Smart Campus Irrigation Monitoring System – Final Specification

Holon Institute of Technology (HIT)

Department of Computer Science

Submitted by: Yoav Levavi, Adane Avia, Hen Tatro, Netanel Zohar, Amit Yehoshafat, Shay Dahari

Date: 20.10.2025

Table of Contents

1. Introduction

The smart campus irrigation monitoring system is a full-stack web application developed to improve the management and control of irrigation systems across a university campus.
It integrates sensors, database management, and modern web technologies to provide a smart, data-driven solution for efficient and sustainable plant maintenance.

The system allows real time monitoring of soil moisture and temperature across different buildings and zones, helping maintenance teams make informed irrigation decisions.
Its goal is to replace outdated manual or timer based monitoring with a responsive and centralized digital platform that helps optimize irrigation decisions.

Developed as part of the final Computer Science project at HIT, this work combines concepts learned throughout the degree - including software engineering, databases, algorithms, and web development.
The project showcases how theoretical academic knowledge can be applied to create a real world, functional system that addresses environmental and operational challenges.

Project Goals

• Develop a centralized platform for managing irrigation data and infrastructure.
• Provide real time visualization of soil moisture and temperature levels.
• Enhance water efficiency by making data driven irrigation decisions.
• Offer an intuitive and minimalistic interface for campus maintenance teams.

2. Problem Definition

Traditional irrigation systems in campuses are usually manual or based on fixed timers, which do not reflect real soil or weather conditions.
This leads to water waste, inconsistent irrigation, and higher maintenance costs.
Since there is no centralized system, data from different zones is not collected or analyzed efficiently, making decision-making slow and unreliable.

Maintenance staff often rely on visual checks instead of real-time data, which increases workload and reduces accuracy.
As a result, the current approach is inefficient, unsustainable, and hard to manage across large areas.

The smart campus irrigation monitoring system aims to fix these problems using automation and real-time monitoring.

The Need for Smart Irrigation

With campuses becoming more digital and sustainability focused, there is a clear need for data driven irrigation management.
A smart system can automatically adjust watering schedules based on live soil and weather data, reducing both waste and manual effort.

Using sensors, databases, and an interactive dashboard, the Smart Campus system helps maintenance teams make informed decisions, improve efficiency, and promote environmental responsibility - all while keeping daily operation simple and intuitive.

3. System Architecture

Our system follows a three-tier architecture, separating the system into three main layers - Client, Server, and Database.
This structure improves scalability, maintainability, and clarity between system components.

As shown in the figure below, the Client (React) communicates with the Server (Express) through API calls, while the server interacts with a MySQL database that stores all irrigation-related data.

3.1 Client (Frontend)

The client side is built using React with Vite as the development environment and TailwindCSS for styling.
It provides a responsive, minimalistic, and user-friendly interface that allows maintenance teams to:

- View real time sensor data (moisture and temperature).
- Navigate between maps, sensors, and buildings.
- Manage irrigation data easily.

The frontend fetches data from the server via RESTful APIs and displays it using dynamic components that update automatically when new sensor data arrives.

3.2 Server (Backend)

- The server is developed using Node.js with Express.js, serving as the bridge between the user interface and the database.
  It handles requests, processes logic, and returns data to the client in JSON format.
  The backend includes:
- Controllers for handling API endpoints (sensors, maps, buildings).
- Services that manage data processing and validation.
- Routes that define structured endpoints for clean and modular development.
- This layer ensures secure and efficient communication between the frontend and the database.

3.3 Database (MySQL)

The database is built with MySQL and designed using a normalized schema to prevent data duplication and maintain consistency.
It stores all entities such as:

- Sensors
- Plants
- Maps
- Buildings
- Measurements

Relationships between these tables ensure that each sensor and measurement is linked to its respective map and building, supporting easy data retrieval and scalable future growth.

3.4 Data Flow

1. The user interacts with the React frontend (selects a building or views a map).
2. The frontend sends a request to the Express server via a REST API.
3. The server processes the request, interacts with MySQL, and retrieves the relevant data.
4.The data is returned to the client and displayed in real-time using visual components.

This modular structure ensures that each part of the system can be improved independently without affecting other layers.

4. Database Design

Our system is built using MySQL, following normalization principles (up to 3NF) to ensure efficient data storage, integrity, and quick query performance.
It contains six core entities - Buildings, Floors, Maps, Sensors, Plants, and Measurements, along with a supporting Users table for future role management.

This relational structure allows the system to easily track every irrigation-related data point, from the building level down to individual sensor measurements.

4.1 Entity Relationships

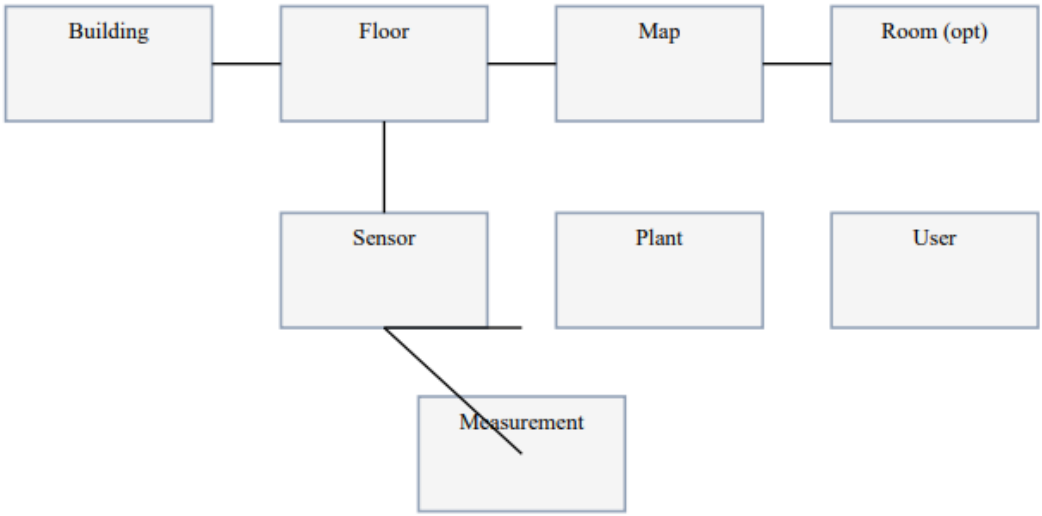
As shown in the diagram below, the main hierarchy is:
Building → Floor → Map → Sensor → Measurement

- A Building contains one or more Floors.
- Each Floor can have one or more Maps, representing different irrigation zones.
- A Map contains multiple Sensors, each associated with soil moisture or temperature.
- Each Sensor produces multiple Measurements over time.
- A Plant is linked to a Sensor in a one-to-one relationship, representing the specific plant type and its watering threshold.

This design allows both real-time monitoring and historical trend analysis of irrigation data.

4.2 Advantages of This Design

- Supports real time and historical data visualization.
- Enables data driven decision making based on accurate sensor feedback.
- Simplifies maintenance and updates with a clear one-to-many structure.
- Prepares the system for future expansion such as multi-campus integration or user-based permissions.



Key Fields (selection):

Table	Important Columns
buildings	id, name
floors	id, building_id, name
maps	id, floor_id, name, image_url
sensors	id, map_id, name, type (moisture/temperature), location_x, location_y
plants	id, sensor_id, species?, watering_threshold
measurements	id, sensor_id, value, unit, timestamp
users	id, username, role (future use)

5. UI and UX Design

The user interface of the system was designed with a strong emphasis on clarity, simplicity, and usability.
Using React and TailwindCSS, the system provides a modern, responsive, and visually balanced layout that adapts seamlessly across desktop and mobile devices.

The overall design focuses on minimalism - removing unnecessary visual clutter and allowing users to easily access information at a glance.

Navigation is managed through a fixed sidebar layout, giving users consistent access to all system sections without confusion or page reloads.

Main UI Pages

- MapView - Presents an interactive campus map showing real-time sensor data.
  Each sensor is represented by a color-coded indicator (blue for normal moisture, red for dry zones, gray for offline).
  Users can click on sensors to view detailed measurements.
- Sensors - Displays a comprehensive list of all sensors across the campus, including their type, latest readings, and operational status.
  The interface allows quick filtering and refreshing, giving maintenance teams a fast overview of system health.
- Buildings - Enables administrators to manage campus infrastructure, including adding new buildings, floors, and maps.
  Each change is automatically reflected across the platform, ensuring an up to date digital representation of the irrigation network.

Smooth animations, transitions, and hover effects provide a more engaging user experience, while preserving the system's professional and lightweight design.
The result is an intuitive, fast, and enjoyable dashboard that supports efficient daily operation even for users without a technical background.

6. Conclusions

The Smart Campus project gave us the opportunity to combine what we learned in different courses into one practical system.
We managed to build a functional platform that helps monitor and manage irrigation across the campus in a simple and smart way.

The system proves how technology can make real impact by saving water, improving efficiency, and making maintenance easier.
Working on it taught us a lot about full-stack development, teamwork, and how to turn an idea into a working product.

Overall, this project was both a learning experience and something we're proud of.