Design Specifications - Machine Learning Platform for Intelligent Water Systems Management - Final Project

Robert Castro
Calvin Chau
Yvan Michel Kemsseu Yobeu
Laila Velasquez
Kassandra Vera

Friday, May 9, 2025

Contents

1 Introduction

1.1 Purpose

The Design Specifications document provides a comprehensive breakdown of the architecture, components, and features of the Intelligent Water Systems Management Platform. It is intended to serve as a reference for developers and stakeholders during implementation and testing phases.

1.2 Scope

This document covers the design and functional components of the platform, including database architecture, API specifications, and user interface design.

1.3 Design Goals

- Optimize water usage through real-time monitoring
- Provide predictive analytics for maintenance
- Enhance user experience with intuitive dashboards

2 System Design Overview

2.1 Architecture Design

The platform is built on a client-server model using Flask for backend APIs and SQL for database management.

2.2 Component Design

- **Frontend**: HTML, CSS, JavaScript for interactive dashboards
- **Backend**: Flask for routing and API management
- **Database**: SQL for structured data storage
- **Deployment**: Docker for containerized services

3 Breakdown of Pages and Components

3.1 Home Dashboard

Displays real-time water usage, health status, and alerts.

3.2 Analytics Page

Provides historical data visualization and comparative analysis.

3.3 Settings

Allows users to configure sensors, set thresholds for alerts, and manage account settings.

3.4 Reports

Generates detailed water consumption reports in PDF and CSV format.

3.5 API Endpoints

- **GET /api/v1/water-usage** - Retrieves real-time water usage data.
- **POST /api/v1/alerts** - Sets alert thresholds for leak detection.
- **GET /api/v1/reports** - Downloads usage reports.

4 Functional Requirements

4.1 Sensor Data Ingestion

Real-time collection of water usage data.

4.2 Real-time Monitoring

Display live data on dashboards.

4.3 Alerts and Notifications

Automated alerts for anomalies (e.g., leaks).

4.4 Data Visualization

Graphical representation of water consumption.

5 Non-Functional Requirements

- **Performance**: Capable of handling large datasets
- **Scalability**: Expandable to multiple regions
- **Security**: Encrypted data storage and secure API communication

6 Technical Specifications

6.1 Database Schema

Optimized for fast querying and retrieval.

6.2 API Endpoints

RESTful design for sensor data, user info, and reporting.

6.3 Docker and Flask Integration

Containerized services for scalability.

7 UI/UX Design Specifications

7.1 Dashboard Layout

Intuitive and user-friendly.

7.2 User Interactions

Real-time updates and interactive graphs.

8 Testing and Validation

- **Unit Testing**: For individual components
- **Integration Testing**: For combined modules
- **Performance Testing**: For large-scale usage

9 Deployment Strategy

9.1 Docker Configuration

Defined services for frontend, backend, and database.

9.2 Cloud Hosting Setup

For scalability and accessibility.