

Module : Symfony4

TP4 : Manipulation des entités

Objectifs :

- Création des entités
- Manipulation des entités
- Relations entre les entités

Partie 1 : Création de l'entité Job

Mysql	Doctrine	Symfony
Table	Entité	Classe

NB : Avant toute manipulation assez délicate, veuillez toujours vider le cache.

php bin/console cache:clear

- Création de la base de données

Commencer par paramétrer la base de données.

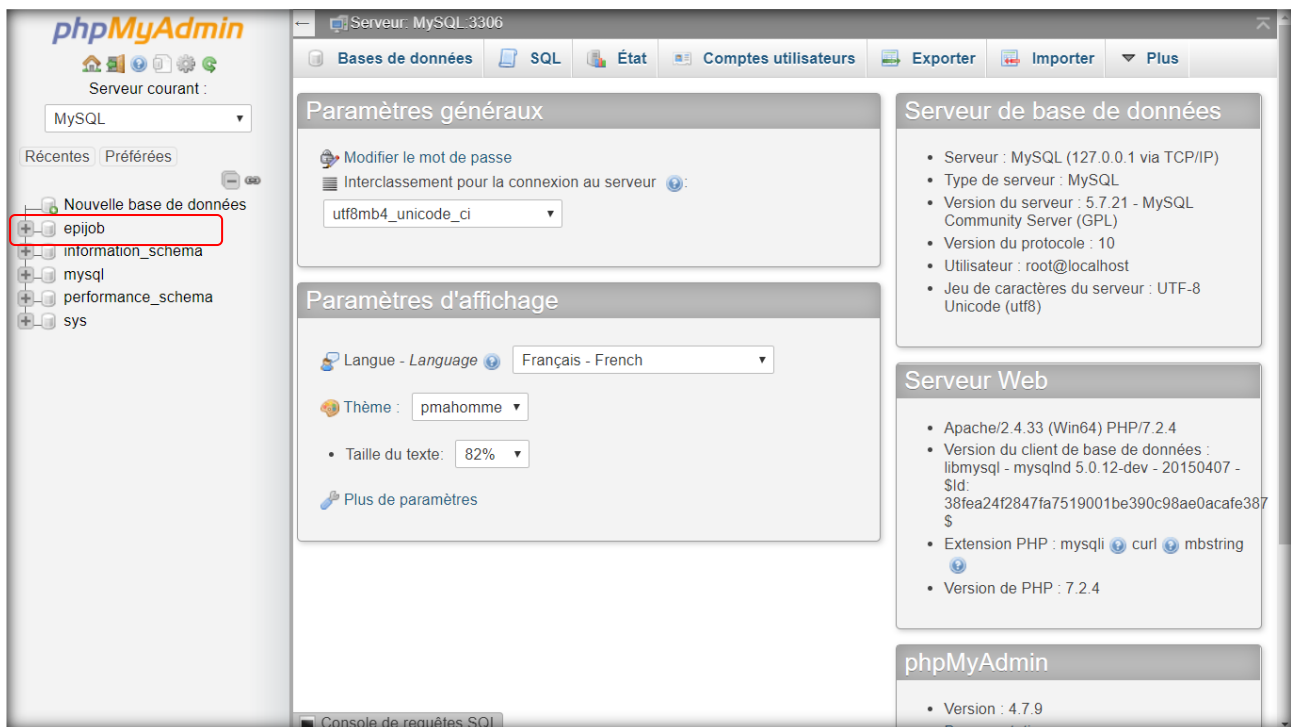


```

1 doctrine:
2   dbal:
3
4     driver: 'pdo_mysql'
5     server_version: '5.7'
6     charset: utf8mb4
7     host: 127.0.0.1
8     port: 3306
9     user: root
10    password:
11    dbname: epijob
12
13
14    #url: '%env(resolve:DATABASE_URL)%'
15
16    # IMPORTANT: You MUST configure your server version,
17    # either here or in the DATABASE_URL env var (see .env file)
18    #server_version: '5.7'
19
20
21  orm:
22    auto_generate_proxy_classes: true
23    naming_strategy: doctrine.orm.naming_strategy.underscore_number_aware
24    auto_mapping: true

```

```
C:\wamp64\www\EPIJOB>php bin/console doctrine:database:create
Created database `epijob` for connection named default
```



➤ Pour visualiser les différentes commandes de doctrine, lancer la cmd suivante :

```
php bin/console list doctrine
```

```
C:\wamp64\www\EPIJOB>php bin/console list doctrine
Symfony 4.3.4 (env: dev, debug: true)

Usage:
  command [options] [arguments]

Options:
  -h, --help            Display this help message
  -q, --quiet            Do not output any message
  -V, --version          Display this application version
      --ansi             Force ANSI output
      --no-ansi          Disable ANSI output
  -n, --no-interaction  Do not ask any interactive question
  -e, --env=ENV          The Environment name. [default: "dev"]
      --no-debug         Switches off debug mode.
  -v|vv|vvv, --verbose  Increase the verbosity of messages: 1 for normal output, 2 for more verbose output and 3 for debug

Available commands for the "doctrine" namespace:
doctrine:cache:clear                Flush a given cache
doctrine:cache:clear-collection-region  Clear a second-level cache collection region
doctrine:cache:clear-entity-region    Clear a second-level cache entity region
doctrine:cache:clear-metadata        Clears all metadata cache for an entity manager
doctrine:cache:clear-query           Clears all query cache for an entity manager
doctrine:cache:clear-query-region     Clear a second-level cache query region
doctrine:cache:clear-result          Clears result cache for an entity manager
doctrine:cache:contains              Check if a cache entry exists
doctrine:cache:delete                Delete a cache entry
doctrine:cache:flush                 [doctrine:cache:clear] Flush a given cache
doctrine:cache:stats                 Get stats on a given cache provider
doctrine:database:create              Creates the configured database
doctrine:database:drop                Drops the configured database
doctrine:database:import              Import SQL file(s) directly to Database.
doctrine:database:production-settings  Verify that Doctrine is properly configured for a production environment
doctrine:generate:entities            [generate:doctrine:entities] Generates entity classes and method stubs from your mapping information
doctrine:mapping:convert              [orm:convert:mapping] Convert mapping information between supported formats
doctrine:mapping:import               Imports mapping information from an existing database
doctrine:mapping:info                 [diff] Generate a migration by comparing your current database to your mapping information.
doctrine:migrations:diff              [diff] Generate a migration by comparing your current database to your mapping information.
doctrine:migrations:dump-schema       [dump-schema] Dump the schema for your database to a migration.
doctrine:migrations:execute           [execute] Execute a single migration version up or down manually.
doctrine:migrations:generate          [generate] Generate a blank migration class.
doctrine:migrations:latest            [latest] Outputs the latest version number
doctrine:migrations:migrate           [migrate] Execute a migration to a specified version or the latest available version.
doctrine:migrations:rollback          [rollback] Rollup migrations by deleting all tracked versions and insert the one version that exists.
doctrine:migrations:status            [status] View the status of a set of migrations.
doctrine:migrations:up-to-date        [up-to-date] Tells you if your schema is up-to-date.
doctrine:migrations:version           [version] Manually add and delete migration versions from the version table.
```

➤ Génération de l'entité

- ✓ Lancer la commande de génération d'entité :

```
php bin/console make:entity
```

Grâce à ce que le générateur vous demande, il faut entrer :

- **Nom** : le nom de l'entité sous le format `NomEntité`.
- **Définition d'un champ** : les noms de nos champs, type du champ, est-il facultatif, est-il unique
- Rajouter les champs suivants :

```
title:

    type: string

    length: 255

    nullable: false

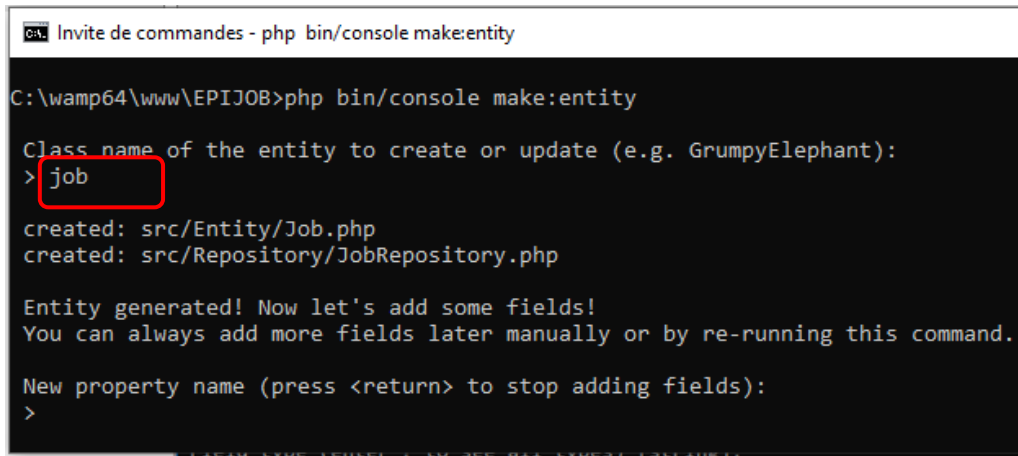
company:

    type: string

    length: 255

description:
```

```
        type: text
is_activated:
    type: boolean
    nullable: true
expires_at:
    type: datetime
```



```
Invite de commandes - php bin/console make:entity

C:\wamp64\www\EPIJOB>php bin/console make:entity

Class name of the entity to create or update (e.g. GrumpyElephant):
> job
created: src/Entity/Job.php
created: src/Repository/JobRepository.php

Entity generated! Now let's add some fields!
You can always add more fields later manually or by re-running this command.

New property name (press <return> to stop adding fields):
>
```

```
Invite de commandes - php bin/console make:entity

New property name (press <return> to stop adding fields):
> title

Field type (enter ? to see all types) [string]:
> string

Field length [255]:
>

Can this field be null in the database (nullable) (yes/no) [no]:
>

updated: src/Entity/Job.php

Add another property? Enter the property name (or press <return> to stop adding fields):
> .
```

Invite de commandes - php bin/console make:entity

```
Add another property? Enter the property name (or press <return> to stop adding fields):
> company
Field type (enter ? to see all types) [string]:
>
Field length [255]:
>
Can this field be null in the database (nullable) (yes/no) [no]:
>
updated: src/Entity/Job.php

Add another property? Enter the property name (or press <return> to stop adding fields):
> description
Field type (enter ? to see all types) [string]:
> text
Can this field be null in the database (nullable) (yes/no) [no]:
>
updated: src/Entity/Job.php

Add another property? Enter the property name (or press <return> to stop adding fields):
> is_activated
Field type (enter ? to see all types) [boolean]:
>
Can this field be null in the database (nullable) (yes/no) [no]:
>
updated: src/Entity/Job.php

Add another property? Enter the property name (or press <return> to stop adding fields):
> expires_at
Field type (enter ? to see all types) [datetime]:
```

➤ Générer l'entité en appuyant « entrée » après le dernier champ :

```
Invite de commandes

>

updated: src/Entity/Job.php

Add another property? Enter the property name (or press <return> to stop adding fields):
>

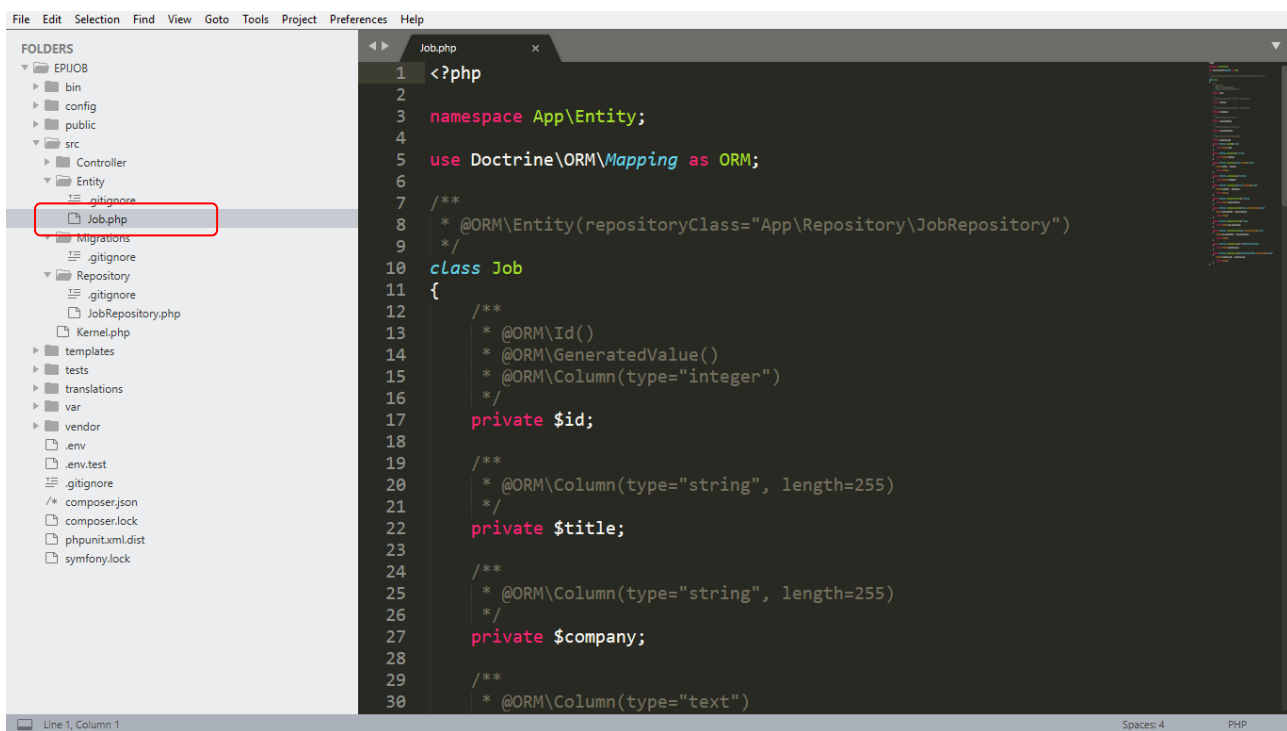
Success!

Next: When you're ready, create a migration with make:migration

C:\wamp64\www\EPIJOB>
```

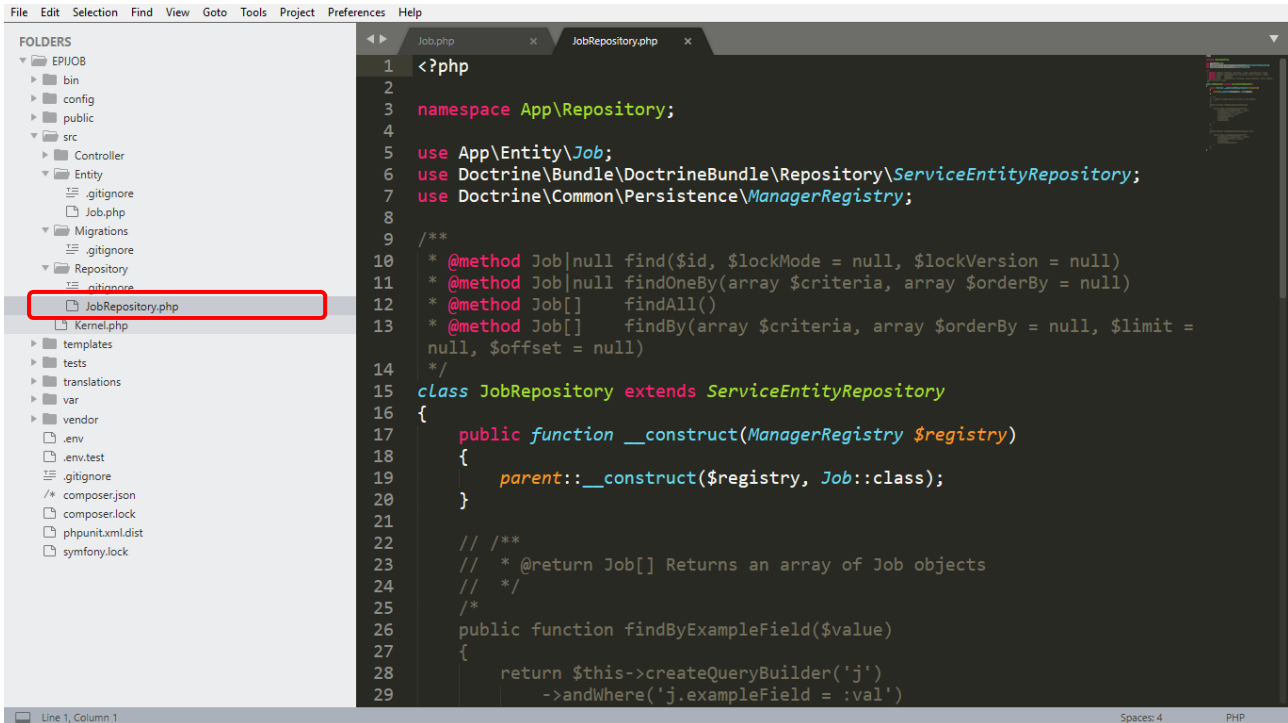
➤ **Vérifier la création des fichiers suivants :**

- ✓ **La classe job.php :**
 - **Path:** nom_projet\src\Entity
 - **Contenu :** champs + getters()+ setters()
- ✓ **Le repository jobRepository.php :**
 - **Path:** nom_projet \src\ Repository
 - **Contenu :** Requetes sql



The screenshot shows an IDE with a file explorer on the left and a code editor on the right. In the file explorer, the 'Entity' folder under 'src' is expanded, and 'Job.php' is highlighted with a red rectangle. The code editor displays the content of 'Job.php' with line numbers 1 through 30. The code is a PHP class named 'Job' within the 'App\Entity' namespace, using 'Doctrine\ORM\Mapping' as 'ORM'. It includes annotations for an ID, a title, and a company, each with a length of 255 characters.

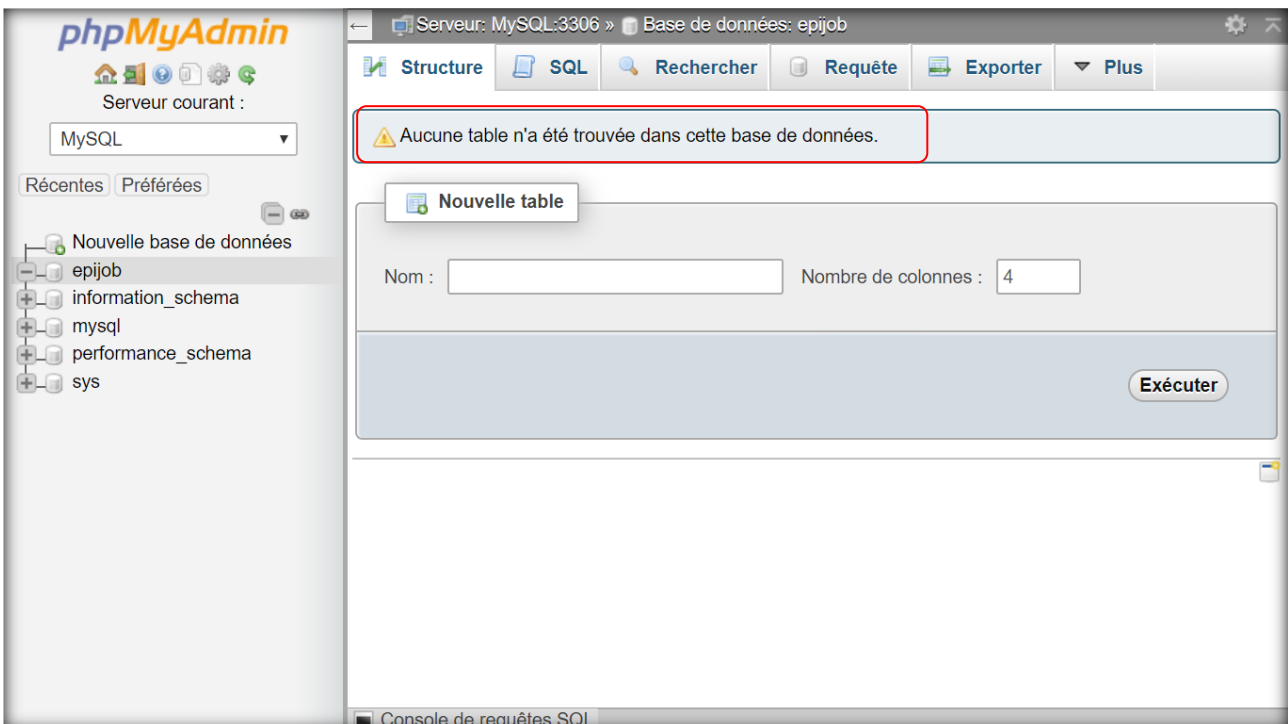
```
1 <?php
2
3 namespace App\Entity;
4
5 use Doctrine\ORM\Mapping as ORM;
6
7 /**
8  * @ORM\Entity(repositoryClass="App\Repository\JobRepository")
9  */
10 class Job
11 {
12     /**
13      * @ORM\Id()
14      * @ORM\GeneratedValue()
15      * @ORM\Column(type="integer")
16      */
17     private $id;
18
19     /**
20      * @ORM\Column(type="string", length=255)
21      */
22     private $title;
23
24     /**
25      * @ORM\Column(type="string", length=255)
26      */
27     private $company;
28
29     /**
30      * @ORM\Column(type="text")
```



```

1  <?php
2
3  namespace App\Repository;
4
5  use App\Entity\Job;
6  use Doctrine\Bundle\DoctrineBundle\Repository\ServiceEntityRepository;
7  use Doctrine\Common\Persistence\ManagerRegistry;
8
9  /**
10   * @method Job|null find($id, $lockMode = null, $lockVersion = null)
11   * @method Job|null findOneBy(array $criteria, array $orderBy = null)
12   * @method Job[]  findAll()
13   * @method Job[]  findBy(array $criteria, array $orderBy = null, $limit =
14   null, $offset = null)
15   */
16  class JobRepository extends ServiceEntityRepository
17  {
18      public function __construct(ManagerRegistry $registry)
19      {
20          parent::__construct($registry, Job::class);
21      }
22
23      // /**
24      //  * @return Job[] Returns an array of Job objects
25      //  */
26      public function findByExampleField($value)
27      {
28          return $this->createQueryBuilder('j')
29              ->andWhere('j.exampleField = :val')

```



phpMyAdmin

Serveur courant : MySQL

Récentes Préférées

- Nouvelle base de données
- epijob
- information_schema
- mysql
- performance_schema
- sys

Structure SQL Rechercher Requête Exporter Plus

Aucune table n'a été trouvée dans cette base de données.

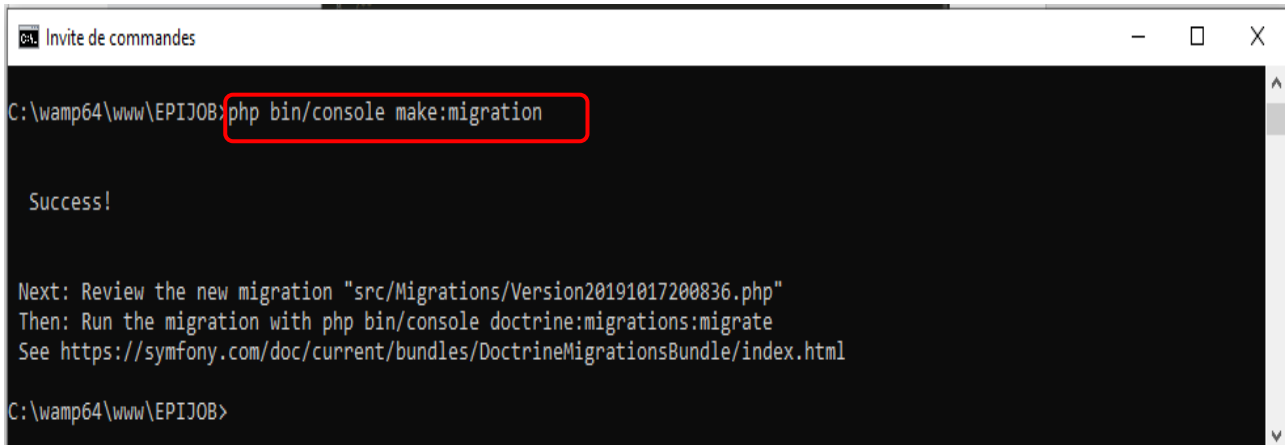
Nouvelle table

Nom : Nombre de colonnes : 4

Exécuter

Console de requêtes SQL


```
php bin/console make:migration
```



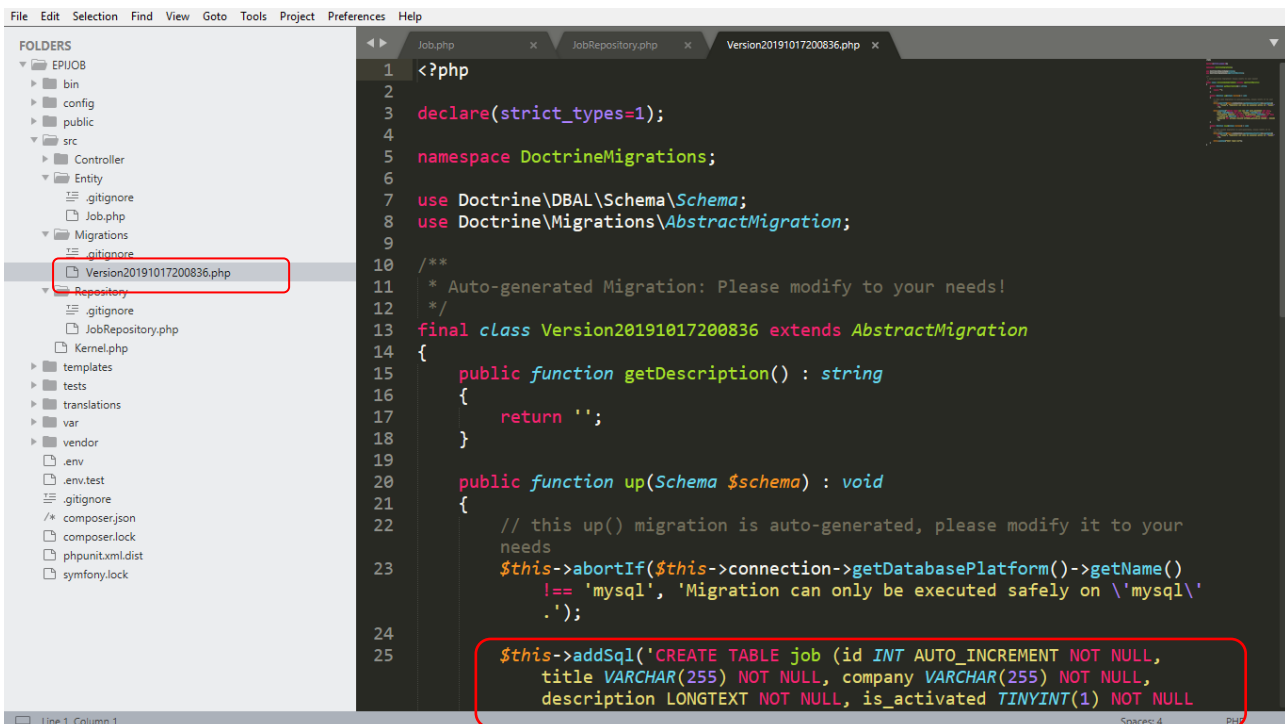
```
Invite de commandes

C:\wamp64\www\EPIJOB>php bin/console make:migration

Success!

Next: Review the new migration "src/Migrations/Version20191017200836.php"
Then: Run the migration with php bin/console doctrine:migrations:migrate
See https://symfony.com/doc/current/bundles/DoctrineMigrationsBundle/index.html

C:\wamp64\www\EPIJOB>
```



```
File Edit Selection Find View Goto Tools Project Preferences Help

FOLDERS
  EPIJOB
    bin
    config
    public
    src
      Controller
      Entity
        .gitignore
        Job.php
      Migrations
        .gitignore
        Version20191017200836.php
      Repository
        .gitignore
        JobRepository.php
        Kernel.php
    templates
    tests
    translations
    var
    vendor
      .env
      .env.test
      .gitignore
      composer.json
      composer.lock
      phpunit.xml.dist
      symfony.lock

1  <?php
2
3  declare(strict_types=1);
4
5  namespace Doctrine\Migrations;
6
7  use Doctrine\DBAL\Schema\Schema;
8  use Doctrine\Migrations\AbstractMigration;
9
10 /**
11  * Auto-generated Migration: Please modify to your needs!
12  */
13 final class Version20191017200836 extends AbstractMigration
14 {
15     public function getDescription() : string
16     {
17         return '';
18     }
19
20     public function up(Schema $schema) : void
21     {
22         // this up() migration is auto-generated, please modify it to your
23         // needs
24         $this->abortIf($this->connection->getDatabasePlatform()->getName()
25             !== 'mysql', 'Migration can only be executed safely on \'mysql\'
26             .');
27
28         $this->addSql('CREATE TABLE job (id INT AUTO_INCREMENT NOT NULL,
29             title VARCHAR(255) NOT NULL, company VARCHAR(255) NOT NULL,
30             description LONGTEXT NOT NULL, is_activated TINYINT(1) NOT NULL
```

```
php bin/console doctrine:migrations:migrate
```

```
Sélection Invite de commandes

C:\wamp64\www\EPIJOB>php bin/console make:migration

Success!

Next: Review the new migration "src/Migrations/Version20191017200836.php"
Then: Run the migration with php bin/console doctrine:migrations:migrate
See https://symfony.com/doc/current/bundles/DoctrineMigrationsBundle/index.html

C:\wamp64\www\EPIJOB>php bin/console doctrine:migrations:migrate

Application Migrations

WARNING! You are about to execute a database migration that could result in schema changes and data loss. Are you sure you wish to continue? (y/n)y
Migrating up to 20191017200836 from 0

++ migrating 20191017200836

-> CREATE TABLE job (id INT AUTO_INCREMENT NOT NULL, title VARCHAR(255) NOT NULL, company VARCHAR(255) NOT NULL, description LONGTEXT NOT NULL, is_activated TINYINT(1) NOT NULL, expires_at DATETIME NOT NULL, PRIMARY KEY(id)) DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci ENGINE = InnoDB

++ migrated (took 4457.2ms, used 16M memory)

-----

++ finished in 4686.3ms
++ used 16M memory
++ 1 migrations executed
++ 1 sql queries

C:\wamp64\www\EPIJOB>
```

phpMyAdmin

Serveur courant : MySQL

Récentes Préférences

- Nouvelle base de données
- epijob
- information_schema
- mysql
- performance_schema
- sys

Serveur: MySQL:3306 » Base de données: epijob

Structure SQL Rechercher Requête Exporter Plus

Filtres

Contenant le mot :

Table Action

☒ **job** ☐ Parcourir ☐ Structure ☐ Rechercher ☐ Insérer ☐ Vider ☐ Supprimer

☐ migration_versions ☐ Parcourir ☐ Structure ☐ Rechercher ☐ Insérer ☐ Vider ☐ Supprimer

2 tables Somme

☐ Tout cocher Avec la sélection :

Imprimer Dictionnaire de données

Nouvelle table

Nom : Nombre de colonnes : 4

Console de requêtes SQL Exécuter

phpMyAdmin

Serveur courant : MySQL

Récentes Préférences

- Nouvelle base de données
- epijob
- information_schema
- mysql
- performance_schema
- sys

Serveur: MySQL:3306 » Base de données: epijob » Table: job

Parcourir Structure SQL Rechercher Insérer Plus

Structure de table Vue relationnelle

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra
1	id	int(11)			Non	Aucun(e)		AUTO_INCREMENT
2	title	varchar(255)	utf8mb4_unicode_ci		Non	Aucun(e)		
3	company	varchar(255)	utf8mb4_unicode_ci		Non	Aucun(e)		
4	description	longtext	utf8mb4_unicode_ci		Non	Aucun(e)		
5	is_activated	tinyint(1)			Non	Aucun(e)		
6	expires_at	datetime			Non	Aucun(e)		

☐ Tout cocher Avec la sélection : ☐ Parcourir ☐ Modifier ☐ Supprimer

☐ Primaire ☐ Unique ☐ Index

Imprimer Suggérer des optimisations de structure Déplacer des colonnes

Améliorer la structure de la table

Ajouter 1 colonne(s) après expires_at Exécuter

Console de requêtes SQL

➤ **Comment modifier une entité ?**

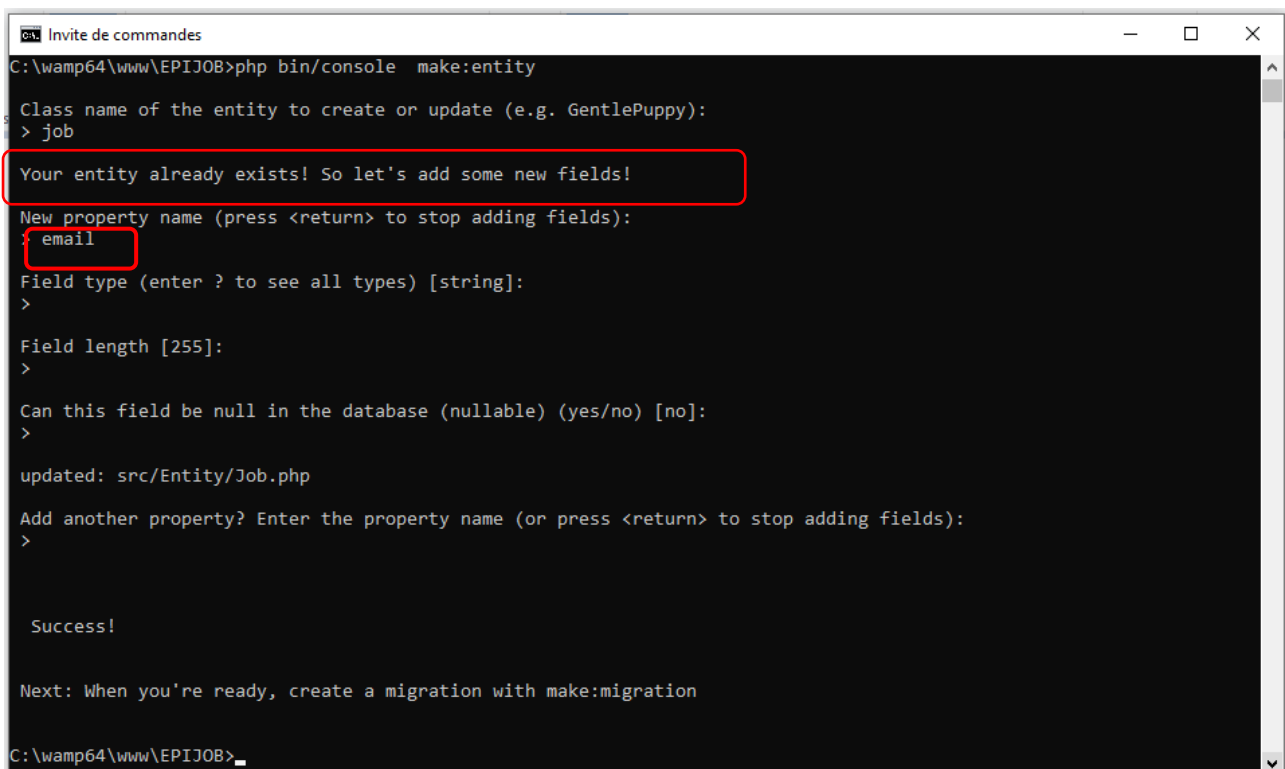
Pour modifier une entité, on peut soit le faire à la main dans `Entity/job.php` ou bien le générer automatiquement

✓ On se propose d'ajouter l'attribut email à notre entité job :

name : email

type: string

length: 255



```
Invite de commandes
C:\wamp64\www\EPIJOB>php bin/console make:entity

Class name of the entity to create or update (e.g. GentlePuppy):
> job

Your entity already exists! So let's add some new fields!

New property name (press <return> to stop adding fields):
email

Field type (enter ? to see all types) [string]:
>

Field length [255]:
>

Can this field be null in the database (nullable) (yes/no) [no]:
>

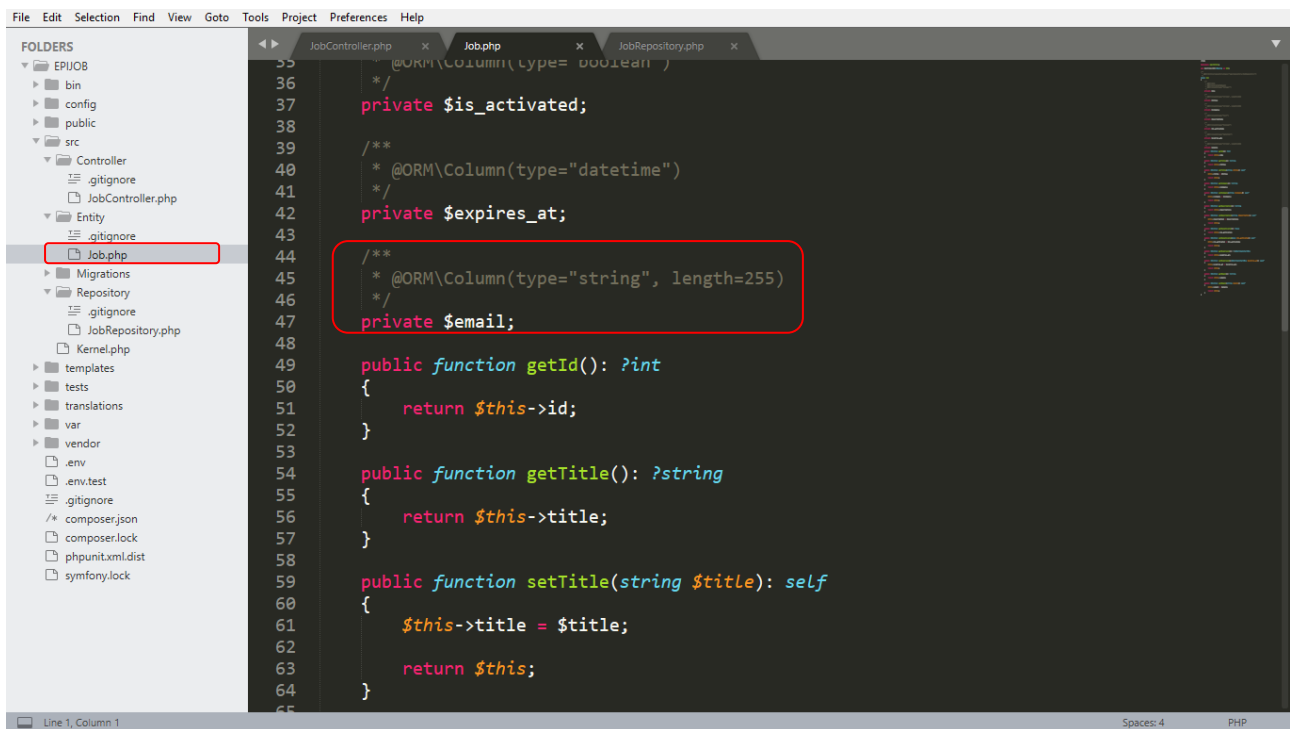
updated: src/Entity/Job.php

Add another property? Enter the property name (or press <return> to stop adding fields):
>

Success!

Next: When you're ready, create a migration with make:migration

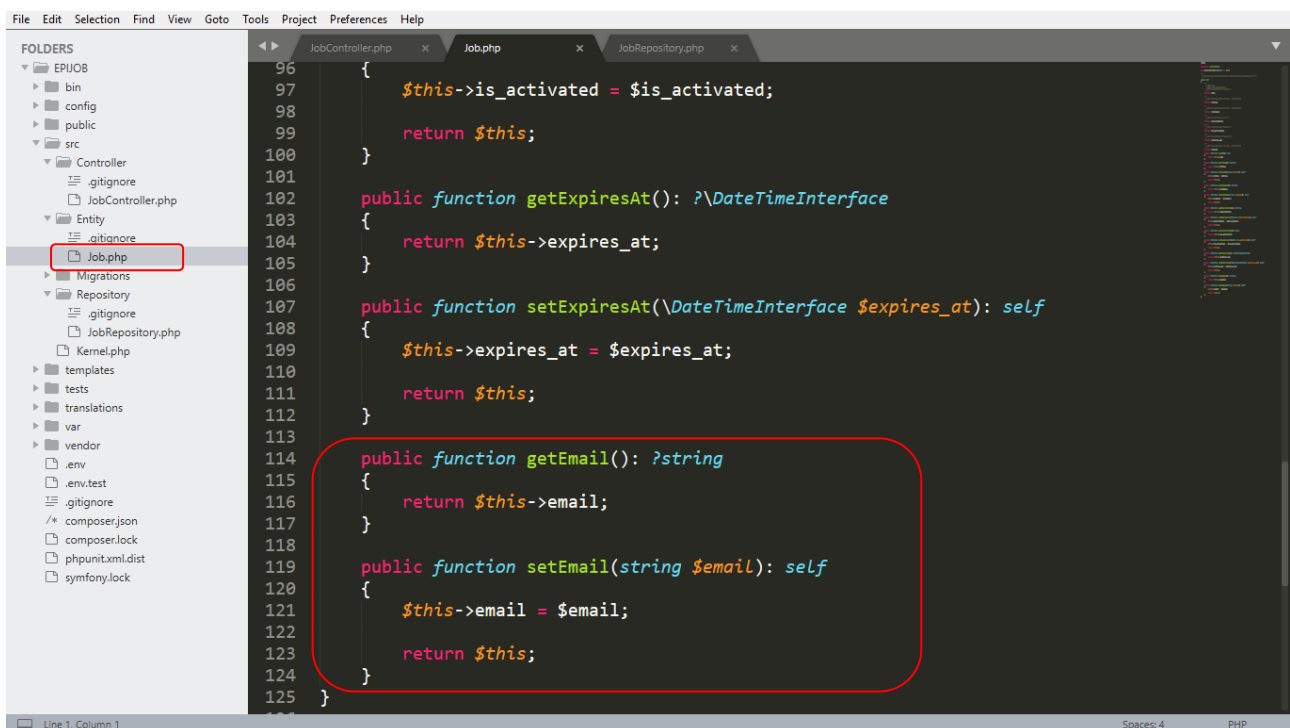
C:\wamp64\www\EPIJOB>
```



```

35  * @ORM\Column(type="boolean")
36  */
37  private $is_activated;
38
39  /**
40   * @ORM\Column(type="datetime")
41   */
42  private $expires_at;
43
44  /**
45   * @ORM\Column(type="string", length=255)
46   */
47  private $email;
48
49  public function getId(): ?int
50  {
51      return $this->id;
52  }
53
54  public function getTitle(): ?string
55  {
56      return $this->title;
57  }
58
59  public function setTitle(string $title): self
60  {
61      $this->title = $title;
62
63      return $this;
64  }
65

```



```

96  {
97      $this->is_activated = $is_activated;
98
99      return $this;
100  }
101
102  public function getExpiresAt(): ?\DateTimeInterface
103  {
104      return $this->expires_at;
105  }
106
107  public function setExpiresAt(\DateTimeInterface $expires_at): self
108  {
109      $this->expires_at = $expires_at;
110
111      return $this;
112  }
113
114  public function getEmail(): ?string
115  {
116      return $this->email;
117  }
118
119  public function setEmail(string $email): self
120  {
121      $this->email = $email;
122
123      return $this;
124  }
125  }

```

```
php bin/console make:migration
```

```
Invite de commandes
C:\wamp64\www\EPIJOB>php bin/console make:migration

Success!

Next: Review the new migration "src/Migrations/Version20191022194333.php"
Then: Run the migration with php bin/console doctrine:migrations:migrate
See https://symfony.com/doc/current/bundles/DoctrineMigrationsBundle/index.html

C:\wamp64\www\EPIJOB>
```

```
php bin/console doctrine:migrations:migrate
```

```
Invite de commandes
C:\wamp64\www\EPIJOB>php bin/console doctrine:migrations:migrate

Application Migrations

WARNING! You are about to execute a database migration that could result in schema changes and data loss. Are you sure you wish to continue? (y/n)y
Migrating up to 20191022194333 from 20191017200836

++ migrating 20191022194333
  -> ALTER TABLE job ADD email VARCHAR(255) NOT NULL
++ migrated (took 5883.2ms, used 16M memory)

-----

++ finished in 6186.9ms
++ used 16M memory
++ 1 migrations executed
++ 1 sql queries

C:\wamp64\www\EPIJOB>
```

phpMyAdmin

Serveur courant : MySQL

Récentes Préférées

- Nouvelle base de données
- epijob
 - Nouvelle table
 - job
 - migration_versions
- information_schema
- mysql
- performance_schema
- sys

Serveur: MySQL:3306 » Base de données: epijob » Table: job

Parcourir Structure SQL Rechercher Insérer Exporter Importer Plus

Structure de table Vue relationnelle

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra	Action
1	id	int(11)			Non	Aucun(e)		AUTO_INCREMENT	Modifier
2	title	varchar(255)	utf8mb4_unicode_ci		Non	Aucun(e)			Modifier
3	company	varchar(255)	utf8mb4_unicode_ci		Non	Aucun(e)			Modifier
4	description	longtext	utf8mb4_unicode_ci		Non	Aucun(e)			Modifier
5	is_activated	tinyint(1)			Non	Aucun(e)			Modifier
6	expires_at	datetime			Non	Aucun(e)			Modifier
7	email	varchar(255)	utf8mb4_unicode_ci		Non	Aucun(e)			Modifier

Tout cocher Avec la sélection : Parcourir Modifier Supprimer Primaire Unique Index

Imprimer Suggérer des optimisations de structure Déplacer des colonnes Améliorer la structure de la table

Ajouter 1 colonne(s) après email Exécuter

Index

Action	Nom de l'index	Type	Unique	Compressé	Colonne	Cardinalité	Interclassement	Null	Commentaire
Éditer Supprimer	PRIMARY	BTREE	Oui	Non	id	0	A	Non	

Console de requêtes SQL

Partie 2: Doctrine

- **Le service Doctrine** : gère la base de données. Il s'occupe de deux choses :

- ✓ **Les différentes connexions à des bases de données :**

C'est la partie **DBAL** de Doctrine (**DataBase Abstraction Layer**).

En effet, vous pouvez utiliser plusieurs connexions à plusieurs bases de données différentes.

Le service Doctrine dispose d'une méthode qui récupère une connexion à partir de son nom : `$doctrine->getConnection($name)`

- ✓ **Les différents gestionnaires d'entités, ou EntityManager :**

C'est la partie **ORM** de Doctrine (**Object-Relational Mapping**).

On peut utiliser plusieurs gestionnaires d'entités, à condition d'en utiliser un par connexion !

Le service dispose de la méthode qui récupère un ORM à partir de son nom : `$doctrine->getManager($name)`.

- **Le service EntityManager**

C'est le service EntityManager qui permet de dire à Doctrine « Persiste cet objet », c'est lui qui va exécuter les requêtes SQL.

```
<?php
$em = $this->getDoctrine()->getManager();
```

Le seul problème c'est qu'il ne peut pas récupérer facilement les entités depuis la base de données.

Pour faciliter l'accès aux objets, on va utiliser un Repository

- **Les repositories**

On parle des repositories au pluriel car il en existe **un par entité**.

Quand on parle d'un repository en particulier, il faut donc toujours préciser le repository de quelle entité, afin de bien savoir de quoi on parle.

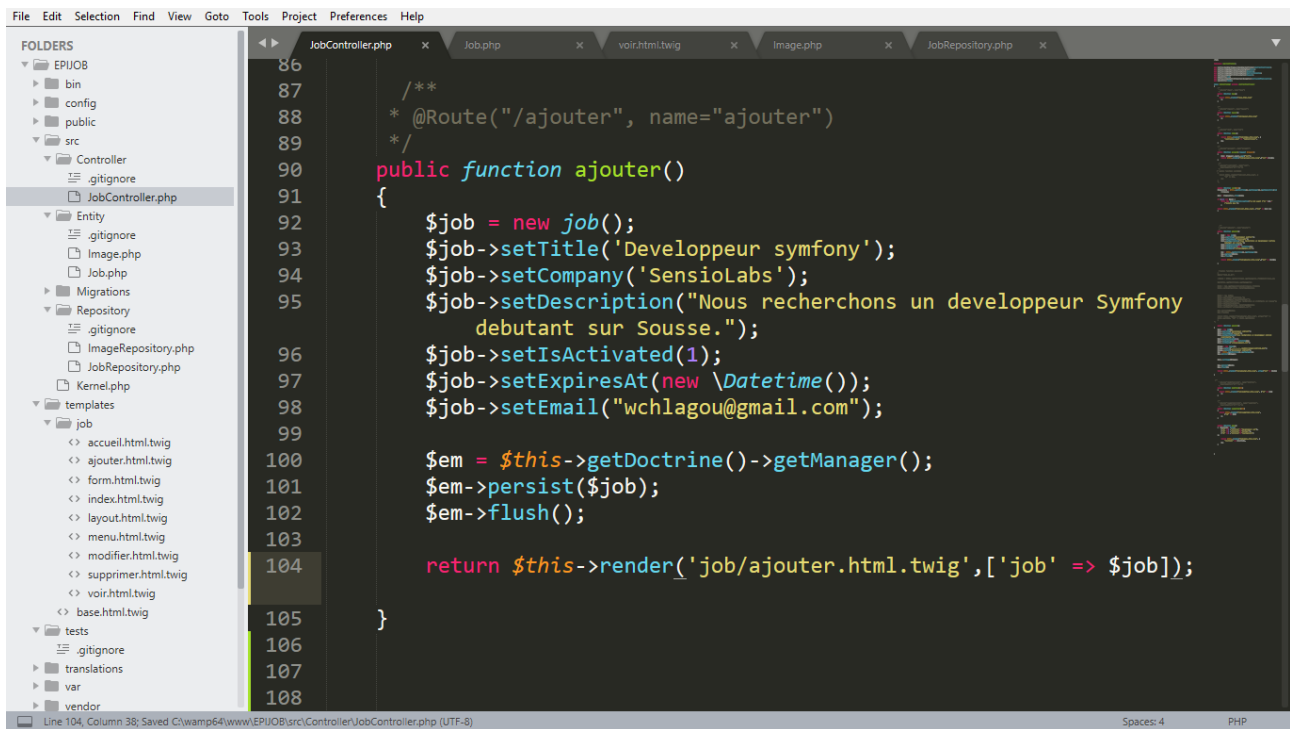
```
<?php
$em = $this->getDoctrine()->getManager();
$jobRepository = $em->getRepository('job');
```

L'EntityManager : sert à **manipuler** les entités,

Les repositories : servent à **récupérer** les entités.

Partie3 : Enregistrement des entités dans la base de données :

➤ Insérer un job en BD via le contrôleur JobController :

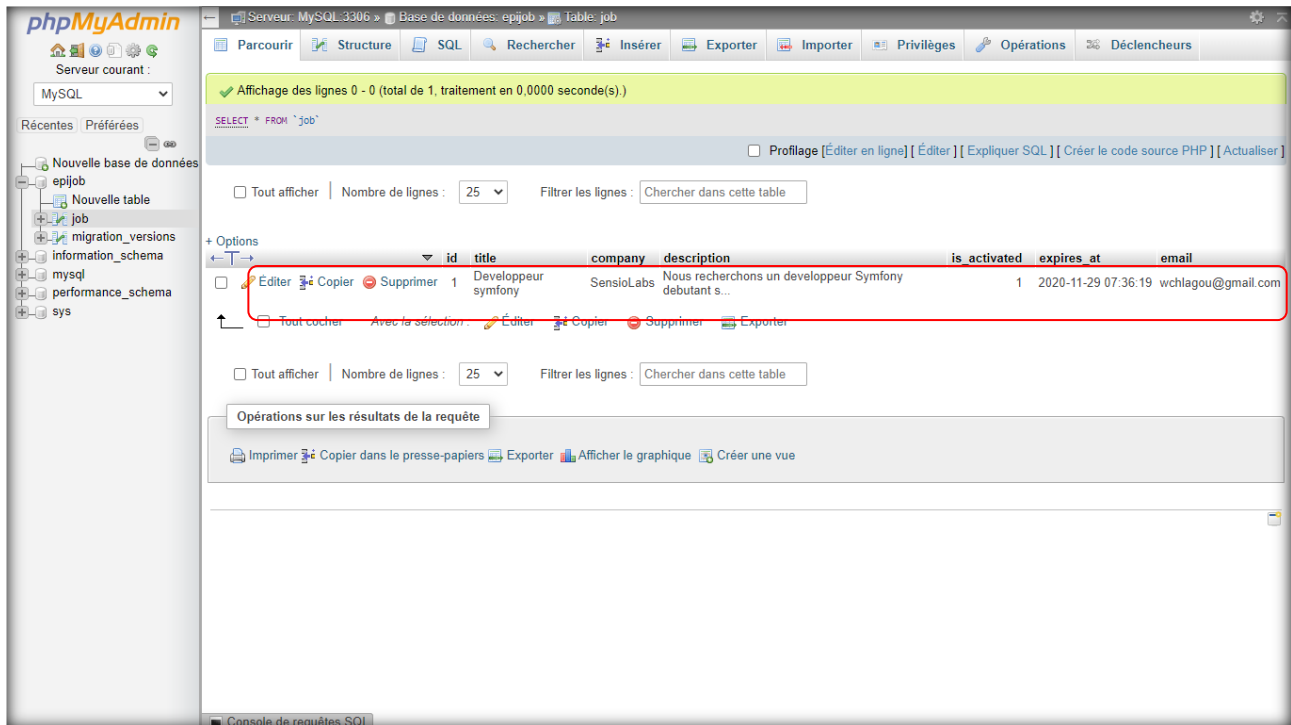


```
86
87
88  /**
89   * @Route("/ajouter", name="ajouter")
90   */
91  public function ajouter()
92  {
93      $job = new job();
94      $job->setTitle('Developpeur symfony');
95      $job->setCompany('Sensiolabs');
96      $job->setDescription("Nous recherchons un developpeur Symfony
97      debutant sur Sousse.");
98      $job->setIsActive(1);
99      $job->setExpiresAt(new \DateTime());
100     $job->setEmail("wchlagou@gmail.com");
101
102     $em = $this->getDoctrine()->getManager();
103     $em->persist($job);
104     $em->flush();
105
106     return $this->render('job/ajouter.html.twig', ['job' => $job]);
107
108 }
```

Sans oublier de rajouter le use suivant dans le contrôleur !

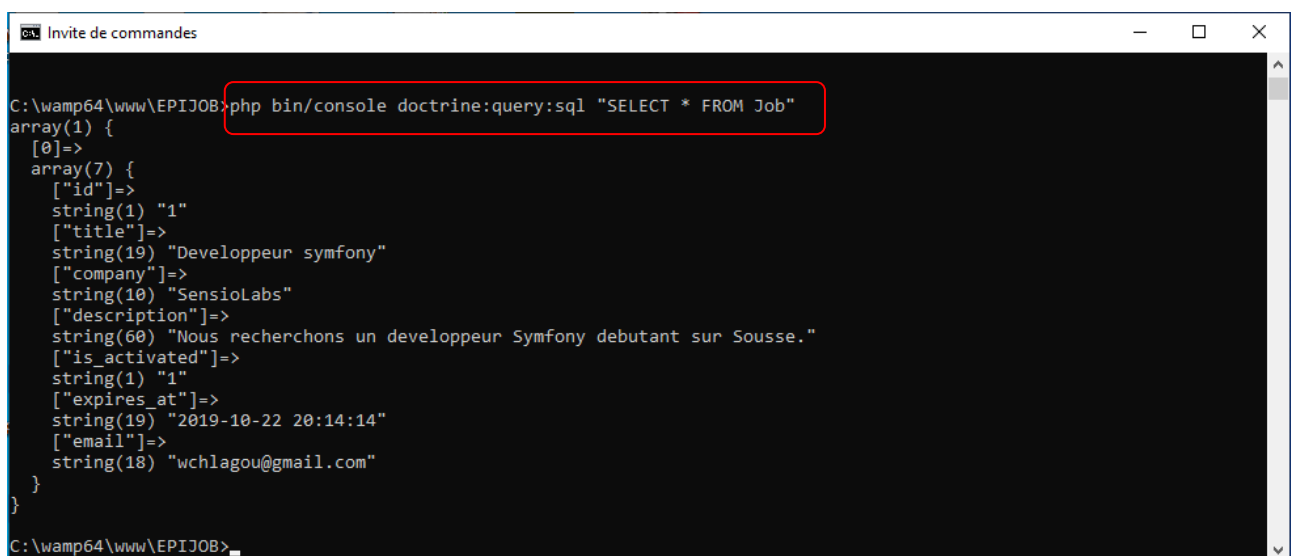
```
use App\Entity\Job;
```

- ❖ `$em->persist()` : Garde cette entité en mémoire, tu l'enregistreras au prochain `flush()`.
- ❖ `$em->flush()` : Ouvre une transaction et enregistre toutes les entités qui t'ont été données depuis le dernier `flush()`.
- ✓ Grace à EM, On n'a plus besoin de faire des requêtes à la main (ni INSERT INTO ni UPDATE)

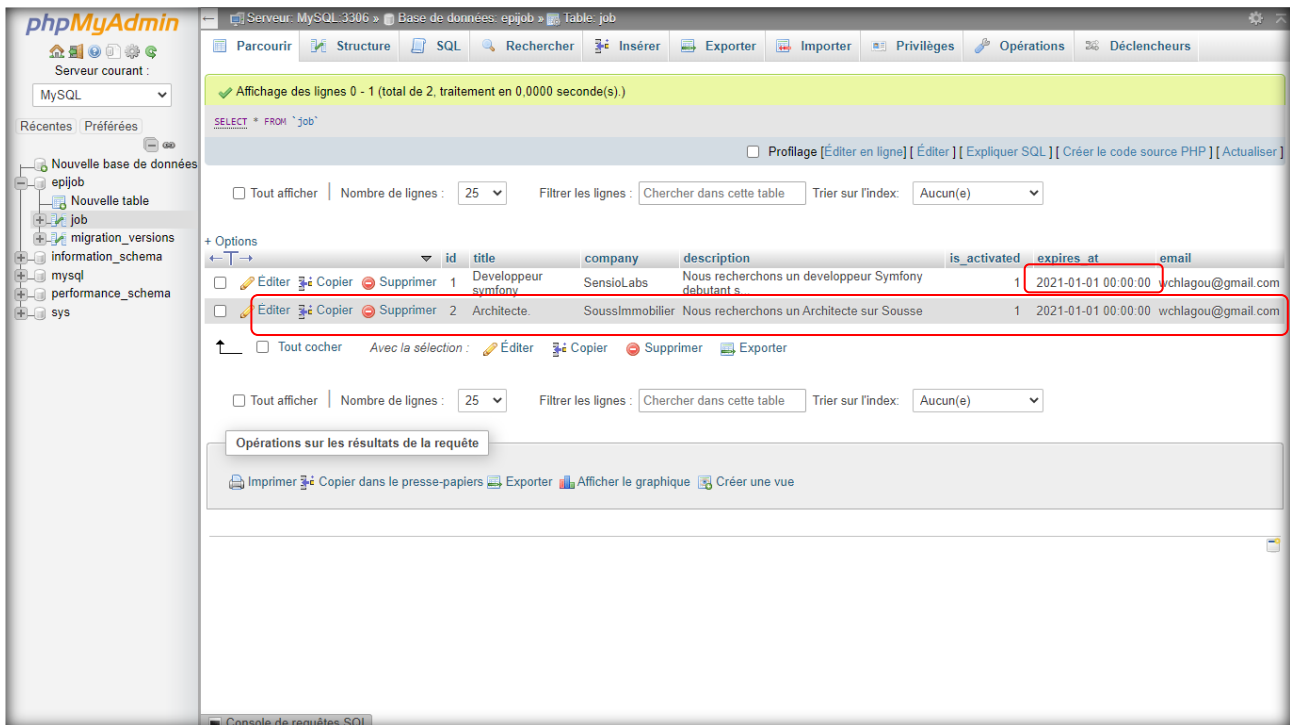


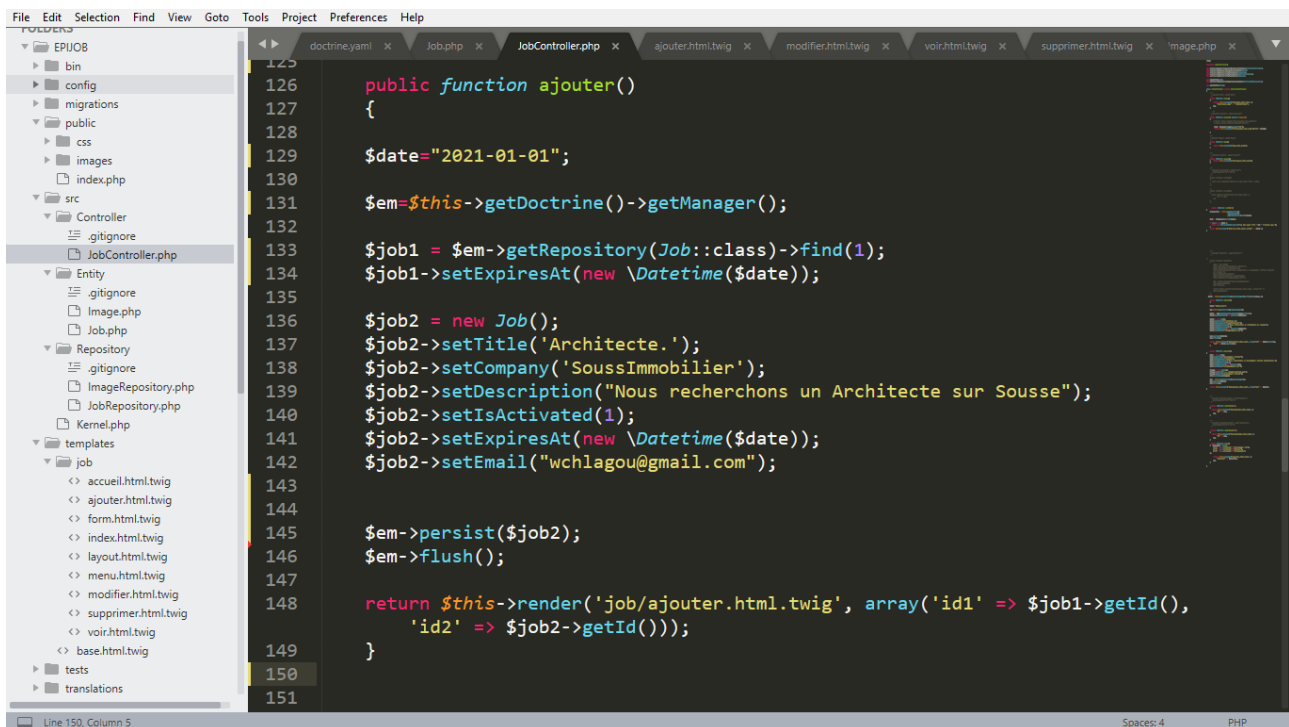
Récupérer toutes les données depuis la BD :

```
php bin/console doctrine:query:sql "SELECT * FROM Job" >
```



- On se propose de modifier code actuel de la fonction ajouter afin qu'on puisse à la fois :
- Modifier la date d'expiration du job déjà créé
 - Créer un nouveau job





```

125
126 public function ajouter()
127 {
128
129     $date="2021-01-01";
130
131     $em=$this->getDoctrine()->getManager();
132
133     $job1 = $em->getRepository(Job::class)->find(1);
134     $job1->setExpiresAt(new \DateTime($date));
135
136     $job2 = new Job();
137     $job2->setTitle('Architecte. ');
138     $job2->setCompany('SoussImmobilier');
139     $job2->setDescription("Nous recherchons un Architecte sur Souss");
140     $job2->setIsActivated(1);
141     $job2->setExpiresAt(new \DateTime($date));
142     $job2->setEmail("wchlagou@gmail.com");
143
144
145     $em->persist($job2);
146     $em->flush();
147
148     return $this->render('job/ajouter.html.twig', array('id1' => $job1->getId(),
149         'id2' => $job2->getId()));
150 }
151

```

- ✓ Une autre syntaxe pour rechercher le job via son id : la méthode find() de l'EntityManager, et non pas du Repository

```
$job1 = $this->getDoctrine()->getManager()->find(job::class,1);
```

- ✓ Pourquoi on n'a pas fait un persist() sur \$job1 ?

En effet, puisque ce job a été récupéré via Doctrine précédemment, il sait déjà qu'il doit gérer cette entité :

Un persist ne sert qu'à donner la responsabilité de l'objet à Doctrine.


➤ Attribut par défaut

Parfois, on a besoin de définir une certaine valeur à notre entité lors de sa création, par exemple si on veut fixer la date d'expiration des job à 2021, dans le fichier job.php:

```
public function __construct()  
{  
    $this->date = new \Datetime(2020);  
}
```

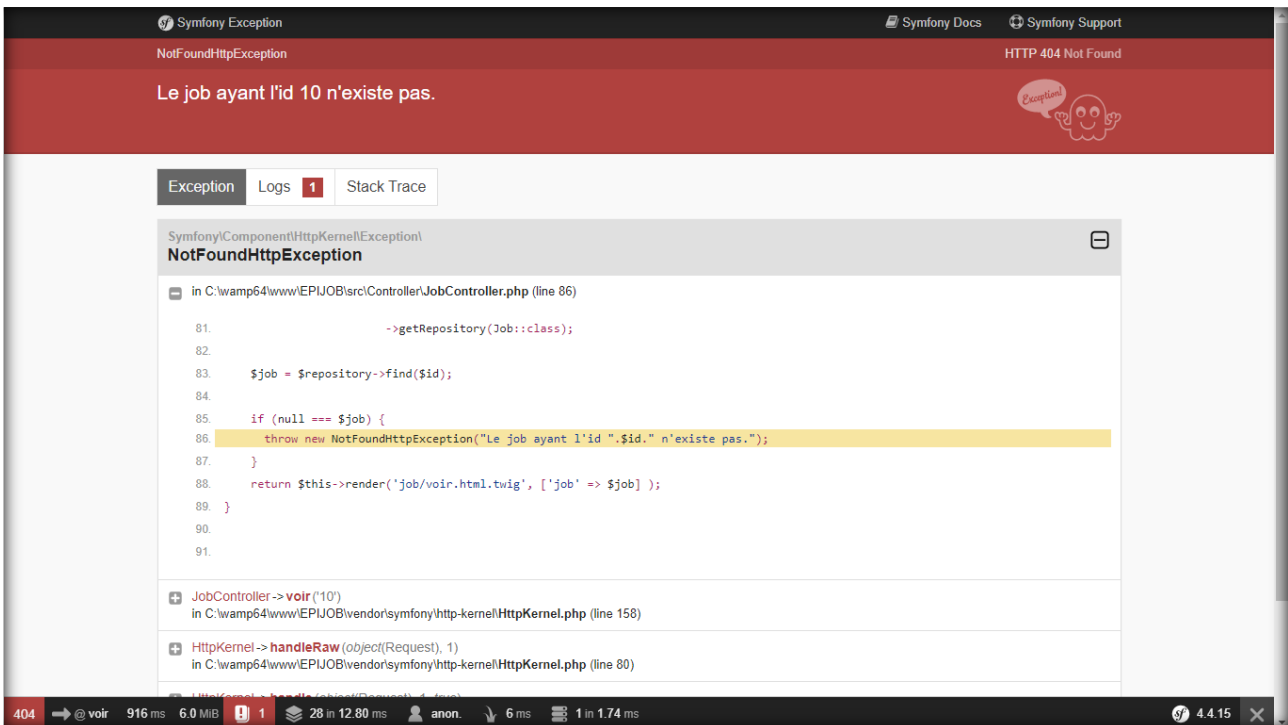
➤ **Exception NotFound :**

On se propose de personnaliser le rendu de la méthode voir(),
Si l'id appelé dans la route existe dans la table job, on aura l'affichage suivant :



The screenshot shows a web application interface. On the left, there is a sidebar with three links: "Developpeur web", "Docteur", and "Journaliste". The main content area displays the following information in red text: "BIENVENU À EPIJOB -JOB -Voir un job", "Bonjour", and "Détails du job ayant l'id: 1". Below this, in black text, are the details: "Intitulé: Developpeur symfony", "Description: Nous recherchons un developpeur Symfony debutant sur Sousse.", and "Société: SensioLabs". At the bottom of the page, there is a status bar with various icons and text: "200 @ voir 779 ms 2.0 MiB 26 in 8.42 ms anon. 113 ms 1 in 1.82 ms 4.4.15".

Dans le cas contraire on aura l'affichage suivant :



Symfony Exception

Not FoundHttpException

HTTP 404 Not Found

Le job ayant l'id 10 n'existe pas.

Exception Logs 1 Stack Trace

Symfony\Component\HttpFoundation\Exception
NotFoundHttpException

in C:\wamp64\www\EPIJOB\src\Controller\JobController.php (line 86)

```

81.             ->getRepository(Job::class);
82.
83.     $job = $repository->find($id);
84.
85.     if (null === $job) {
86.         throw new NotFoundHttpException("Le job ayant l'id ".$id." n'existe pas.");
87.     }
88.     return $this->render('job/voir.html.twig', ['job' => $job]);
89. }
90.
91.

```

JobController->voir("10")
in C:\wamp64\www\EPIJOB\vendor\symfony\http-kernel\HttpKernel.php (line 158)

HttpKernel->handleRaw(object(Request), 1)
in C:\wamp64\www\EPIJOB\vendor\symfony\http-kernel\HttpKernel.php (line 80)

404 → @ voir 916 ms 6.0 MiB 1 28 in 12.80 ms anon. 6 ms 1 in 1.74 ms 4.4.15

- Compléter le code de la méthode voir() permettant de faire ceci:

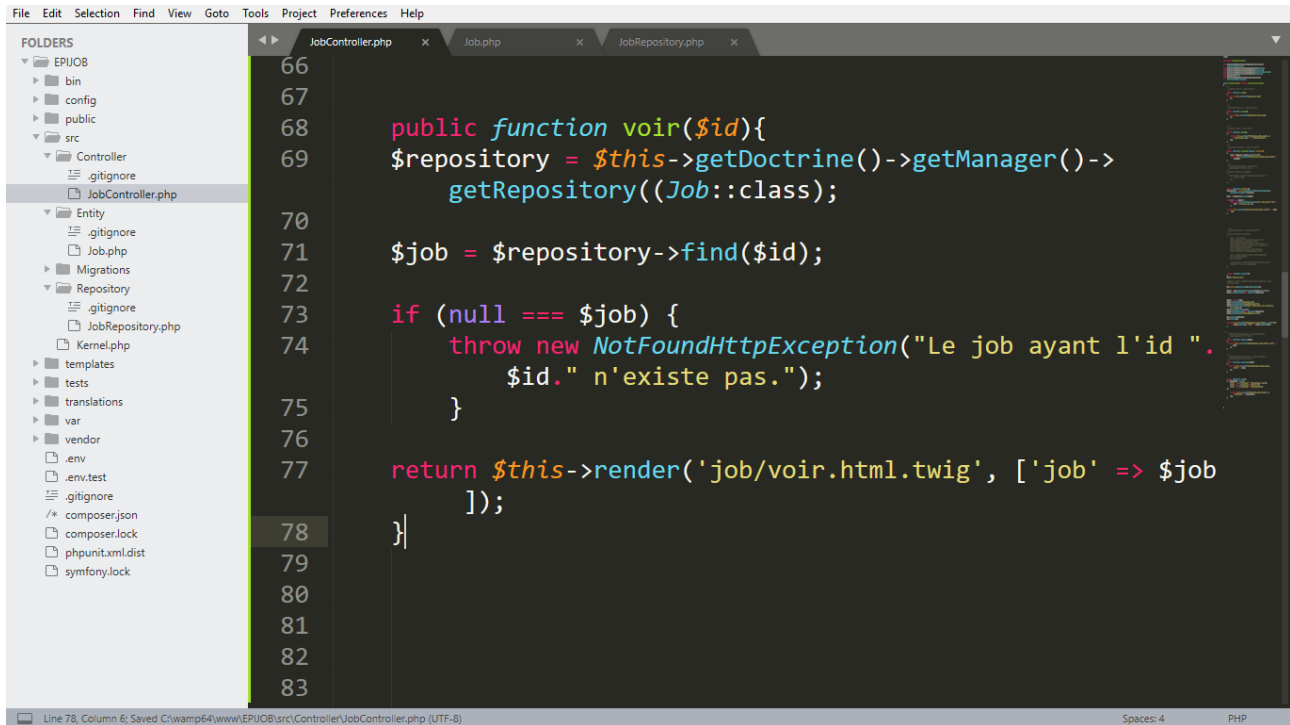
```

public function voir($id){
    $repository = $this->getDoctrine()->getManager()->getRepository(.....);
    $job = .....;
    if (.....) {
        throw new NotFoundHttpException("Le job ayant l'id ".$id."
        n'existe pas.");
    }
    return $this->render(....., [
        'job' => $job,
    ]);
}

```

- Veuillez modifier également le template voir.html.twig pour avoir le rendu cité précédemment.

- La méthode voir



```
66
67
68 public function voir($id){
69     $repository = $this->getDoctrine()->getManager()->
        getRepository((Job::class));
70
71     $job = $repository->find($id);
72
73     if (null === $job) {
74         throw new NotFoundException("Le job ayant l'id ".
            $id." n'existe pas.");
75     }
76
77     return $this->render('job/voir.html.twig', ['job' => $job
        ]);
78 }
79
80
81
82
83
```

Sans oublier de faire appel à NotFoundException!

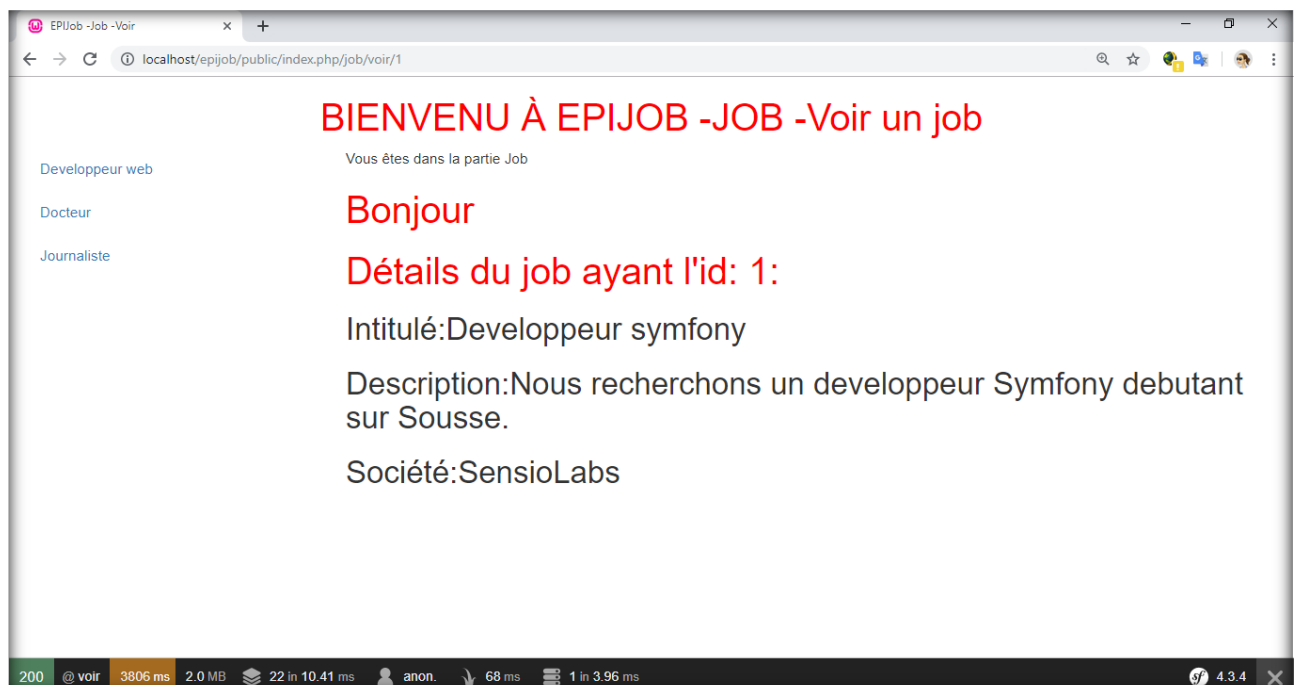
```
use Symfony\Component\HttpFoundation\Exception\NotFoundException;
```

- Voir.html.twig

```

File Edit Selection Find View Goto Tools Project Preferences Help
1 {% extends "job/layout.html.twig" %}
2 {% block title %}
3     {{parent()}} -Voir
4 {% endblock %}
5
6 {% block stylesheets %}
7     {{parent()}}
8     <link rel="stylesheet" type="text/css"
9         href="{{asset('css/supprimer.css')}}">
10 {% endblock %}
11
12
13 {% block titre %}
14     {{parent()|upper}} -Voir un job
15 {% endblock %}
16
17
18 {% block job_body %}
19     <h1>Bonjour</h1>
20     <h1>Détails du job ayant l'id: {{job.id}}:</h1>
21     <h2>Intitulé:{{job.title}}</h2>
22     <h2>Description:{{job.description}}</h2>
23     <h2>Société:{{job.company}}</h2>
24 {% endblock %}

```



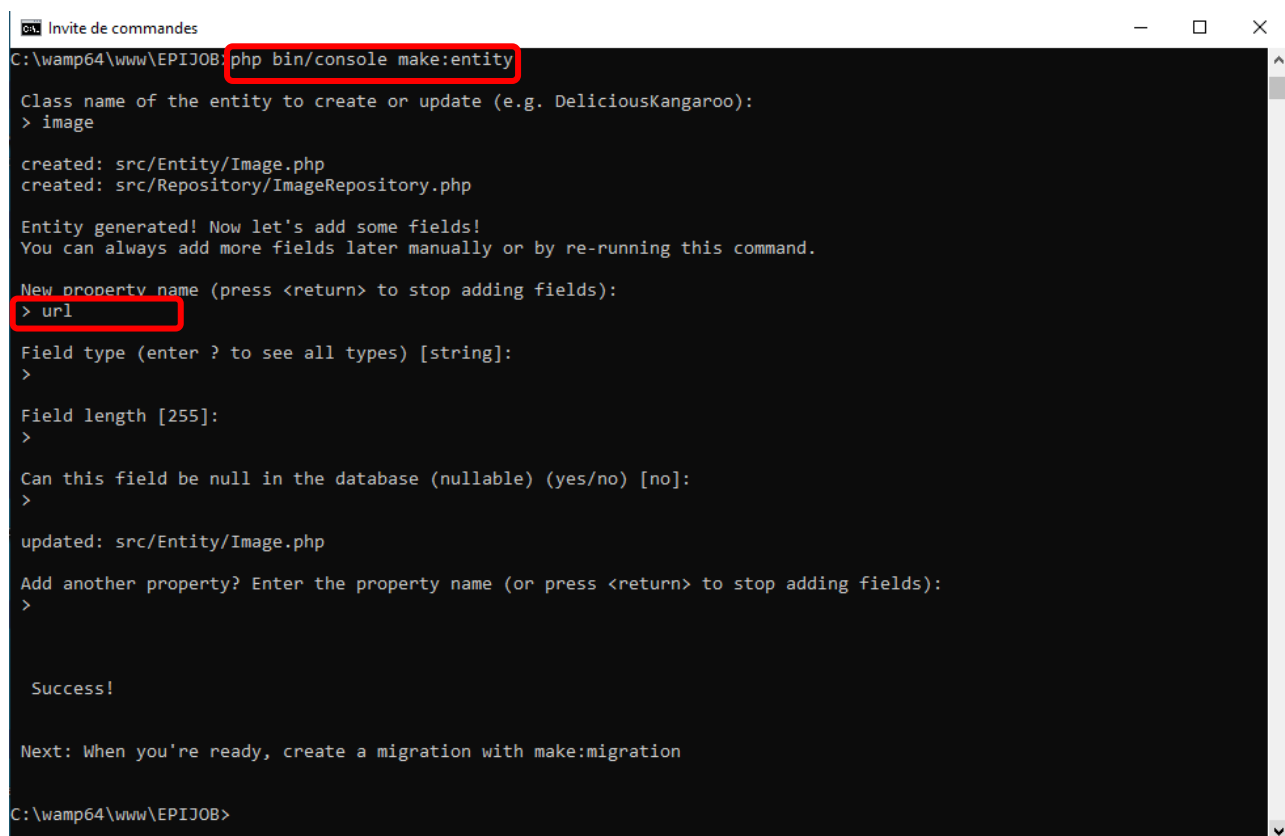
Partie 4 : Relations entre les entités

1. Relation One To One

- ✓ Entités de la relation : Job et Image

- Créer l'entité Image :

```
url:
    type: string
    length: 255
alt:
    type: string
    length: 255
```



```
Invite de commandes
C:\wamp64\www\EPIJOB> php bin/console make:entity
Class name of the entity to create or update (e.g. DeliciousKangaroo):
> image
created: src/Entity/Image.php
created: src/Repository/ImageRepository.php
Entity generated! Now let's add some fields!
You can always add more fields later manually or by re-running this command.
New property name (press <return> to stop adding fields):
> url
Field type (enter ? to see all types) [string]:
>
Field length [255]:
>
Can this field be null in the database (nullable) (yes/no) [no]:
>
updated: src/Entity/Image.php
Add another property? Enter the property name (or press <return> to stop adding fields):
>
Success!
Next: When you're ready, create a migration with make:migration
C:\wamp64\www\EPIJOB>
```

- Rajouter l'attribut « alt »

```

C:\wamp64\www\EPIJOB>php bin/console make:entity
Class name of the entity to create or update (e.g. DeliciousGnome):
> image

Your entity already exists! So let's add some new fields!
New property name (press <return> to stop adding fields):
> alt

Field type (enter ? to see all types) [string]:
>

Field length [255]:
>

Can this field be null in the database (nullable) (yes/no) [no]:
>

updated: src/Entity/Image.php

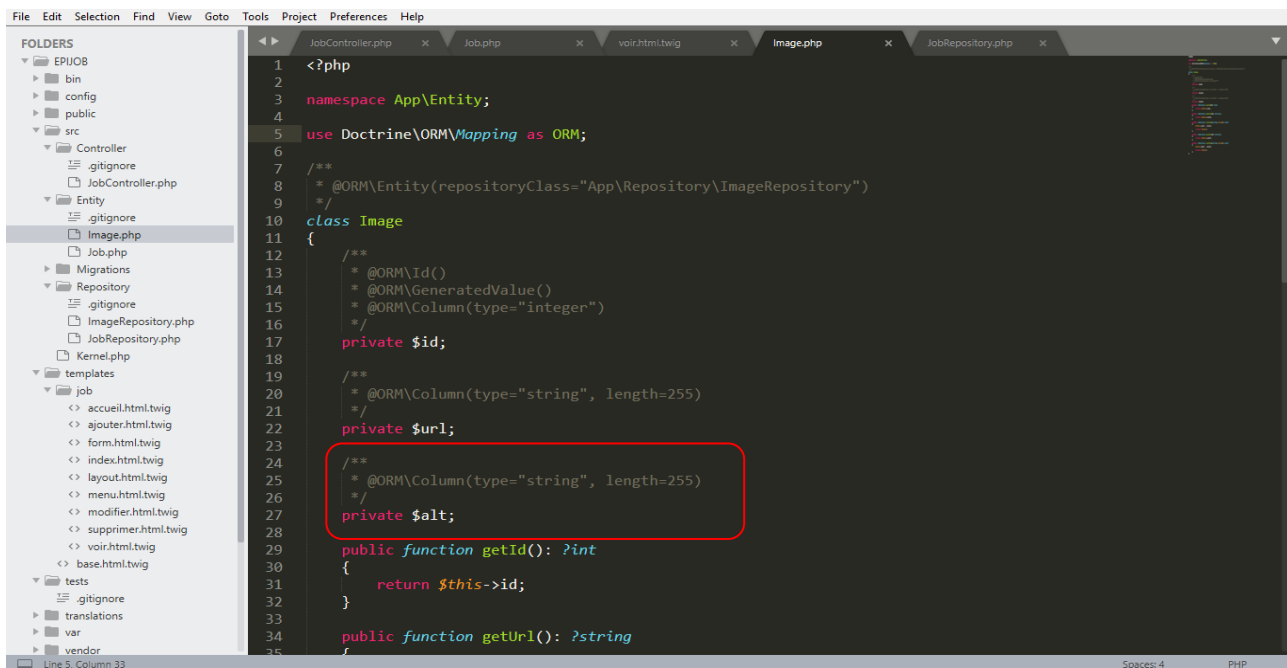
Add another property? Enter the property name (or press <return> to stop adding fields):
>

Success!

Next: When you're ready, create a migration with make:migration

C:\wamp64\www\EPIJOB>

```



```

1  <?php
2
3  namespace App\Entity;
4
5  use Doctrine\ORM\Mapping as ORM;
6
7  /**
8   * @ORM\Entity(repositoryClass="App\Repository\ImageRepository")
9   */
10 class Image
11 {
12     /**
13      * @ORM\Id()
14      * @ORM\GeneratedValue()
15      * @ORM\Column(type="integer")
16      */
17     private $id;
18
19     /**
20      * @ORM\Column(type="string", length=255)
21      */
22     private $url;
23
24     /**
25      * @ORM\Column(type="string", length=255)
26      */
27     private $alt;
28
29     public function getId(): ?int
30     {
31         return $this->id;
32     }
33
34     public function getUrl(): ?string
35

```

- Enregistrer la nouvelle entité en Base de données :

```
php bin/console make:migration
```

```
Invite de commandes

C:\wamp64\www\EPIJOB>php bin/console make:migration

Success!

Next: Review the new migration "src/Migrations/Version20191023033559.php"
Then: Run the migration with php bin/console doctrine:migrations:migrate
See https://symfony.com/doc/current/bundles/DoctrineMigrationsBundle/index.html

C:\wamp64\www\EPIJOB>
```

```
php bin/console doctrine:migrations:migrate
```

```
Invite de commandes

C:\wamp64\www\EPIJOB>php bin/console doctrine:migrations:migrate

Application Migrations

WARNING! You are about to execute a database migration that could result in schema changes and data loss. Are you sure you wish to continue? (y/n)y
Migrating up to 20191023033559 from 20191022194333

++ migrating 20191023033559

--> CREATE TABLE image (id INT AUTO_INCREMENT NOT NULL, url VARCHAR(255) NOT NULL, alt VARCHAR(255) NOT NULL, PRIMARY KEY(id)) DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci ENGINE = InnoDB

++ migrated (took 3673.7ms, used 16M memory)

-----

++ finished in 3829.4ms
++ used 16M memory
++ 1 migrations executed
++ 1 sql queries

C:\wamp64\www\EPIJOB>
```

➤ Définir la relation entre les 2 entités :

On peut créer une relation de deux façons

- Ecrire l'annotation de la relation directement dans job.php
- Utiliser les commandes dans la console.

```
C:\wamp64\www\Symfony-project4>php bin/console make:entity
Class name of the entity to create or update (e.g. GrumpyPuppy):
> job
Your entity already exists! So let's add some new fields!
New property name (press <return> to stop adding fields):
> image
Field type (enter ? to see all types) [string]:
> relation
What class should this entity be related to?
> Image
What type of relationship is this?
Type      Description
-----
ManyToOne Each Job relates to (has) one Image.
           Each Image can relate/has to (have) many Job objects
OneToMany Each Job relates can relate to (have) many Image objects.
           Each Image relates to (has) one Job
ManyToMany Each Job relates can relate to (have) many Image objects.
           Each Image can also relate to (have) many Job objects
OneToOne  Each Job relates to (has) exactly one Image.
           Each Image also relates to (has) exactly one Job.
Relation type? [ManyToOne, OneToMany, ManyToMany, OneToOne]:
> OneToOne
Is the Job image property allowed to be null (nullable)? (yes/no) [yes]:
> yes
Do you want to add a new property to Image so that you can access/update Job objects from it - e.g. $image->getJob()? (yes/no) [no]:
> yes
A new property will also be added to the Image class so that you can access the related Job object from it.
New field name inside Image [job]:
> job
updated: src/Entity/Job.php
```

Image et non pas image :
nom de l'entité

```
Invite de commandes

updated: src/Entity/Job.php
updated: src/Entity/Image.php

Add another property? Enter the property name (or press <return> to stop adding fields):
>

Success!

Next: When you're ready, create a migration with make:migration

C:\wamp64\www\EPIJOB>
```

Ne pas oublier d'appliquer

`bin/console make:migration`

`php bin/console doctrine:migrations:migrate`

```
cat Sélection Invite de commandes
C:\wamp64\www\EPIJOB>php bin/console make:migration

Success!

Next: Review the new migration "src/Migrations/Version20191023035413.php"
Then: Run the migration with php bin/console doctrine:migrations:migrate
See https://symfony.com/doc/current/bundles/DoctrineMigrationsBundle/index.html

C:\wamp64\www\EPIJOB>php bin/console doctrine:migrations:migrate

Application Migrations

WARNING! You are about to execute a database migration that could result in schema changes and data loss. Are you sure y
ou wish to continue? (y/n)y
Migrating up to 20191023035413 from 20191023033559

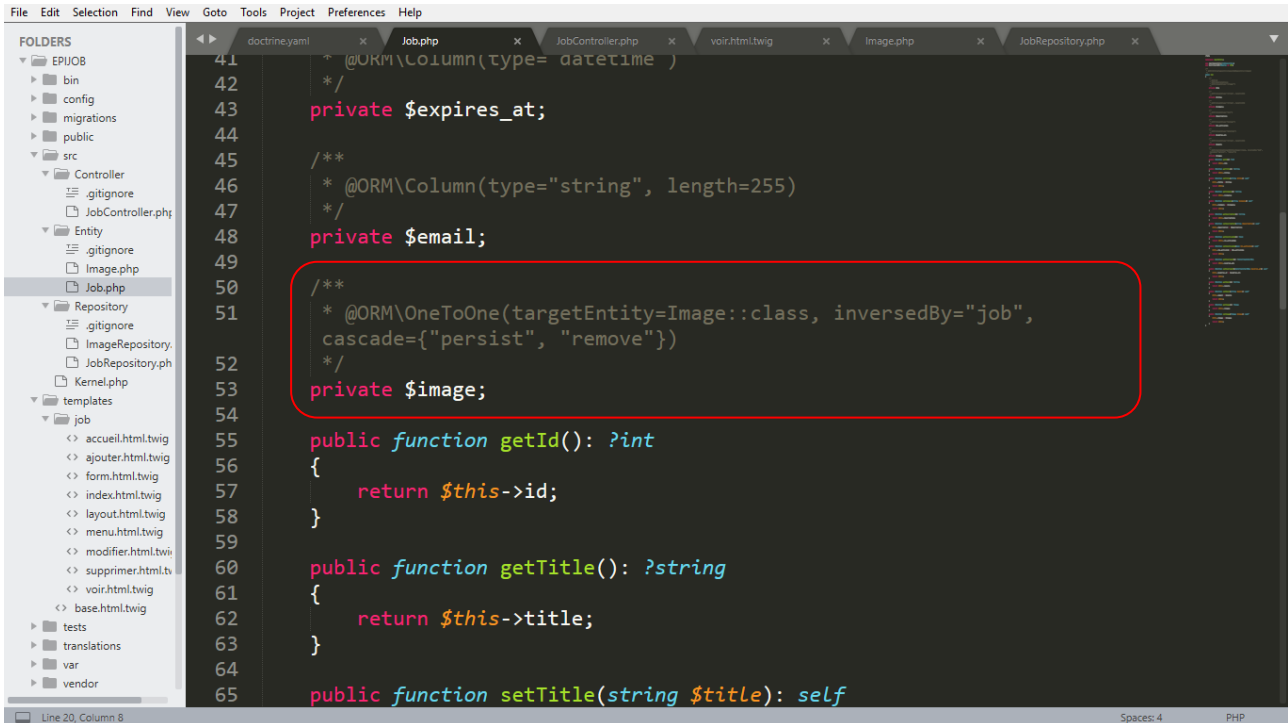
++ migrating 20191023035413
  -> ALTER TABLE job ADD image_id INT DEFAULT NULL
  -> ALTER TABLE job ADD CONSTRAINT FK_FBD8E0F83DA5256D FOREIGN KEY (image_id) REFERENCES image (id)
  -> CREATE UNIQUE INDEX UNIQ_FBD8E0F83DA5256D ON job (image_id)

++ migrated (took 2632.2ms, used 16M memory)

-----

++ finished in 2638.3ms
++ used 16M memory
++ 1 migrations executed
++ 3 sql queries

C:\wamp64\www\EPIJOB>
```

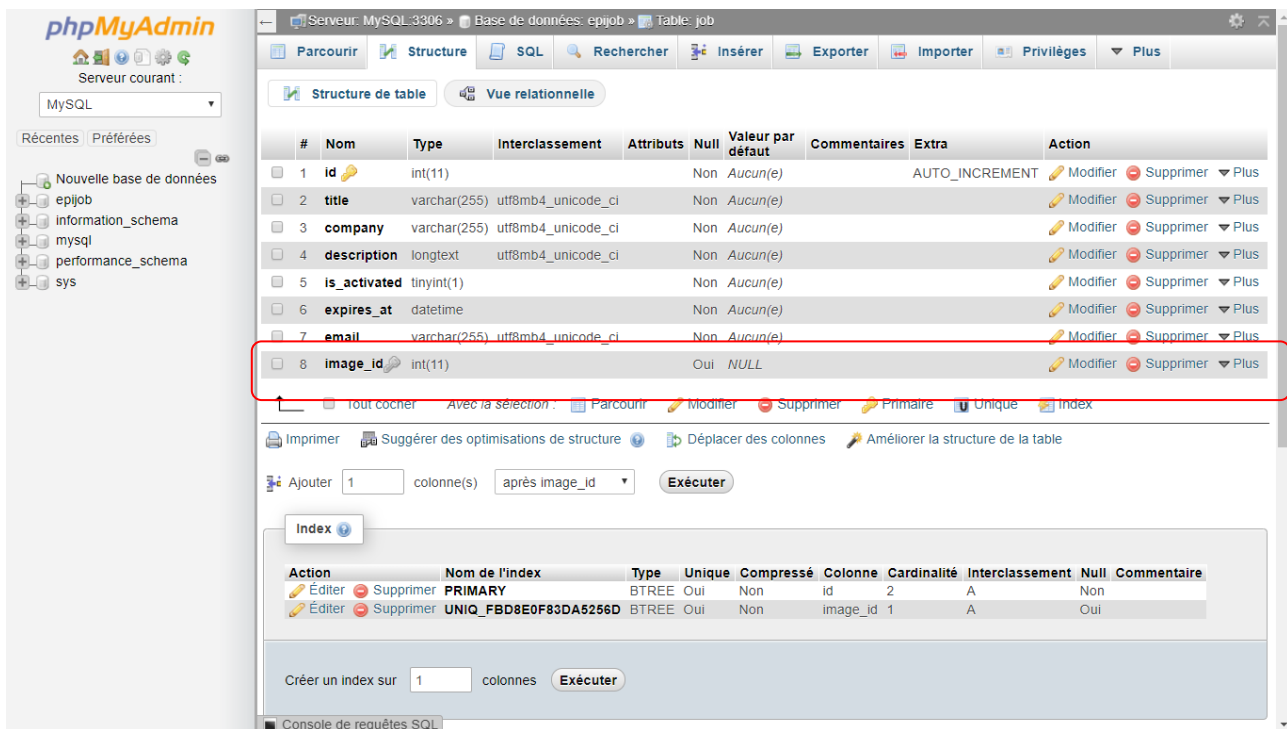


```

41  * @ORM\Column(type= datetime )
42  */
43  private $expires_at;
44
45  /**
46   * @ORM\Column(type="string", length=255)
47   */
48  private $email;
49
50  /**
51   * @ORM\OneToOne(targetEntity=Image::class, inversedBy="job",
52   cascade={"persist", "remove"})
53   */
54  private $image;
55
56  public function getId(): ?int
57  {
58      return $this->id;
59  }
60
61  public function getTitle(): ?string
62  {
63      return $this->title;
64  }
65
66  public function setTitle(string $title): self

```

- ✓ OneToOne et non pas Column ici on définit une relation vers une autre entité et non pas un attribut de notre entité
- ✓ targetEntity : le namespace complet vers l'entité liée.
- ✓ cascade={"persist"} : opérations en cascade : un persist() sur job doit se «propager» sur l' image liée.



✓ Une colonne `image_id` a bien été ajoutée à la table `job`.
Cependant, ne confondez pas cette colonne `image_id` avec notre attribut `image`.

L'objet `job` ne contient pas d'attribut `image_id`.
L'objet `job` contient un attribut `image`.

L'attribut `image` de l'objet `job` ne contient pas l'id de l'image liée,
L'attribut `image` de l'objet `job` contient une instance de la classe `App\Entity\image`.
Cette dernière (instance de la classe `image`), elle, contient un attribut `id`:

➡ Une entité n'est pas une table.

```
$job->getImageId : Incorrect
$job->getImage()->getId() :Correct
```

- Créer un job avec une image

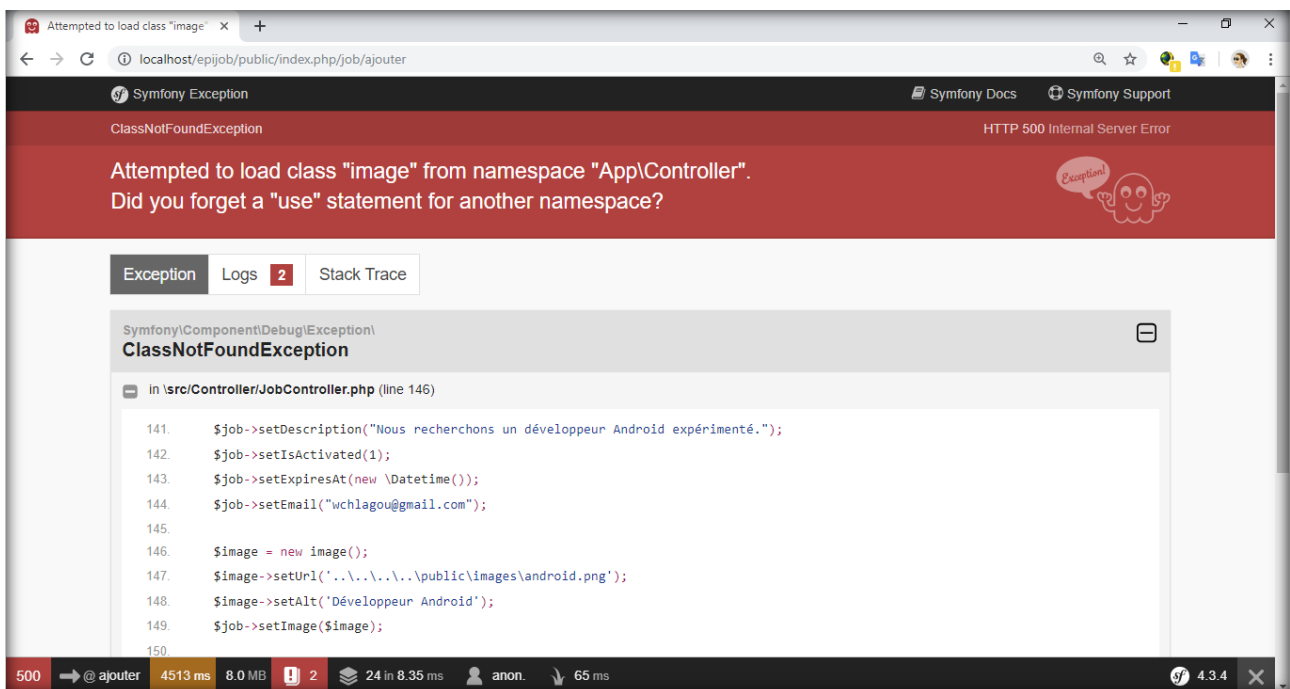
```

153 public function ajouter()
154 {
155     $job = new job();
156     $job->setTitle('Développeur Android');
157     $job->setCompany('sumsung');
158     $job->setDescription("Nous recherchons un développeur Android
159 expérimenté.");
160     $job->setIsActivated(1);
161     $job->setExpiresAt(new \Datetime());
162     $job->setEmail("wchlagou@gmail.com");
163     $image = new image();
164     $image->setUrl('../..../images/android.png');
165     $image->setAlt('Développeur Android');
166     $job->setImage($image);
167
168     $em = $this->getDoctrine()->getManager();
169     $em->persist($job);
170     $em->flush();
171
172     return $this->render('job/ajouter.html.twig', array('job' => $job));
173 }

```

Ne pas oublier de rajouter le « use » de l'image

```
use App\Entity\Image;
```



Case mismatch between loaded and declared class names: "App\Entity\image" vs "App\Entity\Image".

Symfony Exception

RuntimeException

HTTP 500 Internal Server Error

Exception Logs 1 Stack Trace

RuntimeException

```

in vendor/symfony/debug/DebugClassLoader.php (line 183)
178.         return;
179.     }
180.     $name = $refl->getName();
181.
182.     if ($name !== $class && 0 === \strcasecmp($name, $class)) {
183.         throw new \RuntimeException(sprintf('Case mismatch between loaded and declared class names: "%s" vs "%s".', $class,
184.     }

```

500 @ajouter 1609 ms 6.0 MB 1 115 in 508.16 ms anon. 100 ms sf 4.1.4

phpMyAdmin

Serveur: MySQL 3306 » Base de données: epjob » Table: job

Parcourir Structure SQL Rechercher Insérer Exporter Importer Privileges Opérations Déclencheurs

Affichage des lignes 0 - 2 (total de 3, traitement en 0,0000 seconde(s).)

SELECT * FROM `job`

Profilage [Éditer en ligne] [Éditer] [Expliquer SQL] [Créer le code source PHP] [Actualiser]

Tout afficher Nombre de lignes : 25 Filtrer les lignes : Chercher dans cette table Trier sur l'index : Aucun(e)

		id	title	company	description	is_activated	expires_at	email	image_id
<input type="checkbox"/>	Éditer Copier Supprimer	1	Developpeur symfony	SensioLabs	Nous recherchons un developpeur Symfony debutant s...	1	2020-01-01 00:00:00	wchlagou@gmail.com	NULL
<input type="checkbox"/>	Éditer Copier Supprimer	2	Architecte	SoussImmobilier	Nous recherchons un Architecte sur Soussse	1	2020-01-01 00:00:00	wchlagou@gmail.com	NULL
<input type="checkbox"/>	Éditer Copier Supprimer	3	Développeur Android	samsung	Nous recherchons un développeur Android expériment...	1	2019-10-23 04:06:52	wchlagou@gmail.com	1

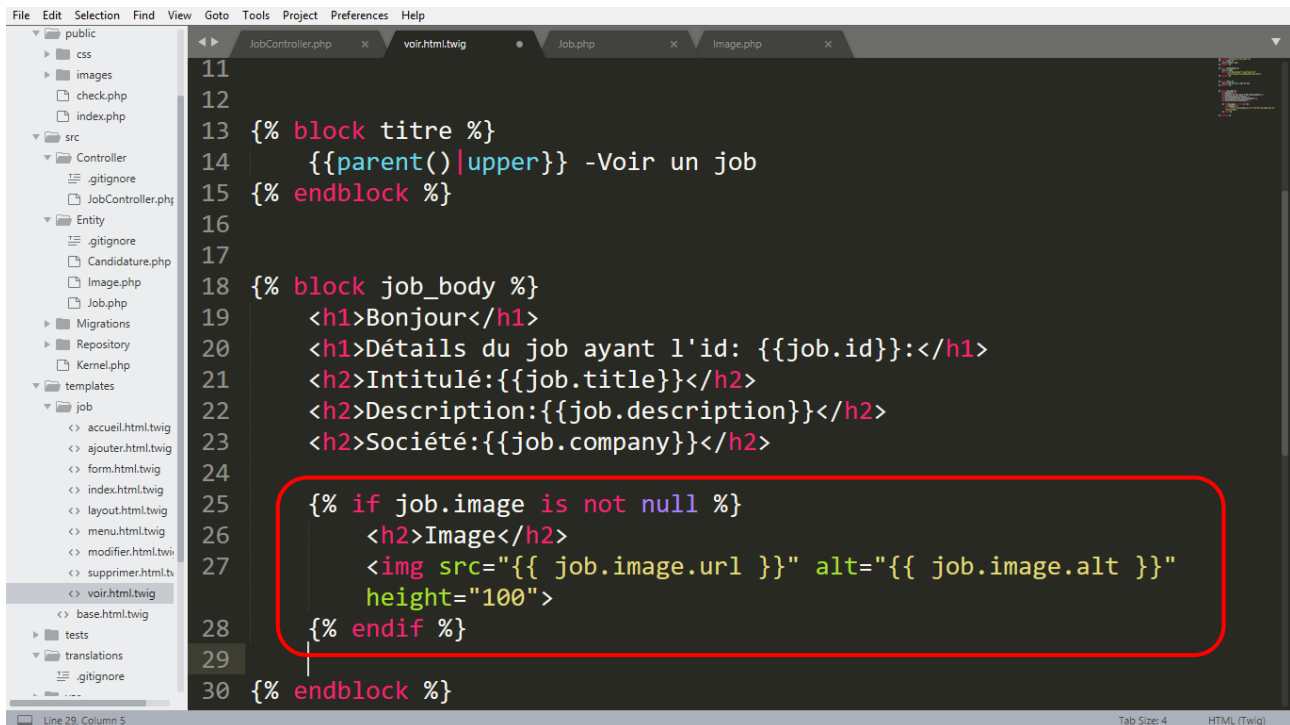
Tout cocher Avec la sélection : Éditer Copier Supprimer Exporter

Opérations sur les résultats de la requête

Imprimer Copier dans le presse-papiers Exporter Afficher le graphique Créer une vue

Console de requêtes SQL

- Adapter le template «voir.html.twig» pour afficher l'image du job



```

11
12
13 {% block titre %}
14     {{parent()|upper}} -Voir un job
15 {% endblock %}
16
17
18 {% block job_body %}
19     <h1>Bonjour</h1>
20     <h1>Détails du job ayant l'id: {{job.id}}:</h1>
21     <h2>Intitulé:{{job.title}}</h2>
22     <h2>Description:{{job.description}}</h2>
23     <h2>Société:{{job.company}}</h2>
24
25     {% if job.image is not null %}
26         <h2>Image</h2>
27         
29     {% endif %}
30 {% endblock %}

```

