READ ME

I - Services Interface

In this project, you can find three different services, all with a different structure.

I-1 - The Employee service interface

This service provide the following methods for the employee: getTask(employeeId) to get the task assigned to him, placeProduct(itemId, width, height, depth, quantity) to place the product taking into account the dimensions, and getProductLocation(itemId) to get the position of a product in the warehouse.

I-2 – The retailer service interface

This service provides the following methods for the retailer: dispatch(RetailerInput input).

The input can be a ProcessDelivery or a ProcessPickup. So he can create a delivery or a pickup request. If he decides to send a delivery request, he will have to reference (hour; minutes; width; height; depth; quantity; productRef; warehouseId). If he decides to send a pickup request, he will have to reference (hour; minutes; warehouseRef; productRef)

As a result, the retailer will get information about his request.

1-3 – the team leader interface

This service provides the following methods for the team leader: createTask(List<String> employeesId, eventId) to assign a task to one or more employees and daylogs() to get the new day logs created by retailers.

2 – Business handling

I did not implement so many things about the business. The creation methods such as the creation of tasks by the team leader, or the pickup/deliveries day logs by the retailer are implemented but not tested.

I do not handle the algorithm about the placement of a product, given its dimensions. Currently, this method returns a default location.

3 – Design choices

I decided to choose the RPC design for the employee because he just need few information to get his work done. Basically, the employee asks the system, then does manual work, asks the system …

The document design for the retailer seemed to be more suitable. Indeed, for a delivery, he has to reference many things such as the dimensions, the quantity etc.

Finally, I decided to choose the resource design for the team leader, limitation of the messages and standardized http is more suitable for this requirement.