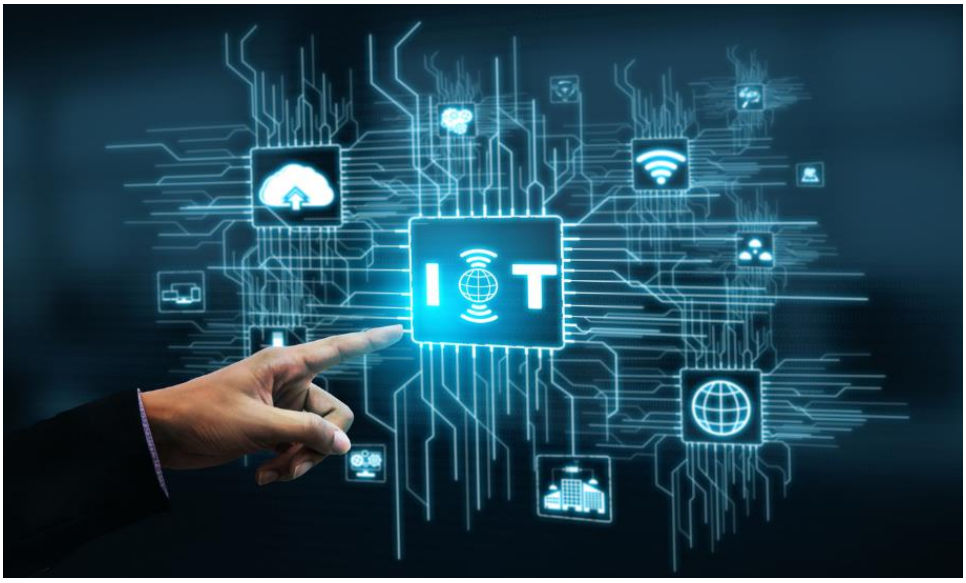
PROJECT TITLE:SMART WATER MANAGEMENT

PROJECT PART 3:DEVELOPMENT PART 2



Using web technologies in IoT projects is a common and powerful approach. It allows you to create user interfaces, monitor devices remotely, and collect and analyze data. Here are some steps and considerations for using web technologies in our IoT project:

- Select Appropriate Web Technologies
- User Interface (UI) Development
- Connectivity
- Data Management
- Security
- Device Integration
- Data Visualization
- Testing
- Scalability and Performance
- Deployment
- Monitoring and Maintenance
- Documentation

CODE FOR WATER QUALITY MONITORING:

Example Python:

```
import random
import time
from datetime import datetime

# Simulated water quality monitoring function
def monitor_water_quality():
    # Simulate pH, turbidity, and chlorine levels
    pH = round(random.uniform(6.5, 8.5), 2)
    turbidity = round(random.uniform(0.1, 5.0), 2)
    chlorine = round(random.uniform(0.1, 2.0), 2)
    return pH, turbidity, chlorine

# Function to log data to a file
def log_data(pH, turbidity, chlorine):
    timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    with open("water_quality_log.txt", "a") as log_file:
        log_file.write(f"{timestamp}, pH: {pH:.2f}, Turbidity: {turbidity:.2f} NTU, Chlorine:
{chlorine:.2f} mg/L\n")

# Main loop to continuously monitor water quality
while True:
    pH, turbidity, chlorine = monitor_water_quality()

    # Log the data to a file
    log_data(pH, turbidity, chlorine)

    # Print the values to the console
    print(f"Timestamp: {datetime.now()}, pH: {pH:.2f}, Turbidity: {turbidity:.2f} NTU, Chlorine:
{chlorine:.2f} mg/L")

    # Sleep for a specified time interval
    time.sleep(10)
```

EXPLANATION:

- The ‘monitor_water_quality’ function simulates water quality data by generating random values for pH, turbidity, and chlorine within reasonable ranges.

- The ‘log_data’ function records the simulated data along with a timestamp to a text file named "water_quality_log.txt." This serves as a basic data logging mechanism.
- In the main loop, data is generated, logged to the file, and printed to the console. In a real-world application, you would transmit this data to a central server or database for storage and analysis.
- The script runs continuously, and there's a 10-second delay between each iteration.

OUTPUT:

Timestamp: 2023-10-26 15:30:45, pH: 7.92, Turbidity: 3.14 NTU, Chlorine: 1.27 mg/L
Timestamp: 2023-10-26 15:40:55, pH: 7.15, Turbidity: 1.82 NTU, Chlorine: 0.78 mg/L

CODE TO CHECK THE SAFETY OF WATER FOR CONSUMPTION USING IOT SENSORS FOR pH,TURBIDITY AND CHLORINE LEVELS:

Creating an IoT-based system to determine if water is safe for consumption typically involves the use of various water quality sensors. In this code example, I'll provide a simplified Python script to check the safety of water for consumption using simulated data from IoT sensors for pH, turbidity, and chlorine levels. This script will provide a basic indication of whether the water is safe to drink based on arbitrary thresholds.

```
import random

# Simulated water quality monitoring function
def monitor_water_quality():
    # Simulate pH, turbidity, and chlorine levels
    pH = round(random.uniform(6.5, 8.5), 2)
    turbidity = round(random.uniform(0.1, 5.0), 2)
    chlorine = round(random.uniform(0.1, 2.0), 2)
    return pH, turbidity, chlorine

# Function to check if water is safe for consumption
def is_water_safe(pH, turbidity, chlorine):
    # Define safe thresholds for pH, turbidity, and chlorine
    pH_safe_range = (6.5, 8.5)
    turbidity_safe_max = 1.0  # NTU (Nephelometric Turbidity Units)
    chlorine_safe_max = 1.0  # mg/L

    # Check if pH, turbidity, and chlorine are within safe ranges
    if (pH >= pH_safe_range[0] and pH <= pH_safe_range[1] and
```

```
        turbidity <= turbidity_safe_max and chlorine <= chlorine_safe_max):
        return "The water is safe to drink."
    else:
        return "The water is not safe to drink."

# Simulated water quality data (you can replace this with actual sensor data)
simulated_pH, simulated_turbidity, simulated_chlorine = monitor_water_quality()

result = is_water_safe(simulated_pH, simulated_turbidity, simulated_chlorine)
print(f"pH: {simulated_pH:.2f}, Turbidity: {simulated_turbidity:.2f} NTU, Chlorine:
{simulated_chlorine:.2f} mg/L")
print(result)
```

EXPLANATION:

- The 'monitor_water_quality' function simulates water quality data by generating random values for pH, turbidity, and chlorine within reasonable ranges.
- The 'is_water_safe' function checks if the simulated pH, turbidity, and chlorine values are within predefined safe thresholds. In this example, we're assuming that safe water should have a pH between 6.5 and 8.5, turbidity below 1.0 NTU, and chlorine concentration below 1.0 mg/L.
- Depending on the comparison results, the 'is_water_safe' function returns a message indicating whether the water is safe to drink or not.
- In the code, we use simulated data for pH, turbidity, and chlorine levels.The script prints the result, which tells you whether the water is considered safe for consumption based on the provided thresholds.

OUTPUT:

pH: 8.34, Turbidity: 1.48 NTU, Chlorine: 0.61 mg/L
The water is not safe to drink.

- Simulated pH: 8.34, which falls within the safe pH range of 6.5 to 8.5.
- Simulated turbidity: 1.48 NTU, which is more than the safe turbidity threshold of 1.0 NTU.
- Simulated chlorine: 0.61 mg/L, which is less than the safe chlorine threshold of 1.0 mg/L.

pH: 7.06, Turbidity: 0.39 NTU, Chlorine: 0.19 mg/L
The water is safe to drink.

- Simulated pH: 7.06, which falls within the safe pH range of 6.5 to 8.5.
- Simulated turbidity: 0.39 NTU, which is less than the safe turbidity threshold of 1.0 NTU.
- Simulated chlorine: 0.19 mg/L, which is less than the safe chlorine threshold of 1.0 mg/L.

CODE TO CHECK FOR WATER CONSUMPTION:

This code provides a more comprehensive example which includes water temperature and total dissolved solids (TDS) in addition to pH, turbidity, and chlorine. It checks each parameter against safe thresholds and provides an overall assessment of water quality.

```
import random

 # Simulated water quality monitoring function

def monitor_water_quality():

    # Simulate water quality parameters within reasonable ranges

    pH = round(random.uniform(6.5, 8.5), 2)

    turbidity = round(random.uniform(0.1, 5.0), 2)

    chlorine = round(random.uniform(0.1, 2.0), 2)

    temperature = round(random.uniform(5.0, 30.0), 2)  # Degrees Celsius

    tds = round(random.uniform(100, 800), 2)  # Total Dissolved Solids in ppm


    return pH, turbidity, chlorine, temperature, tds


# Function to assess water quality

def assess_water_quality(pH, turbidity, chlorine, temperature, tds):

    # Define safe thresholds for each parameter

    pH_safe_range = (6.5, 8.5)

    turbidity_safe_max = 1.0  # NTU
```

```
    chlorine_safe_max = 1.0  # mg/L
    temperature_safe_range = (10.0, 25.0)  # Degrees Celsius
    tds_safe_range = (100, 500)  # ppm

    # Check if each parameter is within safe ranges
    safe = (pH >= pH_safe_range[0] and pH <= pH_safe_range[1] and
            turbidity <= turbidity_safe_max and
            chlorine <= chlorine_safe_max and
            temperature >= temperature_safe_range[0] and temperature <=
temperature_safe_range[1] and
            tds >= tds_safe_range[0] and tds <= tds_safe_range[1])

    return safe

# Simulated water quality data (you can replace this with actual sensor data)
simulated_data = monitor_water_quality()

pH, turbidity, chlorine, temperature, tds = simulated_data
is_safe = assess_water_quality(pH, turbidity, chlorine, temperature, tds)

# Print the simulated data and assessment result
print(f"pH: {pH:.2f}, Turbidity: {turbidity:.2f} NTU, Chlorine: {chlorine:.2f} mg/L")
print(f"Temperature: {temperature:.2f} °C, TDS: {tds:.2f} ppm")

if is_safe:
    print("The water is safe to drink.")
else:
```

```
print("The water is not safe to drink.")
```

EXPLANATION:

- The code defines a 'monitor_water_quality' function that simulates water quality parameters, including pH, turbidity, chlorine, temperature, and total dissolved solids (TDS). These parameters are generated with random values within reasonable ranges.
- There's a 'assess_water_quality' function that checks if each of these parameters falls within safe thresholds. The safe thresholds are predefined for each parameter, including pH, turbidity, chlorine, temperature, and TDS.
- The simulated data is generated by calling 'monitor_water_quality'. The values for pH, turbidity, chlorine, temperature, and TDS are assigned to variables.
- The code then calls the 'assess_water_quality' function with these simulated parameters to check if they are within safe ranges.
- Based on the results of the assessment, the code prints out the simulated values of pH, turbidity, chlorine, temperature, and TDS, along with an assessment of whether the water is safe to drink or not.

OUTPUT:

pH: 7.24, Turbidity: 0.91 NTU, Chlorine: 0.78 mg/L
Temperature: 20.54 °C, TDS: 342.85 ppm
The water is safe to drink.

- Simulated pH: 7.24, which falls within the safe pH range of 6.5 to 8.5.
- Simulated turbidity: 0.91 NTU, which is less than the safe turbidity threshold of 1.0 NTU.
- Simulated chlorine: 0.78 mg/L, which is less than the safe chlorine threshold of 1.0 mg/L.
- Simulated temperature: 20.54 °C, which falls within the safe temperature range of 10.0 to 25.0 °C.
- Simulated TDS: 342.85 ppm, which is within the safe TDS range of 100 to 500 ppm.

Since all parameters are within their respective safe thresholds, the code concludes that "The water is safe to drink."

pH: 7.92, Turbidity: 4.41 NTU, Chlorine: 0.44 mg/L

Temperature: 22.38 °C, TDS: 312.43 ppm

The water is not safe to drink.

- Simulated pH: 7.92, which falls within the safe pH range of 6.5 to 8.5.

- Simulated turbidity: 4.41 NTU, which is more than the safe turbidity threshold of 1.0 NTU.
- Simulated chlorine: 0.44 mg/L, which is less than the safe chlorine threshold of 1.0 mg/L.
- Simulated temperature: 22.38 °C, which falls within the safe temperature range of 10.0 to 25.0 °C.
- Simulated TDS: 312.43 ppm, which is within the safe TDS range of 100 to 500 ppm.

Since the turbitidy value is greater than the safe thresholds the code concludes that "the water is not safe to drink."

PARAMETERS FOR DRINKING WATER:

Drinking water quality is typically assessed using a combination of physical, chemical, and microbiological parameters to ensure it is safe for human consumption. The specific parameters and their allowable ranges can vary by region and are regulated by local authorities and health organizations. Here are some of the key parameters commonly used to assess the quality of drinking water:

- pH: The pH of drinking water should typically fall within the range of 6.5 to 8.5, with 7 being considered neutral. This range ensures that the water is not too acidic or too alkaline.
- Turbidity: Turbidity is a measure of the cloudiness or haziness of water caused by suspended particles. Low turbidity indicates clear water, while high turbidity can be a sign of contamination. The specific allowable levels can vary but are typically less than 5 NTU (Nephelometric Turbidity Units).
- Total Dissolved Solids (TDS): TDS represents the concentration of inorganic and organic substances dissolved in water. Low TDS is generally preferred for drinking water, with typical levels below 500 mg/L.
- Microbiological Parameters: These include the presence of coliform bacteria, E. coli, and other microorganisms. Drinking water should be free of harmful bacteria, with zero total coliforms and E. coli present.
- Chlorine Residual: Chlorine is often used as a disinfectant in drinking water treatment. The residual chlorine level should be maintained to ensure continued disinfection while minimizing taste and odor issues.
- Total Chlorine: The total chlorine concentration, which includes free chlorine and combined chlorine, should be within recommended levels to ensure effective disinfection.
- Disinfection Byproducts (DBPs): DBPs are formed when disinfectants like chlorine react with organic matter in water. Regulations often specify maximum allowable levels for DBPs.

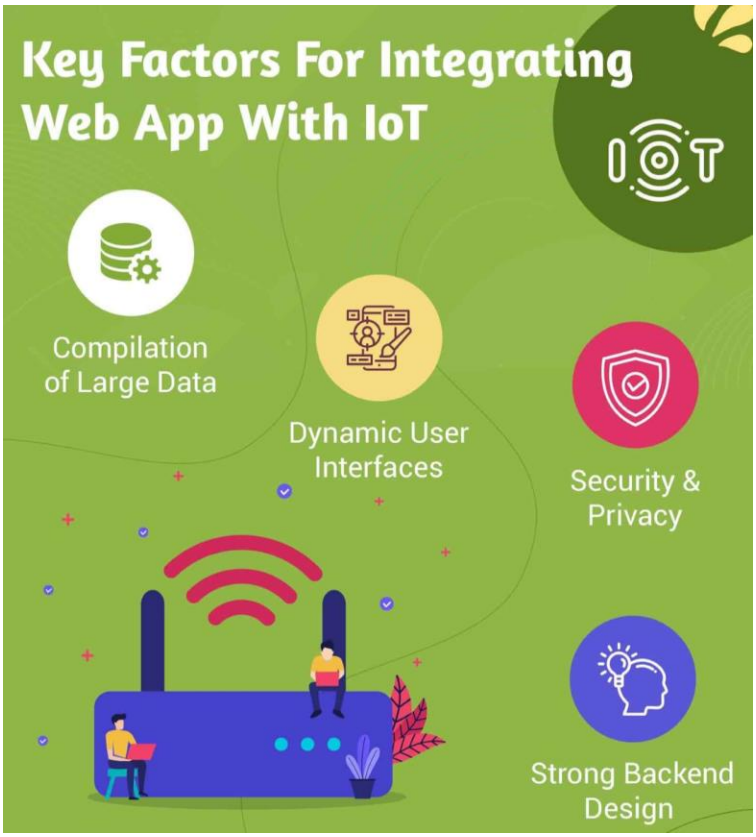
- Heavy Metals: Parameters such as lead, arsenic, mercury, and cadmium should be monitored, and their concentrations should be well below established safety limits.
- Nitrates and Nitrites: Elevated levels of nitrates and nitrites can be a sign of contamination, and their concentrations should be monitored, especially in groundwater sources.
- Fluoride: Fluoride is sometimes added to drinking water to prevent tooth decay. Levels should be controlled to avoid both deficiency and excess.
- Sulfates: High levels of sulfates can affect the taste of water and may have a laxative effect. Standards for sulfate levels are established in water quality regulations.
- Trace Organic Compounds: Monitoring for specific organic compounds like pesticides, herbicides, and volatile organic compounds (VOCs) is important for safety.
- Radionuclides: Radioactive elements like radon and radium should be monitored and kept within safety limits.

APPLICATIONS OF WEB DEVELOPMENT TECHNOLOGIES OVER IOT:

Web development technologies can be used to create user interfaces and manage the data generated by IoT devices. Here are some applications of web development technologies over IoT:

- Remote Monitoring and Control
- Data Visualization
- Healthcare and Telemedicine
- Asset Tracking
- Smart Agriculture
- Smart Cities
- Environmental Monitoring
- Energy Management
- Industrial IoT (IIoT)
- Fleet Management
- Wearable IoT
- Security and Surveillance
- Education and Training
- Consumer Electronics

In these applications, web development technologies provide the interface and data management tools necessary to access and control IoT devices and systems. They also enable data visualization, reporting, and analytics, making IoT data more actionable and valuable for users and businesses.



Web development technologies in the context of IoT,

1. Web-Based User Interface:

Example HTML:

```
<!DOCTYPE html>
<html>
<head>
   <title>IoT Dashboard</title>
   <link rel="stylesheet" type="text/css" href="styles.css">
</head><body>
   <h1>Welcome to IoT Dashboard</h1>
   <div id="sensorData"></div>
   <script src="script.js"></script>
</body>
</html>
```

2. Real-time Data Communication:

Example JavaScript using WebSockets:

```
const socket = new WebSocket('wss://your-iot-server.com');

socket.onmessage = function(event) {
    const data = JSON.parse(event.data);
    document.getElementById('sensorData').innerHTML = `Sensor Data: ${data.value}`;
};
```

3. Web Frameworks

4. Data Storage

5. Authentication and Security

6. Device Integration

7. Data Visualization

8. IoT Protocols

9. Cloud Services

10. Mobile Compatibility

11. Testing and Debugging

12. Version Control

13. Documentation

14. Open Source and Community Resources

15. Project Management

16. Learn as You Go