Contenido

[1. Introduccion al Curso Spring Boot - Angular 4 2](#_Toc498547711)

[1.1. Objetivos 2](#_Toc498547712)

[2. Introducción Spring 2](#_Toc498547713)

[2.1. Que es Spring Boot 2](#_Toc498547714)

[2.2. Entorno de dedesarrollo 2](#_Toc498547715)

[2.3. Herramientas 2](#_Toc498547716)

[2.4. Spring Initializer 2](#_Toc498547717)

[3. Primer Ejemplo Spring Boot 2](#_Toc498547718)

[3.1. Importar IDE 2](#_Toc498547719)

[3.2. Estructura del proyecto 2](#_Toc498547720)

[3.3. Spring Beans / Dependency Injection 2](#_Toc498547721)

[4. Spring MVC 2](#_Toc498547722)

[4.1. REST API Intro 2](#_Toc498547723)

[4.2. Método GET 2](#_Toc498547724)

[4.3. Método POST 2](#_Toc498547725)

[4.4. Método PUT 2](#_Toc498547726)

[4.5. Método DELETE 2](#_Toc498547727)

[5. Introducción Angular 2](#_Toc498547728)

[5.1. Qué es Angular 4 2](#_Toc498547729)

[5.2. Versiones de Angular 2](#_Toc498547730)

[5.3. Herramientas 2](#_Toc498547731)

[6. Primer Ejemplo Angular 3](#_Toc498547732)

[6.1. Estructura del Proyecto 3](#_Toc498547733)

[6.2. Data Binding 3](#_Toc498547734)

[6.3. Qué es TypeScript 3](#_Toc498547735)

[6.4. Usando Bootstrap 3](#_Toc498547736)

[7. Componentes 3](#_Toc498547737)

[7.1. Crear componentes 3](#_Toc498547738)

[7.2. Crear barra de navegación 3](#_Toc498547739)

[7.3. Router – Navegación 3](#_Toc498547740)

[7.4. Directivas – lista bootstrap 3](#_Toc498547741)

[7.5. Creando interfaz para Artículo 3](#_Toc498547742)

[8. API REST 3](#_Toc498547743)

[8.1. Agregar Artículo 3](#_Toc498547744)

[8.2. Editar Artículo 3](#_Toc498547745)

[8.3. Eliminar Artículo 3](#_Toc498547746)

[9. Despedida 3](#_Toc498547747)

[9.1. Promoción 3](#_Toc498547748)

# Introduccion al Curso Spring Boot - Angular 4

## Objetivos

What will your target student be able to do by the end of the course?

Clarify your students’ ending point. This will help you identify the skills you’d like to

help people acquire.

Think about…

what skills do your students need to learn to reach their goal?

what will be different about your students after they take your course?

Remember:

1. Goals should be realistic and measurable, meaning students should be able to demonstrate their skill.

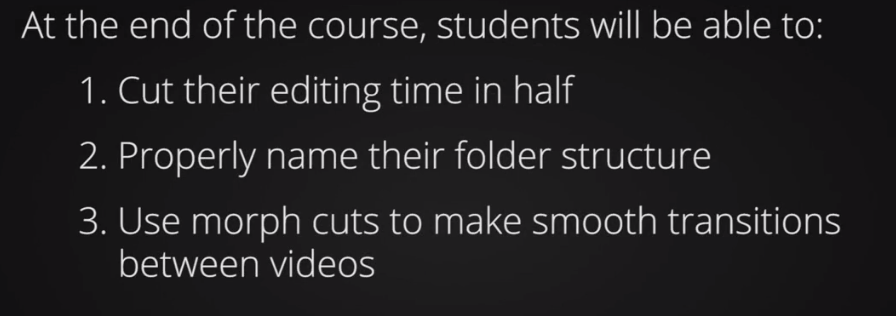
2. Include strong action verbs (“build”, “create”, “explain”, “differentiate”). If possible, avoid fuzzy

words (like “understand”, “know”, “learn”).

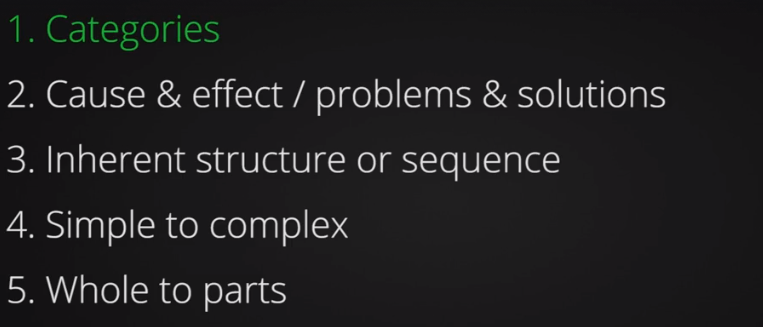
*Example: At the end of my course, students will be able to brainstorm an idea for a novel from*

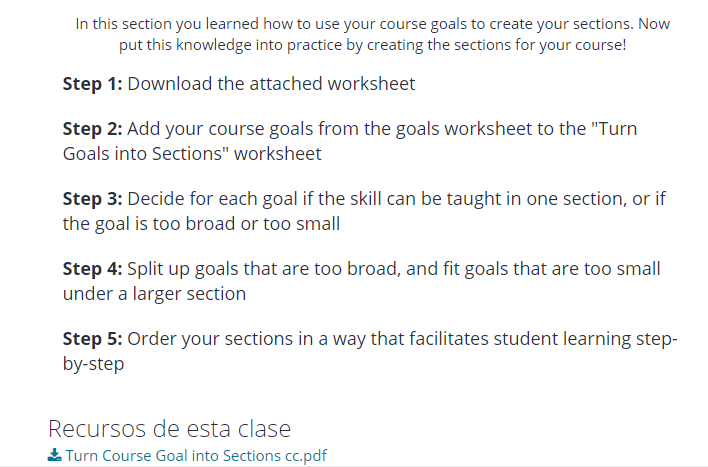
*scratch, write a complete outline for their novel before starting to write, develop compelling*

*character arcs, and confidently begin (or continue) writing their first novel.*



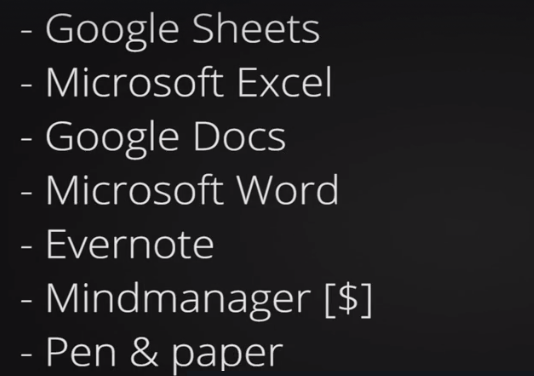
Ordenar las secciones del curso

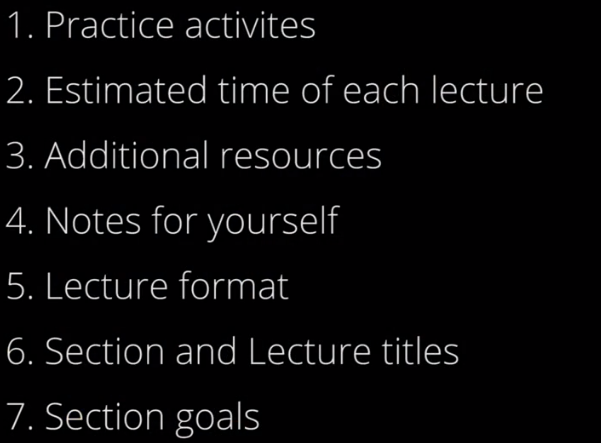


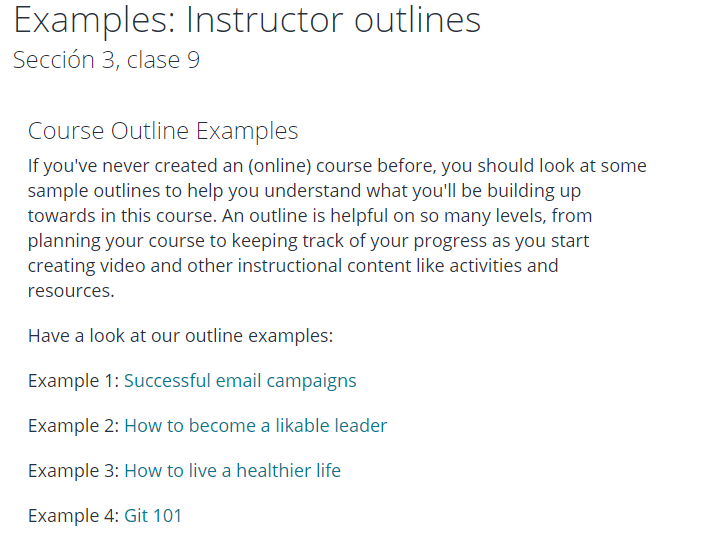


Cada sección debe tener clases y actividades (práctico)

Herramientas y modelos de guion







Activity: Create your outline template

Sección 3, clase 10

Your turn!

After reviewing different outline examples, it's now time for you to pick on an outline template that works for you. We've [created an outline template](https://docs.google.com/spreadsheets/d/1udMRK9QOOz2GzsIoIrATJ2hV70-32ehw8OdbI6lWbqY/edit?usp=sharing)that you can use, and that has been practice-tested by hundreds of instructors.

**Step 1:**Make a copy of the template [here](https://docs.google.com/spreadsheets/d/1udMRK9QOOz2GzsIoIrATJ2hV70-32ehw8OdbI6lWbqY/edit?usp=sharing), so that you can add your curriculum items as we go through the course.

**Step 2:**Add your sections and section goals into the template.

We will help you add more details to the outline as you go through this course.

**Building your own template**

Different instructors have different preferences, so if you'd rather create your own template that's fine too. When creating your template, remember that you want your outline to include the following items:

* Section and lecture titles
* Section goal
* Estimated time of each lecture
* Lecture format: You’ll need to decide if each lecture should be a talking head, article, screencast or slide presentation (you'll learn more about this in this course)
* A place to mark off which lectures are practice opportunities
* A place to mark off where you will add additional resources, like checklists, templates or pdf articles that might be beneficial to add for your students
* A place to take notes (or add the script later on)

**Step 1:** Pick a tool that works for you (e.g. Google sheets, Excel, Google Docs, Word, Evernote).

**Step 2:**Build your own template including the items above.

**Step 3:** Add your sections and section goals into the template.

We will help you add more details to the outline as you go through this course.

# Introducción Spring

Que tal empezamos un nuevo curso que vamos a dedicar a Spring Boot …y angular 4, framework desarrollado por google para la creación de Single Page Application .

## Que es Spring Boot

## 

## Entorno de dedesarrollo

## Herramientas

## Spring Initializer

# Primer Ejemplo Spring Boot

## Importar IDE

## Estructura del proyecto

## Spring Beans / Dependency Injection

# Spring MVC

## REST API Intro

## Método GET

## Método POST

## Método PUT

## Método DELETE

# Introducción Angular

Hola que tal vamos a empezar con el framework de angular 4 para la creación SPA o Simple Page Application de nuestro Frontend, este framework es creado y mantenido por google.

En este punto cabe aclarar que angular 4 es totalmente diferente a angular1 o angularJS, se puede decir que son dos frameworks totalmente diferente.

## Qué es Angular 4

## Versiones de Angular

## Herramientas

Para el desarrollo de nuestro frontend, son necesarias las siguientes herramientas:

NodeJS:

NPM: el manager para la gestión de paquetes, el cual se instala con NodeJS

TypeScript: Y para trabajar con angular necesitaremos TypeScript.

Angular CLI(Interfaz de Línea de Comando): Es una herramienta desarrollada por el equipo oficial de angular, la cual nos facilita la creación de nuestro proyecto base, podemos crear componentes.

Para instalar Angular CLI, vamos a una terminal, ejecutamos como administrador e ingresamos el siguiente comando:

npm install –g @angular/cli

La –g para que se instale de modo global

Una vez instalado ya podemos crear nuestro proyecto utilizando Angular CLI.

Lo cual estaremos viendo en el siguiente video

# Primer Ejemplo Angular

Hola que tal vamos a continuar nuestro curso. En este punto vamos crear nuestro proyecto web utilizando Angular CLI.

Para lo cual nos vamos a una terminal y ejecutamos el siguiente comando:

ng new tiendaOnlineWeb

Este comando nos creara el esqueleto de un nuevo proyecto con el nombre de tiendaOnlineWeb y descargara todos los componentes y dependencias necesarias.

Navegamos a la ubicación de donde esta nuestro proyecto, y en su interior tenemos la estructura básica de una aplicación angular 4, para comprobar su creación y que esta funcionando podemos ejecutar el siguiente comando:

ng server

Comienza a ejecutar nuestra aplicación y podemos ver que se levanta en localhost: y el puerto 4200

Abrimos su navegador de su preferencia ingresamos la dirección <http://localhos:4200>, enter y podemos ver funcionando nuestra aplicación Web con angular 4.

## Estructura del Proyecto

## Data Binding

## Qué es TypeScript

## Usando Bootstrap

# Componentes

## Crear componentes

## Crear barra de navegación

## Router – Navegación

## Directivas – lista bootstrap

## Creando interfaz para Artículo

# API REST

## Agregar Artículo

## Editar Artículo

## Eliminar Artículo

# Despedida

## Promoción

MicroProfile

GitLab

Microservicios con Spring Frameworks – Netflix OSS – Spring cloud

IT Training Course

Manu Academy

joedayz

**Spring Boot**

IDE, se puede utilizar cualquiera, porque gracias a maven o a gradle hasta podemos utilizar sublime via comandos, pero para este caso vamos a utilizar Spring Sources Tool Suite que es el IDE creado por Spring, basado en eclipse.

**JAVA**

Verificamos si tenemos instalado java 8, si no tenemos debemos descargarnos e instalar en nuestra máquina.

Para verificar si se encuentra instalado, vamos a una terminal e ingresamos el siguiente comando:

java –versión,

Verifique que se encuentre como variable de entorno JAVA\_HOME.

**MAVEN**

Así mismo verificamos si tenemos instalado maven, , vamos a una terminal e ingresamos el siguiente comando:

mvn –versión

Si no se encuentra instalado, procedemos a descargar de la página oficial de Maven, descargamos la versión.. una vez descargado, en mi caso ya lo tenía descargado, descomprimimos en C:

Creamos la variable de entorno MVN\_HOME, …… y verificamos si tenemos maven, con el comando mvn –version.

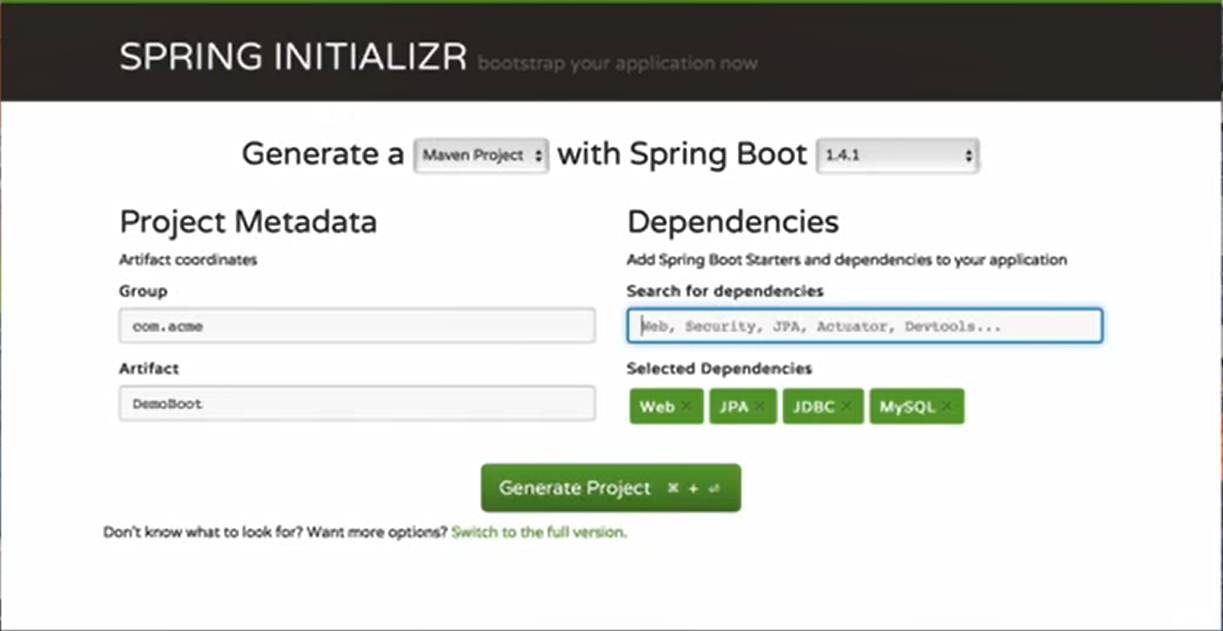
**PROYECTO BASE**

Vamos a crear un proyecto base Spring Boot, y abrirlo en in IDE, para eso nos vamos a un browser, e ingresamos la siguiente URL <http://start.spring.io>, aquí debemos responder una serie de preguntas y seleccionamos las dependencias que nuestro proyecto base va a tenga.

Buenos empecemos con Group: aquí va el grupo de tu empresa, para nuestro ejemplo vamos a poner com.online.tienda

En Artifact: ponemos el nombre de nuestro pryecto: tiendaOnline

Luego seleccionamos las dependencias para nuestro proyecto,



Y generamos nuestro proyecto, a continuación nos descarga el proyecto, nos llevamos a nuestra carpeta del Backend, descomprimimos.

**ESTRUCTURA DEL PROYECTO**

Nos extrae un proyecto base, listo para ser ejecutado.

Nuestro proyecto base, está bien estructurado, podemos ver aquí dentro de src/main/

Tenemos nuestra parte java para nuestro código java

Resources para nuestros recursos estáticos, aquí tenemos nuestro application.properties, para configurar nuestras propiedades.

Aquí tienes la carpeta de test, para nuestra pruebas

Si navegamos tenemos que nos creó una clase …..

Como seleccione maven, nos genera un archivo pom.xml, con toda la información de nuestro proyecto, aquí podemos ver que se encuentra el groupId que es el grupo que ingresamos, el artifactId de igual manera con el nombre de nuestro proyecto, aquí nos dice que vamos a construir un proyecto jar, aquí está el primer cambio de paradigma ya que no vamos a trabajar con war, se trabaja con jar lo que nos permite correrlo con java -jar

Más abajo podemos ver que hereda del proyecto boot padre, la versión que escogiste, tenemos aquí la versión de java, y más abajo están las dependencias que seleccionamos anteriormente en el navegador:

Bien entonces tenemos JPA, JDBC, Web, el driver de MySQL y para los tests, eso es todo, no necesitamos de más dependencias.

Para verificar que nuestro proyecto esta listo para ser ejecutado abrimos una terminal, navegamos hasta la ubicación de nuestro proyecto, una vez ahí ejecutamos el siguiente comando.

mvn install: este comando de maven nos compila nuestra aplicación en un archivo .jar

Ingresamos a la carpeta target y ejecutamos el comando

java –jar y el nombre de nuestro proyecto, podemos ver que nos comienza a correr nuestra aplicación, en la parte de abajo podemos ver el mensaje …….., que nos dice que nuestro aplicación se ejecutó exitosamente.

Aquí arriba podemos ver el puerto por default es 8080.

Abrimos un browser e ingresamos <http://localhost:8080>

**IDE STS**

Que tal chicos, en esta clase vamos abrir nuestro proyecto en nuestro IDE, para lo cual vamos a utilizar STS, si no tenemos, podemos ir a google y descargar STS, descomprimimos el archivo zip, ingresamos a nuestra carpeta y ejecutamos STS.exe.

Aquí nos pide la ubicación de nuestro workspace, seleccionamos y tenemos listo nuestro IDE.

Para importa nuestro proyecto, damos click derecho, damos click en importar, aquí seleccionamos Existing Mavan Projects (Proyectos Maven existentes), buscamos nuestro proyecto y Finalizar

**EXPLICAR CLASE PRINCIPAL**

Aquí tenemos la Clase TiendaOnlineApplication, la cual tiene un método principal

Para ejecutar nuestra aplicación, damos clic derecho en nuestro proyecto, damos cl……..

Para empezar a desarrollar con spring boot vamos hacer un crud de los artículos que va a tener nuestra Tienda Online y explorar los beneficios y bondades que nos brinda Spring Boot

**IMPLEMENTACION DE TIENDA ONLINE - MODELO**

Lo primero que vamos a crear es nuestro modelo.

Para eso creamos un paquete, le damos como nombre .modelo, y aceptar

Luego creamos una clase llamada Artículo. Para que nuestra clase sea un Entity Bean debemos agregarle la anotación @Entity, del paquete jacax.persistence.entity. Listo

Vamos agregar las siguientes propiedad:

* id: de tipo long, agregamos la anotación @Id para decir que es nuestro Id.

Agregamos la anotación @GenerateValue(strategy=GenerationType.AUTO), para la generación automática de nuestro id.

* nombre: de tipo String
* descripcion: de tipo String
* precio: de tipo double
* descuento: de tipo int

Adicionalmente creamos nuestro GET y SET y listo tenemos nuestro modelo Articulo que haría referencia a una tabla llamada ARTICULO, con las siguientes columnas: ID, NOMBRE, DESCRIPCION, PRECIO, DESCUENTO.

**IMPLEMENTACION DE TIENDA ONLINE – REPOSITORY**

Vamos a utilizar JPA

Para crear nuestro repository creamos un nuevo paquete con nombre: repository, y aceptar.

Luego creamos una interfaz llamada ArticuloRepository, extendemos de CrudRepository para la entidad Artículo, con primary key de tipo long, listo y este sería mi repositorio, nada más porque estamos utilizando Spring Data y este va a injectar una implementación que él va a proveer en base a esta estructura (Abrimos la interfaz CrudRepository) en tiempo de ejecución.

Aquí podemos declarar nuestros propios método estos debe empezar con findBy y el nombre del atributo, este atributo debe coincidir con alguno que tengamos en nuestra clase Articulo por ejemplo

findbyTitulo(String titulo), nosotros solo hacemos la declaración del método en nuestra interfaz, el framework de spring boot se encarga de la implementación.

**IMPLEMENTACION DE TIENDA ONLINE – SERVICE**

Para crear nuestro servicio creamos un nuevo paquete con nombre: service, y aceptar.

Creamos primeramente una interfaz llamada ArtituloService, la cual va implementar los siguientes métodos:

Aticulo findById(long id), debemos importar nuestra clase Articulo

Aticulo findByTitulo(String Titulo)

void saveArticulo(Artitulo articulo)

void updateArticulo(Artitulo articulo)

void deleteArticulo(long id)

Lis<Articulo> findAllArticulos()

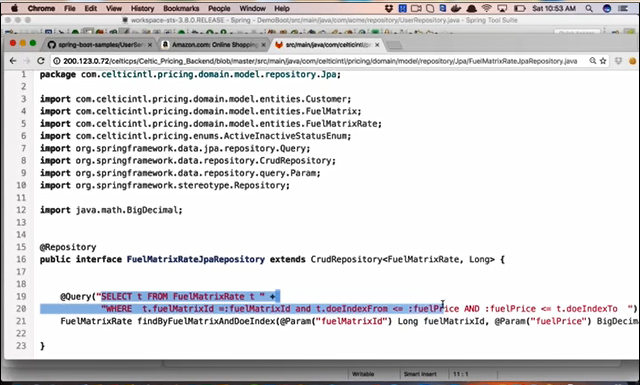
void boolean isArticuloExist(Articulo articulo)

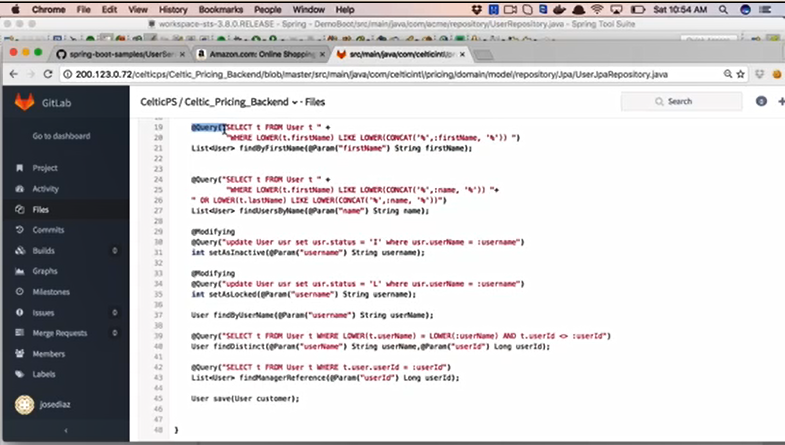
Creamos nueva clase llamada ArtituloServiceImpl en el paquete impl, esta clases va a implementar nuestra interfaz AticuloService, listo, y tendríamos algo así, pero a esta clase la debemos decorar con la anotación @Service, lo que le dice a nuestra clase que es un servicio.

Aquí debemos Injectar nuestro Reporitorio, para acceder a los datos,

Entonces vamos a crear un atributo de tipo ArticuloRepository, le vamos a llamar igual articuloRepository, y para poder injectar utilizamos la anotación @Autowired, esta anotación lo que hace es …….

Y comenzamos a realizar la implementación de cada uno de nuestros métodos **se hace en el video**





**IMPLEMENTACION DE TIENDA ONLINE – SERVICE REST – GET - ALL**

Para crear nuestro servicio REST creamos un nuevo paquete con nombre: rest, y aceptar.

Creamos una clase llamada ArtituloRestController, listo, esta clase debe ser anotada con @RestController. --- esta anotación ha sido creada para spring boot exclusivo para REST, es decir podemos exponer y consumir JSON

Injectamos nuestro servicio ArticuloService, utilizamos la anotación @Autowired, aquí injectamos la interfaz no la implementación, el famework ya se encarga de buscar su implementación.

Aquí le decimos impleméntame el Servicio y el servicio implementa el repositorio, aquí es donde tenemos una arquitectura basada en capas.

Agremos un servicio que nos recupere todo los artículos.

Para eso nos creamos un método, public REsponseEntity<List<Articulo>> getAllArticulos()

Dentro creamos una lista de artículos llamamos articulos, le decimos igual al método articuloService.findAllArticulos()

Hacemos una validación en el caso que venga con null la respuesta, if(artículos .isEmpty() 🡪 retornamos new ResponseEntity<List<Articulo>>(HttpStatus.NO\_CONTENT);

Caso contrario si nuestra lista no es vacia retornamos new ResponseEntity<List<Articulo>>(artículos, HttpStatus.OK);

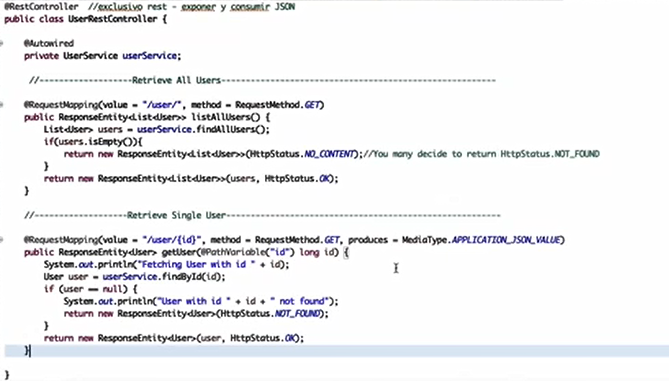
A este método debemos agregarle la anotación @ResquestMapping() dentro le damos un value = “/articulos”, method = ResquestMethod.GET

ResponseEntity que remplaza a ResponseBody, y es un objeto, pero esta entidad va a tener dos datos, lo que deseamos devolver y un status.

Spring Boot utiliza Jackson para convertir nuestra lista de artículos a JSON

**IMPLEMENTACION DE TIENDA ONLINE – SERVICE REST – GET - ID**

Agregamos un nuevo método que nos permite obtener un solo artículo dado su ID



{id} viene en la URL, lo tengo y lo asocio inmediatamente al método gracias a la anotación @PathVariable con el mismo nombre y lo asigno a id de tipo long

Si es nulo es .BOT\_FOUND 🡪 no encontré ese registro, pero si lo encontré le digo toma mi objeto usuario y toma tu código de status.

**IMPLEMENTACION DE TIENDA ONLINE – SERVICE REST – POST**

Agregamos un nuevo método que nos permite crear un artículo.



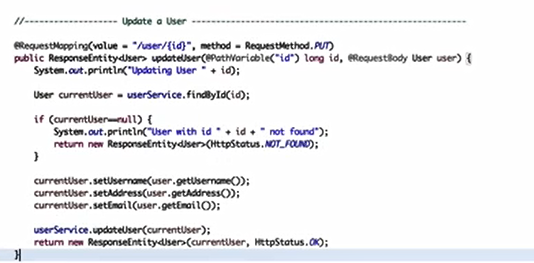
1:57

Para crear un artículo utilizamos el método POST, aquí nos dice que vendar un JSON y se depositara en la anotación @RequestBody Articulo articulo, aquí recibe ese JSON y lo mapea a nuestro objeto Relacionado Articulo

Si existe nuestro articulo debemos retornar un startus de conflicto – 409

**IMPLEMENTACION DE TIENDA ONLINE – SERVICE REST – PUT**

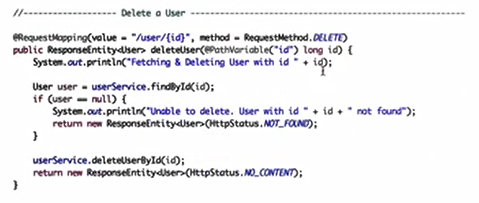
Agregamos un nuevo método que nos permite actualizar un artículo.



Al implementar hacemos la descripción

**IMPLEMENTACION DE TIENDA ONLINE – SERVICE REST – DELETE**

Agregamos un nuevo método que nos permite eliminar un artículo.



**Esto está en el primer video de josediaz**