OCPP Charging Station Management System

System Documentation for Electrical and Hardware Engineers

Version: 2.0 **Last Updated:** January 2025 **Document Type:** Technical System Overview

---

Table of Contents

---

1. Executive Summary

1.1 System Purpose

This system is a production-ready **Central System (CS)** for managing electric vehicle charging infrastructure. It implements the OCPP 1.6 (Open Charge Point Protocol) standard, enabling centralized monitoring, control, and billing of EV charging stations.

Primary Functions: - Real-time communication with OCPP 1.6-compliant charge points - Monitoring of charger status, energy consumption, and electrical parameters - Remote control of charging operations (start, stop, availability management) - Mobile app and QR code-based user authentication with virtual RFID card system - Automated billing based on energy consumption (kWh) - Over-the-air (OTA) firmware updates for connected chargers - User management with role-based access control

1.2 Problem Solved

Managing a network of EV chargers requires: - **Standardized Communication:** OCPP 1.6 provides vendor-neutral protocol for charge point communication - **Real-time Monitoring:** Instant visibility into charger status, electrical parameters, and faults - **Remote Operations:** Ability to start, stop, and control chargers without physical access - **Automated Billing:** Accurate energy measurement and automatic payment processing - **Firmware Management:** Centralized firmware updates across all charge points - **Access Control:** Different permission levels for administrators and end users

1.3 Key Benefits

For Hardware Integration: - Works with any OCPP 1.6-compliant charging hardware - Standard WebSocket-based communication (no proprietary protocols) - Comprehensive message logging for debugging and compliance - Real-time electrical parameter monitoring (voltage, current, power, energy)

For Operations: - Centralized dashboard for monitoring all charging stations - Automated billing eliminates manual intervention - Geographic station mapping for user convenience - Complete transaction history and analytics

For Scalability: - Asynchronous architecture supports hundreds of simultaneous connections - Cloud-based deployment (currently on Render and Vercel) - Redis caching for fast connection state lookups - PostgreSQL database for reliable data persistence

1.4 Technology Overview

The system is built with modern web technologies optimized for real-time communication:

- **Backend:** Python-based asynchronous server using FastAPI framework
- **OCPP Implementation:** python-ocpp library (OCPP 1.6 specification)
- **Communication:** WebSocket protocol for persistent charge point connections
- **Database:** PostgreSQL for transactional data storage
- **Cache:** Redis for connection state tracking
- **Frontend:** Web-based interface using Next.js (React framework)
- **Authentication:** Clerk platform for user management and security
- **Payments:** Razorpay integration for wallet recharges

---

2. System Architecture

2.1 High-Level Architecture

```
┌──────────────────────────────────────────────────────────────────────
│                          OCPP CENTRAL SYSTEM
│
│
│
│  ┌──────────────────────────────┐
```

```
┌──────────────────────────────┐  │
│  │     Web Frontend            │      │   Backend Server
│  │
│  │     (Next.js)               │      │   (FastAPI/Python)
│  │
│  │                             │      │
│  │
│  │  ┌──────────────────────┐   │      │  ┌─────────────────────────┐ │
│
│  │  │ Admin Dashboard      │   │◄────►│  │ REST API Endpoints      │ │
│
│  │  │ - Station Management │   │      │  │ - /api/stations         │ │
│
│  │  │ - Charger Control    │   │      │  │ - /api/chargers         │ │
│
│  │  │ - User Management    │   │      │  │ - /api/transactions     │ │
│
│  │  │ - Firmware Updates   │   │      │  │ - /api/firmware         │ │
│
│  │  └──────────────────────┘   │      │  └─────────────────────────┘ │
│
│  │                             │      │
│  │
│  │  ┌──────────────────────┐   │      │  ┌─────────────────────────┐ │
│
│  │  │ User Interface       │   │◄────►│  │ OCPP WebSocket Server   │ │
│
│  │  │ - Station Finder     │   │      │  │ Endpoint:               │ │
│
│  │  │ - QR Scanner         │   │      │  │ /ocpp/{charger_id}      │ │
│
│  │  │ - Session History    │   │      │  │                         │ │
│
│  │  │ - Wallet Management  │   │      │  │ - Message Handler       │ │
│
│  │  └──────────────────────┘   │      │  │ - Connection Manager    │ │
│
│  │                             │      │  │ - Heartbeat Monitor     │ │
│
│  └─────────────────────────────┘      │  └─────────────────────────┘ │
│
│                                       │
│  │
│                                       │  ┌─────────────────────────┐ │
│
│                                       │  │ Business Logic          │ │
│
│                                       │  │ - Billing Service       │ │
│
│                                       │  │ - Wallet Service        │ │
│
```

```
│                                      │  │ - Firmware Service      │ │
│
│                                      │  └─────────────────────────┘ │
│
│
└──────────────────────────────┘  │
│
│
│  ┌─────────────────────────────┐
┌──────────────────────────────┐  │
│  │  PostgreSQL Database        │      │        Redis Cache
│  │
│  │                             │      │
│  │
│  │  - Users & Authentication   │      │  - Connected Chargers Map
│  │
│  │  - Charging Stations        │      │  - Connection State
│  │
│  │  - Chargers & Connectors    │      │  - Session Data
│  │
│  │  - Transactions             │      │
│  │
│  │  - Meter Values             │      │
│  │
│  │  - Wallets & Billing        │      │
│  │
│  │  - OCPP Message Logs        │      │
│  │
│  │  - Firmware Files           │      │
│  │
│  └─────────────────────────────┘
└──────────────────────────────┘  │
│
│
└─────────────────────────────────────────────────────────────────────┘
                                                   │
                                                   │ WebSocket (OCPP 1.6)
                                                   │ ws://server/ocpp/{charger_id}
                                                   │
         ┌───────────────────────────────────────────────────────┐
         │                                                         │
                                                                │
┌───────▼────────┐    ┌──────────────┐    ┌──────────────┐      │
│ Charge Point 1 │    │Charge Point 2│    │Charge Point N│  ... │
│                │    │              │    │              │      │
│ Type 2         │    │ CCS Combo    │    │ Multi-outlet │      │
│ 7.4 kW AC      │    │ 50 kW DC     │    │ 22 kW AC     │      │
│ Single Outlet  │    │ Single Outlet│    │ Dual Outlets │      │
└────────────────┘    └──────────────┘    └──────────────┘

  OCPP 1.6 Compatible Charging Hardware
```

2.2 Component Descriptions

2.2.1 Web Frontend

- **Purpose:** User interface for administrators and EV drivers
- **Technology:** Browser-based application (desktop and mobile compatible)
- **Key Features:**
  - Interactive maps showing charging station locations
  - Real-time charger status display
  - QR code scanning for quick charger access
  - Transaction history with energy consumption charts
  - Administrative controls for system management

2.2.2 Backend Server

REST API Server: - Handles HTTP requests from web frontend - Manages CRUD operations (Create, Read, Update, Delete) - Enforces authentication and authorization - Provides data endpoints for dashboards and reports

OCPP WebSocket Server: - Maintains persistent connections with charge points - Implements OCPP 1.6 message protocol - Routes messages between charge points and system logic - Monitors connection health with heartbeat mechanism - Stores all OCPP communications for audit trail

Business Logic Services: - **Billing Service:** Calculates charges based on energy consumption and tariffs - **Wallet Service:** Manages user account balances and transactions - **Firmware Service:** Handles OTA updates to charge points - **Retry Service:** Automatically retries failed operations (e.g., billing)

2.2.3 PostgreSQL Database

Primary data store containing: - **User Data:** Account information, roles, virtual RFID card IDs for OCPP identification - **Infrastructure Data:** Stations (locations), chargers, connectors - **Operational Data:** Transactions, meter values, charger status - **Financial Data:** Wallets (INR balances), payments (Razorpay), billing records, tariffs (₹/kWh rates) - **Audit Data:** Complete OCPP message logs - **Firmware Data:** Binary files and update tracking

2.2.4 Redis Cache

High-speed in-memory store for: - Real-time connection status of all chargers - Active session state - Temporary data requiring fast access - Reduces database load for frequently accessed data

2.2.5 Charge Points (External Hardware)

- Any OCPP 1.6-compliant charging station
- Initiates WebSocket connection to central system
- Sends status updates and meter readings
- Receives and executes remote commands
- Examples: ABB Terra, EVBox, Schneider Electric, Delta, and others

2.3 External Service Integrations

Clerk (Authentication Service): - Manages user accounts and login sessions - Issues secure JWT (JSON Web Tokens) for API access - Provides webhooks for user event synchronization - Handles password security and multi-factor authentication

Razorpay (Payment Gateway): - Indian payment gateway for processing wallet recharges - Supports credit/debit cards, UPI, net banking, wallets - All transactions in INR (Indian Rupees) - Provides secure payment checkout interface - Sends payment confirmation webhooks - Handles refunds and transaction disputes

2.4 Data Flow Overview

```
User Action (Web/Mobile) → REST API → Business Logic → Database
                                                      ↓
                                                   Response

Charge Point → WebSocket → OCPP Handler → Business Logic → Database
                                                          ↓
                                                       Response
                                                          ↓
Charge Point ← WebSocket ← OCPP Handler ← Business Logic
```

---

3. OCPP Protocol Integration

3.1 OCPP 1.6 Overview

OCPP (Open Charge Point Protocol) is an application protocol for communication between EV charging stations and a central management system. Version 1.6 is the most widely adopted standard in the industry.

Key Characteristics: - **Transport:** WebSocket (persistent bidirectional connection) - **Message Format:** JSON (JavaScript Object Notation) - **Architecture:** Request-Response pairs with unique message IDs - **Direction:** Bidirectional (charge point ↔ central system)

3.2 Communication Architecture

```
Charge Point                          Central System
     │                                      │
     │  1. TCP/IP Connection Establishment  │
     ├─────────────────────────────────────►│
     │                                      │
     │  2. WebSocket Upgrade Handshake      │
     ├─────────────────────────────────────►│
     │◄─────────────────────────────────────┤
     │                                      │
     │  3. OCPP Session Active              │
```

```
|◄──────────────────────────────────►|
|         JSON Messages              |
|                                    |
```

Connection Details: - **Protocol:** WebSocket (RFC 6455) - **Endpoint Format:** `ws://[server]:[port]/ocpp/{charge_point_id}` - **Sub-Protocol:** `ocpp1.6` (specified in WebSocket handshake) - **Security:** Can be secured with WSS (WebSocket Secure) over TLS - **Persistence:** Connection remains open for duration of charger operation

3.3 OCPP Message Types

OCPP defines two categories of messages:

1. Charge Point → Central System (Initiated by Hardware): - BootNotification - Heartbeat - StatusNotification - StartTransaction - StopTransaction - MeterValues - FirmwareStatusNotification - DiagnosticsStatusNotification

2. Central System → Charge Point (Initiated by System/Admin): - RemoteStartTransaction - RemoteStopTransaction - ChangeAvailability - UpdateFirmware - GetDiagnostics - Reset - UnlockConnector

3.4 Message Structure

All OCPP messages follow this JSON array format:

```
[
  MessageTypeId,
  UniqueMessageId,
  Action,
  Payload
]
```

Example - StartTransaction Request:

```
[
  2,
  "unique-msg-123",
  "StartTransaction",
  {
    "connectorId": 1,
    "idTag": "VIRTUAL_RFID_USER_789abc",
    "meterStart": 0,
    "timestamp": "2025-01-17T10:30:00Z"
  }
]
```

Note: The idTag value comes from the user's `rfid_card_id` field in the database.

Message Type IDs: - 2 = CALL (request) - 3 = CALLRESULT (successful response) - 4 = CALLERROR (error response)

3.5 Supported OCPP Messages

3.5.1 BootNotification

Direction: Charge Point → Central System **Purpose:** Sent when charge point powers on or resets **Timing:** First message after WebSocket connection established

Payload Contains: - Charger vendor name and model - Serial number - Firmware version - ICCID (SIM card ID for cellular connectivity, if applicable)

Central System Response: - **Accepted:** Charge point is authorized to operate - **Pending:** Registration pending, retry later - **Rejected:** Charge point not authorized

System Actions: - Validates charger exists in database - Updates firmware version information - Sets heartbeat interval (default: 300 seconds) - Logs boot event for tracking uptime

---

3.5.2 Heartbeat

Direction: Charge Point → Central System **Purpose:** Keep-alive mechanism to maintain connection **Timing:** Sent at regular intervals (typically every 300 seconds)

Payload: Empty (just timestamp from system)

System Actions: - Updates last heartbeat timestamp - Resets connection timeout counter - If heartbeat not received within 90 seconds of expected time, marks connection as stale

Importance for Hardware: This is critical for connection health monitoring. Chargers must send heartbeats at configured intervals. Missing heartbeats trigger automatic cleanup to prevent ghost connections.

3.5.3 StatusNotification

Direction: Charge Point → Central System **Purpose:** Reports current status of a connector **Timing:** Sent whenever connector status changes

Key Parameters: - **connectorId:** Which outlet (1 for single-outlet chargers) - **status:** Current state (see status codes below) - **errorCode:** Fault code if status is “Faulted” - **timestamp:** When status changed

Status Codes: - `Available` - Ready for use, no vehicle connected - `Preparing` - Preparing to start charging - `Charging` - Active energy transfer in progress - `SuspendedEVSE` - Charging paused by charge point - `SuspendedEV` - Charging paused by vehicle - `Finishing` - Completing charging session - `Reserved` - Reserved for specific user - `Unavailable` - Offline or out of service - `Faulted` - Error condition

System Actions: - Updates connector status in database - Notifies connected web clients via real-time updates - If status changes to non-charging state during active transaction, may terminate transaction - Logs status history for analytics

---

3.5.4 StartTransaction

Direction: Charge Point → Central System **Purpose:** Reports that a charging transaction has started **Timing:** Sent in response to RemoteStartTransaction command (initiated via mobile app)

Key Parameters: - **connectorId:** Which outlet is being used - **idTag:** User's virtual RFID card ID (from user profile) - **meterStart:** Initial meter reading in Watt-hours (Wh) - **timestamp:** Transaction start time

Central System Response: - **transactionId:** Unique ID assigned to this session (used in all subsequent messages) - **idTagInfo:** Authorization status (Accepted, Blocked, Expired, Invalid)

System Actions: - Looks up user by matching `idTag` to `rfid_card_id` field in user database - Checks if user account is active (not blocked/deactivated) - Creates transaction record in database linked to user - Returns unique transaction ID to charge point - If user not found or inactive, rejects transaction

Important: The charge point will NOT start charging if idTagInfo status is not "Accepted"

Note: The `idTag` received here matches the virtual RFID card ID that was sent in the preceding RemoteStartTransaction command.

3.5.5 MeterValues

Direction: Charge Point → Central System **Purpose:** Reports electrical measurements during charging **Timing:** Sent periodically during transaction (typically every 60-300 seconds)

Key Parameters: - **connectorId:** Which outlet - **transactionId:** Associated transaction (if in active session) - **meterValue:** Array of sampled values with timestamps

Sampled Values Include: - **Energy.Active.Import.Register:** Total energy consumed (Wh) - **Voltage:** Supply voltage (V or mV) - **Current.Import:** Current draw (A or mA) - **Power.Active.Import:** Instantaneous power (W or kW) - **Temperature:** (optional) Connector or cable temperature - **SoC:** (optional) Vehicle battery State of Charge percentage

Example MeterValues Payload:

```json
{
  "connectorId": 1,
  "transactionId": 42,
  "meterValue": [
    {
      "timestamp": "2025-01-17T10:35:00Z",
      "sampledValue": [
        {
          "value": "5420",
          "measurand": "Energy.Active.Import.Register",
          "unit": "Wh"
        },
        {
          "value": "230.5",
          "measurand": "Voltage",
          "unit": "V"
        },
        {
          "value": "16.2",
          "measurand": "Current.Import",
          "unit": "A"
        },
        {
          "value": "3.73",
          "measurand": "Power.Active.Import",
          "unit": "kW"
        }
      ]
    }
  ]
}
```

System Actions: - Stores all meter values in database linked to transaction - Updates transaction with latest energy consumption - Provides data for real-time dashboard updates - Used for generating consumption charts and analytics

---

3.5.6 StopTransaction

Direction: Charge Point → Central System **Purpose:** Reports that a charging transaction has ended **Timing:** Sent when charging stops (user action, vehicle full, timeout, or remote stop)

Key Parameters: - **transactionId:** ID from StartTransaction response - **meterStop:** Final meter reading in Wh - **timestamp:** Transaction end time - **reason:** Why transaction stopped - **idTag:** (optional) User identifier used to stop (may differ from start)

Stop Reasons: - `Local` - Stopped by user at charge point - `Remote` - Stopped by central system command - `EVDisconnected` - Vehicle cable disconnected - `HardReset` / `SoftReset` - Charger reset - `PowerLoss` - Loss of grid power - `EmergencyStop` - Emergency stop button pressed - `Other` - Other reasons

System Actions: - Calculates total energy consumed: `(meterStop - meterStart)` - Converts to kWh for billing - Retrieves applicable tariff (price per kWh) - Calculates billing amount: `energy_kwh × tariff_rate` - Deducts amount from user wallet - Updates transaction status to COMPLETED - If billing fails (insufficient balance), marks as BILLING_FAILED and queues for retry - Logs complete transaction details

Important: This triggers automated billing. Accurate meter readings are critical for correct charges.

3.5.7 FirmwareStatusNotification

Direction: Charge Point → Central System **Purpose:** Reports progress of firmware update **Timing:** Sent at each stage of OTA update process

Status Values: - `Downloaded` - Firmware file downloaded successfully - `DownloadFailed` - Download error (network, corrupted file, etc.) - `Downloading` - Download in progress - `Idle` - No update in progress - `InstallationFailed` - Installation error - `Installing` - Installing new firmware - `Installed` - Installation complete (charger will reboot)

System Actions: - Updates firmware update record in database - Notifies administrators of update progress - On `Installed`, expects new BootNotification with updated firmware version - On failures, logs error details for troubleshooting

---

3.6 OCPP Sequence Diagrams

3.6.1 Initial Connection and Boot Sequence

```
Charge Point                     Central System
     │                                 │
     │  1. WebSocket Connection        │
     ├────────────────────────────────►│
     │                                 │ Validate charger exists
     │  2. Connection Accepted         │ Store WebSocket connection
     │◄────────────────────────────────┤ Add to Redis cache
     │                                 │ Start heartbeat monitor
     │                                 │
     │  3. BootNotification            │
     │    - vendor, model              │
     │    - serial number              │
     │    - firmware version           │
     ├────────────────────────────────►│
     │                                 │ Log boot event
     │  4. BootNotification Response   │ Update firmware version
     │    - status: Accepted           │
     │    - interval: 300 (heartbeat)  │
     │◄────────────────────────────────┤
     │                                 │
     │  5. StatusNotification          │
     │    - connectorId: 1             │
     │    - status: Available          │
     ├────────────────────────────────►│
     │                                 │ Update connector status
     │  6. Response: Accepted          │
     │◄────────────────────────────────┤
     │                                 │
     │  [Charger is now operational]   │
     │                                 │
```

3.6.2 Charging Session Lifecycle

```
Charge Point          Central System          User/Vehicle
     │                      │                       │
     │                      │  1. User initiates    │
     │                      │     via mobile app    │
     │                      │◄──────────────────────│
     │                      │ RemoteStart command   │
     │  2. RemoteStart      │                       │
     │◄─────────────────────│                       │
     │                      │                       │
     │  3. StartTransaction │                       │
     │    - connectorId: 1  │                       │
     │    - idTag: VIRTUAL_RFID_USER123             │
     │    - meterStart: 0   │                       │
     │─────────────────────►│                       │
     │                      │ Match idTag to user   │
     │                      │ (lookup rfid_card_id) │
     │                      │ Check user account    │
     │                      │ Create transaction    │
     │                      │                       │
     │  4. Response         │                       │
     │    - transactionId: 42                       │
     │    - idTagInfo: Accepted                     │
     │◄─────────────────────│                       │
     │                      │                       │
     │  5. StatusNotification                       │
     │    - status: Preparing                       │
     │─────────────────────►│                       │
     │                      │ Update status         │
     │                      │                       │
     │  6. StatusNotification                       │
     │    - status: Charging                        │
     │─────────────────────►│                       │
     │                      │ Update status         │
     │                      │ Mark transaction RUNNING
     │                      │                       │
     │  [Charging in progress]                      │
     │                      │                       │
     │  7. MeterValues (periodic)                   │
     │    - Energy: 1500 Wh │                       │
     │    - Voltage: 230 V  │                       │
     │    - Current: 16 A   │                       │
     │    - Power: 3.68 kW  │                       │
     │─────────────────────►│                       │
     │                      │ Store meter values    │
     │                      │ Update dashboard      │
     │                      │                       │
     │  [... more MeterValues every 60-300s ...]    │
     │                      │                       │
     │                      │  8. Vehicle full or   │
     │                      │     user stops via app│
     │                      │◄──────────────────────┤
     │                      │                       │
     │  9. StatusNotification                       │
     │    - status: Finishing                       │
     ├─────────────────────►│                       │
     │                      │                       │
     │  10. StopTransaction │                       │
     │    - transactionId: 42                       │
     │    - meterStop: 8500 Wh                      │
     │    - reason: EVDisconnected                  │
     ├─────────────────────►│                       │
     │                      │ Calculate energy:     │
     │                      │   8500 - 0 = 8500 Wh  │
     │                      │   = 8.5 kWh           │
     │                      │ Get tariff: ₹10/kWh   │
     │                      │ Billing: 8.5 × ₹10    │
     │                      │   = ₹85               │
     │                      │ Deduct from wallet    │
     │                      │ Update transaction    │
     │                      │                       │
     │  11. Response        │                       │
     │    - idTagInfo: Accepted                     │
     │◄─────────────────────┤                       │
     │                      │                       │
     │  12. StatusNotification                      │
     │    - status: Available                       │
     ├─────────────────────►│                       │
     │                      │ Charger ready for     │
     │                      │ next session          │
     │                      │                       │
```

3.6.3 Remote Start Command

```
Admin/User           Central System         Charge Point
     │                      │                       │
     │  1. API Request:     │                       │
     │     Start Charging   │                       │
     ├─────────────────────►│                       │
     │                      │ Validate permissions  │
     │                      │ Check charger online  │
     │                      │ (query Redis)         │
     │                      │                       │
     │                      │  2. RemoteStartTransaction
     │                      │    - connectorId: 1   │
     │                      │    - idTag: VIRTUAL_RFID_USER123
     │                      │      (from user.rfid_card_id)
     │                      ├──────────────────────►│
     │                      │                       │
     │                      │                       │ Validate
```

```
connector
     │                     │                      │ Check
availability
     │                     │                      │
     │                     │  3. Response         │
     │                     │    - status: Accepted │
     │                     │◄─────────────────────┤
     │                     │                      │
     │  4. API Response:   │                      │
     │     Command sent    │                      │
     │◄────────────────────┤                      │
     │                     │                      │
     │                     │  5. StartTransaction │
     │                     │    (see previous seq) │
     │                     │◄─────────────────────┤
     │                     │                      │
     │         [Charging session proceeds normally] │
     │                     │                      │
```

Note: If charger is offline or connector is unavailable, RemoteStart will be rejected.

3.6.4 Firmware Update Sequence

```
Admin              Central System         Charge Point
  │                      │                       │
  │  1. Upload firmware  │                       │
  │     binary file      │                       │
  ├─────────────────────►│                       │
  │                      │ Store file            │
  │                      │ Generate download URL │
  │                      │                       │
  │  2. Trigger update   │                       │
  │     for charger(s)   │                       │
  ├─────────────────────►│                       │
  │                      │ Create update record  │
  │                      │                       │
  │                      │  3. UpdateFirmware    │
  │                      │    - location: URL    │
  │                      │    - retrieveDate: timestamp
  │                      ├──────────────────────►│
  │                      │                       │
  │                      │  4. Response          │
  │                      │    - status: Accepted │
  │                      │◄──────────────────────┤
  │                      │                       │
  │                      │  5. FirmwareStatusNotification
  │                      │    - status: Downloading
  │                      │◄──────────────────────┤
  │                      │ Update record status  │
  │                      │                       │
  │                      │  [Download in progress]
  │                      │                       │
  │                      │  6. FirmwareStatusNotification
  │                      │    - status: Downloaded
  │                      │◄──────────────────────┤
  │                      │ Update record status  │
  │                      │                       │
  │                      │  7. FirmwareStatusNotification
  │                      │    - status: Installing
  │                      │◄──────────────────────┤
  │                      │ Update record status  │
  │                      │                       │
  │                      │  [Charger reboots]    │
  │                      │                       │
  │                      │  8. WebSocket disconnect
  │                      │◄──────────────────────┤
  │                      │ Clean up old connection
  │                      │                       │
  │                      │  9. WebSocket reconnect
  │                      │◄──────────────────────┤
  │                      │                       │
  │                      │  10. BootNotification │
  │                      │     - firmwareVersion: 2.0
  │                      │◄──────────────────────┤
  │                      │ Verify new version    │
  │                      │                       │
  │                      │  11. FirmwareStatusNotification
  │                      │     - status: Installed
  │                      │◄──────────────────────┤
  │                      │ Mark update complete  │
  │                      │                       │
  │  12. Notification:   │                       │
  │      Update success  │                       │
  │◄─────────────────────┤                       │
  │                      │                       │
```

3.7 Connection Management

Heartbeat Monitoring

The system implements robust connection health monitoring:

1. **Heartbeat Interval:** Set during BootNotification (default: 300 seconds)
2. **Timeout Detection:** If no heartbeat received within 90 seconds of expected time, connection considered stale
3. **Automatic Cleanup:** Stale connections automatically disconnected and removed from active pool
4. **Reconnection:** Chargers can reconnect immediately after disconnect

Benefits: - Prevents resource leaks from ghost connections - Quickly detects network failures or charger reboots - Maintains accurate online/offline status

Connection States

Each charger connection can be in one of these states:

- **CONNECTED:** Active WebSocket, receiving regular heartbeats
- **DISCONNECTED:** No active WebSocket connection
- **STALE:** Connected but heartbeat timeout exceeded (transitional state before cleanup)

Message Logging

All OCPP messages (both directions) are logged to database with: - Timestamp with millisecond precision - Message type (CALL, CALLRESULT, CALLERROR) - Action name (e.g., "StartTransaction") - Complete payload - Correlation ID linking requests to responses

Uses: - Debugging charger communication issues - Compliance auditing - Performance analysis - Dispute resolution for billing

---

4. Charging Session Workflow

4.1 Complete Session Data Flow

This section describes the end-to-end flow from a user initiating a charging session to final billing.

```
┌─────────────┐
│ USER ACTION │
└─────────────┘
       │
       │
       ▼
┌─────────────────────────────────────────────────────────┐
│ INITIATION                                              │
│                                                         │
│  Option A: Mobile App QR Scan + Remote Start           │
│  Option B: Mobile App Station Map + Remote Start       │
│  Option C: Admin Remote Start Command                  │
└─────────────────────────────────────────────────────────┘
       │
       │
       ▼
┌─────────────────────────────────────────────────────────┐
│ CENTRAL SYSTEM: Send RemoteStartTransaction             │
│ ─────────────────────────────────────────────────────   │
│ 1. User authenticated via Clerk JWT                    │
│ 2. Lookup user's virtual rfid_card_id from database    │
│    (e.g., "VIRTUAL_RFID_USER123")                       │
│ 3. Send OCPP RemoteStartTransaction command:                │
│    - connector_id: 1                                        │
│    - id_tag: user's rfid_card_id                            │
└─────────────────────────────────────────────────────────────┘
       │
       │
       ▼
┌─────────────────────────────────────────────────────────────┐
│ CHARGER: Receives RemoteStart, Initiates Charging           │
│ ─────────────────────────────────────────────────────────   │
│ - Accepts remote start command                              │
│ - Begins charging preparation                               │
└─────────────────────────────────────────────────────────────┘
       │
       │
       ▼
┌─────────────────────────────────────────────────────────────┐
│ CHARGER → CENTRAL SYSTEM: StartTransaction Message          │
│ ─────────────────────────────────────────────────────────   │
│ - Connector ID: 1                                           │
│ - idTag: VIRTUAL_RFID_USER123 (from RemoteStart)            │
│ - Initial Meter: 0 Wh                                       │
│ - Timestamp: 2025-01-17T10:00:00Z                           │
└─────────────────────────────────────────────────────────────┘
       │
       │
       ▼
┌─────────────────────────────────────────────────────────────┐
│ CENTRAL SYSTEM: User Validation & Transaction Creation      │
│ ─────────────────────────────────────────────────────────   │
│ 1. Match idTag to user by rfid_card_id field               │
│ 2. Check user account status (active/blocked)               │
│ 3. Verify user exists and is authorized                    │
│ 4. Create Transaction record in database:                   │
│    - Status: RUNNING                                        │
│    - User ID: [from rfid_card_id match]                     │
│    - Charger ID: [from connection]                          │
│    - Start Time: [from message]                             │
│    - Start Meter: 0 Wh                                      │
│ 5. Generate unique Transaction ID: 42                       │
└─────────────────────────────────────────────────────────────┘
       │
       │
       ▼
┌─────────────────────────────────────────────────────────────┐
│ CENTRAL SYSTEM → CHARGER: Response                          │
│ ─────────────────────────────────────────────────────────   │
│ - Transaction ID: 42                                        │
│ - Authorization: ACCEPTED                                   │
└─────────────────────────────────────────────────────────────┘
       │
       │
       ▼
┌─────────────────────────────────────────────────────────────┐
│ CHARGER: Begin Energy Transfer                              │
│ ─────────────────────────────────────────────────────────   │
│ 1. StatusNotification: Preparing                          │
│ 2. Close contactors, enable power to vehicle              │
│ 3. StatusNotification: Charging                           │
└───────────────────────────────────────────────────────────┘
       │
       │
       ▼
┌───────────────────────────────────────────────────────────┐
│ CENTRAL SYSTEM: Update Transaction                        │
│ ─────────────────────────────────────────────────────     │
│ - Status: RUNNING (charging active)                      │
│ - Charger Status: Charging                                │
└───────────────────────────────────────────────────────────┘
       │
       │
       ▼
┌───────────────────────────────────────────────────────────┐
│ ACTIVE CHARGING PHASE                                     │
│ ─────────────────────────────────────────────────────     │
│ Every 60-300 seconds:                                     │
│                                                           │
│ CHARGER → MeterValues Message                             │
│ ┌──────────────────────────────────────────────┐          │
│ │ Timestamp: 10:01:00                          │          │
│ │ - Energy: 1,200 Wh (1.2 kWh)                │          │
│ │ - Voltage: 230.5 V                           │          │
│ │ - Current: 16.2 A                            │          │
│ │ - Power: 3.73 kW                             │          │
│ └──────────────────────────────────────────────┘          │
│                                                           │
│ CENTRAL SYSTEM: Store Meter Data                          │
│ - Insert into MeterValue table                           │
│ - Update transaction energy total                        │
│ - Push real-time update to web dashboard                 │
│                                                           │
│ [Repeat for duration of charging session]                │
└───────────────────────────────────────────────────────────┘
       │
       │
       ▼
┌───────────────────────────────────────────────────────────┐
│ TERMINATION                                               │
│                                                           │
│ Trigger: Vehicle battery full, user stops, timeout,      │
│          remote stop command, or error condition          │
└───────────────────────────────────────────────────────────┘
       │
       │
       ▼
┌───────────────────────────────────────────────────────────┐
│ CHARGER: Stop Charging Process                            │
│ ─────────────────────────────────────────────────────     │
│ 1. StatusNotification: Finishing                          │
│ 2. Open contactors, disable power                         │
│ 3. StopTransaction Message:                               │
│    - Transaction ID: 42                                        │
│    - Final Meter: 8,500 Wh                                     │
│    - Reason: EVDisconnected                                    │
│    - Timestamp: 2025-01-17T12:15:00Z                           │
└────────────────────────────────────────────────────────────────┘
       │
       │
       ▼
┌────────────────────────────────────────────────────────────────┐
│ CENTRAL SYSTEM: Billing Calculation                            │
│ ───────────────────────────────────────────────────────────    │
│ 1. Calculate Energy Consumed:                                  │
│    meterStop - meterStart = 8,500 - 0 = 8,500 Wh              │
│    Convert to kWh: 8,500 ÷ 1,000 = 8.5 kWh                   │
│                                                                │
│ 2. Retrieve Tariff:                                            │
│    Check for charger-specific tariff from database            │
│    Fallback to global tariff if none                          │
│    Rate: ₹10 per kWh (example)                                 │
│                                                                │
│ 3. Calculate Billing Amount:                                   │
│    8.5 kWh × ₹10/kWh = ₹85                                     │
│                                                                │
│ 4. Wallet Deduction (Atomic Transaction):                     │
│    a. Begin database transaction                              │
│    b. Lock user wallet record                                 │
│    c. Current balance: ₹500.00                                 │
│    d. New balance: ₹500.00 - ₹85 = ₹415                       │
│    e. Update wallet balance                                    │
│    f. Create WalletTransaction record:                         │
│       - Type: CHARGE_DEDUCT                                   │
│       - Amount: -₹85                                           │
│       - Transaction ID: 42                                     │
│    g. Update transaction record:                               │
│       - Stop Time: 12:15:00                                    │
│       - Energy: 8.5 kWh                                        │
│       - Billing Amount: ₹85                                    │
│       - Status: COMPLETED                                      │
│    h. Commit database transaction                              │
│                                                                │
│ 5. If Billing Fails (e.g., insufficient balance):             │
│    - Mark transaction: BILLING_FAILED                          │
│    - Add to retry queue                                        │
│    - Retry every 30 minutes for up to 24 hours                │
└────────────────────────────────────────────────────────────────┘
       │
       │
       ▼
┌────────────────────────────────────────────────────────────────┐
│ CENTRAL SYSTEM → CHARGER: StopTransaction Response             │
│ ───────────────────────────────────────────────────────────    │
│ - Authorization: ACCEPTED                                      │
└────────────────────────────────────────────────────────────────┘
       │
```

```
         │
         ▼
┌────────────────────────────────────────────────────┐
│ CHARGER: Return to Ready State                     │
│ ────────────────────────────────────────────────   │
│ 1. StatusNotification: Available                   │
│ 2. Ready for next charging session                 │
└────────────────────────────────────────────────────┘
         │
         ▼
┌─────────────┐
│  COMPLETE   │
└─────────────┘
```

4.2 State Transitions

Charger Status States

```
                   ┌─────────────┐
                   │  AVAILABLE  │ ◄─── Initial state
                   └──────┬──────┘
                          │
             Remote start via mobile app
                          │
                          ▼
                   ┌─────────────┐
                   │  PREPARING  │
                   └──────┬──────┘
                          │
             Contactors closed, power enabled
                          │
                          ▼
                   ┌─────────────┐
             ┌────►│  CHARGING   │◄────┐
             │     └──────┬──────┘     │
             │            │            │
Vehicle pauses│           │            │ Vehicle resumes
             │            │            │
             │            ▼            │
        ┌────┴────────┐ Session ends ┌─┴───────────┐
        │ SUSPENDED_EV│              │SUSPENDED_EVSE│
        └─────────────┘              └─────────────┘
                          │
             User stops or vehicle full
                          │
                          ▼
                   ┌─────────────┐
                   │  FINISHING  │
                   └──────┬──────┘
                          │
```

```
                        Contactors opened, session complete
                                     │
                                     ▼
                              ┌──────────────┐
                              │  AVAILABLE   │ ◄─── Ready for next
                              └──────────────┘
```

Transaction Status States

```
User initiates
     │
     ▼
┌──────────┐  Charger confirms       ┌──────────┐
│ STARTED  ├────────────────────────►│ RUNNING  │
└──────────┘                         └────┬─────┘
                                          │
     ┌────────────────────────────────────┼─────────────────┐
     │                                    │                 │
     │ Normal stop                        │ Error           │ Admin
cancels
     ▼                                    ▼                 ▼
┌───────────┐                        ┌──────────┐    ┌────────────┐
│  STOPPED  │                        │  FAILED  │    │ CANCELLED  │
└────┬──────┘                        └──────────┘    └────────────┘
     │
     │ Billing process
     │
     ├─── Success ───────────►  ┌────────────┐
     │                          │ COMPLETED  │
     │                          └────────────┘
     │
     └─── Failure ───────────►  ┌───────────────┐
                                │BILLING_FAILED │
                                └───┬───────────┘
                                    │
                        Retry every 30 minutes
                                    │
                                    ├─── Success ──► COMPLETED
                                    │
                                    └─── Max retries ──► (Manual
intervention)
```

4.3 Remote Start/Stop Workflow

Remote Start Initiated by User via Mobile App

```
1. User opens mobile app, scans QR code on charger
2. App displays charger details and "Start Charging" button
3. User taps button
4. App sends API request to Central System with user's JWT token
```

```
5. Central System:
   a. Validates user authentication
   b. Checks if charger is online (Redis lookup)
   c. Checks if connector is Available
   d. Creates pending transaction record
   e. Sends RemoteStartTransaction command to charger via WebSocket
6. Charger responds: Accepted or Rejected
7. If Accepted:
   a. Charger initiates charging process
   b. Sends StartTransaction message (normal flow continues)
8. App displays "Charging started" confirmation
```

Remote Stop

```
1. User/Admin triggers stop via web/mobile interface
2. Central System:
   a. Validates permissions
   b. Checks charger online and transaction active
   c. Sends RemoteStopTransaction command with transaction ID
3. Charger responds: Accepted
4. Charger stops energy transfer
5. Charger sends StopTransaction message (normal flow continues)
```

4.4 Error Handling Scenarios

Scenario 1: Network Disruption During Charging

Problem: WebSocket connection lost while vehicle is charging

Charger Behavior: - Continues charging (safety: doesn't interrupt vehicle) - Stores meter values locally - Attempts to reconnect to Central System

System Behavior: - Detects missed heartbeat - Marks connection as disconnected - Transaction remains in RUNNING state

Resolution: - When charger reconnects: - Sends BootNotification - May send buffered MeterValues - Continues transaction until vehicle stops - Sends StopTransaction with final meter reading - System processes billing based on final meter values

Scenario 2: User Has Insufficient Wallet Balance

Problem: User starts charging, but wallet balance becomes insufficient for final billing

System Behavior: - Charging proceeds normally (don't interrupt user) - At StopTransaction, billing calculation fails - Transaction marked: BILLING_FAILED - Background retry service attempts billing every 30 minutes - Continues retrying for up to 24 hours

Resolution Options: - User adds funds to wallet → Next retry succeeds - Admin manually adjusts transaction or writes off debt - Account flagged for follow-up if retries exhausted

Scenario 3: Charger Fault During Session

Problem: Charger hardware fault (e.g., ground fault, overvoltage)

Charger Behavior: - Opens contactors immediately (safety) - Sends StatusNotification: Faulted (with error code) - Sends StopTransaction with reason: PowerLoss or Other

System Behavior: - Updates charger status to Faulted - Processes transaction normally (bills for energy consumed up to fault) - Alerts administrators of fault condition - Charger unavailable for new sessions until fault cleared

Administrator Action: - Reviews OCPP logs for error details - Dispatches maintenance if hardware issue - Once repaired, charger sends StatusNotification: Available

---

5. Electrical Concepts & Metering

5.1 Connector Types Supported

The system tracks different connector types to help users find compatible chargers:

AC Connectors (Alternating Current)

Type 2 (Mennekes / IEC 62196-2): - **Usage:** European standard for AC charging - **Voltage:** 230V (single-phase) or 400V (three-phase) - **Current:** Up to 32A (single-phase) or 63A (three-phase) - **Power:** 3.7 kW to 43 kW - **Common Power Levels:** - 7.4 kW (230V, 32A, single-phase) - 11 kW (400V, 16A, three-phase) - 22 kW (400V, 32A, three-phase)

Type 1 (SAE J1772): - **Usage:** North American and Japanese standard - **Voltage:** 120V or 240V (single-phase only) - **Current:** Up to 80A - **Power:** Up to 19.2 kW

DC Connectors (Direct Current - Fast Charging)

CCS (Combined Charging System): - **CCS Type 2 (Europe):** Type 2 connector with additional DC pins - **CCS Type 1 (North America):** Type 1 connector with additional DC pins - **Voltage:** Up to 920V DC - **Current:** Up to 500A - **Power:** Up to 350 kW (though most installations: 50-150 kW)

CHAdeMO: - **Usage:** Japanese standard, common in older EVs - **Voltage:** Up to 500V DC - **Current:** Up to 400A - **Power:** Up to 200 kW (though most: 50 kW)

GB/T: - **Usage:** Chinese standard - **Voltage:** Up to 750V DC - **Current:** Up to 250A - **Power:** Up to 237.5 kW

5.2 Electrical Parameters Monitored

The system collects and stores real-time electrical measurements during charging sessions.

Energy (kWh)

Measurand: `Energy.Active.Import.Register`

Description: Total electrical energy delivered to the vehicle

Units: - Reported by charger: Watt-hours (Wh) - Stored in database: Kilowatt-hours (kWh) - converted by dividing by 1,000

Measurement: - Cumulative register (starts at 0 or previous value, increments throughout session) - Final value at StopTransaction used for billing

Example: - Start: 0 Wh - After 1 hour: 7,400 Wh (7.4 kWh) - After 2 hours: 14,800 Wh (14.8 kWh) - Final: 20,000 Wh (20 kWh) - **Billed energy:** 20 kWh

Importance: This is the primary value for billing calculation

---

Voltage (V)

Measurand: `Voltage`

Description: Electrical potential difference supplied to vehicle

Units: - Reported: Volts (V) or millivolts (mV) - Stored: Volts (V)

Typical Values: - **AC Single-Phase:** 220-240V (Europe), 110-120V (North America) - **AC Three-Phase:** 380-415V line-to-line - **DC Fast Charging:** 200-920V (varies by vehicle and charger capability)

Monitoring Purpose: - Verify grid supply quality - Detect undervoltage or overvoltage conditions - Troubleshoot charging issues

Example: 230.5 V (typical European household supply)

Current (A)

Measurand: `Current.Import`

Description: Electrical current flowing to vehicle battery

Units: - Reported: Amperes (A) or milliamperes (mA) - Stored: Amperes (A)

Typical Values: - **AC Slow Charging:** 6-16A (home charging) - **AC Fast Charging:** 16-32A (public chargers) - **DC Fast Charging:** 50-500A

Relationship to Power: - **AC (single-phase):** Power (kW) = Voltage (V) × Current (A) × Power Factor ÷ 1,000 - **AC (three-phase):** Power (kW) = √3 × Voltage (V) × Current (A) × Power Factor ÷ 1,000 - **DC:** Power (kW) = Voltage (V) × Current (A) ÷ 1,000

Monitoring Purpose: - Verify charger operating within rated capacity - Detect cable overheating risks - Monitor charging speed

Example: 16.2 A at 230V single-phase ≈ 3.7 kW (assuming power factor ≈ 1)

Power (kW)

Measurand: `Power.Active.Import`

Description: Instantaneous electrical power delivered to vehicle

Units: - Reported: Watts (W) or Kilowatts (kW) - Stored: Kilowatts (kW)

Calculation: Power = Voltage × Current (× √3 for three-phase) (× Power Factor)

Typical Values: - **Level 1 (Slow):** 1.4-1.9 kW - **Level 2 (Fast AC):** 3.7-22 kW - **DC Fast Charging:** 50-350 kW

Relationship to Energy: - Energy (kWh) = Power (kW) × Time (hours) - Example: 7.4 kW for 2 hours = 14.8 kWh

Monitoring Purpose: - Real-time charging speed indication - Detect power fluctuations - Verify charging performance

Example: 3.73 kW (typical 230V, 16A single-phase charging)

5.3 Meter Value Collection

Sampling Frequency

- **Configuration:** Determined by charge point configuration (typically 60-300 seconds)
- **System Default:** Recommends 60-120 seconds for granular data
- **Trade-off:**
  - Higher frequency = Better resolution, more storage required
  - Lower frequency = Less data, may miss transient events

Data Storage

All meter values stored in `MeterValue` table with: - Timestamp of measurement - Link to transaction ID - All measured parameters (energy, voltage, current, power) - Original units and converted standard units

Data Visualization

Users can view: - Real-time power graph during active charging - Historical energy consumption charts - Voltage and current trends for troubleshooting - Comparison across multiple charging sessions

5.4 Tariff and Billing

Tariff Structure

Rate Type: Energy-based (per kilowatt-hour)

Database Storage: - Tariffs are stored in the `Tariff` database table - Each tariff has a `rate_per_kwh` field (Decimal type for precision) - Tariffs can be linked to specific chargers or set as global default - All values stored in **INR (Indian Rupees)**

Configuration: - **Charger-Specific Tariff:** Different rates for different chargers - Example: Premium DC fast chargers at ₹15/kWh - Example: Standard AC chargers at ₹10/kWh - **Global Fallback Tariff:** Default rate if no charger-specific rate defined

Precision: Decimal values to 2 decimal places (e.g., ₹10.50/kWh)

Billing Calculation

Formula:

```
Billing Amount = Energy Consumed (kWh) × Tariff Rate (₹/kWh)
```

Example: - Energy: 18.5 kWh - Tariff: ₹12/kWh (from database) - **Billing:** 18.5 × 12 = ₹222

Rounding: Results rounded to 2 decimal places (paise)

Currency: All billing calculations and wallet balances use **INR (₹)**

Energy Calculation from Meters

Method: Difference between start and stop meter readings

```
Energy (Wh) = meterStop - meterStart
Energy (kWh) = Energy (Wh) ÷ 1,000
```

Example: - meterStart: 250 Wh (charger's cumulative meter at session start) - meterStop: 18,750 Wh (charger's cumulative meter at session end) - Energy: 18,750 - 250 = 18,500 Wh = 18.5 kWh

Note: Some chargers reset meter to 0 at start of each transaction; others use cumulative meters. System handles both.

Billing Retry Mechanism

Trigger: Billing fails (e.g., insufficient wallet balance, database error)

Process: 1. Transaction marked: `BILLING_FAILED` 2. Background service runs every 30 minutes 3. Retries billing for all failed transactions 4. On success: Updates status to `COMPLETED` 5. On continued failure: Retries for up to 24 hours 6. After 24 hours: Flags for manual administrator review

Benefits: - Handles temporary issues (user adds funds later) - Reduces manual intervention required - Maintains accurate financial records

---

6. Charger Management & Control

6.1 Real-Time Monitoring Capabilities

Administrators and users can monitor:

Connection Status

- **Online:** Active WebSocket connection, receiving heartbeats
- **Offline:** No active connection
- **Last Heartbeat:** Timestamp of last heartbeat received (updated every 300s for online chargers)

Charger Status

- Current state of each connector (Available, Charging, Faulted, etc.)
- Real-time status updates via WebSocket push notifications to web interface

Active Sessions

- Which users are currently charging
- Real-time power and energy consumption
- Session duration
- Estimated cost based on current consumption and tariff

Historical Data

- All past transactions with complete details
- Energy consumption trends
- Usage patterns (busiest times, average session duration)
- Fault history and uptime statistics

6.2 Remote Control Commands

Administrators can remotely control chargers via the web interface.

Change Availability

Purpose: Make charger operative (available for use) or inoperative (out of service)

Use Cases: - Maintenance: Mark inoperative before servicing - Emergency: Disable faulted charger - Scheduling: Reserve charger for specific events

OCPP Command: `ChangeAvailability`

Parameters: - connectorId: Which outlet (0 = entire charger, 1+ = specific connector) - type: `Operative` or `Inoperative`

Process:

```
Admin clicks "Make Inoperative" in dashboard
       ↓
System sends ChangeAvailability command
       ↓
Charger responds: Accepted or Rejected
       ↓
Charger sends StatusNotification: Unavailable
       ↓
Dashboard updates to show "Offline"
```

Important: Inoperative chargers will reject new transactions but won't interrupt active sessions.

Remote Start Transaction

Purpose: Initiate charging session remotely via mobile app

Use Cases: - User initiates via mobile app (QR scan or station map) - Admin starts charging for testing - Fleet management: Start charging for specific vehicle

OCPP Command: `RemoteStartTransaction`

Parameters: - connectorId: Which outlet to use - idTag: User's virtual RFID card ID (from user profile `rfid_card_id` field)

Prerequisites: - User must have a valid `rfid_card_id` assigned in their profile - System will return error if user's `rfid_card_id` is null/empty

Process:

```
User scans QR code on charger via mobile app
       ↓
App sends start request to API
       ↓
System validates user and checks charger online
       ↓
System sends RemoteStartTransaction command
       ↓
Charger accepts and begins charging process
       ↓
Charger sends StartTransaction (normal flow)
       ↓
App displays "Charging started"
```

Prerequisites: - Charger must be online (connected WebSocket) - Connector must be Available - User must have valid account

---

Remote Stop Transaction

Purpose: Terminate active charging session remotely

Use Cases: - User stops via mobile app - Admin emergency stop - Timeout enforcement (e.g., overstay fees)

OCPP Command: `RemoteStopTransaction`

Parameters: - transactionId: ID of active session to stop

Process:

```
Admin clicks "Stop Charging" in dashboard
       ↓
System sends RemoteStopTransaction command
       ↓
Charger stops energy transfer
       ↓
Charger sends StopTransaction (normal flow with billing)
       ↓
Dashboard shows "Session ended"
```

Important: This will interrupt vehicle charging. Use with caution.

6.3 Firmware OTA Updates

The system supports over-the-air firmware updates for connected chargers, enabling centralized maintenance.

Update Process Overview

```
┌──────────────────────────────────────────────────────────────┐
│ 1. PREPARATION                                               │
│    - Admin uploads firmware binary file (.bin, .hex, etc.)   │
│    - System stores file and generates download URL           │
│    - URL format: https://server/static/firmware/{file_id}    │
└────────────────────────────┬─────────────────────────────────┘
                             │
┌────────────────────────────▼─────────────────────────────────┐
│ 2. INITIATION                                                │
│    - Admin selects target charger(s)                         │
│    - System creates FirmwareUpdate record for each           │
│    - Sends UpdateFirmware command via OCPP                   │
│    - Command includes: download URL, retrieve date/time      │
└────────────────────────────┬─────────────────────────────────┘
                             │
┌────────────────────────────▼─────────────────────────────────┐
│ 3. DOWNLOAD                                                  │
│    - Charger downloads firmware file from URL                │
│    - Sends FirmwareStatusNotification: Downloading           │
│    - System updates status in database                       │
│    - On completion: FirmwareStatusNotification: Downloaded   │
└────────────────────────────┬─────────────────────────────────┘
                             │
```

```
┌──────────────────────────────────▼───────────────────────────┐
│ 4. INSTALLATION                                              │
│    - Charger validates firmware file (checksum, signature)   │
│    - Sends FirmwareStatusNotification: Installing            │
│    - Installs new firmware to flash memory                   │
│    - Sends FirmwareStatusNotification: Installed             │
└──────────────────────────────────┬───────────────────────────┘
                                   │
┌──────────────────────────────────▼───────────────────────────┐
│ 5. REBOOT                                                    │
│    - Charger automatically reboots to load new firmware      │
│    - WebSocket connection drops                              │
│    - System cleans up old connection                         │
└──────────────────────────────────┬───────────────────────────┘
                                   │
┌──────────────────────────────────▼───────────────────────────┐
│ 6. VERIFICATION                                              │
│    - Charger reconnects with new WebSocket                   │
│    - Sends BootNotification with NEW firmware version        │
│    - System verifies version matches expected                │
│    - Marks FirmwareUpdate record: SUCCESS                    │
└──────────────────────────────────────────────────────────────┘
```

Update Status Tracking

The system tracks each update through these states:

Status	Description
PENDING	Update initiated, command sent to charger
DOWNLOADING	Charger actively downloading firmware file
DOWNLOADED	Download complete, ready for installation
INSTALLING	Charger installing new firmware
INSTALLED	Installation complete (charger will reboot)
DOWNLOAD_FAILED	Download error (network, file corruption, etc.)
INSTALLATION_FAILED	Installation error (validation failed, flash write error, etc.)

Bulk Updates

Feature: Update multiple chargers simultaneously

Process: 1. Admin selects multiple chargers or all chargers at a station 2. System creates individual update records for each 3. Sends UpdateFirmware command to each online charger 4. Dashboard shows progress for each charger individually 5. Failed updates can be retried individually

Safety: Staged rollout recommended (update test chargers first, then production)

Failure Handling

Download Failures: - Common causes: Network timeout, file server unavailable, insufficient storage - Solution: Admin can retry update (charger will attempt download again)

Installation Failures: - Common causes: Corrupted file, incompatible firmware, hardware mismatch - Solution: Charger should remain on previous firmware (safe fallback) - Admin reviews error logs and obtains correct firmware version

Rollback: - Not automatically supported by OCPP 1.6 - Requires manual upload and installation of previous firmware version

---

6.4 OCPP Message Logs

Purpose

Complete audit trail of all communications between chargers and central system.

Use Cases: - **Debugging:** Diagnose communication issues or unexpected behavior - **Compliance:** Regulatory requirements for transaction records - **Dispute Resolution:** Verify billing accuracy and transaction details - **Performance Analysis:** Identify slow responses or timeout patterns

Log Contents

Each log entry contains: - **Timestamp:** Precise time (milliseconds) when message sent/received - **Charger ID:** Which charge point - **Direction:** `SENT` (Central System → Charger) or `RECEIVED` (Charger → Central System) - **Message Type:** CALL (request), CALLRESULT (success response), or CALLERROR (error) - **Action:** OCPP action name (e.g., “StartTransaction”) - **Payload:** Complete JSON message content - **Correlation ID:** Links request to response for request-response pairs

Querying Logs

Administrators can filter logs by: - Date/time range - Specific charger - Action type (e.g., show all StartTransaction messages) - Message type (e.g., show all errors)

Example Use: “Show all messages for Charger CP001 on January 15, 2025 between 10:00-11:00”

Data Retention

- Logs retained indefinitely (or per company policy)
- Large databases may archive old logs to separate storage
- Critical for financial auditing (transaction logs must match billing records)

---

7. System Capabilities

7.1 Multi-Station Management

Geographic Organization

Charging Station: - Physical location with one or more chargers - Address and geographic coordinates (latitude, longitude) - Displayed on interactive map in user interface

Chargers: - Individual charge points at a station - Each has unique OCPP identifier - One station can have multiple chargers (e.g., parking lot with 10 charge points)

Connectors: - Physical outlets on a charger - Most chargers have 1 connector (single outlet) - Some have 2+ connectors (dual outlets, or AC + DC)

Hierarchy:

```
Station (Location)
  └── Charger 1 (OCPP ID: CP001)
        └── Connector 1 (Type 2, 22 kW)
  └── Charger 2 (OCPP ID: CP002)
        ├── Connector 1 (Type 2, 22 kW AC)
        └── Connector 2 (CCS, 50 kW DC)
  └── Charger 3 (OCPP ID: CP003)
        └── Connector 1 (CHAdeMO, 50 kW DC)
```

Station Management Features

- **Map View:** All stations displayed on interactive map
- **Filter by Connector:** Users can find stations with specific connector types
- **Availability:** Real-time status of all chargers at each station
- **Directions:** Integration with mapping services for navigation

7.2 Role-Based Access Control (RBAC)

The system enforces different permission levels for users.

User Roles

ADMIN: - Full system access - Can manage stations, chargers, users - Can view all transactions (all users) - Can execute remote commands - Can upload firmware and trigger updates - Can view OCPP message logs - Can configure tariffs (set ₹/kWh rates per charger or globally in database)

USER: - Limited to personal data - Can view public station map - Can start/stop own charging sessions - Can view own transaction history - Can manage own wallet (view balance, add funds) - Cannot access other users’ data - Cannot manage infrastructure

Authentication & Authorization

Authentication (Who are you?): - Handled by Clerk platform - Users sign in with email/password, Google, or other providers - System issues JWT (JSON Web Token) with user identity

Authorization (What can you do?): - Role stored in user database record - Every API request verified: 1. Validate JWT token 2. Lookup user in database 3. Check user role 4. Allow or deny based on role and requested operation

Example:

```
Request: GET /api/users
From: JWT with user ID = user123
    ↓
System checks: Is user123 an ADMIN?
    ├─ Yes → Return list of all users
    └─ No  → Return 403 Forbidden error
```

Virtual RFID Card System

Overview: The system uses a virtual RFID card approach for OCPP protocol compatibility. Users do NOT physically swipe RFID cards at chargers. Instead, each user has a unique RFID card ID stored in their profile, which is used internally for OCPP communication.

How It Works: 1. **User Profile Setup:** - Each user account has an `rfid_card_id` field (unique identifier) - This ID is assigned when the user account is created - Users never interact with this ID directly

2. **Charging Session Initiation:**
   - User opens mobile app and scans QR code on charger OR selects charger from map
   - User taps “Start Charging” in the app
   - App sends authenticated API request to central system (using JWT token)
3. **OCPP Communication:**
   - Central system looks up user’s stored `rfid_card_id` from their profile
   - System sends OCPP `RemoteStartTransaction` command to charger
   - Command includes the user’s `rfid_card_id` as the `idTag` parameter
   - Charger responds and initiates charging
   - Charger sends `StartTransaction` message with the same `idTag`
   - Central system matches `idTag` back to user by looking up `rfid_card_id`
4. **Why This Approach:**
   - **OCPP Compliance:** The OCPP 1.6 protocol requires an `idTag` for user identification
   - **No Hardware Readers:** Eliminates need for physical RFID card readers at chargers
   - **Security:** Users authenticate via mobile app (JWT), not cards that can be lost/stolen
   - **Flexibility:** All charging is remotely initiated, giving users control from their phone

Important Notes: - Chargers do NOT have physical RFID readers installed - All transactions are initiated remotely via mobile app - The `rfid_card_id` is purely for internal OCPP protocol compliance - Users authenticate via Clerk (email/password, Google, etc.), not RFID cards

Database Field: - Model: `User` - Field: `rfid_card_id` (unique string, max 255 characters) - Example value: `"USER_123abc"` or `"VIRTUAL_RFID_456"`

7.3 Wallet System

Wallet Balance

- Each user has one wallet account
- Balance stored as decimal (precise to 2 decimal places for currency)
- **Currency:** All wallet balances in **INR (Indian Rupees ₹)**
- Negative balances allowed (users can charge “on credit”)

Wallet Operations

Top-Up (Add Funds): 1. User selects amount to add (e.g., ₹500) 2. System creates Razorpay payment order (in INR) 3. User completes payment via credit/debit card, UPI, or net banking 4. Razorpay sends webhook to confirm payment 5. System verifies payment signature 6. System adds funds to wallet 7. Creates WalletTransaction record (type: TOP_UP, amount: +₹500)

Deduction (Charging Session): 1. Automatically triggered when transaction completes 2. System calculates billing amount (energy × tariff from database) 3. Deducts from wallet balance 4. Creates WalletTransaction record (type: CHARGE_DEDUCT, amount: -₹85)

Transaction History

Users can view complete wallet transaction history: - Date and time - Transaction type (TOP_UP or CHARGE_DEDUCT) - Amount (+/-) - Running balance after transaction - Associated charging transaction (if CHARGE_DEDUCT)

7.4 Transaction Tracking & Analytics

Transaction Data

For each charging session, system stores: - User who charged - Charger and connector used - Start and stop timestamps - Duration (calculated) - Energy consumed (kWh) - All meter values (voltage, current, power throughout session) - Billing amount - Wallet transaction reference - Stop reason - Status (completed, failed, etc.)

Analytics & Reports

For Users: - Personal charging history - Total energy consumed (lifetime and per period) - Total spending on charging (in INR ₹) - Energy consumption charts (kWh per session, power graph during session) - Average cost per kWh (₹/kWh)

For Administrators: - All transactions across all users - Revenue reports (total income in ₹, per station, per charger) - Usage statistics (sessions per day, peak hours, average duration) - Charger utilization (percentage of time each charger is in use) - Energy delivered (total kWh, per station, per charger) - Fault frequency and downtime tracking - Tariff management (configure ₹/kWh rates per charger or globally)

Data Export

- Transaction data can be exported to CSV format
- Useful for external accounting systems
- Includes all relevant fields for financial reconciliation

---

8. Technical Specifications

8.1 Network Requirements

For Charge Points (Hardware)

Internet Connection: - Required: Stable internet connection (wired Ethernet recommended, cellular acceptable) - Bandwidth: Minimal (~1-10 kbps average for OCPP messages) - Latency: Low latency preferred (<500ms) for responsive remote commands

WebSocket Connection: - Protocol: WebSocket (ws://) or WebSocket Secure (wss://) - Sub-Protocol: `ocpp1.6` - Endpoint: `ws://[server_domain]/ocpp/{charge_point_id}` - Example: `ws://ocpp.example.com/ocpp/CP001`

Firewall Configuration: - Outbound TCP port 80 (ws://) or 443 (wss://) must be open - Persistent connection (not HTTP request-response) - No inbound ports required (charger initiates connection)

Network Address Translation (NAT): - Chargers can operate behind NAT/firewall - Central system does not need to reach charger directly - WebSocket connection initiated by charger (outbound)

For Users

Web Interface Access: - HTTPS connection to frontend (port 443) - Modern web browser required (Chrome, Firefox, Safari, Edge) - JavaScript enabled

Mobile Access: - Responsive web interface (no native app required) - Requires cellular data or Wi-Fi - Camera access for QR code scanning

8.2 Technology Stack Summary

Backend

Component	Technology	Purpose
Application Framework	FastAPI (Python)	Asynchronous web server and REST API
OCPP Implementation	python-ocpp library	OCPP 1.6 protocol handling
WebSocket Server	Uvicorn (ASGI)	WebSocket connection management
Database ORM	Tortoise ORM	Asynchronous database operations
Database	PostgreSQL 15+	Transactional data storage
Cache	Redis 6+	Connection state and session data
Authentication	Clerk	User management and JWT verification
Payments	Razorpay	Payment processing

Frontend

Component	Technology	Purpose
Framework	Next.js 15	React-based web application
Language	TypeScript	Type-safe JavaScript
Styling	Tailwind CSS	Responsive UI styling
Components	Shadcn/ui	Pre-built UI components
State Management	TanStack Query	Server state caching and synchronization
Maps	React Leaflet	Interactive station maps
QR Scanning	ZXing	QR code reading

8.3 Supported Charger Specifications

OCPP Compliance

Required: - OCPP 1.6 specification compliance - JSON message format (not SOAP) - WebSocket transport

Tested Chargers: - This system has been tested with OCPP 1.6 simulators and select hardware - Compatible with any vendor implementing OCPP 1.6 correctly

Supported Features: - Core Profile (required messages) - Remote Control (RemoteStart, RemoteStop, ChangeAvailability) - Firmware Management (UpdateFirmware)

Optional Features (if charger supports): - Reservation (not yet implemented in central system) - Smart Charging (not yet implemented)

Hardware Compatibility

Connector Types: Any (system tracks but doesn't restrict)

Power Levels: - AC: 1 kW to 43 kW - DC: 1 kW to 350 kW (no technical limit)

Communication: - WebSocket only (OCPP 1.6 over WebSocket) - No SOAP support (OCPP 1.5 or earlier)

Charger Configuration: - Must be configured with central system URL - Must use unique charge point identifier (no duplicates) - Heartbeat interval configurable (recommended 300s)

8.4 Data Retention & Logging

Database Storage

Transactional Data: - Users, stations, chargers: Retained indefinitely - Transactions: Retained indefinitely (financial records) - Meter values: Retained indefinitely (for disputes/analysis) - Wallets: Retained indefinitely

Operational Data: - OCPP message logs: Configurable retention (default: indefinite) - Firmware files: Retained until manually deleted by admin

Size Estimates: - Each transaction: ~1-5 KB (depending on meter value frequency) - OCPP logs: ~500 bytes per message - For 100 chargers with 10 sessions/day each: - ~1,000 transactions/day = ~5 MB/day - ~10,000 OCPP messages/day = ~5 MB/day - Total: ~10 MB/day ≈ 3.6 GB/year

Scalability: PostgreSQL can handle millions of transactions with proper indexing

Logging

Application Logs: - Backend server logs: startup, errors, warnings - WebSocket connection events: connect, disconnect, errors - Retention: Typically 30-90 days (configurable per deployment)

OCPP Logs: - Complete message audit trail (as described earlier) - Stored in database for long-term retention

---

9. Integration Guide for Hardware

This section provides guidance for integrating new OCPP 1.6-compliant charging hardware with the system.

9.1 Pre-Integration Checklist

Before connecting a charge point:

- [ ] Charger supports OCPP 1.6 (verify with manufacturer documentation)
- [ ] Charger uses JSON over WebSocket (not SOAP or HTTP)
- [ ] Charger has stable internet connection
- [ ] Charger has unique identifier configured (charge point ID)
- [ ] Central system URL is known
- [ ] Charging station location exists in database (or will be created)

9.2 Configuration Steps

Step 1: Add Station to System (if new location)

Via Admin Web Interface: 1. Navigate to "Stations" section 2. Click "Add Station" 3. Enter station details: - Name (e.g., "Downtown Parking Garage") - Address - Latitude and longitude (for map display) 4. Save station

Step 2: Add Charger to Database

Via Admin Web Interface: 1. Navigate to "Chargers" section 2. Click "Add Charger" 3. Enter charger details: - **Charge Point String ID:** Must match charger's configured ID exactly (e.g., "CP001") - **Station:** Select from dropdown (from Step 1) - **Vendor:** Manufacturer name - **Model:** Model number - **Serial Number:** (optional but recommended) 4. Save charger

Via API (for bulk import):

```
POST /api/chargers
Authorization: Bearer {admin_jwt_token}
Content-Type: application/json

{
  "charge_point_string_id": "CP001",
  "station_id": 123,
  "vendor": "EVBox",
  "model": "BusinessLine",
  "serial_number": "EVB-12345"
}
```

Step 3: Add Connectors

Via Admin Web Interface: 1. Navigate to charger details page 2. Click "Add Connector" 3. Enter connector details: - **Connector ID:** 1 (for first outlet, 2 for second, etc.) - **Connector Type:** Type 2, CCS, CHAdeMO, etc. - **Max Power:** Rated power in kW (e.g., 22 for 22 kW charger) 4. Save connector 5. Repeat for additional connectors (if charger has multiple outlets)

Step 4: Configure Charger Hardware

Access charger configuration interface (method varies by manufacturer): - Web interface (if charger has IP address) - Mobile app (if manufacturer provides) - Physical display and buttons - Configuration software via laptop connection

Set these parameters:

Parameter	Value	Notes
Central System URL	`ws://your-domain.com/ocpp/{charge_point_id}`	Replace with actual domain
Or split configuration:		
- Host	`your-domain.com`	Without protocol
- Port	`80` (ws) or `443` (wss)	
- Path	`/ocpp/{charge_point_id}`	Some chargers auto-append ID
Charge Point ID	`CP001` (example)	Must match database entry exactly
OCPP Version	`1.6`	
OCPP Sub-Protocol	`ocpp1.6`	
Security Profile	Unsecured (0) or TLS (1)	Match server configuration
Heartbeat Interval	`300` seconds	Recommended
Meter Values Interval	`60-120` seconds	Frequency of meter value messages

Save configuration and reboot charger if required.

Step 5: Verify Connection

Expected sequence: 1. Charger establishes WebSocket connection to central system 2. Charger sends `BootNotification` message 3. Central system responds with `Accepted` status 4. Charger sends `StatusNotification` for each connector (typically `Available`)

Check in Admin Dashboard: - Navigate to “Chargers” page - Find charger by ID (e.g., CP001) - Verify: - **Connection Status:** Online (green indicator) - **Last Heartbeat:** Recent timestamp (within last 5 minutes) - **Charger Status:** Available (or appropriate status) - **Firmware Version:** Populated (from BootNotification)

If connection fails: - Check charger event logs for error messages - Verify network connectivity (can charger reach server?) - Check firewall rules (outbound WebSocket allowed?) - Verify URL configuration (correct domain, path, charge point ID) - Check central system logs for connection attempts

9.3 Test Charging Session

Perform test transaction to verify full functionality:

1. **Prepare test user:**
   - Create user account in system
   - Ensure user has a `rfid_card_id` assigned (e.g., "TEST_RFID_001")
   - Add funds to user wallet
2. **Start charging via admin dashboard:**
   - Navigate to charger details page
   - Click "Remote Start" button
   - Select test user from dropdown
   - Click "Start"
3. **Verify:**
   - System sends `RemoteStartTransaction` command
   - Charger responds with "Accepted"
   - Charger sends `StartTransaction` message
   - Transaction visible in dashboard with status "RUNNING"
4. **Monitor session:**
   - Check for periodic `MeterValues` messages (every 60-120s)
   - Verify energy consumption incrementing in dashboard
5. **Stop remotely:**
   - Click "Remote Stop" button
   - Verify `RemoteStopTransaction` command sent
   - Verify `StopTransaction` received
   - Verify transaction status changes to "COMPLETED"
   - Verify billing amount calculated correctly
   - Verify wallet balance deducted

Test successful if: - All OCPP messages exchanged correctly - Transaction created and completed - Energy measured and stored - Billing calculated and wallet deducted - No errors in OCPP logs

9.4 Troubleshooting Common Issues

Charger Won't Connect

Symptoms: No connection status in dashboard, no BootNotification received

Checks: 1. **Network connectivity:** Can charger reach internet? (ping test, access manufacturer cloud) 2. **URL configuration:** Is central system URL correct? - Common mistake: Using `http://` instead of `ws://` - Common mistake: Wrong domain or port 3. **Charge point ID:** Does ID in charger config match database entry exactly? (case-sensitive) 4. **Firewall:** Is outbound WebSocket traffic allowed? 5. **Server status:** Is central system running and accessible?

OCPP Logs: Check for connection attempts and error messages

Charger Connects but Immediately Disconnects

Symptoms: Connection status flashes online briefly then goes offline

Possible Causes: 1. **BootNotification rejected:** System rejected boot (charger not in database) 2. **Protocol mismatch:** Charger using OCPP 1.5 (SOAP), system expects 1.6 (JSON) 3. **Network instability:** Intermittent connection drops 4. **Heartbeat timeout:** Charger not sending heartbeats (check heartbeat interval config)

Check OCPP Logs: Look for BootNotification message and response

---

Transactions Don't Start

Symptoms: User initiates charging via app, but no transaction created

Checks: 1. **RFID Card ID assigned:** Does user have a `rfid_card_id` value in their profile? (Required for OCPP) 2. **User account active:** Is user account status "ACTIVE"? 3. **Connector available:** Is connector in "Available" status? 4. **Charger online:** Is charger connected (check Redis/dashboard)? 5. **OCPP logs:** Does `RemoteStartTransaction` command appear? What is charger response? 6. **OCPP logs:** Does `StartTransaction` message appear? What is response?

Common Issues: - User has no `rfid_card_id` → API returns error "User does not have an RFID card ID assigned" - User account blocked → Response: idTagInfo: Blocked - `rfid_card_id` doesn't match any user → Response: idTagInfo: Invalid - Connector not available → RemoteStart rejected by charger - Charger offline → RemoteStart command fails to send

---

Billing Incorrect

Symptoms: Charged amount doesn't match expected value

Checks: 1. **Meter values:** Review stored meter values - are they accurate? 2. **Energy calculation:** Verify meterStop - meterStart = expected energy (in Wh) 3. **Tariff:** Check applicable tariff rate (charger-specific or global) 4. **Calculation:** energy_kwh × tariff_rate = billing_amount

Example Debug: - meterStart: 100 Wh - meterStop: 5100 Wh - Energy: 5100 - 100 = 5000 Wh = 5 kWh - Tariff: ₹12/kWh (from database) - Expected billing: 5 × 12 = ₹60

If mismatch: - Check charger meter calibration (compare to external energy meter) - Verify charger reporting correct units (Wh vs kWh) - Check for unit conversion errors in system

---

Firmware Update Fails

Symptoms: Update status shows "DOWNLOAD_FAILED" or "INSTALLATION_FAILED"

Download Failures: - Check download URL is accessible from charger's network - Verify file server is running and file exists - Check file size (some chargers have storage limits)

Installation Failures: - Verify firmware file is correct for charger model - Check file integrity (not corrupted) - Review charger manufacturer documentation for update procedure - Some chargers require specific file naming or format

Recovery: Charger should remain on previous firmware if update fails

---

9.5 Best Practices

Naming Conventions

Charge Point IDs: - Use consistent format (e.g., `CP001`, `CP002`, `CP003`) - Include location identifier if managing multiple sites (e.g., `SITE1_CP001`) - Avoid special characters (alphanumeric plus hyphen/underscore only)

Station Names: - Descriptive location (e.g., “Downtown Parking Garage”) - Include address or landmark for easy identification

Network Configuration

- **Prefer wired Ethernet** over cellular for reliability
- Configure static IP for chargers (avoids DHCP issues)
- Use WebSocket Secure (wss://) for production (encryption via TLS)
- Monitor network uptime (cellular data plans should have sufficient allowance)

Maintenance

- **Regular firmware updates:** Check manufacturer for security patches and bug fixes
- **Heartbeat monitoring:** Alert if charger offline for >15 minutes
- **Periodic testing:** Monthly test charge to verify functionality
- **Log review:** Periodically check OCPP logs for errors or warnings

Scalability

- **Bulk charger onboarding:** Use API for adding many chargers (faster than manual entry)
- **Staged rollout:** Connect 1-2 test chargers before deploying entire fleet
- **Geographic distribution:** Ensure server has sufficient bandwidth and latency for all locations

---

10. Glossary

OCPP & Charging Terms

AC (Alternating Current): Electrical current that periodically reverses direction. Used for slower charging (typically 3-43 kW). Standard household electricity is AC.

Availability: Status indicating whether a charger is operative (available for use) or inoperative (out of service for maintenance, errors, etc.).

Boot Notification: OCPP message sent by charge point when it starts up, providing identification and configuration information to central system.

Central System (CS): Server that manages charge points. This system is the central system.

CCS (Combined Charging System): DC fast charging standard combining AC connector with additional DC pins. Common in Europe (CCS Type 2) and North America (CCS Type 1).

CHAdeMO: DC fast charging standard developed in Japan. Common in older electric vehicles.

Charge Point (CP): Physical charging station hardware that delivers electrical energy to electric vehicles.

Connector: Physical outlet or socket on a charge point where vehicle plugs in. Types include Type 1, Type 2, CCS, CHAdeMO.

Connector ID: Numeric identifier for a connector on a charge point (1 for first outlet, 2 for second, etc.).

Charge Point ID / Charge Point String ID: Unique alphanumeric identifier for a charge point (e.g., “CP001”). Used in OCPP communication.

DC (Direct Current): Electrical current flowing in one direction. Used for fast charging (typically 50-350 kW). Vehicle batteries store DC power.

EV (Electric Vehicle): Vehicle powered by electricity stored in a battery.

EVSE (Electric Vehicle Supply Equipment): Technical term for charging station hardware.

Heartbeat: Periodic keep-alive message sent by charge point to central system to maintain connection and indicate operational status.

idTag: Identifier used for authorization in OCPP messages. In this system, it contains the user’s virtual RFID card ID (stored in the `rfid_card_id` database field). This field bridges the gap between mobile app authentication (via Clerk JWT) and OCPP protocol requirements. Users never see or interact with this ID directly.

JSON (JavaScript Object Notation): Text-based data format used for OCPP messages. Human-readable and easy for computers to parse.

kW (Kilowatt): Unit of power (1,000 Watts). Indicates charging speed (e.g., 7 kW charger delivers 7 kilowatts of power).

kWh (Kilowatt-hour): Unit of energy (1 kilowatt for 1 hour). Used for billing (e.g., 10 kWh = 10 kilowatt-hours of energy delivered).

Measurand: Specific electrical parameter being measured (e.g., voltage, current, power, energy).

Meter Values: Periodic measurements of electrical parameters (energy, voltage, current, power) during charging session.

OCPP (Open Charge Point Protocol): Open-source communication standard for electric vehicle charging infrastructure. Version 1.6 is widely adopted.

Remote Start/Stop: OCPP commands allowing central system to initiate or terminate charging sessions remotely via mobile app or admin dashboard.

SoC (State of Charge): Percentage of battery capacity currently charged (e.g., 80% SoC means battery is 80% full).

Status Notification: OCPP message sent by charge point whenever connector status changes (e.g., Available → Charging → Available).

Tariff: Price structure for charging, stored in the database as price per kilowatt-hour in INR (e.g., ₹12/kWh). Each charger can have a specific tariff rate, or use the global default tariff. Stored in the `Tariff` table with `rate_per_kwh` field.

Transaction: Complete charging session from start to stop. Includes energy consumed, duration, billing amount.

Type 1 (SAE J1772): AC charging connector standard used in North America and Japan. Single-phase only.

Type 2 (Mennekes / IEC 62196-2): AC charging connector standard used in Europe. Supports both single-phase and three-phase.

Virtual RFID Card ID: A unique identifier stored in each user's profile (`rfid_card_id` database field) used for OCPP protocol compliance. This system does NOT use physical RFID cards or card readers. Instead, this virtual ID bridges mobile app authentication with OCPP's requirement for user identification via `idTag`. Users authenticate through the mobile app (Clerk), and the system automatically uses their virtual RFID card ID in OCPP messages.

WebSocket: Communication protocol providing persistent bidirectional connection between client (charge point) and server (central system). Used as transport for OCPP messages.

Wh (Watt-hour): Unit of energy (1 watt for 1 hour). Often used by chargers for meter readings (converted to kWh for billing).

System-Specific Terms

Admin: User role with full system access for managing infrastructure, users, and viewing all transactions.

Billing: Process of calculating charge amount (energy × tariff) and deducting from user wallet.

Firmware OTA (Over-The-Air): Remote update of charge point firmware without physical access to hardware.

JWT (JSON Web Token): Secure token used for user authentication in API requests.

Redis: In-memory database used for fast access to connection status and session data.

Wallet: User account balance used for paying charging session fees.

Wallet Transaction: Record of wallet activity (top-up or deduction).

Document Revision History

Version	Date	Changes
1.0	January 2025	Initial documentation for OCPP Charging Station Management System v2.0

Contact & Support

For questions about this system: - **Technical Issues:** Check OCPP logs in admin dashboard - **Hardware Integration:** Review Integration Guide (Section 9) - **System Administration:** Refer to relevant section of this documentation

---

End of Document