



INSTITUTO POLITÉCTICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO
ESCOM



Unidad de Aprendizaje:
Desarrollo de aplicaciones móviles nativas
PRÁCTICA 1

“Herramientas para el desarrollo de aplicaciones
móviles nativas en Android”

Integrantes:

Ávila Juárez Alexis Aramis | 2021630148

Jurado Santos Alberto Isaac | 2021630365

Luna Márquez José Antonio | 2021630374

Morales Palacios Sebastián | 2021630461

Morales Torres Alejandro | 2021630480

Profesor

Gabriel Hurtado Avilés

Fecha de entrega

1 de octubre de 2024

Objetivo

Familiarizarse con algunas de las herramientas más utilizadas en el desarrollo de aplicaciones móviles web, nativas e híbridas en Android, mediante su instalación y configuración, con el fin de preparar el entorno de trabajo para los proyectos del curso.

Introducción

Durante el desarrollo de esta práctica se desarrollará bajo el framework de Spring Boot, donde se tratará de implementar una API REST para calcular la Cerradura de Kleene y la Cerradura Positiva sobre cadenas binarias de longitud especificada por el usuario. La estructura del proyecto sigue las convenciones de arquitectura de Spring Boot, separando las responsabilidades en controladores, servicios y recursos, lo que facilita su mantenimiento y escalabilidad. El archivo `ClosureController.java` define el controlador principal, anotado con `@RestController` para gestionar solicitudes HTTP, y con `@RequestMapping("/api/closure")`, que especifica la ruta base de los endpoints.

El controlador incluye dos endpoints principales, definidos con la anotación `@GetMapping`. El primero, `"/api/closure/star/{number}"`, invoca el método `getKleeneStar` para calcular la Cerradura de Kleene a partir del parámetro `{number}` recibido en la URL. El segundo endpoint, `"/api/closure/plus/{number}"`, llama al método `getKleenePlus` para calcular la Cerradura Positiva. Ambos métodos dependen del servicio `ClosureService`, que se inyecta mediante la anotación `@Autowired`, y cuya lógica de negocio es la encargada de procesar las cerraduras de Kleene y positiva. Los resultados se retornan como un `Map<String, String>`, que es serializado automáticamente en formato JSON para ser enviado en la respuesta HTTP.

Este diseño modular sigue el patrón MVC (Modelo-Vista-Controlador), donde el Controlador (`ClosureController`) recibe y gestiona las solicitudes, y el Servicio (`ClosureService`) encapsula la lógica de negocio. Gracias a la separación de responsabilidades y a la utilización de Spring Boot, el proyecto es fácilmente mantenible y extensible, permitiendo la implementación de futuras funcionalidades relacionadas con el cálculo de conjuntos en cadenas binarias.

Desarrollo

Ejercicio 1: Instalación y configuración de herramientas.

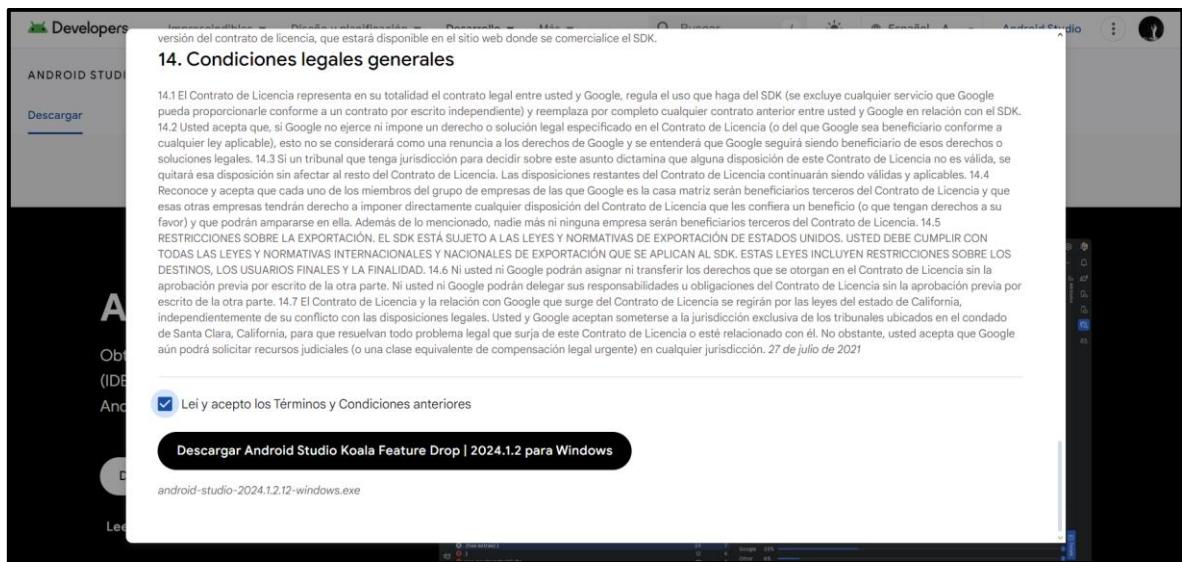
Para el desarrollo de la práctica comenzaremos con la instalación de nuestras herramientas de trabajo en este caso se realizará la guía para dos sistemas operativos debido a que el equipo trabajará de la mano con el sistema operativo Linux y Windows, para la ejecución de programas no deberá de tener ningún inconveniente ya que el código está pensado para ejecutarse de manera que no depende de donde se realizó originalmente hablando respecto al sistema operativo.

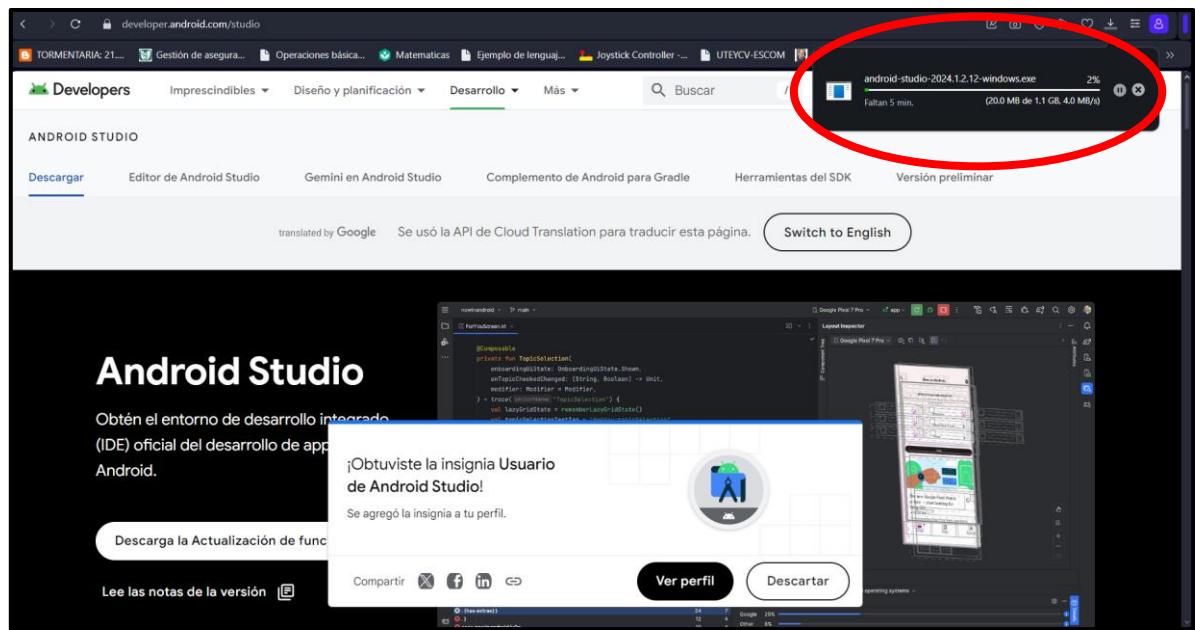
Instalación de herramientas para Windows:

1. Android Studio

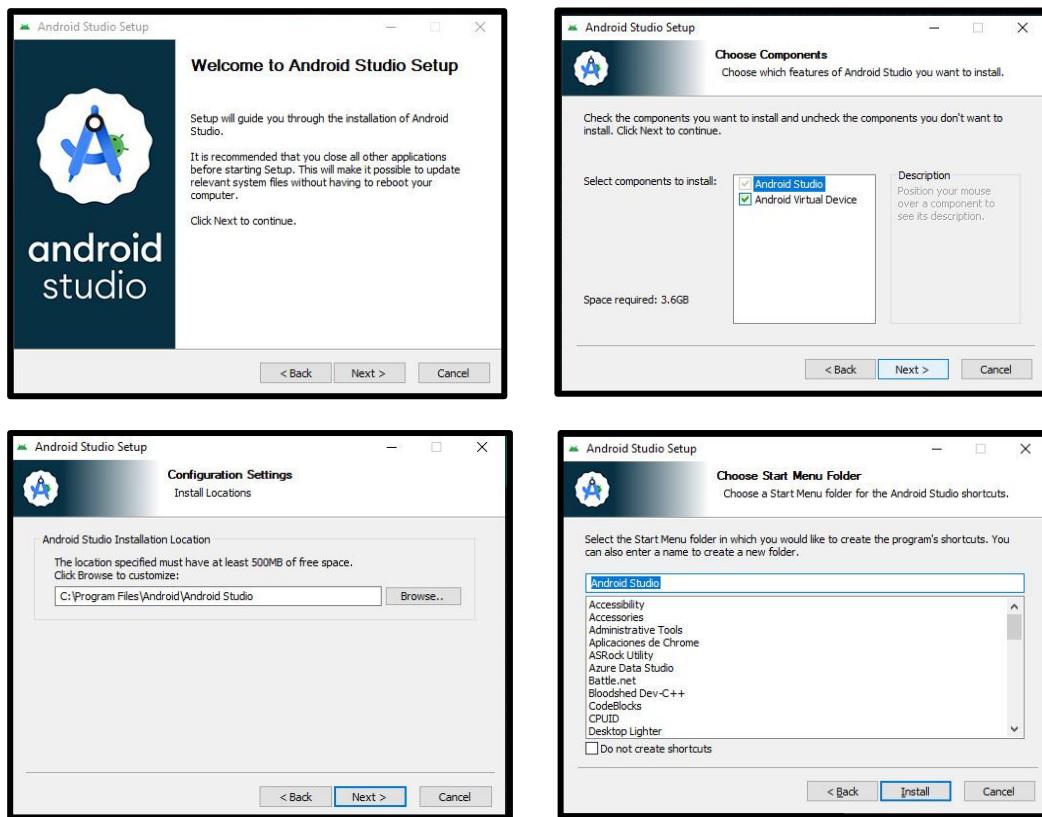
Para descargar Android Studio en Windows debemos seguir los siguientes pasos:

- Accedemos a la página oficial de Android Studio en <https://developer.android.com/studio>.
- Seleccionamos la opción de Descargar Android Studio, nos desplegará una ventana emergente donde debemos aceptar los términos y condiciones, una vez aceptemos podremos descargar el ejecutable.

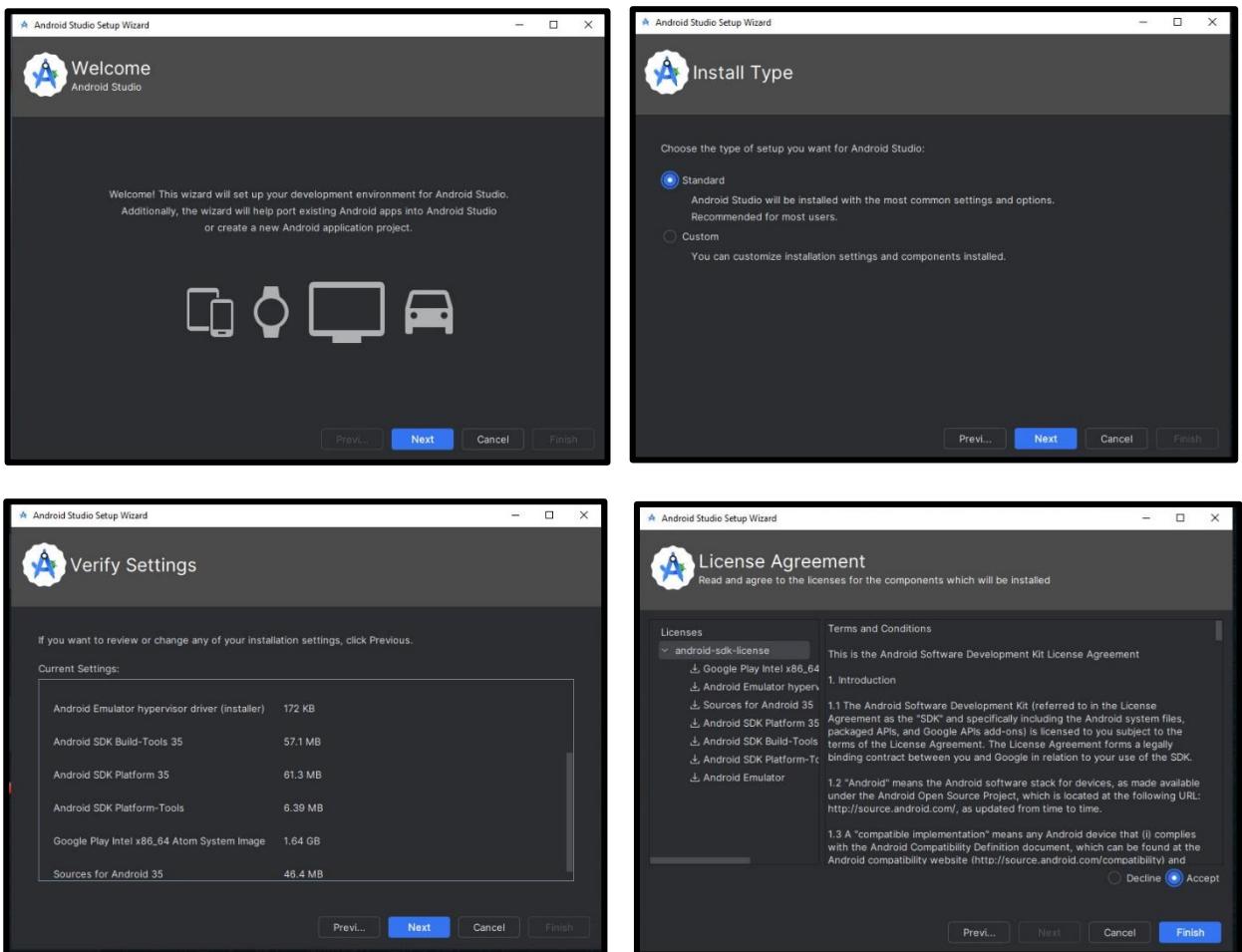




- Una vez este se haya descargado, procedemos a ejecutarlo y seguir la instalación por defecto de cualquier instalador de programas. Seleccionaremos el componente de Android Studio y seleccionaremos la ruta de instalación, para posteriormente confirmarla.



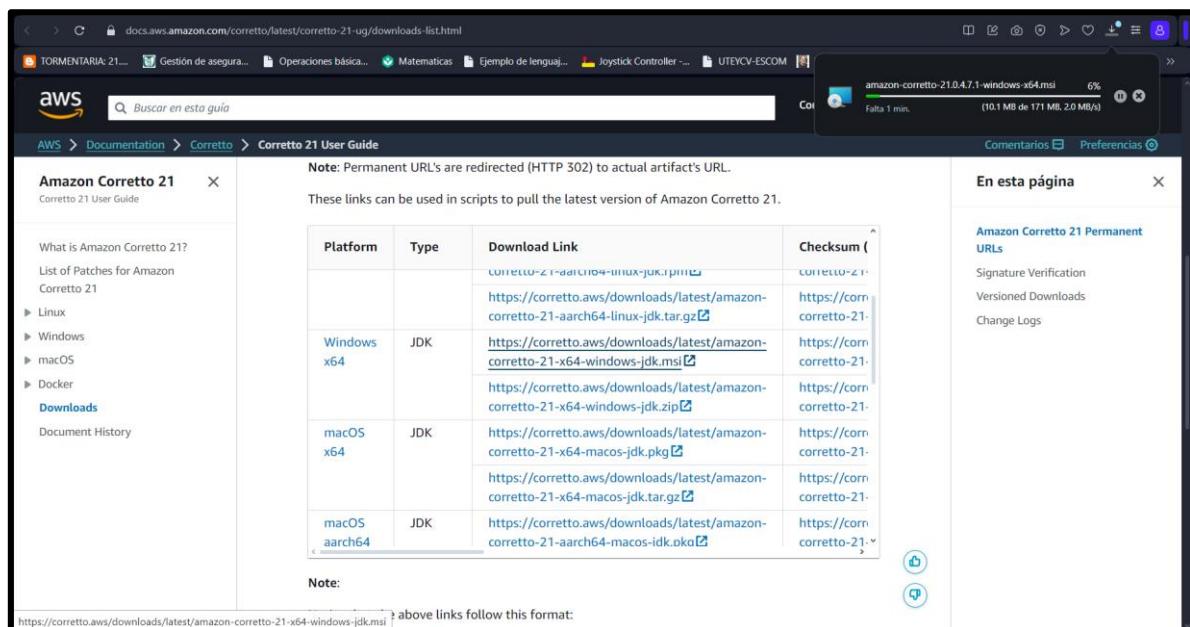
- Ya que se haya instalado, la abrimos para continuar con la instalación, preferentemente daremos en siguiente ya que las opciones que nos da Android Studio por defecto nos funcionarán perfectamente, en su mayoría serán componentes para el entorno.



2. Java Development Kit (JDK):

Para descargar este componente se nos recomienda usar la versión de Amazon Coretto:

- Accedemos al sitio oficial de descarga del JDK Coretto en <https://docs.aws.amazon.com/corretto/latest/corretto-21-ug/downloads-list.html>.
- Buscamos la versión a descargar, en este caso el .msi para Windows y lo descargamos.



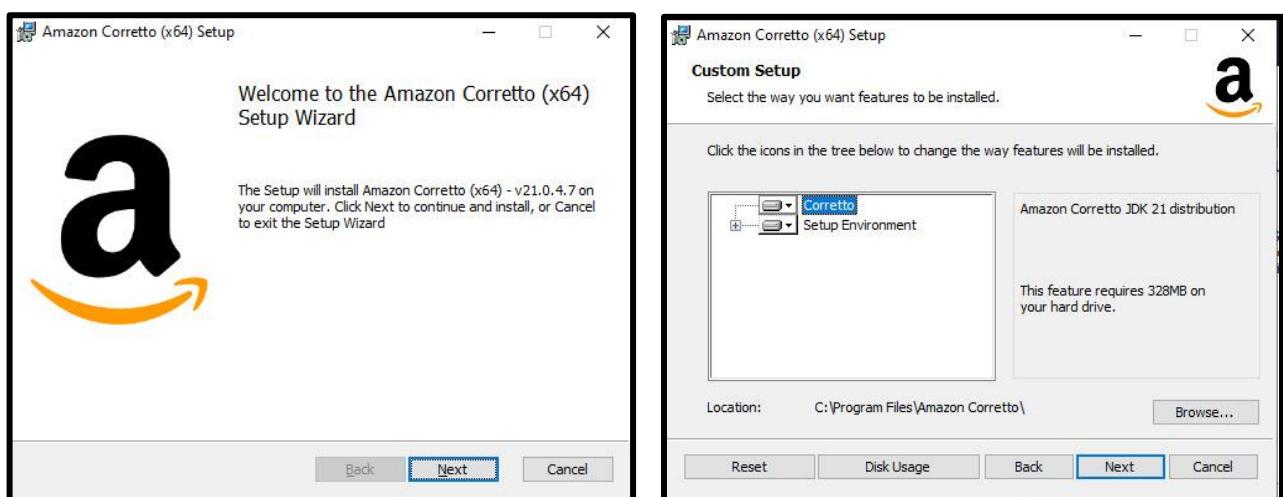
The screenshot shows a web browser window with the URL <https://docs.aws.amazon.com/corretto/latest/corretto-21-ug/downloads-list.html>. The page is titled 'Amazon Corretto 21' and includes a sidebar with links for Linux, Windows, macOS, Docker, and Downloads. The main content area displays a table of download links:

Platform	Type	Download Link	Checksum (SHA256)
Windows x64	JDK	corretto-21-x64-windows-jdk.msi	corretto-21-x64-windows-jdk.msi
		corretto-21-x64-windows-jdk.zip	corretto-21-x64-windows-jdk.zip
macOS x64	JDK	corretto-21-x64-macos-jdk.pkg	corretto-21-x64-macos-jdk.pkg
		corretto-21-x64-macos-jdk.tar.gz	corretto-21-x64-macos-jdk.tar.gz
macOS arm64	JDK	corretto-21-arm64-macos-jdk.pkg	corretto-21-arm64-macos-jdk.pkg

Note: These links can be used in scripts to pull the latest version of Amazon Corretto 21.

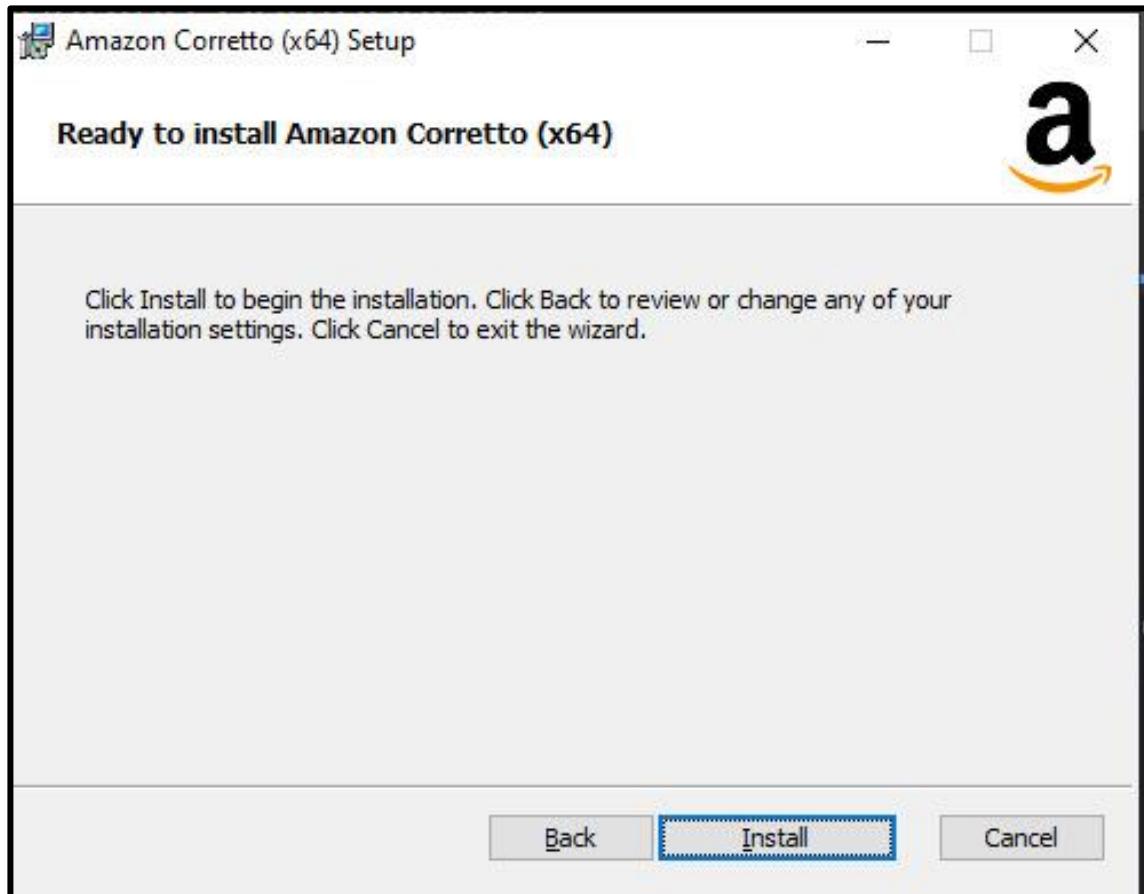
above links follow this format: <https://corretto.aws/downloads/latest/amazon-corretto-21-x64-windows-jdk.msi>

- Una vez instalado, ejecutamos este archivo.

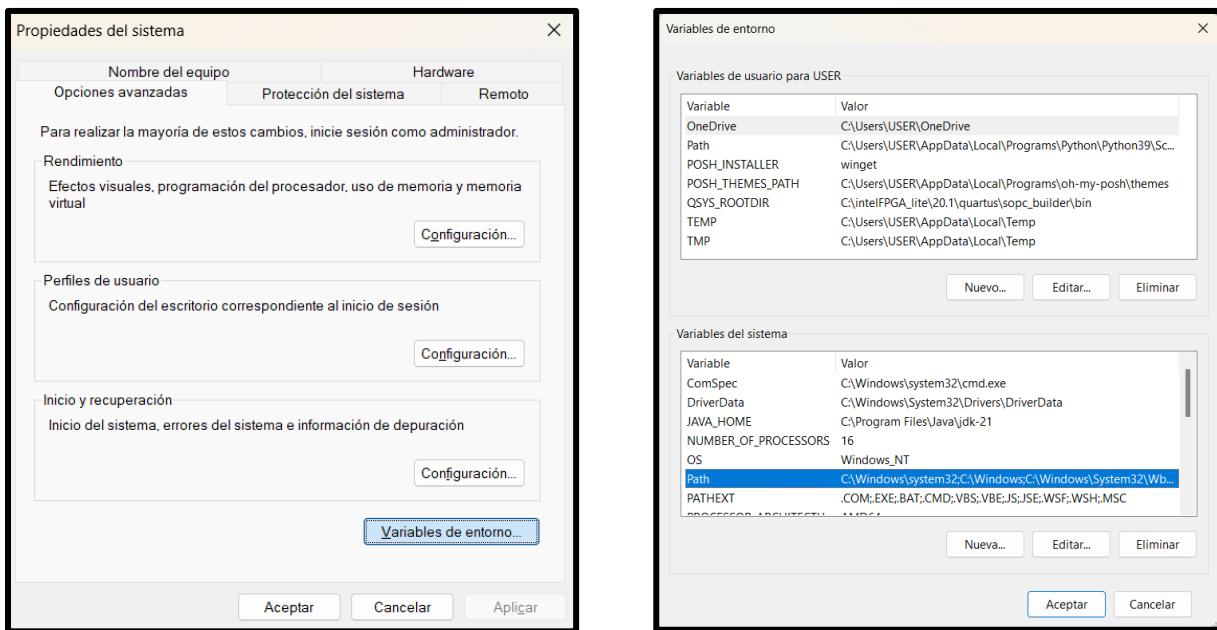


The screenshots show the 'Amazon Corretto (x64) Setup' wizard. The left screenshot is the 'Welcome' screen with the text: 'Welcome to the Amazon Corretto (x64) Setup Wizard'. The right screenshot is the 'Custom Setup' screen with the text: 'Select the way you want features to be installed'. The 'Corretto' feature is selected in the tree view. The 'Setup Environment' checkbox is also present. The 'Location' field is set to 'C:\Program Files\Amazon Corretto\'. The 'Next' button is highlighted in both screenshots.

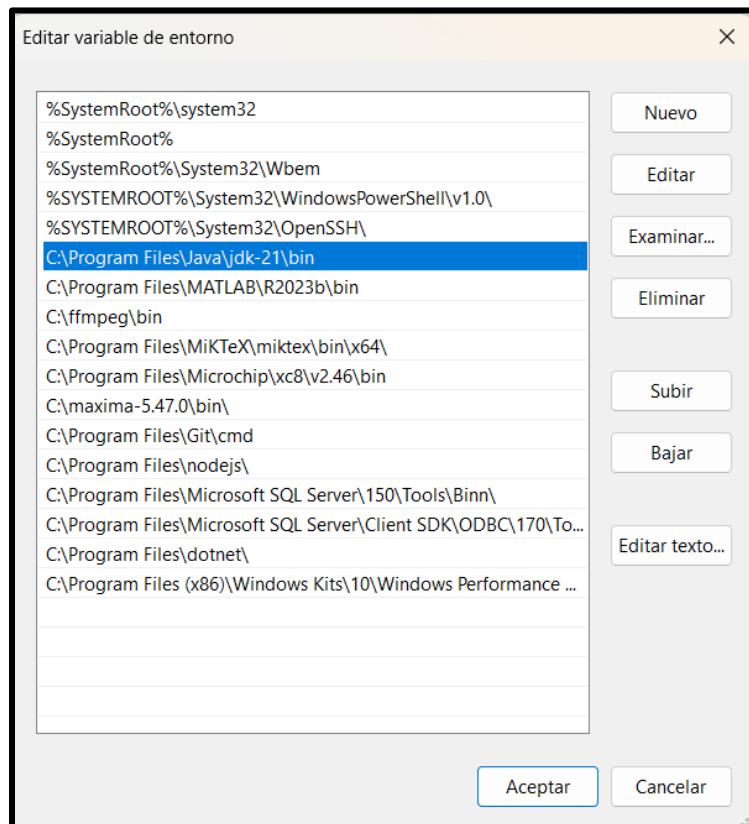
- Daremos siguiente, y en caso de querer instalarlo en otra ruta la seleccionamos, al final solo damos en instalar.



- Finalizaremos con su instalación al crear la variable de entorno del JDK, para esto buscamos el editor de Variables de Entorno en el buscador, una vez abierta esta ventana presionaremos el botón de variables de entorno, esto nos abrirá una nueva ventana, donde debemos seleccionar la ruta de Path en la parte inferior



- Aquí pondremos la ruta a la carpeta bin del JDK:



- Finalmente comprobamos su instalación:

```
Microsoft Windows [Versión 10.0.22631.4169]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\USER>java --version
openjdk 21 2023-09-19
OpenJDK Runtime Environment (build 21+35-2513)
OpenJDK 64-Bit Server VM (build 21+35-2513, mixed mode, sharing)
```

3. Maven

Para descargar Maven seguimos los siguientes pasos

- Accedemos al sitio oficial de descarga del JDK Coreto en <https://maven.apache.org/download.cgi> y descargamos el que tiene la extensión .zip

Files

Maven is distributed in several formats for your convenience. Simply pick a ready-made binary distribution archive and follow the [Installation instructions](#). Use a source archive if you intend to build Maven yourself.

In order to guard against corrupted downloads/installations, it is highly recommended to [verify the signature](#) of the release bundles against the public [KEYS](#) used by the Apache Maven developers.

Link	Checksums	Signature
Binary tar.gz archive	apache-maven-3.9.9-bin.tar.gz	apache-maven-3.9.9-bin.tar.gz.sha512
Binary zip archive	apache-maven-3.9.9-bin.zip	apache-maven-3.9.9-bin.zip.sha512
Source tar.gz archive	apache-maven-3.9.9-src.tar.gz	apache-maven-3.9.9-src.tar.gz.sha512
Source zip archive	apache-maven-3.9.9-src.zip	apache-maven-3.9.9-src.zip.sha512

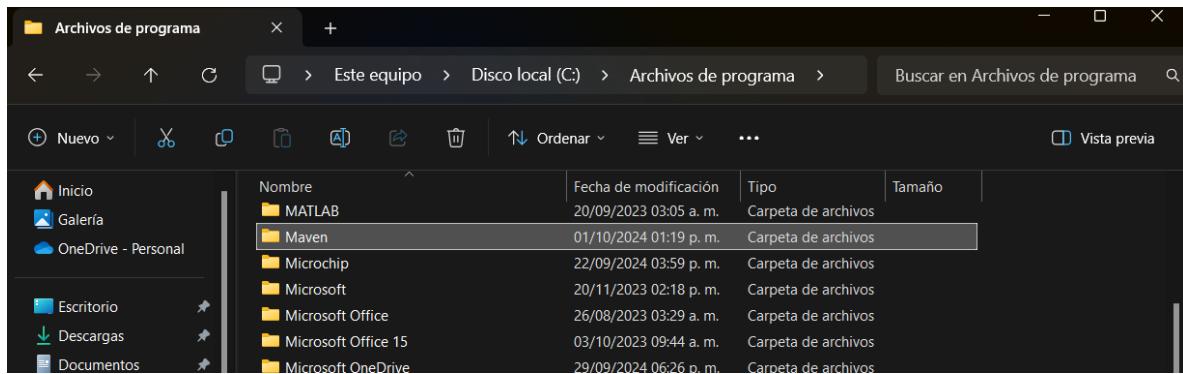
- 3.9.9 [Release Notes](#) and [Release Reference Documentation](#)
- [latest source code](#) from source repository
- Distributed under the Apache License, version 2.0
- other:
 - [All current release sources \(plugins, shared libraries,...\) available at https://downloads.apache.org/maven/](#)

Other Releases

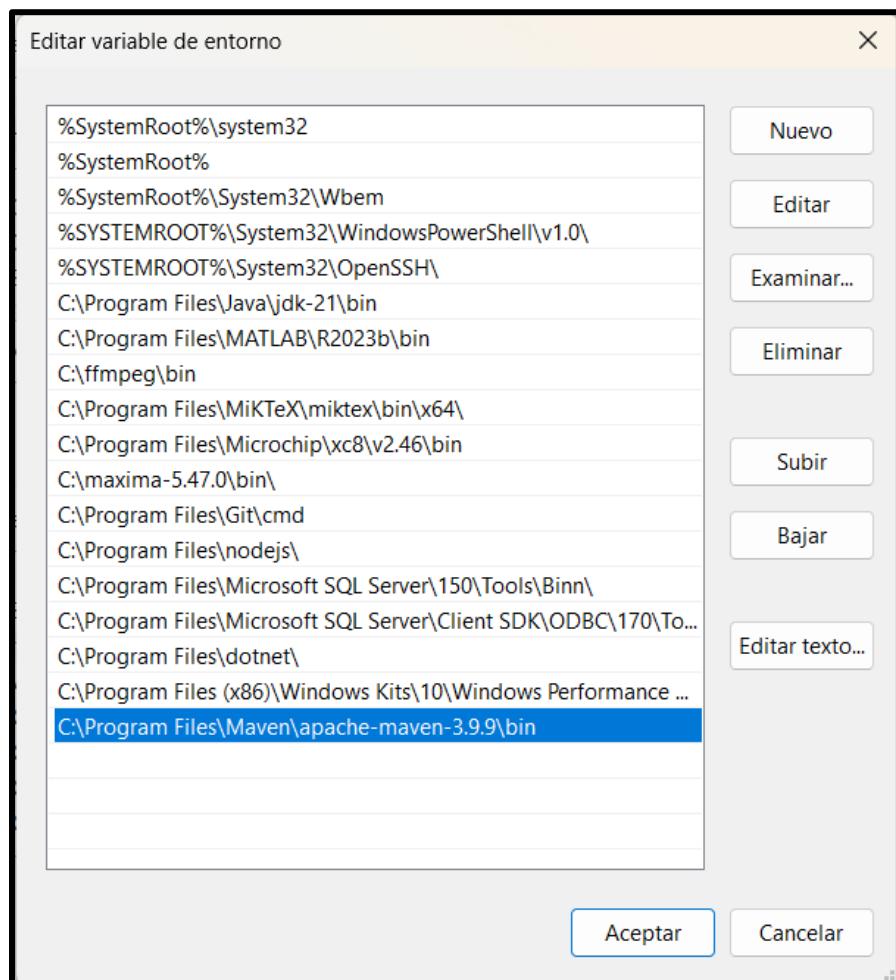
It is strongly recommended to use the latest release version of Apache Maven to take advantage of newest features and bug fixes.

If you still want to use an old version, you can find more information in the [Maven Releases History](#) and can download files from the [Maven 3 archives](#) for versions 3.0.4+ and legacy [archives](#) for earlier releases.

- Descomprimimos la carpeta y la llevamos a una carpeta segura, por convención puede ser en el Disco Local C o el Program Apps:



- Añadimos la ruta del bin de Maven a las variables de entorno, siguiendo los mismos pasos que usamos para el JDK:



- Damos en aceptar y salimos, ahora comprobamos su instalación:

```
Symbolo del sistema X + ▾

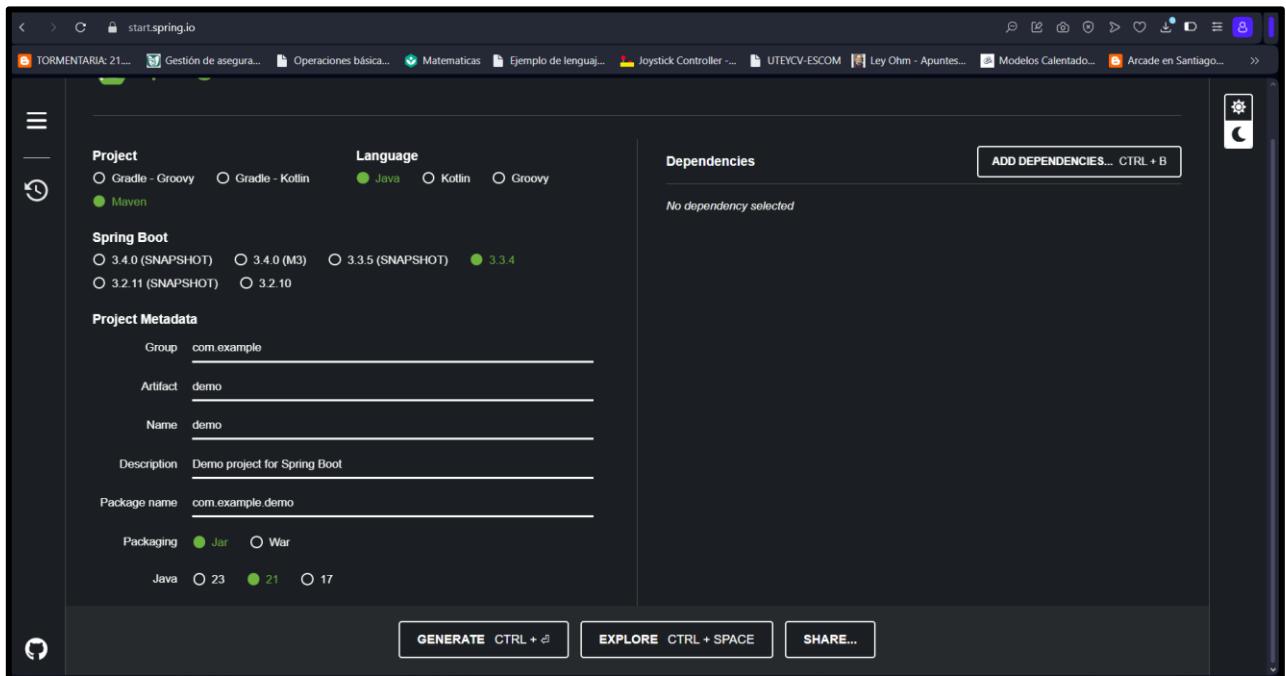
Microsoft Windows [Versión 10.0.22631.4169]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\USER>mvn --version
Apache Maven 3.9.9 (8e8579a9e76f7d015ee5ec7bfc97d260186937)
Maven home: C:\Program Files\Maven\apache-maven-3.9.9
Java version: 21, vendor: Oracle Corporation, runtime: C:\Program Files\Java\jdk-21
Default locale: es_MX, platform encoding: UTF-8
OS name: "windows 11", version: "10.0", arch: "amd64", family: "windows"
```

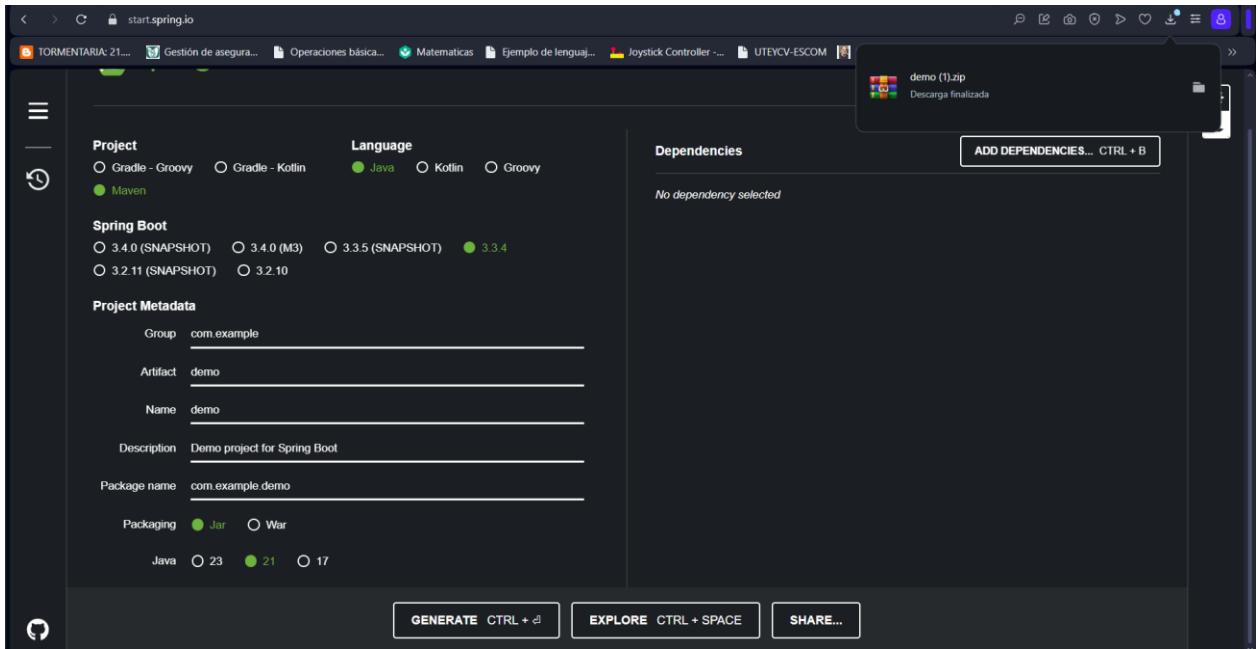
4. Spring Boot

Para crear un proyecto con Spring seguimos los siguientes pasos

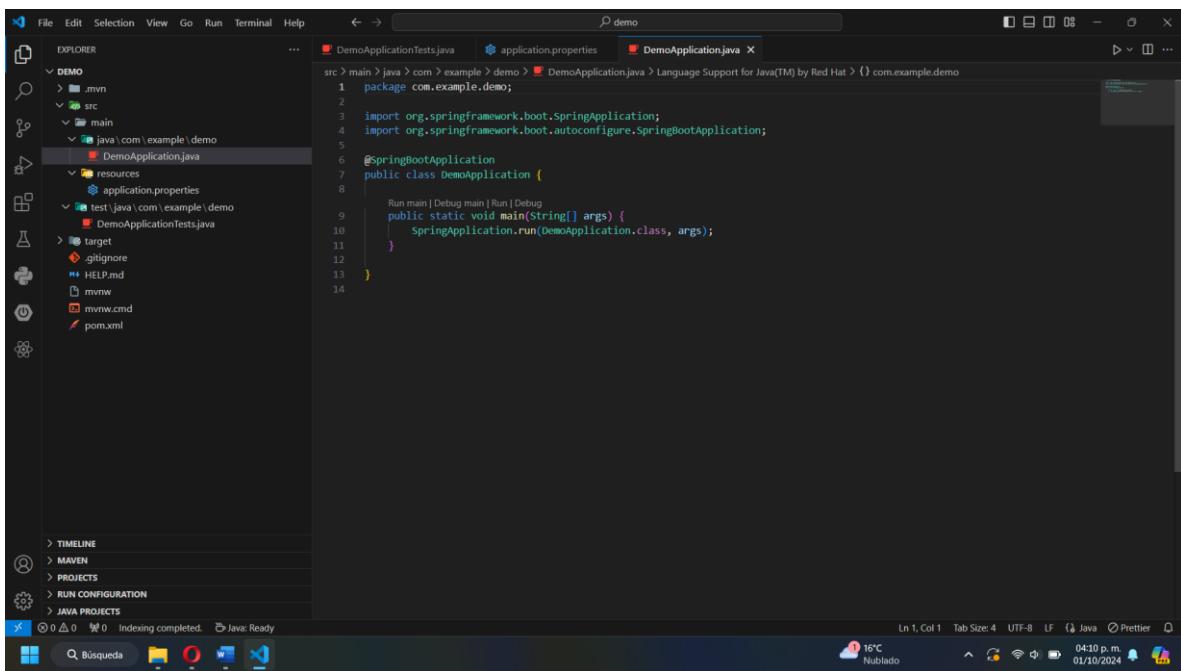
- Accedemos al sitio oficial de descarga del Spring en <https://start.spring.io> y creamos un proyecto acorde a nuestras necesidades, esto depende de el tipo de proyecto, el lenguaje a usar y su versión, la versión de spring y los detalles del proyecto



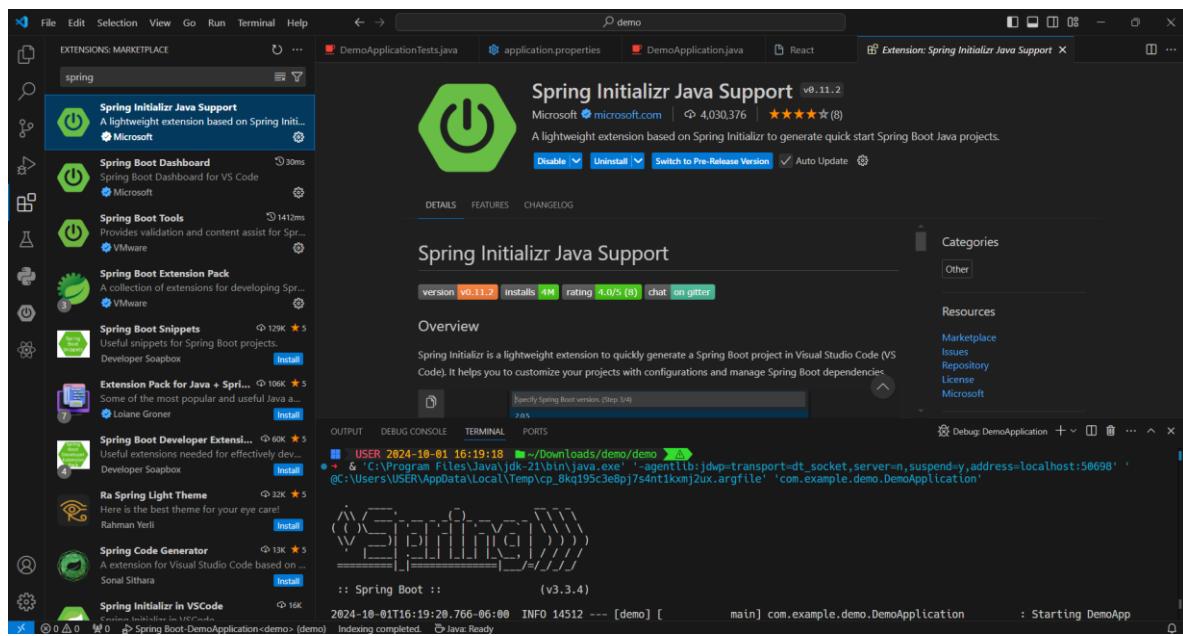
- Una vez seleccionamos todo lo necesario, presionamos el botón inferior de GENERATE y nos dará nuestro proyecto



- Una vez descargado, descomprimimos el zip y abrimos la carpeta con nuestro Vs Code:



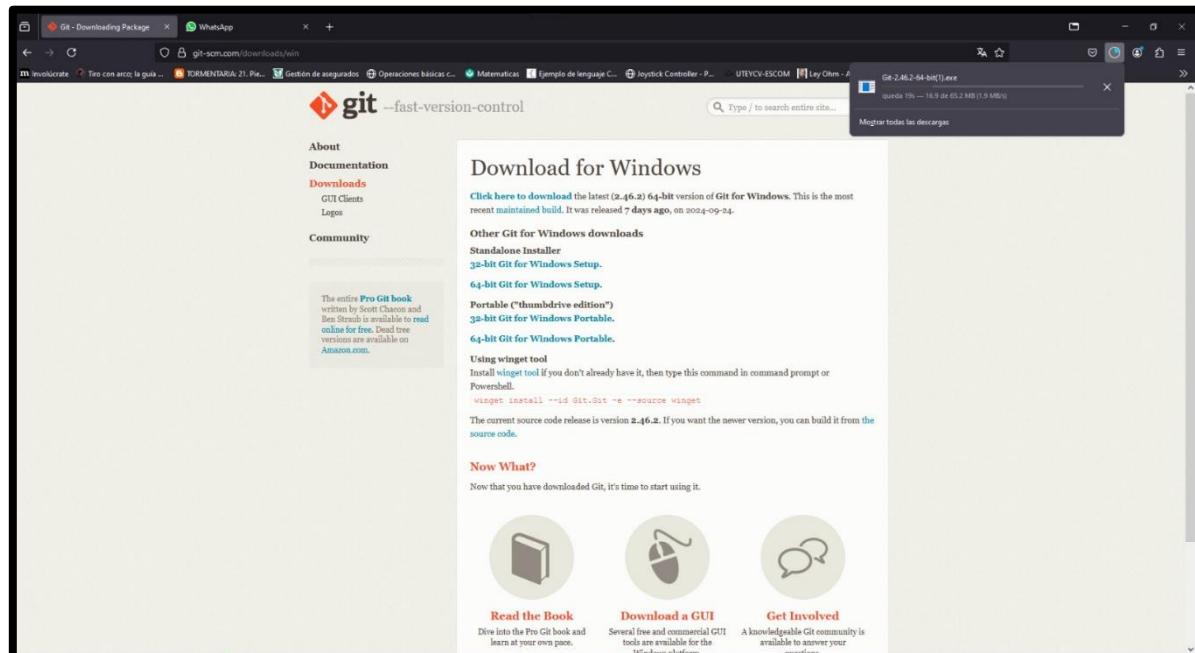
- Y listo, ya tenemos nuestro proyecto de Java con Spring y Maven, y ya de paso descargamos las extensiones para spring en el Vs Code

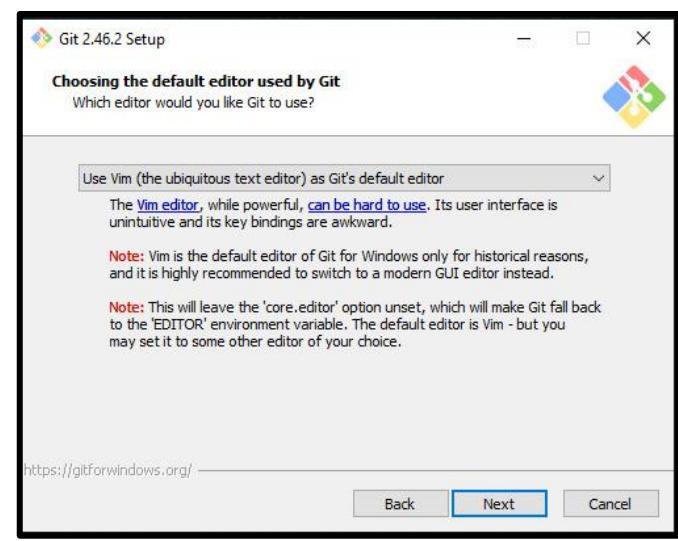
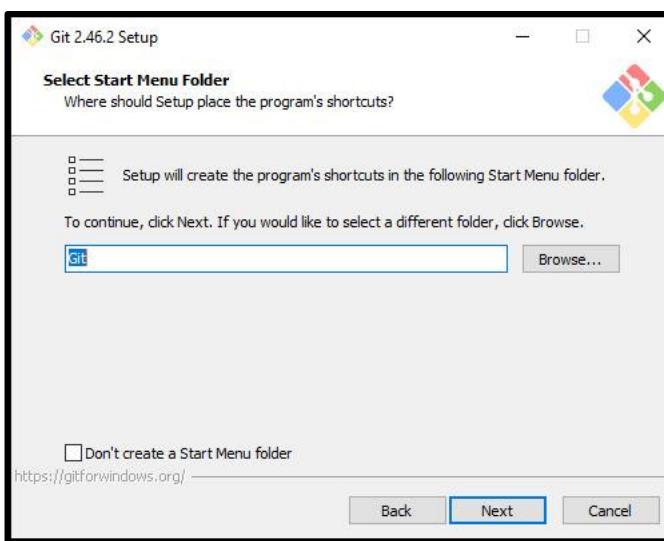
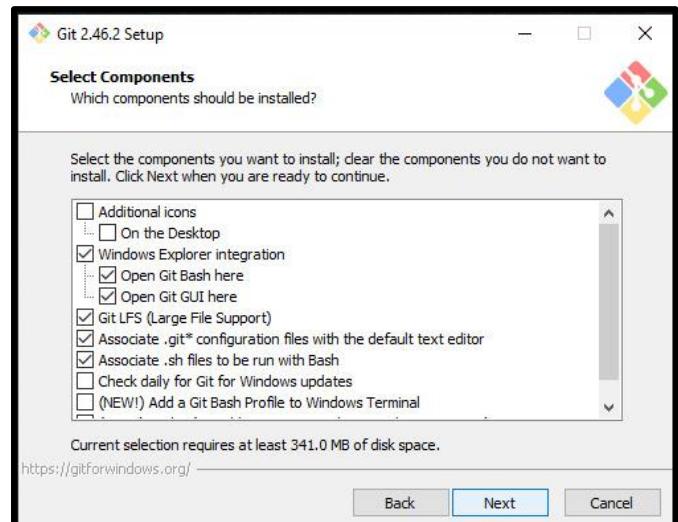
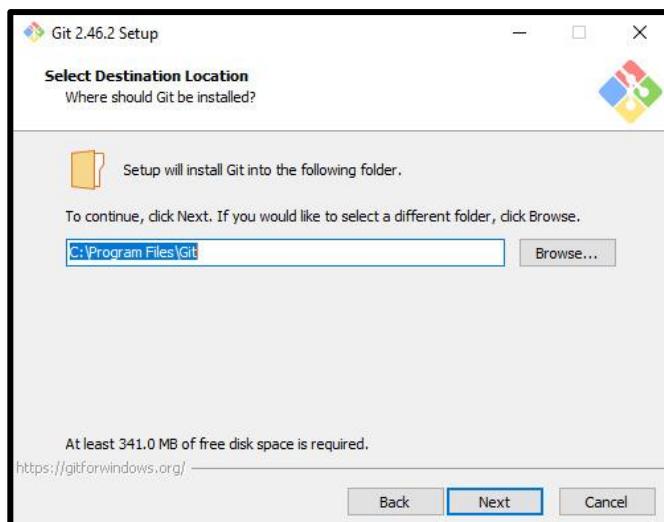
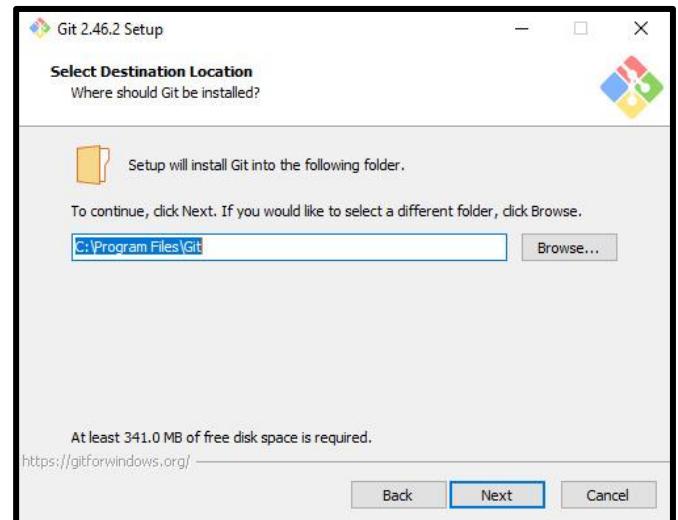
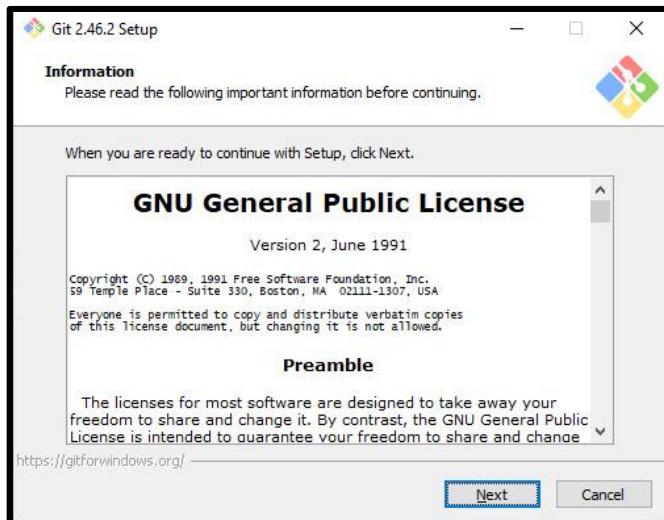


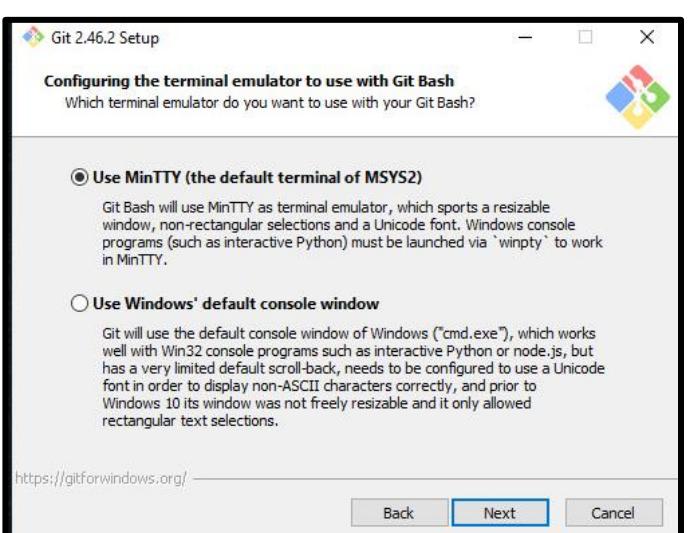
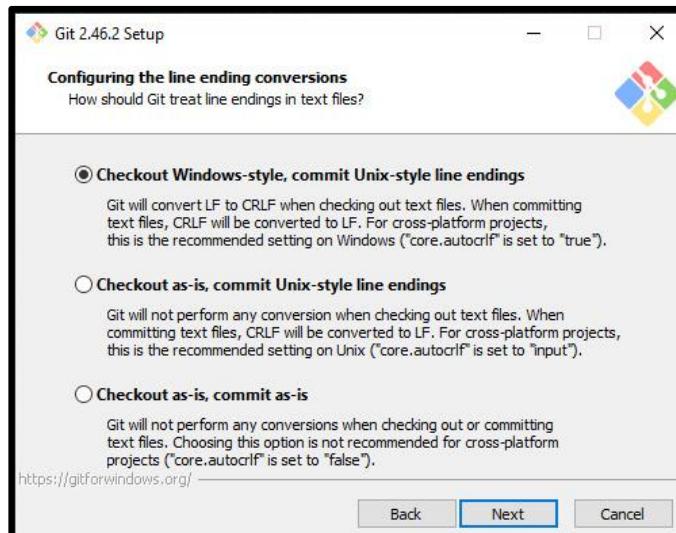
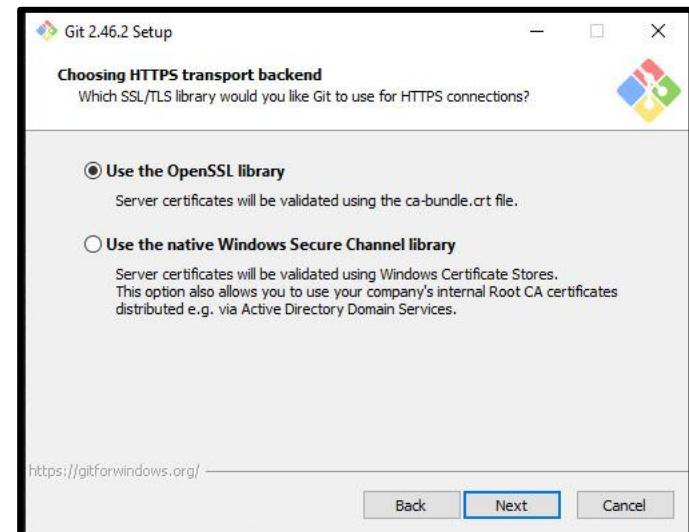
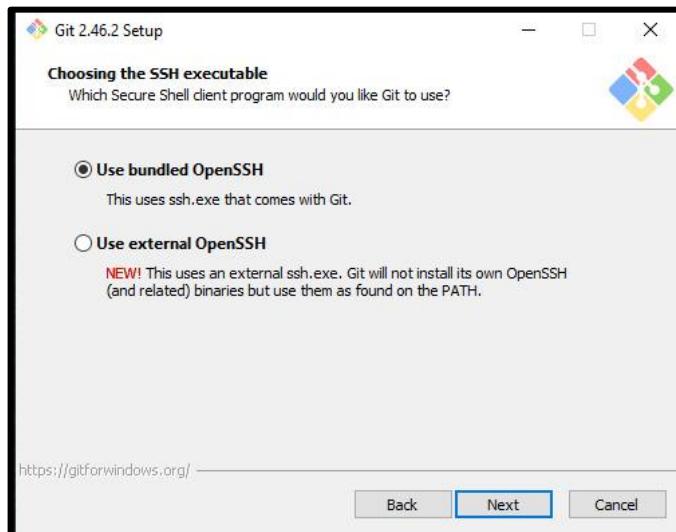
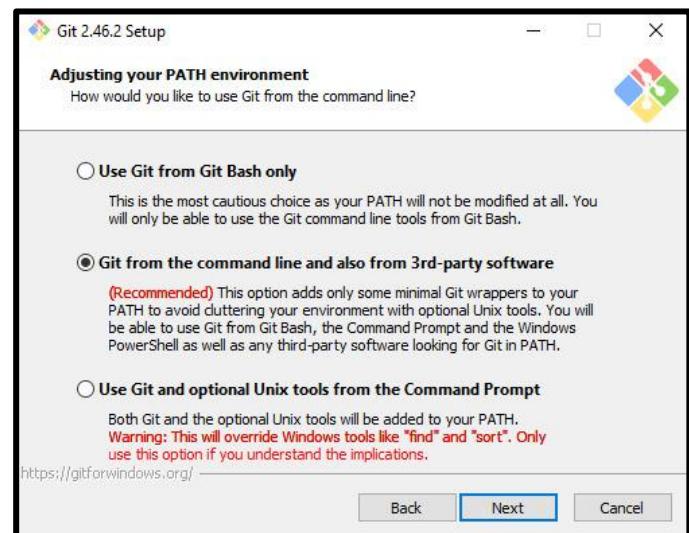
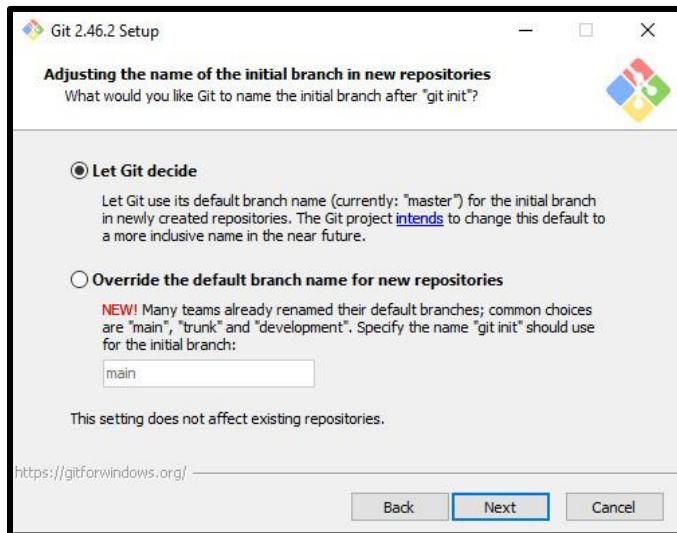
5. Git

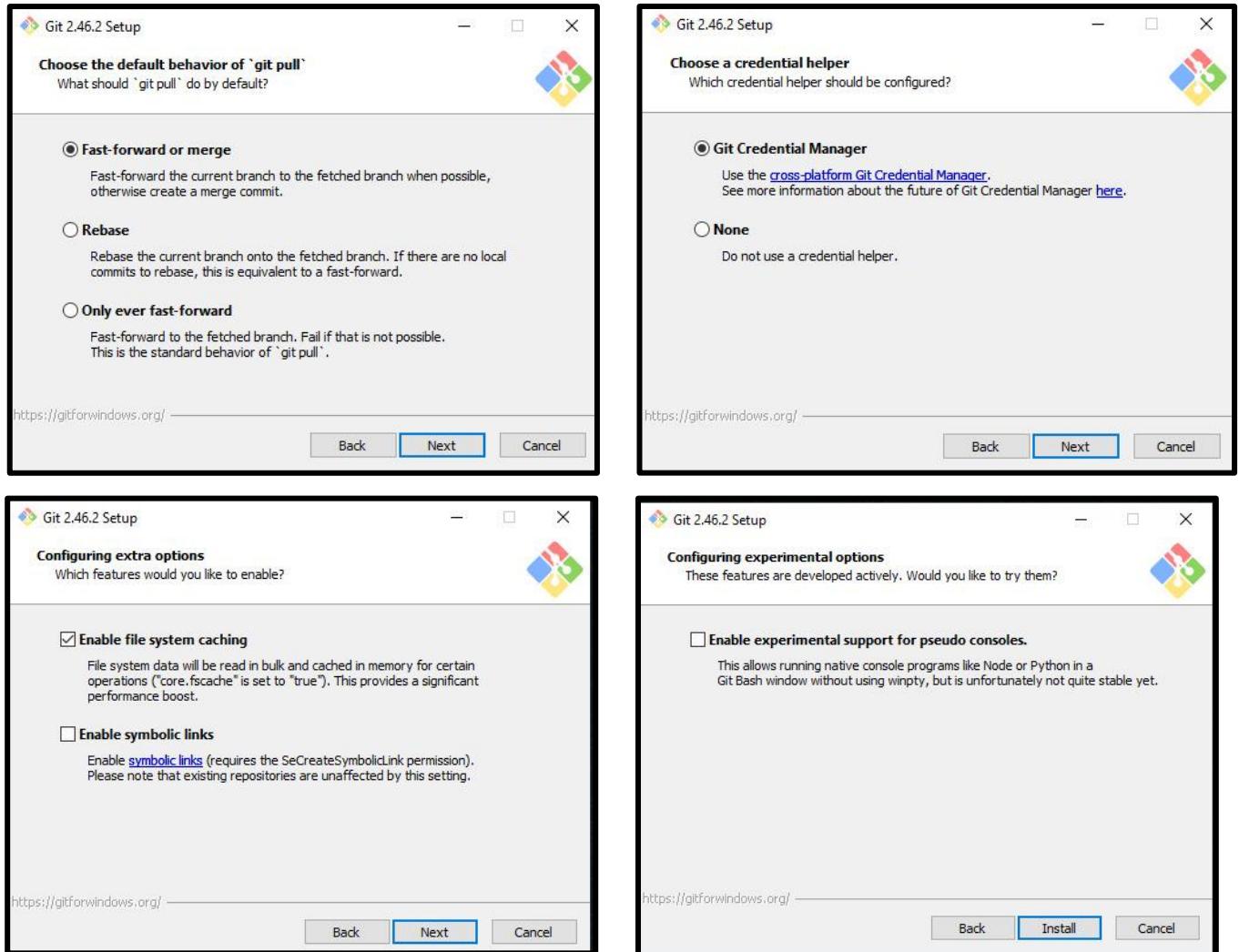
Para poder descargar e instalar git en nuestra computadora seguimos los siguientes pasos:

- Accedemos al sitio oficial de descarga de git en <https://git-scm.com/downloads> y descargamos el instalador.







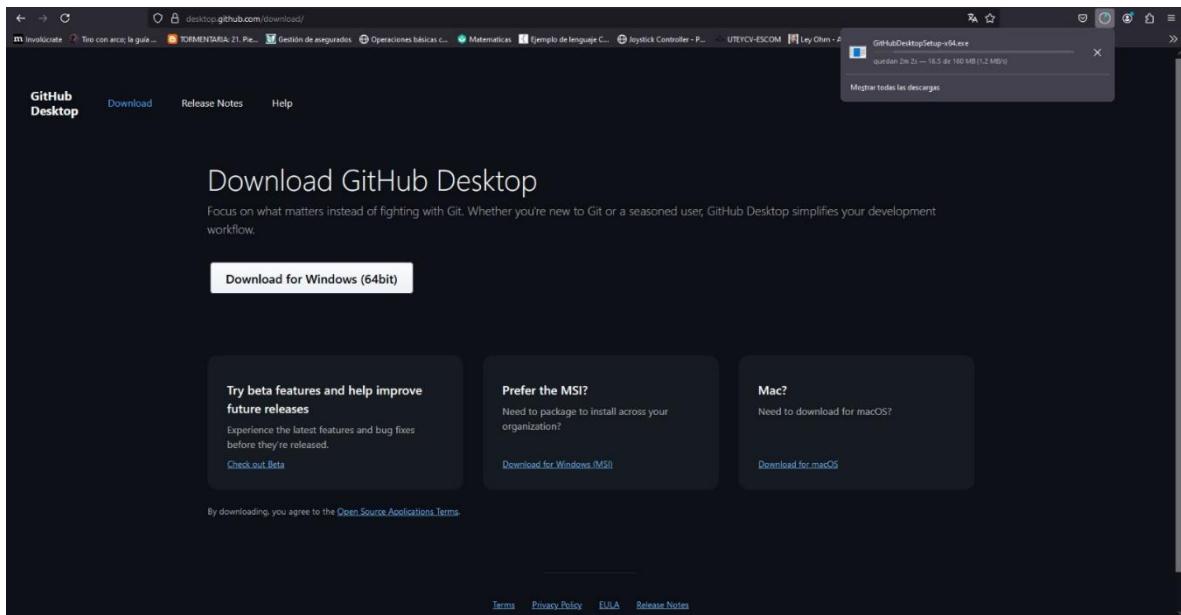


- Solo daremos en siguiente a lo largo de toda la instalación, y así tendrémos instalado nuestro git.

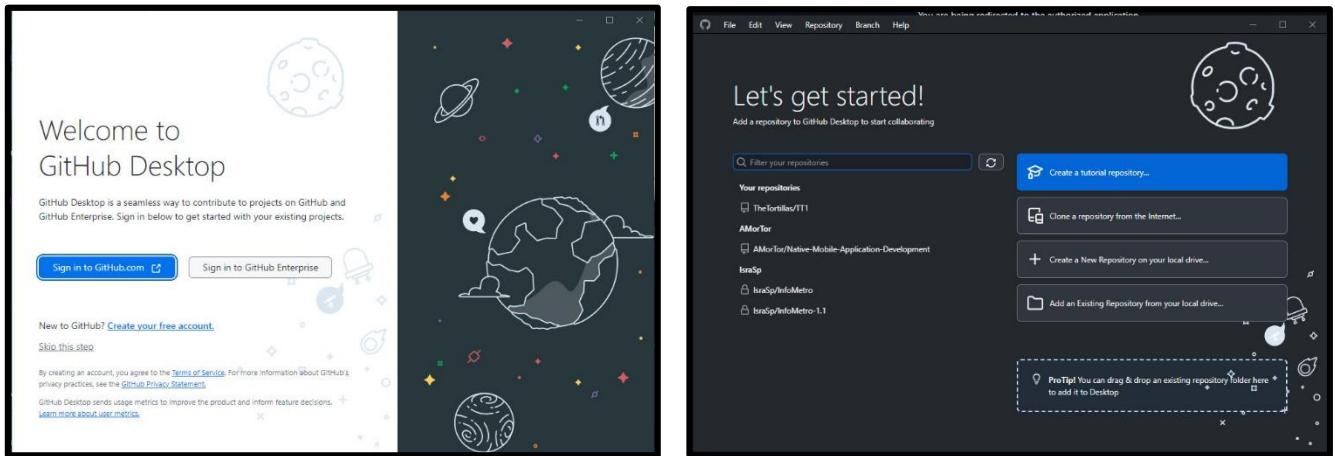
6. GitHub

Para poder descargar e instalar GitHub en nuestra computadora seguimos los siguientes pasos:

- Accedemos al sitio oficial de github desktop para descargarlo: <https://desktop.github.com/download/>



- Al ejecutarlo este se instalará automáticamente, solo nos pedirá nuestra cuenta de github y ya estaría hecho.



Instalación de herramientas para Linux:

1. Android Studio.

Para descargar Android Studio en Linux debemos seguir los siguientes pasos:

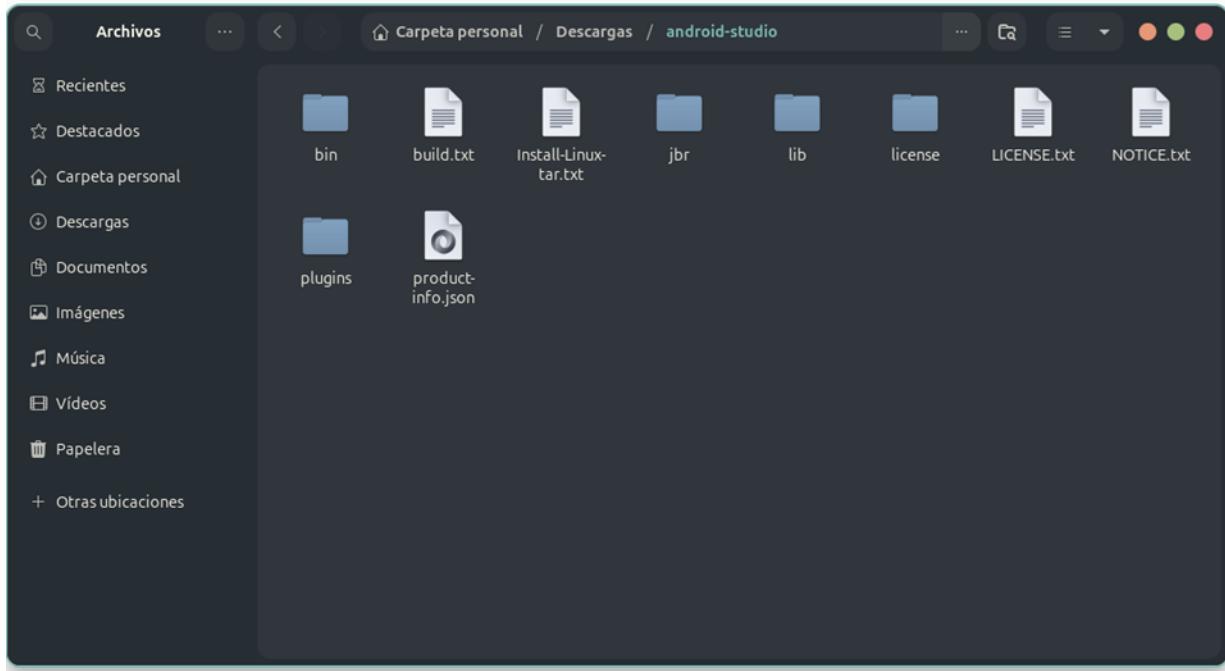
- Ingresamos a la pagina oficial, <https://developer.android.com/studio?hl=es-419> y nos dirigimos a la sección de descargas como se ve en la siguiente imagen. Y descargamos el archivo de Linux (64 bits).

Plataforma	Paquete de Android Studio	Tamaño	Suma de comprobación SHA-256
Windows (64 bits)	android-studio-2024.1.2.12-windows.exe Recomendada	1.2 GB	81aaaf2fa7a6ebbdd7c60eff7ab74dd6a18eeb1c942491c53a42b9efaf59054
Windows (64 bits)	android-studio-2024.1.2.12-windows.zip Sin instalador .exe	1.2 GB	50180dc87d045f4c9a6ba6aa920f8a06780bf27325a2676a5e232871d772f77e
Mac (64 bits)	android-studio-2024.1.2.12-mac.dmg	1.3 GB	1cc2919c25a544be63eace74575df334d1dd96bc86f98bf2cbfd5d9e61c540
Mac (64 bits, ARM)	android-studio-2024.1.2.12-mac_arm.dmg	1.2 GB	e25197654644d4614cfbcb477d0ca8e6f92055d1a2583a4ab7042783a8dc27d1
Linux (64 bits)	android-studio-2024.1.2.12-linux.tar.gz	1.2 GB	745168820e989a9085ff842d47ce541407db09df7b8ab20770f6ea89e41a5e92
ChromeOS	android-studio-2024.1.2.12-cros.deb	992.2 MB	0a01c7dfb3cae55a84987793829816191228a4c451e87aca0d0e7d0cc7763c6

Hay más descargas disponibles en descargar archivos Para Descargas de Android Emulator; consulta la Archivos de descarga del emulador.

Solo herramientas de línea de comandos

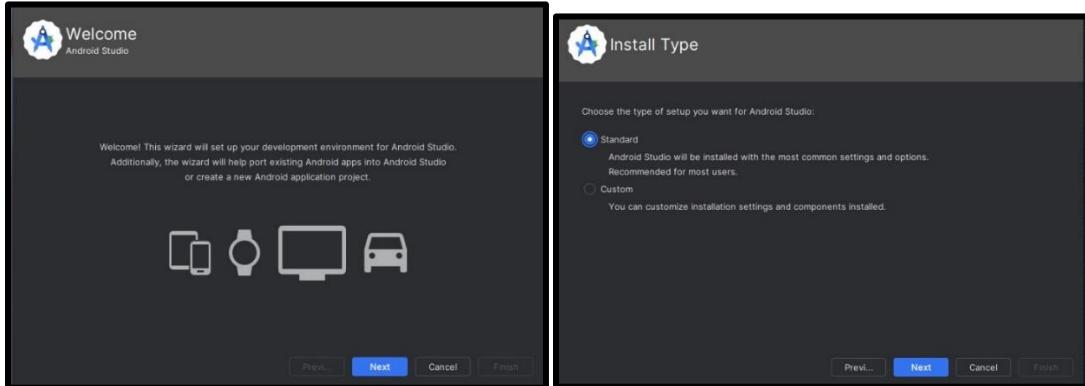
- Una vez descargado el archivo Linux (64 bits), descomprimir el archivo entrar a la carpeta, dentro de esta veremos varios archivos.

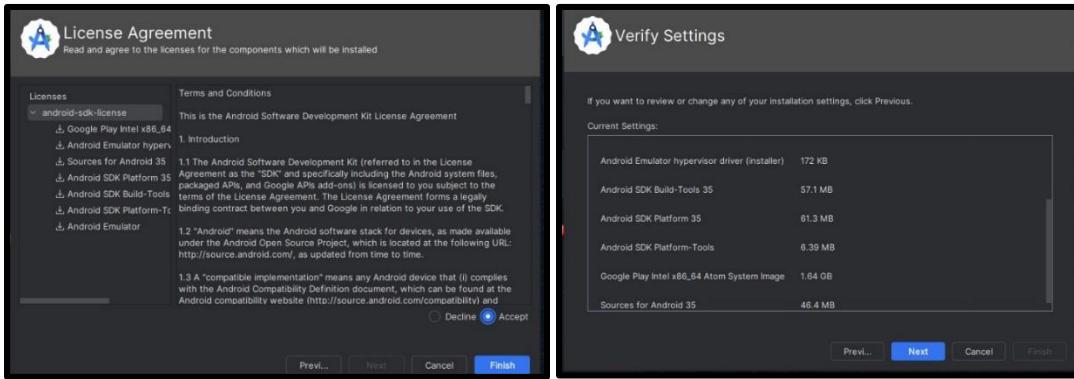


- Aquí leemos el archivo de `Install-Linux.txt` el cual contiene las instrucciones: Desempaquetá el archivo de distribución de Android Studio que descargaste en la ubicación donde deseas instalar el programa. Nos referiremos a esta ubicación como tu `{installation home}`. Para iniciar la aplicación, abre una consola, navega al directorio `"{installation home}/bin"` utilizando el comando `"cd"` y escribe: `"/studio.sh"`. Esto inicializará varios archivos de configuración en el directorio de configuración: `"~/.config/Google/AndroidStudio2024.1"`. Ejecutamos el primer comando, como se ve en la siguiente imagen.

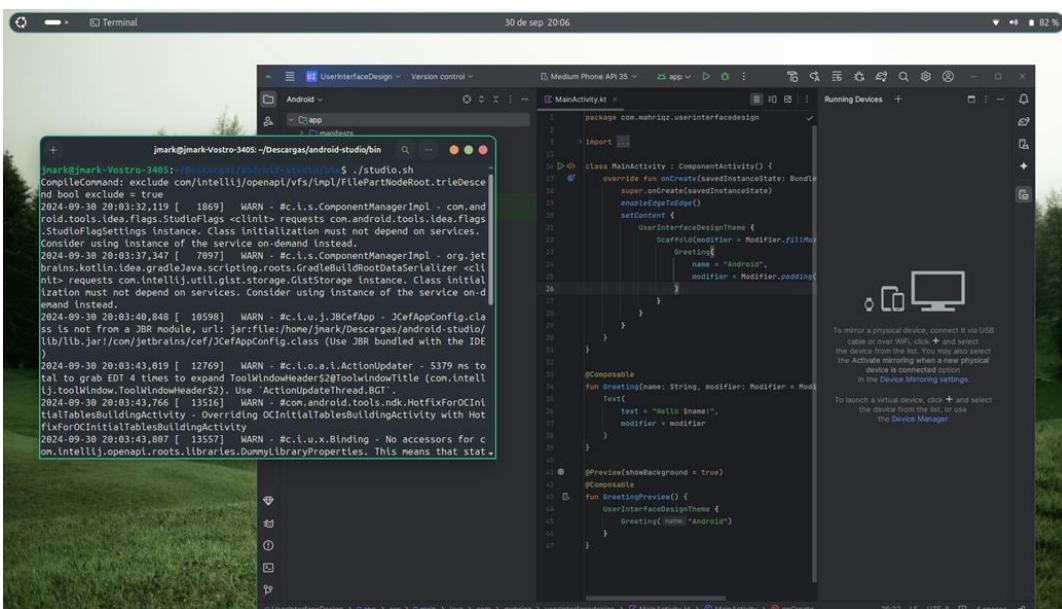
```
jmark@jmark-Vostro-3405: ~/Descargas/android-studio/bin$ ./studio.sh
CompileCommand: exclude com/intellij/openapi/vfs/impl/FilePartNodeRoot.trieDesce
nd bool exclude = true
2024-09-30 20:03:32,119 [ 1869]  WARN - #c.i.s.ComponentManagerImpl - com.and
roid.tools.idea.flags.StudioFlags <clinit> requests com.android.tools.idea.flags
.StudioFlagSettings instance. Class initialization must not depend on services.
Consider using instance of the service on-demand instead.
2024-09-30 20:03:37,347 [ 7097]  WARN - #c.i.s.ComponentManagerImpl - org.jet
brains.kotlin.idea.gradleJava.scripting.roots.GradleBuildRootDataSerializer <cli
nit> requests com.intellij.util.gist.storage.GistStorage instance. Class initial
ization must not depend on services. Consider using instance of the service on-d
emand instead.
2024-09-30 20:03:40,848 [ 10598]  WARN - #c.i.u.j.JBCefApp - JCefAppConfig.cla
ss is not from a JBR module, url: jar:file:/home/jmark/Descargas/android-studio/
lib/lib.jar!/com/jetbrains/cef/JCefAppConfig.class (Use JBR bundled with the IDE
)
2024-09-30 20:03:43,019 [ 12769]  WARN - #c.i.o.a.i.ActionUpdater - 5379 ms to
tal to grab EDT 4 times to expand ToolWindowHeader$2@ToolwindowTitle (com.intell
ij.toolWindow.ToolWindowHeader$2). Use `ActionUpdateThread.BGT`.
2024-09-30 20:03:43,766 [ 13516]  WARN - #com.android.tools.ndk.HotfixForOCIni
tialTablesBuildingActivity - Overriding OCInitialTablesBuildingActivity with Hot
fixForOCInitialTablesBuildingActivity
2024-09-30 20:03:43,807 [ 13557]  WARN - #c.i.u.x.Binding - No accessors for c
om.intellij.openapi.roots.libraries.DummyLibraryProperties. This means that stat.
```

- Al igual que Windows por primera vez se nos abrirá el instalador de Android Studio, después de esto solo es darle en la opción next y esperar a que termine la instalación una vez que esta termine.

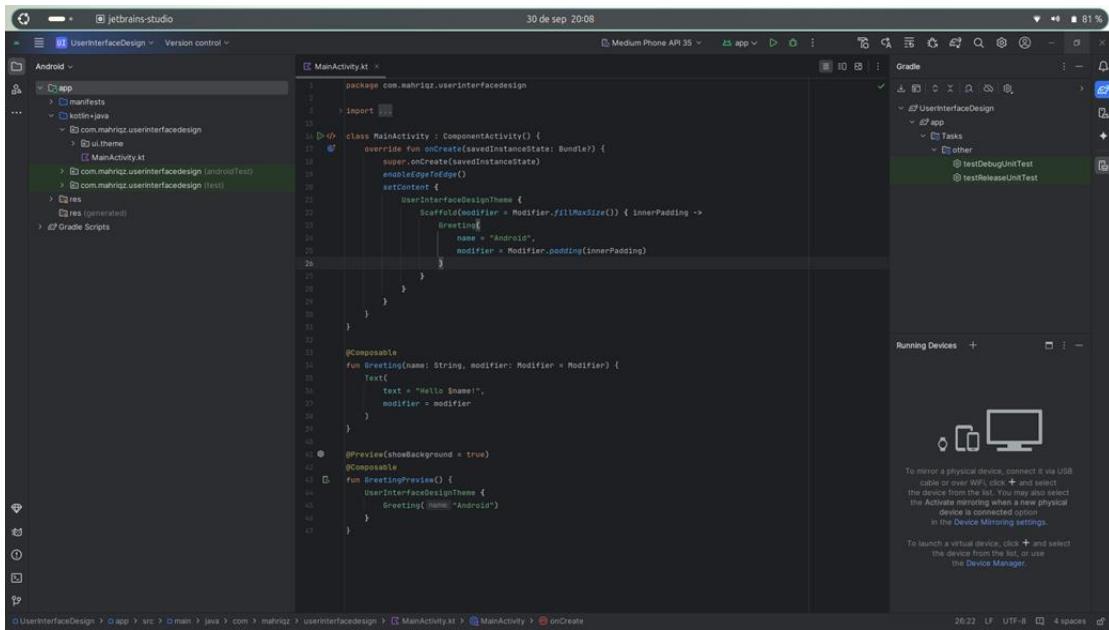




- Para abrir la aplicación de Android Studio es necesario ejecutar el mismo comando como se muestra a continuación:



- Para las dependencias de Gradle solo es necesario crear un nuevo proyecto en Android Studio y conectarse a internet para que estas se descarguen y configuren de manera automática como se ve a continuación.

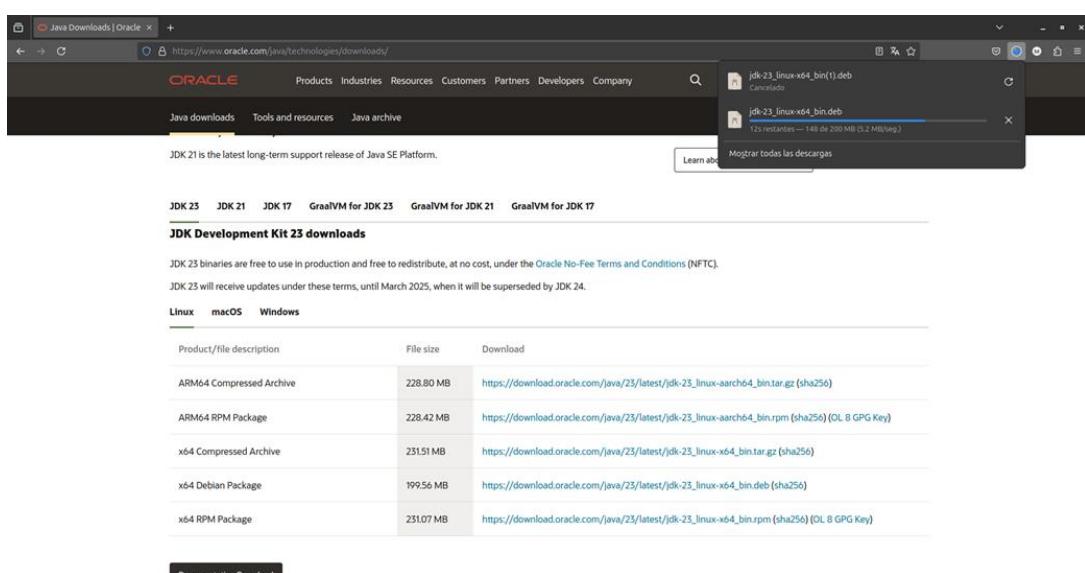


Con todo esto tendríamos listo nuestro Android Studio para el desarrollo y ejecución de programas.

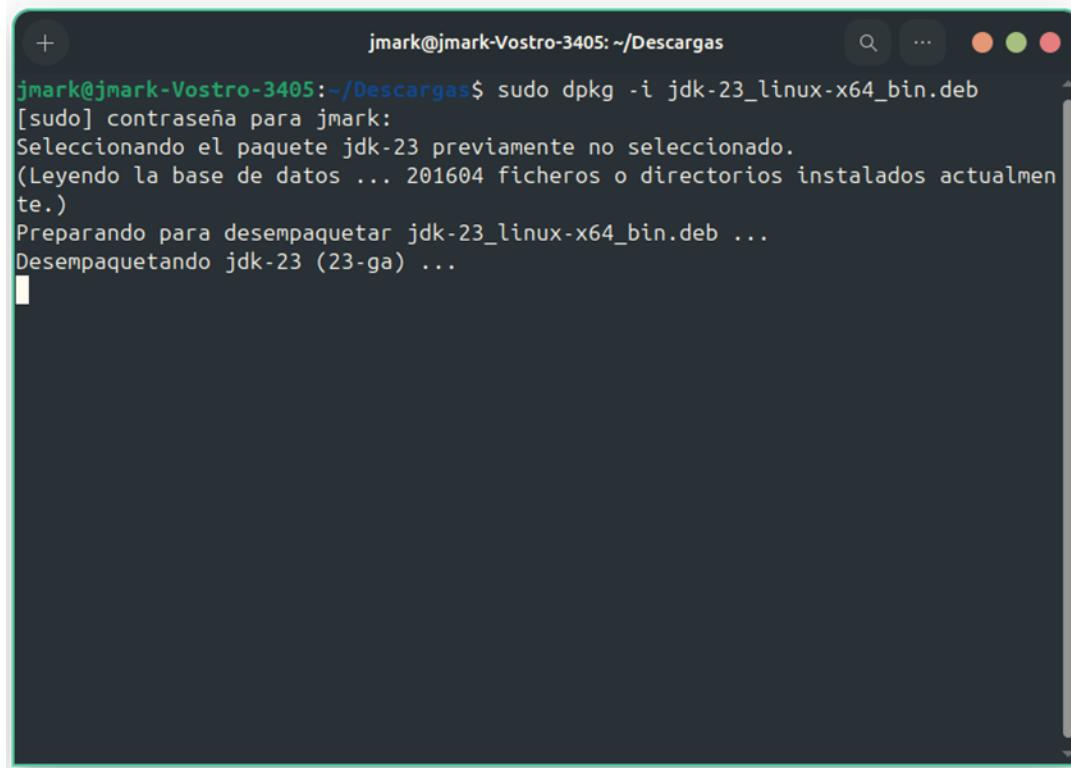
2. Java Development Kit (JDK).

Para asegurarnos que todo esto funcione de manera correcta es necesario establecer el JDK para la compilación y depuración de los archivos, para esto en nuestro caso descargaremos la versión de desarrolladores de Oracle para esto nos vamos al enlace <https://www.oracle.com/java/technologies/downloads/>

- Dentro del enlace descargamos la versión 23 para linux “.deb”, como se ve en la imagen.

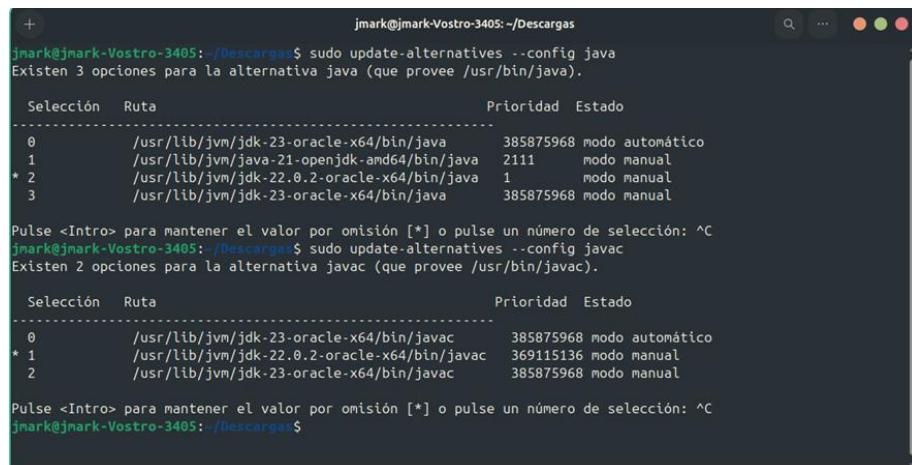


- Una vez descargado lo ejecutamos con el comando “sudo dpkg -i jdk-23_linux-x64_bin.deb” y esperamos a que termine la instalación, como se muestra en la imagen.



```
jmark@jmark-Vostro-3405:~/Descargas$ sudo dpkg -i jdk-23_linux-x64_bin.deb
[sudo] contraseña para jmark:
Seleccionando el paquete jdk-23 previamente no seleccionado.
(Leyendo la base de datos ... 201604 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar jdk-23_linux-x64_bin.deb ...
Desempaquetando jdk-23 (23-ga) ...
```

- Una vez instalado tenemos validar la instalación para esto utilizaremos los comandos, “sudo update-alternatives --install /usr/bin/java java /usr/lib/jvm/jdk-18/bin/java 1” y “sudo update-alternatives --install /usr/bin/javac javac /usr/lib/jvm/jdk-18/bin/javac 1” Como se ve en la siguiente imagen.

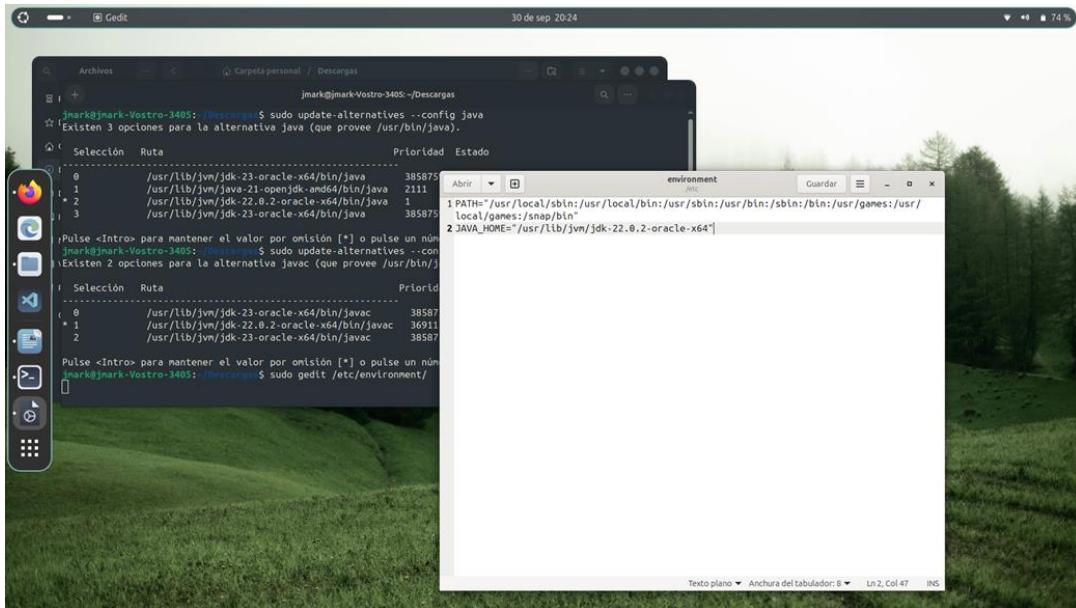


```
jmark@jmark-Vostro-3405:~/Descargas$ sudo update-alternatives --config java
Existen 3 opciones para la alternativa java (que provee /usr/bin/java).
      Selección     Ruta          Prioridad  Estado
      0           /usr/lib/jvm/jdk-23-oracle-x64/bin/java  385875968  modo automático
      1           /usr/lib/jvm/java-21-openjdk-amd64/bin/java  2111     modo manual
*  2           /usr/lib/jvm/jdk-22.0.2-oracle-x64/bin/java  1        modo manual
      3           /usr/lib/jvm/jdk-23-oracle-x64/bin/java  385875968  modo manual

Pulse <Intro> para mantener el valor por omisión [*] o pulse un número de selección: ^C
jmark@jmark-Vostro-3405:~/Descargas$ sudo update-alternatives --config javac
Existen 2 opciones para la alternativa javac (que provee /usr/bin/javac).
      Selección     Ruta          Prioridad  Estado
      0           /usr/lib/jvm/jdk-23-oracle-x64/bin/javac  385875968  modo automático
*  1           /usr/lib/jvm/jdk-22.0.2-oracle-x64/bin/javac  369115136  modo manual
      2           /usr/lib/jvm/jdk-23-oracle-x64/bin/javac  385875968  modo manual

Pulse <Intro> para mantener el valor por omisión [*] o pulse un número de selección: ^C
jmark@jmark-Vostro-3405:~/Descargas $
```

- Ahora que verificamos esta instalación, solo configuraremos la variable de entorno para que podamos ocuparla desde cualquier punto en terminal para esto ejecutamos el comando “sudo update-alternatives --config java”, Ahora abrimos el archivo de las variables de entorno a través del comando “sudo gedit /etc/environment/” y agregamos la variable de JAVA_HOME=”/usr/lib/jvm/jdk-23/”, como se ve en la imagen.



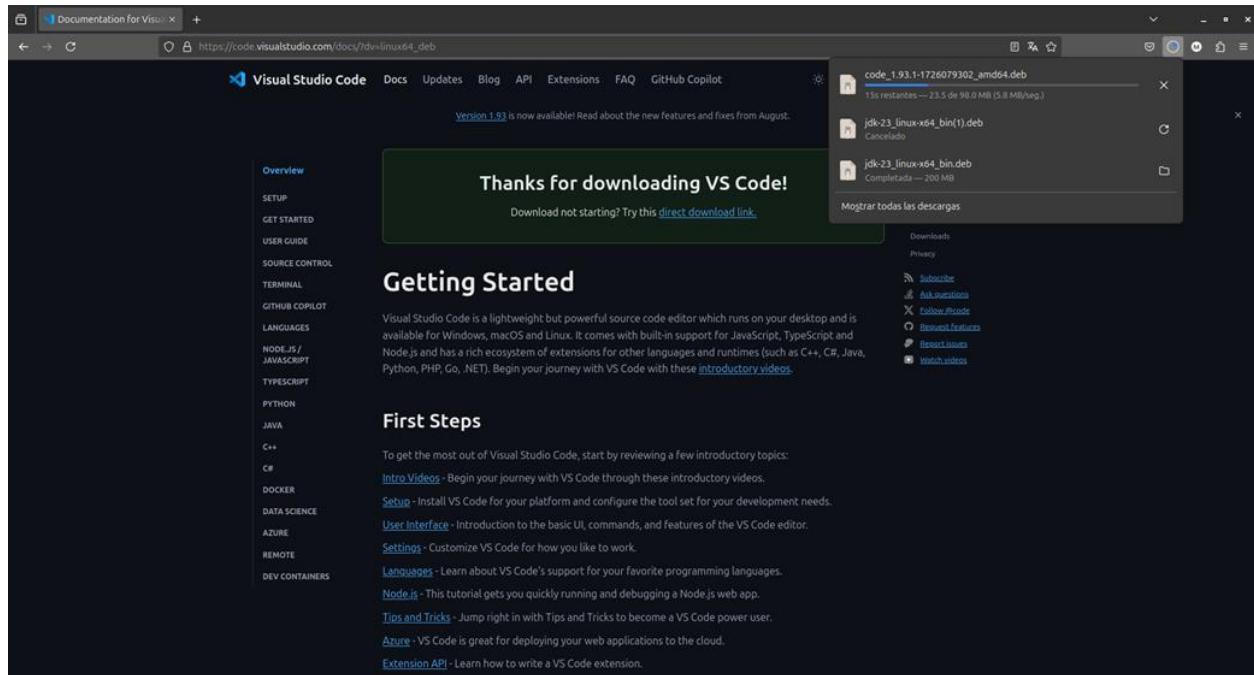
- Por último, actualizamos las variables de entorno con el comando “source /etc/environment/” y probamos si se instaló correctamente con los comandos “javac -version” como se ve a continuación.

```
jmark@jmark-Vostro-3405:~/Descargas$ javac -version
javac 22.0.2
jmark@jmark-Vostro-3405:~/Descargas$ java -version
java version "22.0.2" 2024-07-16
Java(TM) SE Runtime Environment (build 22.0.2+9-70)
Java HotSpot(TM) 64-Bit Server VM (build 22.0.2+9-70, mixed mode, sharing)
jmark@jmark-Vostro-3405:~/Descargas$
```

Ahora con esto terminamos la instalacion y configuracion del JDK en linux.

3. Visual Studio Code y SpringBoot.

Para visual code studio es un proceso muy similar nos vamos a la página oficial cuyo enlace es <https://code.visualstudio.com/Download> y descargamos la versión para Linux en .deb como se ve a continuación.

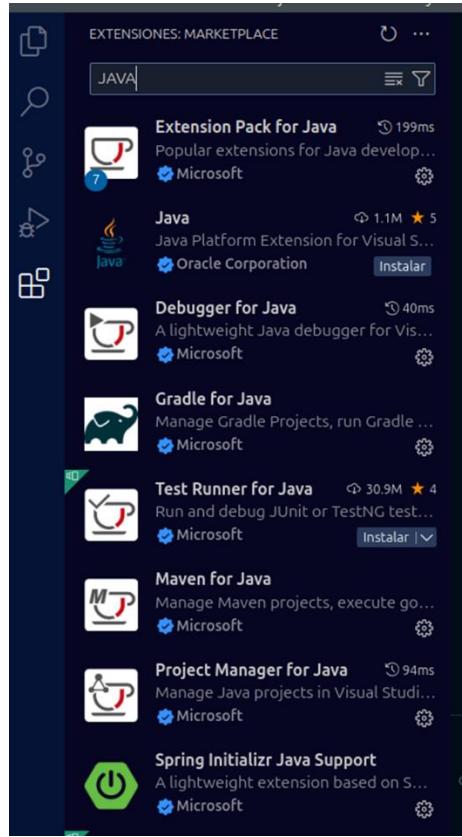


Para realizar la instalación de visual en Linux seguiremos los siguientes pasos:

- Para instalarlo utilizaremos el comando “`sudo dpkg -i code_1.93.1-1726079302_amd64.deb`” con esto validamos y procedemos a esperar que termine la instalación.

```
jmark@jmark-Vostro-3405: ~/Descargas
jmark@jmark-Vostro-3405: ~/Descargas$ sudo dpkg -i code_1.93.1-1726079302_amd64.deb
(Leyendo la base de datos ... 202093 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar code_1.93.1-1726079302_amd64.deb ...
Desempaquetando code (1.93.1-1726079302) sobre (1.93.1-1726079302) ...
Configurando code (1.93.1-1726079302) ...
Procesando disparadores para gnome-menus (3.36.0-1.1ubuntu3) ...
Procesando disparadores para desktop-file-utils (0.27-2build1) ...
Procesando disparadores para shared-mime-info (2.4-4) ...
jmark@jmark-Vostro-3405: ~/Descargas$
```

- Una vez terminado, abrimos el Visual Studio Code y nos vamos a la sección de extensiones, tendremos que instalar. Extension pack for java y Spring Initializr Java Support, como se muestra a continuación.



Ahora con estas herramientas instaladas podemos crear nuestro proyecto.

Para la creación del proyecto de Spring Boot desde Visual Studio Code seguiremos los siguientes pasos:

- Para esto desde nuestro Visual Studio Code ejecutamos la combinación de teclas **ctrl+shift+p** y se nos abrirá un buscador donde escogeremos el proyecto de java que queremos realizar.

```

AdministradorTraficoSSL.java - AdministradorTrafico - Visual Studio Code

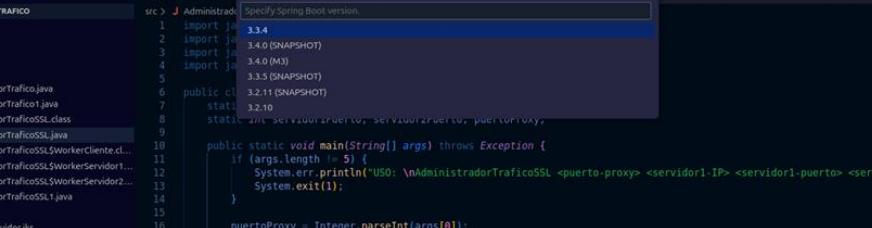
Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda

EXPLORADOR
ADMINISTRADORTRAFICO
> .vscode
> bin
> lib
src
J AdministradorTrafico.java
J AdministradorTrafico1.java
J AdministradorTraficoSSL.class
J AdministradorTraficoSSL.java
J AdministradorTraficoSSLWorkerCliente.class
J AdministradorTraficoSSLWorkerServidor1.class
J AdministradorTraficoSSLWorkerServidor2.class
J AdministradorTraficoSSL1.java
J Cliente.java
keystore_servidor.jks
J Proxy.java
ESQUEMA
LINEA DE TIEMPO
JAVA PROJECTS

src > J AdministradorTraficoSSL.java
1 .net.ssl.
2 io.*;
3 net.Socke
4 security.
5
6 Administ
7 tring ser
8 nt servid
9
10 tatic voi
11 args.leng
12 System.er
13 System.ex
14
15 toProxy =
16 ido1Host = args[1];
17 ido1Puerto = Integer.parseInt(args[2]);
18 ido2Host = args[3];
19 ido2Puerto = Integer.parseInt(args[4]);
20
21 establecer propiedades del keystore
22 em.setProperty("javax.net.ssl.keyStore", "keystore_servidor.jks"); // Cambia a la ruta completa si es necesario
23 em.setProperty("javax.net.ssl.keyStorePassword", "1234567");
24
25
26 obtener la fábrica de sockets SSL
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
287
288
289
289
290
291
292
293
294
295
296
297
297
298
299
299
300
301
302
303
304
305
306
307
308
309
309
310
311
312
313
314
315
315
316
317
317
318
319
319
320
321
321
322
323
323
324
324
325
325
326
326
327
327
328
328
329
329
330
330
331
331
332
332
333
333
334
334
335
335
336
336
337
337
338
338
339
339
340
340
341
341
342
342
343
343
344
344
345
345
346
346
347
347
348
348
349
349
350
350
351
351
352
352
353
353
354
354
355
355
356
356
357
357
358
358
359
359
360
360
361
361
362
362
363
363
364
364
365
365
366
366
367
367
368
368
369
369
370
370
371
371
372
372
373
373
374
374
375
375
376
376
377
377
378
378
379
379
380
380
381
381
382
382
383
383
384
384
385
385
386
386
387
387
388
388
389
389
390
390
391
391
392
392
393
393
394
394
395
395
396
396
397
397
398
398
399
399
400
400
401
401
402
402
403
403
404
404
405
405
406
406
407
407
408
408
409
409
410
410
411
411
412
412
413
413
414
414
415
415
416
416
417
417
418
418
419
419
420
420
421
421
422
422
423
423
424
424
425
425
426
426
427
427
428
428
429
429
430
430
431
431
432
432
433
433
434
434
435
435
436
436
437
437
438
438
439
439
440
440
441
441
442
442
443
443
444
444
445
445
446
446
447
447
448
448
449
449
450
450
451
451
452
452
453
453
454
454
455
455
456
456
457
457
458
458
459
459
460
460
461
461
462
462
463
463
464
464
465
465
466
466
467
467
468
468
469
469
470
470
471
471
472
472
473
473
474
474
475
475
476
476
477
477
478
478
479
479
480
480
481
481
482
482
483
483
484
484
485
485
486
486
487
487
488
488
489
489
490
490
491
491
492
492
493
493
494
494
495
495
496
496
497
497
498
498
499
499
500
500
501
501
502
502
503
503
504
504
505
505
506
506
507
507
508
508
509
509
510
510
511
511
512
512
513
513
514
514
515
515
516
516
517
517
518
518
519
519
520
520
521
521
522
522
523
523
524
524
525
525
526
526
527
527
528
528
529
529
530
530
531
531
532
532
533
533
534
534
535
535
536
536
537
537
538
538
539
539
540
540
541
541
542
542
543
543
544
544
545
545
546
546
547
547
548
548
549
549
550
550
551
551
552
552
553
553
554
554
555
555
556
556
557
557
558
558
559
559
560
560
561
561
562
562
563
563
564
564
565
565
566
566
567
567
568
568
569
569
570
570
571
571
572
572
573
573
574
574
575
575
576
576
577
577
578
578
579
579
580
580
581
581
582
582
583
583
584
584
585
585
586
586
587
587
588
588
589
589
590
590
591
591
592
592
593
593
594
594
595
595
596
596
597
597
598
598
599
599
600
600
601
601
602
602
603
603
604
604
605
605
606
606
607
607
608
608
609
609
610
610
611
611
612
612
613
613
614
614
615
615
616
616
617
617
618
618
619
619
620
620
621
621
622
622
623
623
624
624
625
625
626
626
627
627
628
628
629
629
630
630
631
631
632
632
633
633
634
634
635
635
636
636
637
637
638
638
639
639
640
640
641
641
642
642
643
643
644
644
645
645
646
646
647
647
648
648
649
649
650
650
651
651
652
652
653
653
654
654
655
655
656
656
657
657
658
658
659
659
660
660
661
661
662
662
663
663
664
664
665
665
666
666
667
667
668
668
669
669
670
670
671
671
672
672
673
673
674
674
675
675
676
676
677
677
678
678
679
679
680
680
681
681
682
682
683
683
684
684
685
685
686
686
687
687
688
688
689
689
690
690
691
691
692
692
693
693
694
694
695
695
696
696
697
697
698
698
699
699
700
700
701
701
702
702
703
703
704
704
705
705
706
706
707
707
708
708
709
709
710
710
711
711
712
712
713
713
714
714
715
715
716
716
717
717
718
718
719
719
720
720
721
721
722
722
723
723
724
724
725
725
726
726
727
727
728
728
729
729
730
730
731
731
732
732
733
733
734
734
735
735
736
736
737
737
738
738
739
739
740
740
741
741
742
742
743
743
744
744
745
745
746
746
747
747
748
748
749
749
750
750
751
751
752
752
753
753
754
754
755
755
756
756
757
757
758
758
759
759
760
760
761
761
762
762
763
763
764
764
765
765
766
766
767
767
768
768
769
769
770
770
771
771
772
772
773
773
774
774
775
775
776
776
777
777
778
778
779
779
780
780
781
781
782
782
783
783
784
784
785
785
786
786
787
787
788
788
789
789
790
790
791
791
792
792
793
793
794
794
795
795
796
796
797
797
798
798
799
799
800
800
801
801
802
802
803
803
804
804
805
805
806
806
807
807
808
808
809
809
810
810
811
811
812
812
813
813
814
814
815
815
816
816
817
817
818
818
819
819
820
820
821
821
822
822
823
823
824
824
825
825
826
826
827
827
828
828
829
829
830
830
831
831
832
832
833
833
834
834
835
835
836
836
837
837
838
838
839
839
840
840
841
841
842
842
843
843
844
844
845
845
846
846
847
847
848
848
849
849
850
850
851
851
852
852
853
853
854
854
855
855
856
856
857
857
858
858
859
859
860
860
861
861
862
862
863
863
864
864
865
865
866
866
867
867
868
868
869
869
870
870
871
871
872
872
873
873
874
874
875
875
876
876
877
877
878
878
879
879
880
880
881
881
882
882
883
883
884
884
885
885
886
886
887
887
888
888
889
889
890
890
891
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
900
901
901
902
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
1372
1373
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1380
1381
1381
1382
1382
1383
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1390
1391
1391
1392
1392
1393
1393
1394
1394
1395
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1400
1401
1401
1402
1402
1403
1403
1404
1404
1405
1405
1406
1406
1407
1407
1408
1408
1409
1409
1410
1410
1411
1411
1412
1412
1413
1413
1414
1414
1415
1415
1416
1416
1417
1417
1418
1418
1419
1419
1420
1420
1421
1421
1422
1422
1423
1423
1424
1424
1425
1425
1426
1426
1427
1427
1428
1428
1429
1429
1430
1430
1431
1431
1432

```

- Nos preguntara la versión.



AdministratorTraficoSSL.java - AdministradorTrafico - Visual Studio Code

Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda

EXPLORADOR

ADMINISTRADORTRAFICO

- vscode
- bin
- lib
- src
- AdministradorTrafico.java
- AdministradorTrafico1.java
- AdministradorTraficoSSL.class
- AdministradorTraficoSSL.java
- AdministradorTraficoSSLWorkerCliente.class
- AdministradorTraficoSSLWorkerServidor1.class
- AdministradorTraficoSSLWorkerServidor2.class
- AdministradorTraficoSSL1.java
- Cliente.java
- keystore_servidor.jks
- Proxy.java

ESQUEMA

LÍNEA DE TIEMPO

JAVA PROJECTS

Spring Initializer: Specify Spring Boot version

3.3.4

3.4.0 (SNAPSHOT)

3.4.0 (M3)

3.5 (SNAPSHOT)

3.2.11 (SNAPSHOT)

3.2.10

3.2.10

```
public class AdministradorTrafico {
    static String puertoProxy;
    static String servidor1Host;
    static int servidor1Puerto;
    static String servidor2Host;
    static int servidor2Puerto;
    static String keystorePath;
    static String keystorePassword;

    public static void main(String[] args) throws Exception {
        if (args.length != 5) {
            System.err.println("USO: AdministradorTraficoSSL <puerto-proxy> <servidor1-IP> <servidor1-puerto> <servidor2-IP> <servidor2-puerto>");
            System.exit(1);
        }

        puertoProxy = Integer.parseInt(args[0]);
        servidor1Host = args[1];
        servidor1Puerto = Integer.parseInt(args[2]);
        servidor2Host = args[3];
        servidor2Puerto = Integer.parseInt(args[4]);

        // Establecer propiedades del keystore
        System.setProperty("javax.net.ssl.KeyStore", "keystore_servidor.jks"); // Cambia a la ruta completa si es necesario
        System.setProperty("javax.net.ssl.KeyStorePassword", "1234567");

        // Obtener la fábrica de sockets SSL
        SSLContext sslContext = SSLContext.getInstance("TLS");
        sslContext.init(null, null, null);
        HttpsURLConnection.setDefaultSSLSocketFactory(sslContext.getSocketFactory());
    }
}
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS

jmark@jmark-Vostro-3405:~/Documentos/Distribuidos/Practical1/AdministradorTraficos

Líne 25 col 1 Espacio 4 UTF-8 LF Java

- Ahora el lenguaje de programación a escoger.

- El nombre del paquete.



AdministradorTraficoSSL.java - AdministradorTrafico - Visual Studio Code

Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda

EXPLORADOR

ADMINISTRADORTRAFICO

- vscode
- bin
- lib
- src

J AdministradorTrafico.java
J AdministradorTrafico1.java
J AdministradorTraficoSSLClass
J AdministradorTraficoSSL.java
J AdministradorTraficoSSLWorkerCliente.java
J AdministradorTraficoSSLWorkerServidor1.java
J AdministradorTraficoSSLWorkerServidor1.java
J AdministradorTraficoSSLWorkerServidor2.java
J AdministradorTraficoSSLWorkerServidor2.java
J Cliente.java
keystore_servidores.java
Proxy.java

ESQUEMA

LÍNEA DE TIEMPO

JAVA PROJECTS

AdministradorTrafico < Spring Initializr: Input Group Id

```
src > J AdministradorTraficoSSL.java
1 import java.io.*;
2 import java.net.*;
3 import java.net.Socket;
4 import java.security.KeyStore;
5
6 public class AdministradorTraficoSSL {
7     static String servidor1Host, servidor2Host;
8     static int servidor1Puerto, servidor2Puerto, puertoProxy;
9
10    public static void main(String[] args) throws Exception {
11        if (args.length == 5) {
12            System.out.println("USO: <puerto-proxy> <servidor1-IP> <servidor1-puerto> <servidor2-IP> <servidor2-puerto>");
13            System.exit(1);
14        }
15
16        puertoProxy = Integer.parseInt(args[0]);
17        servidor1Host = args[1];
18        servidor1Puerto = Integer.parseInt(args[2]);
19        servidor1Host = args[3];
20        servidor2Puerto = Integer.parseInt(args[4]);
21
22        // Establecer propiedades del keystore
23        System.setProperty("javax.net.ssl.KeyStore", "keystore_servidor.jks"); // Cambia a la ruta completa si es necesario
24        System.setProperty("javax.net.ssl.KeyStorePassword", "12345678");
25
26        // Obtener la fábrica de sockets SSL
27        SSLContext context = SSLContext.getInstance("TLS");
28        context.init(null, null, null);
29
30        // Crear un socket para el puerto proxy
31        ServerSocket serverSocket = new ServerSocket(puertoProxy);
32
33        // Aceptar conexiones de los servidores
34        Socket servidor1Socket = serverSocket.accept();
35        Socket servidor2Socket = serverSocket.accept();
36
37        // Crear un socket para el puerto proxy
38        Socket proxySocket = new Socket(servidor1Host, servidor1Puerto);
39
40        // Leer datos del servidor 1
41        BufferedReader reader = new BufferedReader(new InputStreamReader(proxySocket.getInputStream()));
42
43        // Escribir datos al servidor 2
44        BufferedWriter writer = new BufferedWriter(new OutputStreamWriter(proxySocket.getOutputStream()));
45
46        // Cerrar los sockets
47        proxySocket.close();
48        servidor1Socket.close();
49        servidor2Socket.close();
50    }
51}
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS

jmark@jmark-Vostro-3405:~/Documentos/Distribuidos/Practical/AdministradorTrafico\$

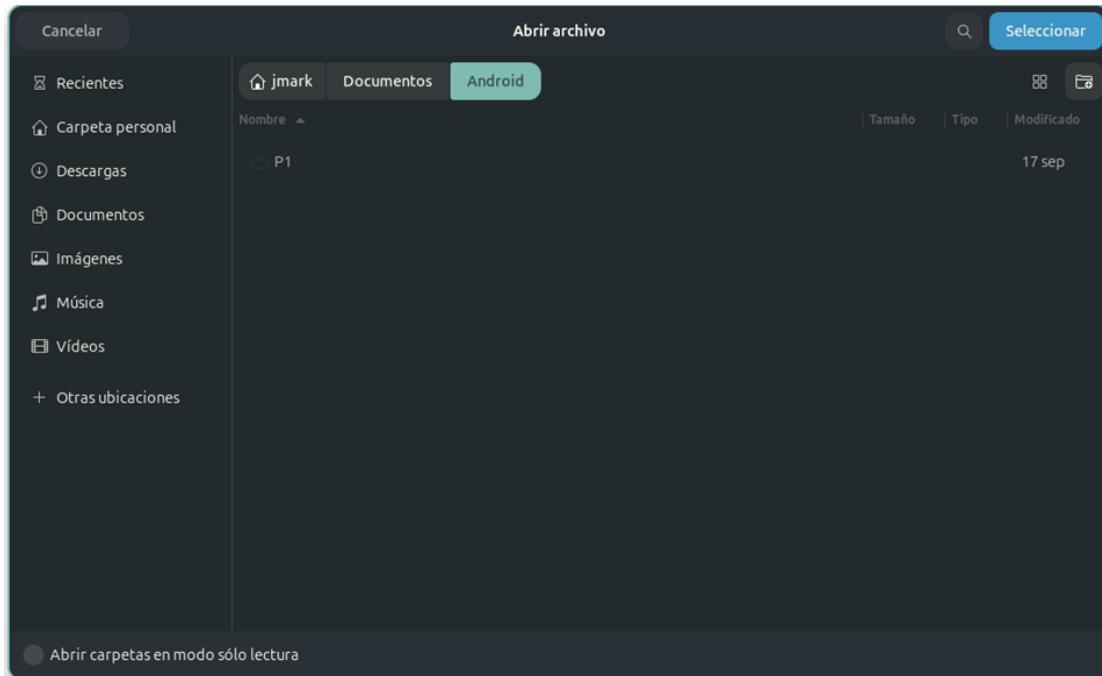
- Nombre del proyecto o clase a realizar.

- Versión del paquete jar o war.

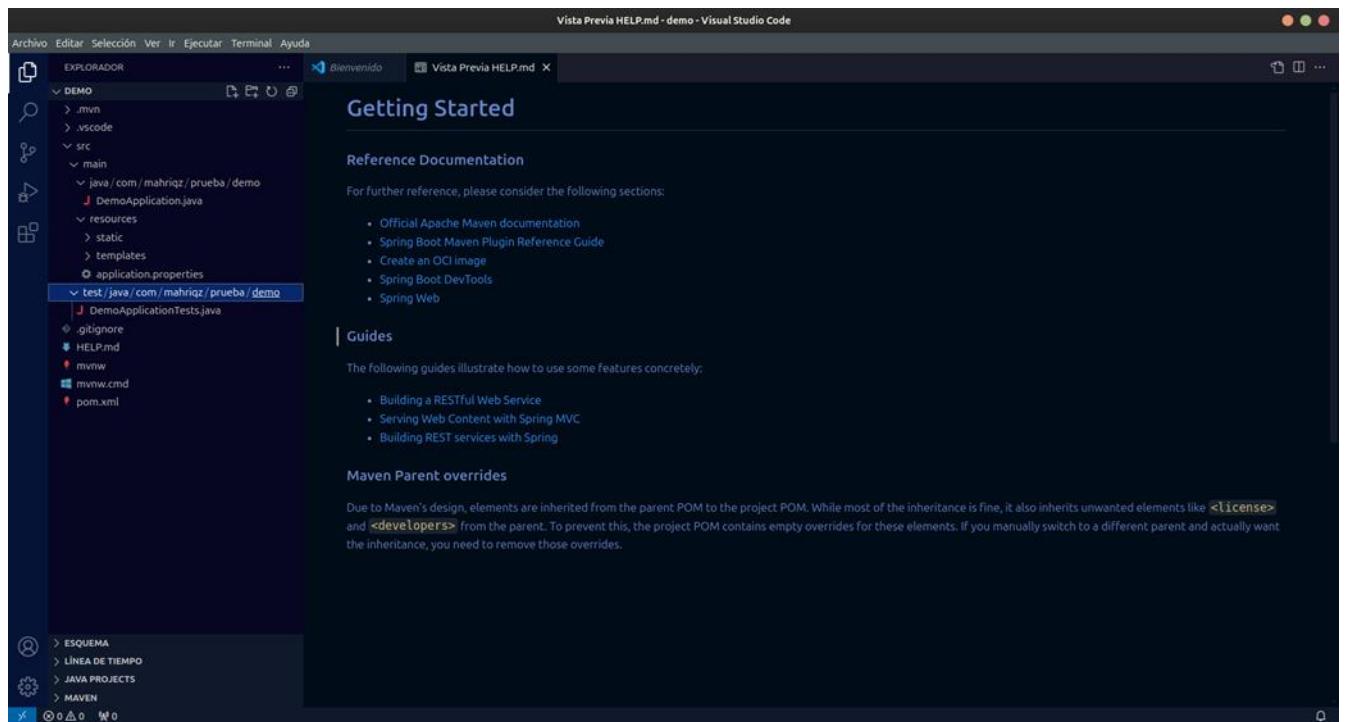
- Versión de Java a ocupar.

- Dependencias de Spring Boot.

- Carpeta donde guardar el proyecto de Spring Boot.



- Abrimos el proyecto y verificamos los archivos de Spring Boot.



Con esto hemos terminado de crear nuestro proyecto de Spring Boot, desde el mismo Visual Studio Code.

4. Github Desktop.

Ahora para la instalación de GitHub ejecutamos los siguientes comandos:

Descarga el paquete de @shiftkey ejecutando el siguiente comando:

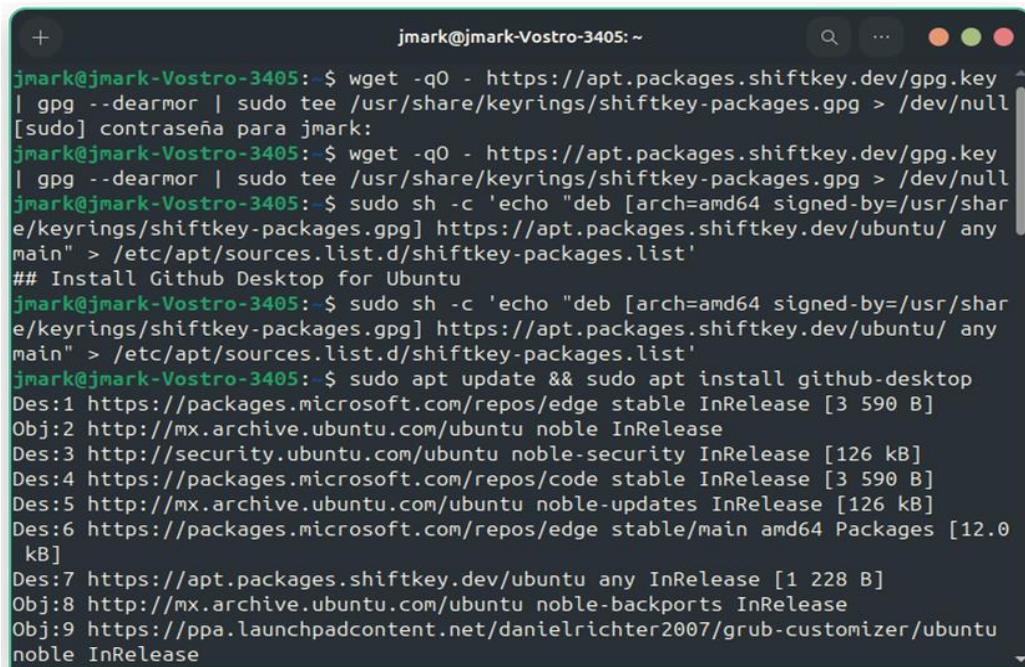
```
"wget -qO - https://apt.packages.shiftkey.dev/gpg.key | gpg --dearmor | sudo tee /usr/share/keyrings/shiftkey-packages.gpg > /dev/null".
```

Luego, añade el repositorio correspondiente con el comando:

```
"sudo sh -c 'echo "deb [arch=amd64 signed-by=/usr/share/keyrings/shiftkey-packages.gpg] https://apt.packages.shiftkey.dev/ubuntu/ any main" > /etc/apt/sources.list.d/shiftkey-packages.list'".
```

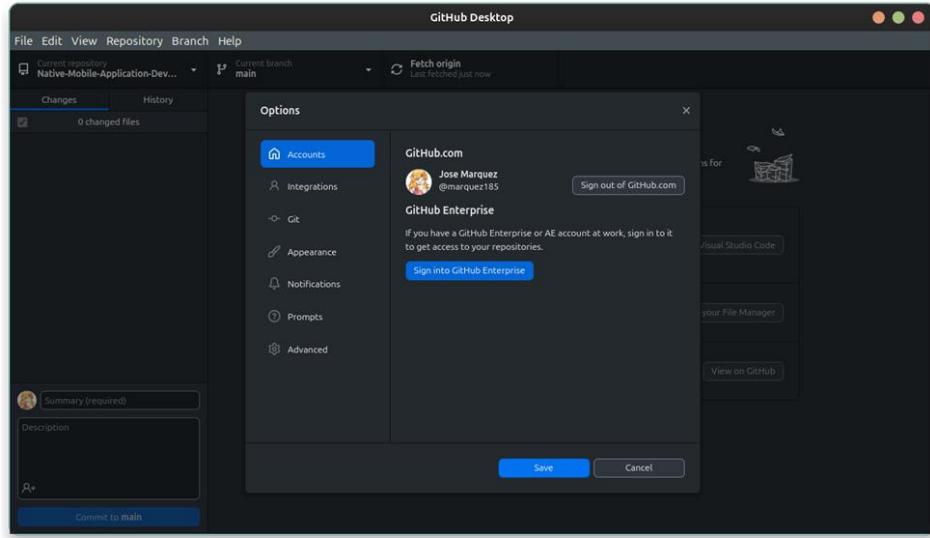
Para instalar GitHub Desktop en Ubuntu, actualiza los paquetes del sistema e instala la aplicación ejecutando:

```
"sudo apt update && sudo apt install github-desktop".
```

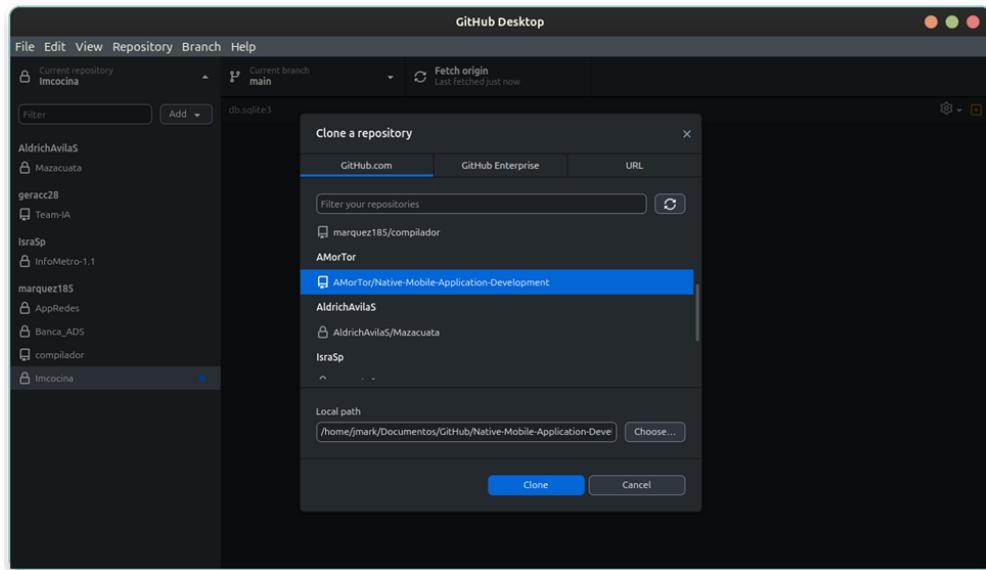


```
jmark@jmark-Vostro-3405:~$ wget -qO - https://apt.packages.shiftkey.dev/gpg.key | gpg --dearmor | sudo tee /usr/share/keyrings/shiftkey-packages.gpg > /dev/null
[sudo] contraseña para jmark:
jmark@jmark-Vostro-3405:~$ wget -qO - https://apt.packages.shiftkey.dev/gpg.key | gpg --dearmor | sudo tee /usr/share/keyrings/shiftkey-packages.gpg > /dev/null
jmark@jmark-Vostro-3405:~$ sudo sh -c 'echo "deb [arch=amd64 signed-by=/usr/share/keyrings/shiftkey-packages.gpg] https://apt.packages.shiftkey.dev/ubuntu/ any main" > /etc/apt/sources.list.d/shiftkey-packages.list'
## Install Github Desktop for Ubuntu
jmark@jmark-Vostro-3405:~$ sudo sh -c 'echo "deb [arch=amd64 signed-by=/usr/share/keyrings/shiftkey-packages.gpg] https://apt.packages.shiftkey.dev/ubuntu/ any main" > /etc/apt/sources.list.d/shiftkey-packages.list'
jmark@jmark-Vostro-3405:~$ sudo apt update && sudo apt install github-desktop
Des:1 https://packages.microsoft.com/repos/edge stable InRelease [3 590 B]
Obj:2 http://mx.archive.ubuntu.com/ubuntu noble InRelease
Des:3 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Des:4 https://packages.microsoft.com/repos/code stable InRelease [3 590 B]
Des:5 http://mx.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Des:6 https://packages.microsoft.com/repos/edge stable/main amd64 Packages [12.0 kB]
Des:7 https://apt.packages.shiftkey.dev/ubuntu any InRelease [1 228 B]
Obj:8 http://mx.archive.ubuntu.com/ubuntu noble-backports InRelease
Obj:9 https://ppa.launchpadcontent.net/danielrichter2007/grub-customizer/ubuntu noble InRelease
```

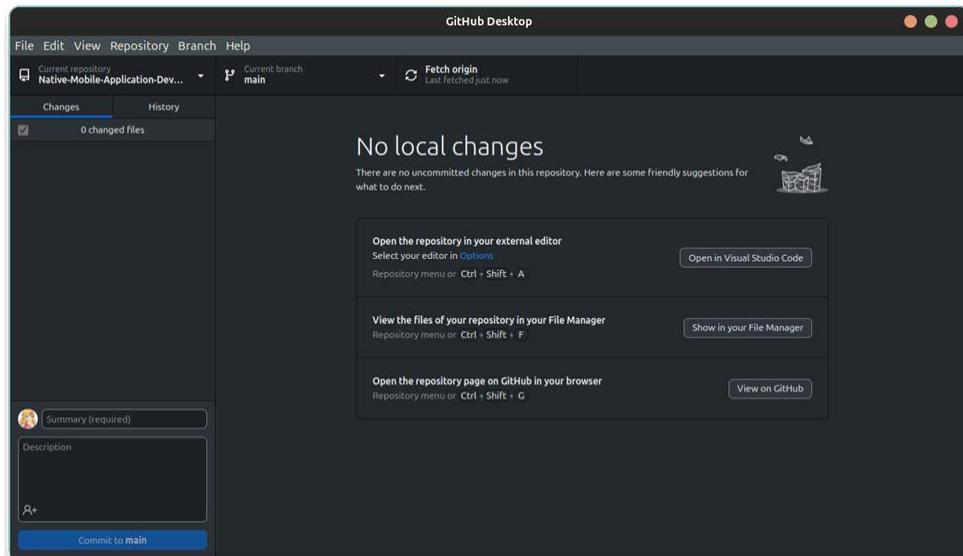
Una vez instalamos la aplicación, ingresamos a ella y nos pide conectarnos a nuestra cuenta de GitHub, como se ve a continuación.



Una vez iniciada sesión clonamos un repositorio para esto escogemos la opción de Add y buscamos el repositorio deseado, como se muestra a continuación.



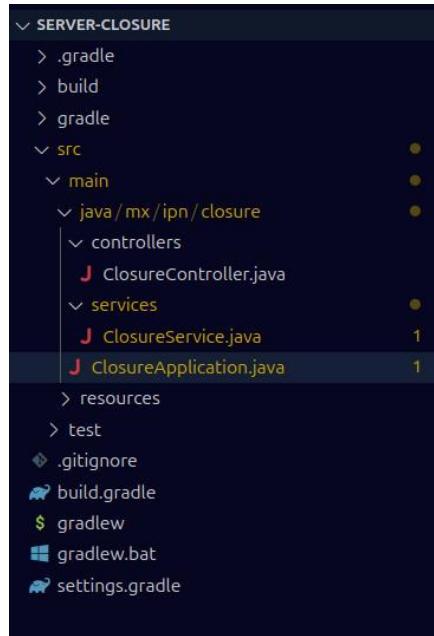
Para terminar, escogemos la opción de fetch para traer todos los archivos del repositorio y poder comenzar a trabajar.



Con todo esto tenemos listo nuestro entorno de desarrollo en Linux para la unidad de aprendizaje de desarrollo de aplicaciones móviles nativas.

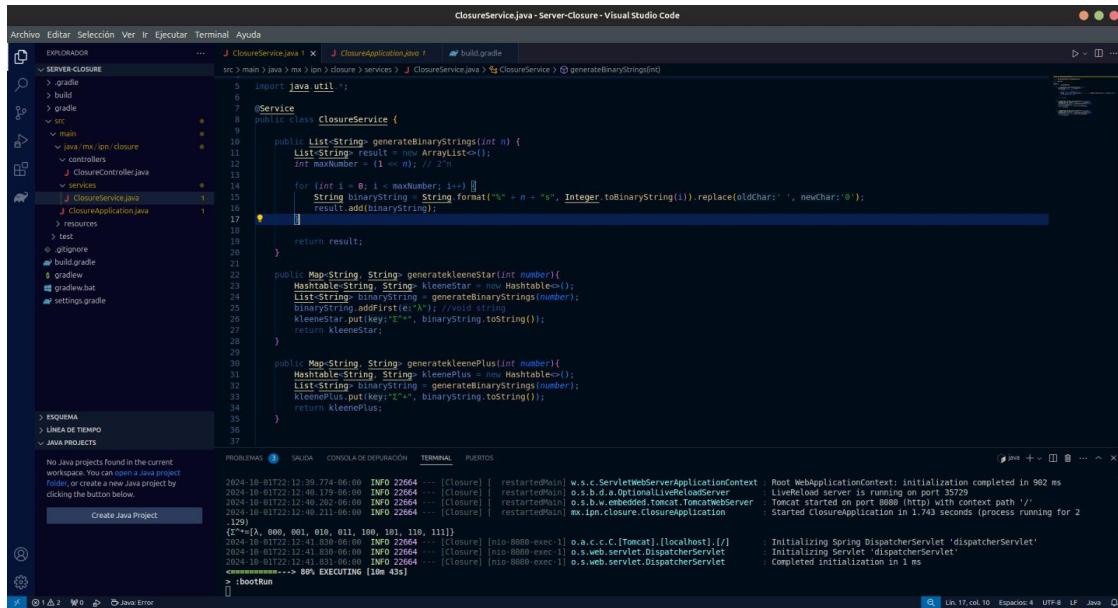
Ejercicio 2: Implementación de un API Rest para operaciones de conjuntos.

Para la implementación del API utilizando Spring Boot que expone dos operaciones principales: la generación de cadenas binarias mediante las operaciones de Kleene Star y Kleene Plus. La lógica está distribuida en tres partes principales: el servicio ClosureService, el controlador ClosureController y la clase de arranque ClosureApplication, véase la estructura del proyecto en la siguiente imagen.



El servicio ClosureService contiene los métodos que implementan la lógica principal para generar cadenas binarias. El método generateBinaryStrings(int n) es responsable de generar todas las combinaciones posibles de cadenas binarias de longitud n. Para ello, se usa un ciclo que recorre desde 0 hasta 2^n , utilizando la función Integer.toBinaryString(i) para convertir cada número a su representación binaria. El método luego completa cada cadena para que todas tengan la misma longitud añadiendo ceros al inicio mediante String.format().

El método generatekleeneStar(int number) utiliza el resultado de generateBinaryStrings(int n) para generar la operación *Kleene Star*, que es el conjunto de todas las cadenas binarias posibles más la cadena vacía (λ). Para incluir esta cadena vacía, se añade al inicio de la lista con binaryString.add(0, " λ "). El método devuelve un Map donde la clave es Σ^* y el valor es la lista de cadenas binarias. De manera similar, el método generatekleenePlus(int number) genera el conjunto *Kleene Plus*, que contiene todas las combinaciones binarias posibles, sin la cadena vacía. Aquí no se añade la cadena vacía y se devuelve directamente la lista como un Map con la clave Σ^+ véase el código en la imagen.



```

ClosureService.java - Server-Closure - Visual Studio Code

Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda

EXPLORADOR SERVER-CLOSURE
src > main > java > mx > ipn > closure > services > ClosureService.java
5 import java.util.*;
6
7 @Service
8
9 public class ClosureService {
10
11     public List<String> generateBinaryStrings(int n) {
12         List<String> result = new ArrayList<>();
13         int maxNumber = (1 << n); // 2^n
14
15         for (int i = 0; i < maxNumber; i++) {
16             String binaryString = String.format("%" + n + "s", Integer.toBinaryString(i)).replace(' ', '0');
17             result.add(binaryString);
18         }
19
20         return result;
21     }
22
23     public Map<String, String> generatekleeneStar(int number) {
24         Map<String, String> kleeneStar = new Hashtable<>();
25         List<String> binaryString = generateBinaryStrings(number);
26         binaryString.addFirst("");
27         kleeneStar.put(key: "Σ*", binaryString.toString());
28         return kleeneStar;
29     }
30
31     public Map<String, String> generatekleenePlus(int number) {
32         Map<String, String> kleenePlus = new Hashtable<>();
33         List<String> binaryString = generateBinaryStrings(number);
34         kleenePlus.put(key: "Σ+", binaryString.toString());
35         return kleenePlus;
36     }
37 }

2024-10-01T22:12:39.774-06:00 INFO 22664 --- [Closure] [ restartMain] w.s.c.ServletWebServerApplicationContext: Root WebApplicationContext: initialization completed in 902 ms
2024-10-01T22:12:40.179-06:00 INFO 22664 --- [Closure] [ restartMain] o.s.b.d.a.OptionalLiveReloadServer: LiveReload server is running on port 35729
2024-10-01T22:12:40.202-06:00 INFO 22664 --- [Closure] [ restartMain] o.s.b.w.embedded.tomcat.TomcatWebServer: Tomcat started on port 8080 (http) with context path '/'
2024-10-01T22:12:40.211-06:00 INFO 22664 --- [Closure] [ restartMain] o.s.ipn.closure.ClosureApplication: Started closureApplication in 1.743 seconds (process running for 2
.129)
{"Σ*": "0", "00", "001", "010", "011", "100", "101", "110", "111}
2024-10-01T22:12:40.211-06:00 INFO 22664 --- [Closure] [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/]
2024-10-01T22:12:41.830-06:00 INFO 22664 --- [Closure] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet: Initializing Spring DispatcherServlet
2024-10-01T22:12:41.831-06:00 INFO 22664 --- [Closure] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet: Completed initialization in 1 ms
[bootRun]

```

El controlador ClosureController expone estos métodos como endpoints REST. Al realizar una solicitud GET a /api/closure/star/{number}, el controlador invoca el método generatekleeneStar(int number) del servicio y devuelve el resultado en formato JSON. De la misma manera, la ruta /api/closure/plus/{number} invoca el método generatekleenePlus(int number) para devolver el conjunto correspondiente véase el código en la imagen.

ClosureController.java - Server-Closure - Visual Studio Code

Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda

EXPLORADOR

... J ClosureController.java J ClosureController.java x build.gradle

✓ SERVER-CLOSURE

src 2 main.java > maven->closure->controllers J ClosureController.java > J ClosureController

src 2 package mx.ipm.closure.controllers;

2 import mx.ipm.closure.services.ClosureService;

4 import org.springframework.beans.factory.annotation.Autowired;

5 import org.springframework.web.bind.annotation.GetMapping;

6 import org.springframework.web.bind.annotation.PathVariable;

7 import org.springframework.web.bind.annotation.RequestMapping;

8 import org.springframework.web.bind.annotation.RestController;

9 import java.util.Map;

10

11 @RestController

12 @RequestMapping("/mx/itm/closure")

13 closureController @

14

15 @Autowired

16 private final ClosureService closureService;

17

18 public ClosureController(ClosureService closureService) {

19 this.closureService = closureService;

20 }

21

22 @GetMapping("star/{number}")

23 Map<String, String> getKleinStar(@PathVariable("number") int number) {

24 return closureService.generateKleinStar(number);

25 }

26

27

28 @GetMapping("plus/{number}")

29 Map<String, String> getKleinPlus(@PathVariable("number") int number) {

30 return closureService.generateKleinPlus(number);

31 }

32 }

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

99

100

101

102

103

104

105

106

107

108

109

110

111

112

113

114

115

116

117

118

119

120

121

122

123

124

125

126

127

128

129

130

131

132

133

134

135

136

137

138

139

140

141

142

143

144

145

146

147

148

149

150

151

152

153

154

155

156

157

158

159

160

161

162

163

164

165

166

167

168

169

170

171

172

173

174

175

176

177

178

179

180

181

182

183

184

185

186

187

188

189

190

191

192

193

194

195

196

197

198

199

200

201

202

203

204

205

206

207

208

209

210

211

212

213

214

215

216

217

218

219

220

221

222

223

224

225

226

227

228

229

230

231

232

233

234

235

236

237

238

239

240

241

242

243

244

245

246

247

248

249

250

251

252

253

254

255

256

257

258

259

260

261

262

263

264

265

266

267

268

269

270

271

272

273

274

275

276

277

278

279

280

281

282

283

284

285

286

287

288

289

290

291

292

293

294

295

296

297

298

299

300

301

302

303

304

305

306

307

308

309

310

311

312

313

314

315

316

317

318

319

320

321

322

323

324

325

326

327

328

329

330

331

332

333

334

335

336

337

338

339

340

341

342

343

344

345

346

347

348

349

350

351

352

353

354

355

356

357

358

359

360

361

362

363

364

365

366

367

368

369

370

371

372

373

374

375

376

377

378

379

380

381

382

383

384

385

386

387

388

389

390

391

392

393

394

395

396

397

398

399

400

401

402

403

404

405

406

407

408

409

410

411

412

413

414

415

416

417

418

419

420

421

422

423

424

425

426

427

428

429

430

431

432

433

434

435

436

437

438

439

440

441

442

443

444

445

446

447

448

449

450

451

452

453

454

455

456

457

458

459

460

461

462

463

464

465

466

467

468

469

470

471

472

473

474

475

476

477

478

479

480

481

482

483

484

485

486

487

488

489

490

491

492

493

494

495

496

497

498

499

500

501

502

503

504

505

506

507

508

509

510

511

512

513

514

515

516

517

518

519

520

521

522

523

524

525

526

527

528

529

530

531

532

533

534

535

536

537

538

539

540

541

542

543

544

545

546

547

548

549

550

551

552

553

554

555

556

557

558

559

550

551

552

553

554

555

556

557

558

559

560

561

562

563

564

565

566

567

568

569

570

571

572

573

574

575

576

577

578

579

580

581

582

583

584

585

586

587

588

589

580

581

582

583

584

585

586

587

588

589

590

591

592

593

594

595

596

597

598

599

590

591

592

593

594

595

596

597

598

599

600

601

602

603

604

605

606

607

608

609

600

601

602

603

604

605

606

607

608

609

610

611

612

613

614

615

616

617

618

619

610

611

612

613

614

615

616

617

618

619

620

621

622

623

624

625

626

627

628

629

620

621

622

623

624

625

626

627

628

629

630

631

632

633

634

635

636

637

638

639

630

631

632

633

634

635

636

637

638

639

640

641

642

643

644

645

646

647

648

649

640

641

642

643

644

645

646

647

648

649

650

651

652

653

654

655

656

657

658

659

650

651

652

653

654

655

656

657

658

659

660

661

662

663

664

665

666

667

668

669

660

661

662

663

664

665

666

667

668

669

670

671

672

673

674

675

676

677

678

679

670

671

672

673

674

675

676

677

678

679

680

681

682

683

684

685

686

687

688

689

680

681

682

683

684

685

686

687

688

689

690

691

692

693

694

695

696

697

698

699

690

691

692

693

694

695

696

697

698

699

700

701

702

703

704

705

706

707

708

709

700

701

702

703

704

705

706

707

708

709

710

711

712

713

714

715

716

717

718

719

710

711

712

713

714

715

716

717

718

719

720

721

722

723

724

725

726

727

728

729

720

721

722

723

724

725

726

727

728

729

730

731

732

733

734

735

736

737

738

739

730

731

732

733

734

735

736

737

738

739

740

741

742

743

744

745

746

747

748

749

740

741

742

743

744

745

746

747

748

749

750

751

752

753

754

755

756

757

758

759

750

751

752

753

754

755

756

757

758

759

760

761

762

763

764

765

766

767

768

769

760

761

762

763

764

765

766

767

768

769

770

771

772

773

774

775

776

777

778

779

770

771

772

773

774

775

776

777

778

779

780

781

782

783

784

785

786

787

788

789

780

781

782

783

784

785

786

787

788

789

790

791

792

793

794

795

796

797

798

799

790

791

792

793

794

795

796

797

798

799

800

801

802

803

804

805

806

807

808

809

800

801

802

803

804

805

806

807

808

809

810

811

812

813

814

815

816

817

818

819

810

811

812

813

814

815

816

817

818

819

820

821

822

823

824

825

826

827

828

829

820

821

822

823

824

825

826

827

828

829

830

831

832

833

834

835

836

837

838

839

830

831

832

833

834

835

836

837

838

839

840

841

842

843

844

845

846

847

848

849

840

841

842

843

844

845

846

847

848

849

850

851

852

853

854

855

856

857

858

859

850

851

852

853

854

855

856

857

858

859

860

861

862

863

864

865

866

867

868

869

860

861

862

863

864

865

866

867

868

869

870

871

872

873

874

875

876

877

878

879

870

871

872

873

874

875

876

877

878

879

880

881

882

883

884

885

886

887

888

889

880

881

882

883

884

885

886

887

888

889

890

891

892

893

894

895

896

897

898

899

890

891

892

893

894

895

896

897

898

899

900

901

902

903

904

905

906

907

908

909

900

901

902

903

904

905

906

907

908

909

910

911

912

913

914

915

916

917

918

919

910

911

912

913

914

915

916

917

918

919

920

921

922

923

924

925

926

927

928

929

920

921

922

923

924

925

926

927

928

929

930

931

932

933

934

935

936

937

938

939

930

931

932

933

934

935

936

937

938

939

940

941

942

943

944

945

946

947

948

949

940

941

942

943

944

945

946

947

948

949

950

951

952

953

954

955

956

957

958

959

950

951

952

953

954

955

956

957

958

959

960

961

962

963

964

965

966

967

968

969

960

961

962

963

964

965

966

967

968

969

970

971

972

973

974

975

976

977

978

979

970

971

972

973

974

975

976

977

978

979

980

981

982

983

984

985

986

987

988

989

980

981

982

983

984

985

986

987

988

989

990

991

992

993

994

995

996

997

998

999

990

991

992

993

994

995

996

997

998

999

1000

1001

1002

1003

1004

1005

1006

1007

1008

1009

1000

1001

1002

1003

1004

1005

1006

1007

1008

1009

1010

1011

1012

1013

1014

1015

1016

1017

1018

1019

1010

1011

1012

1013

1014

1015

1016

1017

1018

1019

1020

1021

1022

1023

1024

1025

1026

1027

1028

1029

1020

1021

1022

1023

1024

1025

1026

1027

1028

1029

1030

1031

1032

1033

1034

1035

1036

1037

1038

1039

1030

1031

1032

1033

1034

1035

1036

1037

1038

1039

1040

1041

1042

1043

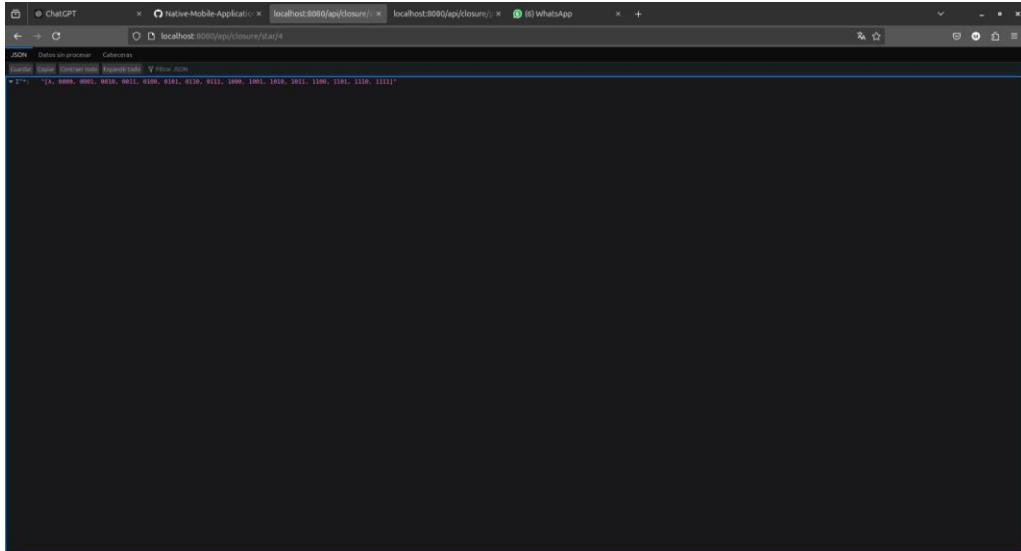
1044

<p

La clase `ClosureApplication` es la entrada principal de la aplicación. Utiliza Spring Boot para arrancar el servidor web integrado y está configurada con un `CommandLineRunner` para ejecutar una prueba al iniciar la aplicación. Este componente ejecuta el método `generatekleeneStar(3)` y muestra los resultados en la consola al arrancar la aplicación, permitiendo validar rápidamente que la lógica funciona correctamente véase el código en la imagen.

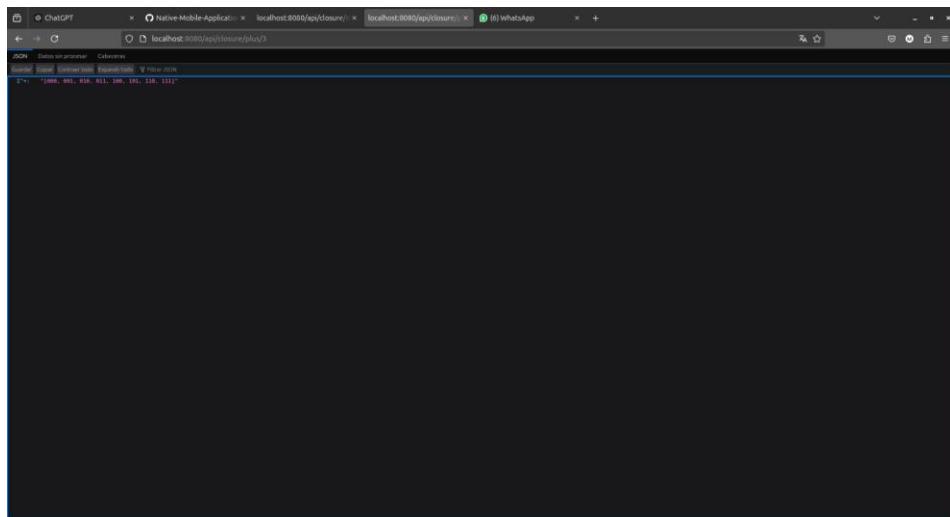
Las pruebas del sistema se realizaron de manera exitosa. Al arrancar la aplicación con el comando “./gradlew bootRun”, el servidor Tomcat integrado se inició correctamente en el puerto 8080. Las rutas expuestas en el controlador respondieron como se esperaba. Al acceder a <http://localhost:8080/api/closure/star/4> desde el navegador, la respuesta fue un

JSON con el conjunto de cadenas binarias generadas y la cadena vacía, tal como se define en la operación *Kleene Star* véase en la siguiente imagen.



```
JSON Datos en proceso: Cadenas binarias
[[], "0000", "0001", "0100", "0101", "0110", "0111", "1000", "1001", "1010", "1011", "1100", "1101", "1110", "1111"]
```

De igual forma, la solicitud a <http://localhost:8080/api/closure/plus/3> retornó las combinaciones binarias sin la cadena vacía, validando el funcionamiento del método *Kleene Plus*. Además, en la consola de la aplicación, el CommandLineRunner imprimió correctamente el resultado de la ejecución de generatekleeneStar(3), confirmando que la lógica se estaba ejecutando sin errores.



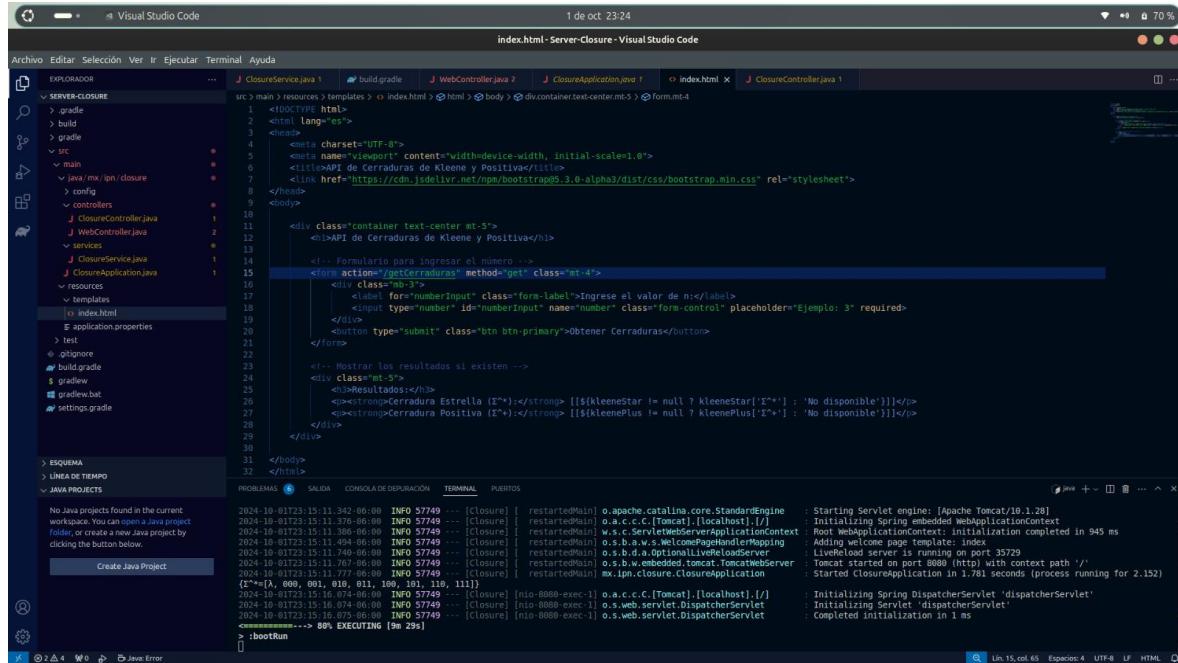
```
JSON Datos en proceso: Cadenas binarias
[[], "000", "001", "010", "011", "100", "101", "110", "111"]
```

Ejercicio 3: Implementación de una aplicación móvil web responsiva.

La implementación de la página web con funcionalidad responsiva y el envío de datos para procesar las operaciones de Kleene Star y Kleene Plus se ha logrado mediante el uso del framework Spring Boot en el backend y Thymeleaf como motor de plantillas para el frontend. La página está diseñada para ser completamente

responsiva utilizando Bootstrap, lo que asegura que se vea correctamente tanto en dispositivos móviles como en pantallas más grandes.

La plantilla HTML está diseñada de manera responsiva utilizando Bootstrap, lo cual facilita que la página se ajuste automáticamente a distintos tamaños de pantalla. El archivo index.html contiene un formulario para que el usuario ingrese el número sobre el cual se realizarán las operaciones de Kleene Star y Kleene Plus, véase en la imagen.



The screenshot shows the Visual Studio Code interface with the following details:

- Explorador (Explorer):** Shows the project structure for "SERVER-CLOSURE" with files like ClosureService.java, WebController.java, ClosureApplication.java, and index.html.
- index.html - Server-Closure - Visual Studio Code:** The content of the index.html file is displayed. It includes a form for entering a number and buttons for "Obtener Cerraduras" and "Obtener Cerradura Estrella". The code uses Bootstrap classes like "mt-4" and "mb-3".
- PROBLEMAS (Problems):** No problems found.
- SALIDA (Output):** Shows the Java application logs from the terminal, including the startup of Tomcat and the execution of the application.
- CONSOLA DE DEPURACIÓN (Debug Console):** Not currently active.
- TERMINAL:** Shows the command "bootRun" and the output "005 EXECUTING (9m 29s)".
- PUERTOS:** Not currently active.

```
index.html content (highlighted line 15):
15  <form action="/getCerraduras" method="get" class="mt-4">
```

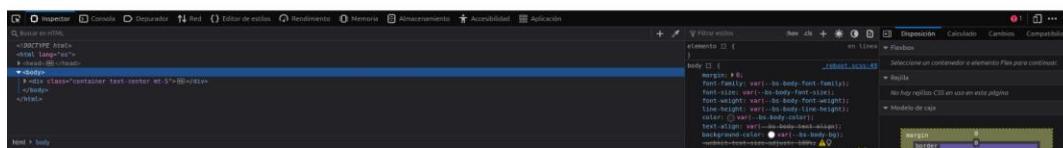
```
Terminal logs (bootRun output):
2024-10-01T23:15:11.342+00:00 INFO 57749 --- [Closure] | restartedMain| o.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/10.1.28]
2024-10-01T23:15:11.376+00:00 INFO 57749 --- [Closure] | restartedMain| o.a.c.c.C.[Tomcat].[localhost].[] : Initializing Spring embedded WebApplicationContext
2024-10-01T23:15:11.386+00:00 INFO 57749 --- [Closure] | restartedMain| w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 945 ms
2024-10-01T23:15:11.401+00:00 INFO 57749 --- [Closure] | restartedMain| o.a.c.c.C.[Tomcat].[localhost].[] : Context [/] is initialized for class: Index
2024-10-01T23:15:11.740+00:00 INFO 57749 --- [Closure] | restartedMain| o.s.b.d.a.OptionalLiveReloadServer : LiveReload server is running on port 35729
2024-10-01T23:15:11.767+00:00 INFO 57749 --- [Closure] | restartedMain| o.s.b.d.a.OptionalLiveReloadServer : Tomcat started on port 8080 (http) with context path '/'
2024-10-01T23:15:11.770+00:00 INFO 57749 --- [Closure] | restartedMain| o.s.b.d.a.OptionalLiveReloadServer : LiveReload server is running on port 35729
2024-10-01T23:15:11.771+00:00 INFO 57749 --- [Closure] | restartedMain| o.s.b.d.a.OptionalLiveReloadServer : Tomcat started on port 8080 (http) with context path '/'
2024-10-01T23:15:11.772+00:00 INFO 57749 --- [Closure] | restartedMain| o.s.b.d.a.OptionalLiveReloadServer : Starting Closure application in 1.781 seconds (process running for 2.152)
2024-10-01T23:15:11.773+00:00 INFO 57749 --- [Closure] | restartedMain| o.s.b.d.a.OptionalLiveReloadServer : Completed initialization in 1 ms
2024-10-01T23:15:16.074+00:00 INFO 57749 --- [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[] : Initializing Spring DispatcherServlet 'dispatcherServlet'
2024-10-01T23:15:16.075+00:00 INFO 57749 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
2024-10-01T23:15:16.075+00:00 INFO 57749 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 1 ms
:bootRun
005 EXECUTING (9m 29s)
```

Cuando el usuario ingresa un número y hace clic en el botón "Obtener Cerraduras", el formulario se envía a través de una solicitud HTTP GET a la URL /getCerraduras. El envío se maneja automáticamente al incluir la ruta de destino del formulario en el atributo action: "<form action="/getCerraduras" method="get" class="mt-4">". Una vez que la solicitud llega al servidor, el controlador de Spring Boot procesa los datos enviados, obteniendo el número ingresado mediante la anotación @RequestParam en el método getCerraduras.

Este método del controlador toma el número ingresado, lo procesa llamando a los métodos del servicio (ClosureService) y luego devuelve la plantilla index.html, pero esta vez con los resultados de las operaciones generados y pasados al modelo para ser renderizados en el frontend, véase en la imagen.

Cuando los resultados se devuelven desde el servidor, son renderizados en la página utilizando las expresiones de Thymeleaf para acceder a los datos del modelo. Si los resultados están disponibles, se muestran en el HTML.

Las pruebas fueron realizadas de manera satisfactoria. Al ingresar un número en el formulario, los datos son enviados correctamente al backend, donde se procesan las operaciones de Kleene Star y Kleene Plus. Posteriormente, los resultados se muestran en la misma página de forma clara y legible, véase en la imagen siguiente.



Además, la responsividad de la página fue verificada accediendo desde diferentes dispositivos y tamaños de pantalla, confirmando que el diseño se adapta perfectamente tanto en dispositivos móviles como en pantallas de escritorio, véase en las imágenes siguientes.

The image consists of three vertically stacked screenshots of a web application interface. The top screenshot shows the application running on a mobile device (Galaxy S10/S) with a dark theme. The middle screenshot shows the application running on a desktop browser (Firefox) with a light theme. The bottom screenshot shows the developer tools (Firefox DevTools) open, with the 'Inspector' tab selected, displaying the HTML structure and CSS styles for the page. The application itself is a simple form with a text input, a button, and a results section. The results section displays two sets of binary strings: 'Cerradura Estrella' and 'Cerradura Positiva'.

API de Cerraduras de Kleene y Positiva

Ingrese el valor de n:

Ejemplo: 3

Obtener Cerraduras

Resultados:

Cerradura Estrella (Σ*): No disponible

Cerradura Positiva (Σ+): No disponible

API de Cerraduras de Kleene y Positiva

Ingrese el valor de n:

Ejemplo: 3

Obtener Cerraduras

Resultados:

Cerradura Estrella (Σ*): [, 0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111]

Cerradura Positiva (Σ+): [0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111]

Elementos: elemento en línea

body { margin: 0; font-family: var(--bs-body-font-family); font-size: var(--bs-body-font-size); font-weight: var(--bs-body-font-weight); line-height: var(--bs-body-line-height); color: var(--bs-body-color); text-align: var(--bs-body-text-align); background-color: var(--bs-body-background-color); width: 100%; height: 100%; }

Disposición: Calculado

Calculado: Cambios: Compatibilidad:

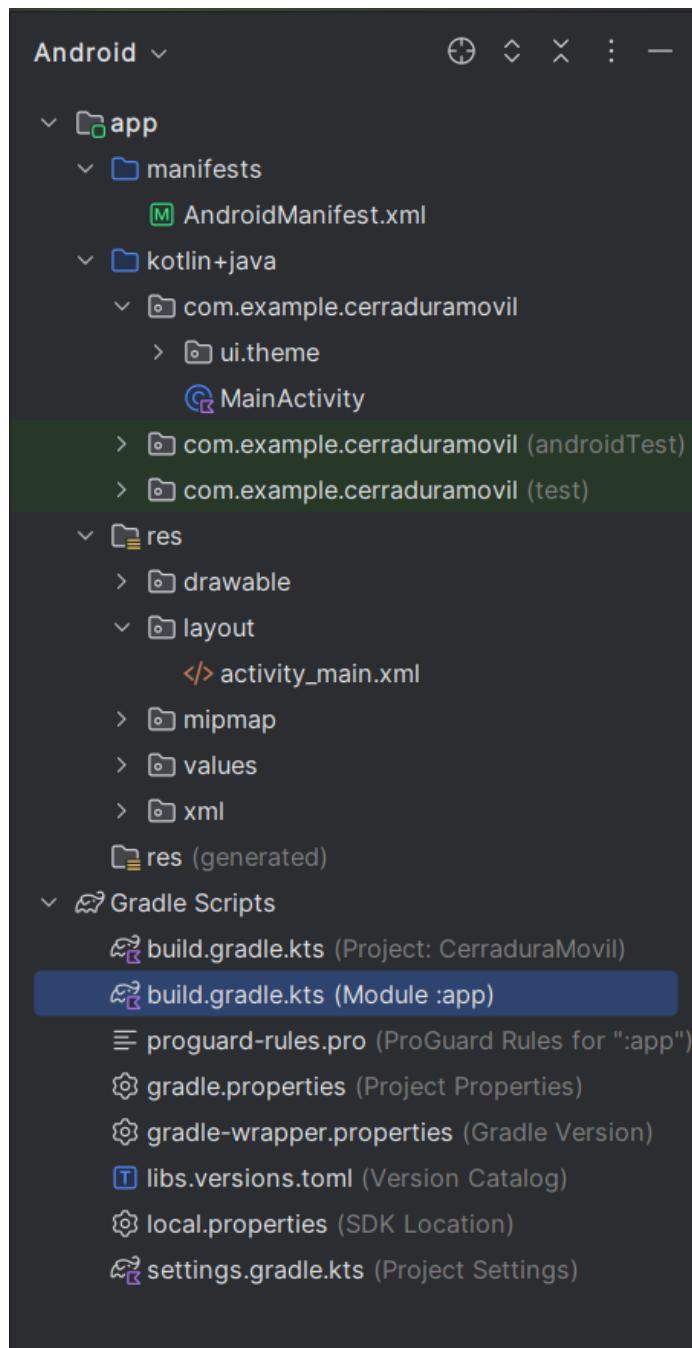
Reglas: No hay reglas CSS en uso en esta página

Modo de caja: Margin: 0 Border: 0

HTML > body

Ejercicio 4 (Opcional): Comenzar el desarrollo de una aplicación móvil (Android) nativa

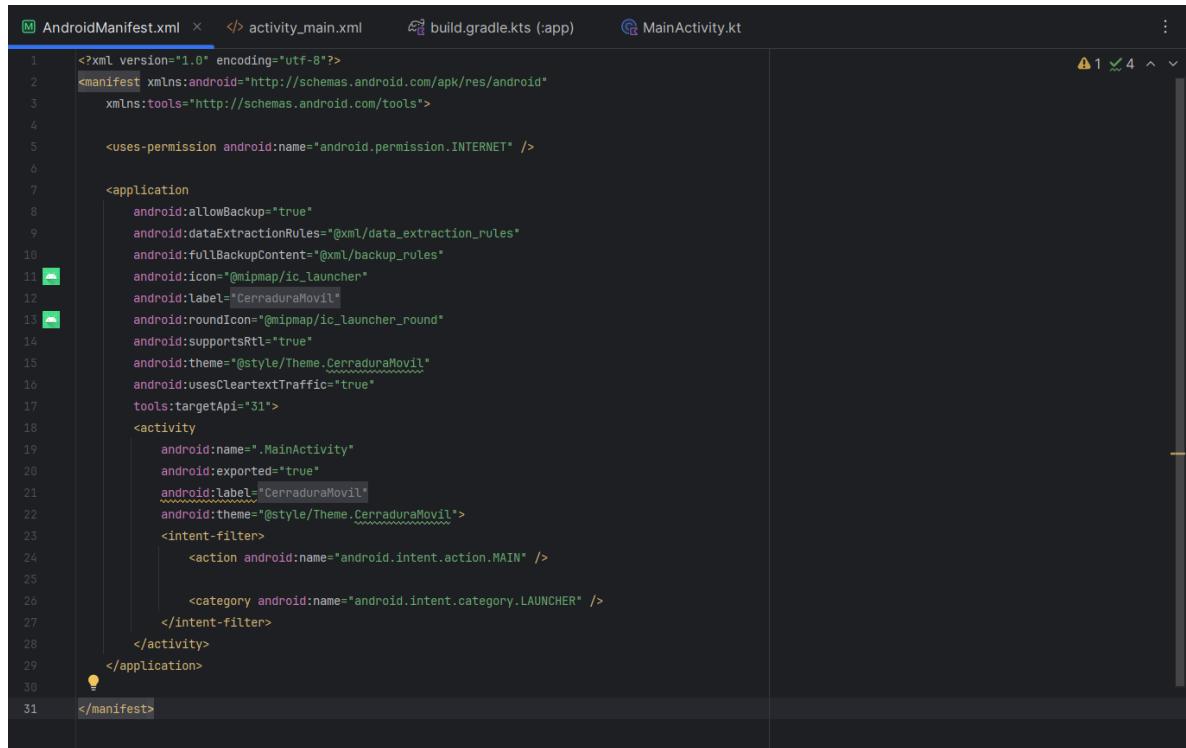
Para poder diseñar nuestra aplicación móvil, usaremos Kotlin con Gradle, donde, una vez finalizado, nos quedará con la siguiente estructura todo el proyecto.



Donde los ficheros que podemos resaltar son 4, los cuales son:

- **AndroidManifest.xml**

Este archivo es esencial en cualquier aplicación Android. Define la estructura general de la app, especificando componentes clave como actividades, servicios, receptores de difusión (broadcast receivers), y proveedores de contenido. También se declara información importante como los permisos que la aplicación necesita, versiones mínimas de la API, nombre del paquete, y configuraciones generales.



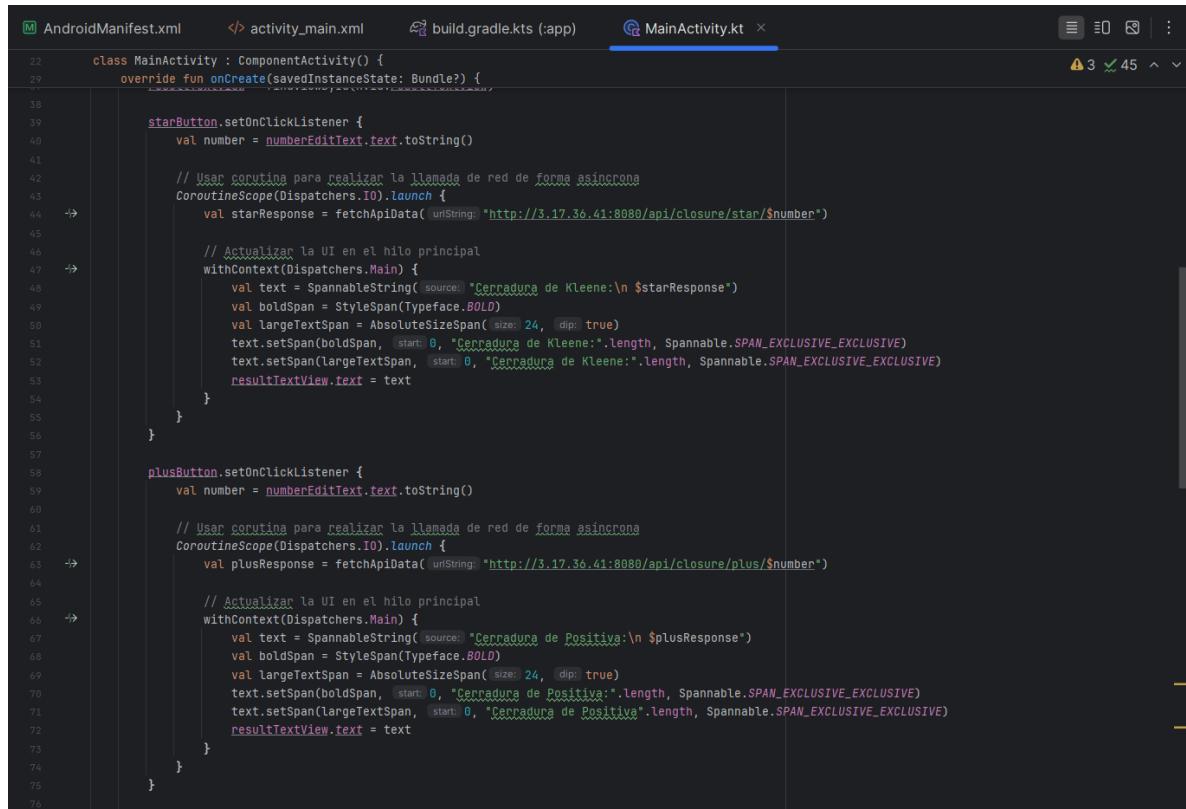
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <uses-permission android:name="android.permission.INTERNET" />

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.CerraduraMovil"
        android:usesCleartextTraffic="true"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true"
            android:label="@string/app_name"
            android:theme="@style/Theme.CerraduraMovil">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

○ MainActivity.kt

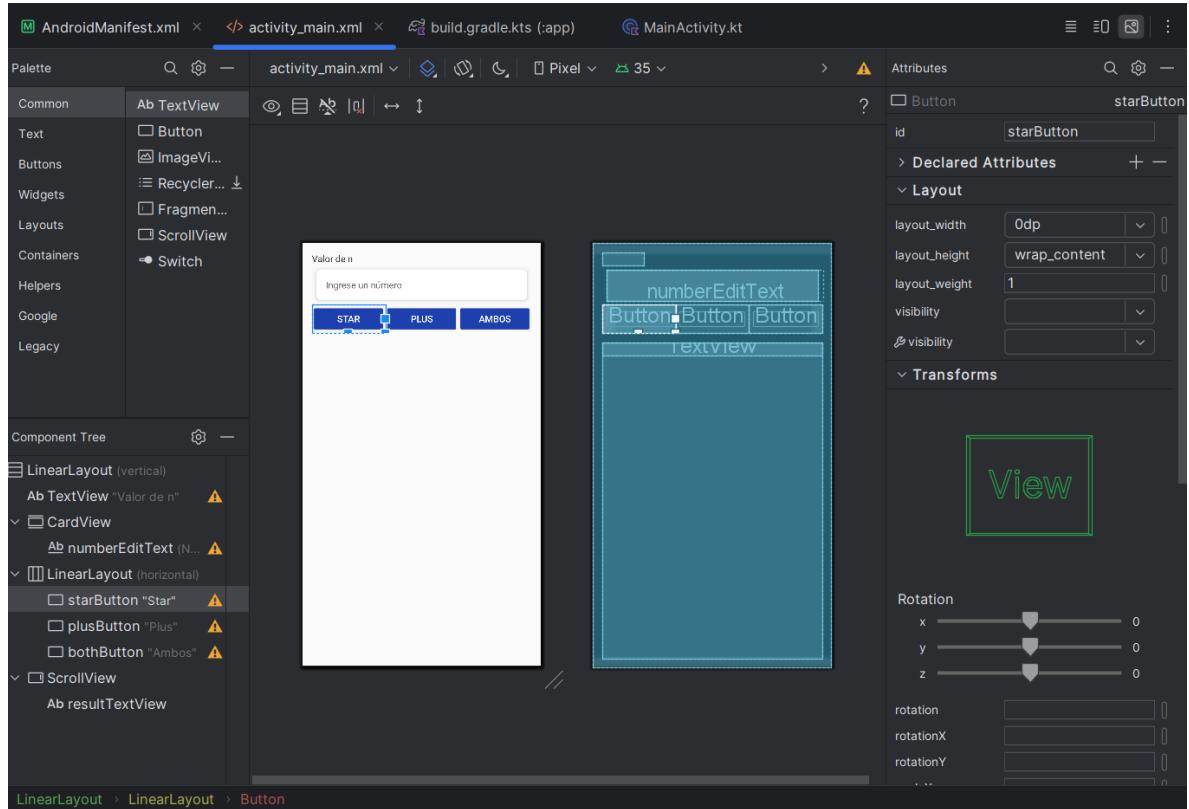
Es el archivo principal en Kotlin donde se gestiona la actividad principal (o pantalla principal) de la aplicación. En Android, una actividad es una pantalla única con una interfaz de usuario, y el archivo MainActivity.kt contiene la lógica y comportamiento de la primera pantalla que el usuario verá al abrir la app, en este caso, el calculo de la Cerradura Positiva, la Cerradura de Kleen o ambas.



```
22     class MainActivity : ComponentActivity() {
23         override fun onCreate(savedInstanceState: Bundle?) {
24             super.onCreate(savedInstanceState)
25             setContentView(R.layout.activity_main)
26
27             starButton.setOnClickListener {
28                 val number = numberEditText.text.toString()
29
30                 // Usar corutina para realizar la llamada de red de forma asincrona
31                 CoroutineScope(Dispatchers.IO).launch {
32                     val starResponse = fetchApiData(urlString: "http://3.17.36.41:8080/api/closure/star/$number")
33
34                     // Actualizar la UI en el hilo principal
35                     withContext(Dispatchers.Main) {
36                         val text = SpannableString("Cerradura de Kleene:\n $starResponse")
37                         val boldSpan = StyleSpan(Typeface.BOLD)
38                         val largeTextSpan = AbsoluteSizeSpan( size: 24, dip: true)
39                         text.setSpan(boldSpan, start: 0, "Cerradura de Kleene:".length, Spannable.SPAN_EXCLUSIVE_EXCLUSIVE)
40                         text.setSpan(largeTextSpan, start: 0, "Cerradura de Kleene:".length, Spannable.SPAN_EXCLUSIVE_EXCLUSIVE)
41                         resultTextView.text = text
42                     }
43                 }
44             }
45
46             plusButton.setOnClickListener {
47                 val number = numberEditText.text.toString()
48
49                 // Usar corutina para realizar la llamada de red de forma asincrona
50                 CoroutineScope(Dispatchers.IO).launch {
51                     val plusResponse = fetchApiData(urlString: "http://3.17.36.41:8080/api/closure/plus/$number")
52
53                     // Actualizar la UI en el hilo principal
54                     withContext(Dispatchers.Main) {
55                         val text = SpannableString("Cerradura de Positiva:\n $plusResponse")
56                         val boldSpan = StyleSpan(Typeface.BOLD)
57                         val largeTextSpan = AbsoluteSizeSpan( size: 24, dip: true)
58                         text.setSpan(boldSpan, start: 0, "Cerradura de Positiva:".length, Spannable.SPAN_EXCLUSIVE_EXCLUSIVE)
59                         text.setSpan(largeTextSpan, start: 0, "Cerradura de Positiva:".length, Spannable.SPAN_EXCLUSIVE_EXCLUSIVE)
60                         resultTextView.text = text
61                     }
62                 }
63             }
64         }
65     }
66 }
```

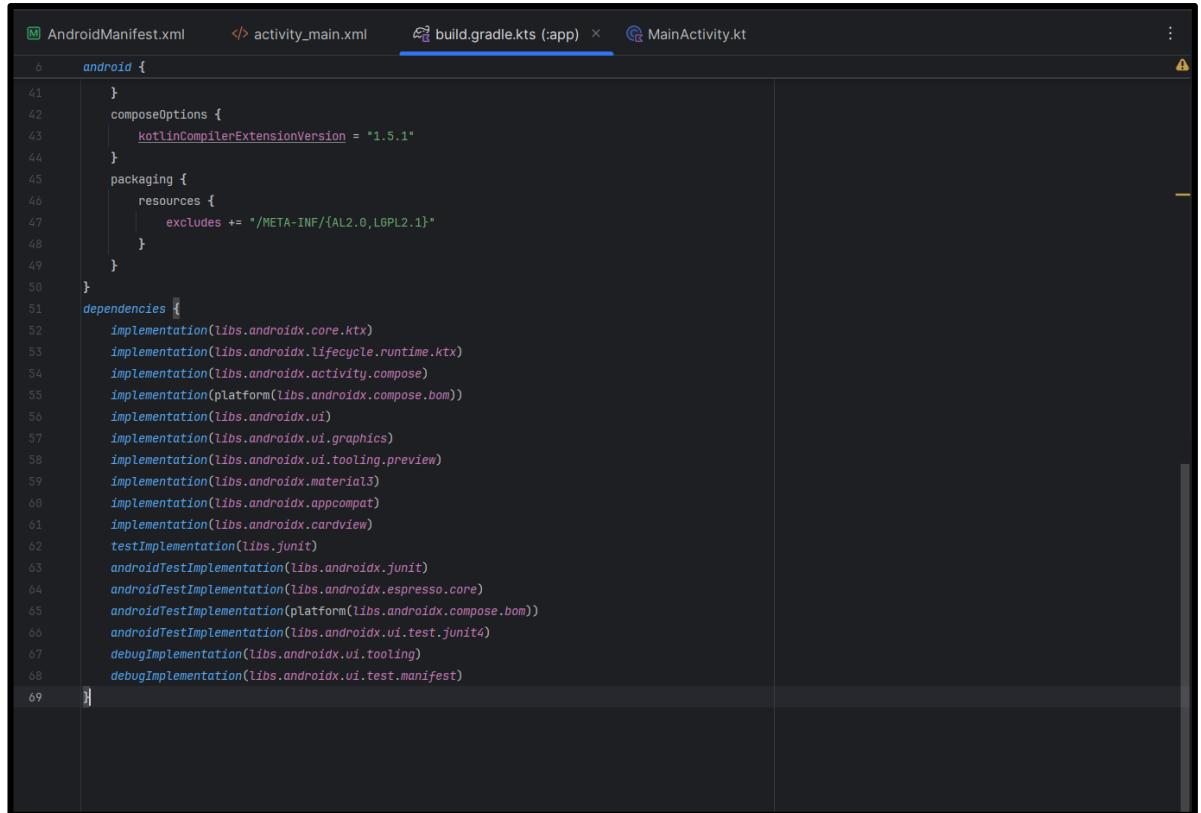
- **activity_main.xml**

Este archivo XML contiene el diseño de la interfaz de usuario (UI) de la actividad principal. Aquí defines cómo se ve la pantalla principal de tu aplicación (colocación de botones, textos, imágenes, etc.). Es el "layout" o diseño visual que luego será gestionado en el archivo Kotlin (MainActivity.kt).



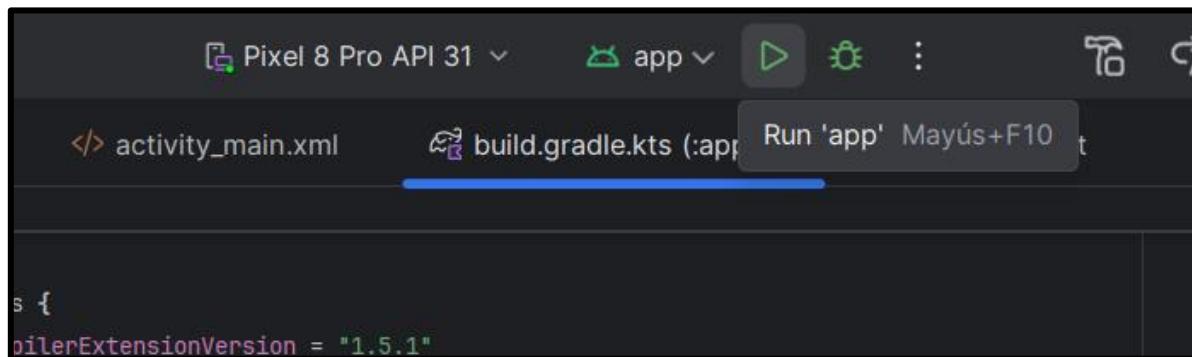
- **build.gradle.kts (module app)**

Es el archivo de configuración del módulo específico de la aplicación en un proyecto Android. Aquí se definen las dependencias (bibliotecas externas) que la aplicación utiliza, la versión de la compilación, versiones de las SDK de Android, entre otros aspectos de configuración de la compilación.

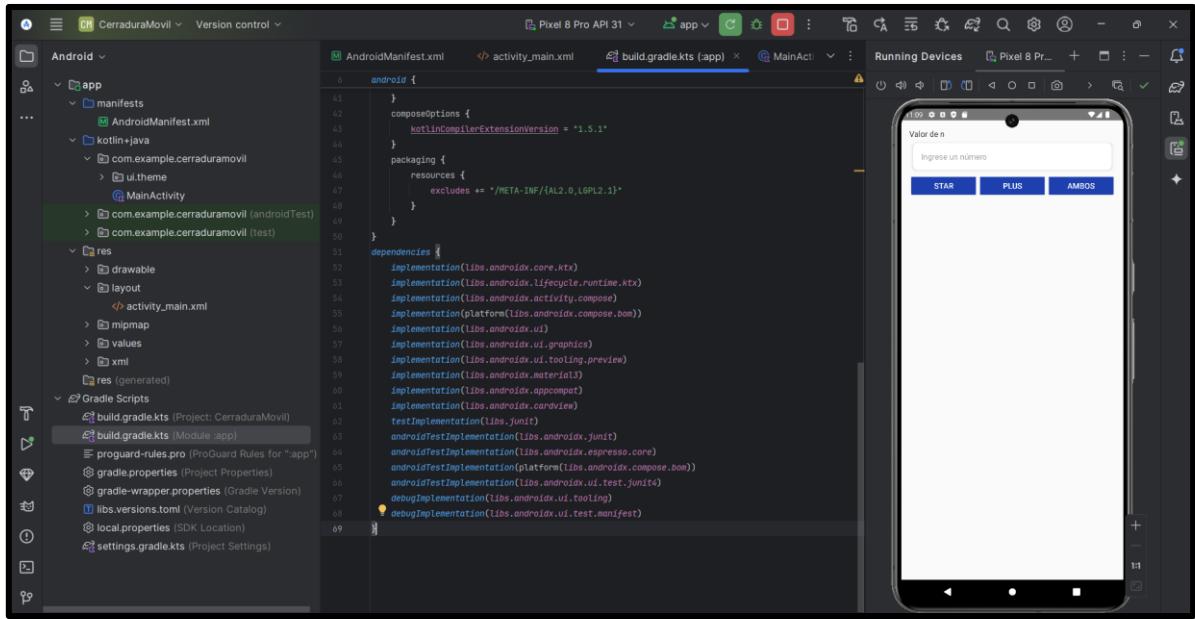


```
6 android {
41
42     composeOptions {
43         kotlinCompilerExtensionVersion = "1.5.1"
44     }
45     packaging {
46         resources {
47             excludes += "/META-INF/{AL2.0,LGPL2.1}*
48         }
49     }
50 }
51 dependencies {
52     implementation(libs.androidx.core.ktx)
53     implementation(libs.androidx.lifecycle.runtime.ktx)
54     implementation(libs.androidx.activity.compose)
55     implementation(platform(libs.androidx.compose.bom))
56     implementation(libs.androidx.ui)
57     implementation(libs.androidx.ui.graphics)
58     implementation(libs.androidx.ui.tooling.preview)
59     implementation(libs.androidx.material3)
60     implementation(libs.androidx.appcompat)
61     implementation(libs.androidx.cardview)
62     testImplementation(libs.junit)
63     androidTestImplementation(libs.androidx.junit)
64     androidTestImplementation(libs.androidx.espresso.core)
65     androidTestImplementation(platform(libs.androidx.compose.bom))
66     androidTestImplementation(libs.androidx.ui.test.junit4)
67     debugImplementation(libs.androidx.ui.tooling)
68     debugImplementation(libs.androidx.ui.test.manifest)
69 }
```

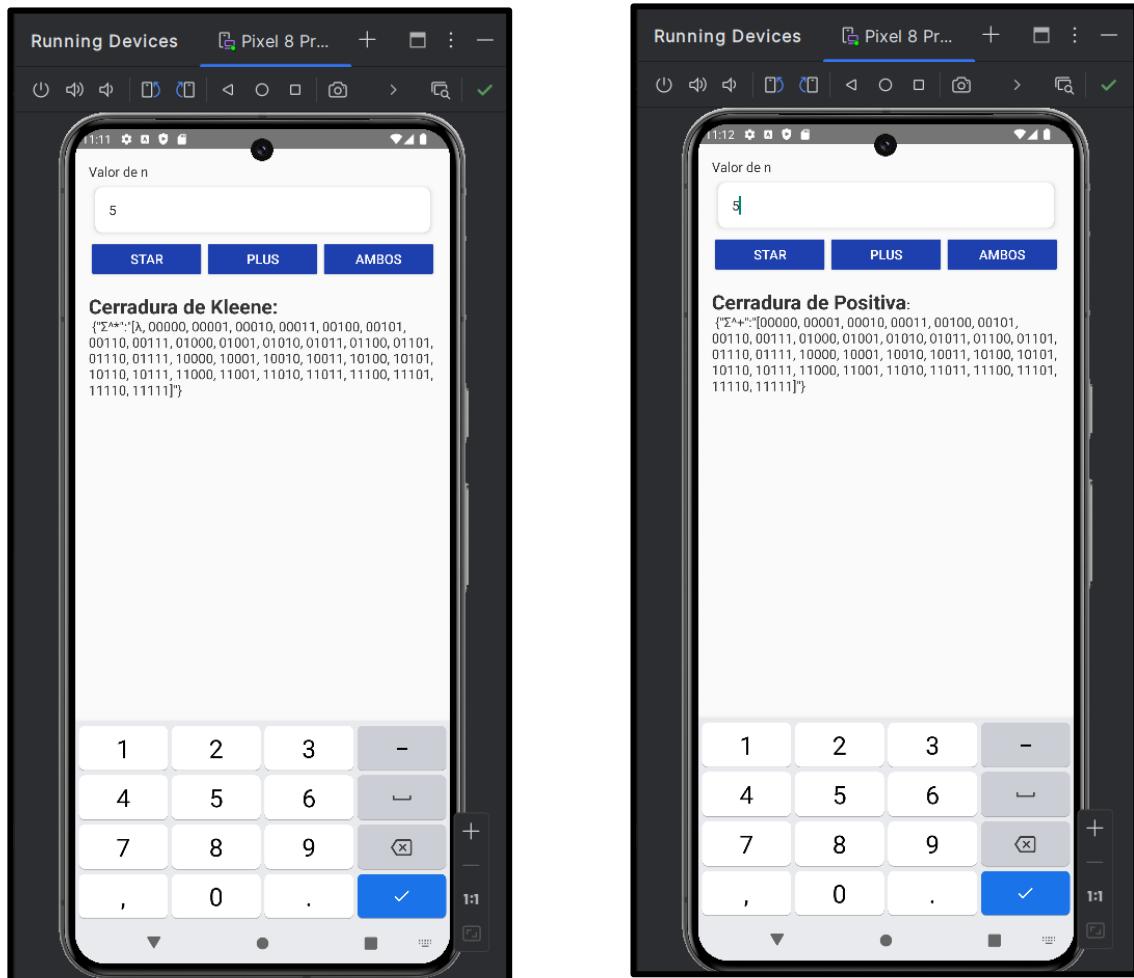
Ahora, podemos dar una ejecución a nuestro programa presionando el icono de Run en la barra superior:



Al compilar el proyecto, se mostrará una pestaña con un emulador de un teléfono Android, en donde nuestra aplicación se desplegará automáticamente.



Ahora podemos hacer varias pruebas de acuerdo con el funcionamiento de dichas cerraduras.





Podemos ver que la aplicación funciona correctamente.

Conclusiones

El objetivo principal de esta tarea fue familiarizarnos con las herramientas más utilizadas para desarrollar aplicaciones web, nativas e híbridas en el ecosistema Android. A través de la instalación y configuración de herramientas clave como Android Studio, Java Development Kit (JDK), Maven, Spring Boot, Git, y GitHub, logramos crear un entorno de desarrollo integral que nos permitió implementar diferentes tipos de aplicaciones.

El propósito de esta práctica se centró en la creación de una aplicación móvil que calcula las cerraduras positiva y estrella de Kleene sobre cadenas binarias de longitud variable. Estas operaciones, fundamentales en la teoría de autómatas y lenguajes formales, fueron implementadas en tres fases distintas:

- Desarrollo de una API Rest: Utilizando Spring Boot y gestionando el proyecto con Maven, implementamos una API que expone dos endpoints principales: uno para la cerradura de Kleene y otro para la cerradura positiva. El API recibe un número entero como parámetro y devuelve el conjunto correspondiente de cadenas binarias, utilizando un enfoque modular y limpio siguiendo el patrón de diseño Modelo-Vista-Controlador (MVC).
- Aplicación web responsive: Posteriormente, se diseñó una interfaz web responsive que interactúa con la API, permitiendo al usuario ingresar un valor y obtener las cadenas resultantes de las cerraduras de Kleene y positiva. Para lograrlo, utilizamos Bootstrap, que facilitó el diseño adaptable a dispositivos móviles.
- Aplicación móvil nativa: Finalmente, se desarrolló una aplicación nativa en Android, utilizando Kotlin y gestionando dependencias con Gradle. Esta aplicación se conecta a la API desarrollada en la primera etapa, permitiendo al usuario ingresar un número y mostrar los resultados directamente en la interfaz nativa de Android.

La práctica representó una excelente oportunidad para adentrarnos y conocer a fondo las diferentes herramientas necesarias para el desarrollo móvil, así como para adoptar buenas prácticas y convenciones de programación y trabajo en equipo. A través de la implementación de las aplicaciones y el uso de herramientas de desarrollo modernas, se fortalecieron las bases para la creación de software de calidad, siguiendo principios de código limpio y metodologías ágiles.