|  |
| --- |
| **Design Specification**  **Version 1.0** |

**NI E-shop**

July, 2019

Table of Contents

[Introduction](#_d8ip8ndlhvk4)

[Purpose](#_p4csedgbvwds)

[Architecture Overview](#_o4dnd2shhigl)

[API Gateway](#_t6qqyqc8mkop)

[Overview:](#_7y5884op4i2g)

[Design Considerations](#_touo3zg6xxz4)

[Service Registry](#_8p3ivgoklczh)

[Overview:](#_biqc3x5uxcb7)

[Design Considerations](#_bsx4k2of26o7)

[Auth Service](#_v49flz8dqys4)

[Overview:](#_8w8xot95r034)

[Design Considerations](#_vj1c6osgahdq)

[Out of scope](#_t0fyzs942i70)

[Service Description](#_535whqsn02hz)

[Request](#_40060wdmh0d9)

[Response](#_j4yqvw4khuwg)

[Sample](#_fo717khyxzj0)

[Database](#_3f2itt27t3e)

[Order Service](#_3dbq2dro2gom)

[Overview:](#_suf2td38fq7w)

[Service Architecture Overview](#_lf314zvm54xu)

[Design Considerations](#_7quhq92en5oo)

[Out of scope](#_kqlsgrunm89i)

[Database Schema](#_7q9dsqtgvlg3)

[Add Order](#_pp18np1m8t4n)

[Service Description](#_gueopdduc9pj)

[Request](#_t5qvshje49b3)

[Response](#_5up8pjad3ke)

[Sample](#_r2ekrfv8nkxm)

[Service Design](#_iazfn19ys8iu)

[Sequence Diagram](#_wcojb8nxnyzr)

[Get Order](#_5m5ytawn2h2l)

[Service Description](#_u8almzrhob5g)

[Request](#_7s4mk6senid2)

[Response](#_ca9ol9a6ye6z)

[Sample](#_nyls00t4i0og)

[Service Design](#_lqg350vqsnba)

[Sequence Diagram](#_o43kjogeucus)

[Product Service](#_545i4qxri3he)

[Overview:](#_g9gr3jwvka3k)

[Service Architecture Overview](#_qt8uuka12otx)

[Service Design](#_b7vj4j4p1sbx)

[Sequence Diagram](#_g850ycqktlwf)

[Design Considerations](#_tdtri51ulxtk)

[Out of scope](#_ck1d8bunfl1i)

[Add Product](#_z6xpg4aos1ve)

[Service Description](#_myzdoklc5d52)

[Request](#_46kd9vmf7xcj)

[Response](#_x085kzi2bw57)

[Sample](#_uzzme9e1oya1)

[Add Products](#_mh2ohbbrbvfy)

[Service Description](#_6tdjv9gqi5z0)

[Request](#_ycdy274kylm)

[Response](#_6dx6fc24ouiw)

[Sample](#_dvmrcsbpwgi8)

[Get Product](#_5atixl1t74c9)

[Service Description](#_sdke0mjw2iqr)

[Request](#_xohns0vv2h11)

[Response](#_5gfaxtqwz225)

[Sample](#_szkdusl4k7mv)

[Get Product Name](#_k2ga34d8fnv4)

[Service Description](#_petundpvihtj)

[Request](#_32m13e304sim)

[Response](#_q9k0q76l6nvo)

[Sample](#_4tzd7l3fwfle)

[Add Items](#_w3cvd0exa449)

[Service Description](#_8sujdu2qdpha)

[Request](#_4xt4pqkhovyy)

[Response](#_53p8clh1ec3c)

[Sample](#_yoshc8rfxndk)

[Book Items](#_qhp0y8ru988l)

[Service Description](#_5r1utqen8owd)

[Request](#_70c3fnqdj04i)

[Response](#_ls3j0h61tg9a)

[Sample](#_fhw4uagbou1g)

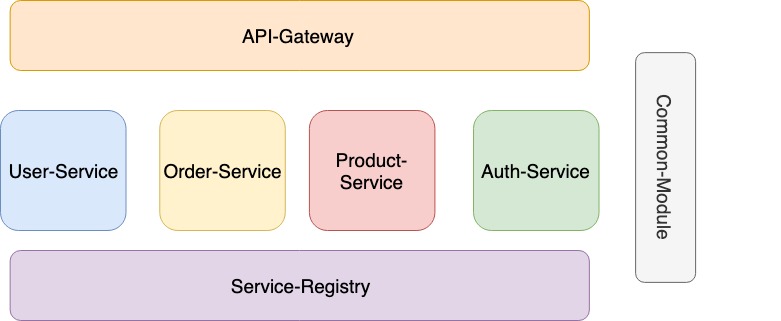
### **Introduction**

This document is designed to be a reference for any person wishing to implement or any person interested in the architecture of the NI E-shop. This document describes project architecture, and section for each service

### **Purpose**

This project implement the NI eshop service

### **Architecture Overview**



### **Design Considerations**

#### Out of scope

This design of the NI-eshop makes several assumptions about software..

* Service availability will be managed in the deployment tool (Ex. Kubernete) by having multiple instances of each service (Ex. Docker container) based on the users access load for the service.

## 

## **API Gateway**

### **Overview:**

It acts as the single entryway into a system allowing multiple APIs or microservices to act cohesively and provide a uniform experience to the user. And provide authentication and authorization services.

### **Design Considerations**

**Technologies**: ZUUL

## **Service Registry**

### **Overview:**

It supports the run-time discovery and evaluation for microservices

### **Design Considerations**

**Technologies**: eureka

## **Auth Service**

### **Overview:**

It implement authentication mechanism using auth protocol

### **Design Considerations**

Technologies Spring

#### Out of scope

This design of the Auth microservice makes below assumptions about software..

* User authenticate the user against in memory list contains 2 users as for simulate users directory

### **Service Description**

#### Request

|  |  |  |
| --- | --- | --- |
| http://Url/auth/login | | |
| Type | Post | |
| **Parameters** | **Type** |  |
| username | String | Mandatory |
| password | String | Mandatory |

#### Response

JWT token in header

#### Sample

**Request**: http://localhost:8762/auth/login

{"username":"admin", "password":"12345"}

**Response**:

Authorization →Bearer eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJhZG1pbiIsImF1dGhvcml0aWVzIjpbIlJPTEVfQURNSU4iXSwiaWF0IjoxNTYzMjE5ODgzLCJleHAiOjE1NjMzMDYyODN9.KSGgW0WzBJTPxEh6ZEnC99uqFTB8u0LPZXyIhAEB7cbm1Z09WvcNT5Pqiu-z30oN96LDVnhmAsR-dtLAM\_iPUg

### **Database**

In memory list

{"username":"admin", "password":"12345" , role:”ADMIN”}

{"username":"user", "password":"12345", “role”:”USER”}

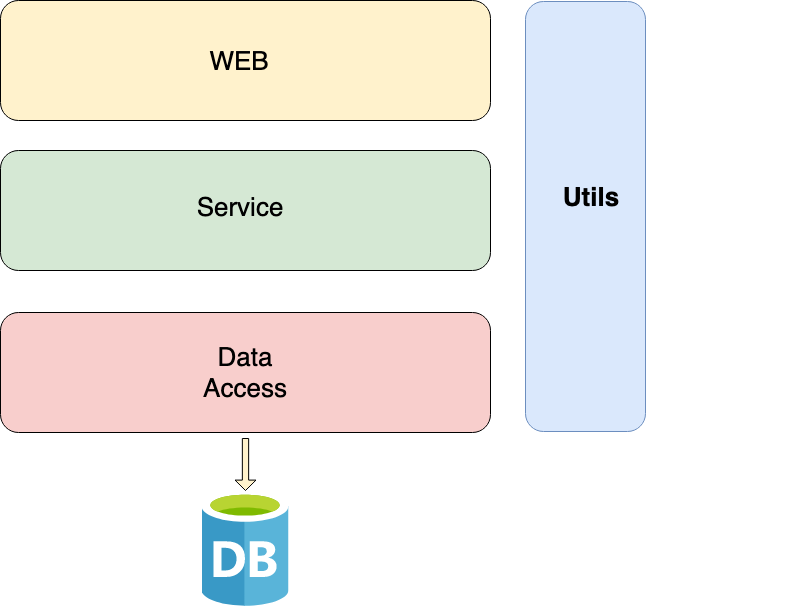
## **Order Service**

### **Overview:**

It is responsible for handling order transactions

* Add order
* Get order by id

### **Service Architecture Overview**



### **Design Considerations**

Technologies Spring, in memory db h2

#### Out of scope

This design of the order microservice makes below assumptions about software..

* Not all order services api implemented

### 

### 

### 

### **Database Schema**

|  |
| --- |
| NiOrder |
| id  productId  itemCount |
|
|
|
|
|
|
|
|
|
|

### **Add Order**

### **Service Description**

#### Request

|  |  |  |
| --- | --- | --- |
| http://localhost:8762/order/add | | |
| Type | Put | |
| **Parameters** | **Type** |  |
| productId | String | Mandatory |
| itemCount | String | Mandatory |

#### Response

|  |  |  |
| --- | --- | --- |
|  | | |
| Type | String | |
| **Parameters** | **Type** |  |
| orderId | Numeric | Mandatory |

#### 

#### Sample

**Request**: <http://localhost:8762/order/add>

{"productId":"1","itemCount":"4"}

**Response**: Status 200

1

### **Service Design**

#### Sequence Diagram

### 

### 

### 

### **Get Order**

### **Service Description**

#### Request

|  |  |  |
| --- | --- | --- |
| http://localhost:8762/order/get/{id} | | |
| Type | Get | |

#### Response

|  |  |  |
| --- | --- | --- |
|  | | |
| Type | JSON | |
| **Parameters** | **Type** |  |
| id | Numeric | Mandatory |
| productId | String | Mandatory |
| productName | String | Mandatory |
| itemCount | String | Mandatory |

#### Sample

**Request**: http://localhost:8762/order/get/1

**Response**: Status 200

{

"id": 1,

"productId": 1,

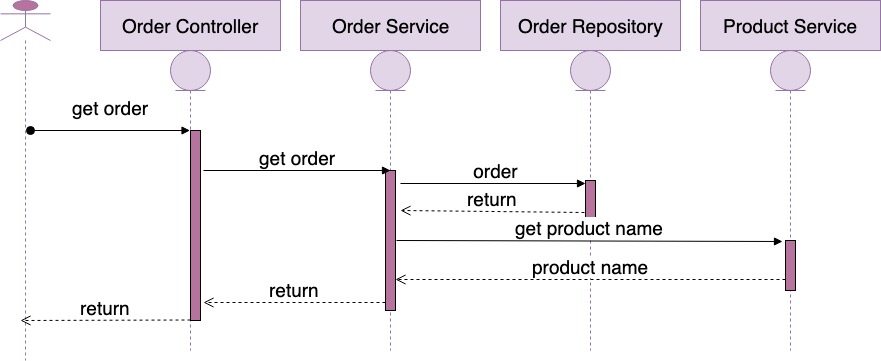
"productName": "test",

"itemCount": 4

}

### **Service Design**

#### Sequence Diagram



## 

## **Product Service**

### **Overview:**

It is responsible for handling product transactions

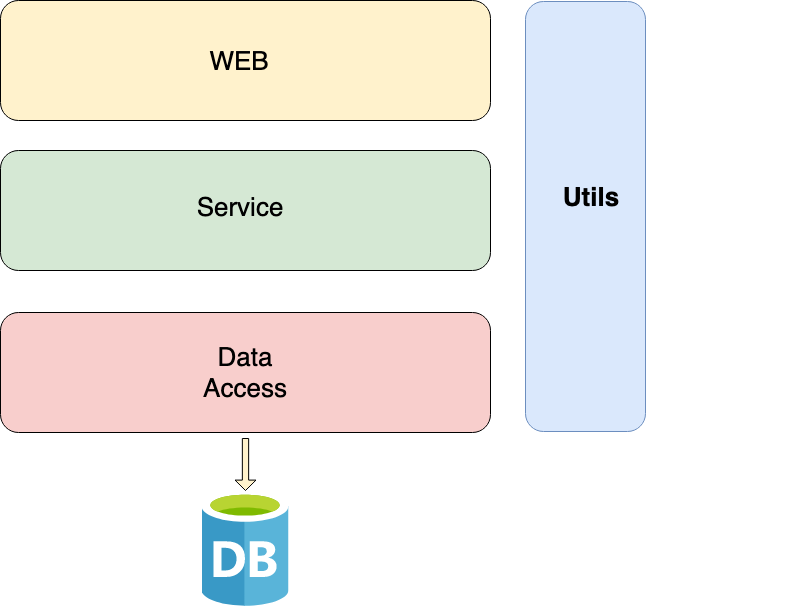
* Add product
* Add products
* Get product by id
* Get product name by id
* Add items for product
* Book product items

### **Service Architecture Overview**

### **Service Design**

#### Sequence Diagram

### 



### 

### **Design Considerations**

Technologies Spring, embedded mongo db

#### Out of scope

This design of the order microservice makes below assumptions about software..

* Not all product service apis implemented

### **Add Product**

### **Service Description**

#### Request

|  |  |  |
| --- | --- | --- |
| http://url/product/add | | |
| Type | Put | |
| **Parameters** | **Type** |  |
| name | String | Mandatory |
| stock | String | Mandatory |

#### Response

|  |  |  |
| --- | --- | --- |
|  | | |
| Type | String | |
| **Parameters** | **Type** |  |
| Product id | Numeric | Mandatory |

#### 

#### Sample

**Request**: <http://localhost:8762/product/add>

{"name":"test3", "stock":"46"}

**Response**: Status 200

1

### **Add Products**

### **Service Description**

#### Request

|  |  |  |
| --- | --- | --- |
| http://url/product/addProducts | | |
| Type | Put | |
| **List** | | |
| **Parameters** | **Type** |  |
| name | String | Mandatory |
| stock | String | Mandatory |

#### Response

|  |  |  |
| --- | --- | --- |
|  | | |
| Type | String | |
| **Parameters** | **Type** |  |
| count | Numeric | Mandatory |

#### 

#### Sample

**Request**: http://localhost:8762/product/addProducts

[{"name":"test", "stock":"46"},{"name":"test4", "stock":"46"},{"name":"1ss", "stock":"46"}]

**Response**: Status 200

3

### **Get Product**

### **Service Description**

#### Request

|  |  |  |
| --- | --- | --- |
| http://url/product/get/{id} | | |
| Type | Get | |

#### Response

|  |  |  |
| --- | --- | --- |
| List | | |
| Type | JSON | |
| **Parameters** | **Type** |  |
| id | Numeric | Mandatory |
| name | String | Mandatory |
| stock | String | Mandatory |

#### Sample

**Request**: http://localhost:8762/order/get/1

**Response**: Status 200

{

"id": 1,

"name": "test",

"stock": 42

}

### **Get Product Name**

### **Service Description**

#### Request

|  |  |  |
| --- | --- | --- |
| http://url/product/getName/{id} | | |
| Type | Get | |

#### Response

|  |  |  |
| --- | --- | --- |
|  | | |
| Type | String | |
| **Parameters** | **Type** |  |
| name | String | Mandatory |

#### Sample

**Request**: http://localhost:8762/order/getName/1

**Response**: Status 200

test

### **Add Items**

### **Service Description**

#### Request

|  |  |  |
| --- | --- | --- |
| http://url/product/addItem/{id}/{count} | | |
| Type | Post | |

#### Response

|  |  |  |
| --- | --- | --- |
|  | | |
| Type | Boolean | |
| **Parameters** | **Type** |  |
|  | boolean | Mandatory |

#### 

#### Sample

**Request**: http://localhost:8762/product/addItem/1/5

**Response**: Status 200

True

### **Book Items**

### **Service Description**

#### Request

|  |  |  |
| --- | --- | --- |
| http://url/product/bookItem/{id}/{count} | | |
| Type | Post | |

#### Response

|  |  |  |
| --- | --- | --- |
|  | | |
| Type | Boolean | |
| **Parameters** | **Type** |  |
|  | boolean | Mandatory |

#### 

#### Sample

**Request**: http://localhost:8762/product/bookItem/1/5

**Response**: Status 200

True

### 

### 

### 