**Project Title**

**PHASE 5:Smart Water Fountain**

**OBJECTIVE:**

The main objectives of a smart water fountain project are to:

* **Improve water quality:**

Smart water fountains can use sensors to monitor water quality parameters such as temperature, pH, and turbidity. This data can be used to identify and address potential water quality problems early on, before they have a chance to impact human health.

* **Reduce water waste:**

Smart water fountains can use sensors to detect water leaks and automatically shut off the water supply. They can also be programmed to dispense water only when needed, which can help to reduce water waste.

In addition to these general objectives, specific smart water fountain projects may have additional objectives, such as:

* **Reduce maintenance costs**
* **Improve data collection and analysis**
* **Promote public health**

**Here are some specific examples of how smart water fountain projects can be used to achieve these objectives:**

* **Improving water quality**
* **Reducing water waste**
* **Improving convenience for users**
* **Reducing maintenance costs**
* **Collecting data and improving data analysis**
* **Promoting public health**

Overall, smart water fountain projects have the potential to improve water quality, reduce water waste, improve convenience for users, reduce maintenance costs, collect data and improve data analysis, and promote public health.

**IOT setup**

**components:**

**Sensors:** Smart water fountains use a variety of sensors to collect data about their operation and the water they dispense.

Common sensors include:

* + Water level sensors to monitor the water level in the fountain
  + Water quality sensors to monitor parameters such as temperature, pH, and turbidity

**Raspberry Pi Pico:**

This will be the processor of your project

**Relay or Transistors:**

To control the water pump based on sensor inputs.

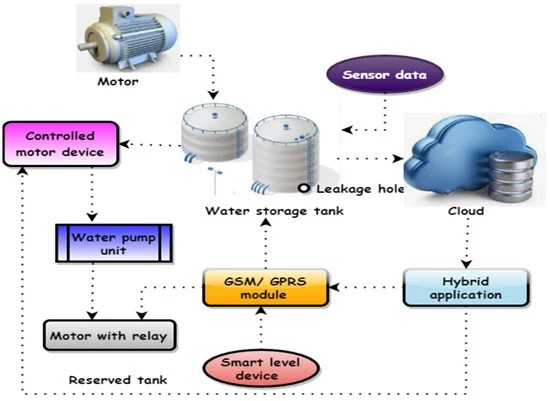
**Power supply:** The power supply provides power to the microcontroller, sensors, and other components of the smart water fountain.

The specific IoT device setup for a smart water fountain will vary depending on the specific features and functionality of the fountain. For example, a smart water fountain with advanced water quality monitoring capabilities may require additional sensors and processing power.

Here Is a diagram of a typical IoT device setup for a smart water fountain:

[Sensors] -> [Processor] -> [cloud(relay)] -> [pump(output)]

The sensors collect data about the water fountain and send it to the processor.it processes the data and using cloud it control the flow of water using relay .



**Platform development**

The development of an IoT platform for smart water fountains typically involves the following steps:

* **Requirements gathering:**

The first step is to gather requirements from the stakeholders, such as the water fountain manufacturer, the water utility, and the end users. These requirements will define the features and functionality of the IoT platform.

* **System design:**

Once the requirements have been gathered, the next step is to design the system architecture of the IoT platform. This includes defining the components of the platform, such as the cloud platform, the edge devices, and the communication network.

* **Hardware development:**

If necessary, new hardware may need to be developed for the IoT platform. This may involve developing new software app to control it with mobile

* **Software development:**

The software for the IoT platform needs to be developed to collect data from the sensors, control the operation of the water fountains, and communicate with the cloud using the Mobile phone.

* **System integration:**

The next step is to integrate the hardware and software components of the IoT platform. This includes configuring the sensors, Rpi and communication modules to work together.

* **Deployment and testing:**

Once the IoT platform has been integrated, it needs to be deployed and tested in a real-world environment. This involves installing the hardware and software at the water fountains and testing the system to ensure that it is working as expected.

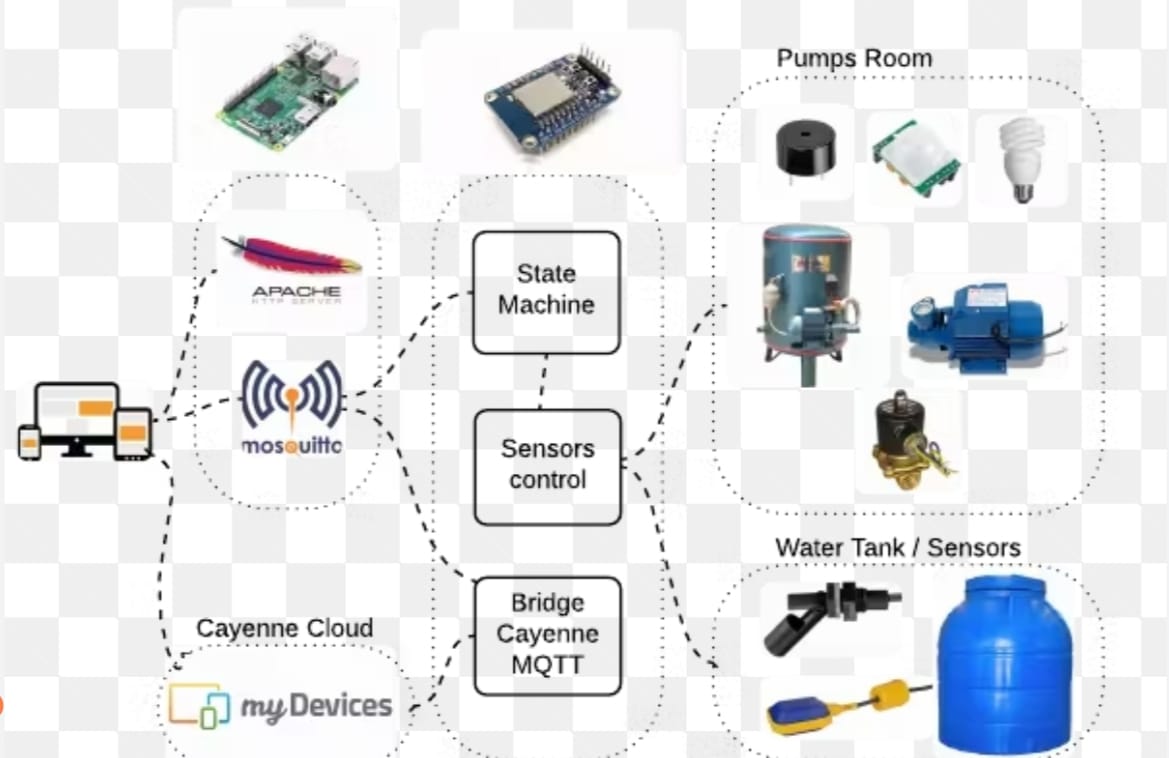
* **Maintenance and support:**

Once the IoT platform is deployed, it needs to be maintained and supported to ensure that it continues to operate reliably. This may involve providing software updates, fixing bugs, and providing technical support to the users of the platform.

The following are some of the key considerations for developing an IoT platform for smart water fountains:

* **Security**
* **Scalability**
* **Reliability**
* **Ease of use**

Overall, an IoT platform for smart water fountains can help to improve the water quality, reduce water waste, improve convenience for users, and reduce maintenance cost.



MOBLIE APP DEVELOPMENT:

Write the code for your mobile app, including the logic for sending commands to the raspberry pi,receiving updates , and displaying the fountains status .ensure that the app can handle different states , such as starting ,stopping ,and error handling.

TEST YOUR APP:

Thoroghly test your mobile app, both locally and with the raspberry pi,ensure that it can connect to your raspberry pi and control the smart water fountain as intended.

DEPLOY THE MOBLIE APP:

Publish your mobile app on the respective app stores(google play store for android and apple app store for ios ) . this will make it available to users for download.

**\*Once you’ve completed these steps ,users can download the app , connect it to your raspberry pi,and control the water fountain remotely .make sure to maintain and update your app and iot system as needed to provide a seamless user experience.**

MOBILE APP CODE :

import requests

import RPi.GPIO as GPIO

# Define the Raspberry Pi's IP address

raspberry\_pi\_ip = 'your\_pi\_ip\_address'

ir\_sensor\_pin = 17

GPIO.setmode(GPIO.BCM)

GPIO.setup(ir\_sensor\_pin, GPIO.IN)

try:

ir\_sensor\_state = GPIO.input(ir\_sensor\_pin)

if ir\_sensor\_state == GPIO.HIGH:

print('IR sensor detected an object')

else:

print('IR sensor did not detect an object')

except KeyboardInterrupt:

pass

finally:

GPIO.cleanup()

WEBDEVELOPMENT TECHNOLOGIES USING JAVASCRIPT CODE:

<html>

<head>

<title>Smart Water Fountain</title>

<style>

/\* CSS for styling the fountain (same as before) \*/

/\* ... \*/

.water-flow.stopped {

animation-play-state: paused;

}

</style>

</head>

<body>

<h1>Smart Water Fountain</h1>

<div class="fountain">

<div class="water-flow"></div>

</div>

<button id="startButton">Start Fountain</button>

<button id="stopButton">Stop Fountain</button>

<input type="range" id="flowRate" min="1" max="10" value="5">

<p>Flow Rate: <span id="flowRateValue">5</span></p>

<script>

const waterFlow = document.querySelector(".water-flow");

const startButton = document.getElementById("startButton");

const stopButton = document.getElementById("stopButton");

const flowRateSlider = document.getElementById("flowRate");

const flowRateValue = document.getElementById("flowRateValue");

// Simulate an IR sensor

let irSensorState = false;

function updateIRSensorState() {

// You can replace this with your actual IR sensor reading logic

// For demonstration, we're alternating between true and false.

irSensorState = !irSensorState;

if (irSensorState) {

startButton.disabled = false;

stopButton.disabled = false;

} else {

startButton.disabled = true;

stopButton.disabled = true;

}

}

// Call the function to update the IR sensor state periodically

setInterval(updateIRSensorState, 1000); // Adjust the interval as needed

let currentFlowRate = 5; // Default flow rate

startButton.addEventListener("click", () => {

if (irSensorState) {

waterFlow.style.animationPlayState = "running";

}

});

stopButton.addEventListener("click", () => {

waterFlow.style.animationPlayState = "paused";

});

flowRateSlider.addEventListener("input", () => {

currentFlowRate = flowRateSlider.value;

flowRateValue.textContent = currentFlowRate;

const animationDuration = 10 / currentFlowRate; // Inverse of flow rate

waterFlow.style.animationDuration = `${animationDuration}s`;

});

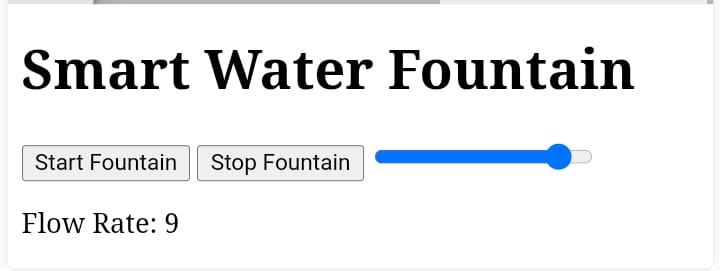
</script>

</body>

</html>

OUTPUT :

The coding is in the language of java script



**Conclusion:**

Smart water fountains offer a number of benefits, including improved water quality, reduced water waste, improved convenience for users, and reduced maintenance costs. By using IoT technology, smart water fountains can be monitored and controlled remotely, and data can be collected to identify areas for improvement.