Nearest Centers API Documentation

1 Overview

The Nearest Centers API is a web service designed to find the closest centers (within 15 km) to a user-provided address in India. It leverages geocoding, database queries, distance calculations, and waiting time estimates to deliver accurate results. The API is built using ASP.NET Core, integrates with a PostgreSQL database, uses the Nominatim geocoding service from OpenStreetMap, and includes a waiting time API for non-center facilities.

2 Purpose

The API enables users to input an address and receive a JSON response containing up to two centers nearest to the provided location, including their names, addresses, distances in kilometers, and estimated travel and waiting times. It is designed for applications requiring location-based services, such as finding nearby facilities or service points.

3 Key Features

- **Geocoding**: Converts user-provided addresses into geographic coordinates (latitude and longitude) using the Nominatim API.
- **Distance Calculation**: Computes the distance between the user's coordinates and center coordinates using the Haversine formula.
- **Waiting Time Estimation**: Provides fixed 10-minute waiting time for centers and dynamic waiting times for non-centers based on patient count (60 minutes per patient) from a dedicated API.
- **Database Integration**: Retrieves center data (SiteId, name, address, coordinates, center status) from a PostgreSQL database.
- **Rate Limiting**: Enforces a 1-second delay between Nominatim API requests to comply with usage policies.

- **Error Handling**: Provides meaningful error messages for invalid addresses, geocoding failures, database issues, or API failures.
- **Logging**: Implements detailed logging for debugging, monitoring, and error tracking.

4 Architecture

The API follows a modular design with the following components:

- **Controller**: Handles HTTP POST requests, validates input, and returns JSON responses. Includes a dedicated waiting time controller for non-center facilities.
- **Service Layer**: Contains logic for geocoding, database queries, distance calculations, and waiting time retrieval via API calls.
- **Data Models**: Defines structures for geographic coordinates, centers (including SiteId), waiting time responses, and API requests/responses.

Dependencies:

- **ASP.NET Core**: For building the web API.
- **Npgsql**: For PostgreSQL database connectivity.
- **System.Net.Http**: For making requests to the Nominatim and waiting time APIs.
- **System.Text.Json**: For JSON serialization/deserialization.
- **Microsoft.Extensions.Logging**: For logging application events and errors.

5 Approach to Geocoding

The API uses a simplified geocoding strategy:

- Extracts a 6-digit pincode from the user-provided address using regex.
- Queries the Nominatim API with the pincode (or full address if no pincode) to obtain coordinates, limiting requests to India (countrycodes=in).
- Applies rate limiting to ensure compliance with Nominatim's usage policy.

6 API Endpoints

6.1 Find Nearest Centers

- **Endpoint**: `POST /api/centers`
- **Request Body**: JSON object with a single property `Address` (string), e.g., `{"Address": "123 Anna Salai, Chennai, 600002"}`
- **Response**: JSON object containing two lists: `centersDistance` (nearest centers by distance) and `centersTime` (nearest centers by total travel and waiting time).

6.2 Waiting Time

- **Endpoint**: `POST /api/centers/wait`
- **Request Body**: JSON object with `siteId` (integer), e.g., `{"siteId": 123}`. Note: `siteId` is included for future compatibility but not currently used.
- **Response**: JSON object with patient count (`totalOP`) and timestamp, e.g., `{"dataValues": [{"totalOP": 5, "UpdatedTime": "12:35 PM"}]}`. Waiting time is calculated as 60 minutes per patient for non-centers.

7 Dependencies

7.1 External Services

- **Nominatim (OpenStreetMap)**: For geocoding addresses to coordinates. Configured to limit requests to India (countrycodes=in).

7.2 NuGet Packages

- **Microsoft.AspNetCore.Mvc**: For API framework.
- **Npgsql**: For PostgreSQL database access.
- **System.Text.Json**: For JSON serialization/deserialization.
- **Microsoft.Extensions.Logging**: For logging.

7.3 Configuration

- **Database**: Requires a PostgreSQL database with a `Centers` table containing `SiteId`, `SiteName`, `SiteLocation`, `Latitude`, `Longitude`, and `IsCenter`.

- **Connection String**: Provide a valid `DefaultConnection` string in the application configuration (e.g., `appsettings.json`).
- **Nominatim Contact**: Update the `User-Agent` header in `NearestCentersService.cs` with a valid email address (e.g., `support@nearestcentersapi.com`).

8 Limitations

- **Nominatim Dependency**: Relies on the availability and accuracy of the Nominatim API.
- **Database Dependency**: Assumes valid coordinate data and `SiteId` in the database; missing or incorrect data may lead to empty results.
- **Waiting Time**: Non-center waiting times depend on the `/api/centers/wait` endpoint; defaults to 60 minutes if the API fails.

9 Conclusion

The Nearest Centers API provides a reliable and efficient solution for finding nearby centers based on a user's address. Its simplified geocoding strategy, Haversine distance calculations, waiting time integration, and robust error handling make it suitable for location-based applications in India. With proper configuration and maintenance, it can serve as a scalable component in larger systems.