Boston University

Electrical & Computer Engineering

EC463 Senior Design Project

First Prototype Testing Plan

by

Team 8

BikeGuard

Beren Donmez: bydonmez@bu.edu

Margherita Piana: mpiana@bu.edu

Marissa Ruiz: mbruiz@bu.edu

Ben Taylor: betaylor@bu.edu

Albert Zhao: albertz@bu.edu

Required Materials

Hardware	Software
Onboard: ● Raspberry Pi Zero 2 W (with 32GB SanDisk SDHC Class 10 card) ● Piezo Buzzer ● MPU-6050 (Accelerometer) ● INIU BI-B61 Portable Charger 22.5W 10000mAh Power Bank ● Raspberry Pi Camera Module V2 ● TP-Link Router ● Small breadboard ● Electrical Tape ● Jumper Cable ● USB-C to micro USB cable ● Small heat sinks Remote: ● Laptop	Raspberry Pi: ● Raspberry Pi OS 32 Bit (Legacy), Debian Bullseye -> better for camera ● node.js -> child_process and fs ● Python3 -> accelerometer reading ● Flask -> pi camera stram Front End: ● Javascript and React ● CSS for formatting ● Flask for connecting to the backend Back End: ● Flask ● SQL for data storage Machine Learning Model: ● Logistic regression model ● Detect theft based on the accelerometer data that Assign labels "0" for normal and "1" for theft ● Extract_features() method ● Train_test_split() method ● StandardScaler() for better performance ● Train model with LogisticRegression. ● True Positives (TP): 49, False Positives (FP): 3, False Negatives (FN): 6, True Negatives (TN): 18 -> Accuracy = ≈0.8816

Set Up:

Our hardware setup includes a Raspberry Pi Zero 2 W, a corresponding Raspberry Pi Camera, an accelerometer, and a buzzer. We use a router to establish our network within the broader BU wifi. We connected the Raspberry Pi to the network and are hosting all our servers on the network. We use SSH to access the Raspberry Pi remotely. We are powering the Raspberry PI with a portable power bank that is intended to charge phones and tablets. We additionally attached small heat sinks to our Raspberry Pi. Our components are in a lock box attached to the bike. The accelerometer's data are saved in a CSV file utilizing a node.js code. In this manner, the data can be used to train the machine learning model to recognize possible bike theft. Once the pitch and roll reach a certain threshold while shaking, the buzzer goes off. The backend file app.py receives constant information from the raspberry pi accelerometer.py folder. If the accelerometer detects motion greater than a pre-set threshold, it sends a push request to the backend and stores the message in the SQL database. Once the front end detects new changes in the database, it displays the most recent message on our website. As for the camera live view, the Raspberry Pi already has a module to convert real video feed into mpng (different format of png), then using flask we can stream the video feed directly to the front end.

Pre-testing Setup Procedure:

Raspberry Pi:

1. Connect and turn on the router.
2. Connect to the Raspberry Pi from the computer through ssh with the following command: ssh Team8@Raspberry_pi_IP.
3. Run the Python script "python3 picam3.py" to start the camera and push it to the background -> ctrl + z, bg
4. Run the node JS script "node log_mpu_data.js" to start the Python code accelerometer.py which connects to the back end and saves the accelerometer data to a CSV file.

Server/Web UI:

5. Enable virtual environment for new systems
6. Make sure system is connected to router
7. Go to back end directory and start flask "python3 app.py"
8. Go to the front end folder and run "npm start"

Testing Procedure:

1. Mount the enclosure to your bike
2. The camera is being consistently streamed on the front-end website
3. Once the bike gets shaken the buzzer goes off and a notification will be seen on the website
4. Once the bike is not shaken anymore the buzzer should stop immediately.

5. All the data are collected and saved in a CSV file that can be used in the machine learning model.
6. The machine learning algorithm utilizes linear classification to train the model and correctly classify new data

Measurable Criteria:

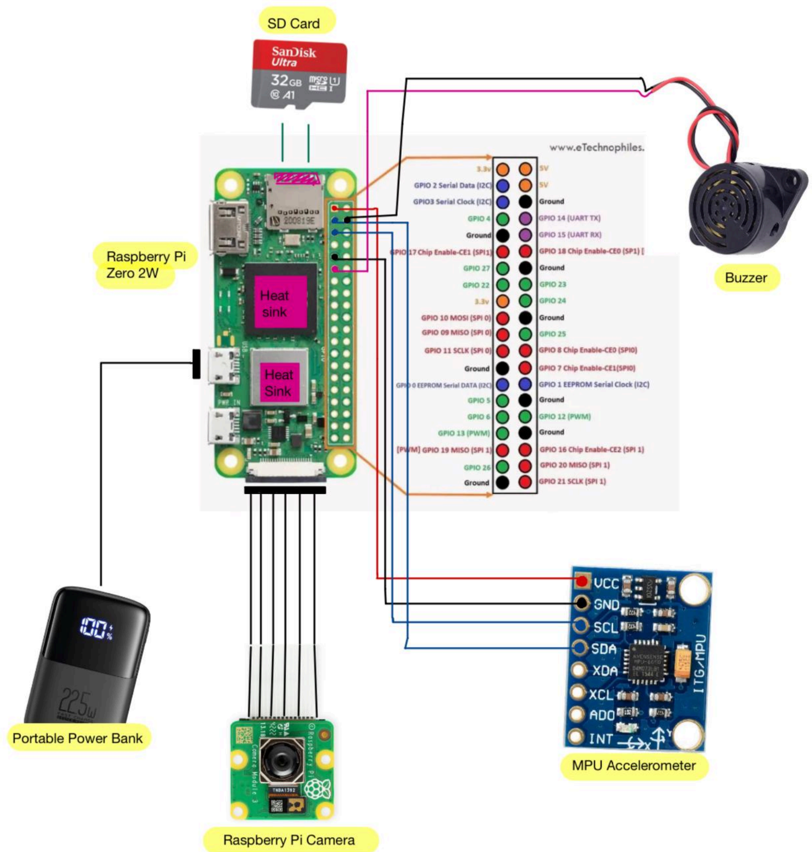
The criteria for a successful running are the following:

1. The enclosure is resistant and does not obstruct the normal function of the bike.
2. When the bike is shaken the buzzer goes off immediately.
3. When the bike is not moved anymore, the buzzer should stop buzzing immediately.
4. When the buzzer goes off a message is printed in our message.
5. Video stream is correctly visible on our website.
6. Once the machine learning model is trained a low error is expected as well as a good confusion matrix, meaning that possible theft is correctly classified.

Hardware Pins:

Raspberry Pi pins	Usage/Description
GPIO 2 Serial Data(I2C) Pin# : 3	SDA pin used for I2C communication between accelerometer (MPU) and raspberry pi
GPIO 3 Serial Data(I2C) Pin# : 5	SCL pin used for I2C communication between accelerometer (MPU) and raspberry pi
Power Pin (3V3) Pin# :1	Powers the accelerometer
Ground Pin Pin# :9	Ground for accelerometer
GPIO 17 Pin# :11	Powers the buzzer, connects it to Raspberry Pi for GPIO usage
Ground Pin Pin# :6	Ground for buzzer
Ribbon Connector Pin# :N/A	Connects Camera to Raspberry Pi

Diagram



Measurable Criteria:

1. Buzzer successfully goes off with excessive shaking motion
2. Sends alert to frontend website