To: Teodoro Dela Cerna

From: Montrichard Group

Date: 2017-03-15

ORDER SCHEDULING

# Description of the task

The task ask you to develop two services to schedule orders.

The first service (“**order service**”) is a customer portal where a user can place orders for products with a quantity and a desired production start date.

The second service (“**production service**”) is a scheduling service receiving order requests from the first service and schedules them or rejects. The result is sent back to the order service which mark the order as either “Confirmed” or “Rejected”.

# UI Requirements

The order service shows a table indicating order number, product, quantity, total price (quantity \* unit price) and production start date. There is no further user action required.

The production service shows a table indicating order number, product, quantity, production start date and status. Status can either be “Waiting For Confirmation”, “Confirmed” or “Rejected”. Pagination and Order columns are nice, but not required and not the focus of this task.

The production service provides a form where the user can place a new production order. The order number can be simply incremented ORDER0001, ORDER0002, ORDER0003, etc...

# Technical specifications

The project must use the Spring Boot/Spring Cloud platform.

The Products can be fixed entities, any persistence can go to a services in-memory storage to keep it simple.

We welcome a solution which is based on asynchronous message queuing and able to handle temporary failure of either production or order service.

# Use scenario

The user connects to the order service and selects a product he/she desires and provides the according quantity and production start date. A product is made of a title, a description and a price. The user presses on “schedule” button which eventually pass the production request to the production service.

The production service checks the order and verifies if the requested production start date is available. If so a positive result is sent back which confirms the order eventually. If the date is not available a negative result is sent back rejecting the order eventually. A date is seen as available if there is no other order scheduled on it (max of 1 order can be produced per day).

The order service shows a visual list of previous placed orders with the according status (“Confirmed”/”Rejected”)

# Delivery

The tested project should be sent in as zipped git bundle including any commit history.

git bundle create fins\_order\_payment\_service.bundle –all

# Evaluation

The test is seen as passed if

* The described use case can be tested and is complete(UI + Use scenario)
* The solution can be easily deployed and build on our local development machines (Linux/Windows Java8)
* The candidate can explain his solution in the review interview