Project Report:

Project Name: Where Am I?
Group Members: Amanda Isenor, Oluwafikayo Oyinloye, Jared Quinn
Project Description: The concept of this project is to periodically report one's location to a desired number in the form of a Google Maps link sent via SMS text messaging. The idea is that when you set out on an adventure you can tell the app a time interval (30 minutes, for example) and a phone number (maybe your moms), that way every 30 minutes your mom will get a text containing your latitude and longitude.

Operating Instructions: The user is first prompted for access to their location, contacts, and SMS messages. These permissions are all necessary for the app to properly function.
The user is initially shown a screen of Google Maps displaying their current location and the only option is a button that says START A NEW ADVENTURE. The user is posed two questions upon pressing this button, a time interval at which to send text messages and a phone number to send those messages to. An error warning will appear in the form of a toast if the user does any of the following:

1. Does not enter a valid phone number (10-11 characters of only digits)

2. Does not enter any phone number

3. Does not provide a time interval (interval cannot be zero hours zero minutes)

Once the user has given sufficient data they may press START which takes the user to a third and final screen. In the final screen the user can see a countdown timer that displays the time remaining in their interval until the next text message will be sent. Below that is a text field displaying the previously entered phone number, this is shown for the user to ensure they entered the correct number. Every time the countdown timer

reaches zero a toast appears to inform the user a text message has been sent, it does not consider whether the user has service or not. The timer is then reset to the given time interval and the process repeats. The app can be minimized and phone locked and will continue to work, if the app is closed completely the messages will not send.
There are 3 buttons on the final screen, STOP, RESTART, and END ADVENTURE. Stop and restart can manipulate the timer allowing the user to stop the timer and restart it without needing to go through the set-up process a second time. Ending the program will return the user to the main screen displaying Google Maps. The user will need to re-enter a phone number and time interval if they wish to resume.
The final build of the app is was tested on Android API 28 and should be compatible with most android devices from android 4.0 upwards.

Design Implementation:

The end product of the app includes certain features such as

- Countdown timer the user can set that will send a message on 0
- Reset automatically after sending a message
- Google Map view within the app to see your location easily
- Pull location data from the phone to place in text

To achieve the above features, a few out of class research had to be done. These out of class features include:

Google Maps: With the aid of google API, the app is able to seamlessly record the current location of the user and periodically send the location(s) without any hiccup. To implement Google Maps, we required an access key from Google to use the Maps API. For the majority of the onMapReady method, we learned from a video on YouTube and the Google Support pages (https://www.youtube.com/watch?v=NHXa96-r8TY https://developers.google.com/maps/documentation/android-sdk/start). The most difficult part was getting the API key that was required in the manifests. We tried to use maps without using an API key as it required a credit card number. Eventually we caved and provided Google with Mandy's credit card, got the API key, and things became clearer. More information about this can be seen in the MainActivity.java Class

Countdown Timer: The countdown used in the app could be consider new but it was implemented thanks to our knowledge of threads. The countdown timer class acts as another thread which counts down from the user specified number to zero. More information about this can be seen in the StartNewActivity.java Class

SMS Manager: Thanks to android studio's extensive library, this allowed the app to send sms messages, bearing in mind Standard messaging rates do apply. This does require the user to accept a few permissions asked upon first use of the app. More information about this can be seen in the DuringActivity.java Class

NumberPicker: To effortlessly allow the user to pick an appropriate amount of time required to send their location, this was chosen as the best method of implementation rather than the traditional method of manually inserting a number which could cause potentially cause bugs. Once again, thanks to Android studio's library, a createPicker() class was implemented

Content Providers: This was only used to provide a list of contacts to the editText with the StartNewAdventure Activity. It is fairly straight forward, while not covered in class the book does make reference to an android tutorial on the subject. (**https://developer.android.com/guide/topics/providers/content-provider-basics**). This page as well as a contacts specific android tutorial (**https://developer.android.com/training/contacts-provider/retrieve-names**) were used in the implementation of the content provider.

Case Uses:

- Long Adventures or trips where taking out your phone is an inconvenience but you want to heck in with someone periodically. Eg hiking trips, long walks
- For safety, if a contact gets the same location twice in a row they know where to find you
- When you need a pick up but are unfamiliar with your surroundings or just lost
- To ease mum trust issues and let her know your location periodically.

Implementation issues and Bugs:

Every file associated with AppCompatActivity becomes an error the majority of time that a fresh copy of the project is pulled from github. The only fix that we have found is to rename the ‘libraries’ directory and then to revalidate caches. It is not known if this is a bug with our code or if it is perhaps a problem with Android Studio. It’s a great mystery.

There was also a totally separate problem that your (Professor Godbout’s) version of Android Studio was throwing when you were trying to help us. I do not know anything about what was causing that error and I am hoping that it not still an issue.

Citations:

Android Developers. (2018). *Retrieve a list of contacts | Android Developers*. [online] Available at: https://developer.android.com/training/contacts-provider/retrieve-names [Accessed 15 Dec. 2018].

Android Developers. (2018). *Content provider basics | Android Developers*. [online] Available at: https://developer.android.com/guide/topics/providers/content-provider-basics [Accessed 15 Dec. 2018].

YouTube. (2018). *Google Maps Tutorial : Part 1 (Android Tutorials)*. [online] Available at: https://www.youtube.com/watch?v=NHXa96-r8TY https://developers.google.com/maps/documentation/android-sdk/start [Accessed 15 Dec. 2018].

Griffiths, D. and Griffiths, D. (n.d.). *Head first Android development*.