

# Compte Rendu TP05

## Création de Formulaires

1. Créer la classe formulaire correspondant à l'entité Article avec la commande **php bin/console make :form**

```
The name of the form class (e.g. VictoriousPuppyType):
> ArticleType

The name of Entity or fully qualified model class name that the new form will be bound to (empty for none):
> Article

created: src/Form/ArticleType.php

Success!
```

2. Modifier la fonction new de la classe IndexController.php

```
public function new(Request $request) {
    $article = new Article();
    $form = $this->createForm(ArticleType::class,$article);
    $form->handleRequest($request);

    if($form->isSubmitted() && $form->isValid()) {
        $article = $form->getData();

        $entityManager = $this->entityManager->getManager();
        $entityManager->persist($article);
        $entityManager->flush();

        return $this->redirectToRoute('article_list');
    }
    return $this->render('articles/new.html.twig',['form' => $form->createView()]);
}
```

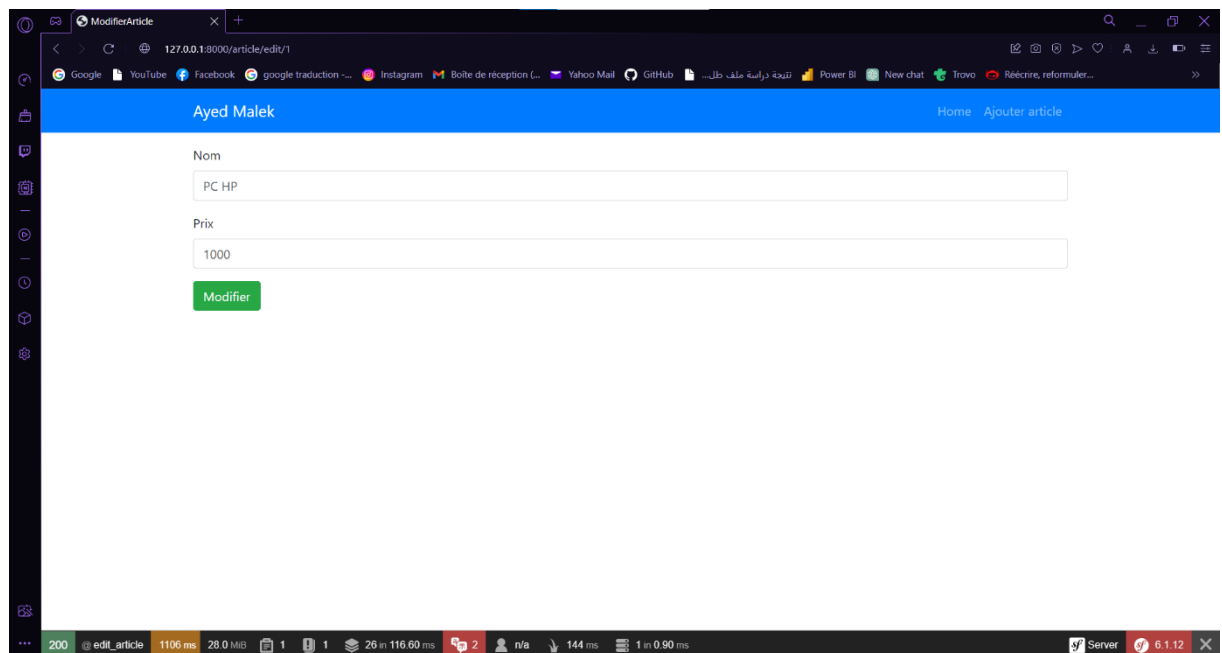
3. Tester la fonction

The screenshot shows a web browser window with the address bar displaying '127.0.0.1:8000/article/new'. The page title is 'Ajouter Article'. The user is logged in as 'Ayed Malek'. The page content shows a form with two input fields: 'Nom' and 'Prix'. Below the 'Prix' field is a green button labeled 'Créer'. The browser's developer tools are open at the bottom, showing the 'new\_article' component with a status of '200' and a response time of '259 ms'.

#### 4. Modifier la fonction edit de la classe IndexController.php

```
public function edit(Request $request, $id) {  
    $article = new Article();  
    $article = $this->entityManager->getRepository(Article::class)->find($id);  
  
    $form = $this->createFormBuilder(ArticleType::class,$article);  
  
    $form->handleRequest($request);  
    if($form->isSubmitted() && $form->isValid()) {  
  
        $entityManager = $this->entityManager->getManager();  
        $entityManager->flush();  
  
        return $this->redirectToRoute('article_list');  
        return $this->render('articles/edit.html.twig', ['form' => $form->createView()]);  
    }  
}
```

#### 5. Tester le travail



### Validation des données

#### 1. Changer l'entité Article

```

#[ORM\Id]
#[ORM\GeneratedValue]
#[ORM\Column]
/**
 * @ORM\Id()
 * @ORM\GeneratedValue()
 * @ORM\Column(type="integer")
 */
private ?int $id = null;

#[ORM\Column(length: 255, nullable: true)]
/**
 * @ORM\Id()
 * @ORM\GeneratedValue()
 * @ORM\Column(type="integer")
 */
private ?string $nom = null;

#[ORM\Column(type: Types::DECIMAL, precision: 10, scale: '0')]
/**
 * @ORM\Column(type="decimal", precision=10, scale=0)
 * @Assert\NotEqualTo(
 *     value = 0,

```

```

#[ORM\Column(type: Types::DECIMAL, precision: 10, scale: '0')]
/**
 * @ORM\Column(type="decimal", precision=10, scale=0)
 * @Assert\NotEqualTo(
 *     value = 0,
 *     message = "Le prix d'un article ne doit pas être égal à 0 "
 * )
 */
private ?string $prix = null;

public function getId(): ?int
{
    return $this->id;
}

public function getNom(): ?string
{
    return $this->nom;
}

public function setNom(?string $nom): self
{
    $this->nom = $nom;

    return $this;
}

```