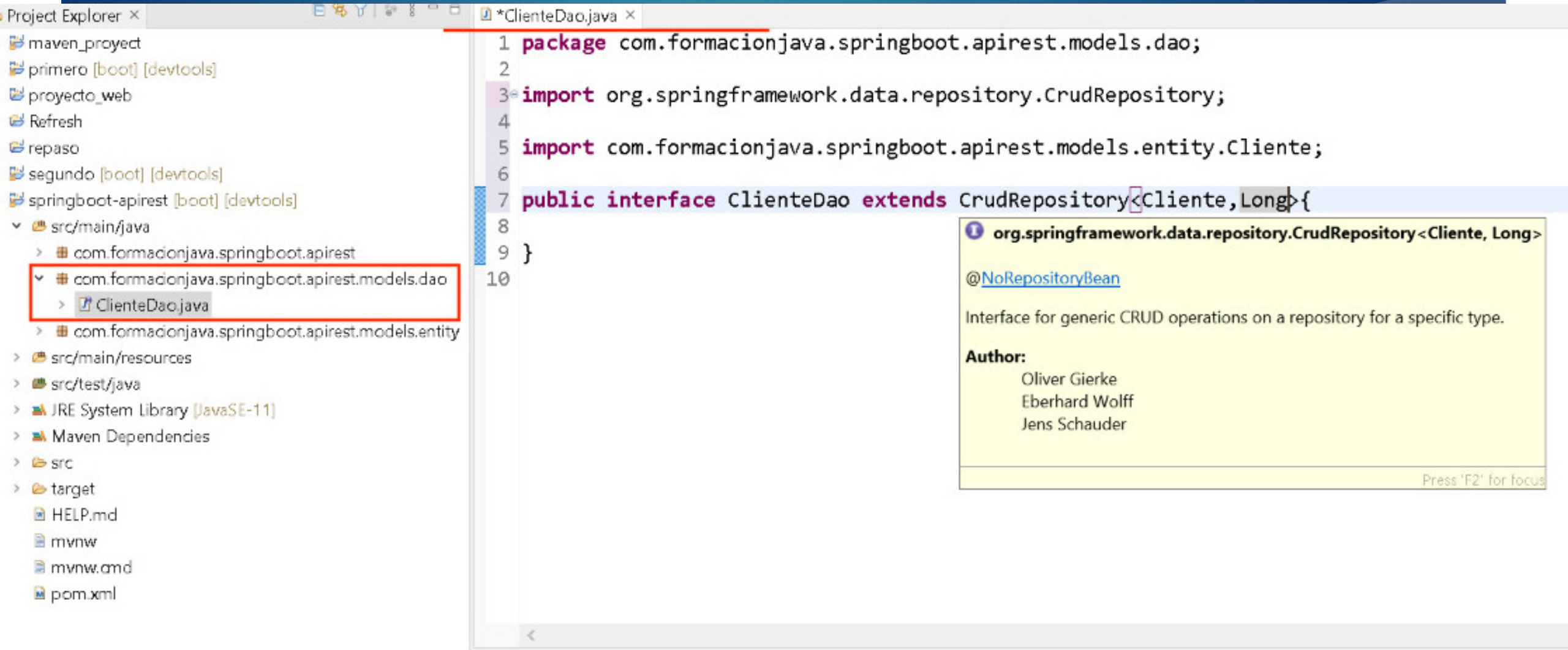
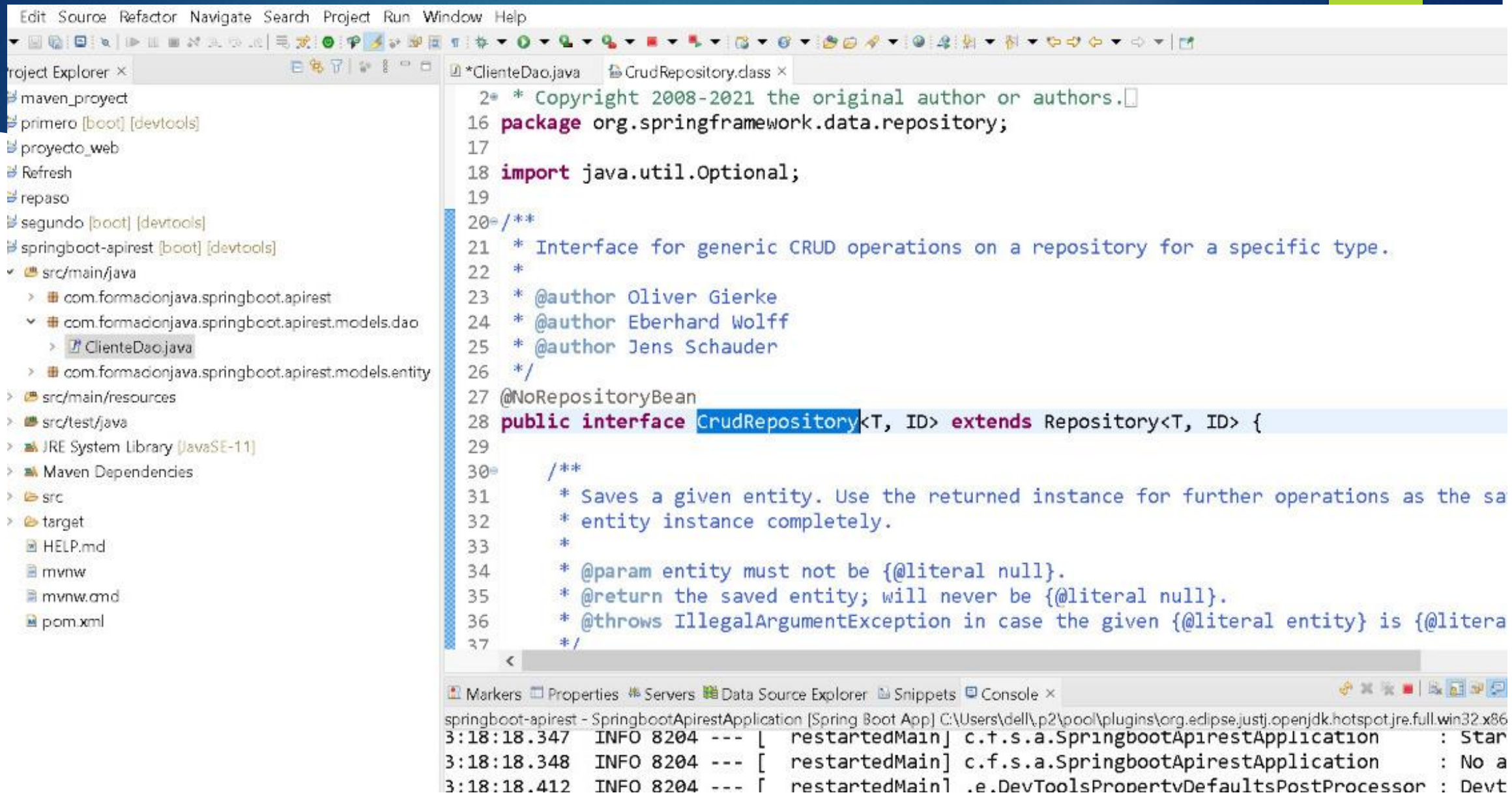
Creamos el paquete y la interfaz para utilizar CrudRepository

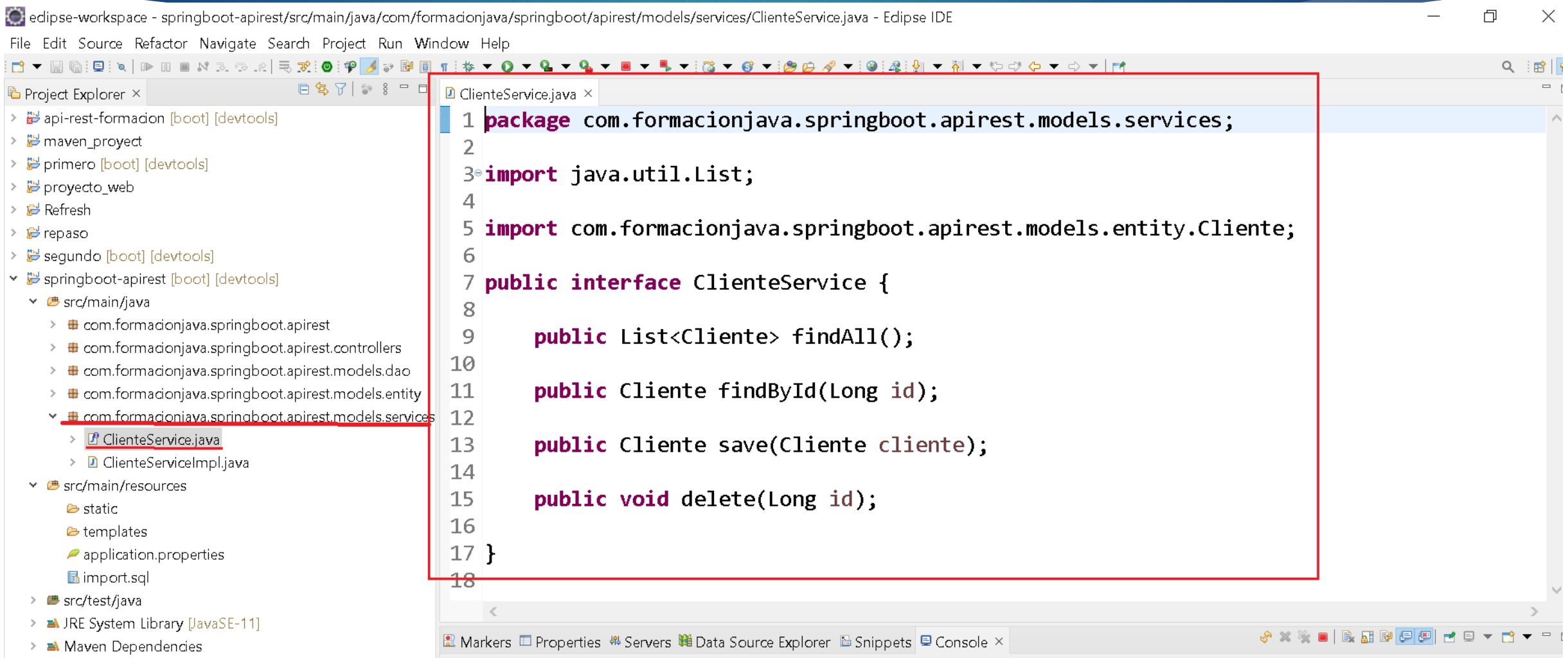


CrudRepository

- CrudRepository implementa operaciones CRUD básicas, que incluyen contar, eliminar, deleteById, save, saveAll, findById y findAll. Lo podemos implementar gracias a SpringDataJPA


Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
maven_proyect
primero [boot] [devtools]
proyecto_web
Refresh
repaso
segundo [boot] [devtools]
springboot-apirest [boot] [devtools]
src/main/java
com.formacionjava.springboot.apirest
com.formacionjava.springboot.apirest.models.dao
ClienteDao.java
com.formacionjava.springboot.apirest.models.entity
src/main/resources
src/test/java
JRE System Library [JavaSE-11]
Maven Dependencies
src
target
HELP.md
mvnw
mvnw.cmd
pom.xml
*ClienteDao.java
CrudRepository.class
2 * Copyright 2008-2021 the original author or authors.
16 package org.springframework.data.repository;
17
18 import java.util.Optional;
19
20 /**
21 * Interface for generic CRUD operations on a repository for a specific type.
22 *
23 * @author Oliver Gierke
24 * @author Eberhard Wolff
25 * @author Jens Schauder
26 */
27 @NoRepositoryBean
28 public interface CrudRepository<T, ID> extends Repository<T, ID> {
29
30 /**
31 * Saves a given entity. Use the returned instance for further operations as the sa
32 * entity instance completely.
33 *
34 * @param entity must not be {@literal null}.
35 * @return the saved entity; will never be {@literal null}.
36 * @throws IllegalArgumentException in case the given {@literal entity} is {@litera
37 */
Markers Properties Servers Data Source Explorer Snippets Console
springboot-apirest - SpringbootApirestApplication [Spring Boot App] C:\Users\dell\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86
3:18:18.347 INFO 8204 --- [restartedMain] c.f.s.a.SpringbootApirestApplication : Star
3:18:18.348 INFO 8204 --- [restartedMain] c.f.s.a.SpringbootApirestApplication : No a
3:18:18.412 INFO 8204 --- [restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : Devt

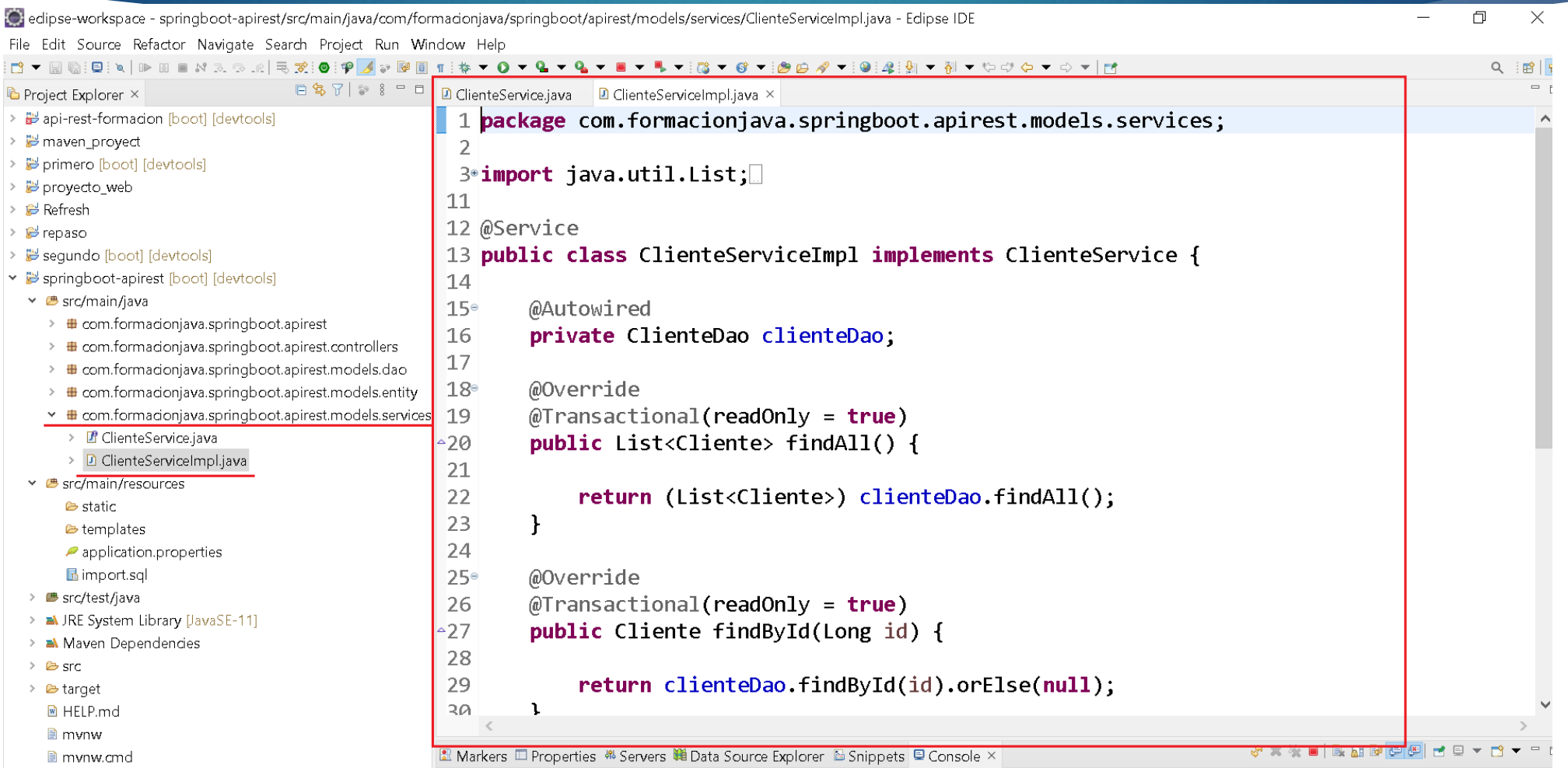
Creamos los servicios en el siguiente paquete y la interfaz

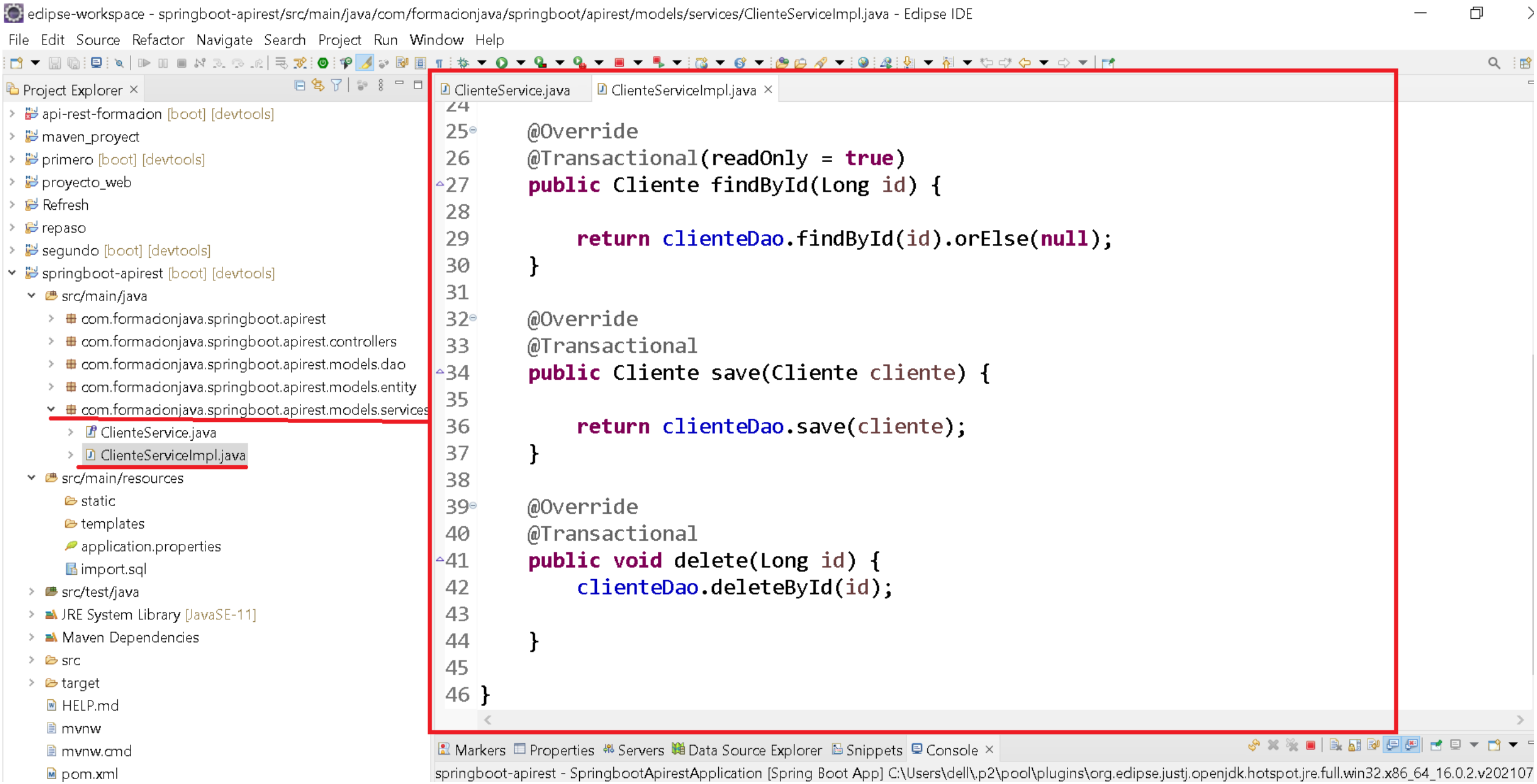


ClienteService

- Implementamos aquí todos los métodos que utilizaremos en nuestra api rest

Creamos la clase ClienteServiceImpl que implementara la interfaz ClienteService y definiendo los métodos

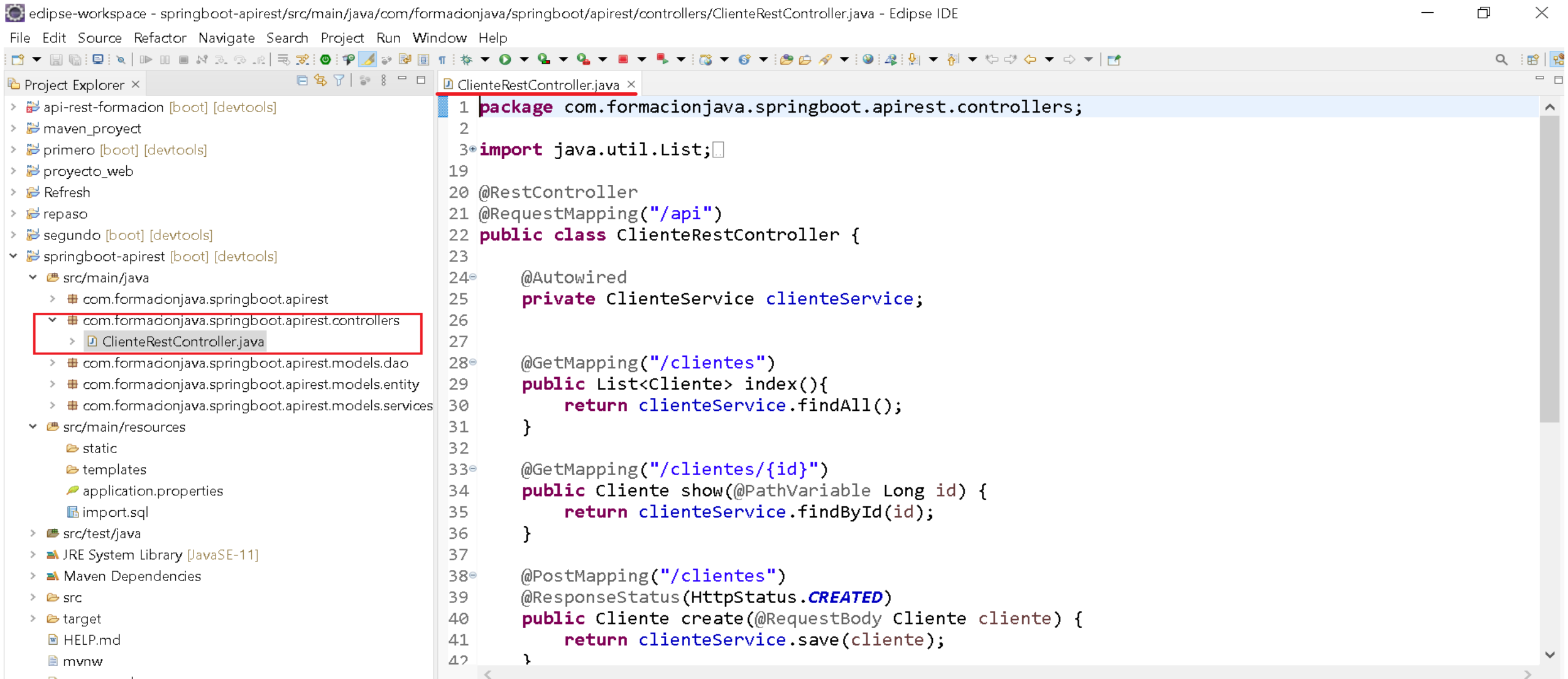


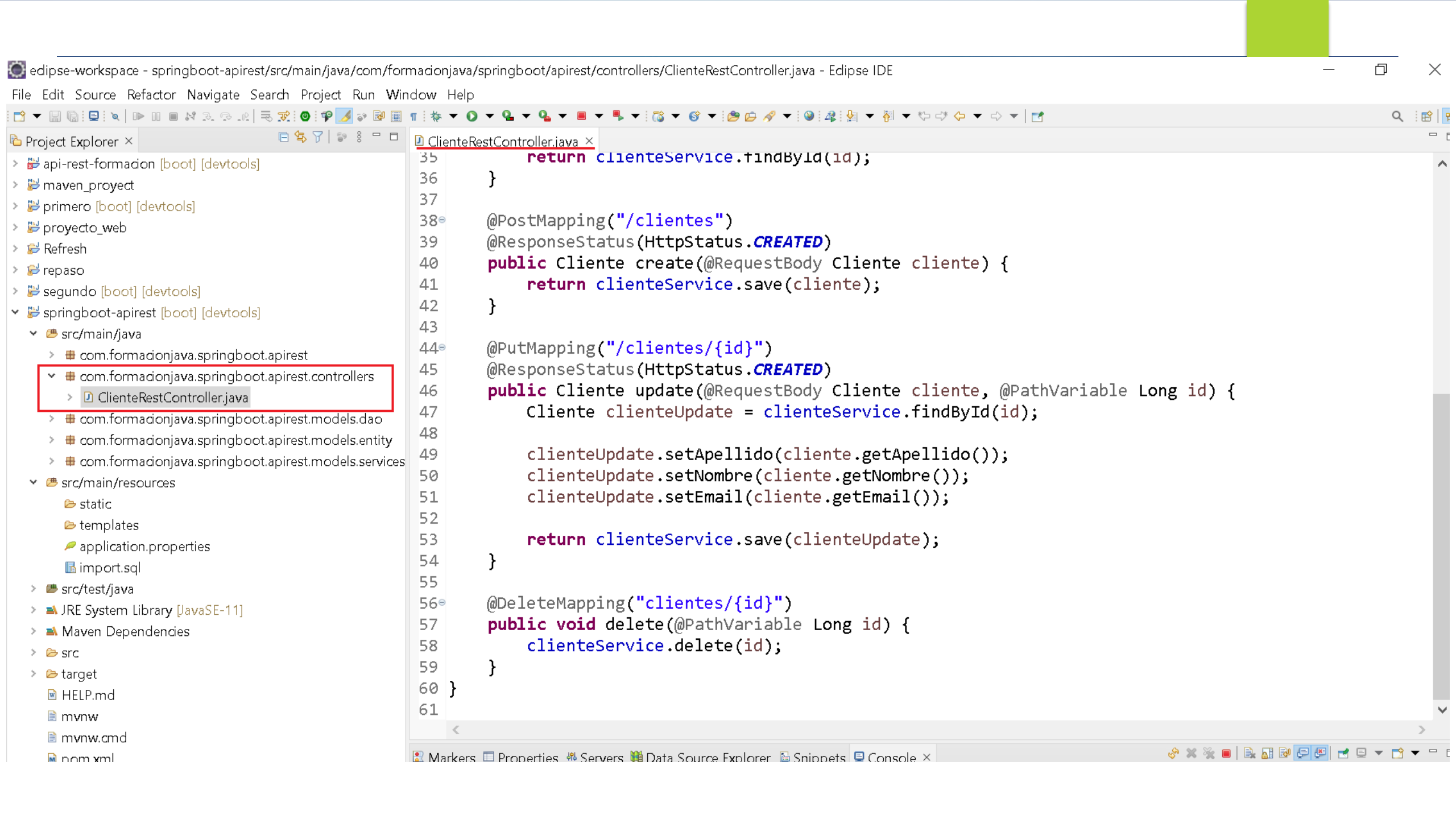

eclipse-workspace - springboot-apirest/src/main/java/com/formacionjava/springboot/apirest/models/services/ClienteServiceImpl.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
api-rest-formacion [boot] [devtools]
maven_proyect
primero [boot] [devtools]
proyecto_web
Refresh
repaso
segundo [boot] [devtools]
springboot-apirest [boot] [devtools]
src/main/java
com.formacionjava.springboot.apirest
com.formacionjava.springboot.apirest.controllers
com.formacionjava.springboot.apirest.models.dao
com.formacionjava.springboot.apirest.models.entity
com.formacionjava.springboot.apirest.models.services
ClienteService.java
ClienteServiceImpl.java
src/main/resources
static
templates
application.properties
import.sql
src/test/java
JRE System Library [JavaSE-11]
Maven Dependencies
src
target
HELP.md
mvnw
mvnw.cmd
pom.xml
ClienteService.java
ClienteServiceImpl.java
@Override
@Transactional(readOnly = true)
public Cliente findById(Long id) {
return clienteDao.findById(id).orElse(null);
}
@Override
@Transactional
public Cliente save(Cliente cliente) {
return clienteDao.save(cliente);
}
@Override
@Transactional
public void delete(Long id) {
clienteDao.deleteById(id);
}
}
Markers Properties Servers Data Source Explorer Snippets Console
springboot-apirest - SpringbootApirestApplication [Spring Boot App] C:\Users\dell\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_16.0.2.v202107

Las anotaciones utilizadas

- @Autowired es una de las anotaciones más habituales cuando trabajamos con Spring Framework ya que se trata de la anotación que permite inyectar unas dependencias con otras dentro de Spring
- @Transactional readonly es una de las características que en muchas ocasiones nos olvidamos al manejar Spring Framework. Spring nos provee de las capacidades de inyección de gestión transaccional distribuida y transparente para nuestros repositorios y servicios
- La pregunta que tenemos que hacernos . ¿En algún momento vamos a solicitar una modificación de la lista?. Es bastante evidente que en el 95% de las ocasiones no y por lo tanto podemos marcar con Spring @Transactional readonly el método para que no guarde el estado y tengamos un mejor rendimiento de la consulta

Por ultimo creamos el siguiente paquete para el controlador ClienteRestController




eclipse-workspace - springboot-apirest/src/main/java/com/formacionjava/springboot/apirest/controllers/ClienteRestController.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
api-rest-formacion [boot] [devtools]
maven_proyect
primero [boot] [devtools]
proyecto_web
Refresh
repaso
segundo [boot] [devtools]
springboot-apirest [boot] [devtools]
src/main/java
com.formacionjava.springboot.apirest
com.formacionjava.springboot.apirest.controllers
ClienteRestController.java
com.formacionjava.springboot.apirest.models.dao
com.formacionjava.springboot.apirest.models.entity
com.formacionjava.springboot.apirest.models.services
src/main/resources
static
templates
application.properties
import.sql
src/test/java
JRE System Library [JavaSE-11]
Maven Dependencies
src
target
HELP.md
mvnw
mvnw.cmd
pom.xml
ClienteRestController.java
return clienteService.findById(id);
}
@PostMapping("/clientes")
@ResponseStatus(HttpStatus.CREATED)
public Cliente create(@RequestBody Cliente cliente) {
return clienteService.save(cliente);
}
@PutMapping("/clientes/{id}")
@ResponseStatus(HttpStatus.CREATED)
public Cliente update(@RequestBody Cliente cliente, @PathVariable Long id) {
Cliente clienteUpdate = clienteService.findById(id);
clienteUpdate.setApellido(cliente.getApellido());
clienteUpdate.setNombre(cliente.getNombre());
clienteUpdate.setEmail(cliente.getEmail());
return clienteService.save(clienteUpdate);
}
@DeleteMapping("clientes/{id}")
public void delete(@PathVariable Long id) {
clienteService.delete(id);
}
}
Markers Properties Servers Data Source Explorer Snippets Console

Anotaciones utilizadas

- ***@Controller*** . Esto es simplemente una especialización de la clase *@Component* , que nos permite detectar automáticamente las clases de implementación a través del escaneo de classpath. Por lo general, usamos *@Controller* en combinación con una anotación *@RequestMapping* para los métodos de manejo de solicitudes.
- ***@RequestMapping*** En pocas palabras, la anotación se utiliza para asignar solicitudes web a los métodos de Spring Controller.
- **@GetMapping** es una versión especializada de @RequestMapping anotación actúa como un atajo para @RequestMapping(method = RequestMethod.GET)
- **@PostMapping** es una versión de @RequestMapping anotación que actúa como un acceso directo a @RequestMapping(method = requestMethod.POST)
- ***@PathVariable*** se puede usar para manejar variables de plantilla en la asignación de URI de solicitud
- ***@RequestBody*** asigna el cuerpo *HttpRequest* a un objeto de transferencia o dominio, lo que permite la deserialización automática del cuerpo *HttpRequest* entrante en un objeto Java.
- @ResponseStatus marca un método o clase de excepción con el código de estado y el mensaje de motivo de devolverse. El código de estado se aplica a la respuesta HTTP cuando se invoca el método controlador o siempre que se lanza la excepción especifica.

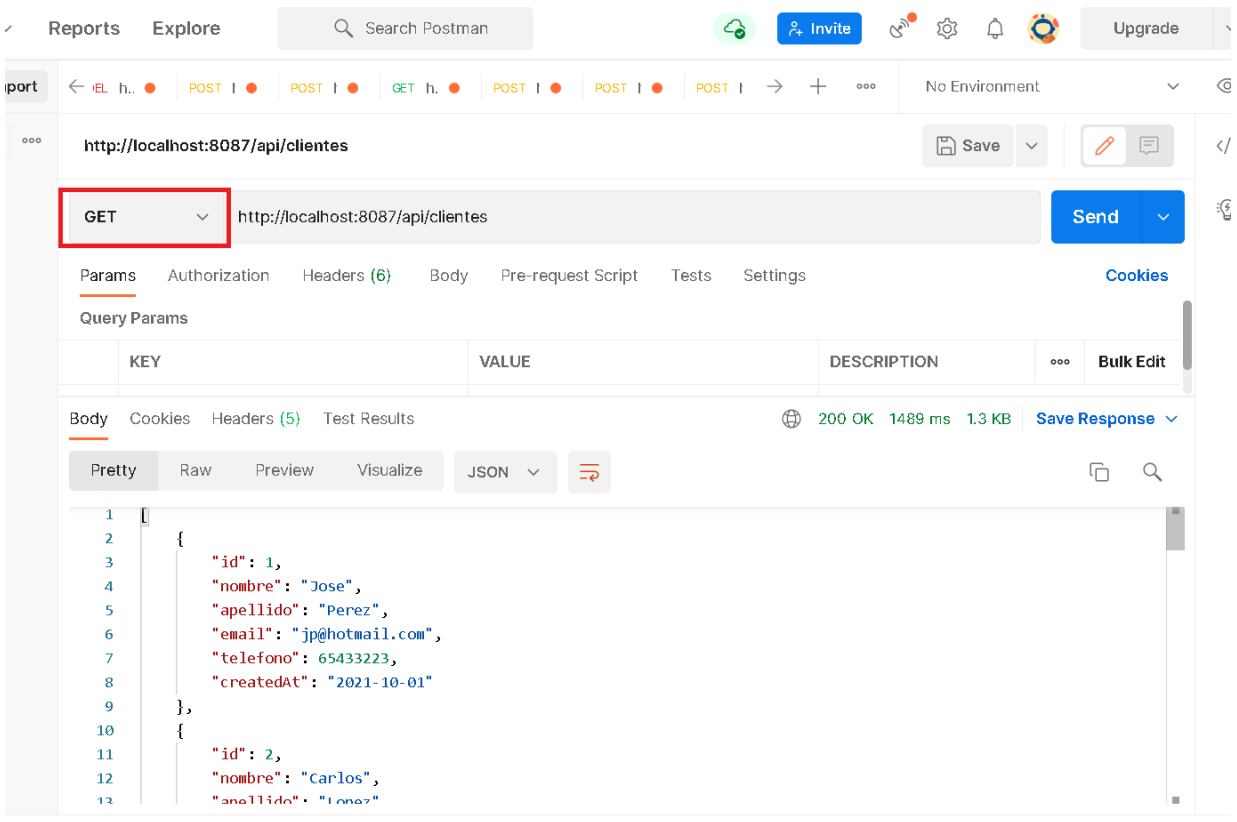
Las url de las apis que hemos creado

- GET http://localhost:8087/api/clientes
- POST http://localhost:8087/api/clientes
- GET http://localhost:8087/api/clientes/{id}
- DELETE http://localhost:8087/api/clientes/{id}

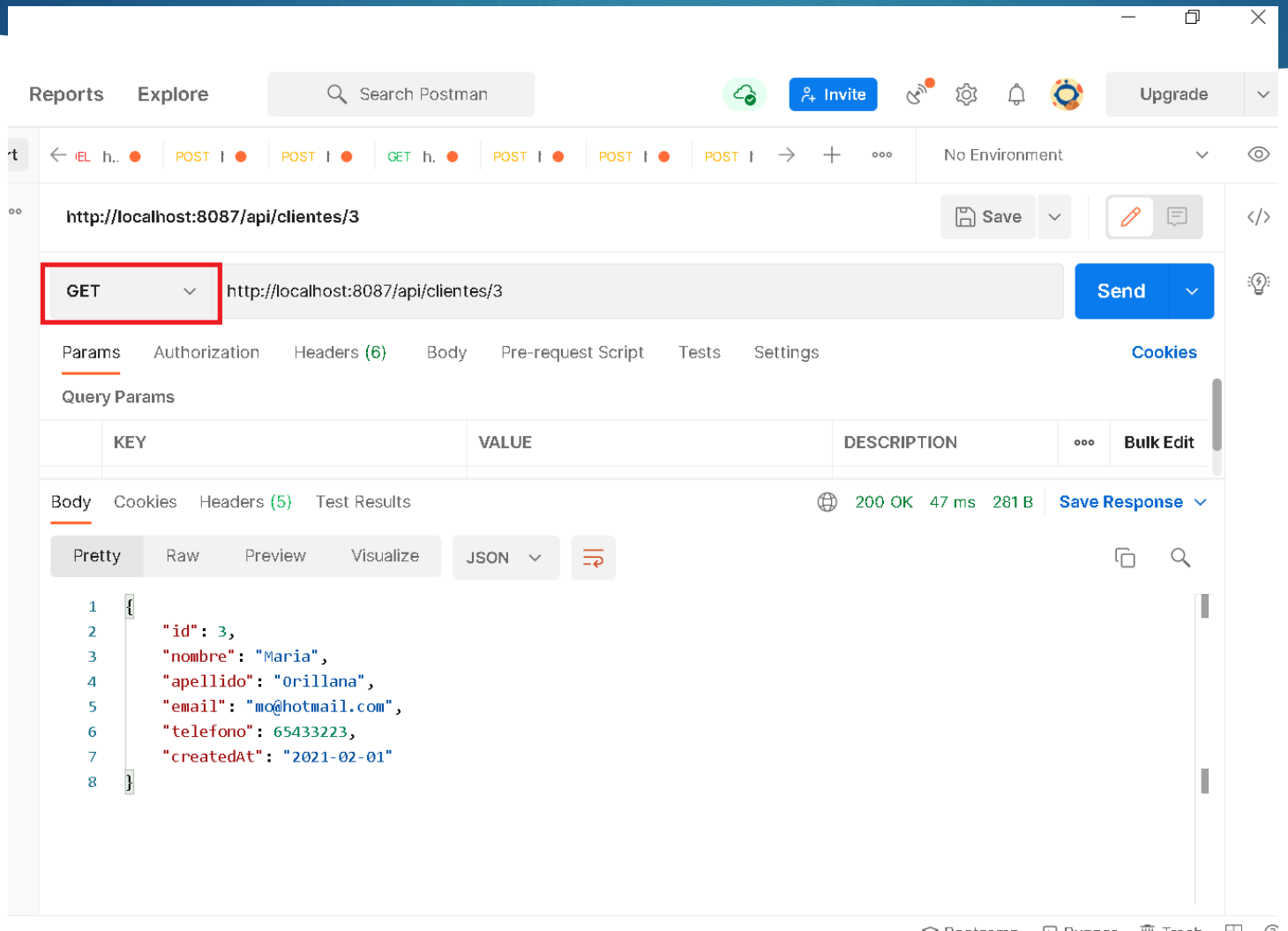
- OBSERVACION: EL PUERTO SERIA EL QUE ESTE FUNCIONANDO PARA USTEDES

Lo probamos con Postman

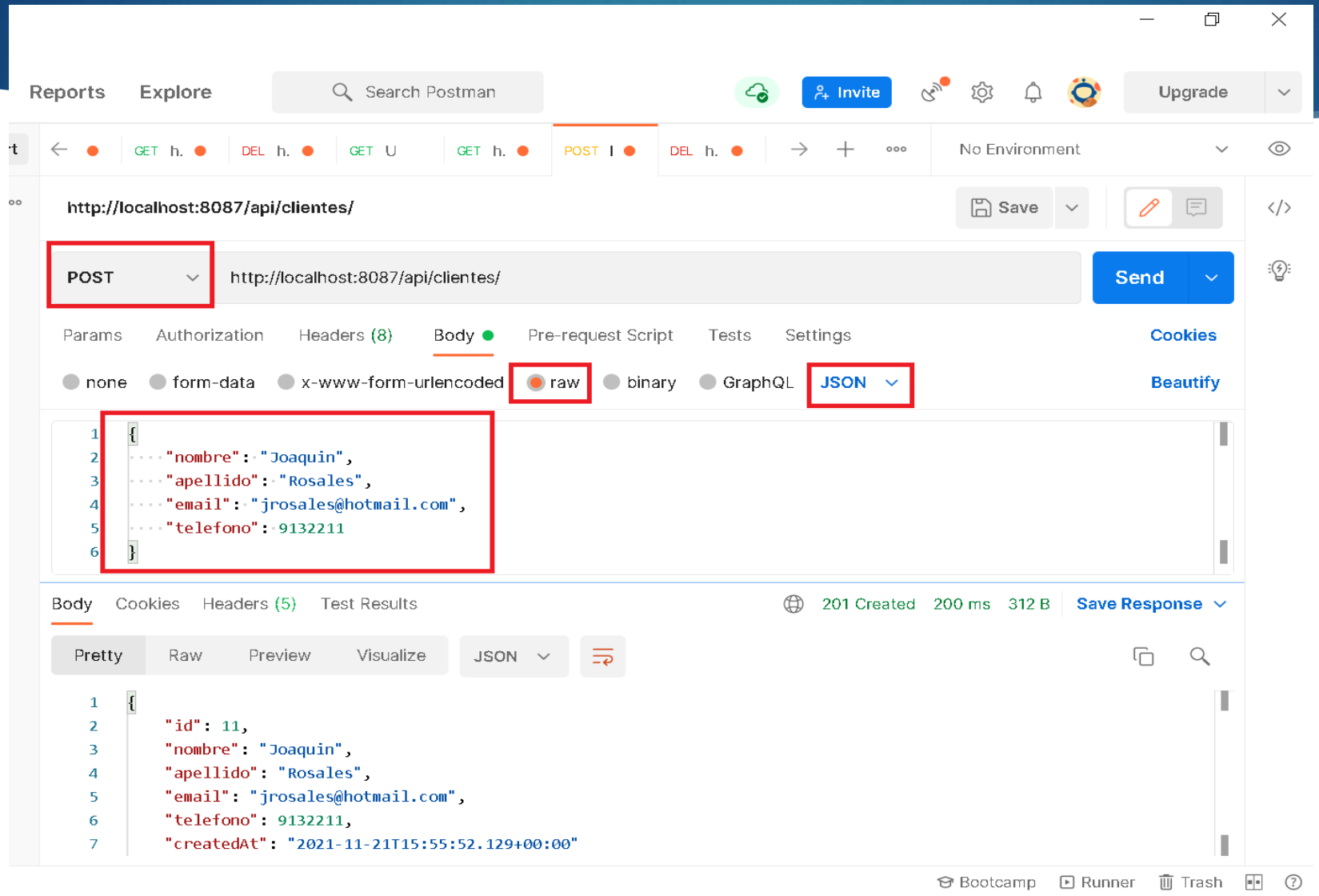
- PROBAR API GET



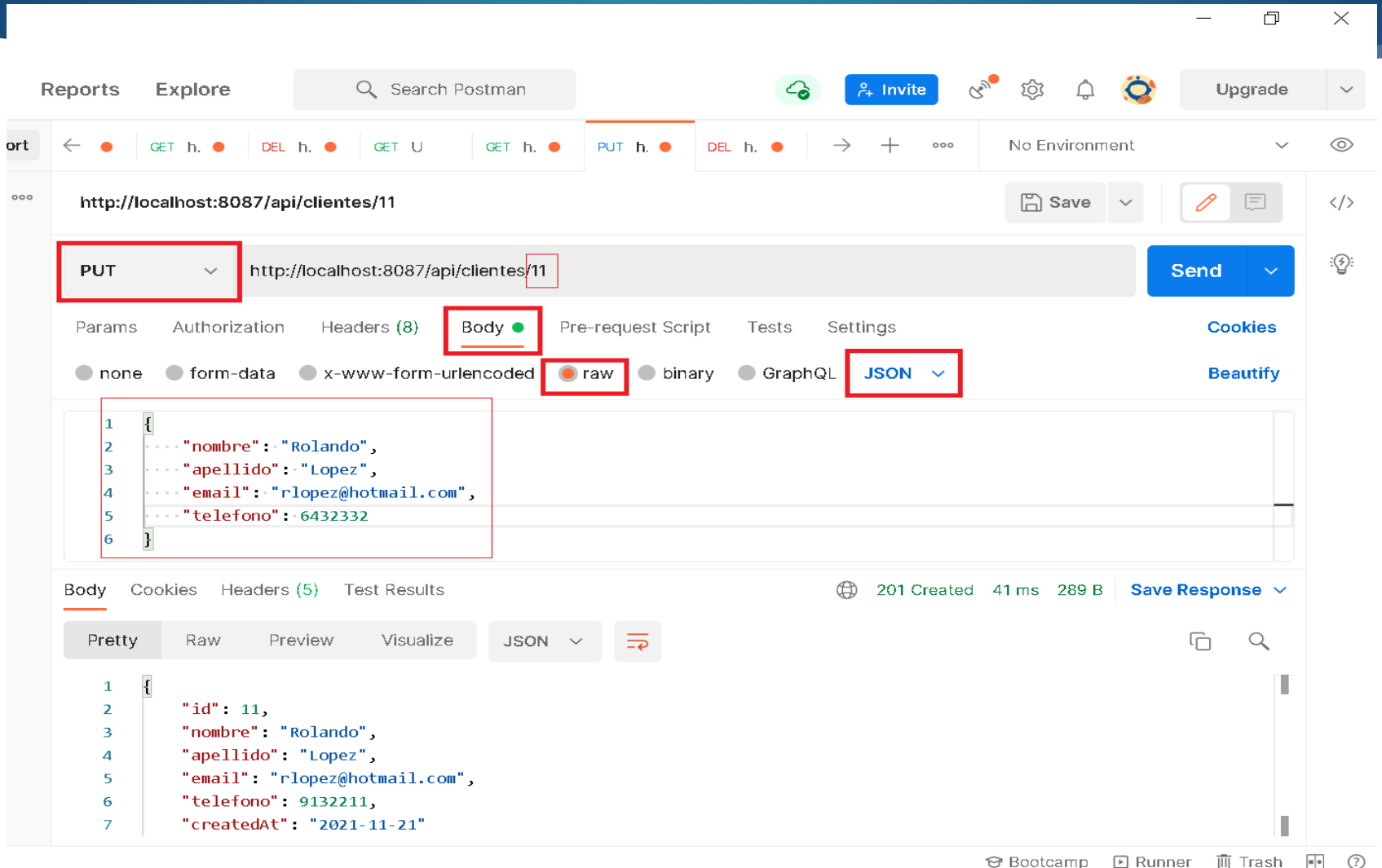
Búsqueda por id



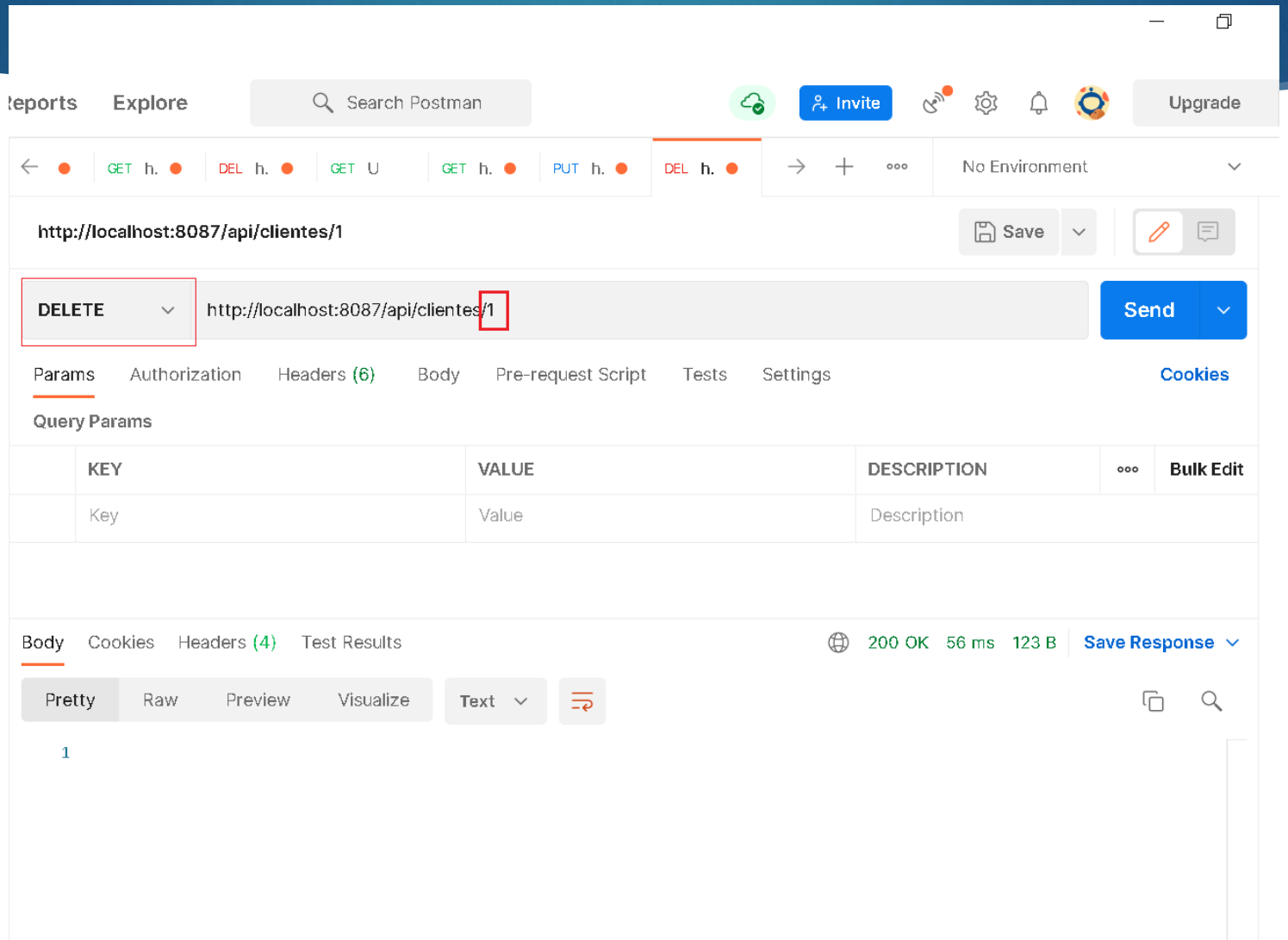
Probamos el método POST



Método PUT

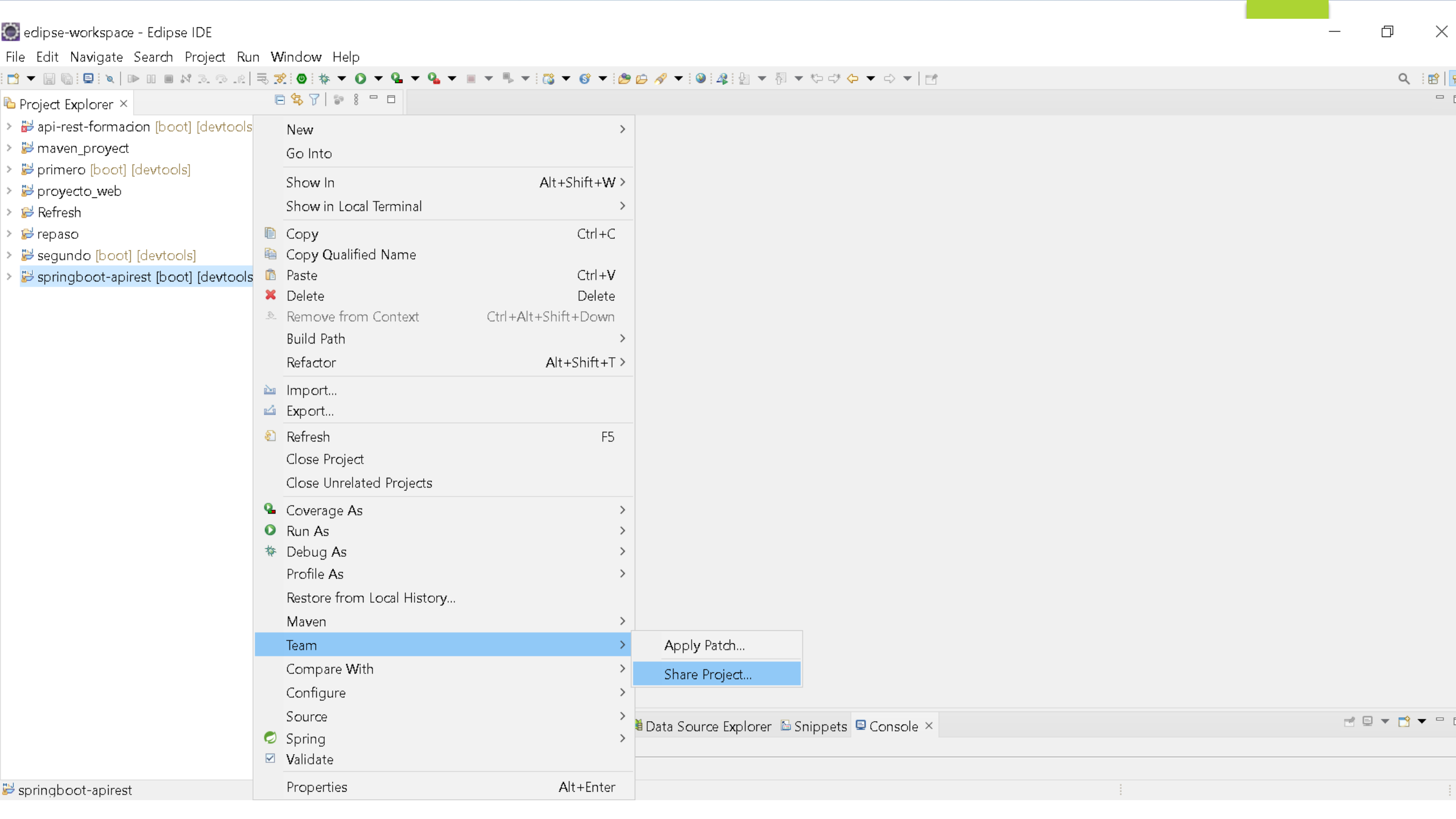

Reports
Explore
Search Postman
Invite
Upgrade
GET h.
DEL h.
GET U
GET h.
PUT h.
DEL h.
No Environment
http://localhost:8087/api/clientes/11
Save
PUT
http://localhost:8087/api/clientes/11
Send
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"nombre": "Rolando",
"apellido": "Lopez",
"email": "rlopez@hotmail.com",
"telefono": 6432332
}
Body
Cookies
Headers (5)
Test Results
201 Created
41 ms
289 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"id": 11,
"nombre": "Rolando",
"apellido": "Lopez",
"email": "rlopez@hotmail.com",
"telefono": 9132211,
"createdAt": "2021-11-21"
Bootcamp
Runner
Trash

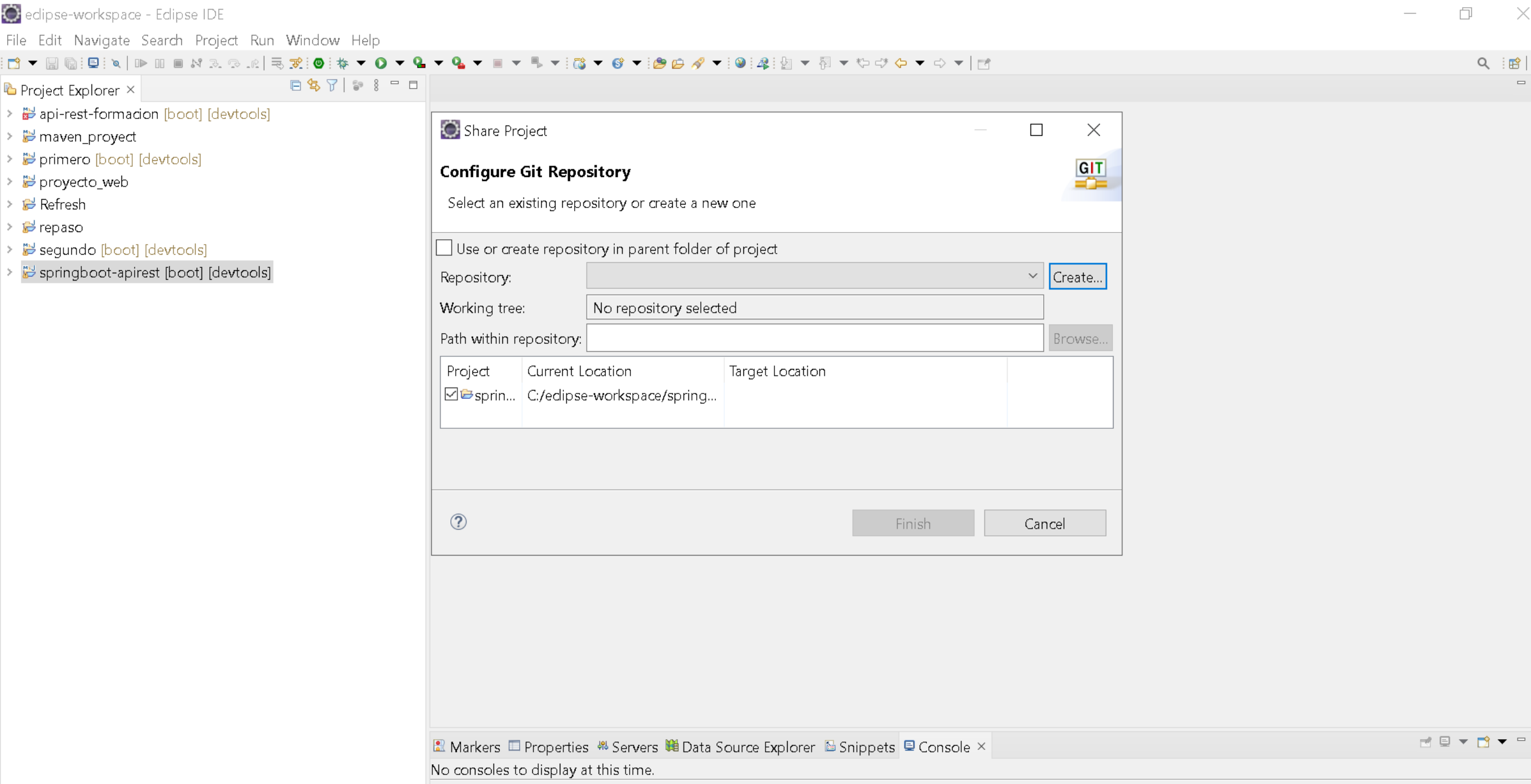
Método DELETE

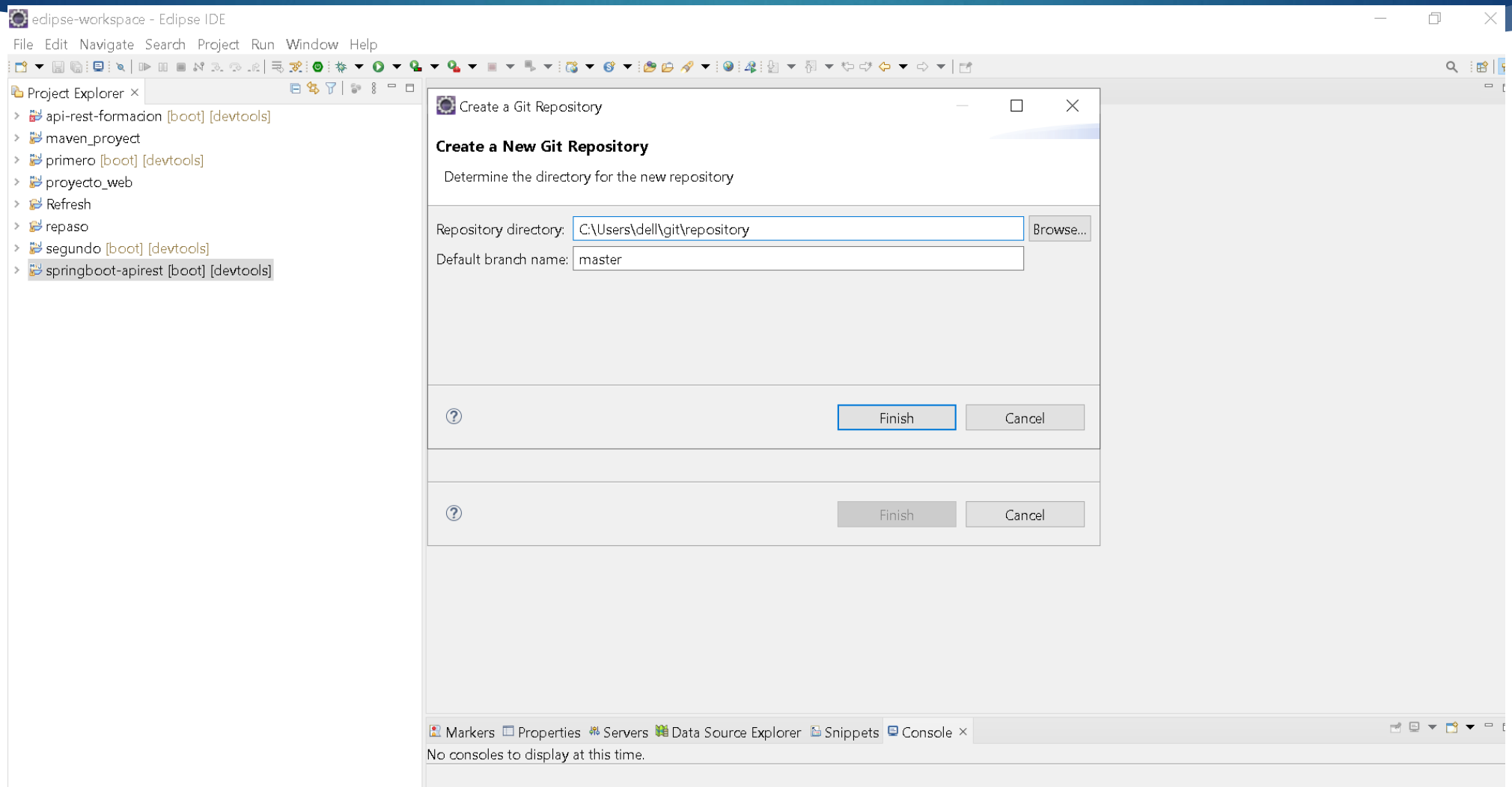


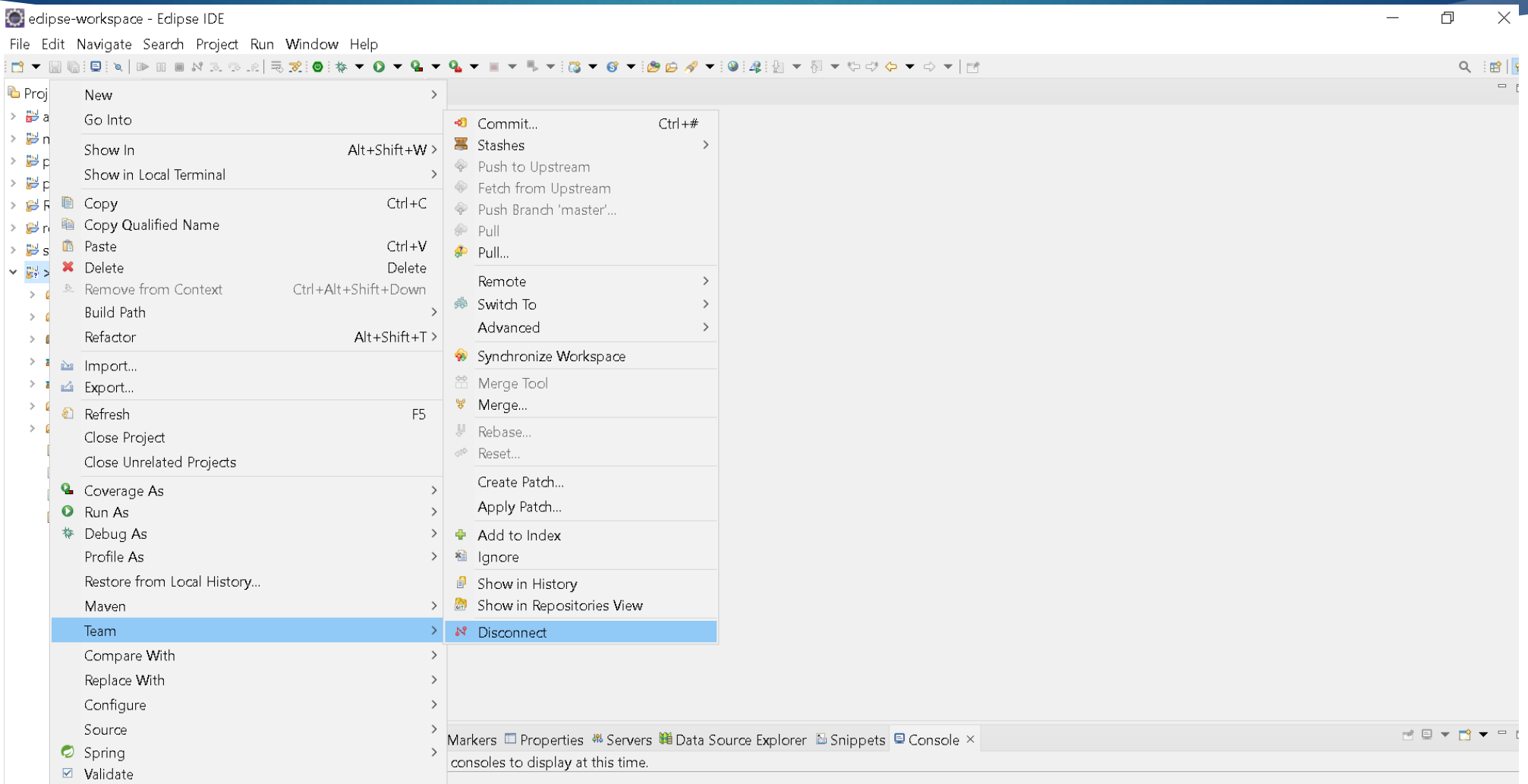
GIT

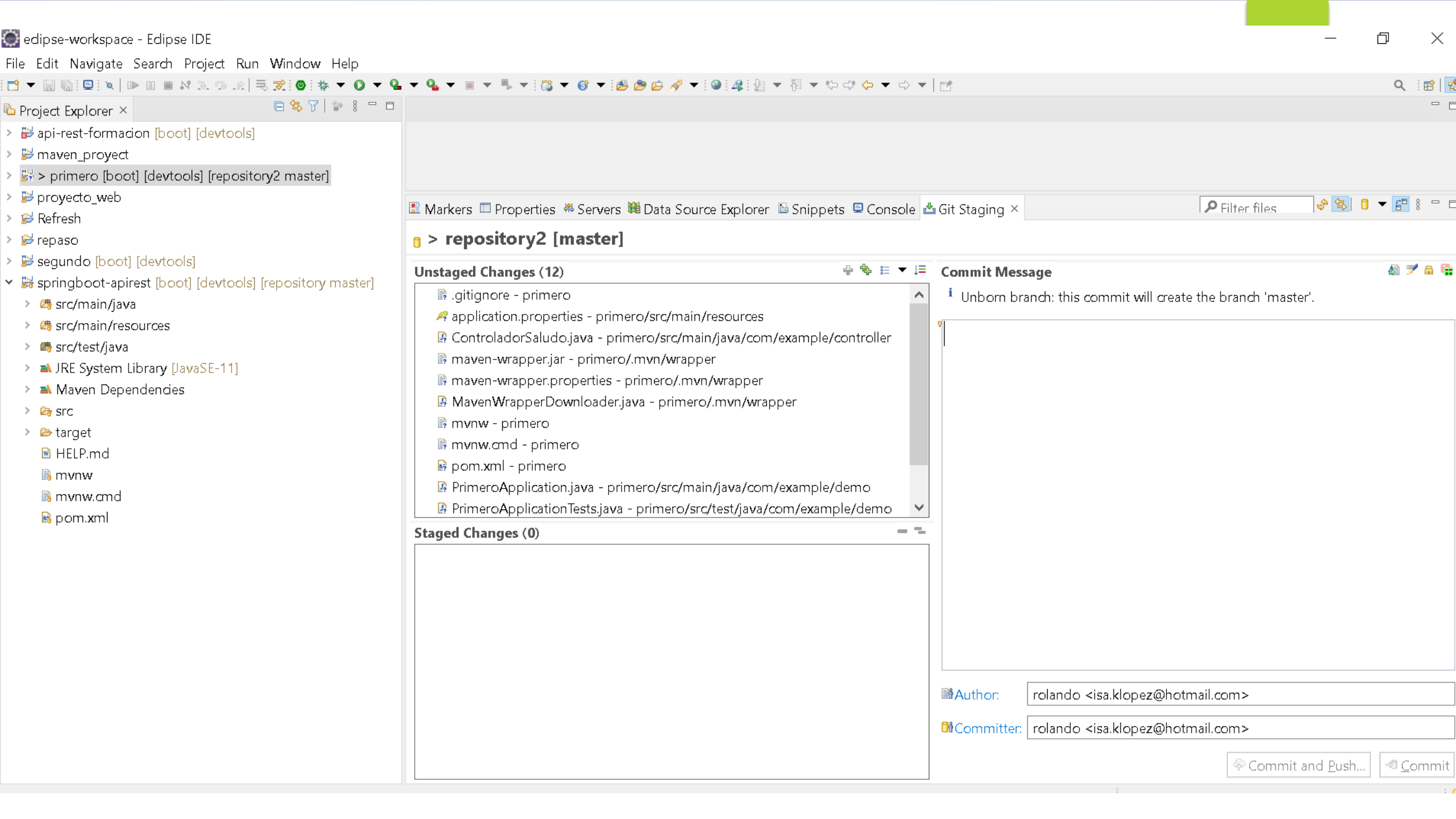
- Subir proyecto a repositorio remoto desde eclipse

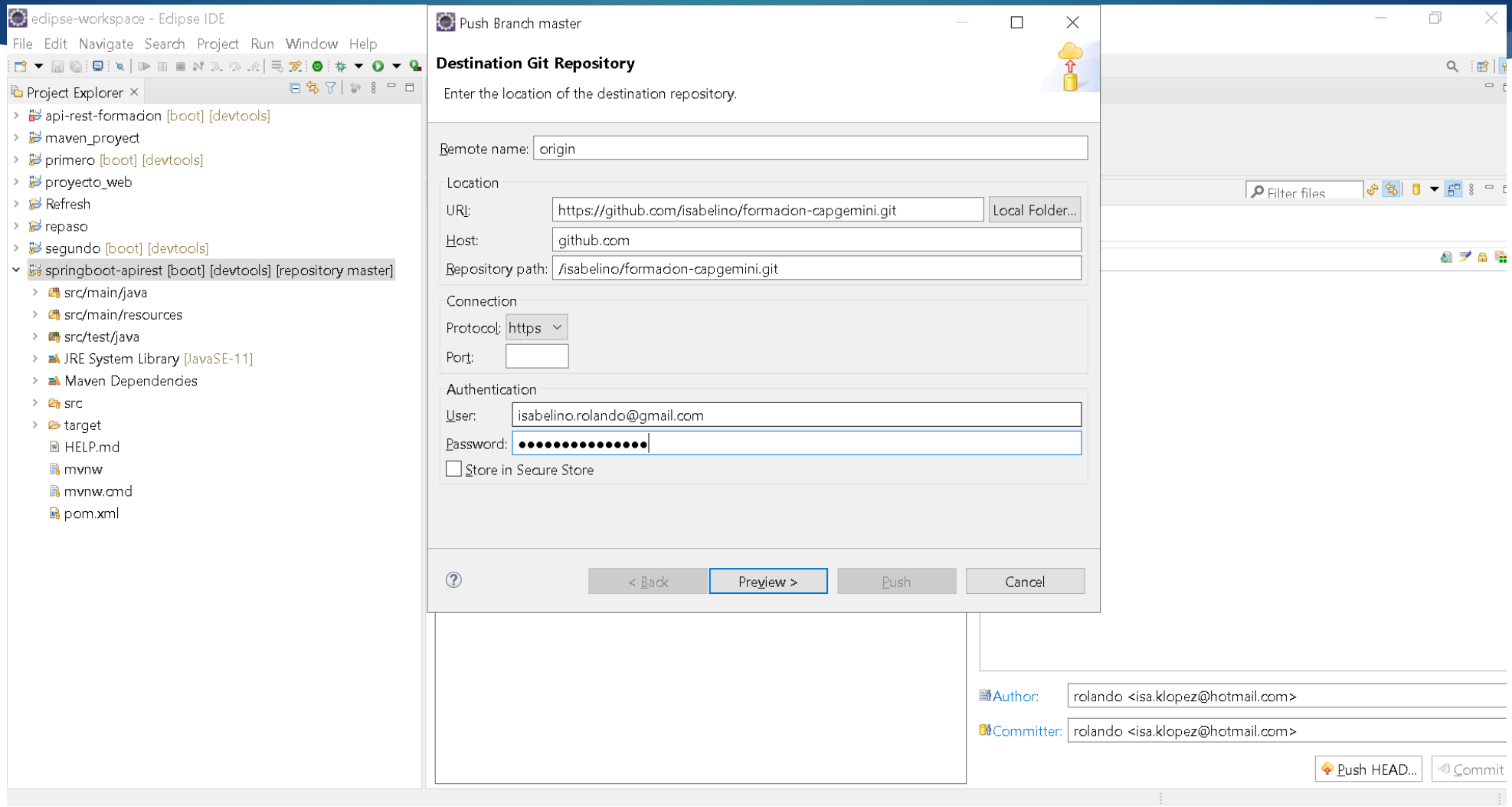

edipse-workspace - Edipse IDE
File Edit Navigate Search Project Run Window Help
Project Explorer
api-rest-formacion [boot] [devtools]
maven_proyect
primero [boot] [devtools]
proyecto_web
Refresh
repaso
segundo [boot] [devtools]
springboot-apirest [boot] [devtools]
New
Go Into
Show In Alt+Shift+W
Show in Local Terminal
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Remove from Context Ctrl+Alt+Shift+Down
Build Path
Refactor Alt+Shift+T
Import...
Export...
Refresh F5
Close Project
Close Unrelated Projects
Coverage As
Run As
Debug As
Profile As
Restore from Local History...
Maven
Team
Compare With
Configure
Source
Spring
Validate
Properties Alt+Enter
Apply Patch...
Share Project...
Data Source Explorer
Snippets
Console
springboot-apirest


edipse-workspace - Eclipse IDE
File Edit Navigate Search Project Run Window Help
Project Explorer
api-rest-formacion [boot] [devtools]
maven_proyect
primero [boot] [devtools]
proyecto_web
Refresh
repaso
segundo [boot] [devtools]
springboot-apirest [boot] [devtools]
Share Project
Configure Git Repository
Select an existing repository or create a new one
GIT
Use or create repository in parent folder of project
Repository:
Create...
Working tree:
No repository selected
Path within repository:
Browse...
Project
Current Location
Target Location
sprin...
C:/edipse-workspace/spring...
Finish
Cancel
Markers
Properties
Servers
Data Source Explorer
Snippets
Console
No consoles to display at this time.

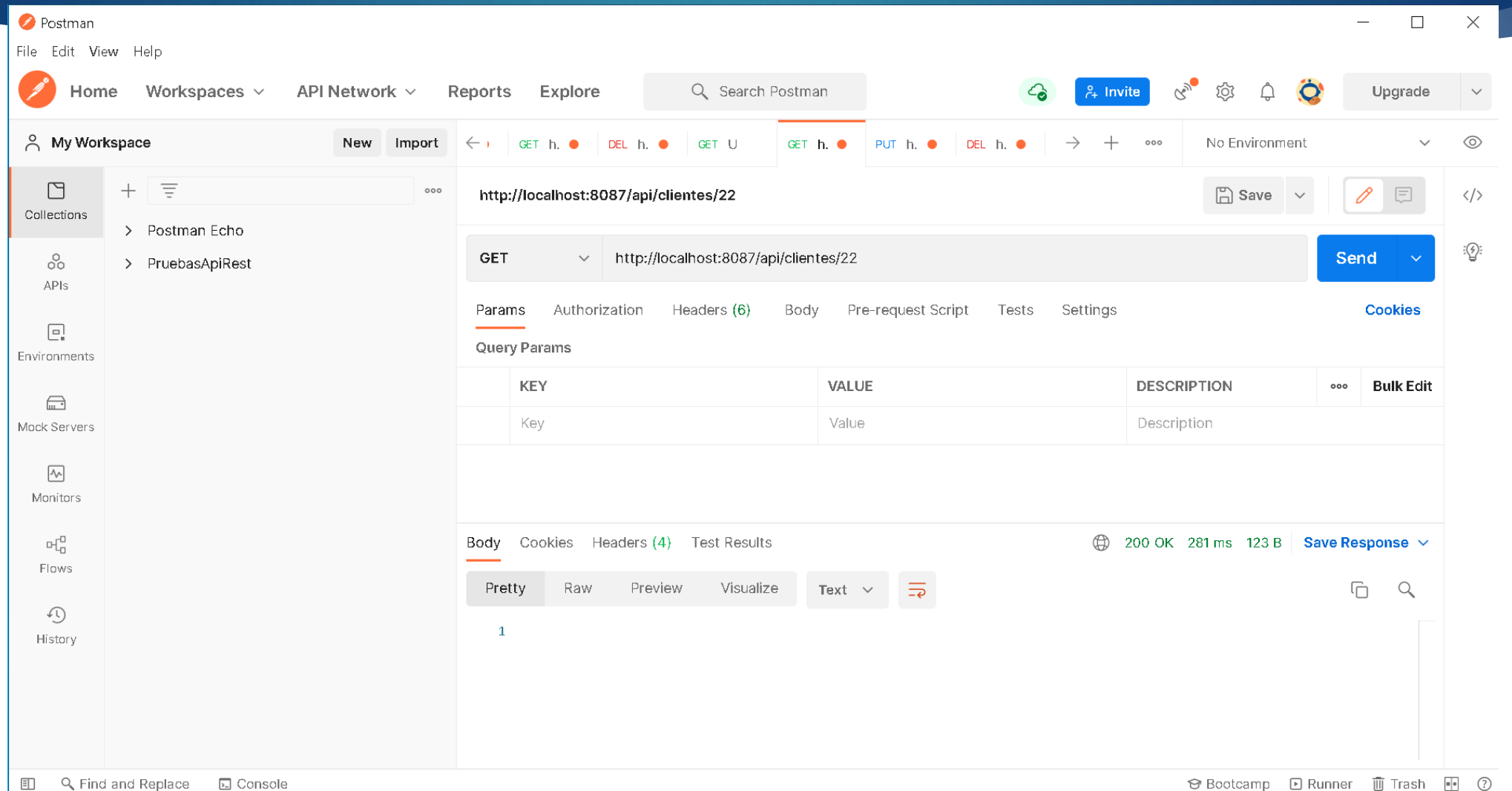

edipse-workspace - Edipse IDE
File Edit Navigate Search Project Run Window Help
Project Explorer
api-rest-formacion [boot] [devtools]
maven_proyect
primero [boot] [devtools]
proyecto_web
Refresh
repaso
segundo [boot] [devtools]
springboot-apirest [boot] [devtools]
Create a Git Repository
Create a New Git Repository
Determine the directory for the new repository
Repository directory:
C:\Users\dell\git\repository
Browse...
Default branch name:
master
Finish
Cancel
Finish
Cancel
Markers
Properties
Servers
Data Source Explorer
Snippets
Console
No consoles to display at this time.


edipse-workspace - Edipse IDE
File Edit Navigate Search Project Run Window Help
New
Go Into
Show In Alt+Shift+W
Show in Local Terminal
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Remove from Context Ctrl+Alt+Shift+Down
Build Path
Refactor Alt+Shift+T
Import...
Export...
Refresh F5
Close Project
Close Unrelated Projects
Coverage As
Run As
Debug As
Profile As
Restore from Local History...
Maven
Team
Compare With
Replace With
Configure
Source
Spring
Validate
Commit... Ctrl+#
Stashes
Push to Upstream
Fetch from Upstream
Push Branch 'master'...
Pull
Pull...
Remote
Switch To
Advanced
Synchronize Workspace
Merge Tool
Merge...
Rebase...
Reset...
Create Patch...
Apply Patch...
Add to Index
Ignore
Show in History
Show in Repositories View
Disconnect
Markers Properties Servers Data Source Explorer Snippets Console
consoles to display at this time.


edipse-workspace - Eclipse IDE
File Edit Navigate Search Project Run Window Help
Project Explorer
api-rest-formacion [boot] [devtools]
maven_proyect
> primero [boot] [devtools] [repository2 master]
proyecto_web
Refresh
repaso
segundo [boot] [devtools]
springboot-apirest [boot] [devtools] [repository master]
src/main/java
src/main/resources
src/test/java
JRE System Library [JavaSE-11]
Maven Dependencies
src
target
HELP.md
mvnw
mvnw.cmd
pom.xml
Markers Properties Servers Data Source Explorer Snippets Console Git Staging
Filter files
> repository2 [master]
Unstaged Changes (12)
.gitignore - primero
application.properties - primero/src/main/resources
ControladorSaludo.java - primero/src/main/java/com/example/controller
maven-wrapper.jar - primero/.mvn/wrapper
maven-wrapper.properties - primero/.mvn/wrapper
MavenWrapperDownloader.java - primero/.mvn/wrapper
mvnw - primero
mvnw.cmd - primero
pom.xml - primero
PrimeroApplication.java - primero/src/main/java/com/example/demo
PrimeroApplicationTests.java - primero/src/test/java/com/example/demo
Staged Changes (0)
Commit Message
Unborn branch: this commit will create the branch 'master'.
Author: rolando <isa.klopez@hotmail.com>
Committer: rolando <isa.klopez@hotmail.com>
Commit and Push...
Commit

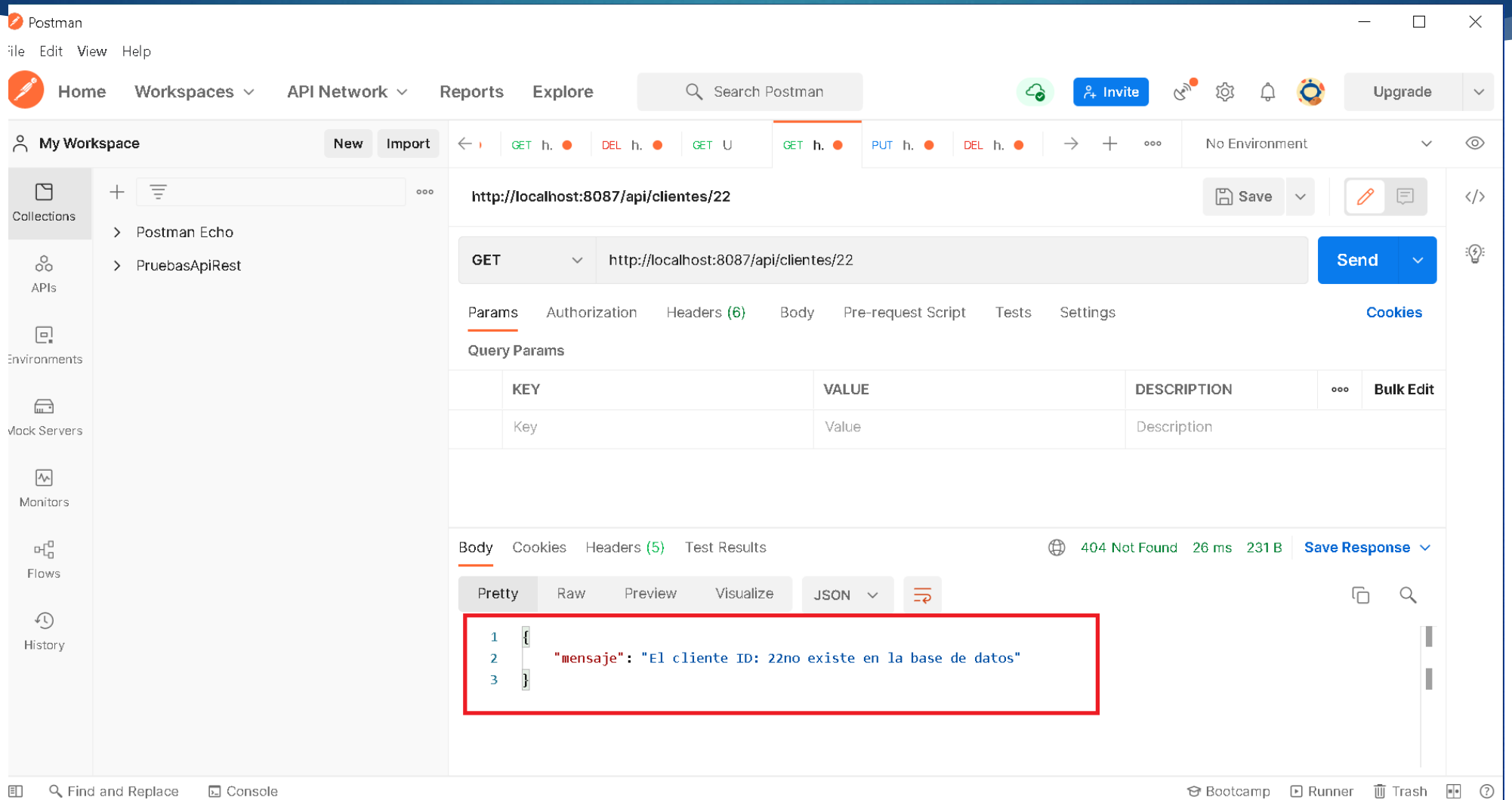

eclipse-workspace - Eclipse IDE
File Edit Navigate Search Project Run Window Help
Project Explorer
api-rest-formacion [boot] [devtools]
maven_proyect
primero [boot] [devtools]
proyecto_web
Refresh
repaso
segundo [boot] [devtools]
springboot-apirest [boot] [devtools] [repository master]
src/main/java
src/main/resources
src/test/java
JRE System Library [JavaSE-11]
Maven Dependencies
src
target
HELP.md
mvnw
mvnw.cmd
pom.xml
Push Branch master
Destination Git Repository
Enter the location of the destination repository.
Remote name: origin
Location
URL: https://github.com/isabelino/formacion-capgemini.git
Local Folder...
Host: github.com
Repository path: /isabelino/formacion-capgemini.git
Connection
Protocol: https
Port:
Authentication
User: isabelino.rolando@gmail.com
Password: ••••••••••••••••
Store in Secure Store
< Back
Preview >
Push
Cancel
Filter files
Author: rolando <isa.klopez@hotmail.com>
Committer: rolando <isa.klopez@hotmail.com>
Push HEAD...
Commit

Manejo de Errores

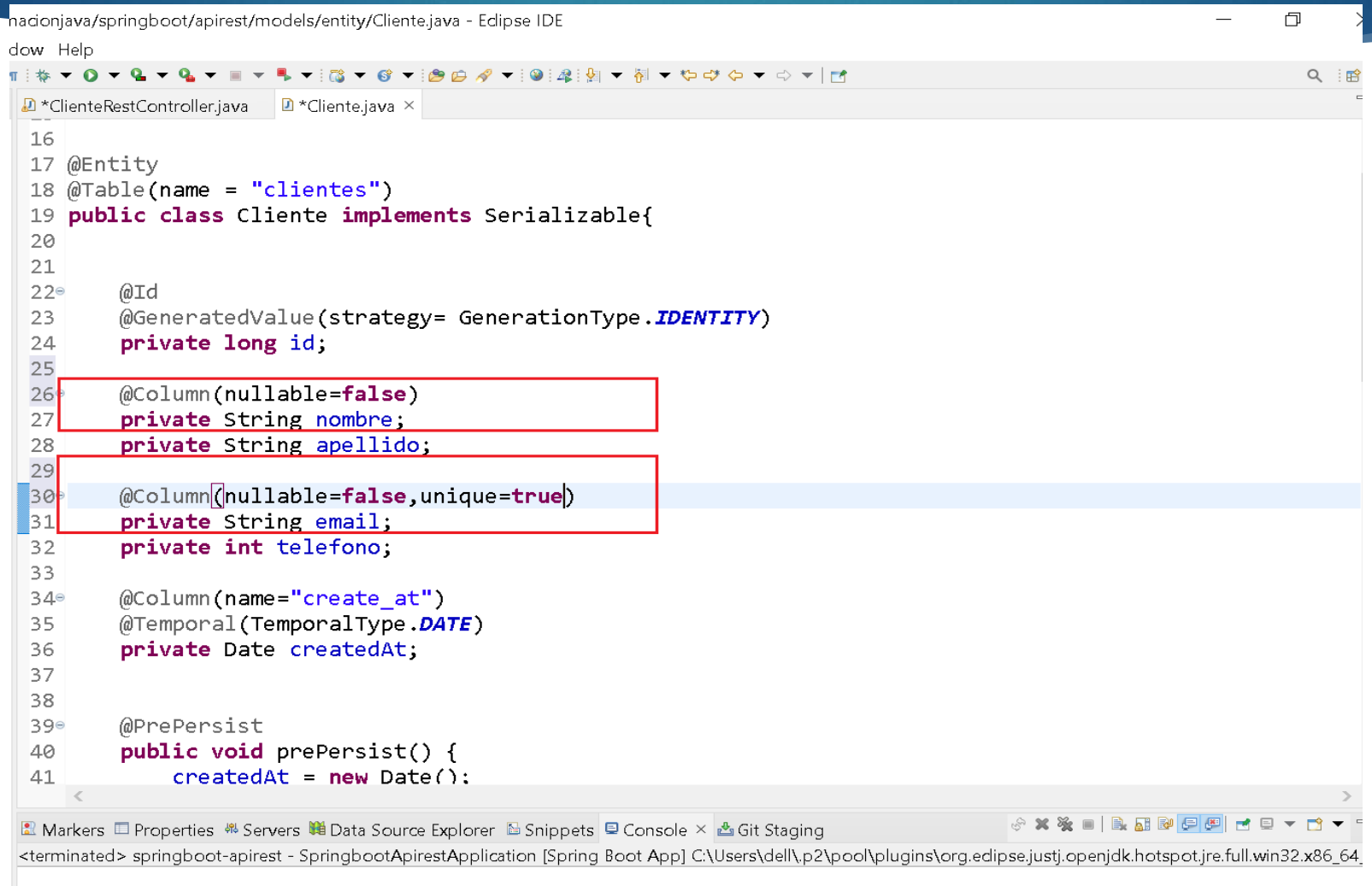


*ClienteRestController.java

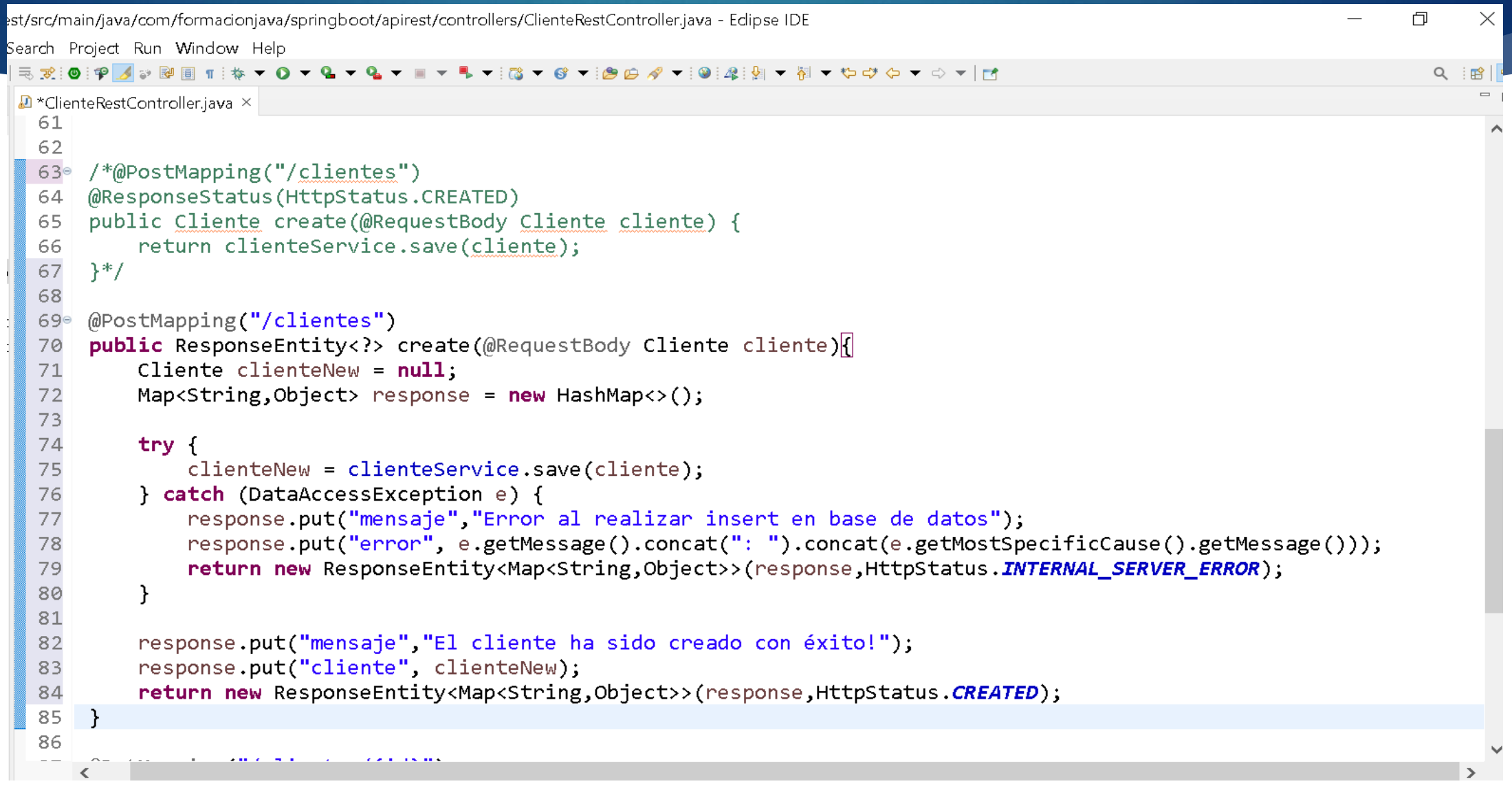
```
36
37      /*@GetMapping("/clientes/{id}")
38      public Cliente show(@PathVariable Long id) {
39          return clienteService.findById(id);
40      }*/
41      @GetMapping("/clientes/{id}")
42      public ResponseEntity<?> show(@PathVariable Long id){
43          Cliente cliente = null;
44          Map<String,Object> response = new HashMap<>();
45
46          try {
47              cliente = clienteService.findById(id);
48          } catch (DataAccessException e) {
49              response.put("mensaje","Error al realizar consulta en base de datos");
50              response.put("error", e.getMessage().concat(": ").concat(e.getMostSpecificCause().getMessage()));
51              return new ResponseEntity<Map<String,Object>>(response,HttpStatus.INTERNAL_SERVER_ERROR);
52          }
53
54          if(cliente == null) {
55              response.put("mensaje", "El cliente ID: ".concat(id.toString().concat("no existe en la base de datos")));
56              return new ResponseEntity<Map<String,Object>>(response,HttpStatus.NOT_FOUND);
57          }
58
59          return new ResponseEntity<Cliente>(cliente,HttpStatus.OK);
60      }
61
```


Postman
File Edit View Help
Home Workspaces API Network Reports Explore
Search Postman
Invite
Upgrade
My Workspace
New Import
Collections
APIs
Environments
Mock Servers
Monitors
Flows
History
Postman Echo
PruebasApiRest
GET h.
DEL h.
GET U
GET h.
PUT h.
DEL h.
No Environment
http://localhost:8087/api/clientes/22
Save
GET
http://localhost:8087/api/clientes/22
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
404 Not Found 26 ms 231 B
Save Response
Pretty Raw Preview Visualize JSON
1 {
2 "mensaje": "El cliente ID: 22no existe en la base de datos"
3 }
Find and Replace
Console
Bootcamp
Runner
Trash

En nuestra entidad cliente programamos que el campo nombre no sea nulo, y el campo email no sea nulo y sea único

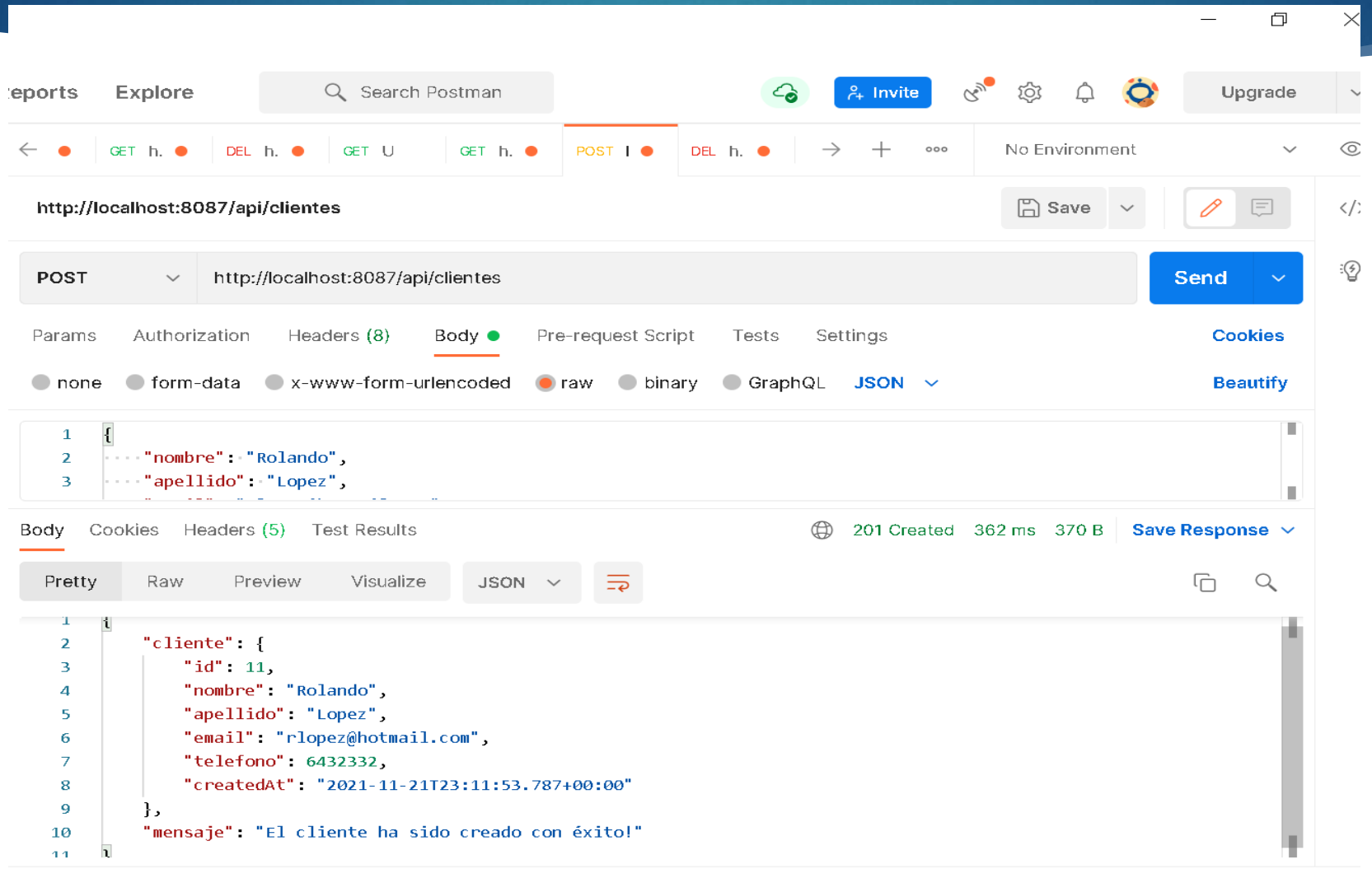


Ahora agregamos a insertar

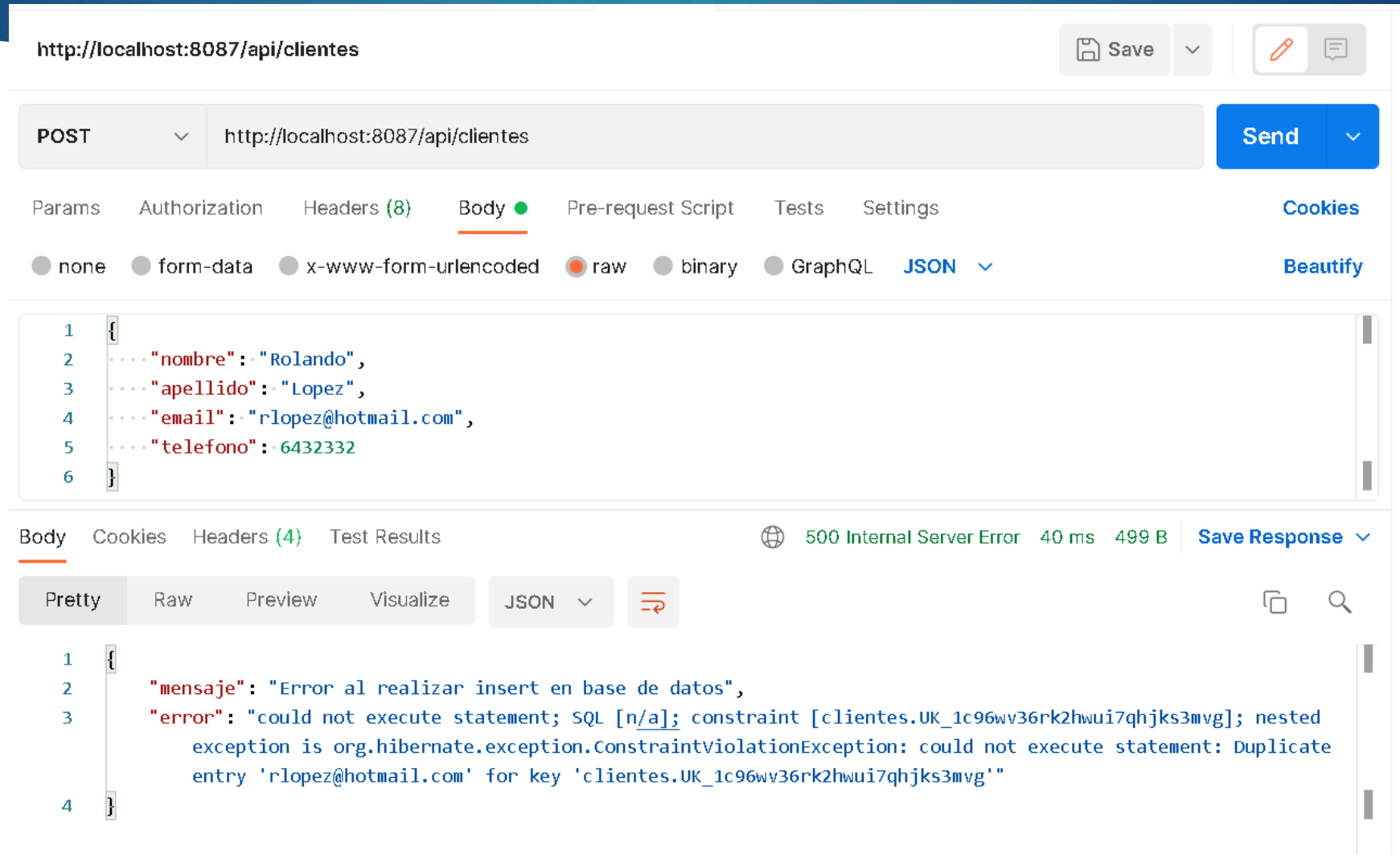



```java
/*@PostMapping("/clientes")
@ResponseStatus(HttpStatus.CREATED)
public Cliente create(@RequestBody Cliente cliente) {
    return clienteService.save(cliente);
}*/

@PostMapping("/clientes")
public ResponseEntity<?> create(@RequestBody Cliente cliente){
    Cliente clienteNew = null;
    Map<String,Object> response = new HashMap<>();

    try {
        clienteNew = clienteService.save(cliente);
    } catch (DataAccessException e) {
        response.put("mensaje","Error al realizar insert en base de datos");
        response.put("error", e.getMessage().concat(": ").concat(e.getMostSpecificCause().getMessage()));
        return new ResponseEntity<Map<String,Object>>(response,HttpStatus.INTERNAL_SERVER_ERROR);
    }

    response.put("mensaje","El cliente ha sido creado con éxito!");
    response.put("cliente", clienteNew);
    return new ResponseEntity<Map<String,Object>>(response,HttpStatus.CREATED);
}
```

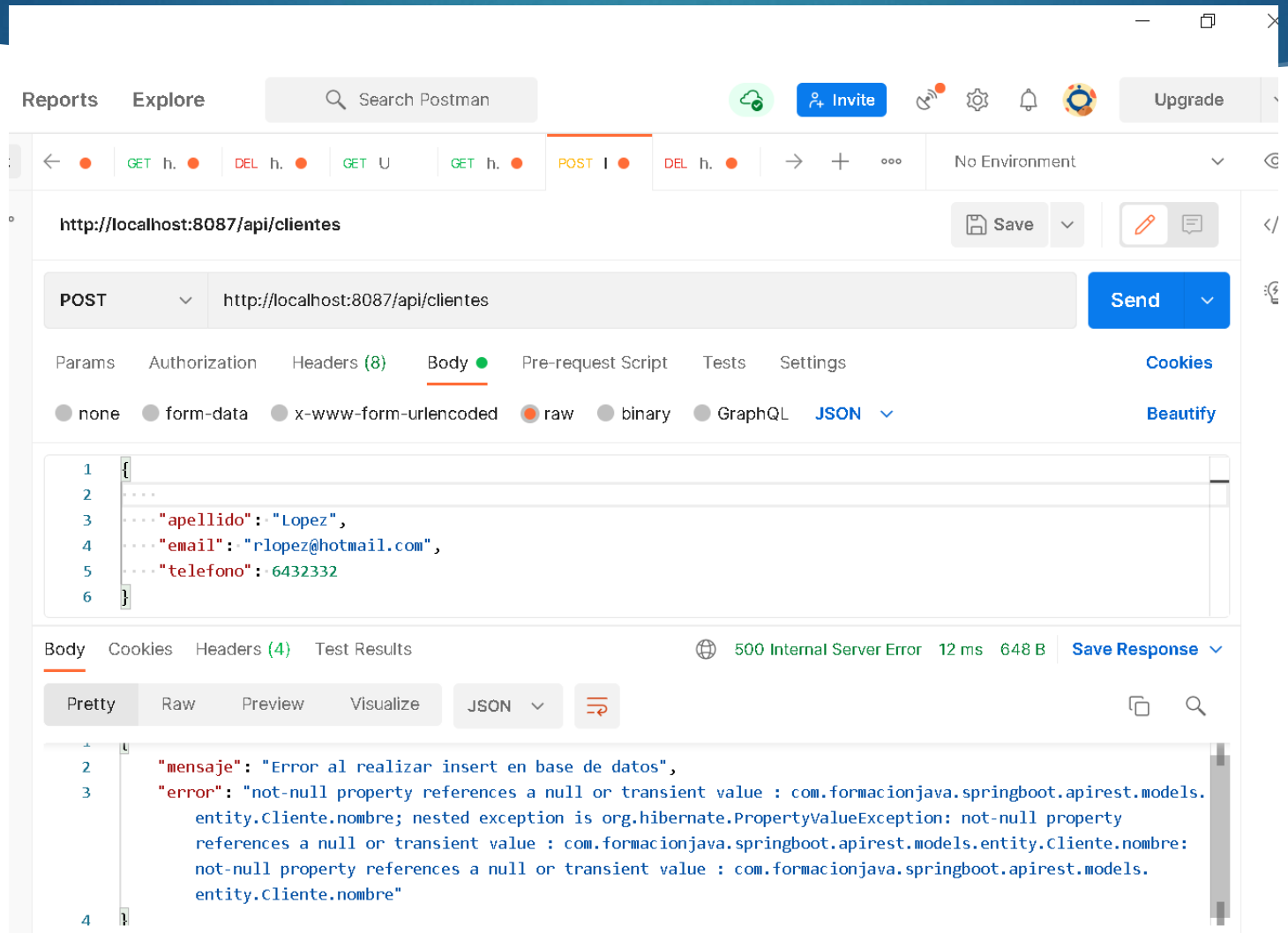
Comprobamos las excepciones



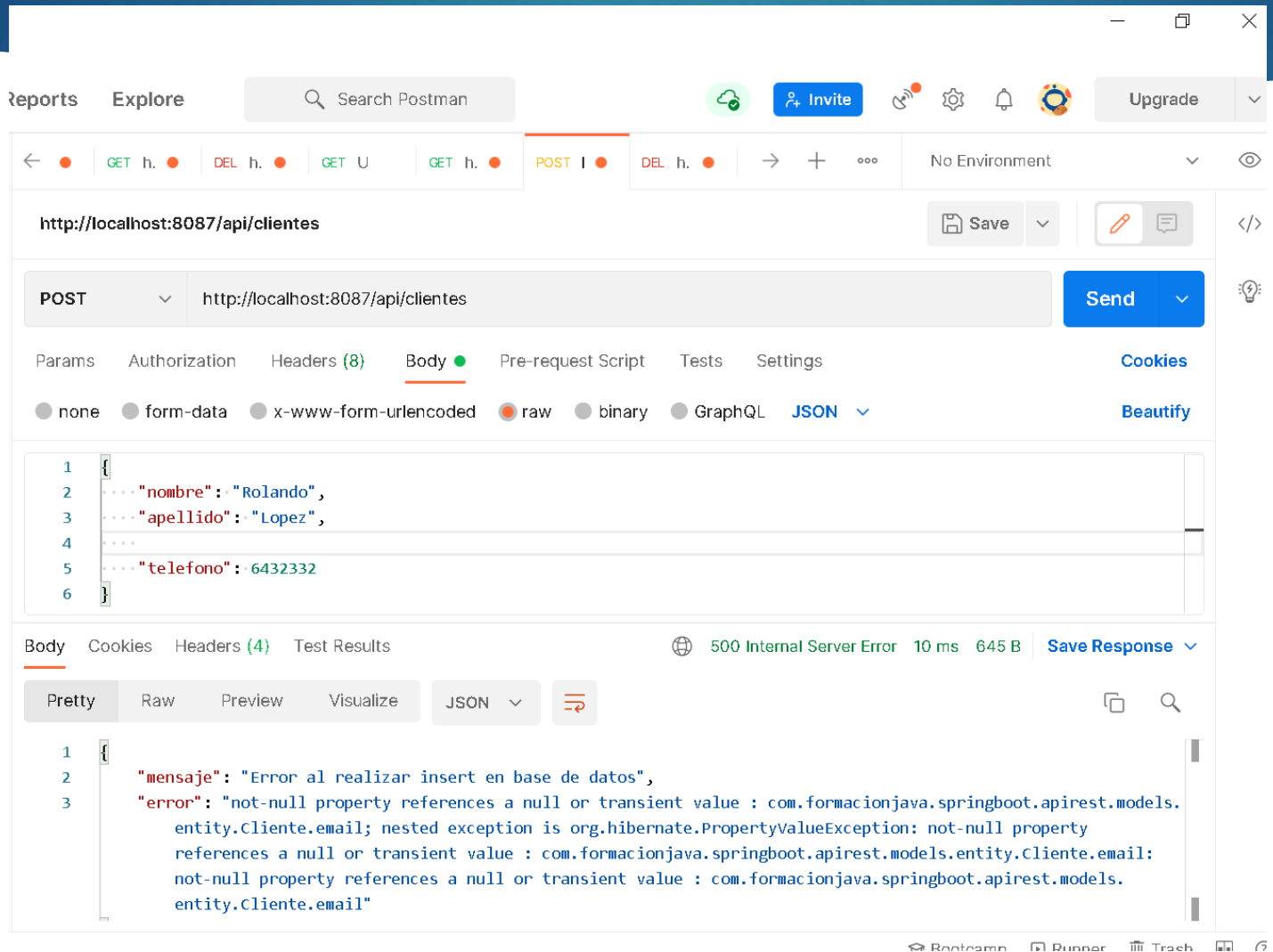
No nos deja registrar un usuario con un mismo email por la propiedad unique



Si no pasamos el campo nombre también nos informa que es requerido



También el campo email es requerido

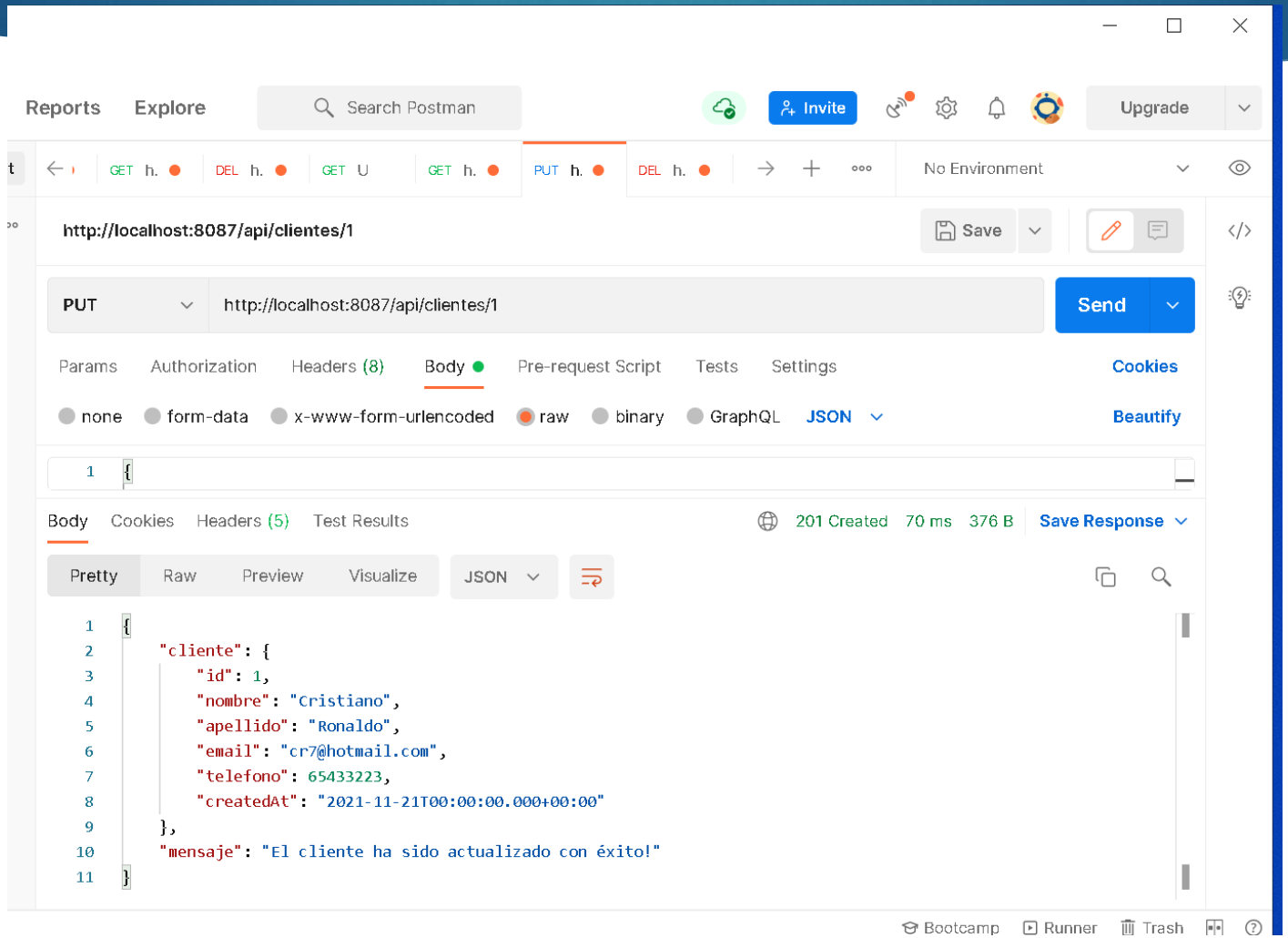


Ahora para el Update

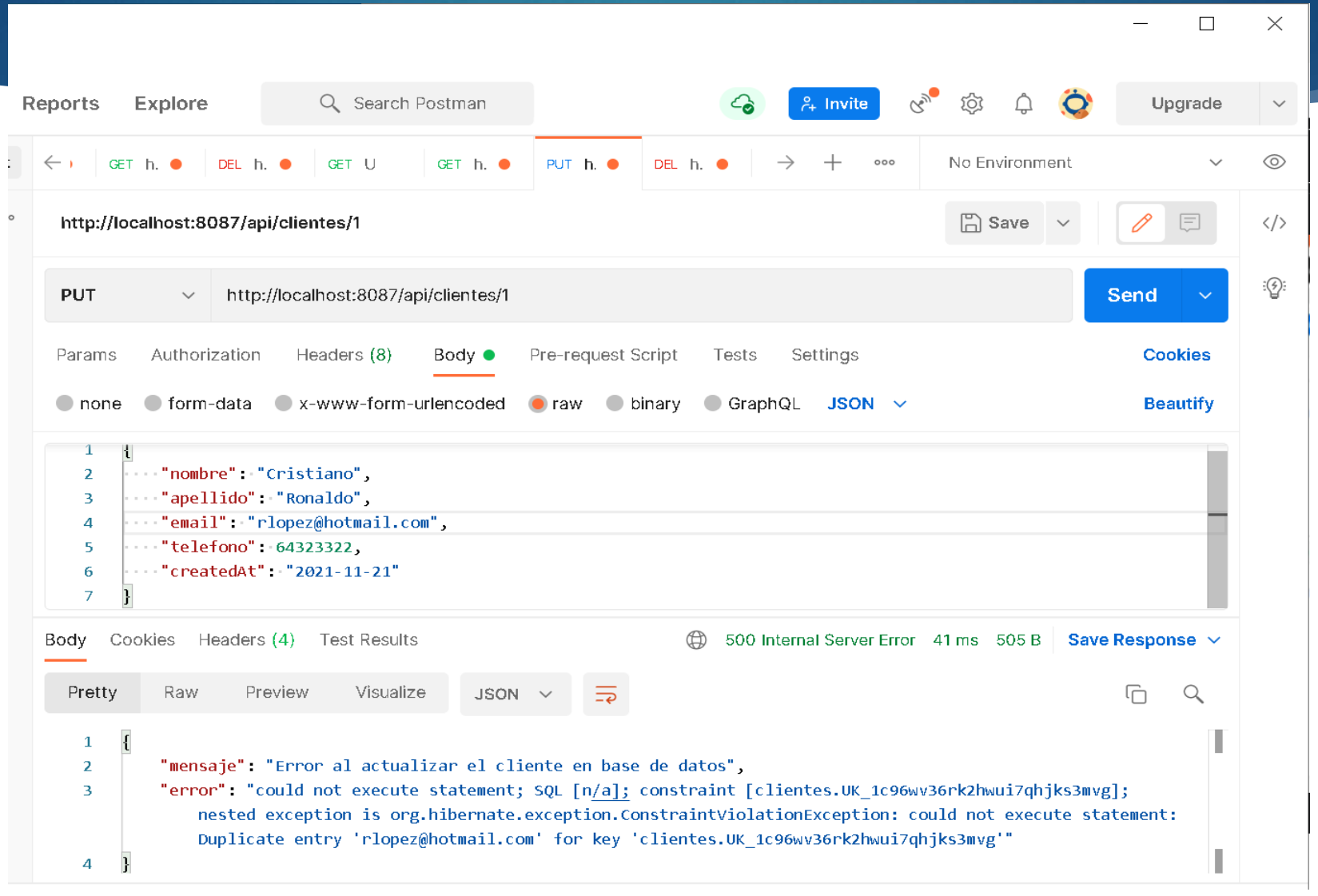
ClienteRestController.java

```
97      }*/
98
99      @PutMapping("/clientes/{id}")
100     public ResponseEntity<?> update(@RequestBody Cliente cliente, @PathVariable Long id) {
101         Cliente clienteActual= clienteService.findById(id);
102
103         Cliente clienteUpdated = null;
104         Map<String,Object> response = new HashMap<>();
105
106         if(clienteActual == null){
107             response.put("mensaje", "Error: no se pudo editar, el cliente ID: ".concat(id.toString().concat("no existe el id en la base de datos")));
108             return new ResponseEntity<Map<String,Object>>(response,HttpStatus.NOT_FOUND);
109         }
110
111         try {
112             clienteActual.setApellido(cliente.getApellido());
113             clienteActual.setNombre(cliente.getNombre());
114             clienteActual.setEmail(cliente.getEmail());
115             clienteActual.setCreatedAt(cliente.getCreatedAt());
116
117             clienteUpdated = clienteService.save(clienteActual);
118         } catch (DataAccessException e) {
119             response.put("mensaje","Error al actualizar el cliente en base de datos");
120             response.put("error", e.getMessage().concat(": ").concat(e.getMostSpecificCause().getMessage()));
121             return new ResponseEntity<Map<String,Object>>(response,HttpStatus.INTERNAL_SERVER_ERROR);
122         }
123
124         response.put("mensaje","El cliente ha sido actualizado con éxito!");
125         response.put("cliente", clienteUpdated);
126
127         return new ResponseEntity<Map<String,Object>>(response, HttpStatus.CREATED);
128     }
```

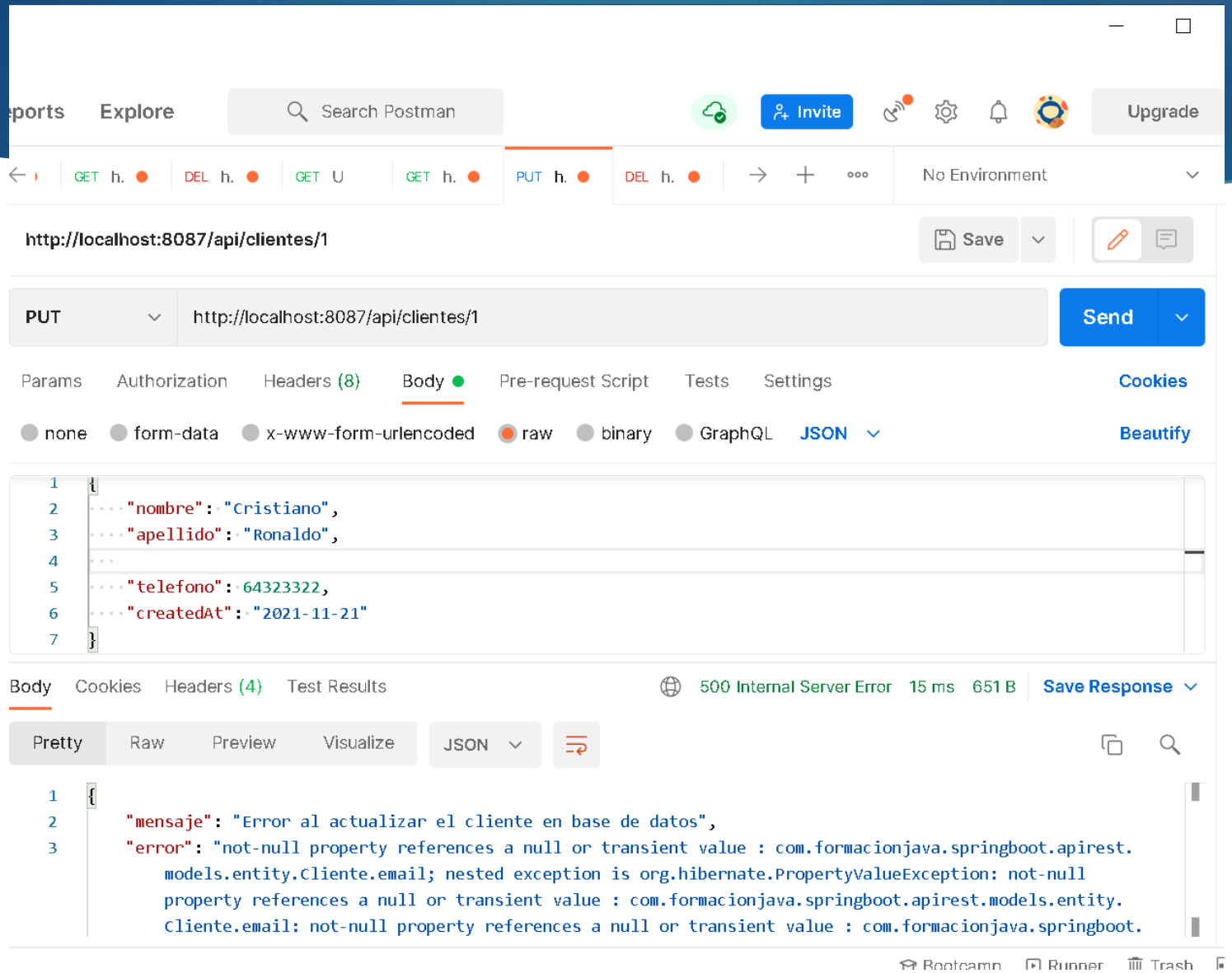
Probamos con postman



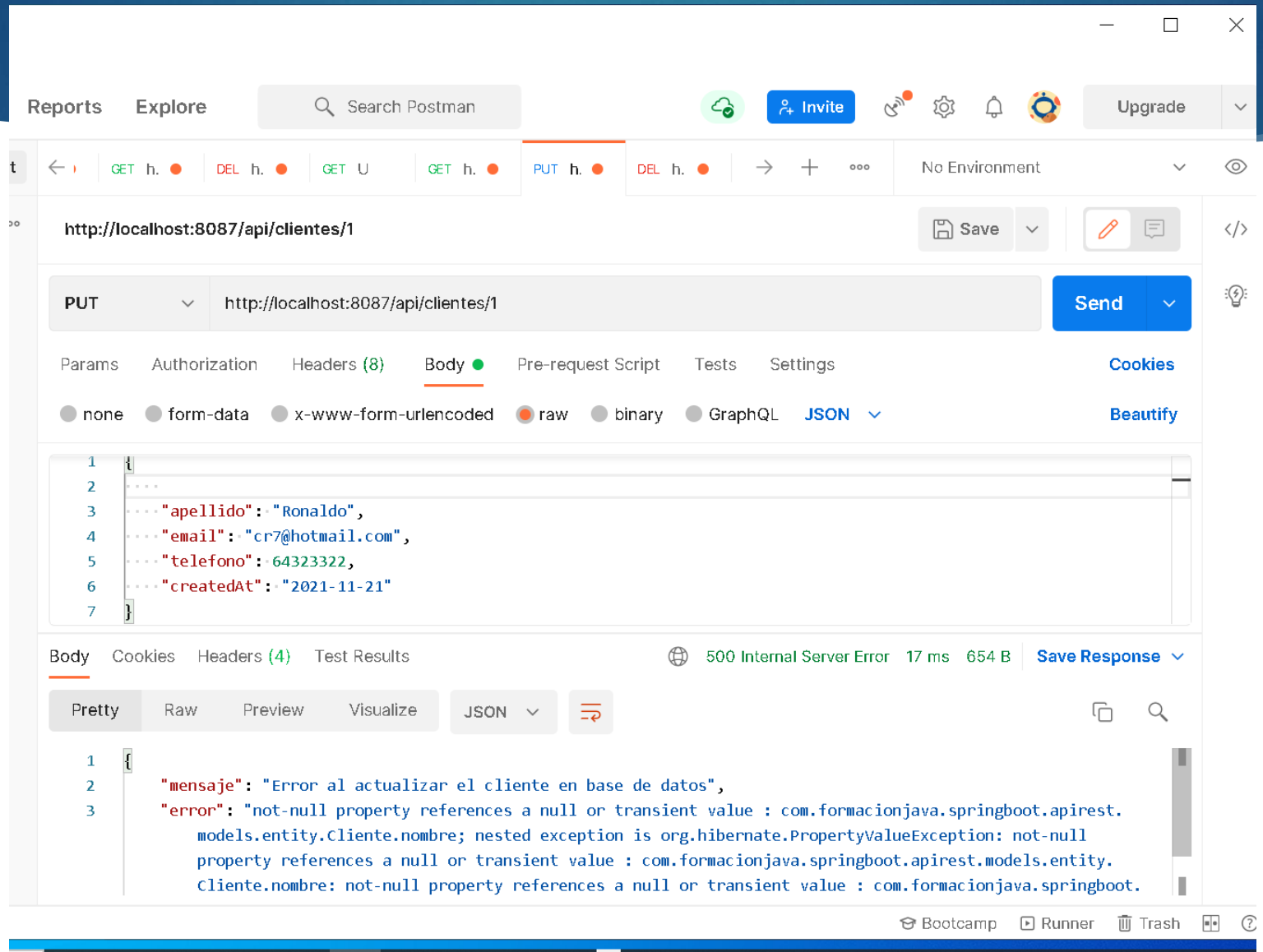
No me deja actualizar un email ya registrado



No me deja actualizar sin el campo email



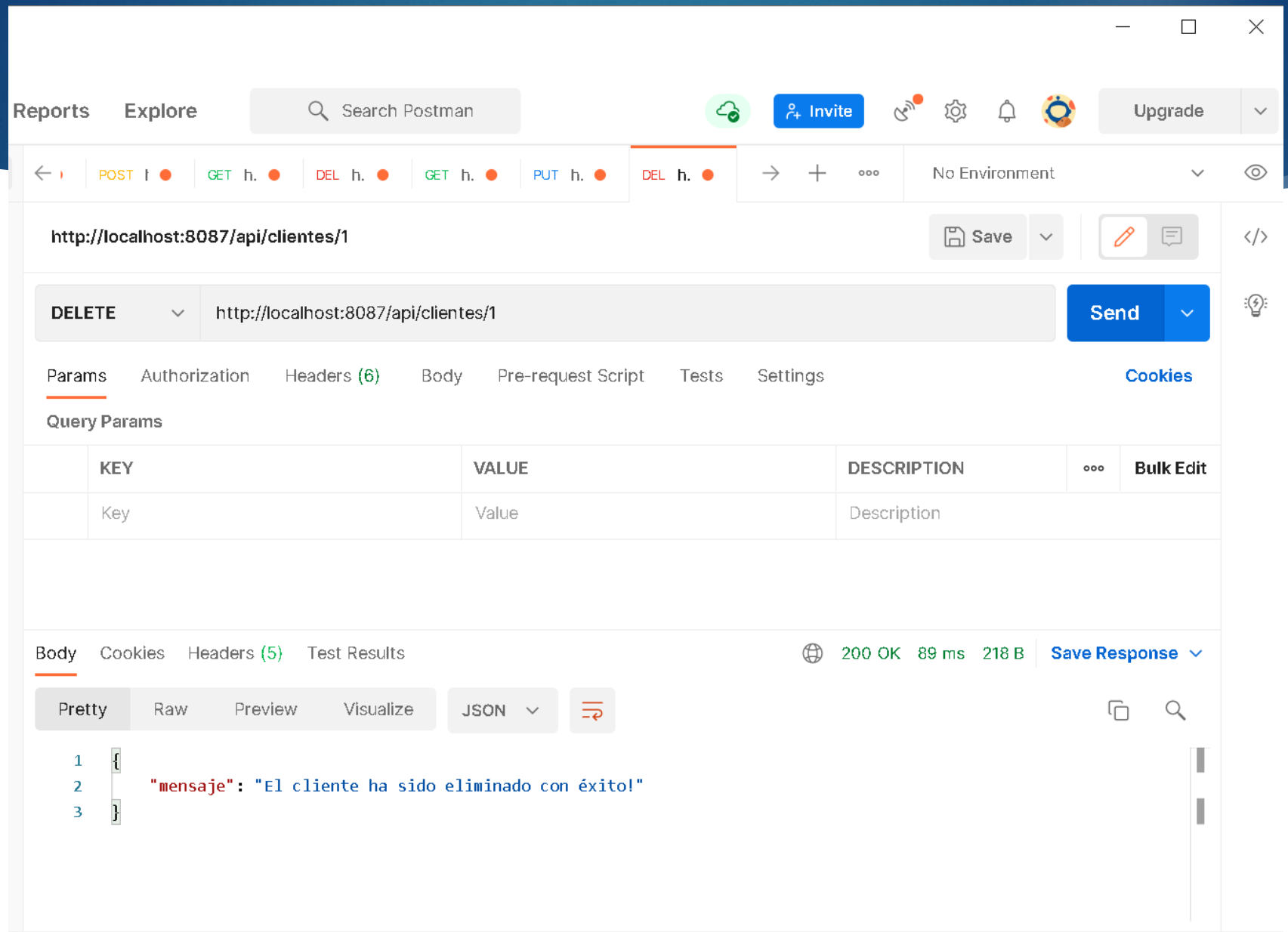
No me deja actualizar sin el campo nombre



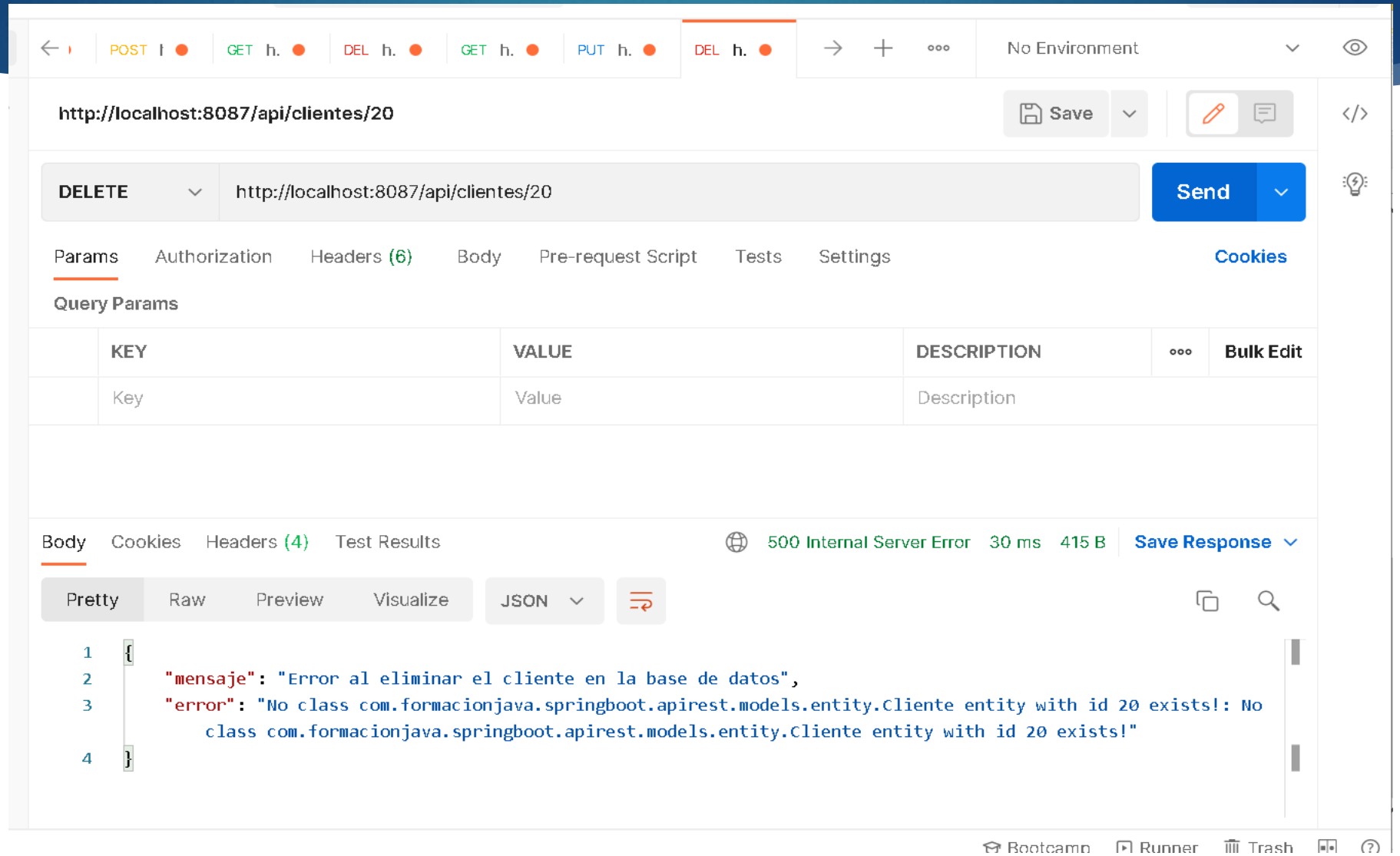
Manejo de error con Delete

*ClienteRestController.java

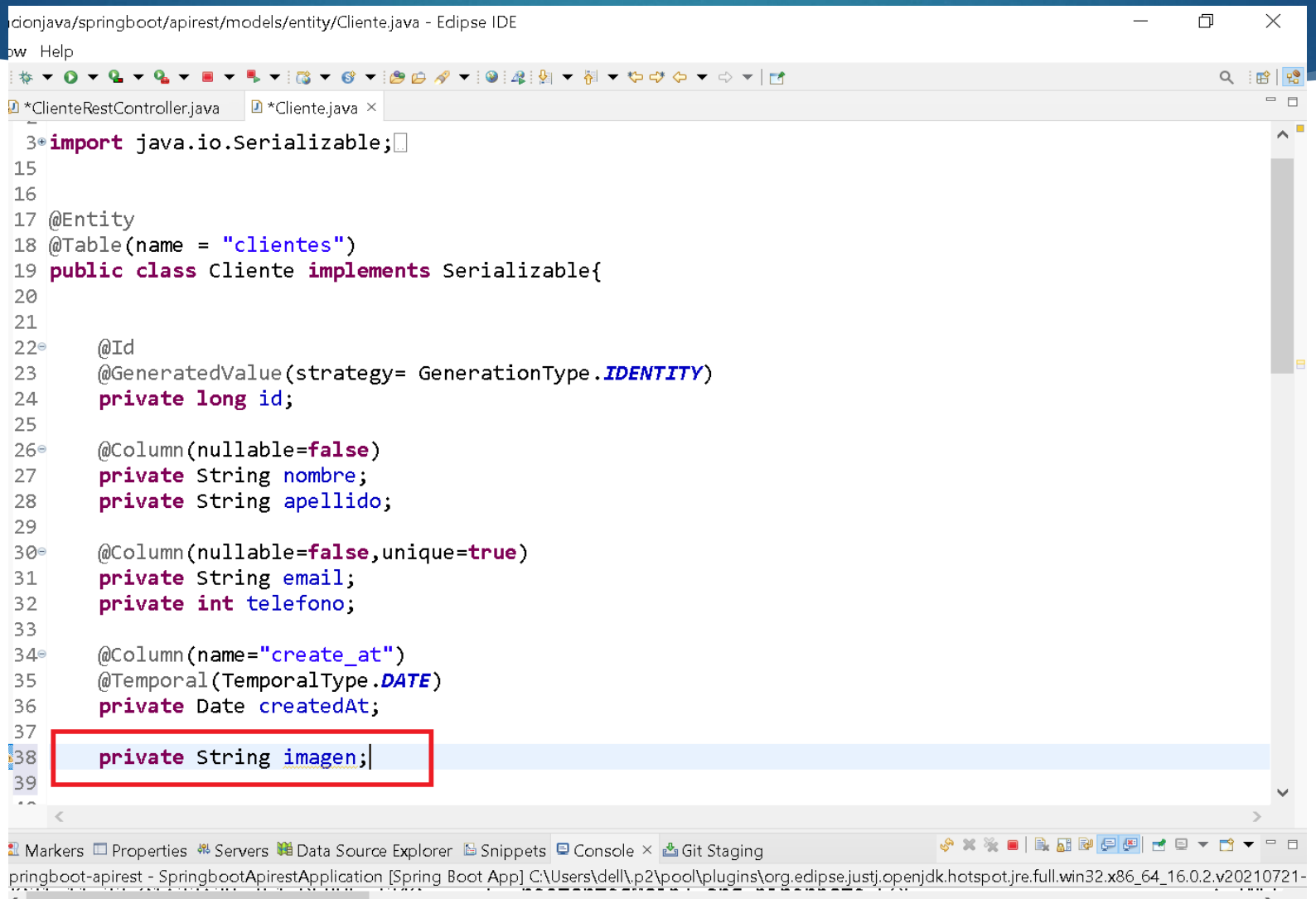
```
126
127         return new ResponseEntity<Map<String,Object>>(response, HttpStatus.CREATED);
128     }
129     /*@DeleteMapping("clientes/{id}")
130     public void delete(@PathVariable Long id) {
131         clienteService.delete(id);
132     }*/
133
134     @DeleteMapping("clientes/{id}")
135     public ResponseEntity<?> delete(@PathVariable Long id) {
136         Map<String,Object> response = new HashMap<>();
137
138         try {
139
140             clienteService.delete(id);
141         } catch (DataAccessException e) {
142             response.put("mensaje", "Error al eliminar el cliente en la base de datos");
143             response.put("error", e.getMessage().concat(": ").concat(e.getMostSpecificCause().getMessage()));
144             return new ResponseEntity<Map<String,Object>>(response,HttpStatus.INTERNAL_SERVER_ERROR);
145         }
146         response.put("mensaje","El cliente ha sido eliminado con éxito!");
147
148
149         return new ResponseEntity<Map<String,Object>>(response, HttpStatus.OK);
150     }
151 }
152
```

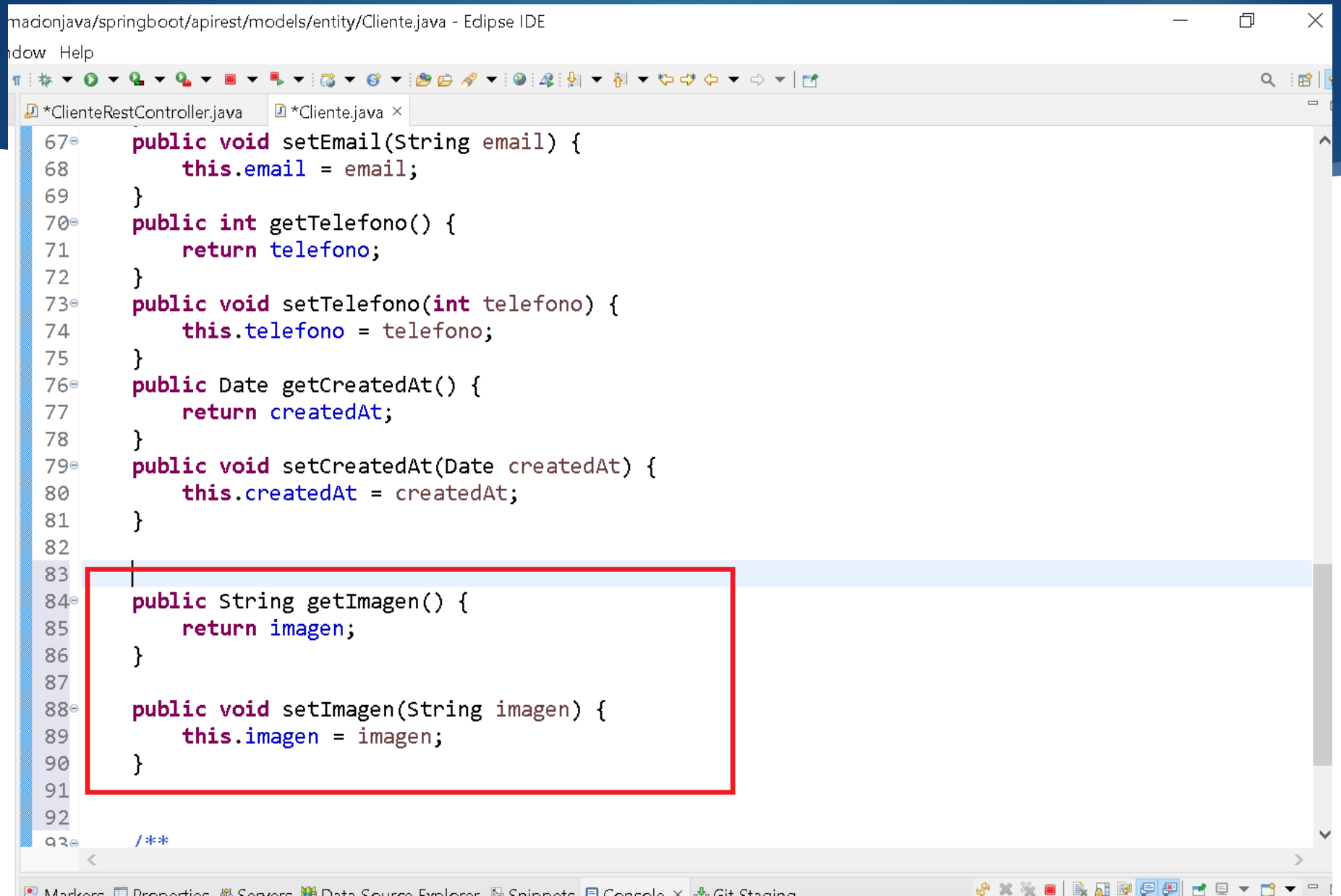

Reports
Explore
Search Postman
Invite
Upgrade
POST
GET h.
DEL h.
GET h.
PUT h.
DEL h.
No Environment
http://localhost:8087/api/clientes/1
Save
DELETE
http://localhost:8087/api/clientes/1
Send
Params
Authorization
Headers (6)
Body
Pre-request Script
Tests
Settings
Cookies
Query Params
KEY
VALUE
DESCRIPTION
Bulk Edit
Key
Value
Description
Body
Cookies
Headers (5)
Test Results
200 OK
89 ms
218 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
1 {
2 "mensaje": "El cliente ha sido eliminado con éxito!"
3 }

Si no encuentra un id valido

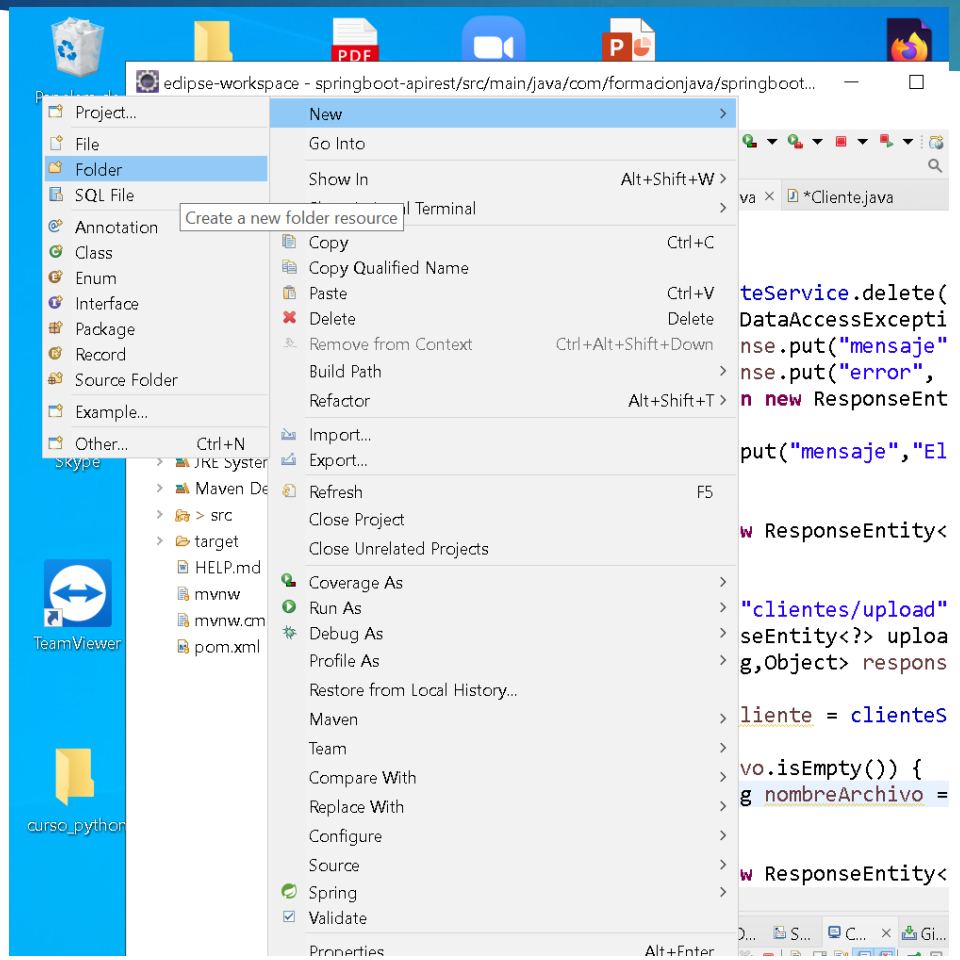


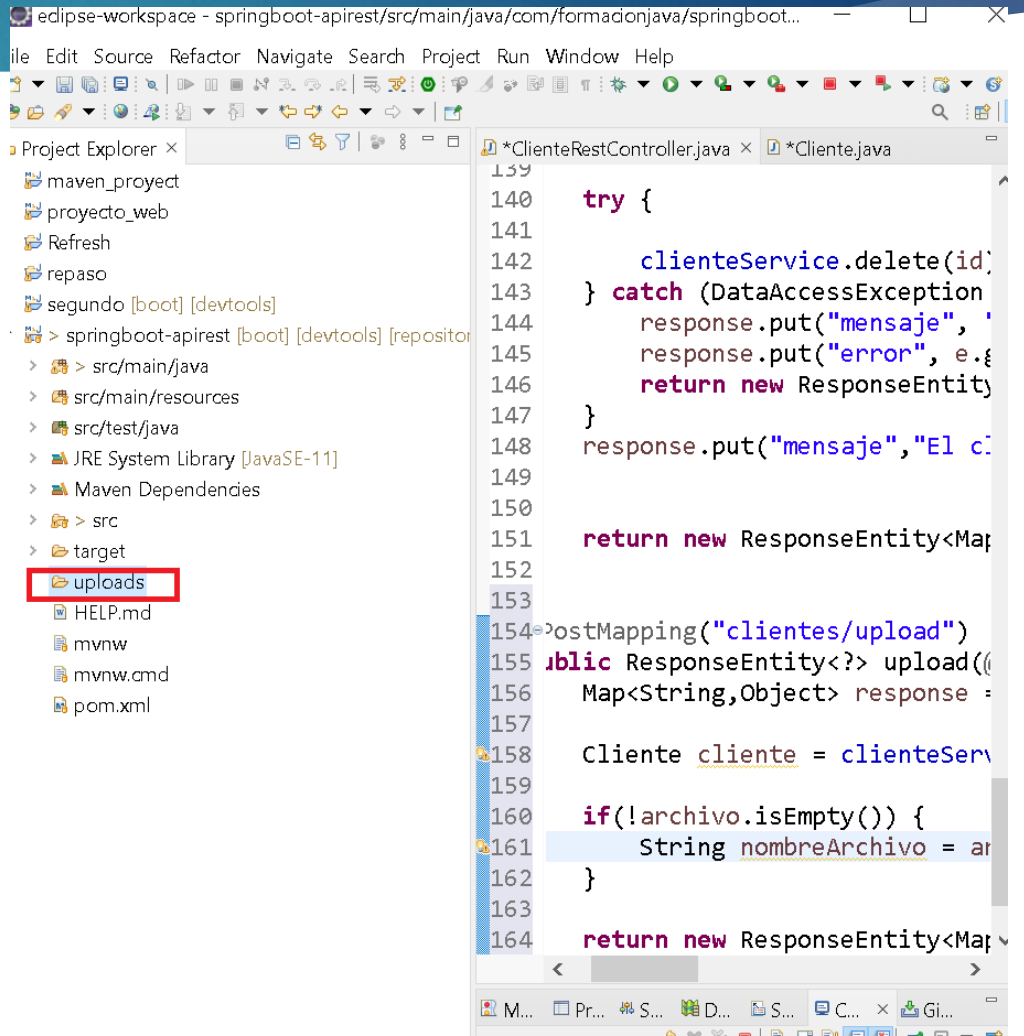
Crear una Api para subida de archivo, agregamos el siguiente campo y sus respectivos setter y getter




macionjava/springboot/apirest/models/entity/Cliente.java - Eclipse IDE
ndow Help
*ClienteRestController.java
*Cliente.java
67 public void setEmail(String email) {
68 this.email = email;
69 }
70 public int getTelefono() {
71 return telefono;
72 }
73 public void setTelefono(int telefono) {
74 this.telefono = telefono;
75 }
76 public Date getCreatedAt() {
77 return createdAt;
78 }
79 public void setCreatedAt(Date createdAt) {
80 this.createdAt = createdAt;
81 }
82
83
84 public String getImagen() {
85 return imagen;
86 }
87
88 public void setImagen(String imagen) {
89 this.imagen = imagen;
90 }
91
92
93 /**

Creamos una carpeta en el directorio principal del proyecto para alojar las imágenes



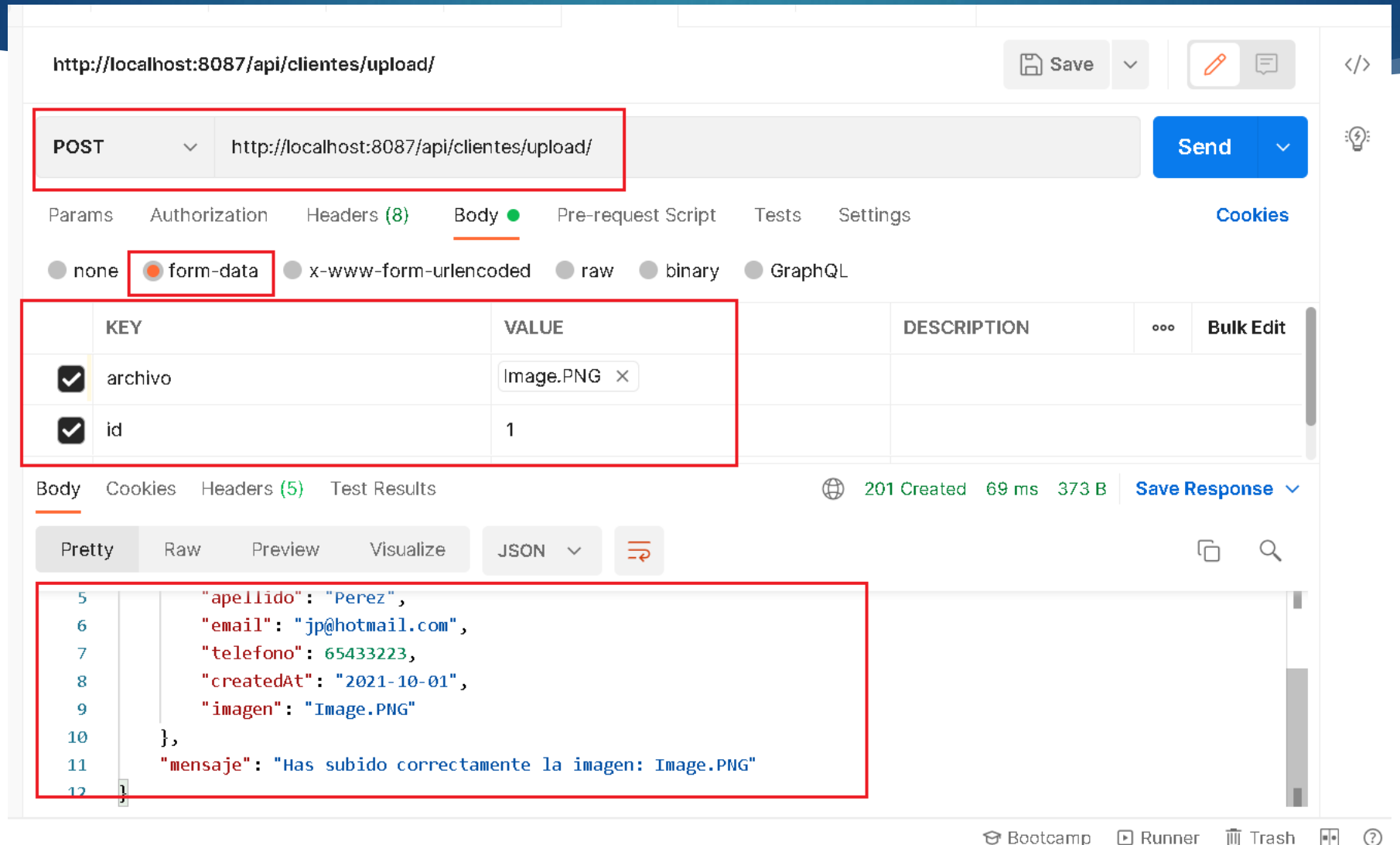


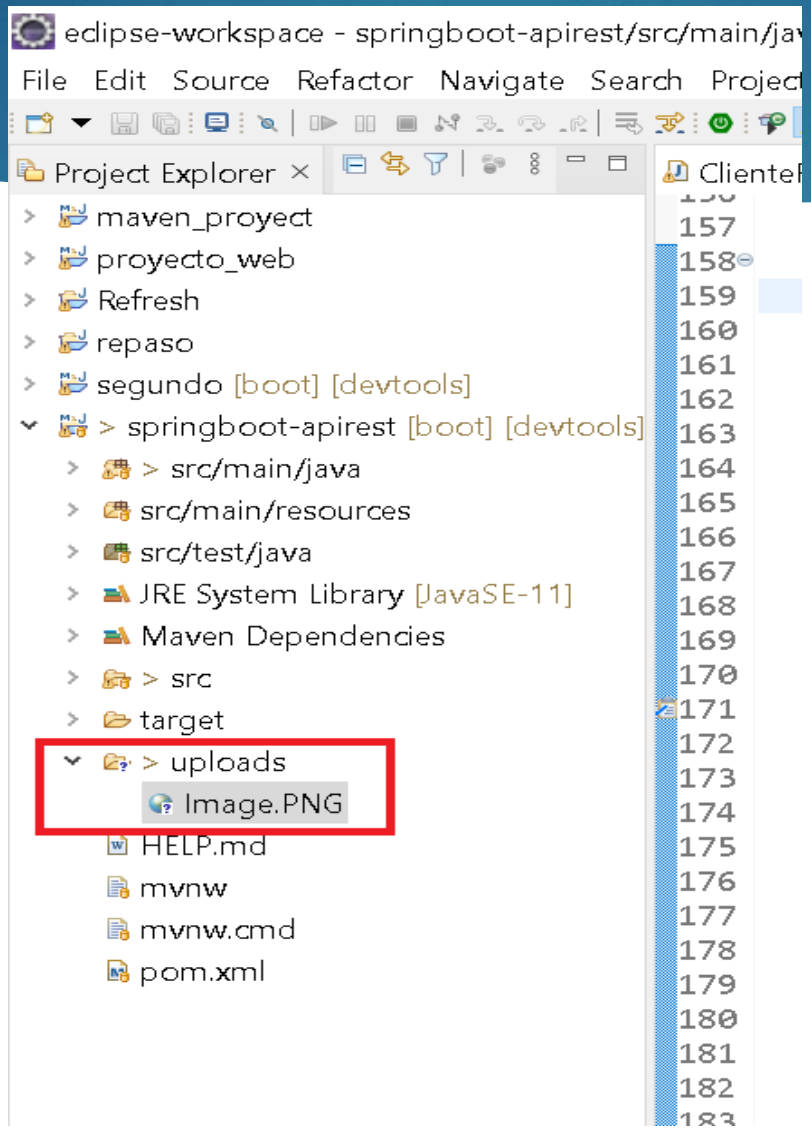
Agregamos un nuevo método al controlador

ClienteRestController.java × Cliente.java

```
157
158     @PostMapping("clientes/upload")
159     public ResponseEntity<?> upload(@RequestParam("archivo") MultipartFile archivo, @RequestParam("id") Long id){
160         Map<String,Object> response = new HashMap<>();
161
162         Cliente cliente = clienteService.findById(id);
163
164         if(!archivo.isEmpty()) {
165             String nombreArchivo = archivo.getOriginalFilename();
166             Path rutaArchivo = Paths.get("uploads").resolve(nombreArchivo).toAbsolutePath();
167
168             try {
169                 Files.copy(archivo.getInputStream(),rutaArchivo);
170             } catch (IOException e) {
171                 // TODO Auto-generated catch block
172                 response.put("mensaje", "Error al subir la imagen del cliente");
173                 response.put("error", e.getMessage().concat(": ").concat(e.getCause().getMessage()));
174                 return new ResponseEntity<Map<String,Object>>(response,HttpStatus.INTERNAL_SERVER_ERROR);
175             }
176
177             cliente.setImagen(nombreArchivo);
178
179             clienteService.save(cliente);
180
181             response.put("cliente",cliente);
182             response.put("mensaje","Has subido correctamente la imagen: "+ nombreArchivo);
183
184         }
185
186         return new ResponseEntity<Map<String,Object>>(response,HttpStatus.CREATED);
187     }
188
```

Probamos la nueva api con postman

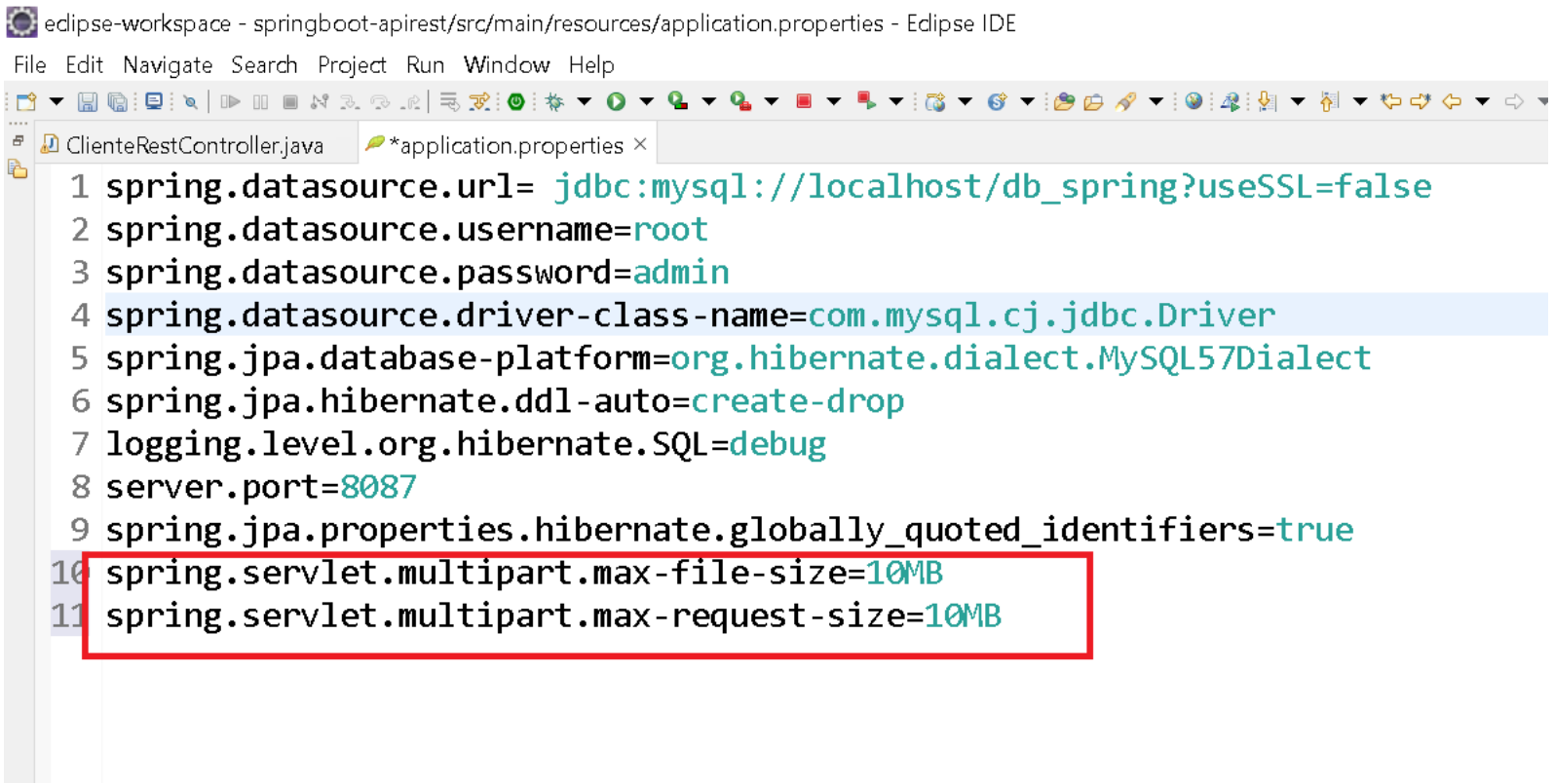



edlipse-workspace - springboot-apirest/src/main/jav
File Edit Source Refactor Navigate Search Project
Project Explorer
maven_proyect
proyecto_web
Refresh
repaso
segundo [boot] [devtools]
> springboot-apirest [boot] [devtools]
> src/main/java
src/main/resources
src/test/java
JRE System Library [JavaSE-11]
Maven Dependencies
> src
target
> uploads
Image.PNG
HELP.md
mvnw
mvnw.cmd
pom.xml
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182

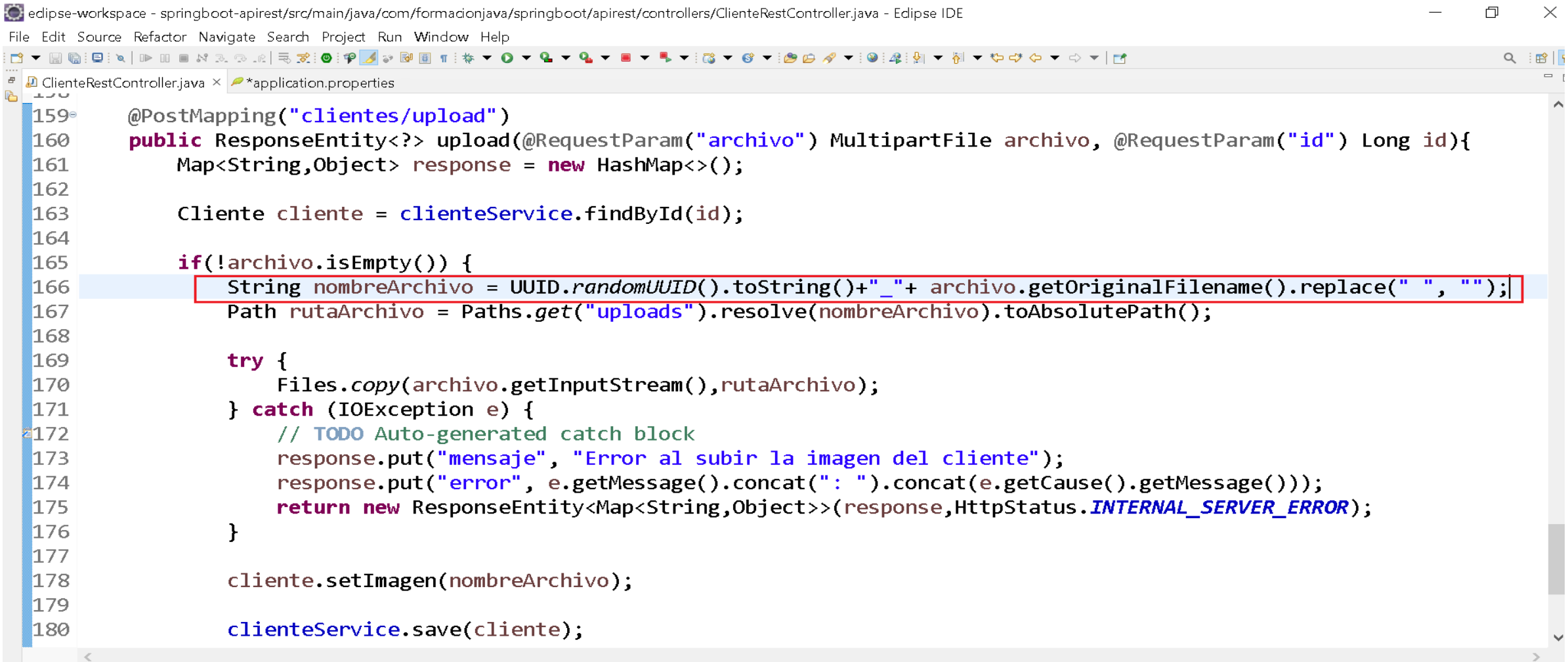
Realizamos mejoras a nuestra api

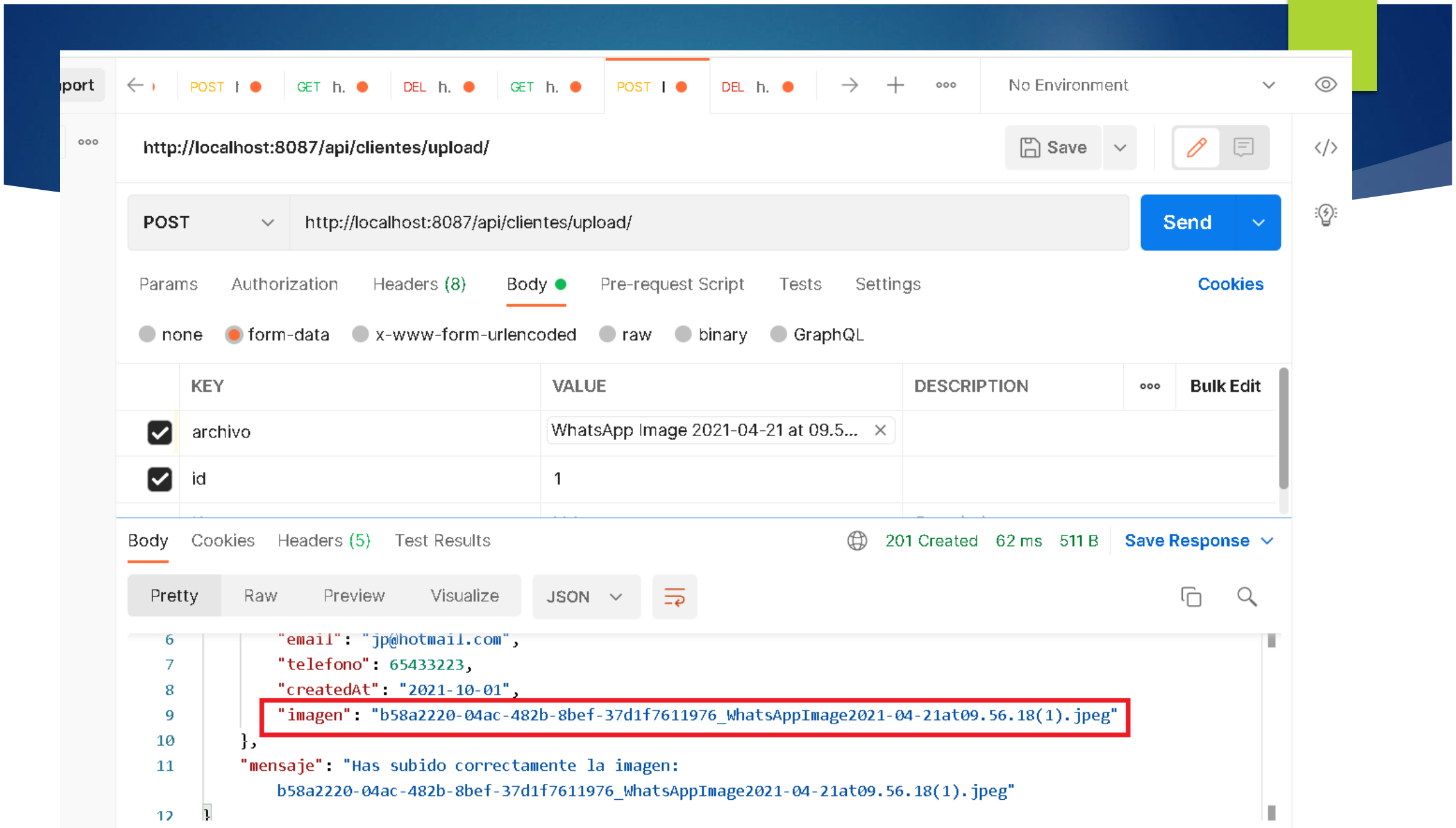
- Aumentar el tamaño permitido del archivo para subir por default esta limitado a 1MB
- Renombrar los archivos al subir para que no entren en conflicto al subir imágenes con el mismo nombre

Configuramos el tamaño permitido para subir archivos en aplication.priorerties



Agregamos la siguiente línea de código para generar nombre de archivos subidos




POST
http://localhost:8087/api/clientes/upload/
Save
POST
http://localhost:8087/api/clientes/upload/
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
archivo
WhatsApp Image 2021-04-21 at 09.5...
id
1
Body
Cookies
Headers (5)
Test Results
201 Created
62 ms
511 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
"email": "jp@hotmail.com",
"telefono": 65433223,
"createdAt": "2021-10-01",
"imagen": "b58a2220-04ac-482b-8bef-37d1f7611976_WhatsAppImage2021-04-21at09.56.18(1).jpeg"
},
"mensaje": "Has subido correctamente la imagen:
b58a2220-04ac-482b-8bef-37d1f7611976_WhatsAppImage2021-04-21at09.56.18(1).jpeg"

Falta una ultima mejora

- Borrar los archivos asociados a un mismo usuario ya no se borra una vez reemplazado.

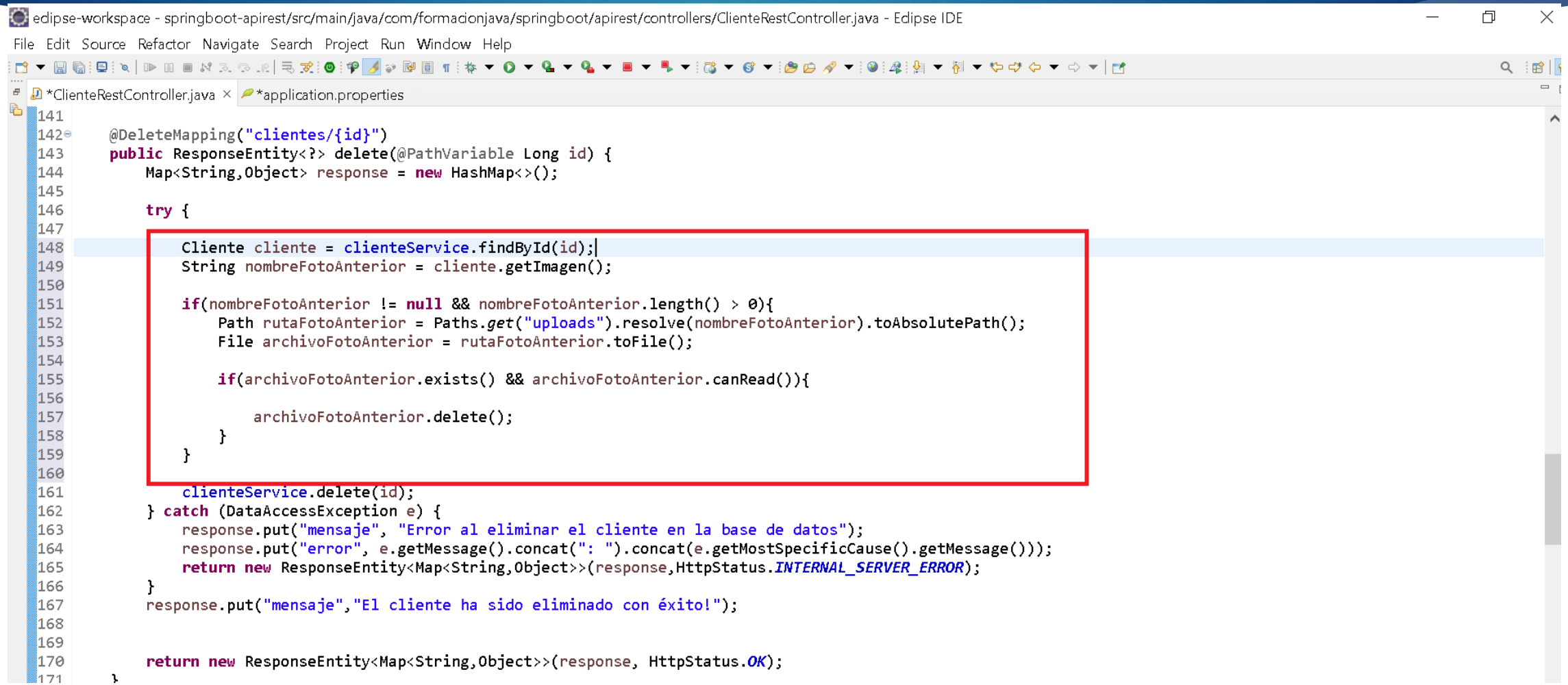
eclipse-workspace - springboot-apirest/src/main/java/com/formacionjava/springboot/apirest/controllers/ClienteRestController.java - Eclipse IDE

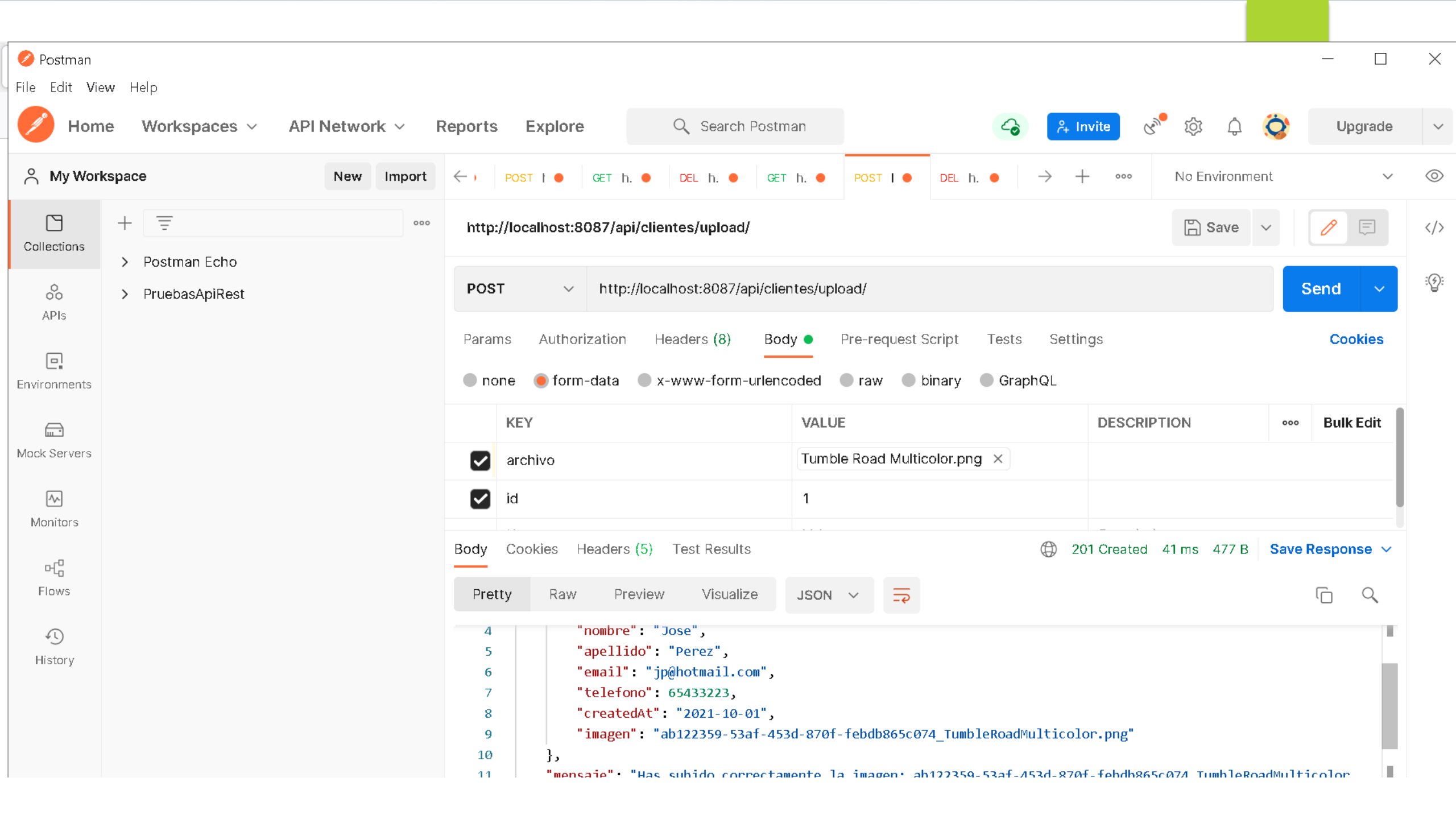
File Edit Source Refactor Navigate Search Project Run Window Help

*ClienteRestController.java × *application.properties

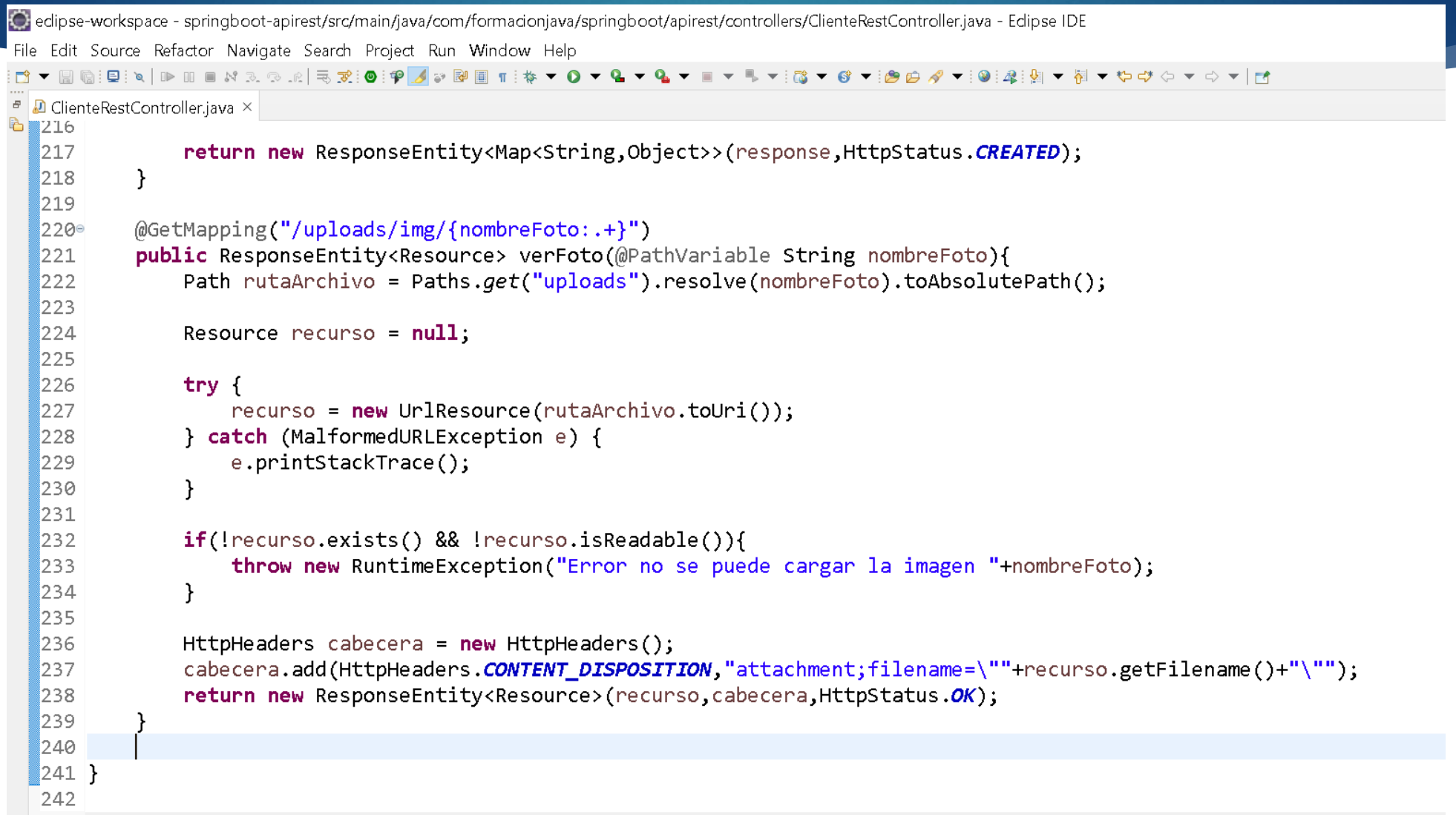
```
160     @PostMapping( "clientes/upload" )
161     public ResponseEntity<?> upload(@RequestParam("archivo") MultipartFile archivo, @RequestParam("id") Long id){
162         Map<String,Object> response = new HashMap<>();
163
164         Cliente cliente = clienteService.findById(id);
165
166         if(!archivo.isEmpty()) {
167             String nombreArchivo = UUID.randomUUID().toString()+"_"+ archivo.getOriginalFilename().replace(" ", "");
168             Path rutaArchivo = Paths.get("uploads").resolve(nombreArchivo).toAbsolutePath();
169
170             try {
171                 Files.copy(archivo.getInputStream(),rutaArchivo);
172             } catch (IOException e) {
173                 // TODO Auto-generated catch block
174                 response.put("mensaje", "Error al subir la imagen del cliente");
175                 response.put("error", e.getMessage().concat(": ").concat(e.getCause().getMessage()));
176                 return new ResponseEntity<Map<String,Object>>(response,HttpStatus.INTERNAL_SERVER_ERROR);
177             }
178
179             String nombreFotoAnterior = cliente.getImagen();
180
181             if(nombreFotoAnterior != null && nombreFotoAnterior.length() > 0){
182                 Path rutaFotoAnterior = Paths.get("uploads").resolve(nombreFotoAnterior).toAbsolutePath();
183                 File archivoFotoAnterior = rutaFotoAnterior.toFile();
184
185                 if(archivoFotoAnterior.exists() && archivoFotoAnterior.canRead()){
186
187                     archivoFotoAnterior.delete();
188                 }
189             }
190
```

Agregamos el mismo código para eliminar imagen a la función delete




Postman
File Edit View Help
Home Workspaces API Network Reports Explore
Search Postman
Invite
Upgrade
My Workspace New Import
POST l GET h. DEL h. GET h. POST l DEL h.
No Environment
Collections
APIs
Environments
Mock Servers
Monitors
Flows
History
Postman Echo
PruebasApiRest
http://localhost:8087/api/clientes/upload/
Save
POST
http://localhost:8087/api/clientes/upload/
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
archivo Tumble Road Multicolor.png
id 1
Body Cookies Headers (5) Test Results
201 Created 41 ms 477 B
Save Response
Pretty Raw Preview Visualize JSON
"nombre": "Jose",
"apellido": "Perez",
"email": "jp@hotmail.com",
"telefono": 65433223,
"createdAt": "2021-10-01",
"imagen": "ab122359-53af-453d-870f-febdb865c074_TumbleRoadMulticolor.png"
},
"mensaje": "Has subido correctamente la imagen: ab122359-53af-453d-870f-febdb865c074_TumbleRoadMulticolor

Agregamos un API mas para mostrar imágenes




```
216
217         return new ResponseEntity<Map<String,Object>>(response,HttpStatus.CREATED);
218     }
219
220     @GetMapping("/uploads/img/{nombreFoto:.+}")
221     public ResponseEntity<Resource> verFoto(@PathVariable String nombreFoto){
222         Path rutaArchivo = Paths.get("uploads").resolve(nombreFoto).toAbsolutePath();
223
224         Resource recurso = null;
225
226         try {
227             recurso = new UrlResource(rutaArchivo.toUri());
228         } catch (MalformedURLException e) {
229             e.printStackTrace();
230         }
231
232         if(!recurso.exists() && !recurso.isReadable()){
233             throw new RuntimeException("Error no se puede cargar la imagen "+nombreFoto);
234         }
235
236         HttpHeaders cabecera = new HttpHeaders();
237         cabecera.add(HttpHeaders.CONTENT_DISPOSITION,"attachment;filename=\""+recurso.getFilename()+"\"");
238         return new ResponseEntity<Resource>(recurso,cabecera,HttpStatus.OK);
239     }
240
241 }
242
```

Postman

File Edit View Help

Home Workspaces API Network Reports Explore Search Postman Invite Upgrade

ST GET h. POST POST POST GET h. DEL h. GET h. POST DEL h.. PUT h. No Environment

http://localhost:8087/api/clientes/upload Save

POST http://localhost:8087/api/clientes/upload Send

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

none form-data x-www-form-urlencoded raw binary GraphQL

archivo	Image (1).PNG	
id	1	
Key	Value	Description

Body Cookies Headers (5) Test Results Status: 201 Created Time: 104 ms Size: 453 B Save Response

Pretty Raw Preview Visualize JSON


```
        "apellido": "Perez",
        "email": "jp@hotmail.com",
        "telefono": 65433223,
        "createdAt": "2021-10-01",
        "imagen": "35c9d9f2-d3de-46ff-92a6-8bee3aa9ba17_Image(1).PNG"
    },
    "mensaje": "Has subido correctamente la imagen: 35c9d9f2-d3de-46ff-92a6-8bee3aa9ba17_Image(1).PNG"
}
```

http://localhost:8087/api/uploads/img/35c9d9f2-d3de-46ff-92a6-8bee3aa9ba17_Image(1).PNG