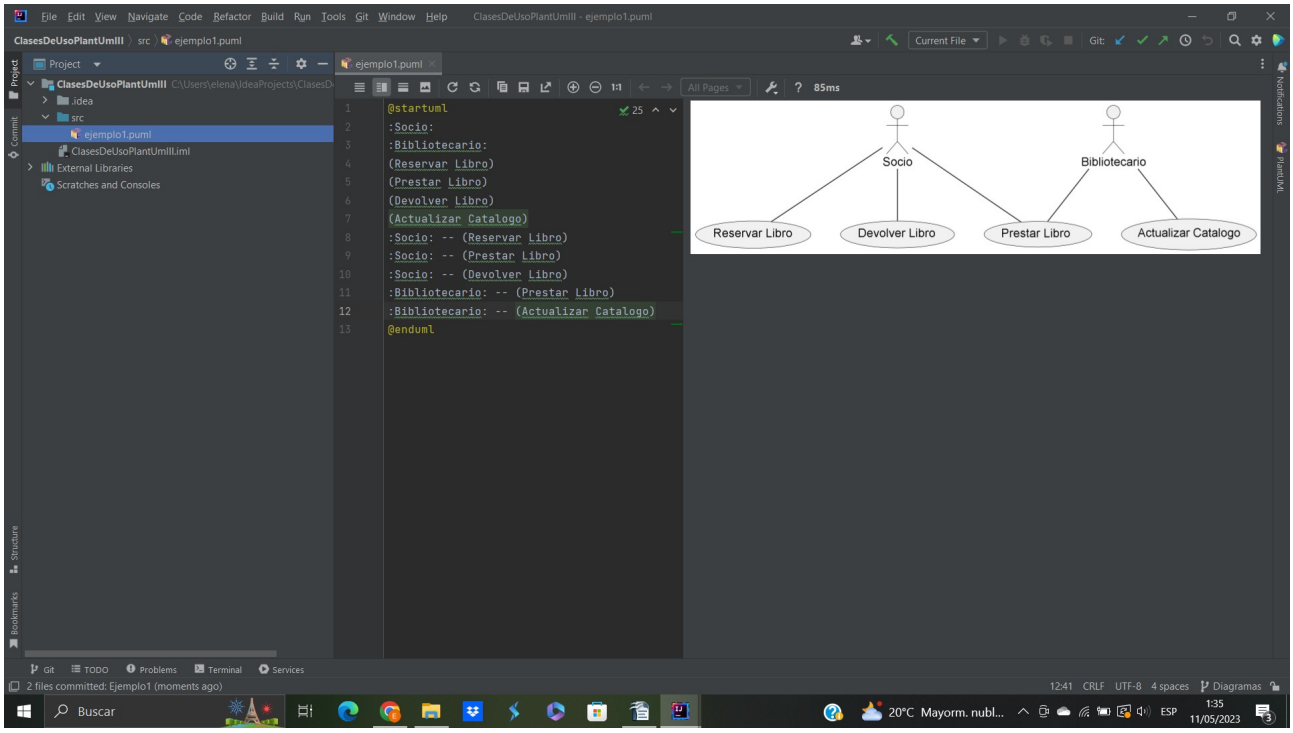
Enlace al repositorio de GitHub: https://github.com/MissTurtle/PracticaClasesDeUsoPlantUmlIIElenaGarciaDiaz.git

Práctica Clases De Uso PlantUml II Memoria

Primero se crea el proyecto, para ello hacemos click en File - New - Project, y le damos un nombre, en este caso ClasesDeUsoPlantUmlII. Entonces, sobre la carpeta src, click derecho, New - PlantUML File y creamos cada una de las clases, que en este caso serán 5:

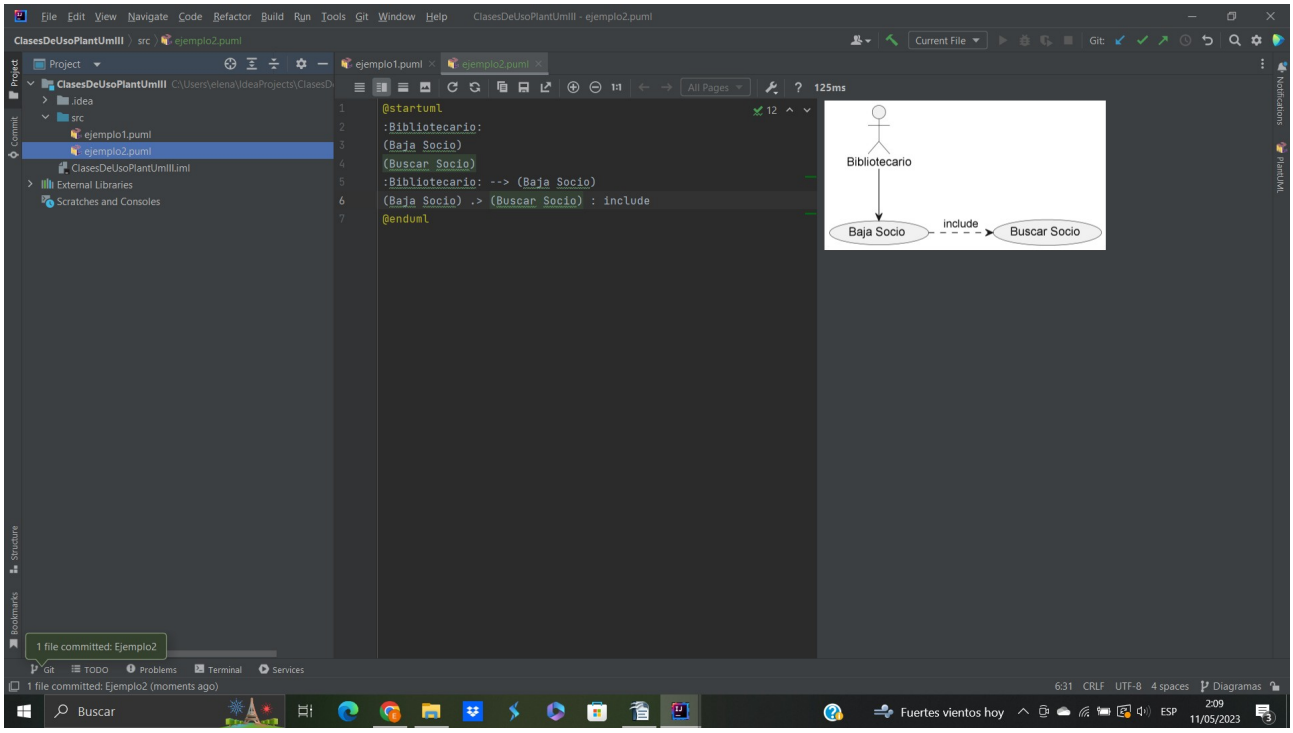
ejemplo1:

La primera clase se llamará ejemplo1.
En el código, primero introducimos los actores poniendo su nombre entre ":", en este caso :Socio: y :Bibliotecario:.
Luego introducimos las clases de uso poniéndolas entre "()", por ejemplo (Reservar Libro).
Finalmente establecemos las relaciones para determinar como funcionaría el programa, en este caso, el Socio debe poder Reservar Libro, Devolver Libro y Prestar Libro, y Bibliotecario de be poder Prestar Libro y Actualizar Catalogo, mediante la sintaxis "- -".



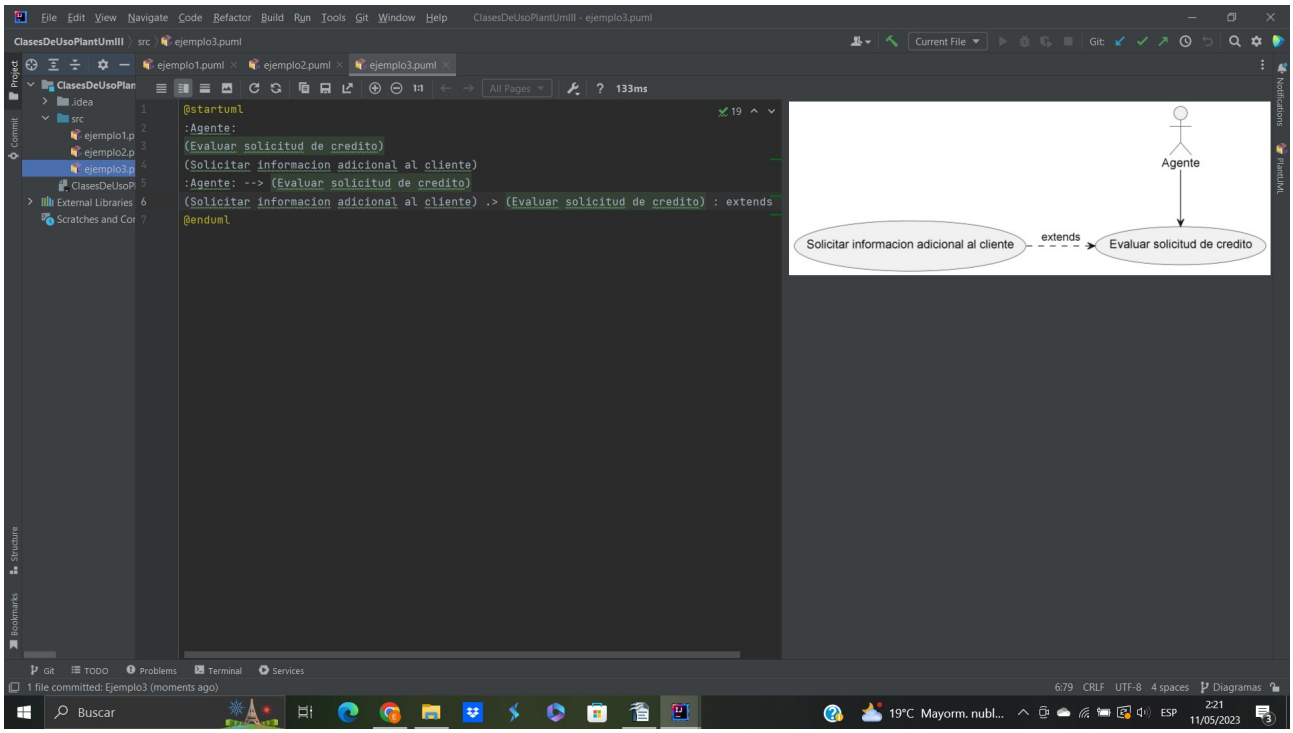
ejemplo2:

La segunda clase se llamará ejemplo2.
En el código, primero introducimos el actor poniendo su nombre entre “:”, en este caso :Bibliotecario:.
Luego introducimos las clases de uso poniéndolas entre “()”, por ejemplo (Baja Socio).
Finalmente establecemos las relaciones para determinar como funcionaría el programa, en este caso, el Bibliotecario debe poder realizar Baja Socio, mediante la sintaxis “- ->”.
En este caso hay una relación de inclusión que nos indica que Buscar Socio se ejecutará en caso de que ocurra Baja Socio, mediante la sintaxis “... .> … : include”.



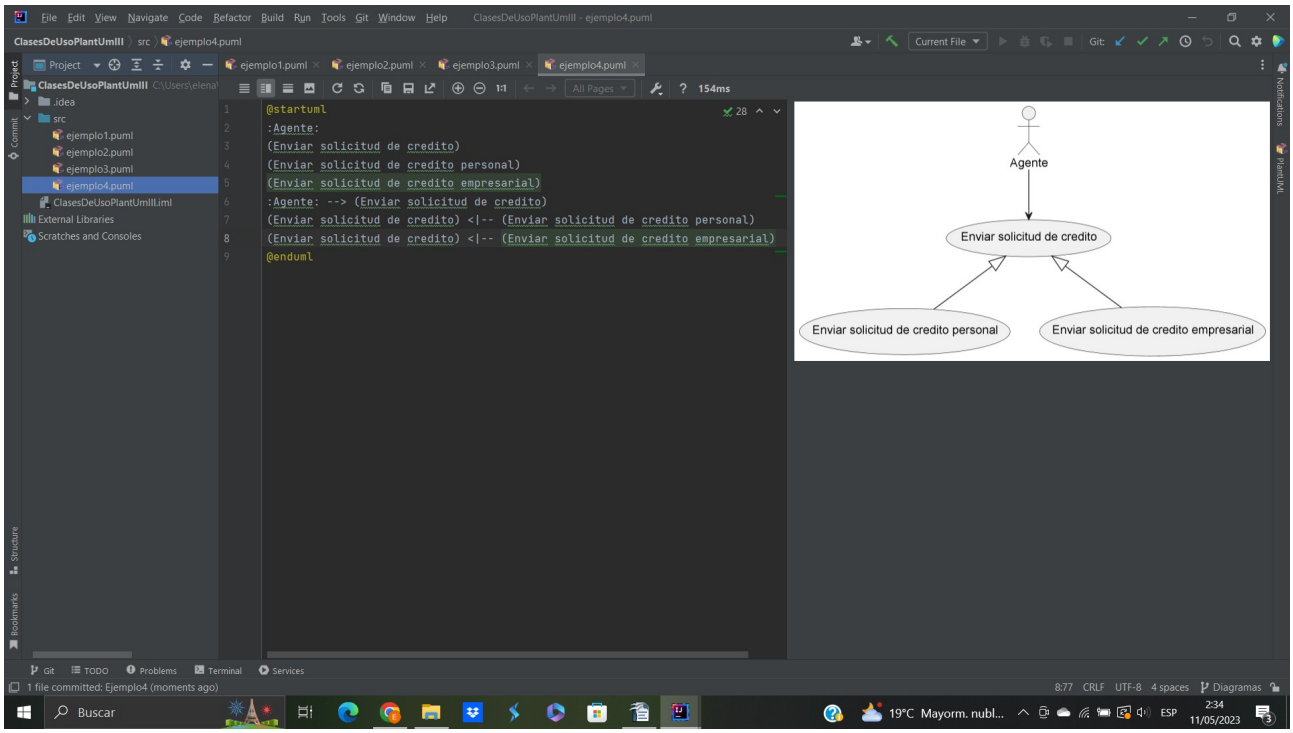
ejemplo3:

La tercera clase se llamará ejemplo3.
En el código, primero introducimos el actor poniendo su nombre entre “:”, en este caso :Agente:.
Luego introducimos las clases de uso poniéndolas entre “()”, por ejemplo (Evaluar solicitud de credito).
Finalmente establecemos las relaciones para determinar como funcionaría el programa, en este caso, el Agente debe poder realizar Evaluar solicitud de credito, mediante la sintaxis “- ->”. En este caso hay una relación de extensión que nos indica que Solicitar informacion adicional al cliente se ejecutará en caso de que se cumpla la condición de que se proceda a Evaluar solicitud de credito, mediante la sintaxis “... .> … : extends”.



ejemplo4:

La cuarta clase se llamará ejemplo4.
En el código, primero introducimos el actor poniendo su nombre entre “:”, en este caso :Agente:.
Luego introducimos las clases de uso poniéndolas entre “()”, por ejemplo (Enviar solicitud de credito).
Finalmente establecemos las relaciones para determinar como funcionaría el programa, en este caso, el Agente debe poder Enviar solicitud de credito, mediante la sintaxis “- ->”. Tambien hay una relación de herencia según la cual Enviar solicitud de credito personal y Enviar solicitud de credito empresarial heredan el comportamiento de Enviar solicitud de credito, mediante la sintaxis “<|- -”.



ejemplo5:

La quinta clase se llamará ejemplo5.
En el código, primero introducimos los actores poniendo su nombre entre “:”, en este caso :Cliente Bancario:, :Cliente Corporativo: y :Cliente Normal:.
En este caso solo intervienen actores.
Finalmente establecemos las relaciones para determinar como funcionaría el programa, en este caso, hay una relación de herencia según la cual Cliente Corporativo y Cliente Normal heredan el comportamiento de Cliente Bancario, mediante la sintaxis “<|- -”.