



El futuro digital
es de todos

MinTIC



Ciclo 4a:

Desarrollo de aplicaciones web



Misión
TIC 2022

VERSIÓN 1.0

Unidad de educación
continua y permanente
Facultad de Ingeniería



Unidad Camilo Torres
Calle 44 # 45-67
Bloque B5 piso 1



(57) + 316 5000
uec_fibog@unad.edu.co

Actividad Práctica

(Microservicio AccountMS - Parte 1)

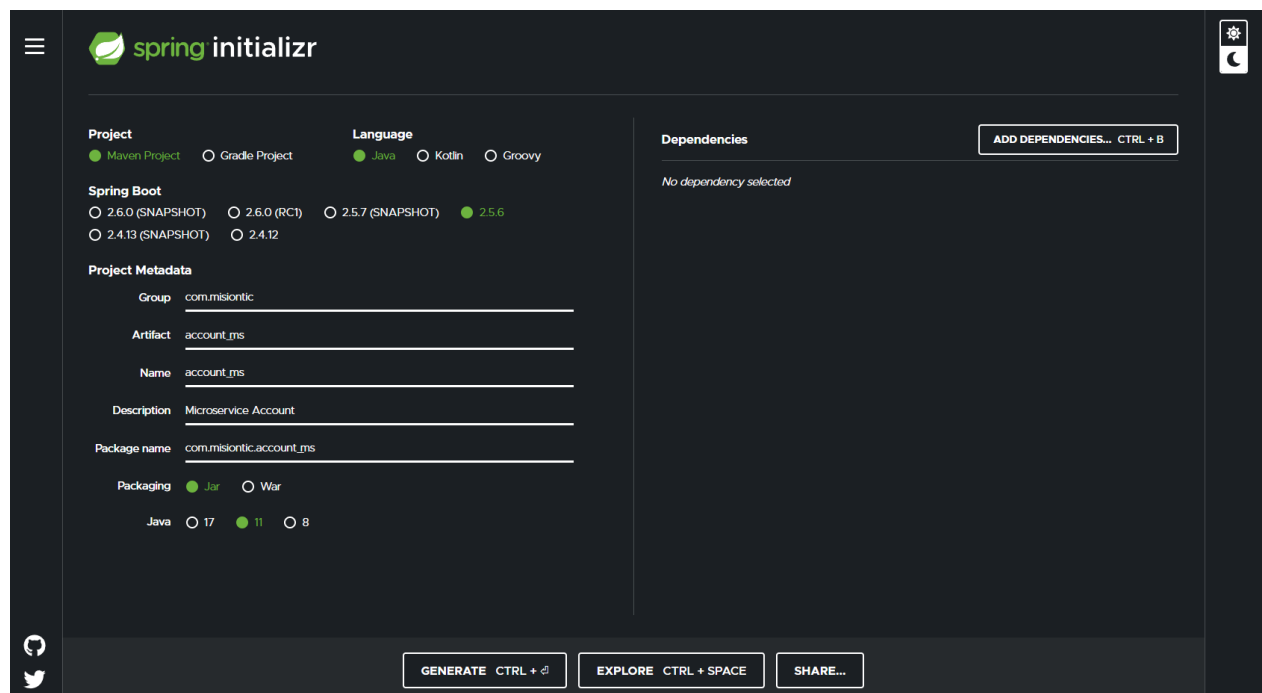
Inicialización de Spring Boot

Spring Boot, al igual que la mayoría de **frameworks**, posee una **herramienta** que permite **inicializar** un **proyecto o espacio de trabajo** (archivos y directorios) sobre el cual se **comienza a trabajar**. Este **espacio de trabajo** es **bastante complejo** y entenderlo por completo puede resultar una tarea bastante tediosa. Sin embargo, para iniciar con el uso de Spring Boot, no es necesario conocer en profundidad cada uno de los aspectos del espacio de trabajo.

A continuación, se creará un **espacio de trabajo**, se explicará brevemente su **estructura**, se establecerá la **conexión** con base de datos de **MongoDB** creada previamente, y, por último, se **ejecutará el espacio de trabajo inicial**.

Inicialización del espacio de trabajo (con Spring Initializr)

Spring Boot ofrece una herramienta web llamada **Spring Initializr**, esta permitirá obtener un **código fuente** sobre el cual se **comenzará a trabajar**. Para inicializar el proyecto se debe ir a la página oficial de **Spring Initializr** <https://start.spring.io/> :



The screenshot shows the Spring Initializr web application interface. It has a dark theme with a sidebar on the left containing a menu icon and social media links. The main content area is divided into several sections:

- Project:** Includes radio buttons for "Maven Project" (selected) and "Gradle Project".
- Language:** Includes radio buttons for "Java" (selected), "Kotlin", and "Groovy".
- Spring Boot:** Includes radio buttons for various versions: "2.6.0 (SNAPSHOT)", "2.6.0 (RC1)", "2.5.7 (SNAPSHOT)", "2.5.6" (selected), "2.4.13 (SNAPSHOT)", and "2.4.12".
- Project Metadata:** Includes text input fields for "Group" (com.misiontic), "Artifact" (account_ms), "Name" (account_ms), "Description" (Microservice Account), and "Package name" (com.misiontic.account_ms).
- Packaging:** Includes radio buttons for "Jar" (selected) and "War".
- Java:** Includes radio buttons for versions "17", "11" (selected), and "8".
- Dependencies:** A section on the right with a button "ADD DEPENDENCIES... CTRL + B" and the text "No dependency selected".

At the bottom, there are three buttons: "GENERATE CTRL + G", "EXPLORE CTRL + SPACE", and "SHARE...".

Para **inicializar** el **proyecto** de manera **correcta**, se debe **rellenar** la **información** de la parte **izquierda** y **agregar** las **dependencias** en la parte **derecha**.

Primero se agrega la **información** necesaria, se pide el tipo de **proyecto** (**Maven** en este caso), el **lenguaje** (**Java**), la **versión** de **Spring Boot** (**2.5.6**), algunos **metadatos** del **proyecto**, el tipo de **Packaging** (**Jar**) y la versión de **Java** (**11**). Todo esto en conjunto debe lucir así:

Project
☒ Maven Project
 ☐ Gradle Project

Language
☒ Java
 ☐ Kotlin
 ☐ Groovy

Spring Boot
☐ 2.6.0 (SNAPSHOT)
 ☐ 2.6.0 (RC1)
 ☐ 2.5.7 (SNAPSHOT)
 ☒ 2.5.6
 ☐ 2.4.13 (SNAPSHOT)
 ☐ 2.4.12

Project Metadata

Group
com.misiontic

Artifact
account_ms

Name
account_ms

Description
Microservice Account

Package name
com.misiontic.account_ms

Packaging
☒ Jar
☐ War

Java
☐ 17
☒ 11
☐ 8

Una vez se tiene la información del proyecto, se deben agregar las respectivas **dependencias**. Una **dependencia** es un **conjunto** de código **adicional** que ayuda a **Spring Boot** a realizar alguna tarea específica.

En este caso, se necesitan dos dependencias: **Spring Web** y **Spring Data MongoDB**, la **primera** se encarga de crear un **servicio** de **tipo REST** y la **segunda** se encarga de **realizar** la **conexión** con la **base de datos**. Para agregar la **dependencia**, se debe hacer **clic** en **ADD DEPENDENCIES**, ingresar el **nombre** y **seleccionarla**, (si aparece más de una, seleccionar la del nombre exacto). Luego de agregar las dependencias, el inicializador debe lucir así:

Dependencies

ADD DEPENDENCIES... CTRL + B

Spring Web
WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Spring Data MongoDB
NOSQL

Store data in flexible, JSON-like documents, meaning fields can vary from document to document and data structure can be changed over time.

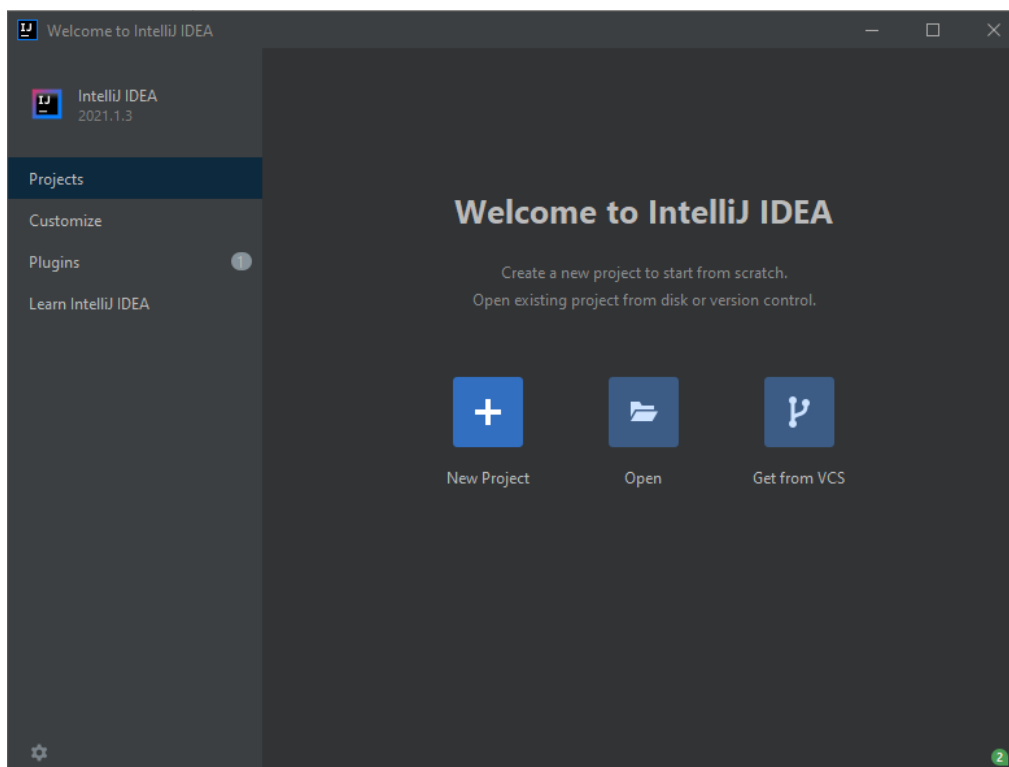
Una vez se tiene la **información** y las **dependencias** agregadas, se genera el **proyecto** haciendo clic en **GENERATE**, esto descargará un **.zip** llamado **account.zip**, el cual contiene el código fuente sobre el cual se trabajará.

Apertura local del espacio de trabajo (con IntelliJ IDEA)

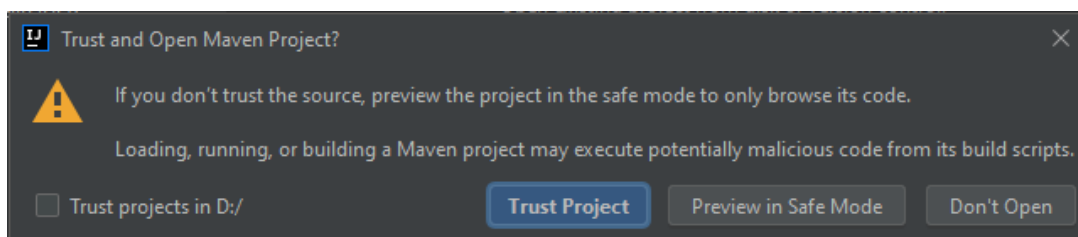
El **.zip** generado por **Spring Initializr** contiene el **código inicial** para comenzar a **trabajar**, este puede ser editado con cualquier herramienta (incluso el bloc de notas), pero en este caso se usará **IntelliJ IDEA**, por tres razones:

- Provee una **estructura** de **archivos simple** (omitiendo algunos), lo cual **facilita** el **desarrollo**.
- **Integra** algunas herramientas como la **terminal** o **consola** de **comandos**, lo cual **facilita** la **ejecución** del **servicio**.
- Y lo más importante, **automáticamente** **descargará** las **dependencias**, evitando el **uso** de **comandos adicionales**.

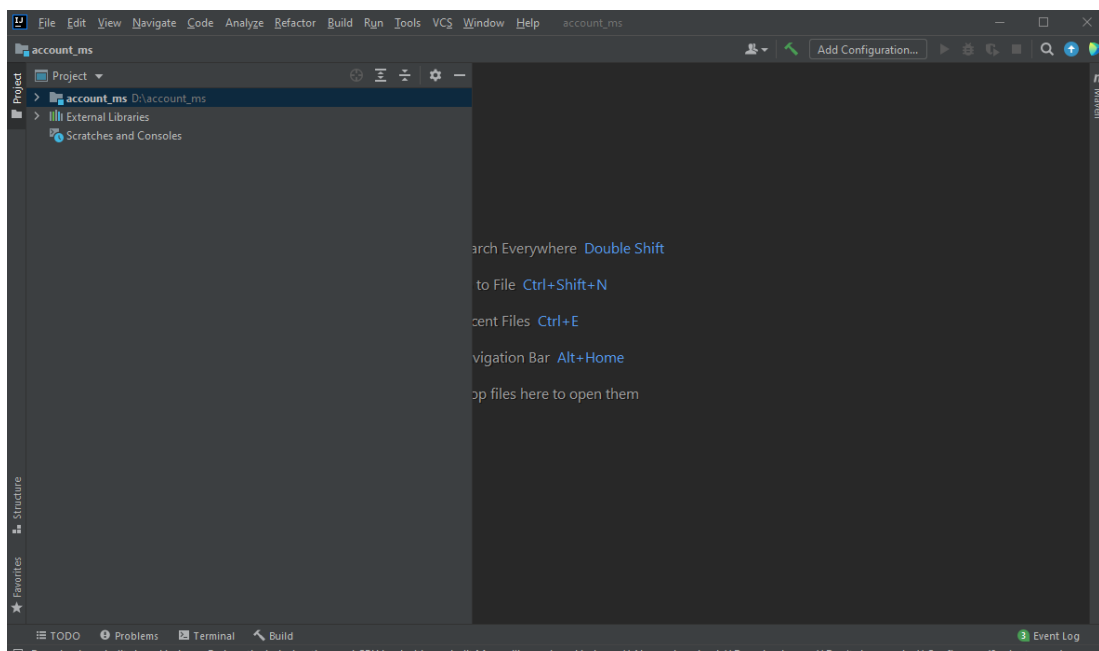
Primero se debe **descomprimir** el **archivo** con alguna herramienta (como **WinRAR**) y luego se debe abrir **IntelliJ IDE**. Si es la primera vez que se abre, es posible que aparezcan algunos mensajes, los cuales se deben aceptar. Luego de esto, se tendrá algo así:



Se debe **elegir** la opción **Open** y **seleccionar** la **carpeta** llamada **account_ms**, generada al realizar la descompresión del archivo. Una vez se seleccione, aparecerá este mensaje:



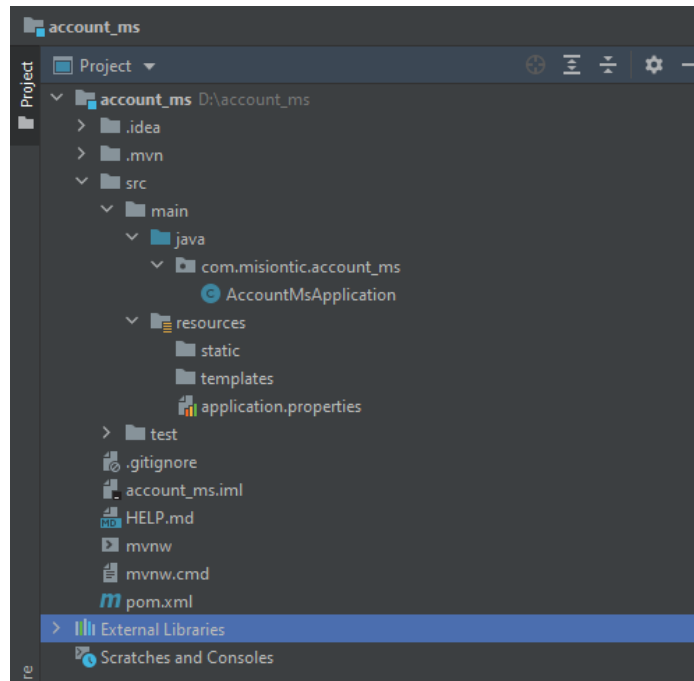
Seleccionar **Trust Project** y de esta forma se abrirá el proyecto, **automáticamente** se **comenzarán** a **descargar** las **dependencias necesarias** y demás paquetes, el **progreso** se puede **observar** en la **esquina inferior derecha**. Es posible que las **descargas tarden algunos minutos**, se debe **esperar**, cuando se hayan terminado las descargar, la barra de progreso desaparecerá y se tendrá lo siguiente:



Con esto se podrá dar inicio al proceso de desarrollo.

Explicación de la Estructura

Antes de comenzar el proceso de desarrollo, se explicará brevemente la **estructura** del **proyecto**, se debe tener claro que **IntelliJ IDEA** ofrece **distintas vistas** y no **solamente** la **tradicional** que se suele tener en el **explorador de archivos**, por lo general se trabaja con vista **Project**, esta luce así:



Algunos elementos importantes de la estructura son:

- El **código fuente** y donde se agregarán las **clases necesarias** para el **desarrollo** se encuentran en **src/main/** (ejemplo: clase inicial **AccountMsApplication**).
- La carpeta **.mvn/** contiene lo necesario para el empaquetado y la ejecución del servicio.
- La información del proyecto se encuentra en el archivo **pom.xml**.
- El archivo **mvnw** contiene los scripts que se usarán para **ejecutar** el proyecto.
- Y en **External Libraries** se encuentran las librerías externas usadas por el proyecto y las dependencias.

Conexión con MongoDB

Ahora que se tiene la **estructura inicial** del **proyecto** o **espacio de trabajo**, se puede **comenzar** el proceso de **desarrollo**. Para esta **primera práctica** se buscará **configurar** el **acceso** a la **base de datos** creada en la **clase pasada**.

Se debe **recordar** la URL **generada**:

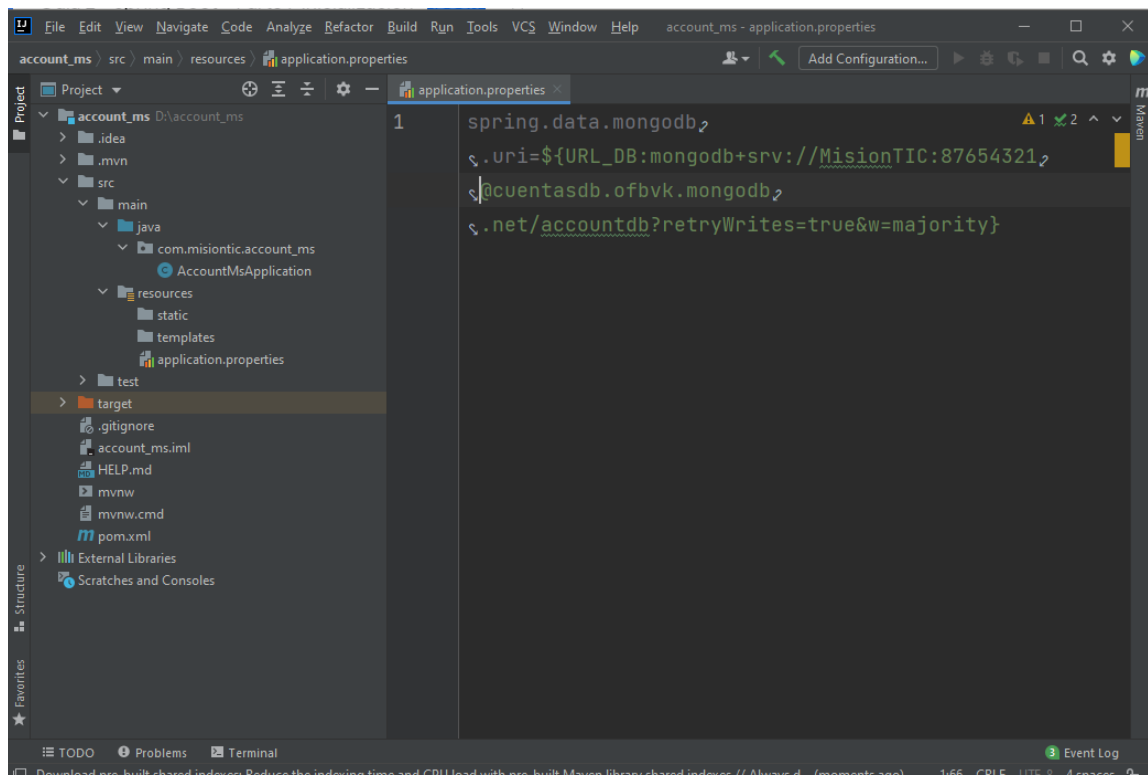
```
mongodb+srv://MisionTIC:87654321@cuentasdb.ofbvk.mongodb.net/accountdb?retryW  
rites=true&w=majority
```

Esta URL le brinda la información **necesaria** a **Spring Boot** para **realizar** la **conexión** con

MongoDB, y se debe ubicar en el archivo `src/main/resources/application.properties` de la siguiente manera:

```
spring.data.mongodb.uri=${URL_DB:su_url}
```

Es decir:

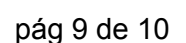


Se guardan los cambios y ya se habrá realizado el primer cambio. Si se está haciendo uso de git, es una buena opción realizar un primer commit.

Ejecución de la Aplicación

En este punto el **proyecto** está conformado solamente por una **clase inicial** llamada `AccountMsApplication` y un archivo de recursos llamado `application.properties`. Es una estructura bastante simple, pero a pesar de eso, se puede **ejecutar** la aplicación. Para esto se deben seguir los siguientes pasos:

Abrir una terminal en IntelliJ IDEA: para esto hacer clic sobre la opción terminal de la esquina inferior derecha, esto mostrará lo siguiente:



Para probar la aplicación se debe abrir la siguiente url: <http://localhost:8080/>, el resultado debe ser algo similar a:

Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

Fri Jul 23 18:12:52 COT 2021

There was an unexpected error (type=Not Found, status=404).

Si bien no se encontró información (debido a que no se tiene nada), el servidor si está escuchando las peticiones en el puerto **8080**, esto se puede evidenciar en la consola.

```
2021-07-23 18:12:52.804 INFO 17036 --- [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet 'dispatcherServlet'
2021-07-23 18:12:52.805 INFO 17036 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
2021-07-23 18:12:52.805 INFO 17036 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 0 ms
```

Y de esta manera se ha completado la ejecución de la aplicación. Para detener el servidor se oprime **Control+C** y luego se ingresa **Y** o **Yes**.

Nota: el anterior flujo de trabajo: cambios y ejecución, se usará a lo largo de las próximas actividades.