

TechTrio Project Summary

Project Manager: {unlisted in [2025 Project Summary.docx](#)}

Project Overview

TechTrio is an AI-based Smart Irrigation System designed to help farmers monitor environmental conditions and automate irrigation decisions. The system uses IoT sensors, edge computing, and machine learning to determine when crops need watering and to activate irrigation automatically.

The platform collects real-time data from field sensors, processes it using a TinyML model running on embedded hardware, and sends information to a central server and dashboard. Farmers can monitor soil and environmental conditions, manage irrigation systems, and review historical data through a web-based dashboard.

The system improves water efficiency, reduces manual labor, and helps maintain optimal crop health by ensuring that irrigation occurs only when needed.

Target Users

The system is designed for the following primary users.

Farmers

Farmers use the dashboard to monitor soil and environmental conditions on their fields. They can view sensor readings, track irrigation activity, and monitor system health.

Farm Managers / Administrators

Administrators manage systems deployed across different fields or farms. They can configure systems, manage users, review analytics, and generate reports.

Technical Operators

Technical users may configure hardware, maintain sensors, and troubleshoot system connectivity or performance issues.

Main System Features

Real-Time Sensor Monitoring

The system continuously collects environmental data from field sensors, such as:

- Soil moisture levels
- Soil temperature
- Air temperature
- Humidity
- Other environmental metrics depending on sensor configuration

These readings are transmitted to the central system for monitoring and analysis.

Automated Irrigation Control

The irrigation pump is automatically controlled based on predictions made by the embedded AI model. The system determines whether irrigation is required by analyzing sensor data.

Possible actions include:

- Activating the irrigation pump when soil moisture is too low
- Keeping irrigation off when conditions are sufficient
- Logging irrigation activity for monitoring and reporting

Dashboard Monitoring

The web dashboard allows users to view and manage the irrigation system through a browser.

Dashboard features include:

- Real-time sensor readings
- Irrigation system status
- Historical data visualization
- System configuration tools
- Alerts and notifications

The interface provides a centralized control panel for managing multiple systems or fields.

Notifications and Events

The system can generate events and notifications for important conditions such as:

- Irrigation activation
- Sensor errors or failures
- System connectivity problems
- Critical environmental conditions

These notifications help users respond quickly to potential issues.

Reporting and Data Export

The system stores historical sensor data and irrigation activity. Users can generate reports and export data for analysis.

Reports may include:

- Sensor readings over time
- Irrigation frequency and duration
- System performance statistics
- Environmental trends

System Architecture

TechTrio uses a distributed IoT architecture consisting of edge devices, a central server, and a web interface.

Main components include:

- Field hardware devices (sensors and controllers)
- Edge AI processing (TinyML model)
- Network communication using MQTT
- Backend server and API
- Web dashboard for monitoring and control

This architecture allows real-time decision making at the edge while maintaining centralized monitoring and management.

Core Hardware Components

The system relies on embedded devices deployed in the field.

Raspberry Pi

Acts as the central local controller and gateway. It collects data from sensors, communicates with other devices, and connects to the server.

ESP32 Microcontroller

Runs the TinyML inference model and controls irrigation actions such as activating the water pump.

Sensors

Sensors collect environmental and soil data used by the AI model to determine irrigation needs.

Possible sensors include:

- Soil moisture sensors
- Temperature sensors
- Humidity sensors

Water Pump / Irrigation System

The pump is automatically activated when the system determines irrigation is necessary.

Software Tools and Libraries

The system is built primarily using Python and IoT communication frameworks.

Core technologies include:

Programming Languages

- Python (primary system logic and backend components)

Embedded Systems

- ESP32 microcontroller firmware
- TinyML inference code for edge AI processing

Machine Learning

- TinyML model used to predict irrigation requirements

Networking

- MQTT messaging protocol for communication between devices and server

Web System

- Server-rendered user interface
- REST / JSON API endpoints for system communication

MQTT Communication

The system uses MQTT topics for communication between devices and the central server.

MQTT is used to:

- Send sensor readings
- Publish irrigation decisions
- Transmit system status information
- Trigger events and alerts

This lightweight messaging protocol allows reliable communication between distributed IoT devices.

System Operation Workflow

The irrigation system operates through a sequence of automated steps.

Step 1 – Initialization

The system boots and initializes hardware components, sensors, and network connections.

Step 2 – Sensor Reading

Environmental data is collected from the sensors installed in the field.

Step 3 – Feature Generation

Collected data is processed and formatted into features suitable for machine learning inference.

Step 4 – Model Inference

The TinyML model analyzes the sensor data to determine whether irrigation is required.

Step 5 – Pump Activation

If the model predicts that irrigation is needed, the system activates the irrigation pump.

Step 6 – Data Publishing

Sensor readings and system actions are published to the server via MQTT for monitoring and storage.

API System

The backend provides a JSON-based API used by the dashboard and connected systems.

Key API areas include:

Authentication and Accounts

- User login and account management

System Management

- Management of irrigation systems and field configurations

Sensor Management

- Registration and monitoring of sensors

Readings and Time-Series Data

- Storage and retrieval of sensor data

Notifications and Events

- Event tracking and alert management

Reports and Data Exports

- Generation of system and environmental reports

Admin and Diagnostics

- Administrative tools and system diagnostics

User Dashboard

The web dashboard is the main interface for interacting with the system.

Main dashboard capabilities include:

- User login and account management
- Monitoring system status
- Viewing sensor data
- Managing irrigation systems
- Reviewing analytics and reports

The dashboard allows farmers and administrators to manage the irrigation system remotely through a standard web browser.

System Setup and Startup

Before using the system, the following prerequisites must be met:

Hardware

- Raspberry Pi gateway installed in the field
- ESP32 controller connected to sensors
- Irrigation pump connected to the controller

Network

- Internet connection for the gateway device

Software

- Server environment running the dashboard and API

Startup process typically includes:

1. Powering the hardware units
2. Establishing network connectivity
3. Launching the backend server
4. Accessing the dashboard via a web browser
5. Logging in and configuring systems

Troubleshooting

Common troubleshooting areas include:

Hardware Issues

- Sensor not responding
- Pump not activating

Network Issues

- MQTT connection failures
- Internet connectivity problems

Software Issues

- Server errors
- API communication failures

Diagnostics tools and logs are used to identify and resolve system problems.

TinyML Model Training and Deployment

The AI component uses TinyML to enable machine learning inference on resource-limited embedded devices.

Key stages include:

- Training a machine learning model using environmental data
- Converting the model for embedded deployment
- Deploying the model to the ESP32 device
- Running real-time inference on incoming sensor data

This approach allows fast decisions directly on the device without relying on cloud processing.