Introduction

Hey there, welcome to the Belsimpel Hackathon! We hope you're ready for a challenge. Below you can find the assignment, but before starting, please read this introduction carefully as it contains quite some important information.

Groups

This hackathon is executed by a group of developers. When reading this document, you should already be in a group. If not, please inform a Belsimpel colleague!
Your group should consist of a mix of developers. Try to let everyone perform a task he/she likes and can handle with their skill level and knowledge. Also, think of a cool team name!

Need help?

It's completely understandable that you will need some help at a certain point. Please do not hesitate to ask your question. Multiple Belsimpel colleagues are walking around, feel free to ask them your questions. Time is short, so let's not waste it! You can recognize our colleagues, they are wearing a Belsimpel t-shirt.

Assignments

All the assignments have an overarching theme, which relates to our warehouse. It's important to note that assignment 1 is the 'base' assignment, without completing it you can't do 2 and/or 3. Also, we do not expect you to finish all the assignments. The jury will focus on assignment 1 when assessing your submissions. Completing just the first one is totally fine, 2 and 3 are more difficult. If you've finished assignment 1 and have time left, you can decide if you want to do assignment 2 or 3 (or both).

Presentation

At the end of the day, every team will present their work. Your presentation should be about 3 minutes and we will time you! If the presentation is too short, you have to improvise until the timer stops. If it's too long we will stop you! Your presentation should contain the following at least:

- A technical design in the form of a system architecture diagram. Please explain it briefly.
- A short demo. We recommend that you take a video/screen capture of the demo to prevent the 'demo effect'.

Note that you will be able to fill three minutes with the above. Here's some inspiration to fill those three minutes:

- What was the hardest to implement?
- What was easier to implement than initially thought?
- What are you the most proud of?
- What do you want to do differently next time?
- What have you learnt today?

Assignment 1

Here at Belsimpel, we've been working on our own software to manage our warehouse for quite some time. Since it's getting a bit old, we think it's time for a bit of a refresh. Therefore, the first and most important assignment of this hackathon will be to build a warehousing system that fulfils a couple of requirements. The goal is to have an MVP (minimum viable product) by the end of the day, which consists of a back-end and a database. Building a front-end is allowed, but not required.

We will provide you with a few things to get you started:

- A small sample dataset of products in the form of a JSON file.
    - This set will be a subset of all product variations that the system should be able to store.
- A big JSON dataset containing a large amount of products.
- A docker setup with a database.
- A Python script to generate a random order.

Implementing the Structure

In the small sample set, you will find several sample products. These products reflect all possible products that can exist in the large sample set. A product will always contain at least a description and an EAN. It's up to you to figure out what other possible properties are important to store. We expect a visualisation of the devised structure by the end of the day. It is necessary to have this structure implemented for the following requirements.

Storing Incoming Products

After implementing a structure, it's time to fill the warehouse with products. The large sample set contains all the products that need to be stored. Come up with an efficient way to store them in your warehouse.

Collecting Orders

Perhaps the most important task is actually gathering products for an order. Your system should be ready to receive an order. It's important to prove that you 'gathered' the products in the order from your warehouse. How you prove that is up to you, a nice way could be to return the locations of the products. If you would be in a real situation then, you can start walking through your warehouse to pick-up the products from those locations. You can test this process by using the *get_order.py* file. Just run it in a terminal and it should print a fictional order that contains a *Customer X* and some order items.

Followup assignments

Assignment 2

There will be times where a customer wants to check their order on our website. They might want to just see what they have ordered. To be able to facilitate this, the 2nd assignment is to implement the ability to store received orders, and to couple the products that have been sent to them. This also means that you will have to be able to keep track of products that are no longer present in the warehouse.

Requirements:

- Be able to store and retrieve orders per customer.
- Be able to couple products to orders.
- Be able to store products for fulfilled orders.

Assignment 3

In an ideal world, a warehouse would be a big box where you ask for something, and it is delivered to you immediately. Sadly this is not the case, and often a warehouse needs people to actually collect orders, and restock the shelves.
To facilitate this process, it is often desirable to think of a way to efficiently store the products in your warehouse, such that the people working to collect orders need to walk a minimal distance.

Requirements:

- Map the warehouse system to a physical layout.
- Be able to decide which product is placed where in the warehouse.
- Create an algorithm to decide which product goes where.