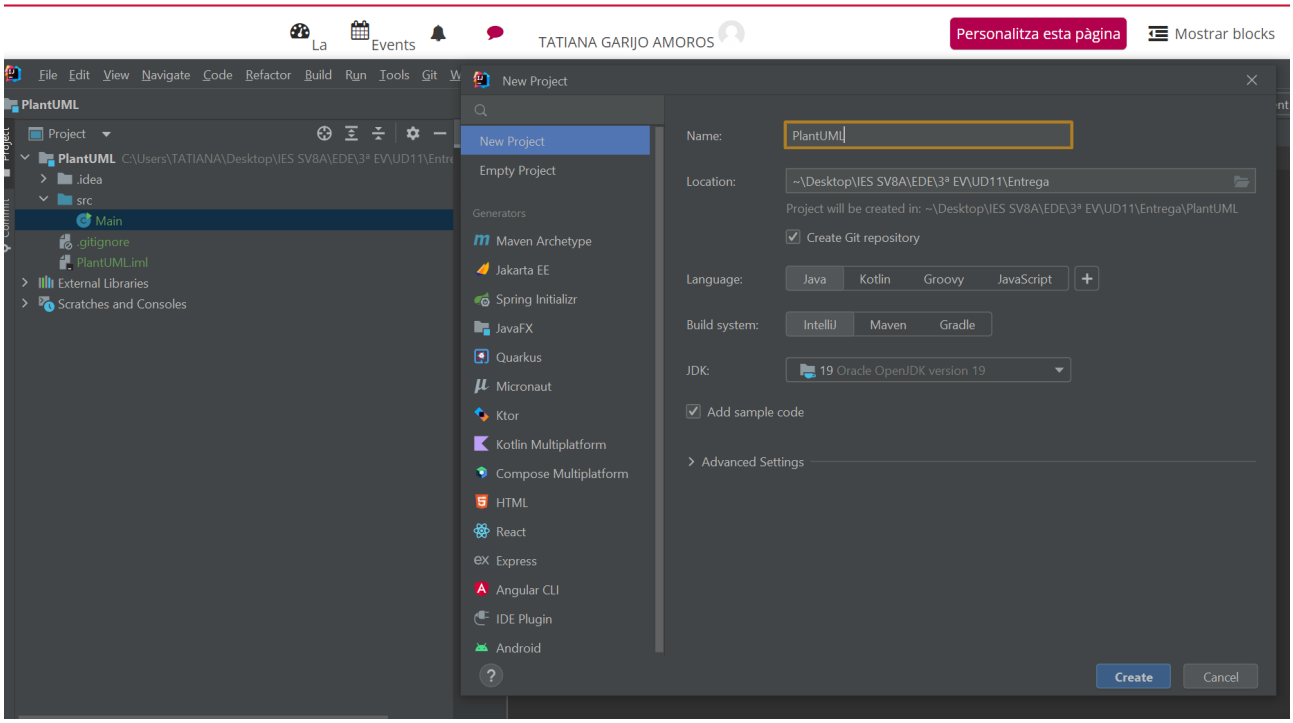
PRÁCTICA Diagramas UML Clases

Repositorio GitHub

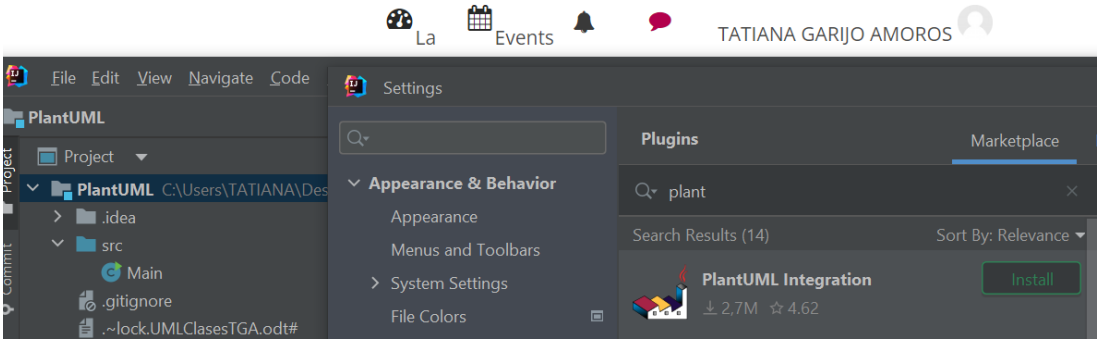
https://github.com/tatianagarijo/UMLClasesTatianaGarijo.git

GENERALITAT VALENCIANA Aules

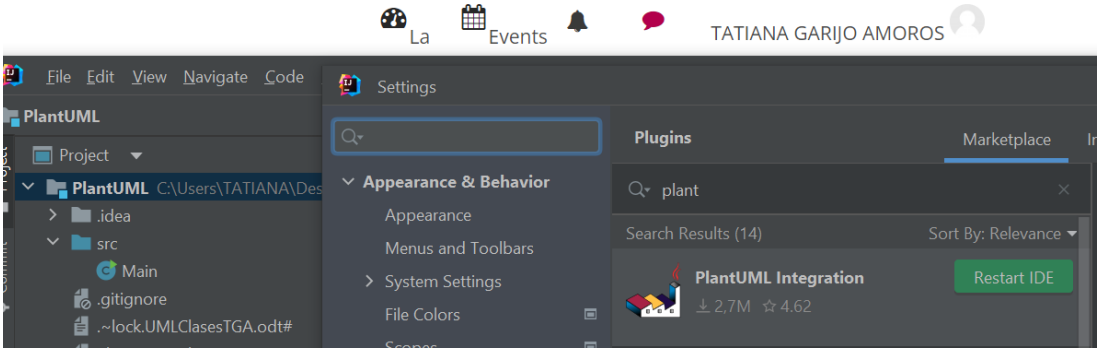


- New Project IntelliJ
- Abrir IntelliJ
- Seleccionar la opción New Project
- Configurar las características del proyecto
  - En mi caso he seleccionado
    - Nombre del proyecto
    - Languaje: Java
    - Build System: IntelliJ
    - version 19 JDK
    - Create Git repository

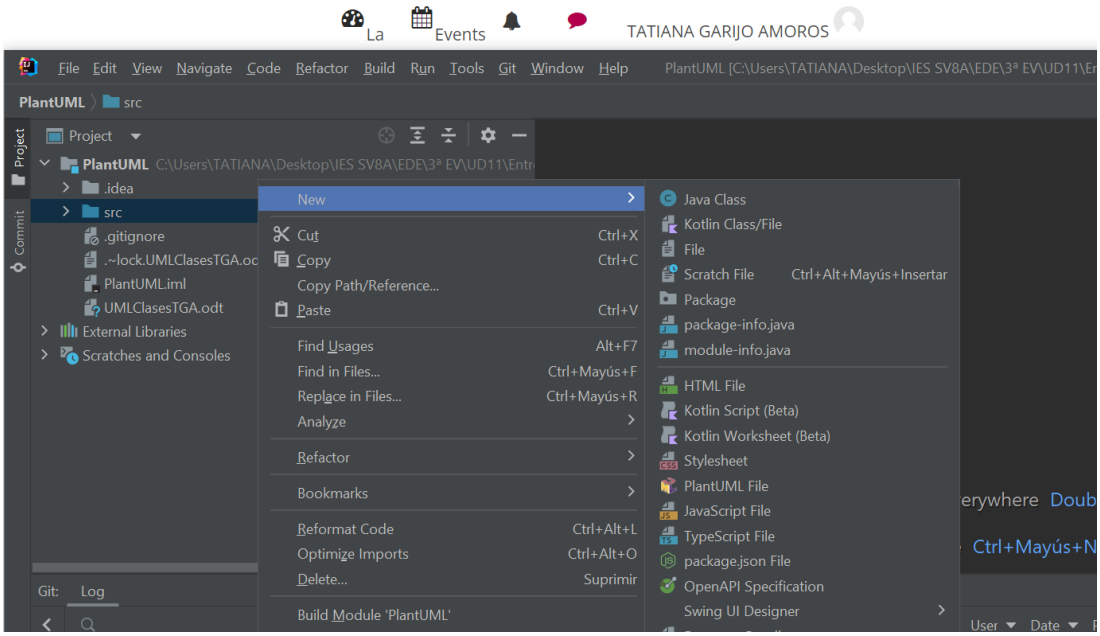
- # PlantUML
  - PlantUML es una herramienta que nos permite crear diagramas tipo UML.
  - Instalamos el plugin PlantUML en Itellij.
    - *En la web oficial https://plantuml.com/es/ podemos encontrar toda la información del plugin.*
    - Navegamos por el menú File --> Settings -->Plugins
      - Buscamos el plugin PlantUML haciendo uso del buscador, una vez localizado hacemos clic sobre el botón install.



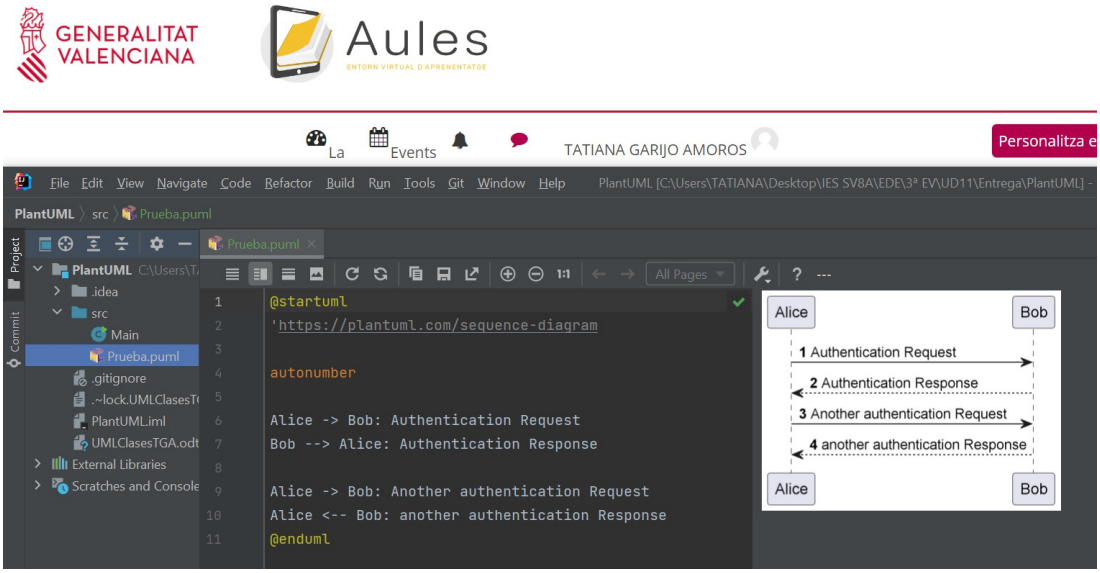
- El IDE nos pide reiniciar para cargar el plugin, hacemos clic sobre el botón Restart IDE



- Hacemos clic botón derecho sobre la carpeta src --> new --> PlantUML File

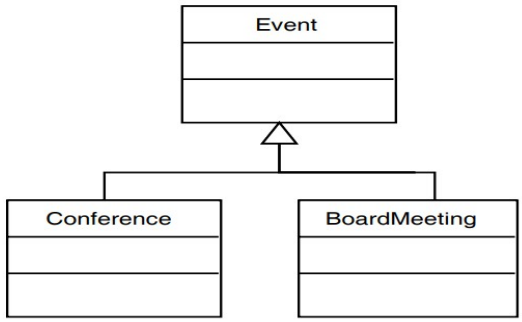


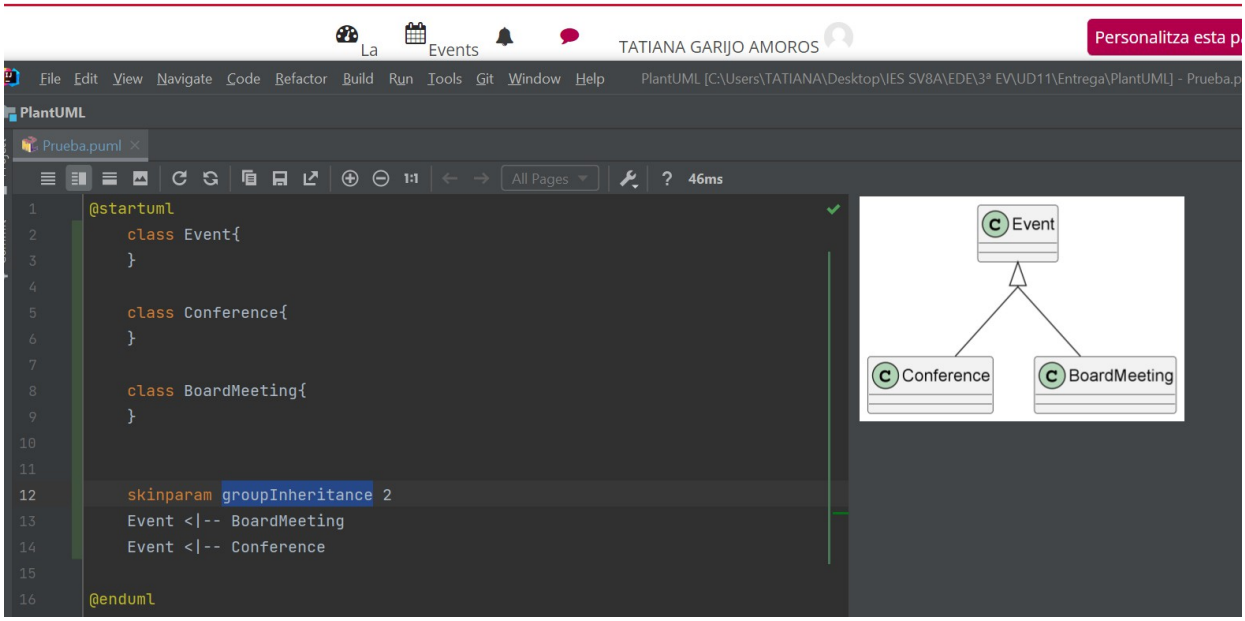
- Le damos el nombre de nuestra elección y nos aparece la siguiente ventana



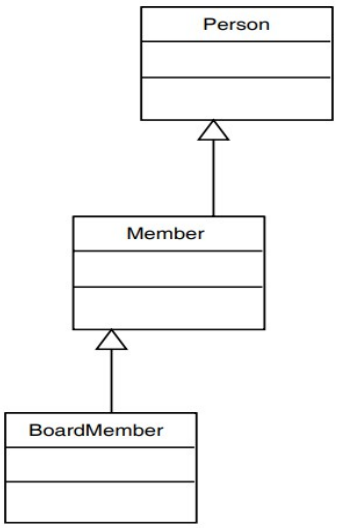
- En el panel central entre las instrucciones @startuml @enduml iremos escribiendo el código necesario para crear el diagrama, en el panel derecho irá apareciendo el diagrama según el código que vayamos generando.
- Para crear los diagramas hacemos uso de la información que nos facilita la web de PlantUML (https://plantuml.com/es/class-diagram), copiamos los fragmentos de código necesarios para crear el diagrama requerido.

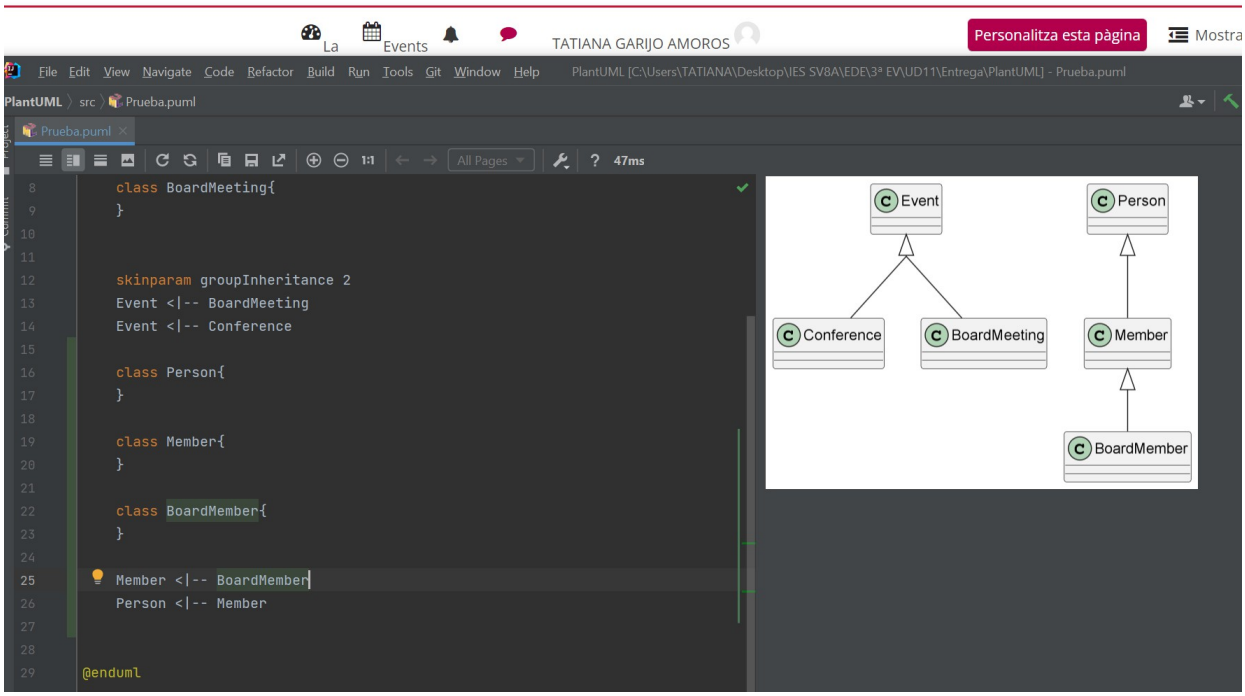
- Replicando las fases del ejemplo resuelto



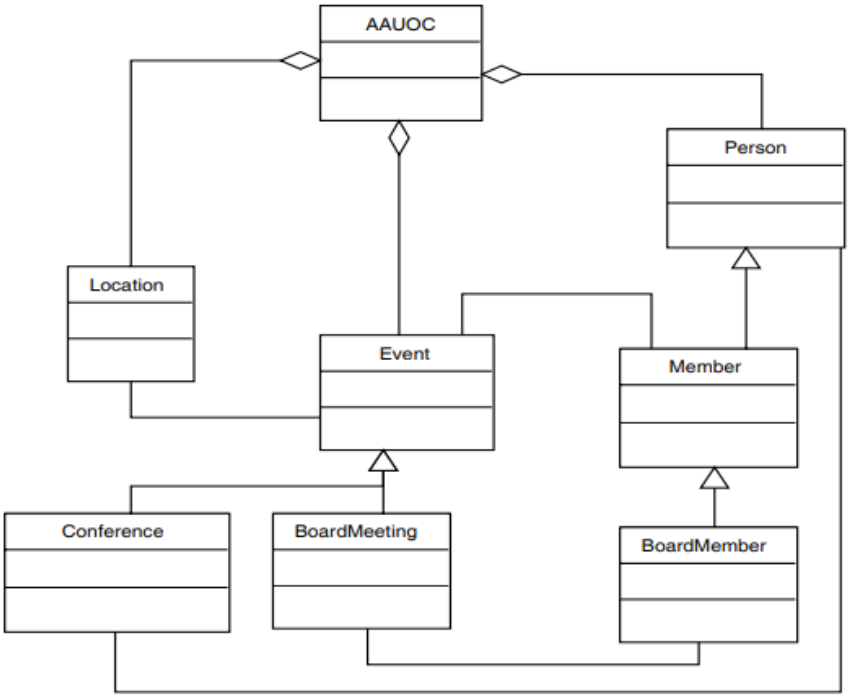


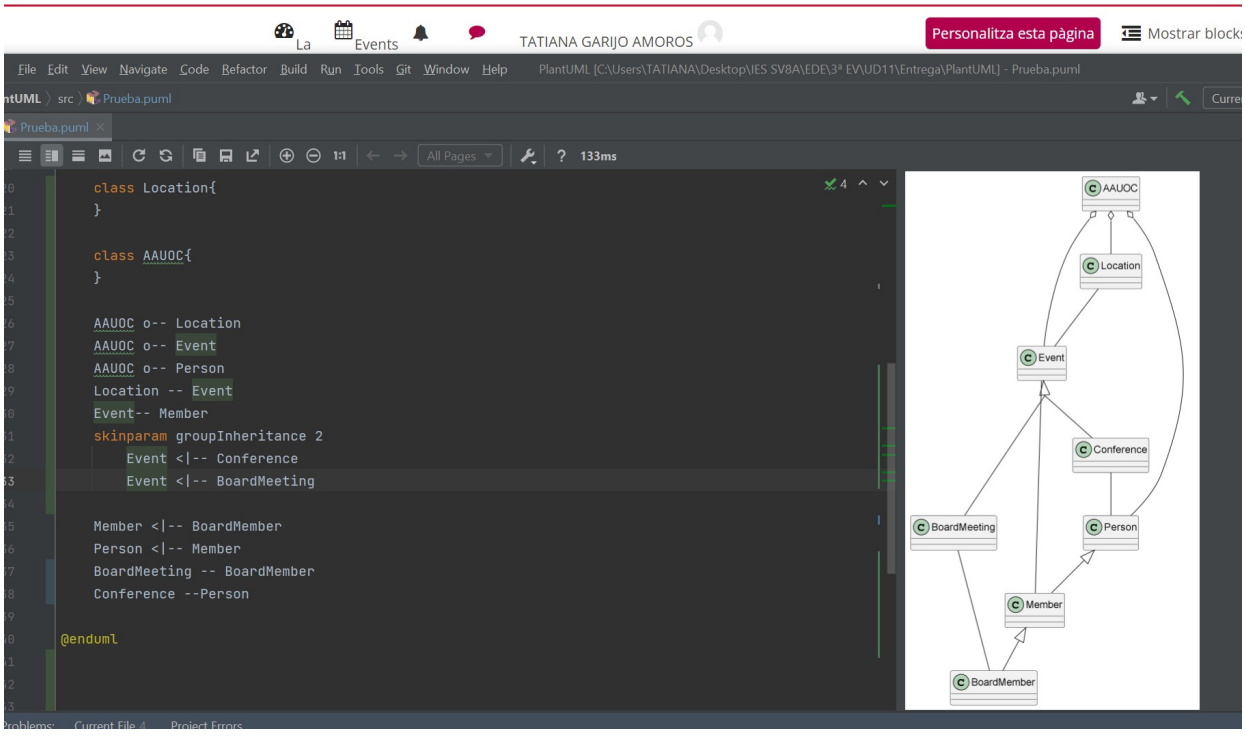
- Declarars las clases:
  - class nombreClase{}
  - Declarar jerarquía de herencia indicamos:
    - skinparam groupInheritance
  - Declarar una relación de tipo extensión:
    - nombreClaseBase <|-- nombreClaseDerivada



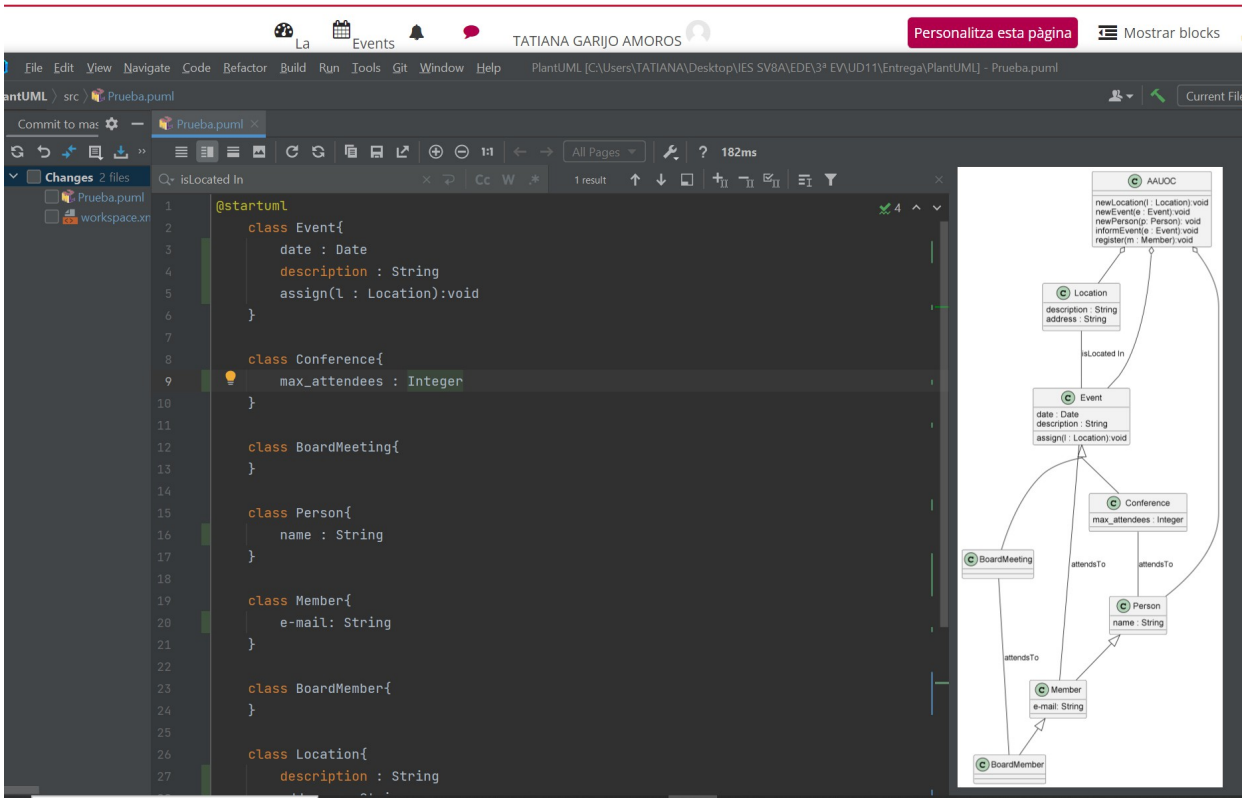


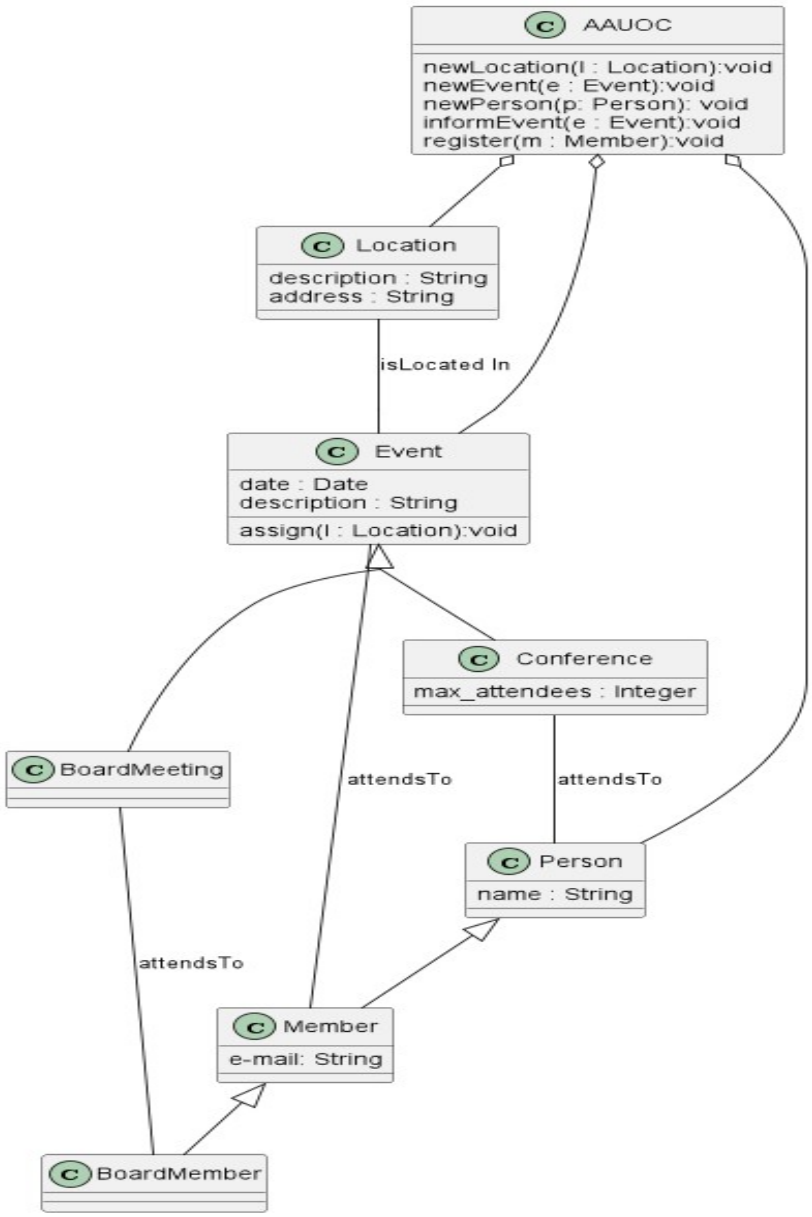
- Declarar las clases:
  - class nombreClase{}
  - Declarar una relación de tipo extensión:
    - <|--


AAUOC
Person
Location
Event
Member
Conference
BoardMeeting
BoardMember


TATIANA GARIJO AMOROS
Personalitza esta pàgina
Mostrar block
File Edit View Navigate Code Refactor Build Run Tools Git Window Help
PlantUML [C:\Users\TATIANA\Desktop\IES SV8A\EDE\3ª EV\UD11\Entrega\PlantUML] - Prueba.puml
Prueba.puml
133ms
class Location{
}
class AAUOC{
}
AAUOC o-- Location
AAUOC o-- Event
AAUOC o-- Person
Location -- Event
Event-- Member
skinparam groupInheritance 2
Event <|-- Conference
Event <|-- BoardMeeting
Member <|-- BoardMember
Person <|-- Member
BoardMeeting -- BoardMember
Conference --Person
@enduml
AAUOC
Location
Event
Conference
BoardMeeting
Person
Member
BoardMember

- Declarar las clases:
  - class nombreClase{}
  - Declarar una relación de tipo extensión:
    - <|--
  - Declarar una una relación de tipo agregación:
    - o--
  - Declarar una una relación de tipo asociación:
    - --

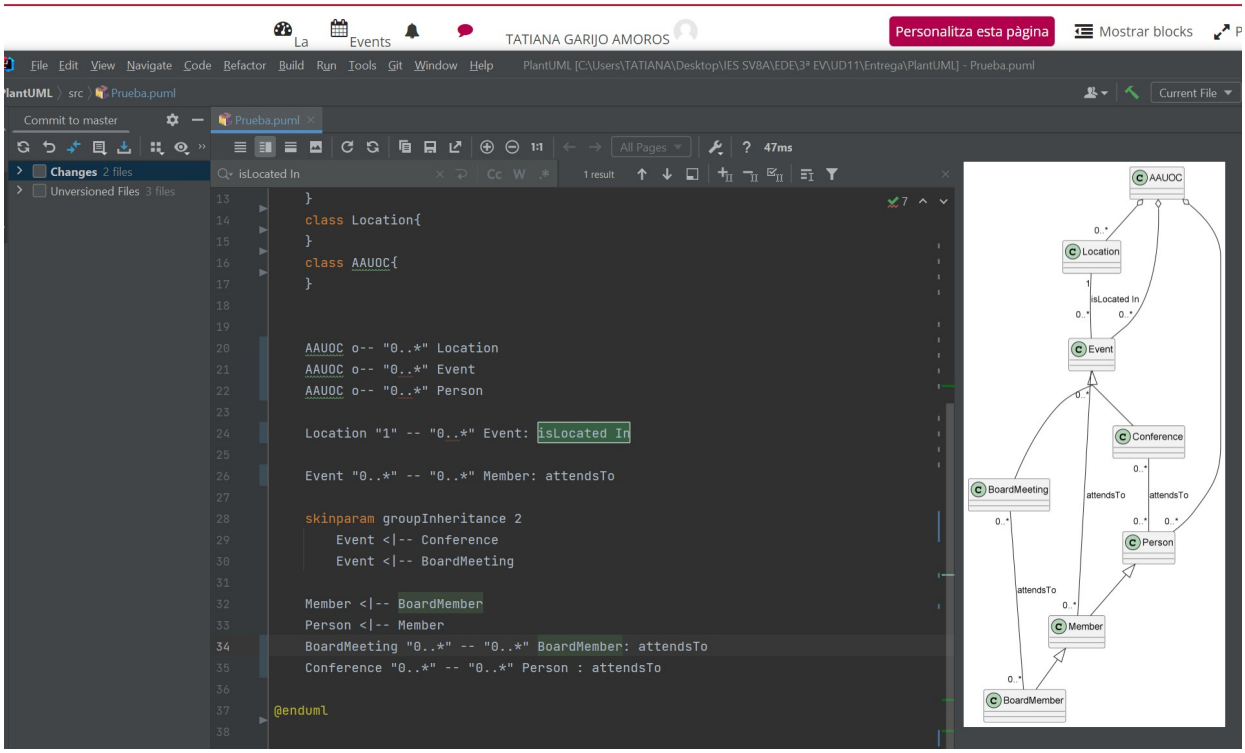



AAUOC
newLocation(l : Location):void
newEvent(e : Event):void
newPerson(p: Person): void
informEvent(e : Event):void
register(m : Member):void
Location
description : String
address : String
isLocated In
Event
date : Date
description : String
assign(l : Location):void
Conference
max_attendees : Integer
BoardMeeting
attendsTo
attendsTo
Person
name : String
attendsTo
Member
e-mail: String
BoardMember

```
@startuml
    class Event{
        date : Date
        description : String
        assign(l : Location):void
    }
    class Conference{
        max_attendees : Integer
    }
    class BoardMeeting{
    }
    class Person{
        name : String
    }
    class Member{
        e-mail: String
    }
    class BoardMember{
    }
    class Location{
        description : String
        address : String
    }
    class AAUOC{
        newLocation(l : Location):void
        newEvent(e : Event):void
        newPerson(p: Person): void
        informEvent(e : Event):void
        register(m : Member):void
    }

    AAUOC o-- Location
    AAUOC o-- Event
    AAUOC o-- Person
    Location -- Event : isLocated In
    Event-- Member: attendsTo
    skinparam groupInheritance 2
        Event <|-- Conference
        Event <|-- BoardMeeting
    Member <|-- BoardMember
    Person <|-- Member
    BoardMeeting -- BoardMember : attendsTo
    Conference --Person : attendsTo
@enduml
```

- Añadir los atributos y métodos indicando los parámetros de entrada y salida, dentro de la clase correspondiente
  - En este caso se ha elegido la sintaxis:
    - *nombreMétodo (nombre : tipo) : tipoparámetroSalida*
    - `newLocation(l : Location):void`
- Añadir anotaciones a los enlaces:
  - declaración de relación : texto_mostrar
  - `Conference -- Person : attendsTo`



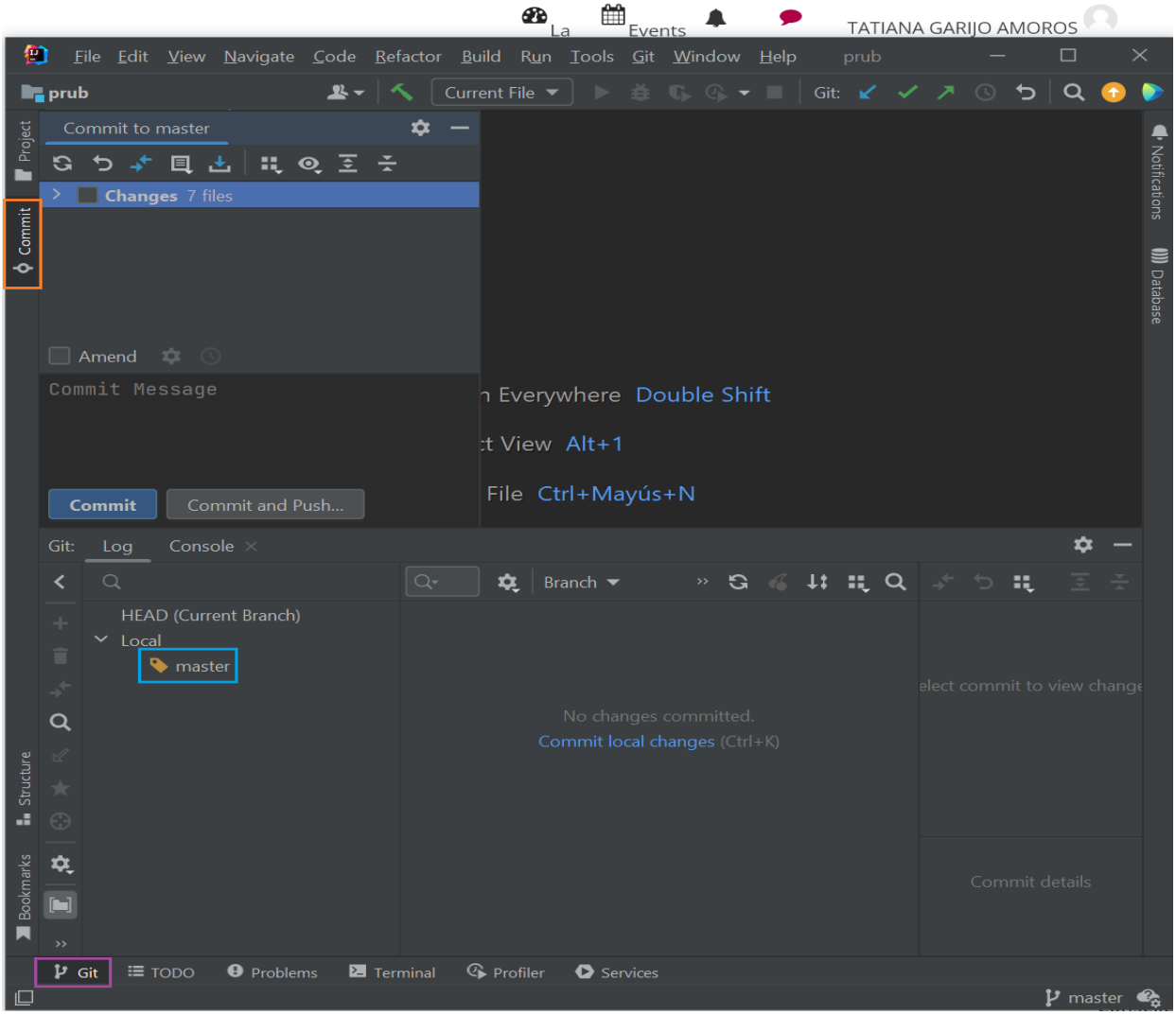
- Borrar los atributos y métodos como se indica en el enunciado.
- Añadir las cardinalidades:
  - introducir entre comillas la cardinalidad “0..*” a cada lado de la relación:
  - `Conference "0..*" -- "0..*" Person : attendsTo`
  -

**Casi al final de la práctica me he dado cuenta que había una forma de ir guardando las imágenes del diagramas, he creado una carpeta dentro del proyecto, he vuelto a generar los diagramas y los he guardado.

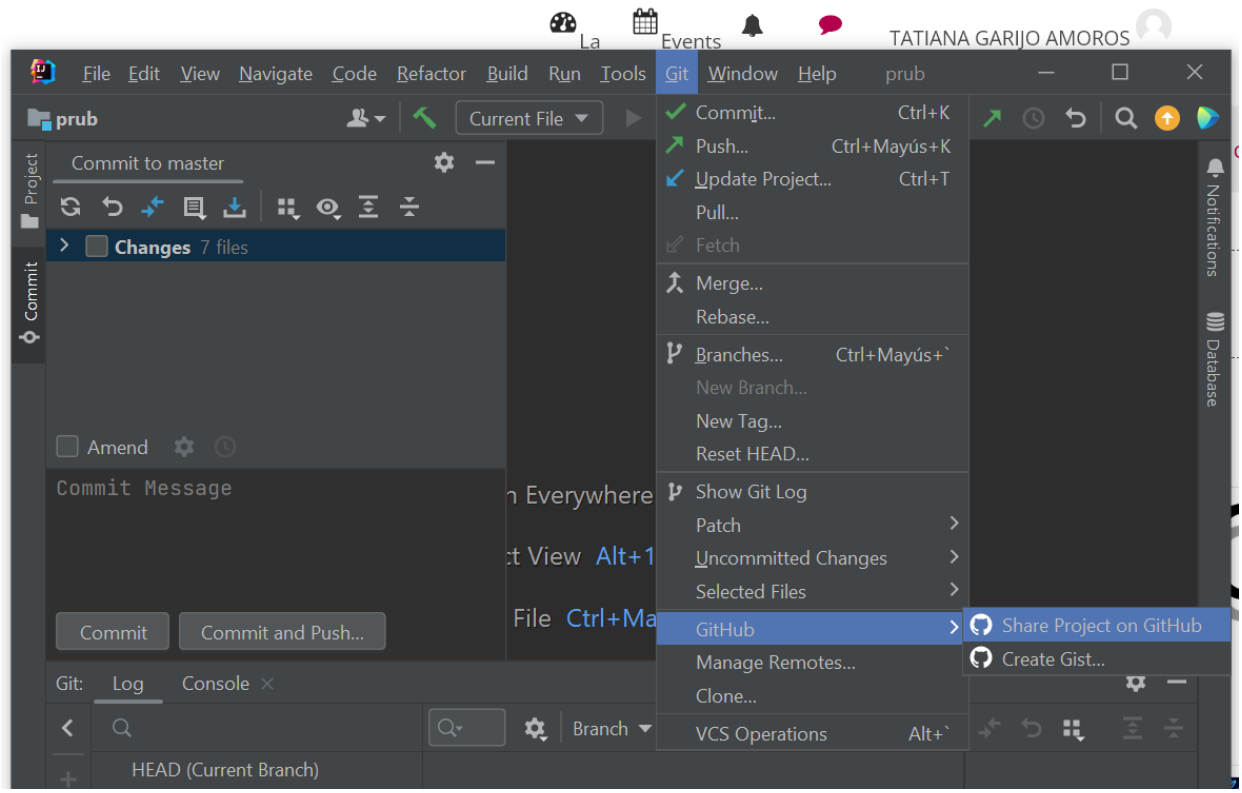
- VSC
- Al crear el proyecto he seleccionado la opción de crear un repositorio Git
- Durante el proceso realizar commit, haciendo uso del panel lateral izquierdo 'commit'
  - Seleccionar los archivos modificados y los que se vayan a incluir en el commit
  - Añadir mensaje del commit
  - Hacer clic sobre el botón commit
-

GENERALITAT VALENCIANA

Aules
ENTORN VIRTUAL D'APRENENTATGE



- Subir repositorio a Github



- Hacer clic sobre el menú superior 'Git' → GitHub → Share Project on GitHub

- Dar nombre al repositorio y hacer clic sobre el botón 'Share'