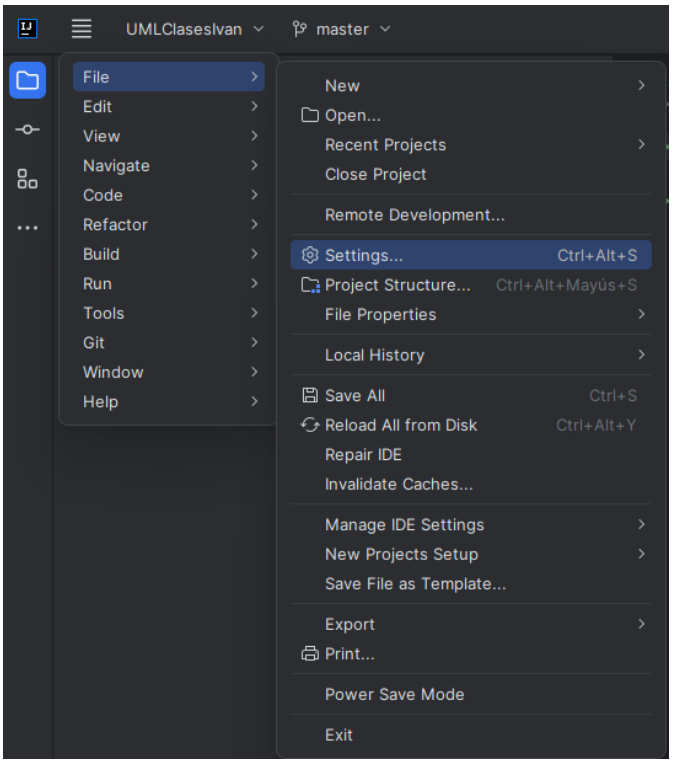
Práctica Diagramas UML Clases.

Enlace repositorio de GitHub: https://github.com/intVand/UMLClasesIvan

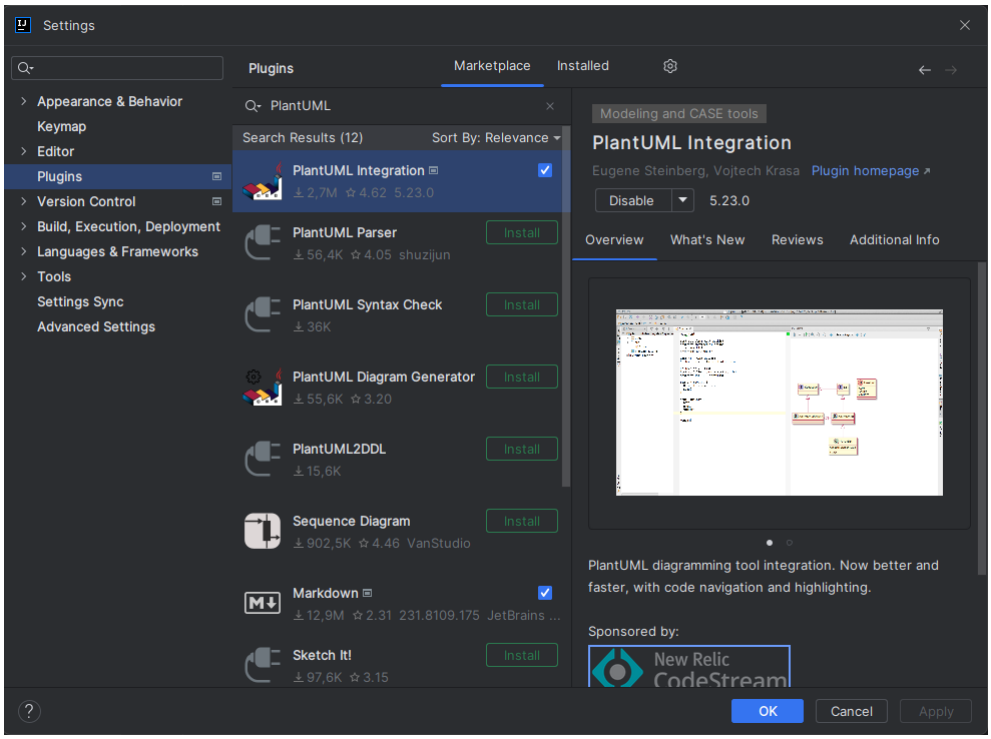
Instalación del plugin PlantUml en Intellij IDEA:



Para realizar la instalación del plugin **PlantUml** en **Intellij IDEA**, en primer lugar accederemos desde el menú superior a la opción **File → Settings**.

A continuación se nos abrirá una nueva ventana como la que aparece en la captura siguiente, en la cual tendremos que desplazarnos a la opción **Plugins**, y buscar desde el buscador superior central, la palabra **PlantUml**.

Finalmente simplemente seleccionamos sobre el botón verde que dice **Install**, solo que en mi caso ya lo tenia instalado.

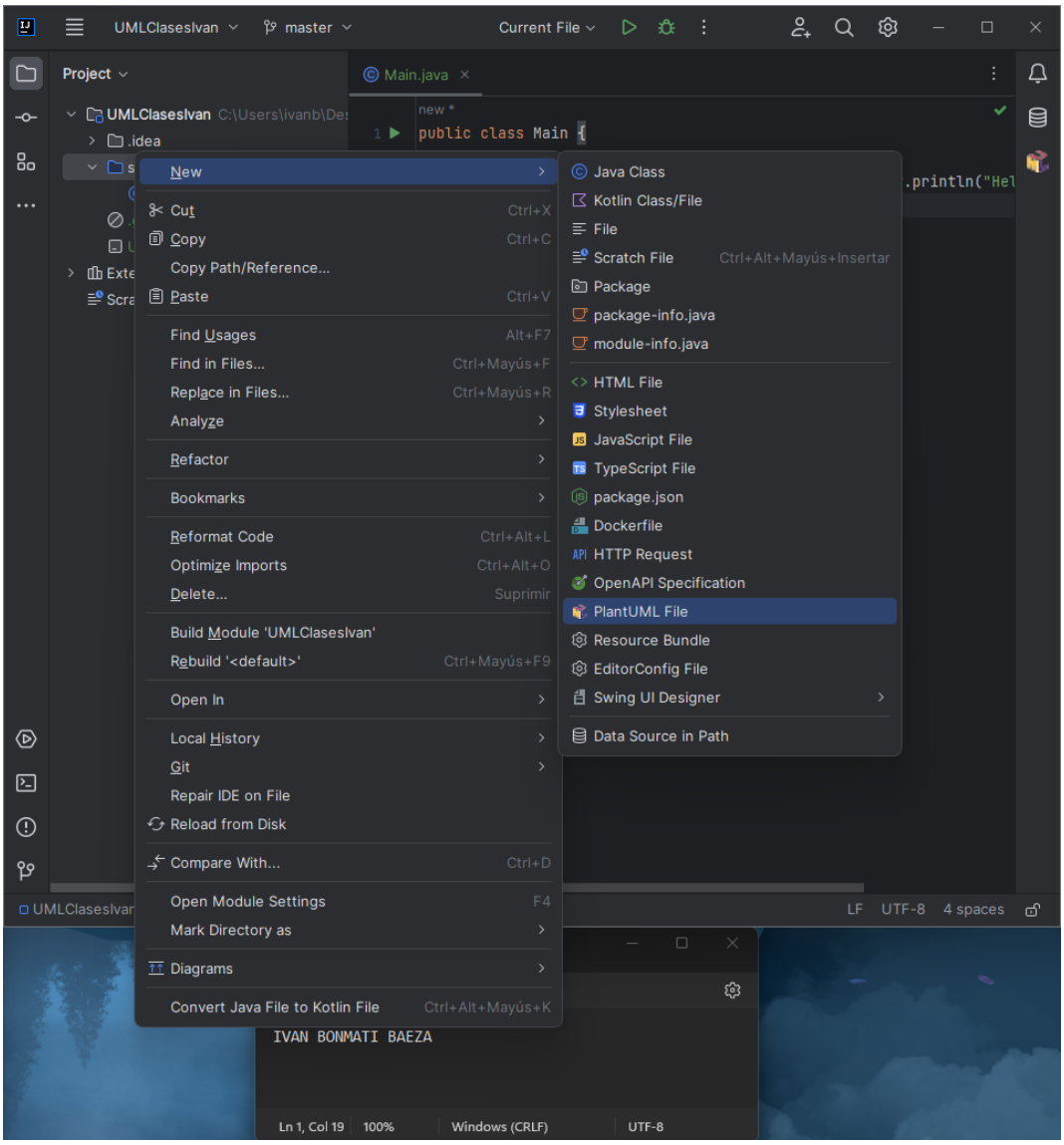


Realización de las diferentes fases de la practica:

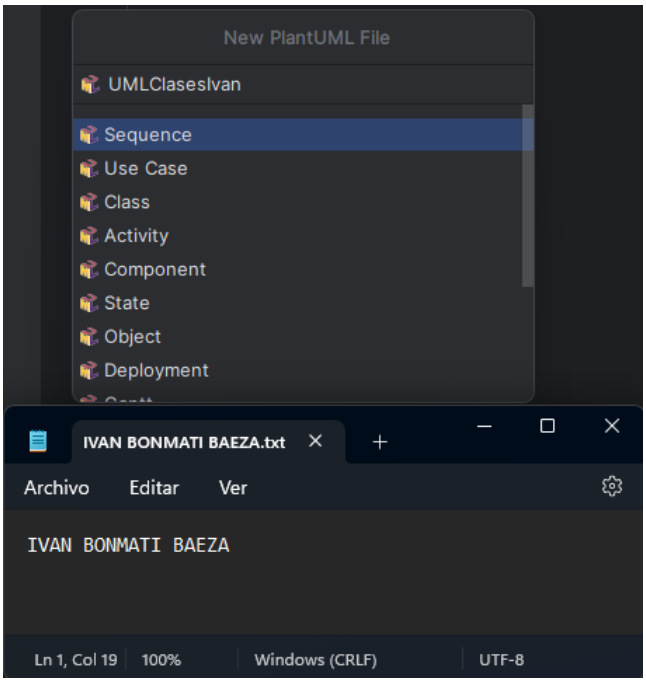
Antes de comenzar con la practica es recomendable tener activado **Git**, de esta forma podremos ir haciendo **commits** en cada una de las fases de la practica y tener una mayor seguridad en caso de error.

Además, antes de proceder con las diferentes fases de la creación del diagrama **UML**, primero tendremos que crear el archivo correspondiente de **PlantUml**, en el cual iremos escribiendo el código necesario para la creación de los diagramas **UML**.

En mi caso creare el archivo de **PlantUml** dentro de la carpeta **src** del proyecto, con el nombre **UMLClasesIvan**. Para ello hacemos clic derecho sobre la carpeta **src**, y seleccionamos la opción **New → PlantUML File** como se puede observar en la siguiente captura.



En la siguiente ventana que aparecerá, simplemente escribimos el nombre del fichero, en mi caso **UMLClaseslvan**, y pulsamos sobre la tecla **Intro**.

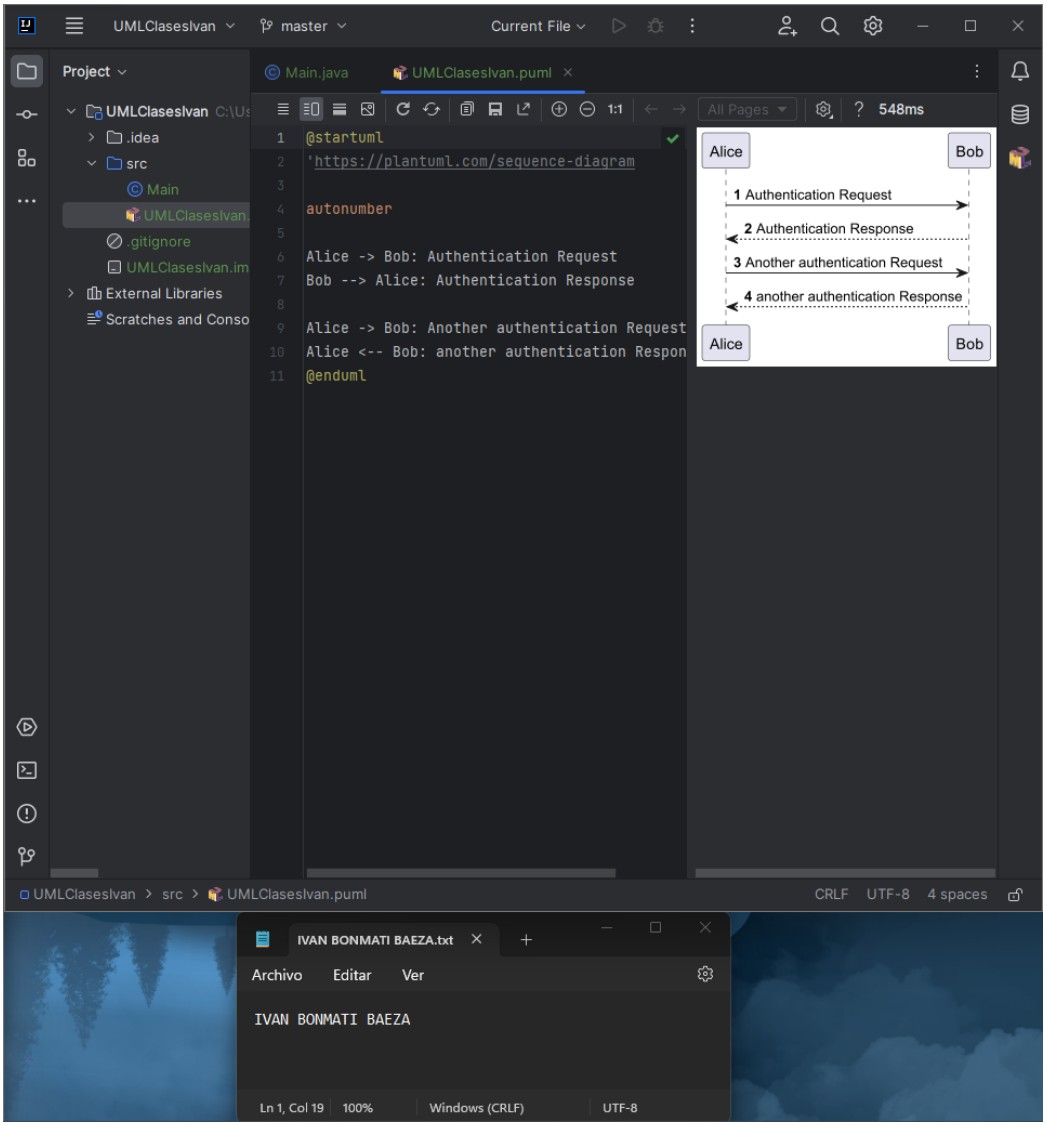


Con esto ya tendremos el fichero creado y podemos proceder a la realización de las diferentes fases

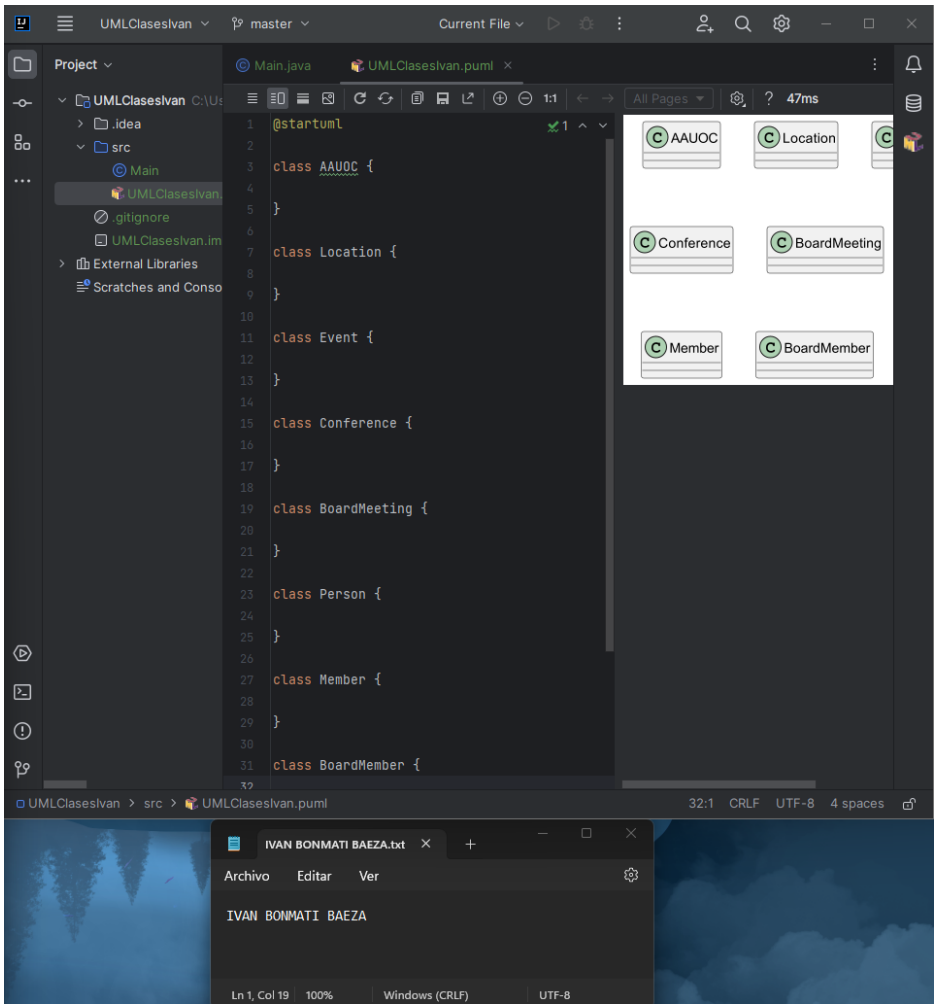
Fase 1 – Identificación de las clases:

En primer lugar y siguiendo el **PDF** de la practica realizaremos la identificación de las clases, en este caso lo que haré sera crear las diferentes clases necesarias para el diagrama **UML**.

Pero si abrimos el fichero **PlantUml** que acabamos de crear, nos encontraremos algo parecido a lo de la captura siguiente, en este caso eliminaremos todo lo que ponga dejando unicamente las frases **@startuml** en la parte superior, y **@enduml** en la parte inferior, ya que son las encargadas de delimitar el fichero.

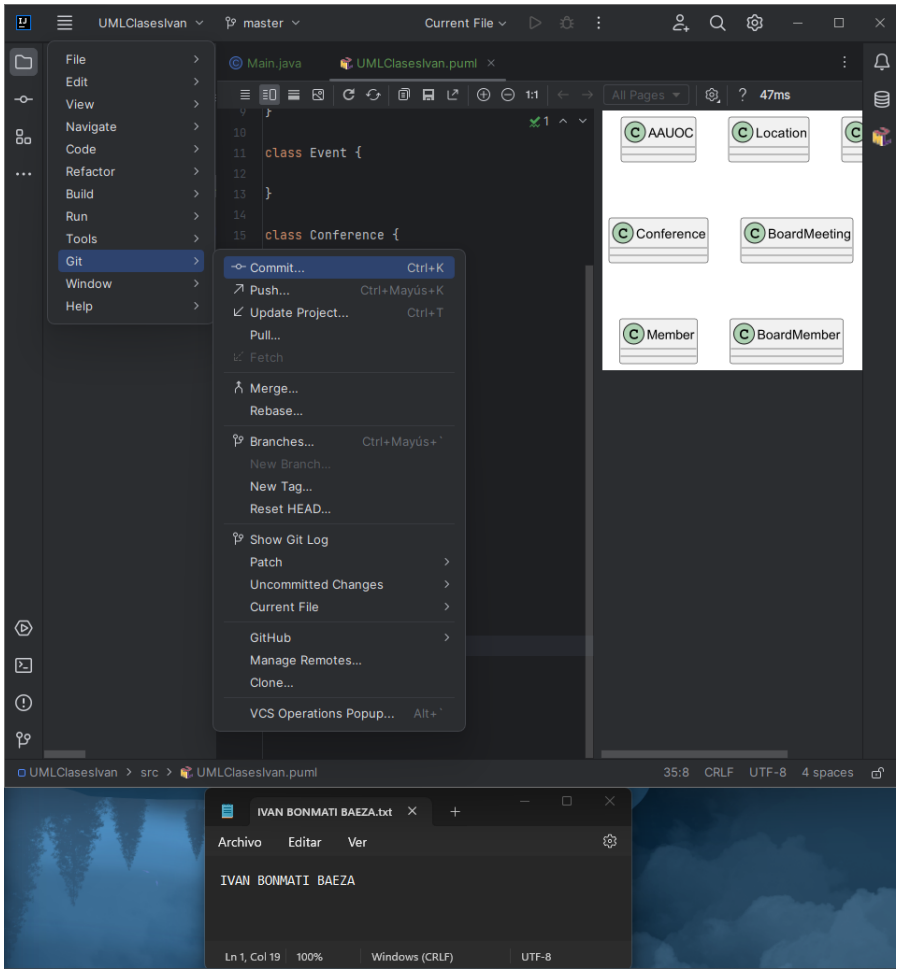


Ahora procedemos a escribir todas y cada una de las clases que vamos a necesitar para el diagrama **UML**, para ello iremos escribiendo cada una de las clases de la misma manera que se indica en la captura siguiente, y teniendo en cuenta que todo lo que hagamos tiene que estar delimitado entre el **@startuml** y el **@enduml**.

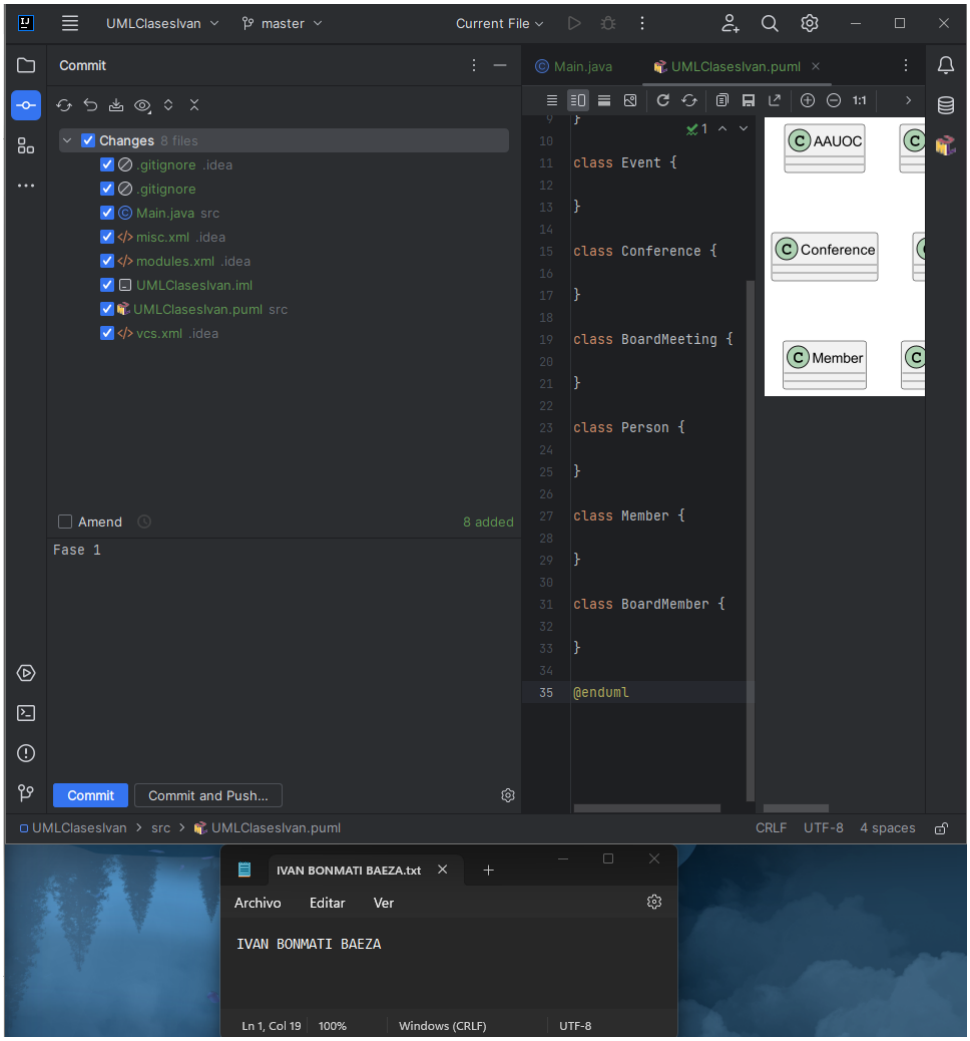


Con esto ya tendríamos todas la clases necesarias identificadas y con ello hemos finalizado la primera fase, así que lo ideal seria realizar ahora el primer **commit**.

Para ello y como se observa en la siguiente captura, desde en el menú superior seleccionaremos la opción **Git → Commit...**



Ahora en la nueva ventana que aparezca nos aseguraremos que todos los archivos que se encuentran en la zona superior izquierda estén marcados, y seguido a ello escribiremos el mensaje del **commit** justo en la parte inferior, en mi caso el mensaje será “Fase 1”.



Con esto ya tendremos nuestro primer **commit** creado, para el resto de **commits** que necesitaremos para la practica, me limitare a indicar cuando y que mensaje escribiré para no repetir la misma información múltiples veces.

Fase 2 – Creación del modelo de datos:

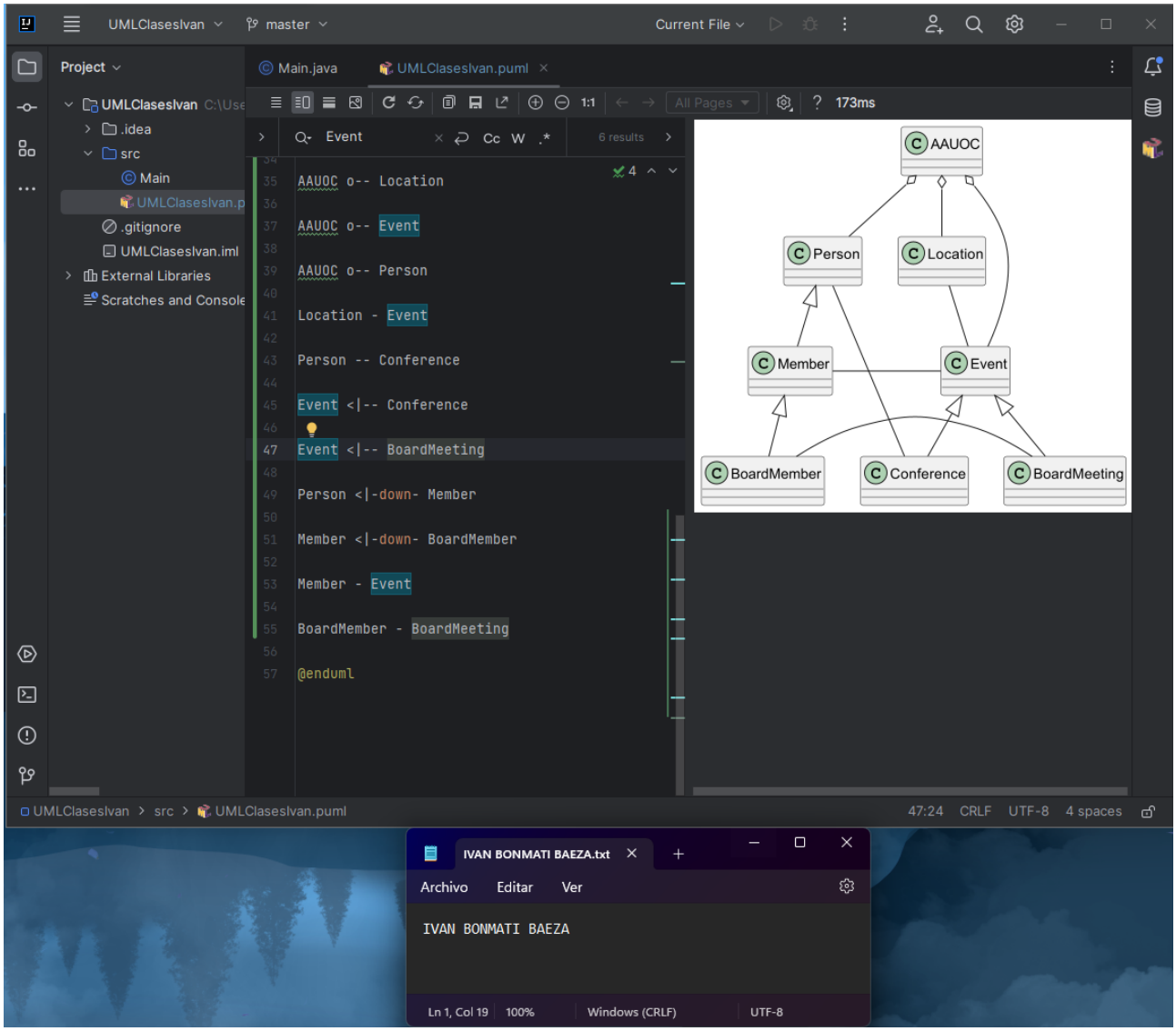
En esta fase nos centraremos en realizar las relaciones de las diferentes clases que tenemos.

Pero ordenar las diferentes clases para que quede exactamente igual al diagrama del PDF es bastante complejo ya que las flechas se mezclan entre si y se cambian las posiciones de las clases.

Por ello me he limitado a que el diagrama quede legible utilizando algunos comandos como:

<|-down- hace que la clase que este a la derecha se sitúe por debajo de la que esta a la izquierda.

- en vez de - - hace que las clases se alineen horizontalmente en vez de verticalmente.

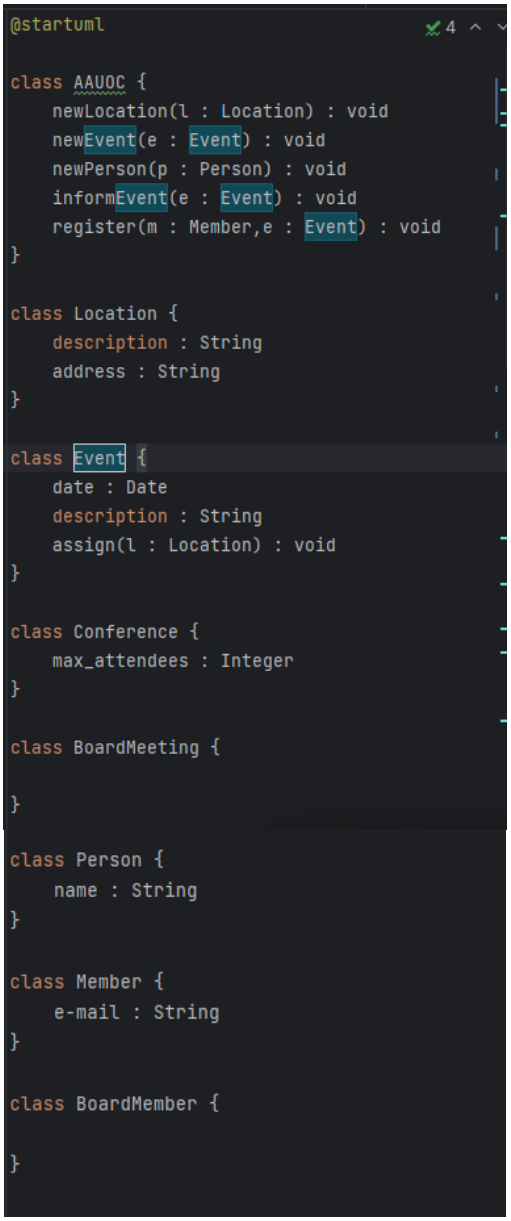


Teniendo hechas las relaciones entre las clases ya habremos finalizado la segunda fase, por ello lo siguiente que realizaremos será un nuevo **commit**, el cual en mi caso pondré de mensaje “Fase 2”.

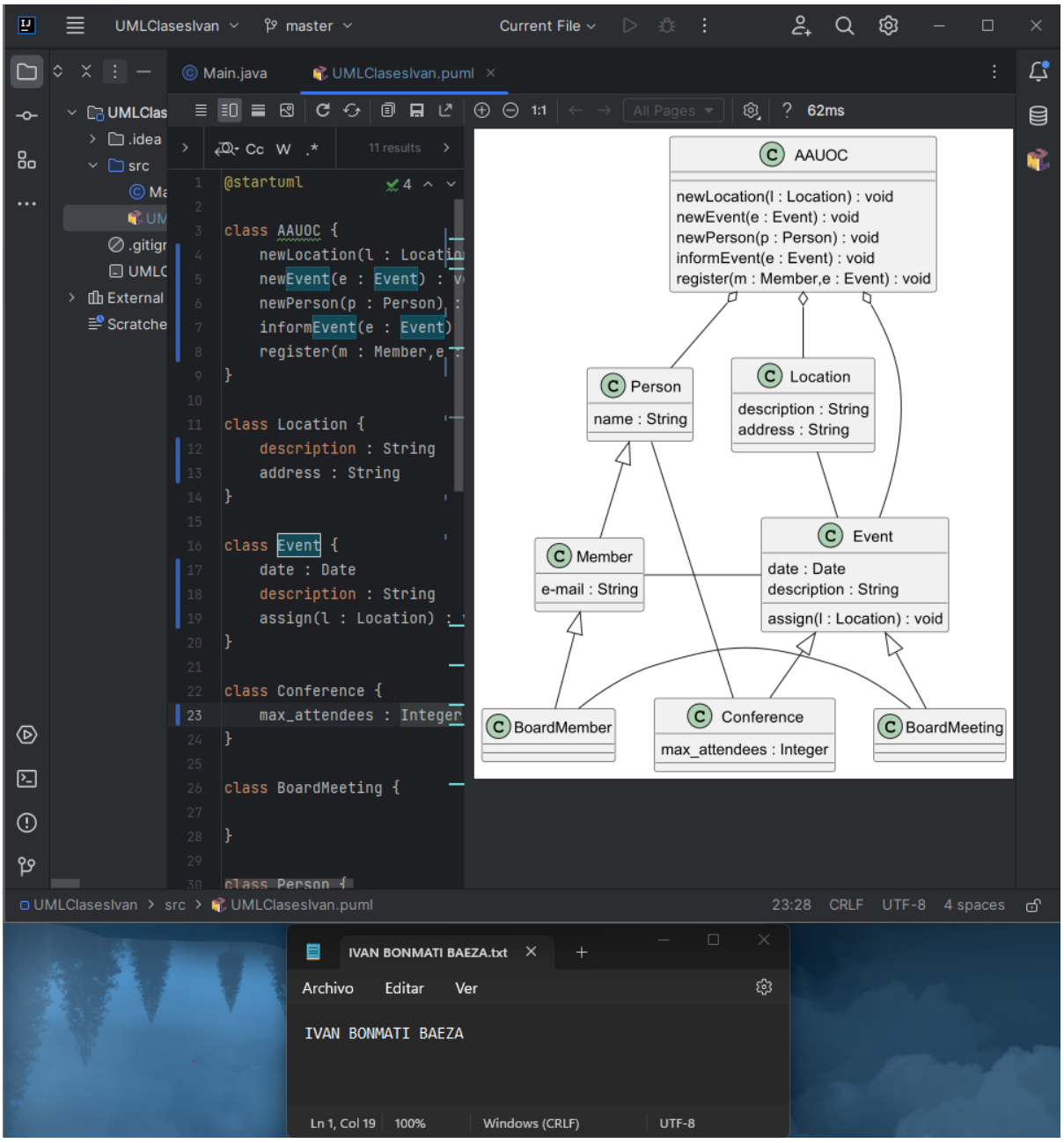
Fase 3 – Inclusión de atributos y métodos:

En esta tercera fase nos centraremos en añadir los atributos y los métodos de las diferentes clases que disponemos.

En esta ocasión en el **PDF** a seguir no se hace ninguna referencia al uso de modificadores de acceso, por ello como hay que limitarse a seguir el **PDF** no los he puesto, pero lo ideal sería que los **atributos** tuviesen modificador de acceso **private** (y acceder a ellos mediante métodos **getters** y **setters**), y que los **métodos** tuviesen modificador de acceso **public**.



Para añadir tanto los **atributos**, como los **métodos** vistos en la anterior captura, hay que escribir su respectivo **código** entre las **{ }** del nombre de su respectiva **clase**.



Finalmente nos quedara un diagrama como el de la captura superior.

A continuación, crearemos un nuevo **commit** al cual llamaremos “Fase 3”.

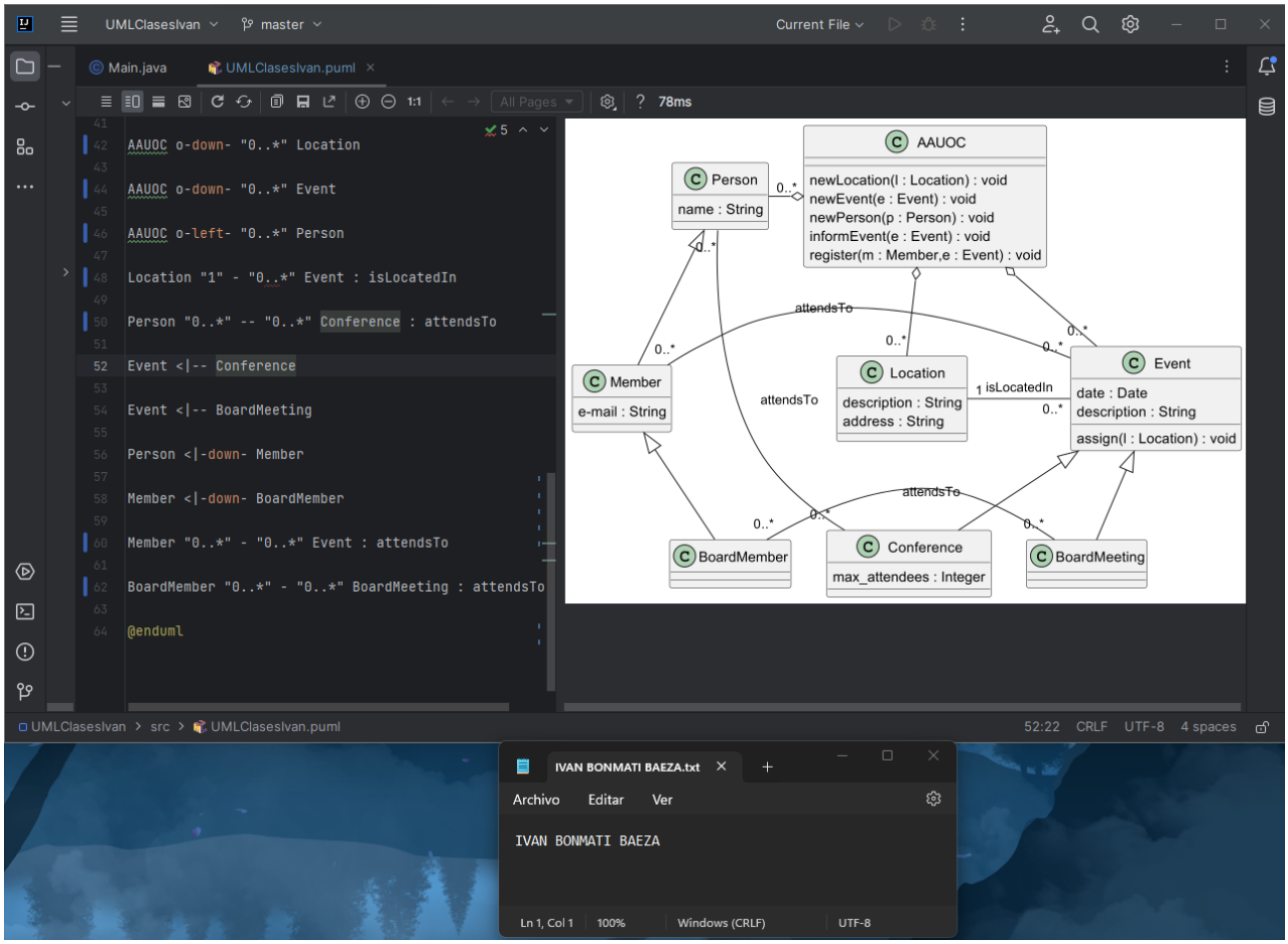
Fase 4 – Inclusión de la cardinalidad y navegabilidad de las relaciones:

En esta cuarta fase nos centraremos en incluir las **navegabilidades** y las **cardinalidades** a las relaciones que tenemos.

Para añadir las **cardinalidades** simplemente escribiremos la cardinalidad entre comillas en el lado correcto de la relación.

Algo como lo siguiente: **Person "0..*" -- "0..*" Conference**

Para el caso de las **navegabilidades**, tendremos que añadir estas escribiéndolas después de añadir **:** al final de la relación, algo como lo siguiente:
Person - - Conference : attendsTo



Como se podrá observar en la captura las posiciones de las clases en el diagrama han sido cambiadas, esto es así debido a que al escribir las **cardinalidades** y las **navegabilidades** estas se habían desordenado por completo, haciéndolas totalmente ilegibles.

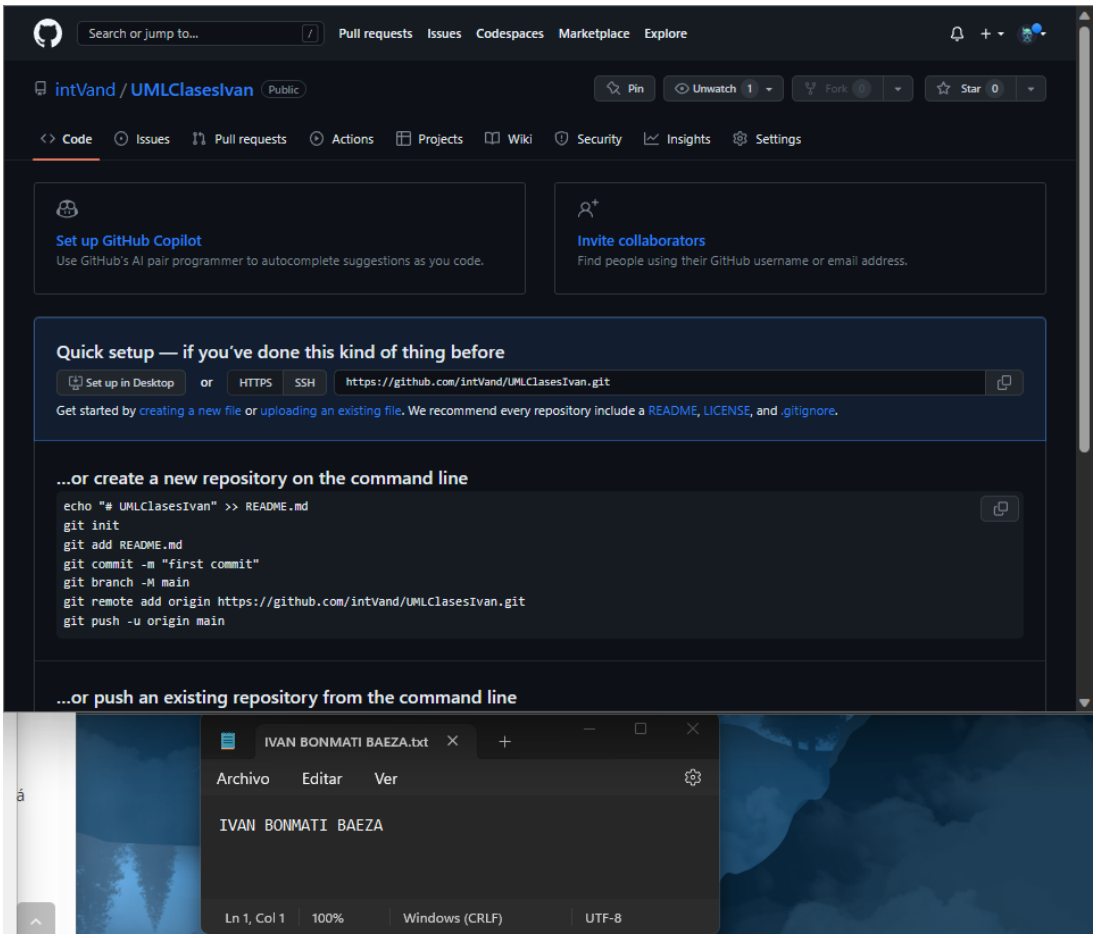
Por ello he tenido que re colocar las relaciones usando código como: **o-down-**, para que mantuviesen un diseño más legible y no tan caótico.

Por ultimo para terminar con la cuarta fase crearemos un ultimo **commit**, al cual llamaremos “Fase 4”.

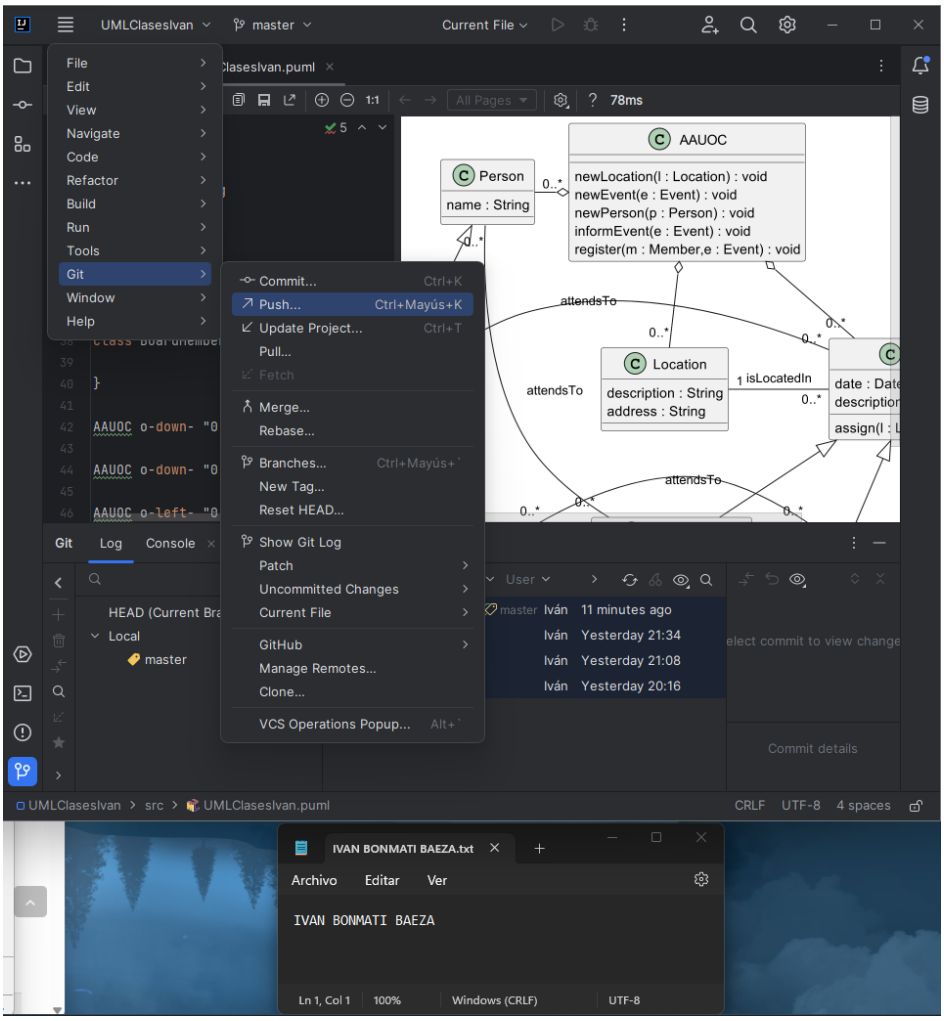
Subir el proyecto a GitHub:

Para subir el proyecto a **GitHub** primero necesitaremos crear el repositorio en la página oficial de **GitHub**, en mi caso el nombre del repositorio será **UMLClasesIvan**.

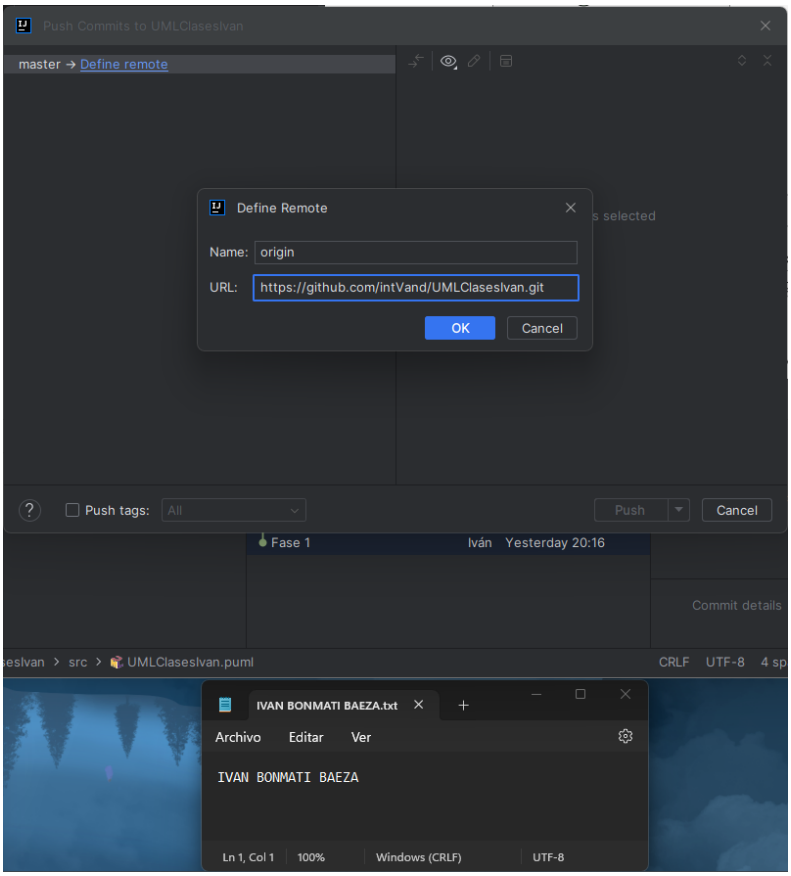
Una vez creado nos quedará algo como lo que aparece en la captura.



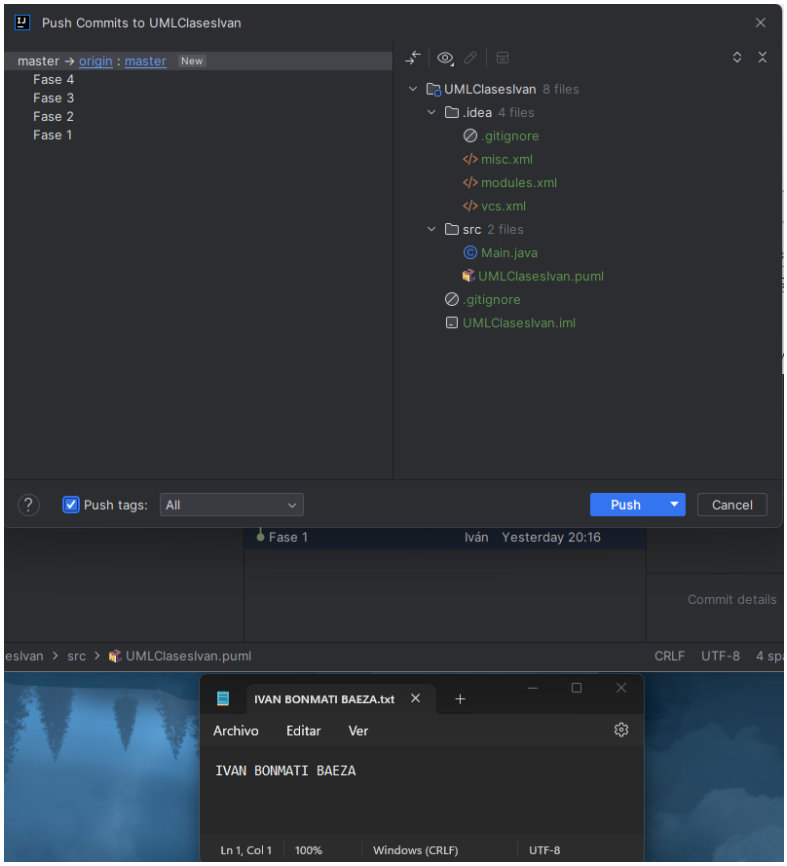
Ahora desde **Intellij** accederemos desde el menú superior a la opción **Git → Push...**



Nos aparecerá la ventana de la siguiente captura, en la cual tendremos que pegar la **URL** de nuestro repositorio recién creado y pulsar sobre el botón “Ok”.



Ahora nos aparecerá algo parecido a lo de la siguiente captura, en este caso como esta todo correcto pulsaremos sobre el botón **Push** para que se suba el repositorio.



Con esto ya tendremos el repositorio subido correctamente a **GitHub**.