

**Business Process Automation Lab Demonstrator 2**

(Guided Project)

**Documentation for Order Management & Production Control Pools**

in Winter 2023/24

**TH Köln - Campus Gummersbach**

Faculty of Computer Science and Engineering Science (F10)

**Project by:** Rahib Nazir Butt (11155035)

Berrak Kücük (11160144)

**A picture containing logo

Description automatically generatedProject Supervisor:** Prof. Dr. Matthias Zapp

**Date:** 1st of February, 2024

**Contents**

[1. Order Management 3](#_Toc160048476)

[1.1 Store customer order (service task) 3](#_Toc160048477)

[1.2 Approve customer order (business rule task) 4](#_Toc160048478)

[1.3 Customer order status check (user task) 5](#_Toc160048479)

[1.4 Check finished product availability (service task) 5](#_Toc160048480)

[2. Production Control 7](#_Toc160048481)

[2.1 Check Components Availability (service task) 7](#_Toc160048482)

[3. Test Cases 9](#_Toc160048483)

[3.1 Single Order – Finished Product Availability 9](#_Toc160048484)

[3.2 Single Order – Finished Product Not Available – Components Available 9](#_Toc160048485)

[3.3 Single Order – Finished Product Not Available – Components Not Available 10](#_Toc160048486)

[3.4 Single Order – Finished Product Not Available – Components Not Available 11](#_Toc160048487)

[3.5 Multiple Order – Finished Product Available 11](#_Toc160048488)

[3.6 Multiple Order – Quantity more than 5 12](#_Toc160048489)

[4. Further Developments 13](#_Toc160048490)

[4.1 Checking multiple components 13](#_Toc160048491)

[4.2 Improving security including authentication 13](#_Toc160048492)

[4.3 AI implementation 13](#_Toc160048493)

# Order Management

Order Management pool handles the incoming customer orders. Process starts with receiving the order from the front-end application.

A diagram of a computer flowchart

Description automatically generated

Figure 1.1 Order Management

A screenshot of a web page

Description automatically generated

Figure 1.2 Front-end application

As it can be seen in figure 1.1, there are many tasks and interactions with databases in this pool. List of the tasks and their purpose are listed below.

## Store customer order (service task)

Customer orders are stored in the customer\_DB database, inside of customer\_order table. Each order has a unique id, product, and quantity, as well as customer information.

A diagram of a customer order

Description automatically generated

Figure 1.3 Store customer order service task



Figure 1.4 customer\_order table

## Approve customer order (business rule task)

Depending on the quantity in the customer order, we categorize the order as either “single order” or “multiple order”. If the quantity is greater than 5, than we cancel this order. In the first version, the decision was made with the DMN tables connected to each other, creating a DRD (Decision Requirements Diagrams). This can be seen in the figure 1.5. But we decided the complexity of this decision is not high, and we don’t need to use DRD, a simple DMN table is enough (figure 1.6).

A screenshot of a computer

Description automatically generated

Figure 1.5 DRD (not in use)

A screenshot of a computer

Description automatically generated

Figure 1.6 DMN Table

## Customer order status check (user task)

Here, the customer order can be viewed as a form, technically to slow down the process and see the order details together. This form takes the data from the customer order automatically, so there is no need to change or edit any field.

A screenshot of a computer

Description automatically generated

Figure 1.7 Customer order form

After the form, we have a gateway to check if the customer order is approved or not. If it is not approved, then we update the customer order status as ORDER\_REJECTED and send a rejection email to the customer. If the order is approved, then we send a confirmation email to the customer that the order is received, and update customer order status as ORDER\_APPROVED. After this step, we have to check if we have the product available.

## Check finished product availability (service task)

We check from the finished\_product\_DB database if we have the ordered product available. Inside of the finished\_product\_stock, we have the product list with the available quantity.

A screenshot of a computer

Description automatically generated

Figure 1.8 finished\_product\_stock table

Depending on the stock, we must go to the production or shipment step. Therefore we have another gateway. When we have enough stock, we send the customer order for shipment. This path connects with the Shipment pool by Send Task and Receive Task, which can be seen in figure 1.9. After shipment is done, customer order status is updated, and the process ends successfully since the order is shipped to the customer.

A diagram of a product production process

Description automatically generated

Figure 1.9

In the case that we don’t have enough stock, we update the order status as ORDER\_PRODUCTION\_REQUIRED and send the customer order for production. This continues in a different pool, “Production Control”, which will be explained in the second part. After the production is completed, we receive the finished production order and again continue with the shipment part.

# Production Control

A diagram of a flowchart

Description automatically generated

Figure 2.1

In the Production Control pool, process starts with receiving a production order from the Order Management pool. To see the production order before the process runs too fast, we have a User Task “Create production order”, which is a Camunda pre-filled form. Then we store the production order inside of production\_order\_DB database, as seen in the figure 2.2

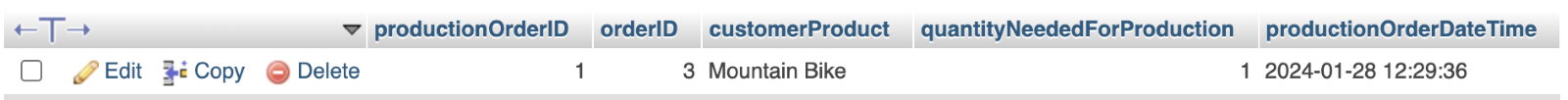


Figure 2.2 production\_order table

After updating the customer order status as “ORDER\_IN\_PRODUCTION”, we need to check the component availability in order to continue with the production. We have available components listed inside component\_DB database, and if there are missing components, we need to go to Purchasing. After purchasing is done, or if there is no need for purchasing, we start the production process. The implementation with the production robots is not done in this version, we have a gateway for checking the production status. Once it is completed, it sends the order to the distribution warehouse pool and prepares the finished product. Finally, the finished product is sent back to the Order Management Pool to be shipped to the customer.

## Check Components Availability (service task)

Inside of component\_DB database, we have the table component\_stock with the list of components and their quantities.

A screenshot of a computer

Description automatically generated

Figure 2.3 component\_stock table

This service task takes the most recent production order from production\_order table (figure 2.2), and according to the name of the order, looks for the component. If the order is Mountain Bike, it will look for Mountain bike frame, or if it is Hybrid 40000 Bicycle, it will look for Hybrid bicycle wheels. If there is no stock of the component, it will go to Purchasing with the values like in figure 2.4 and when the stock is available it will go to Send Production Order by going through the gateway with “No”.

A computer screen shot of a computer code

Description automatically generated A screenshot of a computer program

Description automatically generated

Figure 2.4 Component Availability results

# Test Cases

## Single Order – Finished Product Availability

Selected bicycle type: Mountain Bike

Selected quantity: 1

Is there stock in the finished product database: Yes

Preconditions: Inside of the finished\_product\_stock table, there should be enough (in this case at least 1) productQuantity.

Expected results: Followed path should be

Order approved? - Yes

Order production required? - No

Goes to Shipment

Customer order status is updated

Process ends successfully with “Order shipped successfully”

Actual results: Same as the expected

Status: Passed

Notes: -

## Single Order – Finished Product Not Available – Components Available

Selected bicycle type: Mountain Bike

Selected quantity: 1

Is there stock in the finished product database: No

Is there stock in the component database: Yes

Preconditions: Inside of the finished\_product\_stock table, there shouldn’t be enough (in this case 0) productQuantity. Component of the selected bicycle type should have enough quantity (in this case Mountain bike frame, at least 1 quantity).

Expected results: Followed path should be

Order approved? - Yes

Order production required? - Yes

Customer order status updated (order production required)

Goes to Production Control

Production order is stored in the database

Purchasing required? - No

Production status – Completed

Production order completed

Goes back to Order Management

Goes to Shipment

Customer order status is updated

Process ends successfully with “Order shipped successfully”

Actual results: Same as expected

Status: Passed

Notes: -

## Single Order – Finished Product Not Available – Components Not Available

Selected bicycle type: Mountain Bike

Selected quantity: 1

Preconditions: Inside of the finished\_product\_stock table, there shouldn’t be enough (in this case 0) productQuantity. Component of the selected bicycle type also should not have enough quantity (in this case Mountain bike frame, and 0 quantity).

Expected results: Followed path should be

Order approved? - Yes

Order production required? - Yes

Customer order status updated (order production required)

Goes to Production Control

Production order is stored in the database

Purchasing required? - Yes

Goes to Purchasing

Received back from Purchasing

Production status – Completed

Production order completed

Goes back to Order Management

Goes to Shipment

Customer order status is updated

Process ends successfully with “Order shipped successfully”

Actual results: Same as expected

Status: Passed

Notes: -

## Single Order – Finished Product Not Available – Components Not Available

Selected bicycle type: Hybrid 40000 Bicycle

Selected quantity: 1

Is there stock in the finished product database: No

Is there stock in the component database: No

Preconditions: Inside of the finished\_product\_stock table, there shouldn’t be enough (in this case 0) productQuantity. Component of the selected bicycle type also should not have enough quantity (in this case Hybrid bicycle wheels, and 0 quantity).

Expected results: Followed path should be

Order approved? - Yes

Order production required? - Yes

Customer order status updated (order production required)

Goes to Production Control

Production order is stored in the database

Purchasing required? - Yes

Goes to Purchasing

Received back from Purchasing

Production status – Completed

Production order completed

Goes back to Order Management

Goes to Shipment

Customer order status is updated

Process ends successfully with “Order shipped successfully”

Actual results: Same as expected

Status: Passed

Notes: Logically, the amount for the purchase should be 2 instead of 1, since it is wheels. But this logic is not implemented and we send the amount to Purchasing as 1. It could be also thought of a pair of wheels, or it can be updated as “when it is wheels, multiply by 2”.

## Multiple Order – Finished Product Available

Selected bicycle type: Mountain Bike

Selected quantity: 3

Is there stock in the finished product database: Yes

Preconditions: Inside of the finished\_product\_stock table, there should be enough (in this case at least 3) productQuantity.

Expected results: Followed path should be

Order approved? - Yes

Order production required? - No

Goes to Shipment

Customer order status is updated

Process ends successfully with “Order shipped successfully”

Actual results: Same as the expected

Status: Passed

Notes: -

## Multiple Order – Quantity more than 5

Selected bicycle type: Mountain Bike

Selected quantity: More than 5

Is there stock in the finished product database: Yes

Preconditions: -

Expected results: Order should be rejected and followed path should be

Order approved? - No

Customer order status updated as “order rejected”

Process ends as “Customer order canceled”

Actual results: Same as the expected

Status: Passed

Notes: A solution for the Send Rejection Email task exists that triggers and generates an email to be sent to the customer regarding the current status of their order.

# Further Developments

## Checking multiple components

In the beginning we planned 3 components for each of the bicycles, however in this version, we are checking for one component for each bicycle. This can be improved by using a loop where all the different components are checked, and purchasing of the multiple components could be done.

## Improving security including authentication

One of the improvements that could be done in terms of security is securing the access to the databases with a better authentication protocol. One of the protocols that we discussed was the Oauth which is an open standard used nowadays for the internet users of accessing any application or a server.

## AI implementation

As you might be aware of how AI is progressing these days, we really thought of implementing AI in one of our processes. For example, implementing a chatbot somewhere in our factory model. Let’s say if we have a huge list of products, we could implement ChatGPT by OpenAI using the connector feature in order to generate good product descriptions after giving a specific prompt to the ChatGPT engine.

-----END-----