



Nombre: Darwin Cabrera

Carrera: Computación

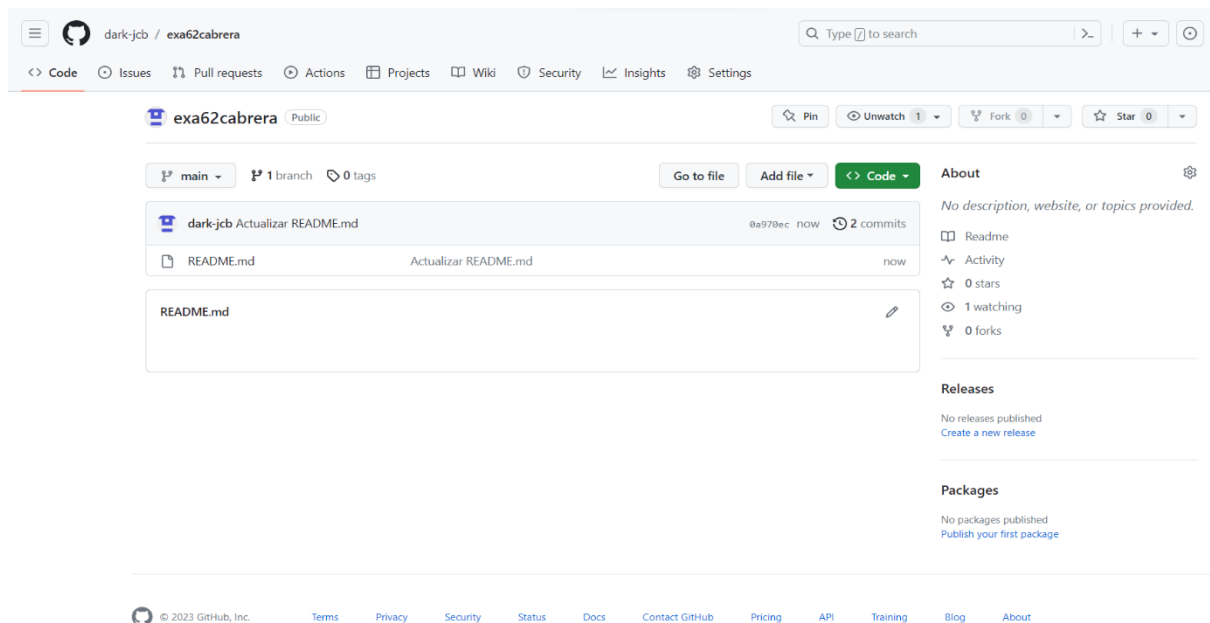
Asignatura: Programación y Plataformas Web

Examen de Desarrollo de Servicios Web REST en Jakarta

Objetivo: Esta evaluación busca evaluar tus habilidades para desarrollar y probar servicios web REST utilizando Jakarta EE. Deberás desarrollar un conjunto de servicios CRUD (Crear, Actualizar y Seleccionar) para un recurso específico, y validar el funcionamiento de estos servicios utilizando Postman.

Parte 1: Repositorio Git (1 punto)

1.1. Crear un repositorio público en GitHub o GitLab



Parte 2: Desarrollo de servicios web REST (4 puntos)

2.1. Crea un nuevo proyecto Jakarta EE con un módulo de servicios web, teniendo en cuenta las siguientes características:

- Nombre del proyecto: eva62<apellido>
- Nombre de la base de datos: mibd62
- Usuario de la base de datos: <cualquier usuario y contraseña>

New Maven Project

Select an Archetype

Catalog: All Catalogs Configure...

Filter: ✕

Group Id	Artifact Id	Version
org.wildfly.archetype	wildfly-html5-mobile-archetype	8.2.0.Final
org.wildfly.archetype	wildfly-html5-mobile-blank-archetype	8.2.0.Final
org.wildfly.archetype	wildfly-jakartaee-ear-archetype	27.0.0.Final
org.wildfly.archetype	wildfly-jakartaee-webapp-archetype	27.0.0.Final
org.wildfly.archetype	wildfly-javaee7-webapp-archetype	8.2.0.Final
org.wildfly.archetype	wildfly-javaee7-webapp-blank-archetype	8.2.0.Final
org.wildfly.archetype	wildfly-javaee7-webapp-ear-archetype	8.2.0.Final

An archetype that generates a starter Jakarta EE project for JBoss WildFly. The project is a WAR archive. It is prepared for Arquillian driven unit tests.
<https://repo1.maven.org/maven2>

☒ Show the last version of Archetype only ☐ Include snapshot archetypes Add Archetype...

► Advanced

? < Back Next > Finish Cancel

New Maven Project

Specify Archetype parameters

Group Id:

Artifact Id:

Version:

Package:

☒ run archetype generation interactively

Properties available from archetype:

Name	Value

Add...
Remove

► Advanced

? < Back Next > Finish Cancel

@ Javadoc Declaration Console × Progress Terminal Error Log Servers

C:\bin\ eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin\javaw.exe (13 jul. 2023 07:51:43) [pid: 13412]

[INFO] Using property: package = ec.edu.ups.ppw.eva62cabrera

Confirm properties configuration:

groupId: ec.edu.ups.ppw

artifactId: eva62cabrera

version: 0.0.1-SNAPSHOT

package: ec.edu.ups.ppw.eva62cabrera

Y: : Y|

```
[INFO] Parameter: groupId, Value: ec.edu.ups.ppw
[INFO] Parameter: artifactId, Value: eva62cabrera
[INFO] Parameter: version, Value: 0.0.1-SNAPSHOT
[INFO] Parameter: package, Value: ec.edu.ups.ppw.eva62cabrera
[INFO] Parameter: packageInPathFormat, Value: ec/edu/ups/ppw/eva62cabrera
[INFO] Parameter: package, Value: ec.edu.ups.ppw.eva62cabrera
[INFO] Parameter: groupId, Value: ec.edu.ups.ppw
[INFO] Parameter: artifactId, Value: eva62cabrera
[INFO] Parameter: version, Value: 0.0.1-SNAPSHOT
[INFO] Project created from Archetype in dir: C:\Users\Darwin\Documents\Documentos-Eclipse\eva62cabrera
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 56.661 s
[INFO] Finished at: 2023-07-13T08:33:56-05:00
[INFO] -----
```

Create - Login/Group Role

General

Definition

Privileges

Membership

Parameters

Security

SQL

Name

RolMotocicleta

Comments

i

?

X Close

↺ Reset

Save

Group Role - RolMotocicleta

General

Definition

Privileges

Membership

Parameters

Security

SQL

Can login?

☒

Superuser?

☒

Create roles?

☒

Create databases?

☒

Inherit rights from the parent roles?

☒

Can initiate streaming replication and backups?

☐

i

?

Close

Reset

Save

Contraseña: RolMotocicletaDarwin1001

Create - Database

General

Definition

Security

Parameters


Advanced

SQL

Database

mibd62

Owner

 RolMotocicleta

▼

Comment

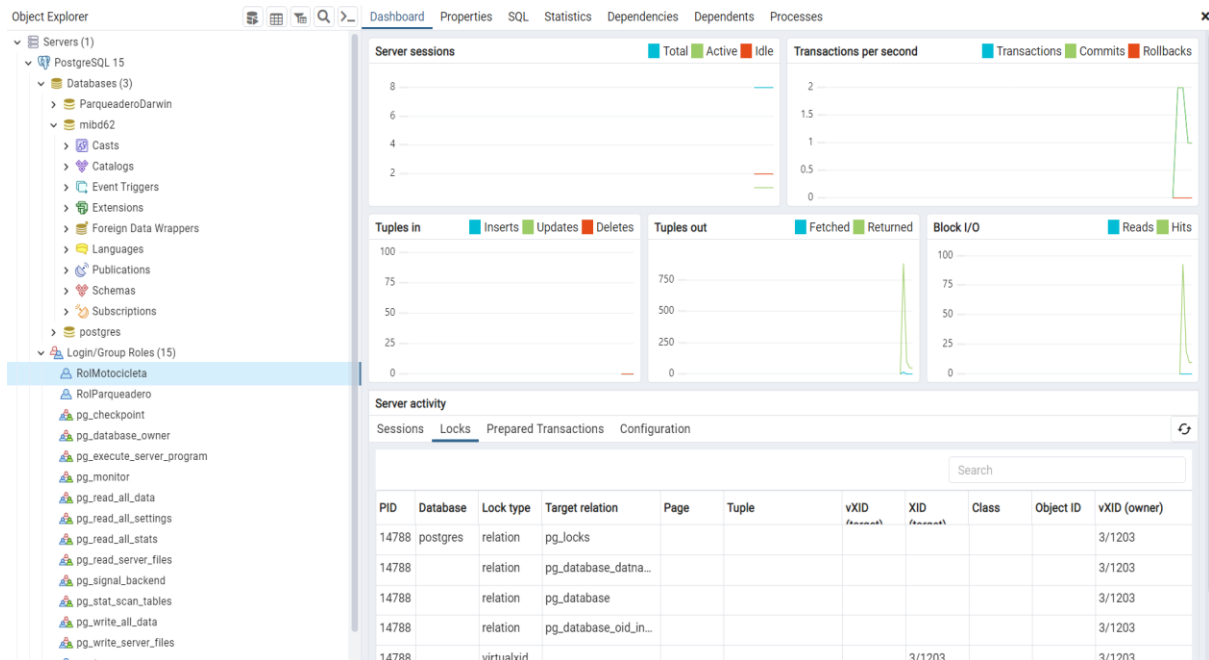
i

?

Close

Reset

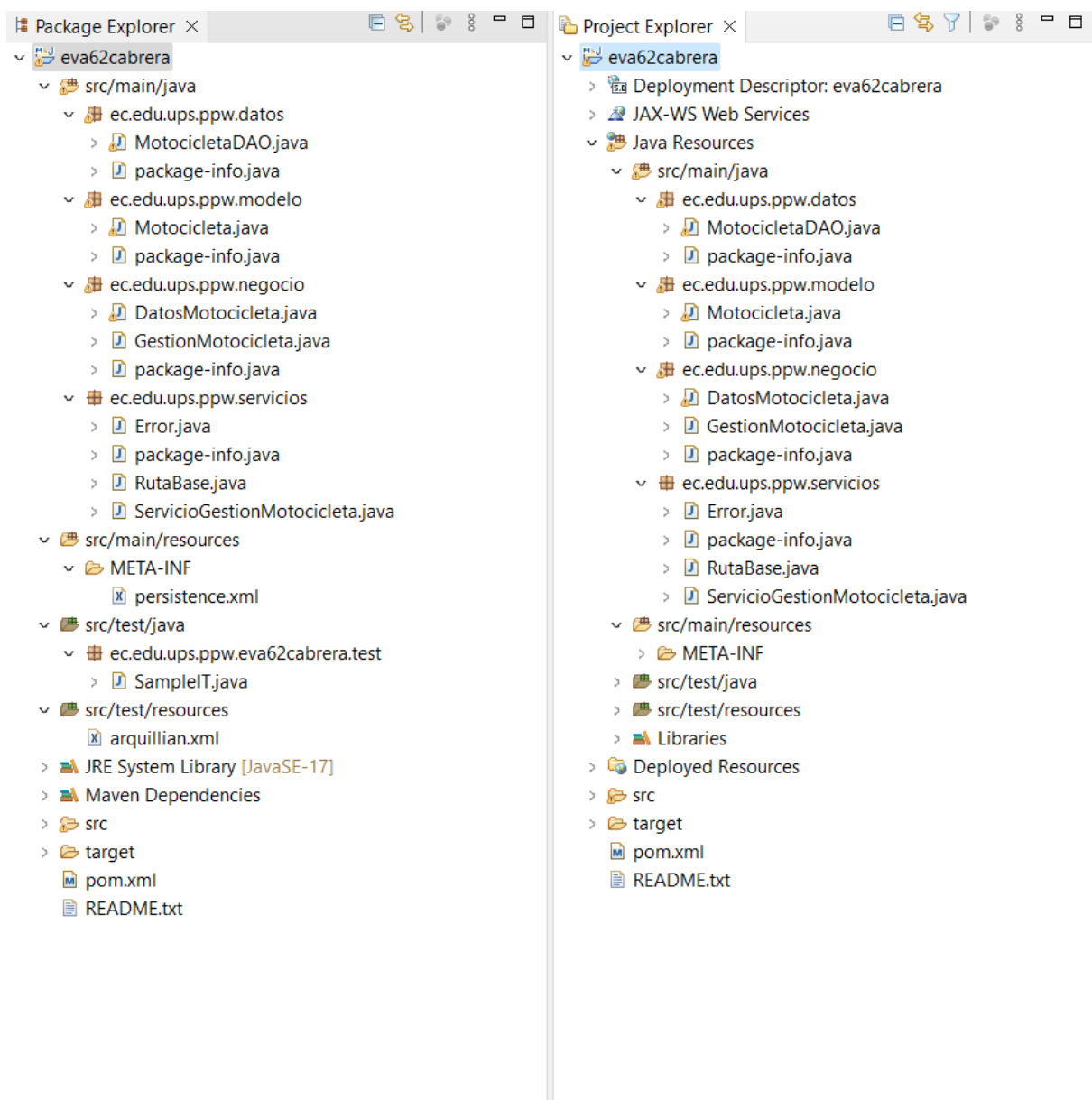
Save



```

</datasource>
<datasource jta="true" jndi-name="java:jboss/datasources/DataSourceMotocicleta" pool-name="DataSourceMotocicleta" enabled="true"
use-java-context="true" use-cm="true">
  <connection-url>jdbc:postgresql://localhost:5432/mibd62</connection-url>
  <driver>postgresql-driver</driver>
  <security>
    <user-name>RolMotocicleta</user-name>
    <password>RolMotocicletaDarwin1001</password>
  </security>
</datasource>
<drivers>
  <driver name="h2" module="com.h2database.h2">
    <xa-datasource-class>org.h2.jdbcx.JdbcDataSource</xa-datasource-class>
  </driver>
  <driver name="oracle-driver" module="com.oracle">
    <driver-class>oracle.jdbc.OracleDriver</driver-class>
  </driver>
  <driver name="postgresql-driver" module="org.postgresql">
    <driver-class>org.postgresql.Driver</driver-class>
  </driver>
</drivers>
</datasources>

```



2.2. Diseña y crea una entidad de persistencia que representará el recurso con el que trabajarán tus servicios. El recurso para considerar es: Motocicletas con al menos 4 atributos distintos.

2.3. Desarrolla 2 servicios web (guardar y seleccionar todos) para esta entidad. Asegúrate de que cada servicio maneje correctamente los códigos de estado HTTP y que envíe una respuesta JSON apropiada.

```
java:global/eva62cabrera/MotocicletaDAO!ec.edu.ups.ppw.datos.MotocicletaDAO
java:app/eva62cabrera/MotocicletaDAO!ec.edu.ups.ppw.datos.MotocicletaDAO
java:module/MotocicletaDAO!ec.edu.ups.ppw.datos.MotocicletaDAO
java:global/eva62cabrera/MotocicletaDAO
java:app/eva62cabrera/MotocicletaDAO
java:module/MotocicletaDAO
```

```
3:01:02,342 INFO [org.jboss.as.ejb3.deployment] (MSC service thread 1-8) WFLYEJB0473: JNDI bindings for session bean named 'DatosMotocicleta' in deployment unit 'deg
```

```
java:global/eva62cabrera/DatosMotocicleta!ec.edu.ups.ppw.negocio.DatosMotocicleta
java:app/eva62cabrera/DatosMotocicleta!ec.edu.ups.ppw.negocio.DatosMotocicleta
java:module/DatosMotocicleta!ec.edu.ups.ppw.negocio.DatosMotocicleta
java:global/eva62cabrera/DatosMotocicleta
java:app/eva62cabrera/DatosMotocicleta
java:module/DatosMotocicleta
```

```
package ec.edu.ups.ppw.servicios;

import java.util.List;

/**
 * @author Darwin Cabrera
 */
@Path("/rutamotocicleta")
public class ServicioGestionMotocicleta {
    @Inject
    private MotocicletaDAO daoMotocicleta;
    @Inject
    private GestionMotocicleta gestionarMotocicleta;

    @GET
    @Path("/datosfinales")
    @Produces("application/json")
    public Motocicleta datosMotocicleta() {
        Motocicleta moto = new Motocicleta();
        moto.setPlaca("FF1231");
        moto.setMarca("Kawasaki");
        moto.setColor("Amarillo");
        moto.setPrecio(1.231);
        return moto;
    }

    @POST
    @Produces("application/json")
    @Consumes("application/json")

    public Response almacenarMotocicleta(Motocicleta motocicleta) {
        try {
            gestionarMotocicleta.guardarMotocicleta(motocicleta);
            return Response.status(Response.Status.OK).entity(motocicleta).build();
        } catch (Exception e) {
            // TODO: handle exception
            e.printStackTrace();
            Error error = new Error();
            error.setCodigoError(99);
            error.setMensajeError("Error al intentar guardar los datos: " + e.getMessage());
            return Response.status(Response.Status.OK).entity(error).build();
        }
    }
}
```

```

        @GET
        @Path("listafinal")
        @Produces(MediaType.APPLICATION_JSON)
        public List<Motocicleta> listadoMotocicleta() {
            return daoMotocicleta.obtenerMotocicleta();
        }
    }

package ec.edu.ups.ppw.negocio;
import java.util.Iterator;

* @author Darwin Cabrera

@Singleton
@Startup
public class DatosMotocicleta {
    @Inject
    private MotocicletaDAO daoMotocicleta;

    @PostConstruct
    public void init() {
        Motocicleta m1 = new Motocicleta();
        m1.setPlaca("BB2121");
        m1.setColor("Verde");
        m1.setMarca("Yamaha");
        m1.setPrecio(2.220);
        daoMotocicleta.insertarMotocicleta(m1);
        Motocicleta m2 = new Motocicleta();
        m2.setPlaca("AA0000");
        m2.setColor("Azul");
        m2.setMarca("Suzuki");
        m2.setPrecio(2.681);
        daoMotocicleta.insertarMotocicleta(m2);

        Motocicleta m3 = new Motocicleta();
        m3.setPlaca("DD2132");
        m3.setColor("Rojo");
        m3.setMarca("BMW");
        m3.setPrecio(1.909);
        daoMotocicleta.insertarMotocicleta(m3);

        Motocicleta m4 = new Motocicleta();
        m4.setPlaca("CC2212");
        m4.setColor("Negro");
        m4.setMarca("Ducati");
        m4.setPrecio(1.359);
        daoMotocicleta.insertarMotocicleta(m4);
        List<Motocicleta> listarMotocicleta = daoMotocicleta.obtenerMotocicleta();
        System.out.println("-----MOTOCICLETAS:-----");
        for (Motocicleta moto : listarMotocicleta) {

```

Message

- i Server Startup Succeeded
- i Module eva62cabrera on WildFly 27 not yet fully deployed. Waiting...
- i Server Starting

Plug-in

- org.jboss.ide.eclipse.
- org.jboss.ide.eclipse.
- org.jboss.ide.eclipse.



Overview GET http://127.0.0.1:9990 No Environment

http://127.0.0.1:9990 Save Send

GET http://127.0.0.1:9990

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (7) Test Results Status: 200 OK Time: 13 ms Size: 849 B Save as Example

Pretty Raw Preview Visualize HTML

```
1 <!DOCTYPE html>
2 <html class="layout-pf layout-pf-fixed" lang="en">
3
4 <head>
5   <title>HAL Management Console</title>
6   <meta charset="utf-8">
7   <meta http-equiv="x-ua-compatible" content="ie=edge">
8   <meta name="viewport" content="width=device-width, initial-scale=1">
9   <link rel="icon shortcut" type="image/x-icon" href="favicon.db74b3b0.ico">
10  <link rel="apple-touch-icon" href="apple-touch-icon.fedc4af6.png">
11  <link rel="stylesheet" href="dev.291b1e16.css" media="screen">
12  <script type="module" src="dev.75953f7e.js"></script>
13  <script src="dev.159a2c8f.js" nomodule defer></script>
14  <script src="hal.nocache.js"></script>
15 </head>
16
17 <body> </body>
18
19 </html>
```

HTTP <http://localhost:8080/eva62cabrera> Save

GET <http://localhost:8080/eva62cabrera> Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (4) Test Results Status: 404 Not Found Time: 5 ms Size: 206 B Save as Example

Key	Value
Connection	keep-alive
Content-Length	74
Content-Type	text/html
Date	Thu, 13 Jul 2023 18:26:30 GMT

Parte 3: Pruebas con Postman (3 puntos)

3.1. Configura Postman para realizar pruebas a los servicios que has creado.

3.2. Realiza y documenta pruebas para cada uno de los servicios CRUD. Asegúrate de probar todos los escenarios posibles, incluyendo peticiones exitosas y fallidas.

Parte 4: Informe (2 puntos)

4.1. Crea un informe que documente el proceso de desarrollo y prueba de tus servicios. Este informe debe incluir:

- Capturas de pantalla de la salida de mensajes en la consola que evidencien el funcionamiento de cada servicio.

- Capturas de pantalla de Postman mostrando los resultados de tus pruebas.

Por favor, asegurarse de que el informe esté bien organizado y sea fácil de seguir. La claridad y precisión de tu informe son tan importantes como el funcionamiento de tus servicios.

Adicional al informe incluir la URL del repositorio git. (el repositorio debe ser público)