Backend Technical Challenge

👋 Welcome to the VW DIGITAL:HUB backend technical test!

We appreciate the time and effort you've invested in reading this document. We believe that you would make an excellent addition to our team, and we would like to learn more about your technical background to proceed with the application process.

📝 Submitting the technical challenge

We recognize that time is our most precious asset, which is why we have devised a technical assessment that should be fairly short and will allow us to identify your technical capabilities. The code challenge is expected to take approximately 4-8 hours, please try to don’t expend more time.

We ask that you submit the challenge through GitHub. Kindly add the specified GitHub user (@kkrusti) as a collaborator to your repository. If you need guidance on this process, please consult the "Inviting collaborators to a personal repository" section in the GitHub Documentation Page. If you have any doubts or questions, feel free to contact our Human Resources email at any time, and we will gladly assist you.

🔧 Technical Aspects to Keep in Mind

- Implementing Hexagonal Architecture and tactical patterns from Domain-Driven Design (DDD) will be highly valued. While it is not mandatory to use these approaches, our projects utilize these patterns and architectures, so incorporating them would be greatly beneficial.
- In our daily work, we use Go, so the challenge should be completed using any version of Go from 1.18 onwards.
- Include a README.md file in the project, where you should explain the technical decisions and technologies you have employed. Feel free to mention any assumptions you have made while interpreting the challenge. Since some aspects of the test may have open interpretations, it is crucial for us to understand your reasons for choosing one approach over another. Please include in the README your name and surname for a better tracking.
- A rich domain model will also be highly appreciated. A rich domain model encapsulates business logic within the domain objects themselves, rather than relying on service classes to handle that logic. This leads to a more cohesive and modular system where logic is placed closer to the data it manipulates, making the code more maintainable, flexible, and easier to reason about. It is also less error-prone since the domain rules and constraints are consistently enforced. Please provide an explanation of the benefits of your chosen domain model in your README file.
- Respecting the input specifications is critical. The instructions are separated by lines, and this is an unchangeable requirement from the client.
- When working on the code challenge, it is important to prioritize writing code that is clean, easy to read, and maintainable. This means paying attention to things like variable names, code organization, and overall structure. Well-organized code makes it easier for other developers to understand, modify, and maintain it in the long run.

- As programmers, we love to sleep soundly at night. Therefore, it would be ideal if you ensure your code works correctly. Feel free to use any testing approach that suits your project best, such as unit tests or integration tests. By thoroughly testing your code, you not only guarantee its functionality and correctness, but also demonstrate a professional commitment to producing reliable and robust software. Remember, a well-tested code is a reliable code, providing peace of mind for both you and your team.

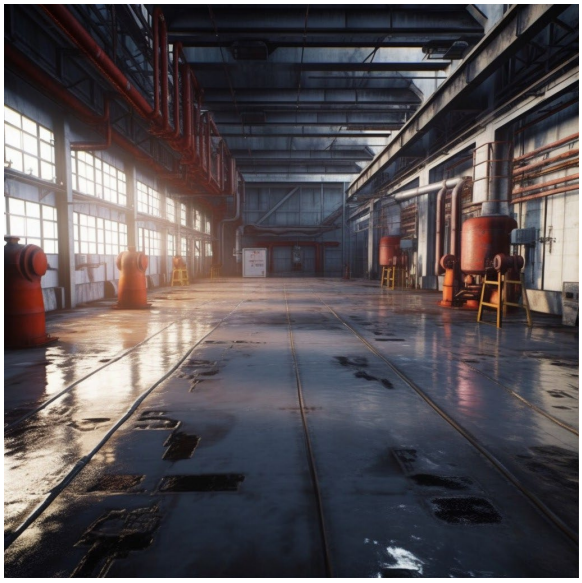
💻 The challenge

Volkswagen digital:hub has been tasked with an essential project. We need to create an application that helps control innovative autonomous cleaning robots at the Volkswagen Wolfsburg Factory. Volkswagen has introduced state-of-the-art robotic cleaners that can efficiently clean the factory floor and utilize their cameras to inspect the environment, identifying any issues or hazards in the workspaces. Our autonomous cleaning robots are truly impressive!

Our impressive robot cleaners

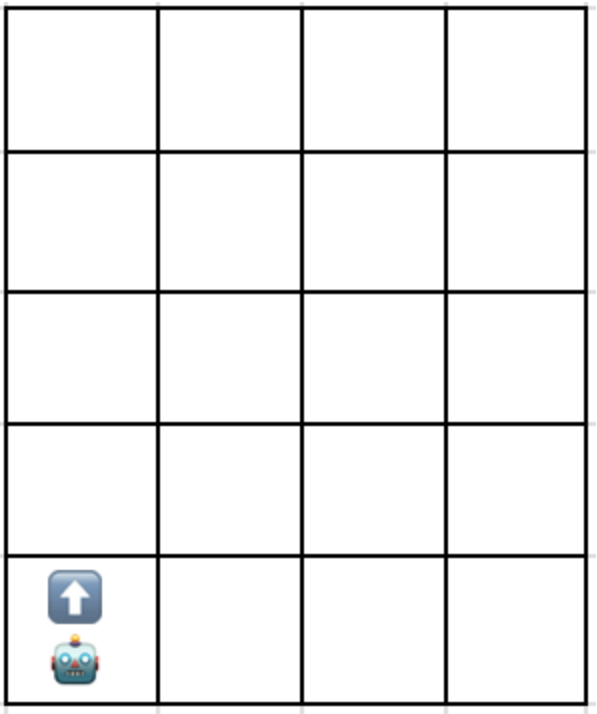
For the pilot test of our cleaning robot, we have selected a rectangular factory floor that is currently vacant and free of obstacles. This floor will provide an ideal testing

ground for our robot, allowing us to evaluate its capabilities in a controlled environment. The robot will be tasked with cleaning the entire floor, including hard-to-reach corners and crevices. We believe that this test will demonstrate the effectiveness of our robot's cleaning mechanism, as well as its ability to navigate around obstacles and operate autonomously. With this successful pilot test, we will be one step closer to launching our cleaning robot into the market, and providing an innovative solution to the challenges of industrial cleaning.



Here you can see the floor that the robot will be cleaning. As you can see, it's quite dirty

A cleaning robot's position and orientation is represented by its X and Y coordinates and a letter representing one of the four cardinal compass points (N, E, S, W). The workspace where the robot operates is divided up into a grid to simplify navigation. For instance, an example position of the robot could be 0, 0, N, which indicates that the robot is at the bottom-left corner and facing North.



The robot cleaner at 0,0, N

To control the robot, the Maintenance Department sends a string of instructions consisting of the letters "L", "R", and "M". The letter "L" makes the robot spin 90 degrees left, "R" makes it spin 90 degrees right, and "M" tells it to move forward one grid point in the same direction it is facing.

It is important to note that the square directly North of (X, Y) is (X, Y + 1).

The input for the robot control consists of two lines. The first line is the upper-right coordinates of the workspace, with the bottom-left coordinates assumed to be 0, 0. The second line provides information on the robots deployed. Each robot is represented by two lines. The first line specifies its position, while the second line is a series of instructions telling the robot how to explore the workspace.

Each robot operates sequentially, meaning that the second robot will start moving only after the first one has finished.

The output should provide the final coordinates and orientation of each robot.

A TEST CASE:

Sample Input:

```
5 5
1 2 N
LMLMLMLMM
3 3 E
MMRMMRMRRM
```

Sample Output:

```
1 3 N
5 1 E
```