POLITECNICO
MILANO 1863

Design Document

An interactive application to support disaster management activities during the prevention and preparedness phase

Junjie Mu, Chaotung Wu, Jiayi Su, Jianwei Deng

Deliverable:	DD
Title:	Design Document
Authors:	Junjie Mu, Chaotung Wu, Jiayi Su, Jianwei Deng
Version:	2.0
Date:	May 28, 2024
Download page:	https://github.com/Junjie-Mu/SE4GEO
Copyright:	Copyright © 2024, M.W.S.D – All rights reserved

目录

List of Tables

List of figures

1. Introduction

1.1 Design Document

Before starting to code and implement the software, an important step is to have a clear understanding of the design goals that our Web application will achieve. All of these goals should be established in this particular design document.

The key reason for writing this document is to define the structure and organization that will continue throughout the creation period until the software project is finalized. This is an important guide document for the engineering team, which they can rely on for any subsequent steps.

A software design document is a technical description that serves both the developer and the client. The developer's goal is to meet the customer's goals, but at the same time help themselves provide an efficient workflow.

It is important to emphasize the fact that the Design Document is built upon the (RASD) Requirements Analysis Specification Document, written by Junjie Mu, Chaotung Wu, Jiayi Su, Jianwei Deng. You can find this document in the following GitHub link.

The connection between the above-mentioned documents helps us to be careful with not avoiding any important functionality.

The characteristics specified would not be changed or described in another way but will remain an implicit knowledge for the development team. At the very least, it should be a description of the application, criteria for completion and milestones.

More specifically it will include the following characteristics:

(1) Project Database:

This is the most important part of the project we are developing. Database design can be defined as a collection of tasks or processes that enhance the designing, development, and maintenance of data management system. With other words, it is the most complex part where the designer determines what data must be stored and how the data elements are connected and interacting with each other.

(2) System Overview:

This section will provide a general description of the structure and functionality of the software system.

(3) Software Structure:

Our software, as a traditional client-server application, will be established as a three-tier architecture. This structure will make possible the interaction between the client and the server, considering both static and dynamic units. According to this architecture the application is organized into three logical and physical layers or tiers, which are:

• Presentation Server

• Application Server

• Database Server

Further details will be explained in the corresponding section in the body of the document.

(4) User Case Application:

This section will include the use cases and requirements map for each component of the software. Further details are prementioned in the RASD Document.

(5) Organization:

This is a key ingredient that makes the project a successful one and as mentioned above helps the design team to easily overcome confusions and fails. The development team is composed of 4 students, who equally participate in each step of the project.

1.2 Product Description

The purpose of developing this product is to inform about visualize and elaborate the exposure at a specific Italian administrative level by means of an Open-Source well application. This web-application will allow users to query, visualize, analyze data, and save their analysis of the chosen data. On the website the user will be able to find general information regarding this environmental issue.

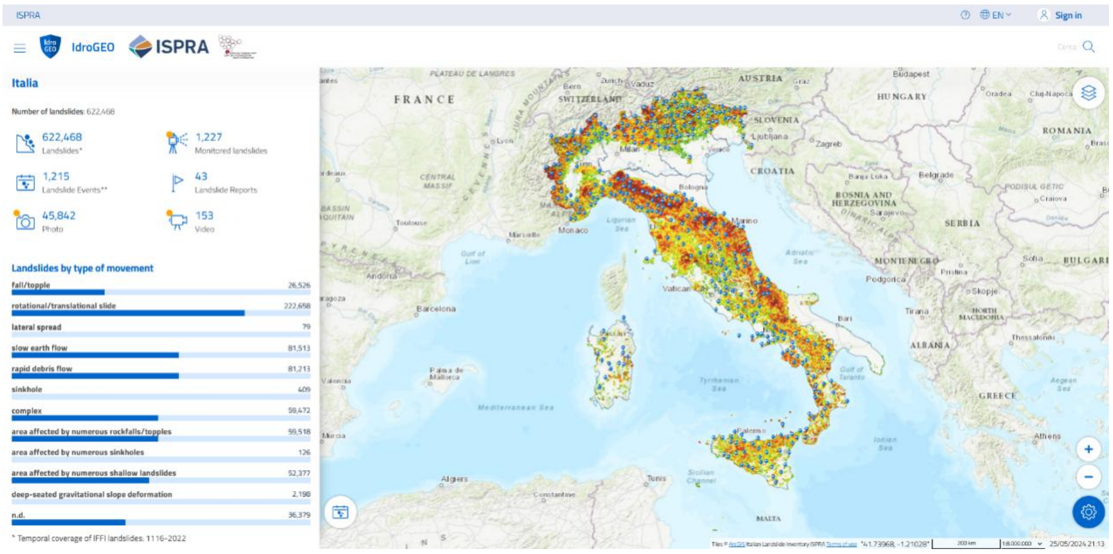
2. Project Database

2.1 ISPRA

The project database for mapping flood and landslide hazards in Italy leverages the IdroGEO API and ISPRA's open data to provide an interactive and comprehensive tool for visualizing geological risks. The IdroGEO API and ISPRA's open data initiatives represent significant advancements in the accessibility and usability of environmental data. By providing detailed and up-to-date information on geological hazards and other environmental parameters, these tools enable better decision-making and enhance our understanding of environmental dynamics.

Information on ISPRA datasets

Background Geodata



Italia

Number of landslides: 622,468

622,468 Landslides*	1,227 Monitored landslides
1,215 Landslide Events**	43 Landslide Reports
45,842 Photo	153 Video

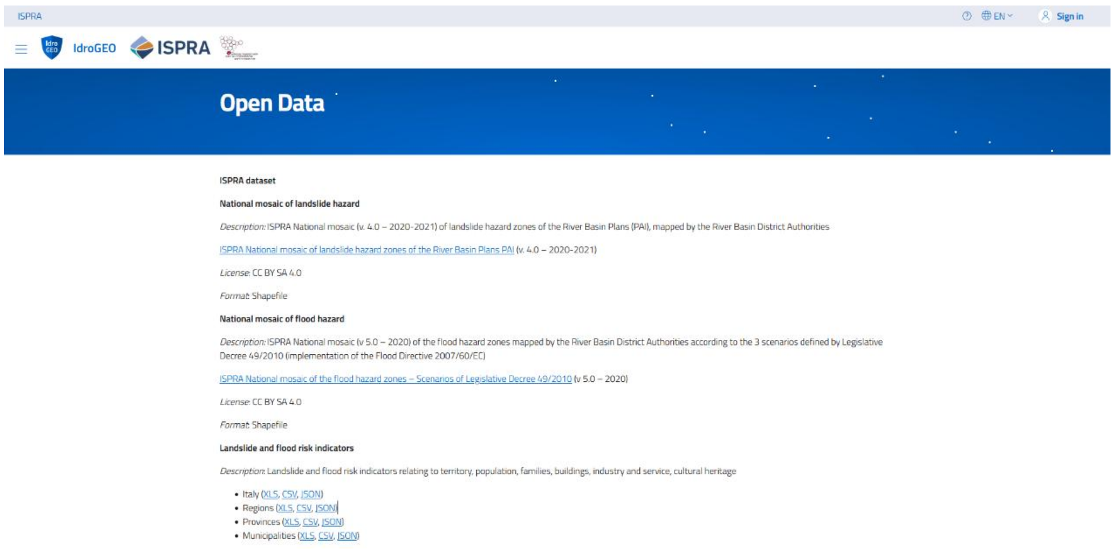
Landslides by type of movement

fall/topple	26,526
rotational/translational slide	222,658
lateral spread	79
slow earth flow	81,513
rapid debris flow	81,213
sinkhole	409
complex	59,472
area affected by numerous rockfalls/topples	59,518
area affected by numerous sinkholes	126
area affected by numerous shallow landslides	52,377
deep-seated gravitational slope deformation	2,198
n.d.	36,379

* Temporal coverage of IFFI landslides: 1116-2022

Figure 1 The interface of Italian Landslide Inventory IFFI

(Resource: https://idrogeo.isprambiente.it/app/iffi)



Open Data

ISPRA dataset

National mosaic of landslide hazard

Description: ISPRA National mosaic (v. 4.0 – 2020-2021) of landslide hazard zones of the River Basin Plans (PAI), mapped by the River Basin District Authorities

ISPRA National mosaic of landslide hazard zones of the River Basin Plans PAI (v. 4.0 – 2020-2021)

License: CC BY SA 4.0

Format: Shapefile

National mosaic of flood hazard

Description: ISPRA National mosaic (v 5.0 – 2020) of the flood hazard zones mapped by the River Basin District Authorities according to the 3 scenarios defined by Legislative Decree 49/2010 (implementation of the Flood Directive 2007/60/EC)

ISPRA National mosaic of the flood hazard zones – Scenarios of Legislative Decree 49/2010 (v 5.0 – 2020)

License: CC BY SA 4.0

Format: Shapefile

Landslide and flood risk indicators

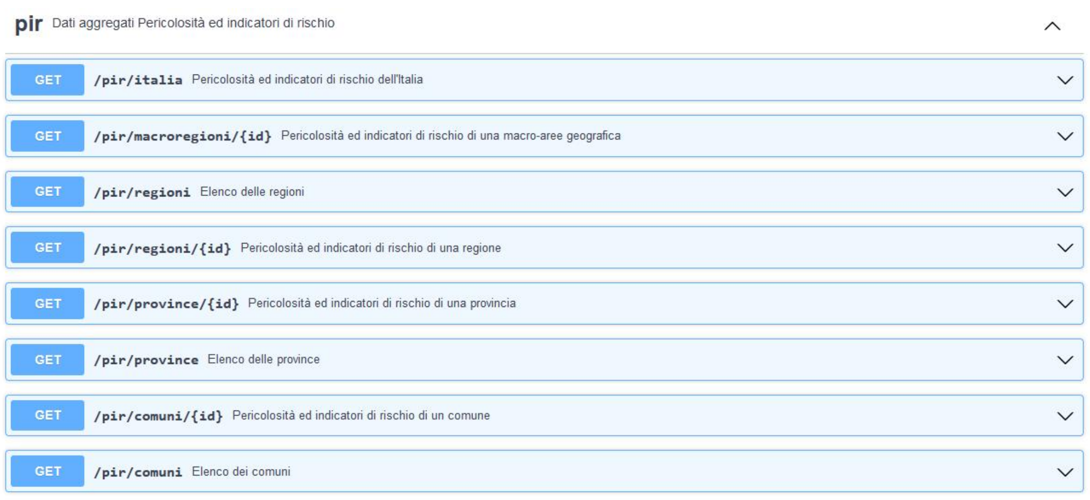
Description: Landslide and flood risk indicators relating to territory, population, families, buildings, industry and service, cultural heritage

- Italy (XLS, CSV, JSON)
- Regions (XLS, CSV, JSON)
- Provinces (XLS, CSV, JSON)
- Municipalities (XLS, CSV, JSON)

Figure 2 ISPRA's open data

(Resource: https://idrogeo.isprambiente.it/app/page/open-data)

By leveraging PostgreSQL, the project database ensures reliable, efficient, and secure management of all data related to flood and landslide hazards in Italy. The app relies on PostgreSQL for robust data storage and management, leading to its reliability, scalability, and powerful feature set. These are essential for handling the dynamic and extensive datasets required by the platform.



pir Dati aggregati Pericolosità ed indicatori di rischio

GET `/pir/italia` Pericolosità ed indicatori di rischio dell'Italia

GET `/pir/macroregioni/{id}` Pericolosità ed indicatori di rischio di una macro-aree geografica

GET `/pir/regioni` Elenco delle regioni

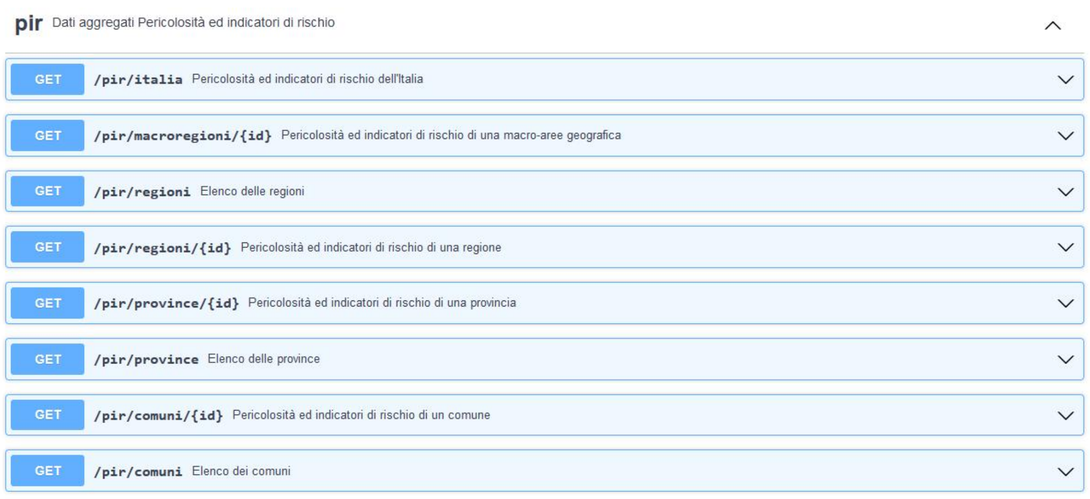
GET `/pir/regioni/{id}` Pericolosità ed indicatori di rischio di una regione

GET `/pir/province/{id}` Pericolosità ed indicatori di rischio di una provincia

GET `/pir/province` Elenco delle province

GET `/pir/comuni/{id}` Pericolosità ed indicatori di rischio di un comune

GET `/pir/comuni` Elenco dei comuni


pir Dati aggregati Pericolosità ed indicatori di rischio

GET /pir/italia Pericolosità ed indicatori di rischio dell'Italia

GET /pir/macroregioni/{id} Pericolosità ed indicatori di rischio di una macro-aree geografica

GET /pir/regioni Elenco delle regioni

GET /pir/regioni/{id} Pericolosità ed indicatori di rischio di una regione

GET /pir/province/{id} Pericolosità ed indicatori di rischio di una provincia

GET /pir/province Elenco delle province

GET /pir/comuni/{id} Pericolosità ed indicatori di rischio di un comune

GET /pir/comuni Elenco dei comuni

Figure 3 Structure of the API response

A data entry contains the following attributes:

- “City”: The geographical coordinates and the name of the location where the sensor providing the data is situated.
- "Landslides by type of movement": index would rely on each hazard analysis of the certain area.
- “Slopes”: it would be identified as “natural” or “artificial”.
- “Level of damage”: it means that the damages from natural hazards are distinguished by the different risks of warning as “slight (aesthetic)”, “moderate (functional)”, and “serious (structural or total loss)”.
- “Types of damage”: " direct damage", "collapse into reservoir", "water course blockage", "blockage and landslide dam break", and "artificial dam break".

2.2 PostgreSQL Database

The data of the web app needs structured data management, which depends on PostgreSQL. It can efficiently store structured data from the IdroGEO API and ISPRA’s open data, ensuring that all information on floods and landslides is organized and easily accessible. With its PostGIS extension, PostgreSQL can store and manage geospatial data, which is crucial for mapping and analyzing the geographic distribution of hazards according to the necessity of spatial data support:

User
User_Name: String User_Password: String

<<Feature Type>> Cities
City_Name: String Landslides by type of movement: String Slopes: String Level of damage: String Types of damage: String Fall/topple: Real Rotational/ translational slide: Real Lateral spread: Integer Slow earth flow: Real Rapid debris flow: Real Sinkhole: Integer Complex: Real Cultural heritage of flood risk: Real Area affected by numerous landslide risks: Real Deep-seated gravitational slope deformation: Real

Table 1 Data Structure

User Table	
User_id	Specific identity of the user from the database.
User_name	Name of the user
User_password	Password of the user

Table 2 User table attributes definition

Cities Tables	
Fall/topple	A type of landslide where rocks or debris fall or topple down a steep slope.
Rotational/translational slide	A landslide where a mass of earth or rock moves down a slope in a rotational (curved) or translational (straight) motion.
Lateral spread	The horizontal movement of ground, often caused by liquefaction during an earthquake.
Slow earth flow	A gradual and slow movement of soil or debris down a slope.
Rapid debris flow	A fast-moving flow of loose soil, rock, and water down a slope, typically triggered by heavy rain.
Sinkhole	A depression or hole in the ground caused by the collapse of a surface layer.
Complex	A landslide involving multiple types of movement, such as a combination of slides and flows.
Cultural heritage of flood risk	Cultural heritage at flood risk involves protecting historical sites and artifacts from water damage and erosion caused by flooding.
Area affected by numerous landslides risks	A region where many landslides occur often in potential risks.
Deep-seated gravitational slope deformation	Slow and gradual movement of a large mass of earth or rock over a long period, affecting deep layers of a slope.

Table 3 Project table attributes definition

3. System Overview

The system which is based on disaster management is designed to support the prevention and preparedness phases of disaster management activities. It leverages geospatial data and advanced visualization tools to provide decision-makers, researchers, and the public with critical insights into natural hazards such as landslides and floods. By splitting into multiple highly interactive pages based on a client-server architecture.

The system is composed of three main components: a database, a web server, and an interactive dashboard.

3.1 System Architecture

The system is designed with a robust architecture to efficiently manage, process, and visualize geospatial data. Below are detailed descriptions of each component within the system:

3.1.1Database

PostgreSQL with PostGIS Extension

1.Spatial Data Storage: The system uses PostgreSQL with the PostGIS extension to store and manage spatial data. This extension allows PostgreSQL to handle geographic objects and perform spatial queries efficiently.

2.Data Source: The database holds information retrieved from the Italian Institute for Environmental Protection and Research (ISPRA) datasets, which include detailed hazards and risk indicators.

3.Data Management: The database schema is designed to handle complex spatial data, with tables for different types of hazards, risk indicators, geographical boundaries, and related metadata.

4.Spatial Queries: PostGIS enables the execution of advanced spatial queries, such as intersection, containment, and proximity analysis, which are essential for geospatial data analysis.

3.1.2Web Server

Flask Framework

1.Backend Development: The backend of the application is built using Flask, a lightweight Python web framework known for its simplicity and flexibility.

2.REST API: The web server exposes a REST API that allows the frontend to query the database and retrieve data. This API includes endpoints for various data operations, such as fetching hazard data, querying risk indicators, and performing spatial analysis.

3.Data Processing: The server handles data processing tasks, including spatial queries and aggregations, ensuring that the data returned to the client is pre-processed and ready for visualization.

4.JSON Responses: The results of data queries and processing are returned in JSON format, which is easily consumable by the frontend.

3.1.3Dashboard

Jupyter Notebooks

1.Interactive Interface: The frontend is implemented using Jupyter Notebooks, providing an interactive interface for users. This setup allows for dynamic interaction with the data and seamless integration of code, text, and visualizations.

2.Widgets and Visualization Tools: The dashboard includes various widgets and visualization tools to enable users to interact with the data through maps, charts, and other visual elements. Key libraries used include:

Matplotlib: For creating static, interactive, and animated plots and visualizations.

Plotly: For creating interactive plots and dashboards.

Bokeh: For creating interactive and real-time web-based visualizations.

IpyLeaflet: For integrating Leaflet maps into Jupyter Notebooks, allowing for interactive mapping.

Folium: For visualizing geospatial data with Leaflet maps, making it easy to create interactive maps.

Geemap: For interactive mapping with Google Earth Engine, enabling advanced geospatial analysis.

Dashboard Functionalities

1.Data Visualization: Users can visualize data on maps, charts, and graphs, allowing for comprehensive analysis and understanding of the geospatial data.

2.Interactive Maps: The dashboard supports interactive maps where users can pan, zoom, and switch between different map layers (e.g., satellite, topographic, weather overlays).

3.Layer Management: Users can toggle different layers of information, such as administrative boundaries, hazard zones, and risk indicators.

4.Charts and Graphs: Various chart types (e.g., bar, line, scatter) are available for data visualization, providing insights into trends and patterns.

5.Real-Time Data Updates: The dashboard can retrieve and display real-time data updates, ensuring that users have access to the most current information.

6.Analytical Tools: The dashboard includes tools for data analysis, such as statistical summaries, trend analysis, and spatial correlations.

7.Export Options: Users can export visualizations and data in various formats (e.g., PNG, CSV) for further use and reporting.

3.1.4 Additional Components

User Management

User Authentication: The system includes functionalities for user registration, login, and logout. Users must authenticate themselves to access certain features, such as the "Project" tab.

Role-Based Access Control (RBAC): The system implements RBAC to define permissions for different user roles (admin, regular user, guest), ensuring secure and appropriate access to system features.

3.1.5 High level Architechture

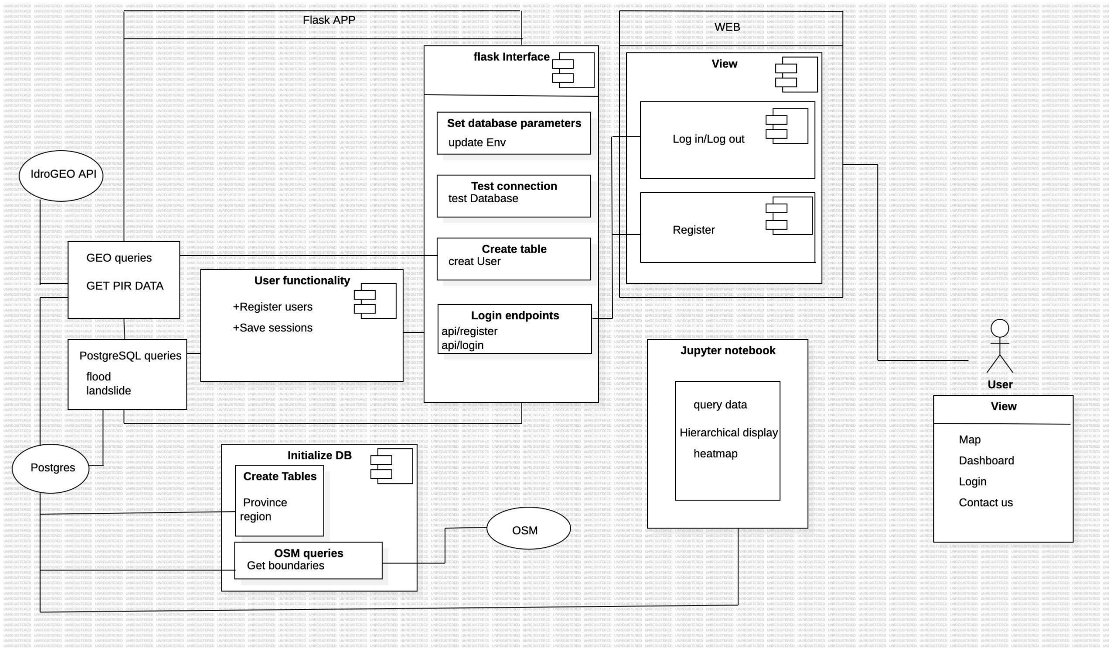


Figure 4：High level architecture of the web app

The diagram shows how these components are interconnected. The Flask APP interacts with the database (Postgres) and APIs (IdroGEO and OSM). The web interface allows users to register, log in, and view data, while the Jupyter Notebook is used for data analysis and visualization. The user can access various views such as maps and dashboards through the web interface.

3.2 Key Features

Data Retrieval and Visualization: Users can search for specific datasets using robust filters and search functionalities. The system provides interactive maps, charts, and graphs to present the data in an easily understandable manner.

Real-time Data Updates: The application ensures that users have access to the most recent data entries, providing real-time updates and visual indicators to show the freshness of the data.

Data Processing: The backend server handles complex data processing tasks, including space-time aggregations and analysis. Techniques like parallel computing and distributed processing are employed to manage large datasets effectively.

Data Export: Users can export visualized data in various formats such as layer packages (.lpk), map packages (.mpk), and other geoprocessing packages.

User Management: The system includes functionalities for user registration, login, and role-based access control. Different user roles (e.g., residents, civil protection, researchers) have specific permissions and access levels.

3.3 Technologies Used

Backend: Flask (Python), PostgreSQL, PostGIS

Frontend: Jupyter Notebooks, Matplotlib, Plotly, Bokeh, IpyLeaflet, Folium, Geemap

Version Control: Git, GitHub

Development Methodology: Agile (Scrum), Jira for project management

4. Software structure

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

- Presentation Server(User's PCs)
- Application Server

- Database Server

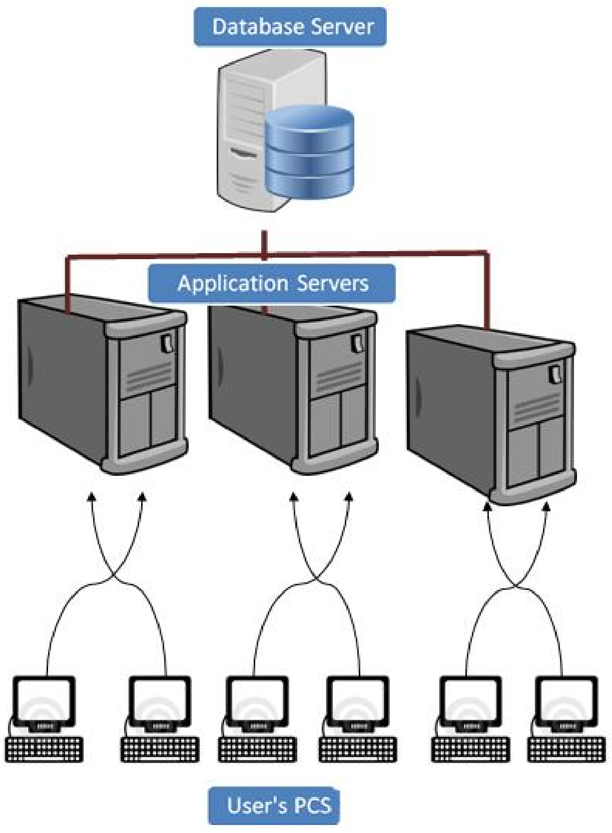


Figure 5: Software structure

4.1 Presentation Server

This top-level layer is the application's user interface, where users interact with the system. Its role is to present information to the users and simultaneously gather input from them.

The presentation layer operates within a web browser. Users can communicate with the HTTP server through actions like entering URLs, clicking, and filling out forms. The primary functions of the HTTP server are to store, process, and deliver web pages to the clients.

In response to user requests, the HTTP server will send a reply to the user's browser. This response will be crafted using HTML, CSS, JavaScript, and Python.

HTML: provides the structure of web pages

CSS: styles and layouts them

Javascript: interactivity and dynamic content.

4.2 Application Server

The application server, also known as the logic tier, handles all the necessary logical operations to meet the software requirements. It processes information gathered from the presentation tier to generate the required web pages. Additionally, the application tier can add, delete, or modify data in the data tier.

Python, along with SQL, will be used to develop the application server. Python will handle all the software's functions, interacting with the DBMS, managing operations between web pages, and controlling the interactive maps that are central to the application's display and analysis.

The essential libraries, categorized by function, are:

- **DBMS:** PostgreSQL
- **Web Development:** Flask
- **Data Analysis:** Pandas, GeoPandas (for data and geodata manipulation)
- **Dashboard:** jupyter dash, plotly

4.2.1 software function

Connect to database: All the credentials are contained in this file. Here the user can find the Database name, the password and the user name.

Continuous data integration: This function is used to continuous update the IdroGEO API to our datebase.

Query data from database: This function is to send query to get from database.

Json data to Dataframe: This function is used to create a Dataframe, starting from the json file.

4.3 Database Server

The data tier, also known as the database server or back-end, is where the information processed by the application is stored and managed. The web application will use PostgreSQL, a free and open-source database management system, to perform queries and select the data to be displayed.PostgreSQL will be integrated with Python, utilizing a database adapter to facilitate communication between the database and the Python programming language.

5. User interface

In this block we will analyze the design of the user interface, our aim is to create an intuitive and user-friendly interface to meet different user roles with different levels of expertise. A well-organized web structure ensures easy navigation and access to key functions.

5.1 Navigation Bar

An important element of the user interface, common to all pages of the application.

Users can click on each tab to navigate to different sections of the application.

Project Tab: Requires users to be logged in before accessing the "Project" page.

More Info Tab: Accessible to all users. Provides information about the general idea of the project, the data involved, and the API used by the application.

Log In Button: Located at the far right of the navigation bar. Directs users to the login page. Unregistered users can register on this page.

5.2 Homepage

Homepage View

Header	Contains the application logo, navigation menu, user account options (login/logout, register)
Main Content	Introduction to the application, latest updates, and quick access links to key features
Footer	Contact information, links to privacy policy, terms of service, and social media profiles

Table 4 Homepage View

5.3 Dashboard

Dashboard	
Navigation Panel	A sidebar that allows users to switch between different sections of the dashboard (e.g., data query, visualization, analysis, export).
Main Content	Displays the selected functionalities and data visualizations. This area is dynamic and updates based on user interactions.
Data Query	Users can input or select query conditions (e.g., location, disaster type) and view the results.
Data Visualization	Interactive maps and charts that display the queried data. Users can toggle different layers, switch base maps, and apply filters.
Data Analysi	Tools for performing various analyses on the data, such as trend analysis, comparison, and risk assessment.
Data Export	Options for exporting visualized data in different formats.

Table 5 Dashboard

5.4 User Management

User Management	
Registration Page	Form for new users to create an account by providing necessary information (username, email, password). Includes validation for unique usernames and strong passwords.
Login Page	Secure login interface for registered users to access their accounts. Includes error messages for invalid credentials and options for password recovery.

Profile Page	Allows users to view and update their profile information, change password, and manage account settings.
Admin Panel	Accessible to users with administrative roles. Provides tools for managing users, monitoring system usage, and configuring application settings.

Table 6 User Management

5.5 Interactive Element (more info)

Interactive Element (more info)	
Maps	Interactive maps that support panning, zooming, and rotating. Users can toggle between different map layers and base maps.
Charts and Graphs	Various types of charts (e.g., line charts, bar charts, pie charts) for data visualization. These elements are interactive, allowing users to hover over data points for more information.
Forms and Filters	Forms for inputting query conditions and filters for refining search results. These forms provide instant feedback and validation to guide users

Table 7 Interactive Element (more info)

6. User Case Application

In this section, we will show the possible functions ans use cases of the website, and consider the exceptions that may occur under different condition.

Use case 1:Specified Condition Data Query	
Name	Specified Condition Data Query
User	Register User
Condition	The users who have successfully logged in and connected to the Internet

Flow of Events	The events follow this order: User inputs or selects query conditions on the interface; After confirming the query conditions, the user clicks the "Query" button; Display eligible data; The user reviews the query results and may perform further filtering or export data operations.
Exit condition	Users can exit the query function at any time by closing the web page or navigating to other pages
Exceptions	The user enters query conditions in an incorrect format There are system errors or data source connection issues The query results are empty

Table 8: Specified Condition Data Query

Use case 2: Data Geographic Visualization	
Name	Data Geographic Visualization
User	Register User
Condition	The users who have successfully logged in and connected to the Internet; The users have completed the criteria query
Flow of Events	The events follow this order: Users trigger the visualization process by clicking the "Visualize" button; The system generates a geographic visualization
Exit condition	They can exit by clicking the "Exit Visualization" button; Users can exit the function at any time by closing the web page or navigating to other pages
Exceptions	No data matches the selected criteria; In case of data retrieval errors or map rendering issues

Table 9: Data Geographic Visualization

Use case 3: Base Map Switch	
Name	Base Map Switch
User	Register User
Condition	The users who have successfully logged in and connected to the Internet; The users have completed the criteria query or data visualization

Flow of Events	The events follow this order: Users locate the base map switch functionality on the map interface; Users click the “base map switch” button, triggering the action to switch the base map; The system loads the appropriate map data based on the user's selected base map type, replacing the current map display
Exit condition	Users can exit the function at any time by closing the web page or navigating to other pages They can exit by clicking the "Exit " button
Exceptions	Users select an invalid base map type; Users encounter issues while switching base maps

Table 10: Base Map Switch

Use case 4: Analyse Data	
Name	Analyse Data
User	Register User
Condition	The users who have successfully logged in and connected to the Internet; The users have completed the data query
Flow of Events	The events follow this order: Users select the data set to analyze and may specify analysis parameters or conditions; Clicking the "Analyse" button; The system presents the analysis results to the user
Exit condition	Users can exit the function at any time by closing the web page or navigating to other pages They can exit by clicking the "Exit " button
Exceptions	Users select an invalid data set or parameters; Analysis results cannot be generated or are invalid

Table 11: Analyse Data

Use case 6: Register	
Name	Register
User	All no-registered user
Condition	The user can register access the website by clicking the “ Register” button.

Flow of Events	The events follow this order: A user open the website; Clicks on the “ Register ” button; User puts the “ username ” and “ password”; After completing the above steps, users successfully register as a new user of the website and relocates to the login screen.
Exit condition	The username is already in use
Exceptions	The user already exists in the database

Table 12: Register

Use case 7: Log In	
Name	Log In
User	Register User
Condition	The users have registered an account.
Flow of Events	The events follow this order: User accesses the web page and navigates to the Log In section; User enters their username and password into the designated fields; The user triggers the log in process by clicking the "Log In" button; Upon successful log in, the user is redirected to a dashboard, account page.
Exit condition	The users can log out by clicking the "Log Out" button; Users can exit the Log In feature by navigating away from the logged-in area of the website
Exceptions	The user enters incorrect username/ password The user forgets their password

Table 13: Log In