Wi-Fi Detection and Analysis

ECE 3333

Project Report

Texas Tech University

Electrical and Computer Engineering

Khisa-Lee Lebrun

R11805898

Group Members: Boyd Paxton, Nicholas Woodward

May 2025

Abstract

This paper describes the design and implementation of a Raspberry Pi 4 [1], routers, and a Raspberry Pi Zero [2] on a web application. All the elements necessary for the project's results will be discussed, including the software and hardware aspects. Additionally, the requirements proper to this project will be decomposed and analyzed to make sure the demands were met. This document’s purpose will mainly be found in the revelation of the definitive results that came out of this project, accompanied by explanations of the approach taken.

Acknowledgement

The results shown in the following report would not have been possible without the help and guidance of classmates and faculty who provided mentorship and advice, as well as the friends who provided moral support. Special recognition is extended to all ECE professors and teaching assistants who always took the time to point the project in the right direction and lend out advice while issues were faced along the making of this project. It is crucial to note that without all this assistance, the project would not be where it is currently.

Table of Contents

Table of Figures

I. Introduction

As common as Wi-Fi can seem to Westerners, it isn't accessible everywhere in the world. The growing demand for connection makes the testing and debugging of its application necessary and a fundamental requirement and skill. This project focuses on the design and implementation of an advanced hardware system connected to a web application that displays relevant information regarding the Wi-Fi connection. However, this functionality must be accompanied by robust security measures to prevent risks such as unauthorized network access, data breaches, privacy violations, or performance issues that may arise due to vulnerabilities, excessive permissions, or malicious exploitation.

With this in mind, the goal of this project is to design a complete solution, integrating hardware elements such as Wi-Fi routers, various microcontrollers, as well as a sophisticated user interface. This project aims to provide a reliable, efficient, and safe system for Wi-Fi debugging and to be presented, as seen on Appendix A, as a final product on May 3rd 2025, as seen on Appendix B.

II. Body of Technical Report

1. Hardware

A. Raspberry Pi 4

As mentioned, a Raspberry Pi 4 was used for this project [1]. Its small size, high computing performance, integrated support for Wi-Fi and Bluetooth, and large developer community make it an ideal candidate. These features make it a robust device for 802.11 protocol analysis. However, the integrated Broadcom BCM43438[2] Wi-Fi/Bluetooth chip does not support the monitor or promiscuous modes required for packet or frame capture.

B. Wi-Fi Routers

To overcome this limitation, a BrosTrend long-range Wi-Fi adapter [3] was used. This adapter is equipped with the Realtek RTL88x2BU chip [4], which supports monitor mode. This mode, combined with its extended range, enables advanced analysis over large areas. With this addition, the network monitoring load is delegated to the external adapter, allowing the Raspberry Pi's internal chip to remain connected to the Internet. This opens up the possibility of collecting and serving data simultaneously, which is essential for communication with clients and integration with the next hardware to be used simultaneously: the Raspberry Pi Zero [5].

C. Raspberry Pi Zero

The raspberry pi zero is a low-power Wi-Fi microcontroller. It provides a lightweight, portable solution for collecting network metrics at various locations. For portable operation, it has its own circuitry as you can see in Figures 1 and 2 below.

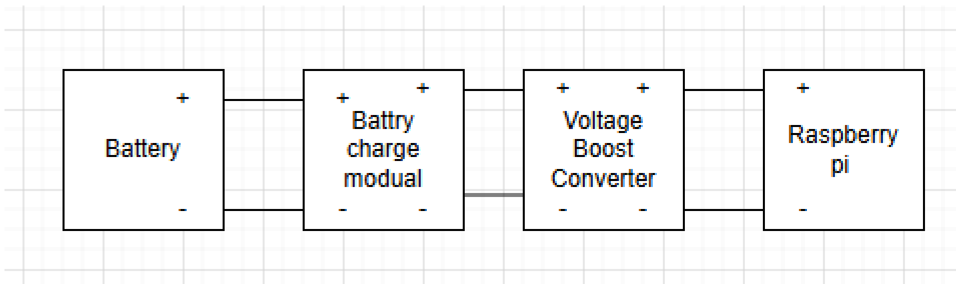


Figure 1: Raspberry Pi Zero Power Train Schematic.[6]

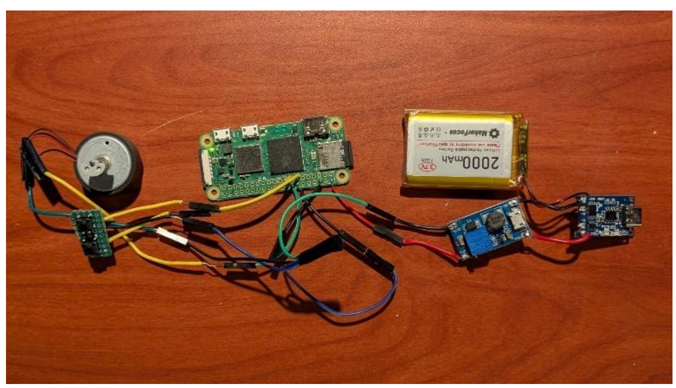


Figure 2: Raspberry Pi Zero Power Train Picture.[7]

As seen on the left side of the picture in Figure 2, a buzzer was implemented to alert the user that the data point was well collected. To ensure portability and to make it more user friendly, the Raspberry Pi Zero also has its own case as you can see on the Figure 3 below as well as in the Appendix C.

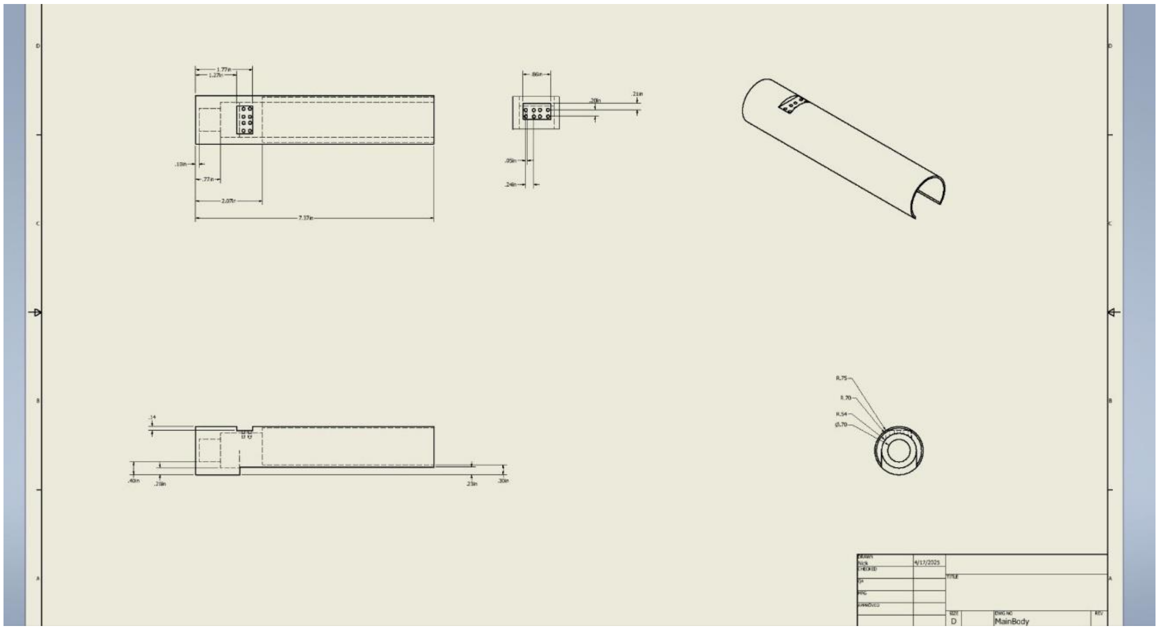


Figure 3: Raspberry Pi Zero Case. [8]

2. Web Application

A. Front End

After a pivot to ensure that the product would function properly on the day of delivery, the user interface went from a React Ionic [9] application to a web application composed of HTML for the format and CSS styling, as you can see in Appendix D. The interface comprises a home screen prompting the user to pick from several modes as well as different buttons for accessing various information, as you can see in the figure 4 below.

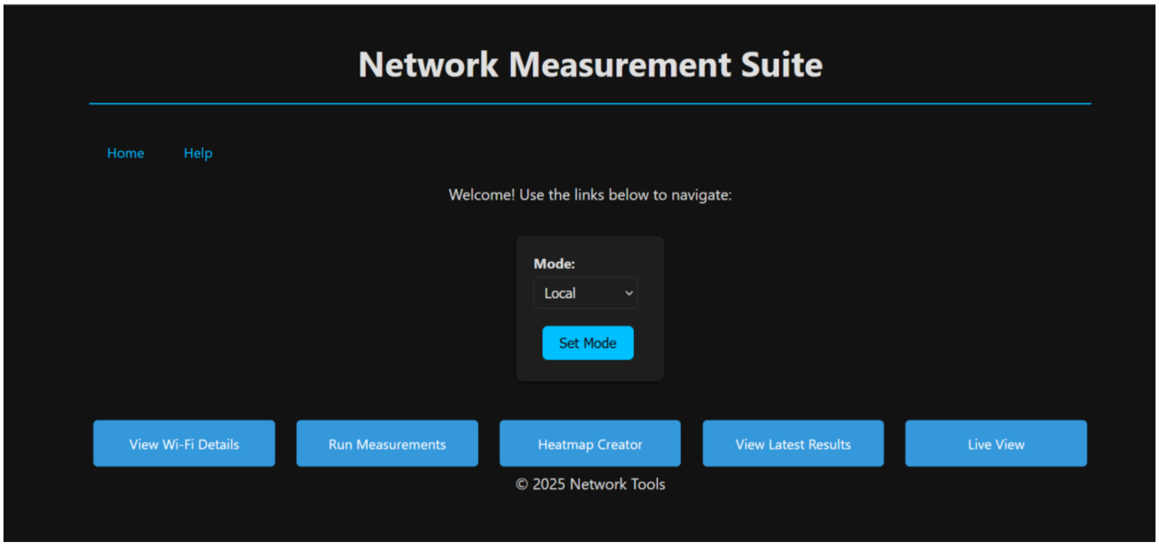


Figure 4: Web Application Homepage.

This dark mode style is present in all tabs of the application as you can see in the figure 5 below.

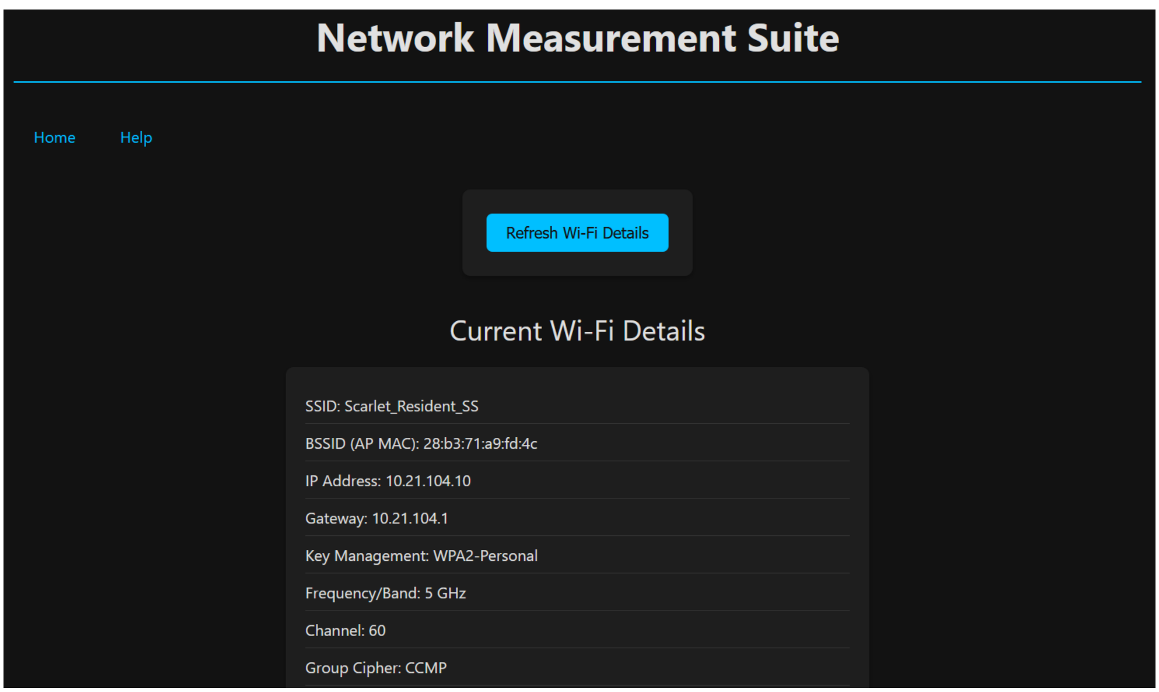


Figure 5: Network Measurement Page.

B. Back End

The software basis for this project is based on the Python language. Python is renowned for its extensive library of modules and its wide compatibility with a variety of devices. This is essential for the integration of more complex network techniques, as these libraries facilitate protocol management, communication between devices, and data processing and analysis. Flask, a web-based microframework in Python, makes it easy to create lightweight, high-performance backend interfaces, while Ionic React offers a modern, responsive structure for frontend development, ensuring a fluid, intuitive user experience. The combination of these technologies makes it possible to develop a user-friendly web application while maintaining a robust, extensible architecture.

C. API

The use of a Flask API [10] plays a fundamental role in communication between the frontend and backend of the web application. Its purpose is to manage requests sent by the user interface or front end and respond to them by providing the requested information from the back end. When a user interacts with the frontend, for example by clicking on the “View Wi-Fi Details” button as seen in Figure 4 above, an HTTP request is sent to the Flask API. The Flask API intercepts the request, processes it, and returns a response in JSON format, which can be displayed on the user interface. All of this relies on the definition of routes in, which are access points for executing different actions depending on the HTTP method used. Flask can also handle POST requests, which are used when the frontend needs to send data to the backend. The Flask API keeps the backend and frontend separated, making the web application more modular and scalable.

The API acts as a gateway between the two, enabling data to be exchanged securely and efficiently.

D. Functionalities

i. Modes

As seen in the homepage in section 2A, the user can pick between different modes in order to collect data. The modes, as seen below in Figure 6, are “Local”, “PiZero” and “ Client-Host”.

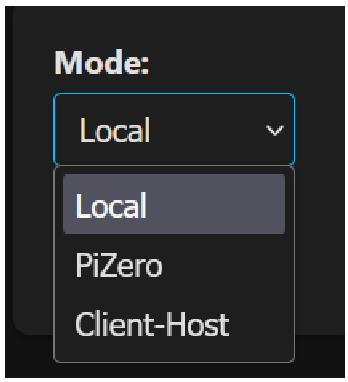


Figure 6: Data Collection Modes.

The “Local” mode means that the data will be collected from the device the application is run. While the “Client-Host” mode allows the user to collect data using two devices. Finally, the “PiZero” mode, as its name says, uses the Raspberry Pi Zero “baton” as seen in Figure 3 to collect data points. That is extremely useful because it allows the user to select the points on their laptop and then walk around the floorplans collecting data on these points using the “baton” Raspberry Pi Zero. This allows for a mobile and quick collection of data.

ii. Wi-Fi Information

This tab displays all the information about the wireless connection the router has collected, such as the name of the SSID, the frequency or the IP address as seen below in Figure 7.

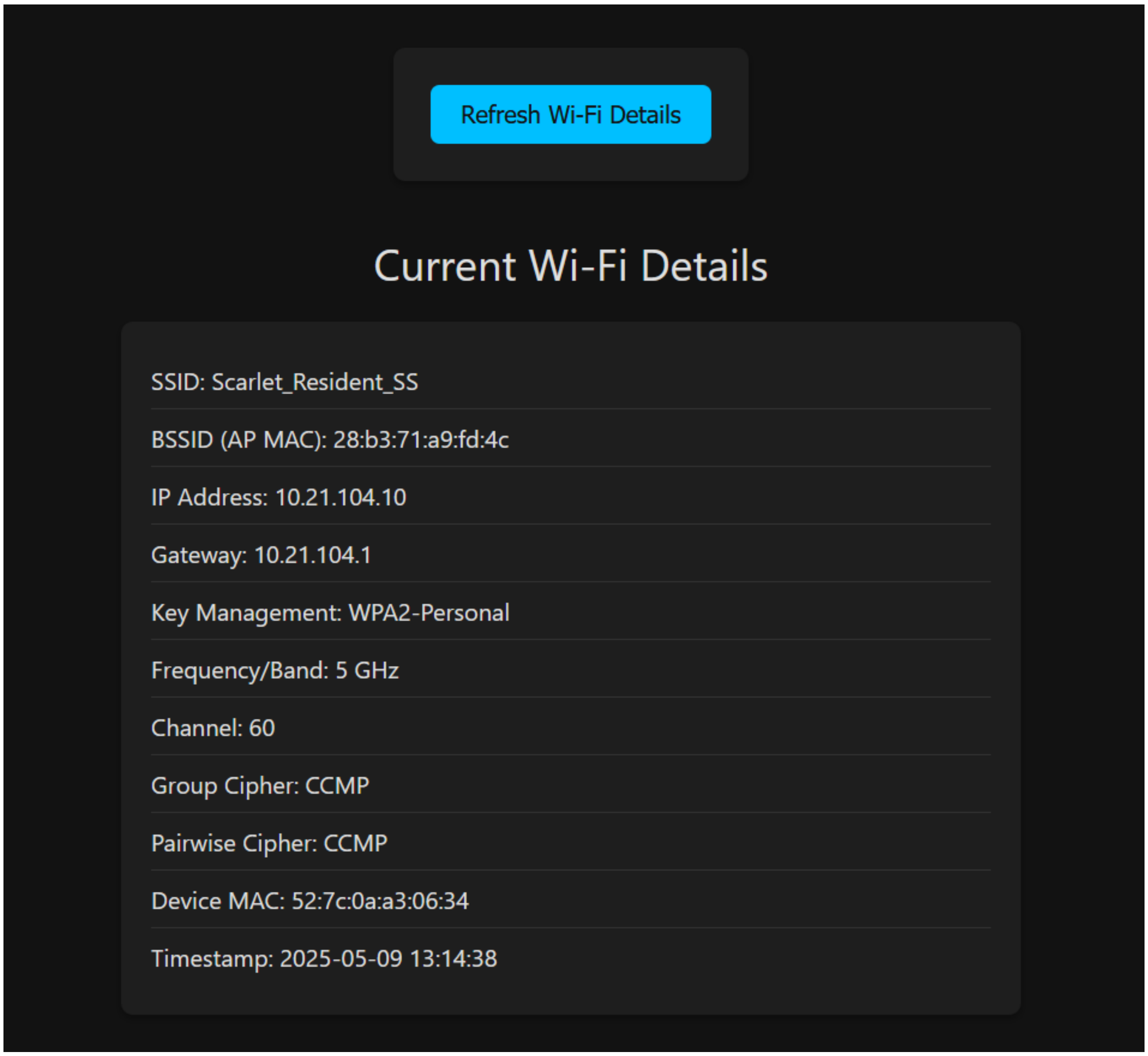


Figure 7: Wi-Fi Details Page.

iii. Heatmaps

One of the best features of this application is its ability to generate heatmaps. As seen below in Figure 8 the user is first prompted to upload a map of the area they want to generate a heatmap of or to reuse an existing floorplan if it isn't the first time the user has generated a heatmap of this area.

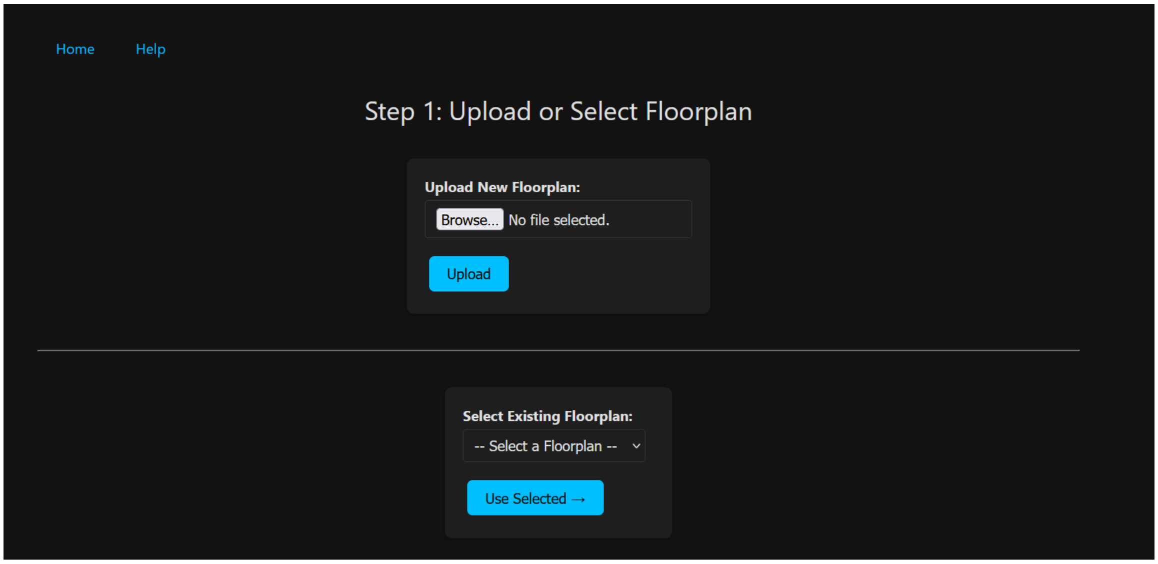


Figure 8: Heatmap Creation Page.

After that, depending on the mode picked previously (as explained in the section D.i) the user picks different points where he wants data to be collected to then move with his device or the raspberry pi zero to them. Once the data collected different heatmaps, such as latency jitter or signal strength will be generated onto the floor plan as seen below in figure 9 and in Appendix E and F.

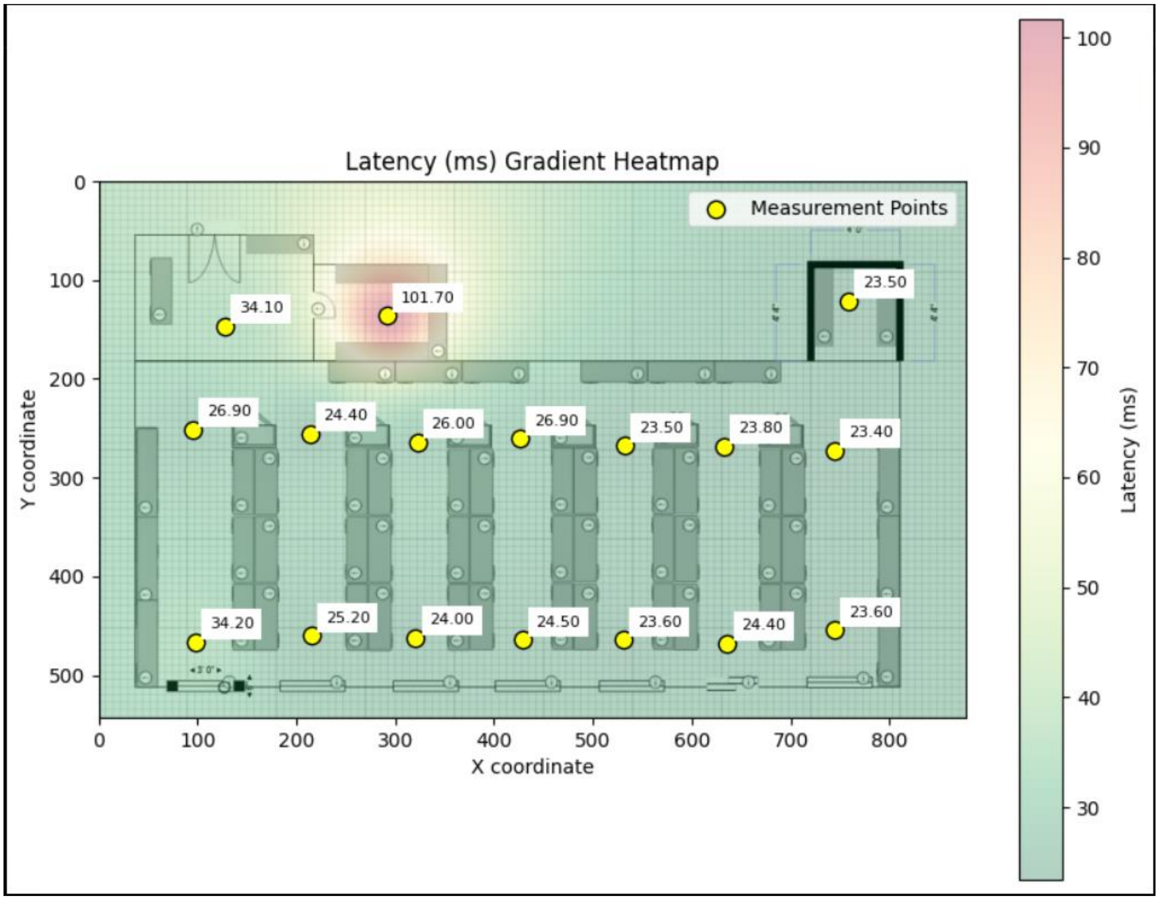


Figure 9: Generated Heatmap of the ECE106 Laboratory.

As seen above in Figure 9, the latency is good everywhere in the laboratory except in the Faraday cage, which makes sense and proves the efficiency of the project.

iv. Live View

The packet analysis process that generated the graphs visible in figure 10 below begins with the continuous capture of every frame crossing the targeted network link. The lightweight, mode-dependent capture tool time-stamps and records each packet, then immediately classifies it according to type: management (tags, probes, associations), control (ACK, RTS/CTS) or data. Totals are calculated in real time to feed the pie chart. At the same time, the tool records the number of retransmissions and CRC or protocol errors.

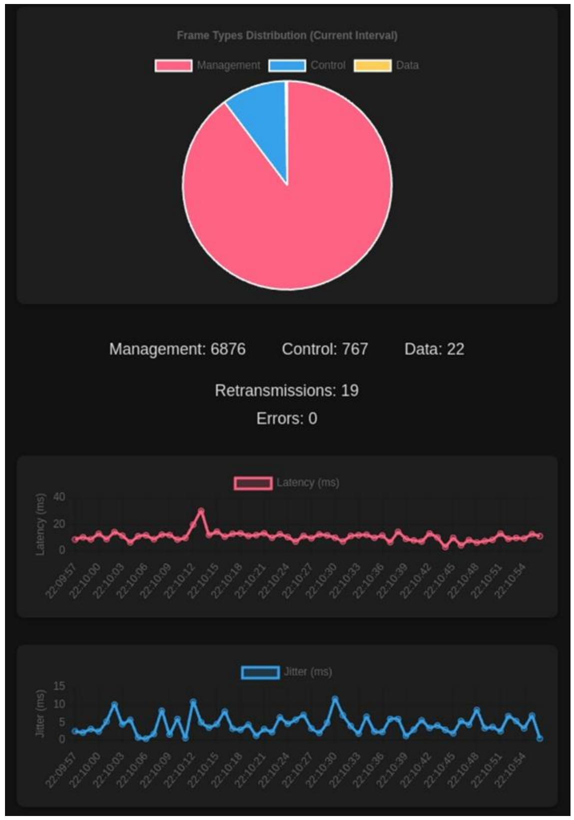


Figure 10: Live View of the Packet Analysis.

For each data packet successfully received, the probe calculates one-way latency by comparing transmission and reception timestamps, and jitter by measuring the variance of inter-arrival times. Lost packets are detected when sequence numbers contain skips or missing acknowledgments, enabling a percentage of packet loss to be deduced for each interval. This creates the graph seen in figure 11 below.

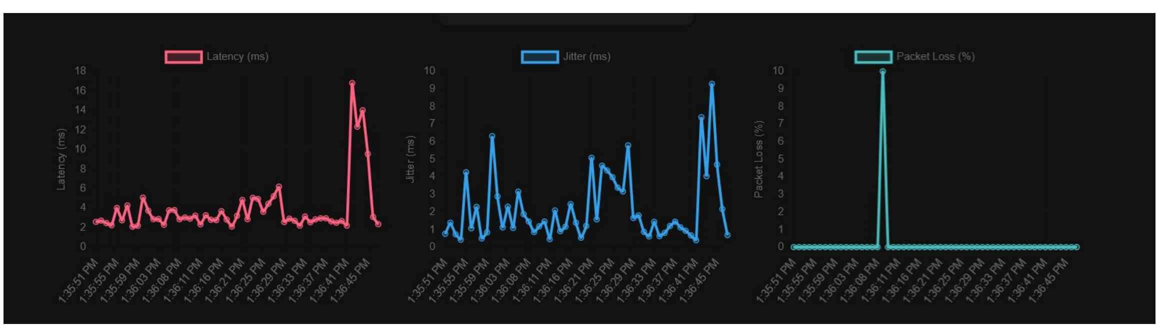


Figure 11: Live View of the Latency, Jitter and Packet Loss.

These visualizations can be used by network engineers and private individuals alike to detect congestion peaks, interference, equipment malfunctions or faulty links; to verify Quality of Service commitments; and for capacity planning or SLA reporting. This continuous packet analysis is essential for both real-time troubleshooting and long-term performance optimization.

v. Measurements

On this page, as seen on the Figure 12 below, users can find recent data on their network. They provide a comprehensive barometer of network health and performance.

Recent Measurements (Max 5)

Timestamp	Latency (ms)	Jitter (ms)	Packet Loss (%)	Download (Mbps)	Upload (Mbps)	Throughput Sent (MB/s)	Throughput Recv (MB/s)	Signal (dBm)
2025-05-03 14:03:06	N/A	N/A	N/A	N/A	N/A	N/A	N/A	-57
2025-04-29 22:02:51	23.82	1.21	N/A	N/A	N/A	N/A	N/A	-63
2025-04-29 22:02:32	N/A	N/A	100.0	N/A	N/A	N/A	N/A	-62
2025-04-25 15:55:27	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
2025-04-25 15:54:43	N/A	N/A	N/A	292.01	32.01	N/A	N/A	-56

© 2025 Network Tools

Figure 12: Previous Measurement Page.

III. Engineering Standards, Specifications, and Intellectual Property Considerations

In this Wi-Fi application project, various standards and specifications were followed to guarantee quality, compatibility, and avoid plagiarism. Web programming tools such as Flask for the backend, Ionic React for the frontend, and HTTP request management tools were used to develop and debug the application. In particular, the use of Flask to create the APIs ensured good request management between server and client, and Ionic React facilitated the interactivity of the user interface.

IV. Safety, Public Health, and Welfare Considerations

In this Wi-Fi application project, user safety, health, and well-being are top priorities. We implement security protocols to protect sensitive data and ensure secure communications between the frontend and backend. The application is designed to minimize risks to users' health by avoiding overloading devices and preventing overheating. The interface is optimized to deliver a smooth, pleasant experience while being energy-efficient to avoid device fatigue or degradation. What's more, the data generated is stored securely and disposed of correctly when no longer required, protecting users' privacy. These measures guarantee a safe environment for users while preserving their health and well-being while using the application.

V. Global, Cultural, Social, Environmental, and Economic Factor Considerations

In this project for the Digital Communication Laboratory at Texas Tech University, a strong emphasis is put on sustainability and efficiency. The integration of security and power management protocols not only protects the application from the risks associated with data overload but also contributes to optimizing energy efficiency. By reducing resource consumption and preventing device overheating, the aim is to extend equipment lifetimes while minimizing electronic waste. Reducing the risk of overheating aims to extend the equipment's lifespan, minimizing electronic waste and promoting resource conservation, like the careful use of the more expensive parts of the project, such as the Raspberry Pi 4 and Raspberry Pi Zero. The hardware parts are important to protect as they total to 77.88 USD, see Appendix G. This mindful approach underlines a strong commitment to responsible engineering practices.

VI. Conclusion

In conclusion, the development of this network analysis application highlights the harmonious integration of various components and systems, each playing an essential role in ensuring efficient and secure control of network connections. The integration of security protocols, data consumption management, and frequency control algorithms illustrates not only advances in wireless communication technologies but also the potential for synergy between sophisticated web applications and hardware solutions. The overall design, both hardware and software, has proven its efficiency and reliability, guaranteeing stable and secure network performance. As technology advances, this project is a testament to the ingenuity and potential of integrating advanced coding solutions with hardware solutions to optimize network utilization.

VII. References

[1] Raspberry Pi 4 - https://www.raspberrypi.com/products/raspberry-pi-4-model-b/

[2] Broadcom BCM43438 - https://forums.raspberrypi.com/viewtopic.php?t=251161

[3] BrosTrend long-range Wi-Fi adapter - https://www.amazon.com/BrosTrend-1200Mbps-Adapter-Wireless-

Antennas/dp/B01IEU7UZ0/ref=asc_df_B01IEU7UZ0?mcid=28d4733cdd2438d4b88d6068367cb79f&hvocijid=1125809801420908264-B01IEU7UZ0-&hvexpln=73&tag=hyprod-20&linkCode=df0&hvadid=721245378154&hvpos=&hvnetw=g&hvrand=1125809801420908264&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=9028540&hvtargid=pla-2281435180058&psc=1

[4] Realtek RTL88x2BU chip - https://forums.raspberrypi.com/viewtopic.php?t=285488

[5] Raspberry Pi Zero - https://www.raspberrypi.com/products/raspberry-pi-zero/

[6] Raspberry Pi Zero Power Train Schematic made by Nicholas Woodward

[7] Raspberry Pi Zero Power Train Picture taken by Nicholas Woodward

[8] Raspberry Pi Zero Case Design made by Nicholas Woodward

[9] React Ionic - https://ionicframework.com/docs/react

[10] Flask API - https://flask.palletsprojects.com/en/stable/

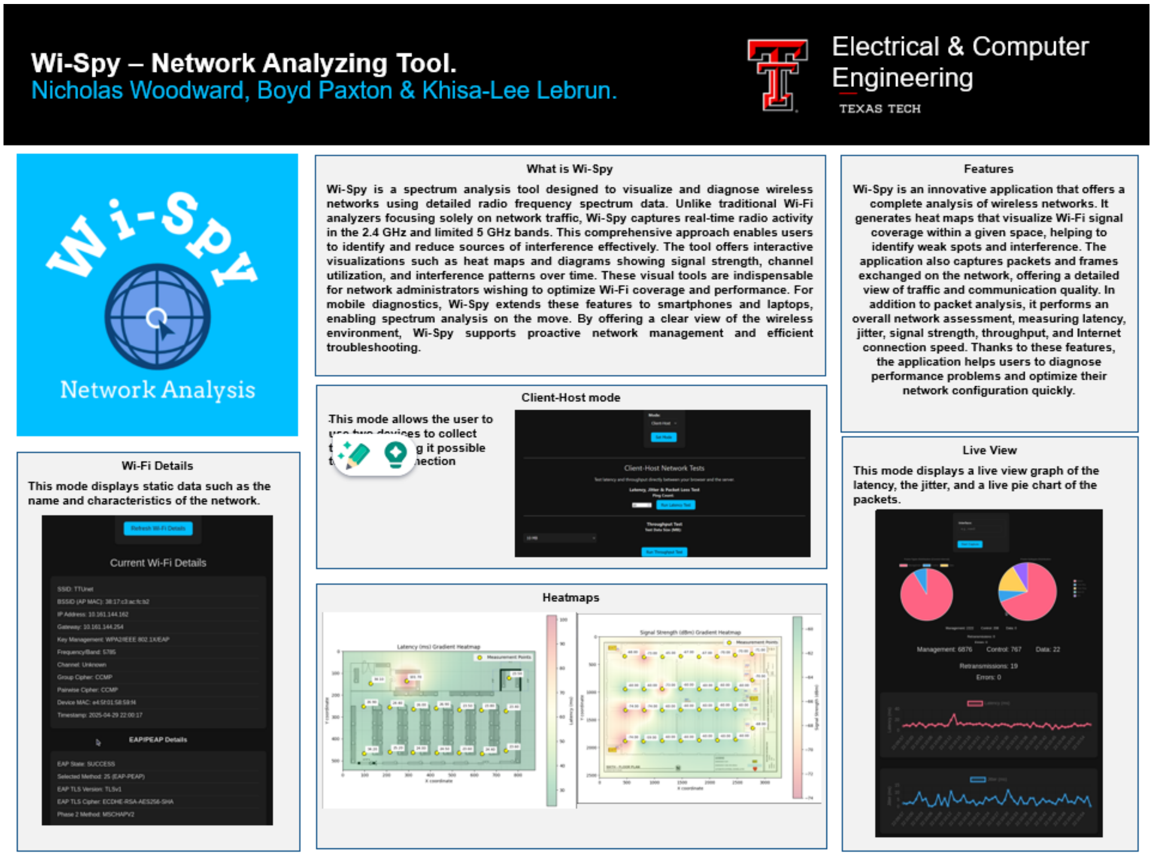
[11] Picture of the Raspberry Pi Zero System taken by Nicholas Woodward

[12] Map of the ECE 106 Laboratory made by Khisa-Lee Lebrun.

VIII. Appendices

Appendix A: Demo Day Presentation Poster

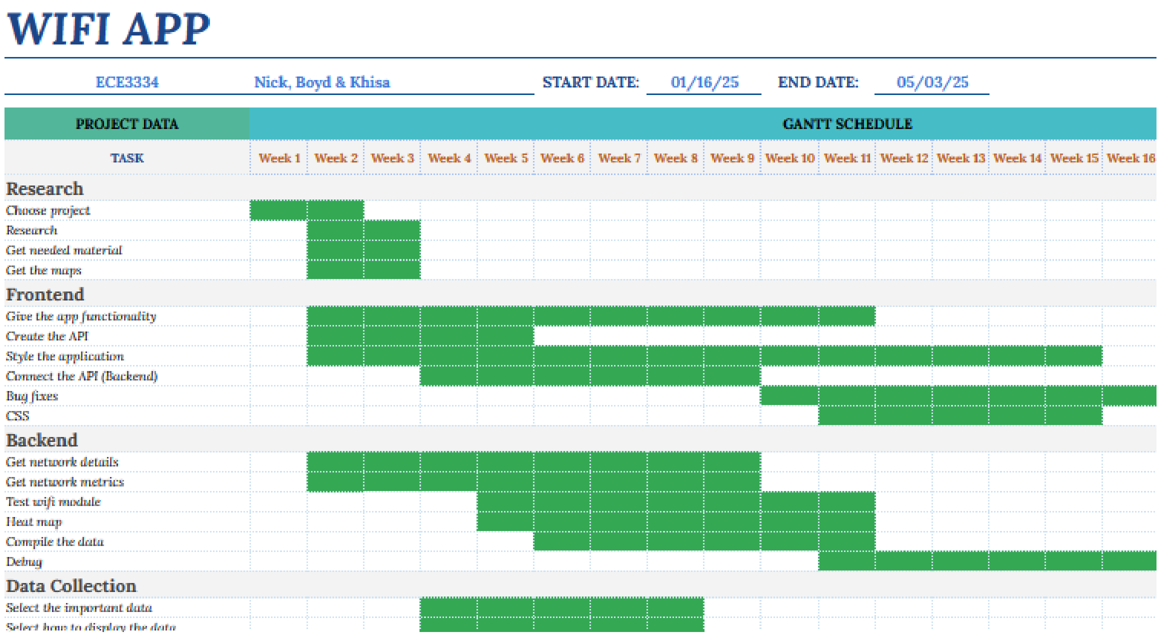
This section provides a view of the poster that was used to present the project. It displays explanations of its functionality in a simple and comprehensible way.



This poster describes the functionalities and features of the Wi-Spy application. This poster was used as a visual aid.

Appendix B: Project Gantt Chart.

This section provides a view of Gantt chart of the project.



This section shows the Gantt chart for this project. The project began on January 16, 2025 and will end on May 3, 2025. The schedule is divided into five main categories: research, frontend development, backend development, data collection and testing. Each task is represented by a green bar indicating its expected duration. Tasks include application creation, API integration, user interface design, network data processing and results analysis. This planning allows you to monitor the project's progress in a clear and organized way, while efficiently distributing responsibilities between group members. The poster also shows the coordination between the various phases, in particular the interconnection between frontend and backend, essential to the smooth running of the application.

Appendix C: Picture of the Raspberry Pi Zero System. [11]

This section provides a view of the "Baton" style casing of the Raspberry Pi Zero System made to collect data points in a portable manner.

This sections shows the "Baton" system used to collect the data points by encasing the Raspberry Pi Zero circuit into this baton shaped case.

Appendix D: CSS Styling Code for the Web Application

This section provides a view of the code used to style the application through CSS.

style.css

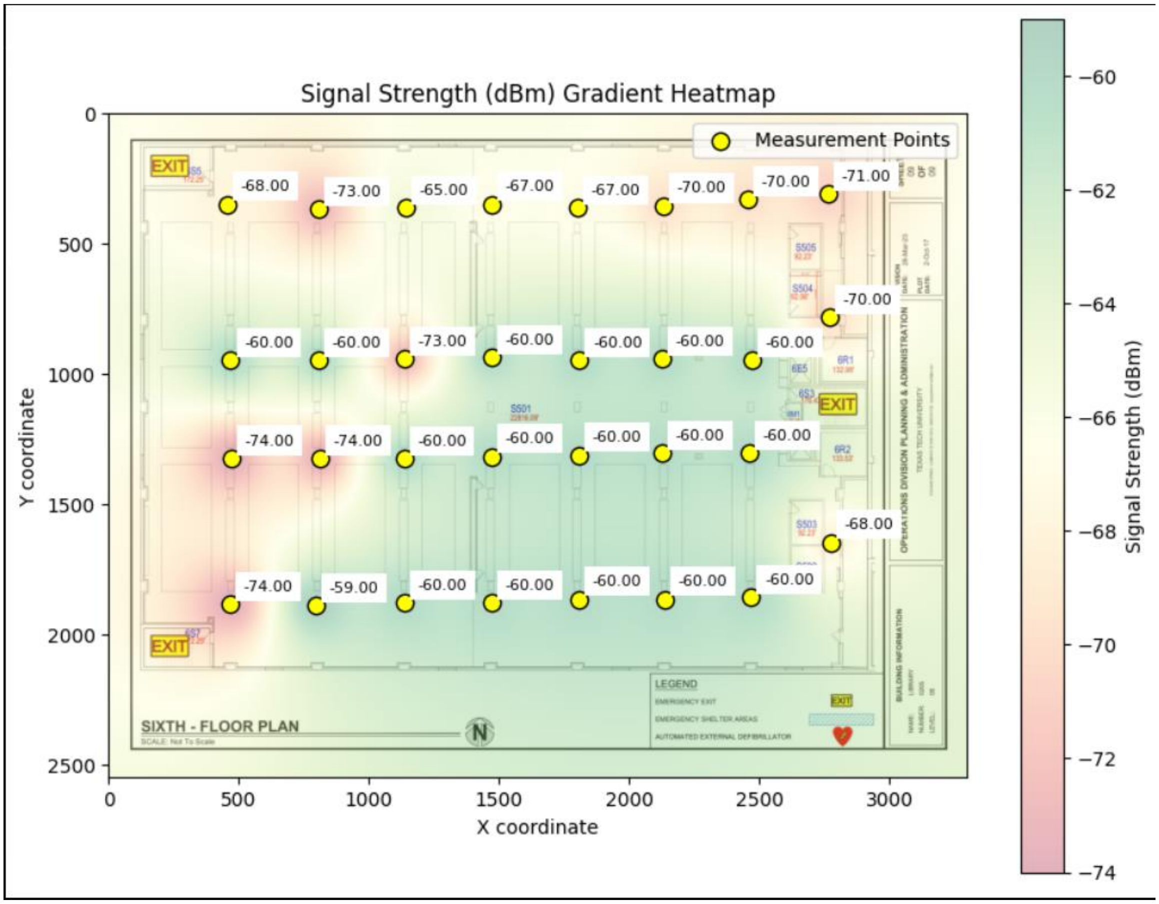
C: > Users > lebru > Desktop > appv7 > static > style.css > *

```css
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

body {
    font-family: 'Segoe UI', Arial, sans-serif;
    background-color: #121212;
    color: #E0E0E0;
    padding: 20px;
    text-align: center;
    margin: 0 auto;
    max-width: 1200px;
}

h1 {
    color: #E0E0E0;
    font-size: 2.5em;
    border-bottom: 2px solid #00BFFF;
    padding: 20px 0;
    margin-bottom: 20px;
}

h2 {
    color: #E0E0E0;
    font-size: 1.8em;
    margin: 15px 0;
    font-weight: 500;
}
```

This section shows the contents of a CSS file called style.css, used to style the web application. The styling begins with a global reset (lines 1 to 4) that removes default margins and paddings, and applies a uniform box template with box-sizing: border-box. The body is styled with a sans-serif font, a dark background and a light text color. Content is centered and limited to a maximum width of 1200 pixels with a padding of 20 pixels. h1 titles are styled with a large font size, light blue underlining and defined vertical spacing. The h2 titles are slightly smaller, with a custom margin and intermediate font size. This CSS file gives a modern, readable style, well suited to a sober, professional and content-centric user interface.

Appendix E: Heatmap of the fifth floor of the Texas Tech Library.

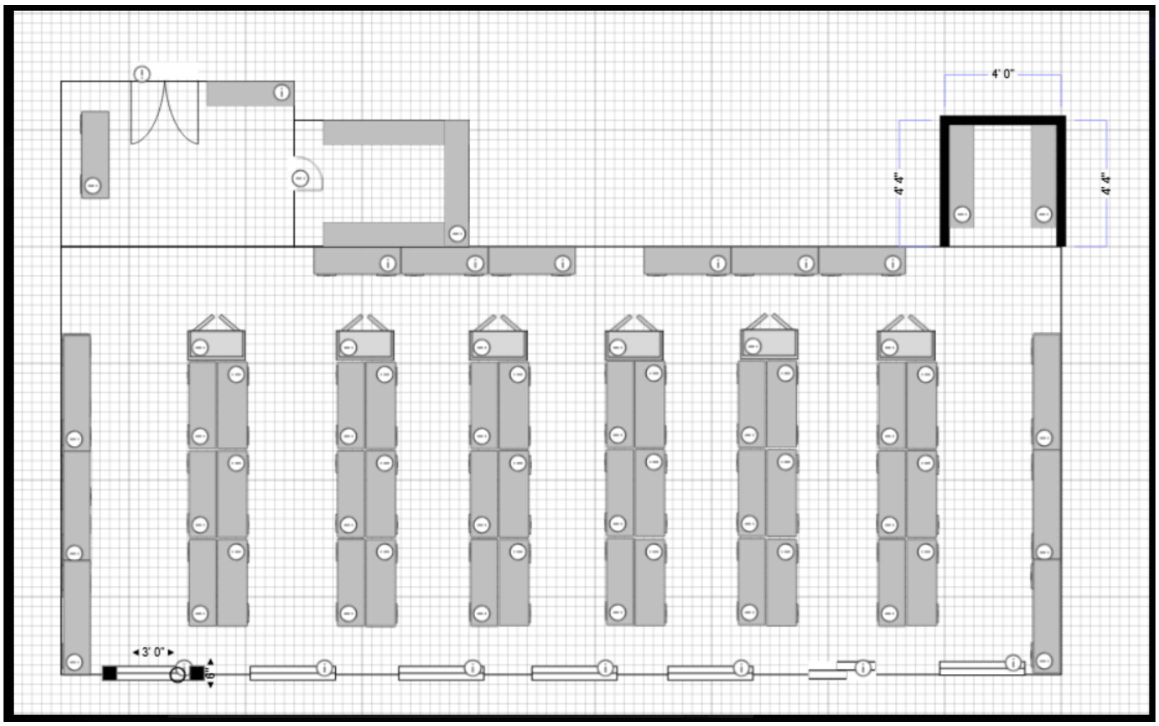
This section provides a view of a signal strengths heatmap of the 5th floor of the library on the Texas Tech University Campus.



This section presents the signal strength heat map (in dBm) of the Texas Tech University library. The image shows a floor plan of the sixth floor, with coordinates in x (0 to 3000) and y (0 to 2500). Measuring points, indicated by yellow circles, show signal values ranging from -74 dBm (weak areas, in red) to -68 dBm (stronger areas, in green). Outputs are marked at the top left, middle right and bottom left corners. The legend on the right shows a gradation from -60 dBm (light green) to -74 dBm (dark red), reflecting signal quality. A notable area around (500, 1500) shows a strength of -74 dBm, while central areas such as (1500, 1000) reach -60 dBm.

Appendix F: Map of the 106 ECE Laboratory. [12]

This section provides a map made by Khisa-Lee Lebrun in order to display heatmap without having to change buildings during the demonstration of the project. This map is not to scale and was only made as a way to display the capabilities of this project.



This section presents a map created by the team to use the laboratory room as a demonstration. The image shows a plan with a grid, representing a rectangular space divided into sections. On the left, a small area includes a door and an adjacent room. In the center, six rows of tables or equipment, each with marked dots, line up vertically, suggesting workstations. To the right, an exit is indicated by a blue rectangle with a door.

Appendix G: Material Cost Budget

This section provides the budget for the project.

WiSPY Budget							
		Running Total			Total Estimate		
Direct Labor:		rate per hour	hrs	total	rate per hour	hrs	total
Nick		18	168	$3 024,00	18	220	$3,300.00
Boyd		18	182	$3 276,00	18	220	$3,300.00
Khisa		18	169	$3 042,00	18	220	$3,300.00
DL subtotal:				$9 342,00			$13,200.00
Material Cost:							
Bought				Price	Budget : $100		
ESP32				$15,99			
Stepup converter				$12.99			
Raspberry pi 0 2w				$23.99			
Rented							
Raspberry Pi 4				$61,89			
MC subtotal:				$77,88			
Total				$9 419,88			

In summary, this diagram describes the price of each element purchased for the project as well as the cost per hour of labor.