



International Journal of Innovative Research in Science Engineering and Technology (IJIRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 8.699

Volume 14, Issue 5, May 2025

Deployment of Solar and IoT-Based Hybrid EV Charging Station with Slot Pre-Booking and Auto- Switching Features

Dr. Zuhaib Baig¹, Namratha C S², Nethravathi³, Sanjay D P⁴, Uday Aravindan S⁵Assistant Professor,

Dept. of EEE, VVIET, Karnataka, India¹

Student, Dept. of EEE, VVIET, Karnataka, India²

Student, Dept. of EEE, VVIET, Karnataka, India³

Student, Dept. of EEE, VVIET, Karnataka, India⁴

Student, Dept. of EEE, VVIET, Karnataka, India⁵

ABSTRACT: This project focuses on the deployment of a Solar and IoT-Based Hybrid EV Charging Station. This modern solution aims to meet the energy demands of Electric Vehicle (EV) users in a sustainable and user-centric manner by utilizing solar power as a primary energy source, thereby reducing reliance on the conventional power grid and lowering carbon footprints. The hybrid nature of the station ensures energy reliability through automatic switching between solar and grid power based on availability, guaranteeing uninterrupted service. The report outlines the design and implementation framework, covering hardware components (solar panels, inverters, batteries, charge controllers, IoT sensors), software architecture for slot booking and energy management, communication protocols, and economic/environmental impact analysis. It also addresses challenges like cybersecurity, system redundancy, and interoperability with various EV models. This project demonstrates the feasibility, benefits, and real-world potential of deploying a smart hybrid EV charging station, setting the stage for future advancements in green energy and intelligent transportation systems.

I. INTRODUCTION

The rapid electrification of the global transportation sector is a cornerstone in the fight against climate change and the effort to reduce dependence on fossil fuels. Electric vehicles (EVs), hailed as the future of mobility, promise cleaner air, lower emissions, and a pathway to sustainable urban development. However, the accelerated adoption of EVs presents a parallel challenge: the need for a widespread, efficient, and intelligent charging infrastructure. As countries and corporations invest heavily in electric mobility, the infrastructure to support this transition must evolve beyond traditional models. One of the most promising avenues is the integration of renewable energy sources, particularly solar power, with smart technologies such as the Internet of Things (IoT). This convergence paves the way for hybrid EV charging stations equipped with advanced features like slot pre-booking and automatic source-switching.

Another key innovation in the proposed model is the automatic power source switching feature. Given the intermittent nature of solar energy, especially during cloudy weather or nighttime, a hybrid charging station must seamlessly switch to the grid when solar power is insufficient. This switch must be instantaneous and intelligent, guided by IoT-based energy monitoring systems that evaluate input from solar panels, battery storage levels, and load demands in real time. The seamless transition ensures that users are not affected by fluctuations in energy supply, thus maintaining service continuity and enhancing trust in the EV infrastructure.

II. METHODOLOGY

The hybrid solar power management system integrates solar, battery, and grid energy sources to provide uninterrupted power to critical loads through intelligent control and monitoring using an ESP-32 microcontroller. The system aims to efficiently harness solar energy, utilize battery storage when necessary, and switch to grid power as a last resort, ensuring continuous power supply to critical loads. The process begins with solar energy generation through a solar panel, which converts sunlight into DC electricity. This DC output is directed to a solar charge controller, responsible for regulating the voltage and current from the solar panels to ensure safe and efficient battery charging. The controller

prevents overcharging, over-discharging, and under-voltage, which could potentially damage the battery backup system. Once regulated, the energy flows into a battery backup system, storing excess solar energy generated during the day for later use when sunlight is unavailable. Voltage sensors connected to the charge controller monitor the voltage levels of the solar input and battery output, providing real-time data to the ESP-32 microcontroller.

In parallel with the solar setup, an AC grid acts as a secondary or backup power source. An AC voltage sensor checks the availability and stability of the grid supply. The AC power then enters an AC to DC rectifier, converting the alternating current into direct current, aligning it with the solar power system's DC architecture. Voltage sensors and a

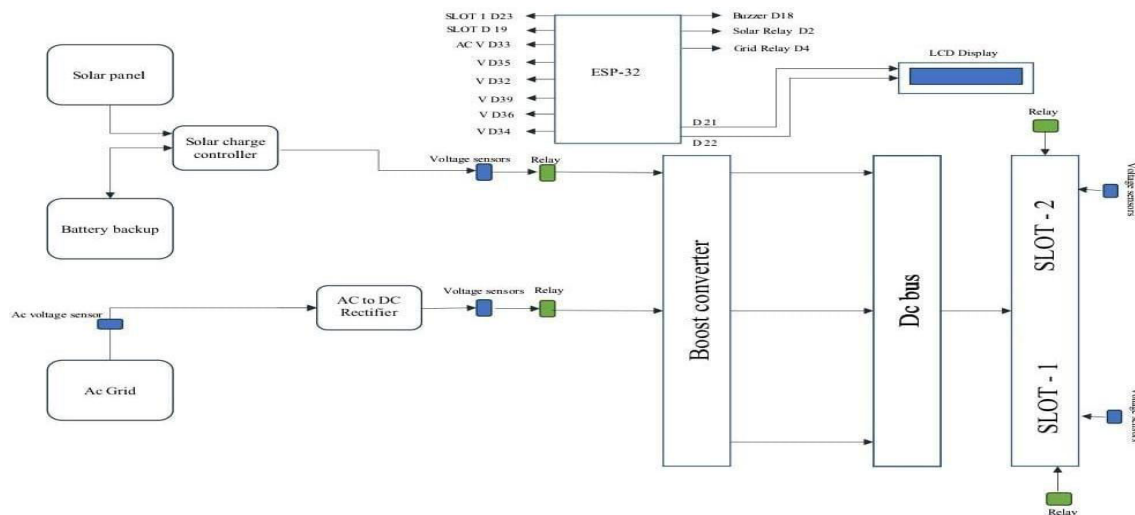


Fig .1.1 Block Diagram

relay are positioned after the rectifier to monitor and control the flow of rectified grid power. If the battery is depleted and solar energy is unavailable or insufficient, the ESP-32 evaluates the condition and commands the grid relay to allow energy flow from the grid to the load.

The regulated DC power—either from the battery (solar) or the grid—flows into a boost converter, increasing the voltage level to match the system's required bus voltage. This ensures a stable and sufficient supply to the DC bus, distributing energy to the connected loads. From the DC bus, power is distributed to SLOT-1 and SLOT-2, each representing separate power-consuming zones or loads.

Each slot is equipped with its own relay and voltage sensors to monitor consumption and enable or disable power delivery based on priority, availability of power, and system health. The ESP-32 microcontroller collects real-time data from all voltage sensors, evaluates system status, and issues control signals to various relays. It interfaces with voltage sensors through analog input pins and drives relays via digital output pins. A buzzer connected to pin D18 provides audible alerts for fault conditions, battery low status, or switching events. An LCD display connected through digital pins D21 and D22 offers a user interface showing system metrics, voltage levels, source status (solar or grid), and load status (SLOT-1 and SLOT-2 activity). This ESP-32-driven control system enables automation, real-time monitoring, and dynamic decision-making, enhancing the system's flexibility and reliability. The control algorithm running on the ESP-32 prioritizes energy sources and loads. It first checks for solar energy availability and battery voltage. If solar energy is insufficient, it evaluates grid availability and activates the grid relay if conditions are favorable. In case neither solar nor grid can supply adequate power, the system can deactivate SLOT-2 to conserve energy for SLOT-1. Voltage sensors continuously monitor for irregularities, and if detected, the corresponding relay is immediately turned off to protect downstream components.

III. SOFTWARE REQUIREMENT

Program

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h> #include <ZMPT101B.h> #include <WiFi.h>
#include <ESPAsyncWebServer.h>
// Define I2C LCD LiquidCrystal_I2C lcd(0x27, 16, 2);
const float R1 = 30000.0; // Resistor R1 value (30k ohms) const float R2 = 7500.0; // Resistor R2 value (7.5k ohms)
// ADC Configuration
const int ADC_MAX_VALUE = 4095; // For ESP32, the ADC resolution is 12 bits const float
ADC_REFERENCE_VOLTAGE = 3.5; // ESP32 reference voltage in volts
// Voltage sensor pins
#define SOLAR_VOLTAGE_PIN 35
#define GRID_VOLTAGE_PIN 32
#define AC_VOLTAGE_PIN 33
#define SLOT_1_VOLTAGE_PIN 36
#define SLOT_2_VOLTAGE_PIN 34
#define BATTERY_VOLTAGE_PIN 39
// Relay pins
#define SOLAR_RELAY_PIN 2
#define GRID_RELAY_PIN 4
#define INVERTER_RELAY_PIN 5
#define SLOT_1_RELAY_PIN 23
#define SLOT_2_RELAY_PIN 19
// Buzzer pin
#define BUZZER_PIN 18
// Voltage thresholds
#define SOLAR_MIN_VOLTAGE 11.0
#define SOLAR_MAX_VOLTAGE 12.0
#define BATTERY_FULLY_CHARGED_VOLTAGE 13.0
#define BATTERY_CONNECTED_THRESHOLD 0.0
// AC Voltage sensor
ZMPT101B acSensor(AC_VOLTAGE_PIN);
// WiFi credentials
const char* ssid = "Major Project"; const char* password = "Project@123";
// Web server AsyncWebServer server(80);
// Global variables
float solarVoltage = 0.0; float gridVoltage = 0.0; float acVoltage = 0.0; float slot1Voltage = 0.0; float slot2Voltage =
0.0; float batteryVoltage = 0.0; String powerSource = "";
String previousSource = "";
unsigned long messageDisplayStart = 0;
unsigned long messageDuration = 2000; // 2 seconds for displaying messages bool displayingMessage = false;
bool slot1Booked = false; bool slot2Booked = false; String slot1Name = ""; String slot2Name = ""; String slot1Id = "";
String slot2Id = "";
void setup() {
// Initialize LCD lcd.init(); lcd.backlight();
// Initialize relays and buzzer pinMode(SOLAR_RELAY_PIN, OUTPUT);
pinMode(GRID_RELAY_PIN, OUTPUT); pinMode(INVERTER_RELAY_PIN, OUTPUT);
pinMode(SLOT_1_RELAY_PIN, OUTPUT); pinMode(SLOT_2_RELAY_PIN, OUTPUT); pinMode(BUZZER_PIN,
OUTPUT);
// Turn off all relays and buzzer initially digitalWrite(SOLAR_RELAY_PIN, HIGH);
digitalWrite(GRID_RELAY_PIN, HIGH); digitalWrite(INVERTER_RELAY_PIN, HIGH);
digitalWrite(SLOT_1_RELAY_PIN, HIGH); digitalWrite(SLOT_2_RELAY_PIN, HIGH);
digitalWrite(BUZZER_PIN, LOW); lcd.setCursor(4, 0);
```

```
lcd.print(" Welcome"); lcd.setCursor(0, 1); lcd.print("Hybrid Grid"); delay(5000);
lcd.clear();
// Connect to WiFi WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED) { delay(1000);
Serial.println("Connecting to WiFi...");
}
Serial.println("Connected to WiFi");
// Initialize the web server
server.on("/", HTTP_GET, [](AsyncWebServerRequest *request) { String html = R"rawliteral(
<!DOCTYPE html><html>
<head>
<title>Hybrid Grid Monitor</title>
<style>
body {
font-family: Arial, sans-serif; background-color: #f4f4f4; margin: 0;
padding: 0; display: flex;
flex-direction: column; align-items: center; justify-content: center;
}
h1 {
color: #333;
}
.container { background: #ffffff; border-radius: 8px;
padding: 20px;
box-shadow: 0 2px 5px rgba(0, 0, 0, 0.2); max-width: 400px;
width: 100%;
}
status {
margin: 10px 0; padding: 10px;
border: 1px solid #ddd; border-radius: 5px; background-color: #f9f9f9; text-align: left;
.status span {
font-weight: bold;
}
form {
margin-top: 20px;
}
select, input[type="text"] { width: 100%;
padding: 10px; margin-bottom: 10px;
border: 1px solid #ccc; border-radius: 5px;
}
input[type="submit"], button {
width: 100%;
padding: 10px;
background-color: #4CAF50; border: none;
border-radius: 5px; color: white; font-size: 16px; cursor: pointer;
}
input[type="submit"]:hover, button:hover { background-color: #45a049;
}
.message {
margin-top: 20px; color: green;
font-weight: bold;
}
.error {
color: red;
font-weight: bold;
}
```

```
}
.button-container { margin-top: 20px;
}
.button-container button { margin-top: 10px;
}
</style>
</head>
<body>
<div class="container">
<h1>Hybrid Grid Monitor</h1>
<div id="data">
<div class="status">Solar Voltage: <span id="solarVoltage">0</span> V</div>
<div class="status">Grid Voltage: <span id="gridVoltage">0</span> V</div>
<div class="status">AC Voltage: <span id="acVoltage">0</span> V</div>
<div class="status">Battery Voltage: <span id="batteryVoltage">0</span> V</div>
<div class="status">Slot 1 Voltage: <span id="slot1 Voltage">0</span> V</div>
<div class="status">Slot 2 Voltage: <span id="slot2Voltage">0</span> V</div>
</div>
<h2>Slot Pre-Booking</h2>
<form id="bookingForm">
<input type="checkbox" id="slot1" name="slot" value="1"> Slot 1<br>
<input type="checkbox" id="slot2" name="slot" value="2"> Slot 2<br>
<input type="text" id="idNumber" name="idNumber" placeholder="Enter 4-digit ID"
maxLength="4" pattern="\d{4}">
<div class="button-container">
<button type="button" onclick="preBookSlot()">Pre-Book Slot</button>
</div>
</form>
<h2>Verify Slot</h2>
<form id="verifyForm">
<input type="text" id="verifyIdNumber" name="verifyIdNumber" placeholder="Enter 4-digit ID to verify"
maxLength="4" pattern="\d{4}">
<div class="button-container">
<button type="button" onclick="verifySlot()">Verify Slot</button>
</div>
</form>
<div id="message" class="message"></div>
<div id="error" class="error"></div>
</div>
<script>
async function fetchData() {
const response = await fetch("/data"); const data = await response.json();
document.getElementById("solarVoltage").innerText = data.solarVoltage;
document.getElementById("gridVoltage").innerText = data.gridVoltage;
document.getElementById("acVoltage").innerText = data.acVoltage;
document.getElementById("batteryVoltage").innerText = data.batteryVoltage;
document.getElementById("slot1 Voltage").innerText = data.slot1 Voltage;
document.getElementById("slot2Voltage").innerText = data.slot2Voltage; updateSlotStatus(data);
}
function updateSlotStatus(data) {
const slot1Status = document.getElementById("slot1"); const slot2Status = document.getElementById("slot2");
slot1Status.disabled = data.slot1Booked; slot2Status.disabled = data.slot2Booked;
}
async function preBookSlot() {
const slot1Checked = document.getElementById("slot1").checked;
```

```
const slot2Checked = document.getElementById("slot2").checked;
const idNumber = document.getElementById("idNumber").value;
if ((slot1Checked || slot2Checked) && idNumber.length === 4) { const slot = slot1Checked ? "1" : (slot2Checked ? "2" : ""); const formData = new URLSearchParams(); formData.append("slot", slot);
formData.append("idNumber", idNumber); const response = await fetch("/prebook", {
method: 'POST', headers: {
'Content-Type': 'application/x-www-form-urlencoded',
},
body: formData.toString(),
});
const message = await response.text(); document.getElementById("message").innerText = message;
document.getElementById("error").innerText = "";
} else {
document.getElementById("error").innerText = "Please select a slot and enter a 4-digit ID.";
document.getElementById("message").innerText = "";
}
}
async function verifySlot() {
const idNumber = document.getElementById("verifyIdNumber").value; if (idNumber.length === 4) {
const formData = new URLSearchParams(); formData.append("idNumber", idNumber);
const response = await fetch("/verify", { method: 'POST',
headers: {
'Content-Type': 'application/x-www-form-urlencoded',
},
body: formData.toString(),
});
const message = await response.text(); document.getElementById("message").innerText = message;
document.getElementById("error").innerText = "";
} else {
document.getElementById("error").innerText = "Please enter a valid 4-digit ID.";
document.getElementById("message").innerText = "";
}
}
setInterval(fetchData, 1000); // Update every second
</script>
</body>
</html>
)rawliteral";
request->send(200, "text/html", html);
});
server.on("/data", HTTP_GET, [(AsyncWebServerRequest *request) { String json = "{";
json += "\"solarVoltage\":" + String(solarVoltage) + ","; json += "\"gridVoltage\":" + String(gridVoltage) + ","; json +=
 "\"acVoltage\":" + String(acVoltage) + ",";
json += "\"batteryVoltage\":" + String(batteryVoltage) + ",";
json += "\"slot1Voltage\":" + String(slot1Voltage) + ","; json += "\"slot2Voltage\":" + String(slot2Voltage) + ",";
json += "\"slot1Booked\":" + String(slot1Booked ? "true" : "false") + ","; json += "\"slot2Booked\":" +
String(slot2Booked ? "true" : "false");
json += "}";
request->send(200, "application/json", json);
});
// Endpoint to handle pre-booking
server.on("/prebook", HTTP_POST, [(AsyncWebServerRequest *request) {
if (request->hasParam("slot", true) && request->hasParam("idNumber", true)) { String slot = request-
>getParam("slot", true)->value();
String idNumber = request->getParam("idNumber", true)->value(); if (idNumber.length() != 4) {
```

```
request->send(400, "text/plain", "ID number must be 4 digits."); return;
}
// Check if the slot is available and pre-book it if (slot == "1" && !slot1Booked) { slot1Booked = true;
slot1Id = idNumber; // Store ID number
request->send(200, "text/plain", "Slot 1 pre-booked successfully.");
} else if (slot == "2" && !slot2Booked) { slot2Booked = true;
slot2Id = idNumber; // Store ID number
request->send(200, "text/plain", "Slot 2 pre-booked successfully.");
} else {
request->send(400, "text/plain", "Slot is already booked or invalid.");
}
} else {
request->send(400, "text/plain", "Please provide both slot and ID number.");
}
});
// Endpoint to handle verification
server.on("/verify", HTTP_POST, [](AsyncWebServerRequest *request) { if (request->hasParam("idNumber", true)) {
String idNumber = request->getParam("idNumber", true)->value(); if (idNumber.length() != 4) {
request->send(400, "text/plain", "ID number must be 4 digits."); return;
}
// Verify ID against pre-booked slots
if (slot1Booked && slot1Id == idNumber) { slot1Booked = false;
digitalWrite(SLOT_1_RELAY_PIN, LOW); // Turn on Slot 1 relay
request->send(200, "text/plain", "Slot 1 verified successfully and relay turned ON.");
} else if (slot2Booked && slot2Id == idNumber) { slot2Booked = false;
digitalWrite(SLOT_2_RELAY_PIN, LOW); // Turn on Slot 2 relay
request->send(200, "text/plain", "Slot 2 verified successfully and relay turned ON.");
}
else {
request->send(400, "text/plain", "Verification failed: Incorrect ID or slot not pre-booked.");
}
} else {
request->send(400, "text/plain", "Invalid verification details.");
}
});
server.begin();
}
void loop() {
// Read sensor values
solarVoltage = readVoltage(SOLAR_VOLTAGE_PIN); gridVoltage = readVoltage(GRID_VOLTAGE_PIN);
acVoltage = acSensor.getRmsVoltage();
batteryVoltage = readVoltage(BATTERY_VOLTAGE_PIN); slot1Voltage = readVoltage(SLOT_1_VOLTAGE_PIN);
slot2Voltage = readVoltage(SLOT_2_VOLTAGE_PIN);
// Determine power source and switch relays
if (solarVoltage > SOLAR_MAX_VOLTAGE) { switchPowerSource("Solar", 1); digitalWrite(SOLAR_RELAY_PIN,
LOW); digitalWrite(GRID_RELAY_PIN, HIGH); digitalWrite(INVERTER_RELAY_PIN, HIGH);
} else if (solarVoltage < SOLAR_MIN_VOLTAGE && gridVoltage > 0) { switchPowerSource("Grid", 2);
digitalWrite(SOLAR_RELAY_PIN, HIGH); digitalWrite(GRID_RELAY_PIN, LOW);
digitalWrite(INVERTER_RELAY_PIN, LOW); // Activate inverter
} else if (solarVoltage == 0 && gridVoltage == 0 && slot1Voltage == 0 && slot2Voltage == 0 && acVoltage == 0) {
switchPowerSource("Grid", 2); digitalWrite(SOLAR_RELAY_PIN, HIGH);
digitalWrite(GRID_RELAY_PIN, LOW); digitalWrite(INVERTER_RELAY_PIN, LOW);
}
// Control Slot 1 and Slot 2 relays
if (!slot1Booked && slot1Voltage > 0) { digitalWrite(SLOT_1_RELAY_PIN, LOW);
```

```
} else {
digitalWrite(SLOT_1_RELAY_PIN, HIGH);
}
if (!slot2Booked && slot2Voltage > 0) { digitalWrite(SLOT_2_RELAY_PIN, LOW);
} else {
digitalWrite(SLOT_2_RELAY_PIN, HIGH);
}
// Display the system status displayDefaultStatus();
delay(1000); // Delay for 1 second before next update
}
float readVoltage(int pin) {
int sensorValue = analogRead(pin);
float sensorValue2 = (sensorValue * ADC_REFERENCE_VOLTAGE) / ADC_MAX_VALUE; float voltage =
sensorValue2 * (R1 + R2) / R2; // Adjust according to the sensor divider circuit return voltage;
}
void switchPowerSource(String source, int beepCount) { if (powerSource != source) {
previousSource = powerSource; powerSource = source;
// Buzzer notification
for (int i = 0; i < beepCount; i++) { digitalWrite(BUZZER_PIN, HIGH);
delay(200); digitalWrite(BUZZER_PIN, LOW); delay(200);
}
// Display switching information lcd.clear();
lcd.setCursor(0, 0);
if (source == "Grid" && previousSource == "Solar") { lcd.print("Switching Solar");
lcd.setCursor(0, 1); lcd.print("to Grid");
} else if (source == "Solar" && previousSource == "Grid") { lcd.print("Switching Grid");
lcd.setCursor(0, 1); lcd.print("to Solar");
}
delay(messageDuration);
}
}
void displayDefaultStatus() { lcd.clear();
// First row: Solar and Grid voltages lcd.setCursor(0, 0);
lcd.print("S:"); lcd.print(solarVoltage); lcd.print("V G:"); lcd.print(gridVoltage); lcd.print("V");
// Second row: Mode with integrated battery status lcd.setCursor(0, 1);
lcd.print("Mode:");
if (acVoltage > 150) {
// Running on mains
if ((slot1Voltage || slot2Voltage) > BATTERY_CONNECTED_THRESHOLD) {
if ((slot1Voltage || slot2Voltage) > BATTERY_FULLY_CHARGED_VOLTAGE) { lcd.print("FC"); // Fully Charged
(on mains)
digitalWrite(BUZZER_PIN, HIGH); delay(200); digitalWrite(BUZZER_PIN, LOW); delay(200);
} else {
lcd.print("MBC"); // Mains with Battery
}
} else {
lcd.print("MA"); // Mains Alone
}
} else if (acVoltage < 50) {
// Running on backup battery
if ((slot1Voltage || slot2Voltage) > BATTERY_CONNECTED_THRESHOLD) {
if ((slot1Voltage || slot2Voltage) > BATTERY_FULLY_CHARGED_VOLTAGE) { lcd.print("FC");
digitalWrite(BUZZER_PIN, HIGH); delay(200); digitalWrite(BUZZER_PIN, LOW); delay(200);
} else {
lcd.print("BBC"); // Backup Battery Connected
```

```
}  
} else {  
  lcd.print("BB"); // Backup Battery Alone  
}  
}  
lcd.setCursor(10, 1); lcd.print("Slot:");  
if (slot1Voltage > 0 && slot2Voltage > 0) { lcd.print("1&2");  
} else if (slot1Voltage > 0) { lcd.print("1");  
} else if (slot2Voltage > 0) { lcd.print("2");  
} else {  
  lcd.print("NC"); // No connection  
}  
lcd.setCursor(0, 2); lcd.print("SLT1:"); lcd.print(slot1Voltage); lcd.setCursor(10, 2); lcd.print("SLT2:");  
lcd.print(slot2Voltage);  
}
```

IV. ADVANTAGES

1. Eco-friendly: Uses solar energy, reducing carbon emissions and promoting sustainability.
2. Reliable Power: Auto-switching between solar, battery, and grid ensures uninterrupted charging.
3. Smart Management: IoT enables real-time monitoring, fault detection, and efficient energy use.
4. User Convenience: Slot pre-booking minimizes wait times and enhances user experience.
5. Cost-Effective: Lowers operational costs through renewable energy and reduced manual intervention.

V. FUTURE SCOPE

Smart electric vehicle (EV) charging stations are transforming urban mobility by integrating advanced technologies and sustainable energy solutions. These stations leverage Internet of Things (IoT) connectivity to enable real-time monitoring, seamless navigation, and adaptive traffic management based on charging demand. This integration allows for efficient utilization of resources and improved traffic flow. Features like slot pre-booking and automatic switching between power sources minimize waiting times and ensure uninterrupted service. Such conveniences enhance user satisfaction and encourage wider EV adoption. Integrating solar energy into EV charging stations reduces reliance on the grid, lowers carbon emissions, and supports global renewable energy targets. Solar-powered stations provide clean energy and contribute to environmental sustainability. Solar-powered, IoT-enabled charging systems offer scalable and modular solutions, making them ideal for remote locations lacking stable grid access. These systems provide reliable charging options and promote clean energy adoption in underserved areas. Smart charging stations enable dynamic pricing, subscription-based access, and energy trading, creating diversified income opportunities for operators. These models enhance the economic viability of EV infrastructure and support its expansion. In summary, the evolution of EV charging infrastructure through smart technologies and sustainable energy integration is pivotal in shaping the future of urban mobility. These advancements not only enhance user experiences but also contribute to environmental sustainability and economic growth.

VI. CONCLUSION

The deployment of Solar and IoT-based hybrid EV charging stations with slot pre-booking and auto-switching features represents a significant step toward sustainable and intelligent transportation infrastructure. By combining renewable energy with smart technology, these stations offer enhanced efficiency, user convenience, and reduced environmental impact. They support the growing demand for electric vehicles while addressing grid challenges and promoting clean energy use. This innovative approach not only aligns with global sustainability goals but also opens new avenues for scalable, data-driven energy solutions in both urban and rural settings.

REFERENCES

1. S. Priya, R. Sanjay, M. S. Aravind "Solar Powered Grid Integrated Charging Station with Hybrid Energy Storage System"2023.
2. Dinh Thi Thanh Loan, Doan Quang Trung, Mai Huu Thuan "Research and Design of an Electric Car Charging Station Using Solar Energy Integrating an IoT Technology Monitoring System" 2024.
3. M.Krithika,V.Abhaikumar,K.P.Anbuuday"IoT-Based EV Charge Prioritizing System for PV- Powered Smart Buildings" 2024.
4. Younes Ledmaoui, Adila El Maghraoui, Ahmed Chebak "Solar Charging Station for Electric Vehicles with IoT Solution for Monitoring Energy Production" 2023 .
5. V. Rad ha Krishna, K. Vijaya, M. Hrishithi, V. Richa Sanjitha, P. Varshitha "Smart Solar Powered Battery System for EV with IoT Voltage Monitoring" 2023



INTERNATIONAL
STANDARD
SERIAL
NUMBER
INDIA



INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

 9940 572 462  6381 907 438  ijirset@gmail.com



www.ijirset.com

Scan to save the contact details