**Symfony**

**Setup:**

Create your new project by running: **composer create-project symfony/website-skeleton my-project**

Adding Rewrite Rules[¶](https://symfony.com/doc/current/setup/web_server_configuration.html#adding-rewrite-rules)

The easiest way is to install the Apache recipe by executing the following command:

|  |
| --- |
| composer require symfony/apache-pack |

Do you want to execute this recipe?

[y] Yes

[n] No

[a] Yes for all packages, only for the current installation session

[p] Yes permanently, never ask again for this project

**Type yes 🡺 Y**

Annotation Routes[¶](https://symfony.com/doc/current/page_creation.html#annotation-routes)

Instead of defining your route in YAML, Symfony also allows you to use *annotation* routes. To do this, install the annotations package:

|  |
| --- |
| composer require annotations |

You can now add your route directly *above* the controller:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | // src/Controller/LuckyController.php  // ...  + use Symfony\Component\Routing\Annotation\Route;  class LuckyController  {  + /\*\*  + \* @Route("/lucky/number")  + \*/  public function number()  {  // this looks exactly the same  }  } |

# **Create your First Page in Symfony:**

## Creating a Page: Route and Controller

 create a "Controller class" and a "controller" method inside of it

<?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;

/\*\*

\* @Route("/lucky")

\*/

class LuckyController extends AbstractController

{

/\*\*

\* @Route("/number")

\*/

public function number()

{

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

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

'number' => $number,

]);

}

}

Template files live in the templates/ directory, which was created for you automatically when you installed Twig. Create a new templates/lucky directory with a new number.html.twig file inside:

|  |  |
| --- | --- |
| 1  2  3 | *{# templates/lucky/number.html.twig #}*  <h1>Your lucky number is {{ number }}</h1> |

The {{ number }} syntax is used to *print* variables in Twig. Refresh your browser to get your *new*lucky number!

# **Routing :**

# **How to Generate Entities from an Existing Database:**

<https://symfony.com/doc/current/doctrine/reverse_engineering.html>

The first step towards building entity classes from an existing database is to ask Doctrine to introspect the database and generate the corresponding metadata files. Metadata files describe the entity class to generate based on table fields.

|  |
| --- |
| php bin/console doctrine:mapping:import 'App\Entity' annotation --path=src/Entity |

## Generating the Getters & Setters or PHP Classes:

// generates getter/setter methods

php bin/console make:entity --regenerate App

## Authentication:

## 1) Installation[¶](https://symfony.com/doc/current/security.html#installation)

In applications using [Symfony Flex](https://symfony.com/doc/current/setup/flex.html), run this command to install the security feature before using it:

|  |
| --- |
| composer require symfony/security-bundle |

## 2a) Create your User Class

php bin/console make:user

The name of the security user class (e.g. User) [User]:

> User

Do you want to store user data in the database (via Doctrine)? (yes/no) [yes]:

> yes

Enter a property name that will be the unique "display" name for the user (e.g.

email, username, uuid [email]

> email

Does this app need to hash/check user passwords? (yes/no) [yes]:

> yes

created: src/Entity/User.php

created: src/Repository/UserRepository.php

updated: src/Entity/User.php

updated: config/packages/security.yaml

php bin/console make:migration

php bin/console doctrine:migrations:migrate

$userStudent = new StudentProfile();

$userStudent->setRole('ll');

$userStudent->setFirstname('Eugene');

$userStudent->setLastname('Duzant');

$userStudent->setSex('Duzant');

$userStudent->setAge(55);

$userStudent->setEmail('jl-test@gmail.com');

$userStudent->setLastdiploma('Bac');

$userStudent->setStreetaddress('73 rue des haies Paris');

$userStudent->setCountry('France');

$userStudent->setZipcode('75020');

$userStudent->setPassword($this->passwordEncoder->encodePassword(

$userStudent,

'the\_new\_password'

));

$manager->persist($userStudent);

$manager->flush();

<?php

namespace App\DataFixtures;

use Doctrine\Bundle\FixturesBundle\Fixture;

use Doctrine\Common\Persistence\ObjectManager;

use Symfony\Component\Security\Core\Encoder\UserPasswordEncoderInterface;

use App\Entity\StudentProfile;

class StudentProfile\_UserFixtures extends Fixture

{

private $passwordEncoder;

public function \_\_construct(UserPasswordEncoderInterface $passwordEncoder)

{

$this->passwordEncoder = $passwordEncoder;

}

public function load(ObjectManager $manager)

{

$userStudent = new StudentProfile();

$userStudent->setRole('ll');

$userStudent->setFirstname('Eugene');

$userStudent->setLastname('Duzant');

$userStudent->setSex('Duzant');

$userStudent->setAge(55);

$userStudent->setEmail('jl-test@gmail.com');

$userStudent->setLastdiploma('Bac');

$userStudent->setStreetaddress('73 rue des haies Paris');

$userStudent->setCountry('France');

$userStudent->setZipcode('75020');

$userStudent->setPassword($this->passwordEncoder->encodePassword(

$userStudent,

'the\_new\_password'

));

$manager->persist($userStudent);

$manager->flush();

}

}

$userStudent = new StudentProfile();

$userStudent->setEmail('test-symfony@hotmail.com');

$userStudent->setUsername("tes-Admin");

$userStudent->setRoles(['User-Admin']);

$userStudent->setPassword($this->passwordEncoder->encodePassword(

$userStudent,

'the\_new\_password'

));

$manager->persist($userStudent);

$manager->flush();