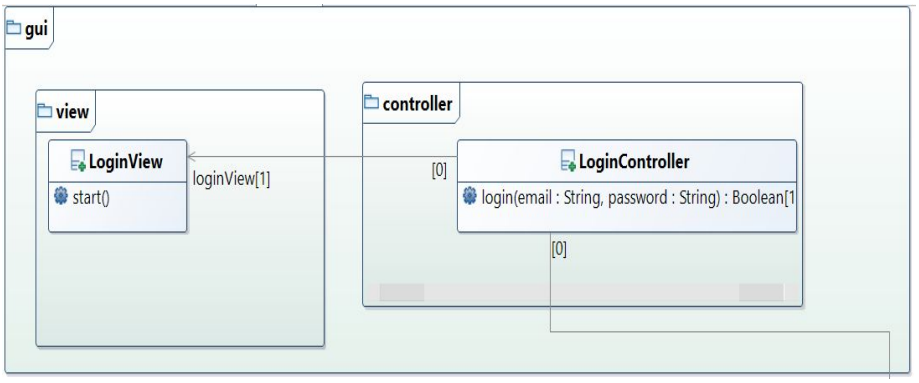
Prototype and Conception

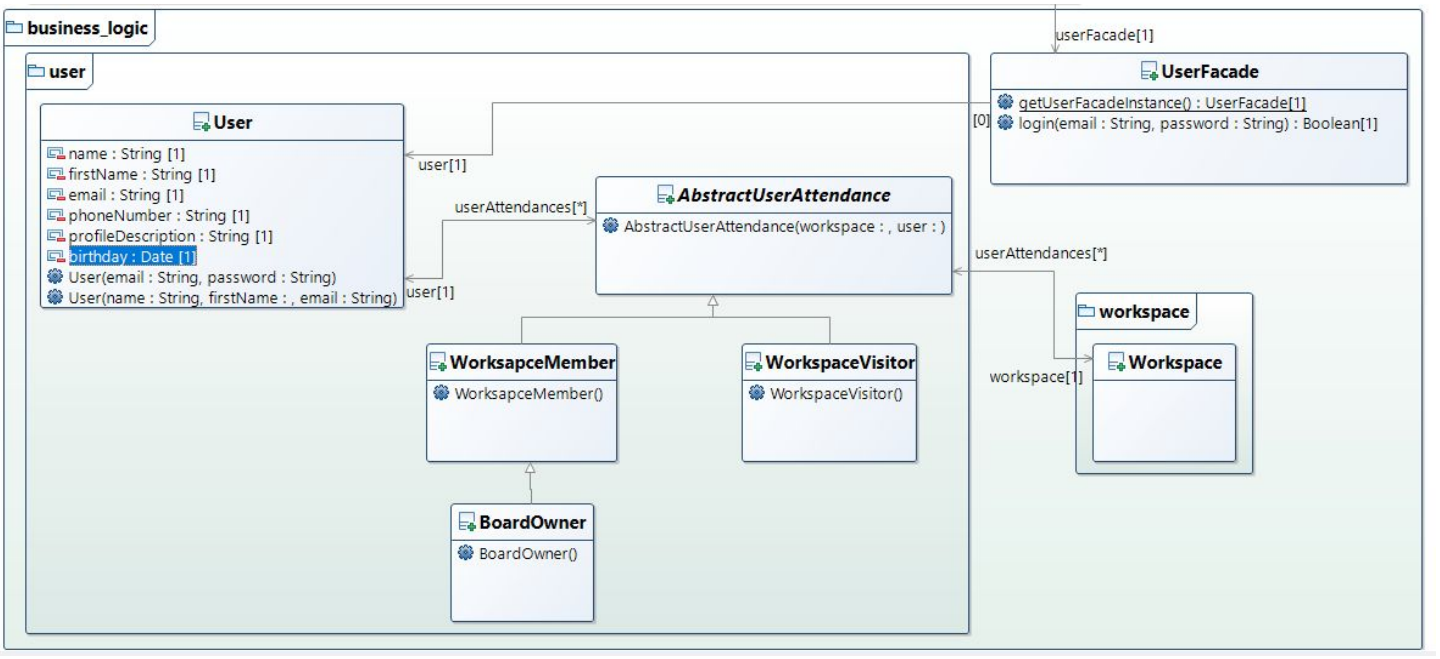
Login

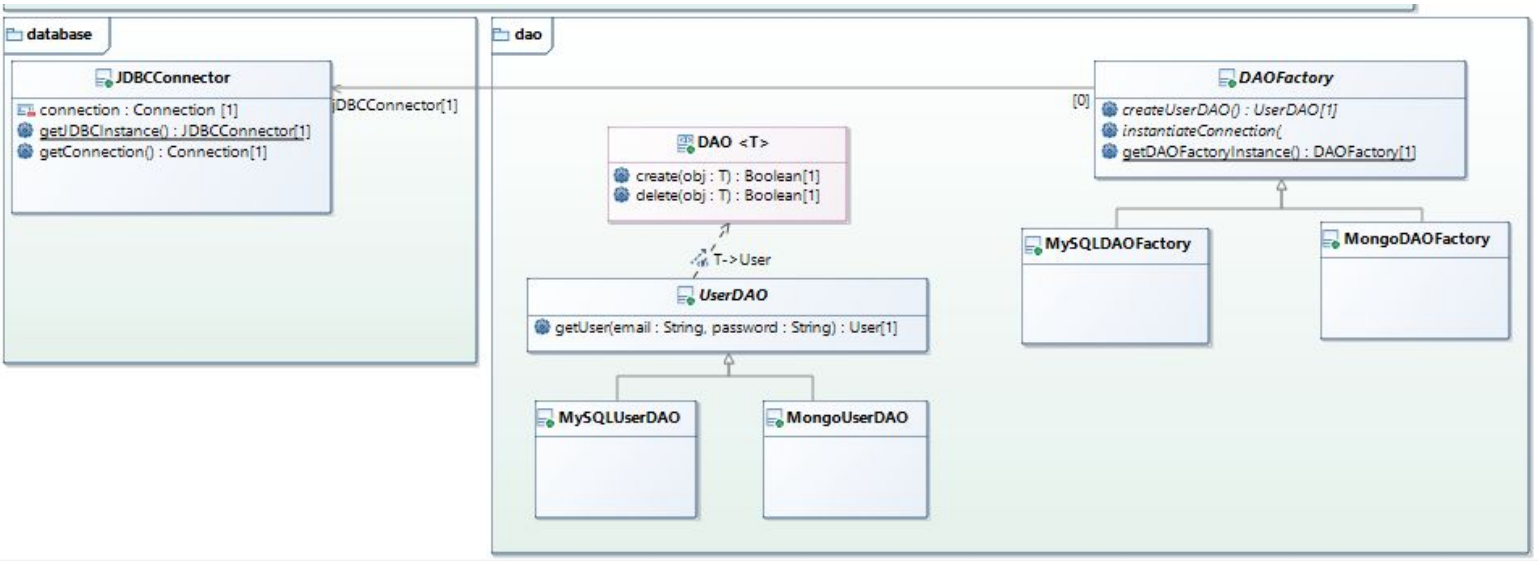
AZHARHOUSSEN Salim
BA Birane
BOURRET Raphaël
GALOIS Nicolas

14/12/2020

Class diagram







Singleton

We use a singleton pattern to make sure that only one class is instantiated, and so that we can access an attribute which will be instantiated only once. In our case we want the connection to the database to be used by many functions in the DAO. We may implement later a close function and an initialize function to handle the connection to the database.

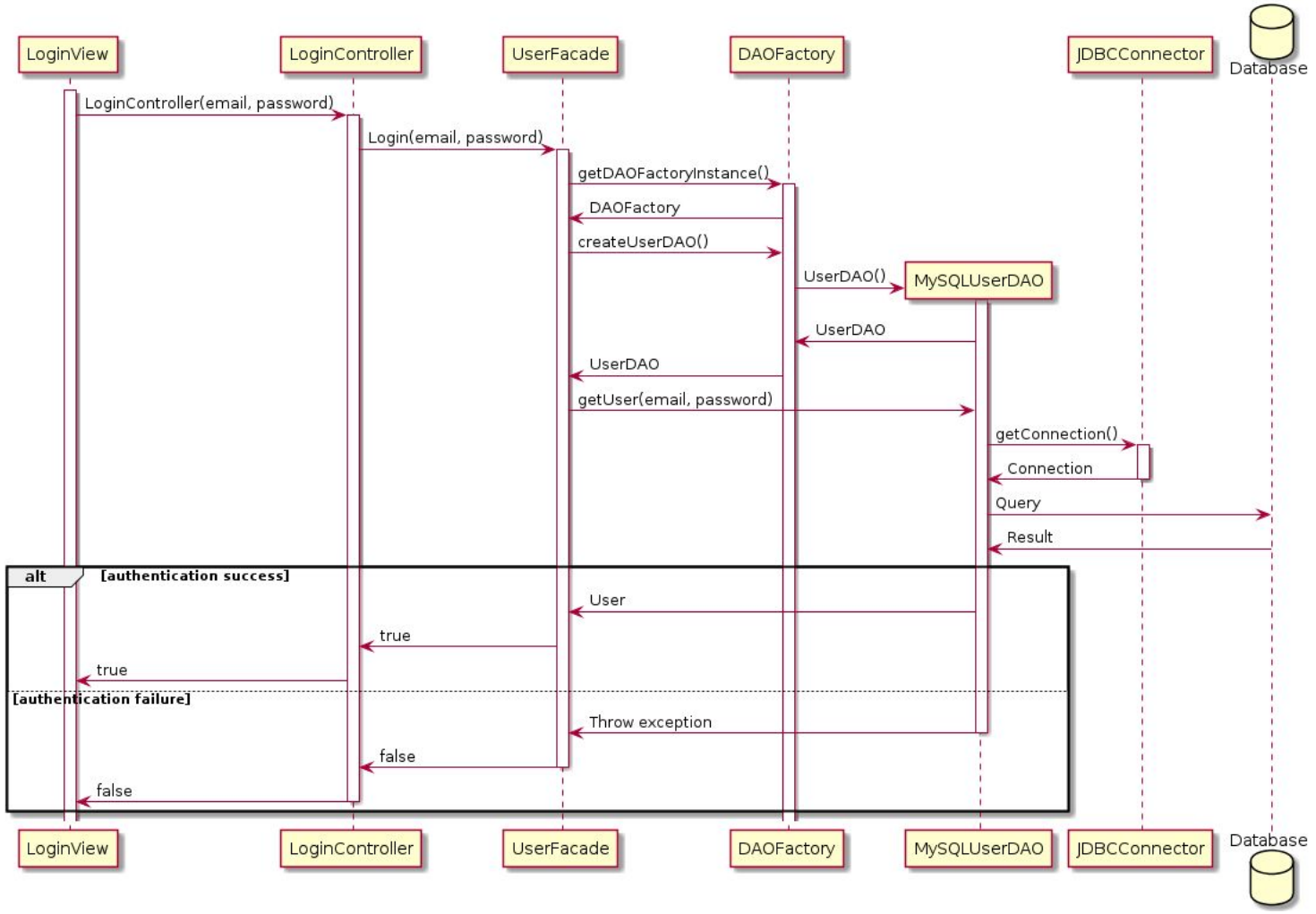
Abstract Factory

The factory design pattern is used to create the family of DAO objects. We need them to collaborate with the database, here we have a MySQL database. The factory will create an object DAO without knowing it's a MySQL DAO. So that we can change the DAO implementation from MySQL to MongoDB later on.
The DAO will return the user to the facade if the user is found and if the password is correct. The program never stores the password, it's the database's task to verify if the password is correct. So no sniffers can intercept the password of our users.

Facade

User facade is a Facade made to hide the complexity of TeamPoint business logic from the GUI. It provides easy to understand functions like login, signUp and deleteUser.
Here the facade asks the factory to create the DAO, after that we can call functions on that DAO to get the user from the database, for example we can call get user for the connection, create user for the inscription and delete user.

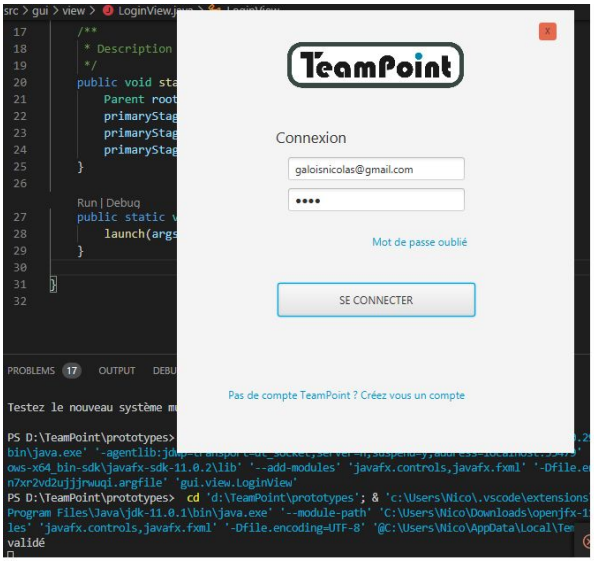
Sequence diagram



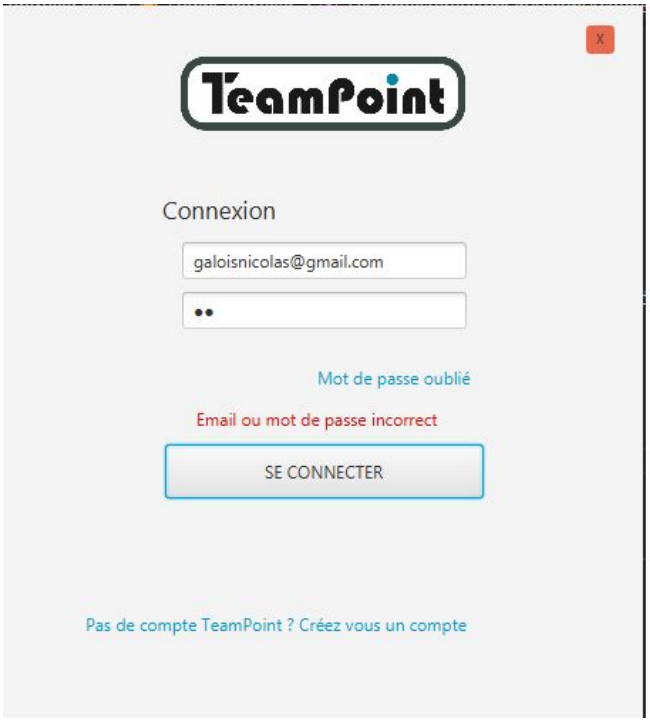
Login Prototype and Conception organization

Connection window :

Correct password :

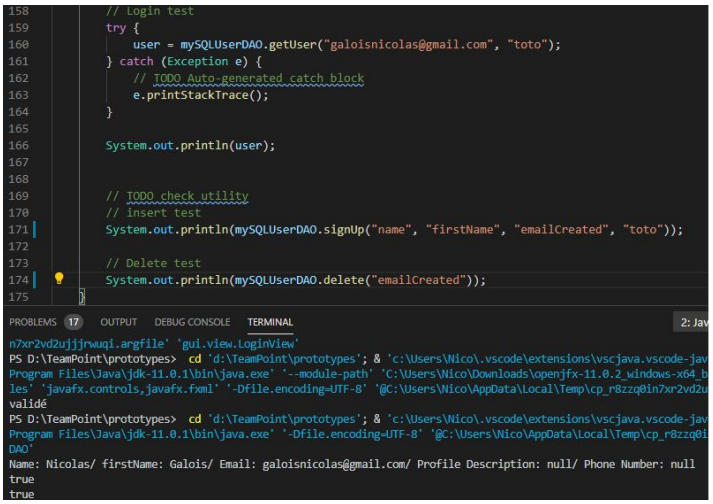


Incorrect password :

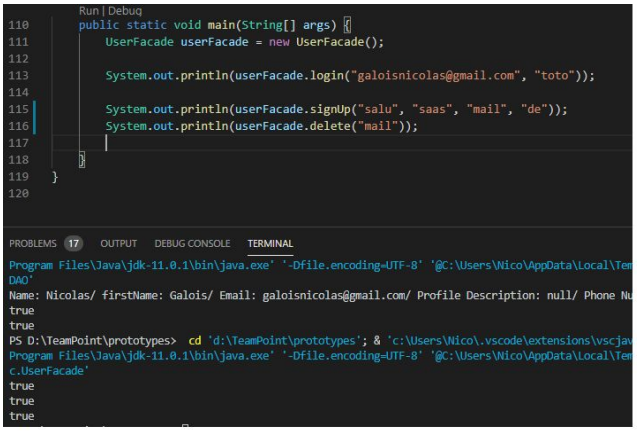


Testing :

Test of UserDAO : getUser, create User and delete user in the database :



Same tests but in UserFacade using DAO :



The Login Prototype manager's word: In this phase we saw and hopefully understood a good implementation of the Abstract Factory and the Facade pattern. Also we have set up the component any team member will work on. The code of the login part is made and working. Some tests have been made directly into the main of some classes to bypass the UI.

Used tools

Class Diagrams :

- We used Google Drive and GitHub to collaborate
- We used Google Docs to collaborate on the final report.
- We used UML Designer to design class diagrams.

Communication:

- Discord
- Messenger

Task breakdown tables

Legend :
CD : Class diagram
BI : Back-end Implementation
FI : Front-end Implementation
SD : Sequence Diagram
T : Testing Implementation

	Salim	Birane	Raphaël	Nicolas
CD				
BI				
FI				
SD				
T				