Class 3 (8th - 11th Graders)

Session 1: Introduction to Arduino & Basic Components

Duration: 90 minutes

Setting up ide and drivers.

Ice breaker activity, introduction to teachers

- **Introduction to Arduino:**
    - What is Arduino and what can it do?
    - Basic Arduino IDE setup and usage
- **Introduction to C++ Programming for Arduino:**
    - Brief introduction to C++ syntax (variables, functions, loops)
    - How C++ is used in Arduino programming
- **Basic Electronics Overview:**
    - Breadboard, Resistors, Capacitors, Jumper Wires
    - LEDs, RGB LEDs, 7-segment Display
- **Hands-On:**
    - Blink an LED using basic C++ commands
    - Simple digital/analog input/output (Button/LED circuit)

Session 2: Working with Motors & Basic Control

Duration: 90 minutes

- **Introduction to Motors:**
  - DC motors, Servo motors
  - L9110 Motor Driver
- **C++ for Motor Control:**
  - Using C++ for PWM control of motors
  - Controlling speed and direction of motors
- **Hands-On:**
  - Build a simple motorized project (e.g., DC motor controlled by a potentiometer)
  - Use a servo to control position

Session 3: Sensor Integration & Displays

Duration: 90 minutes

- **Working with Sensors:**
  - DHT11 Humiture Sensor (Temperature and Humidity)
  - Potentiometer, Button, Reed Switch
- **Working with Displays:**
  - LCD1602 and 7-segment display
- **C++ Programming for Sensors:**
  - Reading sensor data and using conditional statements
  - Sending sensor data to displays using C++ functions
- **Hands-On:**
  - Display temperature on LCD1602 using C++

- Use a potentiometer and button to control a display (e.g., display temperature or input)

Session 4: Smart Car Project - Basic Assembly

Duration: 90 minutes

- **Introduction to the Smart Car:**
  - Motor control for wheels (DC motors)
  - Basic assembly of the car (Chassis, Motors, Wheels)
- **C++ for Car Movement:**
  - Programming the car to move forward, backward, and turn using C++
- **Hands-On:**
  - Assemble a basic smart car and test it with basic movement code

---

Session 5: Smart Car - Obstacle Avoidance & Line Tracking

Duration: 90 minutes

- **Obstacle Avoidance Module:**
  - Use of ultrasonic sensor for obstacle detection
- **Line Tracking Module:**
  - Implement basic line tracking for autonomous movement
- **C++ for Autonomous Movement:**
  - Using C++ to process sensor data for obstacle avoidance and line tracking
- **Hands-On:**

- Modify the smart car to avoid obstacles and follow a line using C++ code

---

Session 6: IoT Projects - Introduction & Control via Blynk

Duration: 90 minutes

- **Introduction to IoT with Arduino:**
  - Overview of IoT and Blynk app
  - Setting up Blynk for remote control
- **C++ for IoT Integration:**
  - Using C++ to interface sensors with Blynk and send data over the internet
- **Hands-On:**
  - Create a basic IoT project (e.g., controlling a LED or motor remotely via Blynk)
  - Push sensor data to the Blynk app (e.g., temperature, humidity)

Session 7: Smart Car - Advanced Features & Remote Control

Duration: 90 minutes

- **Adding Remote Control:**
  - Use IR receiver or Bluetooth for remote control
- **C++ for Remote Control:**
  - Programming the car to respond to IR/Bluetooth commands using C++
- **Hands-On:**
  - Implement remote control for smart car (IR or Bluetooth-based)

- Add advanced features like speed control or turning

Session 8: Final Project & Wrap-Up

Duration: 90 minutes

- **Final Project:**
  - Design and build an autonomous car or a smart system with multiple sensors (students can choose from various project ideas)
  - Integrate IoT elements or sensors for added functionality (e.g., remote monitoring or control)
  - C++ programming for full project control
- **Wrap-Up:**
  - Troubleshoot and finalize the projects
  - Review key concepts learned throughout the course
  - Discuss next steps for continued learning and resources

Key Additions:

1. **C++ Programming Integration**: Each session introduces key C++ concepts (variables, loops, functions, and conditional statements) that are directly applied to the projects. This ensures that students understand how to write the necessary code for controlling motors, sensors, and other components in their smart car and IoT projects.
2. **Hands-On C++ Coding**: Throughout the course, students will practice C++ coding by directly interacting with hardware (e.g., controlling motors, reading sensor data, and managing IoT devices). This hands-on coding approach ensures that they get practical experience with the language as they build functional projects.
3. **Remote Control & Smart Car Programming**: Sessions dedicated to the smart car emphasize C++ programming for autonomous movement, obstacle avoidance, line tracking, and remote control, which will give students a real-world application of C++

skills.