Smart City Synthetic Data Generator 🏙️

A comprehensive web application for generating realistic synthetic data for smart city and governance applications. This tool helps urban planners, researchers, and developers create test data for various smart city scenarios.

Features

Data Types

- **Geographic/Location Data**: Generate geo-coded points with metadata including addresses, building types, elevation, and population density
- **Traffic & Transportation**: Create traffic flow data with vehicle details, congestion levels, and incident reports
- **Social/Demographic Data**: Generate population data with demographics, employment, education, and social indicators
- **Financial/Economic Data**: Create transaction records and budget allocation data
- **Climate/Environmental Data**: Generate weather conditions, air quality measurements, and environmental metrics
- **IoT Sensor Data**: Simulate various sensor types (parking, waste, lighting, water, energy, noise)
- **Public Transport Data**: Generate real-time public transport vehicle positions and status
- **Emergency Services Data**: Create incident reports with response times and severity levels

Key Features

- **Real-time Streaming**: Support for both static generation and real-time data streams via WebSocket
- **Interactive Visualizations**:
  - Map view with Leaflet for geographic data
  - Charts with Chart.js for statistical analysis
  - Table view for detailed data inspection
  - Raw JSON view for developers
- **Export Options**: Export data in JSON, CSV, or Excel formats
- **Customizable Parameters**: Configure data generation with specific parameters for each data type
- **Statistics Dashboard**: Real-time statistics showing record count, data size, and stream status

Installation

1. Clone the repository:

```
git clone <repository-url>
cd Sentetic_Data_WebApp_2025
```

2. Install dependencies:

```
npm install
```

3. Start the server:

```
npm start
```

For development with auto-reload:

```
npm run dev
```

4. Open your browser and navigate to:

```
http://localhost:3000
```

Usage

Basic Data Generation

1. **Select Data Type**: Choose from the dropdown menu (Geographic, Traffic, Social, etc.)
2. **Configure Options**: Set specific parameters for your selected data type
3. **Set Record Count**: Specify how many records to generate (1-10,000)
4. **Choose Mode**:
   - **Static**: Generate data once
   - **Real-time**: Stream data continuously
5. **Click Generate**: Start the data generation process

Visualization Options

- **Map View**: See geographic data plotted on an interactive map
- **Charts**: Analyze data distribution and patterns
- **Table View**: Inspect individual records in a tabular format
- **Raw Data**: View the raw JSON structure

Exporting Data

Click on the export buttons to download your generated data:

- **JSON**: Full data structure with nested objects
- **CSV**: Flattened data suitable for spreadsheets
- **Excel**: Formatted Excel file with data

API Endpoints

REST API

- `GET /api/health` - Check server health status
- `POST /api/generate` - Generate synthetic data

```
{
  "dataType": "geo|traffic|social|financial|climate|iot|transport|emergency",
  "count": 100,
  "options": {}
}
```

WebSocket API

Connect to `ws://localhost:3000` for real-time streaming:

```
// Start stream
{
  "action": "start_stream",
  "dataType": "geo",
  "interval": 1000,
  "options": {}
}

// Stop stream
{
  "action": "stop_stream"
}
```

Data Schema Examples

Geographic Data

```
{
  "id": "uuid",
  "lat": 52.5200,
  "lng": 13.4050,
  "timestamp": "2025-01-14T18:00:00.000Z",
  "type": "residential",
  "metadata": {
    "address": "123 Main St",
    "city": "Berlin",
    "elevation": 75,
    "population_density": 5000
  }
}
```

Traffic Data

```
{
  "id": "uuid",
  "segment_id": "SEG-123",
  "coordinates": {
    "start": { "lat": 52.52, "lng": 13.40 },
    "end": { "lat": 52.53, "lng": 13.41 }
  },
  "current_speed": 45,
  "congestion_level": 0.3,
  "vehicle_count": 25
}
```

Technology Stack

- **Backend**: Node.js, Express.js
- **Real-time**: WebSocket (ws)
- **Data Generation**: Faker.js
- **Frontend**: Vanilla JavaScript, HTML5, CSS3
- **Visualization**: Leaflet (maps), Chart.js (charts)
- **Export**: PapaParse (CSV), SheetJS (Excel)

Configuration

The application uses default coordinates for Berlin (52.5200, 13.4050). You can customize these in the dynamic options for each data type.

Environment Variables

- `PORT`: Server port (default: 3000)

Use Cases

- **Urban Planning**: Test smart city applications with realistic data
- **Research**: Generate datasets for academic research
- **Development**: Create test data for application development
- **Training**: Use for workshops and educational purposes
- **Simulation**: Test system behavior with various data scenarios
- **Prototyping**: Quickly generate data for proof-of-concept projects

Contributing

Feel free to submit issues, fork the repository, and create pull requests for any improvements.

License

MIT License

Support

For issues or questions, please create an issue in the repository.

Built with ❤️ for Smart Cities and Digital Governance