**SMART WATER MANAGEMENT**

**Objectives:**

The objective of a smart water system project is to monitor and manage water resources efficiently using IoT technology. Key goals typically include real-time data collection, analysis, and user-friendly control through a mobile app. This project aims to improve water conservation, quality, and distribution.

**IoT Sensor Setup:**

* Water Quality Sensors: Deploy sensors to monitor various water quality parameters, such as pH levels, turbidity, and chemical contaminants.
* Water Flow Sensors: Install flow meters to measure water consumption, detect leaks, and control water distribution.
* Water Level Sensors: Use ultrasonic or pressure sensors to measure water levels in reservoirs, tanks, or rivers.
* Temperature Sensors: Monitor water temperature for various applications, such as heating and cooling systems or environmental studied.
* Humidity Sensors: Measure humidity levels to prevent water damage or mold growth in sensitive areas.
* Pressure Sensors: Install pressure sensors in pipes to monitor water pressure and detect anomalies.

**Mobile App Development:**

****

To Develop a user-friendly mobile application to provide users with real-time access to data and control over the smart water system. Key features may include:

* Data Visualization: Display real-time data from sensors in an easy-to-understand format, including charts and graphs.
* Alerts and Notifications: Send notifications for critical events like leaks, low water levels, or poor water quality
* Remote Control: Allow users to remotely control water systems, such as turning pumps on/off or adjusting temperature settings.
* Historical Data: Store and present historical data for analysis and decision-making.
* User Management: Implement user accounts, access controls, and permissions to manage who can control and access the system.

**Raspberry Pi Integration:**

****

**Raspberry pi**

Use Raspberry Pi as a central hub to collect data from IoT sensors and communicate with the mobile app. Raspberry Pi can be connected to the sensors via GPIO pins or other interfaces. Additionally, it may be used for data processing and storage. You can also employ the MQTT protocol to facilitate communication between the Raspberry Pi and the IoT devices.

**Code Implementation:**

Here's a simplified overview of code implementation for a smart water system:

* Sensor Data Collection: Write code to interface with each sensor and collect data. This may involve libraries and APIs provided by sensor manufacturers.
* Data Processing: Use Python or other suitable languages to process and normalize sensor data.
* MQTT Integration: Set up an MQTT broker on the Raspberry Pi, and write code to publish sensor data to MQTT topics.
* Mobile App Development: Develop a mobile app (iOS and Android) using appropriate frameworks (e.g., React Native or Flutter) with MQTT support to subscribe to sensor data.
* User Interface: Design and implement the app's user interface to display real-time data and provide control options.
* Alerts and Notifications: Add code to send push notifications to the mobile app based on sensor readings and system events.
* Database Integration: Store historical data in a database (e.g., SQLite or PostgreSQL) for analysis and reporting.

Security: Implement authentication and authorization for user access and data encryption for secure communication.

* Testing and Deployment: Thoroughly test the system, both the hardware and software components. Once tested, deploy the system in the target environment.
* Monitoring and Maintenance: Set up remote monitoring capabilities for the Raspberry Pi and sensors, and establish a maintenance plan for long-term operation.

Real-time water consumption monitoring systems can play a significant role in promoting water conservation and sustainable practices in various ways:

**Awareness and Accountability**:

* These systems provide users with real-time data on their water consumption, making them more aware of their usage patterns.
* Users are more likely to reduce waste and adopt efficient practices when they can see how their actions directly affect water consumption.

**Leak Detection:**

* Real-time monitoring can detect leaks as they occur, helping users take immediate action to fix them.
* By preventing or quickly repairing leaks, these systems help reduce water wastage.

**Targeted Conservation:**

* Users can set consumption targets or receive alerts when their water usage exceeds a certain threshold.
* This encourages individuals and organizations to make informed decisions about water use.

**Remote Control:**

* Users can remotely control water-related devices, such as irrigation systems, to optimize water use based on real-time data.

**Data-Driven Decision-Making:**

* Governments and water utilities can use data collected from these systems to make informed policy decisions.
* Monitoring can be used for resource allocation, identifying areas with high consumption, and planning for drought conditions.

Here are instructions on how to replicate a real-time water consumption monitoring project, including IoT sensor deployment, developing a monitoring platform, and integrating them using Python:

**Replicating the Real-time Water Consumption Monitoring Project:**

**Select IoT Sensors:**

* Choose appropriate water flow sensors and quality sensors based on your project requirements. Common brands include Arduino, Adafruit, and Seeed.



**Arduino**

**IoT Sensor Deployment:**

* Install the selected sensors at relevant points in the water system.
* Connect sensors to a Raspberry Pi or microcontroller board using GPIO pins or appropriate interfaces.

**Developing the Monitoring Platform:**

* Use Python to write code for data collection and processing on the Raspberry Pi.
* Implement a MQTT broker on the Raspberry Pi to facilitate communication.
* Create a database (e.g., SQLite or PostgreSQL) to store sensor data.
* Set up MQTT subscriptions to receive data from sensors.

**Develop a Mobile App:**

****

* Develop a mobile app (iOS and Android) using a framework like Flutter or React Native.
* Implement MQTT support in the app to subscribe to sensor data.
* Design the app's user interface to display real-time data and provide control options.

**Integration Using Python:**

* Write Python code to transmit sensor data from the Raspberry Pi to the MQTT broker.
* Configure the mobile app to receive data from the MQTT broker.
* Develop features for user interaction, such as setting alerts and remote control.

**Example Python Program (Raspberry Pi Data Transmission):**

import paho.mqtt.client as mqtt

# Connect to the MQTT broker

client = mqtt.Client()

client.connect("mqtt.broker.com", 1883

# Publish water consumption data

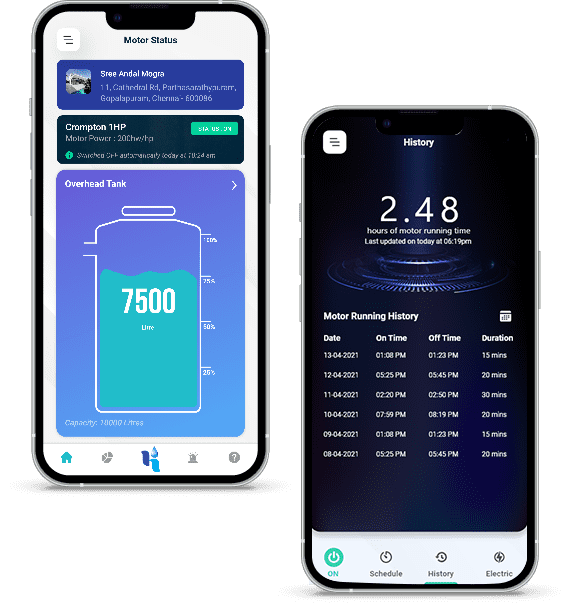
data = "Water consumption: 10 liters"

client.publish("water/consumption", data)

# Close the connection

client.disconnect()

**Example output using IOT MOBILE APP:**



An IoT mobile app is software that integrates data and serves as the touchpoint that makes IoT accessible to users. In simple terms, it provides the interface through which we can control and monitor smart devices.

**Conclusion:**

The incorporation of Internet of Things (IoT) and smart technology has contributed to the improvement of monitoring system. There are different approaches in the development and implementation of online surface water quality monitoring system to provide real-time data collection with lower operating cost.