



Curso de
Java Spring

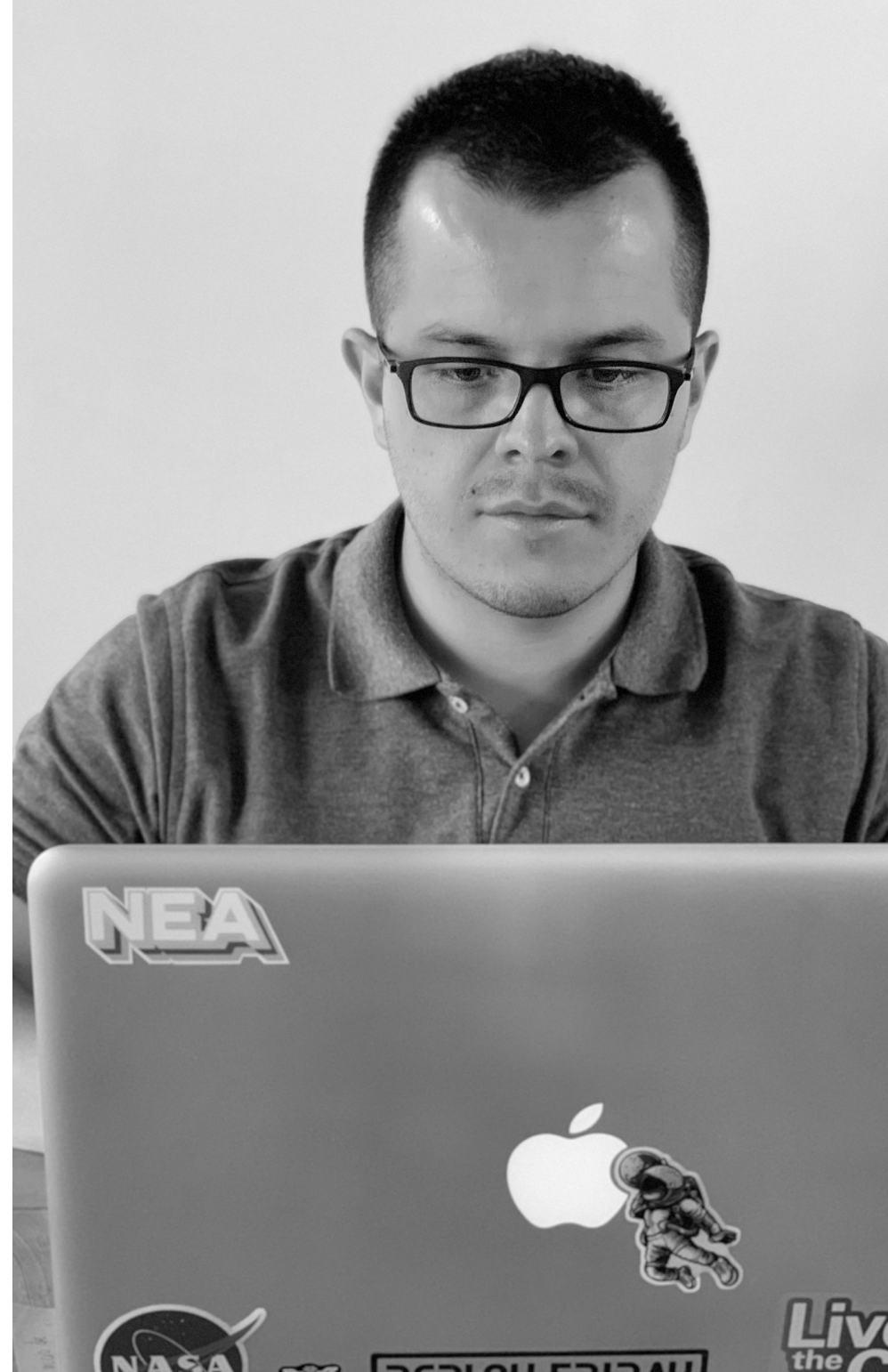
Alejandro Ramírez

¿Quién soy?



- Líder Técnico
- Profesor de cátedra
- Aprendizaje continuo
- Software de calidad

@soyalejoramirez



“

**Sólo se aprende si
existe emoción**

”

Francisco Mora

¿Qué vamos a construir?




¿Java sigue
siendo gratuito?

Cambio de licencia del JDK de Java

- Oracle siempre ha tenido una relación amor / odio con el Open Source
- Desde Java 9, cada 6 meses existe una versión “mayor”
- **JDK 11**
- Nueva licencia para uso en producción / comercial y LTS

Alternativas al JDK de Java

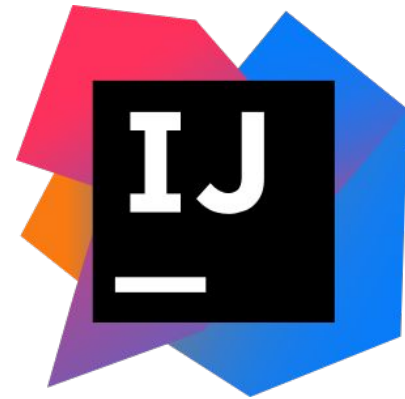
- OpenJDK 
- Alternativas basadas en OpenJDK
 - Amazon Corretto
 - RedHat OpenJDK
 - Otras



Instalar el ambiente de desarrollo

¿Qué necesitamos?

OpenJDK



PostgreSQL



¿Qué es y qué
usaremos de Spring?



¿Qué es Spring?

- Es el framework más popular de Java.
- ¿Qué soluciona y en qué ayuda?
- La comunidad es ENORME.
- Usado por compañías como Netflix y Mercedes-Benz.
- Posee una estructura modular y flexible.

¿Qué usaremos de Spring?



Spring Framework



Spring Boot



Spring Data JPA



Spring Security

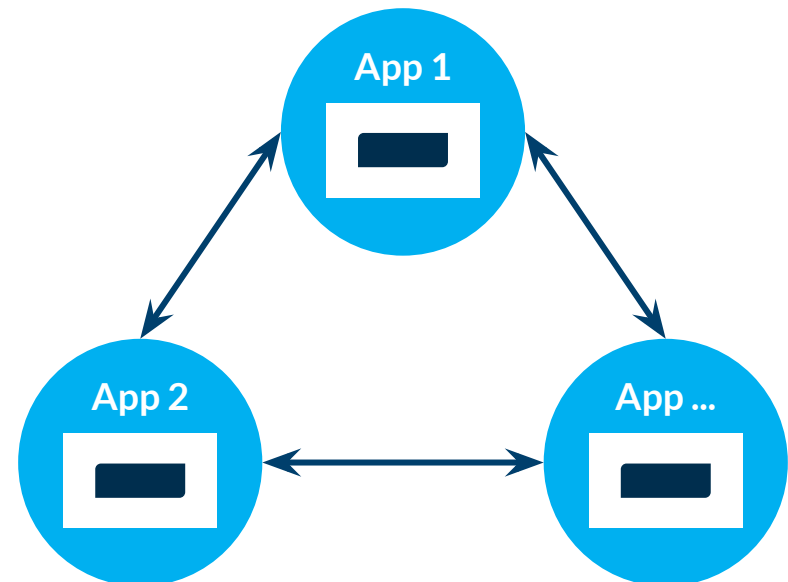
<https://spring.io/projects>

Conocer qué es
una aplicación
autocontenida

¿Cómo funcionan las aplicaciones autocontenidas?



Despliegue usando un servidor de aplicaciones



Despliegue de aplicación autocontenida

Spring Boot

- Es el proyecto de Spring para aplicaciones autocontenidas.
- Olvidarnos de la infraestructura y centrarnos en el desarrollo.
- Puede funcionar con Tomcat (por defecto), Jetty o Undertow.
- Incluye gestión de dependencias iniciales, configuración automática y más.

Crear nuestra aplicación con Spring Initializr

¿Qué es Spring Initializr?

start.spring.io

- Sitio oficial para generar un proyecto de Spring Boot
- En poco tiempo y a nuestra medida
- Con todo lo que necesitamos para empezar
- ¡Hagámoslo!

```
.gitignore
build.gradle
gradlew
gradlew.bat
HELP.md
settings.gradle
```

```
gradle
+---wrapper
      gradle-wrapper.jar
      gradle-wrapper.properties
```

```
src
+---main
|   +---java
|   |   \---com
|   |       \---platzi
|   |           \---market
|   |               PlatziMarketApplication.java
|   \---resources
|       application.properties
\---test
    \---java
        \---com
            \---platzi
                \---market
                    PlatziMarketApplicationTests.java
```

"Hola mundo" con Spring Boot



Configurar Spring Boot

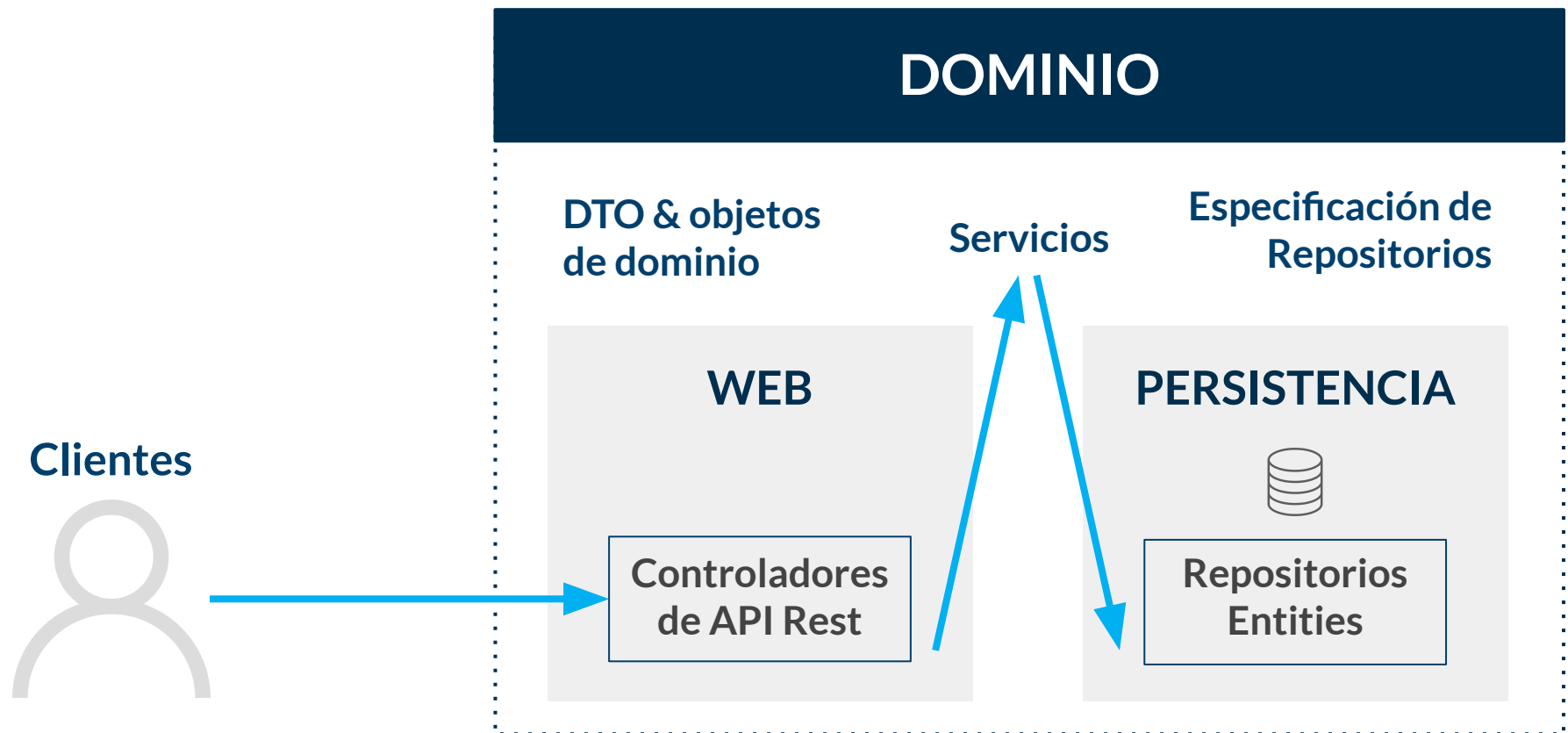


Propiedades de la aplicación

- `application.properties`, `application.yml` o línea de comando.
- Posibilidad de añadir propiedades propias.
- Gestión de perfiles según el tipo de despliegue.

Crear la estructura del proyecto

La estructura de nuestro proyecto



¿Qué es JPA?

JPA

- JPA es una especificación de Java (un estándar) para un framework ORM.
- Interactuar con las tablas de la base de datos en forma de objetos Java.
- Algunas de sus implementaciones son:
 - Hibernate
 - EclipseLink
 - TopLink
 - ObjectDB



Anotaciones de JPA

- @Entity
- @Table
- @Column
- @Id & @EmbeddedId
- @GeneratedValue
- @OneToMany & @ManyToOne

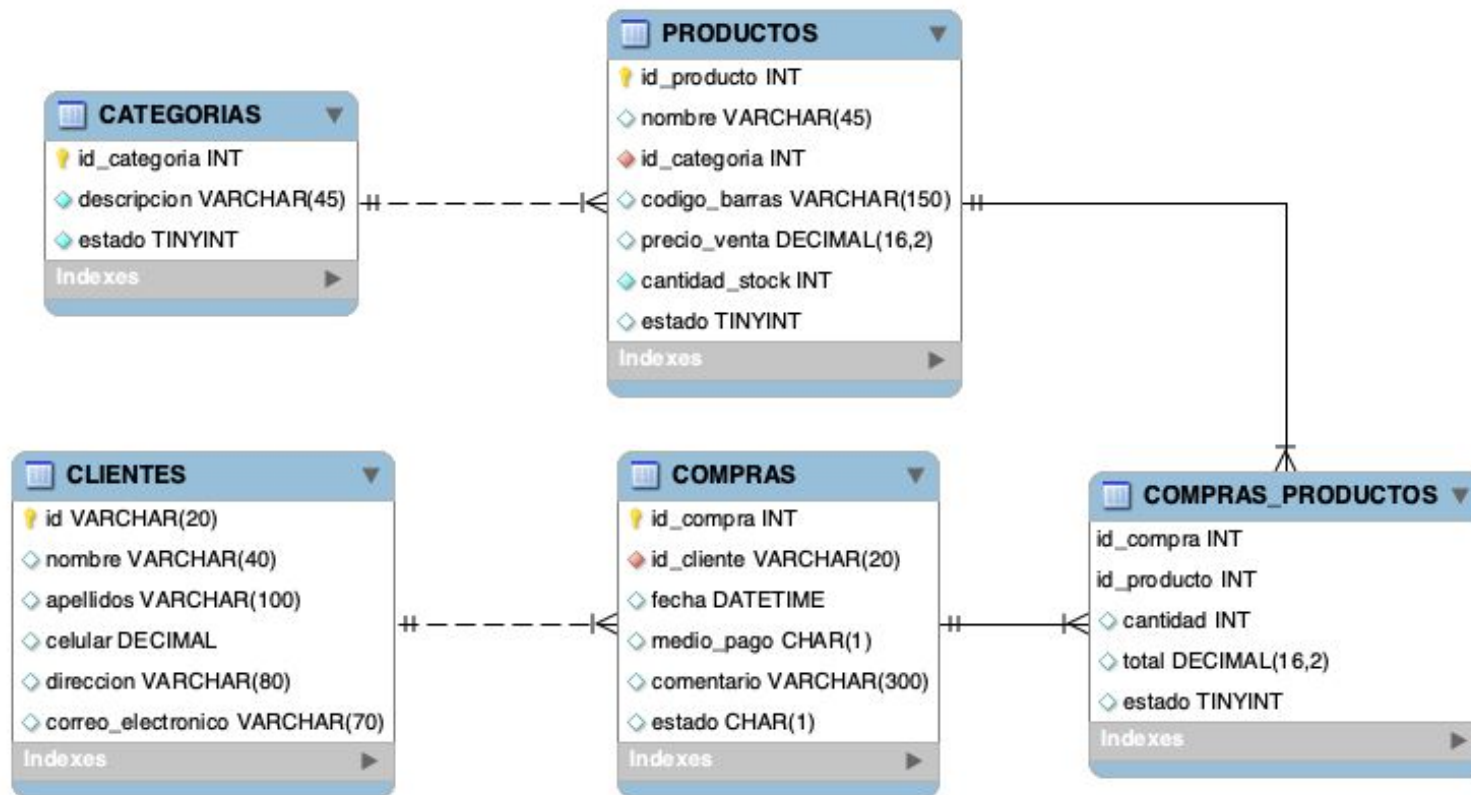
Conocer qué es Spring Data



Spring Data

- Es un proyecto que internamente contiene otros, nosotros usaremos el Spring Data JPA
- Optimización de tareas repetitivas
- Repositorios sin código con JpaRepository, CrudRepository & PagingAndSortingRepository
- Auditorías transparentes

**Conectar la base
de datos a nuestra
aplicación**



Mapear las tablas como clases

Crear Entity cuando
su clave primaria es
compuesta



Mapear relaciones entre clases

Usar la interface
CrudRepository

Spring Data Repositories

- Ahorrar un MONTÓN de código y tiempo de implementación
- Operaciones SIN CÓDIGO en la BD
- Repositorios de Spring Data

CrudRepository

PagingAndSortingRepository

JpaRepository



Query Methods



Uso de Query Methods

- En ocasiones, necesitamos consultas que el Repository de Spring Data no nos puede ofrecer.
- Los Query Methods proveen la posibilidad de generar consultas mediante el nombre de los métodos.
- Tienen la posibilidad de retornar `Optional<T>`

En SQL

```
SELECT    *  
FROM      productos  
WHERE     id_categoria = ?  
ORDER BY  nombre ASC;
```

Con Query Methods

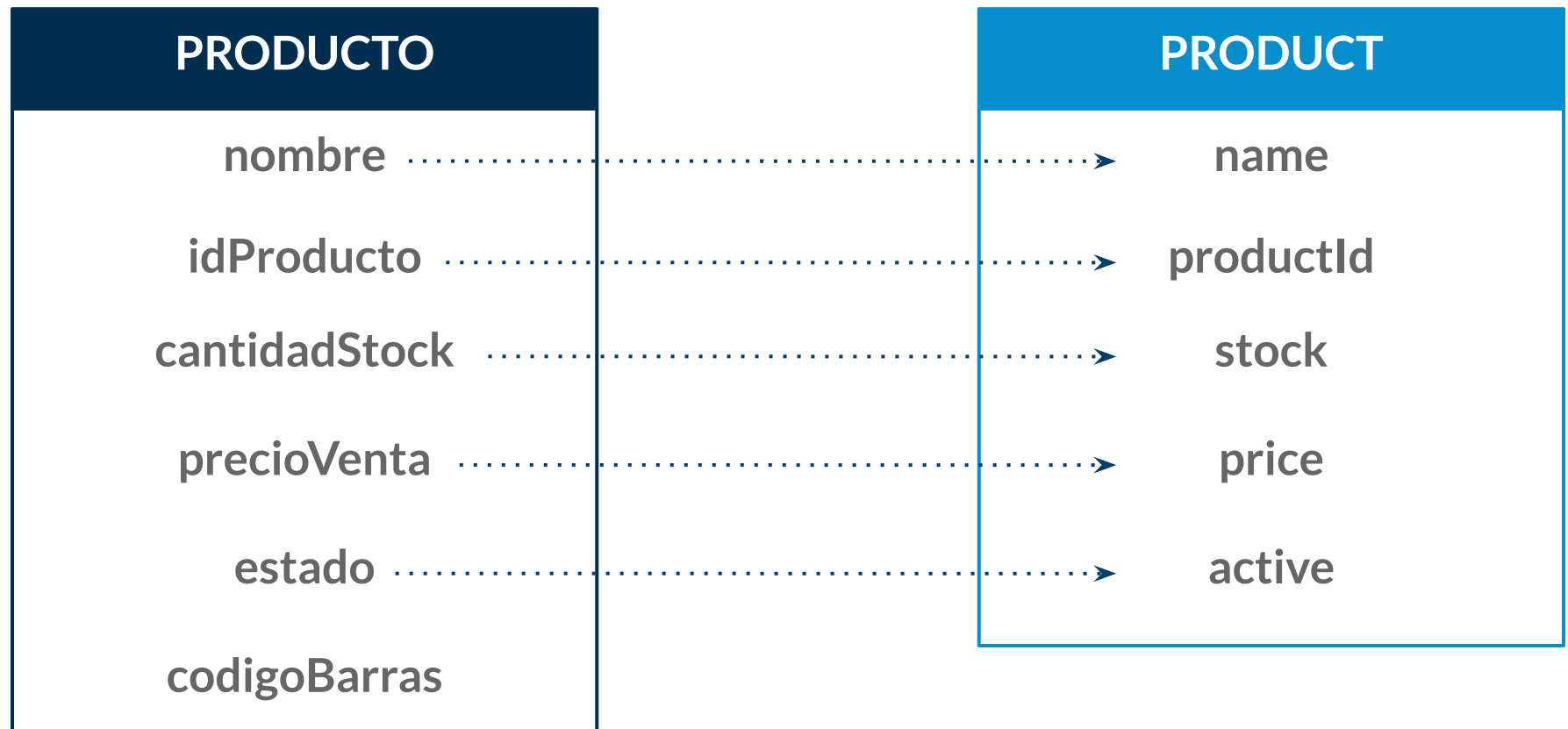
```
findByIdCategoriaOrderByNombreAsc(int idCategoria);
```

<https://docs.spring.io/spring-data/jpa/docs/current/reference/html/#jpa.query-methods>

Implementar la
anotación
`@Repository`

¿Qué es el patrón
Data Mapper y qué
resuelve?

En qué consiste el patrón Data Mapper



¿Y esto en qué nos ayuda?

Logramos varias cosas...

- No exponer la base de datos en el API
- Desacoplar nuestra API a una base de datos puntual
- No tener campos innecesarios en el API
- Sin mezclar idiomas en el dominio

Orientar nuestra API al dominio con MapStruct

Orientar nuestro repositorio a términos del dominio

Inyección de dependencias

Inyección de dependencias

- Principios S.O.L.I.D
- Inyección de dependencias (DI)
- Inversión de Control (IoC)
- Spring y @Autowired

Implementar la anotación @Service

Implementar la
anotación
`@RestController`



Exponer nuestra API

¿Qué anotaciones usaremos?

- Nuestra API se expone por `@RestController`
- Los métodos se exponen con `@GetMapping`, `@PostMapping` ó `@DeleteMapping`

Controlar las respuestas HTTP



ResponseEntity

- ¿Qué es y en qué nos ayuda?
- HttpStatus

Crear el dominio de compras

Mapear el dominio de compras

Crear el repositorio de compras

**Probando nuestros
servicios de compras**

Documentar nuestra API con Swagger

¿Por qué documentar nuestra API?

- Le agregaremos una capa de entendimiento
- Es más fácil de usar
- Es más profesional
- Quien consume tendrá información oficial y de primera mano



Configurar la seguridad de nuestra API con Spring Security



Spring Security

- Autenticación y autorización para aplicaciones de Spring.
- Como todos los proyectos de Spring, es muy fácil de configurar.
- Protección ante ataques como Session Fixation, clickjacking, cross site request forgery, entre otros.
- Configuración por defecto.

Generar un JWT

Autenticación con JWT

Qué es un JWT

- Estándar de código abierto basado en JSON para crear tokens de seguridad
- La autenticación viaja en el header de la petición:

Authorization: Bearer <token>

Cómo funciona un JWT

1

2

3

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MzkwMjQ.DIyfQ.XbPfbIHMI6arZ3Y922BhjWgQzWXcXNrZ0ogtVhfEd2o

1

Header

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

2

Payload

```
{
  "sub": "1234567890",
  "name": "John Doe",
  "iat": 1516239022
}
```

3

Signature

```
HMACSHA256 (
  BASE64URL(header)
  .
  BASE64URL(payload) ,
  secret)
```

Autorización con JWT

**Desplegar nuestra
API desde la ventana
de comandos**

```
java -jar platzi-market-1.0.jar
```

Algunas propiedades adicionales

- -Xmx2048m
- -Dspring.profiles.active=pdn
 - -Dserver.port=88

Desplegar nuestra base de datos con Heroku



¿Qué necesitamos?

- Cuenta de Heroku
- Heroku CLI
- Git

Desplegar nuestra API con Heroku

Conclusiones y despedida del curso

¡FELICIDADES!

- ¿Qué aprendimos?
- Me encantaría ver expuesta tu API

¡Nunca pares de aprender! 🚀

@soyalejoramirez