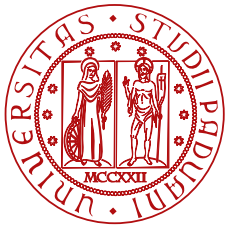


Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione


Geographic Information Systems

# Master Plan


Master Student

**Farzad Shami**

**Student ID 2090160**

Supervisor

**Prof. Massimo Rumor**

**Prof. Sandro Savino**

**University of Padova**


Academic Year
2022/2023

# Contents

# List of Figures

# List of Tables

# List of Algorithms

# 1 Introduction

The primary objective of this project is to develop an advanced Geographic Information System (GIS) application tailored for the management and utilization of the Master Plan of the Italian province of Fano with a population of approximately 250,000 inhabitants, Fano's Master Plan serves as a crucial blueprint for urban and territorial development. This project seeks to address the specific needs of planners and the general public, offering them a platform that facilitates the creation, maintenance, and accessibility of plan variants, thus fostering efficient urban planning and informed decision-making.

In essence, this GIS application aims to streamline the process of managing the Master Plan and its subsequent variants. By providing planning technicians with an intuitive and feature-rich environment, the application will enable them to create and manipulate plan variants. This includes the ability to define new zones, allocate attributes indicating intended land use (such as residential, agricultural, public green, commercial, and industrial), and transition between various plan variant statuses (in progress, planned, in force, not approved) while recording relevant dates for each change. Simultaneously, the application will empower citizens to effortlessly access the Master Plan and its associated variants. Citizens will be able to visualize zoning information, identify land parcels using cadastral identification codes, and obtain printable documents certifying the permissible land uses based on the plan's regulations.

# 2

# Analysis of Requirements

This phase encompasses the identification and definition of both functional and data-related needs, alongside the elucidation of nonfunctional requirements that will shape the architecture and features of the system.

## 2.1 Functional Requirements

1. Seamless Visualization of Master Plan and Variants (Users Types: Planning technicians, public)
   (a) Interface for exploring plan details
   (b) Tracking changes and observing evolving variants over time

2. Empowerment of Municipal Planning Technicians
   (a) Efficient creation and management of plan variants
   (b) Design new zoning geometries
   (c) Assign intended land uses
   (d) Transition between variant statuses

3. Variant Status Management
   (a) Change variant status (e.g., "in progress", "planned", "in force", "not approved")
   (b) Capture relevant dates
   (c) Comprehensive tracking of plan variant evolution

4. Query Application with Cadastral Identification Codes
    (a) Superimpose cadastral parcels on the Master Plan
    (b) Generate printable documents certifying permissible land uses
    (c) Certify intended purposes based on plan specifications

## 2.2 Data Requirements

The data requirements encompass the information necessary for the application's functionality and effectiveness. Geographical data, primarily in the form of shapefiles, will be crucial for representing plan zones and variants. Additionally, cadastral parcel data, which is regularly updated and available in shape format, will be integrated to support the identification of specific land parcels and their attributes. The application will leverage attributes associated with the Master Plan, including intended land use types and status information for plan variants. Furthermore, users will be able to generate and store notes and communications, which will be tied to specific geographical areas and contribute to an enriched understanding of plan-related concerns.

## 2.3 Nonfunctional Requirements

The proposed solution should leverage existing municipal software components including PostgreSQL, PostGIS, and GeoServer where possible, use additional open source components as needed, comply with applicable standards and regulations including national geospatial data standards, respect all applicable laws, and the web-based application portal should be compatible across desktop, tablet, and mobile devices. Overall the system should integrate with the municipality's existing IT infrastructure and geospatial data while meeting legal requirements.

# 3 Technology

The existing technological infrastructure provides a robust starting point for our GIS application development. The municipality currently utilizes PostgreSQL with the PostGIS extension for managing a comprehensive topographic database at a 1:2000 scale. This database, designed in alignment with national standards and maintaining an NC1 level of detail, is a fundamental repository of spatial information. Additionally, Geoserver serves as the platform for sharing and distributing geographical data, ensuring the availability of this critical data to relevant stakeholders. Building upon this established foundation, our application will leverage the PostgreSQL-PostGIS combination for efficient data storage, querying, and manipulation. The familiarity with these technologies will expedite the integration of our GIS application while enhancing data accuracy and accessibility.

## 3.1 Existing Data

Central to our project is the utilization of existing geographical data. The topographic database, meticulously constructed in the ETRF2000 reference system, presents a comprehensive spatial representation of the municipality. This dataset, offering a 1:2000 scale, encompasses intricate details and geometries crucial for accurate plan representation. Additionally, the availability of the current Master Plan and zone geometries in shapefile format, also aligned with the ETRF2000 reference system, furnishes us with foundational spatial information.

This existing data will serve as a cornerstone for our application's visualization and manipulation features. Leveraging shapefiles ensures compatibility and consistency, allowing us to seamlessly integrate these resources into our system's functionalities.

## 3.2 Market Analysis

A thorough market analysis reveals a distinct gap in available solutions that cater to the specific requirements of managing Master Plans and their variants. Packaged software solutions fail to align with the intricacies of Fano's needs, prompting the development of a custom GIS application.

In summary, the starting situation is characterized by a well-established technological foundation, accessible data resources, and an undeniable market need for a comprehensive GIS solution. Building upon these strengths, our GIS application will harness existing technologies and data to deliver a tailored solution that meets the intricate requirements of managing the Master Plan and its variants while filling a void in the available software landscape.

# 4

# Working Hypotheses

The working hypothesis is that a web-based GIS application can be developed to meet the requirements for master plan viewing and querying. This application will leverage the municipality's existing GeoServer, PostgreSQL, and PostGIS components to serve the master plan data through a web mapping interface. Open-source JavaScript libraries like OpenLayers or Leaflet can be used for the interactive map client. The application server can connect to the PostGIS database to retrieve master plan data filtered by parcel ID for zoning lookups.

Additionally, an editing plugin can be developed for the open source OpenJUMP GIS desktop application. This would allow municipal technicians to create and edit new master plan variant geometries and attributes. The plugin can connect to the PostGIS database to load existing plan data, and save any edits made back to the database. Integrating these two components - the web portal and OpenJUMP plugin - will provide both public viewing and internal editing capabilities for managing master plan data.

# 5 Project Draft

## 5.1 Goals

The overall goal of the project is to develop a GIS-based system for managing and providing public access to the master plan data for the municipality of Fano.

### 5.1.1 Detailed Goals:

1. Create an interactive web map application that allows citizens to view the current master plan and look up zoning information for specific parcels
2. Develop an editing plugin for municipal technicians to create, edit, and manage master plan variants
3. Integrate parcel/cadastral data to support parcel ID lookups and displaying zoning on individual parcels
4. Implement a central PostGIS database for master plan data and integrate with existing municipal IT infrastructure
5. Follow best practices for web mapping architecture, data standards, and security

### 5.1.2 Intermediate Goals:

1. Complete market analysis and requirements gathering

2. Design system architecture and draft plan for integration with municipal IT environment
3. Develop prototype web map application for public zoning lookups
4. Develop prototype OpenJUMP plugin for master plan variant editing
5. Integrate sample datasets and test parcel lookup and zoning identification functionality
6. Deliver prototypes to the customer for approval
7. Develop full web map application and OpenJUMP plugin based on prototypes
8. Deploy integrated system to municipal infrastructure
9. Load final master plan dataset into PostGIS database
10. Train municipal technicians on using OpenJUMP plugin
11. Promote web map application access to public

The intermediate goals aim to break the project into manageable milestones, deliver incremental prototypes for feedback, and ensure the final system meets the requirements identified early in the process. The prototypes provide an opportunity for validating functionality and usability before committing to full system development.

## 5.2 Technological Components

The technological components of our GIS project are instrumental in realizing the vision of an efficient and user-friendly Master Plan management application. These components encompass both software and hardware aspects, with careful consideration given to the tools that facilitate seamless interaction, data management, and system performance.

### 5.2.1 Database

The system's database serves as the central component connecting all other technological elements. Moreover, the customer's existing IT department already possesses a PostgreSQL database, which seamlessly aligns with our development needs. This is thanks to its PostGIS extension, designed for handling spatial and geographical data, making it an ideal foundation for our application's development.

### 5.2.2 Desktop GIS System (Plugin)

In accordance with specified criteria, the OpenJUMP plugin operates and helps by loading plan and variant layers, visualizing affected cadastral particles, identifying misaligned boundaries, and showcasing excluded or partially included parcels. Moreover, it accurately measures gaps between variant and cadastral parcel edges, highlighting sections within a 1-meter threshold in a new layer. This systematic process efficiently addresses functional requirements, facilitating precise geospatial analysis.

### 5.2.3 WebGIS Application

The central component of the system will be the web application, which must offer GIS capabilities and integrate various components:

| Component | Description |
|---|---|
| **Frontend Map Management** | Integration of the OpenLayers JavaScript plugin to allow users to utilize maps within the web app. |
| **Background Map** | Displaying a province-level background map using OpenStreetMap maps within the OpenLayers plugin. |
| **Backend Spatial Data Delivery** | Utilizing a dedicated Geoserver instance to deliver and display the spatial data available from the Veneto Region. |
| **Frontend Implementation** | Ensuring certain parts of the web application are accessible to provincial citizens on mobile devices. Implementing full responsiveness to make the system usable on any terminal, including smartphones and tablets. The Bootstrap Italia template is chosen for better integration with other public IT systems. |

Table 5.1: WebGIS Application central components

### 5.2.4 Software Requirements

Many softwares needs to be used to set up the system:

1. **Database**, utilize PostgreSQL 15 as the relational database management system (RDBMS), chosen for its stability, performance, and support for geospatial data manipulation. Additionally, the system shall incorporate the PostGIS extension 2.5.2 into the PostgreSQL database.

2. **Docker**, is a platform that allows you to develop, deploy, and run applications in isolated containers. Containers are lightweight, portable, and self-sufficient units that encapsulate an application along with its dependencies, enabling consistent and efficient deployment across different environments.

3. **Web Map Service (WMS)**, a system capable of delivering web maps is needed: for this, Geoserver is the best solution available.

4. **Tools, Frameworks & Libraries:**

   (a) **FastAPI:** FastAPI is a modern, fast (high-performance), web framework for building APIs with Python 3.6+ based on standard Python type hints.

   (b) **caddy-server:** Caddy 2 is a powerful, enterprise-ready, open source web server with automatic HTTPS written in Go and used as a reverse proxy, which uses namespaces routing,

   (c) **Minio server:** This template allows users can upload their photos. The images are stored using the open source Object Storage Service (OSS) minio, which provides storage of images using buckets in a secure way through presigned URLs.

   (d) **Celery:** Celery is a distributed task queue that allows developers to run asynchronous tasks in their applications. It is particularly useful for tasks that are time-consuming, require heavy computation or access external services, and can be run independently of the main application. It also offers features such as task scheduling, task prioritization, and retries in case of failure.

   (e) **SonarQube:** SonarQube is an automatic code review tool that detects bugs, vulnerabilities, and code smells in a project. You can read this post in order to have a better understanding about what SonarQube can do.

   (f) **Leaflet:** Leaflet is the leading open-source JavaScript library for mobile-friendly interactive maps. It has all the mapping features most developers ever need.

### 5.2.5 Hardware Requirements

When considering the hardware requirements for our GIS application, the choice of utilizing cloud services presents a strategic advantage. Cloud services, such as those provided by platforms like Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP), offer scalable and flexible infrastructure that can adapt to the changing demands of the application.

With cloud services, the hardware resources required to host the application can be provisioned dynamically based on usage, ensuring optimal performance during peak times and efficient resource utilization during periods of lower demand. This elasticity eliminates the need for substantial upfront investments in physical hardware and allows us to scale resources up or down as needed, resulting in cost savings and improved efficiency.

#### Docker Image Deployment and Resource Efficiency

In conjunction with cloud services, the deployment of Docker images further optimizes hardware resource utilization. Docker containers encapsulate applications along with their dependencies, enabling consistent deployment across different environments. By deploying applications as Docker images, we achieve resource efficiency, as each container operates in an isolated environment, utilizing only the resources required for the specific application.

### 5.2.6 Components Schema

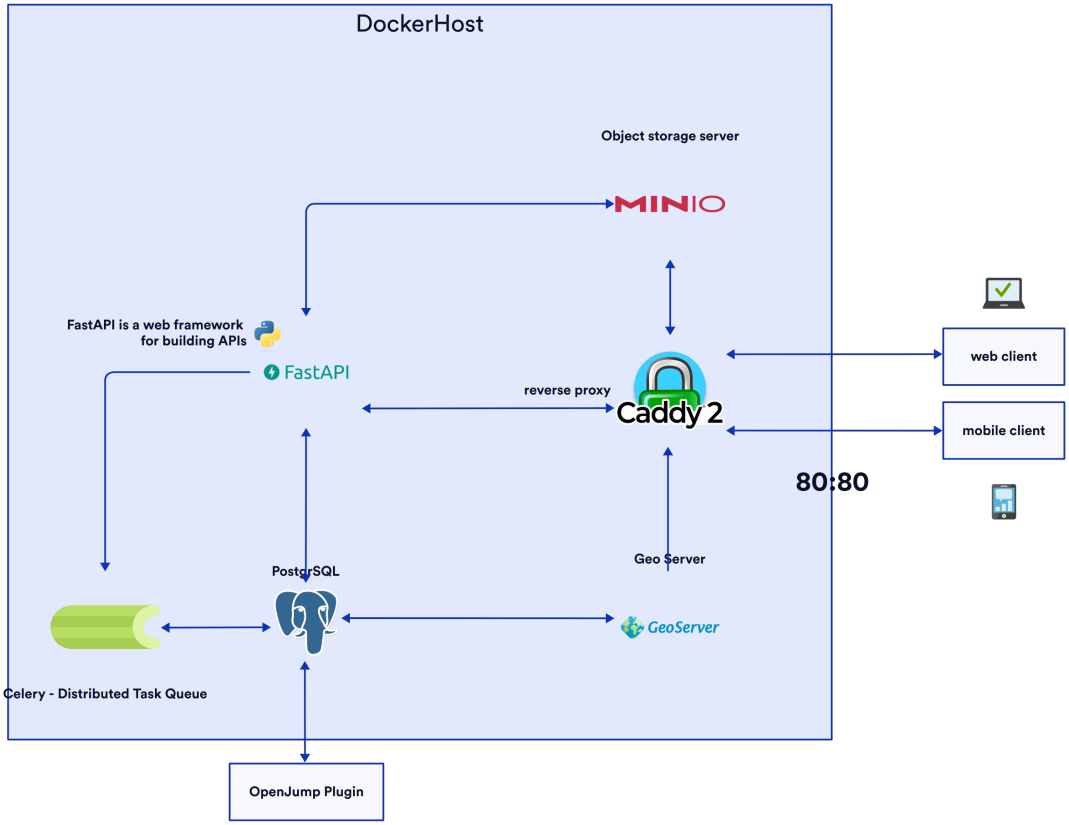


Figure 5.1: components schema

## 5.3 Database

### 5.3.1 Initialize Database

Given data for Particelle data are provided in cadaster mode that relies on local reference points, we need to transform them into a global reference system. To successfully carry out the conversion process, I meticulously followed a step-by-step procedure. Initially, I identified the nearest "punti fiduciali" in proximity to the specific geographic area of interest, which in this case was in the vicinity of Fano. These reference points held coordinates in both the Cassini Soldner and Gauss Braga coordinate reference systems (CRS). By leveraging these paired coordinates, I constructed an Affine Transformation, a mathematical model capable of accurately converting coordinates from the Cassini Soldner system to

the Gauss Boaga system. This transformation was then methodically applied to the target cadastral data, facilitating a seamless and accurate conversion to the desired Gauss Boaga CRS. The integration of these steps culminated in a robust and precise conversion process that was integral to the success of my research project. The "punti fiduciali" dataset provided in OpenJump format was pivotal in enabling the smooth execution of this technique, reaffirming its value in overcoming the challenges associated with coordinate system conversions.

The Algorithm used to convert the coordinates is the following:

**Algorithm 1** Calculate Transformation Matrix

**Require:** $primary, secondary$

1: $n \leftarrow$ shape of $primary[0]$
2: $pad \leftarrow$ function that adds a column of ones to a given array
3: $unpad \leftarrow$ function that removes the last column of a given array
4: $X \leftarrow pad(primary)$
5: $Y \leftarrow pad(secondary)$
6: $A, res, rank, s \leftarrow$ result of solving least squares problem $X * A = Y$
7: $transform \leftarrow$ function that applies the transformation matrix A to a given array
8: **return** $transform$

Now we're ready to create the database, we need to create a new database in PostgreSQL, and then we need to create the schema, the tables and the views. We use shp2sql to import the shapefiles into the database.

### 5.3.2 Database Schema

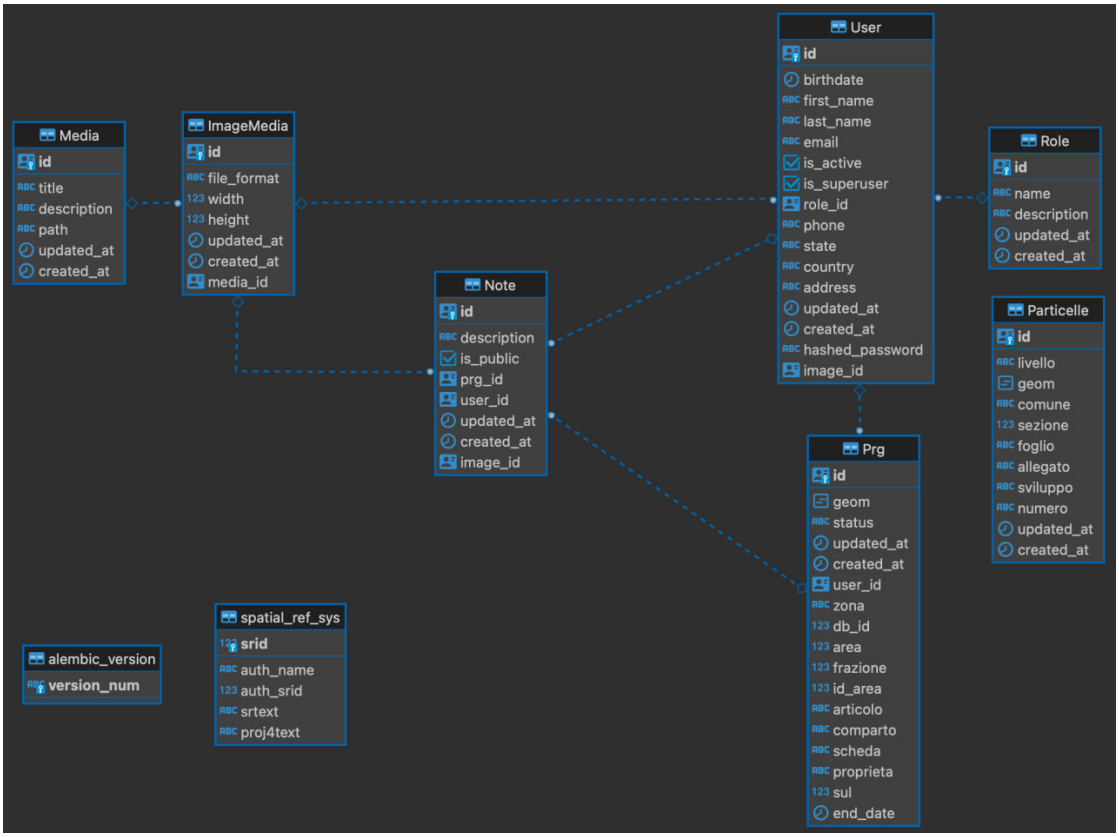


Figure 5.2: db schema

### 5.3.3 Description of Entities and Relationships

The relationships established between the tables in database schema play a crucial role in making the GIS project app work seamlessly. Let's delve deeper into how these relationships contribute to the functionality of the application:

#### ImageMedia and Media Relationship

The `ImageMedia` table contains information about image media files, while the `Media` table holds general media data. This relationship allows you to associate specific images with general media records. For example, you can link a particular image to a video or other media content. It enables the app to manage different types of media assets and efficiently retrieve associated images when needed.

#### Note, ImageMedia, Prg, and User Relationships

The `Note` table is at the core of the app, storing notes related to land usage. The relationships with `ImageMedia`, `Prg`, and `User` are fundamental for organizing and contextualizing the notes. The connection with `ImageMedia` allows users to associate images with their notes, enhancing the visual representation of the content. The link to `Prg` helps tie notes to specific land usage plans, enabling users to provide location-specific information and feedback. The relationship with `User` ensures that each note is attributed to a specific user, facilitating accountability and user-specific access to notes.

#### Prg and User Relationship

The `Prg` table represents urban recovery and redevelopment plans. The relationship with `User` is essential for assigning responsibility for the plans to specific users. This connection allows administrators and designated users to manage and update the details of the plans they are responsible for.

#### User and ImageMedia, Role Relationships

The `User` table represents app users, including administrators and regular users. The relationship with `ImageMedia` allows users to have profile images associated with their accounts. This enhances the user experience by enabling users to personalize their profiles and providing a visual identification. The connection with `Role` defines user roles and permissions, making it possible to differentiate between administrators and regular users. Role-based access control ensures that only authorized users can perform certain actions, enhancing security and data integrity.

#### Overall Functionality

These relationships collectively enable the GIS project app to provide the following functionalities:

- User Authentication and Authorization
- Land Usage Notes and Planning
- Media Management
- User Profiles

- Responsibility Assignment

In summary, the well-defined relationships in the database schema are the backbone of the GIS project app, enabling seamless interactions between different entities, data organization, user authentication, access control, and efficient management of land usage data for the city.

## 5.4 Functional Aspects of the System

### 5.4.1 WebGIS Application

Our web application is mainly composed of 5 parts:

1. **Displaying Parts and Variants**: To display public data to all users and provide technicians with access to manage them.
2. **Notes and Comments**: Users have the ability to write notes and comments for various variants or create a polygon to add notes to. Additionally, users can view all public notes.
3. **Creating Variants**: Technicians can draw new areas and incorporate them as variants. These variants remain inactive and are not visible to the public.
4. **Managing Variants**: Modifying the status of variants and making them visible to all users.
5. **Search and Print**: Searching for parts and printing the associated area with its details.

Explore the interface where parts and their variations are showcased, accessible to both the public and technicians:

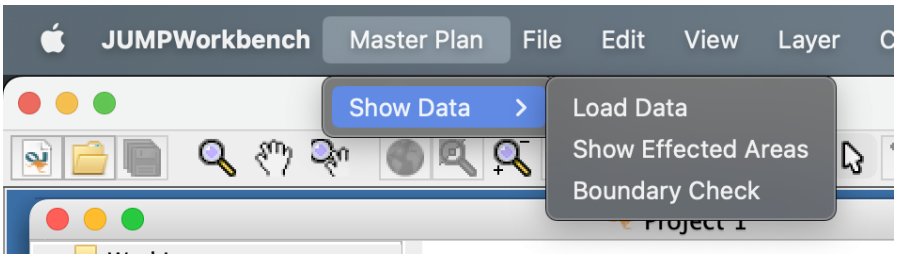


Figure 5.3: Visualizing parts and variants for public and technician access.

Discover how users interact with the system by writing notes and comments for different variants or creating polygons for additional context. Observe the display of public notes.

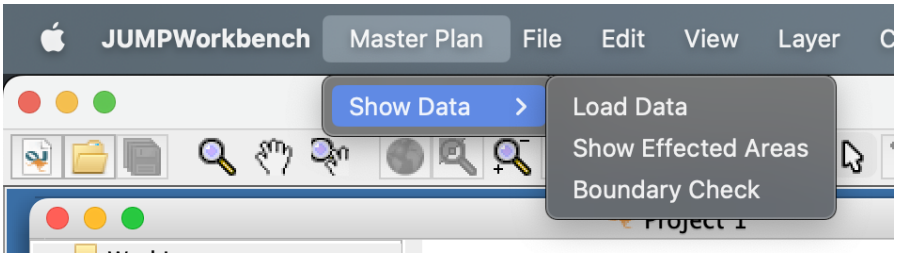


Figure 5.4: User interaction with notes, comments, and polygons, with public notes displayed.

Witness technicians drawing and incorporating new areas as variants within the system. These additions are initially hidden from public view.

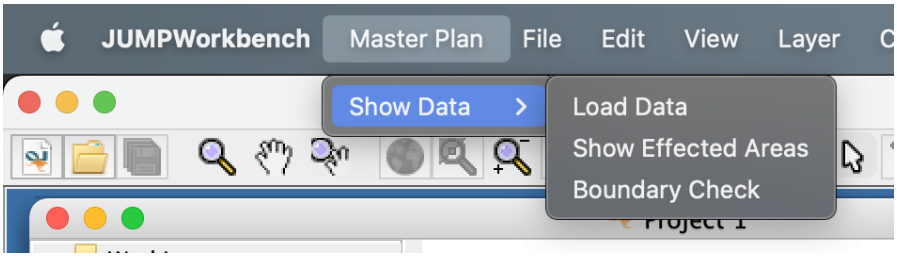


Figure 5.5: Technician adding new areas as variants, initially hidden from the public.

Observe the process of modifying the status of variants and making them visible to all users for effective management.

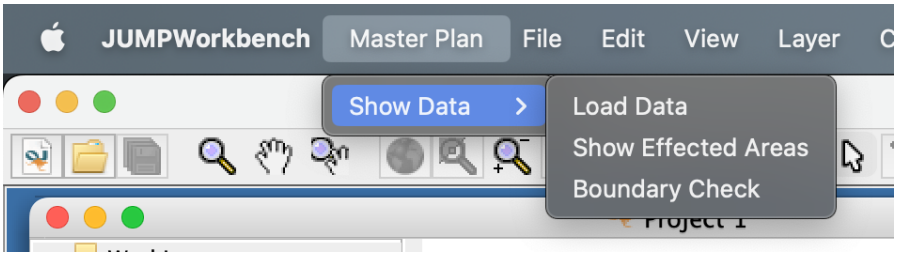


Figure 5.6: Administering variant status and making them visible to all users.

Explore the functionalities of searching for specific parts and printing detailed information along with their associated areas for convenient reference.

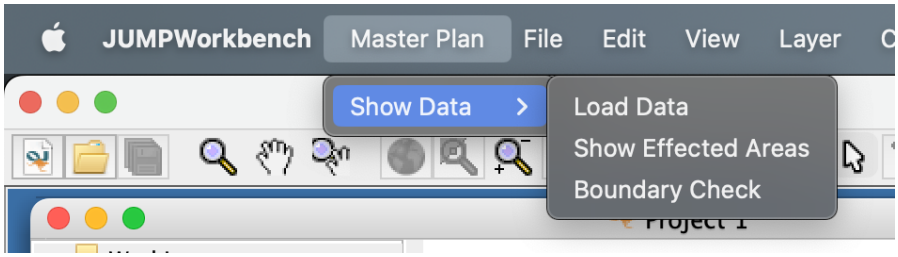


Figure 5.7: Utilizing search functionality and printing part details with associated areas.

### 5.4.2 OpenJUMP plugin

Our plugin is composed of 3 main parts:

1. **Load data from the database**: this part is responsible for loading the data from the database, the data are loaded in the form of layers, and the layers are loaded in the form of features.

2. **Show Effect Areas**: this part is responsible for showing the effect areas of the variants, the effect areas are the areas that are affected by the variant.

3. **Boundary**: this part is responsible for showing the boundary of the effect areas, checking if the boundary is correct, and showing the parcels that are excluded or partially included.

For each part, we have a menu item, and each menu item has a button that is responsible for activating the part.

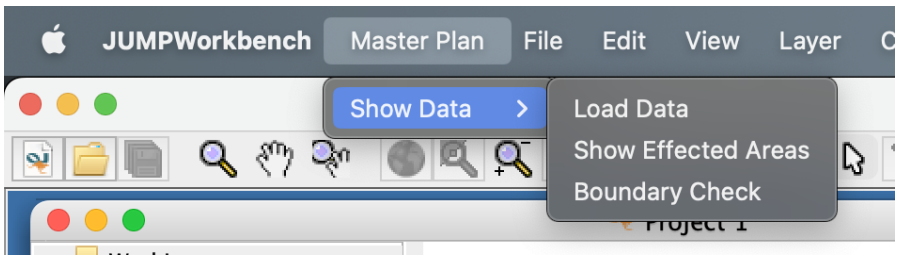


Figure 5.8: Menu of the plugin

After loading the data from the database, we can see the layers in the layer panel, and we can see the features in the map panel.

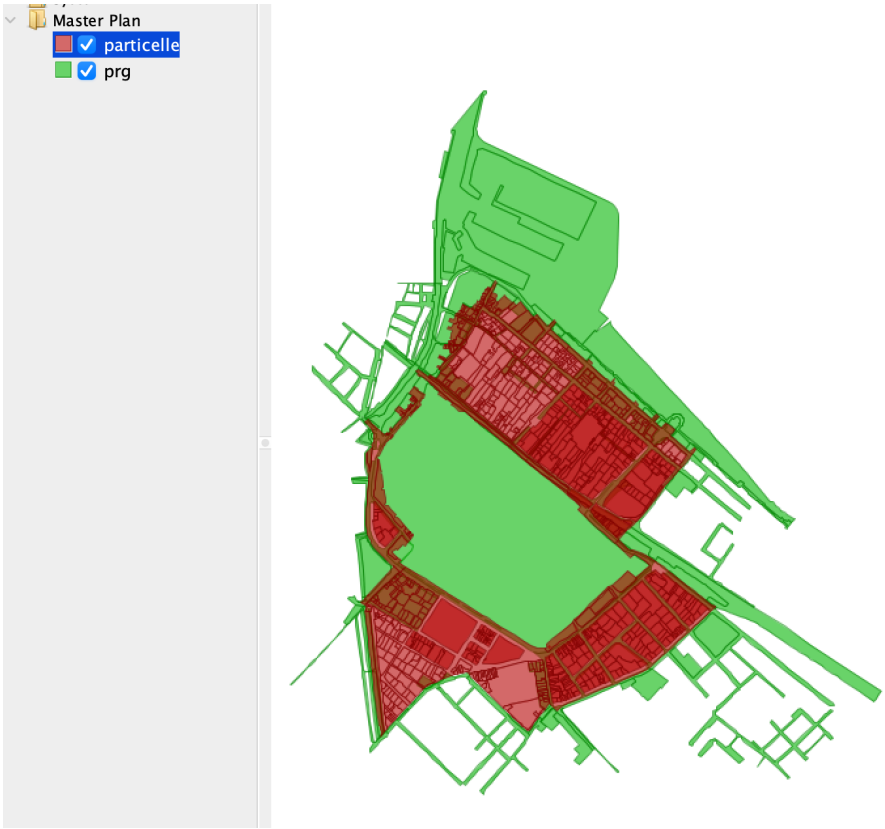


Figure 5.9: Load data from the database

For seeing the effect areas, we need to choose a variant, and then we can see the effect areas of the variant. We choose the variant from the drop down menu, these item id are the id of the variants in the database.

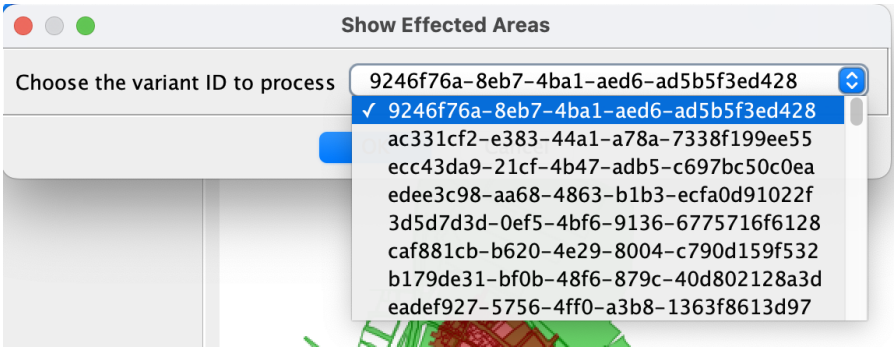


Figure 5.10: Show fffect areas - Choose a variant

After choosing the variant, we can see the effect areas of the variant. For simplicity, we show them under the variant layer and the layer name is `variant-variant_id`.


Figure 5.11: Show Effect Areas

In boundary part, we help the user to check if the boundary of the effect areas is correct or not. We show the parcels that are excluded or partially included in the effect areas.

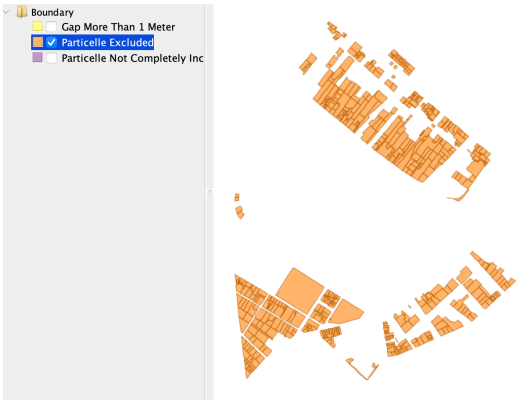


Figure 5.12: Parcels excluded

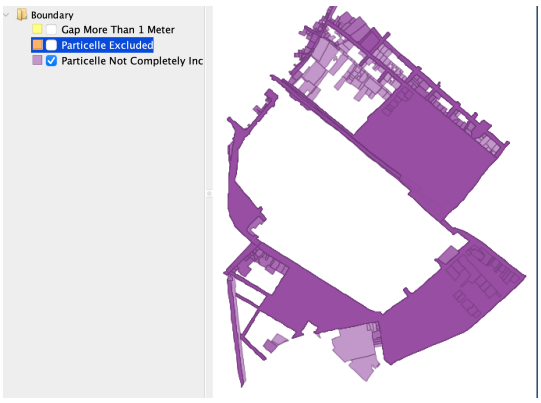


Figure 5.13: Parcels partially included

And we also show the gap between parcels and variant boundary if the gap is in threshold of 1 meter.


Figure 5.14: Gap between parcels that partially included

# 6 Project Guidelines

## 6.1 Deployment Plan

### 6.1.1 Gantt Chart

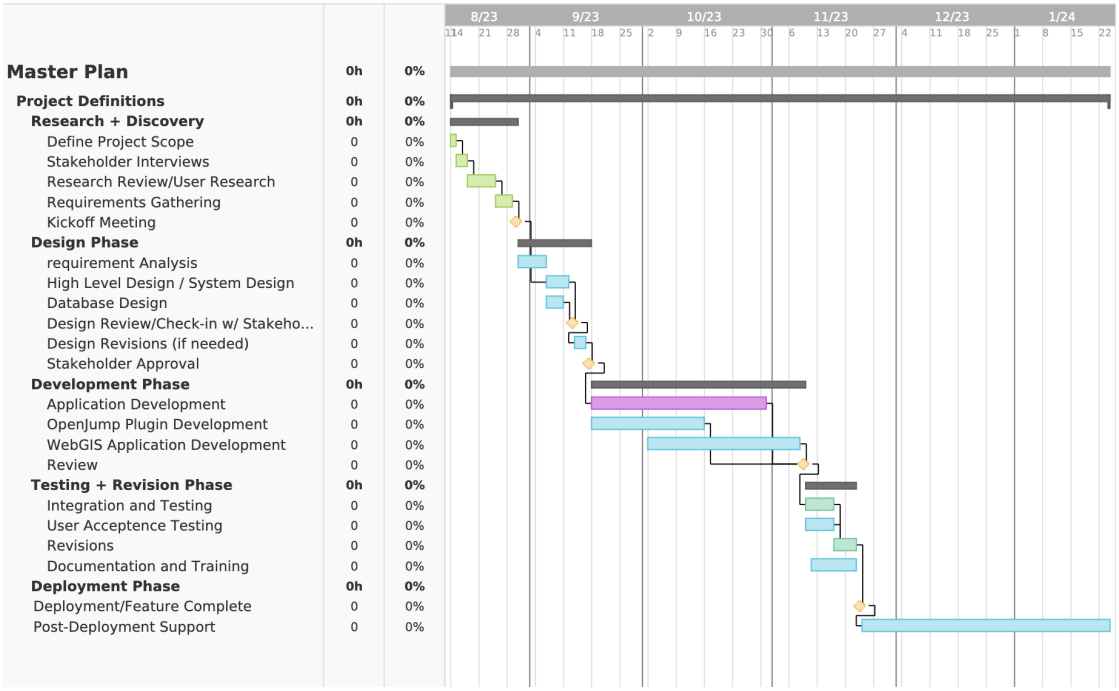


Figure 6.1: Gantt Chart

### 6.1.2 Description of Gantt Chart

The provided Gantt chart outlines the comprehensive timeline and sequence of tasks for the development of the GIS Master Plan Application. It serves as a visual representation of the project's progression, illustrating the major phases, activities, and their interdependencies. The Gantt chart begins with definig project scope, which entails initial planning and team coordination. It then segues into Requirements Analysis, System Design, and Database Design, where the project's fundamental groundwork is laid. Application Development, including the OpenJUMP Plugin and WebGIS Application, constitutes a significant portion of the timeline, culminating in Integration and Testing to ensure seamless operability. Subsequently, User Acceptance Testing, Documentation, and Training phases pave the way for a successful Deployment. Post-Deployment Support extends after deployment to address any issues and facilitate user adaptation. Finally, the project concludes with the Project Conclusion phase. This Gantt chart serves as a roadmap, enabling effective project management and tracking as the GIS Master Plan Application takes shape.

### 6.1.3 Delivery Time

!! TODO

### 6.1.4 Benefits Evaluation

This evaluation can be quantified through various metrics, including time saved in plan variant creation and approval, reduction in errors and inconsistencies, enhanced public participation, and increased efficiency in plan management. By streamlining workflows, providing real-time data access, and facilitating informed decision-making, the application will ultimately lead to more effective urban planning, reducing time-to-implementation and enabling better resource allocation.

### 6.1.5 Benefits Identification

Identifying benefits entails recognizing the tangible and intangible advantages that the GIS Master Plan Application will offer. For municipal planners,

the application's streamlined plan variant management will expedite decision-making, enhance collaboration, and enable informed urban development strategies. Citizens will gain access to user-friendly tools that empower them to visualize and understand the Master Plan's implications, fostering transparent engagement and promoting better-informed decisions. The application's capabilities to manage and analyze spatial data efficiently will lead to improved accuracy, reducing errors in plan management and minimizing resource wastage.

## 6.2 Economic Quantification of the Benefits

In the realm of project management for the GIS Master Plan Application, economic quantification of the benefits holds significant importance.

**Cost Savings and Efficiency**

One key economic benefit lies in cost savings and enhanced operational efficiency. The streamlined plan variant management facilitated by the application will lead to reduced manual effort, shortened approval cycles, and minimized errors. These efficiency gains translate into quantifiable cost reductions, including savings in personnel time, administrative overhead, and resource utilization. Moreover, improved accuracy in plan management will reduce the costs associated with rectifying errors and addressing inconsistencies that might arise from traditional manual processes.

**Enhanced Civic Engagement and Decision-Making**

The application's capabilities to promote transparent public engagement and provide citizens with user-friendly tools for understanding the Master Plan's implications have significant economic implications as well. Informed citizen participation can lead to more effective land use decisions, reducing the potential for legal disputes and costly reversals in development projects. Additionally, by fostering greater community understanding and collaboration, the application can potentially attract investments and development opportunities, thus contributing positively to the local economy.

**Time-to-Implementation Reduction**

Another economic advantage lies in the reduction of time-to-implementation

for plan variants. The streamlined process enabled by the application can lead to faster plan approval, subsequently accelerating the execution of development projects. This reduced time frame results in faster revenue generation for commercial and residential projects, positively impacting the economic growth of the municipality.

**Data-Driven Decision-Making**

The application's ability to provide real-time access to accurate spatial data supports data-driven decision-making. This leads to better resource allocation, optimized land use strategies, and improved urban planning initiatives. By maximizing the efficiency of projects and minimizing resource wastage, the application contributes to economic sustainability and growth.

## 6.3 Cost Evaluation

This process involves a comprehensive analysis of both project costs and ongoing operational expenses, ensuring transparency and informed decision-making regarding resource allocation.

### 6.3.1 Project Costs

These costs include personnel salaries, software and hardware acquisition, third-party services, licensing fees, development tools, and any associated training expenses. Additionally, costs related to quality assurance and testing, documentation, user interface design, and project management tools are considered. Accurate project cost evaluation provides a clear understanding of the investment required to bring the application to fruition, enabling effective budget planning and resource allocation.

Development Team, We'll need developers with expertise in Django, Python, PostgreSQL, PostGIS, Leaflet, Geoserver, Java, and JTS. The estimated costs are writen per year but as we expected this projects needs around 3.5 month of development.

**Django Backend Developers (2-3)**: $60,000–$120,000 per developer per year.

**Front-end Developers (2-3)**: $60,000–$120,000 per developer per year.

**GIS Specialists (1-2)**: $80,000–$150,000 per specialist per year.

**Java Developers (for OpenJump)**: $70,000–$130,000 per developer per year.

**Database**: PostgreSQL (PostGIS) is open-source, so there are no licensing costs, but you may need a DBA (Database Administrator) to optimize the database. A DBA's salary can range from $80,000 to 150,000 per year.

**Leaflet and Geoserver**: These are open-source tools, so there are no licensing costs. However, we may need experts to configure and maintain them, which can add to personnel costs.

**Mobile App Development**: If we want a mobile app that's compatible with both Android and iOS, we'll need mobile app developers (2-3) with expertise in React Native or Flutter. Each developer's salary may range from $60,000 to $120,000 per year.

**Project Management and Quality Assurance**: We should allocate a budget for project management and quality assurance (testing and debugging). Depending on the project's scale, this could range from $20,000 to $60,000 per year.

**OpenJump Plugin**: Development and integration of the OpenJump plugin with Java and JTS could cost an additional $20,000 to 50,000, depending on complexity.

**Infrastructure**: Hosting costs depend on your project's scale and traffic. Consider cloud providers like AWS, Azure, or Google Cloud. Costs may vary widely but could range from $1,000 to $5,000 per month.

**Contingency**: It's advisable to have a contingency fund of 10-20% of the total estimated budget to cover unexpected expenses.

**Maintenance and Updates**: Don't forget to budget for ongoing maintenance and updates, which could be around 20-30% of the initial development cost annually.

| Expense Category | Estimated Cost Range |
|---|---|
| Personnel | $200,000–$450,000 |
| Infrastructure | $12,000–$60,000 |
| Contingency | 10–20% of total |

Table 6.1: Initial Development Costs (First Year)

### 6.3.2 Running Costs

The running costs encompass the ongoing operational expenses to maintain and sustain the GIS Master Plan Application. These costs include hosting and

server maintenance, cloud services, technical support, software updates, security measures, and system monitoring. User training, system administration, and regular data updates also contribute to running costs. By anticipating these ongoing expenses, project stakeholders can ensure the application's continuous functionality, user satisfaction, and alignment with the allocated budget.

| **Expense Category** | **Estimated Cost Range** |
|---|---|
| Maintenance and Updates | 20–30% of initial development cost |
| Hosting | $12,000–$60,000 |
| Database Administration | $80,000–$150,000 (if needed) |

Table 6.2: Ongoing Annual Costs

# Acknowledgments