1-Besoin fonctionnel :

L'application permet de créer un forum en ligne de commande.

D'autres commandes permettent de suivre un fil de discussion et participer.

Exemple d'exécution depuis la console :

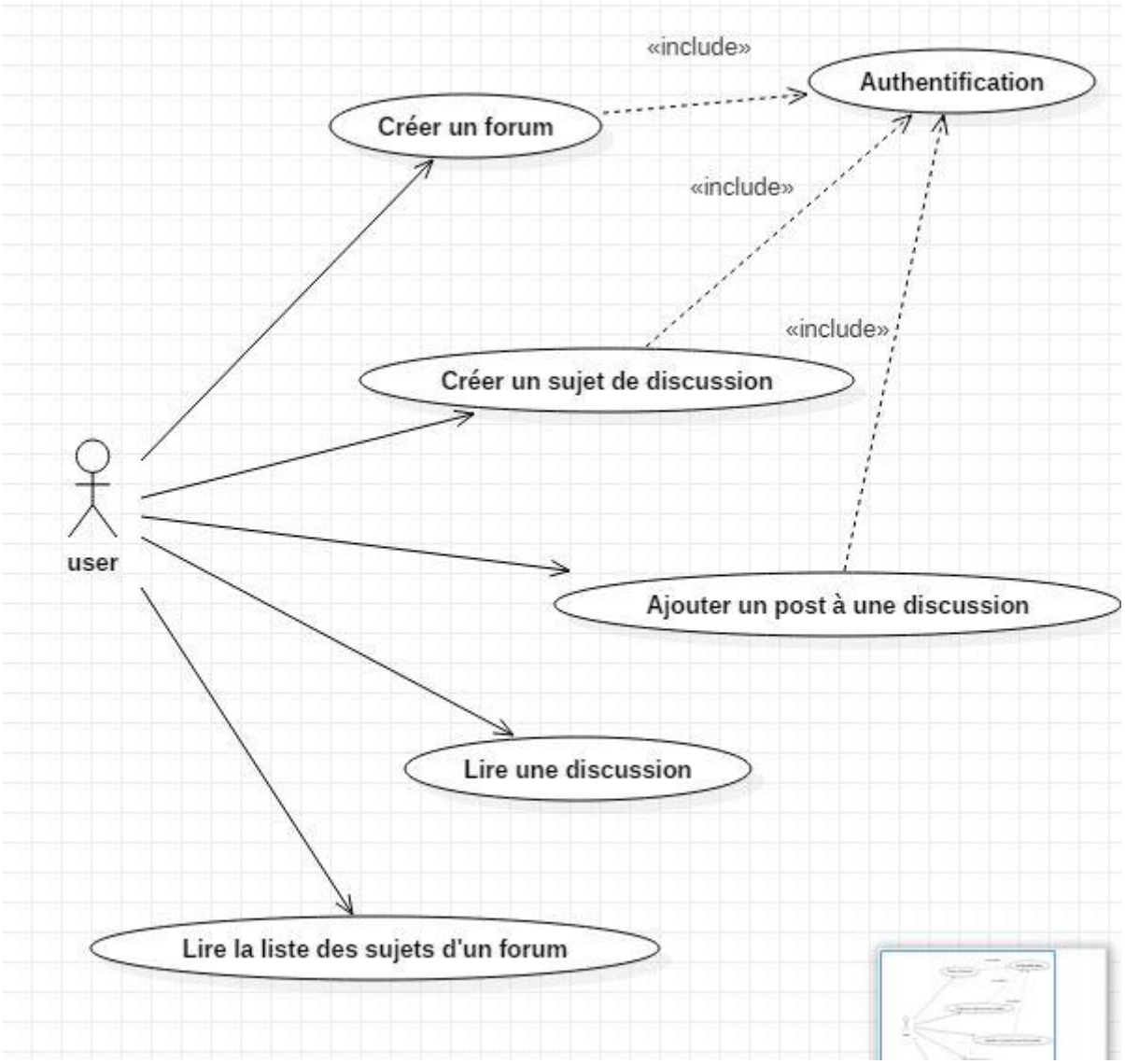
Php appClientForum creer_forum *nom_du_forum*

Php appClientForum ajouter_discussion *nom_du_forum sujet_de_discussion discussion*

Php appClientForum lire_une_discussion *nom_du_forum nom_du_sujet_de_la_discussion*

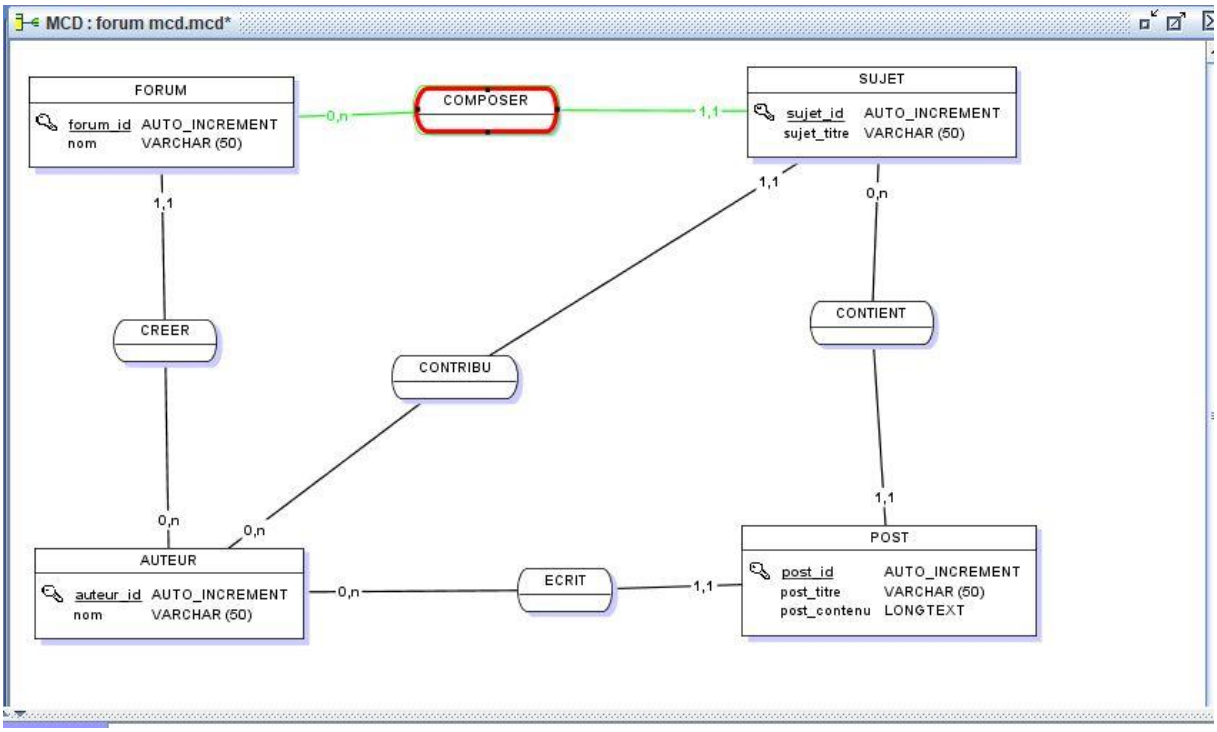
Php appClientForum *Nom_de_la_commande [parametre…]*

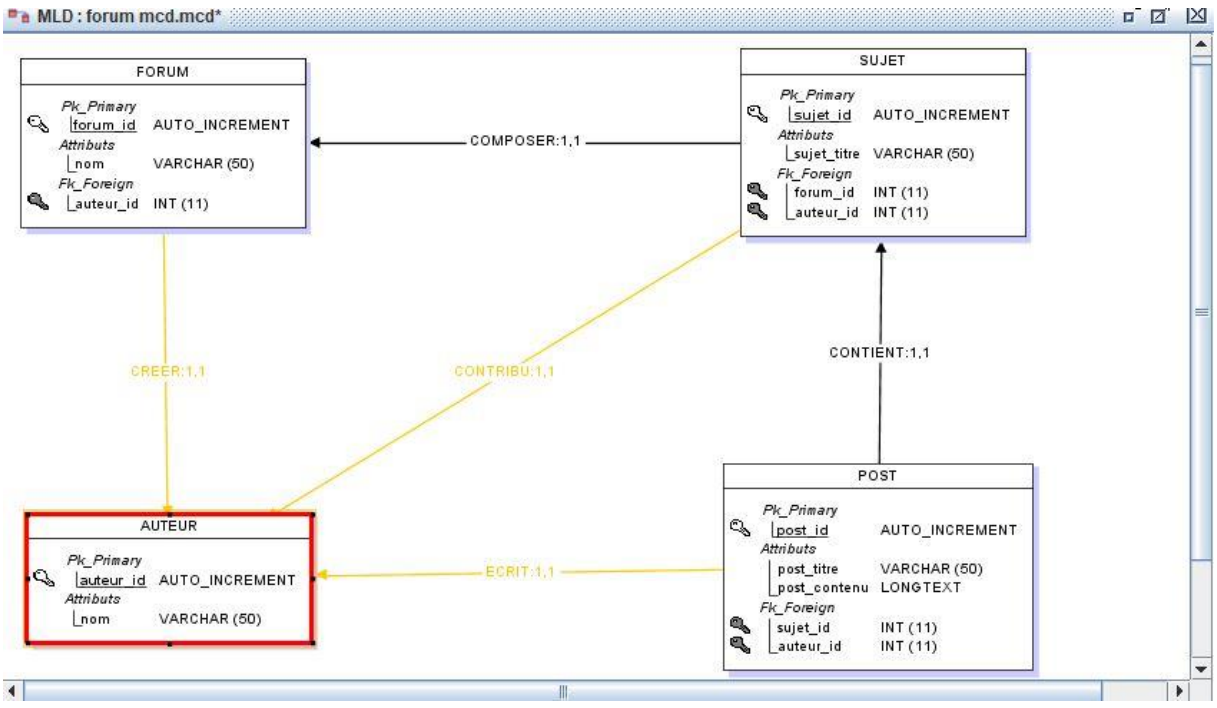
2-Cas d'utilisation :



3-Modèle de données :

MCD/MLD :





Script MySQL :

#----------

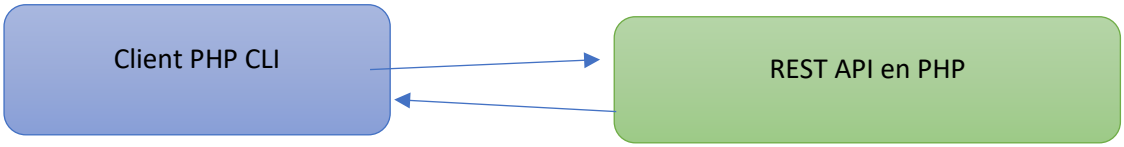
Script MySQL.

#----------

```
#------------------------------------------------------------

# Table: AUTEUR

#------------------------------------------------------------


CREATE TABLE AUTEUR(

      auteur_id Int  Auto_increment  NOT NULL ,

      nom       Varchar (50) NOT NULL

          ,CONSTRAINT AUTEUR_PK PRIMARY KEY (auteur_id)

)ENGINE=InnoDB;



#------------------------------------------------------------

# Table: FORUM

#------------------------------------------------------------


CREATE TABLE FORUM(

      forum_id  Int  Auto_increment  NOT NULL ,

      nom       Varchar (50) NOT NULL ,

      auteur_id Int NOT NULL

          ,CONSTRAINT FORUM_PK PRIMARY KEY (forum_id)


          ,CONSTRAINT FORUM_AUTEUR_FK FOREIGN KEY (auteur_id) REFERENCES
AUTEUR(auteur_id)

)ENGINE=InnoDB;



#------------------------------------------------------------

# Table: SUJET

#------------------------------------------------------------
```

```
CREATE TABLE SUJET(
      sujet_id    Int  Auto_increment  NOT NULL ,
      sujet_titre Varchar (50) NOT NULL ,
      forum_id    Int NOT NULL ,
      auteur_id   Int NOT NULL
          ,CONSTRAINT SUJET_PK PRIMARY KEY (sujet_id)

          ,CONSTRAINT SUJET_FORUM_FK FOREIGN KEY (forum_id) REFERENCES FORUM(forum_id)
          ,CONSTRAINT SUJET_AUTEUR0_FK FOREIGN KEY (auteur_id) REFERENCES AUTEUR(auteur_id)
)ENGINE=InnoDB;



#------------------------------------------------------------
# Table: POST
#------------------------------------------------------------

CREATE TABLE POST(
      post_id     Int  Auto_increment  NOT NULL ,
      post_titre   Varchar (50) NOT NULL ,
      post_contenu Longtext NOT NULL ,
      sujet_id     Int NOT NULL ,
      auteur_id    Int NOT NULL
          ,CONSTRAINT POST_PK PRIMARY KEY (post_id)

          ,CONSTRAINT POST_SUJET_FK FOREIGN KEY (sujet_id) REFERENCES SUJET(sujet_id)
          ,CONSTRAINT POST_AUTEUR0_FK FOREIGN KEY (auteur_id) REFERENCES AUTEUR(auteur_id)
)ENGINE=InnoDB;
```

4-Architecture :



Coté client, utilisation de PHP CLI et d'une API pour interroger une REST API (Httpful, GuzzleHttp…)

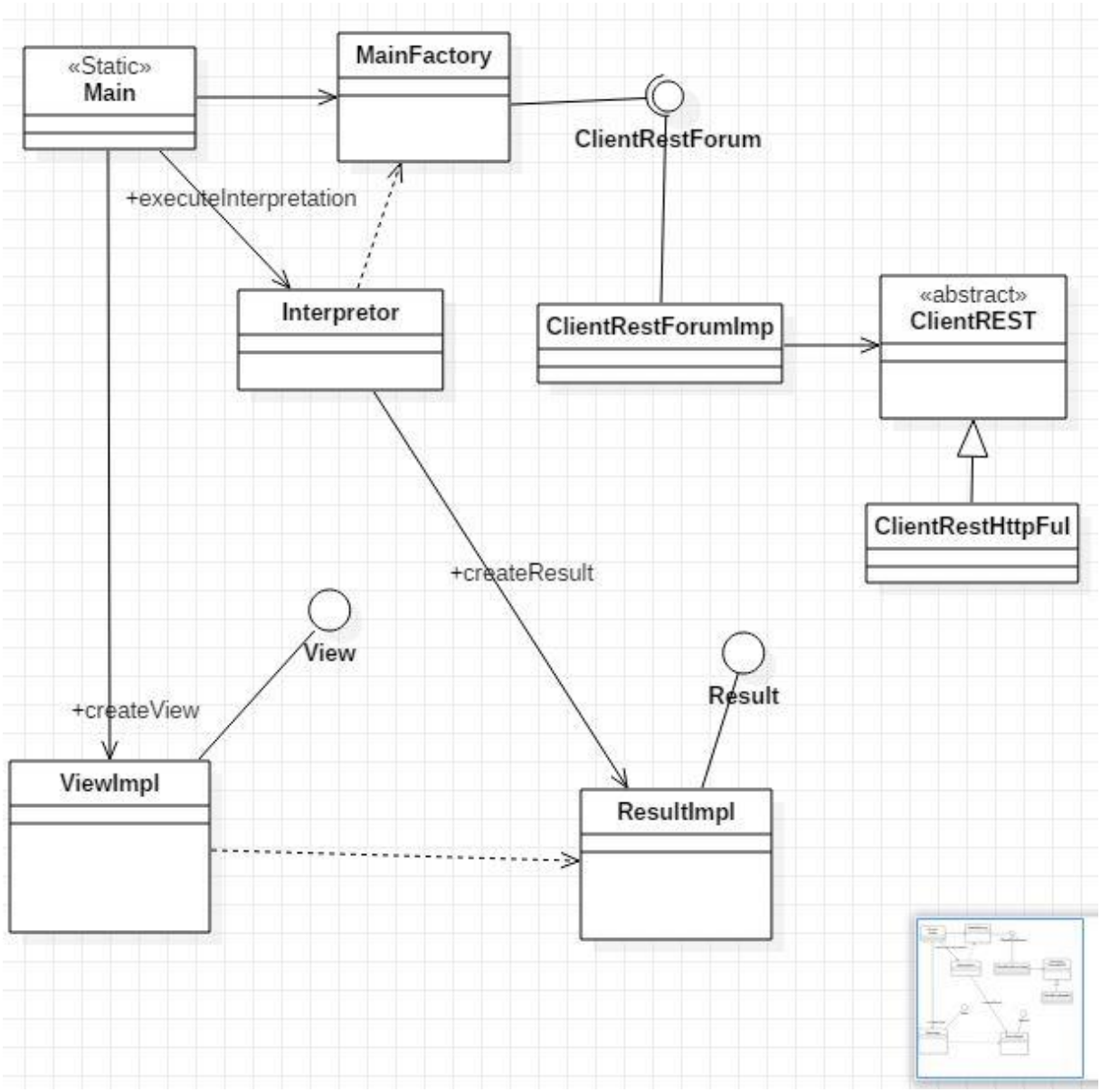
On peut utiliser directement une API pour lancer des requêtes vers la REST API, mais si un jour, on souhaite changer de méthode, ou ne plus utiliser arbitrairement JSON…

L'idée est donc d'encapsuler une solution en utilisant le pattern Strategy , une interface fournira les méthode à utiliser au plus haut niveau de factorisation dans l'application.

On utilise le polymorphisme POO pour pouvoir substituer une classe avec une autre.

On utilise une classe MainFactory (Singleton) pour utiliser l'injection de dépendance assurant un couplage faible entre les objets.

Voici donc le diagramme de classe du client :



Coté Serveur, utilisation d'une REST API avec Symfony et le framework LionFrame pour créer une Rest API, avec les entités mappées avec doctrine.

Voici le lien du repository, pour une ébauche de la solution proposée.

https://github.com/Pakal30/forum_cli.git