Bengaluru GIS Web Application – Report

1. Introduction

The objective of this project was to design and implement a GIS-based web application to visualize key urban datasets of Bengaluru city. The application integrates ward boundaries, schools, tree census data, and a digital elevation model (DEM), providing both spatial and statistical insights through interactive maps and charts.

2. Approach

The workflow followed three major phases:

(a) Data Collection & Preprocessing

- Collected datasets: wards (GeoJSON), schools (Overpass API → GeoJSON), tree census (CSV/GeoJSON), DEM (raster).
- Standardized datasets using **GeoPandas** (coordinate systems, attribute fields).
- Generated a derived dataset ward_tree_counts.csv summarizing tree species counts per ward for visualization.

(b) Backend Development (Flask)

- Developed API endpoints to serve GeoJSON/CSV data:
    - /api/wards
    - /api/schools
    - /api/trees
    - /api/tree_counts
- Preprocessing script (preprocess_data.py) automated conversion and aggregation.

(c) Frontend Development (Leaflet.js)

- Built interactive map with multiple layers (OSM basemap, wards, schools, trees, DEM).
- Implemented **pie chart popups** per ward using Chart.js + Leaflet integration.
- Added toggles for layer visibility.

3. Challenges & Solutions

- **Data inconsistency**: Ward IDs in tree census did not match ward GeoJSON. → Normalized IDs using string cleaning.

- **Large tree dataset**: Initial load slowed the map. → Aggregated counts (ward_tree_counts.csv) and displayed summary instead of plotting every tree.
- **DEM visualization**: Simple raster overlay was too heavy. → Optimized by reducing resolution and loading via Leaflet raster plugin.
- **API errors (404 / FileNotFoundError)**: Fixed by ensuring preprocessing script generated required CSV before Flask ran.

4. Potential Improvements

- Host the app on **cloud platforms** (Heroku/AWS).
- Add **user filters** (tree type, school type).
- Enable **analysis tools** (buffer, nearest school, ward statistics).
- Improve **DEM rendering** with hillshade/contour maps.
- Implement **progressive loading** for large datasets.

5. Conclusion

The project successfully delivered a fully functional GIS web application for Bengaluru. It demonstrates integration of **geospatial data processing (Python, GeoPandas)**, **backend service development (Flask)**, and **frontend visualization (Leaflet.js, Chart.js)**. The system provides a scalable foundation for future urban analytics and geospatial decision-making.