the suggestion (kitchen staff can enter announcement that this dish is going to be available on x day)
- Once meal has been offered it gets removed from the leaderboard to clean it up

Need to add # votes , recipe, and reasoning to meal suggestion database

(Ranking of the users)

Faculty Advisor Signature:  Date: 9/30

Evaluation by Faculty Advisor

- Faculty Advisor: detach and return this page to Dr. Chan (HC 209) or email the scores to pkc@cs.fit.edu
- Score (0-10) for each member: circle a score (or circle two adjacent scores for .25 or write down a real number between 0 and 10)

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| John Smith | 0 | 1 | 2 | 3 | 4 | 5 | 5.5 | 6 | 6.5 | 7 | 7.5 | 8 | 8.5 | 9 | 9.5 | 10 |
| Jane Doe | 0 | 1 | 2 | 3 | 4 | 5 | 5.5 | 6 | 6.5 | 7 | 7.5 | 8 | 8.5 | 9 | 9.5 | 10 |
| Mark Jones | 0 | 1 | 2 | 3 | 4 | 5 | 5.5 | 6 | 6.5 | 7 | 7.5 | 8 | 8.5 | 9 | 9.5 | 10 |

- Faculty Advisor Signature: _____ Date: _____