CSE 237D: Stop The Steal - Bike Anti Theft System Milestone Report

Akshay Prabhu, Gagan Gopalaiah, Grishma Gurbani

May 18, 2023

1 Overview

Our research is focused on creating a highly effective anti-theft system for bicycles that leverages the advanced capabilities of old smartphones. The system operates in two modes: locked and unlocked. In unlocked mode, the user has complete freedom to use their bike without any limitations or notifications. However, when the user engages the lock mode, the smartphone device, equipped with sensors such as GPS and accelerometers, will begin monitoring the bike's location and movements. If the system detects any unauthorized movement, such as a thief attempting to steal the bike, a loud alarm will sound to alert nearby people. Simultaneously, the system will send a notification to the user's phone indicating the theft attempt and providing GPS coordinates of the bike's location. This will enable the user to contact the authorities and recover their stolen bike.

1.1 Project Approach

This research paper outlines the development of a bike locking system that utilizes an old Android device, cloud middleware, and a client-side application. The project is divided into three distinct phases, each of which is discussed in detail below.

1.1.1 Old Android Device

In this phase, we develop an application for an old Android device that interfaces with various sensors, such as the accelerometer and GPS module. Initially, we rely on two sensors for the minimum viable product (MVP), but we plan to integrate more sensors into the prototype as we progress. We implement a simple passcode authentication system to lock and unlock the bike using the old phone itself. From a software perspective, we work on the mechanics of the basic authentication system and report any breach to the cloud middleware if the bike is stolen.

Figure 1: Architecture Diagram (MVP)

1.1.2 Cloud Middleware

The cloud middleware is responsible for handling a ping from the bike lock device and immediately processing the input, which includes sensor data such as GPS, in case of theft. The processed data is sent to the user through messaging services like SMS or email. As we move beyond the MVP, we handle locking through the user's current smartphone. Therefore, the server middleware is required to handle communication between the two devices.

1.1.3 Client-side App

The client-side application provides locking functionality and alerts as a stretch goal. However, this part is not available in the MVP version. Once available, we aim to offer more features in the app.

1.1.4 Conclusion

This project aims to develop a bike locking system that utilizes an old Android device, cloud middleware, and a clientside application. The paper outlines the different phases of development and explains the processes involved in each phase. The resulting system will provide a secure, reliable, and cost-effective bike locking solution.

1.2 Minimum Viable Product

The goal of our project is to build an anti-theft application for bikes which can alert the user if theft is attempted on the bike. The minimum viable product should be able to use the phone's sensors for detecting thefts and send an email/SMS notification to the user.

1.2.1 MVP Goals

1. Web Application for the old phone attached to the bike
    - Can operate in two modes: Locked and Unlocked
    - Has an inbuilt passcode setting mechanism
    - Can switch between the two modes when provided with the passcode
    - Able to detect motion using the phone's accelerometer
    - Able to detect location of the bike using the phone's GPS sensor
2. Cloud Middleware
    - Receives sensor data from the web application
    - Processes the data and detects if the values cross a certain threshold
    - Able to send email/SMS notification to the user

2 Completed Deliverables

2.1 Fetch and format initial sensor data

2.1.1 Fetch data from accelerometer

- **Task:** Fetch the data from accelerometer sensor in the phone we are using the browser web API.
- **Success Criteria:** We have real time access to the raw accelerometer sensor data.
- **Status**: Successful, demonstrated in Figure 3
- **Developer(s)**: Grishma

2.1.2 Format data from accelerometer

- **Task**: Process and transform the raw accelerometer data into a more usable format. This involves converting the x, y, z co-ordinates into a format that can be interpreted and analyzed by the system.
- **Success Criteria**: The success parameters pertaining to this particular task encompass the imperative requirement of guaranteeing precise and faithful translation of the bike's motion into the converted data, and ensuring that the said data is adeptly equipped to detect and apprise concerned personnel of any unwarranted motion or operation.
- **Status**: Successful, demonstrated in Figure 3
- **Developer(s)**: Grishma

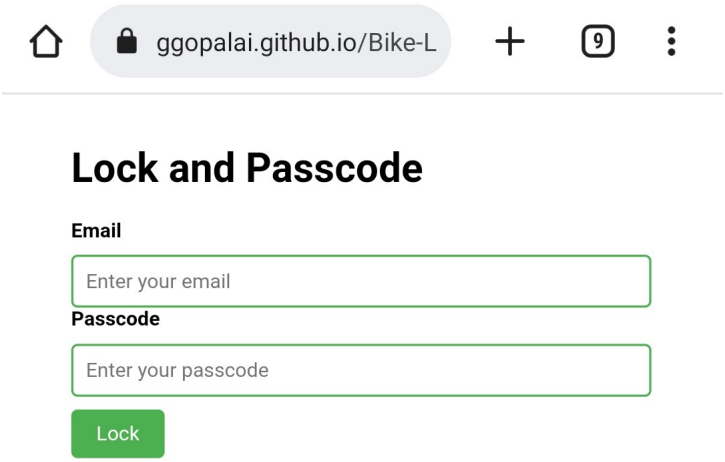

Figure 2: Landing Page

2.1.3 Fetch data from GPS sensor

- **Task:** Fetch the data from GPS sensor in the phone we are using the browser web API.
- **Success Criteria:** We have real time access to the raw GPS sensor data.
- **Status**: Successful, demonstrated in Figure 3
- **Developer(s)**: Grishma

2.1.4 Format data from GPS sensor

- **Task**: Process and transform the raw GPS sensor data into a more usable format. This involves converting the latitude and longitude into a format that can be interpreted and analyzed by the system.
- **Success Criteria**: We have the real time access to the location of the bike.
- **Status**: Successful, demonstrated in Figure 3
- **Developer(s)**: Grishma

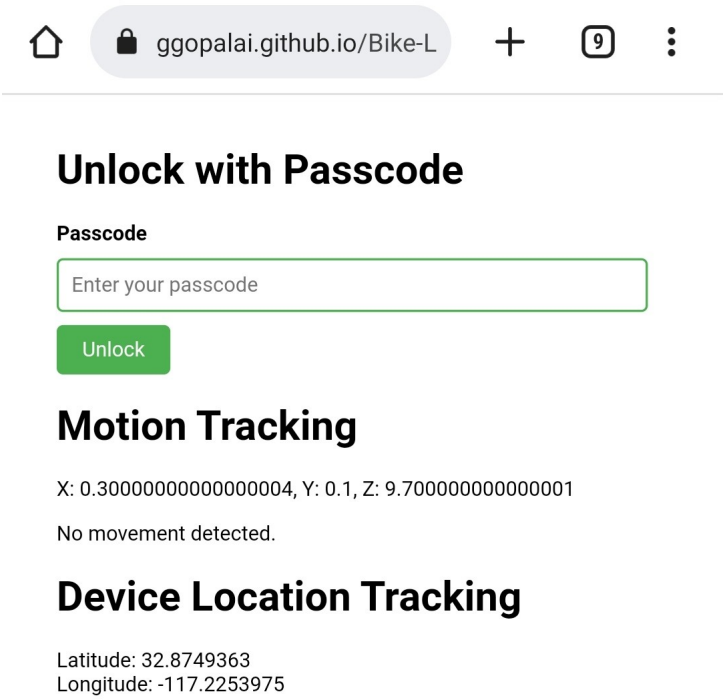

Figure 3: Locked Page with Sensor Data

2.2 Build the v1 interface for the bike app

2.2.1 Passcode feature

- **Task:** As for the MVP we have lock/unlock feature on the bike's application instead of user application, we need to have passcode based authentiation.
- **Success Criteria:** We can store a user given passcode in the web application
- **Status**: Successful, demonstrated in Figure 2
- **Developer(s)**: Akshay

2.2.2 Lock the application

- **Task:** Add a feature for setting the mode to 'Locked'
- **Success Criteria:** We are able to set the mode to 'Locked' by giving any passcode.
- **Status**: Successful, demonstrated in Figure 2
- **Developer(s)**: Akshay

2.2.3 Unlock the application

- **Task:** Add a feature for setting the mode to 'Unlocked'
- **Success Criteria:** We are able to set the mode to 'Unlocked' by using the passcode that was given while locking it.
- **Status**: Successful, demonstrated in Figure 3
- **Developer(s)**: Akshay

2.2.4 API call to cloud middleware

- **Task:** Send the sensor data to the cloud middleware in the email request when 'Locked'
- **Success Criteria:** We are able to make an API call to the cloud middleware and send the values obtained from the phone's sensors.
- **Status**: Successful, demonstrated in Figure 4
- **Developer(s)**: Gagan

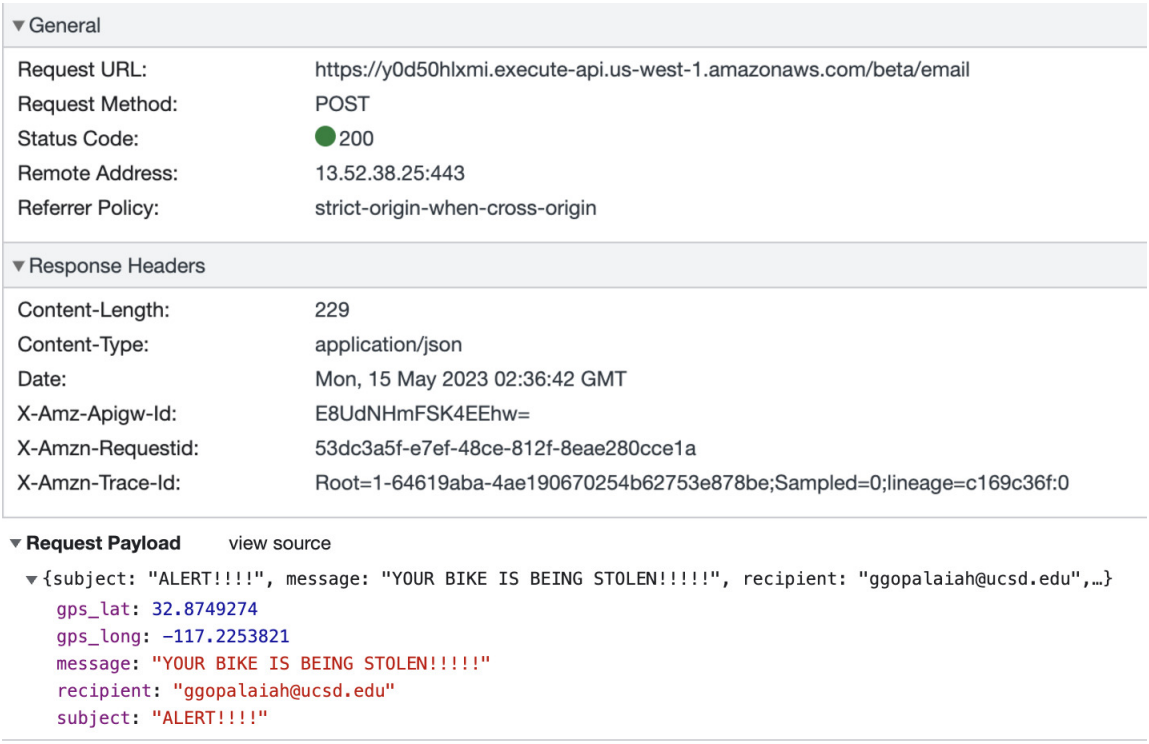

Figure 4: POST Request to send email

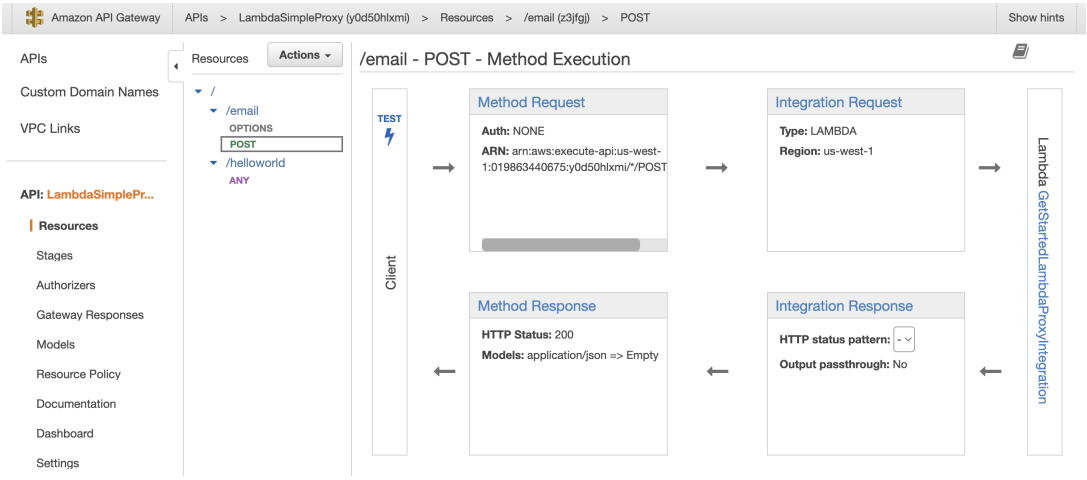

Figure 5: API Gateway Setup

2.3 Setup Cloud Middleware for sending notifications

2.3.1 Provision AWS API Gateway

- **Task:** Manually configure an instance of the AWS API Gateway service.
- **Success Criteria:** This will provide us with an endpoint that will serve as the entry point for the middleware, allowing us to connect to and communicate with other AWS services.
- **Status**: Successful, demonstrated in Figure 5
- **Developer(s)**: Gagan

2.3.2 Provision and setup AWS Lambda

- **Task:** Manually configure an instance of the AWS Lambda function.
- **Success Criteria** : This function will serve as the middleware for our cloud architecture and process incoming data from our bike-lock devices.
- **Status**: Successful, demonstrated in Figure 6
- **Developer(s)**: Gagan

2.3.3 Write business logic for the Lambda function

- **Task**: Define a function that ingests sensor data and other metadata, processes it, and fires emails to the user.

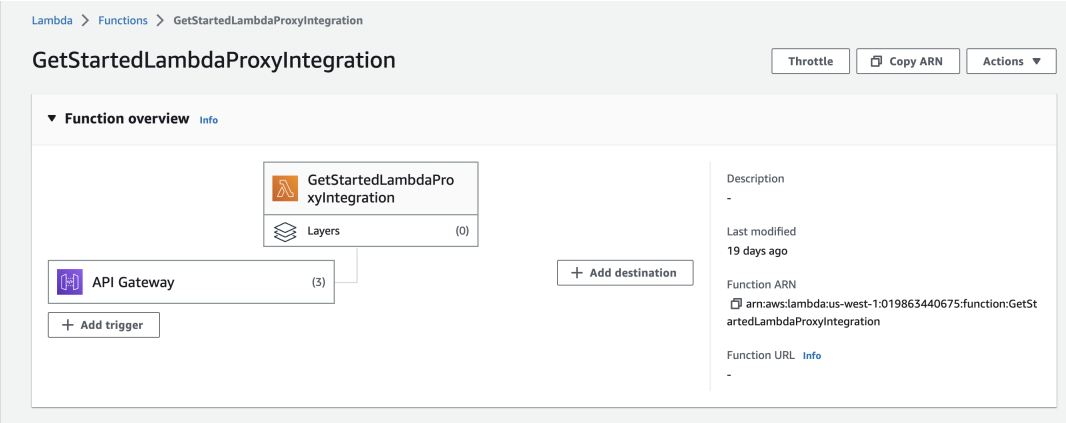

Figure 6: Lambda Function Setup

- **Success Criteria**: A working middleware that can send email to the user.
- **Status**: Successful, code can be seen here.
- **Developer(s)**: Gagan

2.3.4 Unit test lambda function

- **Task** : Unit test Lambda function by manually firing off a Test event that simulates expected sensor data.
- **Success Criteria**: A working Lambda function that can send emails to the user.
- **Status**: Successful, demonstrated in Figure 7
- **Developer(s)**: Gagan

2.3.5 End-to-end testing for API Gateway to Lambda

- **Task**: Test the Lambda function, but instead of a Test event, trigger it via a call to the API Gateway.
- **Success Criteria**: A working Lambda function that can successfully fire an email to the user.
- **Status**: Successful, demonstrated in Figure 8
- **Developer(s)**: Gagan

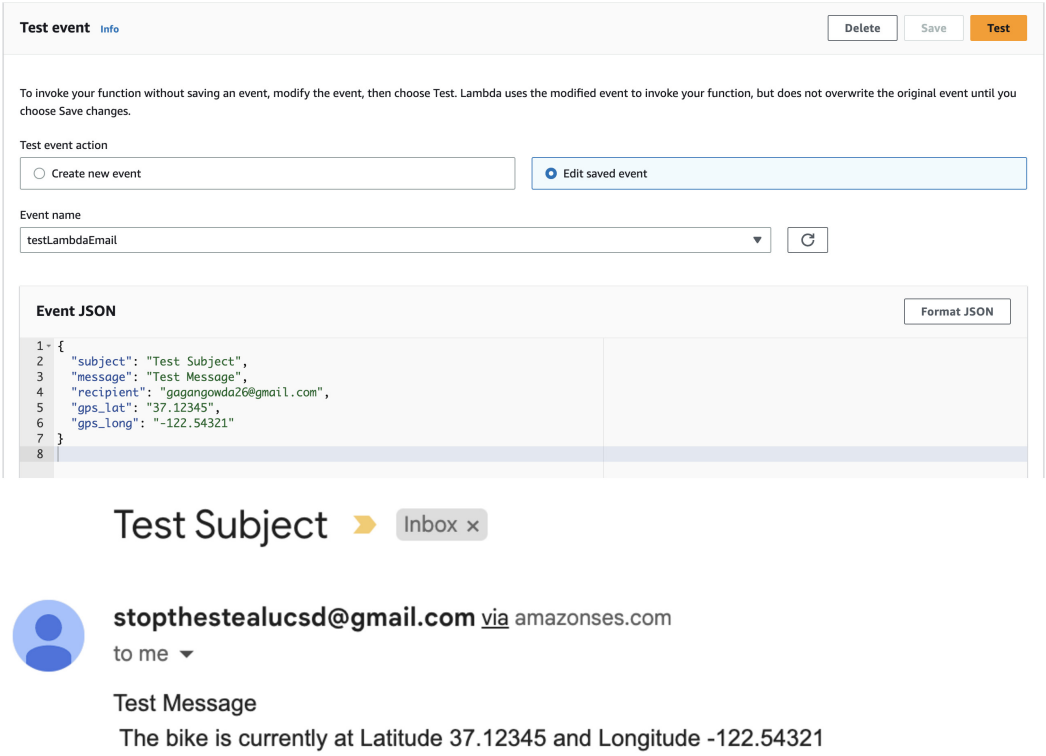

Figure 7: Unit testing lambda using event

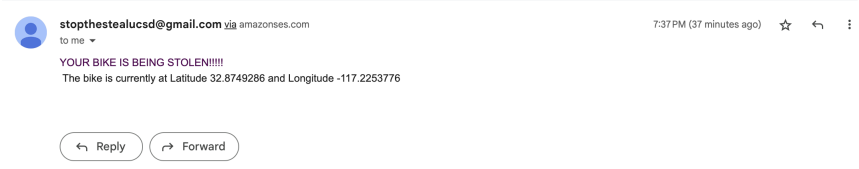

Figure 8: Email received by user

Milestone	Priority
3.1.1	P0
3.1.2	P1

Table 1: Build v2 interface for bike

Milestone	Priority
3.2.1	P1
3.2.2	P0
3.2.3	P0

Table 2: Build client side app interface

3 In-progress/Pending Deliverables

3.1 Build v2 interface for bike

3.1.1 Building a mechanism to register the bike

- **Task:** Since there can be multiple users of our app, we need a mechanism to pair the client app to the actual bike app.
- **Success Criteria:** We successfully implement a way for the two apps to sync and connect with each other.
- **Status**: Pending
- **Developer(s)**: Grishma

3.1.2 New locked screen status screen

- **Task:** Build the UI for a bike app once it is locked.
- **Success Criteria:** Successfully built a UI.
- **Status**: Pending
- **Developer(s)**: Grishma

3.2 Build client side app interface

3.2.1 Register a bike screen

- **Task:** Develop a screen to register a bike that will be monitored using the bike lock application.
- **Success Criteria:** We should be able to register a bike that needs to be monitored via the application.
- **Status**: Pending
- **Developer(s)**: Gagan

Milestone	Priority
3.3.1	P1
3.3.2	P0
3.3.3	P0
3.3.4	P0
3.3.5	P2
3.3.6	P0

Table 3: Update middleware for communication

3.2.2 Lock functionality on the bike

- **Task**: Lock the bike application from the client application screen.
- **Success Criteria**: We should able to remotely lock the bike application from the client application screen.
- **Status**: Pending
- **Developer(s)**: Gagan

3.2.3 Unlock functionality on the bike

- **Task**: Unlock the bike application from the client application screen.
- **Success Criteria**: We should be able to remotely unlock the bike application from the client application screen.
- **Status**: Pending
- **Developer(s)**: Gagan

3.3 Update middleware for communication

3.3.1 Handle bike registration

- **Task:** We will update the cloud middleware to store the details of the bike application.
- **Success Criteria:** We are able to register a bike application on the cloud.
- **Status**: In Progress, Implemented Publisher-Subscriber pattern design pattern to handle communication between the server and multiple subscriber bike applications.
- **Developer(s)**: Akshay

3.3.2 Handle client-bike pairing

- **Task:** We need to store the mappings for each bike application to its user application in order to establish two-way communication.
- **Success Criteria:** We have all the user to bike pairings stored on the cloud.
- **Status**: In Progress, NodeJS based server to handle pairing
- **Developer(s)**: Akshay

3.3.3 Handle lock functionality

- **Task:** We need to find a way to set the mode of the bike application to 'Locked' when we receive a command from its corresponding user application.
- **Success Criteria:** We are able to set the bike application to 'Locked' mode using client application.
- **Status**: In Progress, NodeJS based server to handle locking
- **Developer(s)**: Akshay

3.3.4 Handle unlock functionality

- **Task:** We need to find a way to set the mode of the bike application to 'Unlocked' when we receive a command from its corresponding user application.
- **Success Criteria:** We are able to set the bike application to 'Unlocked' mode using client application.
- **Status**: In Progress, NodeJS based server to handle locking
- **Developer(s)**: Akshay

3.3.5 Automate Email Identity Creation

- **Task**: Deploy AWS Lambda to programatically register an email on AWS SES.
- **Success Criteria**: Should be able to register user's email on AWS SES from the user application.
- **Status**: Pending
- **Developer(s)**: Gagan

Milestone	Priority
3.4.1	P1
3.4.2	P1
3.4.3	P1

Table 4: PWA conversion

3.3.6 Deploy Node.js server on EC2

- **Task**: Determine architecture for EC2 instance, setup Node dependencies and deploy the middleware server.
- **Success Criteria**: Must be able to hit API endpoints exposed by the server.
- **Status**: In Progress
- **Developer(s)**: Gagan

3.4 PWA conversion

3.4.1 Addition of service workers

- **Task:** Add service workers to cache the entire service locally
- **Success Criteria:** Satify the caching criteria required to make PWAs installable
- **Status**: Successful, demonstrated in Figure 9
- **Developer(s)**: Akshay

3.4.2 Setting up manifest and developing other PWA specific code

- **Task:** Setting up manifest and developing other PWA specific code which is needed to make the app installable
- **Success Criteria:** App can be installed locally on the phone
- **Status**: In Progress, will be completed soon. Compatability with multiple devices being tested.
- **Developer(s)**: Akshay

3.4.3 PWA Optimization

- **Task:** Optimization of the PWA
- **Success Criteria:** Complete Google lighthouse criteria for PWAs
- **Status**: Pending, will be started once previous action items are complete

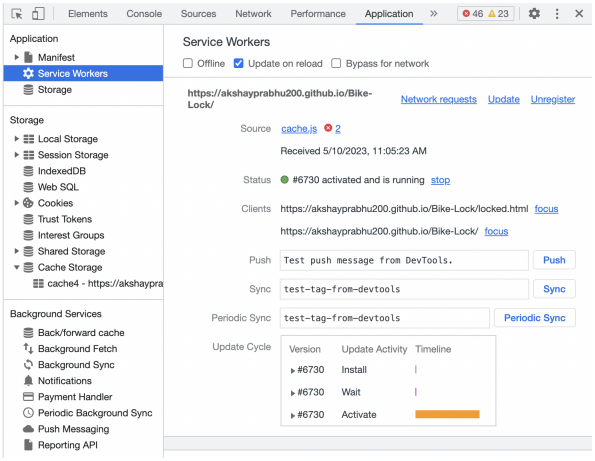

Figure 9: Service Worker Registration

Milestone	Priority
3.5.1	P1
3.5.2	P2

Table 5: Future sensor integration

3.5 Future sensor integration

We contend these are tasks which are a long reach, we do not guarantee the completion of these tasks by the end of this course and must be treated as a wishlist rather than promises. **Note -** We made the decision to de-scope the microphone integration due to a lack of time.

3.5.1 Speaker integration

- **Task**: Integrate the phone's speaker to sound an alarm.
- **Success Criteria**: Successfully sound a loud buzzer/alarm when the bike is being stolen.
- **Status**: Completed
- **Developer(s)**: Gagan
- **Proof of Completion** - Link to video

3.5.2 Camera API integration

- **Task**: Integrate the phone's camera to capture a picture of the thief/assailant.

- **Success Criteria**: We should be able to successfully capture the assailant's photo.
- **Status**: Pending
- **Developer(s)**: Akshay, Grishma

3.6 Documentation

3.6.1 Final Video

- **Task**: Create final video
- **Success Criteria**: Successful creation of final video
- **Status**: Pending
- **Developer(s)**: Akshay, Gagan, Grishma

3.6.2 Final Report

- **Task**: Create final report
- **Success Criteria**: Successful creation of final report
- **Status**: Pending
- **Developer(s)**: Akshay, Gagan, Grishma

3.6.3 Web Presence

- **Task**: Create web presence
- **Success Criteria**: Successful creation of web presence
- **Status**: Pending
- **Developer(s)**: Akshay, Gagan, Grishma

4 Feedback and Responses

Why use and old android device? What are the benefits?
Cost-effectiveness: One of the primary advantages of using an old Android device is its affordability. These devices are often available at significantly lower prices compared to the latest models.

Abundance and Accessibility: Old Android devices are abundant in the market. Many people upgrade to newer models frequently, leading to a surplus of older devices available for purchase. This high supply ensures that these devices are

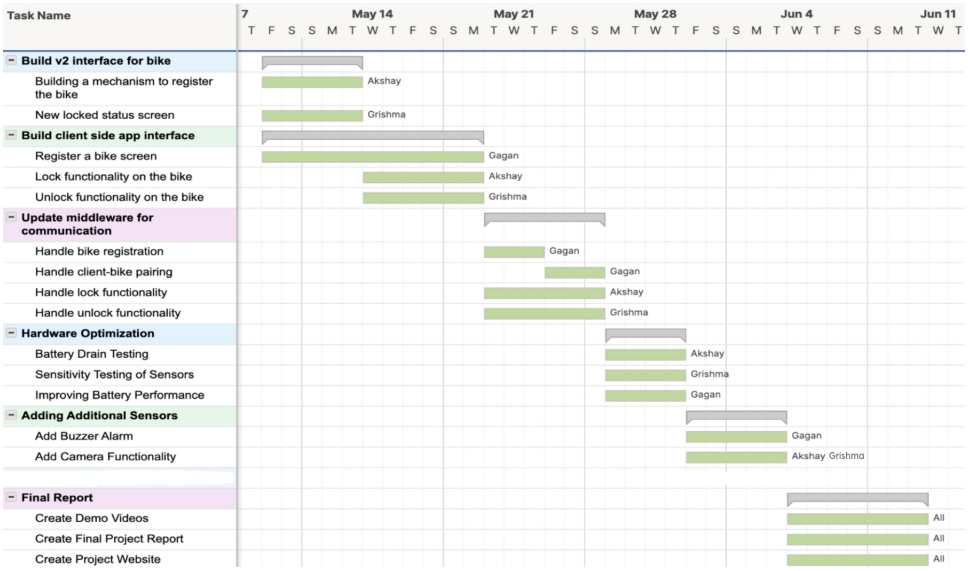

Figure 10: Gantt Chart

easily accessible to a wide range of individuals, making it convenient for those who may not have the means or desire to invest in the latest technology.

Compatibility: Old Android devices often have all the necessary built-in sensors and functionalities required for general day-to-day use. They can still perform common tasks such as calling, texting, web browsing, and running a variety of applications. While they may not support the latest software updates or resource-intensive apps and games, they can still meet the basic needs of many users.

Environmental Considerations: By using an old Android device, individuals contribute to reducing electronic waste and promote sustainability. Instead of discarding these devices prematurely, reusing them extends their lifespan and reduces the demand for new manufacturing. This approach aligns with the principles of recycling and responsible consumption, making it an environmentally conscious choice.

Cost of these devices?
The core assumption is that these are old devices that the users might already own, but if they are to be purchased, then our analysis of Reddit's hardware swap, eBay, and other used item marketplaces places the costs to be under $100. The affordability of old Android devices is a key factor in their appeal. When considering the purchase of a used device, various platforms can offer a wide range of options at budget-friendly prices. For example, Reddit's hardware swap

community allows users to buy and sell used hardware, including Android devices. By browsing through the listings on this platform, users can often find older Android devices priced below $100, depending on the specific model, condition, and demand.

Additionally, eBay is a popular online marketplace that offers a vast selection of used Android devices. Sellers on eBay typically provide detailed information about the condition of the devices, including any cosmetic or functional issues. By carefully evaluating the listings, users can often find affordable options within their desired price range. It's worth noting that the prices on eBay may vary based on factors such as device condition, storage capacity, brand, and model popularity.

Other used item marketplaces, both online and offline, also contribute to the accessibility of affordable old Android devices. Local classifieds websites, thrift stores, and even smartphone repair shops occasionally offer second-hand devices at competitive prices. These platforms provide additional avenues for individuals to find inexpensive options, particularly if they prefer to inspect the device in person before making a purchase., Example of a smartphone that could be purchase and the archive link for the product page

What sort of technology/sensors to you need?
We will need a browser capable of accessing the internet, a SIM card connected to the internet, the internet plan does not need to have a high data limit as our app uses very little. We would also need the phone to have a functional microphone, speaker, screen, accelerometer, GPS sensor, gyroscope and camera.

Argue that the old smartphones are sufficient and readily available.
Our app is expected to run in the browser and most Android phones released in the past 10-15 years should be able to run the application.

Nice articulation of the potential negatives/criticisms of the project.
Thank You!

Getting a smart phone to operate as a sensor is not always trivial. E.g., the smartphone OS will do things like put tasks to sleep to save energy which may not be ideal for sensor.
For V1 and V2, we are relying on application being onscreen with the screen timeout disabled, this is a tradeoff, but a conversion to native would solve this issue completely. However, as mentioned in the milestone oral update this will not be implemented as a part of our course.

I'm interested to see how this turns out.
Worked out well, evidence shared in Figure 11

Clear articulation of the tasks.

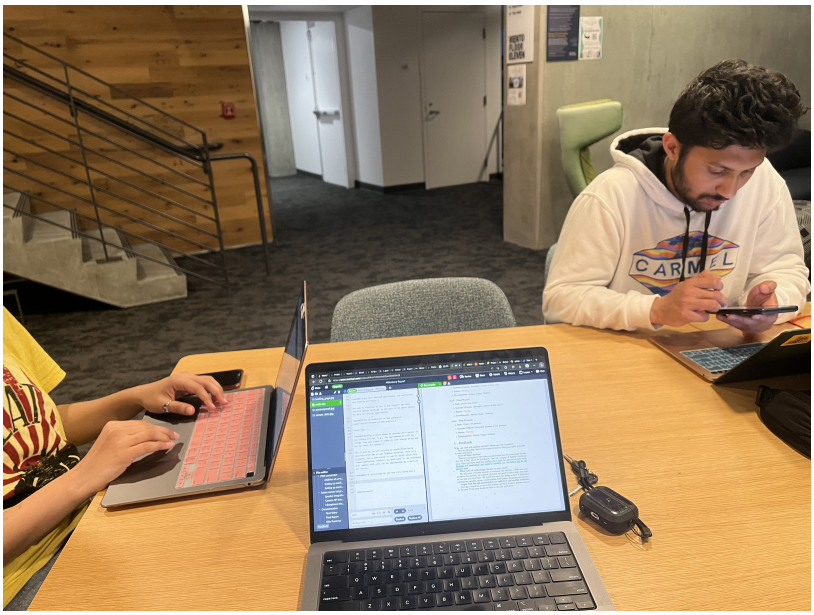

Figure 11: Collaborative Working Session

Thank You!

It is worth explicitly mentioning what the plan is with respect to your driver-navigator and SWOT analysis setup what will be done if a single team member falls behind schedule or misses a milestone.
SWOT analysis, as demonstrated in Figure 12, is simply a method of analysing the benefits and weaknesses of an approach. If a team member is missing then the remaining duo can do the SWOT analysis and share the details with the third member and the voting can be done later on asyncronously.
As for the driver navigator system, as long as atleast two individuals are present the process can be followed. If none of the individuals are present then we would have no choice but to fall back to individual work which happens async. Thankfully, none of this happened this quarter as all members of the team have the same set of courses, as a result we end up meeting in our classes, are meeting up for other coursework, etc.

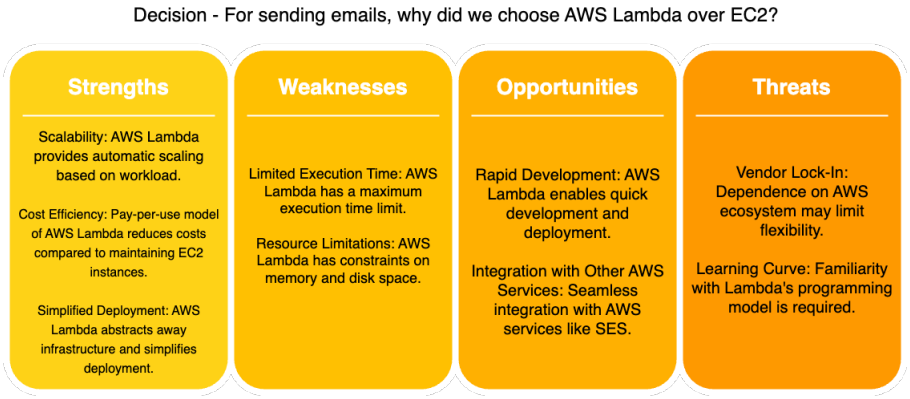

Figure 12: SWOT Analysis