CSE 237D: Stop the Steal - Bike Anti-Theft System Specification

Akshay Prabhu, Gagan Gopalaiah, Grishma Gurbani

April 21, 2023

1 Project Charter

1.1 Project 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.

1.2 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.

1.2.1 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.

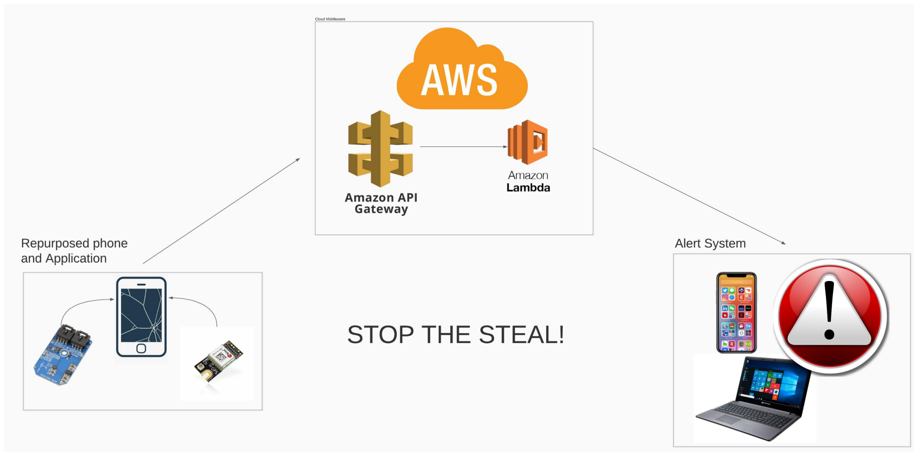

Figure 1: Architecture Diagram (MVP)

1.2.2 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.

1.2.3 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.

1.2.4 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.

1.3 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.

1.3.1 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

1.3.2 Stretch Goals

We'll add more functionalities to the project once we complete all the goals defined for the MVP. The features are as listed below:

1. Web Application for the old phone attached to the bike
    - Able to ring an alarm on the phone when theft is attempted
    - Able to click a picture when theft is attempted using the phone's camera
2. Cloud Middleware
    - Can receive lock/unlock commands from the user application
    - Able to send data to the bike's application
3. User Application
    - Can change the mode of the bike's application from Locked to Unlocked and vice versa
    - Can view the geolocation of the bike on a map

1.4 Constraints, Risk, and Feasibility

1.4.1 Constraints and Feasibility

1. **Environment proofing:** Another risk is achieving complete environmental proofing for the device. Given the time frame, it may not be possible to achieve complete proofing against environmental factors such as rain or physical damage. Therefore, our assumption will be that the device will be used in ideal environments where these factors are not present. We will, however, take measures to ensure that the device is robust and resilient against minor environmental factors.

2. **Time constraints:** Time constraints are another risk that may prevent us from implementing all the desired features in the MVP. To mitigate this risk, we will prioritize the features based on their importance and feasibility, ensuring that we achieve all the essential features within the given time frame.

3. **Device Theft:** It is our underlying assumption that the device in question is of sufficient age to dissuade potential theft or has been discretely concealed from would-be thieves. Furthermore, we contend that our solution is intended for short-term applications, such as students attending a lecture or individuals engaging in a brief shopping excursion, and is not intended to combat the determined efforts of a seasoned thief targeting assets such as bikes stored overnight, as they are likely to have taken adequate antitheft precautions into consideration.

4. **App Being Closed:** We are also operating under the assumption that the device remains impervious to environmental factors, and is ideally situated within a secure enclosure that precludes the possibility of theft, vandalism or unauthorized interference with the device's back button, which could result in the closure of our application. Nonetheless, it should be noted that the design and development of said cage or protective layer falls outside the purview of our current project scope, as it is slated for future consideration.

5. **Limited Battery Life:** The majority of antiquated Android devices are subject to diminished battery life as a result of their lithium-ion cells experiencing degradation due to multiple charge cycles. It is therefore advisable to utilize an external battery bank to mitigate excessive phone usage, as the device's operational capabilities may be curtailed to only a few hours, at best.

6. **Network:** The network functionality of mobile devices constitutes an area of considerable risk. Potential vulnerabilities include deficient LTE or 5G signal reception on the client side. Nevertheless, end-users can optimize their network connectivity by leveraging WiFi while parking their vehicle at home, attending class, or at the workplace, thereby enhancing network performance and reducing battery drain. Moreover, network uptime and the reliability of the cloud services we employ to deliver SMS and email transmissions are additional concerns that warrant close scrutiny.

7. **False Positives:** Striking a balance between optimal accuracy and a low rate of false positives presents a delicate equilibrium. Consequently, we endeavor to ensure that a significant margin of error exists between these two parameters. It is acknowledged, however, that a certain degree of false positives is inevitable.

8. **Thief Unlocking the bike:** In order to achieve the minimum viable product (MVP), the initial implementation will include the locking and

unlocking functionality on the older Android device, which will be kept on the bicycle. It is acknowledged that this approach presents a potential security risk, as a would-be thief could gain access to the device and unlock the bike. However, for subsequent iterations, either in the final version or a future release, the intention is to modify the solution to incorporate both client and bicycle-side applications. To mitigate the risk of easy unlocks for thieves, a straightforward passcode mechanism will be implemented, requiring the user to enter a code to unlock the bike, which the thief would not possess.

1.4.2 Risks

1. **Potential Unforeseen Limitations of the Web as a platform:** The present project employs state-of-the-art application programming interfaces (APIs) that are currently endowed with experimental support in the realm of web-based platforms. The decision to adopt the web-based approach was a deliberate one, given that we repurposed outdated Android devices for our purposes, rendering operating system (OS) level updates impracticable. However, we can avail ourselves of browser updates to overcome this predicament. Nonetheless, we acknowledge that deploying a web-based application entails certain limitations vis-à-vis native apps, which may preclude certain tasks.

2. **Hardware failures:** The utilization of antiquated Android devices poses potential concerns regarding the operability and accessibility of the sensors. The occurrence of inaccurate readings could significantly impede the functionality of our project.

3. **Scope Creep**: Scope creep is a common risk associated with software development projects. Unless we maintain a tight ship, we run the risk of expanding our scope and wasting time. To mitigate this risk, we plan to establish a clear project scope and prioritize tasks based on their importance and impact on the project.

2 Group Management

2.1 Communication

Our team has planned to work and communicate using a combination of different methods. We will be using WhatsApp as our primary source of asynchronous communication, which will allow us to stay in touch and share updates on project progress. In addition to that, we have decided to schedule in-person pair and mob programming sessions at the Nuevo West Viento and Nuevo East Piedra Study Rooms in Graduate housing, which will give us the opportunity to work together and collaborate on specific tasks. For remote work, we will be conducting our meetings over Zoom to discuss any issues, plan future tasks, and ensure everyone is on the same page. By using a variety of methods, we aim to maximize efficiency and productivity while maintaining effective communication within the team.

2.2 Decision Making

Our team has made a conscious decision to follow a structured approach for decision-making. For technical decisions, we will rely on SWOT analysis, which will involve assessing the strengths, weaknesses, opportunities, and threats associated with each approach. This will help us make informed decisions based on the analysis of each approach's technical feasibility, scalability, and potential impact on the project. In addition to this, we have decided to adopt a flat structure centered around group consensus for decision-making. We will not have a leader; instead, all members of the team will have an equal say in decision-making, and the final decision will be made based on the agreement of the majority. This approach ensures that everyone's opinions are heard and considered, leading to more inclusive and effective decision-making.

2.3 Responsibilities

Our team intends to employ a collaborative programming approach, specifically pair and/or mob programming, wherein the team members take turns assuming the roles of a driver and a navigator in a round-robin fashion. This approach will enable us to effectively tackle the project's challenges by allowing each team member to contribute their expertise and knowledge to the project. We have eschewed the traditional model of designating individual team members with specific milestones, instead opting to establish clear and well-defined milestones for the project as a whole.

To facilitate the project's success, we have divided the project into distinct areas of interest. Grishma Gurbani and Akshay Prabhu will spearhead the exploration and research of browser-based web APIs that can interface with sensors. On the other hand, Gagan Gopalaiah will leverage his extensive experience with cloud providers to determine the most effective means of constructing the middleware.

3 Project Development

3.1 Development roles

Developers will not have a specific role in the project, however we plan to take up responsibilities based on areas of expertise and interest. Roughly, one developer will work on the application that will run on the repurposed phone, while another will be responsible for developing the middleware that connects the phone to the cloud. The third developer will focus on developing the notification system that will alert the user in the event of a theft attempt. Each developer will also be responsible for testing their respective components to ensure that they are working properly.

3.2 Hardware and Software Used

We will be using repurposed phones as they already come with the required sensors like accelerometer and GPS. To develop the software, we will be using our local Macbook M1 machines as the development environment. We will be leveraging the powerful processing capabilities of the M1 chip to ensure that our software runs smoothly and without any hiccups.

For the middleware, we will be hosting our code on the Amazon Web Services (AWS) cloud platform. We plan to use AWS API Gateway and Lambda to host the API that will act as the communication channel between the bike-lock device and the user's phone. These AWS services offer a robust and scalable solution that will allow us to handle a large volume of requests and ensure that our system is always available and responsive.

3.3 Resources Required

- An old phone either from a UCSD lab or from eBay.
- AWS credits
- Potentially credits from an email/SMS service such as Twilio or Mailchimp

3.4 Testing

To ensure the quality of the software, we will be implementing rigorous testing procedures throughout the development process. For the web app, we will be using the Jest testing framework, which will allow us to write and run automated tests on the application. We will develop test cases that cover a range of scenarios, such as user inputs, error handling, and edge cases. By running these tests regularly, we can quickly identify any issues and ensure that the application is functioning as expected.

For the middleware hosted on AWS, we will be testing our code using AWS Lambda's built-in testing tools, which allow us to simulate input events and verify the output. We will write test cases for the API endpoints, ensuring that the middleware is correctly processing and storing sensor data, and triggering notifications as expected.

Throughout the development process, we will also conduct manual testing to ensure that the software is user-friendly and meets the requirements outlined in the MVP specification. This will involve using the system in a real-world scenario, and testing each component individually as well as in conjunction with each other.

Milestone	Priority
4.1.1	P0
4.1.2	P1
4.1.3	P2

Table 1: Fetch and format initial sensor data

3.5 Documentation

We will document the project using Markdown and host it on GitHub Pages.

4 Project Deliverables, Milestones and Schedule

The deliverables have been defined as subsections and milestones as sub-subsections. The assignments should be considered advisory in nature as we expect to work in tandem even though someone is taking the lead in the research and development the others will be closely working with them.

4.1 Fetch and format initial sensor data

4.1.1 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.

4.1.2 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.

4.1.3 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.

Milestone	Priority
4.2.1	P0
4.2.2	P1
4.2.3	P2
4.2.4	P2

Table 2: Build the v1 interface for the bike app

4.1.4 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.

4.2 Build the v1 interface for the bike app

4.2.1 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

4.2.2 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.

4.2.3 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.

4.2.4 API call to cloud middleware

- **Task:** Send the sensor data to the cloud middleware 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.

Milestone	Priority
4.3.1	P1
4.3.2	P1
4.3.3	P0
4.3.4	P2
4.3.5	P2

Table 3: Setup Cloud Middleware for sending notifications

4.3 Setup Cloud Middleware for sending notifications

4.3.1 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.

4.3.2 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.

4.3.3 Write business logic for the Lambda function

- **Task**: Define a function that ingests sensor data and other metadata, process it, and fire notifications to the user.
- **Success Criteria**: A working middleware that can communicate with the user on receiving sensor data.

4.3.4 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 successfully fires of a notification to the user.

4.3.5 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 fires of a notification to the user.

Milestone	Priority
4.4.1	P0
4.4.2	P1

Table 4: Build v2 interface for bike

Milestone	Priority
4.5.1	P1
4.5.2	P0
4.5.3	P0

Table 5: Build client side app interface

4.4 Build v2 interface for bike

4.4.1 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.

4.4.2 New locked screen status screen

- **Task:** Build the UI for a bike app once it is locked.
- **Success Criteria:** Successfully built a UI.

4.5 Build client side app interface

4.5.1 Register a bike screen

- **Task:** Develop a screen to register a bike that will be monitored using the bike lock application.
- **Success Criteria:** We are able to register a bike that needs to be monitored via the application.

4.5.2 Lock functionality on the bike

- **Task**: Lock the bike application from the client application screen.
- **Success Criteria**: We are able to remotely lock the bike application from the client application screen.

Milestone	Priority
4.6.1	P1
4.6.2	P0
4.6.3	P0
4.6.4	P0

Table 6: Update middleware for communication

4.5.3 Unlock functionality on the bike

- **Task**: Unlock the bike application from the client application screen.
- **Success Criteria**: We are able to remotely unlock the bike application from the client application screen.

4.6 Update middleware for communication

4.6.1 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.

4.6.2 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.

4.6.3 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.

4.6.4 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.

Milestone	Priority
4.7.1	P1
4.7.2	P1

Table 7: PWA conversion

Milestone	Priority
4.8.1	P2
4.8.2	P2
4.8.3	P2

Table 8: Future sensor integration

4.7 PWA conversion

4.7.1 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

4.7.2 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

4.8 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.

4.8.1 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.

4.8.2 Camera API integration

- **Task**: Integrate the phone's camera to capture a picture of the theif/assailant.
- **Success Criteria**: We are able to successfully capture the assailant's photo.

4.8.3 Microphone integration

- **Task**: Integrate the phone's microphone to record audio.
- **Success Criteria**: We are able to record the audio at the time of attempted robbery.

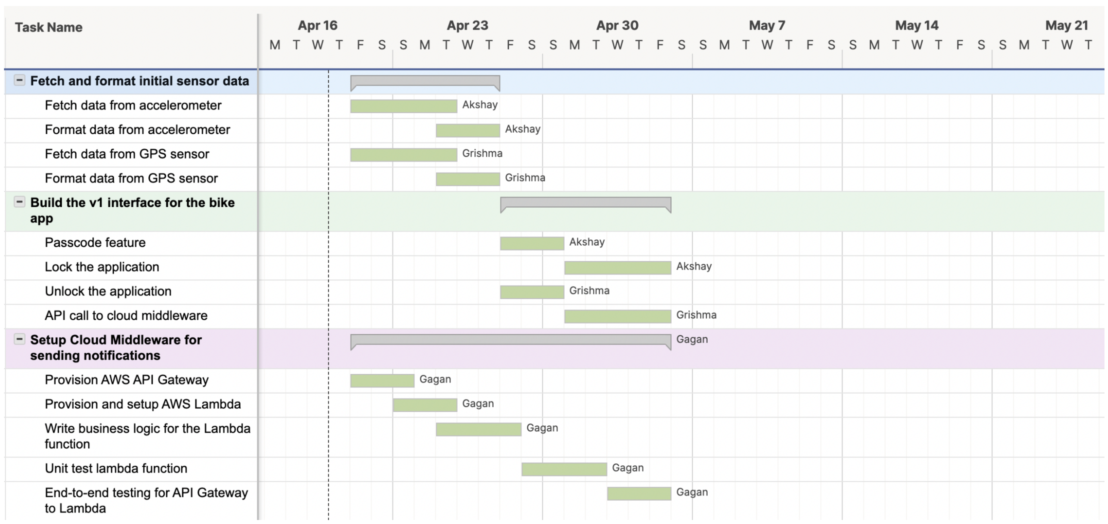

Figure 2: Gantt Chart for MVP

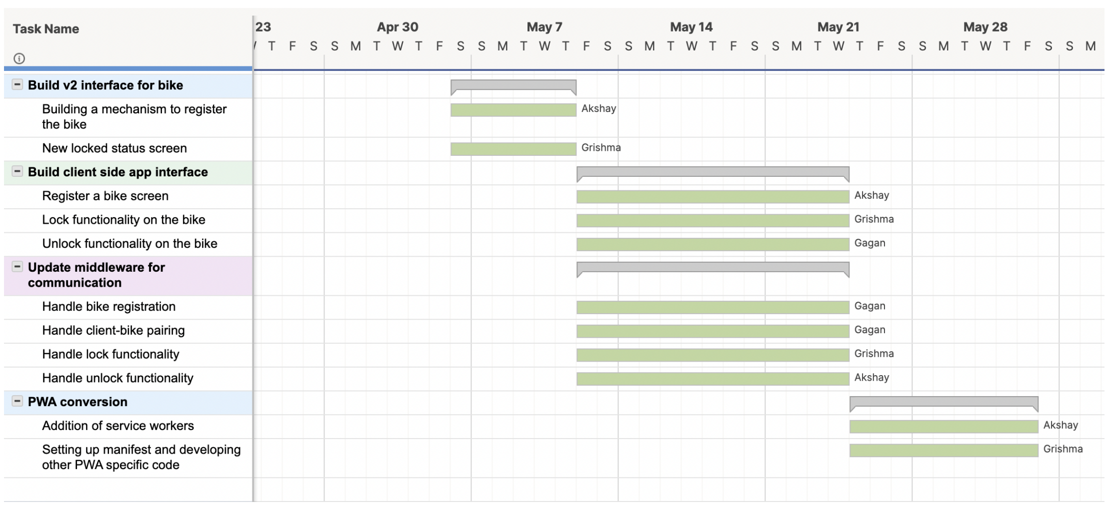

Figure 3: Gantt Chart for Additional Features