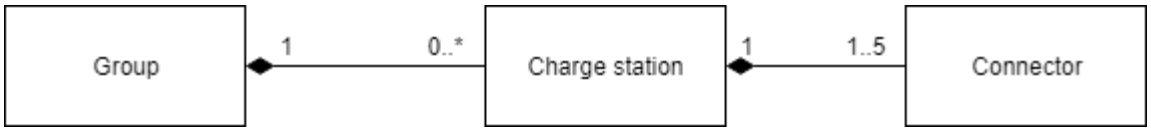
GreenFlux Smart Charging Assignment

Version: 25

Your goal is to develop an API that exposes a simplified smart charging domain.

Domain model:



Group – has a unique Identifier (cannot be changed), Name (can be changed), Capacity in Amps (integer, value greater than zero, can be changed). A Group can contain multiple charge stations.

Charge station – has a unique Identifier (cannot be changed), Name (can be changed), and Connectors (at least one, but not more than 5).

Connector – has integer Identifier unique within the context of a charge station with (possible range of values from 1 to 5), Max current in Amps (integer, value greater than zero, can be changed).

Functional requirements:

1. *Group*/*Charge Station*/*Connector* can be created, updated, and removed.
2. If a *Group* is removed, all *Charge Stations* in the *Group* are removed as well.
3. Only one *Charge Station* can be added/removed to a *Group* in one call.
4. The *Charge Station* can be only in one *Group* at the same time.
   The *Charge Station* cannot exist in the domain without *Group*.
5. A *Connector* cannot exist in the domain without a *Charge Station*.
6. The Max current in Amps of an existing *Connector* can be changed (updated).
7. The Capacity in Amps of a *Group* should always be greater than or equal to the sum of the Max current in Amps of all *Connectors* indirectly belonging to the *Group*.
8. All operations/requests not meeting the above requirement should be rejected.

Further requirements:

1. Design and implement a RESTful HTTP API according to described domain model and functional requirements.
2. Develop your solution in .NET/C#. You can include any third-party libraries or frameworks through NuGet.
3. Use any convenient database to store data. In-memory storage is also an option.
4. Include OpenAPI/Swagger documentation so that we can easily see how your API looks like.
5. Add unit and/or integration tests as you think are necessary.
6. Keep the performance of the solution in mind.
7. Use a local Git repository for your code.
8. Add a *readme.md* file to the root of your repository describing the project. Describe the steps that we need to take to build and run your solution. Pay specific attention if your solution requires extra set up (e.g., creating a database). Also, mention the version of the assignment (see top of this document) in the assignment.

We will evaluate if your code:

1. Implements all requirements.
2. Can be built and run locally.
3. Has no bugs.
4. Is easy to understand, clean, and not overengineered.
5. Is testable and sufficiently covered by tests.
6. Is maintainable (what if we need to extend your solution in the future).
7. Demonstrates your approach and ability to design and implement such an API.

Please don't spend effort on things that are not asked and are not required to solve the problem. After reviewing your assignment, we will also use your solution as a basis for the technical interview. It is important that you can argue clearly about your design decisions.

Sharing your solution publicly on the internet will disqualify your submission.

If you have additional questions about the assignment, feel free to reach out by email as well. We are happy to help you.