SMART WATER FOUNTAIN

Technology used:

IOT,Advanced smart water fountain system,sensors, control unit,voice control,data monitoring ,water filtration and purification

steps

Conceptualization and Planning

⇩

Design and Engineering

⇩

Component Selection

⇩

Software Development

⇩

Prototyping

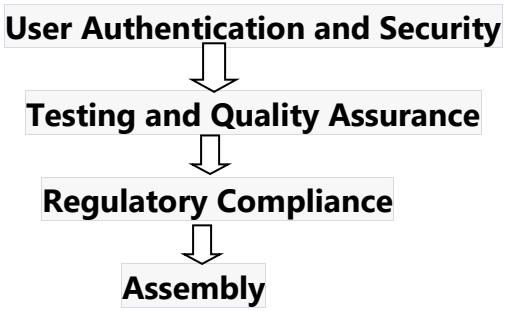
⇩

Water Filtration and Purification

⇩

Voice Control and IoT Integration

⇩



Developing a smart water fountain involves multiple steps, from conceptualization and design to prototyping and final production. Here are the general steps to develop a smart water fountain

1. **Conceptualization and Planning:**
- Define the purpose and goals of the smart water fountain. Is it for personal use, public spaces, or commercial applications?
- Identify the key features and functionalities you want to incorporate into the fountain.
- Consider the target audience and user needs.
- Determine a budget and timeframe for development.

2. **Design and Engineering:**
- Develop a detailed design for the smart water fountain, including the physical design, components, and electronics.
- Create 2D and 3D CAD (Computer-Aided Design) models.
- Collaborate with industrial designers and engineers to ensure the design is functional and aesthetically pleasing.

3. **Component Selection:**
- Choose the sensors, pumps, valves, microcontrollers, and other electronic components required for the fountain.
- Select the materials for the fountain's construction, considering factors like durability and water resistance.

4. **Software Development:**
- Develop the firmware and software that control the fountain's operation.
- Implement user interface design, connectivity with mobile apps, and remote control features.

5. **Prototyping:**
- Build a prototype of the smart water fountain to test the design and functionality.
- Test the integration of sensors, pumps, valves, and the control system.

- Gather user feedback and make necessary improvements.

6. **Water Filtration and Purification:**
- Integrate water filtration and purification systems, if applicable.
- Ensure the water quality meets safety and taste standards.

7. **Voice Control and IoT Integration:**
- If applicable, integrate voice control and IoT capabilities for remote monitoring and control.

8. **User Authentication and Security:**
- Implement user authentication mechanisms to restrict access to the fountain if needed.
- Ensure data security and privacy.

9. **Testing and Quality Assurance:**
- Conduct rigorous testing to ensure all components work as intended.
- Test for durability and reliability under different conditions.

10. **Regulatory Compliance:**
- Ensure the smart water fountain complies with relevant safety and environmental regulations.

IOT Code for water fountain

python
```
import random

def measure_ph():
    # Simulate measuring pH (replace with actual sensor reading)
    return random.uniform(6.5, 7.5)

def measure_turbidity():
    # Simulate measuring turbidity (replace with actual sensor reading)
    return random.uniform(0.1, 1.0)

def measure_temperature():
    # Simulate measuring temperature (replace with actual sensor reading)
    return random.uniform(10.0, 20.0)

def check_water_quality():
    ph = measure_ph()
    turbidity = measure_turbidity()
    temperature = measure_temperature()

    # Define acceptable ranges for water quality parameters
```

```
    ph_range = (6.0, 8.0)
    turbidity_range = (0.0, 1.0)
    temperature_range = (5.0, 25.0)

    # Check the water quality and provide feedback
    if ph_range[0] <= ph <= ph_range[1]:
        ph_status = "within acceptable range"
    else:
        ph_status = "outside acceptable range"

    if turbidity_range[0] <= turbidity <= turbidity_range[1]:
        turbidity_status = "within acceptable range"
    else:
        turbidity_status = "outside acceptable range"

    if temperature_range[0] <= temperature <= temperature_range[1]:
        temperature_status = "within acceptable range"
    else:
        temperature_status = "outside acceptable range"

    print(f"pH: {ph} ({ph_status})")
    print(f"Turbidity: {turbidity} ({turbidity_status})")
    print(f"Temperature: {temperature} ({temperature_status})")

if __name__ == "__main__":
    check_water_quality()
```

Detect leakes

```
    python
    import random
    import time

    # Simulated data collection from the water fountain sensor
    def collect_fountain_data():
        while True:
            # Simulated data: water flow rate
            flow_rate = random.uniform(0.5, 2.0)  # liters per minute

            yield {
                "flow_rate": flow_rate,
                "timestamp": time.time()  # Simulated timestamp
            }
            time.sleep(1)  # Simulated data update interval

    # Detect leaks in the water fountain
    def detect_leaks(data):
        consecutive_low_flow_count = 0
        threshold = 0.2  # Adjust this threshold based on your specific setup

        for reading in data:
            flow_rate = reading["flow_rate"]
```

```
        timestamp = reading["timestamp"]

        if flow_rate < threshold:
            consecutive_low_flow_count += 1
            if consecutive_low_flow_count >= 5:
                print(f"Possible leak detected at {timestamp}. Flow rate: {flow_rate} L/min")
        else:
            consecutive_low_flow_count = 0

# Main program
if __name__ == "__main__":
    fountain_data = collect_fountain_data()
    detect_leaks(fountain_data)


import time
import RPi.GPIO as GPIO

# Define GPIO pins for sensors and actuators
water_level_sensor_pin = 17  # Pin for water level sensor
button_pin = 18  # Pin for the button to dispense water
refill_notification_pin = 27  # Pin for refill notification LED

# Initialize GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(water_level_sensor_pin, GPIO.IN)
GPIO.setup(button_pin, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(refill_notification_pin, GPIO.OUT)

def check_water_level():
    # Check the water level sensor status
    return GPIO.input(water_level_sensor_pin)

def dispense_water():
    # Simulate dispensing water
    print("Water dispensed")
    time.sleep(5)  # Simulated dispensing time

def notify_refill():
    # Turn on the refill notification LED
    GPIO.output(refill_notification_pin, GPIO.HIGH)
    time.sleep(5)  # LED notification time
    GPIO.output(refill_notification_pin, GPIO.LOW)

try:
    while True:
        if not check_water_level():
            print("Low water level. Please refill the fountain.")
            notify_refill()

        if not GPIO.input(button_pin):
```

```
            dispense_water()

        time.sleep(1)  # Check sensors and button every second

except KeyboardInterrupt:
    GPIO.cleanup()
```

HTML

```
<!DOCTYPE html>
<html>
<head>
    <title>Smart Water Fountain Control</title>
    <link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
    <h1>Smart Water Fountain Control</h1>
    <button id="dispenseButton">Dispense Water</button>
    <div id="status"></div>
    <script src="script.js"></script>
</body>
</html>
```

CSS

```
body {
    text-align: center;
    background-color: #f0f0f0;
}

h1 {
    color: #333;
}

button {
    background-color: #007bff;
    color: white;
    border: none;
    padding: 10px 20px;
    cursor: pointer;
}
```

JAVA SCRIPT

```
document.getElementById('dispenseButton').addEventListener('click', function() {
    fetch('control.php', {
        method: 'POST',
        body: JSON.stringify({ action: 'dispense' })
    })
    .then(response => response.json())
    .then(data => {
        document.getElementById('status').innerHTML = data.message;
    })
    .catch(error => {
        document.getElementById('status').innerHTML = 'Error: ' + error.message;
    });
```

```
});
```

Php

```
<?php
header('Content-Type: application/json');

$data = json_decode(file_get_contents('php://input'), true);

if ($data && isset($data['action'])) {
    if ($data['action'] === 'dispense') {
        // Implement code to dispense water (e.g., control GPIO pins).
        $response = ['message' => 'Water dispensed successfully'];
    } else {
        $response = ['message' => 'Invalid action'];
    }
} else {
    $response = ['message' => 'Invalid request'];
}

echo json_encode($response);
```

Conclusion:

In essence, a smart water fountain is a testament to the integration of technology and sustainability in our daily lives, offering a more convenient and environmentally conscious way of accessing clean and refreshing drinking water. It's an exciting example of how innovation can enhance even the most fundamental aspects of our existence..