Smart Water System

Phase 5: Project Documentation & Submission

In this part you will document your project and prepare it for submission.

Document the Smart Water Management project and prepare it for submission.

Documentation

Describe the project's objectives, IoT sensor setup, mobile app development, Raspberry Pi integration, and code implementation.

Include diagrams, schematics, and screenshots of the IoT sensors and mobile app.

Explain how the real-time water consumption monitoring system can promote water conservation and sustainable practices. .

Submission

Share the GitHub repository link containing the project's code and files.

Provide instructions on how to replicate the project, deploy IoT sensors, develop the transit information platform, and integrate them using Python.

Include example outputs of Raspberry Pi data transmission and mobile app UI.

Solution :

Documentation

The project's objectives of IoT sensor setup, mobile app development, Raspberry Pi integration, and code implementation

1.IOT SENSOR SETUP

The IoT sensor setup for a smart water system involves:

1. Choosing appropriate sensors for water monitoring.
2. Establishing a reliable network for data transmission.
3. Placing sensors strategically in the water system.
4. Collecting and securely transmitting real-time data.
5. Analyzing data for insights and decision-making.

6. Setting up alerts for system irregularities.

7. Integrating sensor data with control systems for automated adjustments.

8. Enabling remote monitoring and management.

9. Implementing robust security measures to protect data integrity and privacy.

2.MOBILE APP DEVELOPMENT

Developing a mobile app for a smart water system involves planning and designing the app's features for water monitoring, selecting the appropriate platform for development, integrating features like real-time monitoring and alerts, creating an intuitive user interface, ensuring secure data integration, conducting thorough testing, implementing robust security measures, gathering user feedback for continuous improvement, and deploying the app on app stores with regular maintenance and updates.

3.RASPBERRY PI INTEGRATION

Integrating Raspberry Pi into a smart water system can offer various functionalities, including:

Connecting Raspberry Pi with sensors for monitoring water levels, quality, and flow rates.

Using Raspberry Pi for data processing, analysis, and visualization of water-related parameters.

Enabling automated control of water management systems, such as pumps, valves, and filtration systems, based on sensor data.

Facilitating remote monitoring and management of the smart water system through the Raspberry Pi interface.

Ensuring secure transmission of data to and from the central management system or cloud platform.

Creating a user-friendly interface on Raspberry Pi for users to access real-time data and control the water system efficiently.

Linking Raspberry Pi to IoT networks to enable seamless communication with other devices and systems within the smart water network.

Implementing robust security protocols on Raspberry Pi to safeguard data integrity and protect against potential cyber threats within the smart water system.

4.CODE IMPLEMENTATION

The code implementation involves three main components:

IoT Device Firmware: Code on the IoT devices to read sensor data and transmit it to the platform.

Backend Server: Code to receive, store, and process incoming data.

Frontend Interface: Code for the user interface to display real-time environmental data.

Diagrams, schematics, and screenshots of the IoT sensors and mobile app.

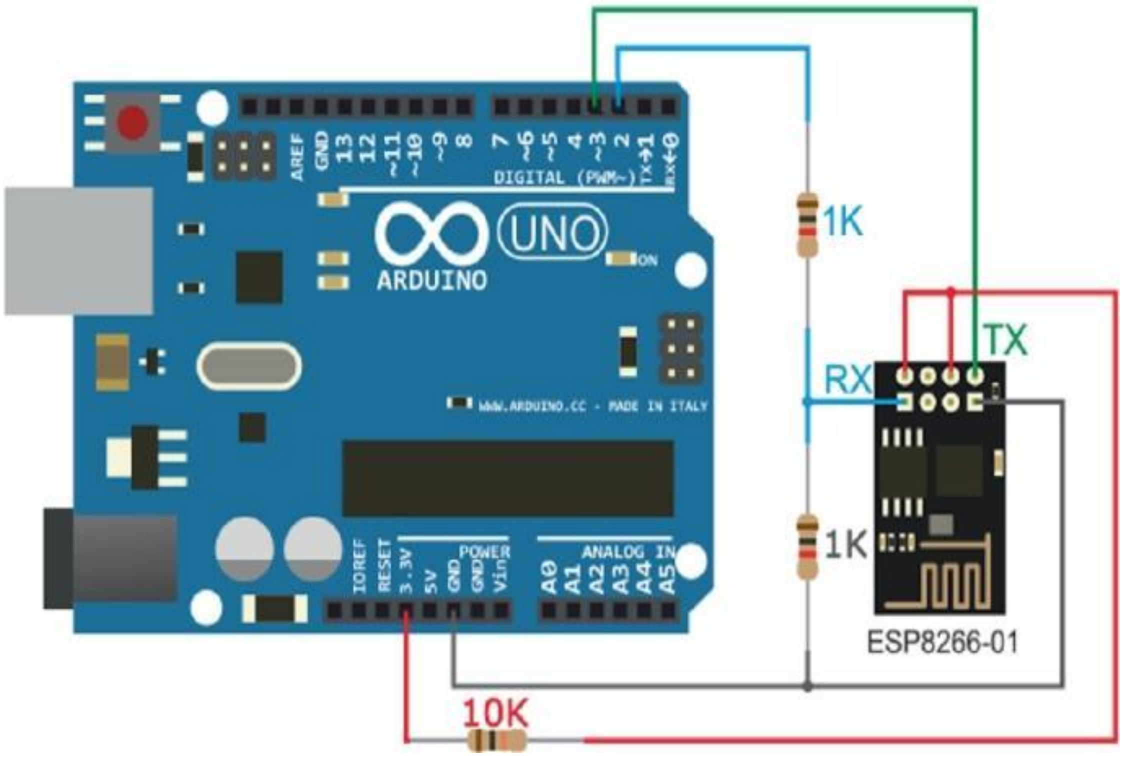
IOT Sensors:

1. Water Flow Sensor: This sensor measures the flow rate of water passing through a pipe or system. It typically uses a paddle wheel or turbine to calculate the flow rate.

2. Water Level Sensor: This sensor detects the water level in tanks, reservoirs, or other water storage devices. It can use ultrasonic, pressure, or capacitive techniques to measure the water level accurately.

3. Water Quality Sensor: This sensor assesses various parameters of water quality such as pH level, conductivity, temperature, turbidity, and dissolved oxygen. It helps monitor the water's suitability for different applications.

4. Leak Detection Sensor: This sensor detects leaks in plumbing systems or pipelines by monitoring changes in pressure or the presence of water in areas where it shouldn't be.



Mobile App:

The mobile app serves as a user interface for interacting with the smart water management system. It enables users to monitor and control various aspects of the system. Here are some key features typically found in such apps:

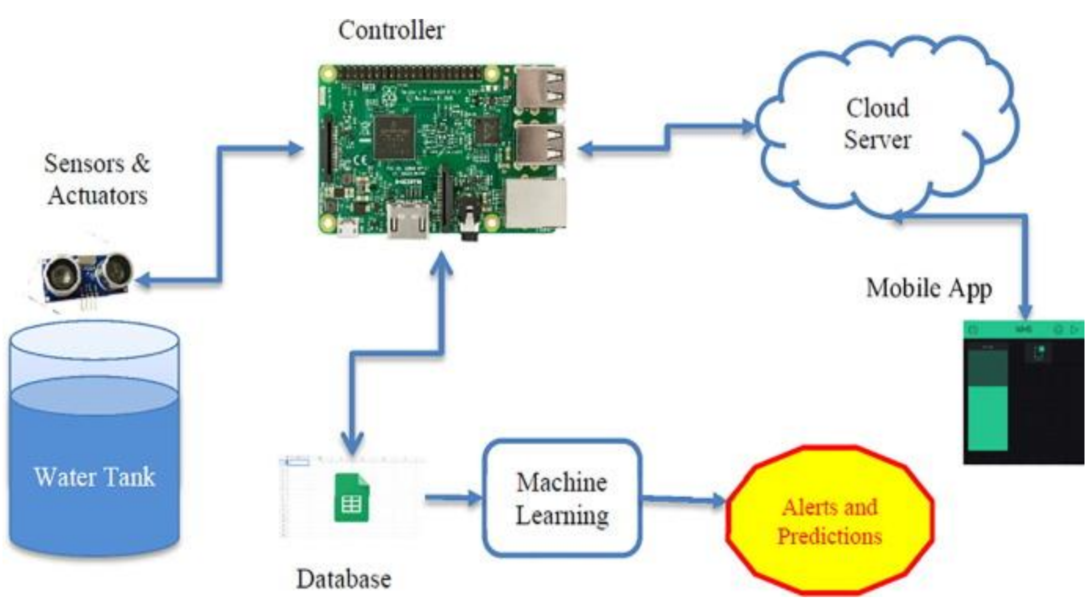
1. Real-time Monitoring: The app provides real-time data on water flow rate, water level, and water quality. It displays this information in intuitive graphical formats for easy comprehension.

2. Alert Notifications: The app sends notifications to the user's mobile device when abnormal conditions are detected, such as leaks, low water levels, or poor water quality. This enables prompt action to be taken.

3. Historical Data Analysis: The app stores historical data collected by the IoT sensors and presents it in the form of charts or graphs. Users can analyze trends and patterns to gain insights into water consumption or identify potential issues.

4. Remote Control: The app allows users to remotely control certain aspects of the system, such as turning on/off pumps, adjusting flow rates, or activating irrigation systems. This provides convenient and efficient management options.

5. Customization and Settings: Users can customize system parameters and set thresholds for different alerts. They can also configure preferences, such as preferred units of measurement or time intervals for data updates.



SUBMISSION

GitHub Repository Link –https://github.com/rajkumar7819/V.Rajkumar

https://github.com/rajkumar7819/Iot-phase-2

INSTRUCTIONS:

1. IOT Sensors Deployment :

   - Choose appropriate IoT sensors for the transit data you wish to collect.

   - Set up the sensors in the desired locations, ensuring proper connectivity and power supply.

   - Configure the sensor to send data to a centralised server or cloud platform.

2. Transit Information Platform Development :

   - Define the scope and features of your transit information platform.

   - Choose a suitable technology stack (e.g., Django, Flask, or FastAPI for backend development, and React, Angular, or Vue.js for the frontend).

   - Develop APIs for data retrieval and processing.

   - Implement user authentication and authorization mechanisms for secure access.

3. Integration Using Python :

   - Utilize Python to handle data from the IoT sensors and the transit information platform.

   - Implement data ingestion and processing scripts to manage the data received from the sensors.

- Develop scripts to feed the processed data into the transit information platform’s database or APIs.

- Establish communication channels between the IoT sensors and the platform using Python libraries like Requests or MQTT.

Example outputs of Raspberry Pi data transmission and mobile app UI

Here are simplified examples of what Raspberry Pi data transmission might look like and a basic representation of a mobile app UI:

1. Raspberry Pi Data Transmission Example:

```
{
  “sensor_id”: “001”,
  “timestamp”: “2023-11-01 12:00:00”,
  “temperature”: 25.6,
  “humidity”: 55.2,
  “location”: {
    “latitude”: 37.7749,
    “longitude”: -122.4194
  }
}
```

2. Basic Mobile App UI Representation :

The UI might include components such as:

- Header with app name and navigation options

- Real-time data display for temperature and humidity

- Map view displaying the location of the sensor
- Buttons for user interaction, such as refreshing data or settings