TQS: Product specification report

Guilherme Rosa [113968], Maria Linhares [113534], Martim Santos[114614], Rui Machado[113765]
v2025-05-15

1 Introduction

1.1 Overview of the project

As part of the TQS (Testing and Quality Software) course, this project offers a practical opportunity to integrate and apply core software engineering practices in a real-world development scenario. The assignment emphasizes a full-stack engineering approach supported by strong foundations in Software Quality Assurance (SQA) and DevOps. Students are expected to deliver not only a technically functional application but also a robust development process, incorporating agile project management, continuous integration and delivery (CI/CD), test automation, static code analysis, and structured quality documentation.

The product developed in this context is **Elytra**—a unified platform designed to streamline interactions between different stakeholders in the electric vehicle (EV) charging ecosystem. While the core technical work includes backend services, APIs, quality pipelines, and deployment workflows, the project also involves architectural planning, iterative backlog refinement, and collaboration tools to simulate professional-grade software development.

Elytra provides a compelling case study for TQS by addressing a multi-actor, integration-heavy use case, making it ideal for exercising quality strategies across functional and non-functional dimensions. It serves as a vehicle for experimenting with testing patterns, deployment automation, and cross-team coordination, reflecting the end-to-end mindset promoted by the course.

1.2 Known limitations

We initially were using geolocation in order for the EV Driver to be able to get the nearest stations based on his location, but since we don't have a certificate, it is not possible to use geolocation. Due to a strict timeline, we weren’t able to implement payments, schedule off-peak charging discounts, nor user account and history management.

1.3 References and resources

https://ui.shadcn.com/
https://www.geeksforgeeks.org/spring-boot-3-0-jwt-authentication-with-spring-security-using-mysql-database/
https://playwright.dev/docs/intro

2 Product concept and requirements

2.1 Vision statement

Elytra addresses one of the most significant obstacles in the electric vehicle (EV) ecosystem: the fragmentation of charging services across multiple networks. Today, EV drivers often face a disjointed experience—switching between incompatible apps, inconsistent interfaces, and limited visibility into station availability or pricing. Our platform provides a unified, user-friendly solution that simplifies the

charging process, empowering users to interact with a diverse ecosystem through a single, seamless interface.

From a functional (black-box) perspective, Elytra enables three core user groups—drivers, station operators, and third-party services—to connect and interact through a central platform. For drivers, the system supports searching for nearby charging stations, booking time slots in advance, initiating charging sessions, making secure payments, and reviewing consumption history and statistics. For operators, Elytra provides tools to register and manage station availability, monitor usage, and configure pricing or promotional rules. The platform also integrates with third-party APIs for maps, payment gateways, and station hardware.

While the original scope included researching and potentially integrating real-world EV charging protocols like OCPI or OCPP, this was ultimately adapted to focus on a simulated environment. This decision allowed the team to prioritize feature completeness, testing coverage, and system stability within the timeframe of the project.

2.2 Personas and scenarios

Persona 1 - Station Operator, Gelson

Figure 1 - Gelson Martins, a station operator

Station Operator

Name: Gelson Martins

Age: 41

Gender: Male

Demographics: Degree in Electrical Engineering

Job Title: Operations Manager

Context: Gelson manages an electric vehicle charging station located in a suburban area. He is responsible for keeping the station running efficiently, ensuring availability, performing maintenance, and controlling operational energy costs.

Needs:

- View real-time status of the station (active, faulty, under maintenance).
- Configure off-peak discount hours to optimize energy costs.
- Schedule maintenance and log faults easily.
- Generate reports on station usage and energy consumption.
- Receive automatic alerts for faults or unauthorized usage.
- Integrate with external systems (maps, billing, maintenance).
- Prevent overbooking and unauthorized access.

Scenario 1: Real-Time Station Monitoring

It's Monday morning at 7:45 AM. Gelson logs into the operations dashboard from his office before the first wave of EV commuters arrives. The dashboard shows 12 chargers, 11 in green (active) and one in red (faulty). That unit flagged a power relay issue around 3 AM.

He clicks on the red unit and opens the activity log. It shows intermittent errors during the night. Gelson adds a comment for the maintenance team and marks it as "awaiting technician." The system updates the unit status to yellow (under maintenance) and removes it from the public booking pool automatically.

This quick morning check allows him to keep availability high and avoid customer complaints during the morning rush.

Scenario 2: Configure Off-Peak Discounts

On Wednesday afternoon, Gelson receives an email from headquarters reminding all regional stations to lower energy costs during peak hours. He opens the pricing control module, reviews recent energy consumption, and notices a clear drop in usage after 9 PM.

To encourage night-time charging, he configures a 20% discount between 9 PM and 6 AM, starting immediately. The system updates the pricing tiers and applies a discount label to off-peak hours in the booking app.

A few hours later, the customer analytics dashboard shows three users scheduled charging during the new discount window. Gelson makes a note to revisit the impact of this change at the end of the week.

Scenario 3: Fault and Security Alerts

At 10:37 PM on a Saturday, Gelson is off duty when he receives a push notification: "Unauthorized access attempt at Unit 4 – RFID mismatch – 3 failed swipes."

He opens the alert on his phone. It shows a failed authentication attempt using a blacklisted RFID tag. The system automatically disables Unit 4 temporarily and logs the incident.

Gelson remotely disables the tag and adds the event to the security incident tracker. He forwards the alert to the central fraud team and sets the unit to be reviewed in person on Monday morning.

Without needing to leave his home, he contained a potential breach within minutes.

Persona 2 - EV Driver, Sofia

Figure 2 - Sofia Martins, a Electrical Vehicle Driver

Electrical Vehicle Driver

Name: Sofia Martins

Age: 34

Gender: Female

Demographics: Urban resident, lives in Aveiro, Portugal

Job Title: Marketing manager at a tech startup

Context: Sofia is a daily EV Hyundai driver who relies on her electric vehicle for both personal and professional travel. She values sustainability and digital convenience but often finds the EV charging experience fragmented and unreliable. She wants a seamless solution that allows her to efficiently manage her charging routine without unnecessary detours, delays, or app overload.

Needs:

- Locate compatible and available EV charging stations in real time;
- Book time slots in advance to avoid waiting or unexpected unavailability;
- Monitor charging sessions and track energy usage history;
- Make quick, secure payments through unified platform;
- Filter stations by connector type, power level, and price;
- Receive notifications when a booked slot is about to start or if issues occur.

Scenario 1: Morning Commute – Booking a Charging Slot Before Work

Sofia finishes her breakfast and opens the Elytra app to check nearby charging stations along her route to work. She notices that her usual charger is already booked, but finds another fast-charger just a few blocks away.

She filters stations by connector type (Type 2) and availability and then books a slot for 8:30 AM. The app confirms the booking and notifies her 10 minutes before the slot begins. Once there, she scans the QR code, unlocks the charger, and starts her session.

She then receives a summary of the session and herb, all logged in her dashboard.

Scenario 2: Weekend Trip – Planning a Long-Distance Route with Charging Stops

Sofia and her partner are planning a weekend getaway to the countryside. She uses Elytra's trip planner tool to map the route, automatically highlighting optimal charging stations based on her car's range and driving preferences.

She books two slots: one mid-way to the destination and one on the way back. Elytra provides a forecast of station occupancy during the selected times, helping her avoid peak hours. On the road, she receives live updates in case of unexpected station downtimes, allowing her to reroute if needed.

Persona 3 - EV Driver, António

Figure 3 - Antonio Silva, a Electrical Vehicle Driver

Electrical Vehicle Driver

Name: Antonio Silva

Age: 53

Gender: Male

Demographics: Urban resident, lives in Aveiro, Portugal

Job Title: Works as a trasporter for a tech company

Context: Antonio Silva is a 53-year-old transporter working for a tech company in Aveiro, Portugal. As an urban resident who drives throughout the day making multiple deliveries or pickups, his work heavily depends on the reliability and availability of his electric vehicle (EV). His daily routine involves constant movement across different parts of the city and surrounding areas, often on a

tight schedule. Ensuring his EV remains sufficiently charged is crucial not only for fulfilling his job responsibilities but also for avoiding delays and downtime.

Needs:

- Ensure his EV remains charged throughout the day to avoid job interruptions.
- Locate nearby charging stations quickly while on the move.
- Filter charging stations by availability, speed, and cost to choose the most efficient option.
- Minimize downtime by identifying stations with fast chargers.
- Book charging slots in advance to avoid waiting in queues.
- Plan charging breaks around his delivery route to maintain efficiency.
- Receive real-time updates on charger status and availability.

Scenario 1: Mid-Day Charging Emergency

Antonio is halfway through his delivery route when he notices that his EV battery has unexpectedly dropped to 15%, likely due to an unplanned detour through a traffic-congested area. With several stops left to complete and time ticking, he needs to find a charging station immediately to avoid falling behind schedule. He opens the Elytra app, which quickly locates nearby charging stations and highlights those with fast chargers that are currently available. A station just five minutes away pops up with a free fast charging slot. Antonio reserves the charger while still en route and follows the app's navigation directly to the station, where he charges up quickly and continues his deliveries without losing valuable time.

Scenario 2: Pre-Route Planning for the Day

At 7:30 in the morning, before beginning his delivery route, Antonio wants to plan his charging break for later in the day. Knowing that stopping mid-route for a charge can cause delays, he uses Elytra to map out stations along his expected delivery path. He's specifically looking for a charging station near a place he can grab lunch, with a lower cost per kilowatt and the option to reserve in advance. Elytra analyzes his route, shows compatible stations, and allows him to book a charging slot at 12:30 PM at a station next to his favorite café. With this planned stop, Antonio aligns his lunch break with his vehicle's charging needs and keeps his schedule intact.

Scenario 3: Peak Hour Congestion

It's 6 P, and Antonio has just finished a long day of deliveries. At this time, many public chargers are occupied, and he dreads the possibility of waiting in line to recharge before heading home. Before driving to any station, he checks Elytra to see which locations are currently busy. The app shows that his usual spot is at full capacity, but a nearby station just two kilometers away still has an available slow charger. Antonio books the slot through Elytra and heads straight there, avoiding unnecessary idling or driving in circles looking for a charger. He arrives, plugs in, and starts winding down while his EV charges, stress-free.

Scenario 4: Unexpected Charger Malfunction

One afternoon, Antonio arrives at a charger he frequently uses, only to find that it's out of service due to ongoing maintenance. He doesn't have time to search manually through various apps or wait for someone from the station to fix it. Instead, he opens Elytra, which automatically flags the charger as offline and presents real-time alternatives nearby. One option stands out—a fast charger that's available just a few minutes away. With a tap, he gets directions and books the charger while on

the move. When he arrives, he reports the issue with the original station through the app, helping keep the system updated for other drivers too.

Scenario 5: Managing Weekly Charging Costs

As someone who drives all day for work, Antonio likes to keep an eye on how much he spends charging his EV each week. Instead of saving receipts or trying to calculate manually, he turns to Elytra's built-in analytics. The platform provides him with a clear breakdown of his weekly charging sessions, showing how much energy he's used, the total cost, and the stations he's visited. With this data, Antonio realizes which stations offer the best balance of speed and price. He adjusts his routine slightly to prioritize those locations, saving money and charging more efficiently. When it's time to report expenses to his employer, he simply exports the data through Elytra and submits it with ease.

Persona 4 - Administrator, Miguel

Figure 4 - Miguel Silva, System Administrator

System Administrator

Name: Antonio Silva

Age: 39

Gender: Male

Demographics: Urban resident, lives in Aveiro, Portugal

Job Title: Station Operations Administrator

Context: Miguel is responsible for overseeing the operations of the charging stations. His role involves managing charging station availability, ensuring maintenance schedules are adhered to, and handling real-time operations. He uses the admin interface to update charging station statuses, monitor usage patterns, and optimize the scheduling of off-peak charging slots. He values efficiency and automation, aiming to minimize manual oversight. With the integration of the Nikola platform, Miguel wants to ensure a smooth user experience for drivers, and he is always looking for ways to improve station utilization.

Needs:

- Ability to register and update charging station status, including availability, maintenance schedules, and configuration of off-peak charging slots.
- View the current usage statistics for all charging stations in the network, including energy consumption, occupancy levels, and payment status.
- Configure discounts for off-peak times and monitor station load to ensure a balanced distribution of users across the network.
- Receive notifications about pending or completed maintenance, and ensure that charging stations are operational.
- Access detailed usage data for both individual drivers and the overall system, helping to monitor trends and identify areas for improvement.
- View aggregate data on station performance, energy consumption, CO2 savings, and other relevant KPIs.

Scenario 1: Real-Time Station Availability Management

It's 8:15 AM on a busy Tuesday morning. Miguel logs into the Nikola platform dashboard from his office to check the current status of the charging stations. The system shows 15 charging stations, with 13 marked as green (available) and 2 marked as yellow (under maintenance). He notices that Station 7 has been under maintenance for the past 12 hours due to a power issue.

He clicks on the yellow status for Station 7 to view its activity log, which shows multiple maintenance alerts in the past 24 hours. He sees that the technician has marked the issue as resolved and updates the station's status back to green (active). This change makes the station available for new bookings immediately.

Miguel also notices Station 4 showing intermittent errors for the last hour. He flags it for urgent inspection and adds a note for the maintenance team to investigate before peak hours.

This proactive monitoring allows Miguel to ensure the availability of charging stations for customers during the morning rush, reducing the chance of complaints.

Scenario 2: Schedule Off-Peak Charging Discounts

On Thursday at 3:30 PM, Miguel receives a notification from Nikola's central operations requesting that all stations in the network implement a 15% discount for off-peak charging hours between 10 PM and 6 AM to encourage usage during low-demand periods.

Miguel opens the pricing management interface and reviews the recent usage trends for all stations. He notices that, historically, usage drops significantly after 10 PM. He then applies the discount for the specified time window and updates the system to reflect these changes.

He also notices that Station 2 has a lot of available slots during the evening, so he configures a higher priority for reservations during the discount window.

The next morning, he checks the user engagement dashboard, and sees that a 5% increase in bookings during the new off-peak hours. Miguel plans to review the full impact of this initiative by the end of the week.

Scenario 3: Monitor Charging Station Performance and Consumption

It's Friday morning, and Miguel is reviewing the monthly performance of the charging stations. He opens the aggregated consumption dashboard, which displays key data on the energy usage, cost, and CO2 savings for each station.

Miguel notices that Station 6 has consumed more energy than usual over the last week, and the CO2 savings are slightly lower than expected. He drills down into the data for that specific station and sees that there was an unusually high volume of users on Tuesday and Wednesday, most likely due to local events in the area.

Miguel sets up a weekly report alert to track any future spikes in usage and considers adjusting the station's availability during high-traffic periods to reduce overload.

Scenario 4: Respond to a Faulty Charging Unit

At 11:45 PM on a Thursday night, Miguel receives a push notification on his phone indicating a charging unit malfunction at Station 5. The system reports that the charging cable has been inadvertently damaged by a vehicle during charging.

Miguel immediately opens the notification and sees that the system has temporarily disabled the charging unit and logged the incident. He confirms the incident details, including damage to the cable, and adds a note for the maintenance team to replace the cable as soon as possible.

He also sends a quick message to the customer support team to notify any affected customers and offer them a discount for the inconvenience. The system automatically triggers an email to the users who had a booking for that station, informing them of the issue and offering alternative stations nearby.

This swift handling of the issue ensures minimal disruption to customer service and avoids negative feedback.

Scenario 5: Respond to a Faulty Charging Unit

On a quiet Wednesday afternoon, Miguel is reviewing the CO2 footprint dashboard to see how well Nikola EV Services is doing in terms of reducing emissions. The dashboard displays data showing that the system has helped save 300 tons of CO2 this month across all stations.

Miguel notes that users are increasingly using the CO2 savings feature to track their personal environmental impact, with several customers sharing their stats on social media. He decides to add a monthly summary report for users, showing their personal CO2 savings, and plans to send an email campaign to encourage users to share their results.

By tracking the CO2 savings, Miguel aims to boost the platform's sustainability image and increase customer loyalty.

2.3 Project epics and priorities

EPIC1 - Stations Management

Covers all operations related to registering, updating, and monitoring charging stations to ensure their availability and operational status.

- As a Station Operator, I want to register a new charging station so that it appears in the system.
- As a Station Operator, I want to view usage data to manage operations effectively.
- As a Station Operator, I want to update the maintenance status of a station so that users see only available chargers.

EPIC2 - Charging Station Discovery

Focuses on enabling users to find and evaluate charging stations based on location and preferences.

- As an EV Driver, I want to search for nearby charging stations so that I can find a convenient location to charge.
- As an EV Driver, I want to filter charging stations by availability, charger type, or cost so that I can choose the best option.

EPIC3 – Slot Reservation and Scheduling

Handles the planning and reservation of charging slots, ensuring coordinated and conflict-free usage of infrastructure.

- As an EV Driver, I want to book a time slot at a charging station so I can ensure availability when I arrive.
- As an EV Driver, I want to receive confirmation of my booking to avoid double booking or overbooking.
- As a Station Operator, I want to manage booking slots and availability so that the station is efficiently used.

EPIC4 – Charging Session Control

Encompasses operations related to the actual use of the charger, including session initiation and monitoring.

- As an EV Driver, I want to unlock a charger remotely so that I can start charging when I arrive.
- As an EV Driver, I want to track the consumption of my vehicle during the charging session so I can manage costs and monitor battery health.

EPIC5 – Payment and Billing

Covers pricing, payment processing, and billing models for charging sessions.

- As an EV Driver, I want to pay for the charging session using a pay-per-use model so that I only pay for what I use.
- As an EV Driver, I want to subscribe to a plan to save on frequent charging sessions.
- As a Platform, I want to integrate with third-party payment gateways so that I can process transactions securely.

EPIC6 – User Account and History Management

Relates to user identity, preferences, and past interactions with the platform.

- As an EV Driver, I want to view my past charging sessions and stats so I can track my usage over time.

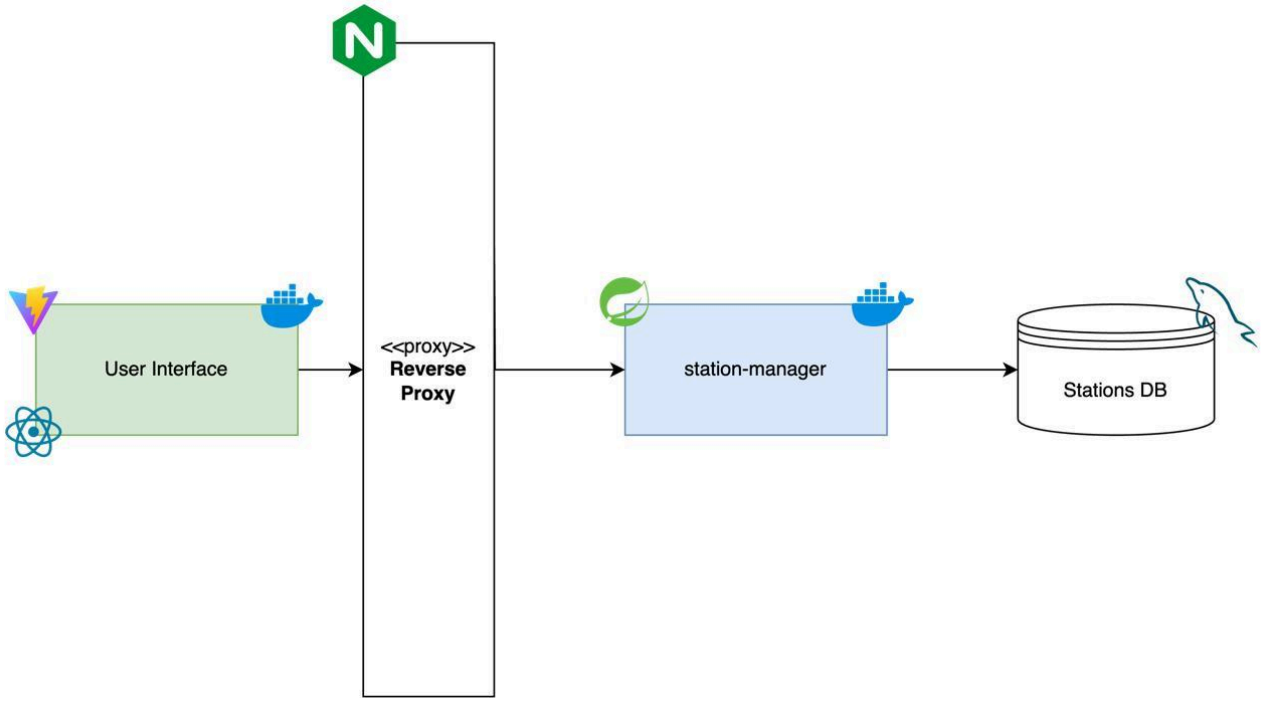
- As an EV Driver, I want to manage my profile and preferences so that my experience is personalized.

3 Architecture notebook

3.1 Key requirements and constrains

The architecture of Elytra was driven by several critical requirements and constraints that shaped our design decisions. First, scalability and performance were paramount: the platform needed to handle a growing user base of EV drivers and an expanding network of charging stations without degradation in responsiveness. To meet this demand, we adopted containerization with Docker to ensure consistent environments across development, testing, and production, and placed NGINX as a reverse proxy to enable efficient request routing, SSL termination, and future load-balancing. Modularity and extensibility were also essential, since we anticipate integrating industry standards like OCPI/OCPP and third-party services (e.g., Google Maps, Stripe) down the road. By isolating these potential dependencies within discrete services, we maintain a clean separation of concerns that simplifies both future feature additions and non-functional testing. Security and maintainability further influenced our choices: sensitive user and payment data must remain protected, and service boundaries must support independent development, automated testing, and peer review in our CI/CD pipelines.

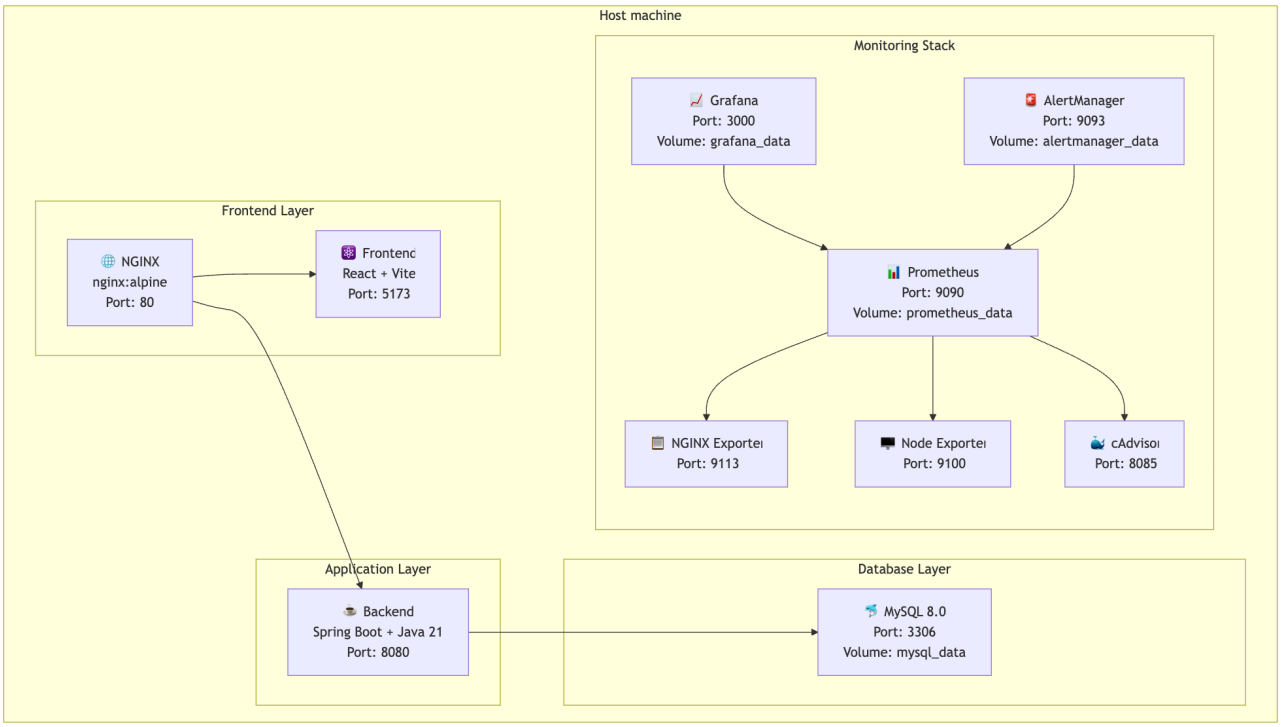
3.2 Architecture view



Elytra's logical architecture comprises three layers: a web-based user interface, a backend "station-manager" service, a relational database, and a reverse proxy between the frontend and backend. The frontend is implemented in React and Vite, packaged via Docker, and exposed to users through NGINX, which abstracts away internal endpoints and manages incoming traffic. When a driver requests station information or books a charging slot, the UI issues an HTTP call that NGINX forwards to our Spring Boot–based station-manager service. This service encapsulates all business logic related to station discovery and booking and interacts with a MySQL database where station metadata is persisted. Responses flow back through the same tunnel, database to backend to proxy to UI, ensuring near-real-time feedback and a responsive user experience.

Beyond these core interactions, our design anticipates more advanced concerns without yet implementing them in the MVP. Finally, our entire stack is integrated into a CI/CD pipeline—automated test suites run on every commit, Docker images are built and scanned, and deployments are managed smoothly—mirroring professional best practices in quality assurance and DevOps.

3.3 Deployment view



The system runs in production using Docker containers that work together to handle user interaction. The setup includes 5 main services: NGINX handles all incoming traffi,, the Spring Boot app runs the business logic, MySQL stores all the charging station and user data, while Prometheus and Grafana handle system monitoring and send alerts when something goes wrong. Everything runs on internal Docker networks so services can talk to each other securely, but only the web interface (port 80) and monitoring dashboards are accessible from outside. The whole system uses environment variables for passwords and settings, stores data in persistent volumes so nothing gets lost when containers restart, and can be updated or scaled without taking the system offline.

4 API for developers

The API developed for our system follows REST architectural principles, focusing on a resource-oriented design to ensure clarity, scalability, and maintainability. The core of the API is structured around domain-specific resources such as **admins**, **users**, **station operators**, **stations**, **chargers**, **cars**, and **bookings**. Each resource is accessed through semantically meaningful endpoints that align with best practices—using HTTP methods (`GET`, `POST`, `PUT`, `DELETE`) to represent actions on resources, rather than embedding actions in the endpoint names themselves.

The API is organized into logical controllers, including `admin-controller`, `auth-controller`, `booking-controller`, `car-controller`, `charger-controller`, `station-controller`, and `user-controller`. These are internally managed using Spring Boot and exposed through a reverse proxy (Nginx) for secure and controlled access. The API documentation is integrated using Swagger UI, which allows for dynamic exploration and testing

of endpoints by developers and stakeholders. This external documentation approach ensures that the implementation remains decoupled from the interface and that changes to business logic are reflected in real-time within the documentation.