IMPLEMENTATION PLAN

### ****1. Data Preparation****

Begin by collecting a clean dataset that links public IP addresses to their known cities. Make sure the data covers a wide range of regions and providers so the model can generalize well. Split your dataset by time and by network groupings to prevent the model from memorizing rather than learning — this ensures it can handle new IPs that appear later.

### ****2. Feature Engineering****

From each IP address, extract useful patterns that can help identify its city. This can include the IP prefix or subnet, simple numerical encodings, and possibly lookup-based features such as the first few bytes (which often correlate with region). You can also include basic metadata like timezone. Keep the feature set clean, consistent, and easily reproducible.

### ****3. Model Training****

Train a supervised machine learning model to predict the most likely city for each IP. Calibrate its output probabilities so that the confidence scores make sense. Optionally, train a small regression model to estimate how far off (in kilometers) a prediction might be, giving you a confidence radius for each result.

### ****4. Evaluation****

Test your model on a separate set of IPs that it has never seen before. Evaluate its accuracy using top-1 and top-3 city prediction metrics, and measure geographic error (for example, the median and 90th percentile distance between the predicted city and the true city). Check calibration using metrics like the Brier score, and analyze where the model tends to make mistakes — such as confusing nearby cities.

### ****5. Serving and Monitoring****

Wrap the trained model in a simple API that allows you to query predictions like /predict?ip=1.2.3.4. The API should return a clear output with the predicted city, probability, latitude, longitude, and estimated confidence radius in kilometers. Log predictions for continuous improvement, and set a clear rule to mark uncertain results as “low confidence” rather than making overconfident guesses. Retrain the model periodically to keep it up-to-date as IP address allocations and usage patterns evolve.