Are U Query-ous? A Web-Based Platform for Democratizing Open Geospatial Data Access

From Queries to Maps, A New Way to See the World!

Universitat Oberta de Catalunya

Nicolas D'Alessandro Calderon

Bachelor's degree in Techniques for Software Application Development

Localization Based Systems and Intelligent Spaces

Project supervisor

Joaquín Torres Sospedra

Coordinating professor

Antoni Perez-Navarro

Date of submission

May 06, 2025

SUMMARY OF THE FINAL PROJECT

Title of the project:	*Are U Query-ous? A Web-Based Platform for Democratizing Open Geospatial Data Access*
Author name:	*Nicolas D'Alessandro Calderon*
Project supervisor:	*Joaquín Torres Sospedra*
Coordinating professor:	*Antoni Perez-Navarro*
Date of submission (MM/YYYY):	*04/2025*
Name of the degree:	*Bachelor's degree in Techniques for Software Application Development*
Topic of the final project:	*Localization Based Systems and Intelligent Spaces*
Language:	*English*
Keywords:	*Geospatial Data, Open Data Visualization, Urban Analytics*

Abstract

This study explores the challenge of making open data more accessible to the general public, addressing the gap between the availability of geospatial information and its practical use. Are U Query-ous? is a web-based application designed to enable individuals, regardless of their technical background, to explore and interpret geographic and demographic data intuitively. By integrating interactive maps and user-friendly visualization tools, the platform allows users to analyze regions based on economic activity, population distribution, and local trends.

Through a simplified interface, users can explore and filter publicly available data, identifying patterns relevant to their interests. The application is intended for individuals who are curious about urban development, seeking optimal locations for personal or professional activities, or analyzing demographic trends for research or decision-making. Additionally, the project examines the potential integration of artificial intelligence to facilitate data retrieval through natural language queries, further enhancing accessibility.

The development and results of the analyses in this data scientific report are intended to address all the concepts targeted above, but the author also expects to describe some fundamental principles underlying web apps development and data science.

Index

List of figures

01 INTRODUCTION

03 METHODS AND RESOURCES

List of Tables

01
INTRODUCTION

1.1 Context and motivation

This final project focuses on developing an intuitive **geo-analytics platform** that simplifies the exploration of open data through **interactive maps**. Many open data sources provide valuable insights into **urban planning, demographics, and economic activity**, but non-experts often struggle to extract meaningful information from them.

Are U Query-ous? aims to solve this problem by creating a **user-friendly interface** that enables individuals to explore and analyze spatial data without requiring technical expertise.

Additionally, if time permits, the project will explore the integration of a natural language processing model, such as **those available from Hugging Face**, to enable users to interact with the data using natural language queries. **This feature is considered an enhancement rather than a core requirement and will be evaluated based on project timelines** and feasibility.

At the end of the project, the system will provide a **fully functional prototype** that allows users to:

- **Filter** and **visualize** location-based open data.
- Identify **regional patterns and trends** based on economic and demographic factors.
- Utilize **intelligent search capabilities** to access relevant insights more intuitively.

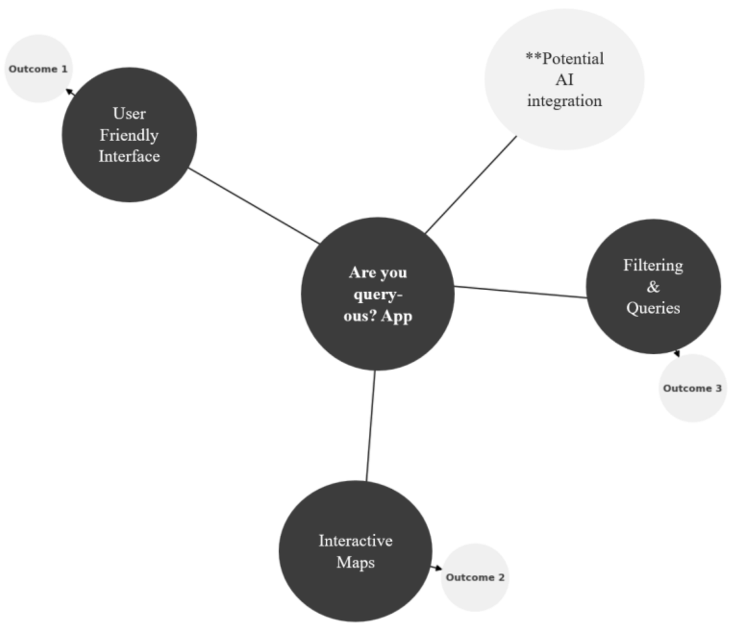

Figure 1 1: Conceptual Representation of Project Summary

Scope

This project focuses on developing a web-based platform that allows users to explore and analyze open geospatial data in an intuitive way. The platform will integrate publicly available datasets from Barcelona and Madrid, two cities with well-structured open data portals that provide reliable and detailed information.

The scope includes the following key aspects:

- **Cities Covered:** The platform will use **open data from Barcelona and Madrid**, ensuring access to urban mobility, economic activity, and demographic datasets for meaningful analysis.
- **Data Integration**: The system will collect and process geospatial, economic, and mobility-related data, allowing users to filter and visualize insights interactively.
- **User Interaction**: The web application will feature an interactive map where users can explore regional trends, compare different areas, and extract useful insights without needing technical expertise.
- **Core Functionalities:**
    - *Mapping & Visualization*: Users will view geospatial data overlaid on maps, with filtering options.
    - *Urban & Economic Insights*: The system will present mobility trends, population density, and economic indicators based on selected areas.
    - *Accessibility & Usability*: The interface will be simple and user-friendly, ensuring that both professionals and non-experts can use it effectively.

By **limiting the project scope to Barcelona and Madrid**, the system will leverage well-organized open datasets while **maintaining a manageable level of complexity within the project timeline**. The structured data availability from these cities will support the development, testing, and validation of the platform, ensuring that it meets its intended objectives.

Rationale

Access to open data has grown exponentially, yet many users struggle to transform this data into actionable insights. While businesses and government entities benefit from sophisticated **geo-analytics tools**, individuals and small organizations often lack the resources or expertise to use these datasets effectively. This project is relevant because it seeks to **bridge this gap**, making open data truly accessible and usable for the **general public, students, researchers, and small businesses**.

Moreover, the relevance of geospatial analysis has expanded in fields such as **smart cities, sustainable urban development, and socio-economic research**. By providing an easy-to-use tool, this project supports the broader goal of promoting **data-driven decision-making at all levels of society**.

Motivation

In the past eight years I have been working and developing my professional career in the field of **data analysis**, so this project aligns with both academic and professional aspirations. The motivation for this project is coming from:

- A personal interest in **making complex data more understandable** for non-experts.
- A desire to provide **access to urban and economic and geomarketing insights** through intuitive visualization for the general public, students, researchers, and small businesses.
- The opportunity **to apply geospatial analytics in a real-world application**.

Additionally, the skills developed through this project, including data processing, backend development, frontend visualization, and user interface design, will be valuable in both academic research and professional settings.

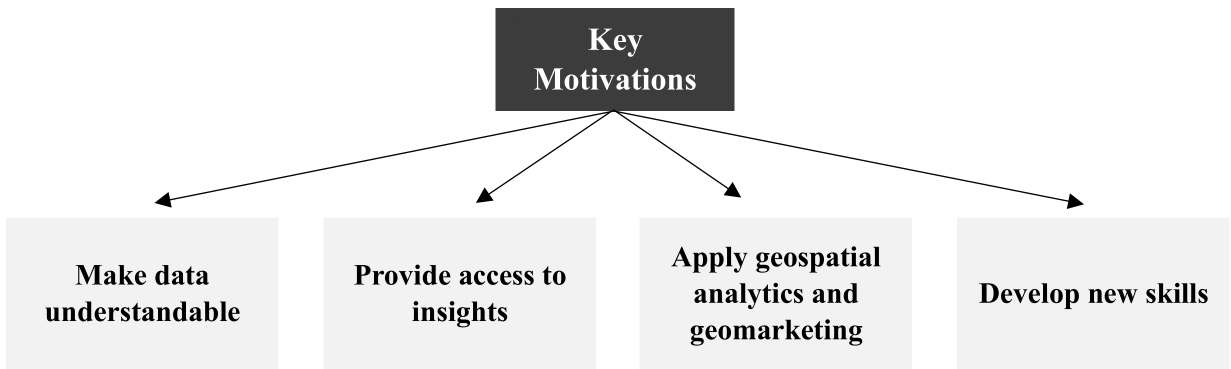

Figure 1 2: Key Motivations for the Project

1.2 Goals

Main Goal

- To **develop a web-based app that enables users to intuitively explore, filter, and analyze open geospatial data**, making location-based intelligence more accessible to a non-technical audience.

Sub-Objectives

- **Develop an interactive mapping system** that allows users to visualize open data in an intuitive and engaging way.
- **Implement filtering and querying functionalities** to help users refine their search and extract relevant insights.
- **Ensure usability and accessibility** by designing a simple and intuitive user interface.
- **If time permits, experiment with integrating an NLP model** to allow natural language queries for filtering and searching data. This is considered a potential enhancement and not a primary project requirement.
- **Validate the effectiveness of the platform** through user feedback and iterative improvements.

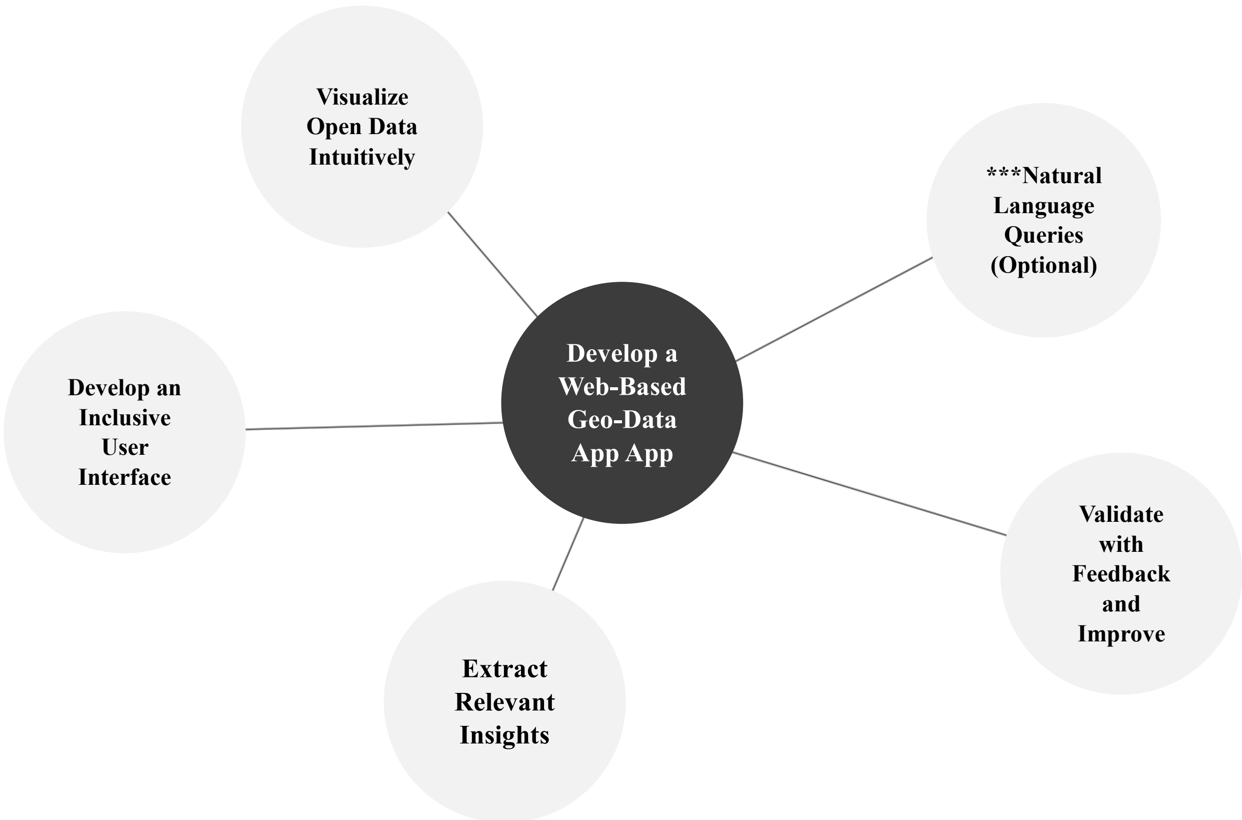

Figure 1.3: Goals Breakdown

1.3 Sustainability, diversity, and ethical/social challenges

Sustainability

This project promotes sustainability by supporting the efficient **use of open data to improve urban and social planning**. By making geospatial information more accessible, individuals and organizations can make informed decisions about resource allocation, mobility, and land use, reducing unnecessary waste and inefficiencies. Additionally, the project aligns with the United Nations Sustainable Development Goals (SDGs), particularly Goal 11 (Sustainable Cities and Communities), by providing insights that encourage responsible urban development. The project has a minimal ecological footprint as it relies on existing digital infrastructure, avoiding additional resource consumption.

Ethical behaviour and social responsibility

The project considers ethical principles by **ensuring data privacy and security**. Since the system processes **publicly available open data**, it does not involve personal or sensitive information. However, the potential risks of misinterpreting data or using insights unethically are acknowledged. To mitigate this, the platform will provide **transparent data sources and disclaimers to ensure users understand the limitations of the information**.

Diversity, gender and human rights

The project is designed to be inclusive and accessible, allowing all individuals, regardless of background, gender, or technical expertise, to explore geospatial data. By offering a simple and user-friendly interface, it ensures that users with limited data experience can engage with geographic insights. Furthermore, **accessibility features will be considered**. The project aligns with the principle of equal access to information, promoting diversity and reducing barriers to data-driven knowledge.

1.4 Approach and Methodology

This project follows a structured development process **based on agile principles**, allowing for **iterative testing and feedback** throughout the semester. The key steps include:

Data Collection and Processing

- **Identify and integrate open datasets** (demographic, economic, urban mobility, etc.).
- **Clean and preprocess data** to ensure usability in the application.

Backend Development

- Build a **RESTful API** to serve geospatial data.
- **Store data in a database** optimized for efficient queries.

Frontend Development

- **Design a responsive user interface** with map-based interaction.
- **Implement data visualization tools** for filtering and exploration.

Testing and Refinement

- **Gather feedback** to enhance usability and features.

1.5 Schedule

A detailed **schedule with milestones** has been defined, ensuring that the development process is structured, manageable, and aligned with the semester timeline following the structure of the phases defined in each CAT:

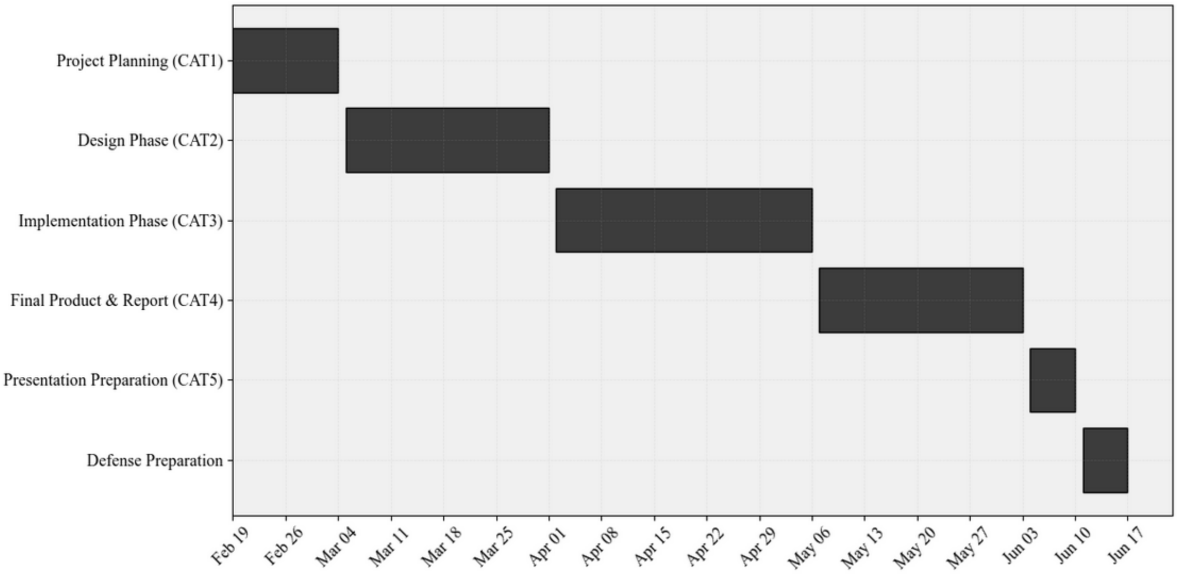

Figure 1 4: Process Diagram

CAT1: Project Planning Phase (Feb 19 - Mar 04)

Task ID	Task Description	Week	Linked
1.1	- **Define project scope, research open datasets, and identify visualization requirements.** - **Define technology stack, assess integration challenges, and create a detailed timeline.**	Feb 19 - Feb 23	-
1.3	**Document sustainability considerations, set up development environment, and compile CAT1 documentation.**	Feb 24 - Mar 02	1.1
1.4	**Review and finalize CAT1 documentation.**	Mar 02 - Mar 03	1.1, 1.2

Table 1.1: Project Planning Phase

CAT2: Design Phase (Mar 05 - Apr 01)

Task ID	Task Description	Week	Linked
2.1	**Define user stories, database schema, and UI wireframes for the mapping interface.**	Mar 03 - Mar 09	1.4
2.2	**Design system architecture, research filtering algorithms, and create frontend class diagrams.**	Mar 10 - Mar 16	2.1
2.3	**Develop database documentation, implement login screen, and document state of the art in geospatial data visualization.**	Mar 17 - Mar 23	2.1
2.4	**Compile and review CAT2 documentation.**	Mar 23 - Mar 31	2.1 - 2.3

Table 1.2: Design Phase

CAT3: Implementation Phase (Apr 02 - May 06)

Task ID	Task Description	Week	Linked
3.1	**Set up backend database and implement data ingestion pipelines for open data.**	Apr 01 - Apr 06	-
3.2	**Develop backend services, authentication, and frontend components for map visualization.**	Apr 07 - Apr 13	3.1
3.3	**Implement filtering, integrate Leaflet.js, and develop data visualization components.**	Apr 14 - Apr 20	3.2
3.4	**Conduct unit testing, perform integration testing, and deploy prototype.**	Apr 21 - Apr 27	3.3, 3.4
3.5	**Compile implementation documentation and finalize CAT3 documentation.**	Apr 27 - May 05	3.1-3.4

Table 1.3: Implementation Phase

CAT4: Final Product & Report (May 07 - Jun 03)

Task ID	Task Description	Week	Linked
4.1	**Refine UI based on testing feedback.**	May 07 - May 11	3.14
4.2	**Implement advanced filtering, economic activity visualization, and optimize database queries.**	May 12 - May 18	4.1
4.3	**Implement (optional) NLP capabilities and conduct comprehensive system testing.**	May 19 - May 25	4.1
4.4	**Fix bugs, optimize performance, and prepare final deployment package. Create user documentation, write results and analysis, draft conclusions, and compile bibliography.**	May 26 - Jun 01	4.1
4.5	**Finalize and format the project report, then submit CAT4.**	Jun 01 - Jun 03	4.1-4.4

Table 1.4: final Product and Report

CAT5: Presentation Preparation (Jun 04 - Jun 10)

Task ID	Task Description	Week	Linked
5.1	- **Create presentation outline, storyboard, and design slides.** - **Prepare demonstration script, record key platform features, and create a narrated presentation video.**	Jun 04 - Jun 08	4.14
5.2	**Review, finalize, and refine presentation based on feedback.**	Jun 08 - Jun 09	5.1

Table 1.5: Presentation Preparation Phase

Defence Preparation (Jun 11 - Jun 17)

Task ID	Task Description	Week	Linked
6.1	- **Prepare defence presentation based on feedback.** - **Anticipate potential questions and prepare responses.** - **Practice presentation delivery**	Jun 11-Jun16	5.2
6.2	**Final defence presentation**	Jun 17	6.1

Table 1.6: Defence Preparation Phase

Ongoing Tasks Throughout the Project

Task ID	Task Description	Week	Linked
7.1	**Gant Planning updates**	Feb 19 - Jun 17	-
7.2	**Weekly supervisor check-ins**	Feb 19 - Jun 17	-
7.3	**Documentation updates**	Feb 19 - Jun 03	-

Table 1.7: Ongoing Tasks Throughout the Project

For a detailed breakdown of the project plan, tasks, and timeline, please refer to the attached **are-you-queryous-planning.xlsx file.**

1.6 Summary of the outputs of the project

By the end of the semester, the project will deliver:

- A functional web application where users can **explore and visualize geospatial data** interactively.
- An intuitive filtering system allowing users **to refine results based on key indicators**.
- A structured API serving **open datasets** with a focus on usability and efficiency.
- A research report detailing **the impact of accessible open data visualization**.

This project contributes to the broader goal of **making open data actionable and meaningful** for a **diverse audience**, reinforcing the importance of **geospatial intelligence in everyday decision-making**.

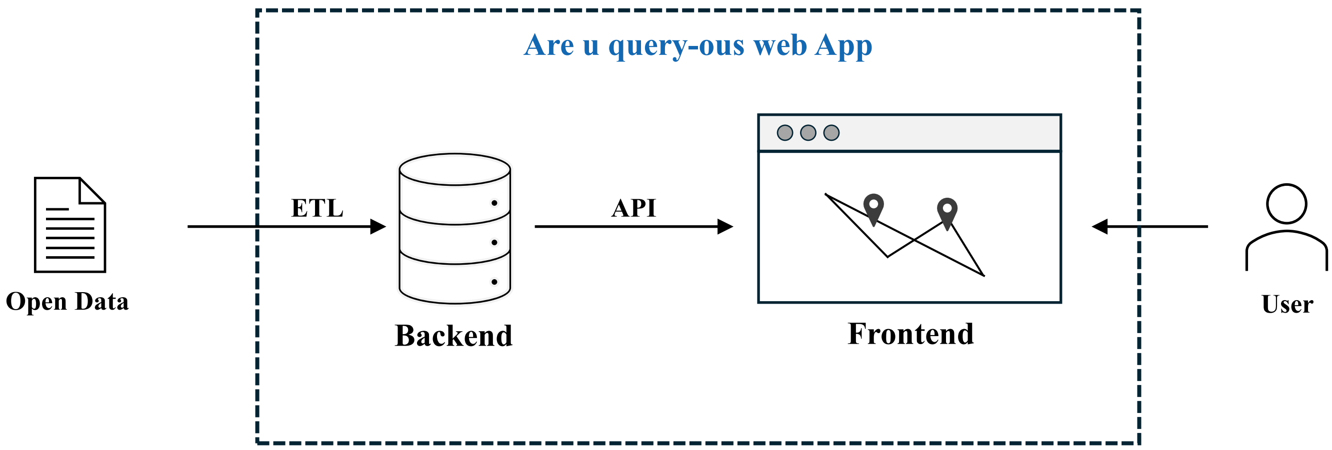

Figure 1 5: Expected Outcome Visual Representation

The project will leverage a modern web technology stack to ensure efficiency, scalability, and a seamless user experience. The core technologies include:

Frontend:

- **React.js** for building a dynamic and interactive user interface.
- **Leaflet.js** for mapping and geospatial visualization.

Backend:

- **FastAPI** to handle API requests and serve processed geospatial data.
- **PostgreSQL/PostGIS** for storing and querying spatial data efficiently.

Data Processing & Integration:

- **Python** and **Pandas** for data preprocessing and transformation.
- **GeoJSON** format to represent geospatial data and serve it dynamically.

Deployment & Hosting:

- **Docker** for containerized development and deployment.
- **GitHub Codespaces** for cloud-based development and collaboration.
- **Vercel** for frontend deployment and **Fly.io** or **Heroku** for backend deployment.

Potential AI Integration (Time-Permitting Feature):

- The project may experiment **with integrating a pre-trained NLP model from Hugging Face** to process natural language queries.
- If implemented, a small widget will allow users to enter queries in plain text (*e.g., "Show me the most populated districts in Madrid").*
- This AI-based functionality is considered an exploratory addition, meaning it will only be developed if time and resources allow.

These technologies will enable the development of a robust and scalable application that can effectively serve users with varying levels of expertise in data analysis and geospatial exploration.

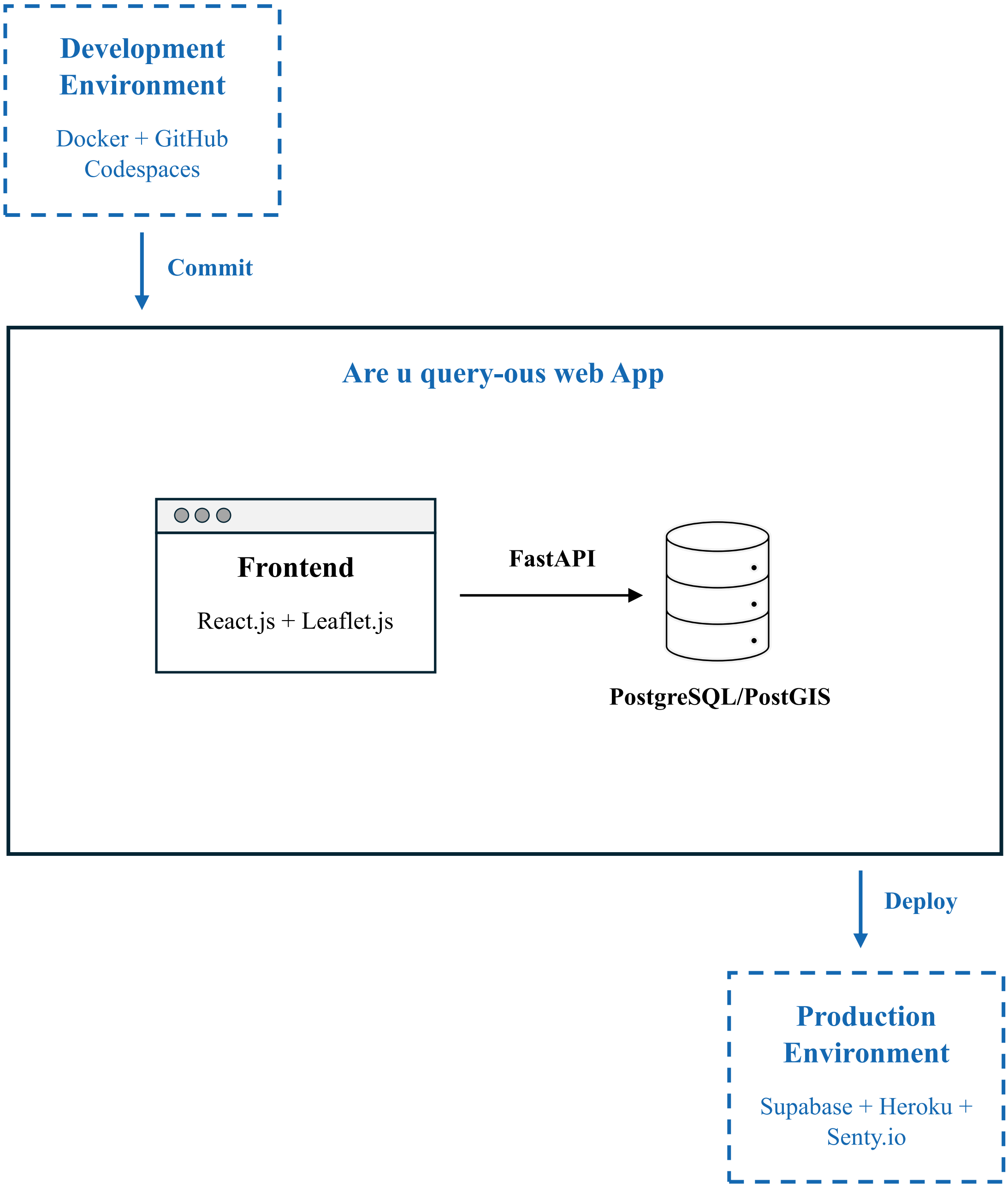

Figure 1 6: Technology Stack Used

1.7 Brief description of the remaining chapters of the report

This section provides a short overview of the chapters that follow in this report:

Chapter 2: State of the Art

This chapter explains what technologies and tools are available already to work with maps and location information. It shows how people are trying to make this kind of information more accessible for any user, not just for professionals. It describes websites and programs that help users to explore and understand maps. We also explain how new technologies like AI are making it easier to ask questions and get responses in the form of maps. Finally, we discuss what problems there still exists and what people are doing to solve them.

Chapter 3: System Design

This chapter introduces the functional architecture of the "Are U Query-ous?" platform, detailing how the system responds to the user's interactions and vice versa. It discusses the primary user stories, brings in the use case diagram, defines the user interface views, and describes the navigation, database, and technical architecture of the web application. This chapter is fundamental to recognizing how the project converts technical concepts into a product that can be used.

Chapter 4: Results

This chapter presents what we accomplished in our project. It shows how the final web application works and how users can explore maps and filter data without technical knowledge. It includes screenshots of the interface and examples of how it makes open data more accessible. This section demonstrates the practical outcomes achieved through our methodology.

Chapter 5: Conclusions and Future Work

The final chapter looks at what we learned from this project and how well we met our goals. It discusses the challenges we faced and what we could improve in the future. It also suggests new features we could add later, like the natural language processing option for easier searching. This chapter evaluates our progress against our original timeline and objectives, and addresses sustainability, diversity, and ethical considerations.

02
STATE OF THE ART

Democratizing Geospatial Data Access: State of the Art

Making geospatial data available to everyone, not just experts, is called "*democratizing geospatial data*." In recent years, new tools, platforms, and AI have made it easier for regular people to use maps and location data (Bhoda, 2023a).

This section looks at the current progress in this field as is intended to cover the state of the art across key areas, including current platforms, visualization technologies, AI integration, challenges, applications, related technologies, and recent research findings.

2.1. Current Geospatial Data Platforms

Modern geospatial platforms help make data available to everyone. Today, we have both free and paid platforms that help people use map data.

A well-known open-source example is **OpenStreetMap (OSM)**, which allows people to create and edit maps (Glasze & Perkins, 2015). This has made mapping more open to the public. Other free tools like **QGIS**, **GRASS GIS**, and **GeoServer** allow users to analyze and visualize spatial data without expensive software (Bhoda, 2023b). These tools support many data formats, making it easier to share information.

On the commercial paid platforms side, **Esri's ArcGIS** is one of the most powerful platforms (Esri, 2023). It was originally a desktop software, but now it also offers cloud-based tools like **ArcGIS Online**. These allow organizations to create and share maps with the public. Another important paid platform is **Google Earth Engine (GEE)**. This popular tool provides satellite images and advanced analysis and also allows researchers to study changes in the environment, such as deforestation, using free satellite data (Gorelick et al., 2017).

There are newer cloud-based web platforms like **CARTO** and **Mapbox** that make it simpler to connect map data with other information and also help developers and businesses to integrate maps into their applications (CARTO, 2024). These platforms provide tools that make geospatial analysis more accessible to non-experts. Finally, **Government and open data APIs** also allow users to access updated maps, weather data, and other statistics (Government of India, 2021).

Thanks to all these platforms, *geospatial data is now available to more people than ever before*.

2.2. Geospatial Data Visualization Technologies

User-friendly tools are key to making map data accessible. Web mapping tools like **Leaflet.js** and **Mapbox GL JS** allow people to create interactive maps in a web browser without advanced skills on *Geographic Information System (GIS)* software (Marten, 2019).

Tools like **Kepler.gl** let users drag and drop data files to create maps instantly. Business tools like **Tableau**, **Microsoft Power BI** and **Amazon QuickSight** now include map features, so business and data analysts can make maps without learning GIS software (Foursquare, 2023). **CesiumJS** is another useful tool that specializes in 3D geospatial visualization, allowing users to create 3D maps of cities and landscapes (Chamberlain et al., 2024).

These visualization tools turn complex data into maps that anyone can understand, hiding the technical details that used to require expert knowledge. Users no longer need advanced training in GIS to work with maps and analyze spatial data.

2.3. AI Integration in Geospatial Analysis

Artificial intelligence (AI) is transforming how we use geospatial data. **GeoAI** refers to the combination of artificial intelligence and geographic data to automate complex tasks (Esri, 2023). AI can quickly analyze satellite images and detect objects like buildings, roads, and forests, reducing the need for human analysis.

One of the most advanced developments is the use of Natural Language Processing (NLP) to allow users to ask questions about maps using everyday language. For example, instead of learning GIS commands, using tools like **ChatGeoAI** or **Chat2Geo** allow users to type questions like "show me areas with high tree loss in 2023" and get maps as answers (Mansourian & Oucheikh, 2024).

Companies like Esri are also developing AI assistants for GIS software. These tools allow users to get quick answers about geospatial data, using Machine learning models that can also predict future trends, such as traffic patterns, climate changes, or urban growth (Lartey, 2024).

As it is happening in other fields, the biggest advantage of AI integration in geospatial analysis is that it reduces the skill barrier. In the past, only experts could perform complex spatial analyses, but now, AI tools make it easier for anyone to work with maps and location data.

2.4. Challenges in Democratizing Geospatial Data

Despite progress, several problems still make map data hard to use for everyone:

- **Cost**: Some advanced data and tools remain expensive (Plunkett, 2019).
- **Complexity**: Working with map data still requires some technical knowledge.
- **Internet access**: Cloud-based tools need good internet connections .
- **Data formats**: Different map data comes in many formats that don't always work together (Hallas et al., 2022).
- **Large file sizes**: Map data files can be very big and hard to share (Gorelick et al., 2017).
- **Awareness**: Many people don't know how useful map data could be for them.

These challenges are being addressed through better education, open data policies, and more user-friendly tools.

2.5. Applications of Geospatial Data

Geospatial data is used in many industries, including:

- **Urban Planning**: Cities use geospatial data to plan infrastructure, transportation, and housing. Maps help identify areas that need better roads, public transport, or green spaces.
- **Transportation & Logistics**: Delivery companies use geospatial data to find the fastest routes. Ride-sharing apps rely on maps to connect drivers and passengers.
- **Environmental Monitoring**: Scientists track deforestation, pollution, and climate change with satellite images and GIS analysis.
- **Disaster Management**: Maps help emergency services respond to floods, earthquakes, and wildfires. Open-source mapping tools like **Ushahidi** allow local communities to share real-time disaster information.
- **Retail & Business**: Companies use geospatial data to find the best store locations, track customer movement, and optimize marketing strategies.
- **Agriculture**: Farmers use satellite images and AI-powered tools to monitor crop health and optimize irrigation.

By making geospatial data available to more people, we can solve real-world problems more effectively.

2.6. Similar Software & Related Technologies

Many technologies help make geospatial data more accessible:

- **Cloud Platforms**: Services like Google Earth Engine and Microsoft Planetary Computer allow users to access and analyze large-scale geospatial data online.
- **Geospatial APIs**: Tools like Google Maps API and Mapbox API make it easy to add maps and location services to applications.
- **Open Data Portals**: Governments and organizations provide free access to maps and location-based data through online portals.
- **Drones & IoT Sensors**: Drones help collect high-resolution aerial imagery, while IoT sensors track environmental changes in real time.

These technologies work helps to bring geospatial analysis to more people, not just specialists.

2.7. Recent Academic Research & Case Studies

In the last five years (2019-2024), researchers have made important progress:

Natural Language Tools (2024)

Researchers created ChatGeoAI, a system that lets people use simple language to make maps and analyze data (Mansourian & Oucheikh, 2024). Users just type what they want, and the computer does the technical work.

Easy Access to Big Data (2020-2022)

CyberGIS projects made it simpler for non-experts to use powerful map tools:

- CyberGIS-Vis created easy-to-use web maps for COVID-19 data (Han et al., 2024).
- CyberGIS-Compute helped regular users run complex map calculations on powerful computers (Michels et al., 2024).

Community Mapping (2019-2023)

Studies on OpenStreetMap showed how regular people can create useful maps (Glasze & Perkins, 2015).:

- Communities developing countries built their own maps instead of relying on external ones
- When people can create maps themselves, it helps improve local government services

Policy Changes in India (2021)

India changed its laws to make map data free for everyone (Government of India, 2021):

- Removed strict rules about who could make and share maps
- Helped local businesses create new map services
- Increased the use of map data in business and farming

Easier Data Sharing (2019-2022)

Researchers worked on making map data easier to find and use:

- A Canadian project created a "Spotify for map data" where users only download what they need
- This helped Indigenous communities and park managers respond to emergencies

AI for Community Help (2022-2024)

Projects used AI to solve social problems:

- The RAMP project used AI and satellite images to help plan healthcare services
- Other studies used AI to spot patterns in citizen-reported issues to improve city planning

Democratizing geospatial data has made great progress, with open-source tools, cloud platforms, and AI making maps and spatial analysis accessible to more people (Bhoda, 2023b). Advances like **GeoAI**, web-based visualization, and natural language processing are lowering technical barriers, allowing non-experts to work with geospatial data more easily (Esri, 2023).

However, challenges remain. Many tools are still expensive, and some require technical skills (Plunkett, 2019). Data formats are not always compatible, making it difficult to share information across platforms (Gorelick et al., 2017). AI models improve geospatial analysis, but they need better transparency to ensure accuracy and trust (Lartey, 2024). In addition, limited internet access in some areas makes it hard to use cloud-based tools (Tula Foundation, 2024).

Future research should focus on standardizing data formats, improving user-friendly AI tools, and reducing dependency on high-cost platforms (Alamri, 2024). By addressing these gaps, geospatial data can become even more accessible, helping businesses, governments, and individuals solve real-world problems more effectively.

2.8. Future Trends in Democratizing Geospatial Data

Looking ahead, these are the trends that will make map data even more accessible to everyone:

- **AI Assistants**: New AI tools will let people talk to maps in normal language. Users will be able to ask questions and get answers without learning special commands.
- **Mobile Apps**: More map tools will work well on phones, helping people in areas with limited computer access.
- **Local Knowledge**: Communities will create their own maps that include cultural information and local knowledge not found in official maps.
- **Mixed Reality**: New technology will combine real-world views with digital map data, making it easier to understand location information.
- **Low-Code Tools**: New software will let people without programming skills create custom map applications.
- **Data Standards**: Better ways to share data between different systems will make it easier to combine information from many sources.

These trends show that geospatial data will become even more useful for everyday decisions. As technology improves and becomes simpler to use, more people will be able to benefit from location-based information in their daily lives, businesses, and communities.

2.9. Conclusions

The review of current geospatial data platforms, visualization tools, and AI integration shows that significant progress has been made in democratizing access to spatial information. Tools like Leaflet.js, OpenStreetMap, and ChatGeoAI provide users to explore maps and datasets without technical expertise. AI is especially promising in reducing barriers through natural language interfaces and predictive modelling. These innovations are complemented by an expanding ecosystem of open data portals and visualization frameworks.

However, challenges remain. Data fragmentation, inconsistent formats, limited awareness, and the need for high internet connectivity are still a barrier for a democratized and general access. Many users also lack the skills or tools to interpret data effectively. Therefore, there is a growing need for platforms that combine simplicity, visual intuition, and smart querying to close this gap. This project contributes to that mission by proposing a low-barrier, map-centric application designed to empower everyday users with meaningful geospatial insights.

References:

- Alamri, S. (2024). *The geospatial crowd: Emerging trends and challenges in crowdsourced spatial analytics*. MDPI Open Access Journals. https://www.mdpi.com/2072-4292/13/23/4745

- BCG Digital Ventures. (2020). *How UP42 is helping institutions combat COVID-19 using geospatial data*. Medium. https://medium.com/bcg-digital-ventures/how-up42-is-helping-institutions-combat-covid-19-using-geospatial-data-eec1b7f81151

- Bhoda, S. K. (2023a). *Making geospatial technology accessible: Why it shouldn't be a privilege*. LinkedIn. https://www.linkedin.com/pulse/making-geospatial-technology-accessible-why-shouldnt-privilege-bhoda-cxrgc/

- Bhoda, S. K. (2023b). *Open geospatial data: Democratizing access to dynamic intelligence. LinkedIn*. https://www.linkedin.com/pulse/open-geospatial-data-democratizing-access-dynamic-santosh-kumar-bhoda-0ktbc/

- CARTO. (2024). *Democratizing spatial analysis with raster data on the cloud*. CARTO Blog. https://carto.com/blog/democratizing-spatial-analysis-raster-data-on-the-cloud

- Chamberlain, H. R., Darin, E., Adewole, W. A., Jochem, W. C., Lazar, A. N., & Tatem, A. J. (2024). *Building footprint data for countries in Africa: To what extent are existing data products comparable?* Computers, Environment and Urban Systems. https://www.sciencedirect.com/science/article/pii/S0198971524000334

- DroneDeploy. (2018). *Democratizing geospatial data with drones*. Geospatial World. https://geospatialworld.net/article/democratizing-geospatial-data/

- Esri. (2023). *The emergence of GeoAI in planning*. Esri Industry Blog. https://www.esri.com/en-us/industries/blog/articles/the-emergence-of-geoai-in-planning/

- Foursquare. (2023). *Foursquare brings enterprise-grade spatial analytics to your browser with Kepler.gl 3.1*. Location Foursquare Blog. https://location.foursquare.com/resources/blog/products/foursquare-brings-enterprise-grade-spatial-analytics-to-your-browser-with-kepler-gl-3-1/

- GeoRetina. (2023). *Chat2Geo Project*. GitHub. https://github.com/GeoRetina/chat2geo

Glasze, G., & Perkins, C. (2015). *Social and political dimensions of the OpenStreetMap project*. In J. Arsanjani (Ed.), OpenStreetMap in GIScience. Springer. https://www.geographie.nat.fau.de/files/2018/02/Glasze-Perkins-Social-and-Political-Dimensions.pdf

- Gorelick, N., Hancher, M., Dixon, M., Ilyushchenko, S., Thau, D., & Moore, R. (2017). *Google Earth Engine: Planetary-scale geospatial analysis for everyone*. Remote Sensing of Environment, 202, 18-27. https://doi.org/10.1016/j.rse.2017.06.031

- Government of India. (2021). *Guidelines for acquiring and producing geospatial data*. T-KARTOR. https://www.t-kartor.com/blogs/democratizing-geospatial-data

- Hallas, M., Price, R., & Haithcoat, J. (2022). *Replicable AI for Microplanning (ramp): Democratizing Geospatial Data Science for Global Health*. AGU Fall Meeting 2022, Chicago, IL. https://ui.adsabs.harvard.edu/abs/2022AGUFMGC42E0755H/abstract

- Han, S. Y., Kim, J. S., Jiang, Y., Kang, J. Y., Park, J., Han, C., Michels, A., & Wang, S. (2024). *CyberGIS-Vis for Democratizing Access to Scalable Spatiotemporal Geovisual Analytics: A Case Study of COVID-19*. Illinois Experts Research & Scholarship. https://experts.illinois.edu/en/publications/cybergis-vis-for-democratizing-access-to-scalable-spatiotemporal-

- Hird, J. (2021). *Satellite time series and Google Earth Engine democratize the process of forest-recovery monitoring over large areas*. MDPI Open Access Journals. https://www.mdpi.com/2072-4292/13/23/4745

- Lartey, D. (2024). *GeoGPT+: Democratizing geospatial data analysis with AI. Data Storytelling Corner*. https://medium.com/data-storytelling-corner/geogpt-revolutionizing-geospatial-data-analysis-with-ai-78e3f7e4c1e1

- Mansourian, A., & Oucheikh, R. (2024). *Physical Geography aChatGeoAI: Enabling Geospatial Analysis for Public through Natural Language, with Large Language Models*. ISPRS International Journal of Geo-Information, 13(10), 348. https://www.mdpi.com/2220-9964/13/10/348

- Marten, K. (2019). *Geospatial analysis made easy with two new spatial functions: MakePoint and MakeLine*. Tableau Resources. https://www.tableau.com/blog/geospatial-analysis-made-easy-two-new-spatial-functions-makepoint-and-makeline

- Michels, A. C., Padmanabhan, A., Xiao, Z., Kotak, M., Baig, F., & Wang, S. (2024). *CyberGIS-Compute: Middleware for democratizing scalable geocomputation*. SoftwareX, 26, 101581. https://doi.org/10.1016/j.softx.2024.101581

- Plunkett, G. (2019). *The role of SDIs in the democratization of geospatial data and services*. Esri Canada. https://resources.esri.ca/news-and-updates/the-role-of-sdis-in-the-democratization-of-geospatial-data-and-services
- Tula Foundation. (2024). *Connecting the dots: Democratizing geospatial data*. Tula.org News. https://tula.org/news/connecting-the-dots-geoconnections

03
METHODS AND RESOURCES

The full source code and documentation for the *Are U Query-ous?* platform is available at: https://github.com/nicodalessandro11/uoc-tfg-auq

3.1. Development Strategy and Project Organization

This project was developed step by step, following a clear and modular structure. Each part of the system was placed in its own folder. Later, all parts were moved into a single repository (monorepo) to improve organization, make integration easier, and allow shared testing.

The full source code and documentation for the *Are U Query-ous?* platform is available at: https://github.com/nicoldalessandro11/uoc-tfg-auq

The development followed this sequence:

1. **Database design**: The database was the first component implemented. It was built using PostgreSQL + PostGIS with schema migrations and RLS policies to support geospatial data. The full schema, test data, and reset tools are included in the **auq_database/** folder.

2. **ETL development**: The second step was building the data engine in the **auq_data_engine/** folder. It collects, cleans, and uploads open datasets into the database. The scripts are written in Python and follow a clear structure based on city and dataset type.

3. **Frontend-first approach using BaaS (ongoing)**: Instead of creating a backend API first, the frontend was developed early using Supabase. Supabase works as a **Backend-as-a-Service (BaaS)**, providing a ready-to-use API, authentication, and direct database access. This allowed the frontend to use real data and help improve the schema during development.

4. **Custom backend (planned)**: After the frontend is complete and well tested, a custom backend (**auq_backend/**) will be developed. This backend will offer full control over API logic and improve security and performance for production use.

5. **Deployment and containerization (planned)**: For local development and testing, the project includes Docker support using **docker-compose.yml**. This allows all services (database, frontend, etc.) to run in containers and simplifies future deployment.

The monorepo folder structure is:

- **auq_database/:** SQL schema, seed data, reset, migrations.
- **auq_data_engine/:** Python ETL scripts organized by city and dataset.
- **auq_frontend/:** Web application built with React, Next.js, Leaflet, and Tailwind CSS.
- **auq_backend/:** Placeholder for the custom backend API (to be implemented after frontend testing).
- **auq_nlp/:** Reserved for optional future natural language features (not yet implemented).
- **shared/:** Common utilities and functions used across modules.

The project uses professional practices such as semantic versioning, a standardized Git commit template, and detailed changelogs.

This modular structure and step-by-step approach helped build a flexible system, where each part could be developed, tested, and improved separately.

3.2. Technologies and Tools Used

The project uses modern technologies, organized in four main areas: database, data processing, frontend, and infrastructure.

3.2.1 Database

- PostgreSQL (v15+) with PostGIS (v3+).
- Supabase as cloud DB with built-in REST API and auth.
- Row-Level Security (RLS) for role-based access (anon, authenticated, service_role).
- Security-definer views were removed for Supabase compatibility.
- SQL scripts for schema definition, seeding, and reset logic.

3.2.2 Data Engine (ETL)

- Python 3.11 with Pandas, GeoPandas, Shapely.
- Folders: **data/raw/, data/processed/**, **scripts/, tests/.**
- Tests: geometry integrity (**test_geometry_integrity.py**).

The ETL process follows three main steps:

1. **Extract**: Download raw datasets into **data/raw/.**
2. **Transform**: Clean and validate data, save results in **data/processed/.**
3. **Load**: Upload clean data to Supabase if all tests pass.

3.2.3 Frontend

- **Next.js** (React-based framework): Used to build the web application.
- **React**: For creating dynamic user interfaces with components.
- **Leaflet**: JavaScript library for interactive maps.
- **Tailwind CSS**: For fast and responsive styling.
- **Supabase Client SDK**: Used to connect the frontend directly to the Supabase database.
- **React Context**: For managing shared state like selected city, filters, and map layers.

The frontend is organized by pages (map, compare, visualize, admin) and uses context providers for state and data loading. Data is always fetched from the real database, not mock files.

3.2.4 Infrastructure and Dev Tools

- **Git**: For version control and project history.
- **GitHub**: Hosting the repository and managing code.
- **Docker & Docker Compose**: For local development and container-based deployment.
- **Makefile**: For running tasks like database setup and tests.
- **Excel calendar**: Used for planning and tracking progress.
- **Markdown files**: All parts of the project include a **README.md** and there's also a **CHANGELOG.md** file in the root of the repository with a summary of the git logs .

3.3. System Design

This section explains how the app works. It shows *who uses it, what they do, and how the app answers their actions*. We also include Hand-drawn sketches of the different GUI elements and the navigation between them aiming at helping to understand how the app is built.

3.3.1 User Stories

General User (Main Actor)

Who?	Any visitor of the web app such as general public, students, researchers, small business owners and other non-technical users.
What they do?	Explore the map. Apply filters by indicators (age, wage, etc.) Compare different areas. Understand data in a simple way. Possibly ask questions (if NLP is enabled).

User Story 1: How can the user see and move around the map?

US01 - View Map and Explore Data	
As a	General User
I want	to open the app and see the map.
So that	I can select an available city (Madrid or Barcelona) and explore it visually.
Acceptance Criteria	Zoom in and out, move around the map, see basic data like population economic information.

User Story 2: How can the user choose what kind of data they want to see?

US02 - Apply Indicators Filters	
As a	General User
I want	to apply filters like age, wage or education.
So that	I can see in the map specific information that I care about.
Acceptance Criteria	Filters easy to select. Map updates immediately to show filtered data.

User Story 3: How can the user look at differences between two places?

US03 - Compare Two Areas	
As a	General User
I want	to compare two zones on the map.
So that	I can see differences in things like population or income and make data driven decisions.
Acceptance Criteria	Areas or polygons well defined with clear numbers, colours and charts

User Story 4: How can the user understand the data?

US04 - Understand Data Visually	
As a	General User
I want	to see the data in a way that is easy to understand.
So that	I don't need any special training or technical knowledge to use the map and understand the information.
Acceptance Criteria	The app uses simple icons and clearly display the information.

User Story 5: How can the user talk to the app in normal words?

US05 - Ask Natural Language Questions	
As a	General User
I want	to ask questions in plain text like "where are more young people?"
So that	I can find and visualize answers without using menus or filters.
Acceptance Criteria	The app accepts a typed question and returns correct answers for visualize in the map.

Admin User (Secondary Actor)

Who?	The person in charge to maintaining and updating the application
What they do?	Upload or update open datasets. Manage system configuration (e.g., which cities or indicators are available). Monitor platform performance. Enable or disable features.

User Story 6: How can the admin add or change the open data?

US06 - Upload or Update Open Datasets	
As a	Admin User
I want	to upload or update datasets in the app.
So that	the application always shows up-to-date and correct information.
Acceptance Criteria	Upload a CSV or GeoJSON file and check it is correctly added to the system.

User Story 7: What kinds of data do we allow users to filter?

US07 - Manage Configuration	
As a	Admin User
I want	to be able to choose which indicators are available for the general users.
So that	I can control what types of data (wage, age, population, education etc.) users can explore.
Acceptance Criteria	There's and enable/disable indicators option in the admin or settings page. The changes should be visible for the general users.

User Story 8: What tools do we allow the user to use?

US08 - Enable or Disable Features	
As a	Admin User
I want	to disable or add new features.
So that	I can maintain the application balanced and easy to use and controls which app functions area available.
Acceptance Criteria	Every feature is listed in the settings page with a switch/option to turn it on/off.

User Story 9: How can the admin check if the app is working well?

US09 - Monitor Platform Performance	
As a	Admin User
I want	to check the system performance and the usage logs.
So that	I can detect if there is any problem and monitor peak usage hours.
Acceptance Criteria	Clearly see the logs errors and user activity data in the settings page.

System (Support Actor)

Who?	The application itself.
What they do?	Process and filter the requests. Manage database operations. Delivers data to the frontend and populate the map. Handles search, filter and comparison.

In this section, we showed the main people and parts that use or support the app. These are called actors, and each one has a special role:

- **General User**:
  This is any person who visits the app. They can be students, researchers, or people curious about the city. They use the map, add filters (like age or income), compare areas, and look at the information. If the AI tool is active, they can also ask questions in normal words.

- **Admin User**:
  This person controls the app. They upload new data, choose what filters and features are active, and check if everything works well. The admin helps keep the app updated and useful for others.

- **System**:
  The system is the app itself. It works in the background. It takes care of the data, processes the filters, and shows the correct information on the map. It also helps show answers when someone uses the search or comparison tools.

These three actors work together. The General User interacts with the map and data. The Admin User manages the content and features. The System makes sure everything works and connects correctly. Together, they create a smart and easy-to-use platform for geospatial data.

3.3.2 Use Case Diagram

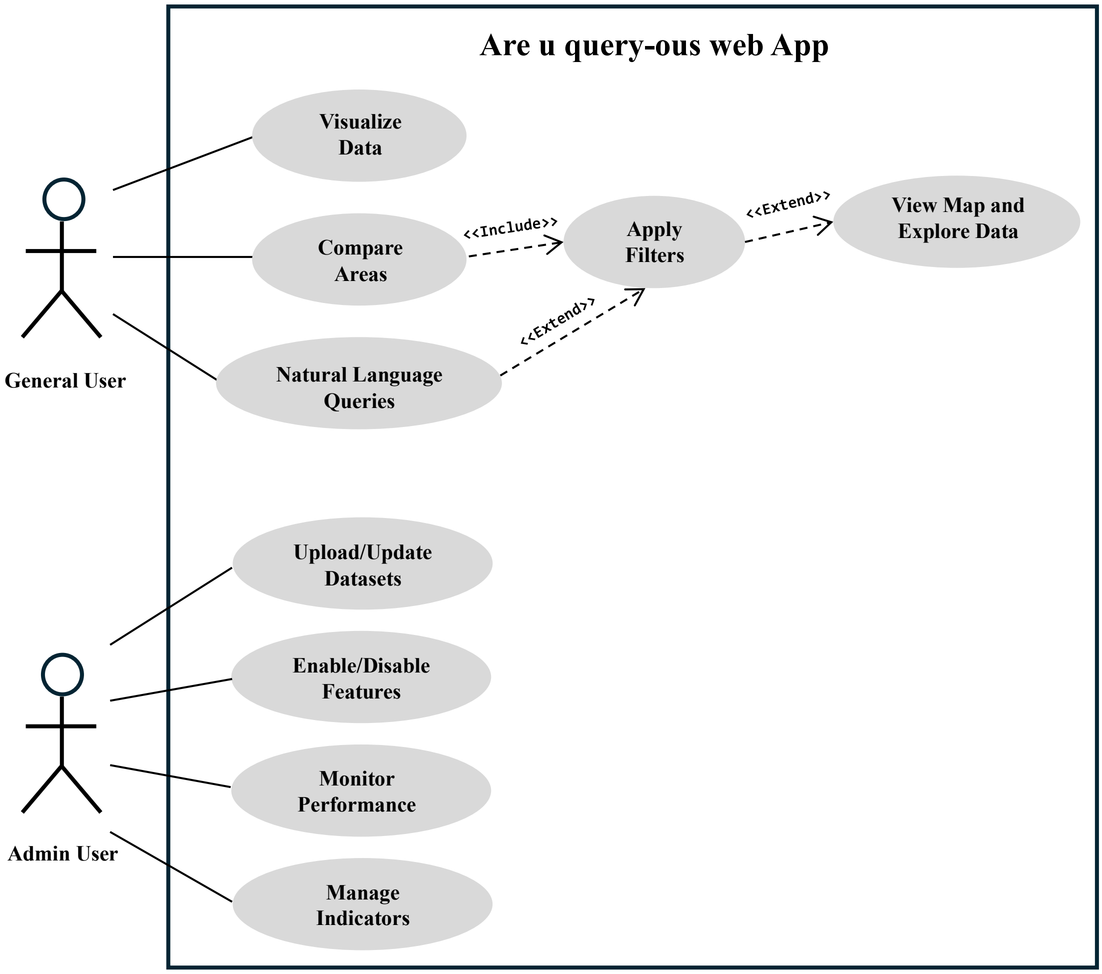

Figure 3 1: Use Case Diagram

This diagram shows how people use the "Are U Query-ous?" app. As we already define, there are two types of users: **General User** and **Admin User**.

- The **General User** can use the map, add filters (like age or income), compare areas, and see simple charts. They can also ask questions in simple words if allowed.
- The **Admin User** can upload new data, select what filters to show, switch features on or off, and check if the app is working well.

The lines show what each user can do. Some tasks need other tasks:

- **Compare Areas «include» Apply Filters**: comparing two areas always uses filters. We cannot compare two areas without first applying filters.
- **Apply Filters «extend» View Map**: applying filters adds more features to viewing the map but is not always required. We can view the map without applying filters, but applying filters makes it more powerful.

3.3.3 User interface design

In this part, we show how the app will look using wireframes. These are simple screen designs made with basic shapes. They are not the final design, but they help us plan where to put each element, like the map, filters, buttons, and charts.

We used low-fidelity wireframes to focus on the layout and the user experience. This method is fast and easy to change. It allows us to improve the design step by step and make sure the app will be easy for everyone to use. These wireframes also help explain the idea to others before we start building the real app.

Home View

This is the first screen of the app. The user can choose a city and see the map. The map has filters like age or income. The user can also ask questions in natural language. It is simple and easy to use.

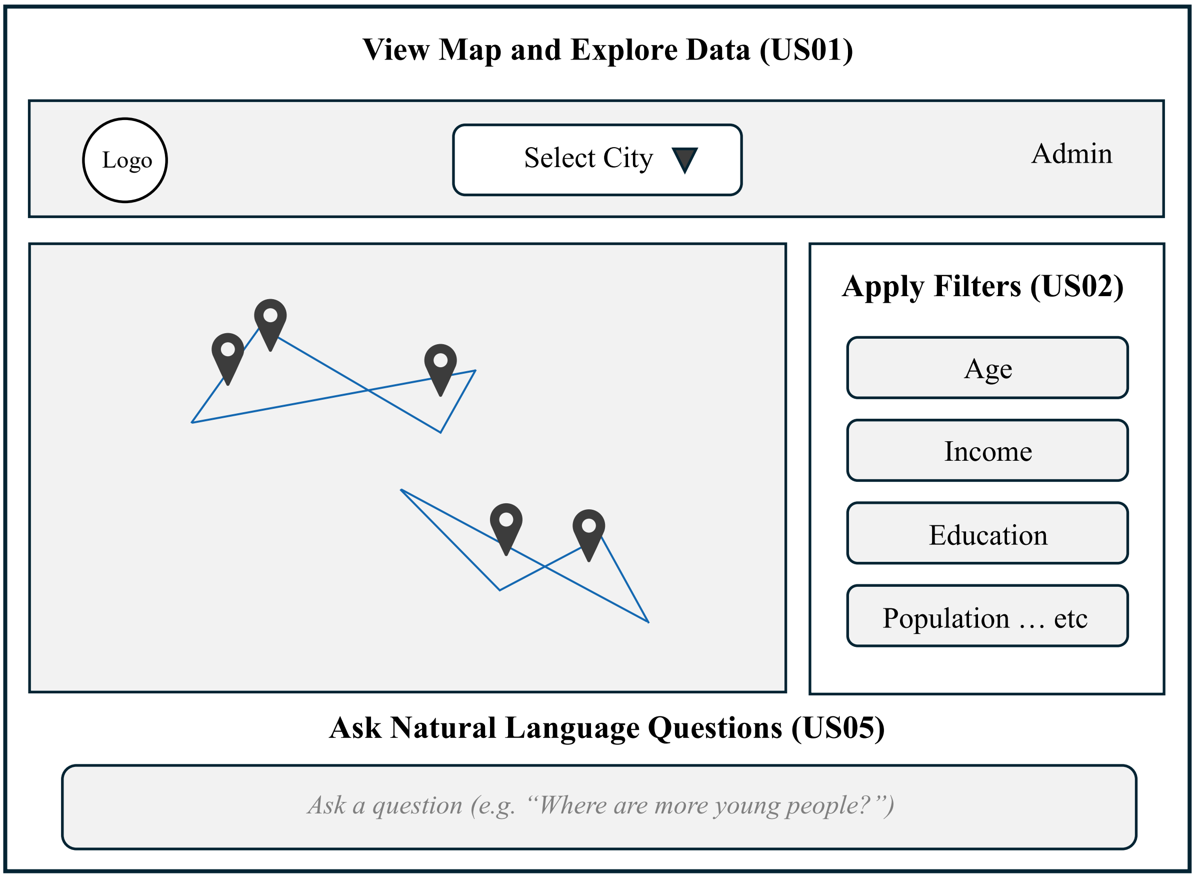

Figure 3 2 - Home Map View

Compare Two Areas View

This screen helps the user compare two areas on the map. The user selects one area and then chooses another area to compare. Both areas are shown side by side, with a simple shape for each one.

Below the areas, there is a table that shows the main information, like population and average income. This helps the user understand the differences between the two areas.

This view is useful for people who want to make decisions based on data, for example choosing the best place to live or start a business. It is easy to read and works without needing technical knowledge.

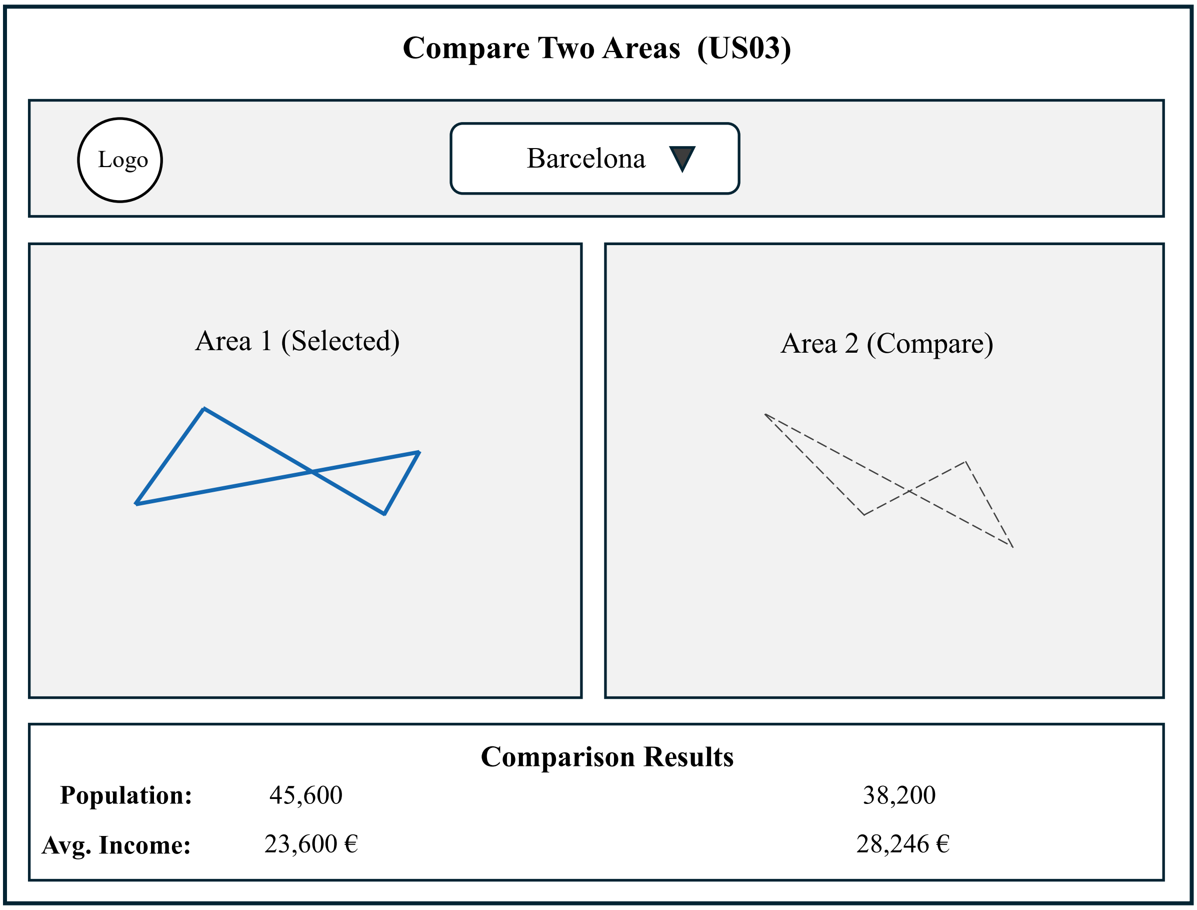

Figure 3 3 - Compare Areas View

Visualize Data View

This screen shows more information using charts and graphs. It helps users understand the data better. For example, they can see population or income for each area. It is good for users who want to learn more after using the map.

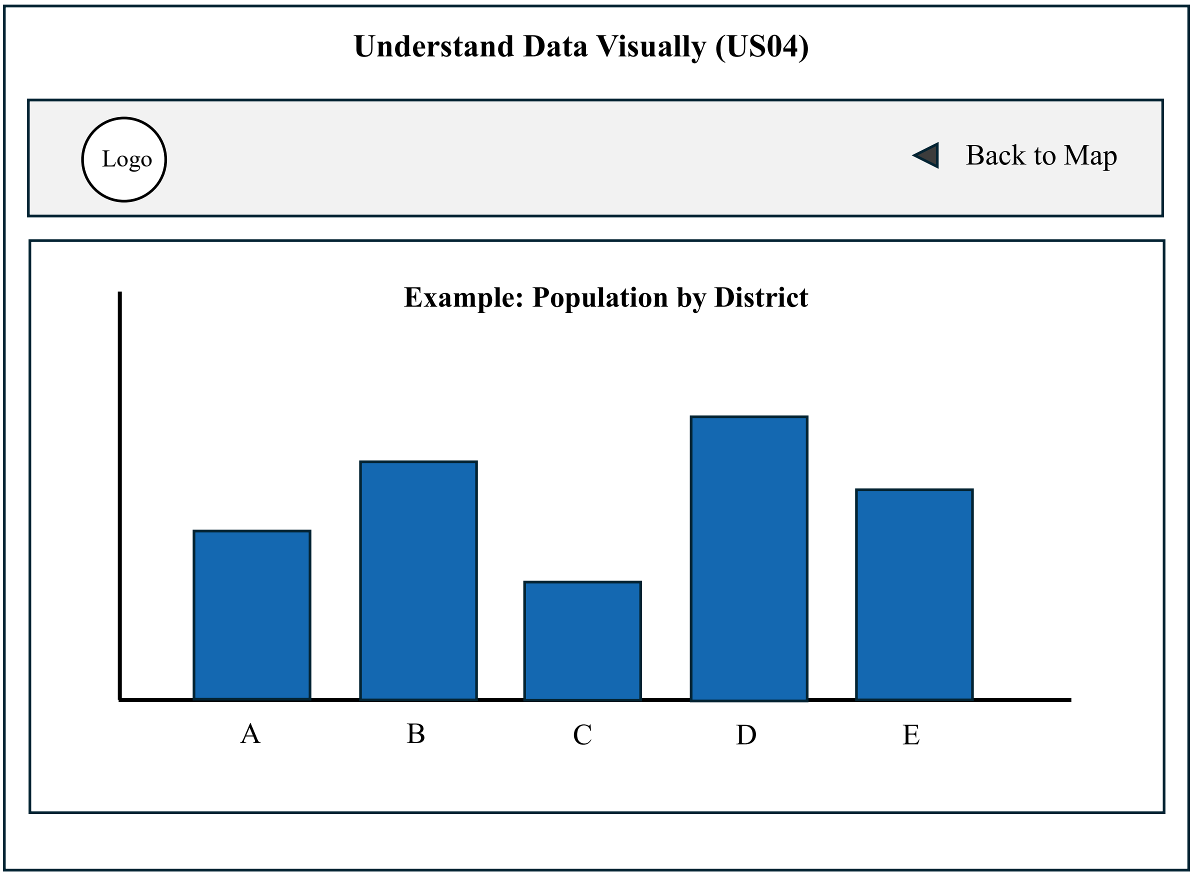

Figure 3 4 - Visualize Data View

Admin View

This is the screen for the admin. The admin can upload new data, change settings, and check if the app is working well. The admin can also turn features on or off. This helps keep the app updated and easy to use.

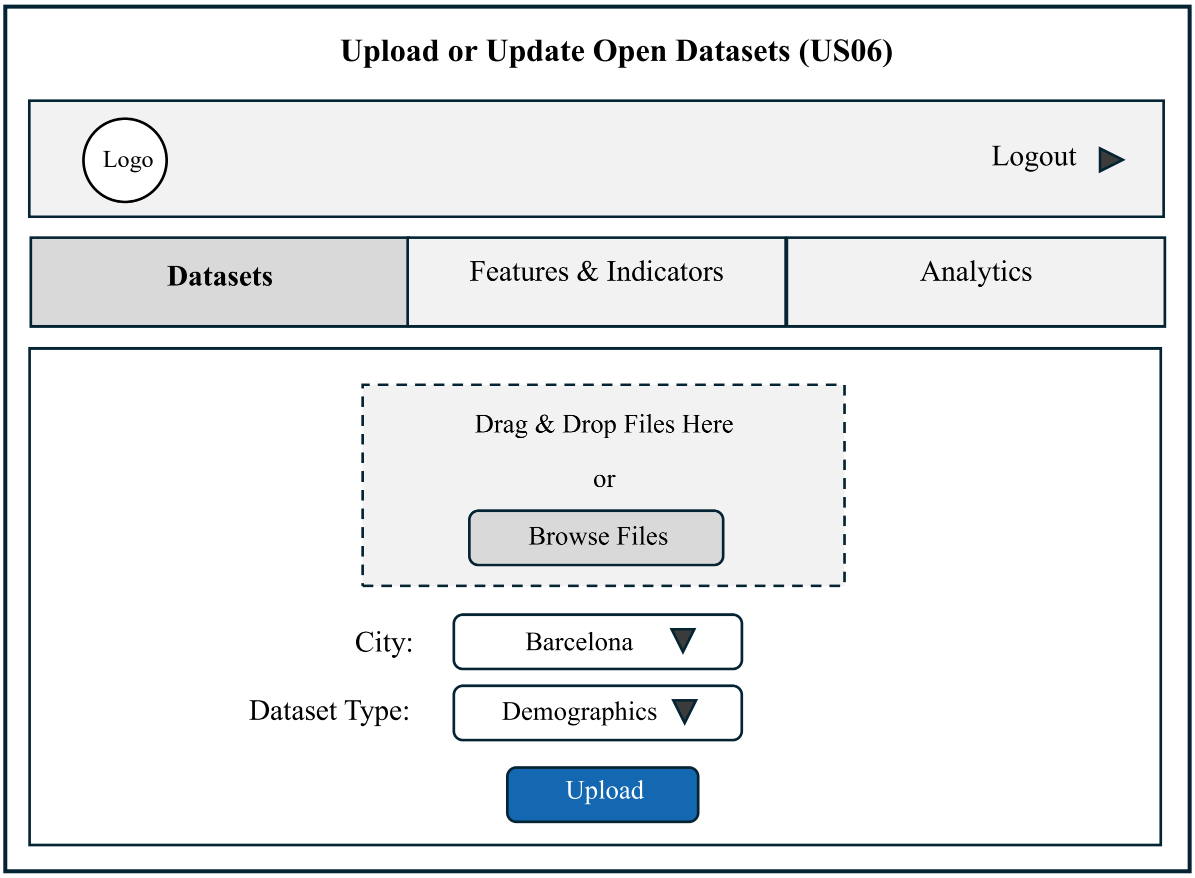

Figure 3 5 - Admin Upload Datasets View

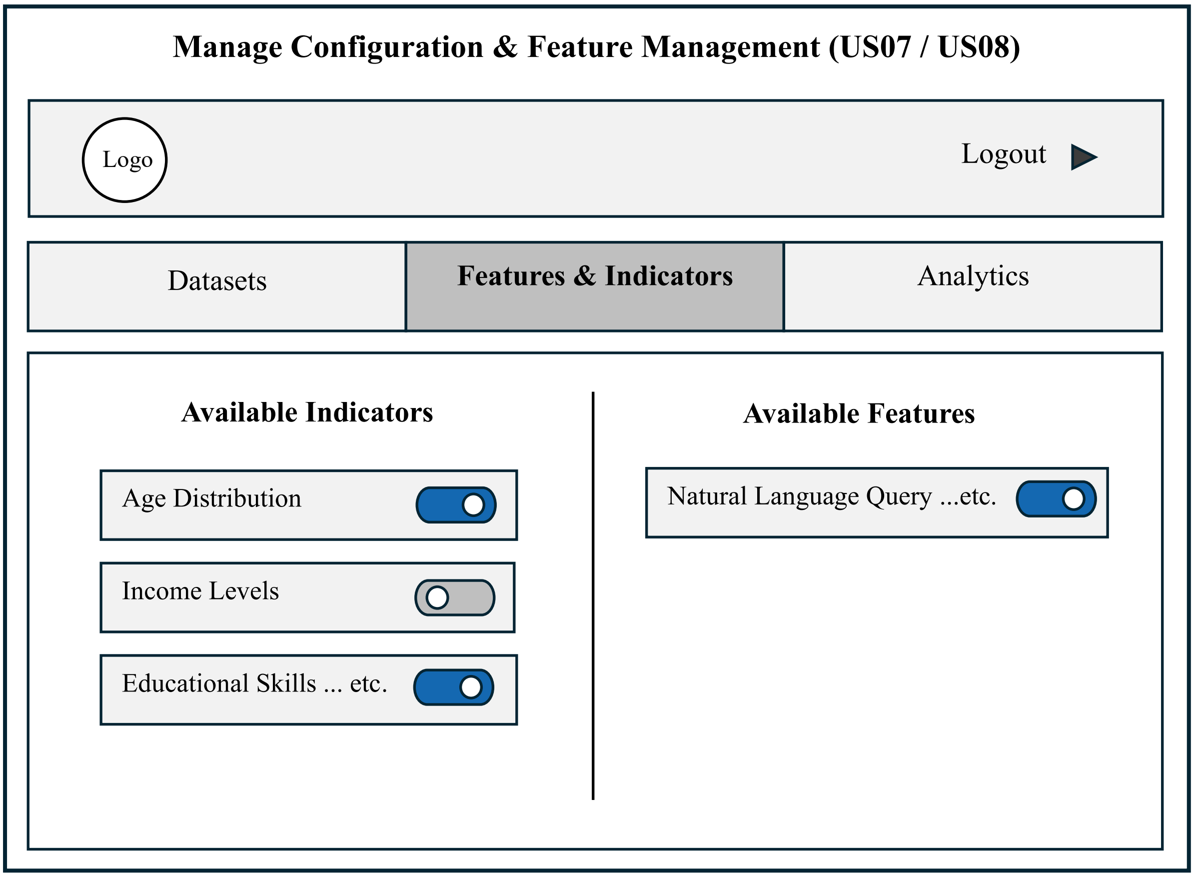

Figure 3 6 - Admin Manage Configuration & Management View

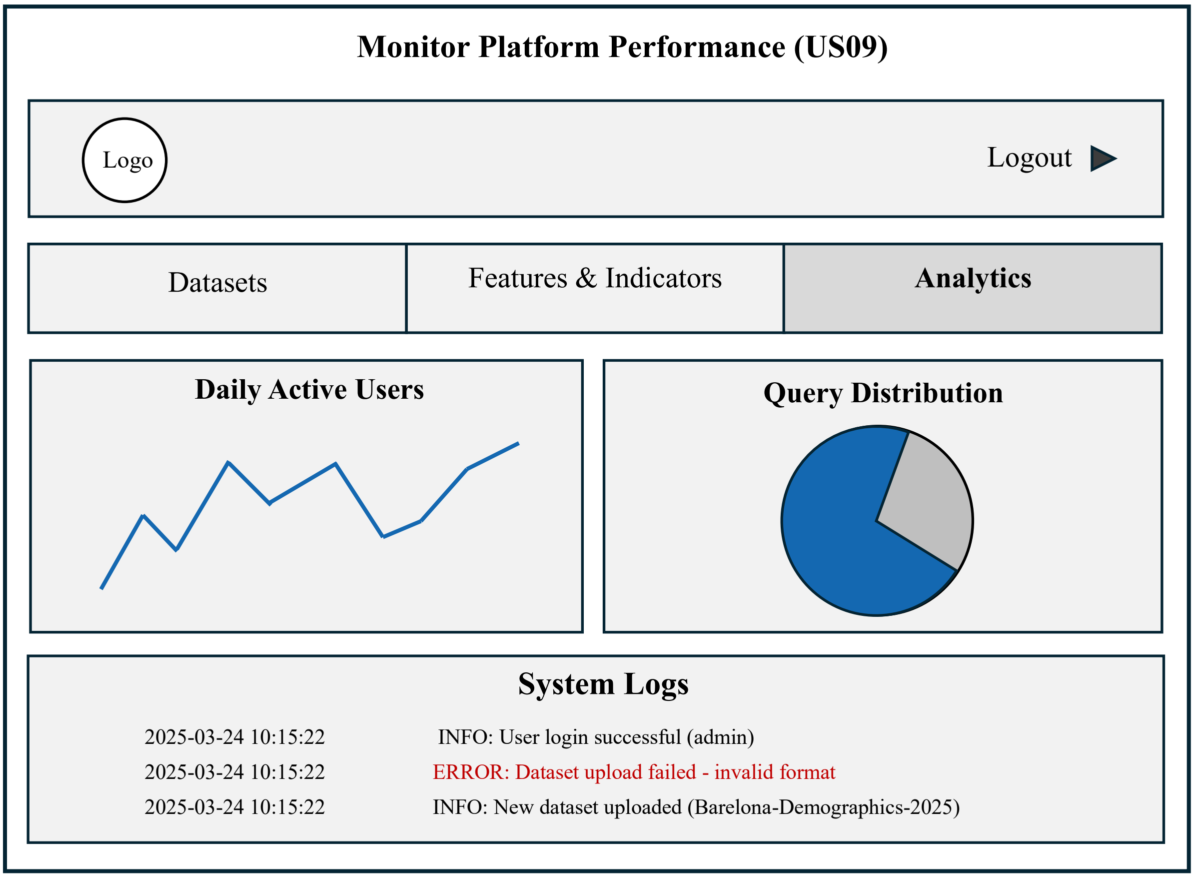

Figure 3 7 - Monitor Platform Performance View

3.3.4 Navigation Flow

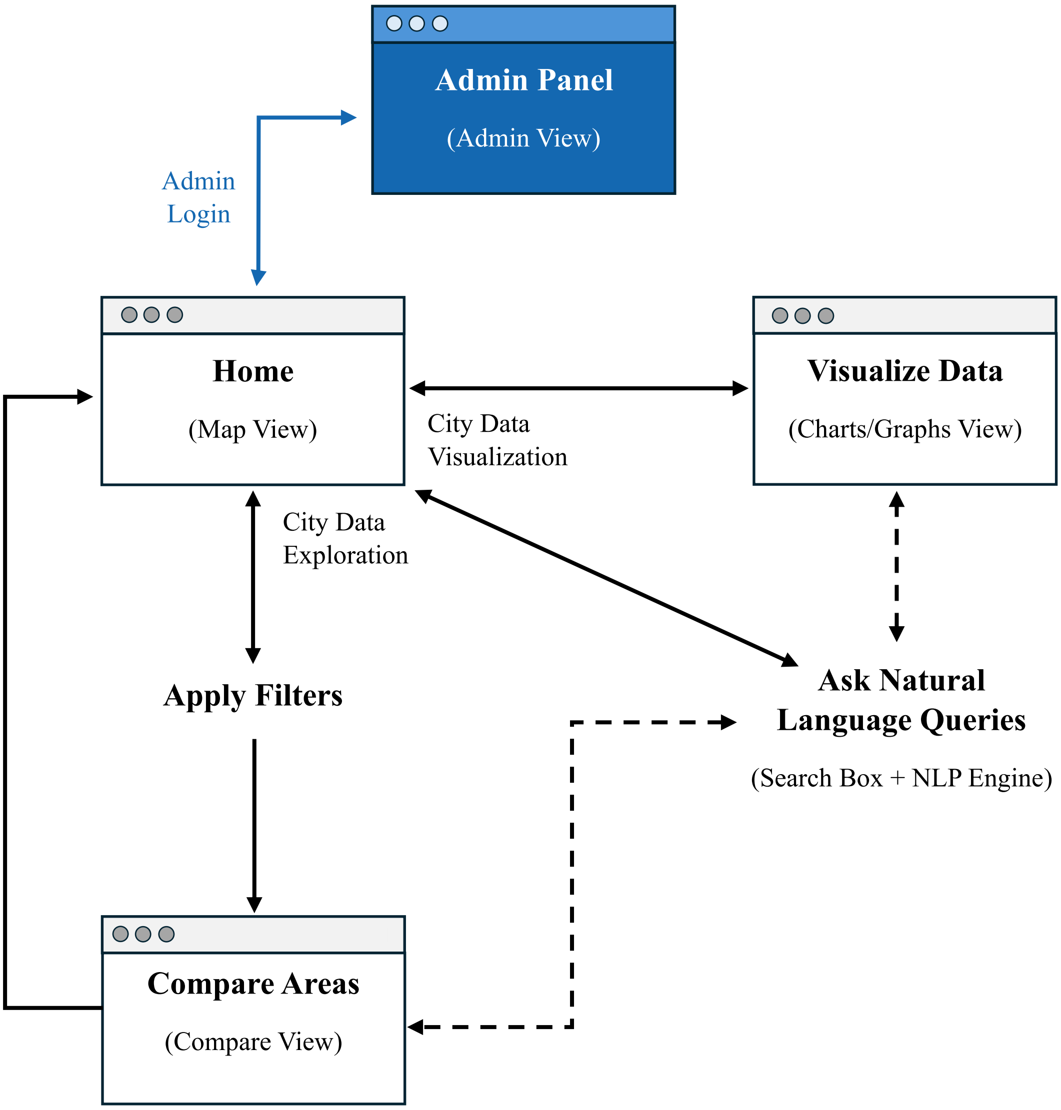

Figure 3 8 - Navigation Flow between the different Views

This diagram shows how the user moves between different parts of the app. The user starts on the **Home** screen. From there, they can:

- **Explore the map** and see data for a city.
- **Apply filters** to choose what kind of data they want to see (like age or income).
- Go to the **Compare Areas** screen to look at two places side by side.
- Go to the **Visualize Data** screen to see charts and graphs.
- If the AI is active, they can also **ask questions** in normal language and see answers.

If the user is an **Admin**, they can log in and go to the **Admin Panel**. There, they can upload data, change settings, and check if the app is working well.

3.3.5 Architecture Diagram

To design the right architecture for this project, we followed these simple steps:

Step 1: Understand the Project Context

We looked at the business domain. The project is about maps and city data. Users can easily see information about cities like Barcelona and Madrid. Our technical tools will be React.js, FastAPI, Python, and Supabase database. We will use cloud services (Heroku and Vercel). We chose simple technology because we have a time constraint, and the project should be delivered on the expected time. We made a list of the important features where we can highlight easy-to-use app, load fast and cost little money.

Step 2: Prioritize Requirements

We made a list of features and put them into three groups:

- **Must-Haves**: Map view, data filtering, and city information.
- **Should-Haves**: Simple user interface and fast responses.
- **Nice-to-Haves**: Natural language questions (NLP), analytics.

Step 3: Evaluate Trade-offs

We asked some questions to choose the best options:

- **Cost vs. Quality:** We picked free or cheap tools to keep costs low.
- **Speed vs. Maintainability:** We want clean code but also quick results since the final project timeline Is tight. We chose simple code that's easy to maintain.
- **Scalability vs. Simplicity:** Right now, simplicity is important. We can make the app bigger later.

Step 4: Apply Simplification Principles

We will follow three simple ideas:

- **YAGNI:** We will build only what we really need.
- **KISS:** Our solutions will be simple and easy to understand.
- **MVP:** We will create a small app first. Later we will add more features.

Step 5: Select an Architectural Pattern

After defined all the requirements, we finally chose a **layered architecture**. It's simple to understand, easy to build, and a good fit for this project timeline. This pattern also makes adding new features easy in the future:

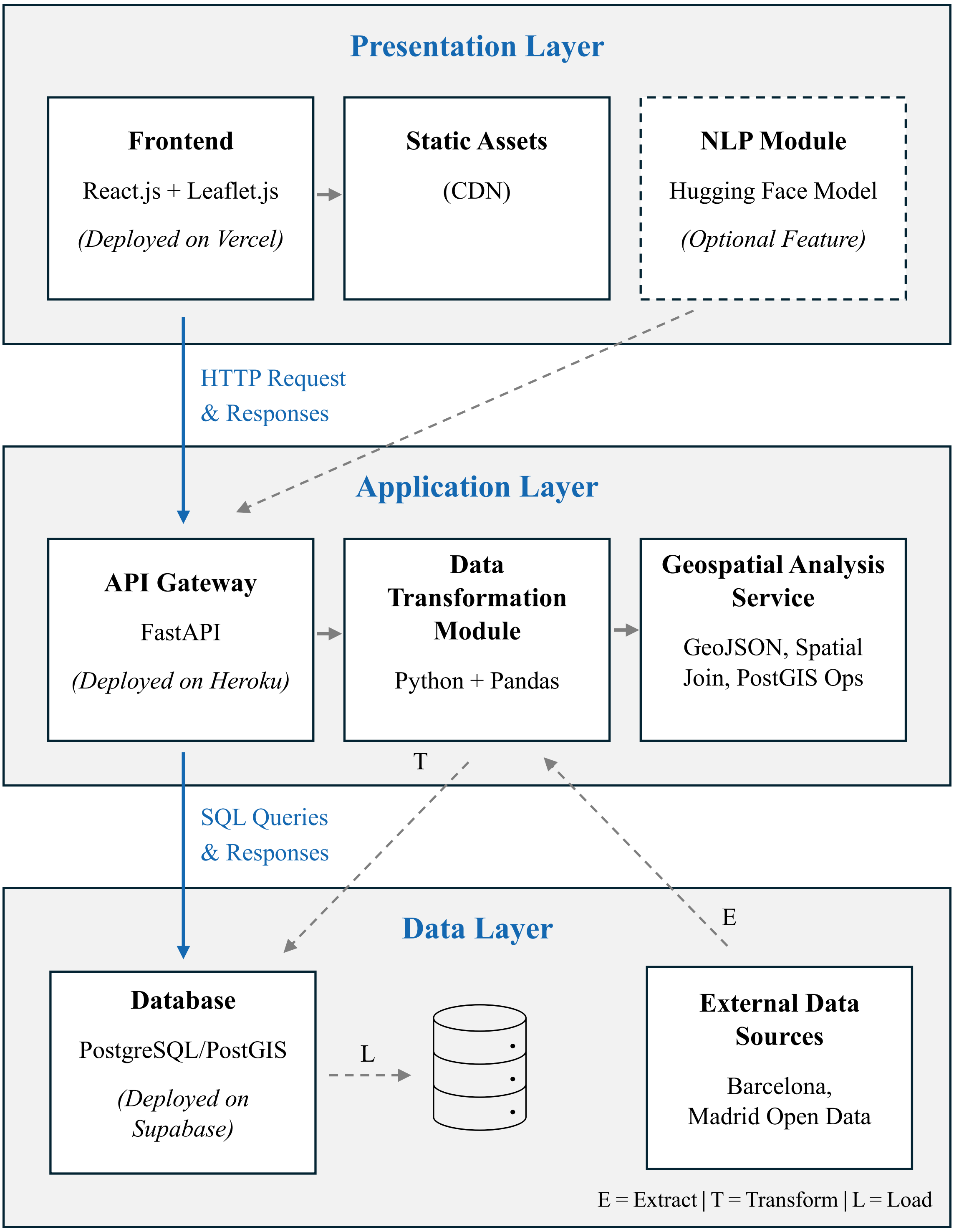

Figure 3 9 - Architectural Diagram

3.3.6 Database design

The database for the "Are U Query-ous?" platform is designed to manage two primary types of geospatial data:

- **Aggregated data** (Indicators such as population or income)
- **Geolocated data** (Point Feature such as schools or bus stops)

To ensure **flexibility and simplicity**, the database uses a traditional relational model with normalized tables and a virtual abstraction layer that allows indicators to be linked to multiple geographic levels.

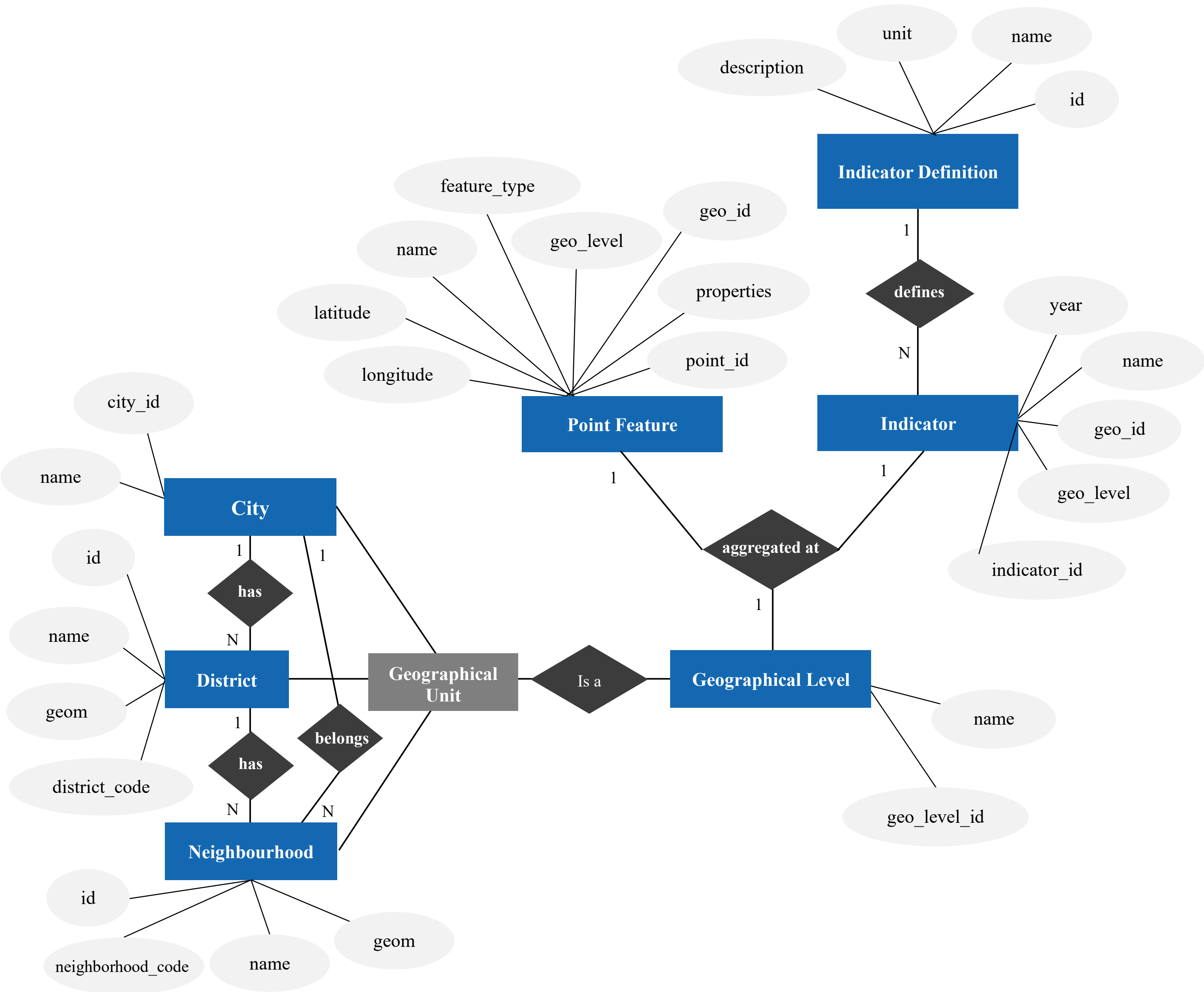

Figure 3 10- Entity Relation Diagram

In the **Entity-Relationship Diagram**, we introduce a virtual entity called **Geographical Unit**. This acts as a common structure that includes City, District, and Neighbourhood, using an "is-a" relationship.

The **Indicator** table stores values for different geographic levels and links to the Geographical Unit by using two fields: *geo_level_id* (which defines the type, like city = 1) and *geo_id* (the ID of the specific city, district, or neighbourhood). This makes the database cleaner, faster, and easier to manage.

The **Point Feature** table uses latitude and longitude to show exact map locations. If needed for visualizations calculations, a point can also be linked to a *Geographical Unit*, so we can group or filter features by area, like showing all schools in one district.

Entity	Description	Attributes
City	Cities covered by the platform	city_id (PK, Integer), name (Text)
District	Areas inside cities	district_id (PK, Integer), name (Text), city_id (FK → City), geom (GEOMETRY (POLYGON, 4326))
Neighbourhood	Areas inside districts	neighborhood_id (PK, Integer), name (Text), district_id (FK → District), geom (GEOMETRY (POLYGON, 4326))
GeographicalLevel	Combines all location types	geo_level_id (Integer), name (Text),
IndicatorDefinition	Indicator types	indicator_def_id (PK, Integer), indicator_name (Text), unit (Text), description (Text)
Indicator	Stores value for one place and year	indicator_id (PK, Integer), indicator_def_id (FK → IndicatorDefinition), geo_level_id (Integer), geo_id (Integer), year (Integer), value (Decimal)
PointFeature	Individual location on map	point_id (PK, Integer), feature_type (Text), name (Text), latitude (Decimal), longitude (Decimal), geo_level_id (Integer), geo_id (Integer), properties (JSONB)

Table 3 1 Entities, Field, Types and Foreign Keys

3.3.7 Class Diagram

This diagram shows how the main entities of the platform are related, including indicators, geographic levels, and point features.

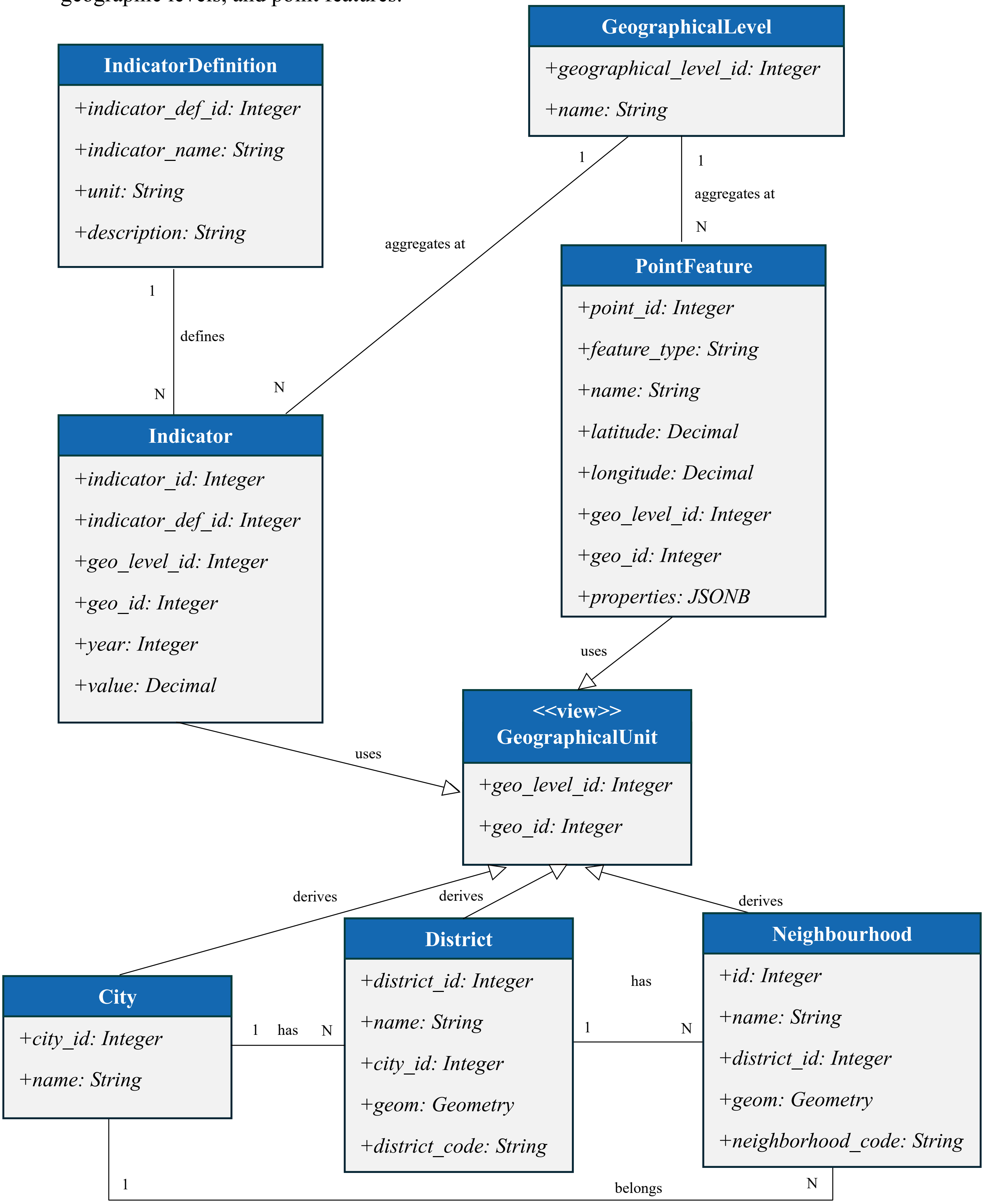

Figure 3 11 - UML Class Diagram

This **UML Class Diagram explains how the system is organized**. It shows how indicator values are connected to different areas like cities, districts, or neighbourhoods. Instead of using names like “city” or “district” directly, we use a table called *GeographicalLevel* with IDs. This keeps the database clean and fast.

The ***GeographicalUnit* is a view** that combines all areas (cities, districts, neighbourhoods) into one, so we can use simple queries. Both *Indicator* and *PointFeature* use this view to link their data to a place on the map.

This design helps keep the database flexible, clear, and ready for new types of data in the future.

How Is the Data Queried?

To avoid using text like “city” or “district” in every row, we use a number called *geo_level_id*.

For example:

- 1 = city
- 2 = district
- 3 = neighbourhood

This keeps the data faster and cleaner.

However, since we store cities, districts, and neighbourhoods in different tables, we need a smart way to join them.

To do this, we create a called *GeographicalUnitView*. This view joins all areas into one table.

With this view, we can write simple queries like:

```
SELECT i.indicator_id, i.year, i.value, g.name
FROM Indicator i
JOIN IndicatorDefinition d ON i.indicator_def_id = d.indicator_def_id
JOIN GeographicalUnitView g ON i.geo_level_id = g.geo_level_id AND
i.geo_id = g.geo_id;
```

Now we can:

- Get the name of the area (like “Eixample”) automatically
- Filter by area type if we want (only districts, only neighbourhoods, etc.)
- Keep the logic simple for future developments.

Example Record: ***Indicator***

We want to store the **average income in Eixample district (Barcelona) in 2023**:

1. *City Table (parent)*

city_id	name
1	Barcelona

2. *District Table (parent)*

district_id	name	city_id	geom
4	Eixample	1	[polygon geom]

3. *GeographicalLevel (parent)*

geographical_level_id	name
2	district

4. *IndicatorDefinition (parent)*

indicator_def_id	display_name	unit	description
1	Average Income	€	Avg. income per household

5. Example record for the average income in Eixample district (Barcelona) in 2023:

Field	Value
indicator_id	501
indicator_def_id	1
geo_level_id	2
geo_id	4
year	2023
value	26800

This record means: "*The average income in district ID 4 (Eixample) in 2023 was €26,800.*"

Example Record: ***Point Feature***

We want to add **a public school in Gràcia neighbourhood**:

1. *City Table (parent)*

city_id	name
1	Barcelona

2. *Neighbourhood Table (parent)*

district_id	name	district_id	geom
22	Gràcia	4	[polygon geom]

3. *GeographicalLevel (parent)*

geographical_level_id	name
3	neighbourhood

4. A public school in Gràcia neighbourhood.

Field	Value
point_id	304
name	"Escola Gràcia"
feature_type	"school"
latitude	41.403
longitude	2.160
geo_level_id	3
geo_id	22
properties	{"type": "public", "students": 420}

This record says: "*There is a public school in Gràcia (neighbourhood ID 22) with 420 students, located at the given coordinates.*"

3.3.8. System Design Conclusions

The "Are U Query-ous?" platform **represents a significant contribution to the democratization of geospatial data through several innovative approaches that differentiate it from existing applications**. While many current platforms focus primarily on providing raw data access or sophisticated tools for experts, this system uniquely bridges the gap between data availability and meaningful interpretation for non-technical users. The design's key contribution it is the level of abstraction of complex geospatial concepts into intuitive interfaces that require minimal technical knowledge, while maintaining the analytical depth necessary for meaningful insights.

The system's flexible database architecture solves a common problem in geospatial applications where users must understand complex relationships between geographic divisions to perform effective analyses. Furthermore, the platform's integration of multi-level filtering with visual comparison tools addresses the critical challenge of making spatial relationships and patterns immediately apparent to non-specialists (something that existing platforms often fail to achieve without requiring users to learn specialized GIS concepts).

Unlike many applications that simply visualize predefined datasets, "Are U Query-ous?" is designed from the ground up to empower users to explore, compare, and draw their own conclusions, transferring agency from developers to users. This represents a fundamental shift from treating the public as passive consumers of map products to enabling them as active participants in spatial knowledge creation.

By combining a user-centric design philosophy with technical innovations in data modelling and visualization, the platform contributes to the broader goal of making spatial intelligence an accessible resource for everyday decision-making rather than a specialized technical domain.

This system design reflects the original architecture and structure defined before the development phase. It is included in this chapter to keep project traceability. In the Results chapter, all changes made during implementation and final deployment will be explained and documented clearly.

3.4. Development Methodology

The project followed a **step-by-step** and **modular** development method. Each part of the system (database, data engine, frontend, and backend) was developed separately, but always with the full system in mind. The development was not based on a formal agile framework, but it used short iterations with clear goals.

Each phase of the project had its own focus:

1. **Database First**:
   The database was the foundation of the project. The first tasks were to design the data structure, define the tables, and insert test data. Geospatial features were added using PostGIS.

2. **ETL Development**:
   After the database was ready, the data engine was created to collect open datasets, clean them, and upload them to the database. Each script was tested with real data. Only after passing all validation checks, the data was uploaded to Supabase.

3. **Frontend-Driven Integration:**
   Instead of using mock data, the frontend was connected directly to Supabase. This allowed the application to use real data from the beginning. While building the frontend, some changes were made to the database schema and ETL process, depending on what was needed in the interface. This method made development faster and more realistic.

   Note: This approach **modified the system design presented in Section 3.3**, which had planned to build the backend first. However, the original design is kept in this report for **project traceability**. All changes and the final version of the system will be explained in the *Results* chapter.

4. **Backend (Planned)**:
   The custom backend was not implemented during this stage. However, it is planned for the next delivery. It will offer more control over API endpoints, business logic, and security for production. Until then, Supabase acts as a temporary backend using its built-in API.

5. **Testing and Validation**:

   Data Engine

   - **Pytest** for data validation and integrity.
   - **test_geometry_integrity.py** ensures geometries match expectations.
   - Scripts include automatic logging and summary outputs.

   Frontend

   - Manual testing with real data.
   - Debug components for **Supabase** connectivity.
   - Logs and browser dev tools for layer validation.

3.6. Project Planning and Workflow Tools

The project was planned to use a **simple but effective workflow** based on personal task management and version control.

1. ***Planning***

   The project Excel was used to organize the main phases of the project, track weekly goals, and set deadlines. This calendar helped follow a clear timeline and avoid delays.

2. ***Version Control***:

   Git was used for version control. Each change was saved in commits using a clear and professional shared commit template including the type of change, date, and a short explanation. This made the history easy to read and follow.

   - Follows semantic versioning (v1.0.0, etc.).
   - Uses standardized commit format (see **commit_template.md**).
   - Changelog updated automatically (**CHANGELOG.md**) using **generate_changelog.py.**
   - GitHub used for version control, backup, and review.

3. **Documentation**

Each part of the project includes its own **README.md** file, with instructions and explanations. A global **CHANGELOG.md** file was used to record all important changes during development. This documentation helps understand what was done and why.

4. **Makefile Automation**:

A **Makefile** was used to simplify common tasks like resetting the database, running tests, or loading seed data. This saved time and reduced the chance of errors. The **Makefile** provides standardized CLI commands, among others we can find:

- **make reset-and-migrate-db**: Drops and rebuilds DB schema.
- **make run-engine**: Runs data ingestion pipelines.
- **make test**: Runs Pytest validations.
- **make install-frontend**: Install frontend dependencies
- **make dev-frontend**: Run frontend development server

5. **Python Development Guidelines (ETL)**

- Based on **python_scripts_guidelines.md**, all ETL scripts follow these:
- Logging for traceability (own module using emojis developed for this project).
- Modular functions (extract, transform, load).
- Validation before upload.

6. **Data Privacy and Access Control**

The project uses only open public data, with no personal or sensitive fields.

- **Supabase RLS policies** restrict access to ensure read-only for anon.
- **service_role** handles ETL uploads securely.
- Admin access is protected via frontend login and Supabase rules.
- Dataset sources and disclaimers are shown in the UI.

7. **Natural Language Queries (Planned – Optional feature not yet implemented)**

   Although not implemented in CAT3:

   - Folder **auq_nlp/** created.
   - Target: Hugging Face models (bert-base, distilGPT).
   - Goal: Convert questions to filter expressions (e.g., "Where are more young people?").
   - Architecture includes optional NLP widget in map view.

This simple setup allowed the project to move forward in a controlled and organized way, without needing complex tools.

CAT3 Delivery Note - Project Status and Implementation Summary

This document summarizes the status of the project at the end of the CAT3 delivery phase. It includes a detailed explanation of the tasks completed so far, the main challenges faced during development, and the next steps required to finish the implementation. This note is part of the project documentation and connects the current progress with the final results that will be delivered in June.

1. Database Development

The development of the database was successful. First, different open data sources were explored and selected to create a strong foundation. A schema was written using SQL, and a **seed.sql** file was created to load sample data into the system. These scripts are included in the **auq_database/** folder.

A Makefile was created to automate database setup, seed, and reset operations. This helped reduce human errors and ensured consistency across local environments.

The database includes cities, districts, neighbourhoods, point features, and indicators. It is designed to support scalable, multi-city geospatial analysis.

2. Data Engine (ETL) Development

This part of the project was one of the most difficult. The main challenge was the lack of consistency between open data formats from different cities (Barcelona and Madrid). Each city uses different standards, which made it hard to build a single process to clean and transform the data.

Several weeks were dedicated to finding suitable datasets and making them compatible. Point features were also selected and are listed in the **seed.sql** file. All source data is stored in an external S3 bucket. The ETL scripts download the raw files, clean the data, and then upload it to the database.

This strategy was chosen to balance the academic time limits with the importance of showing a relevant and working system. The idea was to create a representative project that could be easily extended in the future if needed.

3. Development Methodology

Due to the limited time of the academic calendar, the project used a frontend-first development approach. Normally, a backend API would be created before the frontend, but this method allowed faster progress using real data from the beginning.

Supabase was used as a Backend-as-a-Service (BaaS). It provided instant access to the database through RESTful APIs and authentication. This approach allowed the frontend to be built and tested in a real environment.

This method is often used in professional settings, especially when time is short. It is a modern and efficient solution that also helps build more robust frontend systems. Using real data during development gave better results and fewer surprises in later stages.

The custom backend API is still planned and will be developed after the frontend is finished. An API_DESIGN.md file is already included in the repository. It shows the planned structure and endpoints of the future backend.

4. Current Project Status

As of the CAT3 delivery date, the main focus is on finalizing the frontend. A working prototype has been deployed and is available for testing. However, some features need correction and refinement. The development of the frontend was much more complex than expected. Connecting the frontend to real geospatial data and adapting it to the system design required many adjustments.

There were multiple conflicts between the original frontend design and real implementation constraints. These challenges caused delays in the timeline. Even so, with many hours of work and adjustments, the goal is to complete a functional and polished frontend by the next delivery.

Once this is done, the backend API will be implemented, and the full system will be connected and deployed for the final documentation deadline (June 3).

5. Documentation Approach

This chapter gives a general overview of the methods and resources used in the project. However, because the project has many parts, more detailed information is provided in separate files.

Each module, backend, frontend, and data engine, includes its own **README.md** and supporting documentation. These files explain the internal structure, decisions, and technical details of each part.

Even though the app is not fully finished yet, it is highly recommended to read these files. They are an important part of the project and reflect the professional way it has been developed and documented.

The goal is to not only build a working application, but also to show a clear, traceable, and professional development process.

CAT3 Submission Deliverables

The following documents and files are included in this CAT3 delivery:

- Updated **project Excel calendar** with the actual timeline and completed tasks.
- **CHANGELOG.md + git_history.md**: A full history of commits with dates and types, following the recommended commit format.
- **Various README.md**: A copy of each project module readme that describes the project in its current CAT3 state including the guide to clone the repository and run the project locally for testing the current prototype.
- **Markdown files in docs/**: Includes useful technical documents such as:
    - Commit template
    - Python scripts best practices
    - Other internal development guidelines

04
RESULTS

This section will be further developed in the future Continuous Assessment Tests (CAT4) and project iterations. Additional details, refinements, and validations will be incorporated as the project progresses.

05
CONCLUSIONS AND FUTURE WORK

This section will be further developed in the future Continuous Assessment Tests (CAT4) and project iterations. Additional details, refinements, and validations will be incorporated as the project progresses.

06
GLOSSARY

This section will be further developed in the future Continuous Assessment Tests (CAT4) and project iterations. Additional details, refinements, and validations will be incorporated as the project progresses.

07
BIBLIOGRAPHY

This section will be further developed in the future Continuous Assessment Tests (CAT4) and project iterations. Additional details, refinements, and validations will be incorporated as the project progresses.

08
APPENDICES

This section will be further developed in the future Continuous Assessment Tests (CAT4) and project iterations. Additional details, refinements, and validations will be incorporated as the project progresses