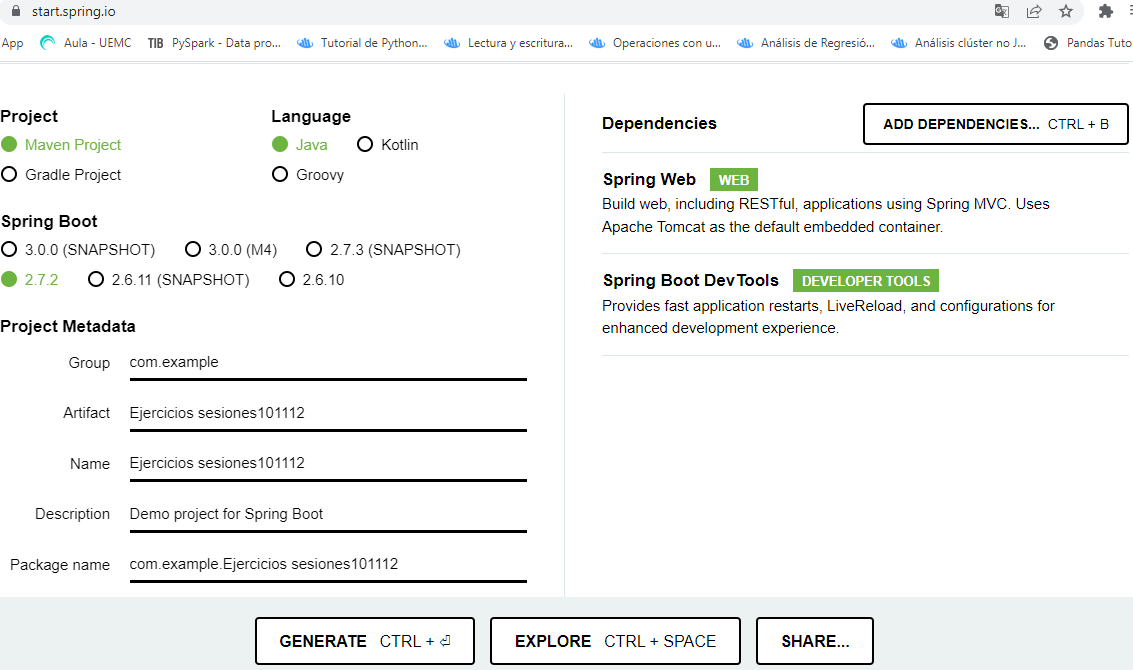
**Ejercicios sesiones 10, 11 y 12**

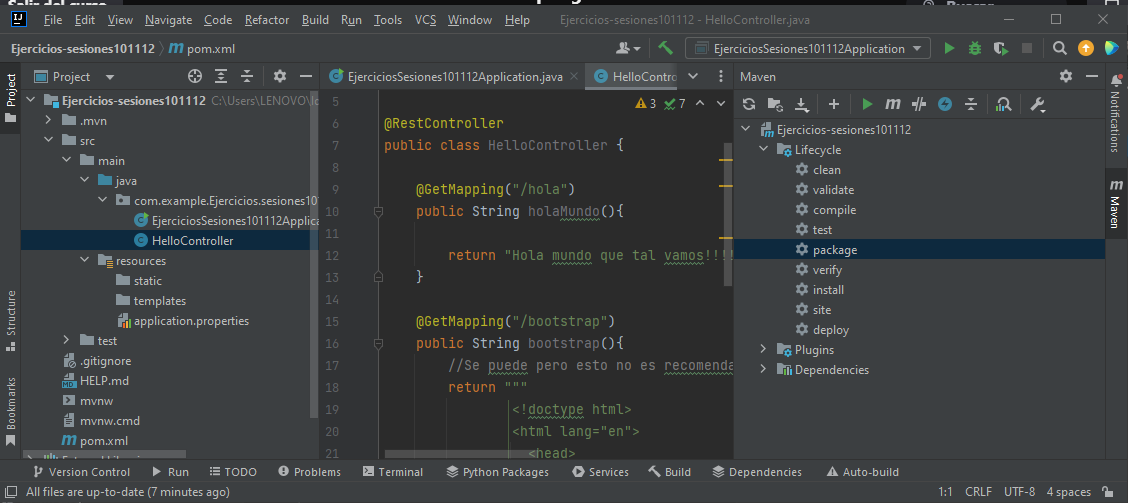
**Ejercicio 1**

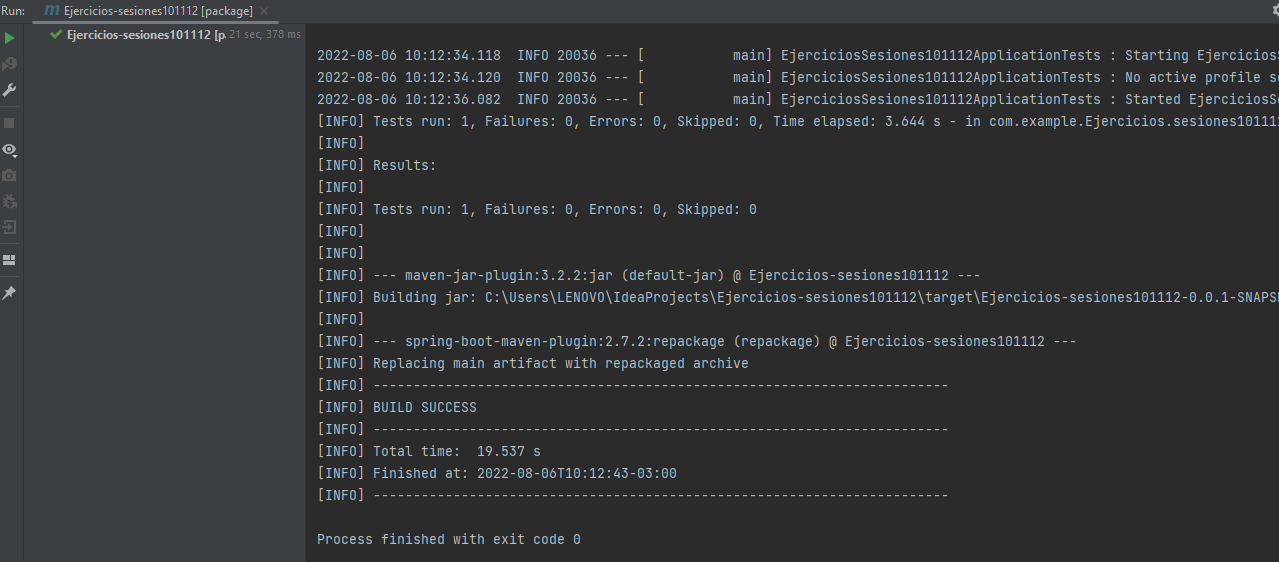
1. **Creo proyecto con Spring initializr**

****

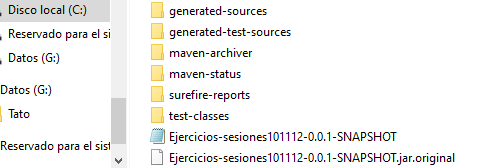
Probar a empaquetar la aplicación con maven package desde Intellij IDEA

1. Para empaquetar la aplicación





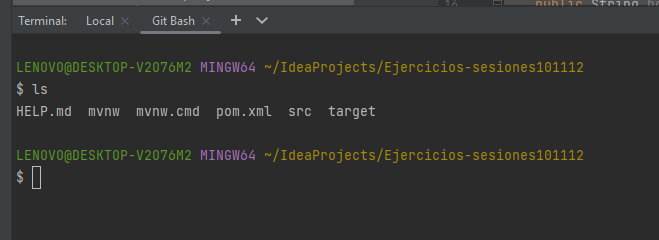
En carpeta TARGET del proyecto se generan los archivos jar



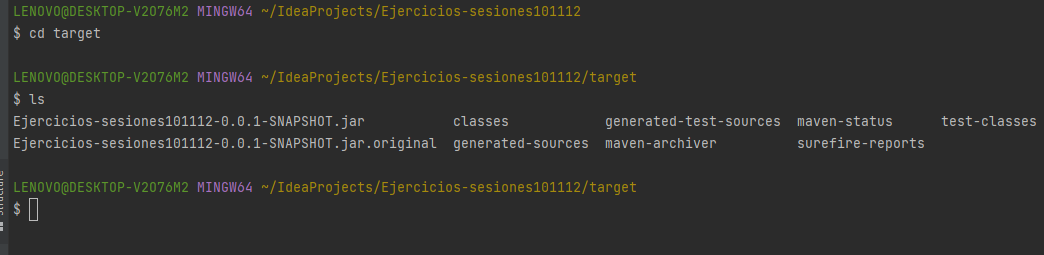
Desde terminal en Intellij IDEA verificar que se se ejecuta correctamente con el comando:

java -jar target/spring-deploy-1.0.jar

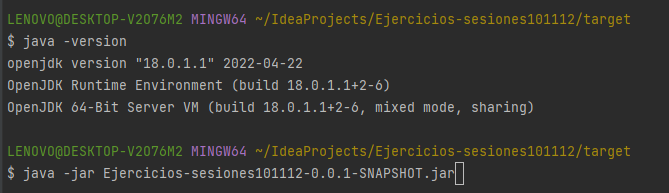
1. Abrimos terminal y verificamos el contenido de la carpeta del proyecto



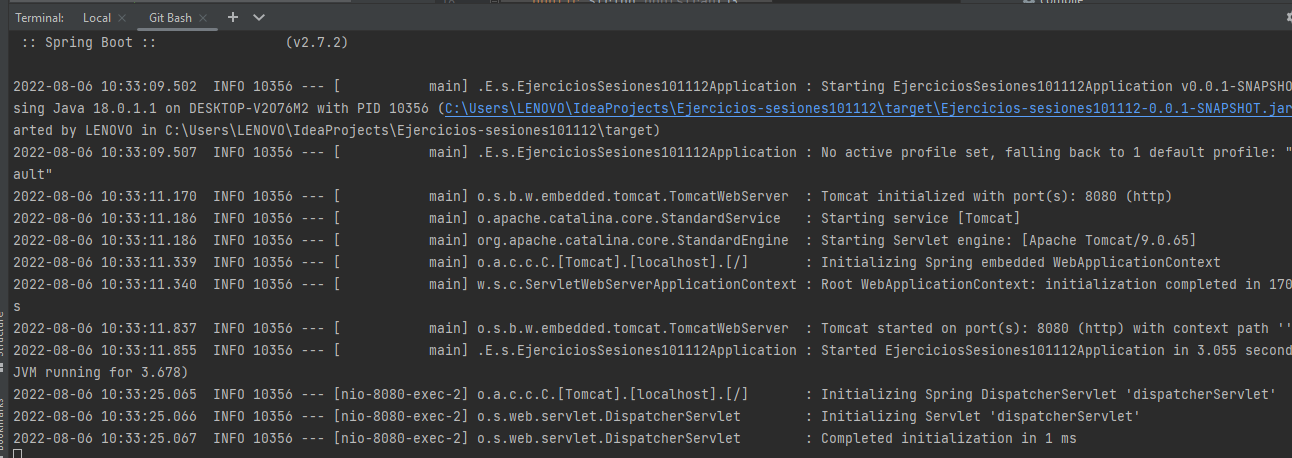
1. Contenido carpeta target

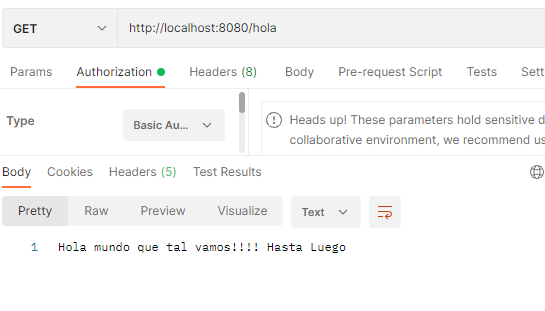


1. Verifico versión de java y saber que esta en el entorno y ejecuto el archivo jar para arrancar la aplicación desde la terminal.

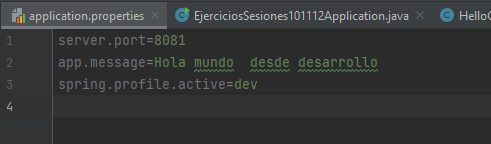


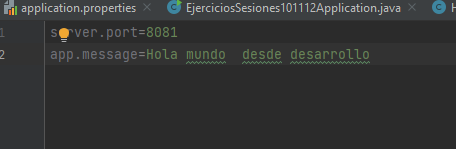
1. Aplicación funcionando

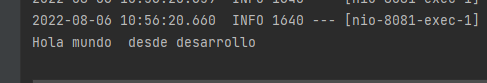




Crear un perfil para dev y otro para test con una propiedad nueva que carguemos en el controlador.



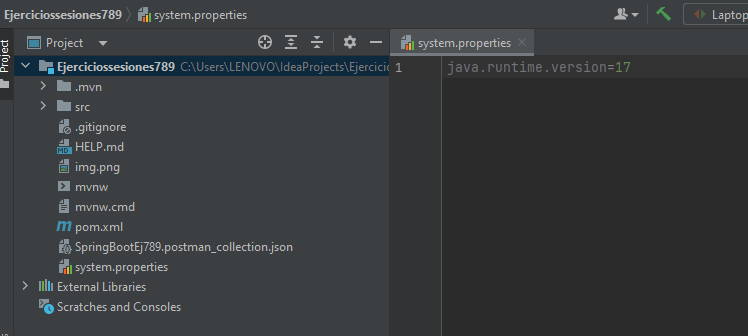




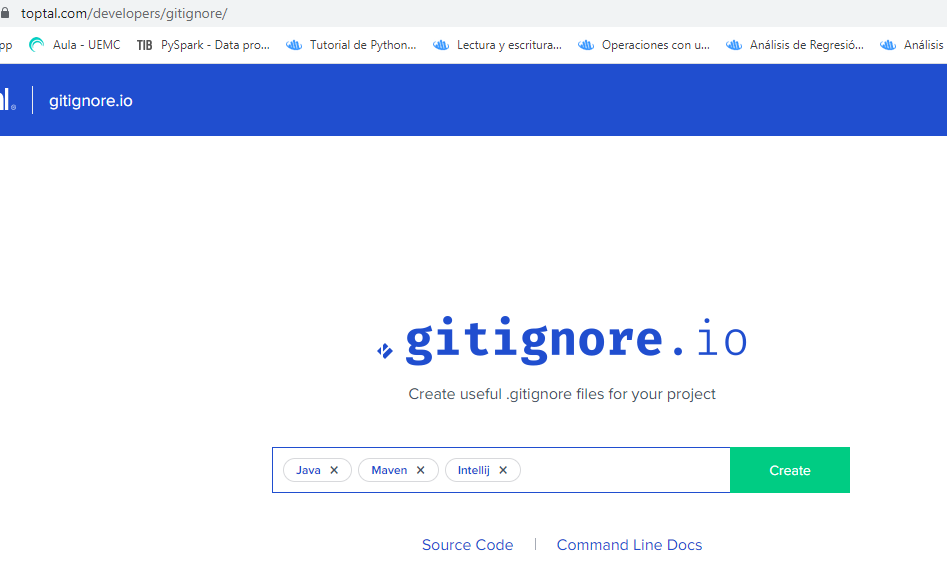
**Ejercicio 2**

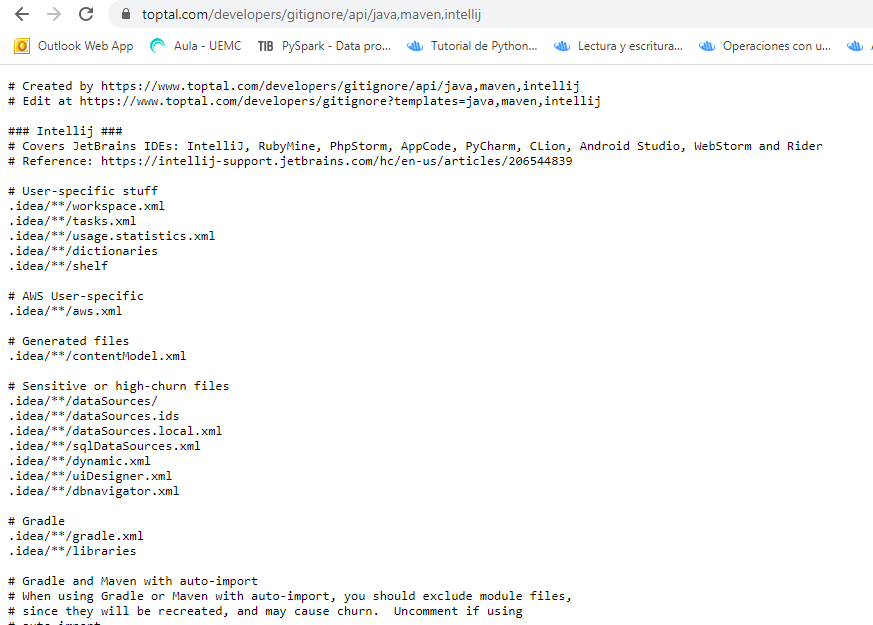
Desplegar el API REST de Laptops en Heroku y verificar funcionamiento desde POSTMAN.

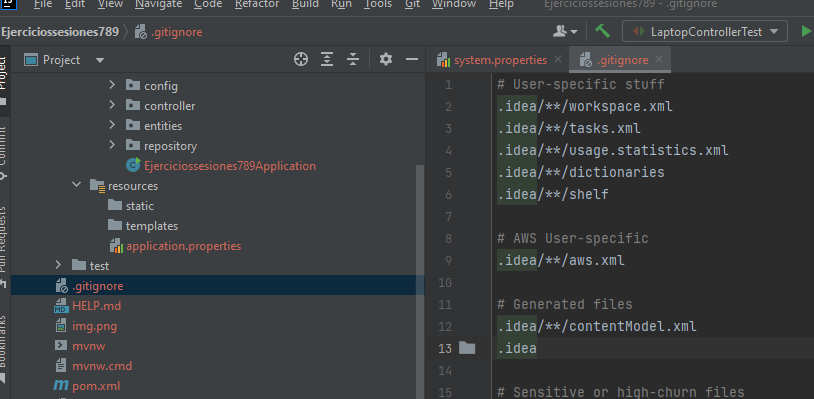
1. Creamos archivo system.properties en el proyecto



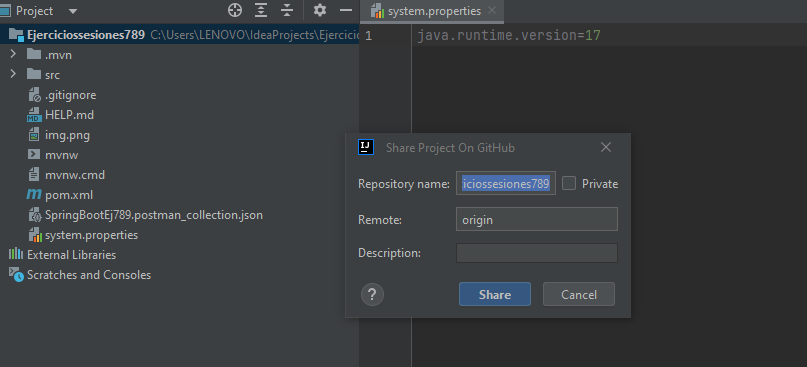
1. Parametrizamos archivo gitignore para no subir información que se ejecuta en local con pagina web gitignore.io y copiamos en archivo gitignore del proyecto

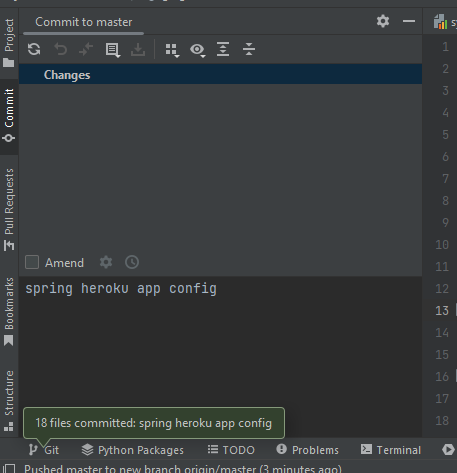


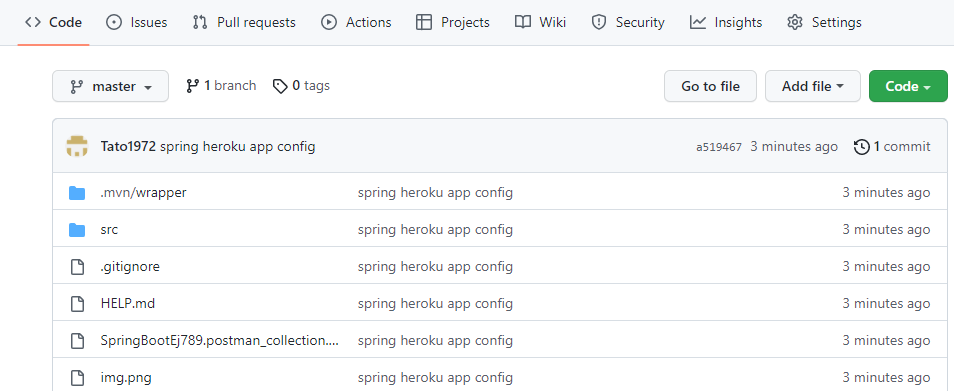




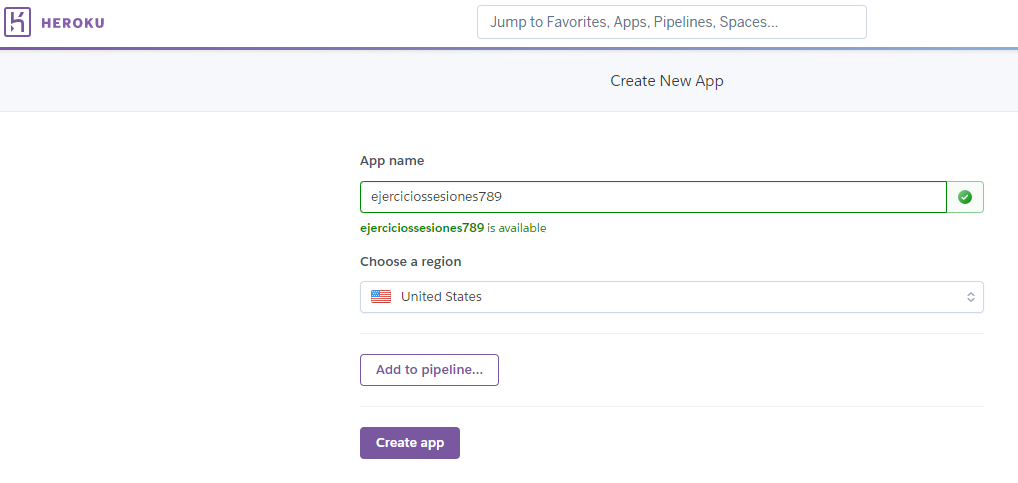
1. Subimos proyecto a github



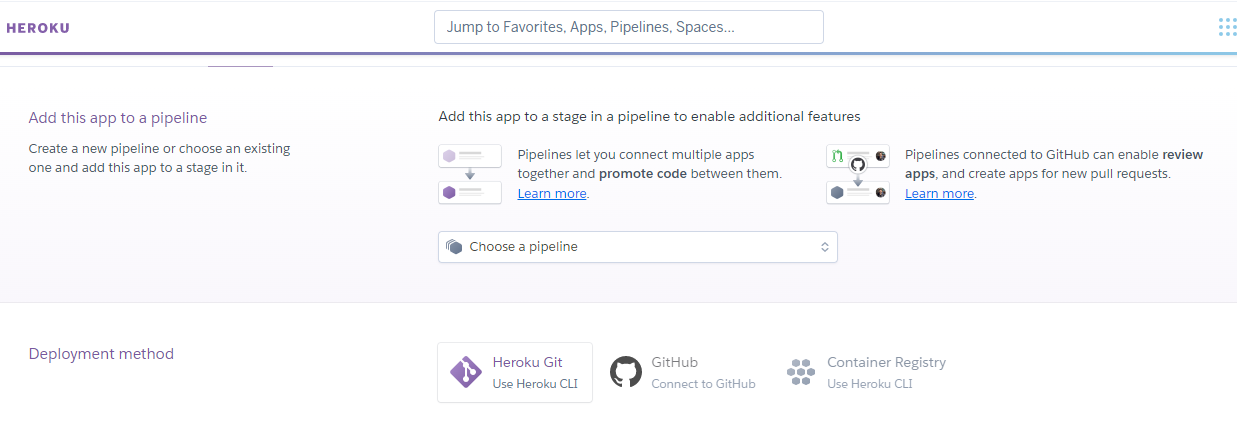




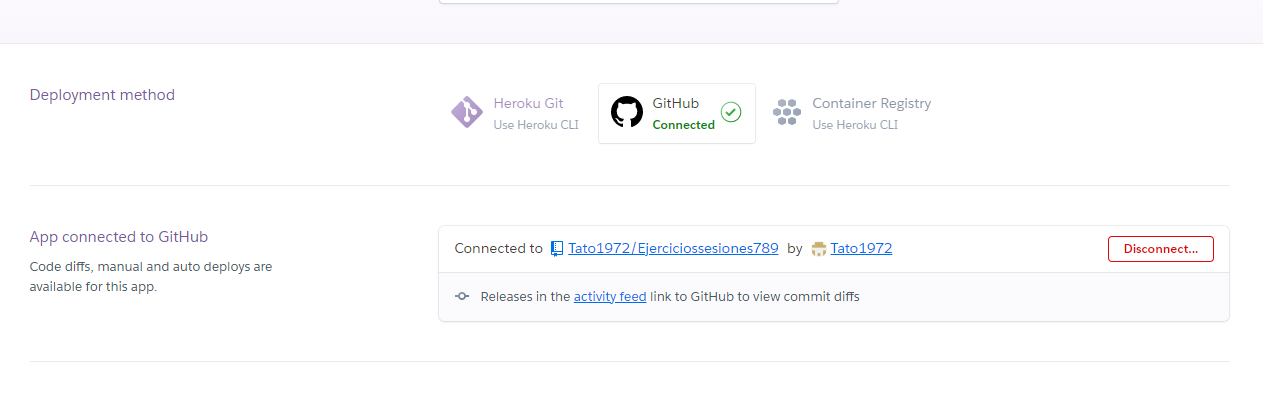
1. Creamos la app en Heroku



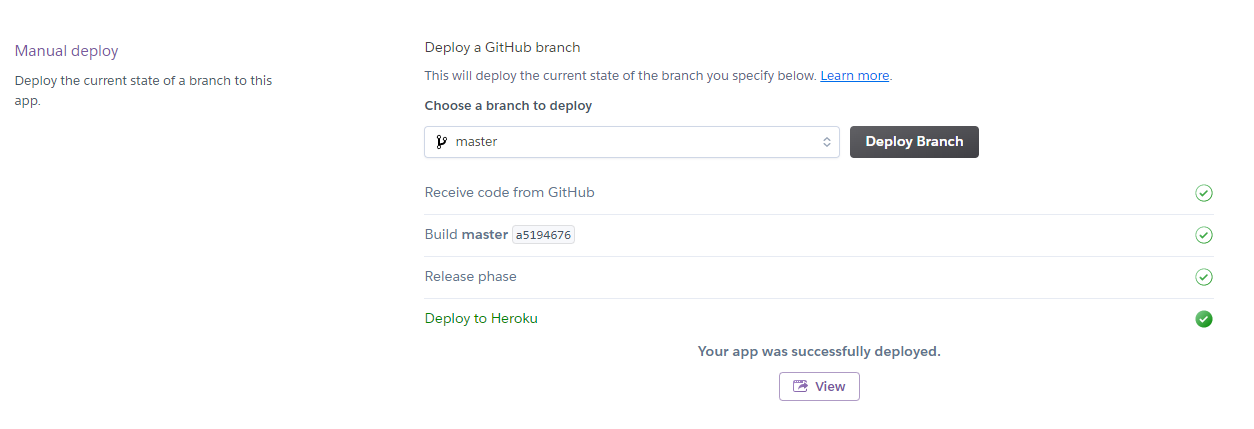
Desplegamos desde GitHub



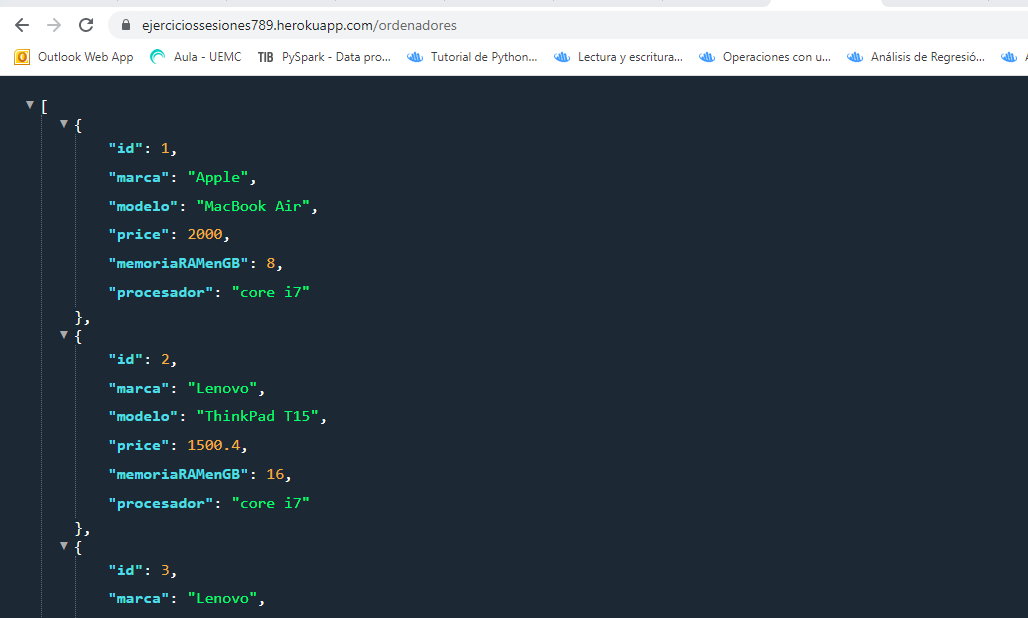
Conectamos con el proyecto en el repositorio GitHUb con Heroku



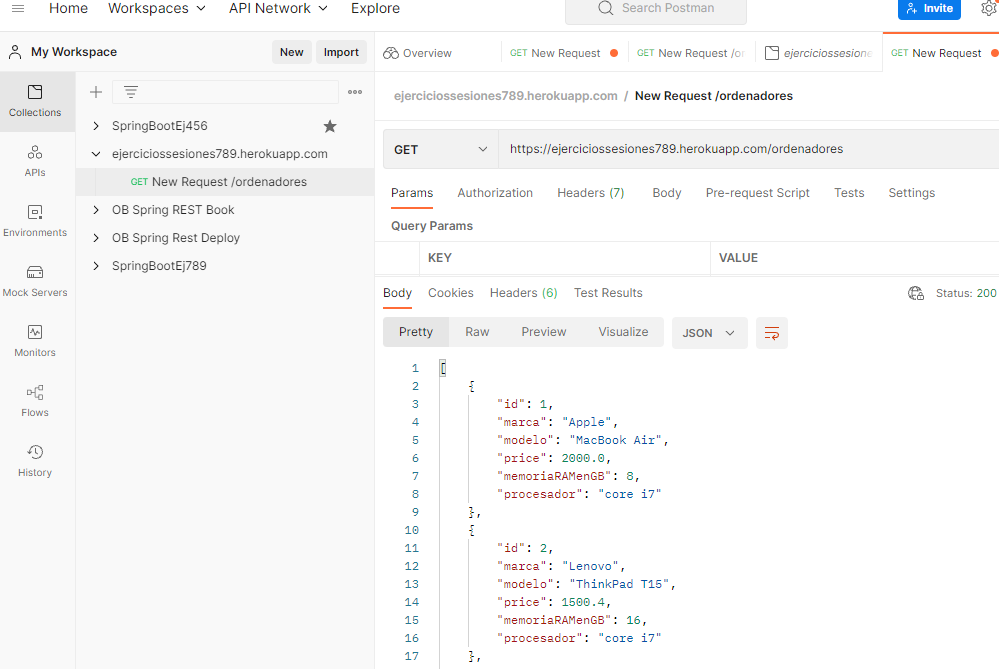
Deployamos manualmente en Heroku



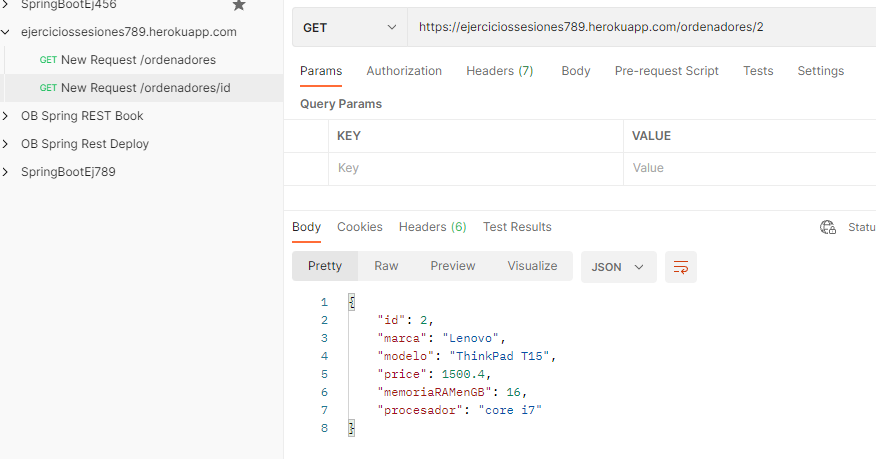
Completamos la URL de Heroku con nuestra ruta y vemos la salida de nuestra app en el navegador



Comprobamos en Postman



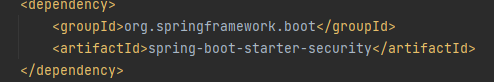
Busqueda por id

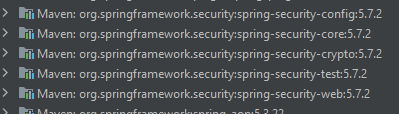
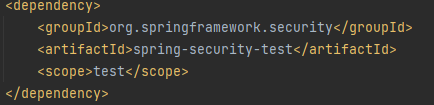


**Ejercicio 3**

Añadir Spring Security sobre el proyecto API REST de Laptops y configurar 2 usuarios en memoria utilizando una clase WebSecurityConfig.

1. Importar dependencias de Spring Security. En Intellij modifique archivo pom.xml agregando dependencias e importe con maven.



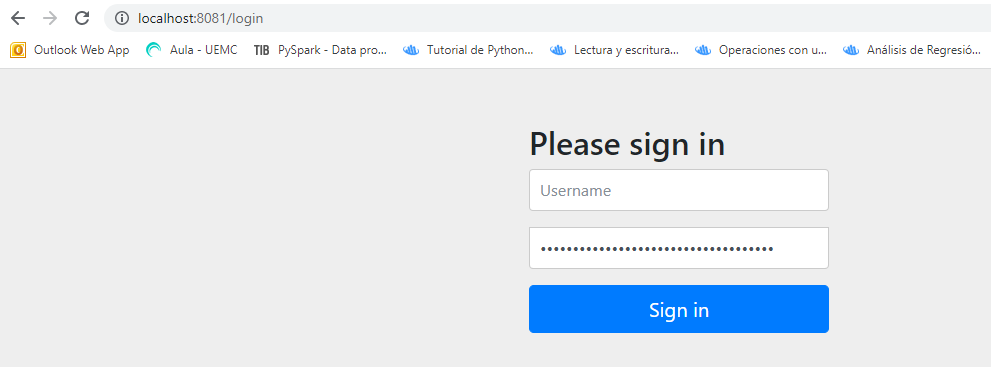


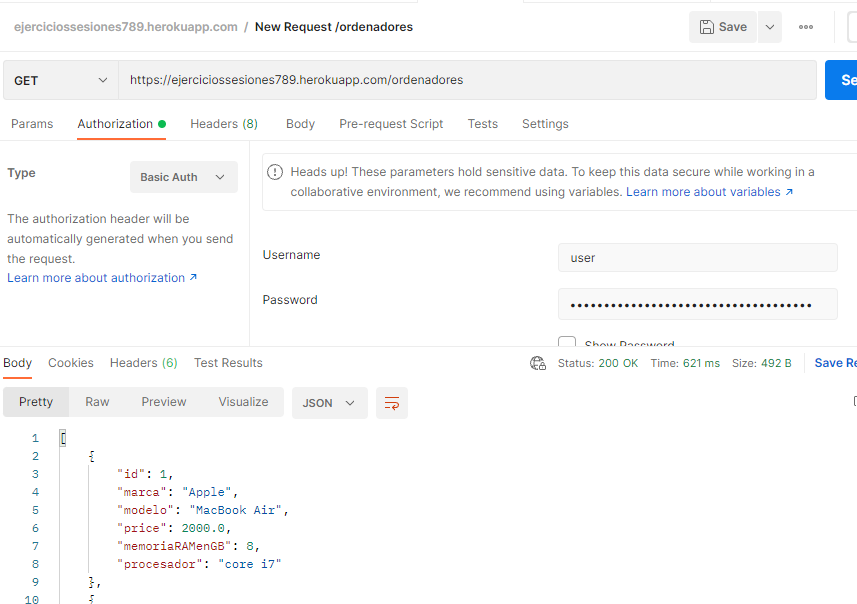
En consola se ve una contraseña



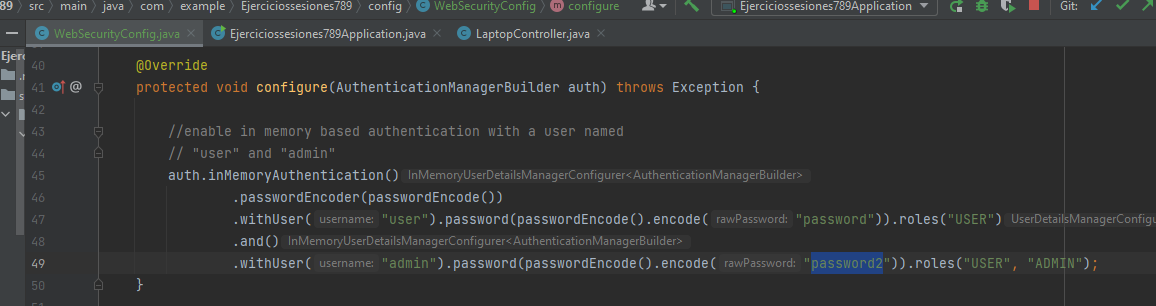


La app en el navegador solicita clave y contraseña para ingresar

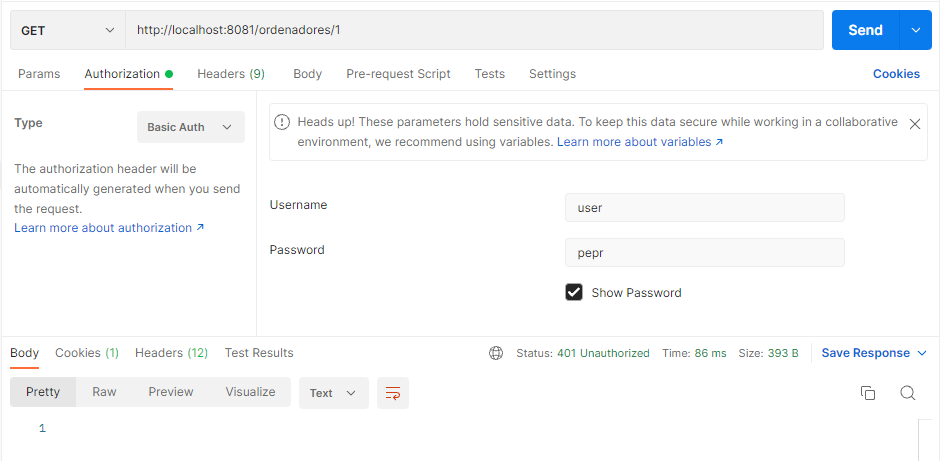




Configuramos dos usuarios (user y admin) con password (password y password1) para ingresar a la app a búsquedas por id



Con usuario y clave no válidas.



Con usuario y clave válidas.

