Bonus features

1) **Timing-based play**
   The player would have a limited time to collect the coins. If they collect all coins on time, a bonus of some gold will be added into their bank account. The amount of bonus gold will depend on the time interval that the player will choose. The smaller the interval, the more gold will the user get for accomplishing this challenge. The chosen time interval will have an upper bound, in order to avoid situations when the player would get extra gold for making it in 20 hours, and a lower bound to provide some limit on the amount of extra gold that the player can obtain by using this feature. The mathematical function that will take the chosen time interval as an input and will give the number of extra gold as an output will be defined differently on distinct parts of its domain. For illustration - assume the player fulfills the challenge on 4 distinct days in all of the following time intervals: 30 minutes, 29 minutes, 10 minutes and 9 minutes. Let us call ‘x’ the extra amount of gold that the player makes if they make it in 29 minutes instead of 30 minutes. This should be smaller than the extra amount of gold that the player makes if they make it in 9 minutes instead of 10.
   Coins that the player does not pick in the time interval will be left untouched and can be picked after the challenge. This bonus feature can only be used once a day.
2) **Recording the distance walked by the user**
   The app will keep track of the distance the player has walked while playing the app. An approximate amount of calories burnt while playing the game will be shown as well. This will be a very approximate value based on player’s weight, gender and age. If the player does not wish to provide some of this information, some default value will be chosen that will be used for calculating the calories. Neither speed, nor weather, nor any other property will impact the calculation for the number of calories burned by playing. In order to record the distance, a reasonable time interval will have to be set for requests asking for the location so that the distance is updated often enough. The time interval between 2 consecutive requests cannot be too small, because that could cause problems if the internet connection is not sufficient or the phone is older and does not have such a strong processing power.
3) **Friends list & Leaderboard**. Friends list will be a simple list of players that a player can add in order to make sending coins easier. This list will be sorted based on their names. Leaderboard will display a list of players from the friends list sorted based on the number of gold that each of the player has in the bank. If a player decides to keep their money stored in a different currency, leaderboard table will transfer this currency into gold based on the current exchange rate.

Programming language

I decided to choose Java as the programming language for this app. Justification for this choice is provided below:

1. Java is a much more broadly used programming language. It is likely that when this app is handed over to a team of developers, their knowledge of Kotlin could be limited. This would require extra time for learning this language in order to maintain and develop this application further. If the team has not been chosen yet, then it is also more probable that a team with appropriate Java knowledge is easier to find than a team with Kotlin knowledge. Further, experienced developers in Kotlin are still quite rare[1], so finding an experienced developer - such as the team leader, will be a lot easier for Java than for Kotlin.
2. Being one of the most famous programming languages ever made, Java has a lot bigger developer community. On Stack-overflow, there is 1.37 million questions tagged with Java, while only around 8000 questions about Kotlin.[2] Hence, finding help on the internet to figure out problems is easier in Java than in Kotlin.
3. On average, Kotlin has slower compilation speed than Java. “For clean builds with no Gradle daemon, Java compiles 17% faster than Kotlin. For clean builds with the Gradle deamon, Java compiles 13% faster than Kotlin.”[3] However, for partial builds with incremental compilation the compilation speed of both is very similar.
4. Java can be used for cross-platform development. It is quite likely that the team would like to release a version of this mobile app for iOS. If this is the case, then this will be easy as Java can also be used to develop application for iOS.
5. Java has checked exceptions. Even though Java is famous for the NullPointerException, which arguably could be considered a downside of the language, many programmers have got used to working with checked exceptions that Java provides and consider them an advantage. Kotlin does not provide them.
6. Java has primitive types that are not classes. Working with primitives instead of objects has 2 significant advantages. It is memory efficient and it has a better runtime performance. Primitives store only value, while objects have to store a reference to the object, the value in the object and a reference to the class object. In addition, Java uses extra memory to support garbage collection for objects.[4]

[1,2] https://www.netguru.co/blog/kotlin-java-which-one-you-should-choose-for-your-next-android-app

[3] https://medium.com/keepsafe-engineering/kotlin-vs-java-compilation-speed-e6c174b39b5d

[4] https://www.javaworld.com/article/2150208/java-language/a-case-for-keeping-primitives-in-java.html

7. Java is still the most used language for Android development as well as a widely used language for development of web applications, GUI based programs, database connectivity, image processing and many more.[5] This makes it more demanded than Kotlin by IT companies which makes it a better match for junior developers. A team of junior developers under some experienced senior developer could also be cheaper for developing this app further. Arguably, a senior developer should pursue learning Kotlin as it might become the programming language for Android development in the future.
8. Big Java projects will require maintenance for many years to come. As already mentioned, Kotlin might replace Java's role for Android development in the future. Even if this happens, it will take some time. However, there will still be many projects that might not be so easily transferred into Kotlin for architectural reasons and will require maintenance in Java. An experienced Java programmer in this case could become highly valuable and developing further this app in Java will provide additional experience of this language.
9. Java has ternary operators.[6] Java's code is usually more robust than Kotlin's code which is a disadvantage. However, ternary operators, which Kotlin does not support, are one of the things that can make Java's code a bit more concise.

[5] https://www.quora.com/What-can-Java-programming-do

[6] https://kotlinlang.org/docs/reference/control-flow.html#if-expression

Timetable

Week	To do:
2	Upgrade GitHub account and create a private repository for this coursework. Choose the programming language. Define unspecified rules for the game (design decisions).[1] (annotation below) Come up with bonus features.
3	Make layout of the different pages the app should have and make wireframes for each of the pages.
4	Implement Mapbox to display the map. Render coins onto the map using GeoJSON. Make a firebase account and learn how to use it.
5	Create a GUI based on the wire frames.[2] (annotation below)
6	Do user authentication. Implement the user account.
7	Implement the game - trace user's location, pick up coins within 25m radius, collect distance walked by the user so far.
8	Implement the money flow - a coin that has been picked up will be automatically added into the wallet. A coin cannot be picked if the wallet is already full. A coin can be put into the bank using wallet.
9	Implement the 3rd bonus feature (friends list and leaderboard). Adding a friend will require verifying that the account to be added is registered. Implement money transfer between 2 friends.
10	Implementat the 2nd bonus feature (distance walked by the user).
11	Implement the 1st bonus feature (time-based play). Do proper user testing.
12	Leave this week for finishing things that have not been done.
13	Leave this week for studying for other exams.

Week	Have already:
1	Installed Android Studio
2	Upgraded GitHub account and created a private repository for this coursework. Chosen the programming language. Defined unspecified rules for the game.
3	Come up with bonus features. (I was ill this week)
4	Made layout of the different pages the app should have and made wireframes for each of the pages. Implemented Mapbox to display the map. Rendered coins onto the map using GeoJson. Have created a firebase account.

Annotations from the timetable

[1]**Define unspecified rules (design decisions):**

1. Coins put into the bank can be either transformed into the gold or kept in their respective currencies. However, the trace of individual coins will get lost. Instead, for each of the currencies the user will be able to see the sum of their coins they have in the bank for this currency. Once a coin is put into the bank, it cannot be withdrawn. E.g.: If I put SHIL coins of values 8 and 9 into the bank, I will be able to see that I have 17 SHIL, but will not be able to tell if it was the sum of two coins with values 8 and 9 or of three coins with values 3, 6, 8.
2. Money can be received by and send to friends. This way, a coin can be put into the bank by the same player more than once a day.
3. Coins can be kept in the wallet for more days. However, the wallet will have a capacity of 50 coins and the coins will expire (disappear from the wallet) after 3 calendar days. This opens space for speculations - the player can keep coins of high value in the wallet if their currency has a low exchange rate hoping that the next day they will become more valuable. Any coin in the wallet can be dropped. Once dropped, it can not be picked again.

[2]**Create a GUI based on the wire frames for all parts of the application:**

1. After the application is launched, the user can either log in or register as a new player. User's session will be set to 72 hours, so that if the user uses the app actively, they do not need to log in every time.
2. After authentication, the main menu will be displayed. User can hit:
    1. Play - the map will be displayed and user can start playing

2. Bank - user's balance in gold and today's exchange rates will be displayed
3. Wallet - user's list of coins will be displayed. One coin will correspond to one row in the list. The row will consist of the coin's currency, value, expiry date and a possibility to 'drop' the coin and 'put in bank'.
4. Leaderboard - list of players from the friends list will be displayed displaying their name, amount of gold they have in the bank and their order. The list will be sorted based on the amount of gold they have.
5. Friends list - at the top, a button to add a new friend will be displayed. Below, the list of friends will be displayed. This will be sorted based on their names and next to every name will be a button to send coins to that friend.
6. Time challenge - an input for the time interval and a potential bonus in the form of gold will be displayed. Below will be a button which directly will start the game.
7. Account - user's personal information will be displayed. This will include name, email, password, gender, weight and age. I am not sure at this point which information will be the user allowed to modify.