**Standard Operating Procedure (SOP) – Geo Routing Program Design**

**1. Purpose**

This SOP outlines the design and development process for the **Geo Routing Program**, which processes **area codes and branch locations** to generate **Dialpad-compatible geo-routing rules** and provides **interactive mapping for coverage visualization**.

**2. System Overview**

The Geo Routing Program is a **browser-based interface** that processes routing rules by:

* Uploading and managing **area code and branch location data**.
* Assigning **geo-routed area codes** to branches based on **city hit count**.
* Generating **Dialpad-compatible routing rules**.
* Providing **interactive mapping of branch coverage areas**.

**3. System Design & Workflow**

**3.1 Data Processing & Storage**

**3.1.1 CSV File Inputs and Hard-Coded Data**

* **US Area Codes Data (Hard-Coded)**
* **Format: Area Code, City, State, Country, Latitude, Longitude**
* **Contains 2,766 area codes.**
* **This data is hard-coded into the system as it remains static and does not change frequently.**
* **System displays a processing status to confirm record count upon initialization.**
* **Branch List CSV (User-Uploaded)**
* **Format: Branch Name, Branch Code, Address, City, State, Country, Latitude, Longitude**
* **System displays the number of branches detected upon upload.**

**3.1.2 Data Storage & Management**

* **Branch data is stored locally with timestamps.**
* **Users can retrieve older branch data files.**
* **Users can clear all stored branch data to reset the system.**
* **Hard-coded US area codes data remains constant and unaffected by user actions.**

**3.2 User Interface – Tab Navigation**

**Tab 1: Data Upload**

* Allows uploading of **Area Code CSV** and **Branch List CSV**.
* Displays **processing status updates**.
* Includes **data management controls** (clear/reset stored data).

**Tab 2: Branch Management**

* Displays **branches** in a **searchable and filterable table**.
* Users can **categorize branches** into:
  1. **General Rental**
  2. **Trench Rental**
  3. **Industrial Rental**
  4. **Other**
  5. **Inactive**
* Allows **merging of multiple branches into Metro regions**.
* Metro regions are **editable and deletable**.

**Tab 3: Routing Rule Generation**

* Users **select a category** to generate **Dialpad-compatible routing rules**.
* Each **of the 317 geo-routed area codes** is assigned to **one branch only**.
* **Duplicate area codes** are assigned to the **branch with the highest city match count**.
* Output follows the **Dialpad routing table format**:
  + **Inbound Codes to Route**: +1205, +1256, +1678, +1706, etc.
  + **Destination Branch**: SSE-Atlanta Metro
* Displays a **count of area codes assigned per branch**.

**Tab 4: Mapping & Visualization**

* **Interactive Map** displays branches by:
  + **Category (General, Trench, Industrial, Other, Inactive)**
  + **Metro regions**
  + **Standalone branches**
* Selecting **Metro or Standalone branches** **highlights**:
  + **Assigned area codes**
  + **Geographic coverage** using a **color-coded overlay or outlined regions**.
* Users can **select multiple branches** via a **dropdown or checkbox selector**.
* The **map dynamically updates** as selections change.

**4. Technical Specifications**

| **Component** | **Technology** |
| --- | --- |
| Backend | Flask / FastAPI |
| Frontend | React.js |
| Storage | IndexedDB / SQLite / JSON |
| Mapping | Leaflet.js / Google Maps API |
| CSV Handling | Pandas |
| Real-time Updates | WebSockets / AJAX |

**5. Implementation Steps**

1. **Develop Backend**
   * Implement CSV processing logic.
   * Build a local database for storing processed data.
2. **Develop Frontend**
   * Create tab-based navigation UI.
   * Implement upload and processing feedback features.
3. **Generate Routing Rules**
   * Develop logic to **assign area codes to branches**.
   * Ensure **Dialpad-compatible output format**.
4. **Implement Mapping Features**
   * Integrate **Leaflet.js / Google Maps API**.
   * Develop **interactive selection for branches & metro areas**.
   * Implement **dynamic highlighting of coverage areas**.
5. **Testing & Deployment**
   * Validate **data parsing and rule generation accuracy**.
   * Test **map visualization responsiveness**.
   * Deploy system in a **browser-accessible format**.

**6. Version Control & Maintenance**

* Maintain **version history** for updates.
* Allow **reloading of previous data versions**.
* Provide **logs for processed data**.

**7. Summary**

This SOP documents the design and implementation process for the **Geo Routing Program**. It ensures structured development, proper functionality, and seamless deployment. This document will guide **developers, engineers, and stakeholders** involved in the project lifecycle.

This SOP is **ready for documentation and process tracking**. Let me know if you'd like any additions!