

Documento Despliegue proyecto FRANQUICIA

Springboot

Para:
xxxxxx

Por:
Javier Orlando Mantilla Portilla
"jmantillap"

Tabla de contenido

1	ASPECTOS TECNICOS	3
1.1	HERRAMIENTAS UTILIZADAS PARA EL DESARROLLO BACKEND	3
1.1.1	JDK17	3
1.1.2	Spring Tool Suite.....	3
1.1.3	GIT	3
1.1.4	POSTMAN	4
1.2	ASPECTO TÉCNICO DE DESARROLLO DEL SISTEMA.....	4
1.2.1	Base de datos.	4
1.2.2	Configuración del proyecto	5
1.2.3	Estructura del proyecto.....	5
1.2.4	Pruebas unitarias de cobertura.....	6
2	RECURSOS NECESARIOS PARA MONTAJE LOCAL	7
2.1	Archivos	7
2.1.1	Montaje de la base de datos:.....	7
2.1.2	Colección de Postman para el consumo de los servicios.	7
3	CREACION DE IMAGEN DOCKER Y CONTENEDOR	8
3.1	Pasos para el montaje.....	8
3.1.1	Archivo DockerFile del proyecto	8
3.1.2	Creación de la imagen del contenedor.	8
3.1.3	Creación del contenedor a partir de la imagen.	9
3.1.4	Prueba de resultados desde postman.	10

1 ASPECTOS TECNICOS

1.1 HERRAMIENTAS UTILIZADAS PARA EL DESARROLLO BACKEND

1.1.1 JDK17

Java™ Development Kit (JDK) es un software para los desarrolladores de Java. Incluye el intérprete Java, clases Java y herramientas de desarrollo Java (JDT): compilador, depurador, desensamblador, visor de applets, generador de archivos de apéndice y generador de documentación.

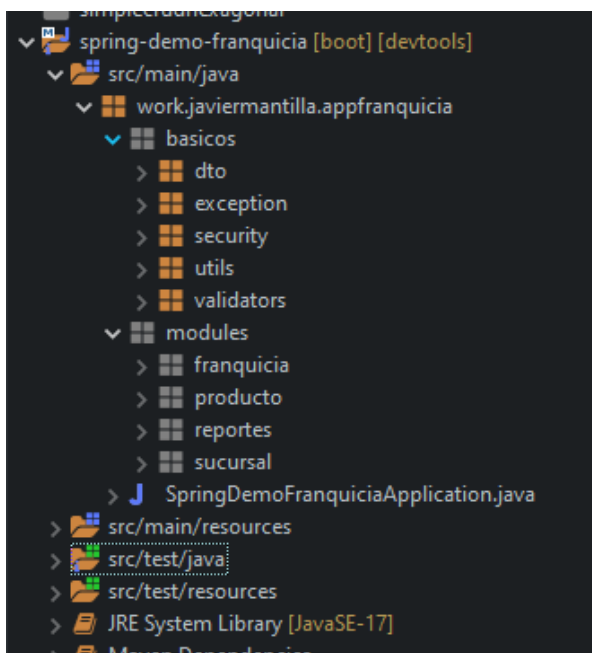
<https://jdk.java.net/java-se-ri/17>

1.1.2 Spring Tool Suite

Spring Tool Suite es un IDE para desarrollar aplicaciones Spring. Es un entorno de desarrollo basado en Eclipse. Proporciona un entorno listo para usar para implementar, ejecutar, implementar y depurar la aplicación. Valida nuestra aplicación y proporciona soluciones rápidas para las aplicaciones.

<https://www.javatpoint.com/spring-boot-download-and-install-sts-ide>

<https://www.springla.io/spring/spring-tool-suite>



1.1.3 GIT

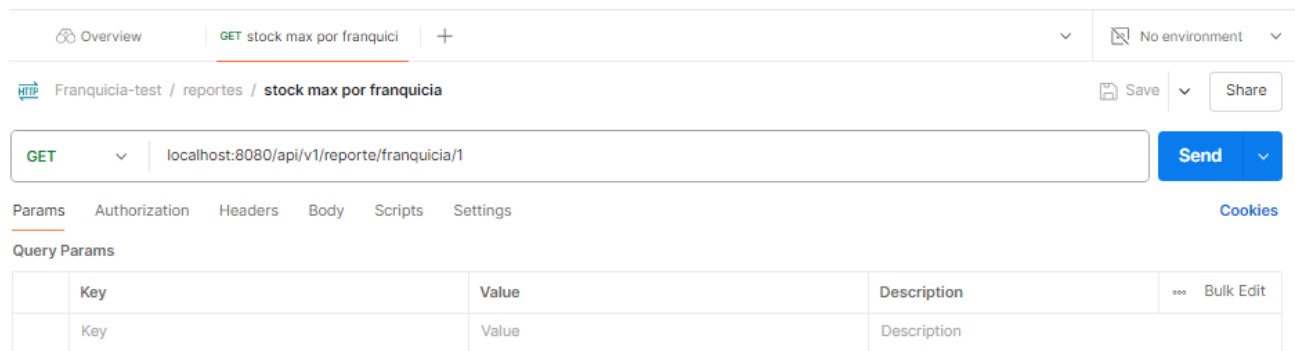
Git es un sistema de control de versiones distribuido: seguimiento de cambios en cualquier conjunto de archivos, generalmente utilizado para coordinar el trabajo entre programadores que desarrollan en colaboración el código fuente durante el desarrollo del software

La ubicación del proyecto está en la ruta.

<https://github.com/jmantillap/spring-demo-franquicia>

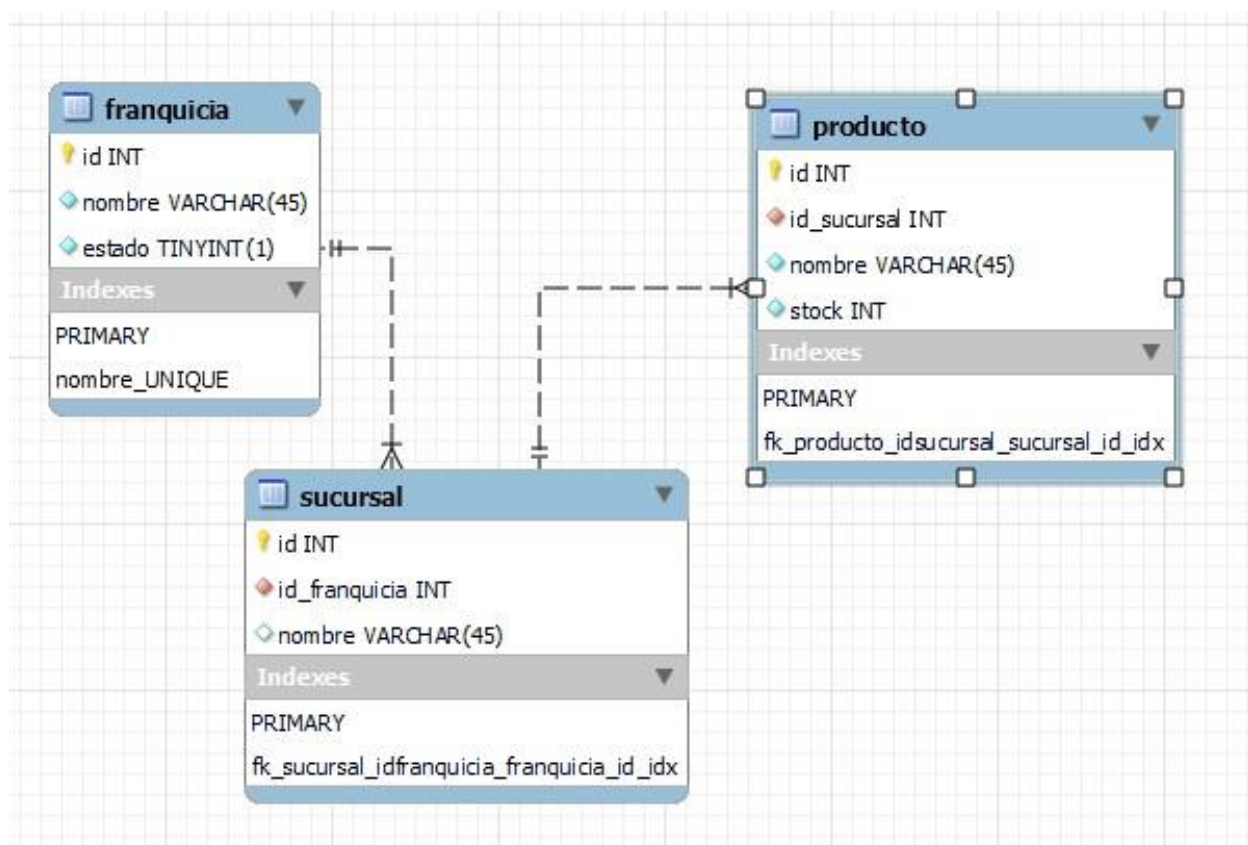
1.1.4 POSTMAN

Postman es una popular herramienta utilizada para probar APIs, permitiendo a los desarrolladores enviar peticiones a servicios web y ver respuestas



1.2 ASPECTO TÉCNICO DE DESARROLLO DEL SISTEMA

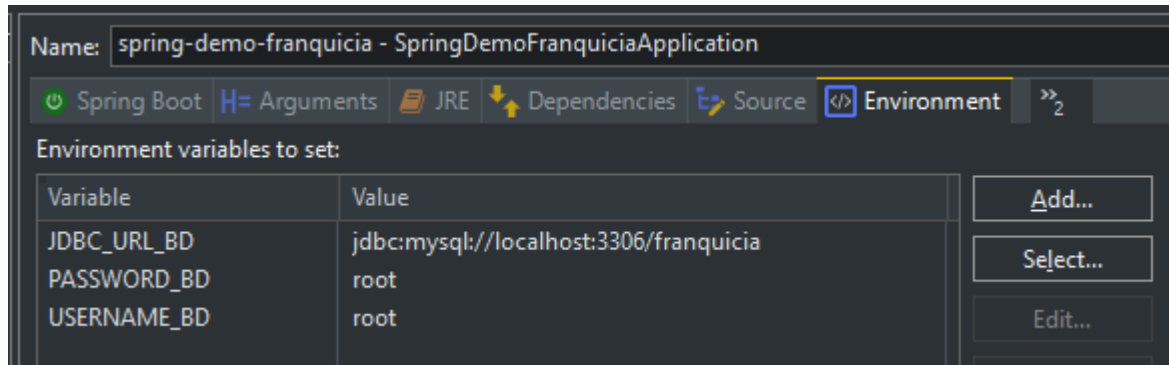
1.2.1 Base de datos.



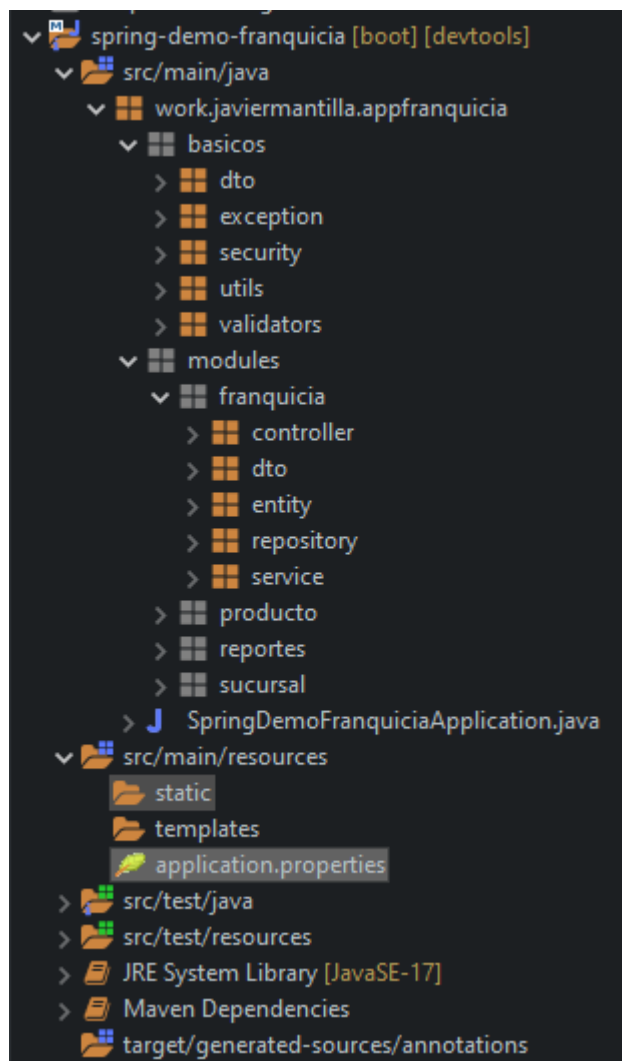
1.2.2 Configuración del proyecto

Para que funcione el proyecto localmente y conectar a la base de datos:

- Valores:
 - JDBC_URL_BD= jdbc:mysql://localhost:3306/franquicia
 - PASSWORD_BD=root
 - USERNAME_BD=root




1.2.3 Estructura del proyecto








1.2.4 Pruebas unitarias de cobertura

\$ mvn clean test

 demo-franquicia

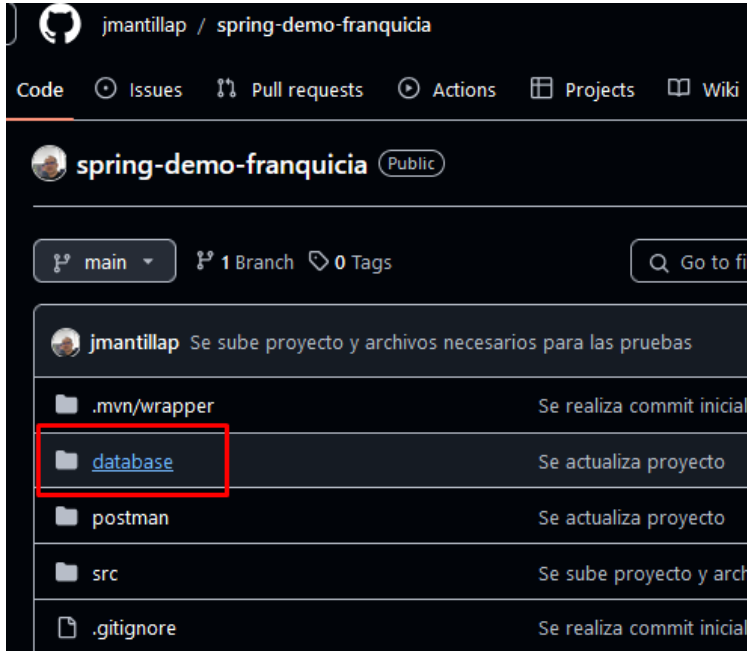
demo-franquicia

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed
 work.javiermantilla.appfranquicia.modules.franquicia.service.impl		100 %		100 %	0
 work.javiermantilla.appfranquicia.modules.franquicia.controller		100 %		n/a	0
Total	0 of 186	100 %	0 of 6	100 %	0

2 RECURSOS NECESARIOS PARA MONTAJE LOCAL

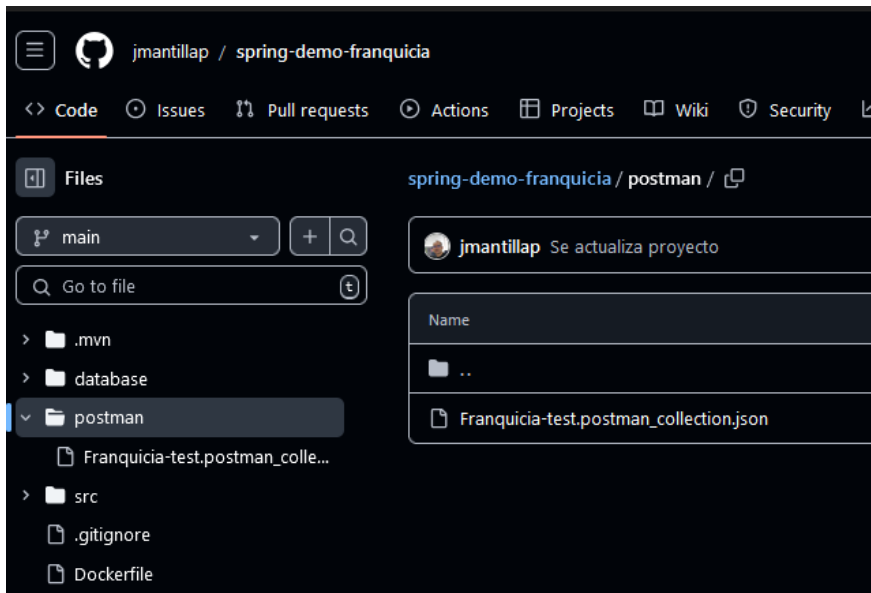
2.1 Archivos

2.1.1 Montaje de la base de datos:



[database/Fanquicia20240915.sql](#)

2.1.2 Colección de Postman para el consumo de los servicios.



[postman/Franquicia-test.postman_collection.json](#)

3 CREACION DE IMAGEN DOCKER Y CONTENEDOR

3.1 Pasos para el montaje

3.1.1 Archivo DockerFile del proyecto

```
Dockerfile X
1 FROM openjdk:17-jdk-slim
2
3 LABEL org.opencontainers.image.authors="jmantillap@gmail.com"
4 #host.docker.internal --> maquina fisica donde esta la base de data. host anfitrión
5 ENV JDBC_URL_BD=jdbc:mysql://host.docker.internal:3306/franquicia
6 ENV USERNAME_BD=root
7 ENV PASSWORD_BD=root
8
9 COPY target/demo-franquicia-1.0.jar /app/app.jar
10
11 ENTRYPOINT ["java", "-jar", "/app/app.jar"]
12
```

3.1.2 Creación de la imagen del contenedor.

Se debe tener el plugin en el pom.xml de dockerfile-maven-plugin

```
<plugin>
  <groupId>com.spotify</groupId>
  <artifactId>dockerfile-maven-plugin</artifactId>
  <version>1.4.13</version>
  <executions>
    <execution>
      <id>default</id>
      <goals>
        <goal>build</goal>
        <goal>push</goal>
      </goals>
    </execution>
  </executions>
  <configuration>
    <repository>jmantillap/franquicia-app</repository>
    <tag>${project.version}</tag>
    <buildArgs>
      <JAR_FILE>${project.build.finalName}.jar</JAR_FILE>
    </buildArgs>
  </configuration>
</plugin>
```

Ejecutar el comando maven que genera el jar y crea la imagen con base en el archivo Dockerfile que tiene el proyecto.

\$ mvn clean verify

```
[INFO]
[INFO] ---> Running in a0dd2cddf408
[INFO] Removing intermediate container a0dd2cddf408
[INFO] ---> 7c9efc17c3a5
[INFO] [Warning] One or more build-args [JAR_FILE] were not consumed
[INFO] Successfully built 7c9efc17c3a5
[INFO] Successfully tagged jmantillap/franquicia-app:1.0
[INFO]
[INFO] Detected build of image with id 7c9efc17c3a5
[INFO] Building jar: D:\workspace-spring-tool\spring-demo-franquicia\target\demo-franquicia-1.0-docker-info.jar
[INFO] Successfully built jmantillap/franquicia-app:1.0
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 04:09 min
[INFO] Finished at: 2024-09-15T22:02:33-05:00
[INFO] -----
```


\$ docker images

Images

[Give feedback](#)

Local



Hub


Artifactory

EARLY ACCESS

0 Bytes / 573.78 MB in use

5 images



<input type="checkbox"/>	Name	Tag
<input type="checkbox"/>	jmantillap/franquicia-app 7c9efc17c3a5 	1.0

```
$ docker run -it --rm --name franquicia-contenedor-app -p 8080:8080 <identificadorImagen>
```

Containers [Give feedback](#)

Container CPU usage

0.15% / 200% (2 cores allocated)

Container memory usage

219.2MB / 953.7MB

Show charts

Search

Only show running containers

	Name	Image	Status	CPU (%)	Port(s)	Last started	Actions
	<div>franquicia-contenedor</div> <div>a3d690de9ea9</div>	7c9efc17c3a5	Running	0.15%	8080:8080	3 minutes ago	<div></div> <div></div> <div></div>

3.1.4 Prueba de resultados desde postman.

Francia-test / reportes / stock max por franquicia

GET

localhost:8080/api/v1/reportes/franquicia/1

Send

Params

Authorization

Headers (7)

Body

Scripts

Settings

Cookies

Query Params

Key	Value	Description	Bulk Edit
Key	Value	Description	

Body

Cookies

Headers (8)

Test Results

200 OK · 1654 ms · 738 B · Save Response

Pretty

Raw

Preview

Visualize

JSON

```
1 {
2   "data": {
3     "franquicia": {
4       "id": 1,
5       "nombre": "Franquicia 1",
6       "estado": true
7     },
8     "sucursales": [
9       {
10        "sucursal": {
11          "id": 1,
12          "nombre": "Sucursal Actualizada"
13        },
14        "productoXSucursal": [
15
```

franquicia-contenedor-app

7c9efc17c3a5

a3d690de9ea9

8080:8080

STATUS

Running (4 minutes ago)

Logs

Inspect

Bind mounts

Exec

Files

Stats

```
2024-09-15 22:10:08 2024-09-16T03:10:08.482Z INFO 1 --- [spring-demo-franquicia] [main] o.s.d.j.r.query.QueryEnhancerFactory : Hibernate is
in classpath; If applicable, HQL parser will be used.
2024-09-15 22:10:09 2024-09-16T03:10:09.596Z WARN 1 --- [spring-demo-franquicia] [main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.op
en-in-view is enabled by default. Therefore, database queries may be performed during view rendering. Explicitly configure spring.jpa.open-in-view to disabl
e this warning
2024-09-15 22:10:10 2024-09-16T03:10:10.819Z INFO 1 --- [spring-demo-franquicia] [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat starte
d on port 8080 (http) with context path '/'
2024-09-15 22:10:10 2024-09-16T03:10:10.866Z INFO 1 --- [spring-demo-franquicia] [main] w.j.a.SpringDemoFranquiciaApplication : Started Sprin
gDemoFranquiciaApplication in 18.422 seconds (process running for 20.549)
2024-09-15 22:13:37 2024-09-16T03:13:37.518Z INFO 1 --- [spring-demo-franquicia] [nio-8080-exec-2] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing
Spring DispatcherServlet 'dispatcherServlet'
2024-09-15 22:13:37 2024-09-16T03:13:37.522Z INFO 1 --- [spring-demo-franquicia] [nio-8080-exec-2] o.s.web.servlet.DispatcherServlet : Initializing
Servlet 'dispatcherServlet'
2024-09-15 22:13:37 2024-09-16T03:13:37.527Z INFO 1 --- [spring-demo-franquicia] [nio-8080-exec-2] o.s.web.servlet.DispatcherServlet : Completed ini
tialization in 4 ms
2024-09-15 22:13:37 2024-09-16T03:13:37.681Z INFO 1 --- [spring-demo-franquicia] [nio-8080-exec-2] w.j.a.m.r.controller.ReporteController : Consulta de l
ista de productos con más stock en la sucursal perteneciente a la franquicia id: 1
2024-09-15 22:13:37 Hibernate: select fe1_0.id,fe1_0.estado,fe1_0.nombre from franquicia fe1_0 where fe1_0.id=?
2024-09-15 22:13:38 Hibernate: select se1_0.id,se1_0.id.franquicia,f1_0.id,f1_0.estado,f1_0.nombre,se1_0.nombre from sucursal se1_0 join franquicia f1_0 on
f1_0.id=se1_0.id.franquicia where se1_0.id.franquicia=?
2024-09-15 22:13:38 Hibernate: SELECT s.id as idSucursal,s.nombre,p.id as idProducto,p.nombre,p.stock FROM sucursal s INNER JOIN producto p ON (p.id_sucursal
=s.id) WHERE s.id = ? AND p.stock= (SELECT max(p2.stock) FROM sucursal s2 INNER JOIN producto p2 ON (p2.id_sucursal = s2.id) WHERE s2.id=
?)
2024-09-15 22:13:38 Hibernate: SELECT s.id as idSucursal,s.nombre,p.id as idProducto,p.nombre,p.stock FROM sucursal s INNER JOIN producto p ON (p.id_sucursal
=s.id) WHERE s.id = ? AND p.stock= (SELECT max(p2.stock) FROM sucursal s2 INNER JOIN producto p2 ON (p2.id_sucursal = s2.id) WHERE s2.id=
?)
```

Gracias

Contacto developer:

jmantillap@gmail.com

Floridablanca, Colombia
