Locate a Socket - Design Document

Use Cases

UC1: Find Nearby Charging Stations.

Actor: EV Driver

Goal: Locate available charging stations near current location.

Flow: User opens app → GPS detection → Map display → Filter application → Station selection

UC2: Plan Route with Charging Stops.

Actor: EV Driver

Goal: Plan optimal route with necessary charging stops.

Flow: Enter destination → Specify battery level → Calculate route → Confirm → Navigate

UC3: Make Payment for Charging.

Actor: EV Driver

Goal: Securely pay for charging services.

Flow: Initiate charging → View rates → Confirm payment → Process → Start charging

UC4: Manage Station Information.

Actor: Station Operator

Goal: Update station information and availability.

Flow: Login → Select station → Update info → Validate → Publish updates

User Stories

EV Drivers:

- US1: Find nearest available charging station quickly.
- US2: See real-time availability to avoid occupied stations.
- US3: Filter by connector type for compatibility.
- US4: Plan routes with charging stops for long journeys.
- US5: Pay through app without multiple cards.

Station Operators:

- US6: Update station availability for accurate information.
- US7: Set dynamic pricing to optimize revenue.

System Administrators:

- US8: Monitor system performance for optimal user experience.

Locate a Socket - Design Document

User Requirements

Functional Requirements

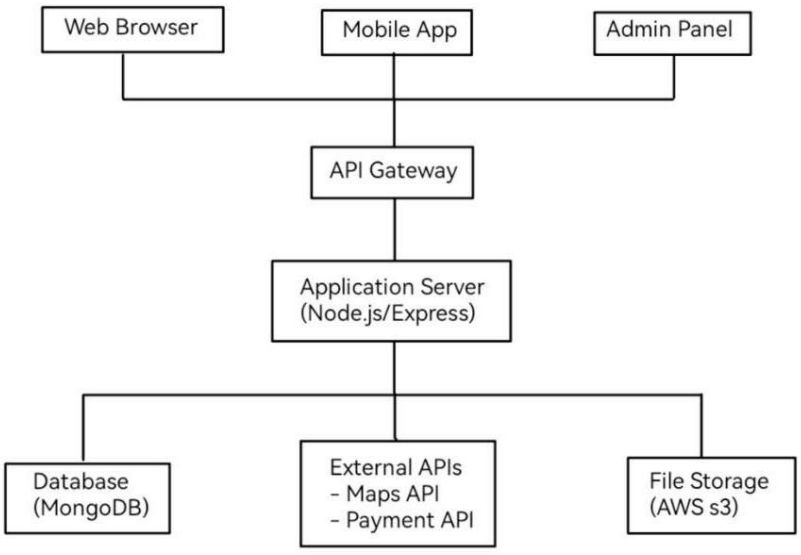
1. Location Services: GPS-based station discovery with real-time availability.
2. Route Planning: Optimal route calculation considering vehicle range and traffic.
3. Payment Processing: Multi-method payment with PCI-DSS compliance [2].
4. Station Management: Real-time status updates from operators.
5. Cross-Platform Access: Responsive web application for all devices.

Non-Functional Requirements

1. Performance: <3 second response time, 50,000+ concurrent users.
2. Security: AES-256 encryption, HTTPS, OAuth 2.0 authentication.
3. Availability: 99.5% uptime with failover mechanisms.
4. Usability: WCAG 2.1 compliance [3], ≤3 clicks for key functions.

Design Specifications

System Architecture



Database Design

Key Collections:

- Users: Profile, vehicle info, payment methods (_id: PRIMARY KEY, email: UNIQUE)
- Charging Stations: Location, operator details, connectors, status (_id: PRIMARY KEY, location: GEO INDEX)

- Sessions: Active charging sessions with tracking (session_id: PRIMARY KEY, user_id/station_id: FOREIGN KEYS)

API Design

Core Endpoints:

- GET /api/stations/nearby?lat={lat}&lng={lng}&radius={radius}
- POST /api/auth/login
- GET /api/routes/plan?origin={origin}&destination={dest}
- POST /api/payments/process
- PUT /api/stations/{stationId}/status

Design Traceability Matrix

Use Case	User Stories	Requirements	Design Components
UC1: Find Stations	US1, US2, US3	Location Services	Map API, GPS Service, Station Database
UC2: Route Planning	US4	Route Optimization	Routing Algorithm, Traffic API
UC3: Payment	US5	Payment Processing	Payment Gateway, Encryption
UC4: Station Management	US6, US7	Station Updates	Admin Interface, Real-time Updates

References

[1] Google Developers. (2024). Maps Platform Documentation. Google LLC. https://developers.google.com/maps/documentation

[2] Payment Card Industry Security Standards Council. (2022). Payment Card Industry Data Security Standard v4.0. https://www.pcisecuritystandards.org/

[3] World Wide Web Consortium. (2018). Web Content Accessibility Guidelines (WCAG) 2.1. https://www.w3.org/TR/WCAG21/