Smart WATER FOUNTAIN

PHASE 2 IOT_PHASE2

INNOVATION

Innovation in smart water fountains has the potential to improve water quality, accessibility, and sustainability, while also enhancing user experience. Here are some innovative features and technologies that can be incorporated into smart water fountains:

Water Quality Monitoring:

Integration of sensors to continuously monitor water quality, detecting contaminants, and pathogens. If an issue is detected, the fountain can shut off and send an alert for maintenance.

Filtration Systems:

Advanced filtration systems, such as UV, reverse osmosis, or activated carbon filters, can be used to ensure clean and safe drinking water.

IoT Connectivity:

Smart water fountains can be connected to the Internet of Things (IoT), allowing for remote monitoring and control. This enables real-time data collection on usage, water quality, and maintenance needs.

User Authentication:

Implementing user authentication through RFID cards, smartphone apps, or biometrics can provide a personalized experience, ensuring that only authorized users can access the fountain.

Hydration Tracking:

The fountain can track the volume of water consumed by users, encouraging healthy hydration habits. This data can be integrated with health apps for a holistic approach to well-being.

Touchless Operation:

In a post-COVID-19 world, touch less operation with motion sensors or foot pedals can minimize the risk of contamination.

Bottle Refill Stations:

Smart fountains can be designed with dedicated refill stations for reusable water bottles, reducing single-use plastic waste.

Solar-Powered Operation:

Integrating solar panels to power the fountain can make it eco-friendly and sustainable.

Customizable User Experience:

Users can choose the water temperature (hot or cold), carbonation levels, and even add flavours like fruit-infused water.

Data Analytics:

Smart fountains can collect and analyse data to optimize water usage, inform maintenance schedules, and improve the overall efficiency of water distribution systems.

Water Conservation Features:

Incorporating features like automatic shutoff after a certain amount of water has been dispensed, reducing water wastage.

Interactive Displays:

Adding touchscreens for educational or informative purposes, showcasing data on water conservation, quality, and local water sources.

Voice Activation:

Voice-controlled smart water fountains can enhance accessibility for people with disabilities.

Emergency Alerts:

Integrating the fountain with emergency alert systems, so it can dispense water in case of natural disasters or emergencies.

Artificial Intelligence:

Using AI algorithms to predict usage patterns, detect anomalies, and optimize maintenance schedules, reducing downtime and costs.

Water Recycling:

Advanced water fountains can include systems that treat wastewater for reuse, promoting water sustainability.

Aesthetic Design:

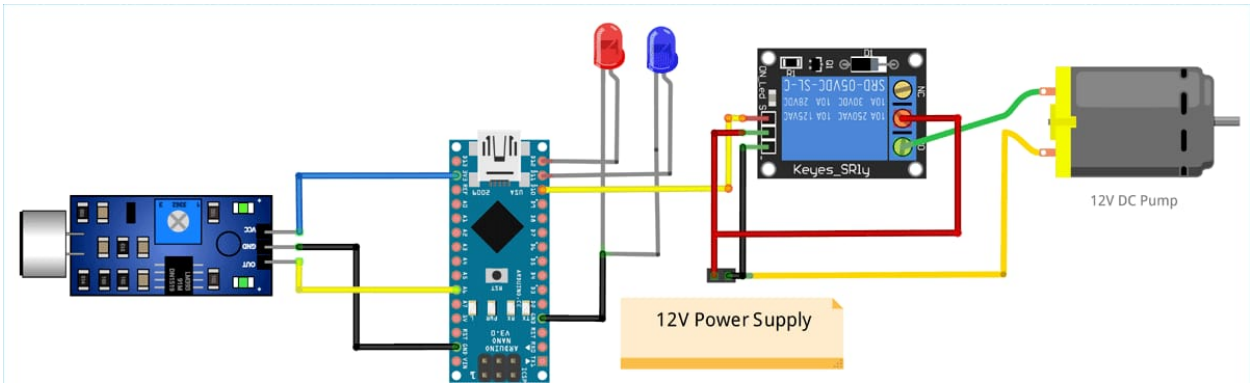
An innovative approach to design can make smart fountains more appealing and encourage their use in public spaces.

Localization and Multilingual Support:

Smart fountains in diverse areas can offer multilingual interfaces and consider local water needs and cultural preferences.

PROJECT CONCLUSION:

Innovation in smart water fountains can significantly impact water quality, accessibility, and sustainability while improving the user experience. By combining advanced technology with sustainable practices, smart water fountains have the potential to play a crucial role in the future of public water access.



PROGRAM

```
const int red = 9;
const int green = 10;
const int blue = 11;
 int val=0;
void setup()
{
Serial.begin(9600);
   pinMode(red, OUTPUT);
  pinMode(green, OUTPUT);
  pinMode(blue, OUTPUT);
}
void loop(){
val=Serial.read();
if (val=='a'){
ran();
}
if (val=='b'){
    analogWrite(red,0);
     analogWrite(green,222);
      analogWrite(blue,100);
```

```
}
if (val=='c'){
  analogWrite(red,222);
     analogWrite(green,0);
      analogWrite(blue,100);
}
if (val=='d'){
   analogWrite(red,222);
     analogWrite(green,165);
      analogWrite(blue,0);
}
}
void ran(){
  while (1){
     analogWrite(red,0);
     analogWrite(green,222);
      analogWrite(blue,100);
      delay(60);
      analogWrite(red,222);
     analogWrite(green,0);
      analogWrite(blue,100);
       delay(100);
        analogWrite(red,222);
     analogWrite(green,165);
      analogWrite(blue,0);
```

```
    delay(50);
analogWrite(red,0);
   analogWrite(green,222);
    analogWrite(blue,0);
    delay(60);
    analogWrite(red,0);
   analogWrite(green,190);
    analogWrite(blue,100);
     delay(100);
      analogWrite(red,222);
   analogWrite(green,0);
    analogWrite(blue,160);
     delay(50);
          analogWrite(red,0);
   analogWrite(green,222);
    analogWrite(blue,100);
    delay(60);
    analogWrite(red,222);
   analogWrite(green,55);
    analogWrite(blue,120);
     delay(100);
      analogWrite(red,222);
   analogWrite(green,165);
    analogWrite(blue,0);
     delay(50);
```

```
  analogWrite(red,0);
     analogWrite(green,222);
      analogWrite(blue,0);
      delay(60);
      analogWrite(red,222);
     analogWrite(green,19);
      analogWrite(blue,10);
       delay(100);
        analogWrite(red,222);
     analogWrite(green,186);
      analogWrite(blue,195);

 delay(100);

   }
}
```