

# ENTORNOS DE DESARROLLO

UD 11: Prácticas Diagramas UML clases

Francisco Salar Pérez 1º S

**Práctica Diagramas UML Clases.**

Lee atentamente el pdf adjunto. Es un ejemplo resuelto de un diagrama UML.

Está formado por el enunciado y diferentes fases de la creación del modelo UML que da solución al problema ejemplo.

En esta práctica debes realizar los siguientes trabajos:

1. Instala el plugin PlantUml en tu entorno de desarrollo IntelliJ IDEA. En el siguiente enlace tienes un videotutorial de como hacerlo paso a paso. Esto te debe costar menos de dos minutos.

Entrega de la práctica.

Copia un enlace al repositorio gitHub donde has subido la práctica.

Asegúrate de que cada rama contiene lo que se pide.

Crea una rama que se llame "Memoria" y sube a esta rama la memoria realizada. \*\*\*  
FORMATO PDF O WORD \*\*\*

Tiempo estimado para realizar la práctica: 1h:45 minutos.

2. Replica cada una de las fases que se desarrollan en la solución del ejemplo (archivo "Ejemplo resuelto para la práctica. pdf" cuyo enlace de descarga se encuentra al final de esta explicación.

Para crear el diagrama UML con la solución final debes utilizar la información contenida en el siguiente enlace, que es el manual de uso de PlantUml en su sección de Diagramas de Clases.

Como verás es muy muy sencillo.

<https://plantuml.com/es/class-diagram>

Crea un repositorio GitHub que se llame "UMLClases+TuNombre" donde debes ir subiendo el proyecto y las diferentes fases de la creación de la memoria, en este caso solo usaremos la rama Master y la memoria debe estar en esta rama para su corrección.

Realiza una memoria del paso a paso, con pantallazos y comentarios de cada uno de los movimientos que vas ejecutando para realizar la práctica.

Es importante que esta memoria contenga una breve explicación de lo que vas a hacer y como lo consigues, junto con el pantallazo acreditativos del trabajo realizado.

En el repositorio debe estar el proyecto de IntelliJ que contiene el modelo UML del ejercicio, el documento de la memoria en el formato que hayas elegido para realizar la memoria y el mismo documento (el de la memoria) en formato PDF.

Te recomiendo realizar commits de forma rutinaria después de realizar cada paso en la ejecución de la práctica y la realización de la memoria.

Piensa que si algo falla lo pierdes todo y tienes que volver a empezar. Si utilizas el VCS tendrás puntos de recuperación y el trabajo perdido será mínimo.

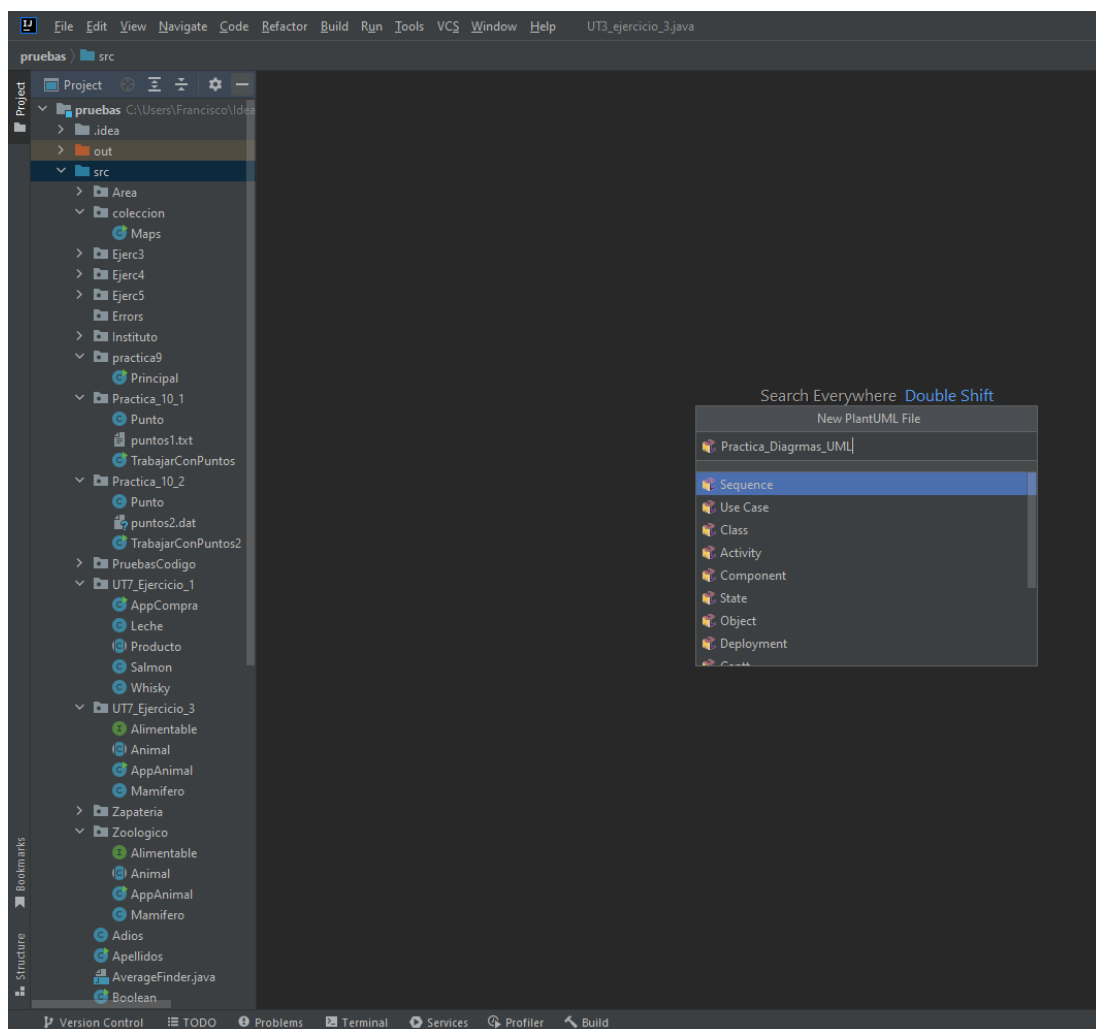
Sube aquí el archivo de la memoria con extensión .pdf

Dentro del archivo, en la primera página, se debe encontrar el enlace al repositorio GitHub donde está el proyecto completo para su corrección.

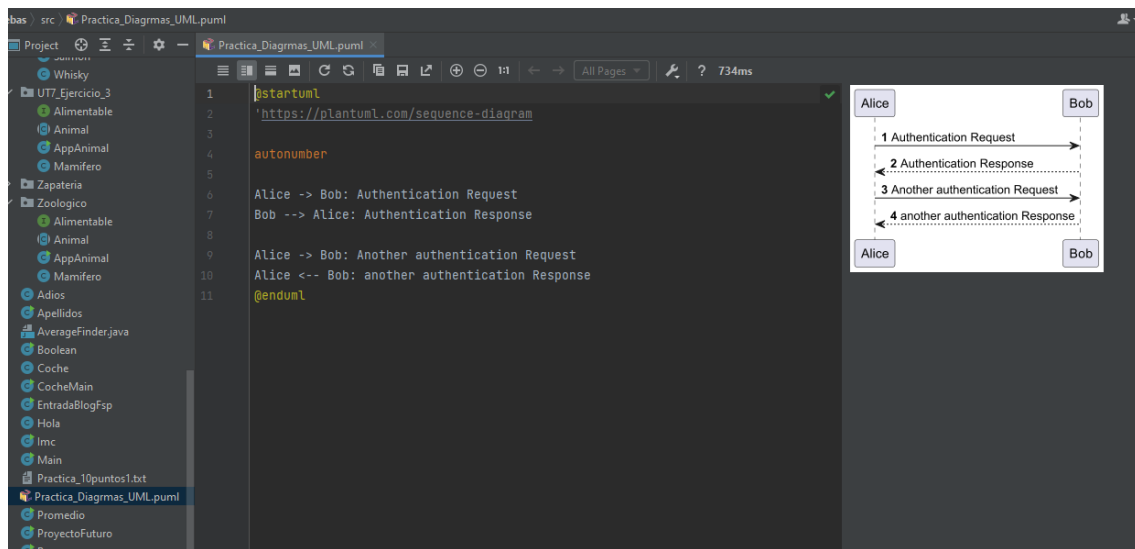
### **Solución:**

**Enlace al repositorio en GitHub:** <https://github.com/CorazaNegra/UMLClasesFSP.git>

Primero creamos el package “Ede\_11” donde albergaremos el archivo “Practica\_Diagramas\_UML.puml”, hacemos click secundario en nuestro package y seleccionamos en el nuevo dialogo la opción New PlantUML File:



Al crear el archivo obtendremos el siguiente resultado:

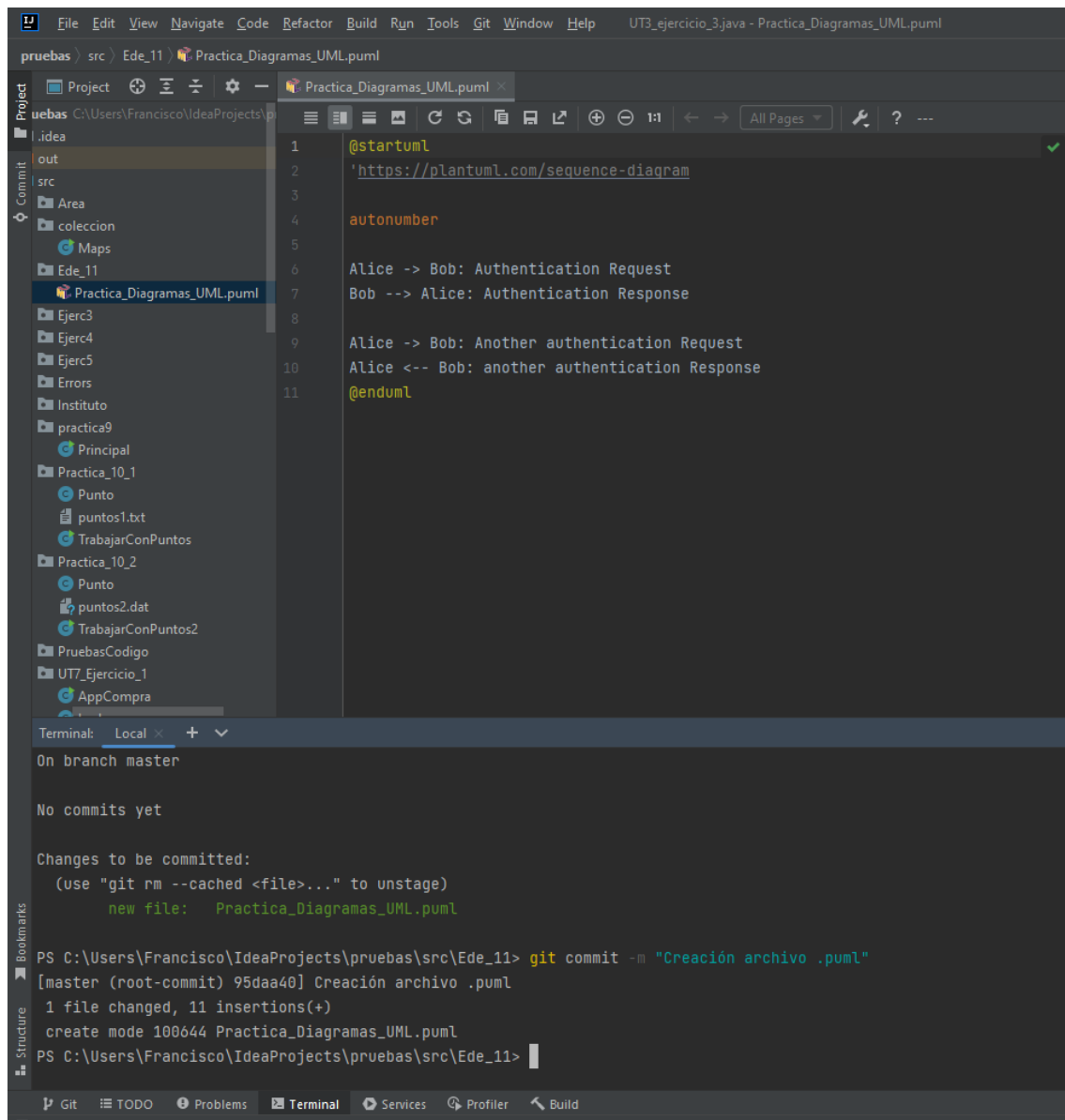


Seguidamente procedemos a crear un repositorio nuevo en GitHub con el nombre “UMLClasesFSP” tal y como se ve en la siguiente imagen:

The screenshot shows the GitHub 'Create a new repository' page. The page includes the following fields and options:

- Owner \***: CorazaNegra
- Repository name \***: UMLClasesFSP (with a green checkmark indicating it is available)
- Description (optional)**: A text input field.
- Visibility**: ☒ Public (Anyone on the internet can see this repository. You choose who can commit.) and ☐ Private (You choose who can see and commit to this repository.)
- Initialize this repository with:** ☐ Add a README file (This is where you can write a long description for your project. [Learn more about READMEs.](#))
- Add .gitignore**: .gitignore template: None (Choose which files not to track from a list of templates. [Learn more about ignoring files.](#))
- Choose a license**: License: None (A license tells others what they can and can't do with your code. [Learn more about licenses.](#))
- Footer**: You are creating a public repository in your personal account.
- Create repository**: A green button at the bottom right.

En este punto hemos realizado nuestro primer “commit”, justo después de crear el archivo:



The screenshot shows an IDE with a project named 'pruebas' and a file named 'Practica\_Diagramas\_UML.puml'. The file contains a UML sequence diagram with the following content:

```
1 @startuml
2 'https://plantuml.com/sequence-diagram
3
4 autonumber
5
6 Alice -> Bob: Authentication Request
7 Bob --> Alice: Authentication Response
8
9 Alice -> Bob: Another authentication Request
10 Alice <-- Bob: another authentication Response
11 @enduml
```

The terminal window shows the output of a git commit command:

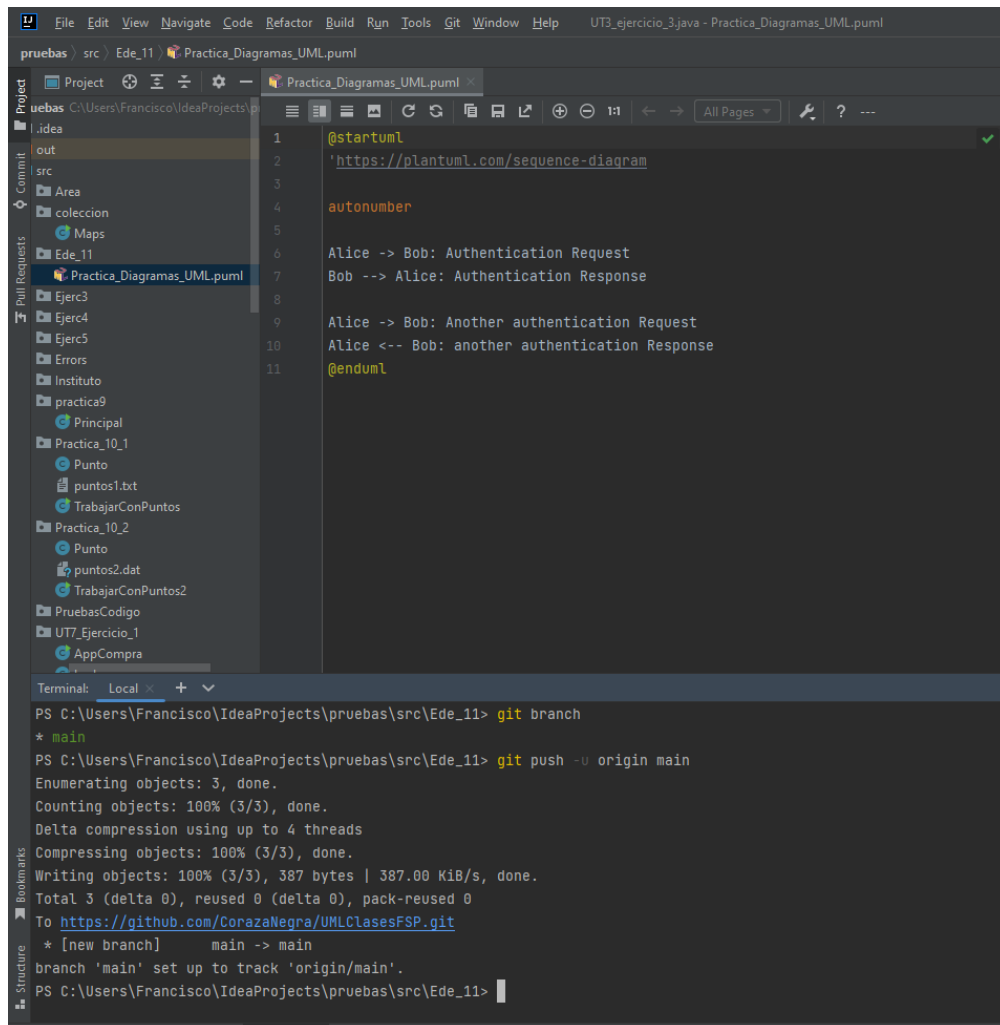
```
Terminal: Local x + v
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   Practica_Diagramas_UML.puml

PS C:\Users\Francisco\IdeaProjects\pruebas\src\Ede_11> git commit -m "Creación archivo .puml"
[master (root-commit) 95daa40] Creación archivo .puml
 1 file changed, 11 insertions(+)
 create mode 100644 Practica_Diagramas_UML.puml
PS C:\Users\Francisco\IdeaProjects\pruebas\src\Ede_11>
```

Posteriormente enlazamos nuestro repositorio local al remoto, con la instrucción correspondiente y enviamos nuestro proyecto al repositorio creado en GitHub:

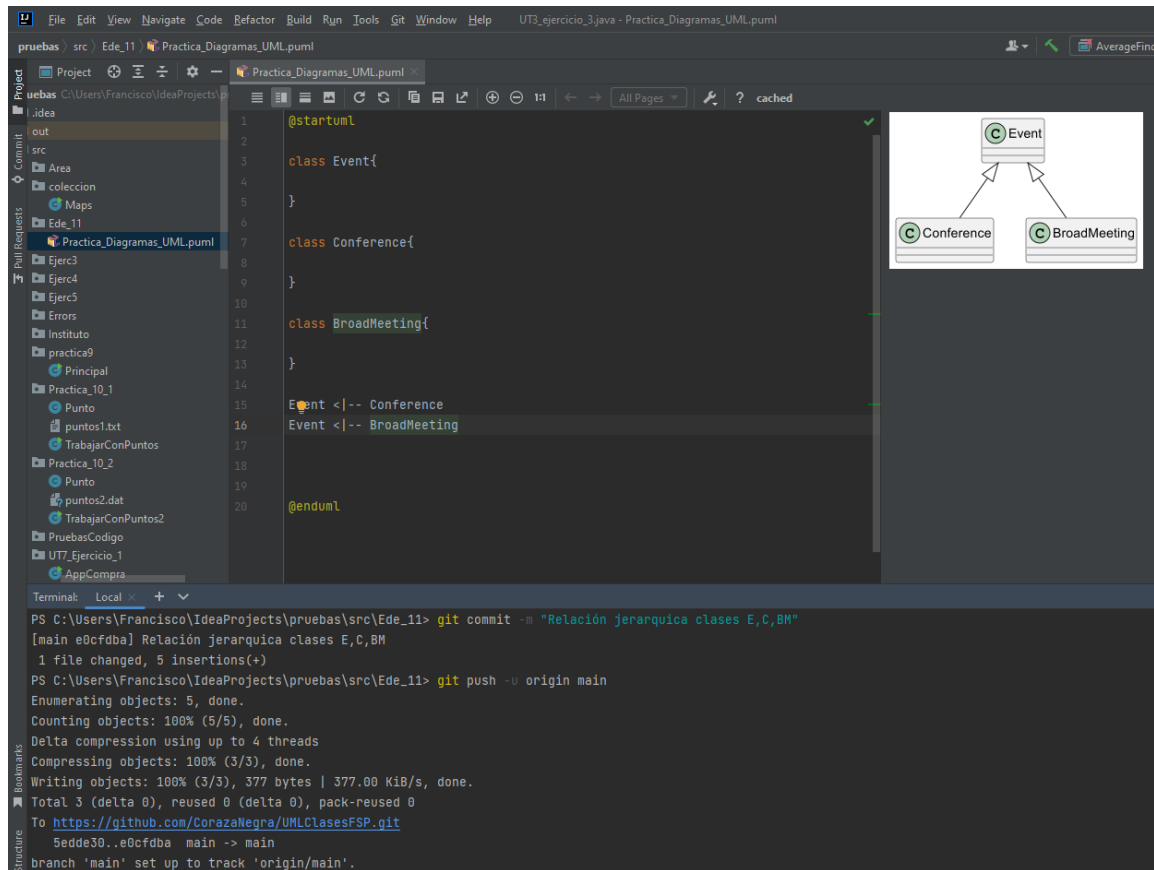


The screenshot shows an IDE with a project named 'pruebas' and a file 'Practica\_Diagramas\_UML.puml'. The diagram is a sequence diagram with two participants, Alice and Bob. It starts with a message from Alice to Bob: 'Authentication Request', followed by a return message from Bob to Alice: 'Authentication Response'. Then, Alice sends another message to Bob: 'Another authentication Request', and Bob returns 'another authentication Response'. The diagram is enclosed in @startuml and @enduml tags. The terminal window at the bottom shows the following commands and output:

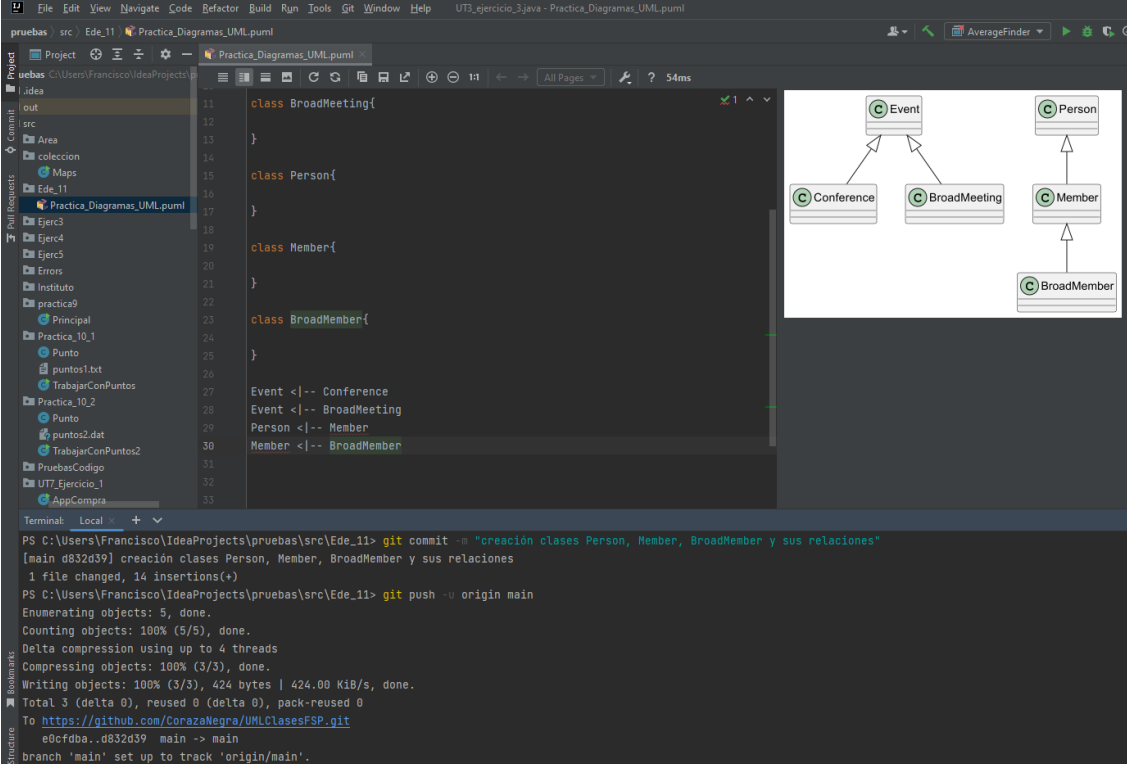
```
PS C:\Users\Francisco\IdeaProjects\pruebas\src\Ede_11> git branch
* main

PS C:\Users\Francisco\IdeaProjects\pruebas\src\Ede_11> git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 387 bytes | 387.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/CorazaNegra/UMLClasesFSP.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
PS C:\Users\Francisco\IdeaProjects\pruebas\src\Ede_11>
```

En nuestro proyecto creamos las tres primeras clases: Event, Conference y BroadMeeting y la relación jerárquica entre ellos, realizando un commit con los cambios realizados y su correspondiente push:



Ahora creamos las clases: Person, Member y BroadMember y sus relaciones jerárquicas, y procedemos de igual forma que antes, realizando un commit con los cambios y un push para actualizar el repositorio remoto:



The screenshot shows an IDE with the following components:

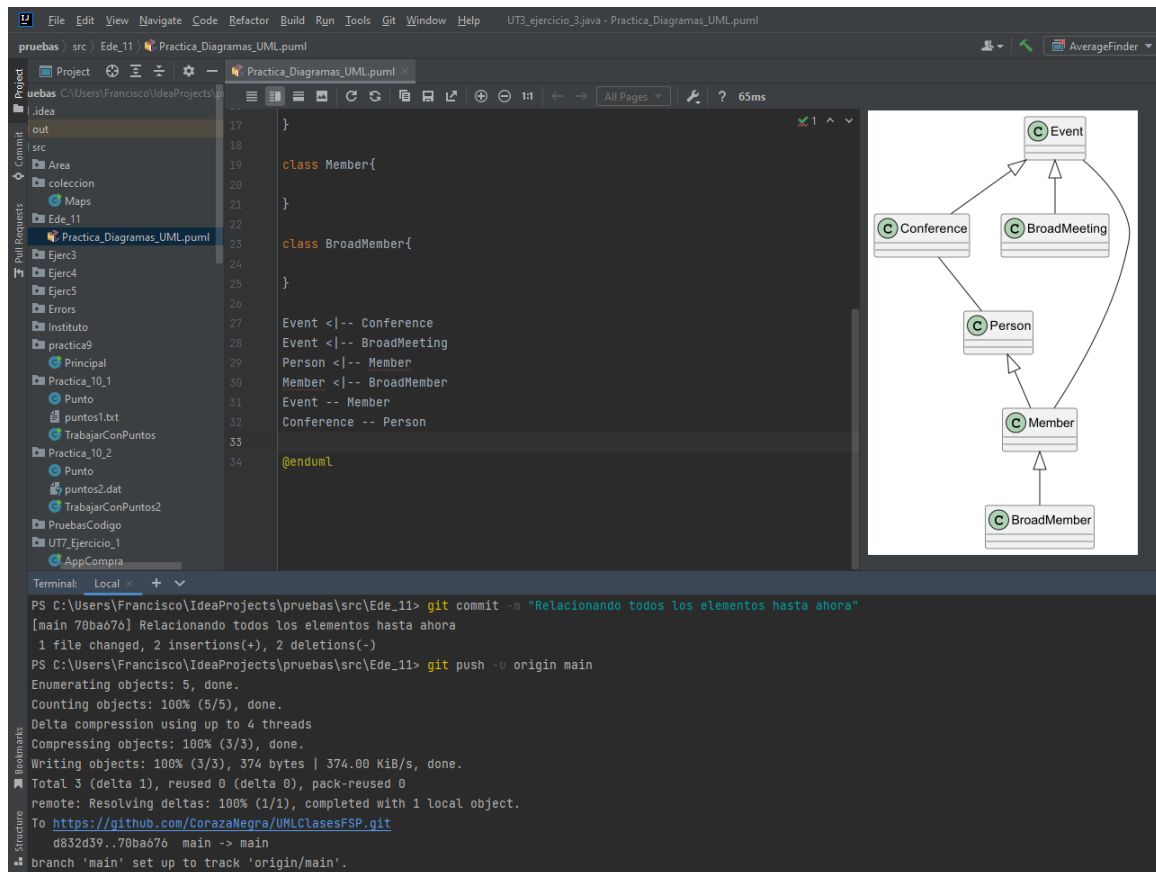
- Left Panel (Project Explorer):** Displays the project structure with folders like 'src', 'Area', 'coleccion', 'Maps', and 'Ede\_11'. The file 'Practica\_Diagramas\_UML.puml' is selected.
- Center Panel (Code Editor):** Contains the Java code for the classes:

```
class BroadMeeting{  
      
}  
  
class Person{  
      
}  
  
class Member{  
      
}  
  
class BroadMember{  
      
}  
  
Event <|-- Conference  
Event <|-- BroadMeeting  
Person <|-- Member  
Member <|-- BroadMember
```
- Right Panel (UML Diagram):** Displays the UML class diagram corresponding to the code. It shows a hierarchy where 'Event' is the superclass for 'Conference' and 'BroadMeeting'. 'Person' is the superclass for 'Member', which is in turn the superclass for 'BroadMember'.
- Bottom Panel (Terminal):** Shows the execution of git commands:

```
PS C:\Users\Francisco\IdeaProjects\pruebas\src\Ede_11> git commit -m "creación clases Person, Member, BroadMember y sus relaciones"  
[main d832d39] creación clases Person, Member, BroadMember y sus relaciones  
1 file changed, 14 insertions(+)  
PS C:\Users\Francisco\IdeaProjects\pruebas\src\Ede_11> git push -u origin main  
Enumerating objects: 5, done.  
Counting objects: 100% (5/5), done.  
Delta compression using up to 4 threads  
Compressing objects: 100% (3/3), done.  
Writing objects: 100% (3/3), 424 bytes | 424.00 KiB/s, done.  
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0  
To https://github.com/corazaNegra/UMLClasesFSP.git  
e0cfdba..d832d39 main -> main  
branch 'main' set up to track 'origin/main'.
```



Comenzamos entonces a definir las relaciones y jerarquías de las clases anteriormente creadas, obteniendo lo que se aprecia en la siguiente imagen:



Podemos observar también que hemos realizado el commit y push correspondiente.

Ahora creamos las clases AAUOC y Location y, hacemos un commit con las modificaciones y su respectivo push:

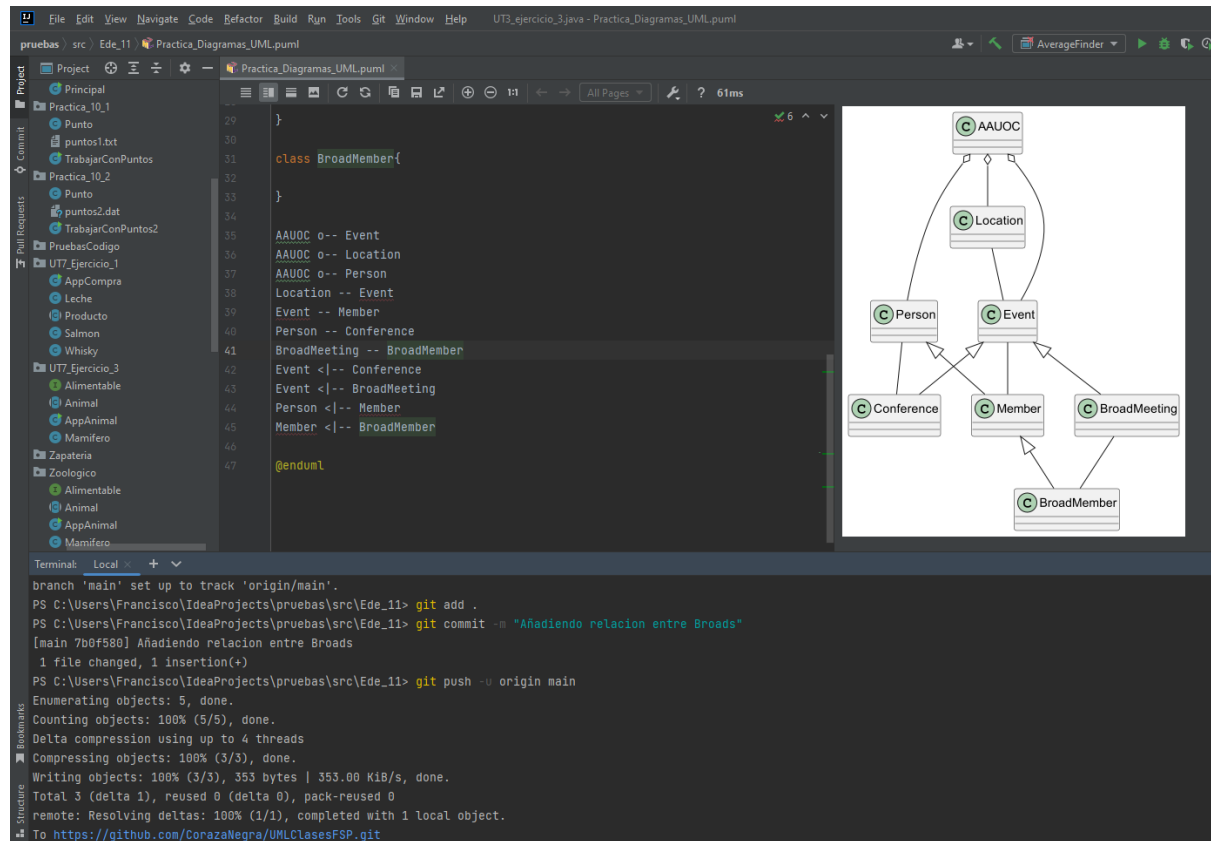
The screenshot displays an IDE window with the following components:

- Project Explorer:** Shows a project structure with folders like 'src', 'out', 'Area', 'coleccion', 'Maps', 'Ede\_11', and 'Practica\_Diagramas\_UML.puml'.
- Code Editor:** Contains the following Java code:

```
@startuml
class AAUOC{
}
class Location{
}
class Event{
}
class Conference{
}
class BroadMeeting{
}
class Person{
}
class Member{
}
class BroadMember{
}
```
- UML Diagram:** A class diagram showing the relationships between the classes. It includes classes AAUOC, Location, Event, Conference, BroadMeeting, Person, Member, and BroadMember. Arrows indicate inheritance or association relationships.
- Terminal:** Shows the execution of git commands:

```
branch 'main' set up to track 'origin/main'.
PS C:\Users\Francisco\IdeaProjects\pruebas\src\Ede_11> git add .
PS C:\Users\Francisco\IdeaProjects\pruebas\src\Ede_11> git commit -m "creación clases AAUOC y Location"
[main 60ed30a] creación clases AAUOC y Location
1 file changed, 8 insertions(+)
PS C:\Users\Francisco\IdeaProjects\pruebas\src\Ede_11> git push -u origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 369 bytes | 369.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/CorazaNegra/UMLClasesFSP.git
```

En este punto relacionamos cada clase y jerarquizamos sus relaciones si así procede:



Observamos que, como hemos hecho anteriormente, realizamos un commit y un push a remoto para tener actualizado el proyecto con sus cambios.

Una vez hecho lo anterior, podemos empezar con la adición de los atributos y métodos que cada clase tiene, empezando por la clase AAUOC:

The screenshot shows an IDE with the following components:

- Project Explorer:** Lists various project files and folders, including 'Practica\_10\_1', 'Practica\_10\_2', 'PruebasCodigo', and 'UT7\_Ejercicio\_1'.
- Code Editor:** Displays the following Java code:

```
@startuml
class AAUOC{
    newLocation(l : Location): void
    newEvent(e : Event): void
    newPerson(p : Person): void
    informEvent(e : Event): void
    register(m : Member, e : Event): void
}
class Location{
}
class Event{
}
class Conference{
}
class BroadMeeting{
}
```
- UML Class Diagram:** Visualizes the classes and their relationships. AAUOC is at the top, connected to Location, Event, and BroadMeeting. Location is connected to Person and Event. Event is connected to Conference, Member, and BroadMeeting. BroadMeeting is connected to Member. All classes have a green circle icon next to their name.
- Terminal:** Shows the execution of git commands to add, commit, and push the changes to the main branch.

```
branch 'main' set up to track 'origin/main'.
PS C:\Users\Francisco\IdeaProjects\pruebas\src\Ede_11> git add .
PS C:\Users\Francisco\IdeaProjects\pruebas\src\Ede_11> git commit -m "Añadiendo métodos a AAUOC"
[main 701ce3c] Añadiendo métodos a AAUOC
1 file changed, 5 insertions(+), 1 deletion(-)
PS C:\Users\Francisco\IdeaProjects\pruebas\src\Ede_11> git push -u origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 418 bytes | 418.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/CorazaNegra/UMLClasesFSP.git
```

En este caso, nuestra clase tan solo cuenta con métodos.

Pasamos ahora a introducir los atributos en la clase Location quedando de la siguiente forma:

The screenshot displays an IDE with a Java code editor on the left and a UML class diagram on the right. The code editor shows the following code:

```
@startuml
class AAUOC{
  newLocation(l : Location): void
  newEvent(e : Event): void
  newPerson(p : Person): void
  informEvent(e : Event): void
  register(m : Member, e : Event): void
}

class Location{
  description : String
  address : String
}

class Event{
}

class Conference{
}

class Person
class Event
class Conference
class Member
class BroadMeeting

AAUOC --> Location
AAUOC --> Event
AAUOC --> Conference
AAUOC --> Member
AAUOC --> BroadMeeting
Person --> Event
Event --> Conference
Event --> Member
Event --> BroadMeeting
```

The UML diagram on the right shows the following classes and relationships:

- AAUOC** (Class): Methods: `newLocation(l : Location): void`, `newEvent(e : Event): void`, `newPerson(p : Person): void`, `informEvent(e : Event): void`, `register(m : Member, e : Event): void`.
- Location** (Class): Attributes: `description : String`, `address : String`.
- Event** (Class):
- Conference** (Class):
- Person** (Class):
- Member** (Class):
- BroadMeeting** (Class):

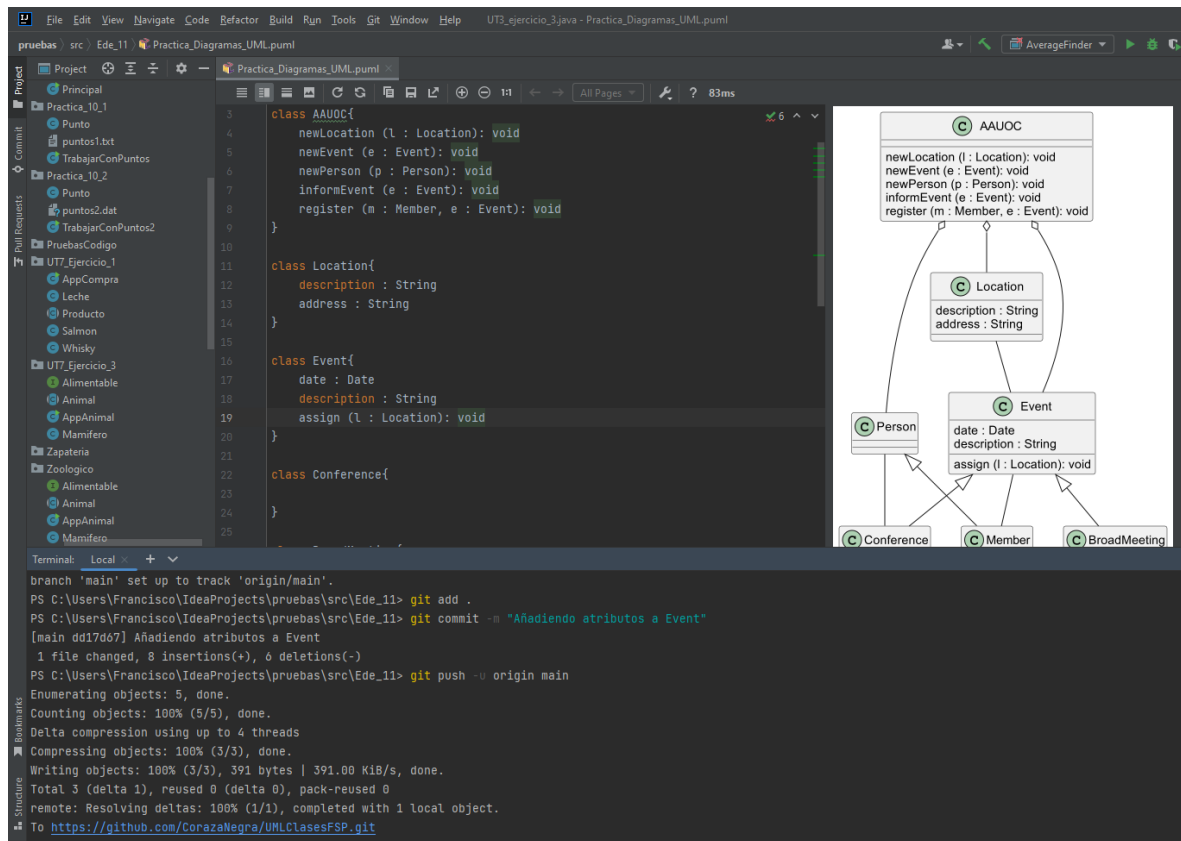
Relationships (Associations):

- AAUOC to Location
- AAUOC to Event
- AAUOC to Conference
- AAUOC to Member
- AAUOC to BroadMeeting
- Person to Event
- Event to Conference
- Event to Member
- Event to BroadMeeting

The terminal at the bottom shows the following commands and output:

```
branch 'main' set up to track 'origin/main'.
PS C:\Users\Francisco\IdeaProjects\pruebas\src\Ede_11> git add .
PS C:\Users\Francisco\IdeaProjects\pruebas\src\Ede_11> git commit -m "Añadiendo atributos a Location"
[main 47ce0d0] Añadiendo atributos a Location
1 file changed, 2 insertions(+), 1 deletion(-)
PS C:\Users\Francisco\IdeaProjects\pruebas\src\Ede_11> git push -u origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 362 bytes | 362.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/CorazaNegra/UMLClasesFSP.git
```

Toca el turno ahora de la clase Event, que en este caso posee tanto atributos como métodos, como se aprecia en la imagen siguiente.



Con cada paso realizamos commits para mantener actualizado nuestro proyecto tanto en nuestro repositorio local como en remoto.

En las siguientes tres imágenes se procede a introducir los atributos en las clases: Conference, Person y Member.

The screenshot shows an IDE with the following components:

- Project Explorer:** Shows a project structure with folders like 'Practica\_10\_1', 'Practica\_10\_2', and 'UT7\_Ejercicio\_3'.
- Code Editor:** Displays the code for the `Conference` class. The attribute `max_attendees : Integer` has been added to the class definition.
 

```

class Event{
    date : Date
    description : String
    assign (l : Location): void
}

class Conference{
    max_attendees : Integer
}

class BroadMeeting{
}

class Person{
}

class Member{
}
```
- UML Diagram:** The diagram on the right shows the updated class structure. The `Conference` class now includes the `max_attendees` attribute. The `Event` class is associated with the `Location` class. The `Person` class is associated with the `Event` class. The `Member` class is associated with the `BroadMeeting` class. The `BroadMember` class is also shown.
- Terminal:** Shows the execution of git commands to add, commit, and push the changes.
 

```

branch 'main' set up to track 'origin/main'.
PS C:\Users\Francisco\IdeaProjects\pruebas\src\Edo_11> git add .
PS C:\Users\Francisco\IdeaProjects\pruebas\src\Edo_11> git commit -m "Añadiendo atributos a Conference"
[main 3c22b43] Añadiendo atributos a Conference
1 file changed, 1 insertion(+), 1 deletion(-)
PS C:\Users\Francisco\IdeaProjects\pruebas\src\Edo_11> git push -u origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 358 bytes | 358.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/corazaNegra/UMLClassesFSP.git
```

The screenshot shows the IDE with the following components:

- Code Editor:** Displays the code for the `Person` class. The attribute `name : String` has been added to the class definition.
 

```

class Event{
    date : Date
    description : String
    assign (l : Location): void
}

class Conference{
    max_attendees : Integer
}

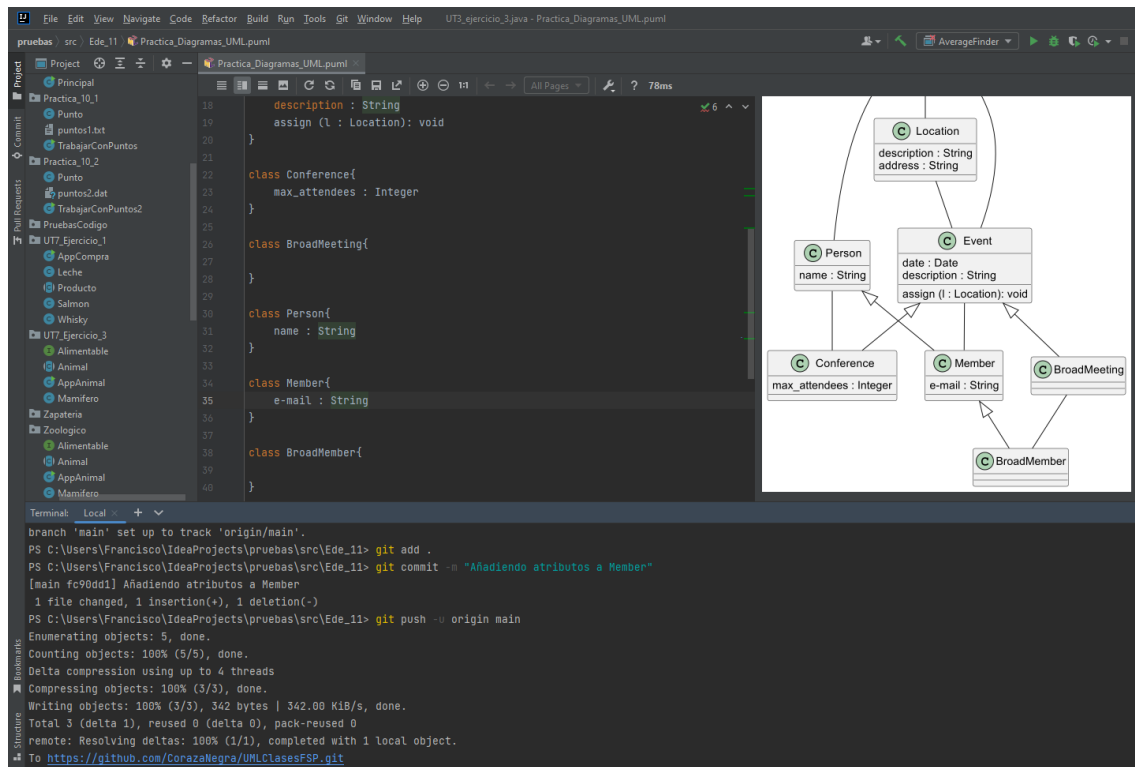
class BroadMeeting{
}

class Person{
    name : String
}

class Member{
}
```
- UML Diagram:** The diagram on the right shows the updated class structure. The `Person` class now includes the `name` attribute. The `Event` class is associated with the `Location` class. The `Person` class is associated with the `Event` class. The `Member` class is associated with the `BroadMeeting` class. The `BroadMember` class is also shown.
- Terminal:** Shows the execution of git commands to add, commit, and push the changes.
 

```

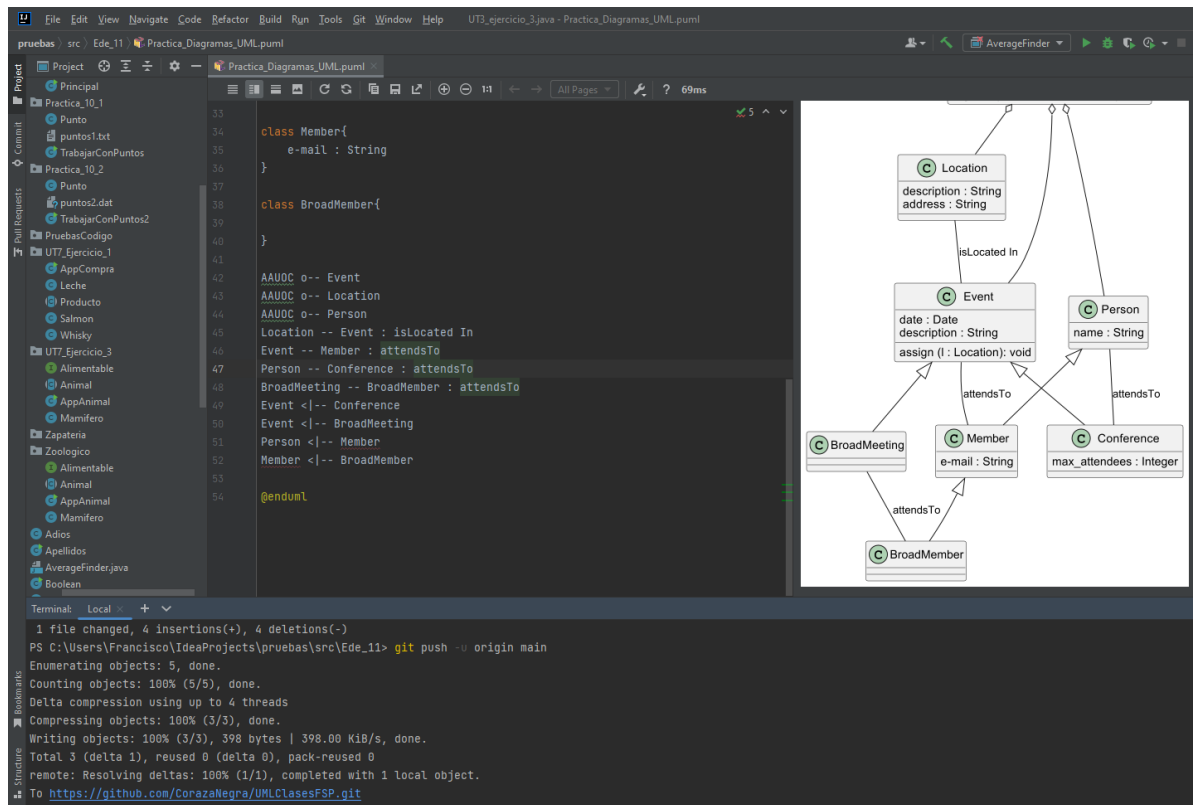
branch 'main' set up to track 'origin/main'.
PS C:\Users\Francisco\IdeaProjects\pruebas\src\Edo_11> git add .
PS C:\Users\Francisco\IdeaProjects\pruebas\src\Edo_11> git commit -m "Añadiendo atributos a Person"
[main 7cd510c] Añadiendo atributos a Person
1 file changed, 1 insertion(+), 1 deletion(-)
PS C:\Users\Francisco\IdeaProjects\pruebas\src\Edo_11> git push -u origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 340 bytes | 340.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/corazaNegra/UMLClassesFSP.git
```



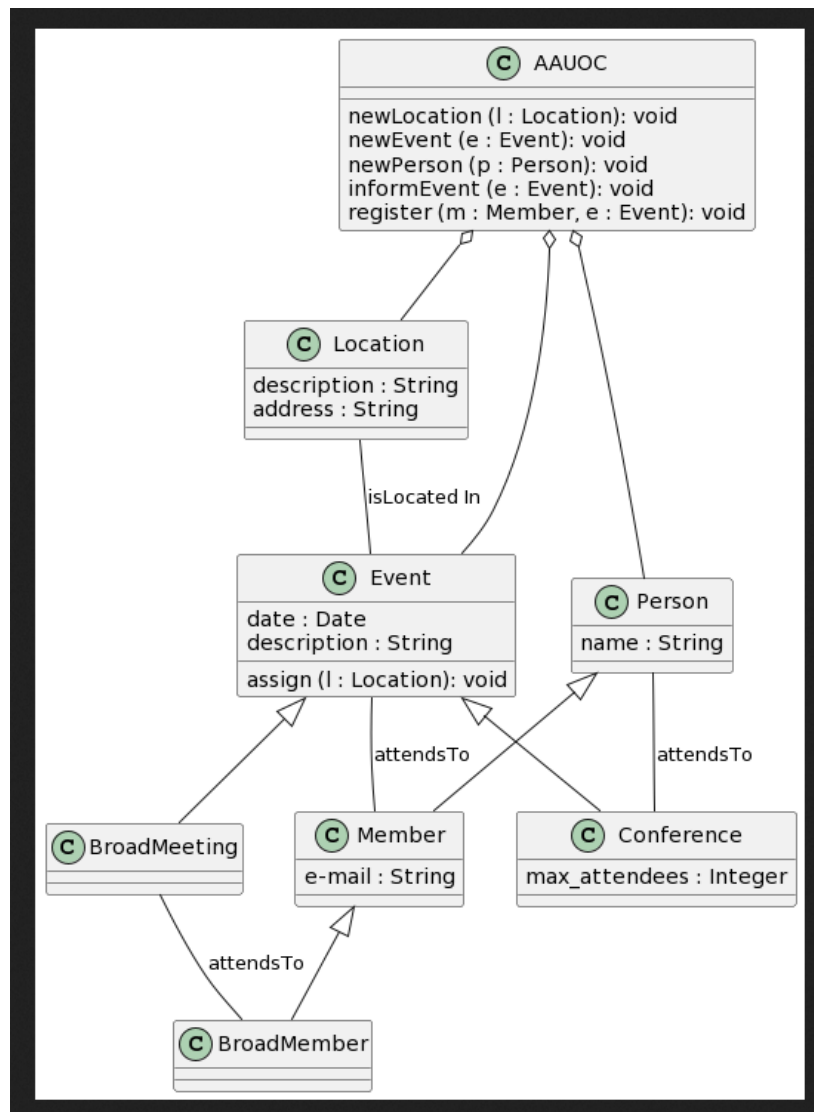
Como se aprecia en la parte baja de cada una de las imágenes, estamos utilizando los comandos Git para actualizar nuestro proyecto.



Una vez realizado lo anterior, toca el turno de definir las relaciones que existen entre clases, como se ve en la imagen que sigue.

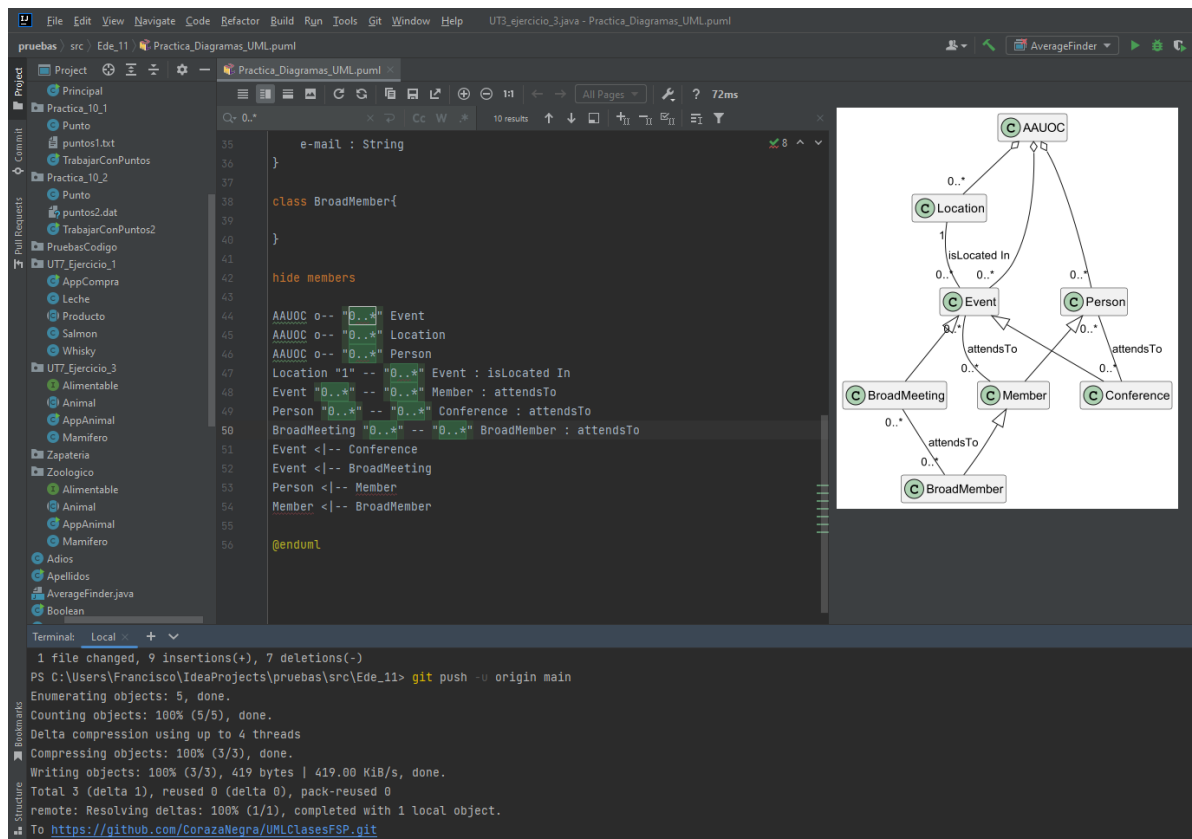


Una vez realizado todos los pasos obtendremos el siguiente resultado:.



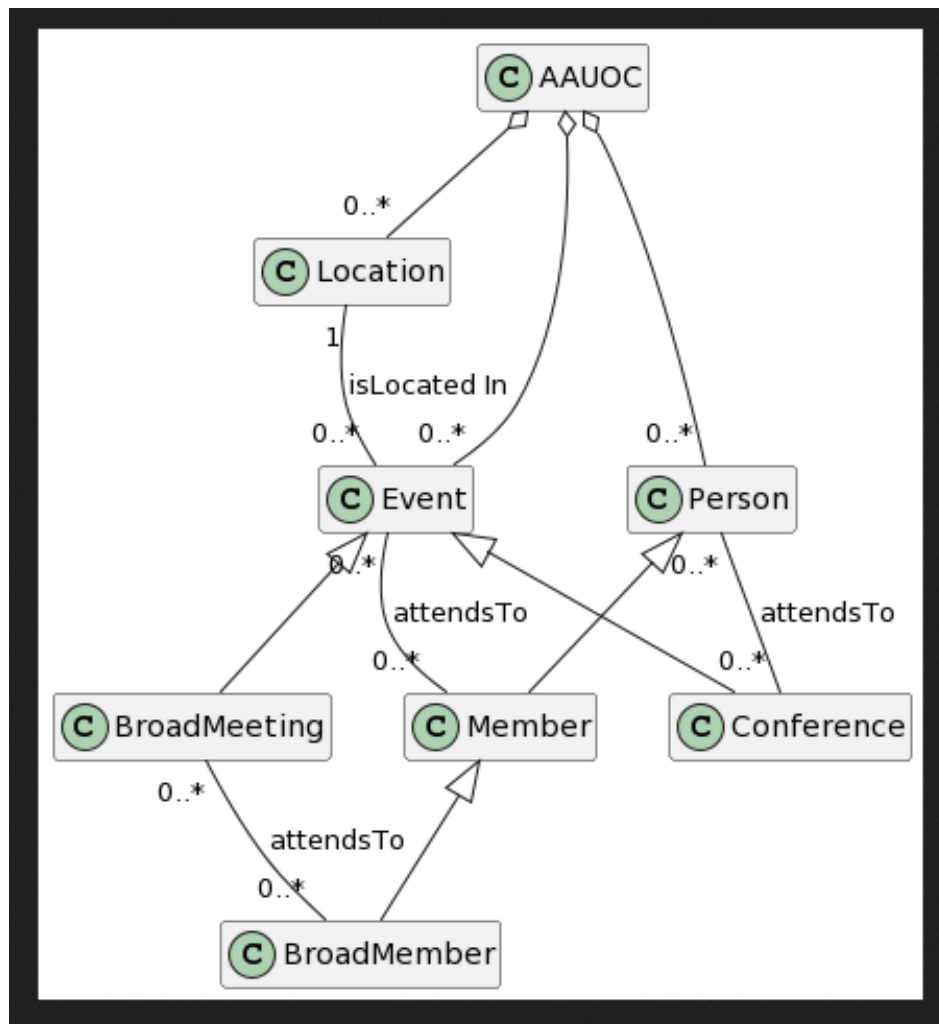
No es un resultado muy estético, pero después de múltiples cambios en el orden de las clases y sus respectivas relaciones, no hemos podido obtener ningún resultado parecido como el que aparece en el pdf de la práctica.

También se nos pide que realicemos un diagrama reducido, que nos es más que el mismo diagrama con la ocultación de los atributos y métodos de cada una de las clases y añadiendo las correspondientes cardinalidades entre las relaciones.



Me imagino que también se podrá poner las cardinalidades en diagramas no reducidos, como el primero que hemos realizado, pero en el pdf las cardinalidades sólo aparecen en el momento en que obtenemos el diagrama reducido.

Vemos, en la siguiente imagen como quedaría el diagrama reducido:



Por último, y a sabiendas de ser pesado, mencionar que con cada cambio o adicción en el proyecto, se han hecho commit y sus correspondientes push.