POLITECNICO
MILANO 1863

Design Document

Group name: PolimiGeo

Project Subject: Flood Risk Assessment

Professor Giovanni Quattrocchi

Professor Vasil Yordanov

Seyed Mohammad Moein Peyghambar Zadeh: 10921320

Saeed Mehdizadeh: 10891394

Hadi Kheiri Gharajeh: 10946669

Vanessa Goletti: 10706682

Table of Contents

1. Introduction:

1.1. Context and motivations

Flood risk management is one of the most common and devastating natural disasters, causing significant economic losses, environmental damage, and loss of life. Effective flood risk management is essential to mitigate these impacts, requiring accurate data collection, analysis, and timely dissemination of information to stakeholders.

Combining geographic information systems (GIS), remote sensing, and spatial data analysis, plays a critical role in understanding and managing flood risks. By leveraging geoinformatics, it is possible to visualize flood-prone areas, model flood scenarios, and predict future risks based on various parameters and environmental conditions.

The primary motivation for developing this flood risk management software is to provide a comprehensive and user-friendly tool that enables stakeholders to make informed decisions regarding flood mitigation and response.

The key stakeholders for this software are government agencies, researchers and scientists, emergency responders, general publics and environmental and non govermental organizations.

The development and implementation of this flood risk management software are expected to deliver several significant benefits including Improved Flood Preparedness, Reduced Economic Losses, Enhanced Collaboration, Data-Driven Policy Making.

1.2. Definitions, Acronyms, and Abbreviations

Definitions:

1. **Flood Risk Management:** The process of assessing, planning, and implementing measures to reduce the impact of flooding.
1. **Geoinformatics:** A field of science that uses data and tools from geographic information systems (GIS), remote sensing, and spatial analysis to solve complex geographic and environmental problems.
2. **Dashboard:** A user interface that provides a visual summary of key metrics and data points.
3. **GIS (Geographic Information System):** A framework for gathering, managing, and analyzing spatial and geographic data.
4. **Spatial Data:** Information about the physical location and shape of geographic features and the relationships between them.

5. **REST API (Representational State Transfer Application Programming Interface):** A set of rules that allows for interaction with web services using HTTP requests.

Acronyms:

1. **GIS:** Geographic Information System
2. **ISPRA:** Istituto Superiore per la Protezione e la Ricerca Ambientale (Institute for Environmental Protection and Research, Italy)
3. **IdroGEO:** A specific portal managed by ISPRA for geospatial data related to hydrological risks
4. **PostGIS:** An extension for PostgreSQL that adds support for geographic objects
5. **API:** Application Programming Interface
6. **HTML:** HyperText Markup Language
6. **HTTP:** HyperText Transfer Protocol
7. **HTTPS:** HyperText Transfer Protocol Secure
8. **CSV:** Comma-Separated Values
9. **JSON:** JavaScript Object Notation
10. **SQL:** Structured Query Language
11. **Uptime:** A measure of system reliability, indicating the amount of time a system is operational

Abbreviations

1. **UI:** User Interface
2. **UX:** User Experience
3. **VSCode:** Visual Studio Code
4. **HTTPS:** HyperText Transfer Protocol Secure
5. **IP:** Internet Protocol

1.3. Purpose

This software engineering project's main purpose is to develop an intuitive and interactive system that lets users identify and assess the risk of flooding for specific areas, families, heritage sites, and buildings. This system will provide essential information to support effective flood risk management, preparedness, and mitigation efforts. The objective of this system is to create a user-friendly interface that visually represents flood risk data for various areas, highlighting vulnerable families, heritage sites, and buildings. It also provides critical information to residents, local authorities, emergency responders, and

conservationists regarding flood-prone areas and at-risk assets. Moreover, it equips decision-makers with accurate and detailed flood risk data to inform planning, resource allocation, and emergency response strategies and raises awareness among the public about flood risks and encourage proactive measures to mitigate potential damage.
The development of this flood risk management system is significant for several reasons including protecting lives and property, enhanced preparedness, conservation of heritage, community engagement.

1.4. Scope and Limitations:

The **scope** of this flood risk management system is composed of the following key functionalities and components:

1. user registration, login, profile management, secure user authentication and authorization.
2. An interactive dashboard providing access to flood risk data, and visualization tools to display flood-prone areas, vulnerable families, heritage sites, and buildings.
3. Integration of geospatial data to create detailed flood risk maps, and interactive map features allowing users to explore different regions and identify at-risk areas.
4. Generation of detailed reports on flood risks for specific areas, families, heritage sites, and buildings, and option for users to download reports in various formats (e.g., excel, CSV).
5. Tools allowing users to customize flood risk analysis parameters (e.g., rainfall intensity, river levels), and dynamic updating of risk assessments based on user-defined parameters.
6. Encryption of data in transit and at rest to protect sensitive information, and implementation of security best practices to ensure system integrity and user privacy.

While the flood risk management system aims to provide comprehensive and accurate information, there are several **limitations** to be aware of:

1. The system's accuracy is dependent on the **quality and timeliness of data** from external sources (e.g., IdroGEO, ISPRA), and limited data availability or inaccuracies in source data may impact the reliability of the risk assessments.
2. The system is designed primarily for regions covered by the data sources used (e.g., areas monitored by ISPRA), and some regions may not have **sufficient data coverage** to provide detailed risk assessments.

3. The accuracy of customized risk assessments based on user-defined parameters depends on the **validity and relevance of the input data**, and users may require training or guidance to set meaningful parameters for accurate risk assessments.
4. The performance of the system (e.g., response times) may vary based on server load, network conditions, user activity. **High traffic volumes or complex queries** may impact system responsiveness.
5. The system **focuses on flood risk visualization and reporting**; it does not include features for direct flood response management (e.g., emergency dispatching), and advanced predictive modeling and simulations beyond basic risk assessments are not within the current scope.

2. Architectural Design

The architectural design of the flood risk management system is structured to provide a robust, scalable, and secure platform for visualizing flood risks and generating detailed reports. The system integrates various data sources, processes spatial data, and delivers interactive user interfaces. This section provides an overview of the key architectural components and their interactions.

2.1. Overview

The system architecture is composed of several layers and components that work together to achieve the functional and non-functional requirements. The main components are the presentation layer, application layer, data layer and external services.

Presentation layer:

Web Application Interface:
Built using HTML to provide a responsive and interactive user experience and contain the user dashboard for accessing flood risk maps, adjusting parameters, and viewing reports.
Interactive Map Visualization:
Utilizes libraries such as Folium and Plotly to render dynamic maps and data visualizations.

Application Layer

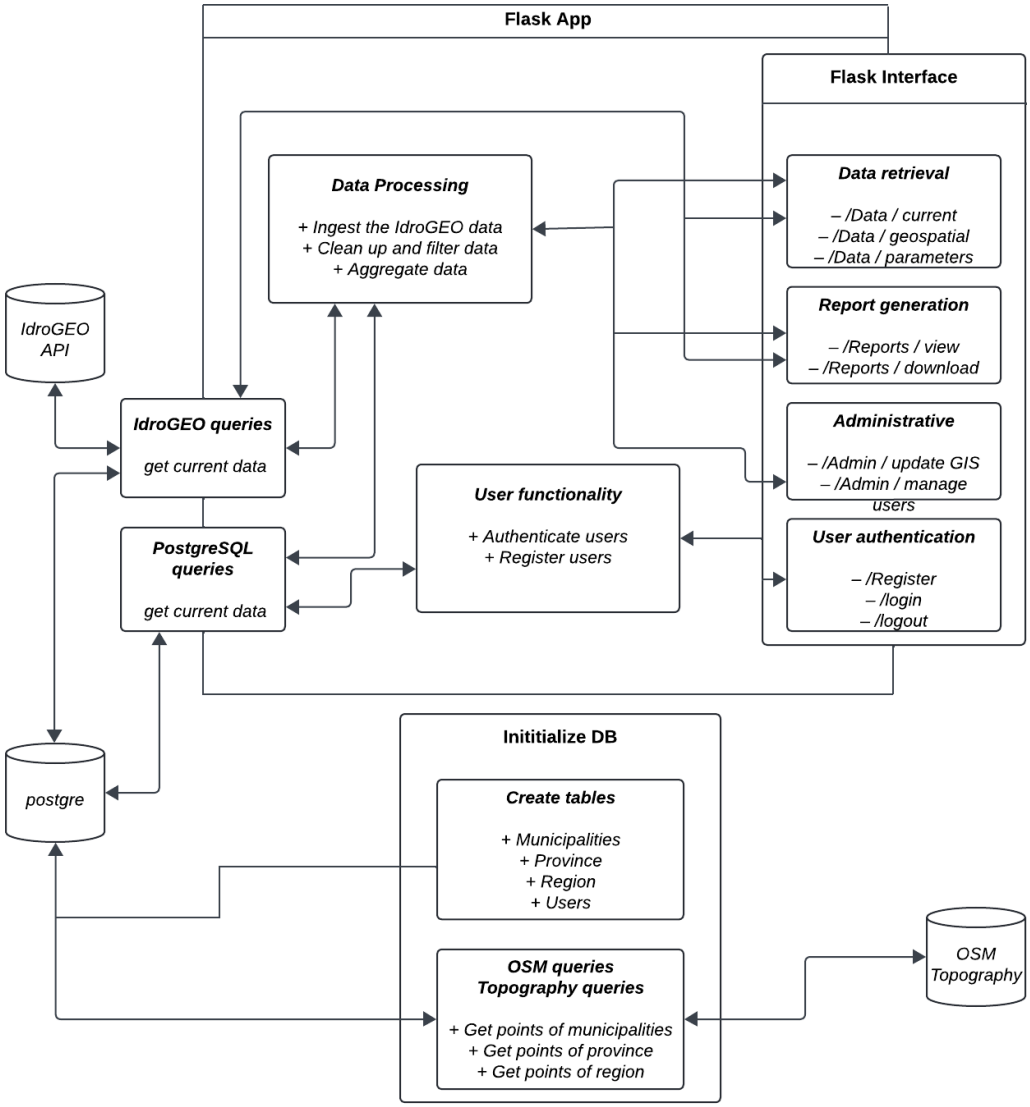
Backend API:
Developed using Flask, exposing RESTful endpoints for data retrieval and processing and handles user authentication, data requests, and report generation.
Data Processing Modules:
Utilizes Python libraries such as Pandas and GeoPandas for data cleaning, transformation, and analysis and includes real-time data integration and parameter adjustment functionalities.

Data Layer

Spatial Data Storage:
Uses PostgreSQL with the PostGIS extension to store and manage spatial data efficiently and handles geospatial queries and data indexing.
User and Application Data:
Stores user profiles, preferences, and system configuration data in a relational database.

External Services
Data Sources:
Integrates data from external sources like the IdroGEO website managed by ISPRA, and periodically fetches and updates data to ensure the latest information is available.
Security Services:
Implements encrypted communication using HTTPS and uses secure authentication protocols to protect user data.

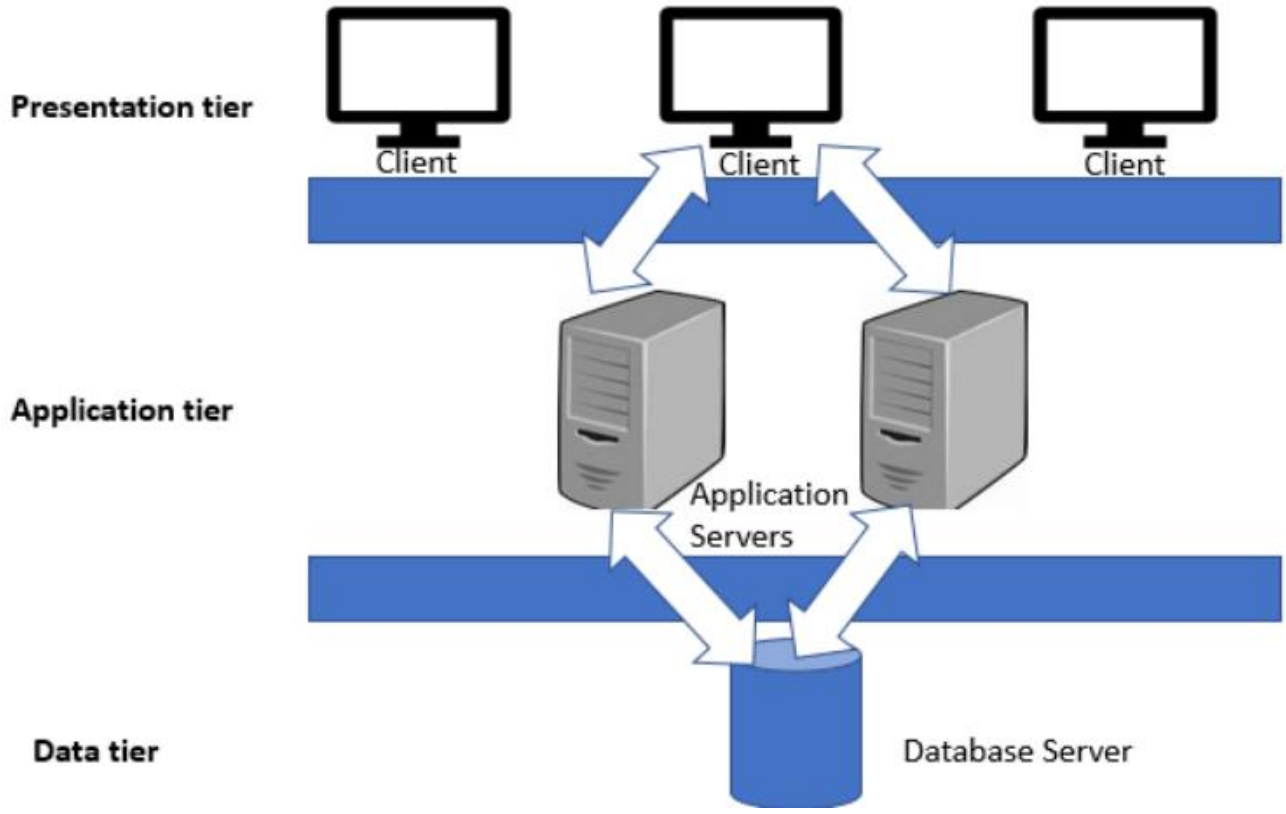
2.2. Component diagram



3. Software design

The software's structure will be organized into three logical and physical computing tiers or system layers, which are:

- *Presentation Tier also called Web Server*
- *Application Tier also called Logical Server*
- *Data Tier also called Database Server*



3.1 Presentation Tier (Web Server)

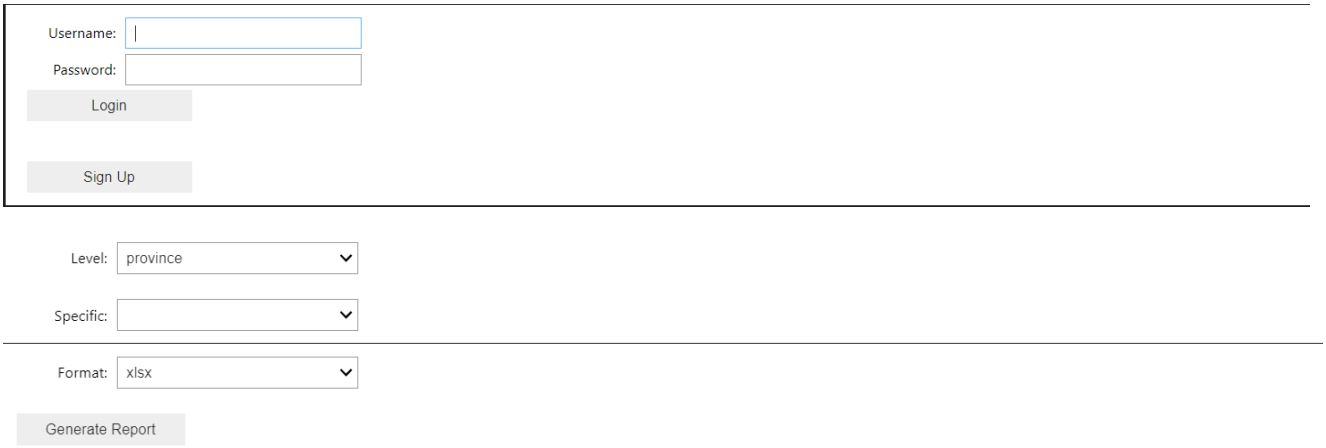
The Presentation Tier handles user interactions and presents data in a user-friendly manner. It includes the user interface components and the client-side logic that runs in the browser.

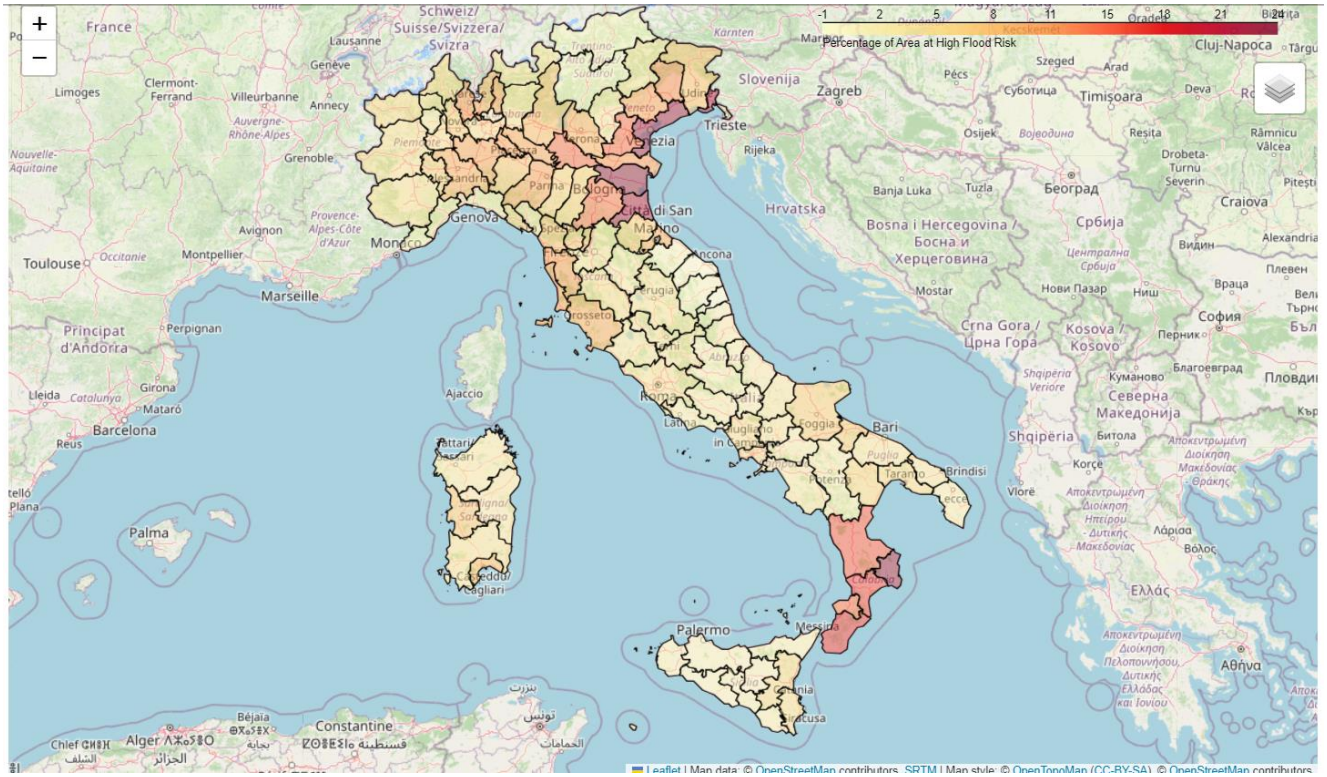
Components

- **User Interface (UI)**: The UI is built using HTML, CSS, and JavaScript, providing a responsive and interactive experience for users.
- **Frameworks and Libraries**: Utilizes front-end frameworks like Flask and Jupyter Notebook to create dynamic and scalable interfaces.

- **Routing**: Manages navigation between different pages or views within the application.

- **Wireframe** A wireframe illustrating the layout and design of the key components within the Presentation Tier.





3. 2 Application Tier (Logical Server)

The Application Tier contains business logic and server-side functionality. It processes requests from the Presentation Tier, performs operations, and interacts with the Data Tier to fetch or store data.

Components

- **REST APIs**: The server-side APIs that handle HTTP requests from the client, process them, and return responses.
- **Business Logic**: Implements the core functionality and rules of the application.
- **Authentication and Authorization**: Manages user authentication and ensures secure access to resources.

API Endpoints: Detailed list of the primary API endpoints and their functionalities:

- **User Authentication**
  - **POST /api/v1/auth/register**: Register a new user.
  - **POST /api/v1/auth/login**: Authenticate an existing user and return a token.

- **Projects Management**
  - **GET /api/v1/projects**: Retrieve all projects for the authenticated user.
  - **POST /api/v1/projects**: Create a new project.
  - **PUT /api/v1/projects/{id}**: Update a specific project.
  - **DELETE /api/v1/projects/{id}**: Delete a specific project.
- **Tasks Management**
  - **GET /api/v1/projects/{id}/tasks**: Retrieve all tasks for a specific project.
  - **POST /api/v1/projects/{id}/tasks**: Create a new task in a specific project.
  - **PUT /api/v1/tasks/{id}**: Update a specific task.
  - **DELETE /api/v1/tasks/{id}**: Delete a specific task.

3.3 Data Tier (Database Server)

The Data Tier is responsible for data storage, retrieval, and management. It ensures data integrity, consistency, and availability.

Components:

- **Database Management System (DBMS)**: The software that handles database operations (PostgreSQL).
- **Schema Design**: The structure of the database, including tables, relationships, and constraints.
- **Data Access Layer**: The layer that interfaces with the DBMS to perform CRUD (Create, Read, Update, Delete) operations.

Database Schema The schema of the primary tables in the database:

- **Users Table**: Stores information about application users.
  - **Columns**: UserID (Primary Key), Username, Password, role.
- **Municipalities/ Provience / Region**: Stores location-related information for flood risk assessments.
  - **Columns**: Flood Risk Related columns

Data Access Layer :
The code snippets or pseudocode showing how the application interacts with the database:

4. Implementation and test plan

The implementation and test plan outlines the steps necessary to deploy and verify the application's functionality.

4.1 Implementation Plan

Environment Setup

- Configure development and production environments.
- Install necessary software dependencies and PostgreSQL database.

Code Deployment

- Deploy the back-end server with REST API endpoints.
- Deploy the front-end application using Flask and Jupyter Notebook.

Database Migration

- Run migration scripts to set up the database schema.
- Populate the database with initial data if necessary.

Integration

- Integrate front-end and back-end components for seamless communication.

4.2 Test Plan

Unit Testing

- Test individual components to ensure they perform as expected.

Integration Testing

- Verify interactions between different modules and services.

System Testing

- Conduct end-to-end testing of the entire application.

Performance Testing

- Assess application performance under various loads.

User Acceptance Testing (UAT)

- Gather feedback from end users and make necessary adjustments.

Regression Testing

- Re-test functionalities after any code changes to ensure stability.

4.3 Continuous Integration/Continuous Deployment (CI/CD)

Automation Tools

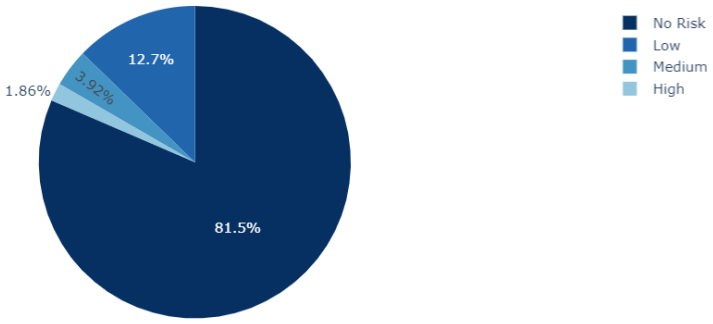
- Use CI/CD tools like Jenkins or GitHub Actions to automate build, test, and deployment processes.

Pipeline Configuration

- Set up pipelines to automatically run tests and deploy the application.


0 2 4 6 9 11 13 15 17
Percentage of Area at High Flood Risk
Lombardia
Piacenza
Novara
Schweiz/
Suisse/Svizzera/
Svizra
France
Lausanne
Genève
Clermont-
Ferrand
Limoges
Villeurbanne
Annecy
Grenoble
Toulouse
Montpellier
Avignon
Marseille
Monaco
Perpignan
Principat
d'Andorra
Girona
Barcelona
Palma
Alessandria
Genova
Parma
Bologna
Venezia
La Spezia
Firenze
Città di San
Marino
Ancona
Perugia
Grosseto
Terni
Roma
Latina
Italia
Foggia
Bari
Taranto
Brindisi
Potenza
Lecce
Ajaccio
Sardigna/
Sardegna
Casteddu/
Cagliari
Palermo
Messina
Catania
Siracusa
Sicilia
Calabria
Trieste
Udine
Slovenija
Zagreb
Hrvatska
Rijeka
Bosna i Hercegovina
Београд
Србија
Crna Gora
Kosovo
Shqipëria
Софија
Ελλάς
Αθήνα
Timișoara
Cluj-Napoca
Craiova
Alger
Constantine

Population Distribution at Different Flood Risk Levels in the region of Lombardia



index	25
Name of the Province	Como
Surface area of areas with low flood risk (%)	11.376
Surface area of areas with medium flood risk (%)	10.068
Surface area of areas with high flood risk (%)	9.193
Cultural heritage at low flood risk (n.)	392
Cultural heritage at medium flood risk (n.)	243
Cultural heritage at high flood risk (n.)	144
Cultural heritage at low flood risk (%)	26.739
Cultural heritage at medium flood risk (%)	16.576
Cultural heritage at high flood risk (%)	9.823
Province Code	13
Region Code	3
Buildings at low flood risk (n.)	10703
Buildings at medium flood risk (n.)	3057
Buildings at high flood risk (n.)	1694
Buildings - 2011 Census (n.)	141890
Buildings at low flood risk (%)	7.543
Buildings at medium flood risk (%)	2.154
Buildings at high flood risk (%)	1.194
Families at low flood risk (n.)	16227
Families at medium flood risk (n.)	4475
Families at high flood risk (n.)	1847
Families - 2011 Census (n.)	245455
Families at low flood risk (%)	6.611
Families at medium flood risk (%)	1.823
Families at high flood risk (%)	0.752
id	12
Local business units at low flood risk (n.)	4734
Local business units at medium flood risk (n.)	1728
Local business units at high flood risk (n.)	403
Local business units - 2011 Census (n.)	50075
Local business units at low flood risk (%)	9.454
Local business units at medium flood risk (%)	3.451
Local business units at high flood risk (%)	0.805
Population at low flood risk (no. of inhabitants)	36299
Population at medium flood risk (no. of inhabitants)	9839
Population at high flood risk (no. of inhabitants)	4205
Resident population - 2011 Census (no. of inhabitants)	586735
Population at low flood risk (%)	6.187
Population at medium flood risk (%)	1.677
Population at high flood risk (%)	0.717

5. Bibliography

- **PostgreSQL Documentation**

- PostgreSQL Global Development Group. "PostgreSQL 14.0 Documentation." Accessed June 2024. Available at: https://www.postgresql.org/docs/14/

- **Flask Documentation**

  Pallets Projects. "Flask Documentation." Accessed June 2024. Available at: https://flask.palletsprojects.com/

- **Jupyter Notebook Documentation**

  Project Jupyter. "Jupyter Notebook Documentation." Accessed June 2024. Available at: https://jupyter-notebook.readthedocs.io/

- **REST API Design**

  Richardson, Leonard. "RESTful Web APIs." O'Reilly Media, 2013.

- **Web Application Development**

  Krasner, Glenn E., and Stephen T. Pope. "A Description of the Model-View-Controller User Interface Paradigm in the Smalltalk-80 System." Journal of Object Oriented Programming, 1988.

- **Front-End Development**

  Duckett, Jon. "HTML and CSS: Design and Build Websites." John Wiley & Sons, 2011.

- **Software Testing**

Ammann, Paul, and Jeff Offutt. "Introduction to Software Testing." Cambridge University Press, 2016

- **Continuous Integration/Continuous Deployment (CI/CD)**

  Humble, Jez, and David Farley. "Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation." Addison-Wesley, 2010