Post-Mining Multi-Hazards evaluation for land-planning

PoMHaz

WP4: GIS development tools

D13 - Deliverable 4.2: Implemented interfaces, database and DSS toolbox

Authors:
Dr. Moncef Bouaziz, *Institute for Mine Surveying and Geodesy, Freiberg University of Technology, 09599 Freiberg, Germany*
Prof. Dr. -Ing Jörg Benndorf, *Institute for Mine Surveying and Geodesy, Freiberg University of Technology, 09599 Freiberg, Germany*
M.Eng. Haske Benjamine*, Research Center of Post-Mining, Technische Hochschule Georg Agricola University, 44787 Bochum, Germany*
Dr. Al Heib Marwan, *The French National Institute for Industrial Environment and Risks, Rue Jacques Taffanel, 60550 Verneuil-en-Halatte, France*

Status: Final
Report Date: 17.01.2025

Confidentiality Level: Public

Research Fund for Coal & Steel

This project has received funding from the Research Fund for Coal and Steel under Grant Agreement No 101057326.

Deliverable 4.2	
Due date of Deliverable	17.01.2025
Start – End Date of the Project	03.10.2022 – 02.10.2025
Duration	3 years
Deliverable Lead Partner	*Freiberg University of Technology*
Dissemination level	Public
Digital file name	POMHAZ-WP4-D13-D4.2-Interfaces_DB_DSS-TUBAF-v1
Keywords	GIS; DSS; Interface; data; end users; guide

Content

List of tables

List of figures

Acronyms

API	Application Programming Interface
CLI	Command-Line Interfaces
CSS	Cascading Style Sheets
CPU	Central Processing Unit CPU
DSS	Decision Support System
GDAL	Geospatial Data Abstraction Library
GIS	Geographic Information System
GPU	Graphics Processing Unit
GUI	Graphical User Interface
HDD	Hard Disk Drives
KML	Keyhole Markup Language
MCDA	Multi-Criteria Decision Analysis
NAS	Network-Attached Storage
RDBMS	Relational Database Management System
RFCS	Research Fund for Coal and Steel
SAN	Storage Area Network
SDSS	Spatial Decision Support System
SQL	Structured Query Language
UPS	Uninterruptible Power Supplies
VR	Virtual Reality
WFS	Web Feature Service
WMS	Web Map Service

Executive Summary

This deliverable is part of the POMHAZ project, **Post-Mining Multi-Hazards evaluation for land-planning.**

The main objective of POMHAZ is to identify the interaction between the post-mining hazards for coalmines in Europe and to develop tools for facilitate the management of the post-mining hazards in coal region.

Deliverable 4.2, titled "Implemented interfaces, data base and DSS toolbox" is related to Task 4.3 “GIS and DSS Development and Advanced data visualization”. This deliverable focuses on the development and implementation of user-friendly interfaces of the Decision Support System (DSS), a comprehensive spatial database, and a DSS toolbox to support the monitoring and management of European coalmines post-mining hazards. These components were built using open-source technologies, ensuring cost-effectiveness, accessibility and flexibility.

In the POMHAZ project, the present deliverable is part of the WP4 that is dedicated to the development of a Post-Mining Risk Information System. This WP supports planning and decision-making and provides information to a broad range of stakeholders. Deliverable 4.2 is linked to Deliverable 3.3 which has layed the groundwork by developing a Decision Support System (DSS) and documenting its application for hazard assessment. Deliverable 3.3 focuses on the conceptualization and implementation of a tool that integrates data analysis and visualization to guide stakeholders in making informed decisions about post-mining hazards.

The **user interfaces** were developed using Leaflet, a lightweight JavaScript library for interactive maps, and GeoServer, an open-source server for geospatial data sharing. These tools allow users to visualize and interact with spatial data related to post-mining hazards, such as land subsidence and environmental risks. The interfaces were designed with accessibility in mind, featuring keyboard navigation and auditory feedback to accommodate blind users. Leaflet provides a front-end for interactive mapping, while GeoServer efficiently serves GIS data for web-based interactions.

The **database of post mining areas** was developed using PostgreSQL with the PostGIS extension for spatial data management. PgAdmin was used to manage the database, while QGIS, an open-source desktop GIS, was employed for spatial data analysis and visualization. The database integrates diverse data sources, including satellite imagery (Sentinel-1 and Sentinel-2), UAV data, and ground-based sensors. It is optimized for handling large datasets and complex spatial queries, with techniques like spatial indexing applied to improve performance. QGIS was used to manage spatial data workflows, and pgAdmin allowed for database management and query execution.

The **DSS toolbox** was built using mainly Flask, an open-source web framework, to provide decision-making tools for post-mining hazard assessment. This system integrates the PostgreSQL/PostGIS database, enabling users to perform spatial analyses, generate hazard risk maps, and make informed decisions based on real-time data. The DSS offers interactive dashboards and risk mapping functionalities, providing actionable insights for decision-makers and supporting real-time hazard management.

By leveraging open-source tools such as PostgreSQL/PostGIS, pgAdmin, QGIS, Leaflet, and GeoServer, Deliverable 4.2 establishes a robust and scalable system for post-mining hazard monitoring and decision-making, aligning with the overall goals of the POMHAZ project.

The annexes of this deliverable provide comprehensive guidelines for effectively utilizing the interfaces, database, and decision support system.

1 Overview

This section involves a comprehensive examination of hardware, software, data formats, data management, and security aspects necessary to support the system's functionalities. This integrated system aims to leverage the power of spatial data analysis and decision support functionalities to facilitate informed decision-making and proactive risk management in post-mining environments in the context of POMHAZ project. Figure 1 presents the GIS-based Decision Support System (DSS) architecture and the main elements.

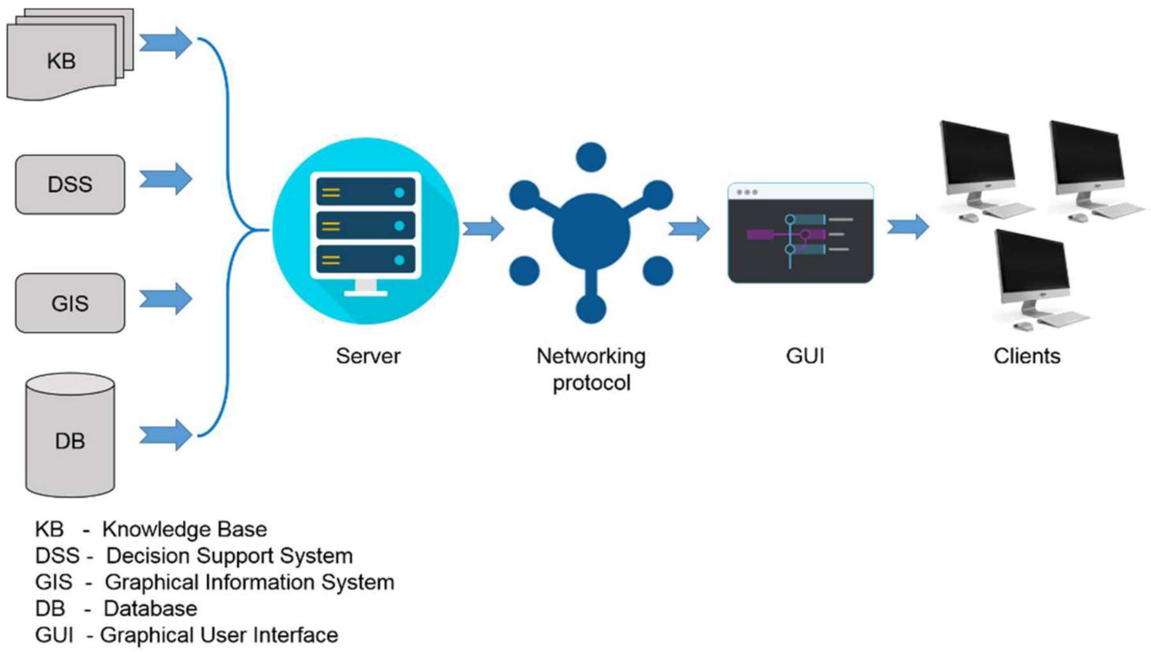


Figure 1. GIS-based Decision Support System architecture

To address the system requirements and carefully craft the architecture of the GIS-based Decision Support System, we worked collaboratively with our partners to develop a questionnaire. This method ensures comprehensive consideration of all relevant data and IT-input requirements, thereby boosting the accuracy and effectiveness of the DSS. Moreover, this questionnaire will guide us in aligning with the specific needs and preferences of our partners concerning the proposed DSS.

The main objective of the POMHAZ project is to tackle the intricate challenges linked to post-mining hazards. Collaborating with seven partners from four EU countries, our goal is to formulate a comprehensive system design and framework for a GIS-based Decision Support System (DSS). This system aims to effectively manage, analyze, and visualize spatial information pertaining to post-mining hazards. To achieve the objective of the POMHAZ project, it is crucial to leverage an open-source GIS stack that supports the development of a robust database, web interface, and Decision Support System (DSS) tool. This stack includes powerful tools such as **PostgreSQL** with the **PostGIS** extension for spatial data management, enabling efficient storage, querying, and analysis of geospatial datasets. For the web interface, frameworks like **Leaflet** or **OpenLayers** can be utilized to create interactive maps and visualizations that enhance user engagement and data accessibility. Additionally, the **GeoServer** platform can serve as a vital component for sharing geospatial data across different clients and applications. By integrating these tools, the system will not only meet the specified requirements outlined in Figure 2 but also provide a comprehensive solution that facilitates informed decision-making, effective risk assessment, and sustainable land planning in impacted areas. This open-source GIS stack ensures that the most suitable IT tools are available for developing the database, web interface, and DSS, ultimately enhancing the project's outcomes.

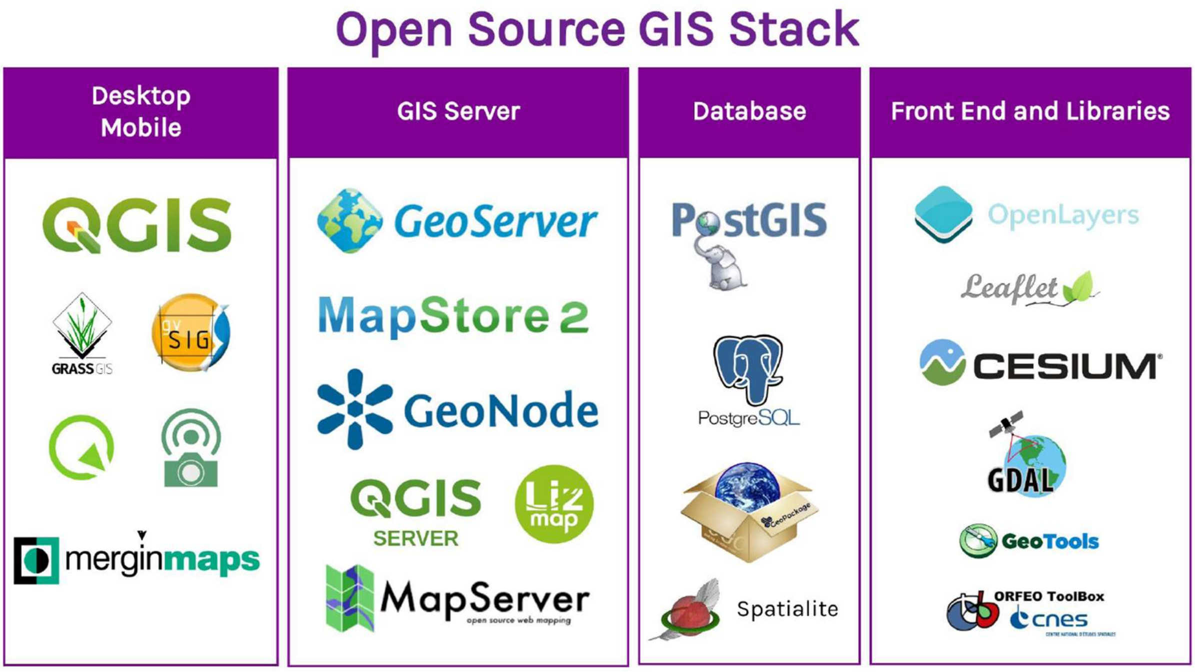


Figure 2. Available Open-source GIS Stack for the Data base, Web Interface and DSS development

2 Implementation of Interfaces

The **implementation** of the web system interface stands as a cornerstone in the effective visualization and management of hazard layers associated with post-mining areas. This **implementation** process is essential for transforming complex hazard and spatial data into an accessible, interactive platform that end-users can readily navigate. By focusing on robust, user-centric **implementation**, the system supports effective hazard monitoring, risk assessment, and decision-making—all critical to post-mining safety management.

The **implementation** phase has integrated advanced geospatial technologies, dynamic data processing, and real-time monitoring capabilities, ensuring that users access precise, updated information on subsidence, environmental factors, and hazard zones. Through rigorous **implementation**, the interface connects users to critical insights, enabling both technical and non-technical users to interact with complex spatial data.

Key Implementation Features of the Web System Interface:

- **User-Friendly Interactive Exploration**: The **implementation** of intuitive navigation tools enables users to explore various hazard layers seamlessly. They can zoom, pan, and interact directly with geographic features to view hazard information, a feature made possible through deliberate interface **implementation** tailored to enhance user engagement and data comprehension.
- **Advanced Layer Management Functionality**: In this **implementation**, users can switch hazard layers on and off, tailoring their views to specific needs. This layer management feature, rigorously implemented, provides flexibility and control, empowering users to explore distinct risk factors, analyze data across different contexts, and make informed, situational decisions.
- **Real-Time Visual Data Representation**: By prioritizing real-time data integration during **implementation**, the system delivers up-to-the-minute visual insights into mining hazards. Whether it’s risk maps or environmental monitoring data, this visual representation—meticulously implemented—offers users a clear, actionable view of post-mining conditions, thereby supporting prompt, informed responses to emerging hazards.
- **Responsive User Interface Design**: The **implementation** focused on responsiveness ensures the system performs optimally across various devices and screen sizes, broadening accessibility for diverse user groups. This adaptable design, integral to the **implementation** strategy, meets users' needs in different field conditions, strengthening its application for on-site decision-making.

The dedicated **implementation** of this web system interface enhances its functionality, reliability, and user adaptability. Through continuous improvements, this interface will remain a vital tool for post-mining hazard management, offering a comprehensive, data-driven foundation to support long-term safety and environmental stewardship.

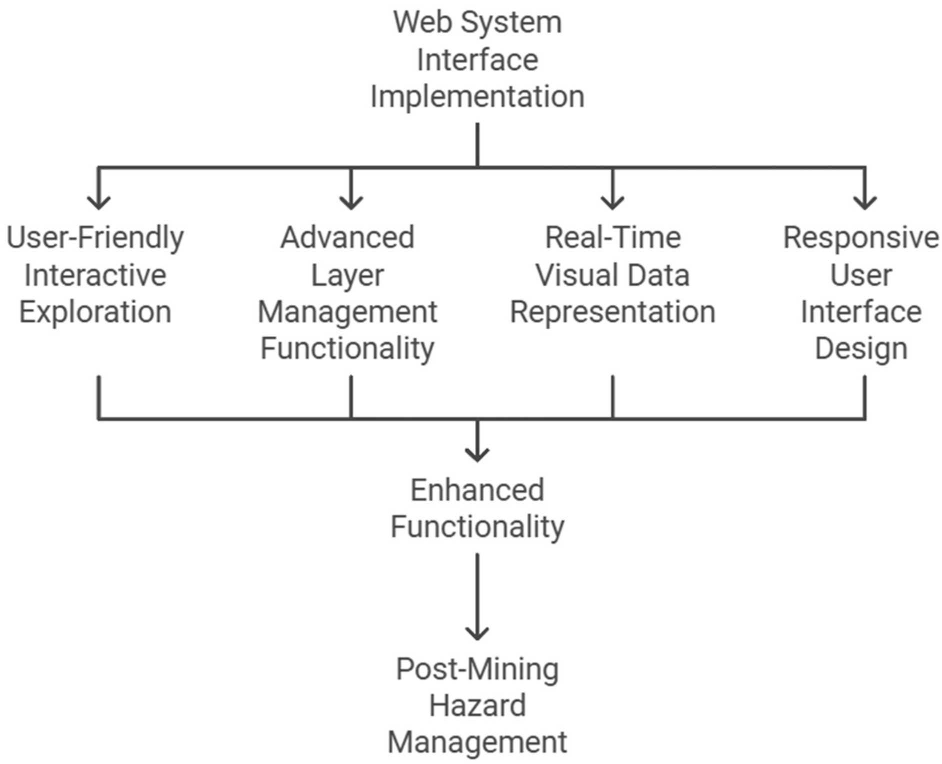


Figure 3. Comprehensive Schematic of Implemented Interfaces

2.1 Web-based Interface

The **web-based interface** of the POMHAZ project functions as a comprehensive, interactive platform that empowers users to remotely access, analyze, and manage multi-dimensional hazard layers across post-mining areas. Built on a robust stack of open-source technologies, this system delivers real-time, actionable insights by integrating several powerful tools tailored for high-performance geospatial applications.

2.1.1 Key Tools and Technical Implementation

1. **Database Management with PostgreSQL and PostGIS**

   At the core of the web interface is a **PostgreSQL** database equipped with **PostGIS** extensions, which enable spatial data handling essential for accurate geospatial analysis. PostgreSQL serves as the primary data repository, managing extensive datasets on subsidence zones, environmental conditions, and hazard assessments. PostGIS further extends this functionality by providing spatial functions that support complex queries, such as proximity analysis and spatial joins, critical for retrieving and processing hazard data at high speed.

2. **Geospatial Data Rendering with GeoServer**

   To render geospatial data on the web efficiently, **GeoServer** is employed as a dedicated map server. GeoServer interacts with the PostgreSQL/PostGIS database to fetch and format geospatial layers in standard web formats such as Web Map Service (WMS) and Web Feature Service (WFS), allowing seamless integration into web applications. This setup enables real-time rendering of complex hazard layers and environmental data directly on the web interface, allowing users to visualize layered spatial information such as risk zones, land subsidence, and vegetation impact areas.

3. **Interactive Mapping with Leaflet**

   Leaflet, a leading JavaScript library for interactive maps, powers the web interface's map visualizations. Leaflet's lightweight structure and compatibility with GeoServer make it ideal for embedding rich, dynamic maps within the web interface. Users can zoom into specific locations, pan across regions, and interact with individual map features, retrieving detailed hazard data on demand. Additional functionality was implemented to allow users to toggle hazard layers on and off, enabling a customized visualization experience tailored to specific monitoring and analysis needs.

4. **Back-End and Front-End Integration with Flask and JavaScript**

   Flask, while primarily dedicated to the development of the Decision Support System (DSS) as detailed in Deliverable 3.3, also supports certain backend functions within the web interface, managing API calls and facilitating smooth communication between the client and server. Flask processes requests for data retrieval and analysis, directing data from the PostgreSQL database to the front-end. **JavaScript** then integrates with Flask's API endpoints on the client side, providing responsive, asynchronous updates to hazard data as new information becomes available, ensuring users have access to the latest insights.

5. **Responsive Design and Real-Time Data Integration**

   The web-based interface is engineered with a responsive design to ensure accessibility across devices, from desktops to tablets and smartphones. Using HTML and CSS for adaptable layouts, the interface adjusts seamlessly to varying screen sizes and resolutions. Dynamic data loading techniques, combined with JavaScript's asynchronous requests, allow the interface to present live hazard updates as soon as data is ingested, making real-time monitoring possible. This capability is essential for timely hazard assessment and rapid response in post-mining areas, where environmental conditions can change unpredictably.

2.1.2 Enhanced Functionality for Proactive Risk Management

This integrated toolset, along with intuitive design choices, allows users not only to visualize but also to interact with complex spatial data in a meaningful way. For example, users can overlay multiple hazard layers, assess spatial correlations between risk factors, and generate real-time reports on areas impacted by subsidence or other environmental hazards. By implementing these capabilities, the **web-based platform** bridges data science and decision-making, giving stakeholders the insights needed for proactive, data-driven management in post-mining areas.

This robust, thoughtfully designed web interface not only facilitates remote data access but also empowers users with a practical tool for comprehensive hazard monitoring, ensuring that the POMHAZ project achieves its mission of effective risk management and environmental sustainability in post-mining contexts.

2.2 Desktop Interface

The desktop interface offers a high-performance environment ideal for advanced analysis and management of complex hazard data, designed specifically for technical users and GIS specialists within the POMHAZ project. Developed using also QGIS for geospatial data processing and visualization, the desktop interface enables users to perform detailed analyses, including high-resolution mapping, spatial modeling, and in-depth risk assessments.

For efficient database access and management, PostgreSQL is integrated with pgAdmin 4 to handle extensive spatial datasets, allowing users to store, retrieve, and manipulate hazard data effortlessly. Additionally, the Python programming language, combined with specialized GIS libraries such as GDAL (Geospatial Data Abstraction Library) and PyQGIS, supports custom scripts for advanced spatial analysis tasks.

This desktop setup prioritizes high-speed data processing and advanced data manipulation capabilities, making it essential for users who require resource-intensive operations for complex post-mining hazard evaluations. By combining these powerful tools, the desktop interface serves as an indispensable asset within the POMHAZ framework, providing deep analytical insights and fostering data-driven decision-making.

3 Database Development

The POMHAZ project aims to build an advanced, GIS-based decision support system (GIS-DSS) specifically designed for assessing and managing post-mining hazards. For this purpose, we used PostgreSQL database enhanced with the PostGIS extension to store and process a wide range of spatial data, which is essential for accurately mapping and evaluating risk in post-mining areas. PostGIS enables PostgreSQL to handle spatial queries and functions, allowing for efficient storage and manipulation of complex geographic datasets such as soil erosion patterns, groundwater levels, and historical mining data, all of which are critical in hazard assessment.

By linking this spatial database with QGIS, the POMHAZ project provides an accessible interface for visualizing and analyzing hazard-related data. Through QGIS, users can load various spatial layers stored in the PostgreSQL database and overlay them to examine the relationships between different environmental factors. For instance, users can analyze how environmental hazards and risks intersect within former mining regions, identifying areas that may pose heightened risks to communities or ecosystems. QGIS's suite of analytical tools supports POMHAZ's goal of providing a thorough understanding of the spatial dynamics at play in these post-mining landscapes.

This setup also enables continuous monitoring and updating, allowing stakeholders to incorporate new data—such as satellite imagery or sensor inputs—into the system as conditions change over time. This real-time data integration enhances the GIS-DSS tool's responsiveness, ensuring that it can support proactive hazard management by providing up-to-date information on risk zones. Additionally, this GIS-based platform can generate visual representations and maps, making complex data accessible and interpretable for a diverse range of users, from local authorities to environmental scientists and policymakers.

Overall, the POMHAZ project's GIS-based approach, integrating PostgreSQL, PostGIS, and QGIS, is designed to support informed decision-making and improve risk mitigation in post-mining areas. By enabling the analysis of spatial data and visualization of potential hazards, POMHAZ equips stakeholders with the tools they need to make data-driven decisions that prioritize safety, sustainability, and resilience in communities affected by mining legacies.

This comparative Table 1 outlines the key strengths and suitability of each database system based on the specific requirements and characteristics of the post-mining multi-hazards evaluation for land-planning. The information provided can assist in selecting the most appropriate database system that aligns with the project's data management, analysis, and reporting needs.

Table 1. Comparative table of database systems for post-mining multi-hazards evaluation for land-planning

Database System	Strengths and Suitability	Suitability for POMHAZ
Relational Database Management System (RDBMS)	- PostgreSQL with PostGIS Extension: Robust spatial data capabilities, suitable for managing complex geospatial data. Suitable for large-scale projects requiring advanced spatial analysis and management capabilities.	Yes
	- MySQL with Spatial Extensions: Efficient handling of smaller to mid-sized projects. Suitable for projects with moderate geospatial data requirements.	Yes
	- Microsoft SQL Server: Seamless integration with the Microsoft technology stack. Excellent spatial data support for medium to large-scale projects.	No
NoSQL Databases	- MongoDB: Efficient handling of unstructured or semi-structured geospatial data. Suitable for dynamic data needs and projects with flexible data structures.	No
	- Cassandra: Distributed database suitable for managing large volumes of time-series and spatial data. Ideal for handling complex environmental monitoring data.	No
Spatial Databases	- SpatiaLite: Open-source option with GIS functionality and SQL support. Suitable for projects with moderate spatial data requirements and a need for basic geospatial analysis.	Yes
	- Oracle Spatial: Commercial option with advanced geospatial data management capabilities. Suitable for large-scale projects with extensive spatial data requirements and complex analysis needs.	Yes
Geospatial Data Warehouses	- Data warehousing solutions (e.g., Amazon Redshift, Google BigQuery, Snowflake): Suitable for extensive data analytics and reporting. Ideal for handling large datasets and complex queries in the context of post-mining multi-hazards evaluation.	Yes
Graph Databases	- Neo4j: Suitable for analyzing complex relationships between geospatial entities, such as ecological networks or geological strata. Ideal for representing and querying intricate spatial data networks.	No
Time-Series Databases	- InfluxDB: Specialized database for storing and querying time-series data. Suitable for managing temporal data in post-mining assessments and facilitating time-based analysis.	No

Considerations for choosing the right database system include data volume, data complexity, the need for spatial indexing and analysis (which is the POMHAZ case), and the compatibility with existing infrastructure and tools. Ultimately, the choice should align with the specific requirements of the Post-Mining Hazard Assessment project and the capabilities required by the Decision Support System.

3.1 PostgreSQL and PostGIS Setup

In the present Deliverable 4.2 of the POMHAZ project, the establishment of the database using PostgreSQL and PostGIS was a multi-step process designed to ensure a comprehensive and efficient data management system for hazard and risk assessment in post-mining areas.

The first step was to set up PostgreSQL as the core relational database management system (RDBMS), which allows for structured data storage and robust querying capabilities. During this phase, the database schema was defined, outlining the various tables required to store the collected hazard and risk data. These included tables for both spatial and non-spatial attributes, ensuring that all relevant information were captured systematically.

Next, PostGIS was integrated to extend the functionality of PostgreSQL with spatial capabilities. This involved enabling the PostGIS extension on the PostgreSQL database, which provided tools to store and manipulate geographic information directly within the database. Geometry columns were created in the tables to store spatial data types, such as points, lines, and polygons, which represent the geographical features relevant to case studies.

Data integration from various formats—such as vector files (e.g., shapefiles), raster images (e.g., satellite imagery), and tabular datasets (e.g., CSV files)—was facilitated through pgAdmin, a powerful graphical interface for managing PostgreSQL databases. Data import functionalities was employed in pgAdmin to upload and convert these diverse data formats into the database. For instance, the Shapefile import feature was used to bring in vector data, which PostGIS seamlessly converted into spatial formats, allowing us to perform spatial queries.

To ensure data integrity and usability, normalization practices were implemented, reducing data redundancy and improving database performance. Relationships were also established between tables using foreign keys, which enabled complex queries that could link spatial data with corresponding hazard assessments.

Throughout the development process, rigorous documentation was maintained and best practices were utilized for database design to ensure scalability and maintainability. This structured approach to building PostgreSQL and PostGIS database not only facilitates efficient data storage and retrieval but also lays the groundwork for advanced spatial analysis and visualization within the Decision Support System (DSS) that supports hazard management in post-mining areas.

Figure 4 illustrates the six main steps taken to build the database, providing a visual representation of the process. For more technical details regarding the database schema, data types used, and specific integration processes, please refer to the annexes.

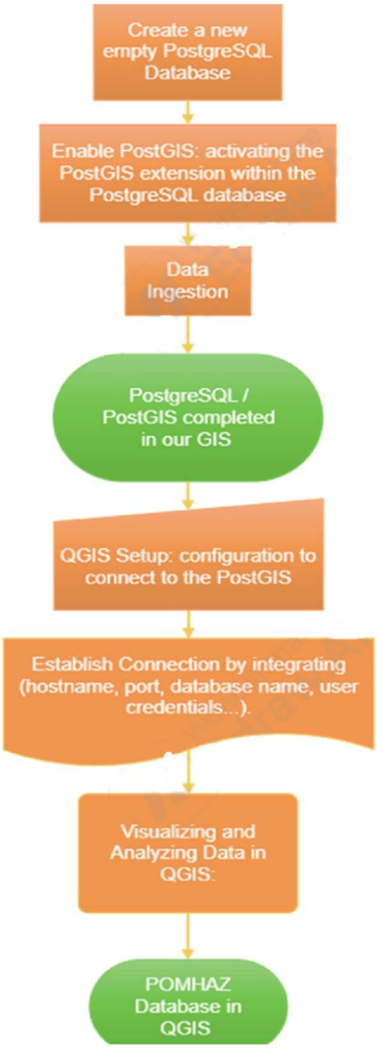


Figure 4. Steps to build the POMHAZ's spatial database

4 DSS Toolbox Development

In the present Deliverable, the development of the Decision Support System (DSS) toolbox represents a crucial step in providing stakeholders a user-friendly interface for hazard assessment and management in post-mining areas. The DSS was developed using Flask, a lightweight web framework for Python, enabling the creation of web applications that are both flexible and scalable.

The toolbox is designed to facilitate the visualization and analysis of hazard data integrated into PostgreSQL and PostGIS database. Utilizing Flask, a web-based interface was built allowing users to interact with various data layers, including risk assessments and hazard maps. HTML, CSS, and JavaScript were employed to create a responsive and intuitive front-end, ensuring that users can easily navigate through the application and access the information they need.

Hazard computations within the DSS are performed using Python, leveraging libraries such as NumPy and Pandas for data manipulation and analysis. These computations involve applying predefined algorithms to assess the risks associated with specific post-mining conditions, using the integrated data from our database. The DSS processes input data, calculates relevant hazard metrics, and displays the results through visual representations such as maps and charts, allowing stakeholders to make informed decisions.

This structured approach to developing the DSS toolbox ensures that it is not only functional but also tailored to the needs of its users. For more in-depth technical details regarding the algorithms used, the specific development processes, and additional functionalities of the DSS toolbox, please refer to Deliverables 3.2 and 3.3.

5 Required Hardware and facilities

To ensure the successful deployment and smooth operation of the **interfaces, database, and DSS toolbox** developed for the **POMHAZ project**, it is crucial to specify the appropriate hardware requirements. These requirements depend on factors such as the volume of data being processed, the complexity of geospatial analyses, and the number of users accessing the system. Below is a detailed list of recommended hardware components and their specifications to ensure optimal performance.

5.1 Server Hardware for Database and DSS Backend

The database and DSS toolbox will handle large amounts of geospatial data, perform complex computations, and serve multiple users. Therefore, the backend hardware must provide high processing power, memory, and storage capacity.

Processor (CPU):
For server infrastructure, it is recommended to use Intel Xeon E5 or AMD EPYC processors with 8-12 cores. These multi-core, server-grade CPUs are essential for handling the parallel processing of queries and large geospatial operations such as spatial indexing and raster calculations. The additional cores allow for improved performance and efficiency, particularly when managing complex datasets and running multiple processes simultaneously. This ensures that the server can handle heavy workloads associated with geospatial data management and analysis in a reliable and scalable manner.

Memory (RAM):
To efficiently process large datasets, such as Sentinel-1 and Sentinel-2 satellite imagery, the server should be equipped with 64 GB - 128 GB of DDR4 ECC RAM. High memory capacity is crucial for running geospatial databases like PostgreSQL/PostGIS and managing real-time data streams with multiple users. Error-correcting code (ECC) RAM enhances system stability by preventing data corruption, making it suitable for high-availability environments where large volumes of geospatial data are processed concurrently. This ensures optimal performance in both real-time and batch data processing scenarios.

Storage (HDD/SSD):
A storage system with at least 4 TB SSD configured with RAID (e.g., RAID 1 or RAID 5) is recommended. SSDs provide fast read/write speeds, which are critical when handling large raster files and managing big datasets in the geospatial database. RAID configurations ensure data redundancy and fault tolerance, reducing the risk of data loss in the event of hardware failure. This setup ensures the server can handle large volumes of data efficiently while maintaining system resilience and high availability.

Graphics Processing Unit (GPU) (optional for advanced geospatial calculations):
For advanced computational tasks, such as machine learning models or real-time geospatial analysis, a dedicated NVIDIA Tesla GPU or similar with CUDA support is recommended. GPUs can significantly accelerate processing for tasks that require heavy computation, such as 3D visualizations or virtual reality (VR) integrations. The use of a powerful GPU enables the server to handle complex geospatial calculations and render high-fidelity models in real-time, making it ideal for advanced users performing high-demand analyses.

Network Interface Card (NIC):
A 10 Gbps Ethernet network interface card is recommended to ensure high-speed data transmission between the server and external clients. This is particularly important for handling large datasets, such as those generated by sensor data or UAV imagery, which users may upload to the system. A high-bandwidth NIC ensures that the server can manage data traffic efficiently, reducing latency and allowing for faster data access and uploads, which is essential for supporting real-time geospatial operations and multiple concurrent users.

5.2 Client-Side Hardware for Users (Web Interface)

End users interacting with the web interface for data visualization and analysis require robust client machines, especially when working with GIS data or large hazard maps.

Processor (CPU):
For client-side operations in the POMHAZ project, a mid-range processor such as an Intel Core i5 or AMD Ryzen 5 or higher is recommended. While client-side tasks are generally less demanding than server-side processing, these processors are essential for efficiently rendering maps and interacting with GIS layers. Users will benefit from the processing power required for smooth visualization of geospatial data and basic analytical tasks without experiencing slowdowns or performance issues, ensuring a responsive user experience.

Memory (RAM):
It is recommended that client machines are equipped with 8 GB - 16 GB of RAM. This amount of memory is critical for loading and interacting with geospatial layers in web-based GIS applications like Leaflet or Mapbox. Since these applications require large amounts of data to be loaded into memory for seamless map rendering, having adequate RAM ensures that users can work with multiple layers, datasets, or complex maps without experiencing performance lags, ensuring smooth and efficient workflows.

Storage:
For storage, a 512 GB SSD is recommended. Solid-state drives (SSDs) provide faster data access and file loading speeds compared to traditional hard drives, making them essential for users handling large local datasets such as shapefiles or drone footage. The speed and efficiency of SSDs reduce delays in data retrieval and writing, allowing users to quickly load, edit, and save geospatial data, which is particularly important when working with large or complex files.

Graphics Card (optional for users dealing with 3D visualizations):
For users involved in 3D hazard modeling or virtual reality (VR) interactions, a dedicated graphics card such as the NVIDIA GTX 1660 or higher is recommended. A powerful GPU allows for smoother and faster rendering of complex geospatial visualizations, ensuring that 3D models and interactive environments can be explored without lag. This is especially important for advanced users who need to visualize large datasets in 3D or in immersive environments, where detailed and responsive graphics are critical.

Display:
A Full HD (1080p) or higher resolution monitor is recommended, with a dual monitor setup preferred for enhanced productivity. High-resolution displays are essential for viewing detailed maps, charts, and dashboards, ensuring that users can see geospatial data clearly and accurately. The addition of a second monitor allows users to view multiple data layers or reports simultaneously, improving

efficiency and reducing the need to switch between windows, which is particularly useful when conducting in-depth spatial analysis or when working with large amounts of data.

5.3 Cloud-Based or Virtual Server

If the system is hosted on a cloud platform rather than on physical on-site servers, the following cloud configurations are recommended:

- **Cloud Providers**:
  For hosting the POMHAZ project's infrastructure, using a reliable cloud provider such as **Amazon Web Services (AWS)**, **Google Cloud Platform (GCP)**, or **Microsoft Azure** is highly recommended. These platforms offer powerful compute instances, such as **AWS EC2 m5.2xlarge**, which provide a balance of performance and cost-effectiveness. Each instance should have **32-64 GB of RAM** to handle the memory-intensive tasks associated with geospatial analysis and database management. Additionally, **Elastic Block Storage (EBS)** or equivalent cloud storage solutions with a minimum of **4 TB SSD** capacity are necessary to manage large datasets efficiently. Regular snapshot backups should be enabled to ensure that all system data and configurations are secure and easily restorable in the event of failure or data loss.

- **Load Balancing and Scalability**:
  To manage the system's performance during periods of high traffic, such as after hazard alerts or when large volumes of data are being processed, implementing **auto-scaling groups** is essential. These enable dynamic resource allocation by automatically adjusting the number of active instances based on system load. This horizontal scaling ensures that the system remains responsive during peak usage times, maintaining performance and preventing bottlenecks. Additionally, a **load balancer** distributes traffic evenly across all available instances, preventing any single server from becoming overwhelmed. This setup provides both scalability and reliability, ensuring the GIS-DSS tool can handle fluctuating demand without interruptions.

5.4 Backup and Redundancy

Backup and redundancy systems are crucial to ensure data integrity, safety and system reliability.

The table below provides a summary of key hardware specifications.

Backup Storage:
It is highly recommended to use a **Network-Attached Storage (NAS)** system with a minimum capacity of **10 TB** to handle the backup requirements of the POMHAZ project. A NAS provides a centralized and scalable solution for storing large datasets, system configurations, and project-related geospatial data. Regular backups of the **database** and **DSS toolbox** can be automated to ensure data integrity, allowing quick recovery in case of system failure or corruption. This centralized storage also facilitates easy access for authorized users across the project team, making it an essential component of the overall data management strategy. In the event of a disaster, such as data loss or hardware failure, the NAS ensures minimal disruption by maintaining multiple backup copies that can be restored quickly.

Backup Power (UPS):
To safeguard the server infrastructure from unexpected power outages, the use of an **Uninterruptible Power Supply (UPS)** is strongly recommended. A UPS acts as a buffer by supplying backup power during short-term outages, preventing abrupt shutdowns that could lead to data corruption or hardware damage. In case of longer power interruptions, the UPS allows enough time for a controlled, graceful shutdown of servers and critical systems. This minimizes data loss and ensures that ongoing processes, such as geospatial data analysis or database transactions, are not interrupted suddenly. Having a reliable UPS system in place is crucial for maintaining the stability and resilience of the server environment.

Table 2. Summary of Key Hardware Specifications

Component	**Recommended Specifications**
CPU	Intel Xeon E5/AMD EPYC (Server) / Intel i5 or higher (Client)
RAM	64 GB - 128 GB (Server) / 8 GB - 16 GB (Client)
Storage	4 TB SSD (Server), 512 GB SSD (Client)
GPU (Optional)	NVIDIA Tesla (Server) / NVIDIA GTX 1660 (Client)
Network	1 Gbps fiber-optic (External) / Gigabit Ethernet (Internal)
Cloud (Optional)	AWS EC2 m5.2xlarge, 32-64 GB RAM, 4 TB SSD
Backup	NAS (10 TB), UPS for power protection

This hardware specification provides a scalable and robust infrastructure to support the complex demands of the GIS-DSS tool, ensuring high performance for both server-side processing and client-side interaction.

6 Perspectives and Recommendations

In the context of the present Deliverable, future work should focus on addressing both short-term operational needs and long-term strategic goals to improve the system's functionality, scalability, and adaptability. The current infrastructure—comprising web interfaces, geospatial databases, and a decision-support toolbox—forms a solid foundation, but further enhancements are necessary to ensure that the system remains responsive to evolving project requirements and stakeholder needs.

As the POMHAZ project scales up, the ability to manage larger datasets, incorporate more advanced analytical tools, and integrate real-time data streams will be vital. This involves not only expanding the system's technical capabilities but also making it more intuitive and accessible for diverse user groups ranging from environmental researchers to policy makers, mining companies, and local communities.

For example, the web interface, which serves as the primary point of interaction for many users, must evolve to become more dynamic and interactive. This will facilitate better data exploration, visualization, and decision-making, allowing users to engage with complex geospatial data more effectively. Likewise, the geospatial database must be expanded to handle additional types of data while ensuring that the system's performance remains optimized. A robust database structure will be essential for storing and managing the increasing volume of satellite imagery, drone footage, and sensor data that will be incorporated into the GIS-DSS tool.

The DSS toolbox, which acts as the analytical core of the system, must also be enhanced to offer more predictive capabilities and allow for multi-scenario analysis. As the project moves forward, it will be important to not only refine the existing hazard models but also integrate new types of hazard data and simulation tools to provide decision-makers with a comprehensive view of risks and mitigation options.

Below are detailed recommendations for these enhancements across three key areas: Database Expansion, and DSS Toolbox Enhancements. These improvements will ensure the system continues to support the project's overall objective of creating a comprehensive, user-friendly platform for monitoring, predicting, and mitigating post-mining hazards in vulnerable regions.

Database Expansion

The developed database is the backbone of the GIS-DSS tool, storing vast amounts of geospatial, temporal, and sensor data. To expand and enhance the future-proof the database, the following expansions could be taken in consideration:

- **Incorporation of Additional Data Sources**: Expand the existing database by integrating **new types of data** such as high-resolution imagery from **drones** or **ground-based sensors** (e.g., methane sensors, soil moisture detectors). These data sources can complement the existing **Sentinel-1 and Sentinel-2 satellite imagery** and add another layer of precision to hazard prediction models.
- **Historical Data Integration**: Collect and store **historical datasets** related to post-mining hazards (e.g., records of past flooding, erosion events, or subsidence incidents). This historical data is critical for validating models and enhancing the accuracy of predictive tools.

- **Enhanced Metadata Management**: Improve the management of metadata to provide more detailed information about each dataset (e.g., origin, date of collection, resolution, accuracy). This will help researchers and decision-makers assess data quality and suitability for specific analyses.
- **Real-Time Data Ingestion**: Implement a real-time data ingestion framework that allows for continuous updating of the database from sensors deployed in the field. Technologies like **Kafka** or **RabbitMQ** can be used to stream data in real-time, ensuring that hazard predictions are based on the latest available information.
- **Big Data Storage and Optimization**: As the database grows in size, optimizing for performance becomes critical. Use **partitioning** and **indexing** techniques in **PostGIS** to ensure quick query times, especially for large geospatial datasets. Additionally, **cloud storage solutions** such as **AWS S3** or **Google Cloud Storage** can be used to store massive amounts of satellite and drone imagery, which can be dynamically linked to the system when needed.
- **Advanced Data Interoperability**: Ensure that the database can communicate with external systems, offering **APIs** or **data services** that allow third-party applications or tools to access and retrieve relevant datasets. This is particularly important for collaboration with other projects or stakeholders.

DSS Toolbox Enhancements

The Decision Support System (DSS) toolbox is central to enable users to make informed decisions based on hazard predictions and risk assessments. Several key enhancements are suggested for improving its performance and usability. Deliverable 3.3 provides:

- Predictive Modeling Improvements: Expand the predictive models currently in use by integrating machine learning algorithms (e.g., random forests, support vector machines) for more accurate hazard predictions. This would allow the DSS to dynamically adjust its predictions based on new data, continually improving over time.
- Scenario Simulations: Develop features that allow users to simulate different hazard scenarios. For instance, users could simulate a heavy rainfall event and visualize potential soil erosion or flooding in vulnerable areas. These simulations will help decision-makers prepare for various risk conditions.
- Multi-Criteria Decision Analysis (MCDA): Introduce tools for multi-criteria decision analysis, allowing users to evaluate hazards based on multiple factors (e.g., economic impact, environmental risk, and human safety). The system should enable users to weight different criteria based on their needs, generating customized risk assessments.
- Integration with External Models: Ensure that the DSS toolbox is capable of integrating with external modeling tools and data services. For instance, connecting with hydrological models (for flood risk) or geotechnical models (for subsidence and erosion risks) will enrich the toolbox's capabilities.
- Visualization and Reporting: Improve the visualization components of the DSS by integrating more dynamic and detailed tools for data exploration. Use libraries like Plotly or Dash to create interactive dashboards that allow users to explore different hazard layers, filter by time or region, and generate reports. Ensure these visualizations are exportable in formats that can be shared with stakeholders.
- Collaboration Features: Implement features that allow for collaborative decision-making. This could include multi-user access, where different users can annotate, comment on, or

even manipulate certain datasets within the DSS. A version control system can also track changes to the data and models, ensuring transparency.

- User Feedback Loop: Introduce a feedback mechanism where users can submit feedback or suggest improvements to the toolbox. This can help the development team address real-world challenges that may arise as users interact with the system.

By addressing these points, the present Deliverable can evolve into a highly capable and user-friendly system for managing, predicting, and mitigating post-mining hazards, ultimately supporting the broader objectives of the POMHAZ project. These recommendations also ensure that the system remains scalable, accessible, and adaptable to future challenges in hazard management.

7 Conclusion

The POMHAZ project aims to meticulously design the requirements to develop a GIS-based Decision Support System (DSS) to tackle the multifaceted challenges posed by post-mining hazards. This report specifically focuses on the implementation and development of the database and interface for the DSS, detailing the necessary tools and methodologies employed for effective post-mine multi-hazard risk assessment.

A crucial component of this initiative is the construction of a robust database using PostgreSQL and PostGIS, which facilitates the integration and management of diverse data formats essential for comprehensive hazard analysis. The interface has been developed using open-source tools, ensuring flexibility and accessibility while catering to varied user needs through diverse front-end interfaces. This emphasis on stakeholder engagement is vital for creating an inclusive tool that empowers users to interact with and analyze data effectively.

Overall, these strategic choices aim to establish a reliable, flexible, and effective tool for post-mining hazard assessment, enabling stakeholders to make informed decisions based on accurate and comprehensive data analysis.

As the POMHAZ project advances, the next crucial step, still within Work Package 4 (WP4), involves finalizing the specifications for the interfaces and the database system. This preparation is integral to the subsequent implementation of the Decision Support System (DSS), ensuring that the system meets both the technical requirements and the practical needs of its users in managing post-mining hazards effectively.

8 Annexes

List of annexes:

- Annex 1: Guide 1 related to using the System Interface
- Annex 2 : Guide 2 related to Interacting with the PostgreSQL Database

In the context of the POMHAZ project, which focuses on identifying and mitigating risks associated with post-mining hazards, Deliverable 4.2 covers the implementation of system interfaces, a comprehensive database, and a Decision Support System (DSS) toolbox. This deliverable plays a vital role in enabling stakeholders, researchers, and analysts to effectively access, manage, and analyse spatial data related to post-mining hazards.

The "Guides for Using the System Interfaces, Database, and DSS Toolbox" component offers users detailed, step-by-step instructions for navigating and utilizing the suite of tools developed within the POMHAZ framework. These guides cover essential functionalities, from querying spatial data within the PostgreSQL/PostGIS database to accessing visualizations and hazard assessments through the QGIS interface.

These guides are designed to ensure that all users, regardless of their technical expertise, can fully leverage the capabilities of the GIS-DSS tool. By simplifying access to data and analytical tools, this documentation facilitates informed decision-making and proactive hazard management. This alignment with the core mission of POMHAZ enhances safety and resilience in post-mining environments.

Guide 1 related to Using the System Interface

The web system interface is a pivotal tool for visualizing and managing various hazard layers associated with post-mining areas. It allows users to interactively explore spatial data related to hazards, subsidence, and environmental monitoring. By providing a comprehensive platform for data visualization, the interface enhances users' understanding of spatial relationships and risk factors, ultimately facilitating informed decision-making and effective hazard management.

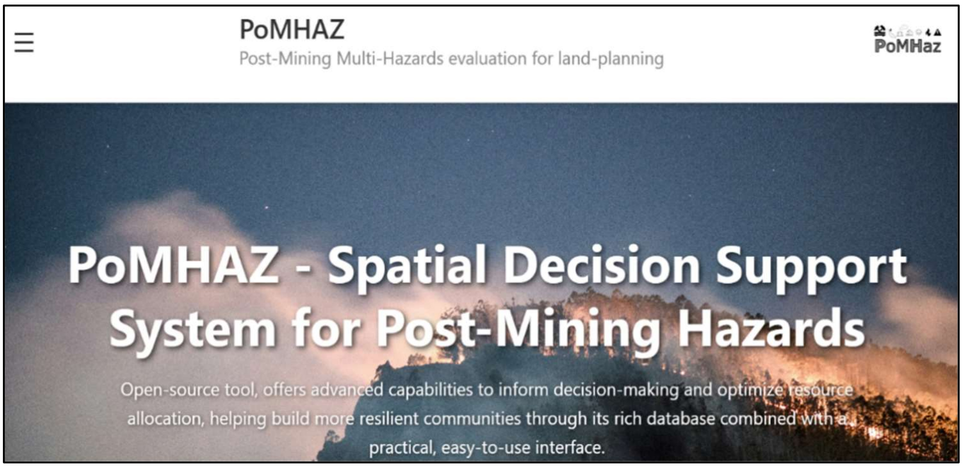
Key Features of the Web System Interface:

- **Interactive Exploration:** Users can engage with the interface to navigate through different hazard layers, zoom in on areas of interest, and click on features to retrieve detailed information about specific hazards and monitoring results.
- **Dynamic Layer Management:** The interface allows users to toggle multiple hazard layers on and off, giving them the flexibility to customize their view based on specific needs. This feature enhances the user's ability to assess different risk factors in various contexts.
- **Visual Data Representation:** The system provides real-time visualizations of mining hazards, including maps that highlight risk zones and monitoring data, enabling users to make informed decisions based on up-to-date information.

1- Accessing the Web Interface and upload your DSS data and documentation:

To begin utilizing the Decision Support System (DSS), first, access the web interface. Ensure you have your login credentials ready, as authentication is required to maintain the security and integrity of the system. The login credentials will be shared by THGA in Bochum and TUBAF, granting authorized users access to both the DSS and the database developed under the POMHAZ project.

Once logged in, you will be directed to the main dashboard, where you can navigate to the "Upload your content to the DSS" section. This area is designed to facilitate the seamless upload of your DSS data and relevant documentation. Here the link: https://dss.fzn.thga.de/



Screenshot of the system interface for the POMHAZ project

2- Navigating the Web Map:

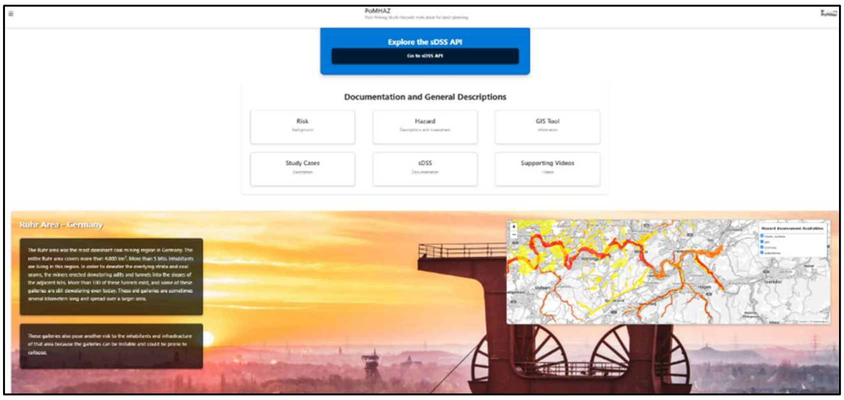
- **Map Controls:**
    - **Zoom In/Out:** Use the + and - buttons or scroll your mouse wheel to zoom in and out.
    - **Pan the Map:** Click and drag the map to move around the area of interest.
- **Layer Control:**
    - On the left sidebar, you will see a list of hazard layers (e.g., subsidence areas, risk zones, Shafts).
    - **Layer Visibility:** Check or uncheck boxes next to each layer to display or hide them on the map.
- **Hazard Layer Information:**
    - Click on any feature (e.g., a polygon representing a hazard area) on the map to open a pop-up window that provides detailed information about that feature.
    - Information may include:
        - Hazard type
        - Risk assessment results
        - Monitoring dates
        - Relevant mitigation measures

3- Interacting with the Map:

- **Search Functionality:**
    - Use the search bar located at the top of the interface to locate specific geographic areas or hazard features by name.
- **Drawing Tools:**
    - Utilize drawing tools (if available) to mark specific areas of interest on the map.
    - You can create polygons or lines to outline areas for further analysis.

4- Accessibility Features:

- The interface includes keyboard shortcuts for navigating map controls.
- Ensure that features for screen readers and auditory feedback are enabled for users with visual impairments.



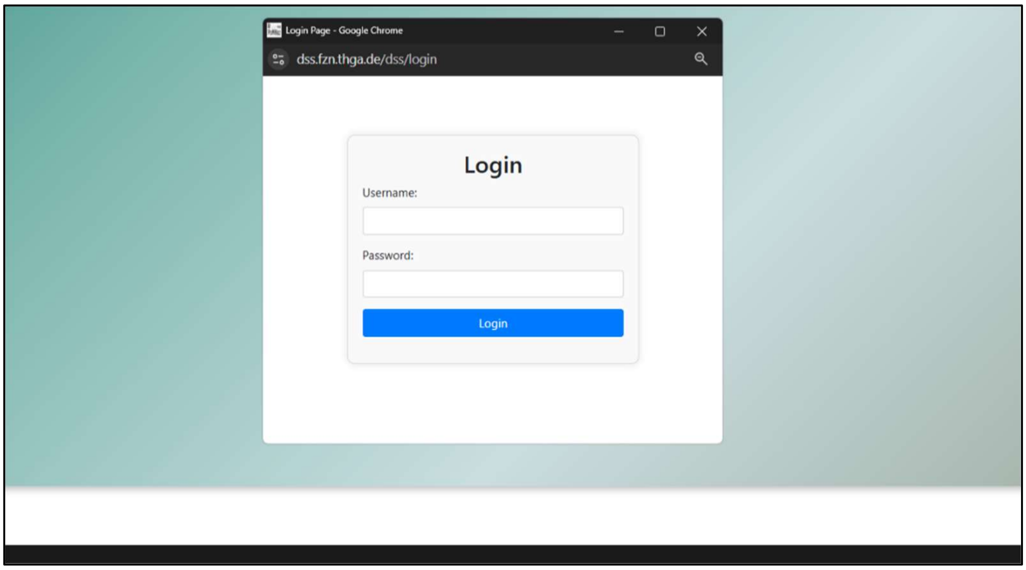
Screenshot of the interactive Web mapping highlighting the hazards and the link to access the developed sDSS

5- Prepare the Data in QGIS

Steps:

- **Load GIS Data**: Open **QGIS** and load your GIS data (e.g., shapefiles, raster data, or other vector data). QGIS is used to prepare and style the geospatial data before publishing it to the web.
- **Style Layers**: Apply symbology to the data to define how the layers will look when rendered on the web (e.g., color schemes for different features). The styles will later be applied when serving these layers through **GeoServer**.
- **Export the Data**: Once the data is prepared, export the layers in a format that GeoServer can use, such as:
    - **Shapefiles** or **GeoJSON** for vector data
    - **GeoTIFF** for raster data

  This data will be served over the web via GeoServer.



Screenshot of the Access Point to Decision Support Systems (DSS) via the Interface

6- Set Up GeoServer

GeoServer is used to publish the geospatial data you prepared in QGIS and make it accessible via web services (WMS, WFS).

Steps:

- **Install GeoServer**: Download and install GeoServer from geoserver.org.
- **Add PostGIS Data Source** (Optional): If your data is stored in a PostgreSQL/PostGIS database, configure GeoServer to access it.
- **Create a Workspace**: A workspace in GeoServer organizes layers. Create a workspace for your project.
- **Publish Layers**: Add your data (e.g., shapefiles, GeoJSON, GeoTIFF) to GeoServer by creating a new data store. Then, publish these data stores as layers that can be accessed via WMS (Web Map Service) or WFS (Web Feature Service) URLs.

- **Generate WMS/WFS Links**: GeoServer will generate URLs to your data layers, which will be used by Leaflet to display the map in your web interface.

7- Python for Backend Data Processing

Python can be used for additional geospatial data processing, automation, or integration with the backend. You can use Python libraries such as **GDAL**, **Fiona**, **Shapely**, and **GeoPandas** to preprocess geospatial data before serving it via GeoServer.

Example Use Case:

- Process Data with Python: we used Python to clean, merge, or analyze geospatial data. For example:

```
import geopandas as gpd

# Load shapefile and reproject it to WGS84 (EPSG:4326)
gdf = gpd.read_file("data/land_parcels.shp")
gdf = gdf.to_crs(epsg=4326)

# Save the processed data as GeoJSON for GeoServer
gdf.to_file("data/land_parcels.geojson", driver='GeoJSON')
```

8- **Set Up Leaflet for Web Mapping**

Leaflet is a lightweight, open-source JavaScript library used for creating interactive maps on the web. You'll embed Leaflet in your HTML interface to interact with the data published by GeoServer.

- **Create HTML Structure**: Set up the HTML file with a container (<div>) for the map.

```
html
<!DOCTYPE html>
<html>
<head>
  <title>Web Mapping Interface</title>
  <link rel="stylesheet"
href="https://unpkg.com/leaflet@1.7.1/dist/leaflet.css" />
  <script
src="https://unpkg.com/leaflet@1.7.1/dist/leaflet.js"></script>
</head>
<body>
  <h1>Interactive Web Map</h1>
  <div id="map" style="width: 100%; height: 500px;"></div>

  <script>
    // Initialize the map and set its view to your preferred center
and zoom level
    var map = L.map('map').setView([51.505, -0.09], 13);

    // Add a tile layer to the map (this example uses OpenStreetMap)
    L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png',
{
      attribution: '© OpenStreetMap contributors'
    }).addTo(map);
  </script>
```

```
</body>
</html>
```

- **Add GeoServer Layers (WMS/WFS)**: Use the WMS or WFS URLs generated by GeoServer to add your layers to the Leaflet map.

```
javascript
Code kopieren
<script>
  // Initialize the map
  var map = L.map('map').setView([51.505, -0.09], 13);

  // Add OpenStreetMap base layer
  L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {
    attribution: '© OpenStreetMap contributors'
  }).addTo(map);

  // Add WMS layer from GeoServer
  var wmsLayer =
L.tileLayer.wms('http://localhost:8080/geoserver/wms', {
    layers: 'workspace:layername',
    format: 'image/png',
    transparent: true,
    attribution: "GeoServer"
  }).addTo(map);
</script>
```

Replace `workspace:layername` with the actual workspace and layer name from GeoServer.

9- Integrating the Web Map in the Interface

Incorporate the map into your web interface by adding the Leaflet map container into the desired section of your HTML layout. For example:

```
html
<div id="webmap-section">
  <h2>GIS Data Visualization</h2>
  <div id="map" style="width: 100%; height: 500px;"></div>
</div>
```

You can style the map section using **CSS** to fit seamlessly into the overall web interface layout. For example:

```
css
#map {
  width: 100%;
  height: 500px;
  margin: 20px auto;
  border: 1px solid #ddd;
}
```

10- Testing and Deployment

- **Local Testing**: Test your web interface locally by running the HTML Ensure data is displayed accurately and the interactivity (zoom, pan, click events) works smoothly.

- **Deploy to a Web Server**: Once the web interface works as expected, deploy it to a live web server. Ensure that your GeoServer instance is also accessible online or within the relevant network.
- **Optional Backend Enhancements with Python**: If your project requires dynamic data updates or user interactions, you could develop a backend using **Flask** or **Django** (Python frameworks) to interact with the database and serve new data to the front-end dynamically.

With these steps, you'll have a dynamic web interface that integrates GIS data, served by GeoServer, displayed interactively using Leaflet, and managed by Python for backend processing.

Guide 2 related to Interacting with the PostgreSQL Database

Introduction

QGIS supports a wide range of file types and offers robust integration with PostgreSQL and PostGIS, making it a stored in a PostgreSQL database. The PostgreSQL database is used to store and manage spatial data related to post-mining hazards, enabling efficient retrieval and analysis of large datasets.

This guide provides a step-by-step overview of connecting QGIS to a PostgreSQL database with PostGIS enabled, along with the fundamental operations that allow seamless data interaction between these systems.

When working with pgAdmin and QGIS, ensure that your spatial data is in a compatible format for PostGIS, such as having a geometry column, to avoid issues when loading data. If you are dealing with large datasets, consider indexing your spatial data in PostGIS for improved query performance. Familiarizing yourself with QGIS tools can help you effectively manipulate and analyze the data you are working with. Additionally, regularly backing up your database is advisable to prevent data loss, especially when making significant changes or updates.

Managing and visualizing spatial data steps

Step 1: Create the POMHAZ Database in pgAdmin

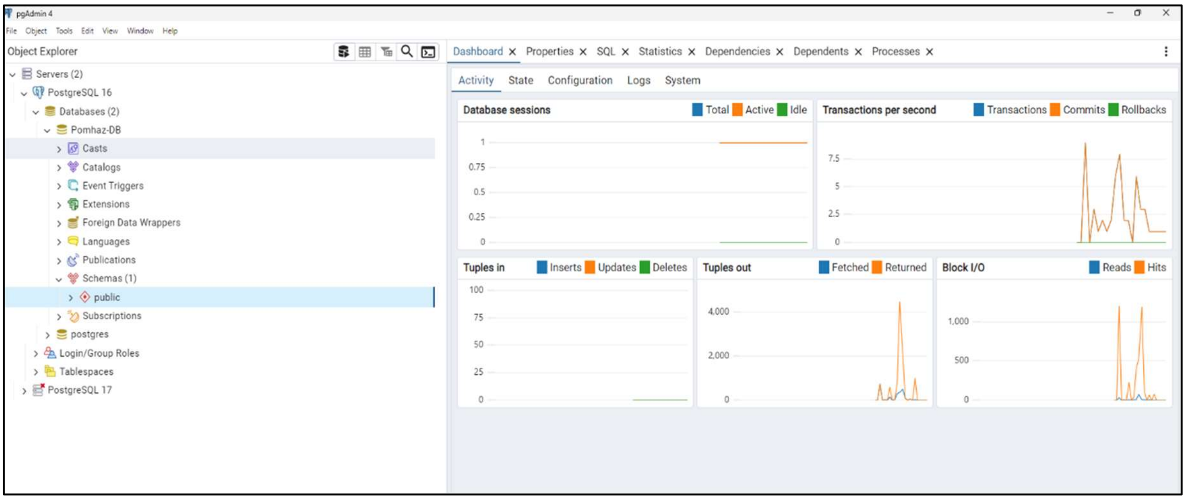
Open pgAdmin:

Launch the pgAdmin application. create a new data base and the login group role to manage and use the POMHAZ data base

Connect to the PostgreSQL Server:

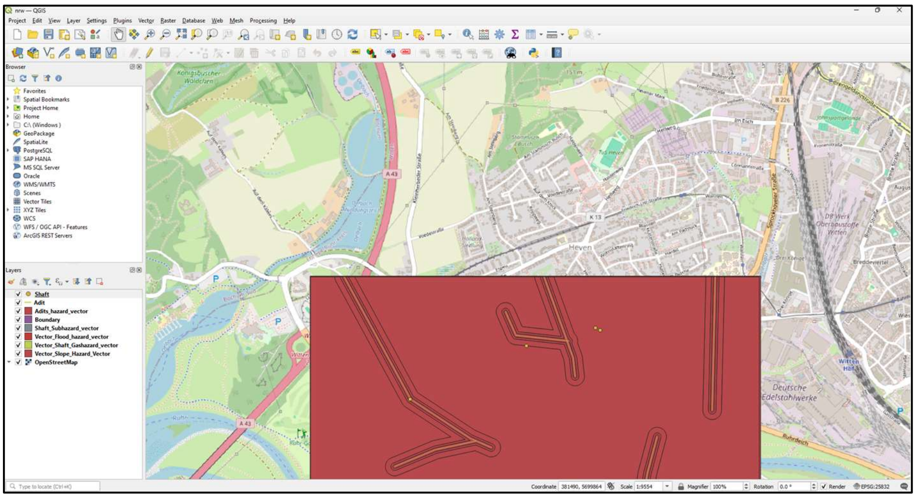
In the Browser panel, click on your PostgreSQL server. You may need to enter your password to connect.

1. **Create a New Database**:
    - Right-click on the **Databases** node in the tree view.
    - Select **Create** > **Database**.
    - In the dialog that appears:
        - **Name**: Enter POMHAZ as the database name.
        - **Owner**: Choose the appropriate owner from the dropdown (default is usually your user).
    - Click **Save** to create the database.



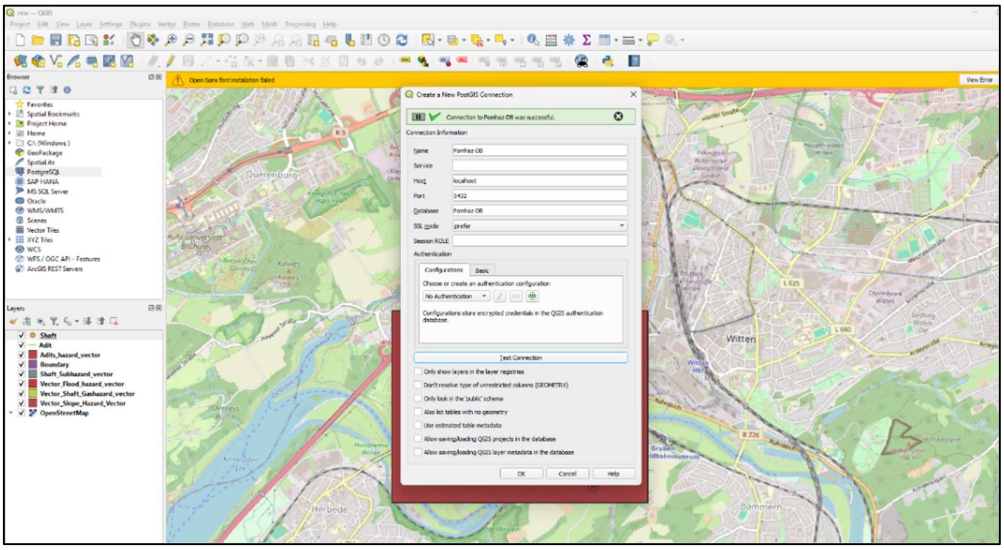
Step 2: Connect the POMHAZ Database to QGIS

1. **Open QGIS**: Launch the QGIS application and load the provided layers of the study case which will be included in the POMHAZ database.



2. **Add PostgreSQL Connection**:
    - Go to the **Browser** panel (if not visible, enable it via View > Panels > Browser).
    - Right-click on **PostgreSQL** and select **New Connection**.
    - In the connection settings dialog:
        - **Name**: Give your connection a name (e.g., POMHAZ Connection).
        - **Host**: Usually localhost if you're working on your local machine.
        - **Port**: Default is 5432.
        - **Database**: Enter POMHAZ.
        - **Username**: Enter your PostgreSQL username.
        - **Password**: Enter your PostgreSQL password.
    - Click **Test Connection** to ensure it's working, then click **OK**.

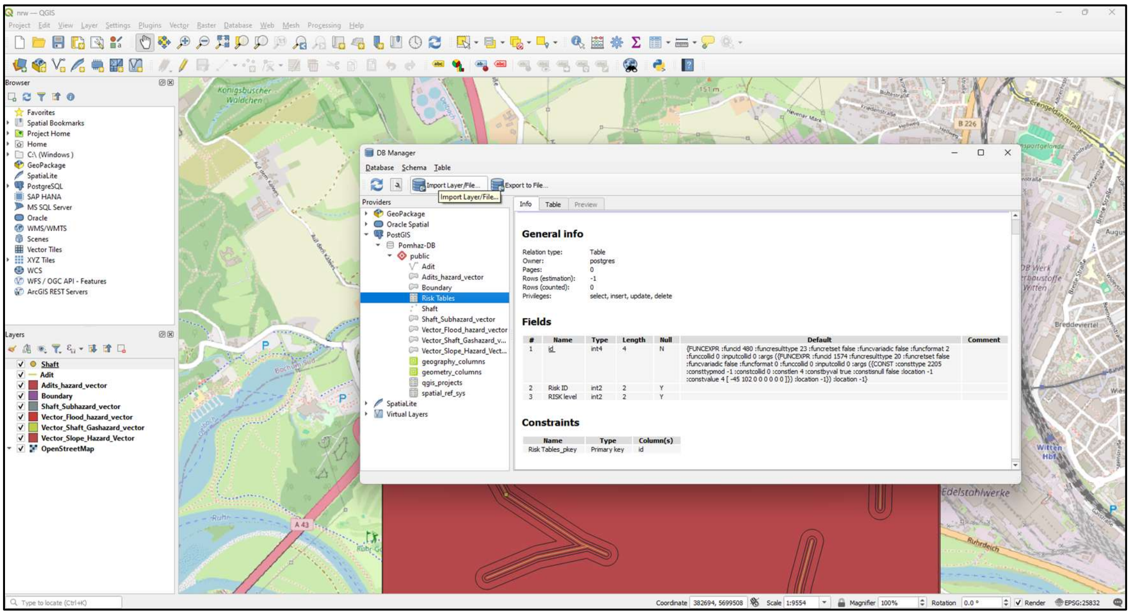
3. **Connect to the Database**:
   - After creating the connection, expand the **PostgreSQL** section in the Browser panel.
   - Click on your newly created connection to connect to the POMHAZ database.



Step 3: Load Layers for the Study Case in North Rhine-Westphalia

This step contains three sub-steps.

1. **Import Spatial Layers**:
   - If you have spatial data files (e.g., shapefiles, GeoJSON) related to the study case, you can import them into the POMHAZ database.
   - Right-click on the POMHAZ database in pgAdmin and choose **Query Tool** to open a new SQL query window. You can use the shp2pgsql tool to convert shapefiles to SQL or use QGIS to load the layers directly.
2. **Load Layers Directly in QGIS**:
   - In QGIS, go to **Layer** > **Add Layer** > **Add Vector Layer**.
   - In the dialog, select the **Source Type** (e.g., file or database).
   - If you're loading from a file, browse to your local file (shapefile, GeoJSON, etc.) and add it.
   - If you want to add layers already in your PostgreSQL? database, right-click on your PostgreSQL connection in the Browser panel, and you'll see the tables available in your POMHAZ database.
   - Select the layer(s) you need and click **Add Selected Layer(s)**.



3. **Visualize and Analyze the Data**:
    - Once loaded, you can visualize the spatial data in the QGIS map canvas.
    - Use various QGIS tools for analysis and interpretation of the data related to your study case in North Rhine-Westphalia.

This guide should assist you in creating the POMHAZ database in pgAdmin, connecting it to QGIS, and loading the necessary layers for your study case effectively.

Step 4: Backup the POMHAZ Database in pgAdmin

Backing up your POMHAZ database in pgAdmin is essential for data safety and recovery. Here's a detailed breakdown of how to perform the backup:

1. **Open pgAdmin**: If it's not already open, launch the pgAdmin application and connect to your PostgreSQL server.
2. **Locate the POMHAZ Database**:
    - In the **Browser** panel on the left side, expand the **Databases** node.
    - Find and select the POMHAZ database that you created.
3. **Initiate the Backup Process**:
    - Right-click on the POMHAZ database.
    - From the context menu, select **Backup**. This action opens the Backup dialog.
4. **Configure Backup Options**: In the Backup dialog, you'll need to set various options:
    - **General Tab**:
        - **Filename**: Specify the path and name for the backup file. For example, you might choose C:\backups\POMHAZ_backup.sql (ensure the folder exists and you have the right permissions).
        - **Format**: Choose the format for the backup file. Options include:
            - **Plain**: A plain text SQL script that can be executed to recreate the database.
            - **Custom**: A compressed format that allows for flexible restoration options.

        - **Tar**: A tarball format that can contain multiple files and is useful for larger databases.
    - **Dump Options #1 Tab**:
        - Here, you can specify what you want to back up.
            - If you want to back up only the data, check **Data**.
            - If you want to back up the schema (structure), check **Schema**.
            - You can also choose to back up roles, tablespaces, and other options as needed.
    - **Dump Options #2 Tab**:
        - This tab allows you to set additional options for the backup. You can choose to include:
            - **Use INSERT commands**: If checked, the backup will use INSERT statements rather than COPY, which might be useful for some restoration scenarios.
            - **OIDs**: Object identifiers; typically, you can leave this unchecked unless you have specific needs.
            - **Blobs**: If your database contains large objects (BLOBs), you can choose to include these in your backup.
5. **Perform the Backup**:
    - Once you've configured all the options, click the **Backup** button at the bottom of the dialog.
    - A progress window will appear, showing the status of the backup operation. Once complete, a message will indicate whether the backup was successful.
6. **Verify the Backup**:
    - Navigate to the specified backup location to ensure that the backup file was created.
    - Optionally, you can open the file (if it's in plain text format) with a text editor to verify its contents.

By following these detailed steps, you can successfully back up your POMHAZ database using pgAdmin, ensuring that your data is safe and can be restored if necessary.

Inserting and Updating Data:

- To add new records to the database, use the following SQL command:

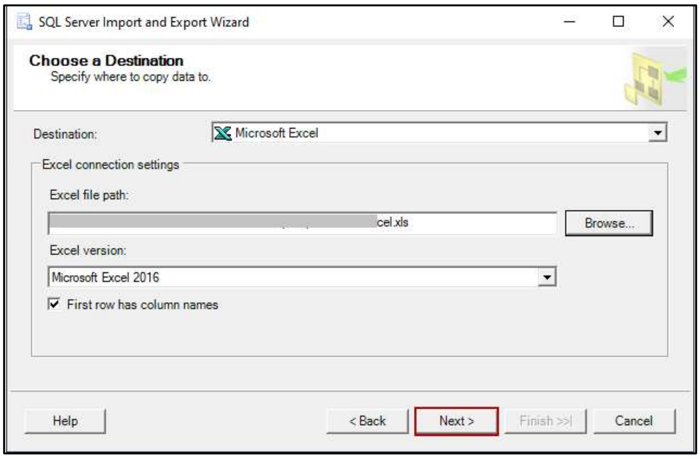
  sql
  INSERT INTO hazard_data (name, geometry, hazard_type) VALUES ('New Hazard', ST_GeomFromText('POLYGON((...))', 4326), 'subsidence');

- To update existing records:

  sql
  UPDATE hazard_data SET hazard_type = 'updated_type' WHERE id = 'specific_id';

Data Export:

- To export query results, use the "Export Data" option in pgAdmin to save your query output in formats like CSV or Excel for further analysis.



Best Practices:

- Regularly back up the database to prevent data loss.
- Use meaningful naming conventions for tables and columns to enhance data management.

Here are the following link and password for the SQL Database example used for this deliverable:
Link:https://cloud.tu-freiberg.de/s/TadDn4isMKJrDer
Password:POMHAZ2024

What is PoMHaz?

The goal of PoMHaz is to improve methodological and practical knowledge for the assessment and management of multi-hazards, at the scale of a coal mining basin, through the active and continuous engagement of key stakeholders involved in or affected by post-mining activities.

PoMHaz is a project funded by the Research Fund for Coal and Steel programme.

Further information can be found under https://www.pomhaz-rfcs.eu.

For feedback on the PoMHaz project or the published deliverables, please contact contact@pomhaz-rfcs.eu.

The PoMHaz Consortium

Public Power Corporation

INERIS maîtriser le risque pour un développement durable

Technische Hochschule Georg Agricola

GIG Instytut Badawczy

CERTH CENTRE FOR RESEARCH & TECHNOLOGY HELLAS

SRK S.A.

TECHNISCHE UNIVERSITÄT BERGAKADEMIE FREIBERG The University of Resources. Since 1765.