Software Requirements Document

Austrian Flood Monitoring and Emergency Response System

Alexander Berger, Lisa-Maria Hollaus, Christian Hummel, Erwin Lenart, Selin Meseli

1. Introduction

This document provides a detailed description of the product software, Austrian Flood Monitoring and Emergency Response System. The initial points under the section of Introduction below will focus on the purpose and general overview of this documentation with description of product scope, potential definitions (terms, acronyms and abbreviations) and references.

After the introduction, a descriptive insight into the product and its content will be presented under the Product Overview section. Hereby, the product will be portrayed regarding its functionality, potential constraints, user characteristics, possible assumptions & dependencies and requirements apportioning.

Following to 2. Product Overview, this paper will continue to deepen the context of requirements of the project under the section of Requirements.

This section will split requirements into five subsections; the first subsection from 3.1 to 3.13, will provide an insight into various interfaces (External, User, Hardware and Software Interfaces), while the second subsection 3.2 Functional will share a description of the product's functionality.

The third subsection 3.3 Quality of Service from 3.3.1 to 3.3.4 will handle service quality and touch on essential themes of performance, security, reliability and availability.

The fourth subsection 3.4 Compliance will present contents of compliance. The final subsection 3.5. Design and implementation will have a general focus on the design and the implementation of the product , where technical and organizational topics will be pointed out.

The final section, 4. Verification, will include verification approaches and methods.

1.1 Document Purpose

The main purpose of this document is to present the overall organization of the project along with its functionality and its performance. This paper should contain all necessary and critical information to be read by both the project team and the client.

1.2 Product Scope

The software being specified in this document is planned to be a platform/application to monitor flooding events in Austria and coordinate the response activities. It is also aimed for this platform to share information to public, so that local people can be informed about emergency situations regarding the flooding.

The main features of this application should contain various informative visualizations. A visualization of water levels of surface water with a map, critical areas and historical data with statistical values/chart representations should assist the users to determine risk and danger levels in parts of Austria. Emergency reporting will be done by crowdsourcing with the help of verifications and visualizations.

1.3 References

In order to prepare this document, the file AFM-Project Outline was mainly used as inspiration. However, to follow a correct structure, a very useful GitHub repo and a website were additionally utilized. References for both are provided below:

- Jam, A. (2018). *SRS-Template* [Software]. GitHub. https://github.com/jam01/SRS-Template
- Rosencrance, L. (2019). Software requirements specification (SRS). Retrieved from https://www.techtarget.com/searchsoftwarequality/definition/software-requirements-specification#:~:text=Purpose%20of%20an%20SRS,the%20teams%20are%20in%20agreement

1.4 Document Overview

As the structure of this paper is meticulously described in the Introduction part, the following parts after this section will give a profound insight into the product itself. After the product's essential components, requirements including interfaces and functional sub requirements will be presented. Next part will demonstrate the notions on the design and the implementation of the project by touching on numerous other themes. The paper will be finalized after the presentation of verification.

2 Product Overview

2.1 Product Perspective

The product is a web-based application focused on supporting Austrian citizens and emergency response organizations by providing accurate and reliable data on flood events. Offering real-time information, as well as historical data.

The platform makes use of open government data sources, but also enables the public to post flood-related emergencies reports. Users will have access to real-time data on water levels from Austrian government sources, which is visualized on a map interface to help users quickly interpret the situation. The map includes layers for current water levels, historical data, and critical zones.

The application will be optimized for both desktop and mobile use, providing flexibility for users in the field or at emergency centres without the need for a separate mobile app. Different user roles (e.g., administrators, emergency managers, agents and public users) have specific permissions, allowing emergency response managers to coordinate tasks and verify reports, while public users can view data and submit emergency reports within set limitations. This role-based setup ensures accurate data sharing and limits the risk of misinformation, making it a reliable source during flood events.

2.2 Product Functions

The platform includes several key features to support emergency response and keep the public informed:

- **Map-Based Visualization**
  - Displays real-time and historical water levels as well as critical areas.
  - Uses color-coded markers and aggregates information to make it easy to understand.
- **Emergency Reporting and Verification**
  - Allows users to report emergencies with location data, description and optional photo uploads, when logged in.
  - Reports can then be reviewed and verified by emergency managers and nearby users can upvote or downvote reports from other users.
  - Verified reports are displayed on the map for public tracking.
- **Emergency Management Dashboard**
  - Dashboard for emergency managers to oversee reports, manage incident data, and verify user reports.
  - For managers to assign tasks to emergency response agents and monitor their completion status for effective resource allocation and response.
- **User Interaction and Anti-Spam Features**
  - Provides different roles (like admin, manager, registered public user, and general user) with specific permissions for interaction and information visibility.
  - Includes anti-spam measures, like automatic bans for multiple false reports and inappropriate reports e.g. advertisements, to keep data accurate and trustworthy.

2.3 Product Constraints

The following limitations and standards will help guide the development of the platform:

- **User Interface**
  - Designed to work on both desktop and mobile devices, so a separate mobile app won't be necessary.
  - Uses role-based views, to make sure different types of users only see information they have permission to access.
- **Integration of Data**
  - Connects with Open Government Data sources (like Austrian water level portals) for up-to-date information.
- **Data Security and Quality Control**:
  - Includes anti-spam features, like automatic bans for users who submit multiple false reports.
  - Focuses on high-quality data by verifying reports and allowing feedback from users.
- **Compliance with Standards**:
  - Follows data privacy and security standards to make sure user data is managed safely and securely.

2.4 User Characteristics

The platform is designed for different types of users, each with specific roles, responsibilities, and permissions:

- **Administrator**
  - Manages user roles and oversees the entire platform, with the highest level of access.
  - Only limitation is that he cannot give his role (permissions) to other users.
- **Emergency Response Managers**
  - Responsible for creating and assigning tasks, inviting agents to assist, and verifying reports.
- **Emergency Response Agents**
  - Get tasks assigned by managers.
  - Update task statuses and report on their progress.
  - Typically, experienced professionals, such as firefighters or disaster response coordinators.
- **Registered Public Users**
  - Can submit location-based emergency reports with descriptions and images.
  - Must have a verified account to ensures accountability (real name).
- **General Public Users (Unregistered)**
  - Limited access to the platform, mainly viewing basic information such as water levels and general emergency areas.

  - They can see information but can’t submit or interact with data, helping to reduce the spread of misinformation.

2.5 Assumptions and Dependencies

The application requires continuous, reliable access to open data sources (e.g surface water and their water levels) provided by official governmental sources and trusted datasets from Austria. It depends on APIs from these sources to deliver real time and historical data. Any disruption in the data accessibility or API changes by the data providers could impact the functionality of the application.

We would like to achieve that our applications web-based design is fully responsive, which would allow it to function effectively on a wide range of devices and screen sizes. Cross-browser and cross-platform testing will ensure compatibility, but changes to browsers or mobile operating systems could require further adaptation and changes from our side.

Users of our application will have stable internet connections for real time updates. But what if e.g emergency managers and emergency response agents experience problems in their connection which could impact their ability to access data or post updates. Offline functionality would solve this problem and could be considered in the future to mitigate this risk.

Within our application we want to have different roles for each user which would change the possibility of interactions he gets. For this we need robust authentication and authorization mechanism which are provided by external providers.

2.6 Apportioning of Requirements

Live water display is one of the core functions of our application. With it you get immediate information about floods all around Austria. Through retrieving real time data from different sources we can guarantee the visualization of water levels and critical areas.

The visualization of historical data, which would be HQ30, HQ100 or all time high is done by the Data Retrieval API. It allows users to view historical flood data to understand flood trends and to do predictive planning. It’s important for users who need to analyse historical flood patterns and/or predict potential future events.

Public users should be enabled to submit flood-related incident reports via the User Interface. It collects user input which is stored for verification and analysis by emergency managers. This is a core functionality which allows crowd sourcing of emergency reports and increase the awareness during flood events.

Providing a central interface for emergency managers to coordinate response efforts is another requirement it must fulfil in the current version. It shows real time flood data which helps the communication and coordination of all people involved in emergencies. Furthermore, Role Base Access Control ensures that only authorized people can access this interface.

Offline Data Access allows users (focus would lie on emergency response agents, emergency managers) to access our application in areas with limited or no connection. Offline Data Access is not critical for initial deployment and could be addressed later in the creation process.

Functionality	Software Element	Current / Future Version
Map with visualization of water levels of surface water	Data Retrieval API, Mapping Interface	Current Version
Visualization of critical areas	Data Retrieval API, Mapping Interface	Current Version
Historical Data Access	Data Retrieval API, Historical Data	Current Version
Crowd sourcing of emergency reports	User Interface (Frontend), Admin Dashboard	Current Version
Emergency response planning	Admin Dashboard, Role Based Access Control	Current version
Offline Data Access	Local Data System	Future Version

3. Requirements

3.1 External Interfaces

This subsection defines all the inputs into and outputs requirements of the software system.

3.1.1 User interfaces

The application requires a well structured, intuitive user interface that's meets usability and convenience standards given our broad range of users, including emergency managers, emergency response agents, registered and not-registered users from the general public.

User Interface Components:

- Map Interface: It should display real-time water levels, historical data and critical zones. Users have different options while viewing this data e.g zooming in the map
- Emergency Manager Dashboard: Provides tools for incident management and report verification
- Public Reporting Interface: This allows public users to report flood-related incidents, with input for location, description and severity of the flood

Design Standards and Layout:

- Gui Standards: The interface will follow to recognized standards to ensure a consistent look and feel across the whole application
- Error Messages: The application will display clear and concise error messages to the user in case of any issue with the platform or one of its features, to fulfil usability requirements.

- Standard Elements: Navigation bars, buttons and interactive elements of the application need to be consistent and intuitive, to avoid confusion of the intended users.

Usability Requirements

- Responsive Design: The interface will adapt to both desktop and mobile devices, which ensures accessibility for users in the field and at home.
- Keyboard Shortcuts: Common shortcuts (e.g zooming) will be provided to desktop users
- Color Codes: A consistent color scheme will be used to denote water levels, critical zones and other flood risk indicators

Convenience Features

- Real-Time Data Refresh: The map will automatically update water levels and flood statuses, with a refresh rate of every few minutes
- Location Based Auto Detection: in the best case, if GPS settings from the user are activated, the application will be able to extract GPS information of reports from the post itself, but users will have to provide information about the location of the incident.

3.1.2 Hardware interfaces

The application must work with different hardware components, especially mobile devices and workstations used by emergency managers and public users.

Supported Device Types:

- Desktop and Laptop Computers: For emergency centres and users with stable internet access
- Mobile Devices: Smartphones and tablets for field agents and public users.

Control Interactions:

- Touch Input: For mobile devices, so smartphones and tablets, the application will support touch input for map interactions (e.g pinching to zoom, dragging to pan)
- Camera and Image Upload: For public incident reports, the application will support image uploads from the camera or gallery to provide visual evidence of flood conditions

Communication Protocols

- HTTP/HTTPS: All data interactions will occur over secure HTTPS to ensure data privacy and integrity
- API integration: APIs will manage data exchange with hardware, ensuring smooth functionality across different device types

3.1.3 Software interfaces

In this section the connections between the product and other software components, databases, operating systems, tools and many more will be further described.

Connections to other software Components:

- Data Source APIs: Connects with open data APIs to retrieve water levels and flood alerts
- Database Management System (DBMS): Uses a DBMS to store historical data, user reports and verified incident records
- Operating System Compatibility: The application will be browser-based but compatible with the major operating systems (Windows, macOS, Android, iOS)

Data Items and Messages:

- Input Data Items: Realtime water levels, historical flood data, public report data and emergency updates
- Output Data Items: Display real-time water levels , critical flood zones and user submitted reports
- Additional functionality for emergency managers to send a notification to emergency workers and assign them to tasks this way

3.2 Functional

This section will describe the services that the product will provide, what its behaviors are, and what the results are. The main functional requirement is that it can report data concerning flooding in Austria. The web app will also include features about emergency response tracking. It is used as a dedicated social media platform to report information of disaster.

Our website will provide a login system including different types of roles: Administrator, emergency response manager, emergency response agents and normal users. Not logged in users can still view some content, but most of it is hidden from them. Anti-spam measures will be implemented to ensure only related content is displayed. Another feature will be automatic bans for users that have submitted three faulty reports. There will be a map in the front end to visualize historical and present water levels. Ideally there might be options for users who have impaired eyesight.

Another feature will be Emergency Report – Georeferencing. The reports have coordinates and tasks that must be executed by assigned people. Emergency response managers will primarily oversee the management and validation of reports submitted by registered users. In cases where reports are determined to be incorrect or implausible, it is the responsibility of the emergency response managers to delete them.

Additionally, a verification system will be implemented through a community voting mechanism. Registered users located within proximity to a reported incident can upvote or downvote the report. Should a report's rating drop below a predetermined threshold, the system will alert an emergency response manager, who will then evaluate the report and decide whether to remove it or keep it active.

3.3 Quality of Service

The quality of service standards encompass a range of essential non-functional attributes that contribute to the overall effectiveness, and appeal of a product. These attributes have a focus on the characteristics that enhance the user experience and safeguard data. The non-functional standards include performance, security, and reliability, availability, and compliance each playing an important role in how well the system meets the needs and expectations of its users.

3.3.1 Performance

The product should be able to load and process pages fast enough to satisfy the user. Data updates of critical information will be automatically refreshed on the frontend as fast as possible. The platform should be able to allow access to it for a minimum number of users without impacting the performance in a negative way. Ideally it will be designed to support increased loads from sudden influxes of user connections during disaster events. The system should be capable of processing features like reporting, upvotes/downvotes and emergency tasks within acceptable latency limits to ensure real-time accuracy.

3.3.2 Security

The application will implement a role-based access control system to restrict permissions based on user roles. For example, administrators have more permissions than regular users. This is necessary to prevent regular users abusing sensitive functionalities for the application. Ideally the data transmitted, such as login credentials and user data, will be encrypted using HTTPS. Users are required to have strong passwords to ensure security. Users submitting more than three fake reports will experience automatic ban to prevent misuse of the web application.

3.3.3 Reliability

The application should ensure that reports, especially regarding critical areas, are processed accurately, with regular checks for data integrity by users with more permissions, such as the administrator or emergency report managers. Ideally there will be a mechanism that alerts the administrator of issues that may impact the user experience in a negative way, ensuring fast response to errors.

3.3.4 Availability

The application aims to ensure a high uptime, this is required to ensure that the population will have access to critical information about disaster events in real-time. Since geolocation services are crucial for emergency reporting, the system should provide accurate mapping services, with backup map providers if the primary service is unavailable.

3.4 Compliance

The system will comply with data privacy regulations according to the GDPR law. User data collection, storage and other processing practices will follow privacy and data protection standards. Any user of our application will be informed to what purpose their data will be used. If a user deletes their account, any data that was part of them will be deleted as well. There will be a possibility to change user data in their profile section to allow the user to update their personal information.

3.5 Design and Implementation

3.5.1 Installation

This application will be web-based, and does not need additional installation. It will be running on a public domain located in Austria. It can be accessed by typing in the corresponding url into your preferred browser.

3.5.2 Distribution

The Austrian Flood Monitoring System will be directly distributed to the users via a webserver, owned by the company. A stable deployment environment is very important for the reliability of the application and its services.

Distribution of data

Necessary information for the visualizations, e.g. surface water levels, historical data about critical areas, will be accessible for the system in a database, which will provide this information in processed form to the users.

3.5.3 Maintainability

In order to maintain the stability and efficiency of the Austrian Flood Monitor system even after the release, it has to be developed as efficiently as possible. Some of the supplementary methods to increase maintainability are listed below:

- Code has to be written in a clear and structured way and dead code should be avoided
- Boundaries of the modules and services have to be specified, for better refactoring
- The project has to be well documented, with changelogs for newer versions of the application

Maintenance tasks for the software development team include:

- Code reviews to spot potential vulnerabilities or errors in the code
- Adjustment of existing test cases and add new ones
- Performance optimization
- Bugfixes and health checks of the system

- Management of the database

3.5.4 Reusability

Certain features of this product will be reusable for future projects. A login system is common for many different software applications. The basis for the visualization schema could also be adapted and reused for similar projects.

3.5.5 Portability

This software will be developed in a windows software environment, but the data it displays to registered and non-registered users is meant to be shown to users of different operating systems as well, like linux distributions and macOS.

Furthermore, it has to be accessible for web browsers of mobile phones of different operating systems, to ensure direct access for a vast range of devices.

3.5.7 Deadline

The deadline for this Software project is on the 9th of January 2025. Until then it needs to be tested accordingly in order to be deployed to the general public and its intended users.

4 Verification and Validation

4.1 Software verification

Software verification aims to ensure consistency, completeness and correctness of the project. Three main methods to achieve this are:

- Unit Testing
- Integration Testing
- System Testing

Regular assessments of the software against the requirements are necessary for the development process. The development team will write, execute and evaluate software tests to prevent errors and bugs as early as possible. Continuous testing in the development phase will provide useful information for the developers about the progress and functionality of the project.

4.1.1 Requirements verification

Requirements verification describes the process of checking if the fully developed software meets the requirements that are collected in the development phase of the project. Each requirement has to be checked for the development process, to prove their importance to the project. The preferred method for verification will be to inspect and evaluate the requirements by the development team.

4.2 Software validation

Software validation is about evaluating the product against its requirements, to check if it will be able to fulfil its intended purpose and that functional and non-functional requirements are satisfied.

Software validation activities include:

- Functional testing
- Requirements validation
- Documentation validation

4.2.1 Functional testing

Functional testing does not only contain tests to check for functionality of the code, it also checks for other design aspects like usability and performance of the product.

4.2.2 Requirements validation

Requirements validation is about the user requirements for the system. The main goal is to reduce the likelihood of errors and defects before the product is being shipped. Techniques to make sure that the system features align with the requirements involve:

- Requirement reviews
- Prototyping
- Test-case generation

Requirements validation also intends to specify requirements for stakeholders and developers, to avoid ambiguity and misconceptions about the software. There will be multiple iterations of requirements validations and therefore regular reviews and meetings with the client are highly recommended, as the system requirements can change in the development process or even after the release of the product. These changes will be documented and evaluated once they are forwarded to the development team to maintain consistency of the requirements.

4.2.3 Documentation validation

Documentation validation checks for accuracy and actuality of the documents regarding this product.