Caso Práctico 2: Instalación y Comparativa de IntelliJ

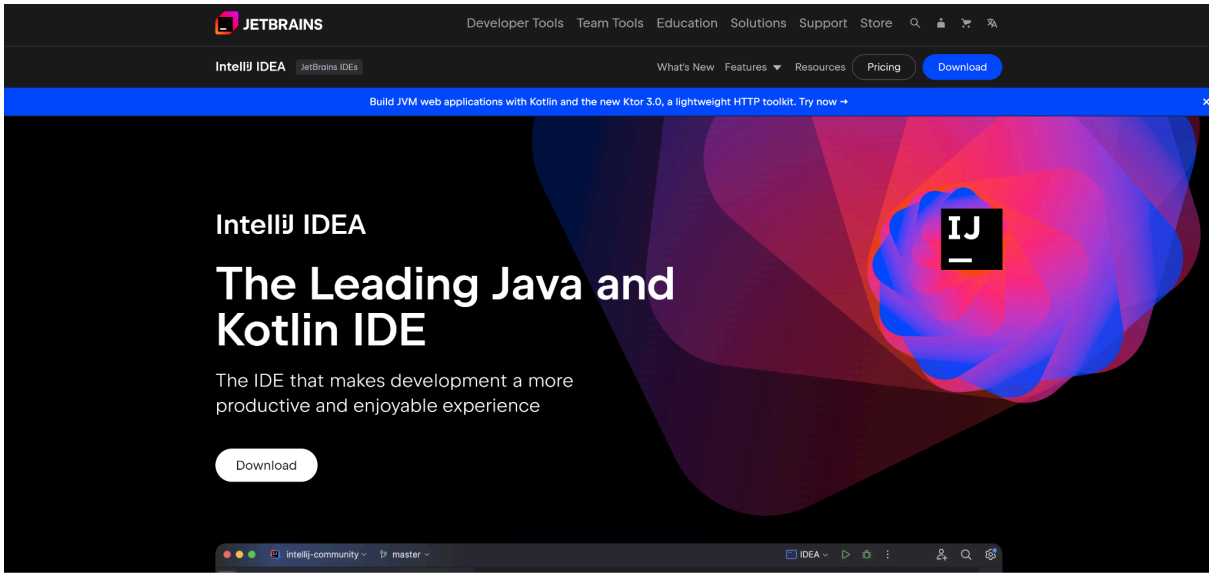
1. Introducción

El presente documento aborda la instalación y configuración del entorno de desarrollo integrado (IDE) IntelliJ, su personalización, así como la comparativa con otros IDEs relevantes. Además, incluye pruebas unitarias, generación de ejecutables y un análisis detallado de las características de IntelliJ frente a herramientas similares. Este enfoque integral permite evaluar la idoneidad de IntelliJ como herramienta clave para el desarrollo de proyectos.

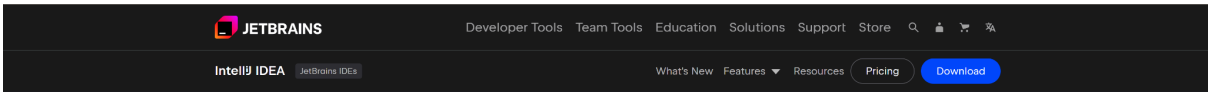
2. Instalación de IntelliJ

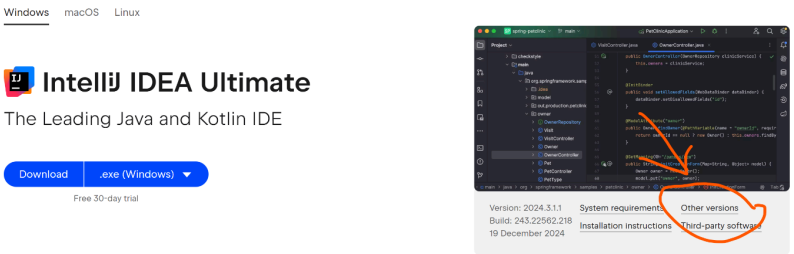
2.1 Proceso de instalación

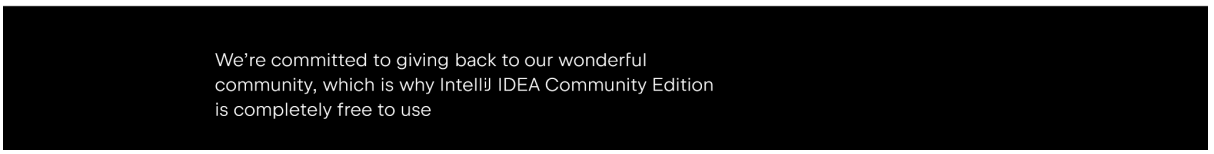
1. **Descarga del software**:
    - Para iniciar, es necesario visitar la página oficial de JetBrains (https://www.jetbrains.com/idea/). Desde allí se puede seleccionar entre las versiones "Ultimate" (de pago, con más funcionalidades) y "Community" (gratuita, ideal para proyectos educativos y pequeños).



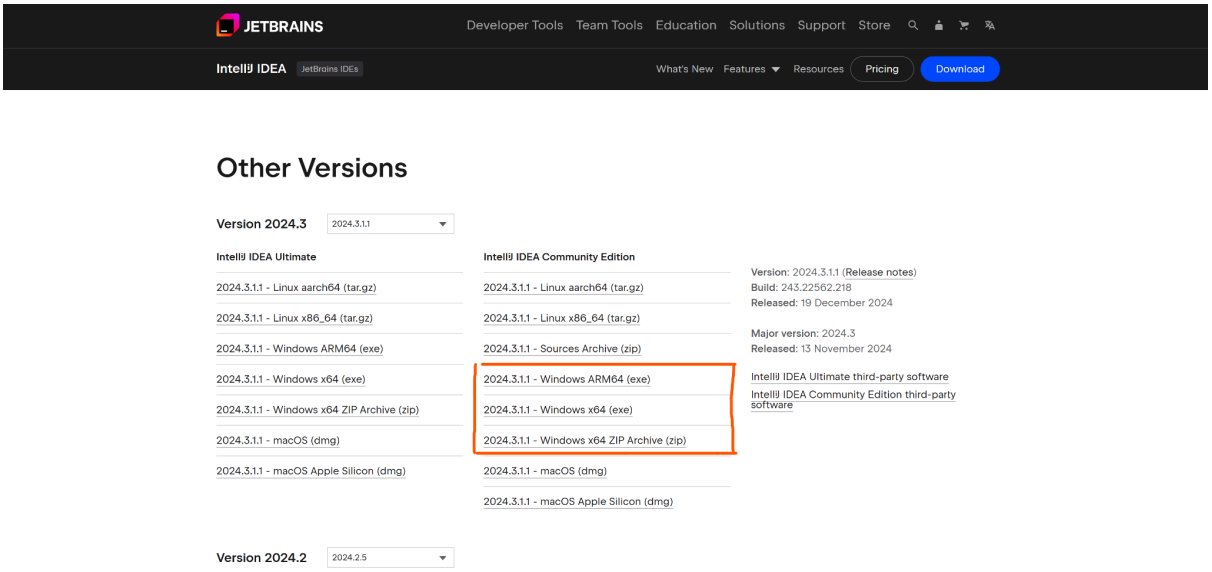
    - Cuidado, porque la versión gratuita, la Community Edition, la han escondido. Para descargarla, debemos pulsar



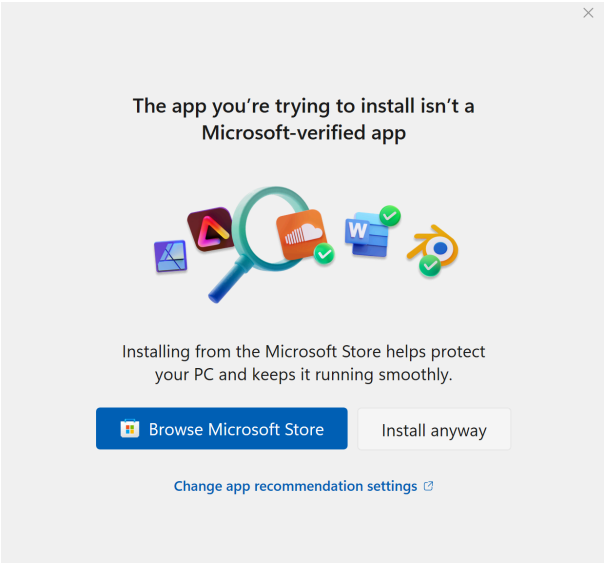




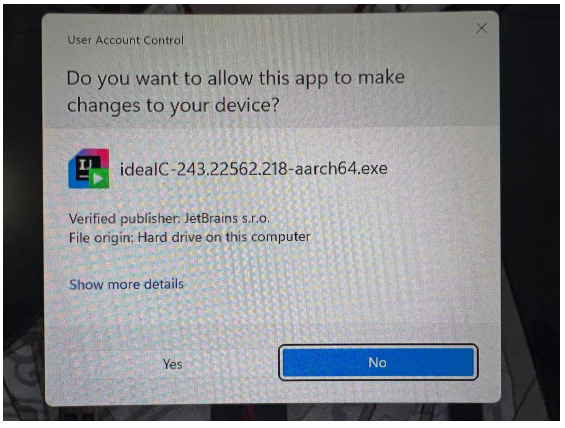
- Una vez seleccionada la versión adecuada, se descarga el archivo instalador compatible con el sistema operativo utilizado (Windows, macOS o Linux).



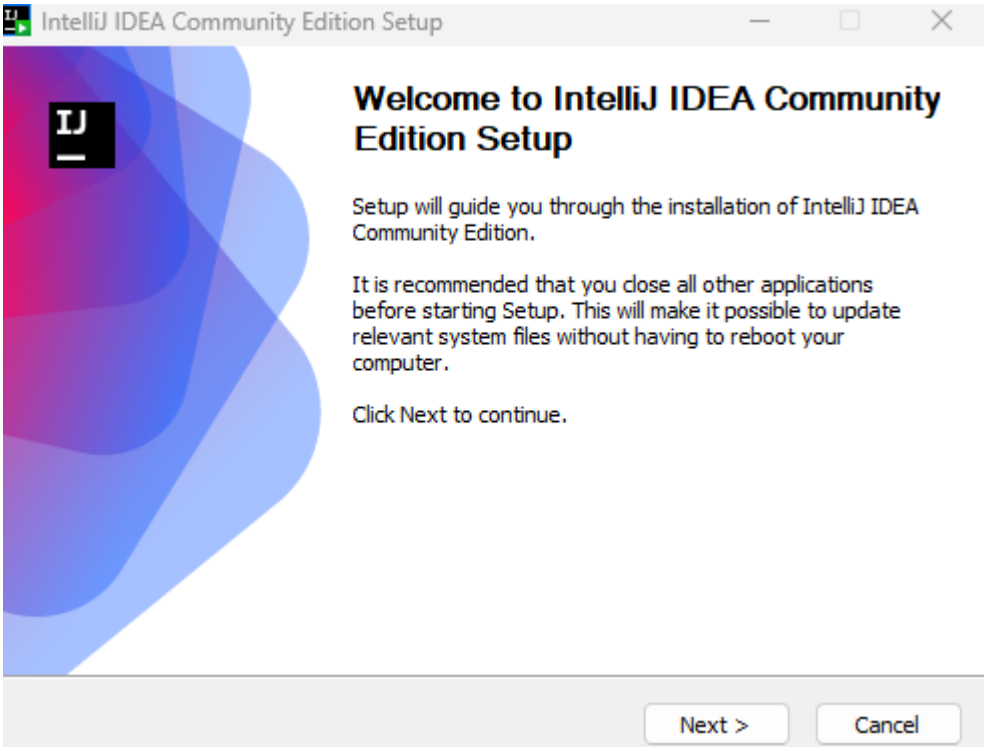
2. **Ejecución del instalador**:
   - Se procede a abrir el archivo descargado. En Windows, al ser una descarga que no está en la Microsoft Store, te avisa primero diciendo que no está verificada. Pulsa en instalar de todas formas. (“Install anyway” en mi caso, por tener Windows en inglés)

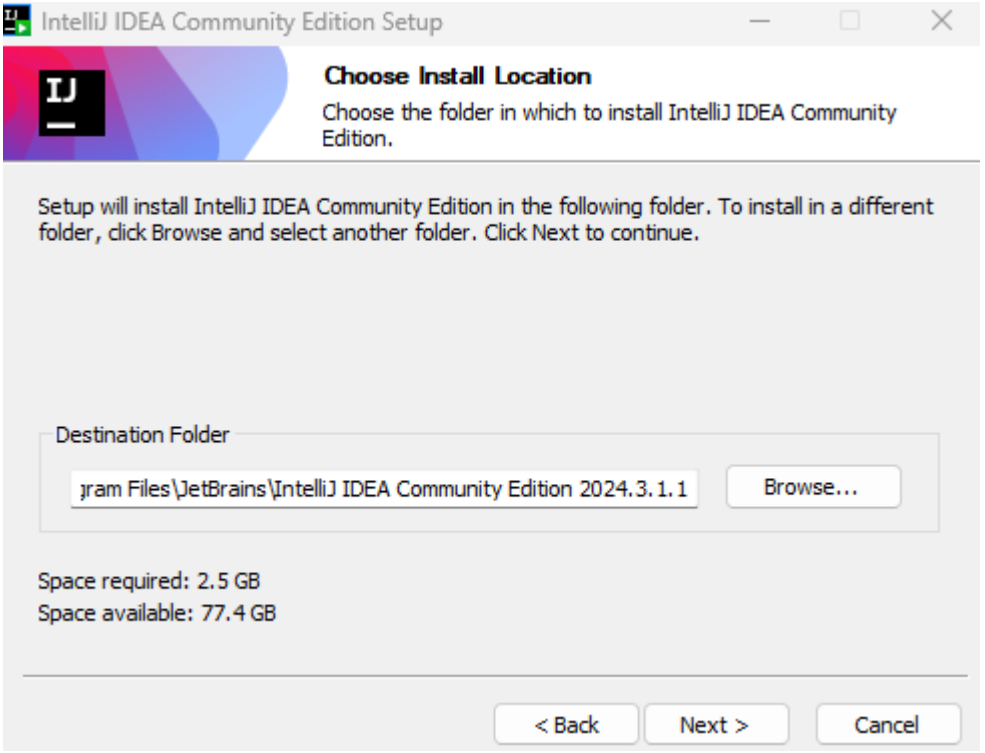


   - A continuación, saldrá una alerta como esta, que también deberemos aceptar con “Sí”, o en mi caso “Yes”.

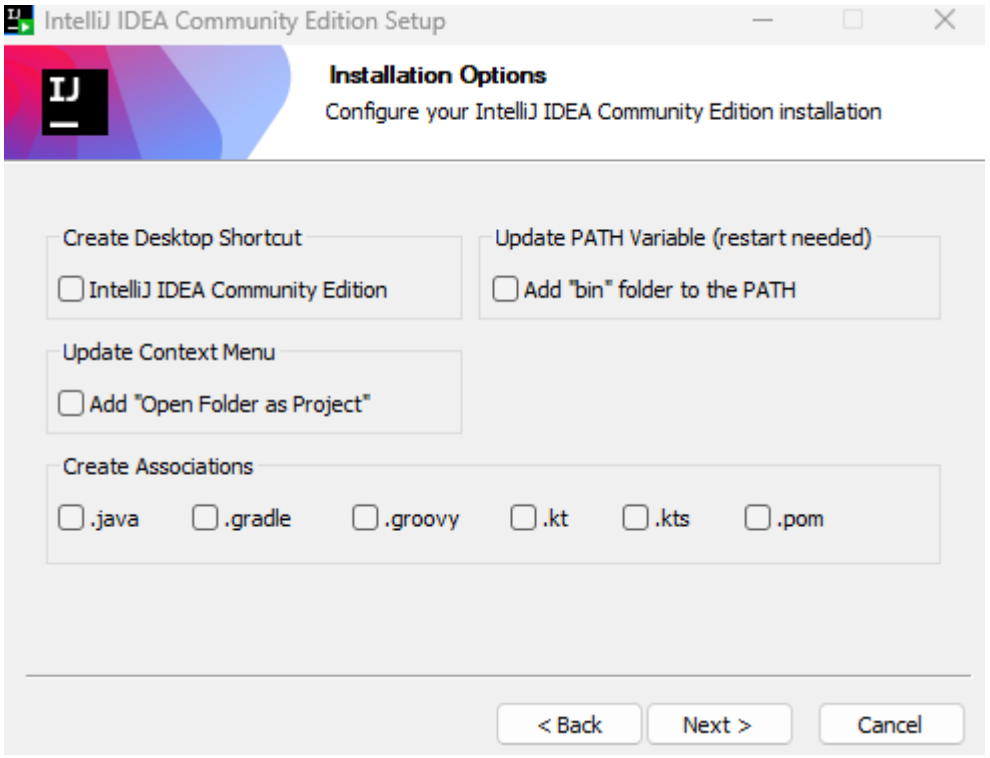


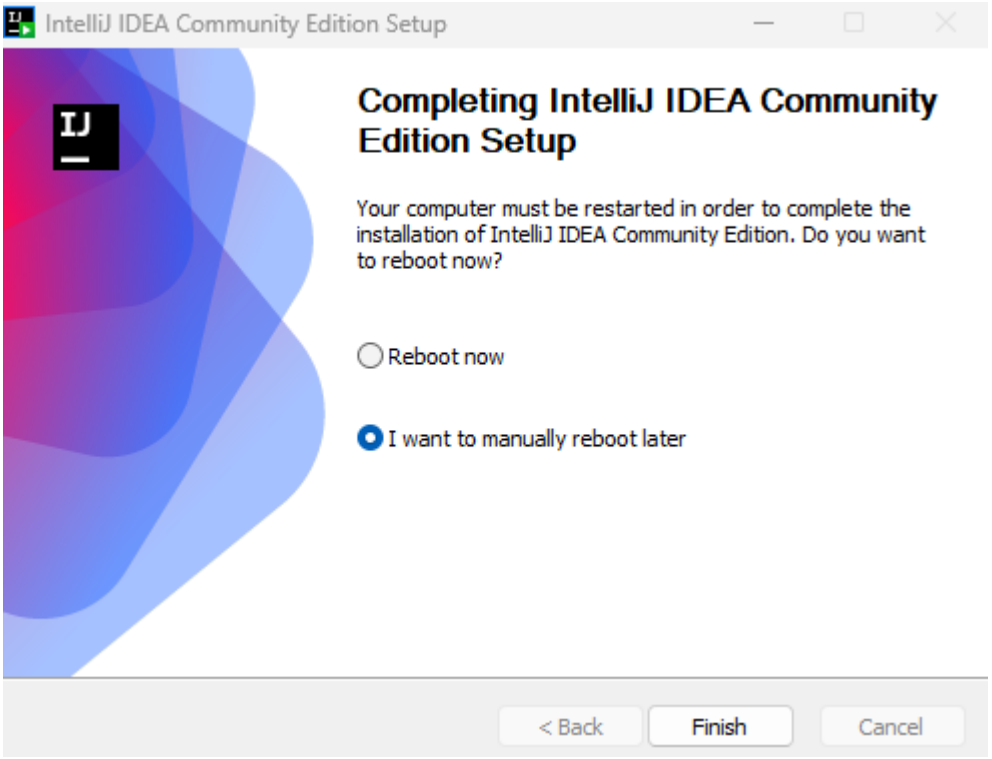
- Iremos siguiendo los pasos de la instalación marcando “Next”, eligiendo la carpeta de instalación, extensiones, etc.



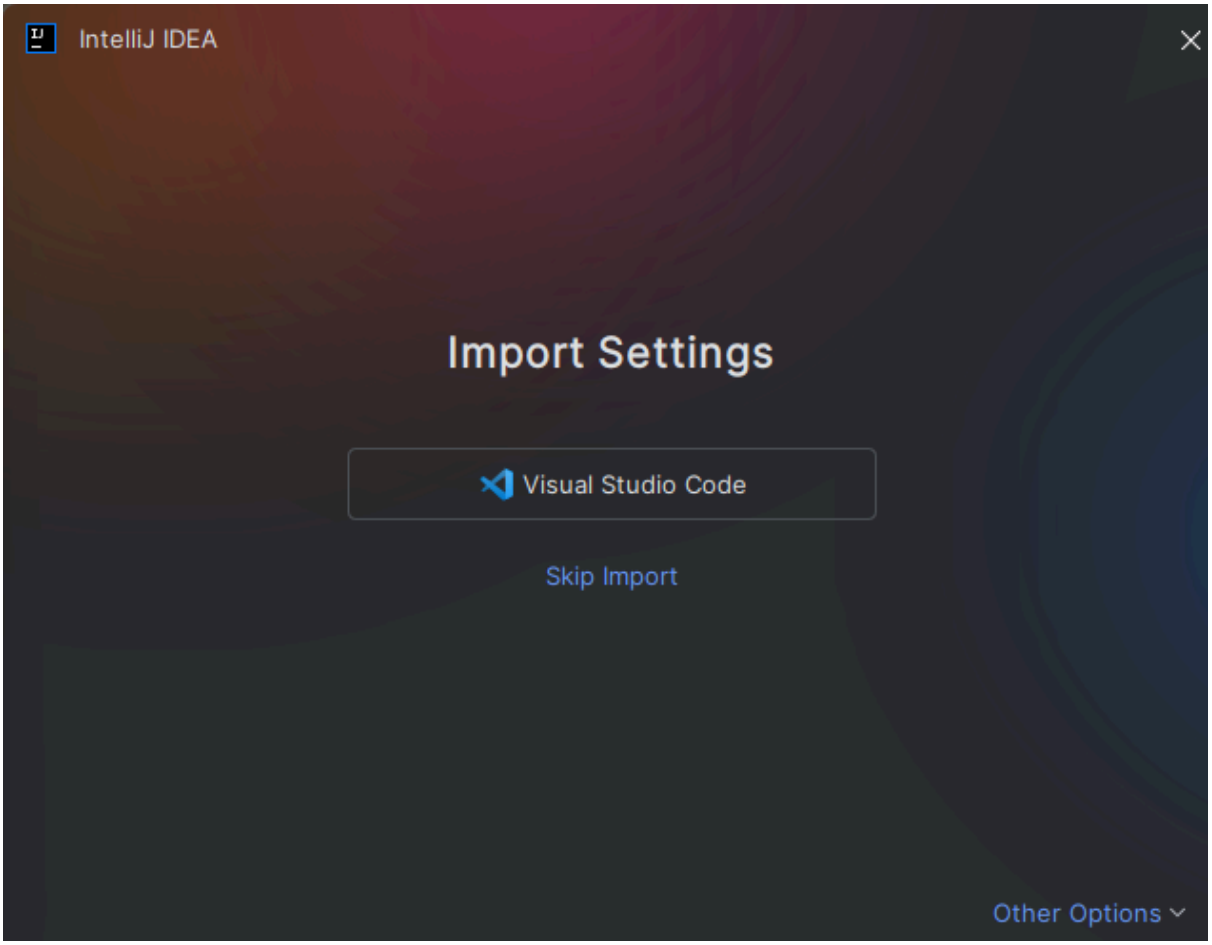


- Durante este proceso, se pueden seleccionar configuraciones adicionales, como la creación de atajos en el escritorio y la asociación de archivos .java a IntelliJ.

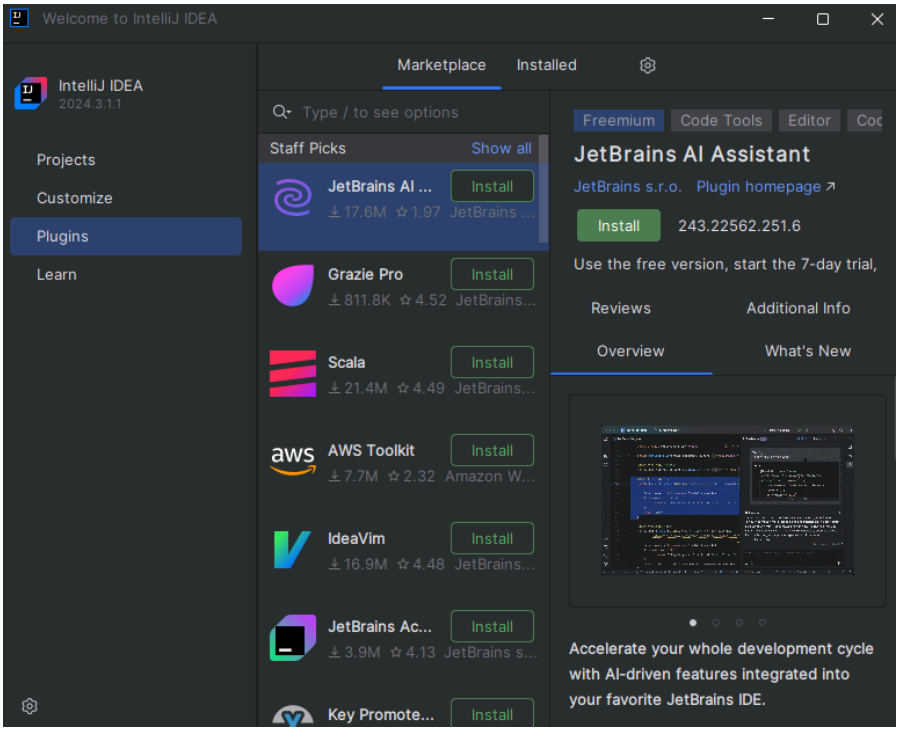

IntelliJ IDEA Community Edition Setup
Installation Options
Configure your IntelliJ IDEA Community Edition installation
Create Desktop Shortcut
IntelliJ IDEA Community Edition
Update PATH Variable (restart needed)
Add "bin" folder to the PATH
Update Context Menu
Add "Open Folder as Project"
Create Associations
.java
.gradle
.groovy
.kt
.kts
.pom
< Back
Next >
Cancel


IntelliJ IDEA Community Edition Setup
Completing IntelliJ IDEA Community Edition Setup
Your computer must be restarted in order to complete the installation of IntelliJ IDEA Community Edition. Do you want to reboot now?
Reboot now
I want to manually reboot later
< Back
Finish
Cancel

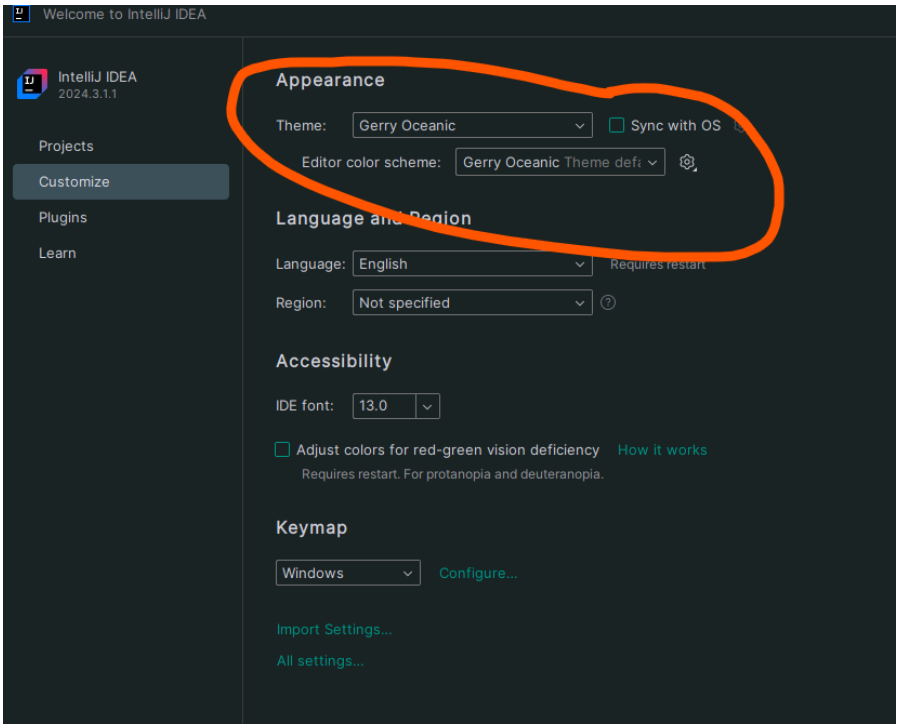
3. **Configuración inicial**:
    - Al iniciar IntelliJ por primera vez, se puede elegir importar configuraciones de una instalación previa o comenzar desde cero. Yo he utilizado Visual Studio Code de manera autodidacta antes de comenzar este curso, por lo que me proponía importar las configuraciones, cosa que aproveché para aceptar.

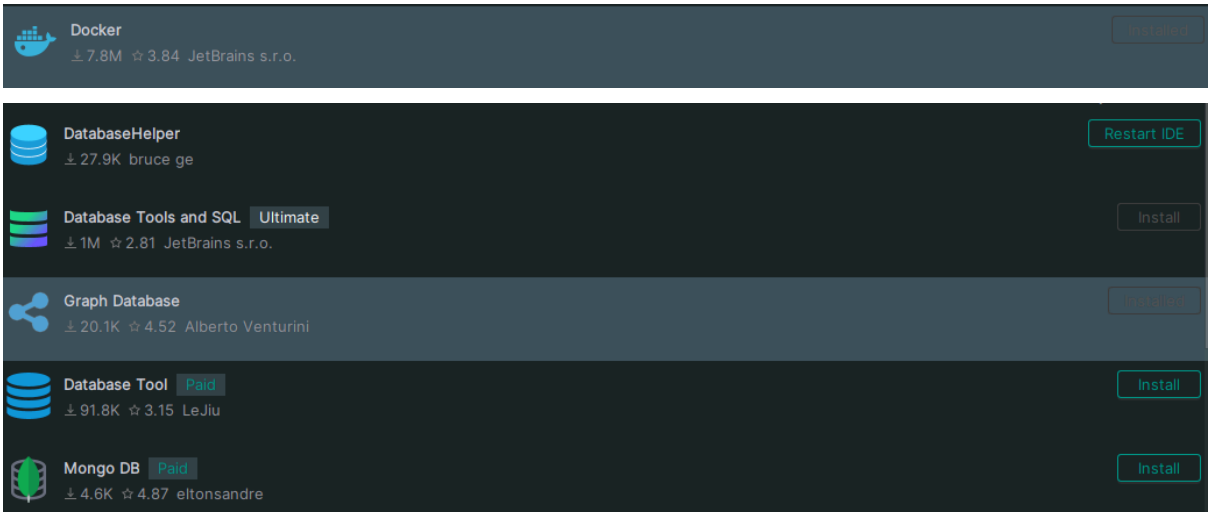


    - Es importante seleccionar los plugins necesarios desde el marketplace integrado. Esto incluye herramientas como soporte para Maven, Gradle y Git.

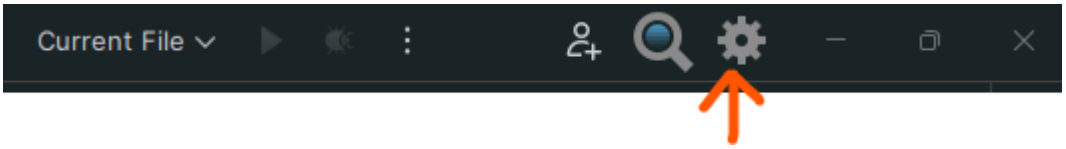


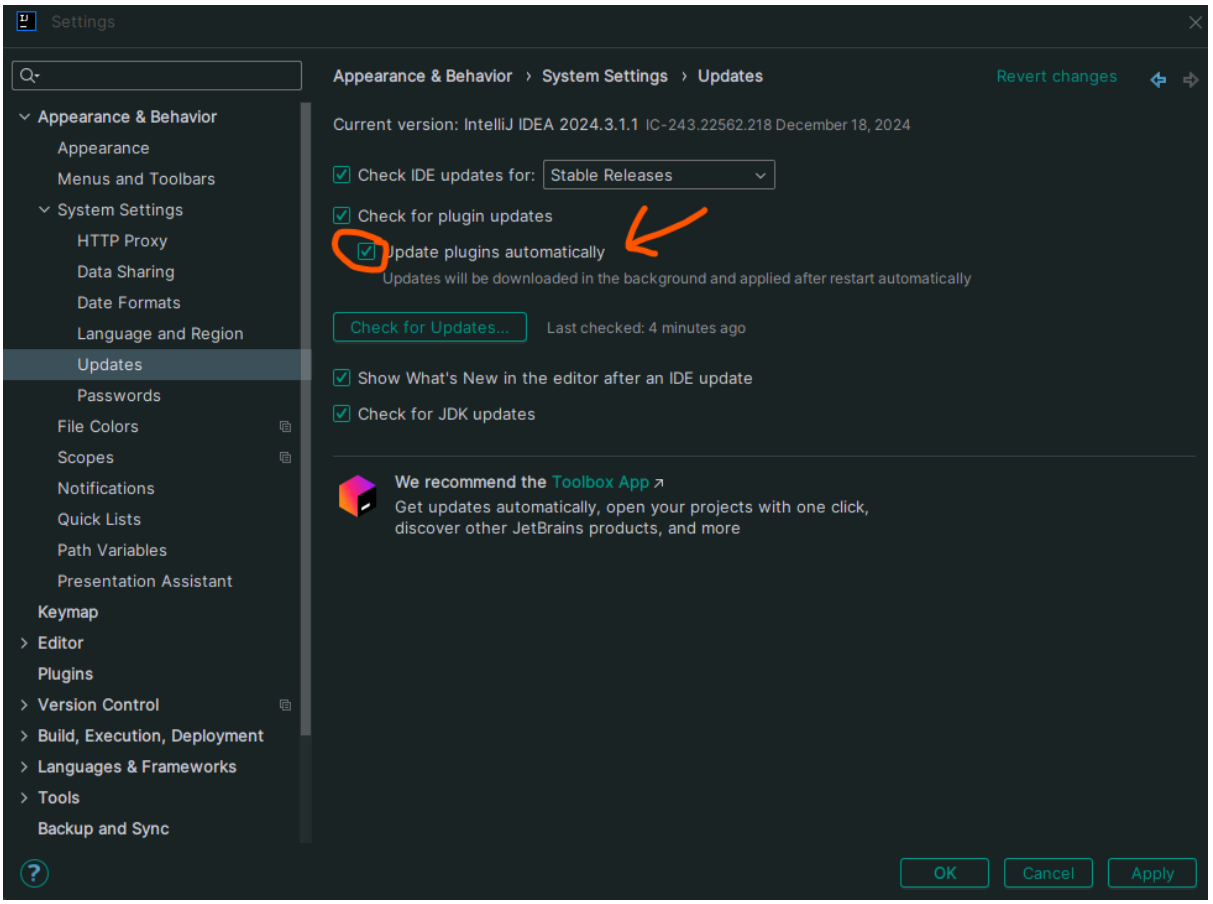
- Finalmente, el usuario puede personalizar la apariencia (temas claros u oscuros) y configurar la integración con herramientas externas como Docker o bases de datos.





- **Sistema de actualizaciones**: IntelliJ permite configurar actualizaciones automáticas pulsando en la rueda de engranaje (Settings) > Appearance & Behavior > System Settings > Updates.

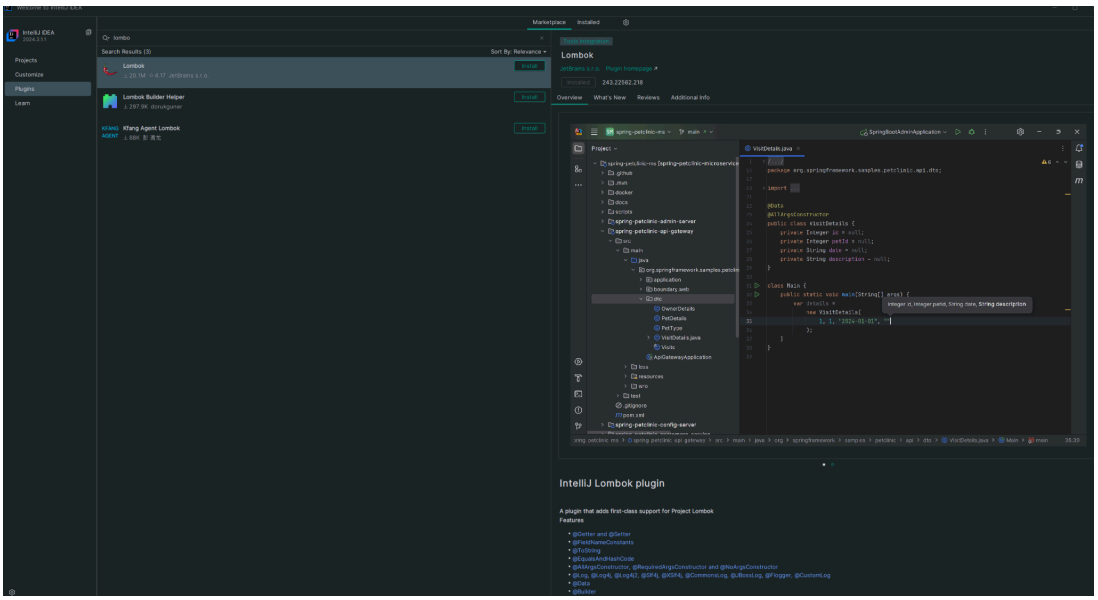




- Es recomendable habilitar esta opción para recibir mejoras constantes y parches de seguridad.

2.2 Personalización

- **Plugins adicionales**:
  - El marketplace de IntelliJ ofrece una gran variedad de plugins para lenguajes adicionales (como Python o Ruby), herramientas de colaboración (como Code With Me) y funcionalidades avanzadas (como soporte para Kubernetes).
  - Un ejemplo común es la instalación del plugin Lombok, que facilita el trabajo con anotaciones en Java.



- **Temas y esquemas de código**:
    - La posibilidad de personalizar colores y fuentes es útil para reducir el cansancio visual. También se pueden importar esquemas prefabricados desde la comunidad.

- **Configuraciones avanzadas**:
    - Los desarrolladores pueden definir atajos de teclado personalizados y configurar perfiles de ejecución adaptados a sus necesidades.

2.3 Ventajas e inconvenientes

- **Ventajas**:
    - IntelliJ destaca por su potente sistema de autocompletado inteligente (IntelliSense) y sus herramientas integradas para refactorización.
    - Soporte nativo para una amplia gama de lenguajes, como Java, Kotlin, Scala, entre otros.
    - Integración fluida con sistemas de control de versiones como Git, GitHub y Bitbucket.

- **Inconvenientes**:
    - Su consumo de recursos puede ser elevado, lo que podría generar problemas en equipos de gama baja.
    - La versión gratuita (Community) carece de funcionalidades avanzadas, como herramientas de desarrollo web y soporte para Spring.

3. Comparativa con otras IDEs

3.1 IntelliJ vs Eclipse

- **Ventajas de IntelliJ**:
    - La interfaz de usuario es más moderna y limpia, ofreciendo una experiencia más intuitiva para el usuario.
    - El sistema de autocompletado y las sugerencias son más precisas, lo que mejora la productividad.
    - Las actualizaciones son frecuentes y garantizan mejoras continuas.

- **Ventajas de Eclipse**:
    - Eclipse es ligero en cuanto a consumo de recursos, lo que lo hace ideal para equipos con especificaciones limitadas.
    - Dispone de una gran cantidad de plugins gratuitos y una amplia comunidad de soporte.

3.2 IntelliJ vs Visual Studio Code

- **Ventajas de IntelliJ**:
    - Las herramientas para depuración de código Java son mucho más avanzadas en IntelliJ.
    - Integra directamente funcionalidades avanzadas como perfiles de rendimiento y generación de artefactos JAR y WAR.

- **Ventajas de Visual Studio Code**:
    - Es una herramienta más ligera y versátil, adecuada para lenguajes como JavaScript, TypeScript y Python.
    - La variedad de extensiones disponibles lo hace flexible para diferentes proyectos.

3.3 Tabla comparativa IntrlliJ vs. Eclipse vs. Visual Studio Code

Característica	IntelliJ	Eclipse	Visual Studio Code
Interfaz de usuario	Moderna y limpia	Clásica, menos intuitiva	Minimalista y altamente personalizable
Sistema de autocompletado	Avanzado y preciso	Menos desarrollo	Versátil, depende de extensiones
Actualizaciones	Frecuentes con mejoras contínuas	Menos frecuentes	Constantes, impulsadas por la comunidad
Consumo de recursos	Alto	Bajo	Bajo
Comunidad y plugins	Activa, plugins de calidad	Amplia comunidad, gran cantidad de plugins gratuitos	Extensa, gran variedad de extensiones
Especialización	Ideal para Java y Kotlin	Java y otras tecnologías empresariales	Lenguajes de script y desarrollo web
Herramientas avanzadas	Perfiles de rendimiento, artefactos JAR/WAR	Limitadas a plugins	Depende de extensiones
Integración con bases de datos	Robusta	Depende de plugins adicionales	Depende de extensiones
Facilidad de uso	Alta para desarrolladores experimentados	Intermedia	Intuitiva para principiantes

4. Pruebas unitarias y generación de ejecutables

4.1 Pruebas unitarias Las pruebas unitarias son fundamentales para garantizar que cada módulo de código funcione de forma independiente. En IntelliJ, estas pruebas se implementaron utilizando JUnit:

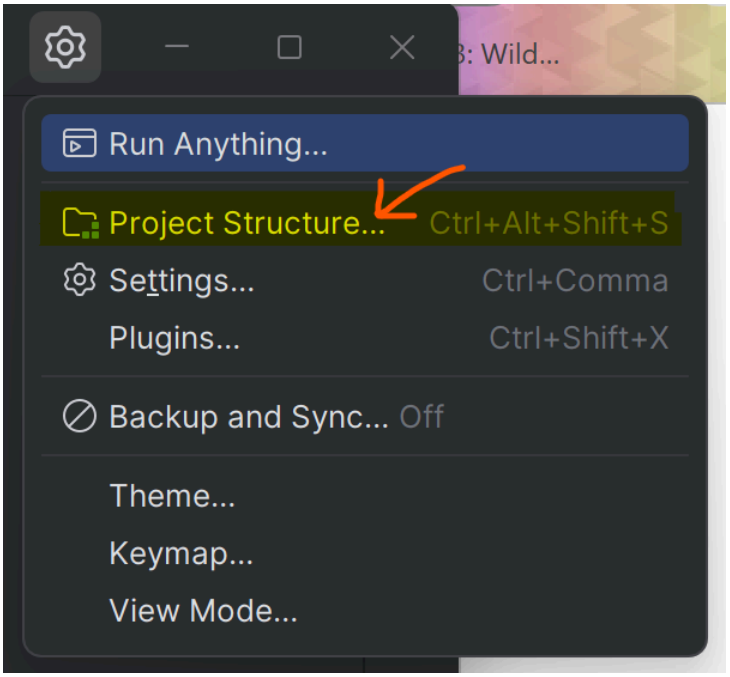
```
import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.*;

class CalculadoraTest {
    @Test
    void suma() {
        Calculadora calc = new Calculadora();
        assertEquals(5, calc.suma(2, 3));
    }

    @Test
    void resta() {
        Calculadora calc = new Calculadora();
        assertEquals(1, calc.resta(3, 2));
    }

    @Test
    void multiplicacion() {
        Calculadora calc = new Calculadora();
        assertEquals(6, calc.multiplicacion(2, 3));
    }

    @Test
```

```
    void division() {
        Calculadora calc = new Calculadora();
        assertEquals(2, calc.division(6, 3));
    }
}
```

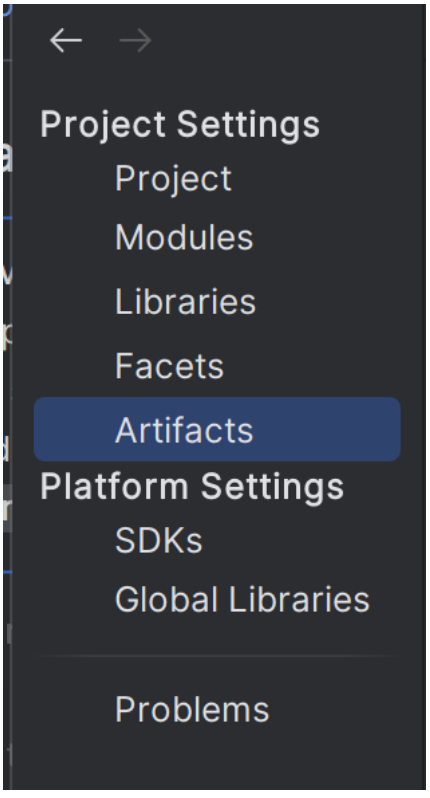
- **Resultados**: IntelliJ proporciona reportes detallados de las pruebas, permitiendo identificar fácilmente errores en el código. Los resultados incluyen tiempos de ejecución, éxito y detalles de fallos, si los hubiera.

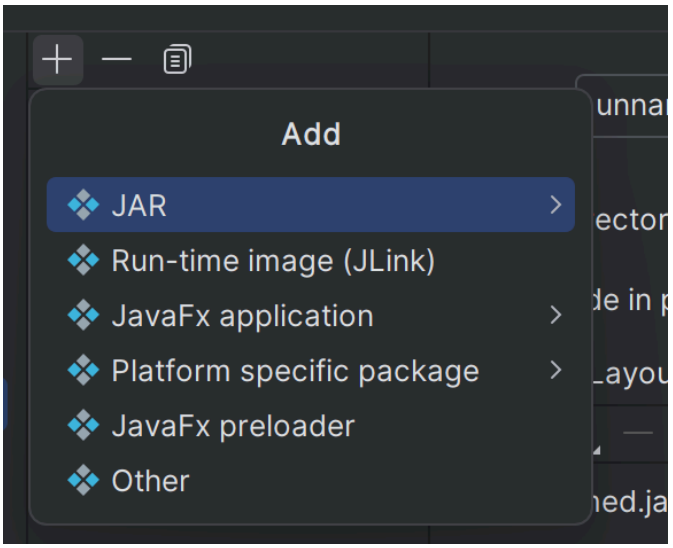
4.2 Generación de ejecutables Para empaquetar aplicaciones en IntelliJ:

1. Configurar un perfil de construcción mediante "File > Project Structure".



2. Seleccionar "Artifacts" y definir el formato de salida (JAR, WAR, etc.).





3. Ejecutar la construcción desde "Build > Build Artifacts".
4. Probar el ejecutable generado en el entorno de destino.

Ejemplo:

```
java -jar MiAplicacion.jar
```

Esto garantiza que el ejecutable funcione correctamente en cualquier sistema con Java instalado. Además, es posible empaquetar configuraciones para aplicaciones móviles o servicios web utilizando herramientas integradas como Gradle o Maven.

5. Conclusión y recomendaciones

IntelliJ se posiciona como una de las herramientas más completas para desarrolladores, especialmente en proyectos Java y Kotlin. Su robustez y facilidad de uso lo convierten en una opción preferida para proyectos de mediana y gran escala. No obstante, se deben considerar las siguientes recomendaciones:

- Contar con un equipo con suficiente memoria RAM y procesador adecuado.
- Invertir tiempo en explorar plugins y configuraciones iniciales.
- Aprovechar las herramientas de integración continua para maximizar la productividad.
- Evaluar las necesidades del proyecto para elegir entre IntelliJ Ultimate y Community Edition.

6. Fuentes consultadas

- https://medium.com/nerd-for-tech/why-you-should-choose-intellij-idea-ultimate-84656cfebcf7
- https://www.gartner.com/reviews/market/integrated-development-environment-ide-software/vendor/jetbrains/product/intellij-idea
- https://stackshare.io/stackups/eclipse-vs-intellij-idea-vs-visual-studio-code
- https://www.reddit.com/r/java/comments/13zv498/vscode_bluej_visual_studio_eclipse_intellijvim/
- https://dev.to/nisachampagne/intellij-vs-eclipse-vs-vscode-1g09
- https://www.capterra.com/p/136010/IntelliJ-IDEA/reviews/
- https://medium.com/@wagnersignoretti/first-impressions-of-intellij-ai-assistant-461d07dc7a64
- https://www.reddit.com/r/java/comments/1976lzh/is_there_ever_any_reason_not_to_use_intellij/
- https://www.softwareadvice.com/app-development/intellij-idea-profile/