SMART HOME AUTOMATION SYSTEM USING ESP32 MICROCONTROLLER

**System Features**

1. **Real-Time Control**: Uses RTC for scheduling and time-based automation.
2. **Remote Access**: Controls devices using a remote or a mobile application.
3. **Manual Override**: Pushbuttons provide manual control.
4. **Status Display**: LCD displays real-time information and status of devices.
5. **Indicators**: LEDs signal the state of the devices.
6. **Relay Modules**: Act as switches to control high-power appliances.
7. **Wi-Fi Connectivity**: ESP32 provides wireless control and IoT functionality.

**Components**

1. **ESP32 Microcontroller**: Central unit for processing and communication.
2. **RTC Module (e.g., DS3231)**: Maintains accurate time for scheduling.
3. **Relay Modules**: Control AC appliances like lights and fans.
4. **Pushbuttons**: Manual on/off control for devices.
5. **LCD Display (16x2 or 20x4)**: Displays the status of devices and time.
6. **LEDs**: Indicate device states (e.g., ON/OFF).
7. **Remote (IR)**: Allows remote control using an IR receiver module.
8. **Breadboards & Jumper Wires**: For circuit connections.
9. **Power Supply**: Powers the ESP32 and connected components.

**Circuit Design**

1. **ESP32**:
   * Connected to the RTC via I2C (SDA, SCL).
   * GPIO pins connected to relay modules, LEDs, pushbuttons, and IR receiver.
2. **Relay Modules**:
   * Controlled by GPIO pins of ESP32 to switch appliances.
   * Connect the relay's NO (Normally Open) pin to the appliance.
3. **LCD**:
   * Use I2C module to simplify connections (SDA, SCL to ESP32).
4. **RTC Module**:
   * SDA and SCL to ESP32 for time synchronization.
5. **Pushbuttons**:
   * Connected to GPIO pins with pull-up or pull-down resistors.
6. **IR Receiver**:
   * Connect the data pin to ESP32 for remote control input.

**Software Development**

Use **Arduino IDE** to program the ESP32.

**Libraries Required**

1. WiFi.h for Wi-Fi connectivity.
2. Wire.h for I2C communication.
3. LiquidCrystal\_I2C.h for LCD.
4. RTClib.h for RTC module.
5. IRremote.h for IR receiver.

**Code Structure**

1. **Setup**:
   * Initialize peripherals: RTC, LCD, Wi-Fi, etc.
   * Configure GPIO pins.
2. **Loop**:
   * Check RTC for scheduled actions.
   * Read pushbutton states and IR remote commands.
   * Update LCD with device status.
   * Control relay modules based on inputs.
3. **Wi-Fi Integration**:
   * Set up a web server or MQTT for remote control via smartphone or PC.

**Enhancements**

1. **Wi-Fi Control**:
   * Add a web server or MQTT for IoT functionality.
2. **Mobile App**:
   * Use Blynk or a custom app to control devices.
3. **Voice Commands**:
   * Integrate with Google Assistant or Alexa using IFTTT and MQTT.

Project Link:- https://wokwi.com/projects/417727218276875265