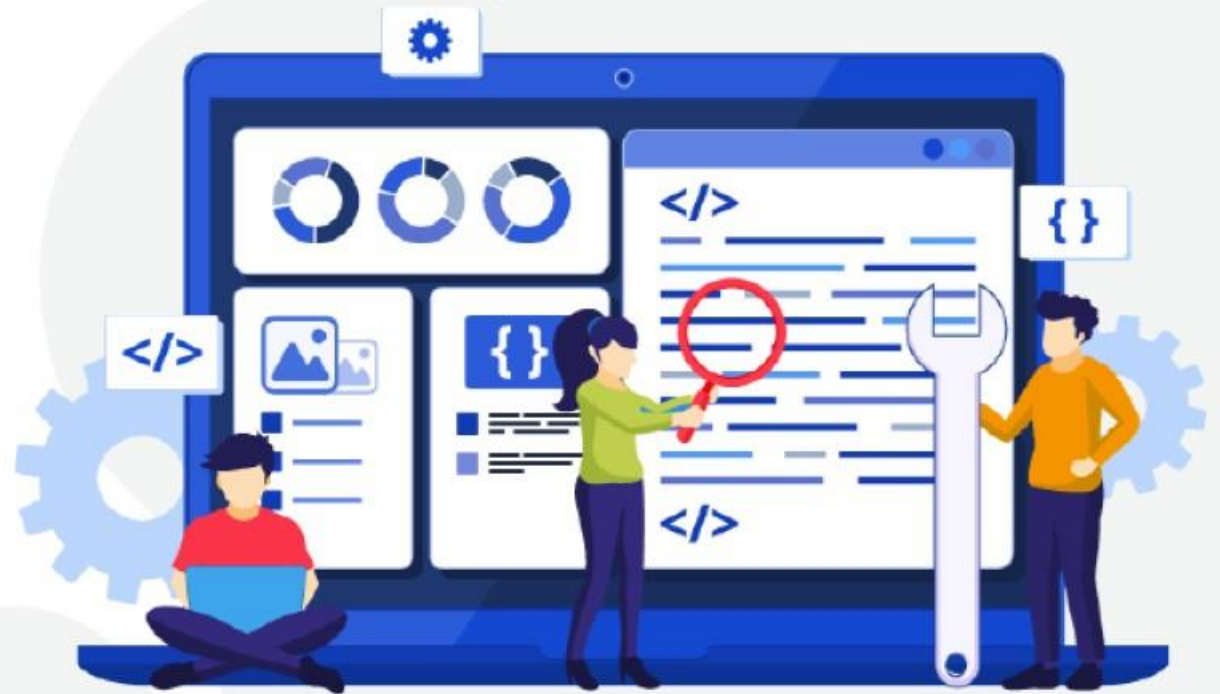




CICLO 4a

[FORMACIÓN POR CICLOS]
Desarrollo de
APLICACIONES WEB

Introducción a
microservicios
con Spring Boot



**UNIVERSIDAD
DE ANTIOQUIA**

Facultad de Ingeniería

Introducción a microservicios con Spring Boot

Diego Iván Oliveros Acosta @scalapp.co



Microservicios

- Los microservicios son un sistema de desarrollo de software que ha venido creciendo en popularidad en los últimos años, influyendo de manera positiva en aspectos como el tiempo, el rendimiento y la estabilidad de los proyectos.
- Los microservicios proponen su propia arquitectura permitiendo funcionar como un conjunto de pequeños servicios que se ejecutan de manera independiente y autónoma, creados con diferentes lenguajes de programación.

Microservicios

Para aplicar una arquitectura orientada a microservicios sobre soluciones IT ya desarrolladas, se propone interactuar con dos capas:

- Una capa que actúa de manera interna conteniendo los componentes del microservicio puro con acciones complejas menores.
- Una capa externa alrededor que se construye del microservicio y que contendrá las capacidades requeridas de los servicios expuestos en estas arquitecturas.



Microservicios

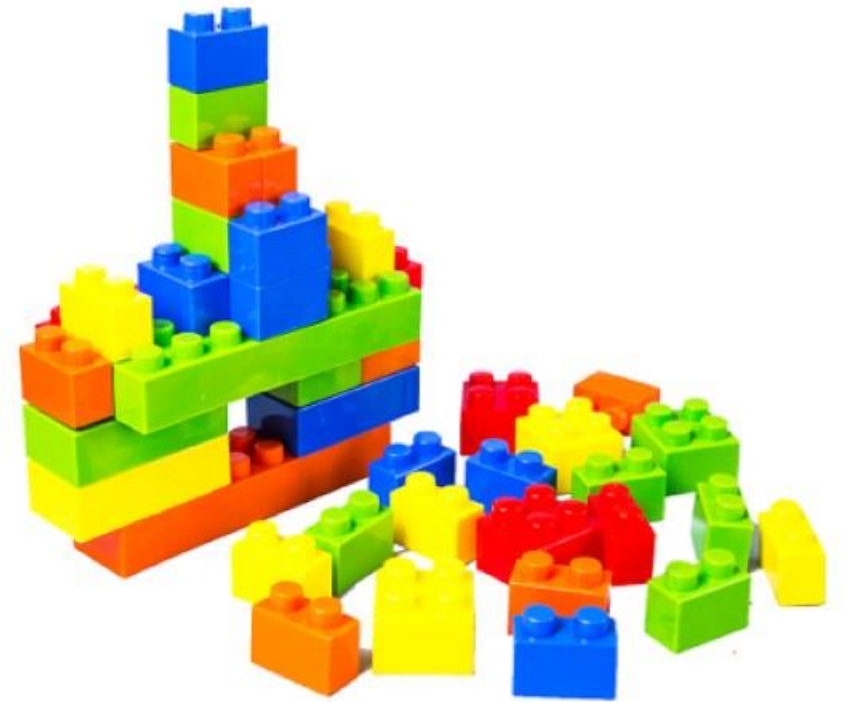
Los microservicios se comunican entre sí a través de peticiones HTTP hacia sus API.

En arquitecturas de microservicios debe haber un grupo mínimo de microservicios que gestionen elementos comunes.

Uno de los puntos fuertes de este tipo de servicios es su **escalabilidad**.

Microservicios

- Pueden ser desplegados y modificados sin afectar otros aspectos funcionales de la aplicación en general del proyecto o producto.
- Cada microservicio es independiente porque posee su propia base de datos, evitando así la sobrecarga y el encolamiento de solicitudes (*request*) desde el cliente.



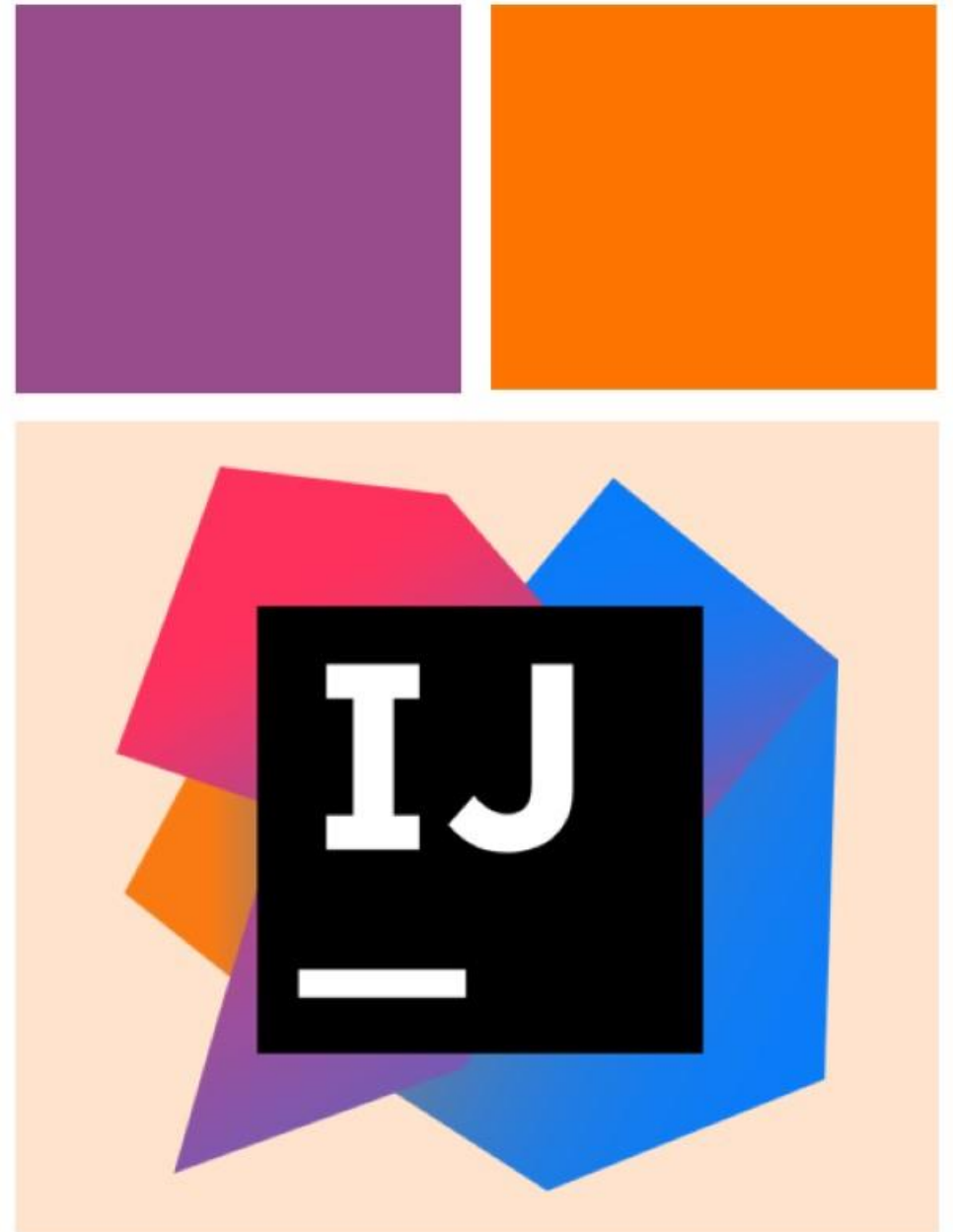
Microservicios

- Las ventajas de los microservicios son:
 - Equipo de trabajo mínimo
 - Escalabilidad
 - Funcionalidad modular, módulos independientes
 - Libertad del desarrollador de desarrollar y despliegue de servicios de manera independiente
 - Uso de contenedores que permiten rápidamente el despliegue y el desarrollo de la aplicación

Download IntelliJ IDEA

- Windows /macOS/ Linux
- Ultimate: For web and enterprise development (Free 30-day trial)
- Community: For JVM and Android development
- **Learning or teaching programming:** IntelliJ IDEA Edu with special support for learners and educators

<https://www.youtube.com/user/intellijideavideo>



Microservicios con Spring Boot

- Cuando pretendemos crear microservicios sobre un IDE en particular, como por ejemplo IntelliJ IDE, **requerimos un plug-in** para tener un asistente.
- El **plug-in** que utilizaremos será “**Spring Assistant**”, el cual podemos referenciar para instalar directamente desde el **IDE** mediante la opción:

File/Settings/Plugins

- y le damos la opción de “Install”.



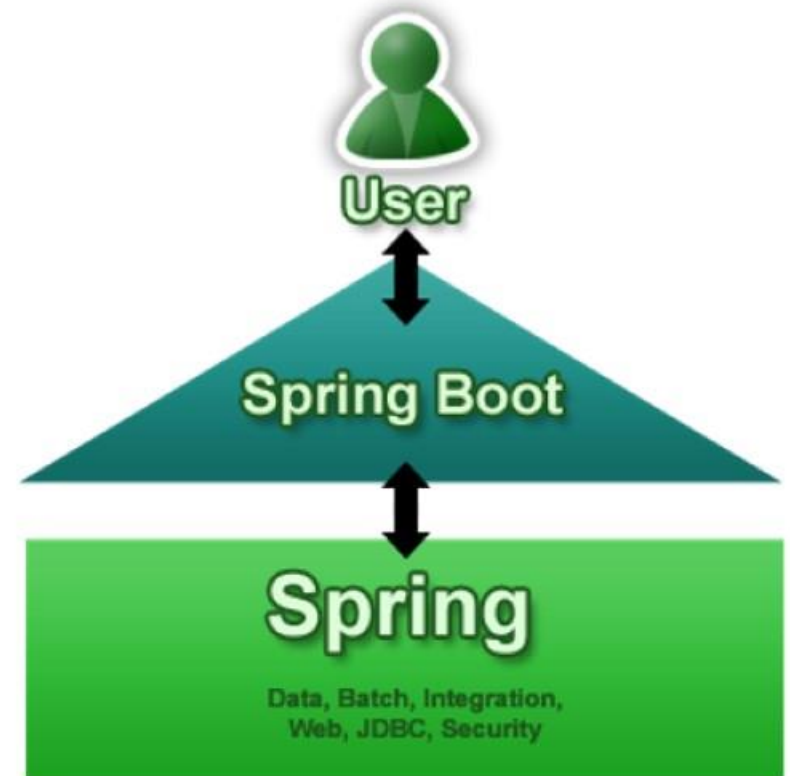
Spring Boot®

<https://spring.io/projects/spring-boot>

<https://adoptium.net/?variant=openjdk11>

Microservicios con Spring Boot

- **El objetivo de Spring Boot:** es proporcionar un conjunto de herramientas para construir de manera rápida y ágil aplicaciones Spring, fáciles de crear, configurar y mantener en el tiempo.
- **Spring Boot:** facilita la creación de aplicaciones basadas en **Spring**, autónomas y del nivel de producción para ejecutar.
- **Poca configuración:** Básicamente es posible poner en funcionamiento una aplicación de Spring con muy poco trabajo





```
gradle myTask

$ gradle :mySubproject:taskName
$ gradle mySubproject:taskName

$ gradle test
gradle test deploy //Ejecutando multiples tareas
gradle dist --exclude-task test gradle dist -x test
gradle test --continue
gradle build //assembling all outputs and running all checks
gradle tasks --all
gradle myTask --scan
```

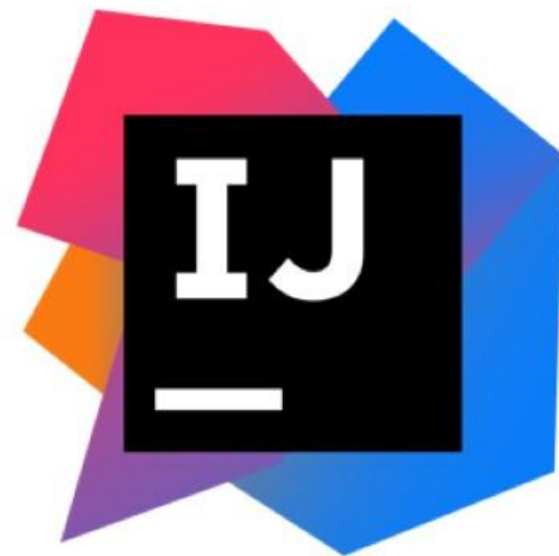
Accelerate developer productivity

<https://gradle.org/>

Microservicios con Spring Boot

Requisitos previos

- Para sacar el máximo provecho de la información de este curso debemos estar familiarizados y tener instalado lo siguiente:
 - Java Development Kit 1.8 (JDK)
 - IntelliJ IDEA para 2020.2 o superior
 - Gradle 6.0 o superior



Gradle

Microservicios con Spring Boot

- Para instalar el **Plug-in de “Spring Assistant”** sobre el IDE de IntelliJ IDEA diríjase al siguiente link:
<https://plugins.jetbrains.com/plugin/10229-spring-assistant/versions>
- Presione el botón “Get” y ajuste la compatibilidad según su versión de IntelliJ IDEA y presione “Download” en la última versión o la que prefiera.
- Con esto tendrá un archivo descargado con la extensión .zip que contendrá el plugin.

Framework integration

Spring Assistant



Development Team @ 1Ton Technologies

Microservicios con Spring Boot

JET
BRAINS

Marketplace

Edu Courses

Themes

Plugin Ideas

Build Plugins

Sign In



Framework integration

Spring Assistant



Development Team © 1Ton Technologies

Overview

Versions

Reviews

Get

Compatible with IntelliJ IDEA (Ultimate, Community, Educational), Android Studio

Version History

Android Studio

IntelliJ IDEA Community

IntelliJ IDEA Educational

IntelliJ IDEA Ultimate

Compatibility with [IntelliJ IDEA Community](#)

Stable

EAP

Nightly

Version

Compatibility with [IntelliJ IDEA Ultimate](#)

Update Date

2018

0.12.0

2017.2 — 2019.3.5

Apr 11, 2018

Download

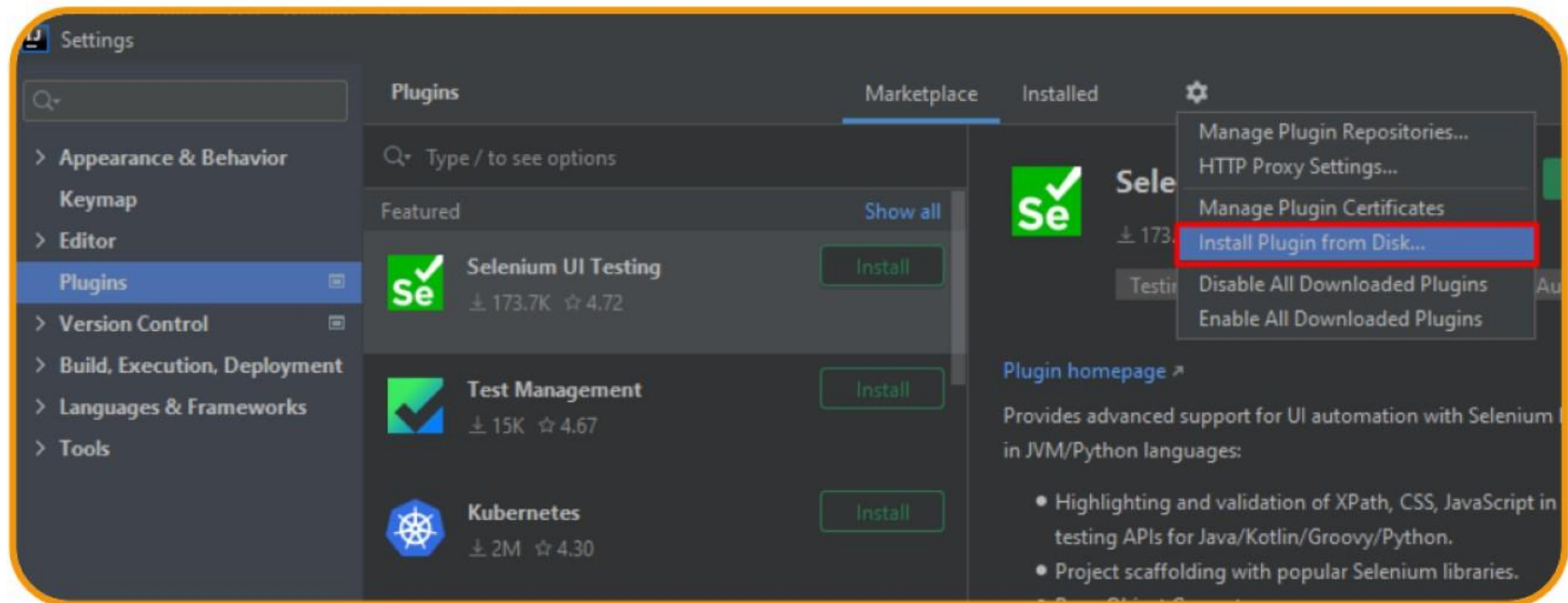
0.11.0

To have full functionality you have to accept [Plugin Marketplace Agreement](#).



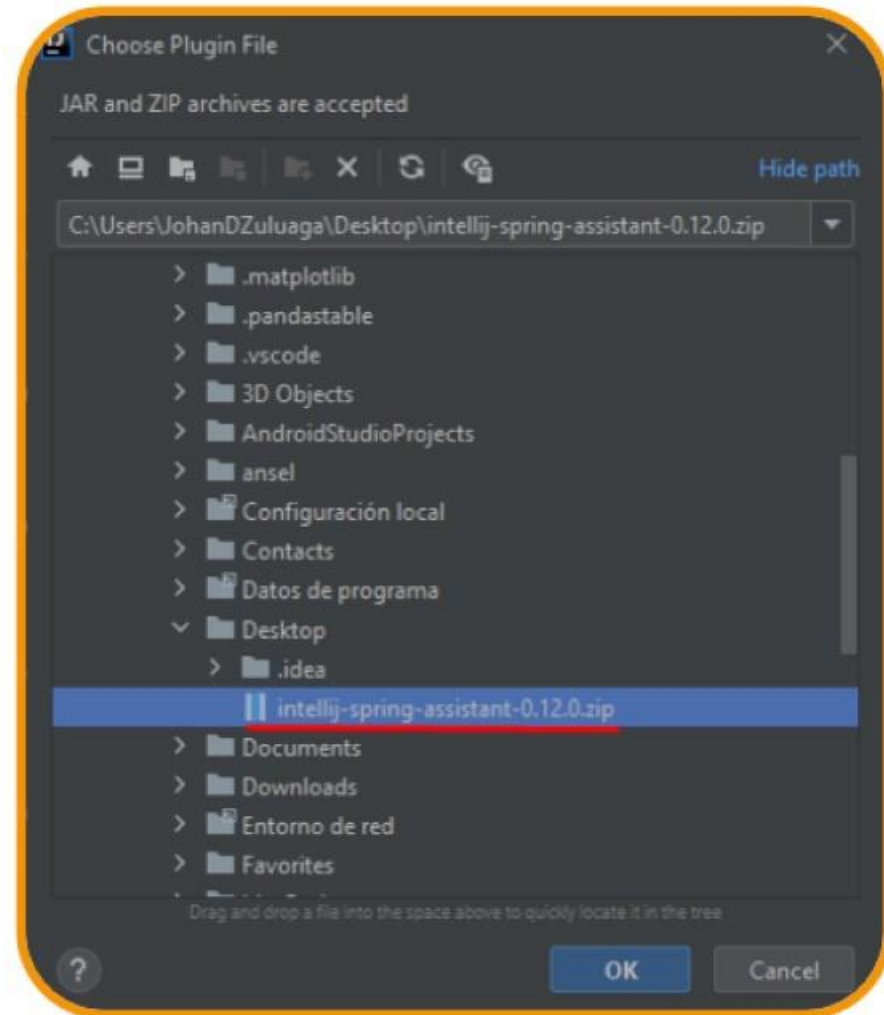
Microservicios con Spring Boot

- Diríjase a IntelliJ y presione las teclas Ctrl + Alt + s, luego presione el botón del engranaje y la opción "Install Plugin from Disk..."



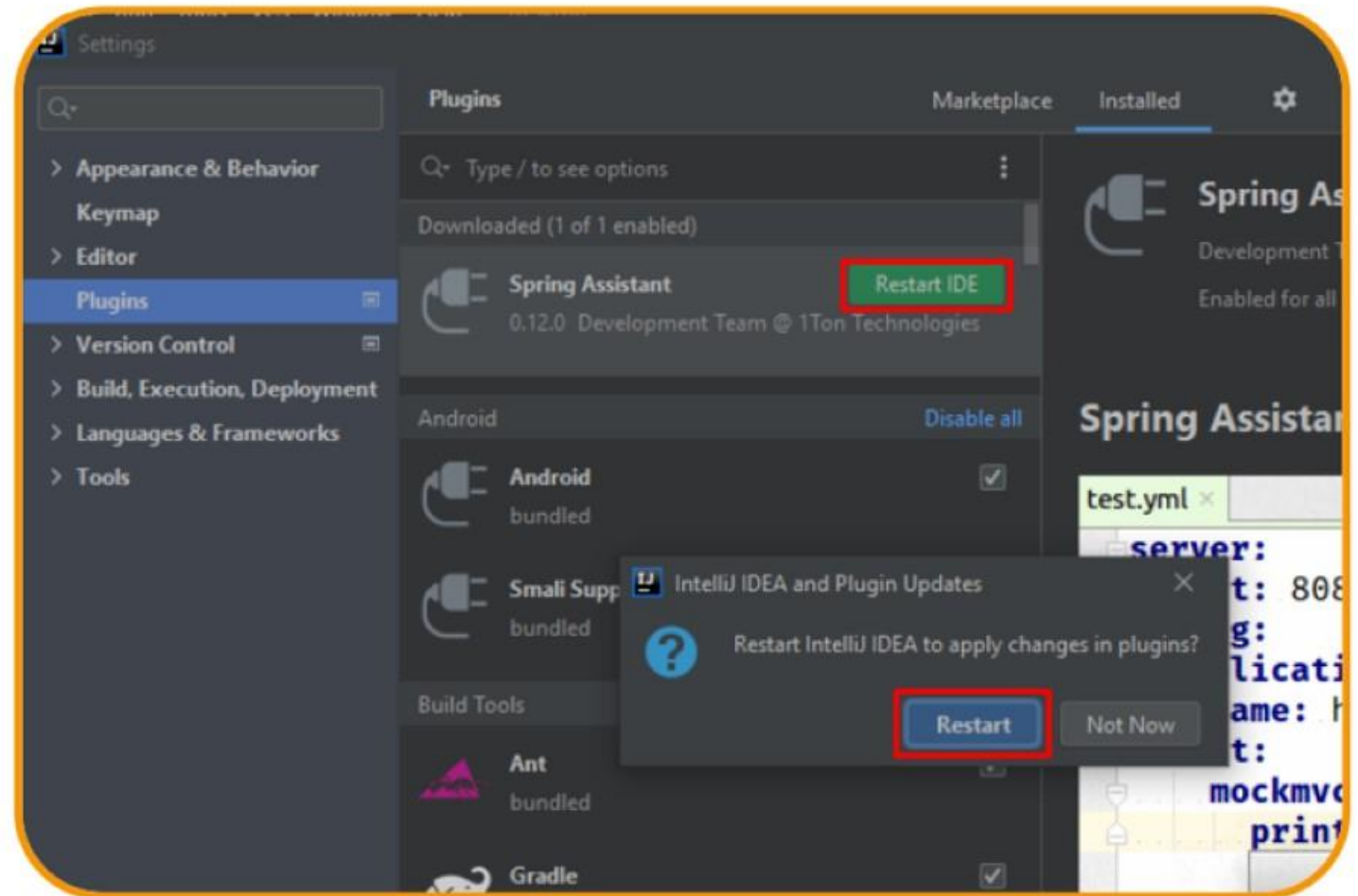
Microservicios con Spring Boot

- Busque el archivo descargado y presione “OK”



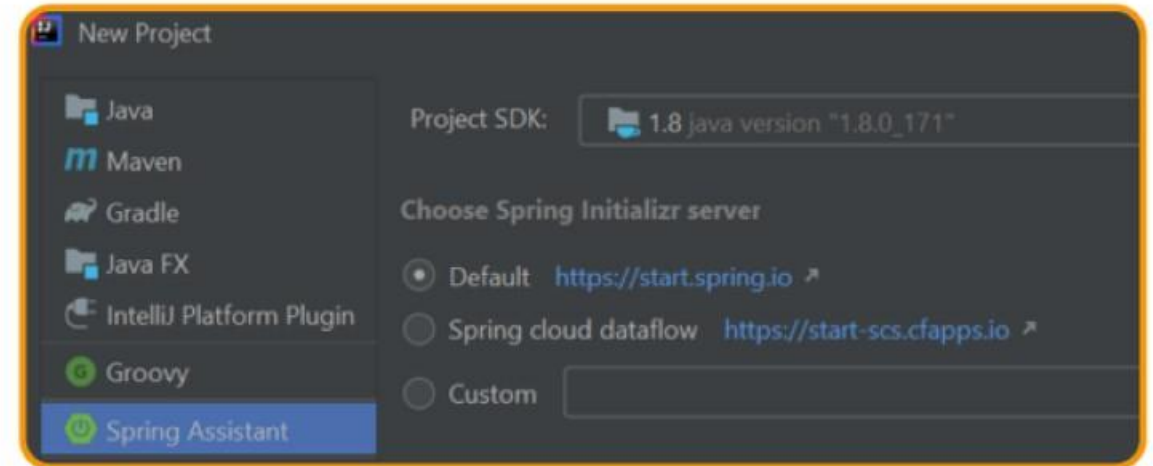
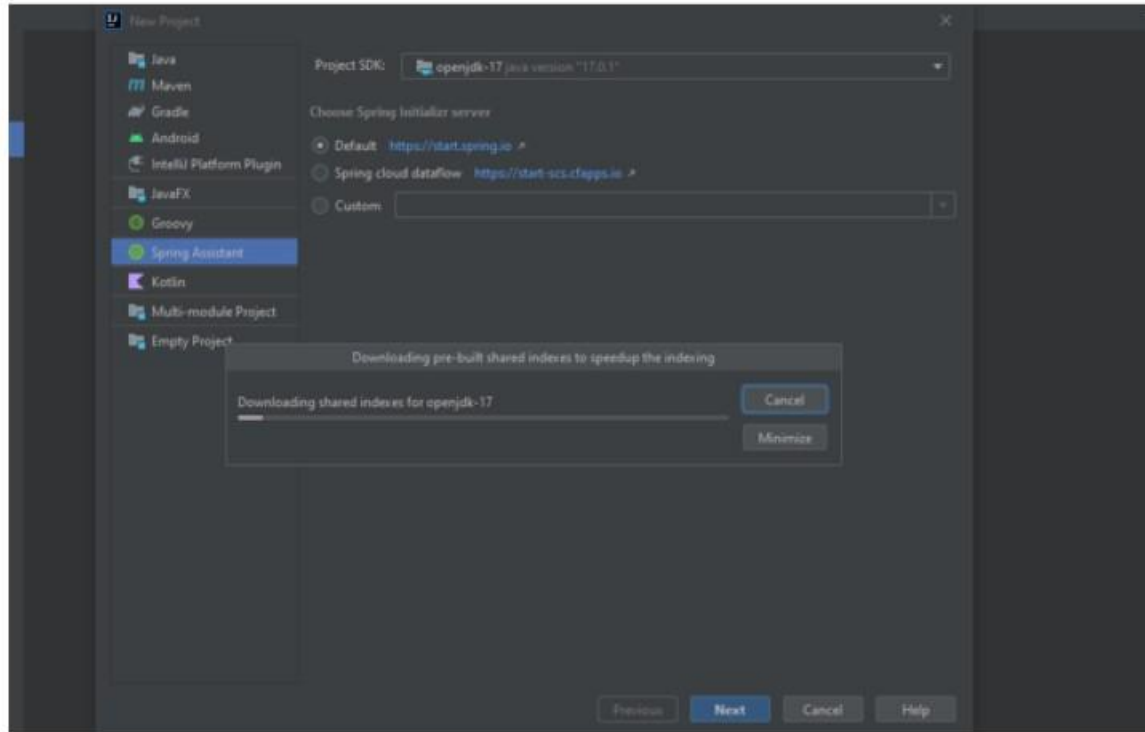
Microservicios con Spring Boot

- Presione “Restart IDE” y luego “Restart”



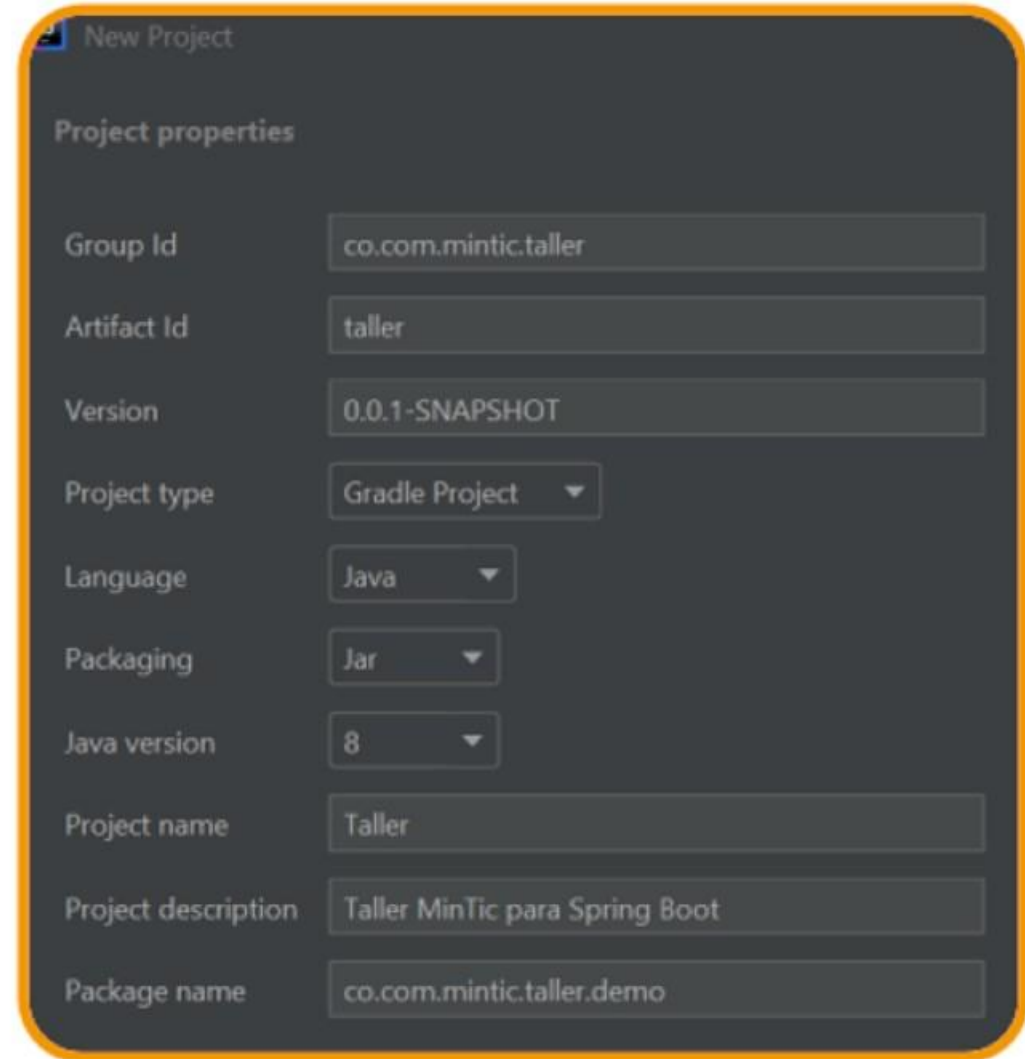
Microservicios con Spring Boot

- Para crear un nuevo proyecto de Spring Boot sobre el IDE de IntelliJ seleccionamos la opción File/New/ Project. Seleccionamos luego “Spring Assistant” y después “Next”:



Microservicios con Spring Boot

- Llenamos las propiedades del nuevo proyecto tal como se requiere para un taller de ejemplo y procedemos a dar "Next":

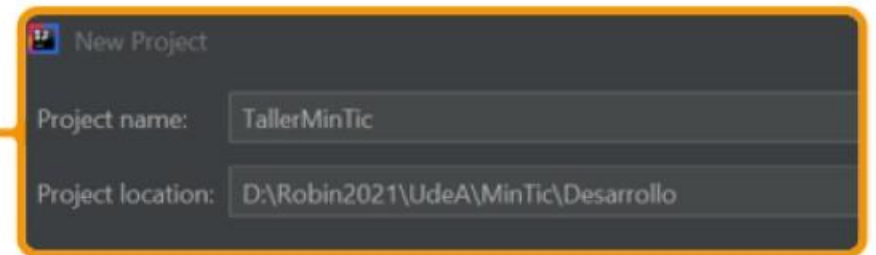
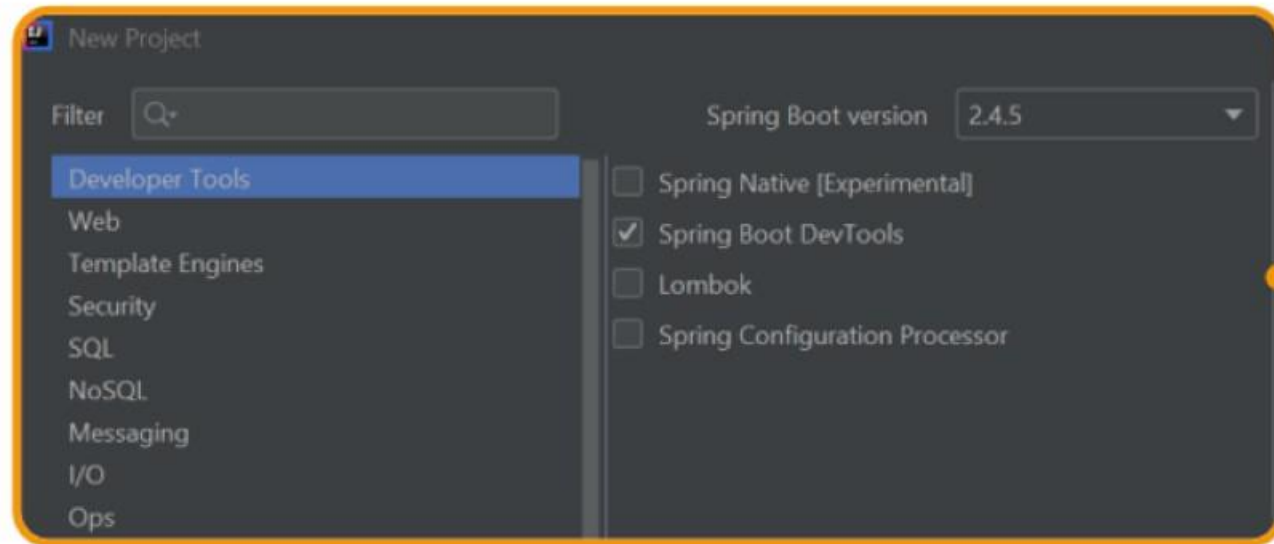


The screenshot shows the 'New Project' dialog box with the following fields and values:

Project properties	
Group Id	co.com.mintic.taller
Artifact Id	taller
Version	0.0.1-SNAPSHOT
Project type	Gradle Project
Language	Java
Packaging	Jar
Java version	8
Project name	Taller
Project description	Taller MinTic para Spring Boot
Package name	co.com.mintic.taller.demo

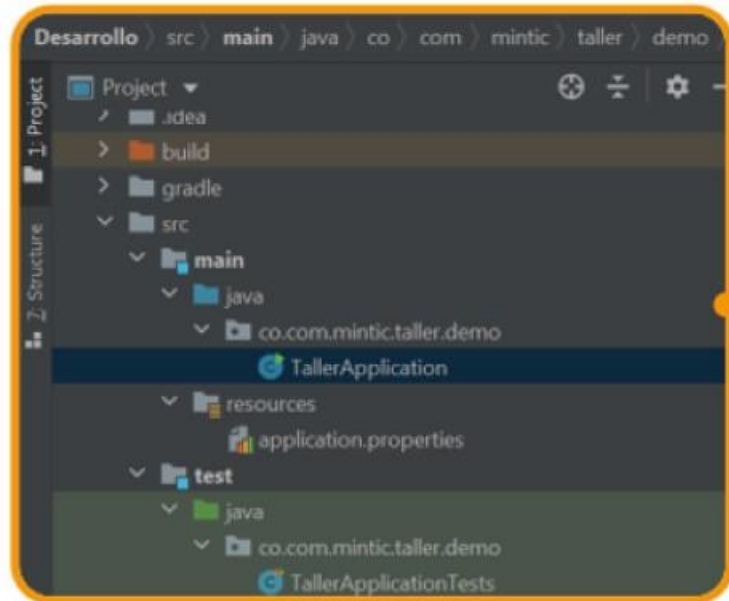
Microservicios con Spring Boot

- Seleccionamos la opción de herramientas de desarrollo “Spring Boot DevTools” y luego continuamos con “Next”. Por último, finalizamos el asistente poniendo el nombre del nuevo proyecto en la ruta en que quedará:



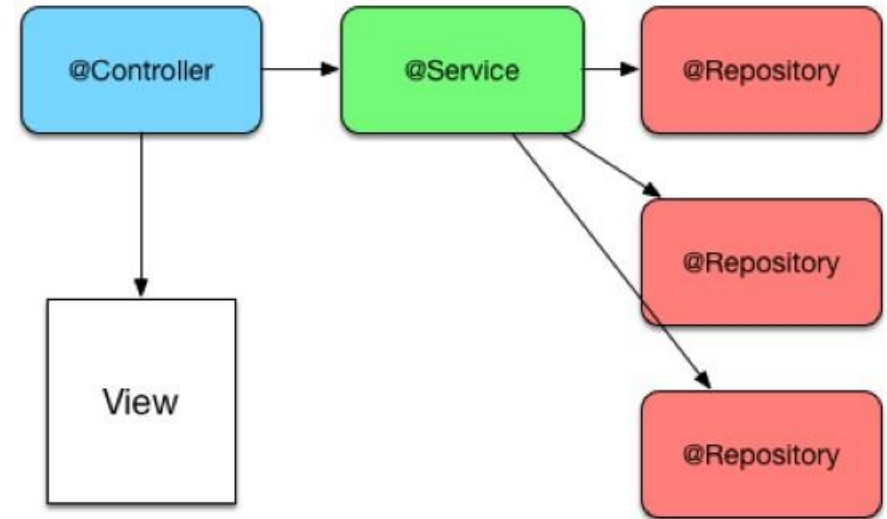
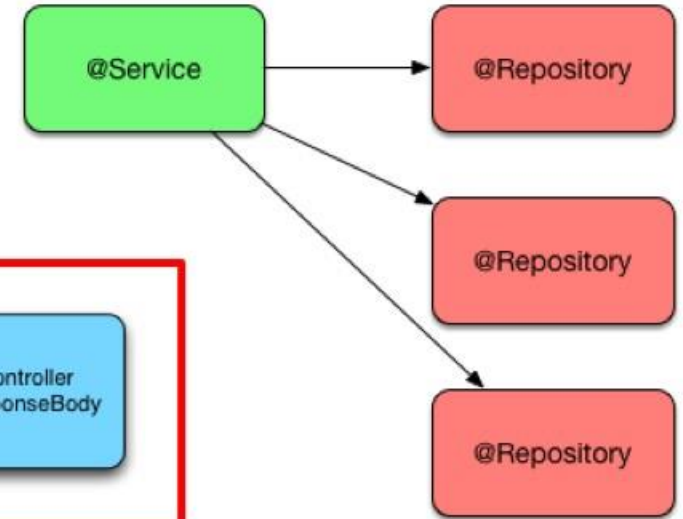
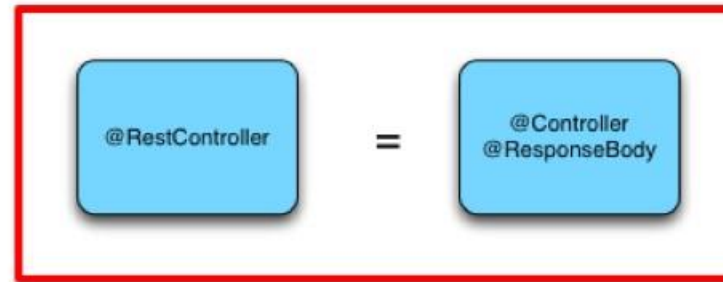
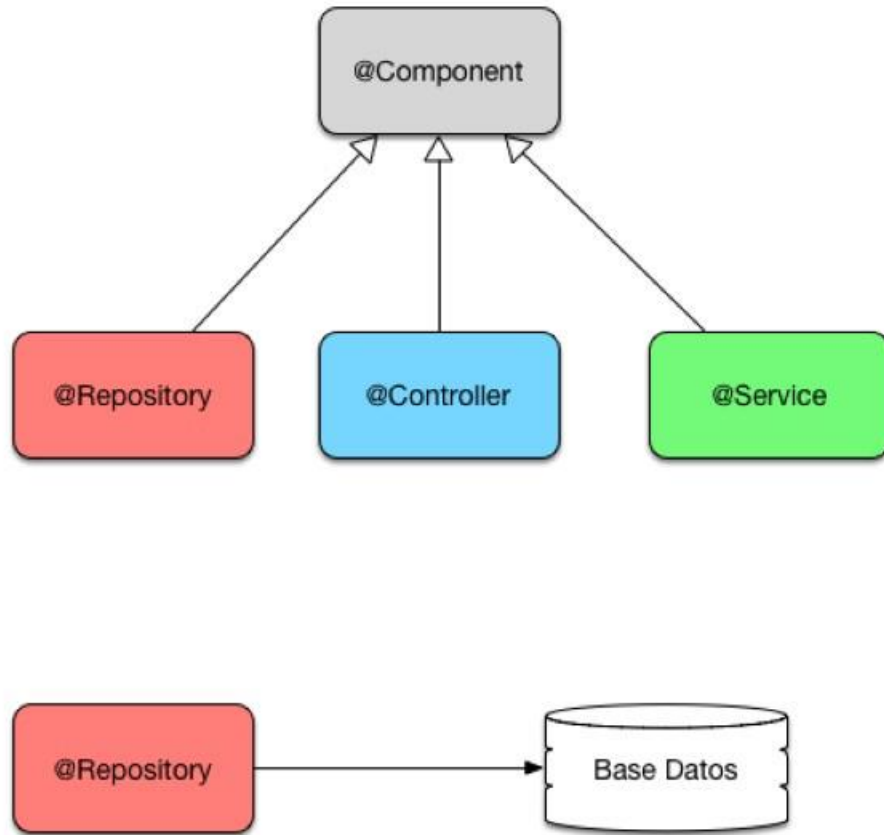
Microservicios con Spring Boot

Una vez finalizado el paso a paso de “Spring Assistant”, abrimos el nuevo proyecto para verificar el arquetipo inicial para la construcción de los microservicios basados en el framework de Spring Boot y revisamos las librerías de Spring Boot en el archivo build.gradle:



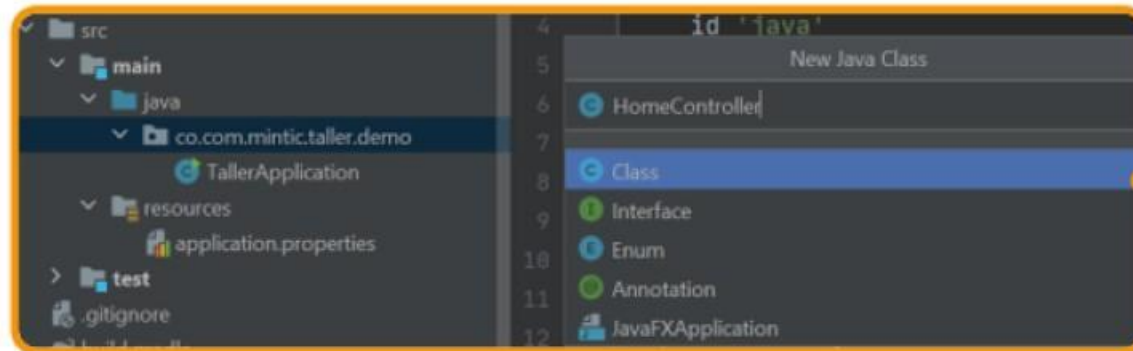
```
dependencies {  
    implementation 'org.springframework.boot:spring-boot-starter-web-services'  
    implementation 'org.springframework.boot:spring-boot-starter-webflux'  
  
    implementation group: 'org.springframework.boot', name: 'spring-boot-starter-parent', version: '2.1.6.RELEASE'  
    implementation group: 'org.springframework.boot', name: 'spring-boot-starter-web', version: '2.1.6.RELEASE'  
  
    developmentOnly 'org.springframework.boot:spring-boot-devtools'  
    testImplementation 'org.springframework.boot:spring-boot-starter-test'  
    testImplementation 'io.projectreactor:reactor-test'  
}
```

Spring estereotipos y anotaciones



Microservicios con Spring Boot

- Creamos un controlador utilizando las anotaciones básicas en una nueva clase, denominada HomeController:
- @RestController
- @GetMapping



```
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class HomeController {
    @GetMapping("/")
    public String homePage(){
        return "Curso Spring Boot para MinTic";
    }
}
```

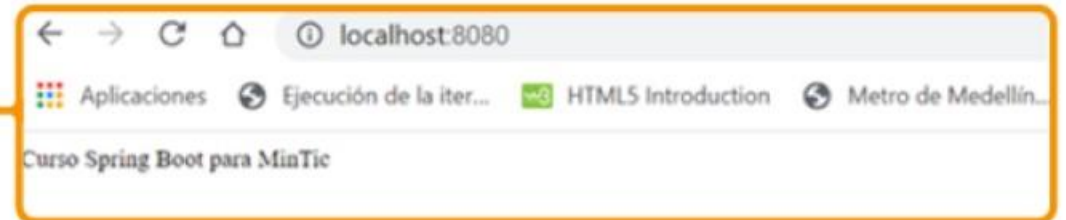
Microservicios con Spring Boot

- Ejecutamos la aplicación desde la clase TallerApplication.java para enviar el mensaje desde la clase controladora hacia el browser y ver el mensaje. Así podemos ver que se encuentra adecuadamente el servidor web de Spring Boot:

```
@SpringBootApplication
public class TallerApplication {

    public static void main(String[] args) {
        SpringApplication.run(TallerApplication.class, args);
    }

}
```



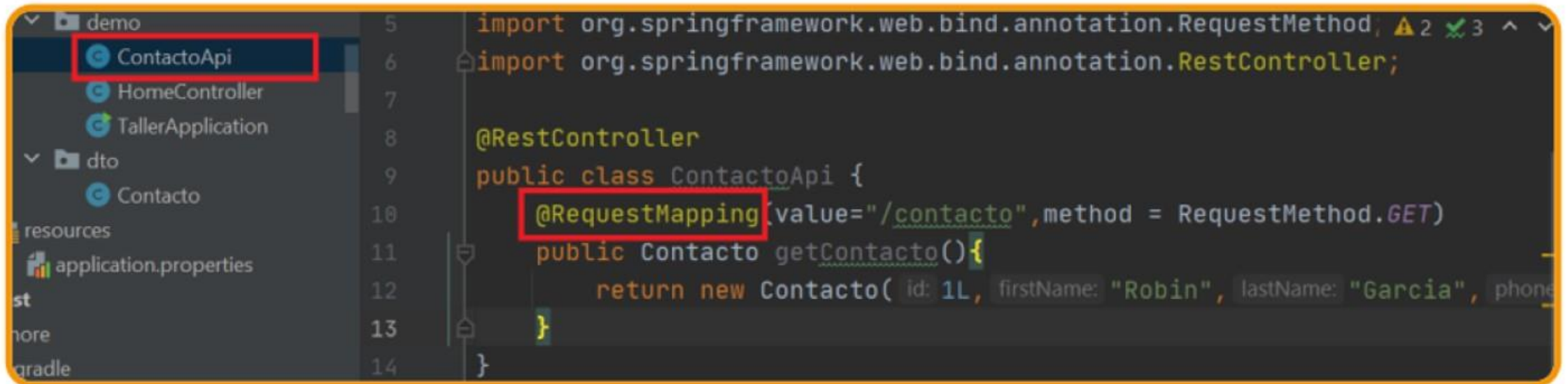
Microservicios con Spring Boot

- Ahora creamos su primer servicio REST. Para ello creamos un paquete llamado “dto” y una clase llamada “Contacto”, y ponemos las propiedades del contacto y un constructor de la clase:



Microservicios con Spring Boot

- Creamos una clase denominada ContactoApi para retornar la información del contacto en formato JSON sobre el browser.
- Utilizamos la anotación `@RequestMapping`, que permiten tener un microservicio básico y funcional con un método que retorne una variable del tipo Contacto con la información configurada por defecto:

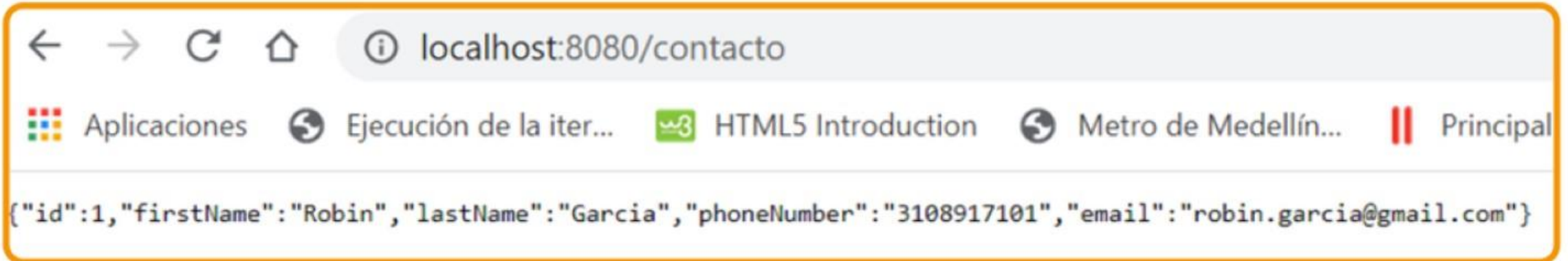


The screenshot shows an IDE with a project named 'demo'. In the left sidebar, the 'demo' folder is expanded, showing 'ContactoApi' (highlighted with a red box), 'HomeController', and 'TallerApplication'. Below it, the 'dto' folder contains 'Contacto'. The main editor displays the code for 'ContactoApi.java'.

```
5 import org.springframework.web.bind.annotation.RequestMethod;
6 import org.springframework.web.bind.annotation.RestController;
7
8 @RestController
9 public class ContactoApi {
10     @RequestMapping(value="/contacto", method = RequestMethod.GET)
11     public Contacto getContacto(){
12         return new Contacto( id: 1L, firstName: "Robin", lastName: "Garcia", phone
13     }
14 }
```

Microservicios con Spring Boot

- Ejecutamos su aplicación construida con el arquetipo de Spring Boot y vamos a la URL local para observar la información del contacto que retornamos en el método `getContacto` de la clase `ContactoApi` en formato JSON sobre el browser:



Referencias:

- <https://plugins.jetbrains.com/plugin/10229-spring-assistant/versions/nightly>

Ciclo4 misión tic 2022 Introducción a microservicios con Spring Boot

- https://lms.misiontic2022udea.com/pluginfile.php/81115/mod_resource/content/9/2021_000127_DW_Semana5_MinTic_Microservicios%20Spring%20Boot_VFfinal.pdf
- www.scalapp.co
- <https://escuelafullstack.com/slides/curso-de-microservicios-con-spring-boot-13>
- <https://www.oracle.com/java/technologies/downloads/#jdk17-windows>
- <https://www.jetbrains.com/idea/download/other.html>

Referencias:

- 5 Trucos del IntelliJ IDEA

<https://www.youtube.com/watch?v=DOTL82UiQjo>

- IntelliJ IDEA | Full Course | 2020

<https://www.youtube.com/watch?v=yefmcX57Eyg>

https://www.jetbrains.com/idea/features/editions_comparison_matrix.html

<https://docs.oracle.com/en/java/javase/17/index.html>

<https://gradle.org/releases/>

<https://gradle.org/guides/>

https://gradle.com/training/?_ga=2.146957936.690131247.1638469529-173266704.1638469529

<https://gradle.org/release-candidate/>