**МОЛДАВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**ФАКУЛЬТЕТ МАТЕМАТИКИ И ИНФОРМАТИКИ**  
**ДЕПАРТАМЕНТ ИНФОРМАТИКИ**

**Индивидуальная работа**

**Framework-uri pentru Dezvoltarea Aplicațiilor Web**

Выполнила: студентка группы IA-1902

Гоян Анна

Проверяет: преподаватель

Бодруг Светлана

**КИШИНЁВ-2021**

Шаг 1: Создать проект

composer create-project symfony/skeleton FlightsApp

composer require twig

composer require annotations

Шаг 2:

Создайте контроллер и маршрут

<?php

namespace App\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;

use Symfony\Component\HttpFoundation\Response;

use Symfony\Component\Routing\Annotation\Route;

class FlightsController extends AbstractController{

/\*\*

\* @Route("/flights/destination")

\*/

public function destinationAction() {

return $this->render('flights/destination.html.twig');

}

}

## Шаг 3:**Создание представления**

“destination.html.twig”

<h3> Simple flights application</h3>

1. Установка Doctrine.  
2. Настройка базы данных  
3. создание базы данных

1. composer require symfony/orm-pack

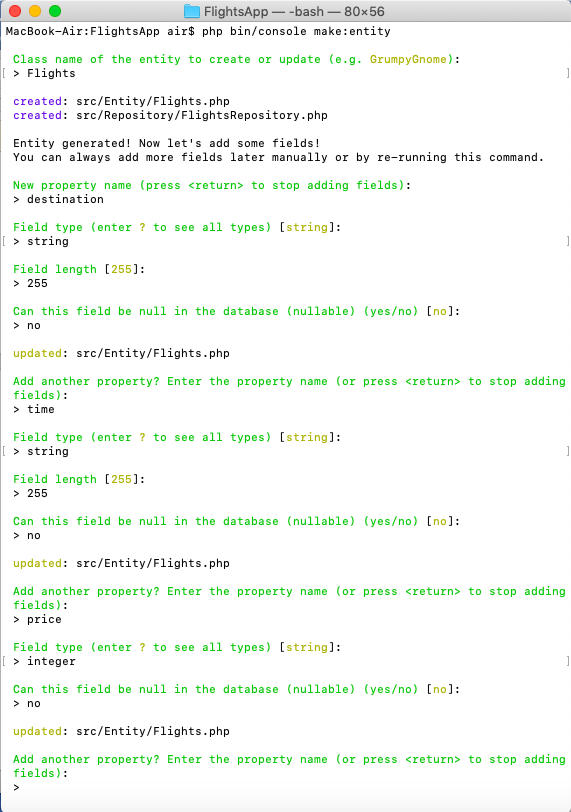
* composer require --dev symfony/maker-bundle

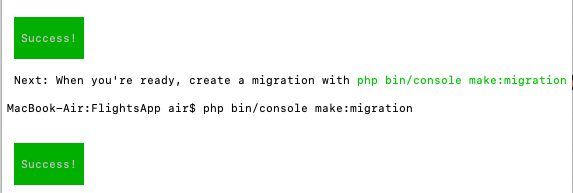
2. внутри .env нужно составить чтобы работала БД:

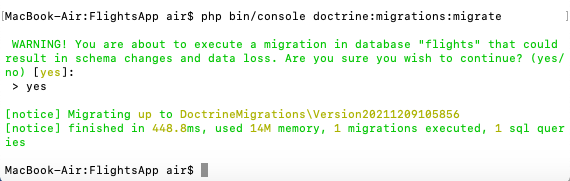
DATABASE\_URL="mysql://air:123@127.0.0.1:3306/flights?serverVersion=mariadb-10.4.11"

* php bin/console doctrine:database:create

Создание класса сущности







В базе данных появилась таблица



Появился файл Flights.php:

<?php

namespace App\Entity;

use App\Repository\FlightsRepository;

use Doctrine\ORM\Mapping as ORM;

/\*\*

\* @ORM\Entity(repositoryClass=FlightsRepository::class)

\*/

class Flights

{

/\*\*

\* @ORM\Id

\* @ORM\GeneratedValue

\* @ORM\Column(type="integer")

\*/

private $id;

/\*\*

\* @ORM\Column(type="string", length=255)

\*/

private $destination;

/\*\*

\* @ORM\Column(type="string", length=255)

\*/

private $time;

/\*\*

\* @ORM\Column(type="integer")

\*/

private $price;

public function getId(): ?int

{

return $this->id;

}

public function getDestination(): ?string

{

return $this->destination;

}

public function setDestination(string $destination): self

{

$this->destination = $destination;

return $this;

}

public function getTime(): ?string

{

return $this->time;

}

public function setTime(string $time): self

{

$this->time = $time;

return $this;

}

public function getPrice(): ?int

{

return $this->price;

}

public function setPrice(int $price): self

{

$this->price = $price;

return $this;

}

}

Вывод на экран содержимое бд

/\*\*

\* @Route("/list", name="product\_list")

\*/

public function showAll()

{

$products = $this->getDoctrine()->getRepository(Product::class)->findAll();

return $this->render('product/list.html.twig',[

'products' => $products,

]);

}

{% extends('base.html.twig') %}

{% block stylesheets %}

<style>

body {

background-color: #E5CCFF;

}

h2 {

text-align: center;

color: #4C0099;

}

form {

text-align:center;

}

button{

padding: 10px;

color: #4C0099;

}

.table { border-collapse: collapse; }

.table th, td {

border-bottom: 1px solid #4C0099;

padding: 10px;

width: 250px;

text-align: left;

align: left;

}

</style>

{% endblock %}

{% block body%}

<h2>List of flights</h2>

<table class="table table-striped">

<thead>

<tr>

<td>ID</td>

<td>Destination</td>

<td>Time</td>

<td>Price</td>

<td></td>

<td></td>

</tr>

</thead>

<tbody>

{% for product in products %}

<tr>

<td>{{product.id}}</td>

<td>

{{product.destination}}

</td>

<td>

{{product.time}}

</td>

<td>

{{product.price}} €

</td>

<td><a href = "{{ path('product\_edit', { 'id' : product.id }) }}">Edit</a></td>

<td><a href = "{{ path('product\_delete', { 'id' : product.id }) }}">Delete</a></td>

</tr>

{%endfor%}

</tbody>

</table>

<br>

<form action="{{path('new\_flight')}}">

<button type="submit" class="btn btn-primary btn-lg btn-block">Create new record</button>

</form>

{% endblock %}



**composer require symfony/form** для создания формы

Добавление новой записи:

/\*\*

\* @Route("/new", name="new\_flight")

\*/

public function newAction(Request $request) {

$stud = new Product();

$category = new Category();

$category->setDestination('Flights');

$form = $this->createFormBuilder($stud)

->add('destination', TextType::class)

->add('time', TextType::class)

->add('price', TextType::class)

->add('save', SubmitType::class, array('label' => 'Submit'))

->getForm();

$form->handleRequest($request);

if ($form->isSubmitted() && $form->isValid()) {

$stud->setCategory($category);

$stud = $form->getData();

$doct = $this->getDoctrine()->getManager();

// tells Doctrine you want to save the Product

$doct->persist($stud);

$doct->persist($category);

//executes the queries (i.e. the INSERT query)

$doct->flush();

return $this->redirectToRoute('product\_list');

} else {

return $this->render('product/new.html.twig', array('form' => $form->createView(),));

}

}

В new.html.twig

{% extends 'base.html.twig' %}

{% block stylesheets %}

<style>

body {

background-color: #E5CCFF;

}

h3{

text-align: center;

color: #4C0099;

font-size: 24px;

}

#simpleform {

width:600px;

border:2px solid #4C0099;

padding:14px;

margin: 0 auto;

}

#simpleform label {

font-size:18px;

float:left;

width:300px;

text-align:right;

display:block;

}

#simpleform span {

font-size:11px;

color:grey;

width:100px;

text-align:right;

display:block;

}

#simpleform input {

border:1px solid grey;

font-family:verdana;

font-size:14px;

color:light blue;

height:24px;

width:250px;

margin: 0 0 10px 10px;

}

#simpleform textarea {

border:1px solid grey;

font-family:verdana;

font-size:14px;

color:light blue;

height:120px;

width:250px;

margin: 0 0 20px 10px;

}

#simpleform select {

margin: 0 0 20px 10px;

}

#simpleform button {

clear:both;

margin-left:250px;

background: #4C0099;

color:#FFFFFF;

border:solid 1px #666666;

font-size:16px;

padding:10px;

cursor: pointer;

}

</style>

{% endblock %}

{% block body %}

<h3>Flight details:</h3>

<div id = "simpleform">

{{ form\_start(form) }}

{{ form\_widget(form) }}

{{ form\_end(form) }}

</div>

{% endblock %}

Изменение записи:

/\*\*

\* @Route("/product/edit/{id}", name="product\_edit")

\*/

public function update($id, Request $request)

{

$doct = $this->getDoctrine()->getManager();

$bk = $doct->getRepository(Product::class)->find($id);

if (!$bk) {

throw $this->createNotFoundException(

'No flight found for id '.$id

);

}

$form = $this->createFormBuilder($bk)

->add('destination', TextType::class)

->add('time', TextType::class)

->add('price', TextType::class)

->add('save', SubmitType::class, array('label' => 'Submit'))

->getForm();

$form->handleRequest($request);

if ($form->isSubmitted() && $form->isValid()) {

$stud = $form->getData();

$doct = $this->getDoctrine()->getManager();

// tells Doctrine you want to save the Product

$doct->persist($stud);

//executes the queries (i.e. the INSERT query)

$doct->flush();

return $this->redirectToRoute('product\_list');

} else {

return $this->render('product/new.html.twig', array('form' => $form->createView(),));

}

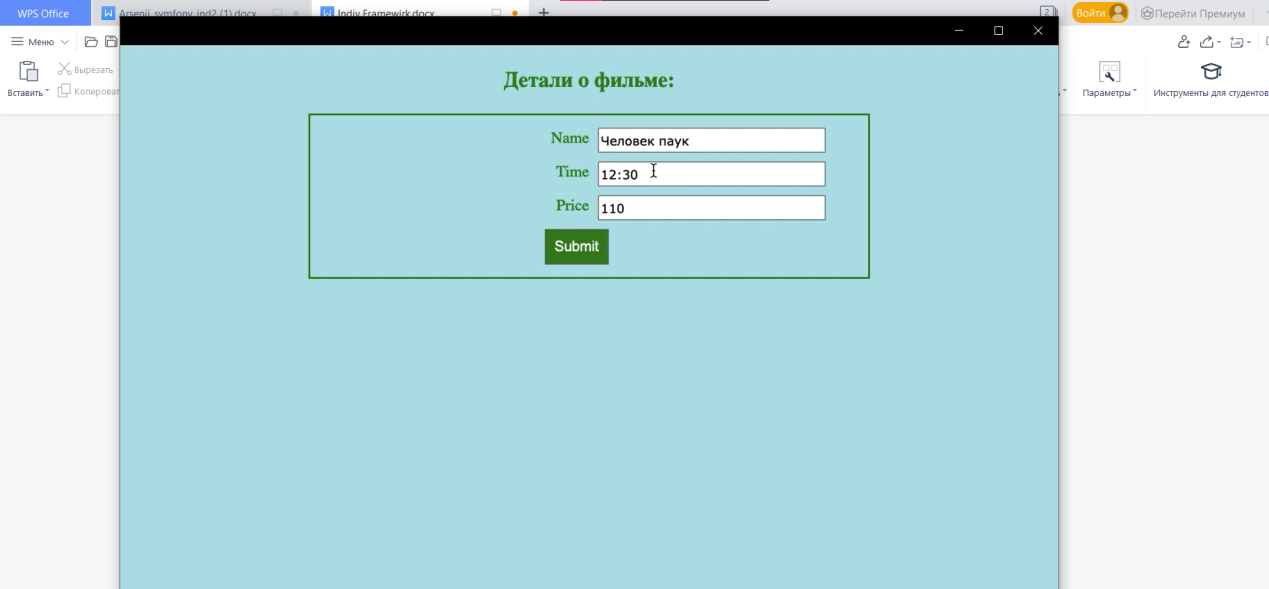
}

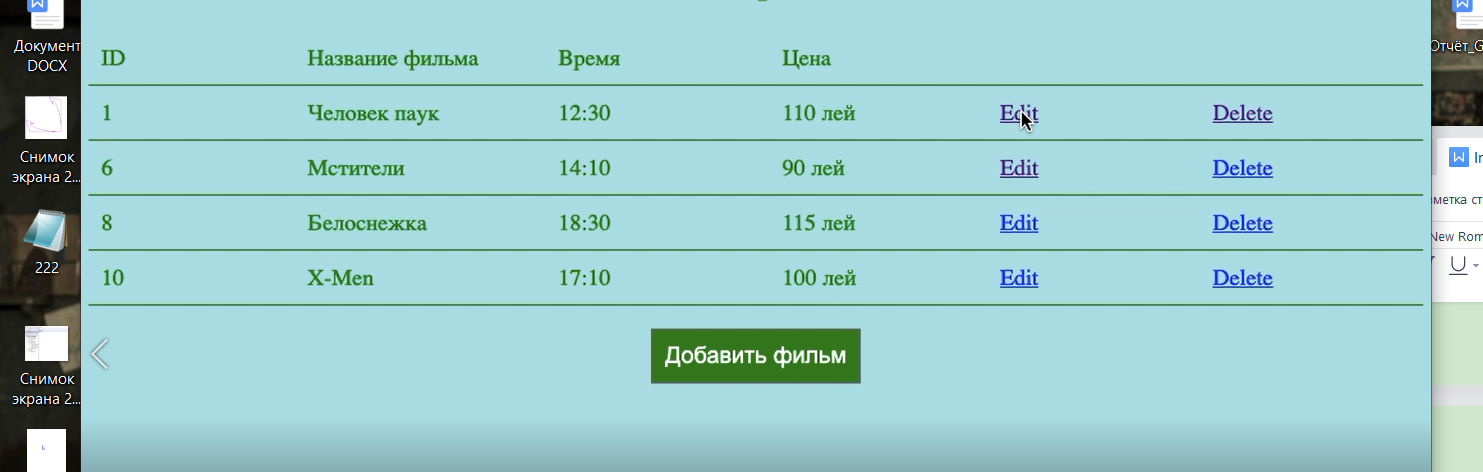
Нажимаем на edit и меняем выбранную запись

Либо указываем путь /product/edit/{id}

Поменяла цену на 180 и нажала на Submit







Удаление записи:

/\*\*

\* @Route("/product/delete/{id}", name="product\_delete")

\*/

public function delete($id){

$product = $this->getDoctrine()->getRepository(Product::class)->find($id);

$entityManager = $this->getDoctrine()->getManager();

if (!$product) {

throw $this->createNotFoundException('No flight found for id '.$id);

}

//remove the product with id = $id

$entityManager -> remove($product);

$entityManager -> flush();

return $this->redirectToRoute('product\_list');

}

Нажимаем на delete или указываем путь /product/delete/{id}

И запись удаляется

