

PROYECT

BASIC:

Dessert

- Entities
- Dtos
- Repositories - **@Repository** – Query - Buscar
- ServiceInterfaces – Insertar – Listar - Buscar
- ServiceImplements - **@Service**
- Controllers - **@RestController**

Ingredient

- Entities
- Dtos
- Repositories
- Serviceinterfaces
- Serviceimplements
- Controllers

SECURITY

Crear Users and Role

- Entities (Users and Role)
- Repositories (User- findby - JwtUserDetailsService)

Crear JwtUserDetailsService

- ServiceImplements - **@Service**

Crear el paquete Security

1. JwtTokenUtil - **@Component**
2. JwtAuthenticationEntryPoint - **@Component** - implements AuthenticationEntryPoint
3. JwtRequest - Copiar
4. JwtRequestFilter – **@Component** - extends OncePerRequestFilter

OBSERVACIÓN: Depende de la clase **JwtUserDetailService** y **JwtTokenUtil**

5. JwtResponse - Copiar
6. WebSecurityConfig –**OBSERVACIÓN:** Depende de la clase **JwtAuthenticationEntryPoint**

No tiene extends

@Configuration

@EnableWebSecurity

@EnableGlobalMethodSecurity(prePostEnabled = true)

Crear JwtAuthenticationController

- **Controllers**

@RestController

@CrossOrigin

DEPENDENCIAS

BASIC:

```
<!-- https://mvnrepository.com/artifact/org.springdoc/springdoc-openapi-ui -->
<dependency>
  <groupId>org.springdoc</groupId>
  <artifactId>springdoc-openapi-ui</artifactId>
  <version>1.7.0</version>
</dependency>
<!-- https://mvnrepository.com/artifact/org.modelmapper/modelmapper -->
<dependency>
  <groupId>org.modelmapper</groupId>
  <artifactId>modelmapper</artifactId>
  <version>3.1.1</version>
</dependency>
```

SECURITY:

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-security</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.security</groupId>
  <artifactId>spring-security-test</artifactId>
  <scope>test</scope>
</dependency>
<dependency>
  <groupId>io.jsonwebtoken</groupId>
  <artifactId>jjwt</artifactId>
  <version>0.9.1</version>
</dependency>
```

CONTRASEÑA

- user
- 976bcd5f-f93a-4ec2-8173-af342656bbbe

```
spring.jpa.database=POSTGRESQL
spring.jpa.show-sql=false
spring.jpa.hibernate.ddl-auto=update
spring.datasource.driver-class-name=org.postgresql.Driver
spring.datasource.url=jdbc:postgresql://localhost/db_ejercicio1
spring.datasource.username=postgres
spring.datasource.password=130902
server.port=8080

spring.mvc.pathmatch.matching-strategy = ANT_PATH_MATCHER
jwt.secret=4t7w!z%C*F-JaNdRgUkXn2r5u8x/A?D(G+KbPeShVmYq3s6v9y$B&E)H@McQfTjW
spring.devtools.restart.log-condition-evaluation-delta=false
```

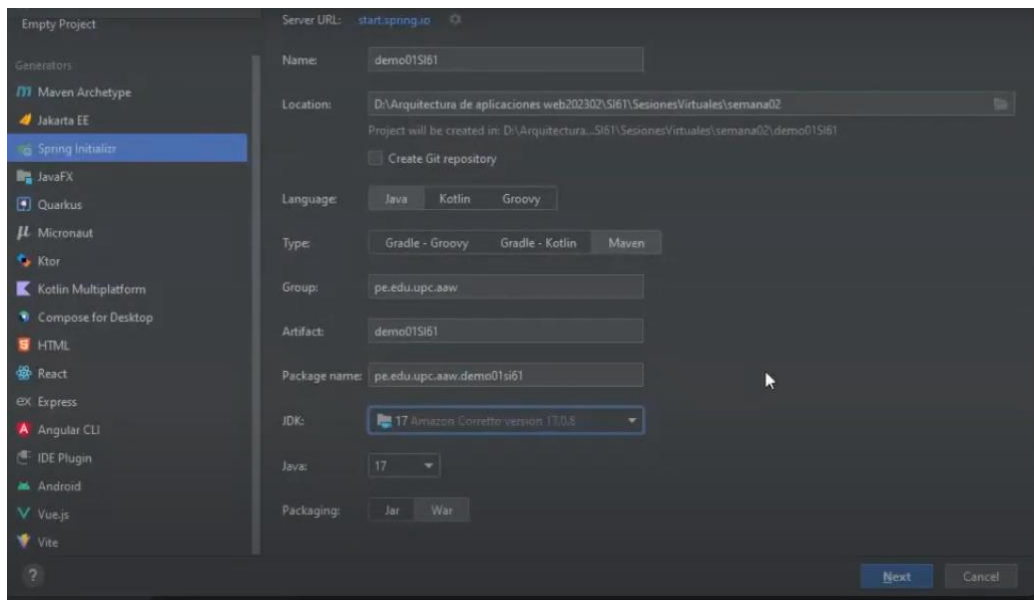
POST: localhost:8080/authenticate

```
{
  "jmtrUsername": "usuario",
  "jmtrPassword": "web"
}
```

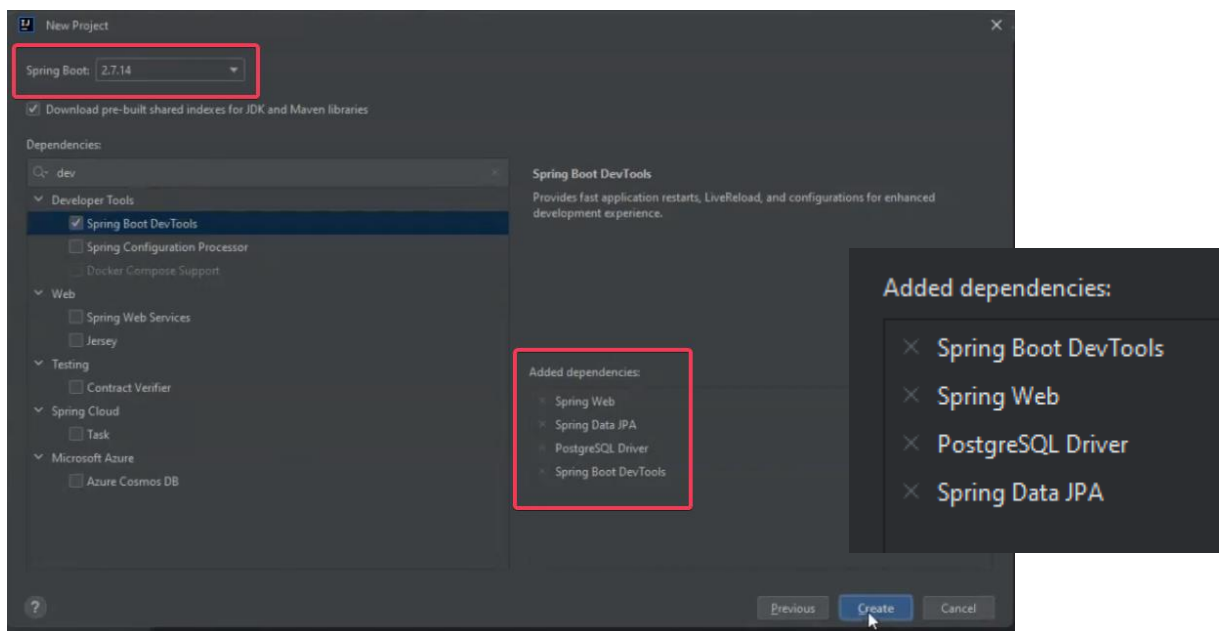
localhost:8080/api2/software2/aaa

Headers: Authorization

Bearer



pe.edu.upc.aaw



para obtener mas informacion.

```
//Para crear una variable de tipo Long que se autoincrementa, puedes usar la anotación
@GeneratedValue con la estrategia AUTO
@Id
@GeneratedValue(strategy = GenerationType.AUTO)
private Long id;

//Para crear una variable de tipo String que sea única, puedes usar la anotación @Column con
el atributo unique
@Column(name = "code", unique = true)
private String code;
```

```
1 {
2   ... "jmtUsername": "administrador",
3   ... "jmtPassword": "web"
4 }
```

1. Como asistente de cocina quiero listar los ingredientes por categoría de postre para gestionarlos. **Listar INGREDIENTES por categoría de POSTRE**

QUERY

```
SELECT ing.jmtr_id_ingredient, ing.jmtr_name_ingredient, ing.jmtr_quantity_ingredient,  
ing.jmtr_id_dessert
```

```
FROM ingredient ing
```

```
INNER JOIN dessert des ON ing.jmtr_id_dessert = des.jmtr_id_dessert
```

```
WHERE des.jmtr_category_dessert = 'dulce'
```

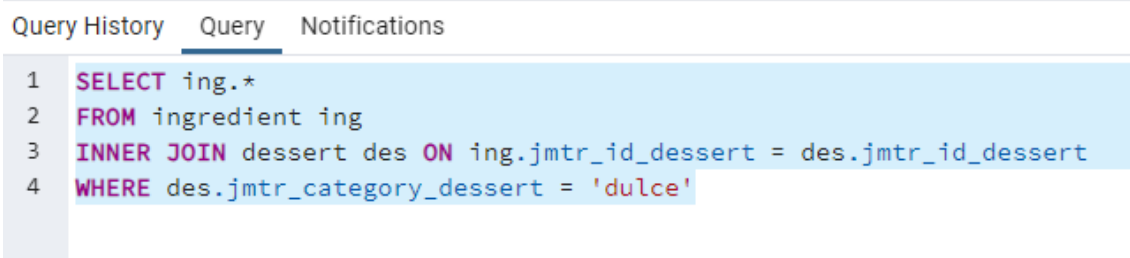
QUERY2

```
SELECT ing.*
```

```
FROM ingredient ing
```

```
INNER JOIN dessert des ON ing.jmtr_id_dessert = des.jmtr_id_dessert
```

```
WHERE des.jmtr_category_dessert = 'dulce'
```



The screenshot shows a SQL query editor with three tabs: 'Query History', 'Query', and 'Notifications'. The 'Query' tab is active. The query text is as follows:

```
1 SELECT ing.*  
2 FROM ingredient ing  
3 INNER JOIN dessert des ON ing.jmtr_id_dessert = des.jmtr_id_dessert  
4 WHERE des.jmtr_category_dessert = 'dulce'
```

2. Como cocinero quiero mostrar la cantidad de ingredientes por dificultad del postre.

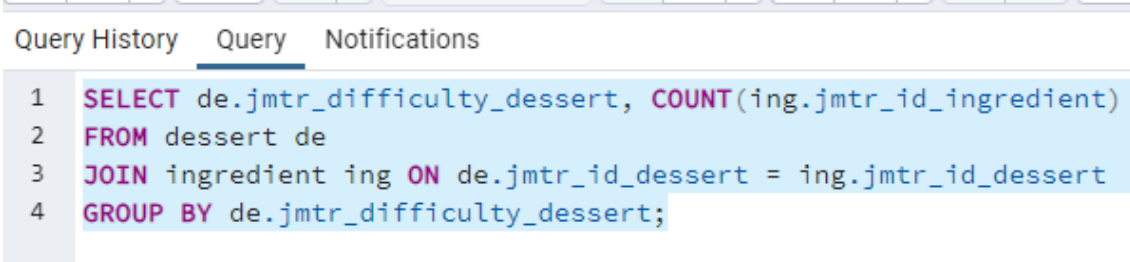
QUERY:

```
SELECT de.jmtr_difficulty_dessert, COUNT(ing.jmtr_id_ingredient)
```

```
FROM dessert de
```

```
JOIN ingredient ing ON de.jmtr_id_dessert = ing.jmtr_id_dessert
```

```
GROUP BY de.jmtr_difficulty_dessert
```



The screenshot shows a SQL query editor with three tabs: 'Query History', 'Query', and 'Notifications'. The 'Query' tab is active. The query text is as follows:

```
1 SELECT de.jmtr_difficulty_dessert, COUNT(ing.jmtr_id_ingredient)  
2 FROM dessert de  
3 JOIN ingredient ing ON de.jmtr_id_dessert = ing.jmtr_id_dessert  
4 GROUP BY de.jmtr_difficulty_dessert;
```

3. Como administrador quiero listar los ingredientes dado el id de postre para gestionarlos

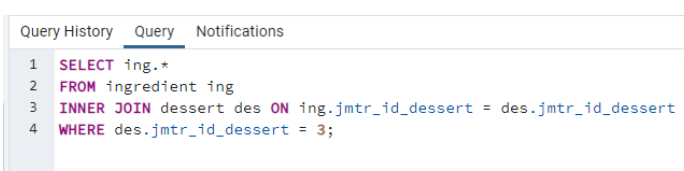
QUERY:

```
SELECT ing.*
```

```
FROM ingredient ing
```

```
INNER JOIN dessert des ON ing.jmtr_id_dessert = des.jmtr_id_dessert
```

```
WHERE des.jmtr_id_dessert = 3
```



The screenshot shows a SQL query editor with three tabs: 'Query History', 'Query', and 'Notifications'. The 'Query' tab is active. The query text is as follows:

```
1 SELECT ing.*  
2 FROM ingredient ing  
3 INNER JOIN dessert des ON ing.jmtr_id_dessert = des.jmtr_id_dessert  
4 WHERE des.jmtr_id_dessert = 3;
```

MEMBER:

Como usuario quiero obtener la cantidad miembros del Club cuyos pagos están en estatus false en un mes dado para gestionarlos.

```
SELECT me.jmtr_name as club, COUNT(me.jmtr_id_member) AS quantityMembers  
FROM member me  
JOIN payment p ON me.jmtr_id_member = p.jmtr_id_member  
WHERE p.jmtr_status = false AND p.jmtr_month = 'SEPTEMBER'  
GROUP BY me.jmtr_name
```

```
1 SELECT me.jmtr_name as club, COUNT(me.jmtr_id_member) AS quantityMembers  
2 FROM member me  
3 JOIN payment p ON me.jmtr_id_member = p.jmtr_id_member  
4 WHERE p.jmtr_status = false AND p.jmtr_month = 'SEPTEMBER'  
5 GROUP BY me.jmtr_name
```

MEMBER2:

Como usuario quiero listar los pagos dado un código de un integrante ordenados ascendentemente para gestionarlos.

```
SELECT pa.*  
FROM Member me  
JOIN Payment pa ON me.jmtr_id_member = pa.jmtr_id_member  
WHERE me.jmtr_memeber_code = 2  
ORDER BY pa.jmtr_month
```

Query History	Query	Notifications
1	SELECT pa.*	
2	FROM Member me	
3	JOIN Payment pa ON me.jmtr_id_member = pa.jmtr_id_member	
4	WHERE me.jmtr_memeber_code = 2	
5	ORDER BY pa.jmtr_month	

UNIVERSIDAD:

Como usuario quiero listar estudiantes con promedio ponderado mayor o igual a una calificación dada ordenados alfabéticamente por nombre para gestionarlos.

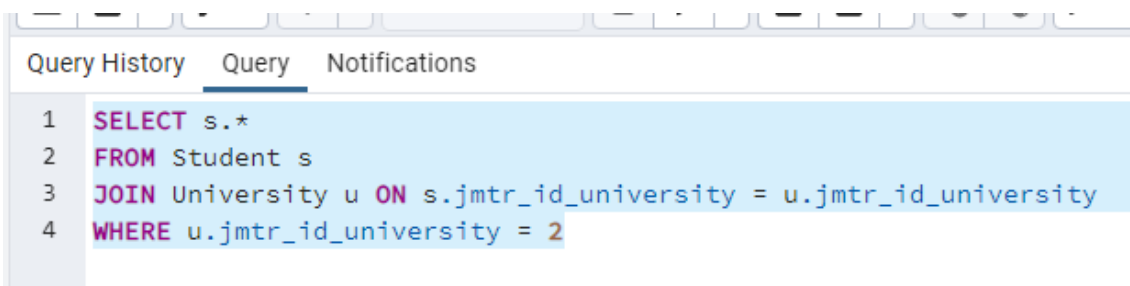
```
SELECT s.jmtr_name, s.jmtr_weighted_average  
FROM Student s  
JOIN University u ON s.jmtr_id_university = u.jmtr_id_university  
WHERE s.jmtr_weighted_average >= 15  
ORDER BY s.jmtr_name asc;
```

```
1 SELECT s.jmtr_name, s.jmtr_weighted_average  
2 FROM Student s  
3 JOIN University u ON s.jmtr_id_university = u.jmtr_id_university  
4 WHERE s.jmtr_weighted_average >= 15  
5 ORDER BY s.jmtr_name asc;
```

UNIVERSIDAD2:

Como usuario quiero listar los estudiantes de una universidad dado el id de Universidad para gestionarlos.

```
SELECT s.*  
FROM Student s  
JOIN University u ON s.jmtr_id_university = u.jmtr_id_university  
WHERE u.jmtr_id_university = 2
```



ENTERPRISE:

Como usuario quiero listar software dado el nombre de la empresa ordenados
ascendentemente por nombre para gestionarlos

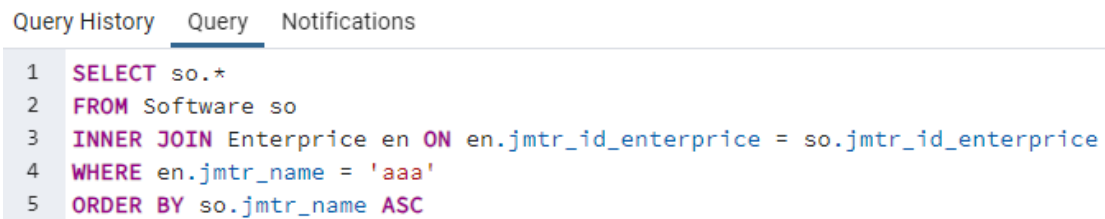
SELECT so.*

FROM Software so

INNER JOIN Enterprise en ON en.jmtr_id_enterprice = so.jmtr_id_enterprice

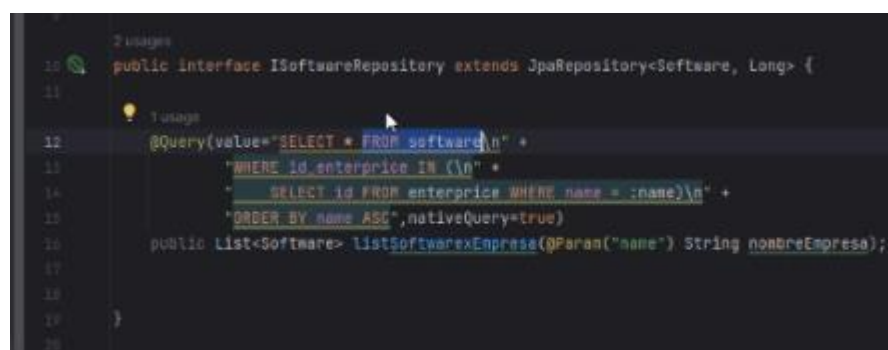
WHERE en.jmtr_name = 'aaa'

ORDER BY so.jmtr_name ASC



Query History Query Notifications

```
1 SELECT so.*
2 FROM Software so
3 INNER JOIN Enterprise en ON en.jmtr_id_enterprice = so.jmtr_id_enterprice
4 WHERE en.jmtr_name = 'aaa'
5 ORDER BY so.jmtr_name ASC
```



```
2 usages
10 public interface ISoftwareRepository extends JpaRepository<Software, Long> {
11
12     1 usage
13     @Query(value="SELECT * FROM software\n" +
14             "WHERE id_enterprice IN (\n" +
15             "    SELECT id FROM enterprise WHERE name = :name)\n" +
16             "ORDER BY name ASC",nativeQuery=true)
17     public List<Software> listSoftwareExEmpresa(@Param("name") String nombreEmpresa);
18
19 }
20
```