

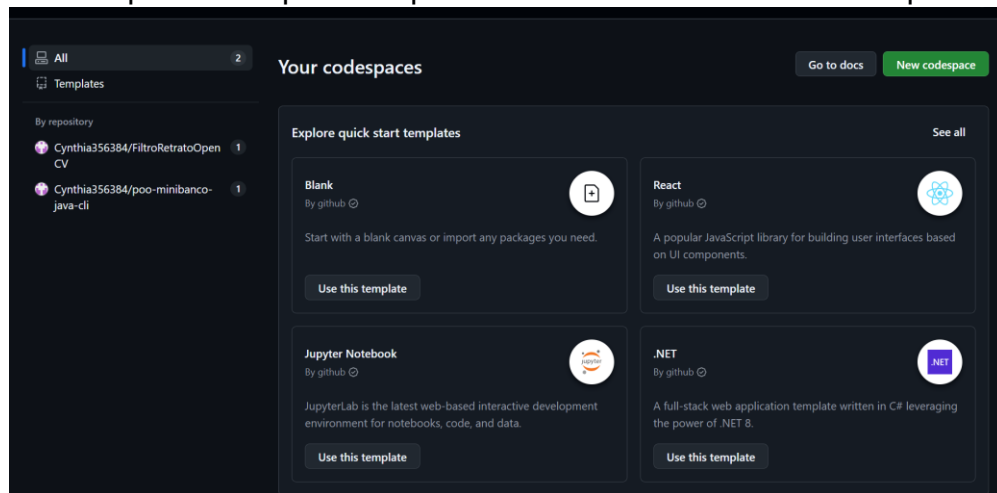
ADA3(ACTIVIDAD EN CLASE- JAVA-GITHUB-CLI)

Mini banco

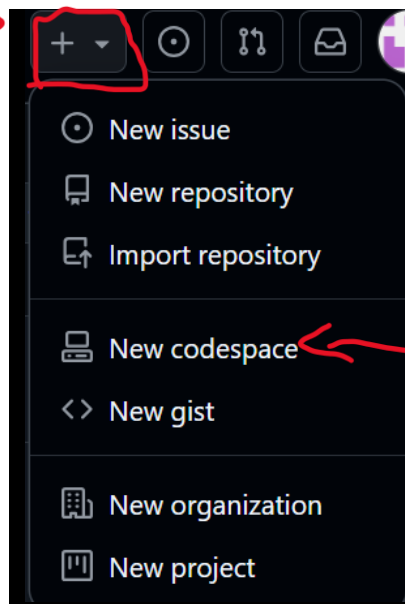
Cynthia Alessandra Aguilar uicab

ADA3(Actividad en clase-java-Github-CLI)

1. En esta practica lo primero que hicimos fue abrir nuestro codespaces

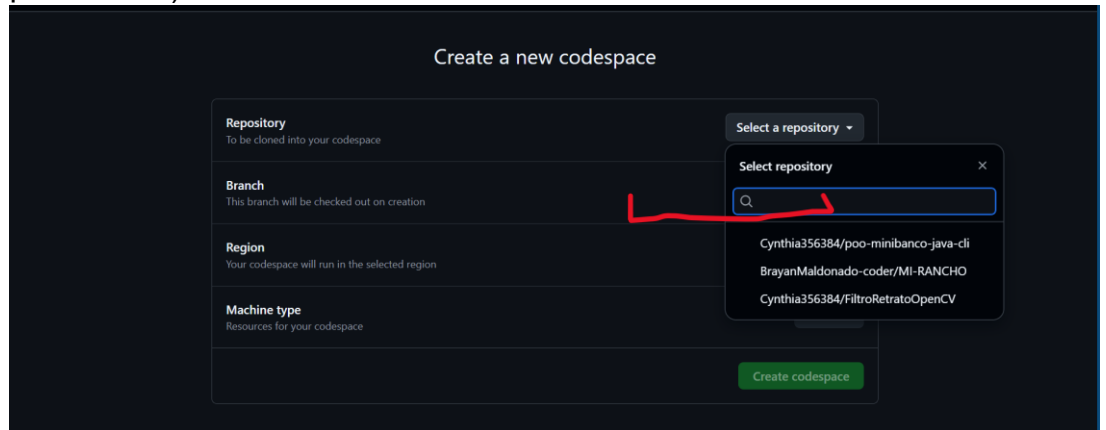


2. A continuación tenemos que crear un nuevo proyecto para poder iniciar y esto lo conseguimos a través de el signo mas que encontramos en la parte superior derecha como se muestra a continuación:

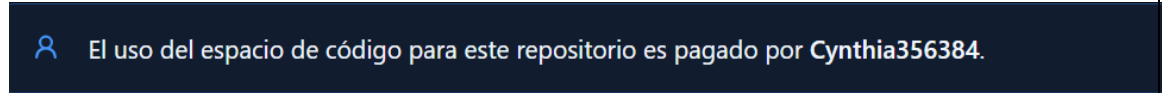


3. y ya como parte de este segundo paso le damos a new codespace.

4. Seguidamente hay que elegir en el repositorio un deposito para que este pueda ser clonado: (el señalado en rojo es el depósito donde lo pondremos)

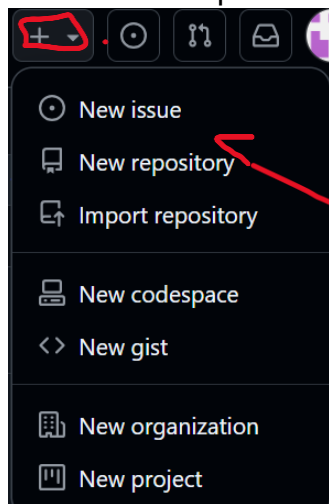


5. Una vez elegido nos dará la opción créate codespace donde podremos iniciar. Y nos saldrá algo como esto.



En dado caso de aun no tener un repositorio habrá que crearlo y esto se logra de la siguiente manera:

- Volvemos a la parte de :



- Una vez ubicado crear un nuevo repositorio le damos clic y nos mandara a la siguiente pantalla:

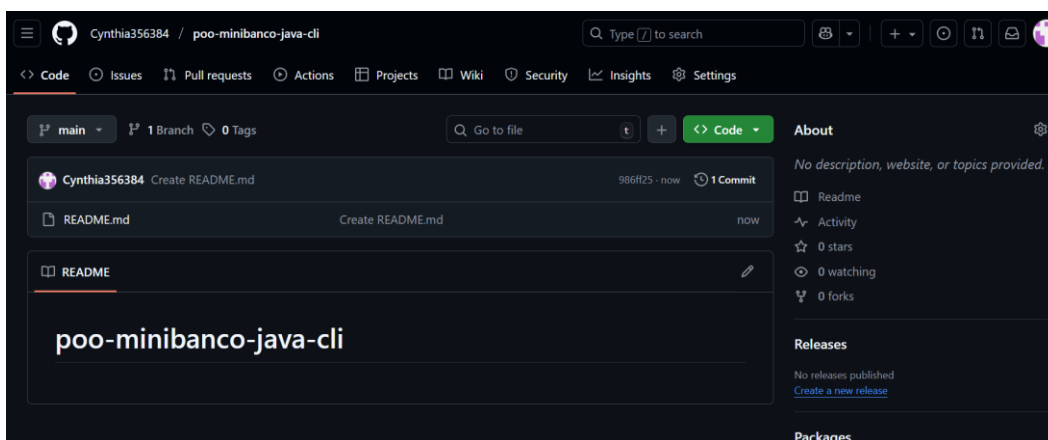
En esta pantalla nos aparecen los siguientes espacios donde en el apartado de nombre del repositorio le daremos el nombre de la actividad o el de nuestra preferencia. Y en la parte de abajo nos aparecerá el botón de crear repositorio y se creara automáticamente pero al momento de volver al apartado de crear un nuevo codespace

The screenshot shows the 'Crear un nuevo repositorio' (Create a new repository) form. It includes sections for 'General' (Owner, Repository name, Description), 'Configuración' (Visibility, README, .gitignore, License), and 'Agregar archivo README' (Add README file). The 'Repository name' field is highlighted with a red asterisk, indicating it is required.

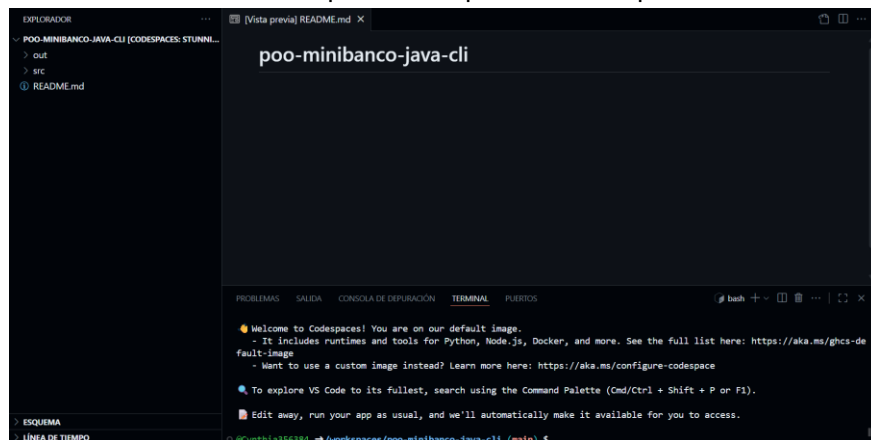
- Y al volver a el apartado de:

The screenshot shows the 'Create a new codespace' dialog. It includes fields for 'Repository', 'Branch', 'Region', and 'Machine type'. A 'Select a repository' dropdown menu is open, showing a list of repositories: 'Cynthia356384/poo-minibanco-java-cli', 'BrayanMaldonado-coder/MI-RANCHO', and 'Cynthia356384/FiltroRetratoOpenCV'. The 'Create codespace' button is at the bottom.

Y elegir el repositorio posiblemente nos aparezca que no se puede en un archivo vacío y es allá donde podemos crear un archivo cualquiera en este caso creamos un raedme y ahora si podremos iniciar el proyecto.

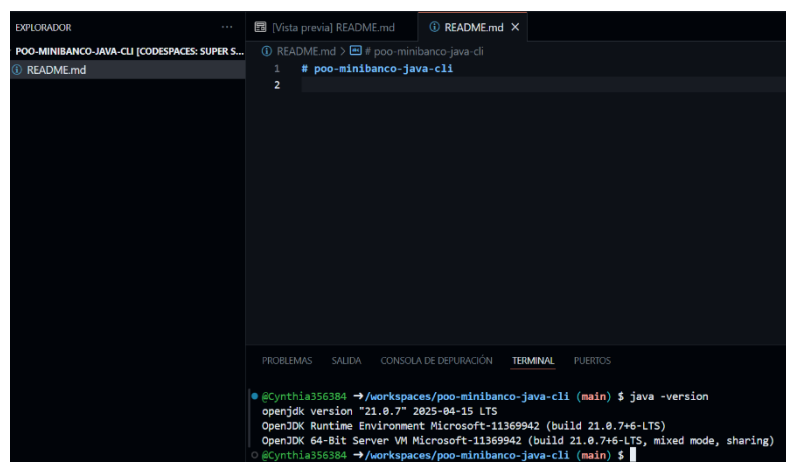


6. Una vez entrado al codespace nos aparecerá una pantalla como la siguiente



Donde pondremos a un lado del signo \$ verificaremos java con lo siguiente

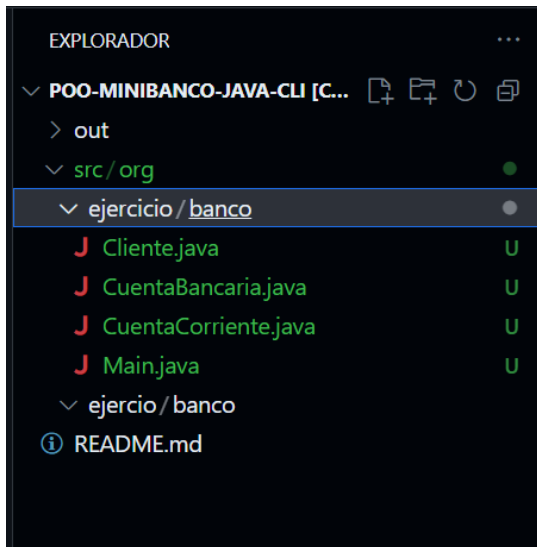
- ❖ Java -version le damos enter y ponemos el siguiente
- ❖ Javac -version y le volvemos a dar enter y queda algo así:



7. Seguidamente tenemos que crear la estructura y los archivos con lo siguiente que aparece en pantalla: a partir de donde comienza mkdir -p src/org/ejercicios/banco hasta donde dice mkdir -p out.



8. Una vez hecho el paso 7 nos debe aparecer algo como el de la imagen y ya sabremos que tiene la estructura correcta:



9. Ahora si en cada apartado podremos programar:

❖ Cliente.java:

```
1  package org.ejercicio.banco;
2  public class Cliente {
3      String nombre;
4      long dni;
5      Cliente (String str, long num) {
6          nombre = str;
7          dni = num;
8      }
9
10 }
```

❖ CuentaBancaria.java

```
Code  Blame
1  package org.ejercicio.banco;
2
3  class CuentaBancaria {
4      long numero;
5      Cliente titular;
6      long saldo;
7  CuentaBancaria(long num, Cliente clt, long s)
8  {
9      this.numero = num;
10     this.titular = clt;
11     this.saldo = s;
12 }
13 }
```

❖ CuentaCorriente.java:

```
Code Blame
1 package org.ejercicio.banco;
2 class CuentaCorreinte
3 {
4     static double interes;
5 }
```

❖ Main.java

```
Code Blame
1 package org.ejercicio.banco;
2 public class Main
3 {
4     public static void main(String[] args) {
5         Cliente cliente1 = new Cliente ("cynthia",121424);
6         CuentaBancaria cuenta1 = new CuentaBancaria(140806, cliente1, 10000);
7     }
8 }
```

10. Una vez lista los programas podremos ejecutar el programa con un: `javac -d out src/org/ejercicios/banco/*. java` y una vez que no nos marque errores tenemos que poner el control de versión básico:






- ❖ `Git status`
- ❖ `Git add`
- ❖ `Git commit -m "Actividad poo: minibanco inicial"`
- ❖ `Git push origin main`









```
Cynthia356384 →/workspaces/poo-minibanco-java-cli (main) $ git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    out/
    src/

nothing added to commit but untracked files present (use "git add" to track)
Cynthia356384 →/workspaces/poo-minibanco-java-cli (main) $ 
@Cynthia356384 →/workspaces/poo-minibanco-java-cli (main) $ git add .
@Cynthia356384 →/workspaces/poo-minibanco-java-cli (main) $
```

11. Una vez terminado esto ya tendremos en el github nuestro repositorio con el programa ya listo como se muestra en lo siguiente:

 Cynthia356384 Actividad Poo: Minibanco inicial	c1f6cec · 4 minutes ago	 2 Commits
 out/org/ejercicio/banco	Actividad Poo: Minibanco inicial	4 minutes ago
 src/org/ejercicio/banco	Actividad Poo: Minibanco inicial	4 minutes ago
 README.md	Create README.md	1 hour ago

poo-minibanco-java-cli / src / org / ejercicio / banco / 			Add file ▾	⋮
 Cynthia356384 Actividad Poo: Minibanco inicial	c1f6cec · 4 minutes ago	 History		
Name	Last commit message	Last commit date		
 ..				
 Cliente.java	Actividad Poo: Minibanco inicial	4 minutes ago		
 CuentaBancaria.java	Actividad Poo: Minibanco inicial	4 minutes ago		
 CuentaCorriente.java	Actividad Poo: Minibanco inicial	4 minutes ago		
 Main.java	Actividad Poo: Minibanco inicial	4 minutes ago		

Y con eso ya tendremos lista esta actividad.

Reflexión:

En esta actividad nos encontramos con varias dificultades ya que muchas de las cosas que vimos eran nuevas para mí pero es demasiado sencillo una vez que entendamos como crearlo lo resolvimos con ayuda del profesor que fue guiándonos pero también tuve que poner de mi parte el autoaprendizaje para que pueda llegar a concluirlo también al principio me salían varios errores los cuales leyendo pude solucionar,}.