**PHP for Exam**

**1.Отивам в папката където съм разархивирал проекта(пътят за php трябва да е къс,защото немогат да се инсталират всички файлове) отварям си 1 конзола(Shift+right click) и пиша:**

**php composer.phar install(чакам да се се свялят всички файлове,през това време си пускам и PHPStorm да зарежда проекта).Отварям и XAMPP+MySQL+HeidiSQL.Търся файла app/config/parameters.yml.В него се намира името на базата с данни-сменявам го с име\_php(zaradi JAVA).Връщам се пак в конзолата и пиша:**

**php bin/console doctrine:database:create --if-not-exists за да си създам база с данни.**

**2.В PHPStorm изтривам от папка Entity/файла и от папка Repository/файла.Давам save и се връщам в конзолата: php bin/console doctrine:generate:entity**

**След това давам име на ентитито,което трябва да съдържа името на Бъндала:AppBundle:Film**

**И след това попълвам данните за колоните на таблицата.Като съм готов натискам enter.**

**След това трябва да вкарам новите колони в таблицата:**

**php bin/console doctrine:schema:update --force**

**Когато стартирам проекта да проверя в Edit Configuration дали няма нещо излишно,за да неможе да ми се стартира проекта!!!**

**. Вече всичко е генерирано автоматично,но трябва да отида в WebStorm entity/ime fail най-отгоре да направя детайлите да са в мн.число(генерират се автоматично в единствено):**

*\* Film  
\*  
\** ***@ORM\Table****(name="films")  
\** ***@ORM\Entity****(repositoryClass="AppBundle\Repository\FilmRepository")*

**Правя и метода за Index:**

**public function** index(Request $request)  
{  
 $repo = $this->getDoctrine()->getRepository(Film::***class***);  
 $films = $repo->findAll();  
 **return** $this->render(  
 **":film:index.html.twig"**,  
 [**"films"** => $films]  
 );  
}

**И тествам дали всичко работи.**

**3.Отивам в папка Form/ime na fail и си построявам билдър:**

**public function** buildForm(FormBuilderInterface $builder, **array** $options)  
{  
 $builder  
 ->add(**'name'**, TextType::***class***)  
 ->add(**'genre'**, TextType::***class***)  
 ->add(**'director'**, TextType::***class***)  
 ->add(**'year'**, IntegerType::***class***);  
  
}

**Ако не ми се показват нещата след ръчно въвеждане на продукт през базата данни,правя и метода Create за да видя правилната база с данни.**

**4.Всички методи от Наков:**

*/\*\*  
 \** ***@param*** *Request $request  
 \** ***@Route****("/create", name="create")  
 \** ***@return*** *\Symfony\Component\HttpFoundation\Response  
 \*/***public function** create(Request $request)  
{  
 $film = **new** Film();  
 $form = $this->createForm(FilmType::***class***, $film);  
 $form->handleRequest($request);  
 $errorMsg = **""**;  
 **if** ($form->isSubmitted()) {  
 **if** ($form->isValid()) {  
 $em = $this->getDoctrine()->getManager();  
 $em->persist($film);  
 $em->flush();  
 **return** $this->redirect(**"/"**);  
 } **else** $errorMsg = **"Invalid form data"**;  
 }  
 **return** $this->render(  
 **":film:create.html.twig"**,  
 [**"film"** => $film, **"form"** => $form->createView(),  
 **"errorMsg"** => $errorMsg]  
 );  
}  
*/\*\*  
 \** ***@Route****("/edit/{id}", name="edit")  
 \*  
 \** ***@param*** *$id  
 \** ***@param*** *Request $request  
 \** ***@return*** *\Symfony\Component\HttpFoundation\Response  
 \*/***public function** edit($id, Request $request)  
{  
 $filmRepo = $this->getDoctrine()->getRepository(Film::***class***);  
 $film = $filmRepo->find($id);  
 **if** ($film == **null**) {  
 **return** $this->redirect(**"/"**);  
 }  
 $form = $this->createForm(FilmType::***class***, $film);  
 $form->handleRequest($request);  
 $errorMsg = **""**;  
 **if** ($form->isSubmitted()) {  
 **if** ($form->isValid()) {  
 $em = $this->getDoctrine()->getManager();  
 $em->merge($film);  
 $em->flush();  
 **return** $this->redirect(**"/"**);  
 } **else** $errorMsg = **"Invalid form data"**;  
 }  
 **return** $this->render(  
 **":film:edit.html.twig"**,  
 [**"film"** => $film, **"form"** => $form->createView(),  
 **"errorMsg"** => $errorMsg]  
 );

}  
*/\*\*  
 \** ***@Route****("/delete/{id}", name="delete")  
 \*  
 \** ***@param*** *$id  
 \** ***@param*** *Request $request  
 \** ***@return*** *\Symfony\Component\HttpFoundation\Response  
 \*/***public function** delete($id, Request $request)  
{  
 $filmRepo = $this->getDoctrine()->getRepository(Film::***class***);  
 $film = $filmRepo->find($id);  
 **if** ($film == **null**) {  
 **return** $this->redirect(**"/"**);  
 }  
 $form = $this->createForm(FilmType::***class***, $film);  
 $form->handleRequest($request);  
 **if** ($form->isSubmitted()) {  
 $em = $this->getDoctrine()->getManager();  
 $em->remove($film);  
 $em->flush();  
 **return** $this->redirect(**"/"**);  
 }  
 **return** $this->render(  
 **":film:delete.html.twig"**,  
 [**"film"** => $film, **"form"** => $form->createView()]  
 );  
}

**5.View/Create:**

|  |
| --- |
| {%extends"base.html.twig"%} |
|  |  |
|  | {% block main %} |
|  | <h1>Create Film</h1> |
|  | <section> |
|  | <form method="POST"> |
|  | <div> |
|  | <label for="name">Name</label> |
|  | <input type="text" id="name" name="film[name]" |
|  | value="{{ film.name }}" /> |
|  | <label for="genre">Genre</label> |
|  | <input type="text" id="genre" name="film[genre]" |
|  | value="{{ film.genre }}"/> |
|  | <label for="director">Director</label> |
|  | <input type="text" id="director" name="film[director]" |
|  | value="{{ film.director }}"/> |
|  | <label for="year">Year</label> |
|  | <input type="text" id="year" name="film[year]" |
|  | value="{{ film.year }}"/> |
|  |  |
|  | {{ form\_row(form.\_token) }} |
|  |  |
|  | <button type="submit" class="accept">Create</button> |
|  | <button type="button" onclick="location.href='/'" class="cancel">Cancel</button> |
|  | </div> |
|  | </form> |
|  | </section> |
|  | {% endblock %} |