Cuevas Kyllian
Mirbey Thomas

5G Antenna Configuration Web Application

Introduction

The 5G Antenna Configuration Web Application is designed to assist users in selecting optimal antenna configurations based on location-specific parameters. Users can pick a location on an interactive map, and the application provides recommendations on frequency, subcarrier size, cyclic prefix mode, output power, and coverage radius.

Key Features

- Interactive Map Selection: Users can click on a map to choose a location.

- Dynamic Configuration Recommendations: The system calculates optimal antenna settings based on terrain, building density, and elevation.

- Integration with External Data: Uses location-based factors such as road speed and elevation.

- Result Visualization: Displays coverage area and configurations on an interactive map.

Technologies Used

- HTML, CSS, JavaScript for frontend development

- Leaflet.js for interactive mapping

- Flask (Python) for backend logic

- OpenStreetMap for map tiles

- FontAwesome for icons and UI enhancement

Implementation Details

1. index.html (User Input Page)

This page provides an interactive map where users select a location. The coordinates (latitude and longitude) are captured and submitted for analysis.

2. result.html (Configuration Display)

Displays calculated 5G antenna configurations with additional geographical information. Users see a highlighted coverage area on the map.

3. Backend Processing

A Flask server processes location data, retrieves geographical attributes, and determines optimal 5G antenna settings based on predefined logic.

4. Configuration logic

If mountains are present:

- The first condition checks if there are mountains. If mountains evaluates to True, it returns a configuration that is suitable for mountainous areas, with a high output power, 1.5 km coverage, and a frequency of 3.5 GHz.

If average speed is greater than 80:

- If mountains are not present, the next condition checks if the average speed (avg_speed) is greater than 80 (km/h). If true, it returns a configuration with a lower frequency of 700 MHz, wider coverage radius of 10 km, and a high output power.

If building density is greater than 500:

- If neither of the previous conditions is met, it checks if the building density (building_density) is greater than 500. If so, it assumes that the area is heavily built-up and returns a configuration suited for urban areas with high density. This includes a high-frequency band (26 GHz), small coverage (0.5 km), and medium output power.

If building density is greater than 100:

- If the building density is not greater than 500 but greater than 100, it returns a network configuration with medium output power, normal cyclic prefix mode, and a coverage radius of 2 km, using 3.5 GHz frequency.

If none of the conditions are met (default case):

- If none of the previous conditions are met, the code defaults to a configuration suitable for a wider coverage area, lower output power, and a frequency of 700 MHz.

Requirements

```
pip install flask requests jinja2
```

Note

This setup may not work as intended on Windows due firewall configuration (port not opening). If you encounter problems, try running it on a Linux system.