Smart Water Management

Objective:

This proposal outlines a comprehensive solution for monitoring and conserving water in public places using advanced technology. The key components of the system include water flow sensors, microcontrollers (e.g., Arduino or Raspberry Pi), ESP32 or Wi-Fi modules for wireless communication, and a user-friendly application accessible.

Monitoring water consumption and leakage detection in public places like parks

1.Climate-Based Watering:The application collects weather data to adjust plant watering schedules based on humidity and temperature, minimizing water usage during rainy or high moisture conditions.

2.Leak Detection:Water flow sensors in pipes continuously monitor water flow rates, sending alerts to the application in the event of abnormal flow. This allows for rapid leak detection and prevention of water wastage.

3.User-Friendly Application:The application provides real-time water monitoring, climate-based scheduling, leak alerts, historical data viewing, and remote water flow control, catering to both administrators and the general public.

The benefits of implementing this system in public spaces include water conservation, cost savings, reduced environmental impact, convenience, and early leak detection. The aim is to promote sustainable and environmentally conscious water management practices.

Advancing Water monitoring in Public Spaces:

The project aims to enhance the water monitoring and conservation system introduced in Phase 1 by implementing innovative features and technologies. These innovations include:

1. Enhanced Sensor Technology: The integration of more accurate water flow sensors for better control and optimization of water usage.

2. Data Analytics: The use of advanced data analytics tools to analyze historical data and optimize water conservation efforts.

3. Remote Control and Automation: Real-time adjustment of watering schedules based on weather forecasts and user customization through a user-friendly application.

4. IoT Integration: Improved communication between sensors and the control unit for faster response times and better data synchronization.

These innovations aim to make the system more efficient and environmentally sustainable, focusing on energy efficiency, reduced water wastage, and carbon footprint reduction. The project also contributes to ecosystem protection and biodiversity preservation by conserving water resources.

The user-friendly interface ensures ease of use, and the Phase 2 timeline includes stages such as sensor integration, AI development, data analytics implementation, IoT integration, user testing, and refinement of the user interface.

In conclusion, it is dedicated to leveraging technology for water conservation in public places, prioritizing environmental concerns, ease of use, and the adoption of the best available technology. The ultimate goal is to create a system that saves water and promotes eco-friendly water usage.

Explain how the real-time water consumption monitoring system can promote water conservation and sustainable practices.

A real-time water consumption monitoring system can significantly contribute to promoting water conservation and sustainable practices by providing crucial data and insights that enable effective management of water resources. Here are some ways in which such a system can bring about positive changes:

1. Awareness and Behavior Change:Real-time monitoring systems can provide users with instant feedback on their water usage, making them more aware of their consumption patterns. This awareness often leads to more mindful water usage and encourages individuals to adopt water-saving practices.

2.Leak Detection and Prevention:These systems can detect leaks in the water supply network promptly. By identifying and addressing leaks in real-time, water wastage can

be minimized, leading to significant conservation of water resources and reduced operational costs for water utilities.

3. Optimized Water Distribution:By monitoring water consumption patterns in real-time, authorities can better understand the demand for water in different areas. This allows for the optimization of water distribution, ensuring that water is distributed efficiently to meet demand without unnecessary waste.

4.Data-Driven Decision Making:Real-time monitoring provides a wealth of data that can be analyzed to identify trends and patterns in water consumption. This data can inform policy decisions and urban planning, enabling authorities to implement targeted water conservation strategies that are based on actual usage data.

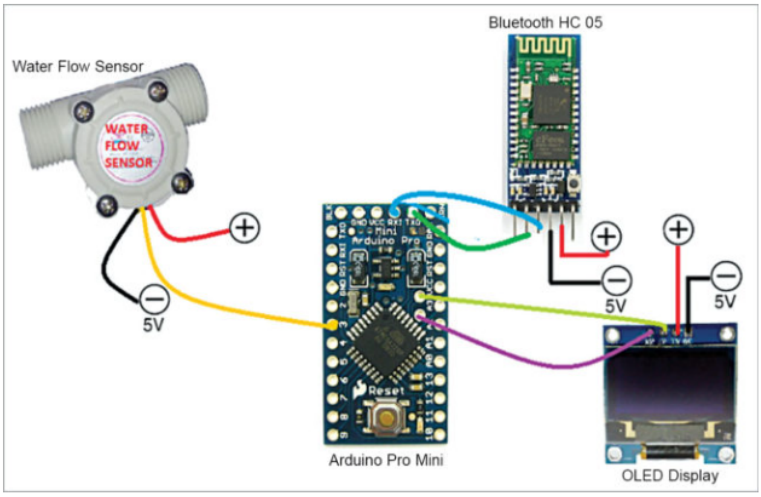
5. Behavioral Feedback and Incentives: Some real-time monitoring systems incorporate features that provide users with feedback on their water consumption habits. This can include suggestions for reducing consumption or incentives for meeting specific conservation targets, encouraging users to actively participate in sustainable water practices.

6. Educational Initiatives:The data generated by these systems can be used to create educational campaigns and initiatives aimed at promoting water conservation and sustainable practices. By leveraging the insights gained from the monitoring system, educational programs can be tailored to address specific water usage challenges within a community.

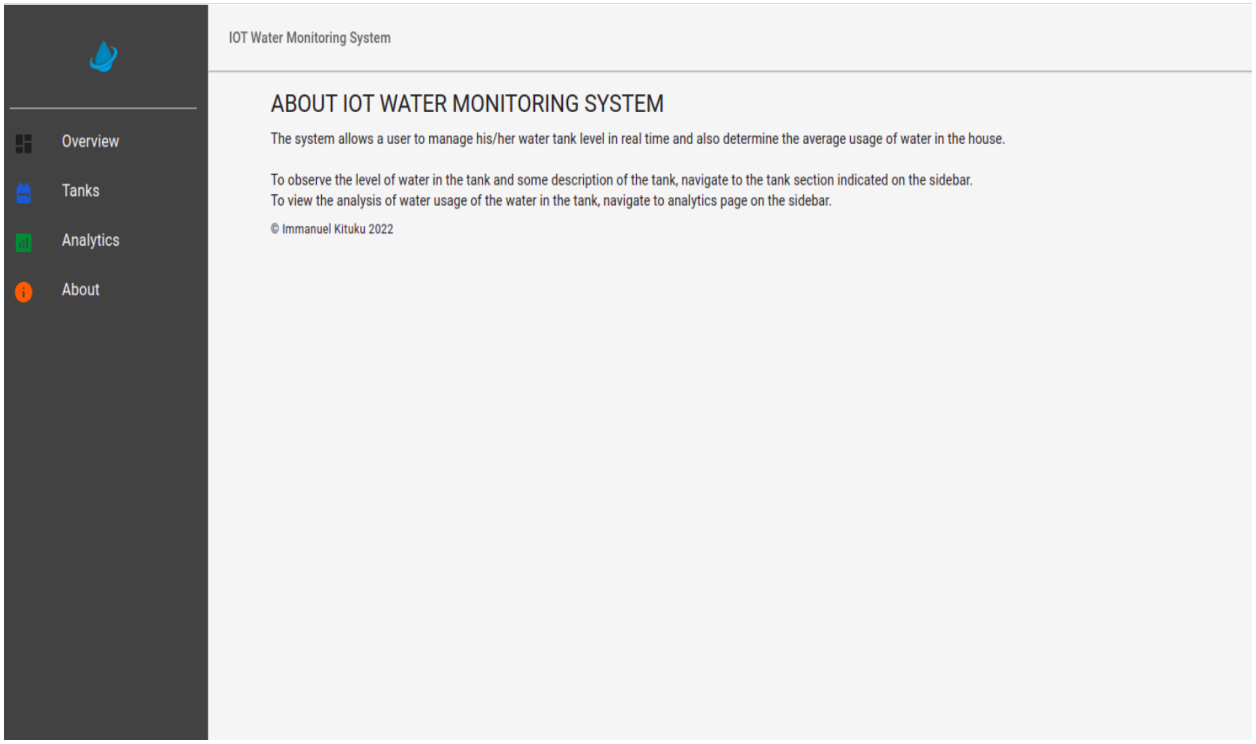
7.Remote Management and Control:Real-time monitoring systems often come with remote management capabilities, allowing authorities to control and manage water distribution systems from a centralized location. This enables quick responses to any anomalies or emergencies, leading to efficient and prompt water management.

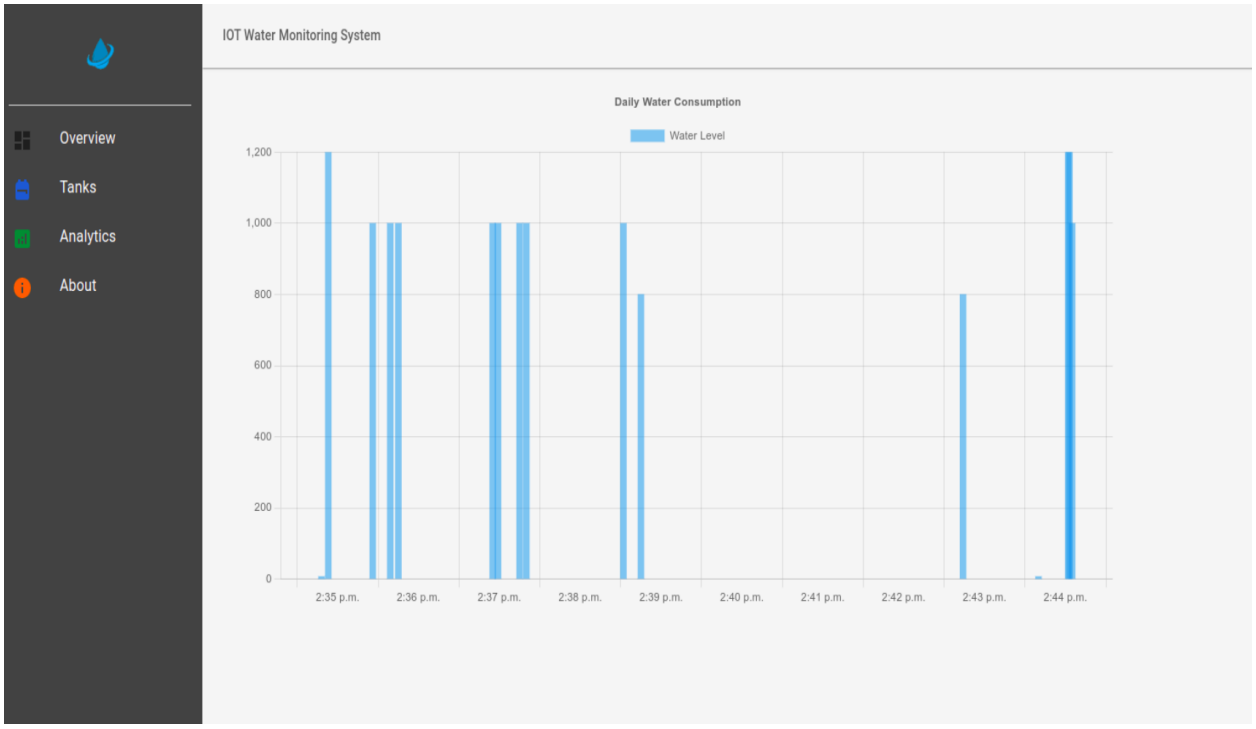
By leveraging these capabilities, a real-time water consumption monitoring system can not only promote water conservation but also contribute to the overall sustainability of water resources, leading to long-term environmental benefits and improved water management practices.

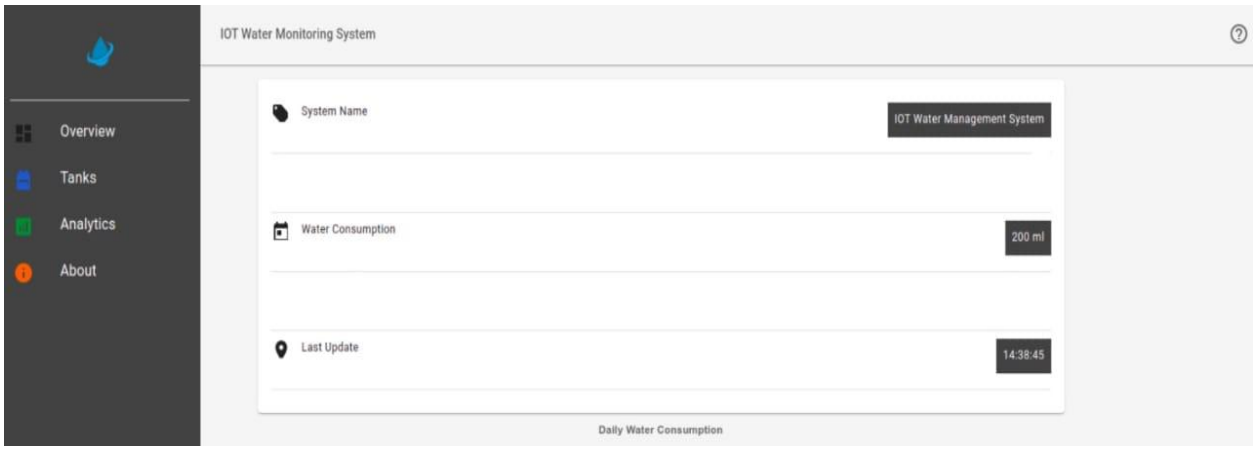
Connectivity of sensor



Example Output of the Application




IOT Water Monitoring System
Overview
Tanks
Analytics
About
Daily Water Consumption
Water Level
1,200
1,000
800
600
400
200
0
2:35 p.m.
2:36 p.m.
2:37 p.m.
2:38 p.m.
2:39 p.m.
2:40 p.m.
2:41 p.m.
2:42 p.m.
2:43 p.m.
2:44 p.m.


IOT Water Monitoring System
Overview
Tanks
Analytics
About
System Name
IOT Water Management System
Water Consumption
200 ml
Last Update
14:38:45
Daily Water Consumption

Steps to replicate and deploy the project

Components Required:

Arduino Board: You can use an Arduino Uno, Arduino Nano, or another compatible board.
Water Flow Sensor: To measure water flow/consumption.
Water Leakage Sensor: A sensor that can detect water leakage (e.g., a moisture sensor).
Breadboard and Jumper Wires: For prototyping and connecting components.
LEDs/Buzzer: To indicate leakage or certain consumption levels.
WiFi/Bluetooth Module (optional): If you want to send data or alerts wirelessly.

Steps:

Setting up the Water Flow Sensor:

- Connect the water flow sensor to the Arduino following its datasheet or pinout.
- Calibrate the sensor if required to accurately measure water flow rates.

Connecting the Water Leakage Sensor:

- Connect the water leakage sensor to the Arduino. This could involve placing the sensor in an area prone to leaks or around pipes where water might accumulate in case of a leak.

Writing the Arduino Code:

- Write code that reads data from the water flow sensor at regular intervals. The sensor will provide data that you can convert to actual water consumption.
- Write code to read data from the water leakage sensor. This sensor might provide a digital signal indicating the presence of water.
- Utilize conditional statements to check for certain thresholds of water flow that might indicate issues or leaks. This could trigger an alert (LED lighting up, buzzer sounding, etc.).

Alert Mechanism:

- Implement an alert system to notify users if a leak is detected or if water consumption surpasses a set threshold.
- You can integrate LEDs or buzzers for local alerts, or incorporate Wi-Fi/Bluetooth modules to send notifications to a mobile device or a server.

Testing and Integration:

- Assemble all the components and test the system by simulating water flow and leaks.
- Ensure the system accurately detects flow and leaks and triggers the appropriate alerts.

Python script on the IoT sensors to send real-time water consumption data to the data-sharing platform.

```
import serial
import time

SERIAL_PORT = COM3
BAUD_RATE = 9600

# Thresholds for excess consumption and low moisture
WATER_CONSUMPTION_THRESHOLD = 150
SOIL_MOISTURE_THRESHOLD = 30

# Function to send an alert message
def send_alert(message):
    print(f'ALERT: {message}')

# Initialize the serial connection to the Arduino
ser = serial.Serial(SERIAL_PORT, BAUD_RATE)

try:
    while True:
        data = ser.readline().decode().strip()
        try:
            category, value = data.split(":")
            value = float(value)

            if category == "SoilMoisture":
                print(f'Soil Moisture Level: {value}%')
                if value < SOIL_MOISTURE_THRESHOLD:
                    alert_message = 'Low soil moisture level detected. Water your
garden.'
                    send_alert(alert_message)

            elif category == "Humidity":
```

```
            print(f'Humidity: {value}%')

        elif category == "WaterConsumed":
            print(f'Water Consumed: {value} liters')
            if value > WATER_CONSUMPTION_THRESHOLD:
                alert_message = f'Excess water consumption detected: {value}
liters.'
                send_alert(alert_message)

    except ValueError:
        print(f'Invalid data received: {data}')

except KeyboardInterrupt:
    ser.close()
```

Code for the Application:

Index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <link rel="icon" type="image/png" href="./logo.png" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <meta name="theme-color" content="#000000" />
    <meta
      name="description"
      content="This is an IOT water monitoring system that shows you level
of water in tank and gives you analysis of water usage"
    />
    <link rel="apple-touch-icon" href="logo.png" />
    <link rel="manifest" href="./manifest.json" />
    <title>IOT Water Monitoring System</title>
  </head>
```

```
  <body>
    <div id="root"></div>
    <script>
      if ("serviceWorker" in navigator) {
        window.addEventListener("load", () => {
          navigator.serviceWorker
            .register("./serviceWorker.js")
            .then((reg) => console.log("Success:", reg.scope))
            .catch((err) => console.log("Failure:", err));
        });
      }
    </script>
  </body>
</html>
```

Index.js

```
import React from "react";
import ReactDOM from "react-dom";
import "./Assets/styles.js";
import App from "./App";

ReactDOM.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>,
  document.getElementById("root")
);
```

App.js

```
import { CssBaseline } from "@mui/material";
import Layout from "./Components/Layout";
import Overview from "./Components/Overview page/Overview";
import TankDetails from "./Components/Tanks/TankDetails";
import Analytics from "./Components/Analytics";
```

```
import About from "./Components/About";
import { BrowserRouter as Router, Routes, Route } from "react-router-dom";

document.body.style.background = "#f5f5f5";

function App() {
  return (
    <div>
      <Router>
        <CssBaseline />
        <Layout>
          <Routes>
            <Route path="/" element={<Overview />} />
            <Route path="/tanks" element={<TankDetails />} />
            <Route path="/analytics" element={<Analytics />} />
            <Route path="/about" element={<About />} />
          </Routes>
        </Layout>
      </Router>
    </div>
  );
}
export default App;
```

About.js

```
import React from "react";
import { Typography, Container, Box } from "@mui/material";

const About = () => {
  return (
    <div>
      <Box>
```

```
        <Container>
          <Typography
            variant="h5"
            component="h1"
            sx={{ textTransform: "uppercase" }}
            gutterBottom
          >
            About IOT Water Monitoring System
          </Typography>
          <Typography variant="body2" component="p" gutterBottom>
            The system allows a user to manage his/her water tank level in real
            time and also determine the average usage of water in the house.
            <br />
            <br />
            To observe the level of water in the tank and some description of
            the tank, navigate to the tank section indicated on the sidebar.
            <br />
            To view the analysis of water usage of the water in the tank,
            navigate to analytics page on the sidebar.
          </Typography>
          <Typography variant="caption">&copy; Immanuel Kituku
2022</Typography>
        </Container>
      </Box>
    </div>
  );
};
export default About;
```

Analytics.js

```
import React from "react";
import LineChart from "./Charts/LineChart";

const Analytics = () => {
```

```
  return (
    <div>
      <LineChart />
    </div>
  );
};
export default Analytics;
```

Layout.js

```
import React from "react";
import Navbar from "./Navbar";
import Sidebar from "./Sidebar";
import { NavProvider } from "./NavContext";
import { useStyles } from "../Assets/styles";

const Layout = ({ children }) => {
  const classes = useStyles();
  return (
    <div>
      <NavProvider>
        <Sidebar />
        <Navbar />
      </NavProvider>
      <div component="main" className={classes.container}>
        {children}
      </div>
    </div>
  );
};
export default Layout;
```

NavContext.js

```
import React, { createContext, useState } from "react";
export const navContext = createContext();
```

```
export const NavProvider = ({ children }) => {
  const [mobileOpen, setMobileOpen] = useState(false);
  const handleDrawerToggle = () => {
    setMobileOpen(!mobileOpen);
  };
  return (
    <div>
      <navContext.Provider value={{ mobileOpen, handleDrawerToggle }}>
        {children}
      </navContext.Provider>
    </div>
  );
};

export default NavProvider;
```

Navbar.js

```
import { useContext } from "react";
import { Link } from "react-router-dom";
import { AppBar, Toolbar, Typography, Button, IconButton } from
"@mui/material";
import MenuIcon from "@mui/icons-material/Menu";
import HelpOutlineIcon from "@mui/icons-material/HelpOutline";
import { drawerWidth } from "./Sidebar";
import { navContext } from "./NavContext";

const Navbar = () => {
  const { handleDrawerToggle } = useContext(navContext);
  return (
    <nav>
      <AppBar
        position="fixed"
        sx={{
          width: { sm: `calc(100% - ${drawerWidth}px)` },
          ml: { sm: `${drawerWidth}px` },
```

```
    background: "#f5f5f5",
    color: "#616161",
  }}
  elevation={1}
>
  <Toolbar sx={{ display: "flex", justifyContent: "space-between" }}>
    <IconButton
      color="inherit"
      aria-label="open drawer"
      edge="start"
      onClick={handleDrawerToggle}
      sx={{ mr: 2, display: { sm: "none" } }}
    >
      <MenuIcon sx={{ flexGrow: 1 }} />
    </IconButton>
    <Typography
      variant="h6"
      sx={{ flexGrow: 1, textTransform: "capitalize", fontSize: "14px" }}
    >
      IOT Water Monitoring System
    </Typography>
    <HelpOutlineIcon />

    <Link
      to="/about"
      style={{
        textDecoration: "none",
        color: "black",
      }}
    >
      <Button
        color="inherit"
        variant="text"
        sx={{ textTransform: "capitalize" }}
      >
```

```
            Help
          </Button>
        </Link>
      </Toolbar>
    </AppBar>
  </nav>
 );
};
export default Navbar;
```

Sidebar.js

```
import { useContext } from "react";
import {
 Toolbar,
 List,
 ListItem,
 ListItemIcon,
 ListItemText,
 Drawer,
 Divider,
 Icon,
 Box,
} from "@mui/material";
import DashboardIcon from "@mui/icons-material/Dashboard";
import BackpackIcon from "@mui/icons-material/Backpack";
import AnalyticsIcon from "@mui/icons-material/Analytics";
import InfoIcon from "@mui/icons-material/Info";
import logo from "../Assets/images/logo.svg";
import { Link } from "react-router-dom";
import { useStyles } from "../Assets/styles";
```

```
import { navContext } from "./NavContext";

export const drawerWidth = 240;
const Sidebar = () => {
  const classes = useStyles();
  const { mobileOpen, handleDrawerToggle } = useContext(navContext);
  const drawerMenu = [
    {
      label: "Overview",
      icon: <DashboardIcon />,
      color: "white",
      path: "/",
    },
    {
      label: "Tanks",
      icon: <BackpackIcon />,
      color: "primary",
      path: "/tanks",
    },
    {
      label: "Analytics",
      icon: <AnalyticsIcon />,
      color: "success",
      path: "/analytics",
    },
    { label: "About", icon: <InfoIcon />, color: "warning", path: "/about" },
  ];
  const drawer = (
    <div>
      <Toolbar>
        <img
          src={logo}
          alt="logo"
          width="100"
          height="100"
```

```
        className={classes.logo}
      />
    </Toolbar>
    <Box className={classes.divider}>
      <Divider />
    </Box>
    <List>
      {drawerMenu.map(({ label, icon, color, path }, index) => (
        <Link
          to={path}
          style={{
            textDecoration: "none",
          }}
          key={index}
        >
          <ListItem
            button
            key={label}
            sx={{ "&:hover": { background: "#f5f5f5" } }}
          >
            <ListItemIcon>
              <Icon color={color}>{icon}</Icon>
            </ListItemIcon>
            <ListItemText
              primary={label}
              sx={{ "&:hover": { color: "#616161" }, color: "#fcfcfc" }}
            />
          </ListItem>
        </Link>
      ))}
    </List>
  </div>
);
return (
  <div>
```

```
<Box
  component="nav"
  sx={{ width: { sm: drawerWidth }, flexShrink: { sm: 0 } }}
>
  <Drawer
    variant="temporary"
    open={mobileOpen}
    onClose={handleDrawerToggle}
    ModalProps={{
      keepMounted: true,
    }}
    sx={{
      display: { xs: "block", sm: "none" },
      "& .MuiDrawer-paper": {
        boxSizing: "border-box",
        width: drawerWidth,
        background: "#424242",
      },
    }}
  >
    {drawer}
  </Drawer>
  <Drawer
    variant="permant"
    sx={{
      display: { xs: "none", sm: "block" },
      "& .MuiDrawer-paper": {
        boxSizing: "border-box",
        width: drawerWidth,
        background: "#424242",
      },
    }}
    open
  >
    {drawer}
```

```
        </Drawer>
      </Box>
    </div>
  );
};

export default Sidebar;
```

LineChart.js

```
import { useState, useEffect, useRef, useMemo } from "react";
import {
  Chart as ChartJS,
  ArcElement,
  LineElement,
  BarElement,
  PointElement,
  BarController,
  BubbleController,
  DoughnutController,
  LineController,
  PieController,
  PolarAreaController,
  RadarController,
  ScatterController,
  CategoryScale,
  LinearScale,
  LogarithmicScale,
  RadialLinearScale,
  TimeScale,
  TimeSeriesScale,
  Decimation,
  Filler,
  Legend,
```

```
  Title,
  Tooltip,
  SubTitle,
} from "chart.js";
import { Chart } from "react-chartjs-2";
import "chartjs-adapter-date-fns";
import { db, ref, onValue } from "../../firebase-config";

ChartJS.register(
  ArcElement,
  LineElement,
  BarElement,
  PointElement,
  BarController,
  BubbleController,
  DoughnutController,
  LineController,
  PieController,
  PolarAreaController,
  RadarController,
  ScatterController,
  CategoryScale,
  LinearScale,
  LogarithmicScale,
  RadialLinearScale,
  TimeScale,
  TimeSeriesScale,
  Decimation,
  Filler,
  Legend,
  Title,
  Tooltip,
  SubTitle
);
```

```
const LineChart = () => {
  const [data, setData] = useState({ labels: [], datasets: [] });
  const [chartData, setChartData] = useState({});

  useEffect(() => {
    let dataArr = [];
    onValue(ref(db, "graph"), (snapshot) => {
      snapshot.forEach((childSnapshot) => {
        let xVal = snapshot.val()[childSnapshot.key].timestamp;
        let yVal = snapshot.val()[childSnapshot.key].level;
        dataArr.push({ x: xVal, y: yVal });
      });
      setChartData(dataArr);
    });
  }, []);

  const myChartRef = useRef();

  const config = useMemo(
    () => ({
      datasets: [
        {
          label: "Water Level",
          barThickness: 7,
          maxBarThickness: 10,
          minBarLength: 5,
          data: chartData,
          borderColor: "rgb(53, 162, 235)",
          backgroundColor: "rgba(53, 162, 235, 0.5)",
        },
      ],
    }),
    [chartData]
  );
```

```
  useEffect(() => {
    setData(config);
  }, [config]);

  const options = {
    responsive: true,
    plugins: {
      title: {
        display: true,
        text: "Daily Water Consumption",
        fontSize: 25,
      },
    },
    scales: {
      x: {
        type: "time",
        time: {
          unit: "minute",
        },
      },
      y: {
        beginAtZero: true,
      },
    },
  };

  return (
    <>
      <Chart type="bar" data={data} options={options} ref={myChartRef} />
    </>
  );
};
export default LineChart;
```

Style.js

```
import { makeStyles } from "@mui/styles";
export const useStyles = makeStyles({
  logo: {
    margin: "0 auto",
  },
  container: {



    margin: "5% 18%",
    "@media(max-width: 800px)": {
      margin: "15% auto",
      width: "100%",
    },
  },
  divider: {
    background: "#f5f5f5",
    width: "90%",
    margin: "0 auto",
  },
});
```