

AppsFactory Insurance Premium Calculation Project

The purpose of this technical document is to write down the steps required to run the Demo project developed in Spring boot with Java 11.

The zipped file named “premium-insurance-calculation-project” can be downloaded from the link file section provided in the email.

Link: <https://drive.google.com/drive/folders/1-x6wFeLwdd4Bk9eYdFFm3cesaC3a8yma?usp=sharing>

Download the file and extract all the folders to your destination folder.

The folders contain the following the project.

1. **api-gateway**
2. **naming-server**
3. **location-services**
4. **auto-services**
5. **mileage-services**
6. **premium-insurance**

Prerequisites (System Environment)

- Java 11
- Maven

(Optional - Not done)

- a) Unfortunately, the docker-compose file is not setup to run the project easily and make it platform independent as it is not mentioned in the task requirement.
- b) The User Interface is not designed to calculate the Premium Insurance as it was an optional task.
- c) Database's setup is not required as In-Memory H2 databases is used in the demo project.

Start Up Spring Boot Applications

Run Naming-Server

Our task is to run each project individually. Therefore, go the destination folder where you extracted all the files of the project and run the following commands in the command prompt.

- Navigate inside the naming-server project to run the project with the help of maven commands.
 - Run the following command in the destination folder to navigate inside naming-server project.

```
cd naming-server
```

- Run the following command to run the naming-server as spring-boot project.

```
mvn spring-boot:run
```

Run API-Gateway

Our task is to run each project individually. Therefore, go the destination folder where you extracted all the files of the project and run the following commands in the command prompt.

- Navigate inside the api-gateway project to run the project with the help of maven commands.
 - Run the following command in the destination folder to navigate inside api-gateway project.

```
cd api-gateway
```

- Run the following command to run the api-gateway as spring-boot project.

```
mvn spring-boot:run
```

Run Location-Services

Our task is to run each project individually. Therefore, go the destination folder where you extracted all the files of the project and run the following commands in the command prompt.

- Navigate inside the location-services project to run the project with the help of maven commands.
 - Run the following command in the destination folder to navigate inside location-services project.

```
cd location-services
```

- Run the following command to run the naming-server as spring-boot project.

```
mvn spring-boot:run
```

Run Auto-Services

Our task is to run each project individually. Therefore, go the destination folder where you extracted all the files of the project and run the following commands in the command prompt.

- Navigate inside the auto-services project to run the project with the help of maven commands.
 - Run the following command in the destination folder to navigate inside auto-services project.

```
cd auto-services
```

- Run the following command to run the auto-services as spring-boot project.

```
mvn spring-boot:run
```

Run Mileage-Services

Our task is to run each project individually. Therefore, go the destination folder where you extracted all the files of the project and run the following commands in the command prompt.

- Navigate inside the auto-services project to run the project with the help of maven commands.
 - Run the following command in the destination folder to navigate inside mileage-services project.

```
cd mileage-services
```

- Run the following command to run the auto-services as spring-boot project.

```
mvn spring-boot:run
```

Run Premium-Insurance

Our task is to run each project individually. Therefore, go the destination folder where you extracted all the files of the project and run the following commands in the command prompt.

- Navigate inside the auto-services project to run the project with the help of maven commands.
 - Run the following command in the destination folder to navigate inside premium-insurance project.

```
cd premium-insurance
```

- Run the following command to run the premium-insurance as spring-boot project.

```
mvn spring-boot:run
```

Microservices Registration and Discovery with Spring Cloud and Netflix Eureka

Hence, our all microservices are registering with Eureka Naming server and cannot be accessed individually. All the services must access through the api-gateway to perform the necessary actions.

Accessing Premium Insurance API to Calculate Premium Insurance Value.

Description

The Annual Insurance Value request to get the value of the premium insurance based on location (postcode), the mileage of the car (mileage) and the car type (auto).

API Request

<http://localhost:8765/insurance/insurance-calculator/<postcode>/<mileage>/<auto>>

Key	Description	Format	Type
postcode	Location postcode	Integer	Mandatory
mileage	The number of mileage driven by car	Integer	Mandatory
auto	Car type	Integer	Mandatory

Example of Get Request

<http://localhost:8765/insurance/insurance-calculator/79189/100/BMW>

Response

```
{  
  
  • uuid: "6eed5109-9631-4e2e-a98c-31d5b6d0db8c",  
  • countryISO: "DE",  
  • location: "Bad Krozingen",  
  • postcode: 79189,  
  • regionalFactor: 1.8032957213724221,  
  • autoType: "BMW",  
  • autoTypeFactor: 1.2,  
  • autoMileage: 100,  
  • mileageMinimumValue: 0,  
  • mileageMaximumValue: 5000,  
  • autoMileageFactor: 0.5,  
  • currency: "Euro",  
  • annualPremiumInsurance: 1.08  
  
}
```

Key	Description	Format	Type
uuid	The unique transaction Id	Uuid	Mandatory
countryISO	The country name where the postcode belongs	String	Mandatory
location	The name of the location where postcode belongs	String	Mandatory
Postcode	The postcode of location	Integer	Mandatory
regionalFactor	The regional factor	Double	Mandatory
autotype	The auto type	String	Mandatory
autoTypeFactor	The type factor	Double	Mandatory
autoMileage	The range of auto mileage	Integer	Mandatory
mileageMinimumValue	Minimum value of the range mileage	Integer	Mandatory
mileageMaximumValue	Maximum value of the range mileage	Integer	Mandatory
autoMileageFactor	Mileage Factor	Double	Mandatory
Currency	Type of Currency	String	Mandatory
annualPremiumInsurance	<i>The annual premium insurance value calculated based on the parameters</i>	Double	Mandatory

API-Documentation.

The API documentation of all the microservices is given below that include request type, request example and response.

Naming-Server

There is no API documentation for the naming server. However, if the user wants to see the services registered with the Eureka server can see at the following URL.

<http://localhost:8761/>

The screenshot shows the Spring Eureka web interface in a browser. The address bar shows localhost:8761. The page has a dark header with the Spring Eureka logo and navigation links for HOME and LAST 1000 SINCE STARTUP. The main content area is divided into sections: System Status, DS Replicas, and General Info. The System Status section shows environment and data center as N/A, and current time as 2021-11-20T13:16:31 +0500. The DS Replicas section shows instances currently registered with Eureka, including API-GATEWAY, AUTO, INSURANCE, LOCATION, and MILEAGE. The General Info section shows a table with Name and Value columns. An 'Activate Windows' watermark is visible in the bottom right corner.

System Status	
Environment	N/A
Data center	N/A
Current time	2021-11-20T13:16:31 +0500
Uptime	00:54
Lease expiration enabled	false
Renews threshold	10
Renews (last min)	4

EMERGENCY! EUREKA MAY BE INCORRECTLY CLAIMING INSTANCES ARE UP WHEN THEY'RE NOT. RENEWALS ARE LESSER THAN THRESHOLD AND HENCE THE INSTANCES ARE NOT BEING EXPIRED JUST TO BE SAFE.

DS Replicas

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
API-GATEWAY	n/a (1)	(1)	UP (1) - DESKTOP-0P34541.api-gateway:8765
AUTO	n/a (1)	(1)	UP (1) - DESKTOP-0P34541.auto:8100
INSURANCE	n/a (1)	(1)	UP (1) - DESKTOP-0P34541.insurance:8500
LOCATION	n/a (1)	(1)	UP (1) - DESKTOP-0P34541.location:8000
MILEAGE	n/a (1)	(1)	UP (1) - DESKTOP-0P34541.mileage:8200

General Info

Name	Value
------	-------

API-Gateway

The API Gateway is used as the entry point for client to access the microservices.

Location-Services

1. Get Request

Description

The regional factor value based on the postcode.

API Request

<http://localhost:8765/location/postcode-factor/<postcode>>

Key	Description	Format	Type
postcode	Location postcode	Integer	Mandatory

Example of Get Request

<http://localhost:8765/location/postcode-factor/79189>

Response

```

{
  • id: 1,
  • countryISO: "DE",
  • stateISO: "DE-BW",
  • regionOne: "Baden-Württemberg",
  • regionTwo: "Freiburg",
  • regionThree: "Breisgau-Hochschwarzwald",
  • regionFour: "Bad Krozingen",
  • postLeitZahl: 79189,
  • ort: "Bad Krozingen",
  • regionalFactor: 1.8032957213724221,
  • status: "ACTIVE"
}

```

Key	Description	Format	Type
id	The unique Id	Integer	Mandatory
countryISO	The Country name in ISO 3166 standards	String	Mandatory
stateISO	The State name name in ISO 3166 standards	String	Mandatory
regionOne	Region One	String	Mandatory
regionTwo	Region Two	String	Mandatory
regionThree	Region Three	String	Mandatory
regionFour	Region Four	String	Mandatory
postLietZahl	The postcode of the location	Integer	Mandatory
Ort	location	String	Mandatory
regionalFactor	Regional type factor value	Double	Mandatory
status	The status of the soft delete	String	Mandatory

2. Delete Request

Description

The regional factor value based on the postcode.

API Request

<http://localhost:8765/location/postcode-factor/<postcode>>

Key	Description	Format	Type
postcode	Location postcode	Integer	Mandatory

Example of Delete Request

<http://localhost:8765/location/postcode-factor/79189>

Response

The screenshot shows a REST client interface with the following details:

- Toolbar:** Includes a 'Save' button, a dropdown menu, and a 'Send' button.
- Request Bar:** The method is 'DELETE' and the URL is 'http://localhost:8765/location/postcode-factor/79189'.
- Auth Tab:** The 'Authorization' tab is selected, showing 'Type: Inherit auth from parent'. A note states: 'This request is using No Auth from collection AppsFactory.'
- Response Tab:** The 'Body' tab is selected, showing a '204 No Content' status. The response time is '23 ms' and the size is '64 B'. A 'Save Response' button is visible.
- Response Content:** A tooltip displays the message: '204 No Content. The server successfully processed the request, but is not returning any content.'

3. Add Request

Description

Add the regional factor value based on the postcode.

API Request

<http://localhost:8765/location/postcode-factor>

Request Body

Key	Description	Format	Type
countryISO	The Country name in ISO 3166 standards	String	Mandatory
stateISO	The State name name in ISO 3166 standards	String	Mandatory
regionOne	Region One	String	Mandatory
regionTwo	Region Two	String	Mandatory
regionThree	Region Three	String	Mandatory
regionFour	Region Four	String	Mandatory
postLietZahl	The postcode of the location	Integer	Mandatory
Ort	location	String	Mandatory
regionalFactor	Regional type factor value	Double	Mandatory

Example of Add Request

<http://localhost:8765/location/postcode-factor>

POST

localhost:8765/location/postcode-factor

Send

ParamsAuthorizationHeaders (8)BodyPre-request ScriptTestsSettingsCookies

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/>	countryISO	DE			
<input checked="" type="checkbox"/>	stateISO	DE-BW			
<input checked="" type="checkbox"/>	regionOne	Rheinland-Pfalz			
<input checked="" type="checkbox"/>	regionTwo	Koblenz			
<input checked="" type="checkbox"/>	regionThree	Neuwied			
<input checked="" type="checkbox"/>	regionFour	Neuwied			
<input checked="" type="checkbox"/>	postLeitZahl	99999			
<input checked="" type="checkbox"/>	ort	kross			
<input checked="" type="checkbox"/>	regionalFactor	1.00			

Response

```
{
  "id": 22899,
  "countryISO": "DE",
  "stateISO": "DE-BW",
  "regionOne": "Rheinland-Pfalz",
  "regionTwo": "Koblenz",
  "regionThree": "Neuwied",
  "regionFour": "Neuwied",
  "postLeitZahl": 99999,
  "ort": "kross",
  "regionalFactor": 2.5367129124309713,
  "status": "ACTIVE"
}
```

Key	Description	Format	Type
id	The unique Id	Integer	Mandatory
countryISO	The Country name in ISO 3166 standards	String	Mandatory
stateISO	The State name name in ISO 3166 standards	String	Mandatory
regionOne	Region One	String	Mandatory
regionTwo	Region Two	String	Mandatory
regionThree	Region Three	String	Mandatory
regionFour	Region Four	String	Mandatory

postLietZahl	The postcode of the location	Integer	Mandatory
Ort	location	String	Mandatory
regionalFactor	Regional type factor value	Double	Mandatory
status	The status of the soft delete	String	Mandatory

4. Update Request

Description

Update regional factor value based on the postcode.

API Request

<http://localhost:8765/location/postcode-factor/<postcode>>

Key	Description	Format	Type
postcode	Location postcode	Integer	Mandatory

Request Body

Key	Description	Format	Type
countryISO	The Country name in ISO 3166 standards	String	Mandatory
stateISO	The State name name in ISO 3166 standards	String	Mandatory
regionOne	Region One	String	Mandatory
regionTwo	Region Two	String	Mandatory
regionThree	Region Three	String	Mandatory
regionFour	Region Four	String	Mandatory
postLietZahl	The postcode of the location	Integer	Mandatory
Ort	location	String	Mandatory
regionalFactor	Regional type factor value	Double	Mandatory

Example of Update Request

<http://localhost:8765/location/postcode-factor/99999>

PUT

localhost:8765/location/postcode-factor/99999

Send

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Cookies

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/>	countryISO	DE			
<input checked="" type="checkbox"/>	stateISO	DE-BW			
<input checked="" type="checkbox"/>	regionOne	Rheinland-Pfalz			
<input checked="" type="checkbox"/>	regionTwo	Koblenz			
<input checked="" type="checkbox"/>	regionThree	Neuwied			
<input checked="" type="checkbox"/>	regionFour	Neuwied			
<input checked="" type="checkbox"/>	postLeitZahl	99999			
<input checked="" type="checkbox"/>	ort	krossen			
<input checked="" type="checkbox"/>	regionalFactor	1.00			

Activate Windows

Response

```
{
  "id": 22899,
  "countryISO": "DE",
  "stateISO": "DE-BW",
  "regionOne": "Rheinland-Pfalz",
  "regionTwo": "Koblenz",
  "regionThree": "Neuwied",
  "regionFour": "Neuwied",
  "postLeitZahl": 99999,
  "ort": "krossen",
  "regionalFactor": 1.0,
  "status": "ACTIVE"
}
```

Key	Description	Format	Type
id	The unique Id	Integer	Mandatory
countryISO	The Country name in ISO 3166 standards	String	Mandatory
stateISO	The State name name in ISO 3166 standards	String	Mandatory
regionOne	Region One	String	Mandatory
regionTwo	Region Two	String	Mandatory
regionThree	Region Three	String	Mandatory
regionFour	Region Four	String	Mandatory
postLietZahl	The postcode of the location	Integer	Mandatory
Ort	location	String	Mandatory
regionalFactor	Regional type factor value	Double	Mandatory
status	The status of the soft delete	String	Mandatory

Auto-Services

1. Get Request

Description

The Type Class factor value based on the auto type.

API Request

<http://localhost:8765/auto/auto-factor/<auto>>

Key	Description	Format	Type
auto	Auto Type	String	Mandatory

Example of Get Request

<http://localhost:8765/auto/auto-factor/BMW>

Response

```
{
  "id": 1,
  "carName": "BMW",
  "status": "ACTIVE",
  "factor": 1.2
}
```

Key	Description	Format	Type
id	The unique Id	Integer	Mandatory
carName	The auto type	String	Mandatory
factor	The Type class factor	Double	Mandatory
status	The status of the soft delete	String	Mandatory

2. Delete Request

Description

The regional factor value based on the postcode.

API Request

<http://localhost:8765/auto/auto-factor/<auto>>

Key	Description	Format	Type
auto	Auto Type	String	Mandatory

Example of Delete Request

<http://localhost:8765/auto/auto-factor/BMW>

Response

DELETE

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (1) Test Results 136 ms 64 B

Pretty Raw Preview Visualize

1

Mileage-Services

1. Get Request

Description

The mileage factor value based on the number of mileages driven by Auto.

API Request

<http://localhost:8765/mileage/mileage-factor/<mileage>>

Key	Description	Format	Type
mileage	The number of miles driven by the car.	Integer	Mandatory

Example of Get Request

<http://localhost:8765/mileage/mileage-factor/100>

Response

```
{
  "id": 1,
  "minMileage": 0,
  "maxMileage": 5000,
  "factor": 0.5,
  "status": "ACTIVE"
}
```

Key	Description	Format	Type
id	The unique Id	Integer	Mandatory
minMileage	The minimum mileage range for the same mileage factor	Integer	Mandatory
maxMileage	The maximum mileage range for the same mileage factor	Integer	Mandatory
factor	The mileage factor	Double	Mandatory
status	The status of the soft delete	String	Mandatory

2. Delete Request

Description

The mileage factor value based on the range of the mileages.

API Request

<http://localhost:8765/mileage/mileage-factor/<minValue>/<maxValue>>

Key	Description	Format	Type
minValue	The minimum value for the range factor	Integer	Mandatory
maxValue	The maximum value for the range factor	Integer	Mandatory

Example of Delete Request

<http://localhost:8765/mileage/mileage-factor/0/5000>

Response

The screenshot shows the AppsFactory REST client interface. At the top, it displays 'AppsFactory / Delete mileage Factor'. Below this, there's a 'DELETE' method selected and the URL 'http://localhost:8765/mileage/mileage-factor/0/5000'. A 'Send' button is visible. The interface has tabs for 'Params', 'Authorization', 'Headers (6)', 'Body', 'Pre-request Script', 'Tests', and 'Settings'. The 'Params' tab is active, showing a table with columns 'KEY', 'VALUE', 'DESCRIPTION', and 'Bulk Edit'. Below the table, there's a 'Query Params' section. The 'Body' tab is also visible, showing a '204 No Content' status, '138 ms' response time, and '64 B' body size. There's a 'Save Response' button and a 'Pretty' button for formatting the response.

Premium-Insurance

1. Get Request

Description

The Annual Insurance Value request to get the value of the premium insurance based on location (postcode), the mileage of the car (mileage) and the car type (auto).

API Request

<http://localhost:8765/insurance/insurance-calculator/<postcode>/<mileage>/<auto>>

Key	Description	Format	Type
postcode	Location postcode	Integer	Mandatory
mileage	The number of mileage driven by car	Integer	Mandatory
auto	Car type	Integer	Mandatory

Example of Get Request

<http://localhost:8765/insurance/insurance-calculator/79189/100/BMW>

Response

```
{  
  
  • uuid: "6eed5109-9631-4e2e-a98c-31d5b6d0db8c",  
  • countryISO: "DE",  
  • location: "Bad Krozingen",
```

- `postcode: 79189,`
- `regionalFactor: 1.8032957213724221,`
- `autoType: "BMW",`
- `autoTypeFactor: 1.2,`
- `autoMileage: 100,`
- `mileageMinimumValue: 0,`
- `mileageMaximumValue: 5000,`
- `autoMileageFactor: 0.5,`
- `currency: "Euro",`
- `annualPremiumInsurance: 1.08`

}

Key	Description	Format	Type
uuid	The unique transaction Id	Uuid	Mandatory
countryISO	The country name where the postcode belongs	String	Mandatory
location	The name of the location where postcode belongs	String	Mandatory
Postcode	The postcode of location	Integer	Mandatory
regionalFactor	The regional factor	Double	Mandatory
autotype	The auto type	String	Mandatory
autoTypeFactor	The type factor	Double	Mandatory
autoMileage	The range of auto mileage	Integer	Mandatory
mileageMinimumValue	Minimum value of the range mileage	Integer	Mandatory
mileageMaximumValue	Maximum value of the range mileage	Integer	Mandatory
autoMileageFactor	Mileage Factor	Double	Mandatory
Currency	Type of Currency	String	Mandatory
annualPremiumInsurance	<i>The annual premium insurance value calculated based on the parameters</i>	Double	Mandatory