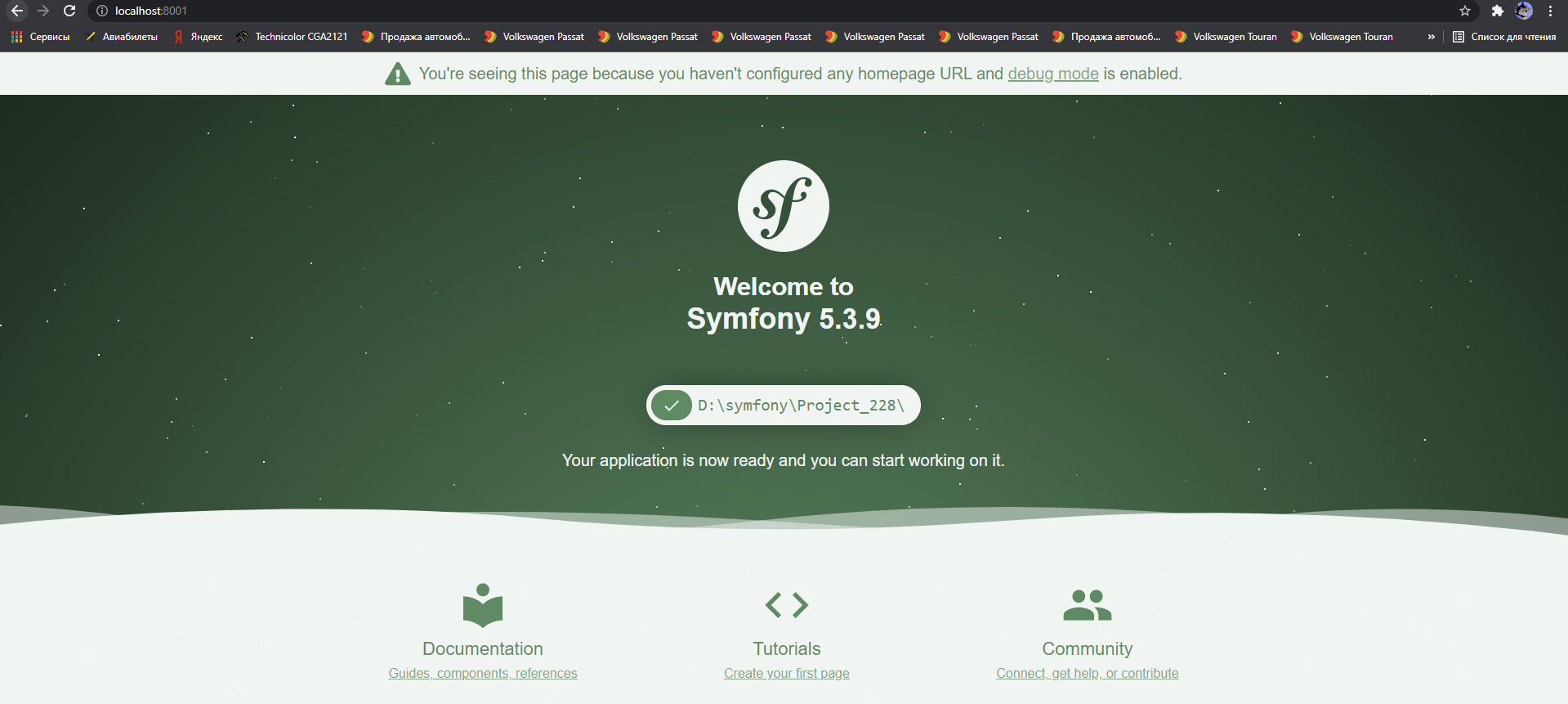
Создание приложений Symfony

symfony new my\_project

cd my\_project

symfony server:start



Создаем свою первую страницу в Symfony:

Примером создания страницы, послужит создание страницы /lucky/number которая генерирует счастливое число и печатает его. Для этого создайте класс «Controller» и метод «controller» внутри него:

<?php

// src/Controller/LuckyController.php

namespace App\Controller;

use Symfony\Component\HttpFoundation\Response;

class LuckyController

{

public function number()

{

$number = random\_int(0, 100);

return new Response(

'<html><body>Lucky number: '.$number.'</body></html>'

);

}

}

Cвяжем эту функцию контроллера с общедоступным URL , чтобы number()метод выполнялся, когда пользователь просматривает его. Эта связь определяется путем создания маршрута в config/routes.yaml файле:

app\_lucky\_number:

path: /lucky/number

controller: App\Controller\LuckyController::number

Маршруты аннотаций

Устанавливаем маршруты c помощью :composer require annotations

Дополняем наш php код:

<?php

// src/Controller/LuckyController.php

namespace App\Controller;

use Symfony\Component\HttpFoundation\Response;

use Symfony\Component\Routing\Annotation\Route;

class LuckyController

{

/\*\*

\* @Route("/lucky/number")

\*/

public function number()

{

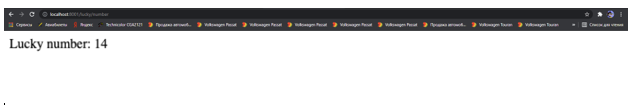
$number = random\_int(0, 100);

return new Response(

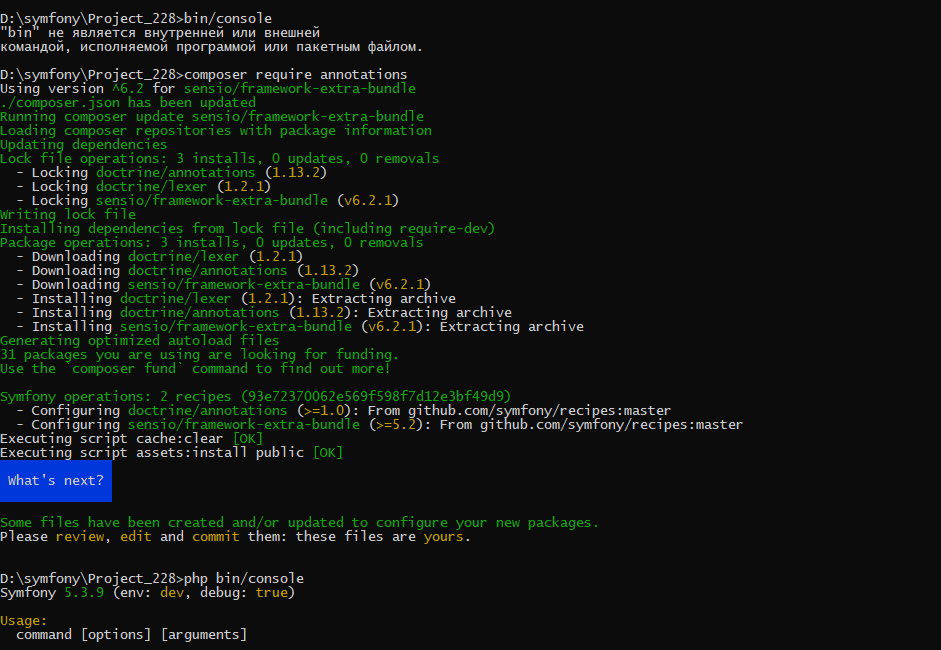
'<html><body>Lucky number: '.$number.'</body></html>'

);

}

}

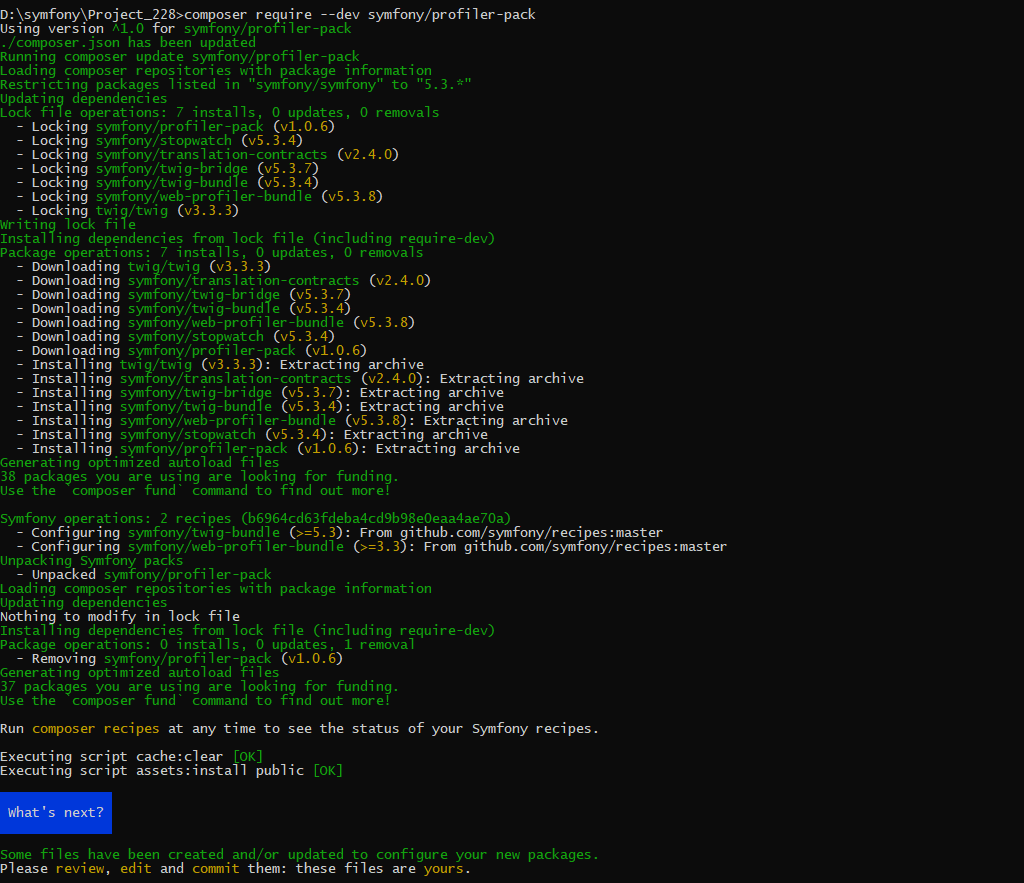
Прописываем команду bin/console



Чтобы получить список всех маршрутов в вашей системе, используйте debug:router



Прописываем composer require --dev symfony/profiler-pack



Проверяем



Необходимо проверить ,что LuckyController расширяет базовый AbstractController класс Symfony :

<?php

// src/Controller/LuckyController.php

namespace App\Controller;

use Symfony\Component\HttpFoundation\Response;

use Symfony\Component\Routing\Annotation\Route;

use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;

class LuckyController extends AbstractController

{

/\*\*

\* @Route("/lucky/number")

\*/

public function number()

{

$number = random\_int(0, 100);

return $this->render('lucky/number.html.twig', [

'number' => $number,

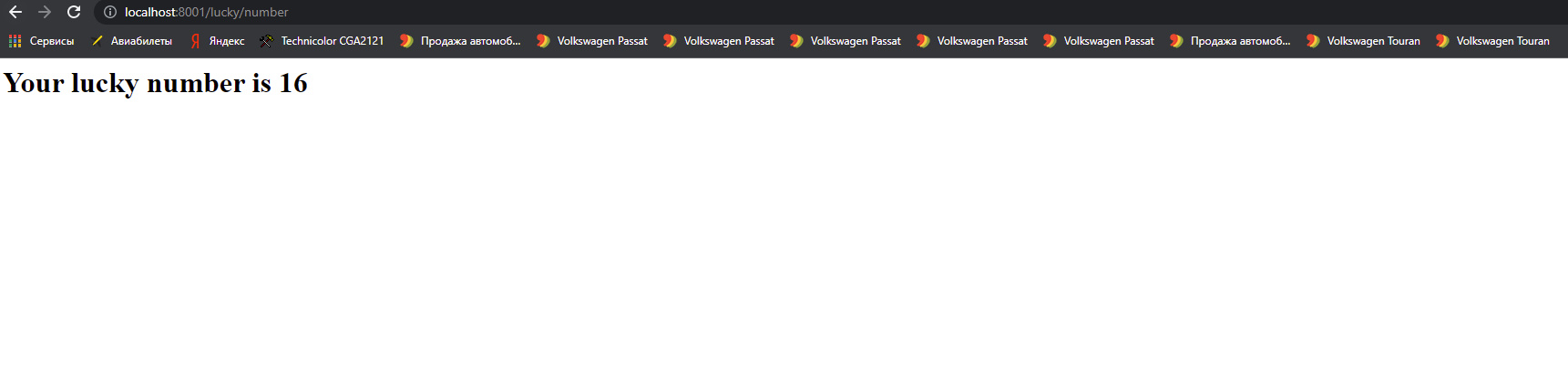
]);

}

}

Файлы шаблонов находятся в templates/каталоге, который был создан для вас автоматически при установке Twig. Создайте новый templates/lucky каталог с новым number.html.twig файлом внутри:

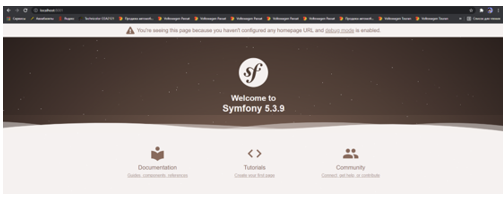
{# templates/lucky/number.html.twig #}

<h1>Your lucky number is {{ number }}</h1>

Маршрут, Контроллер, Ответ:

Для начала создадим проект

composer create-project symfony/skeleton quick\_tour



Меняем config/routes.yaml:

index:

path: /

controller: App\Controller\DefaultController::index



Дописываем path в routes.yaml и пишем скрипт :

index:

path: /hello/{name}

controller: App\Controller\DefaultController::index

<?php

namespace App\Controller;

use Symfony\Component\HttpFoundation\Response;

class DefaultController

{

public function index($name)

{

return new Response("Привет, $name!");

}

}

У нас поменялся URL: теперь он /hello/\*: {name}



Так что давайте установим поддержку аннотаций:

composer require annotations

Теперь, закомментируйте маршрут YAML

#index:

# path: /hello/{name}

# controller: App\Controller\DefaultController::index

Вместо этого, добавьте маршрут прямо над методом контроллера:

<?php

namespace App\Controller;

use Symfony\Component\Routing\Annotation\Route;

use Symfony\Component\HttpFoundation\Response;

class DefaultController

{

/\*\*

\* @Route("/hello/{name}")

\*/

public function index($name)

{

return new Response("Привет, $name!");

}

}



Добавляем ешё один маршрут и метод в DefaultController:

/\*\*

\* @Route("/simplicity")

\*/

public function simple()

{

return new Response(' simple! Легко! Прекрасно!');

}

