SafeWalk

By Samantha Puterman, Alexandra Pineiro, Byron Mitchell, and Jean Marc Achkar

Project Overview:

The goal of our project is to direct users on safe routes to their desired destination, and provide a service that notifies police, or an emergency contact, in case of emergency. It is mostly helpful for college students coming back from the library at late hours, tourist wandering around an unknown country, and women walking back home from work. Our app has two main views that are controlled by a tab controller. These are: the user profile and a map/location. The User profile is the default screen when the app is opened. The user inputs their information and then saves it using the save button, this information will be saved even if the user exits and relaunches the app, but can be changed any time. There is also a "Notify the Police" button, which can be used any time the user enters the app in case of an emergency. On the second view, we see a map of the user's location. There is a Go Home button which will calculate the estimated time of arrival and the safest route to the user's address. After a certain amount of time, the user will be notified, asking if they are okay. If they do not respond in a certain amount of time, the police or an emergency contact will be contacted. Once the user has reached their destination, the user should press the "I Arrived" button so that he does not get more notifications.

Technology and Functions Used:

We decided to build an IOS app using Swift5 in Xcode, as it would be easily accessible to us. The reason we decided to code in Swift5 using the Xcode IDE was because it includes a lot of functionalities that made designing the app a lot easier. In fact, since our app includes a map, and we were unable to use a Google Maps API (see more in Challenges) we realised that using Xcode, we can simply import the Mapkit & CoreLocation libraries and get most of the features

needed to make our app. With these libraries, we were able to track the user's location, calculate the fastest route from the user's position to their desired destination (even though our main idea involved around calculating the safest route, we couldn't find any free algorithms that accomplished that), and get the address at any point in the map. We also implemented a **Go** button that calculates the fastest route between the user's location and the location of the **Pin** in the map. This button also starts a timer that when it reaches 5 minutes, it will send the user a notification asking if he/she is okay. If the user doesn't answer in a reasonable amount of time, lets say 30 seconds, the app will automatically alert the police. To stop the app from sending notifications and prevent it from automatically alerting the police, once the user gets to his/her destination, the **Arrived** button has to be pressed, terminating the current route in progress.

Challenges:

The first challenge we faced was trying to choose a Map SDK for our application. With Google being the most famous and used one, we thought implementing a Google API would make our app perfect. However, what we forgot to consider is that working with a Google API would probably cost money, require a lot of time to understand, and is not as simple as we thought it seems to be. In fact, the first thing we noticed is that to get access to several APIs for testing, we had to create a Google Maps Platforms account, in which we received an amount of credits to use on Google Maps functions and services to implement on the app. After a long while of researching and watching Youtube tutorials on how to simply implement a map on Xcode using Google, we concluded that it would be much easier if we just used the built-in features that Xcode offers to directly implement a map on the storyboard of our app.

Since the group was divided into 2 smaller subgroups, one working on the map and the other working on the user interface, each of us was working on a different Swift file, hence different Xcode workspaces. When each of the subgroups finally finished their part of the code, we came to the realisation that we might face some issues trying to merge both codes into one, and as obvious as it is, we did. The first few problems were that a map would display on the second tab but none of the functions were working. Then, after fixing that issue, we faced another challenge which was that the application was not receiving correct data, thus show the wrong location of the user.

Other types of challenges we face include the issue that we were unable to have the application reroute the user's directions if the user deviated from the initial path. This was due to the limited functions that the MapKit included. We also tried to use the user's address input on the first tab as a variable in the second tab, but unfortunately, we were unable to transfer the data from one view controller to another. Not to mention that we had difficulty coding a function that would geocode the user's address input, and turn it into readable coordinates for the map to read and set the destination.

Moreover, when it comes to messaging the authorities or an emergency contact, we were not able to test that our code worked since the messaging cannot be tested in the simulator as it does not have a SIM Card and when we tried to download the app to our phones with the messaging notification included, we found that Xcodes requires an additional version, which costs a lot of money, in order to use the messaging part of the app in an actual phone. Because of this we decided to display a message that says "Contacting Authorities" in the situations where the user needs help or does not answer to the Are you okay? prompt. Despite not being able to

test it, we were able to write the code, as seen below, that will do so when we have the resources to test this function.

```
import UIKit
import MessageUI
class SecondViewController: UIViewController,
    MFMessageComposeViewControllerDelegate {

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view.
    }

    var timer = Timer()
    var isTimerOn = false
    var duration = 0
    var phoneNumber = 7869731005;


    func displayMessageInterface() {
        let composeVC = MFMessageComposeViewController()
        composeVC.messageComposeDelegate = self

        // Configure the fields of the interface.
        composeVC.recipients = ["7869731005"]
        composeVC.body = "I need help"

        // Present the view controller modally.
        if MFMessageComposeViewController.canSendText() {
            self.present(composeVC, animated: true, completion: nil)
        } else {
            print("Can't send messages.")
        }
    }

    func messageComposeViewController(_ controller:
        MFMessageComposeViewController, didFinishWith result:
        MessageComposeResult) {
    }
}
```

For the messaging feature of the app we would have used the MessageUI as shown above which is a framework that provides specialized view controllers that allow the interface to send

emails and SMS text messages. After that we just had to customize the message and add the phone number of the recipient.

Despite having faced all these problems, we were still able to have a functioning app that saved the user's name and address, that notified the police in case of emergency, and that included a map where the user could manually pinpoint the destination and press go, to have a path light up on the route to follow.

Things to improve:

- If we had more time, we would have implemented a way for the map to sense when the user got to the destination without the need of having a manual input of arrival.
- Possibly include service for people nearby to walk user home
- More accurate location and safer routes
- Request access to user's Apple Health application where all relevant information are present in case of emergency
- Better interface with the user and easier input of destination
- Used the home address provided in the "User Profile" section and automatically calculate a route.
- Test the text messaging feature (explained in challenges)