Disaster-Resilient Telecommunication Infrastructure: A Systematic Approach

24-25J-049

Team

Supervisor :
Ms. Dinithi Pandithage

Co-Supervisor :
Prof. Pradeep Abeygunawardhana

Widanage W.T.N

IT21377426

Bandara H.K.K.T

IT21176074

S.M.T.S Senaratna

IT21337512

Muhandiramge M.D.A.D

IT21383434

Introduction

- Disasters disrupt communication networks, complicating rescue efforts. To solve this, our project creates a resilient system using ad-hoc networks.
- Our project establishes a centralized Mobile Ad-hoc Network (MANET) using Bluetooth Low Energy (BLE) and Wi-Fi Direct to enable direct device-to-device communication.
- Designing a reliable SOS messaging system for ad-hoc networks to ensure low-latency, high-reliability communication during disasters, enabling effective emergency response in infrastructure-deprived areas.

- A self-healing algorithm ensures the network remains stable by dynamically reconfiguring connections when nodes disconnect.
- We use SQLite for local storage on each device and Cassandra for centralised data management at the base station, ensuring reliable data access during disasters and after.
- Victim localisation is implemented using Wi-Fi signal strength analysis, helping rescuers determine the approximate location of stranded individuals.

Research Problem

How can we create a reliable system that detects and relays the presence of people in disaster areas, while also transmitting SOS alerts and instructions when traditional networks fail?

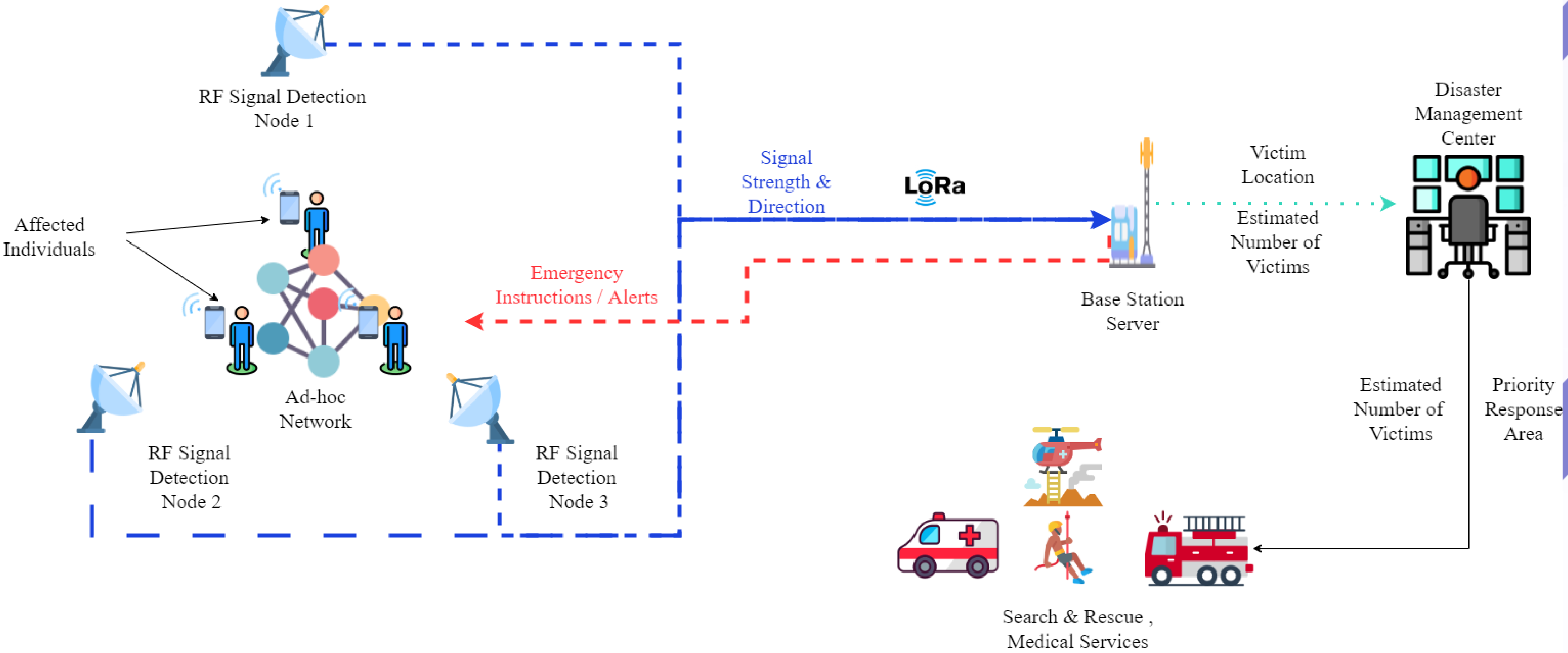
Main Objective:

To develop a robust communication system using multiple Wi-Fi signal detection devices and an ad-hoc network, enhancing disaster response by accurately identifying the presence of people in disaster areas where traditional networks have failed.

Sub-Objectives:

- Design a centralized ad-hoc network using BLE and Wi-Fi Direct for disaster communication.
- Enable efficient SOS message transmission for emergency responders.
- Implement a self-healing mechanism to ensure continuous network connectivity.
- Develop a system to approximate location of stranded individuals Wi-Fi signals.
- Integrate a database system for the base station to manage and store critical data.

Overall System Diagram



IT21176074 | Bandara H. K. K. T.

Computer System and Network Engineering

INTRODUCTION

Background

- Reliable communication is critical during emergencies to coordinate rescue efforts and ensure safety.
- Ad-hoc networks enable dynamic, infrastructure-independent communication, ideal for disaster-hit areas.
- Can be quickly established and scaled using Wi-Fi-enabled devices, providing immediate, widespread coverage.
- Utilizes existing consumer devices, minimizing the need for specialized equipment.

Research Gap

Research Paper	Objectives & Tasks	
	To design and implement a dynamic ad-hoc network architecture.	To implement and test server within the network.
Quality of Sustainability Optimization Design for Mobile Ad Hoc Networks in Disaster Areas	✓	✗
Emergency Alert Networks for Disaster Management: Applications Perspective	✗	✗
A Novel Technique for Mobile Phone Localization for Search and Rescue Applications	✗	✓
Proposed System	✓	✓

Research Question

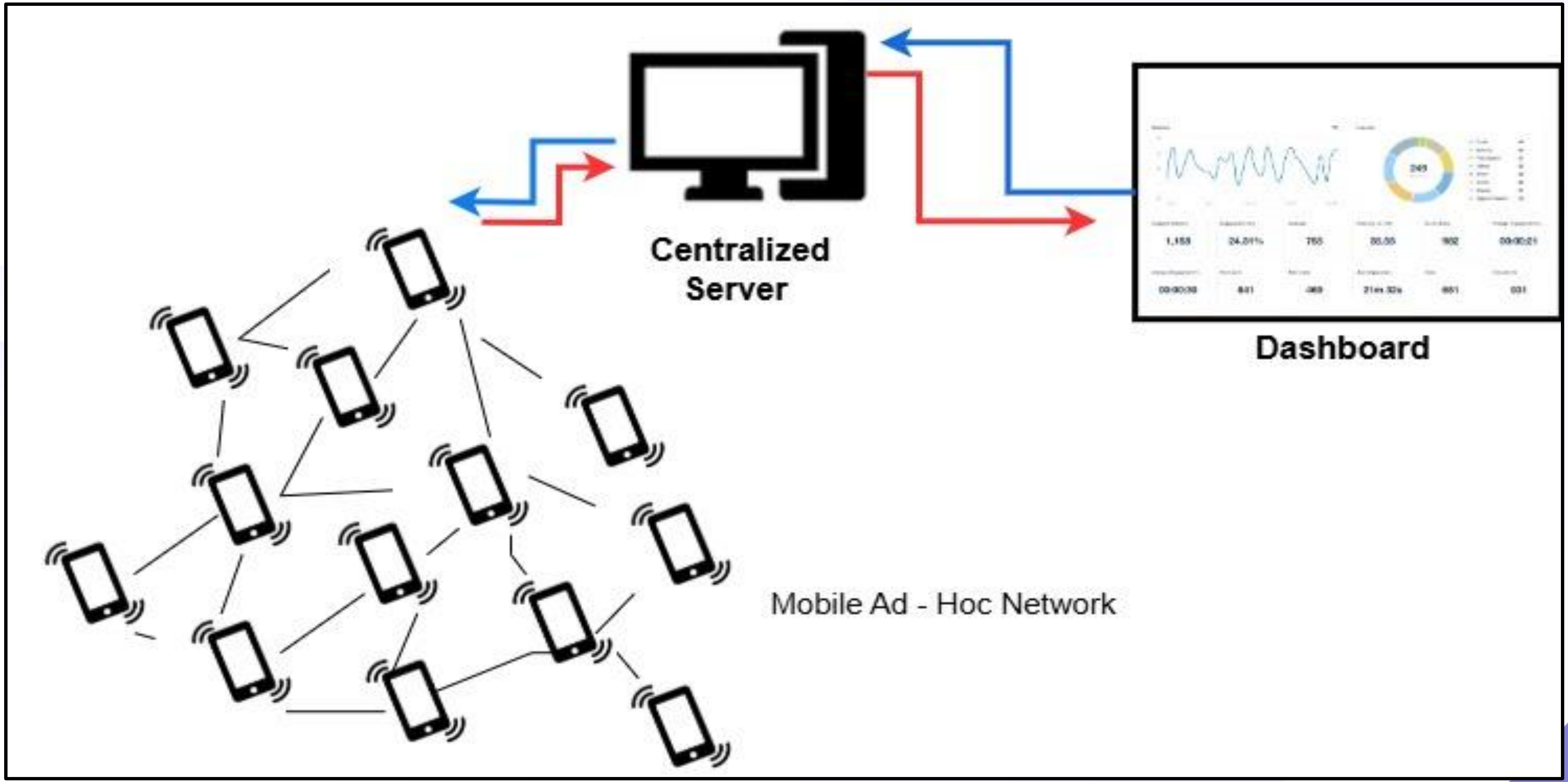
- How can a dynamic ad-hoc network architecture be designed and implemented to facilitate reliable emergency communication in disaster-affected areas without relying on existing infrastructure?

- **Main Objective**
  - To develop a disaster-resilient telecommunication infrastructure to ensure continuous communication during natural disasters.

- **Sub-Objectives:**
  - Establish an ad-hoc network for emergency communication.
  - To design and implement a dynamic ad-hoc network architecture.
  - Utilize Centralize Server to enhance the ad-hoc network.
  - To implement and test server within the network.
  - Develop a Cassandra database to store device information and all messages transmitted through the ad-hoc network.
  - Implement SQLite on each device to maintain local message logs for historical records.

METHODOLOGY

System Diagram



Technologies

1. Ad-hoc Network Technologies:

- Wi-Fi Direct-based Mobile Application – Facilitates direct peer-to-peer communication without requiring traditional network infrastructure.

2.Data Management Technologies:

- Cassandra – Stores device information and all messages sent via the ad-hoc network, ensuring scalability and reliability.
- SQLite – Maintains local message logs on each device for historical records.

Requirements

- Hardware
  - Wi-Fi Enabled Centralize Server.
  - Mobile devices

- Software
  - Custom Firmware

Completion of the project

- Created an Ad Hoc network using Wi-Fi Direct.
- Creating Ad-Hoc network Application
- Creating local database for all nodes.

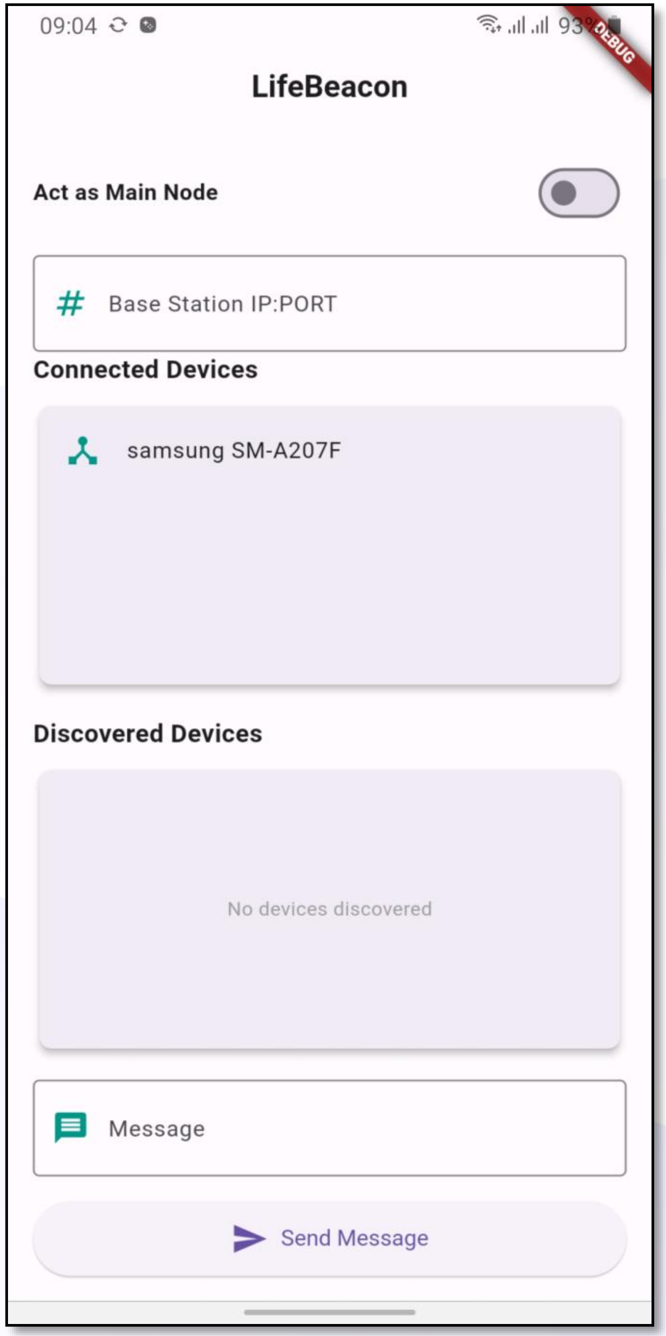
Demonstration

```
@override
void initState() {
  super.initState();
  initNearbyService();
  initNotifications();
}

void initNearbyService() {
  nearbyService.init(
    serviceType: 'mesh-network',
    strategy: Strategy.P2P_CLUSTER,
    callback: (isRunning) {
      if (isRunning) {
        // Start both advertising and discovering
        nearbyService.startAdvertisingPeer();
        nearbyService.startBrowsingForPeers();
      }
    },
  );
```

```
nearbyService.stateChangedSubscription(callback: (deviceList) {
  for (var element in deviceList) {
    switch (element.state) {
      case SessionState.connected:
        setState(() {
          if (!connectedDevices
              .any((device) => device.deviceId == element.deviceId)) {
            connectedDevices.add(element);
          }
        });
        break;
```

3/18/2025
21

```
   case SessionState.notConnected:
   // Automatically connect to found devices
     if (!connectedDevices.any((d) => d.deviceId == element.deviceId)) {
       nearbyService.invitePeer(
           deviceID: element.deviceId, deviceName: element.deviceName);
     }
     setState(() {
       if (!connectedDevices
           .any((device) => device.deviceId == element.deviceId)) {
         connectedDevices.add(element);
       }
     });
     break;
   default:
     break;
  }
 }
});
```


09:04
93%
DEBUG
LifeBeacon
Act as Main Node
Base Station IP:PORT
Connected Devices
samsung SM-A207F
Discovered Devices
No devices discovered
Message
Send Message

Gantt Chart

Process	Months											
	May	June	July	August	September	October	November	December	Janurary	February	March	April
Requirement Gathering & Initial Planning	■	■	■	■								
Network Design					■							
Middlewre Development & Hardware Setup						■	■					
Integration & Testing						■	■	■				
Final Deployment									■			
Project Review & Documentation										■	■	
Final Presentation												■

IT21383434 | MUHANDIRAMGE M.D.A.D

Computer System and Network Engineering

BACKGROUND

- The function aims to create an ad-hoc network using victims' mobile devices.
- Bluetooth Low Energy (BLE) is the one of main technology used to build the network.
- The system includes a self-healing mechanism that automatically rebuilds the network if a node disconnects.
- It ensures reliable communication in disaster areas where traditional networks are unavailable.

RESEARCH GAP

Research Paper	Objectives & Tasks		
	Implementing energy-efficient data transmission techniques.	Real-Time Data Transmission for Disaster Management	Self-healing and BLE-based communication
New Ordered Policy Routing Protocol for Active Data Transmission in Mobile Ad-hoc Networks	✗	✗	✗
Fairness Improvement and Efficient Rerouting in Mobile Ad Hoc Networks	✗	✗	✗
A Novel Technique for Mobile Phone Localization for Search and Rescue Applications	✗	✓	✗
Proposed System	✓	✓	✓

Research Question

- How can an ad-hoc network be effectively created using Bluetooth Low Energy (BLE) and mobile devices in disaster-affected areas to ensure reliable communication?
- What self-healing mechanisms can be implemented to automatically restore and maintain network connectivity in ad-hoc networks when a connected node disconnects during a disaster?

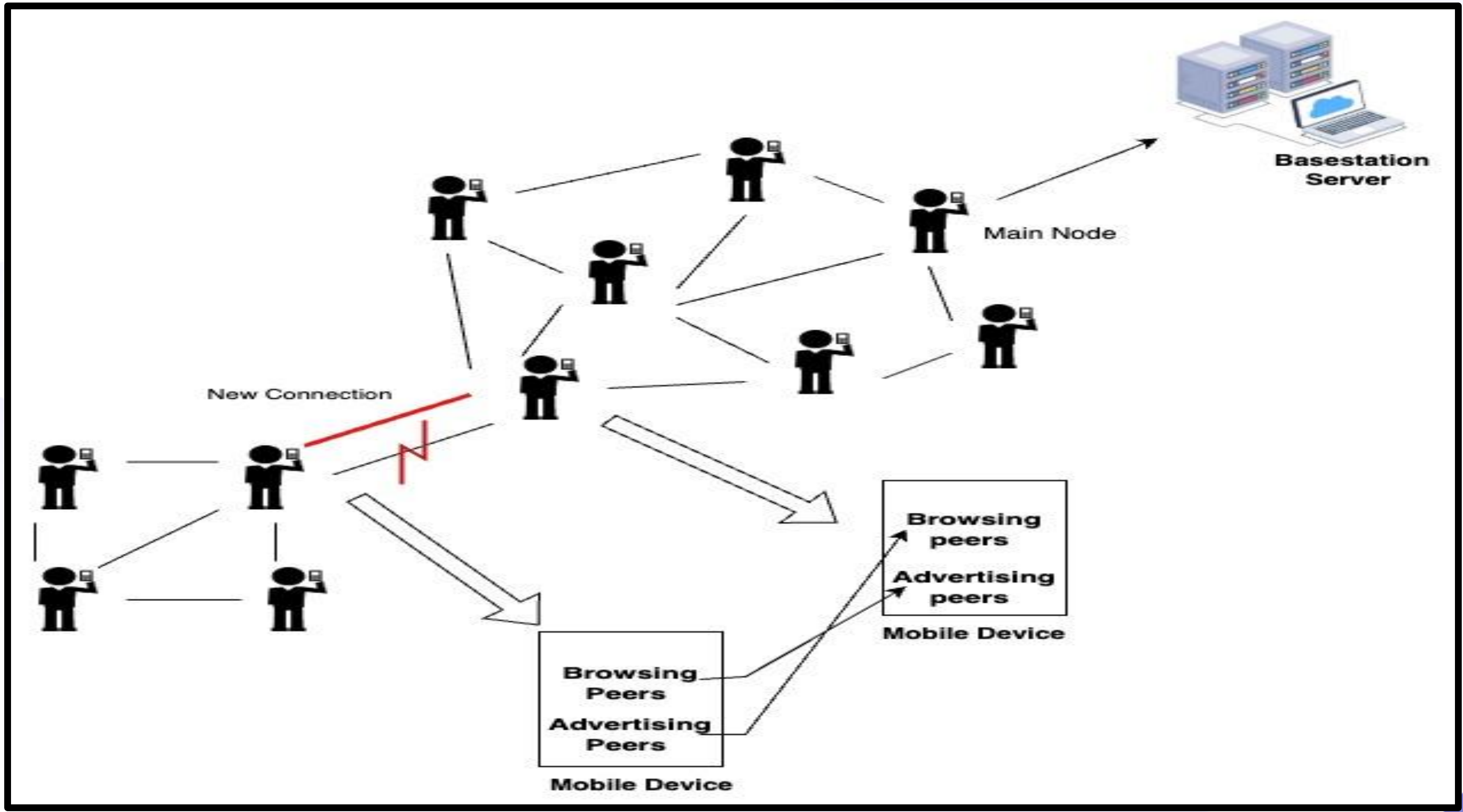
MAIN OBJECTIVES

- To create a reliable ad-hoc network using Bluetooth Low Energy (BLE) and mobile devices, ensuring continuous communication and recreating the network when a node disconnects in disaster-affected areas.

SUB OBJECTIVES

- Build an ad-hoc network using mobile devices as servers and clients with BLE technology.
- Implement a self-healing mechanism to rebuild the network when a node disconnects.
- Optimize BLE for low-power, efficient communication in the network.
- Test and validate the network's reliability and self-healing function in disaster scenarios

METHODOLOGY – SYSTEM DIAGRAM



METHODOLOGY

- Create an ad-hoc network using mobile devices in the disaster area with Bluetooth Low Energy (BLE).
- Ensure devices communicate efficiently within the network.
- Implement a self-healing mechanism to reconnect devices if a node disconnects.
- Rebuild the network automatically when a node leaves or joins. Enable dynamic device management to maintain continuous network operation.
- Test the system to ensure reliable performance in disaster scenarios.

METHODOLOGY - TECHNOLOGIES

1. Bluetooth Low Energy (BLE)
2. Self-Healing Algorithm

SYSTEM, PERSONAL, AND SOFTWARE REQUIREMENTS SPECIFICATION

System Requirements

- Centralized server:
- Ad-Hoc Network Components:

Personnel Requirements

- System Engineers
- Network Engineers

Software Requirement

- Communication Protocol Software
- Software development tools

Completion of the project

- Created an Ad Hoc network using Bluetooth Low Energy .
- Creating Ad-Hoc network Application
- Implementing self-healing mechanism that automatically rebuilds the network if a node disconnects.

Need to be done

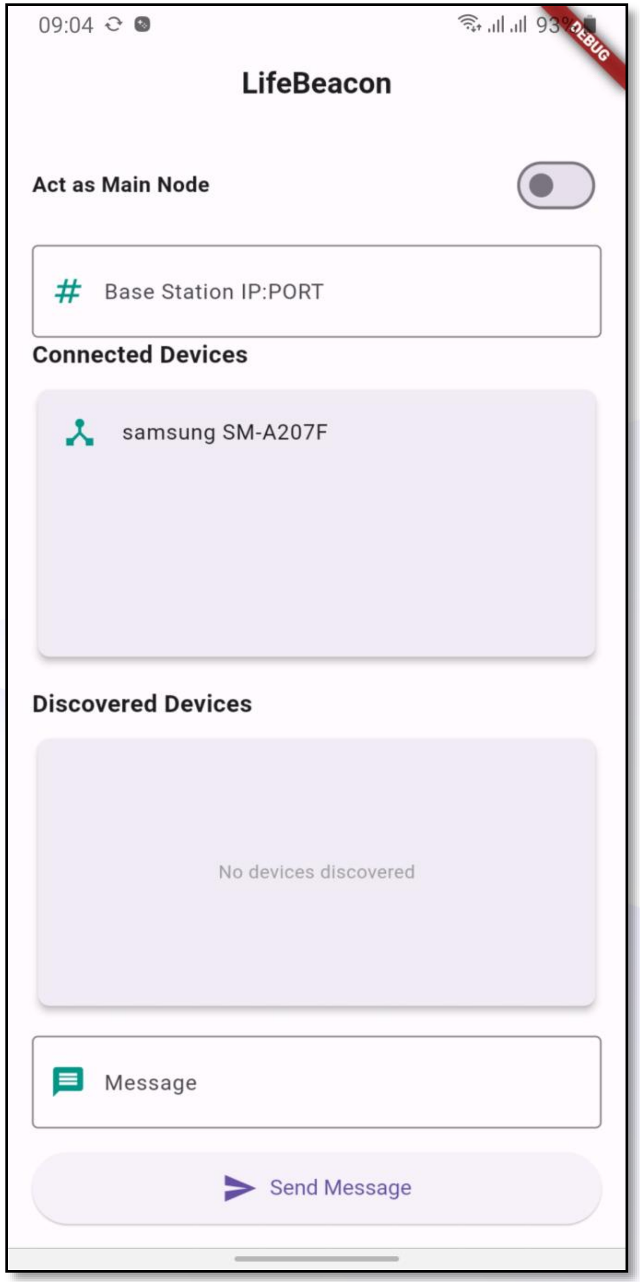
- Need to complete the testing phase and work on improving performance to ensure optimal functionality.

Demonstration

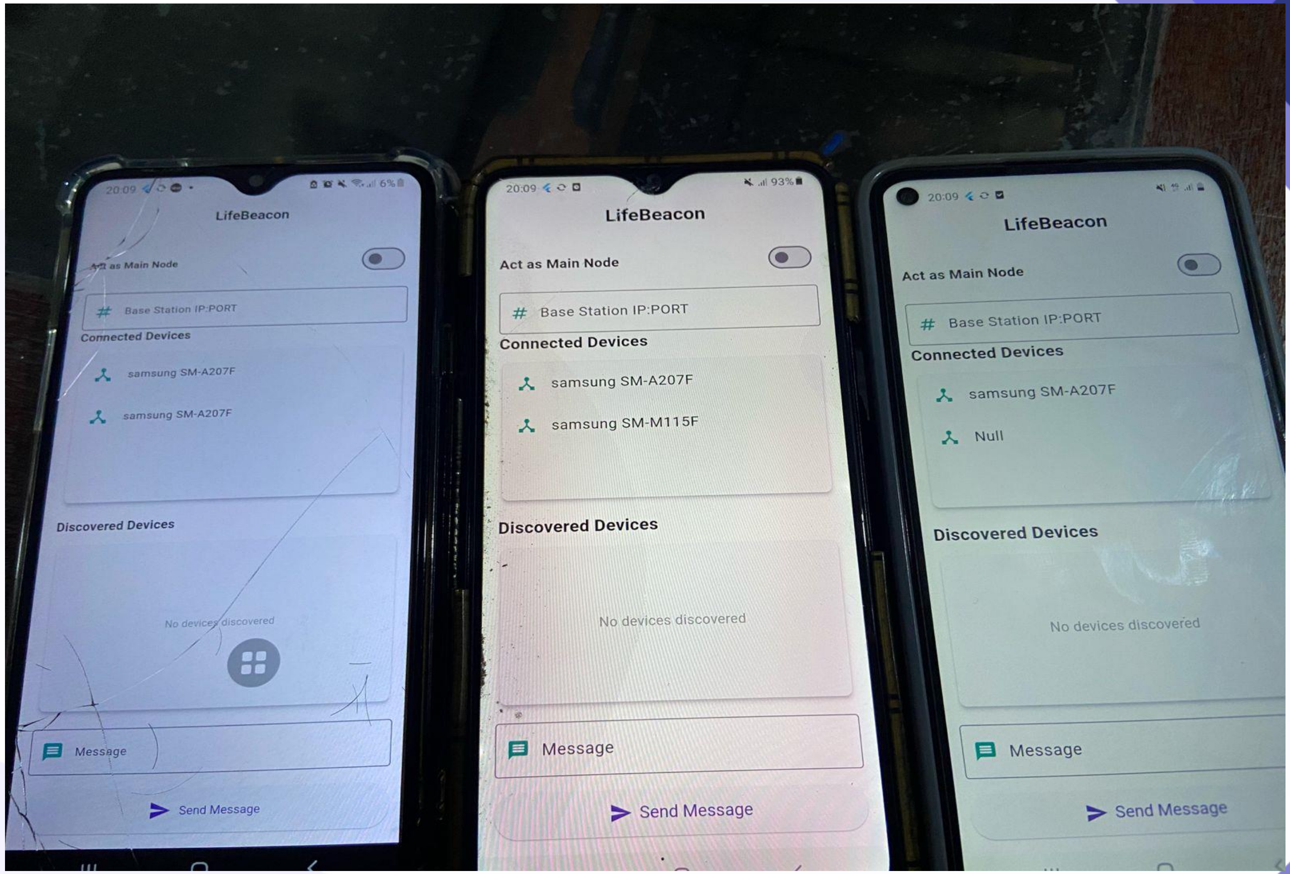
```dart
import 'dart:async';
import 'dart:convert';
import 'dart:typed_data';
import 'package:flutter_bluetooth_serial/flutter_bluetooth_serial.dart';

class BackgroundCollectingTask {  //Manages the Bluetooth connection.
  final BluetoothConnection _connection;
  final List<int> _buffer = [];   //A temporary list to store raw data received from the Bluetooth device.
  final List samples = [];   // A list that stores parsed data for later use.
  bool inProgress = false;

  BackgroundCollectingTask._fromConnection(this._connection) {
    _connection.input!.listen((data) {
      _buffer.addAll(data);   //Sets up a listener for incoming data and appends it to

      while (true) {
        int index = _buffer.indexOf('t'.codeUnitAt(0));
        if (index >= 0 && _buffer.length - index >= 7) {
          _buffer.removeRange(0, index + 7);  // Removes processed data from the buffer.
          samples.add(_buffer);  //Stores valid data in samples.
        } else {
          break;
        }
      }
    }).onDone(() {   //Triggered when the Bluetooth connection is closed, setting inProgress to
      inProgress = false;
    });
  }

  static Future<BackgroundCollectingTask> connect(  //Establishes a connection to a specified Bluetooth device and returns an instance of BackgroundCollectingTask.
      BluetoothDevice server) async {
    final connection = await BluetoothConnection.toAddress(server.address);  //Initiates the connection using the device's address.

    return BackgroundCollectingTask._fromConnection(connection);
  }

  void dispose() {
    _connection.dispose();  //Cleans up the connection to release resources.
  }

```

Main code is used to stablish connection using BLE

35



Interface of the mobile application



Connected device for the application

GANTT CHART


Process
Months
May
June
July
August
September
October
November
December
Janurary
February
March
April
Requirement Gathering & Initial Planning
Network Design
Middlewre Development & Hardware Setup
Integration & Testing
Final Deployment
Project Review & Documentation
Final Presentation

IT21377426 | Widanage W. T. N.

Computer Systems and Network Engineering

BACKGROUND

- Disasters can disrupt traditional communication, leaving affected areas without reliable means to connect. Ad-hoc networks play a crucial role in these situations, enabling spontaneous and independent communication.
- This project focuses on designing and implementing an **SOS messaging system** for **reliable emergency communication**. Key features include:
  Broadcast SOS System – Mass alerts for wide coverage
  Encrypted Direct Messaging – Secure, targeted assistance
  Centralized Dashboard – Message management, victim monitoring, and device data access
- The system ensures **low latency** and **high reliability**, making it ideal for the dynamic nature of ad-hoc networks in disaster scenarios.

Research Question

- How can an SOS messaging system be optimized to ensure low-latency, high-reliability communication and secure encrypted messaging for effective emergency response in ad-hoc networks?

Main Objective

- To design and optimize a robust SOS messaging system that ensures low-latency, high-reliability communication and secure, encrypted messaging within ad-hoc networks during disaster situations.

Sub Objectives

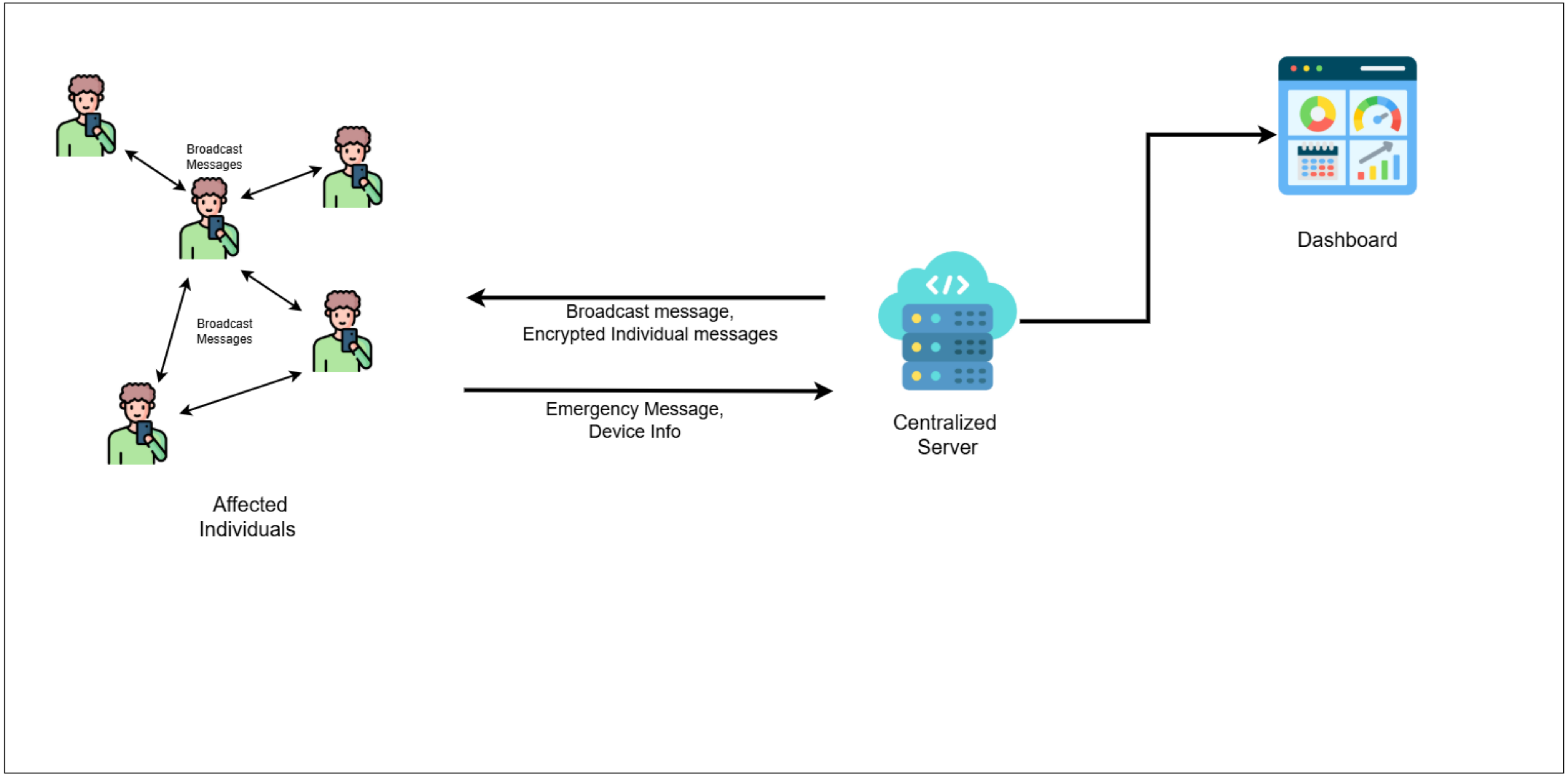
- To assess the performance of existing SOS messaging systems in terms of latency, reliability, and security, and identify key areas for improvement within ad-hoc networks.

- To develop and implement an optimized SOS messaging system that integrates broadcast and encrypted direct messaging for effective communication in disaster situations.

- To design a centralized dashboard for managing broadcasts, tracking victim data, and enabling real-time communication, ensuring efficient coordination of emergency response efforts.

- To ensure the enhanced SOS messaging system operates seamlessly on mobile devices used by rescue teams, providing reliable communication even in dynamic and infrastructure-less environments.

System Diagram



Methodology– Technologies

- **Flask (Python):**
  - Serves as the backend framework for managing SOS messaging.
  - Implements Flask-SocketIO for real-time WebSocket communication.
  - Handles message broadcasting, direct messaging, and encryption.
- **Flutter (Mobile App):**
  - Integrates the messaging system within the mobile app.
  - Uses flutter_nearby_connections for peer-to-peer communication.
  - Receives, decrypts, and displays messages.
- **Flutter Nearby Connections Plugin:**
  - Enables peer-to-peer messaging using the P2P_CLUSTER strategy.
  - Establishes communication even in infrastructure-less environments.
- **Notification Service:**
  - Uses Flutter Local Notifications for standard alerts.
  - Implements Raw Resource Sounds for high-priority emergency notifications.

Implementation Steps:

1. **Server-Side Messaging:** Flask-SocketIO establishes WebSocket connections, manages broadcasts, and encrypts direct messages.
2. **Mobile App Messaging:** The Flutter app handles P2P messaging, receives, decrypts, and displays SOS messages.
3. **Device Information & Logging:** The server retrieves and manages active devices, logs messages, and maintains connectivity data.
4. **Encryption & Security:** RSA encryption ensures private message security using dynamically generated key pairs.
5. **Real-Time Communication:** The centralized dashboard controls message dispatch, monitors active devices, and manages encrypted messaging.

Completion of the project

- Developed a real-time SOS messaging system for ad-hoc networks.
- Integrated encryption for secure, targeted messaging.
- Implemented a centralized dashboard for message management, victim tracking, and device monitoring.

3/18/2025 46

System Requirements:

- **Server:** Python server for Flask-based messaging.
- **Mobile Devices:** Android smartphones with Bluetooth, Wi-Fi, and Internet connectivity for SOS messaging.
- **Network:** Ad-hoc network capability (Wi-Fi Direct, Bluetooth, WebSocket) for reliable message transmission.

Software Requirements:

- **Backend:** Flask (Python) with Flask-SocketIO.
- **Frontend (Mobile App)**: Flutter (Dart) with flutter_nearby_connections.
- **Encryption Library:** PyCrypto (Python) & PointyCastle (Dart) for RSA encryption.
- **Real-Time Communication:** WebSockets (Socket.IO), Bluetooth, Wi-Fi Direct.
- **Notification Services:** Flutter Local Notifications for alerts.

Progress

- Sending messages to connected peers.

```
void sendMessage(String message) {
  for (Device device in connectedDevices) {
    nearbyService.sendMessage(device.deviceId, message);
  }
  print('Message Sent: $message');
}
```


3/18/2025 49

- Receiving and processing messages from peers.

```
nearbyService.dataReceivedSubscription(callback: (data) {
  _showNotification(data['message']);
  showDialog(
    context: context,
    builder: (context) => AlertDialog(
      title: Text('Message Received'),
      content: Text(data['message']),
      actions: [
        TextButton(
          onPressed: () => Navigator.pop(context),
          child: Text('OK'),
        ), // TextButton
      ],
    ), // AlertDialog
  );
});
}
```

- Real-time messaging integration with a server.

```
void startListeningToSocketServer() {
  print("Listening to sock");
  IO.Socket socket = IO.io(
    'http://192.168.43.210:12345',
    IO.OptionBuilder()
        .setTransports(['websocket'])
        .disableAutoConnect()
        .build(),
  );

  socket.connect();

  socket.onConnect((_) {
    print('Connected to server');
  });

  socket.on('message', (data) {
    print('Message received from server: $data');
    sendMessage(data.toString());
  });

  socket.onDisconnect((_) {
    print('Disconnected from server');
  });
}

void stopListeningToSocketServer() {
  print("Stopped listening to the socket server...");
}
```

- User alerts for incoming messages.

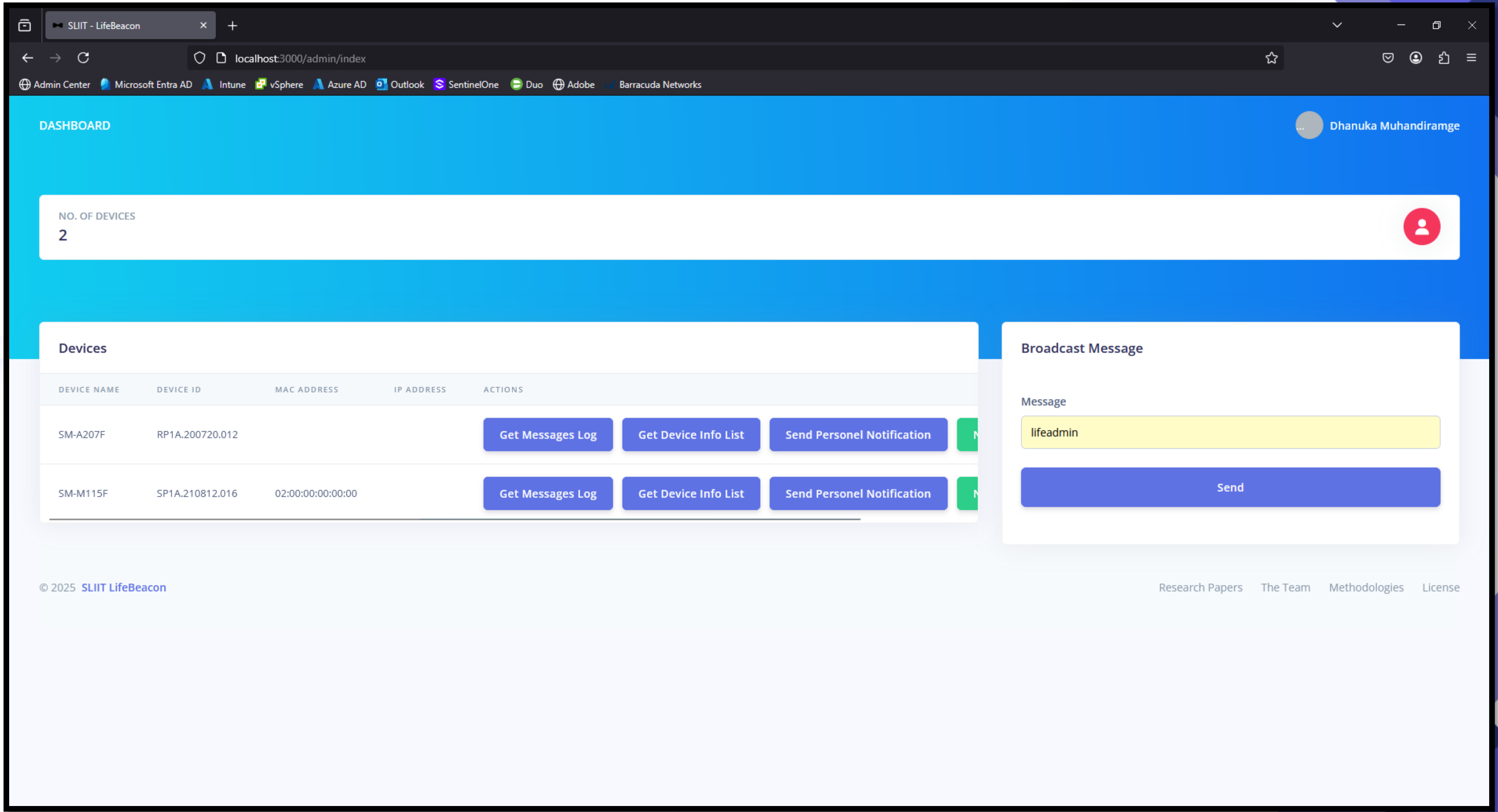
```dart
Future<void> _showNotification(String message) async {
  const AndroidNotificationDetails androidNotificationDetails =
      AndroidNotificationDetails(
    'mesh_channel', // Channel ID
    'Mesh Messages', // Channel Name
    importance: Importance.high,
    priority: Priority.high,
    playSound: true,
    sound: RawResourceAndroidNotificationSound('siren'),
  ); // AndroidNotificationDetails
  const NotificationDetails notificationDetails =
      NotificationDetails(android: androidNotificationDetails);
  await flutterLocalNotificationsPlugin.show(
    0, // Notification ID
    'Message Received', // Notification Title
    message, // Notification Body
    notificationDetails,
  );
}
```

- Message input and send button.

```
// Message Input Field
TextField(
  controller: messageController,
  decoration: InputDecoration(
    labelText: 'Message',
    border: OutlineInputBorder(),
    prefixIcon: Icon(Icons.message, color: Colors.teal),
  ), // InputDecoration
), // TextField
SizedBox(height: 16),
// Send Message Button
ElevatedButton.icon(
  onPressed: () {
    if (messageController.text.isNotEmpty) {
      sendMessage(messageController.text);
      messageController.clear();
    }
  },
  icon: Icon(Icons.send),
  label: Text('Send Message'),
  style: ElevatedButton.styleFrom(
    padding: EdgeInsets.symmetric(vertical: 12),
    textStyle: TextStyle(fontSize: 16),
  ),
), // ElevatedButton.icon
```


3/18/2025
53

- Dashboard



3/18/2025
54

IT21337512 | Senaratna S.M.T.S

Computer Systems & Network Engineering

Background

- Traditional communication networks are often disrupted during natural disasters, making it difficult to identify victims through conventional means.
- A RF Signal (Wi-Fi) based detection and analysis methodology is proposed to identify victims by detecting Wi-Fi probe requests emitted by devices in affected areas.
- Portable devices equipped with sensors and communication modules detect and measure Wi-Fi signal strength.
- Collected data is transmitted to a central base station via an ad-hoc network, ensuring continuous data relay without relying on conventional communication infrastructure.
- The collected data is analyzed centrally to estimate the approximate coordinates of the victim's device, aiding in search and rescue operations.

Research Gap

Research Paper	Objectives & Tasks		
	Integration with Disaster Management Centers	Non-Intrusive Operation	Use of multiple devices for increased accuracy.
A Novel Technique for Mobile Phone Localization for Search and Rescue Applications	✗	✗	✗
A Doppler Effect Based Framework for Wi-Fi Signal Tracking in Search and Rescue Operations	✗	✓	✗
A Smartphone-assisted Post-Disaster Victim Localization Method	✗	✓	✗
Proposed System	✓	✓	✓

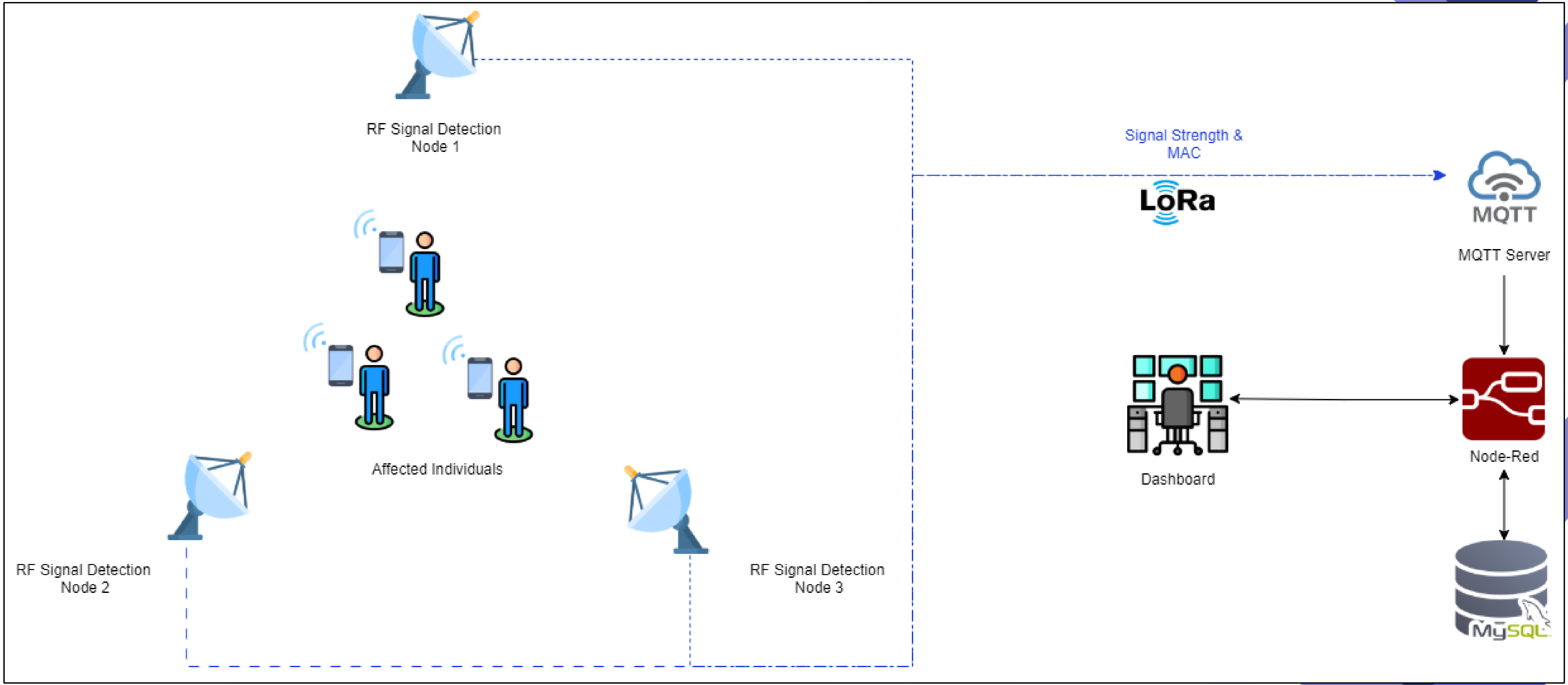
Research Question

- How can RF technologies be leveraged to detect and determine the approximate location of individuals in disaster-affected areas?

- Objective
  - To develop a system that utilizes RF technologies to collect data and determine the approximate location of individuals in disaster-affected areas.

- Sub-objectives
  - To Implement a Data Collection System:
    Develop mechanisms to collect data using RF signals emitted from smartphones and compatible devices. Utilize Wi-Fi or Bluetooth technology to gather data from devices within range.
  - To Ensure Data Accuracy:
    Implement filtering techniques to enhance data accuracy, eliminating redundancies or false positives in locating affected individuals.
  - To Evaluate and Optimize the System:
    Evaluate the system's performance and optimize the system.

Methodology – System Diagram



Methodology - Technologies Used

1. Microcontroller-Based Platform
    - Utilizes a flexible and programmable embedded system for building RF signal detection devices.
    - Supports integration with various sensors and communication modules.
2. Wi-Fi Modules & Antennas
    - Employed to detect and measure Wi-Fi signal strength and facilitate data transmission.
3. LoRa
    - Long-range, low-power communication technology for transmitting RF signals over large distances.
    - Ideal for remote detection nodes with low data transmission needs, covering the data transmission requirements.
4. MQTT
    - Lightweight messaging protocol for efficient data transmission.
    - Ensures reliable communication between devices and the data processing & storage node.
5. Node-RED
    - Facilitates real-time data processing and visualization.
6. MySQL Database
    - For storing and managing RF signal data.
    - Supports efficient querying, storage, and retrieval of large datasets from the detection devices.

System, personal, and software Requirements specification

- Hardware Requirements

  - Microcontrollers
  - Wi-Fi modules & antennas for RF signal detection.
  - LoRa modules & antennas for LoRa Communication
  - Power supply components / batteries for device operation in the field.
  - Enclosures and mounting hardware for device protection and deployment.
  - Displays to convey device status & critical information

System, personal, and software Requirements specification - Required Skills and Knowledge

- Microcontrollers
  - Proficiency in programming and interfacing with microcontroller-based platforms for detecting RF signals and enabling communication with various modules.
- Wi-Fi and LoRa Modules
  - Knowledge of integrating and configuring Wi Fi modules and antennas to detect and measure RF signal strength.
  - Understanding of LoRa technology for long range, low power communication in remote sensing applications.
- MQTT & Data Communication
  - Understanding of lightweight messaging protocols, specifically MQTT, for efficient and reliable data transmission.
  - Knowledge of ensuring seamless communication between devices and data processing/storage nodes.
- Networking & Data Protocols
  - Understanding of networking protocols and configurations, with a focus on RF signal detection and communication.
  - Familiarity with integrating data transmission technologies like Wi Fi, Bluetooth, and LoRa for networked devices.
- Database Management
  - Knowledge of MySQL or other relational databases for storing, managing, and querying large datasets.
  - Understanding of efficient data retrieval and processing for real time applications.

System, personal, and software Requirements specification - Software Requirements

- Development Environment
  - Programming tools and environments suitable for developing code for microcontroller-based platforms.
- Custom Firmware
  - Firmware to interface with detection nodes & LoRa modules, as well as handle data transmission and reception efficiently.
- Networking Libraries
  - Libraries for handling communication.
  - Support for LoRa communication.
- Database Integration
  - Tools for storing, querying, and retrieving data efficiently in real-time applications.

Progress as of Now

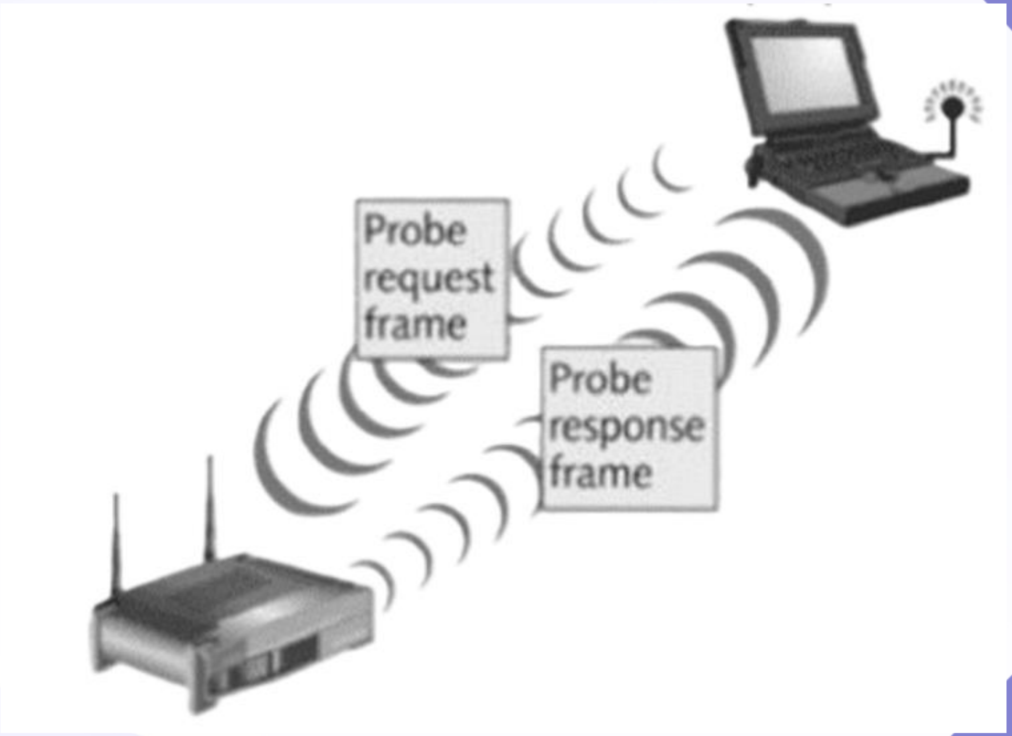
1. **Detection Technology Comparison & Selection – Completed**
2. **Microcontroller Selection, Hardware Acquisition & Implementation**
    1. Selection of Embedded System (Microcontroller) – Completed
    2. Hardware Acquisition – Completed
    3. Final Hardware Implementation – In-progress
3. **Detection Device Programming**
    1. Device Detection – Completed
    2. Data Filtering Mechanism Finalization – Completed
    3. Signal Strength to distance conversion technique - Completed
    4. Location Approximation (Signal Trilateration Algorithm) – Pending
    5. Connectivity between Nodes – In-progress
4. **Hardware Design & Implementation – In Progress**

1. Detection Technology Comparison & Selection

- **GSM Signal Capturing**:
  - **Advantages**: Could detect devices connected to GSM networks.
  - **Challenges**:
    - Requires use of a **Jammer** to disrupt existing connections, which can be highly disruptive.
    - Complex implementation due to mimicking GSM network security features.
    - High cost and complexity of required hardware.
    - Legal challenges in obtaining and deploying GSM-capturing & jamming hardware.
- **Wi-Fi Probe Request Capturing** (Selected):
  - **Advantages**:
    - No disruption to existing networks.
    - Ability to leverage promiscuous mode for non-intrusive data collection.
    - Cost-effective and simpler implementation.
    - Minimal legal or regulatory restrictions.

Understanding Wi-Fi Probe Requests

- A Wi-Fi probe request is a type of management frame defined in the IEEE 802.11 standard.
- It is sent by Wi-Fi clients to actively search for available networks.
- These requests are transmitted even when no network is present, as the device is scanning for networks to join.



Wi-Fi Probe Requests

- **Key Features:**
  - Broadcasted: Probe requests are sent publicly and can be captured by any device in range.
  - Contains Metadata:
    - MAC Address: Unique identifier for the device.
    - SSID: Network name (if the device is searching for a specific one).
- **Relevance to the Project:**
  - Enables real-time tracking of devices in disaster-affected areas.
  - Provides non-disruptive data collection without affecting existing network operations.

2. Embedded System / Microcontroller Selection, Hardware Acquisition & Implementation

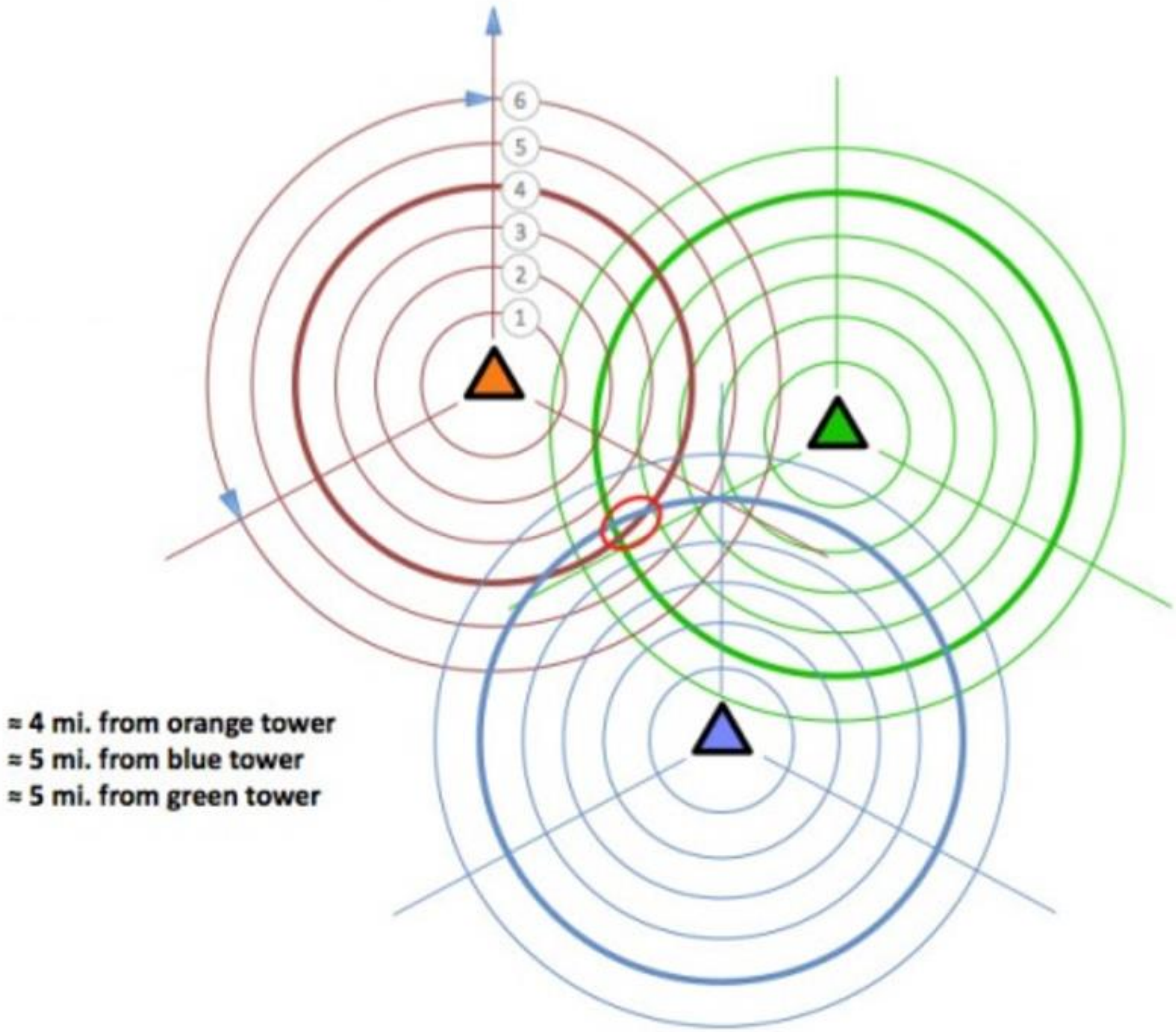
- Selected Device : Espressif Systems 32-bit Microcontroller : (ESP32-WROOM-32U)

- **Why ESP32?**
  - Built-in Wi-Fi capabilities.
  - ESP32-WROOM-32U model allows the integration of external antennas for longer range and accuracy.
  - Supports promiscuous mode for packet sniffing.
  - Efficient power consumption allows for more portable & user-friendly device implementation.
  - Compact, cost-effective, and suitable for portable applications.

3. Detection Device Programming

- Wi-Fi Probe Request Capturing:
  - Operates in promiscuous mode to sniff packets.
  - Captures and processes metadata like:
    - RSSI (Signal Strength)
    - Channel
    - MAC Address
    - SSID (if available).
- Features:
  - Channel Hopping: Scans channels (1–13) for wider coverage.
  - Packet Processing: Filters for probe requests and extracts relevant metadata.
- Data Filtering:
  - Eliminates redundancies and false positives.
  - Filters out incomplete packets and corrupted packets to ensure accurate data.
  - Ensures accuracy for estimating individual counts in disaster scenarios.

Location Approximation Methodology (Trilateration)



Progress

1. Probe-request Capturing

```
10:30:56.026 -> RSSI: -39 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSID: Test 2
10:30:56.026 -> RSSI: -39 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSID: Test3SSID
10:30:56.069 -> RSSI: -39 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSID:
10:30:56.069 -> RSSI: -38 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSID: SLT fiber
10:30:56.069 -> RSSI: -37 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSID: SLT fiber
10:30:56.069 -> RSSI: -37 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSID: Test 2
10:30:56.069 -> RSSI: -38 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSID: Test3SSID
10:30:56.214 -> RSSI: -69 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSID:
10:30:56.214 -> RSSI: -78 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSID: SLT fiber
10:30:56.214 -> RSSI: -68 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4
10:30:56.214 -> RSSI: -68 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4
10:30:56.214 -> RSSI: -68 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSID:
10:30:56.245 -> RSSI: -71 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSID: SLT fiber
10:30:56.245 -> RSSI: -72 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSID: Test3SSID
10:30:56.245 -> RSSI: -77 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSID:
10:30:56.245 -> RSSI: -70 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSI
10:30:56.245 -> RSSI: -70 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSI
10:30:56.245 -> RSSI: -70 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSID: Test 2
10:30:56.279 -> RSSI: -79 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSID: Test3SSID
10:30:56.279 -> RSSI: -69 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSID: Test 2
10:30:56.279 -> RSSI: -69 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSID:
10:30:56.320 -> RSSI: -68 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSID: SLT fiber
10:30:56.320 -> RSSI: -84 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSID: Test3SSID
10:30:56.320 -> RSSI: -85 Ch: 8 Peer MAC: 7a:20:82:c7:54:d4 SSID:
10:31:01.983 -> RSSI: -69 Ch: 1 Peer MAC: 00:00:00:00:01:00 SSID:
10:31:01.983 -> RSSI: -28 Ch: 1 Peer MAC: 4e:eb:d0:8f:de:95 SSID:
10:31:02.015 -> RSSI: -64 Ch: 1 Peer MAC: a8:80:55:21:f3:c6 SSID:
10:31:02.196 -> RSSI: -67 Ch: 1 Peer MAC: 4e:eb:d0:8f:de:95 SSID:
10:31:02.230 -> RSSI: -27 Ch: 1 Peer MAC: a8:80:55:21:f3:c6 SSID:
10:31:02.680 -> RSSI: -64 Ch: 1 Peer MAC: c2:86:0f:82:a4:b4 SSID:
10:31:02.772 -> RSSI: -68 Ch: 1 Peer MAC: a8:80:55:21:e3:15 SSID:
10:31:02.805 -> RSSI: -34 Ch: 1 Peer MAC: 9e:56:ab:6c:86:bb SSID:
```

SSID

Device MAC Address

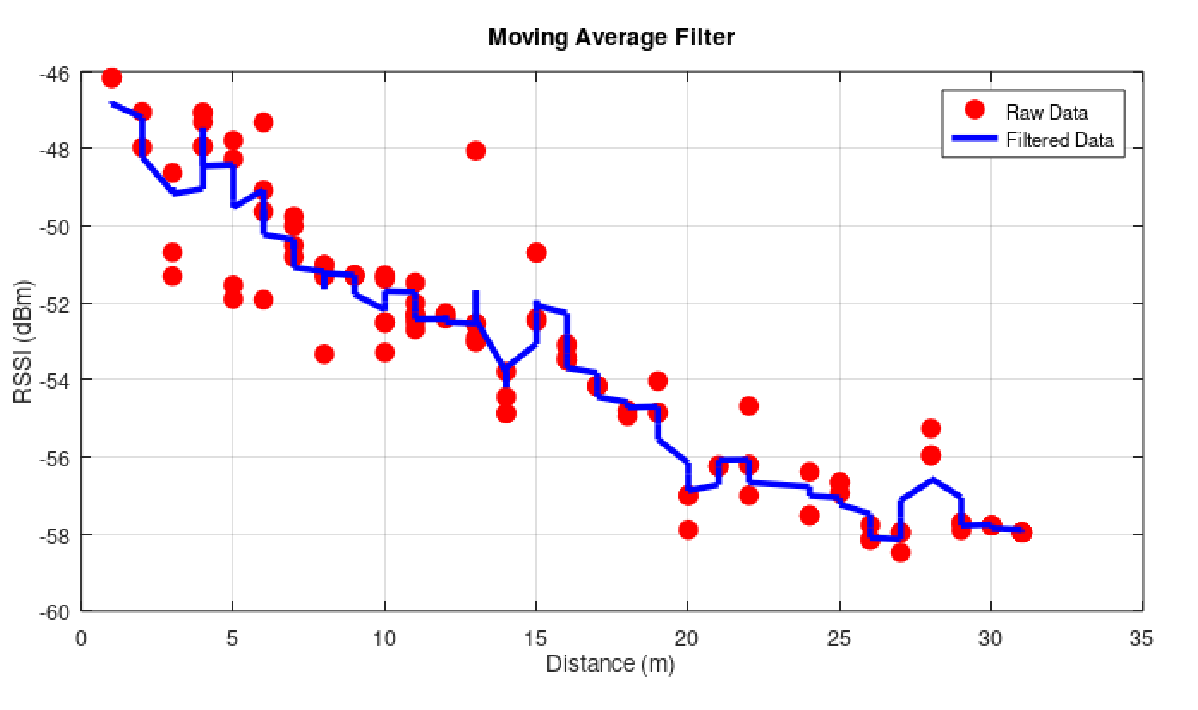
Detected Wi-Fi Chennel

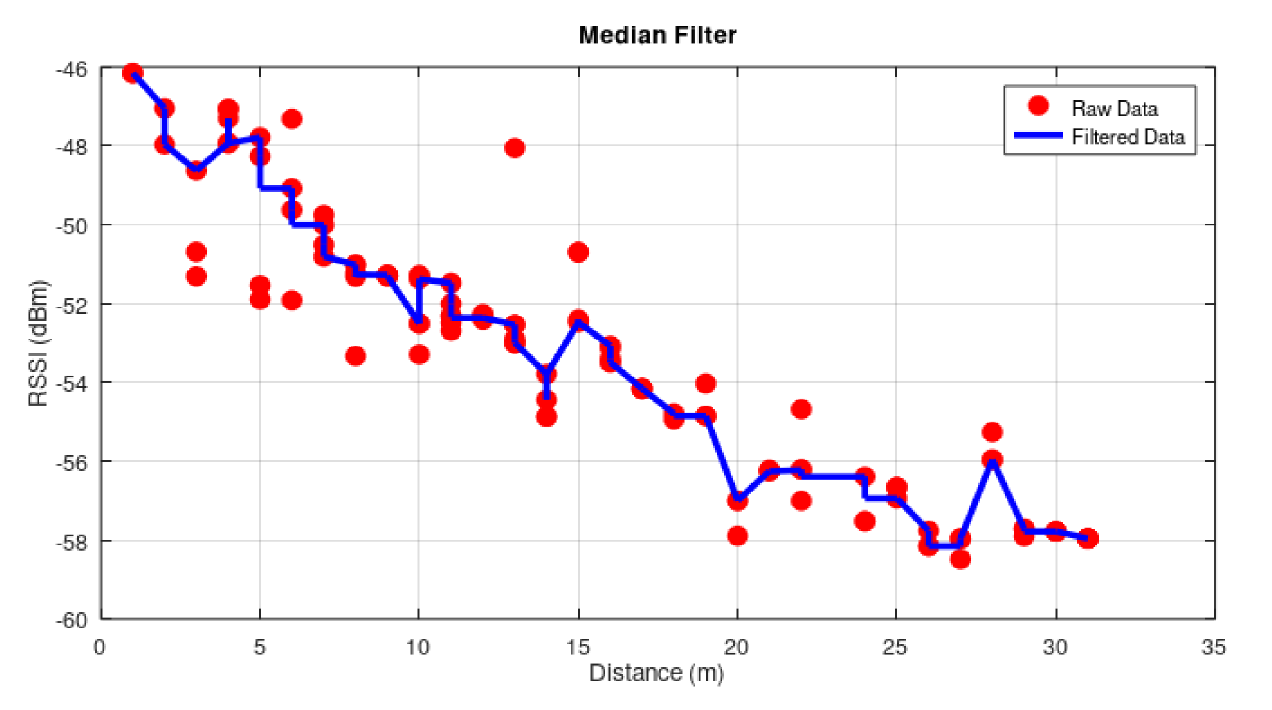
Signal Strength

Timestamp

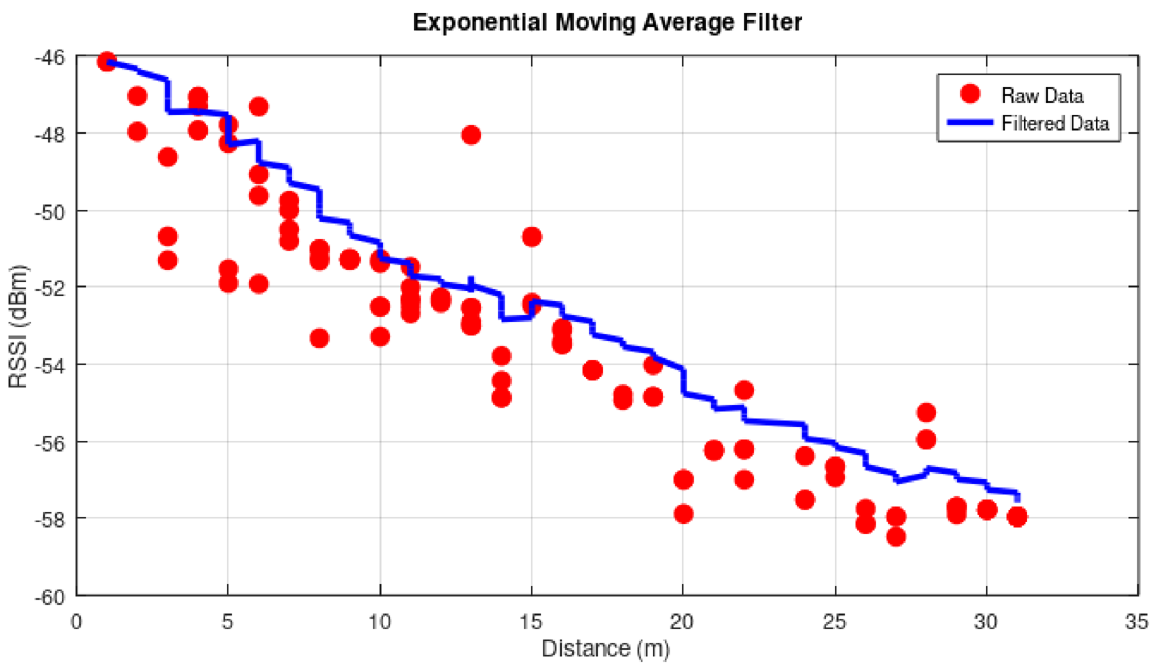
72

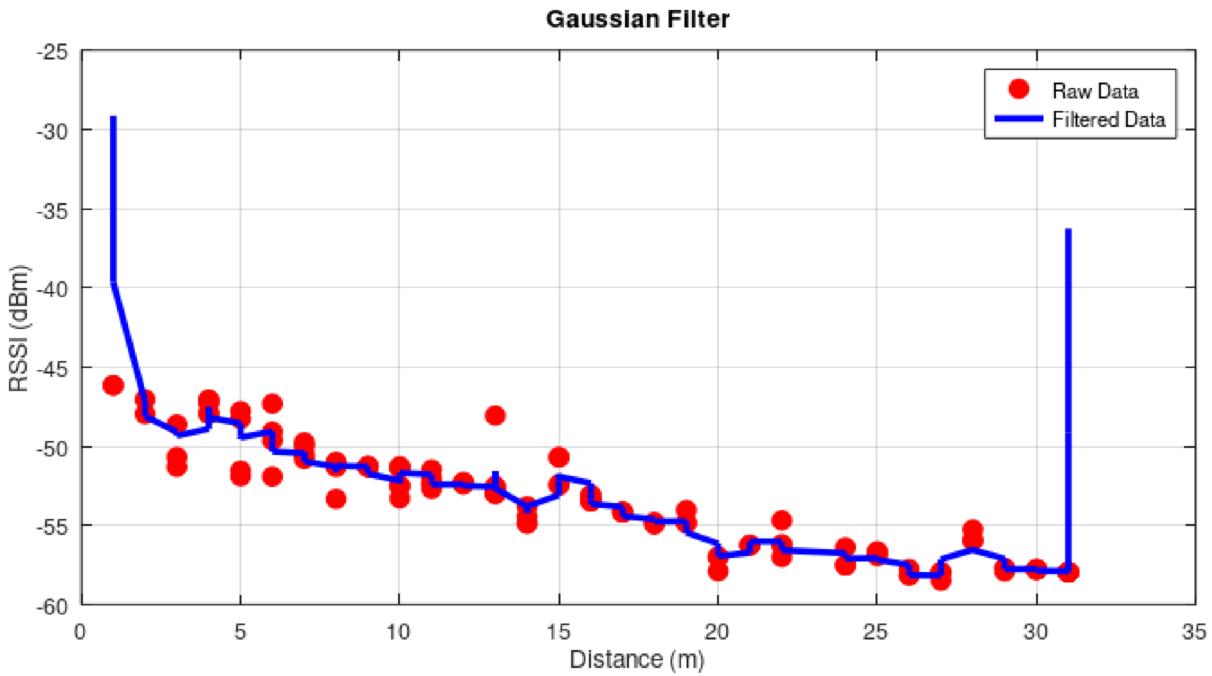
Progress – 2. Data Filtering





Progress – 2. Data Filtering



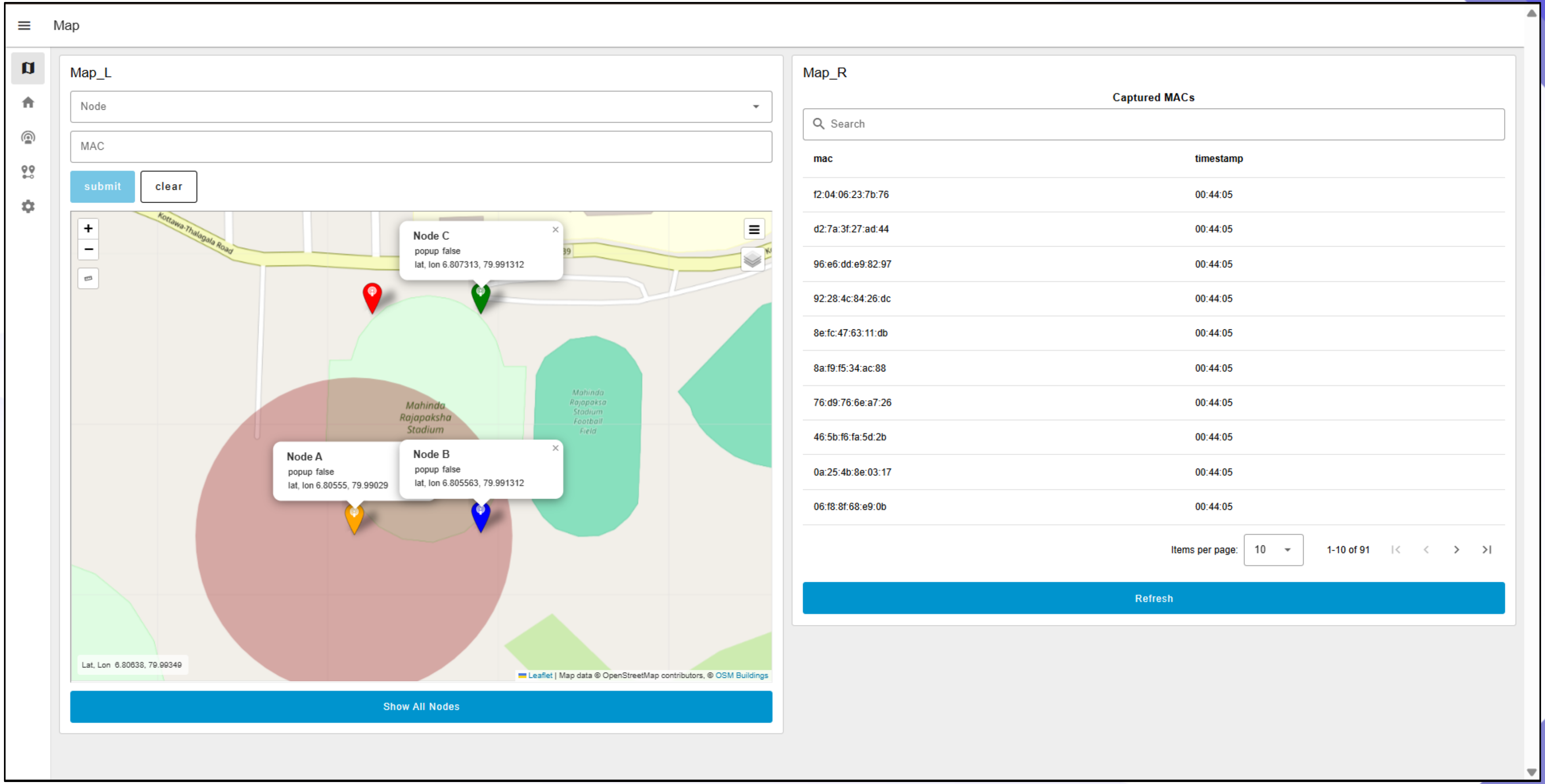


Demonstration - Equation

$$RSSI(d) = P_t - 10nlog10(d) \longrightarrow d = 10^{\frac{RSSI(d) - P_t}{-10n}}$$

$P_t t$ → Reference RSSI
n → path loss exponent
d → Distance
$RSSI(d)$ → Received signal strength

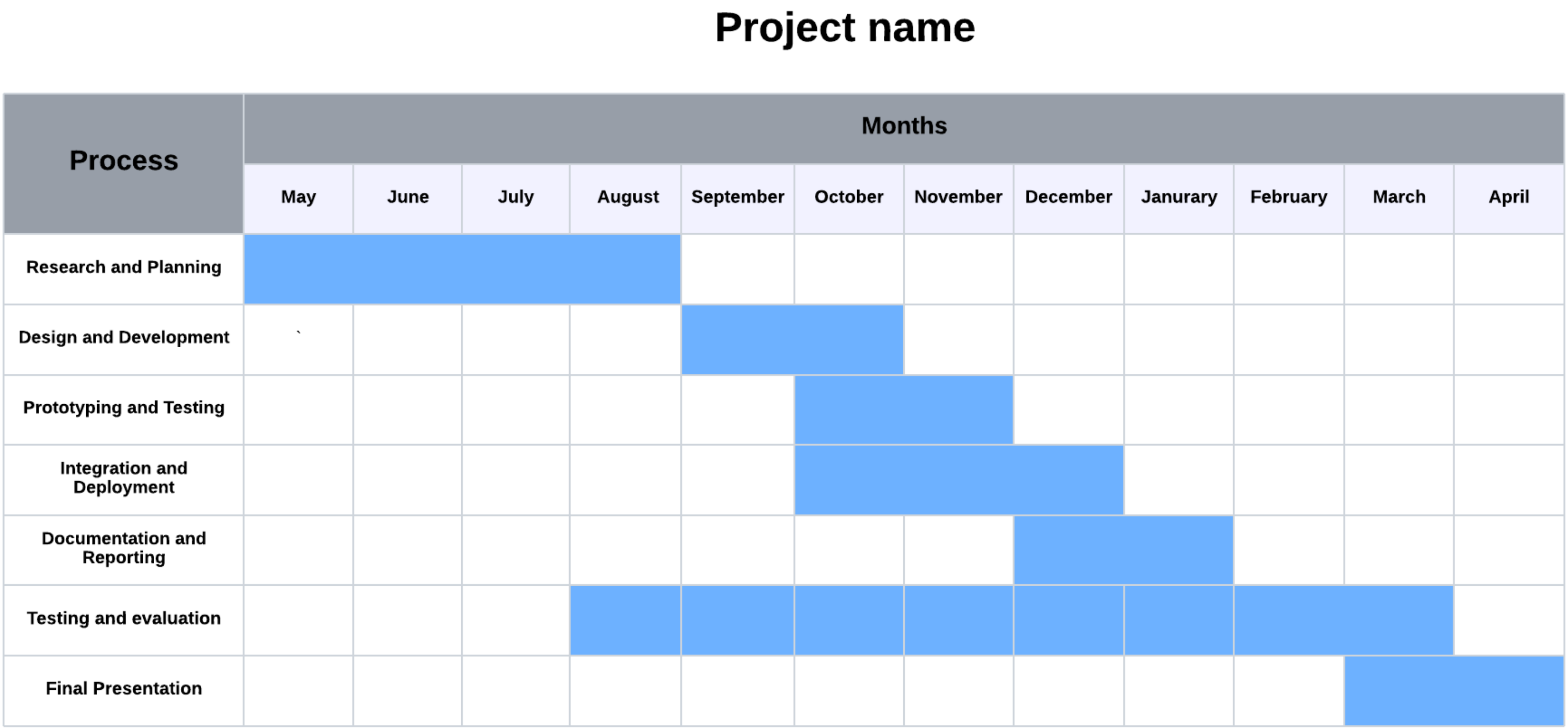
Demonstration - Dashboard



Progress – Hardware Implementation

- Hardware Procurement
  - Antennas
    - Wi-Fi
    - Lo-Ra
  - ESP32 Modules
    - Lo-Ra
    - Detection Nodes
  - Batteries
- Enclosure
  - Design
  - Print
- Batter & power supply configurations - In progress
- LoRa Communication Module & Detection Module Integration - Pending

Gannt Chart



Project name
Process
Months
May
June
July
August
September
October
November
December
Janurary
February
March
April
Research and Planning
Design and Development
Prototyping and Testing
Integration and Deployment
Documentation and Reporting
Testing and evaluation
Final Presentation

Thank You !