SMART WATER SYSTEM USING TINKERCAD

PHASE 5 : PROJECT DOCUMENTATION & SUBMISSION

Creating an smart water system project using Arduino and Tinkercad is a great way to simulate and prototype your system.

PROJECT OBJECTIVE :

The objective of a project for a smart water system using Tinkercad can be to create an innovative and efficient system that leverages IoT (Internet of Things) technology to monitor and manage water resources effectively. Here's a more detailed project objective:

1. **Remote Monitoring** :
   Develop a system that can remotely monitor water parameters such as water level, flow rate, and temperature in tanks or reservoirs. Utilize Tinkercad to create a virtual representation of the hardware and IoT components.

2. **Data Collection** :
   Enable the system to collect and store data from the sensors, creating a database that logs historical water quality and quantity information. This data can be accessed and analyzed for insights.

3. **Alerts and Notifications** :
   Implement a notification system that alerts users via email, SMS, or a mobile app when abnormal conditions are detected, such as low water levels, leaks, or water quality issues.

4. **Water Quality Control :**
   Develop a mechanism to control water quality parameters by integrating components like water purification systems or chemical dosing systems. Use Tinkercad to model and simulate these systems.

5. **Efficient Water Usage** :

Create features for optimizing water usage by automating tasks like irrigation or water distribution in smart homes or agriculture. Implement scheduling and automation rules.

6. **User-Friendly Interface** :
   Design a user-friendly web or mobile interface to access and control the smart water system, allowing users to visualize data, set parameters, and receive real-time updates.

7. **Energy Efficiency** :
   Ensure that the system is energy-efficient by implementing power-saving features and using energy-efficient components.

8. **Cost Efficiency :**
   Optimize the system for cost-effectiveness, considering both hardware and software components.

9. **Scalability** :
   Plan for scalability, allowing the system to be expanded or adapted for larger water management applications.

10. **Educational Component :**
    If this project is for educational purposes, provide resources and documentation that help users learn about IoT, sensor integration, and water management principles.

COMPONENTS REQUIRED :

1. Arduino board (e.g., Arduino Uno or Arduino Nano)
2. Water level sensor (e.g., ultrasonic sensor)
3. Wi-Fi module (e.g., ESP8266 or ESP32)
4. Relay module (if you want to control a pump)

5. Breadboard and jumper wires
6. Power supply (e.g., USB cable and adapter)
7. Tinkercad account (https://www.tinkercad.com/)

PROCEDURE :

1. Design the Circuit :

- Open Tinkercad and create a new project.
- Add components to your project from the Tinkercad library, such as Arduino, water level sensor, Wi-Fi module, and relay (if using a pump).
- Connect the components on the breadboard using jumper wires. Ensure the connections are correct and secure.

2. Write Arduino Code :

- Write the Arduino code that reads the water level from the sensor and sends the data to the cloud (you can use a platform like ThingSpeak, Adafruit IO, or Firebase).
- If you're using a relay to control a pump, also write the code to turn the pump on or off based on the water level.

3. Simulate the Circuit :

- In Tinkercad, you can simulate your circuit to ensure it's working as expected.
- Upload the Arduino code to the Arduino board in the simulation.
- Test the circuit by adjusting the water level sensor and monitoring the simulated output.

4. Connect to the Cloud :

- If you're using a Wi-Fi module, you need to configure it to connect to your Wi-Fi network.
- Modify your code to send data to a cloud platform of your choice. Most platforms provide libraries and APIs for easy integration.

5. Monitor and Control Remotely :

- Access the data on your chosen cloud platform. You can visualize the water level and set up alerts or notifications.
- If you're controlling a pump, you can remotely turn it on or off through the cloud platform's interface.

6. Test and Iterate :

- Test your smart water system in the simulation and ensure it works as intended.
- Make any necessary adjustments to the circuit or code.

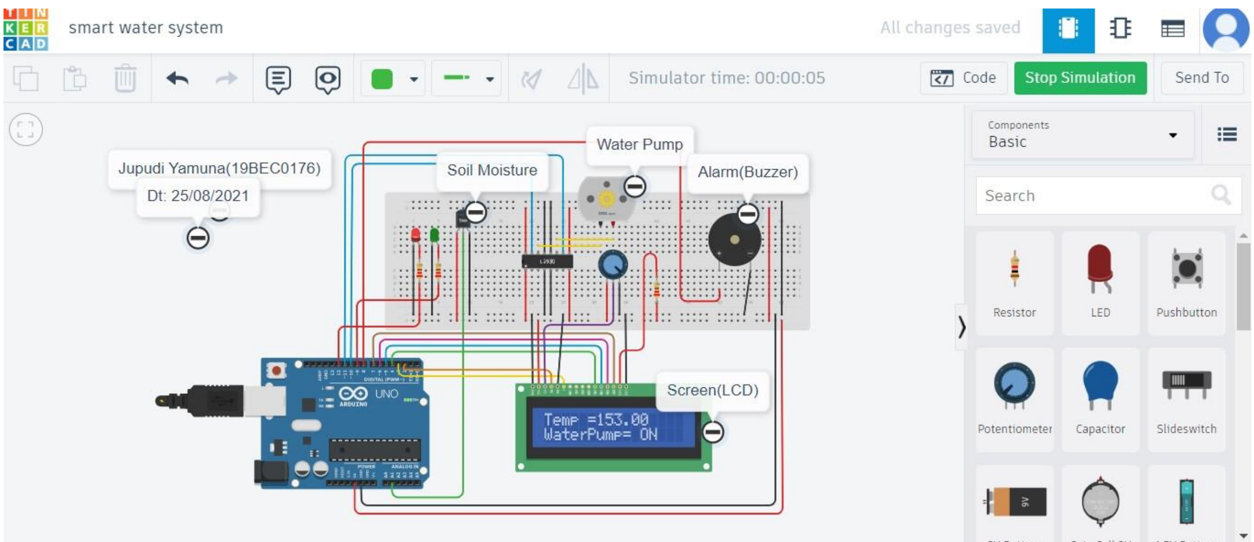
PROGRAM :

```
#include <LiquidCrystal.h>


const int temp = A1;
const int motor_terminal1 = 10;
const int motor_terminal2 = 11;
const int LedRed = 12;
const int LedGreen = 9;
const int Buzzer = 8;


LiquidCrystal lcd(2, 3, 4, 5, 6, 7);

void setup() {
  Serial.begin(9600);
  Serial.print("Smart water system");
  Serial.print("\n");
  Serial.print("\n");
  lcd.begin(16, 2);
  lcd.print("Smart water system");
  lcd.setCursor(4,1);
  lcd.print("System!!");
  pinMode(Buzzer, OUTPUT);
  pinMode(LedRed, OUTPUT);
```

```
  pinMode(LedGreen, OUTPUT);
  pinMode(motor_terminal1, OUTPUT);
  pinMode(motor_terminal2, OUTPUT);
  delay(2000);
  lcd.clear();
  lcd.print("Temp = ");
  lcd.setCursor(0,1);
  lcd.print("WaterPump= ");
}

void loop() {

  int value = analogRead(temp);
  float Temperature = value;
  Serial.print("Soil Temperature = ");
  Serial.print(Temperature);
  Serial.print("\n");Serial.print("\n");
  lcd.setCursor(6,0);
  lcd.print(Temperature);
  lcd.setCursor(11,1);


  if (Temperature > 50){
    digitalWrite(motor_terminal2, HIGH);
    digitalWrite(motor_terminal1, LOW);
    digitalWrite(LedRed, HIGH);
    digitalWrite(LedGreen, LOW);
    tone(Buzzer,220,100);
    lcd.print("ON ");
    Serial.print("Warning...!!!! Soil temperature is high");
    Serial.print("\n");Serial.print("\n");
    Serial.print("Need water!! Switch on water pump");
    Serial.print("\n");Serial.print("\n");
  }
  else {
    digitalWrite(motor_terminal2, LOW);
    noTone(Buzzer);
    digitalWrite(LedRed, LOW);
    digitalWrite(LedGreen, HIGH);
    lcd.print("OFF");
```

```
    Serial.print("Soil Temperature is fine...!!!");
    Serial.print("\n");Serial.print("\n");
  }

  delay(1000);
}
```

CIRCUIT DIAGRAM :



CONCLUSION :

This project in Tinkercad provides a great starting point for understanding the basics of smart water system using Arduino. Remember that for a real-world implementation, you would need to adapt the project to physical components and consider factors like power supply, data transmission, and enclosure design.