

Taller Arquitectura Cliente-Servidor: Sockets, WebServer y API Rest



Universidad
del Cauca

Vigilada Mineducación

Ingeniería de software II

Presentado por:

Milthon Ferney Caicedo Jurado.

Profesor:

Julio Ariel Hurtado Alegría

Universidad del Cauca

Facultad de Ingeniería Electrónica y Telecomunicaciones

Programa de Ingeniería de Sistemas

Popayán, Junio del 2022

Taller Arquitectura Cliente-Servidor: Sockets, WebServer y API Rest

Tareas

1. Descargar, instalar y correr la clase EchoServerExample que viene en el proyecto strategyserver. Valor 0.5
2. Descargar, instalar y correr el AgencyTravelServer como servidor tcp/ip y el AgencyTravelClient (infra.tcpip). Valor 0.5
3. Correr el Web Server(infra.web) que vienen en el mismo proyecto AgencyTravelServer. Usar un cliente postman para hacer la consulta, recuerden que el protocolo cambió y ahora sólo van los parámetros. Ver la consulta postman. Valor 0.5
4. Realizar el taller de la API Rest y probar las consultas a través de postman o un Jersey Client. Sigán el paso a paso del taller. Valor 1.5
5. Hacer una API Rest para el AgencyTravelServer. Probarla desde un Jersey Client o a través de postman. Valor Valor 2.0.

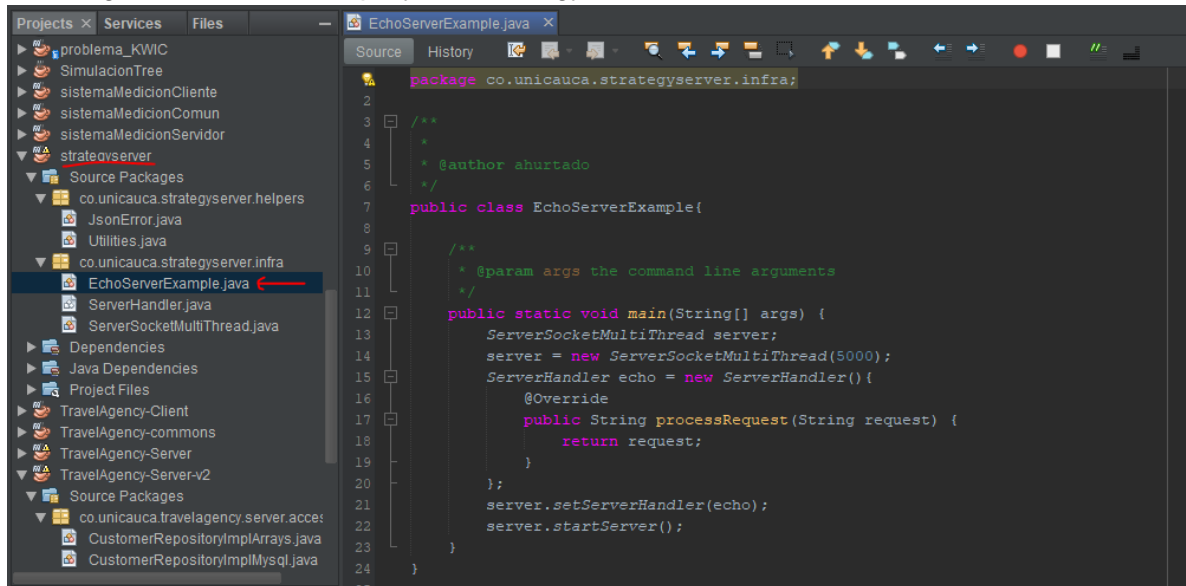
Deberán entregar un link de la solución que debe tener el código del último punto y los pantallazos que evidencien la ejecución de las aplicaciones.

La fecha de entrega 21 de junio de 2022 a través del sitio del curso en Moodle

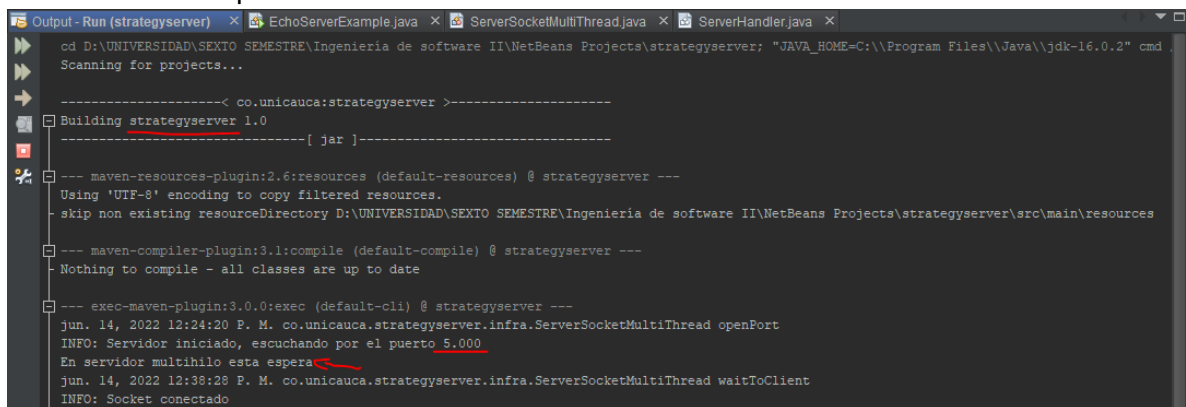
Solución:

1. Descargar, instalar y correr la clase EchoServerExample que viene en el proyecto strategyserver. Valor 0.5

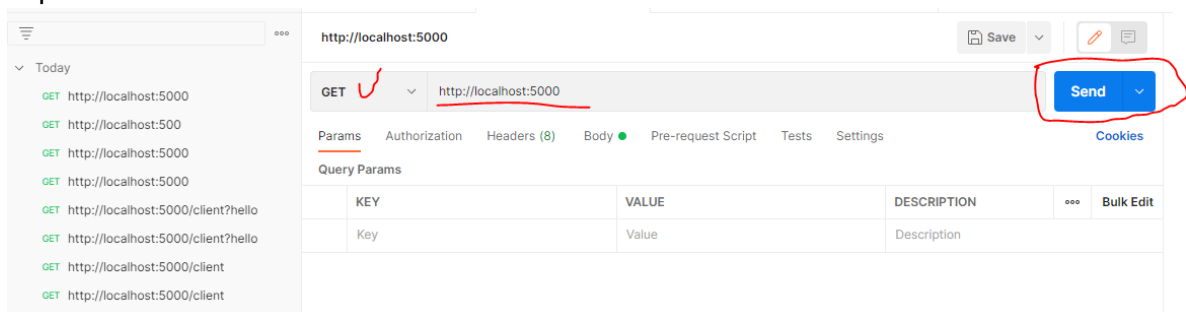
Descarga e instalación del proyecto strategyserver



Ahora pasamos a correr el proyecto el cual tiene su arranque main en EchoServerExample.



Le hice una petición Get desde Postman y claramente el servidor está escuchando por el puerto 5000.



Salida por pantalla al enviar la petición de Postman, claramente al darle lógica al cliente podemos obtener respuestas con información.

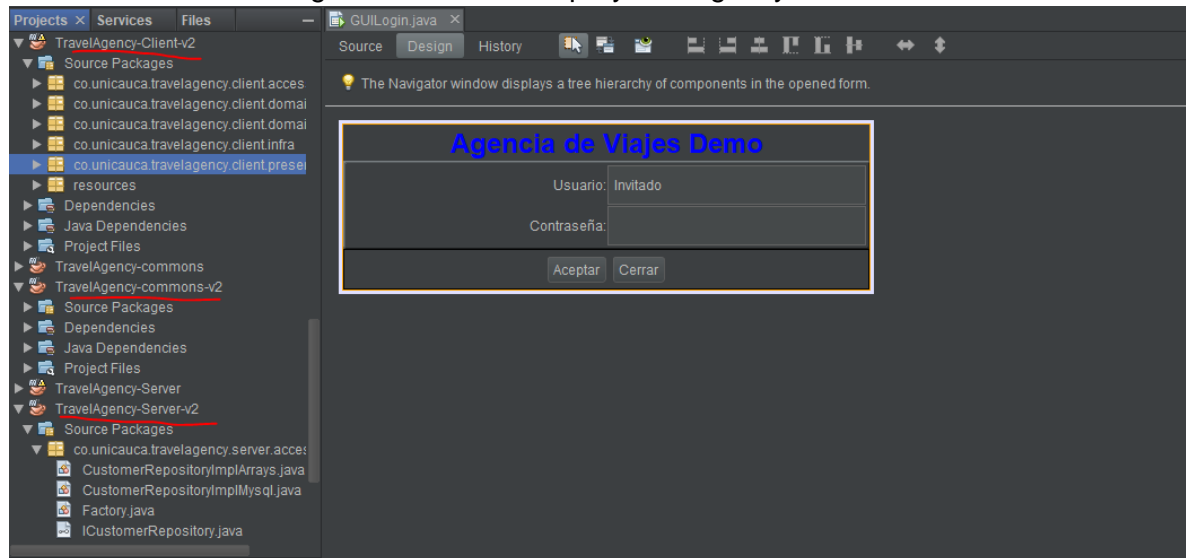
```

En servidor multihilo esta espera
jun. 14, 2022 12:38:28 P. M. co.unicauca.strategyserver.infra.ServerSocketMultiThread waitToClient
INFO: Socket conectado
En servidor multihilo esta espera
jun. 14, 2022 12:38:57 P. M. co.unicauca.strategyserver.infra.ServerSocketMultiThread waitToClient
INFO: Socket conectado
En servidor multihilo esta espera
jun. 14, 2022 12:38:57 P. M. co.unicauca.strategyserver.infra.ServerSocketMultiThread waitToClient
INFO: Socket conectado
En servidor multihilo esta espera
jun. 14, 2022 12:39:11 P. M. co.unicauca.strategyserver.infra.ServerSocketMultiThread waitToClient
INFO: Socket conectado

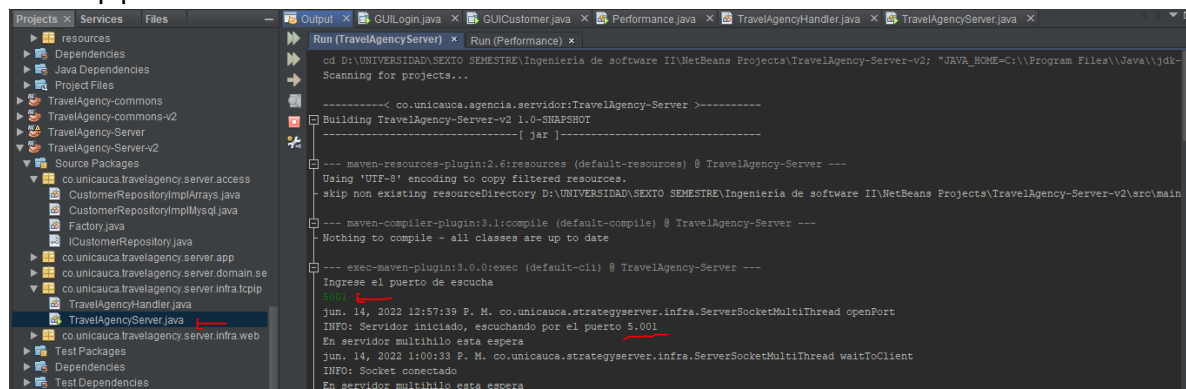
```

2. Descargar, instalar y correr el AgencyTravelServer como servidor tcp/ip y el AgencyTravelClient (infra.tcpip). Valor 0.5

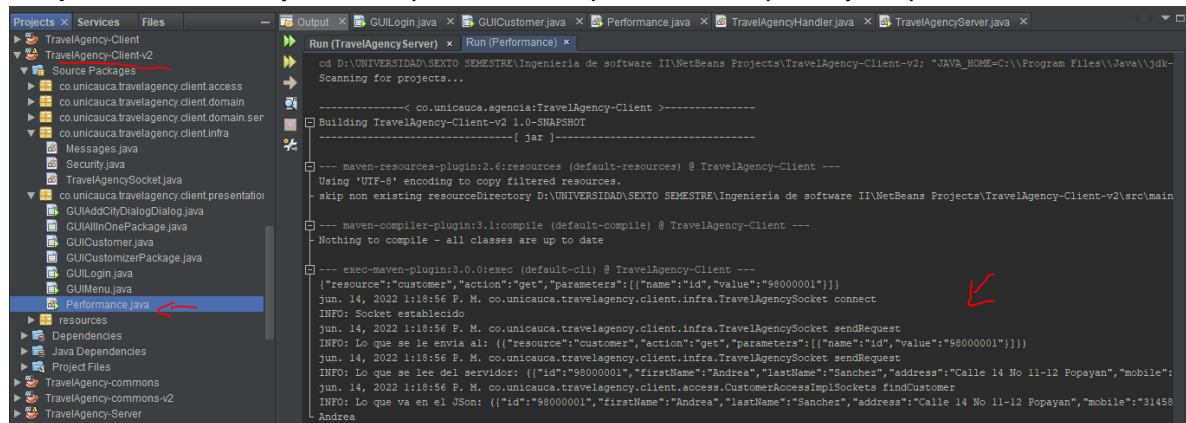
Resultado de la descarga e instalación del proyecto AgencyTravelServer



Ahora pasamos a construirlo a ejecutarlo el servidor como tcp/ip y el cliente como infra.tcpip.

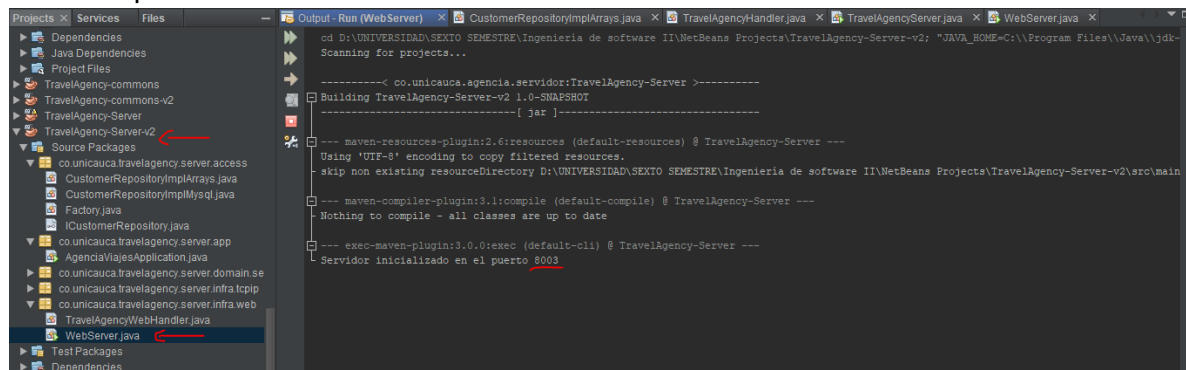


Al ejecutar el cliente y mandar peticiones se puede notar que hay respuesta correcta.

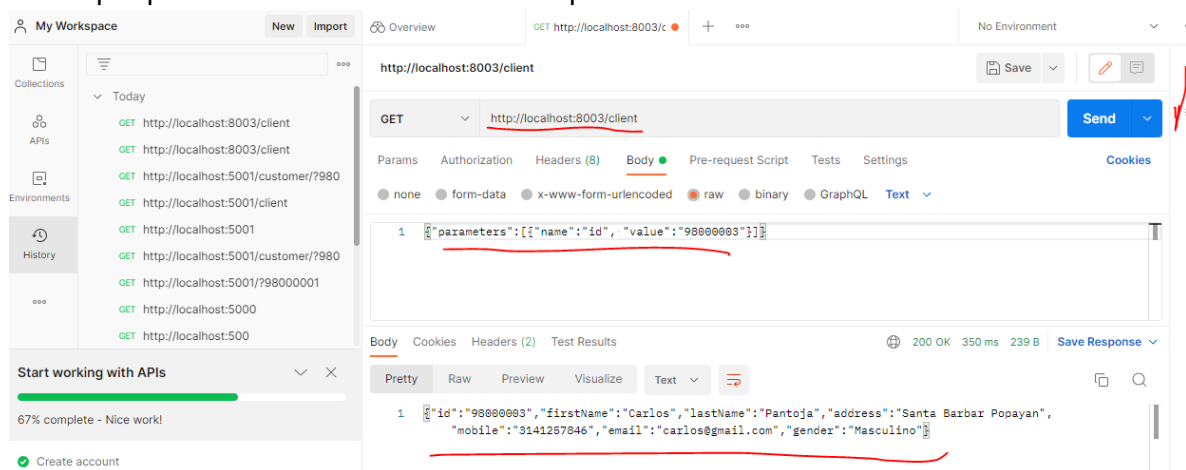


3. Correr el Web Server(infra.web) que vienen en el mismo proyecto AgencyTravelServer. Usar un cliente postman para hacer la consulta, recuerden que el protocolo cambi  y ahora s lo van los par metros. Ver la consulta postman. Valor 0.5

En este caso vamos a correr al Web Server (infra.web) y usaremos un cliente Postman para hacer la consulta.



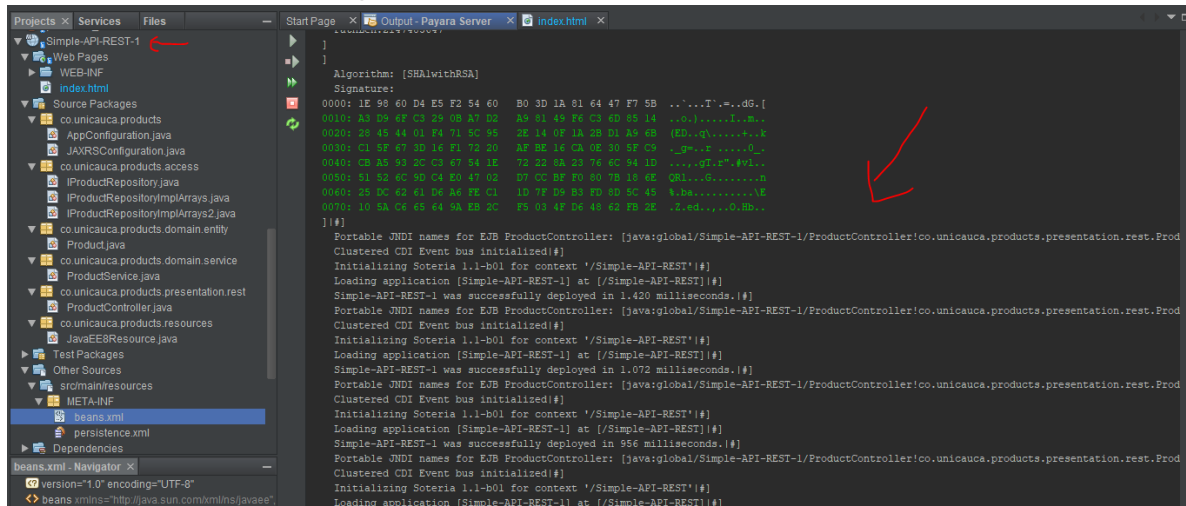
Ahora por parte del cliente vamos a enviar peticiones haciendo uso de Postman.



Claramente podemos observar que al enviar una petici n al puerto 8003 se obtuvo respuesta por parte del servidor y en este caso se recuper  los datos de un cliente.

4. Realizar el taller de la API Rest y probar las consultas a través de postman o un Jersey Client. Siguen el paso a paso del taller. Valor 1.5

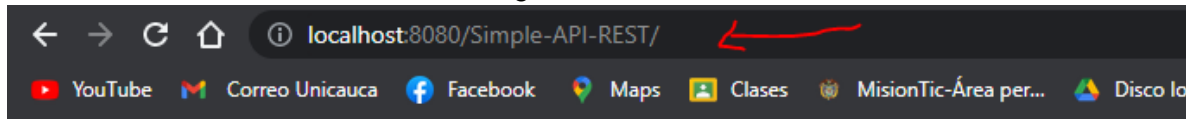
Evidencias del API-REST que se creo en NetBeans



```
Algorithm: [SHA1withRSA]
Signature:
0000: 1E 98 60 D4 E5 F2 54 60 B0 3D 1A 01 64 47 F7 5B ...T...dg.
0010: A3 D9 4F C3 29 08 A7 D2 A9 01 49 F6 C3 6D 85 14 ...b).....m.
0020: 26 45 44 01 F4 71 8C 95 2E 14 0F 1A 2B D1 A9 6B (E...q.....k
0030: C1 5F 67 3D 16 F1 72 20 AF BE 16 CA 0E 30 5F C9 ...ge...k...0...
0040: CB A5 93 2C C3 67 54 1E 72 22 5A 23 76 6C 94 1D ...g...k...v...
0050: 51 52 40 9D C4 E0 47 02 D7 CC BF F0 60 7B 16 4E QRI...G.....h
0060: 25 DC 62 61 D6 A6 FE C1 1D 7F D9 B3 F0 8D 9C 45 k.be.....E
0070: 10 5A C4 65 64 9A E8 2C F9 03 4F D6 48 62 F8 2E .2.ed.....O.Rh...

[14]
Portable JNDI names for EJB ProductController: [java:global/Simple-API-REST-1/ProductController!co.unicauca.products.presentation.rest.Prod
Clustered CDI Event bus initialized(##)
Initializing Soteria 1.1-b01 for context '/Simple-API-REST'(!)
Loading application [Simple-API-REST-1] at [/Simple-API-REST](!)
Simple-API-REST-1 was successfully deployed in 1.420 milliseconds.(!)
Portable JNDI names for EJB ProductController: [java:global/Simple-API-REST-1/ProductController!co.unicauca.products.presentation.rest.Prod
Clustered CDI Event bus initialized(##)
Initializing Soteria 1.1-b01 for context '/Simple-API-REST'(!)
Loading application [Simple-API-REST-1] at [/Simple-API-REST](!)
Simple-API-REST-1 was successfully deployed in 1.072 milliseconds.(!)
Portable JNDI names for EJB ProductController: [java:global/Simple-API-REST-1/ProductController!co.unicauca.products.presentation.rest.Prod
Clustered CDI Event bus initialized(##)
Initializing Soteria 1.1-b01 for context '/Simple-API-REST'(!)
Loading application [Simple-API-REST-1] at [/Simple-API-REST](!)
Simple-API-REST-1 was successfully deployed in 956 milliseconds.(!)
Portable JNDI names for EJB ProductController: [java:global/Simple-API-REST-1/ProductController!co.unicauca.products.presentation.rest.Prod
Clustered CDI Event bus initialized(##)
Initializing Soteria 1.1-b01 for context '/Simple-API-REST'(!)
Loading application [Simple-API-REST-1] at [/Simple-API-REST](!)
```

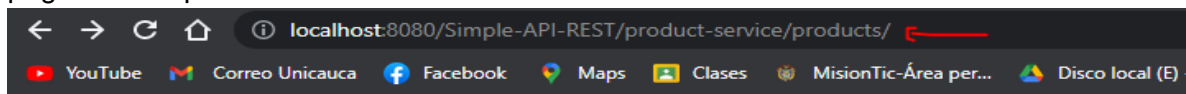
Ahora vemos los resultados en el navegador



Hello World!

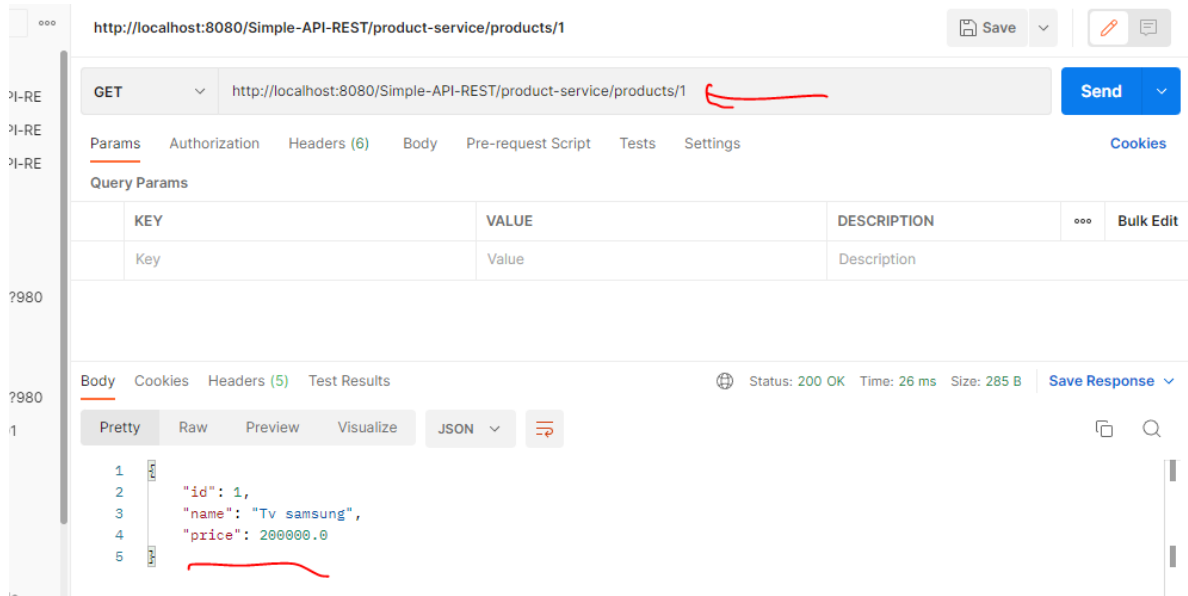
Esta es mi primera Aplicación API-REST

Finalmente podemos hacer las respectivas peticiones, GET, POST, etc con la misma página o con postman

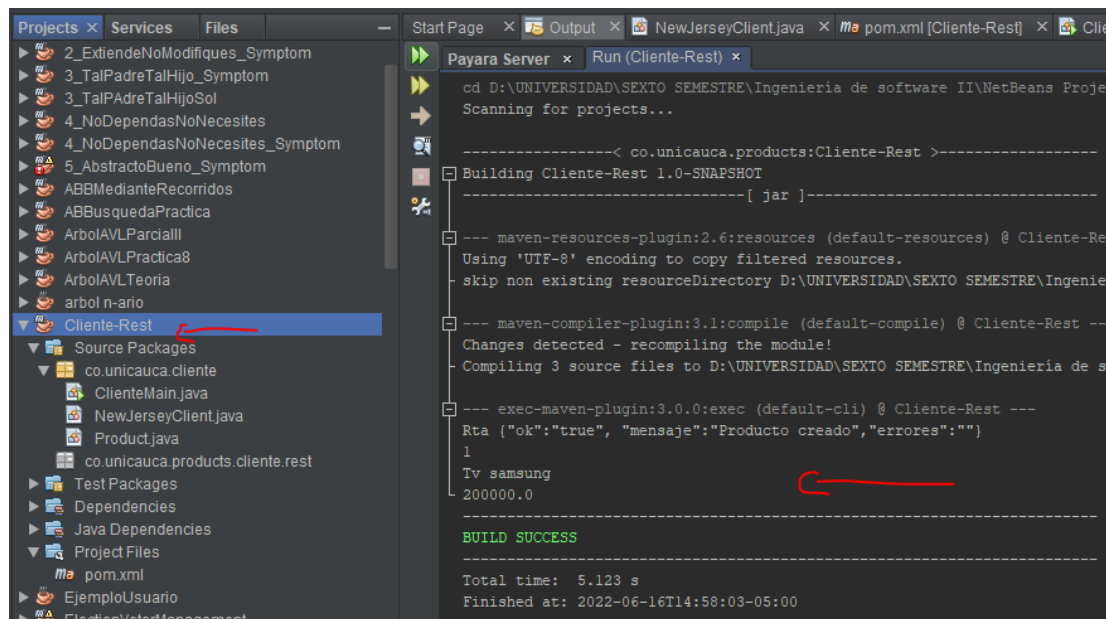


```
[
  {
    id: 1,
    name: "Tv samsung",
    price: 200000
  },
  {
    id: 2,
    name: "Tv lg",
    price: 300000
  },
  {
    id: 1,
    name: "Tablet asus RESD-TD-34",
    price: 400000
  }
]
```

Con Postman



Finalmente probamos desde un Jersey Client.



5. Hacer una API Rest para el AgencyTravelServer. Probarla desde un Jersey Client o a través de postman. Valor Valor 2.0.

Link del repositorio GitHub de la solución:

<https://github.com/mfcaicedo/API-REST-TravelAgency>

Evidencias de que el API-REST si funcionó:

Corriendo la API-REST

```
Payara Server x Run (API-REST-TravelAgency-1) x
No tests to run.

--- maven-war-plugin:2.3:war (default-war) @ API-REST-TravelAgency ---
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by com.thoughtworks.xstream.converters.collections.TreeMapConverter (file:/C:/Users/HP/.m2/repository/com/thoughtworks/xstream/xstream/1.4.10/xstream-1.4.10.jar) method com.thoughtworks.xstream.converters.collections.TreeMapConverter.readFromCollection(java.lang.Object,java.lang.Class)
WARNING: Please consider reporting this to the maintainers of com.thoughtworks.xstream.converters.collections.TreeMapConverter
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release

Packaging webapp
Assembling webapp [API-REST-TravelAgency] in [D:\UNIVERSIDAD\SEXTO SEMESTRE\Ingenieria de software II\Taller-API-REST\API-REST-TravelAgency\t
Processing war project
Copying webapp resources [D:\UNIVERSIDAD\SEXTO SEMESTRE\Ingenieria de software II\Taller-API-REST\API-REST-TravelAgency\src\main\webapp]
Webapp assembled in [75 ms]
Building war: D:\UNIVERSIDAD\SEXTO SEMESTRE\Ingenieria de software II\Taller-API-REST\API-REST-TravelAgency\target\API-REST-TravelAgency-1.wa

BUILD SUCCESS

Total time: 3.666 s
Finished at: 2022-06-24T22:20:37-05:00

Deploying on Payara Server
  profile mode: false
  debug mode: false
  force redeploy: true
Starting Payara Server
Payara Server is running.
Undeploying ...
In-place deployment at D:\UNIVERSIDAD\SEXTO SEMESTRE\Ingenieria de software II\Taller-API-REST\API-REST-TravelAgency\target\API-REST-TravelAg
Deploying on Payara Server
  profile mode: false
  debug mode: false
  force redeploy: true
Undeploying ...
In-place deployment at D:\UNIVERSIDAD\SEXTO SEMESTRE\Ingenieria de software II\Taller-API-REST\API-REST-TravelAgency\target\API-REST-TravelAg
```

Usando del GET

<http://localhost:8080/Simple-API-REST/product-service/products/1> Save ⌵ ⌵ ⌵

GET ⌵ <http://localhost:8080/API-REST-TravelAgency/customer-service/customers/> Send ⌵

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

Query Params


KEY	VALUE	DESCRIPTION	⋮	Bulk Edit
Key	Value	Description		

Body Cookies Headers (5) Test Results ⌵ Status: 200 OK Time: 56 ms Size: 1.9 KB Save Response ⌵

Pretty Raw Preview Visualize JSON ⌵ ⌵ ⌵ ⌵

```
1 {
2   {
3     "address": "Calle 14 No 11-12 Popayan",
4     "email": "andrea@hotmail.com",
5     "firstName": "Andrea",
6     "gender": "Femenino",
7     "id": "98000001",
8     "lastName": "Sanchez",
9     "mobile": "3145878752"
10  },
11  {
12    "address": "Santa Barbar Popayan",
13    "email": "libardo@gmail.com",
14    "firstName": "Libardo",
15    "gender": "Masculino",
16    "id": "98000002",
```

Usando del POST


http://localhost:8080/API-REST-TravelAgency/customer-service/customers/ 



POST http://localhost:8080/API-REST-TravelAgency/customer-service/customers/ **Send**

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL ☒ JSON Beautify


```
1 { "address": "Calle 14 No 11-12 Popayan", "email": "andrea@hotmail.com", "firstName": "Felipe", "gender": "Masculino", "id": "12345", "lastName": "Caicedo", "mobile": "3145878752" }
```

body Cookies Headers (5) Test Results  Status: 200 OK Time: 11 ms Size: 298 B **Save Response**

Pretty Raw Preview Visualize **Text**  

```
1 { "ok": "true", "mensaje": "Cliente creado con éxito", "errores": "" }
```

Usando del PUT


http://localhost:8080/API-REST-TravelAgency/customer-service/customers/ 



PUT http://localhost:8080/API-REST-TravelAgency/customer-service/customers/ **Send**

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL ☒ JSON Beautify


```
1 { "address": "Calle 14 No 11-12 Popayan", "email": "andrea@hotmail.com", "firstName": "Milthon Ferney", "gender": "Masculino", "id": "123", "lastName": "Caicedo", "mobile": "3145878752" }
```

body Cookies Headers (5) Test Results  Status: 200 OK Time: 19 ms Size: 304 B **Save Response**

Pretty Raw Preview Visualize **Text**  

```
1 { "ok": "true", "mensaje": "Cliente modificado exitosamente", "errores": "" }
```

Usando del DELETE


http://localhost:8080/API-REST-TravelAgency/customer-service/customers/98000009 



DELETE http://localhost:8080/API-REST-TravelAgency/customer-service/customers/98000009 **Send**

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

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

body Cookies Headers (5) Test Results  Status: 200 OK Time: 9 ms Size: 299 B **Save Response**

Pretty Raw Preview Visualize **Text**  

```
1 { "ok": "true", "mensaje": "Cliente borrado con éxito", "errores": "" }
```