Coding Challenge

The goal of the challenge is to create a very simple Spring Boot application for managing charge points used to charge electric vehicles.

Setup

Use Spring Initialzr to create a basic Spring Boot application with a configuration allowing to manage entities in an H2 database and accessing the data using a REST API with authentication.

Basic REST API

The version information of the web application consists of (a) Version of the application and (b) Version of the database schema.

1. Both application version and database schema version are specified as properties in `application.properties`.
2. Create a Spring service allowing access to the version information from `application.properties`.
3. Implement an unauthenticated REST endpoint returning the version information obtained from the previously created service.

Entity Model

Create an entity model using the description below. We listed key properties of the respective entity in parentheses if they are not part of the description.

- The system manages customers (name) and their charge points (name, unique serial number). Customers do not share charge points.
- A charge point has one or more connectors (connector number).
- RFID tags (name, number) are used to authenticate the access to charge points.
    - RFID tags are owned by customers. Customers do not share RFID tags.
    - The numbers of RFID tags are globally unique.
    - RFID tags are linked to at most one vehicle.
- After a vehicle (name, registration plate) is plugged into a connector, a charging session is started. The time and meter value are persisted in the database.
- When the charging session is ended, the end time and end meter value are persisted in the database.
- The charge point transmits an error message if the charging session cannot be completed successfully. The error message is stored as part of the charging session.

Make sensible assumptions about the data model and the properties of the entities to fill in gaps in the description.

Provide means to easily populate the database with test data (e.g., SQL file imported during the start-up of application).

Date Converter

Implement a utility class allowing you to convert a string value to a Java object representing a date with time. The converter shall support different patterns and automatically determine the pattern to be used for a string value. For example, it should be able to convert the dates “2020/03/01” and “2020-03-05 13:10”.

Write unit tests for the utility class.

REST API

Develop a REST API which is protected by username-password authentication. Users can either have the role Admin or Customer (not both) and can only access the endpoints allowed for their role.

Admin REST API: Implement an endpoint that

- Returns the list of charging sessions stored in the database (pagination is not required). It should allow you to specify a date range to filter the result.
- Adds a new connector to an existing charge point.

Customer REST API: Implement an endpoint for

- Starting a charging session.
- Ending a charging session.

Describe in words what unit tests you would implement to ensure the correct behaviour of the date range filter used in the service listing the charging session.

Code Versioning

Create a github repository and upload your web application.