UNIVERSITÀ DEGLI STUDI DELL'AQUILA

DISIM
Dipartimento di Ingegneria e Scienze dell'Informazione e Matematica

Final project assignment

A.A. 2022/2023

Business Process Development

Scenario description

Public billposting service

It is required to realize a system that allows users to book spaces for public billposting.

1. The user specifies the cities (one ore more) where she wants to affix posters, the poster format (e.g., 60x80, 200x100, etc.), and the maximum price that can be paid for each of the selected cities. Those information are sent through an API.
2. After having received the message from the exposed API, the process gets the user details (user personal data) and the list of the available zones of the cities where posters can be affixed (with the prices) from two web services.
3. For each chosen city, zones need to be selected in such a way that their total price is lower than the maximum price. Zones are chosen by taking the most expensive ones until the maximum price is reached.
4. A posting service is invoked sending the list of the selected zones and the user details. It will return a request ID.
5. The request ID and the list of the selected zones are returned to the user

Scenario description

6. The user sends a message with the request ID and the decision: "confirm" or "cancel"

If the decision is "confirm":

7. The confirmation is sent to the posting service. The service will return the billing information with the total amount due.
8. The order information are printed into a file
9. The billing information are returned to the user

If the decision is "cancel":

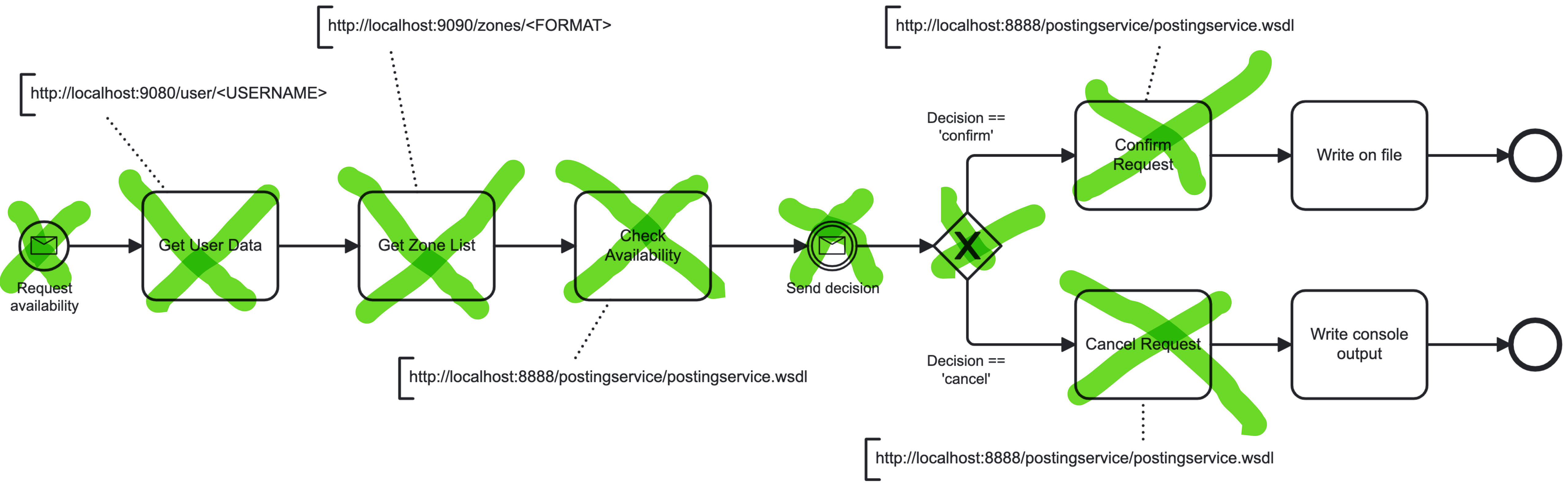
7. The order cancelation is sent to the posting service
8. A cancelation notice is printed in the standard output
9. An empty bill is returned to the user

Project Requirements

Students are required

- To model a BPMN process realizing the described scenario
- The BPMN process must be executed through the Camunda platform embedded into a Spring Boot project
- The project must expose two API endpoints (REST or SOAP)
- The API endpoints must interact with the process and start/resume its execution through messages
  - Messages must be correlated
- Communication with external services must be implemented by using connectors
- Extra tasks may be added to perform extra computation (e.g., zones selection)

Example process


http://localhost:9080/user/<USERNAME>
http://localhost:9090/zones/<FORMAT>
http://localhost:8888/postingservice/postingservice.wsdl
Request availability
Get User Data
Get Zone List
Check Availability
Send decision
Decision == 'confirm'
Confirm Request
Write on file
Decision == 'cancel'
Cancel Request
Write console output
http://localhost:8888/postingservice/postingservice.wsdl
http://localhost:8888/postingservice/postingservice.wsdl

APIs to expose

- Can be realized as REST or SOAP services
- If REST, methods (GET, POST, etc.) can be chosen according to what is the best-fitting ...choose wisely ;)

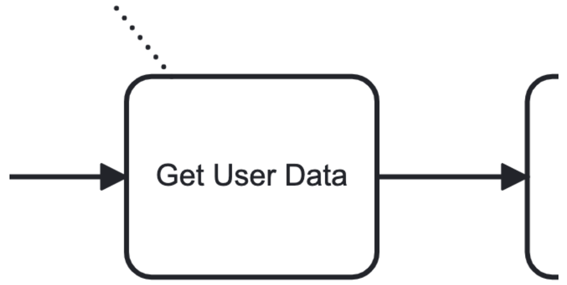
Request availability

- Input data:
  - Username
  - List of cities
  - Maximum price for each of the listed cities
  - Poster format
- Output data:
  - List of selected zones
  - Total price
  - Request ID

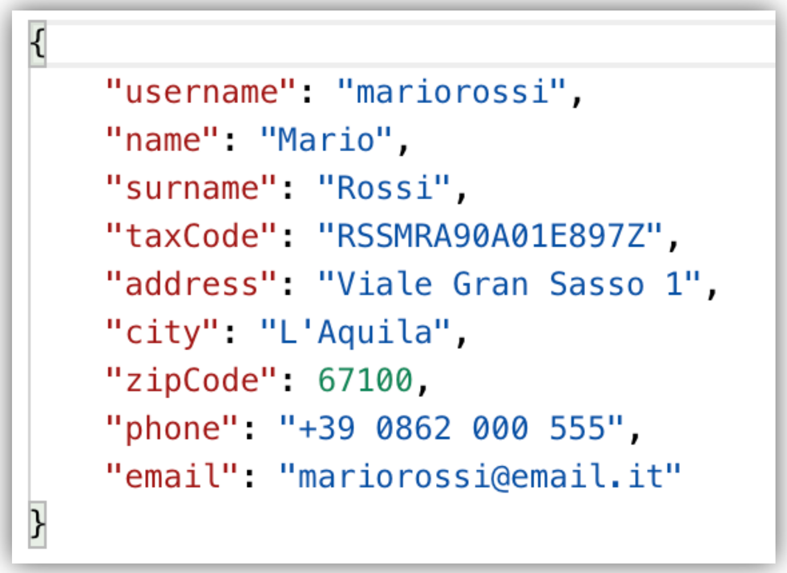
Send decision

- Input data:
  - Request ID
  - Decision ("confirm" or "cancel")
- Output data:
  - Billing information

Process details



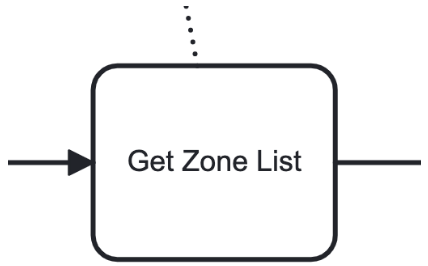
- REST Service endpoint: [GET] http://localhost:9080/user/<USERNAME>
- Example Request
  - [GET] http://localhost:9080/user/mariorossi
- Response



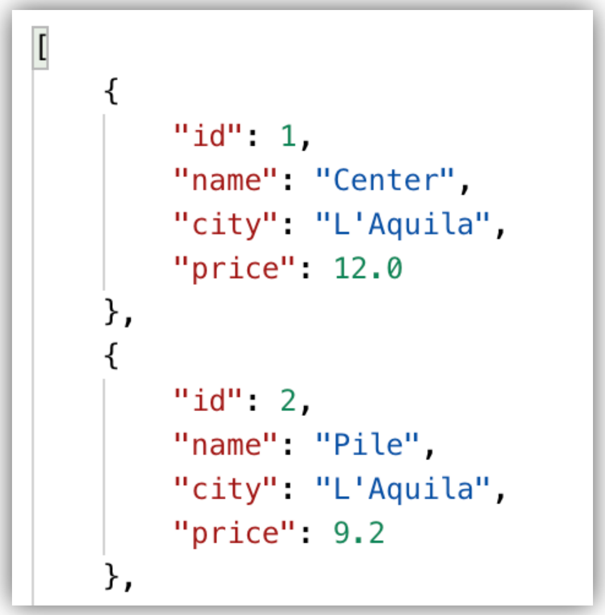
```
{
    "username": "mariorossi",
    "name": "Mario",
    "surname": "Rossi",
    "taxCode": "RSSMRA90A01E897Z",
    "address": "Viale Gran Sasso 1",
    "city": "L'Aquila",
    "zipCode": 67100,
    "phone": "+39 0862 000 555",
    "email": "mariorossi@email.it"
}
```

- Available usernames:
  - mariorossi
  - sarabianchi
  - johndoe

Process details



- REST Service endpoint: [GET] http://localhost:9090/zones/<FORMAT>
- Example Request
  - [GET] http://localhost:9090/zones/60x80
- Response



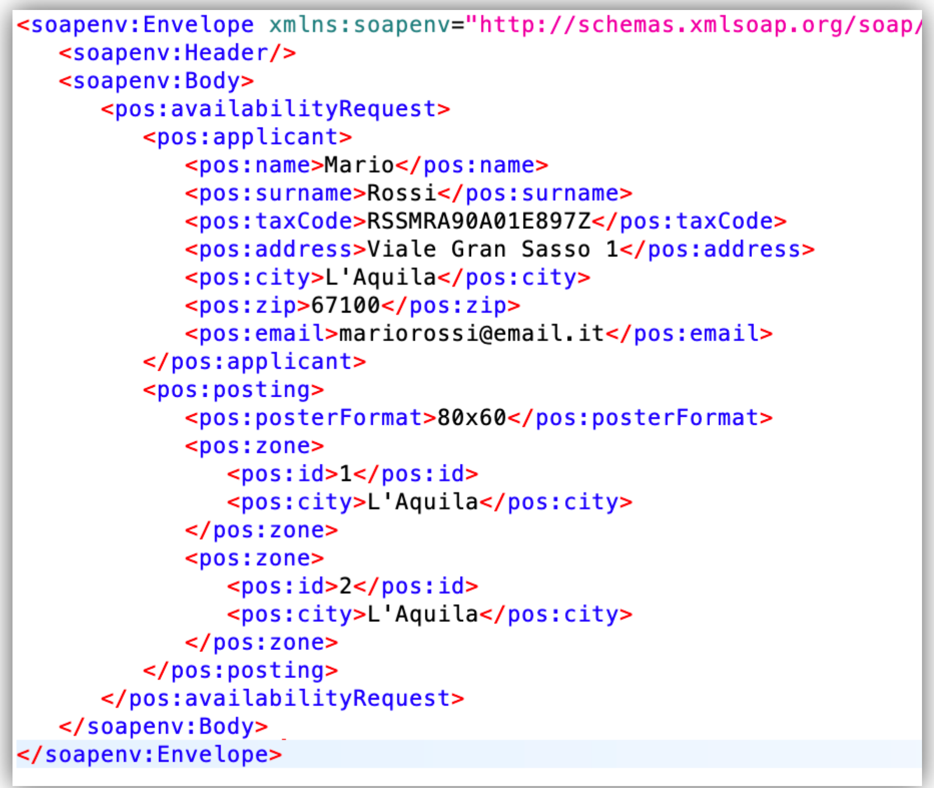
```
[
    {
        "id": 1,
        "name": "Center",
        "city": "L'Aquila",
        "price": 12.0
    },
    {
        "id": 2,
        "name": "Pile",
        "city": "L'Aquila",
        "price": 9.2
    },
```

- This service is mocked
  - It returns the same list for any specified format
- Cities: L'Aquila, Rome, Milan

Process details

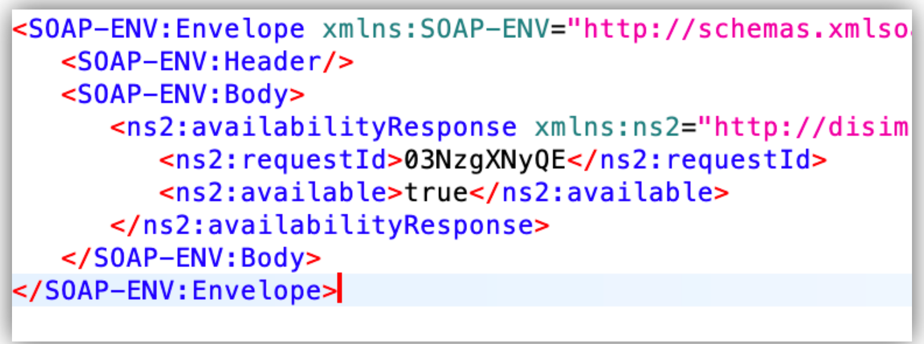
Check Availability

- WSDL location: http://localhost:8888/postingservice/postingservice.wsdl
- Example request



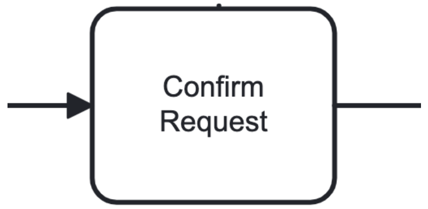
```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/
   <soapenv:Header/>
   <soapenv:Body>
      <pos:availabilityRequest>
         <pos:applicant>
            <pos:name>Mario</pos:name>
            <pos:surname>Rossi</pos:surname>
            <pos:taxCode>RSSMRA90A01E897Z</pos:taxCode>
            <pos:address>Viale Gran Sasso 1</pos:address>
            <pos:city>L'Aquila</pos:city>
            <pos:zip>67100</pos:zip>
            <pos:email>mariorossi@email.it</pos:email>
         </pos:applicant>
         <pos:posting>
            <pos:posterFormat>80x60</pos:posterFormat>
            <pos:zone>
               <pos:id>1</pos:id>
               <pos:city>L'Aquila</pos:city>
            </pos:zone>
            <pos:zone>
               <pos:id>2</pos:id>
               <pos:city>L'Aquila</pos:city>
            </pos:zone>
         </pos:posting>
      </pos:availabilityRequest>
   </soapenv:Body>
</soapenv:Envelope>
```

- Example response



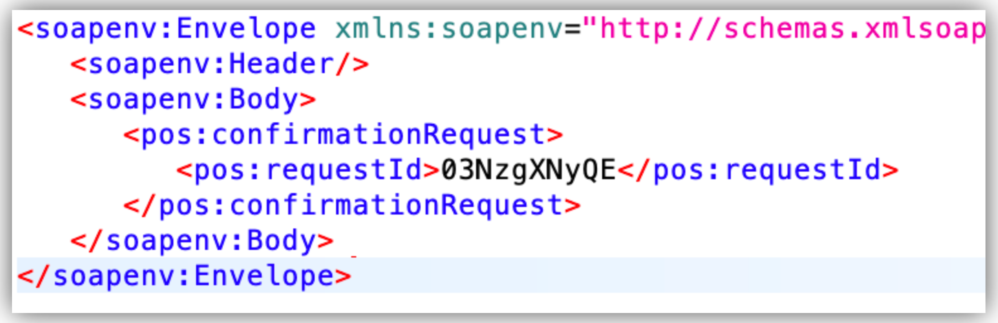
```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlso
   <SOAP-ENV:Header/>
   <SOAP-ENV:Body>
      <ns2:availabilityResponse xmlns:ns2="http://disim
         <ns2:requestId>03NzgXNyQE</ns2:requestId>
         <ns2:available>true</ns2:available>
      </ns2:availabilityResponse>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Process details



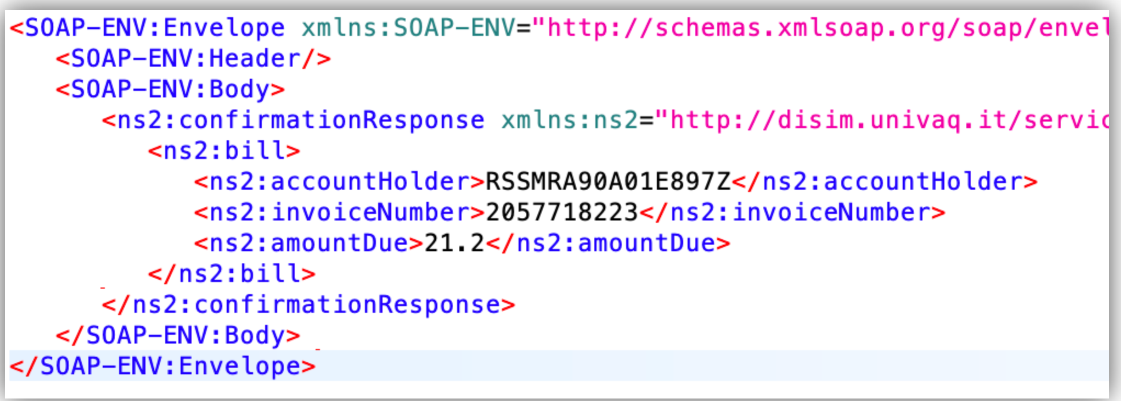
- WSDL location: http://localhost:8888/postingservice/postingservice.wsdl

- Example request



```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap
   <soapenv:Header/>
   <soapenv:Body>
      <pos:confirmationRequest>
         <pos:requestId>03NzgXNyQE</pos:requestId>
      </pos:confirmationRequest>
   </soapenv:Body>
</soapenv:Envelope>
```

- Example response



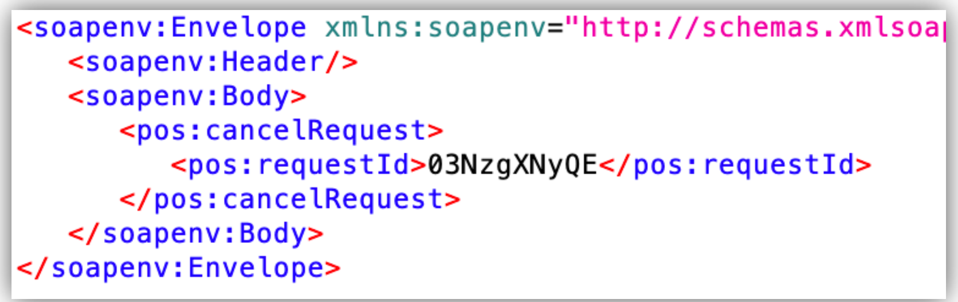
```xml
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envel
   <SOAP-ENV:Header/>
   <SOAP-ENV:Body>
      <ns2:confirmationResponse xmlns:ns2="http://disim.univaq.it/servic
         <ns2:bill>
            <ns2:accountHolder>RSSMRA90A01E897Z</ns2:accountHolder>
            <ns2:invoiceNumber>2057718223</ns2:invoiceNumber>
            <ns2:amountDue>21.2</ns2:amountDue>
         </ns2:bill>
      </ns2:confirmationResponse>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Process details

Cancel Request

- WSDL location: http://localhost:8888/postingservice/postingservice.wsdl

- Example request



```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoa
   <soapenv:Header/>
   <soapenv:Body>
      <pos:cancelRequest>
         <pos:requestId>03NzgXNyQE</pos:requestId>
      </pos:cancelRequest>
   </soapenv:Body>
</soapenv:Envelope>
```

- The response is empty (if the request is acceptable)

Process details

Write on file

- Add a new line into a file named "posting_requests.txt"
  - Create the file if it does not exist
  - Line format: *username*, *request ID*, *invoice number*, *amount due*;

- Example:

  mariorossi, 03NzgXNyQE, 2057718223, 21.2;
  sarabianchi, 7uu2kCpFPX, 6390163826, 89.9;

Services to compose

- Available as *.jar files in the Team class material
  - user-service.jar, zones-service.jar, posting-service.jar
- Run from terminal with java -jar command
  - Es.: java -jar user-service.jar
- Java 8 required
  - To run the services
  - To run the Camunda engine with Javascript tasks