**DS SYMFONY**

1/ Environnement :

Xampp

Vscode

Composer

Symfony CLI

2/ Les commandes utilisées dans le projet :

Symfony new *Projet-DS* –full

Cd Projet\_DS

symfony server:start

php bin/console doctrine:database:create Vente

composer require annotations

composer require twig

composer require symfony/orm-pack

composer require –dev symfony/maker-bundle

php bin/console make:entity Voiture (marque, couleur, prix)

php bin/console make:migration

php bin/console doctrine:migrations:migrate

php bin/console make:controller Car Controller

composer require –dev orm-fixtures

php bin/console make:fixtures

php bin/console doctrine :fixtures :load

3/ Partie code :

1. .env :
2. ###> symfony/webapp-pack ###
3. MESSENGER\_TRANSPORT\_DSN=doctrine://default?auto\_setup=0
4. ###< symfony/webapp-pack ###
5. ###> symfony/framework-bundle ###
6. APP\_ENV=dev
7. APP\_SECRET=cc641fe177f5249b0ae881e084a59560
8. ###< symfony/framework-bundle ###
9. ###> doctrine/doctrine-bundle ###
10. # Format described at https://www.doctrine-project.org/projects/doctrine-dbal/en/latest/reference/configuration.html#connecting-using-a-url
11. # IMPORTANT: You MUST configure your server version, either here or in config/packages/doctrine.yaml
12. #
13. # DATABASE\_URL="sqlite:///%kernel.project\_dir%/var/data.db"
14. DATABASE\_URL="mysql://root:@127.0.0.1:3306/vente?serverVersion=8&charset=utf8mb4"
15. # DATABASE\_URL="postgresql://app:!ChangeMe!@127.0.0.1:5432/app?serverVersion=15&charset=utf8"
16. ###< doctrine/doctrine-bundle ###
17. Controller :

* Fonction Index et use :
* <?php
* namespace App\Controller;
* use App\Form\CarType;
* use App\Entity\Voiture;
* use Doctrine\ORM\EntityManager;
* use App\Repository\CarRepository;
* use App\Repository\VoitureRepository;
* use Doctrine\ORM\EntityManagerInterface;
* use Symfony\Component\HttpFoundation\Request;
* use Symfony\Component\HttpFoundation\Response;
* use Symfony\Component\Routing\Annotation\Route;
* use Symfony\Component\HttpFoundation\JsonResponse;
* use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
* use Doctrine\Persistence\ManagerRegistry as PersistenceManagerRegistry;
* Function allCar :

class CarController extends AbstractController

{

    /\*\*

     \* @Route("/", name="car")

     \*/

     public function index(): Response {

        return $this->render("home.html.twig");

    }

    /\*\*

     \* @Route("/allCar", name="allCar")

     \*/

     public function allCar(VoitureRepository $rep): Response {

        $cars = $rep->findAll();

        return $this->render(

            "allCar.html.twig",

            ["cars"=>$cars]

        );

    }

* Function detail :
* /\*\*
* \* @Route("/detail/{id}", name="details")
* \*/
* public function oneCar (VoitureRepository $rep, $id) : Response {
* $oneCar = $rep->find($id);
* return $this->render("detail.html.twig", ["onecar"=>$oneCar]);
* }
* Function addCar :
* /\*\*
* \* @Route("/add", name="addCar")
* \*/
* public function addCar(Request $req , PersistenceManagerRegistry $doctrine){
* $car = new Voiture();
* $form = $this->createForm(CarType::class, $car);
* $form->handleRequest($req);
* if ($form->isSubmitted() && $form->isValid()) {
* $car = $form->getData();
* $entityManger = $doctrine->getManager();
* $entityManger->persist($car);
* $entityManger->flush();
* $this->addFlash('notice','Une nouvelle voiture a été ajoutée avec succès ...');
* return $this->redirectToRoute('allCar');
* }
* return $this->render('addCar.html.twig', [
* 'form' => $form->createView()
* ]);
* }

1. Entity :
2. <?php
3. namespace App\Entity;
4. use App\Repository\VoitureRepository;
5. use Doctrine\ORM\Mapping as ORM;
6. #[ORM\Entity(repositoryClass: VoitureRepository::class)]
7. class Voiture
8. {
9. #[ORM\Id]
10. #[ORM\GeneratedValue]
11. #[ORM\Column]
12. private ?int $id = null;
13. #[ORM\Column(length: 255)]
14. private ?string $marque = null;
15. #[ORM\Column(length: 255)]
16. private ?string $couleur = null;
17. #[ORM\Column(length: 255)]
18. private ?string $prix = null;
19. public function getId(): ?int
20. {
21. return $this->id;
22. }
23. public function getMarque(): ?string
24. {
25. return $this->marque;
26. }
27. public function setMarque(string $marque): self
28. {
29. $this->marque = $marque;
30. return $this;
31. }
32. public function getCouleur(): ?string
33. {
34. return $this->couleur;
35. }
36. public function setCouleur(string $couleur): self
37. {
38. $this->couleur = $couleur;
39. return $this;
40. }
41. public function getPrix(): ?string
42. {
43. return $this->prix;
44. }
45. public function setPrix(string $prix): self
46. {
47. (string) $this->prix = $prix;
48. return $this;
49. }
50. }
51. Fixtures :
52. <?php
53. namespace App\DataFixtures;
54. use Doctrine\Bundle\FixturesBundle\Fixture;
55. use Doctrine\Persistence\ObjectManager;
56. use App\Entity\Voiture;
57. class CarFixture extends Fixture
58. {
59. public function load(ObjectManager $manager): void
60. {
61. for ($i = 1; $i <= 10; $i++) {
62. $car = new Voiture();
63. $car->setMarque("Audi " . $i);
64. $car->setCouleur("Blanc " . $i);
65. $car->setPrix("100.000" . $i);
66. $manager->persist($car);
67. }
68. $manager->flush();
69. }
70. }
71. Form :
72. <?php
73. namespace App\Form;
74. use App\Entity\Voiture;
75. use Symfony\Component\Form\AbstractType;
76. use Symfony\Component\Form\FormBuilderInterface;
77. use Symfony\Component\OptionsResolver\OptionsResolver;
78. class CarType extends AbstractType
79. {
80. public function buildForm(FormBuilderInterface $builder, array $options): void
81. {
82. $builder
83. ->add('marque')
84. ->add('couleur')
85. ->add('prix')
86. ;
87. }
88. public function configureOptions(OptionsResolver $resolver): void
89. {
90. $resolver->setDefaults([
91. 'data\_class' => Voiture::class,
92. ]);
93. }
94. }
95. Les interfaces :









