XDNATIS

Les services

Michael
XDNATIS

XWIKI NATIS

Compétence demandée :
Savoir mettre en place des services

XKNATIS

Qu’est-ce qu’une architecture ?

XWNATIS

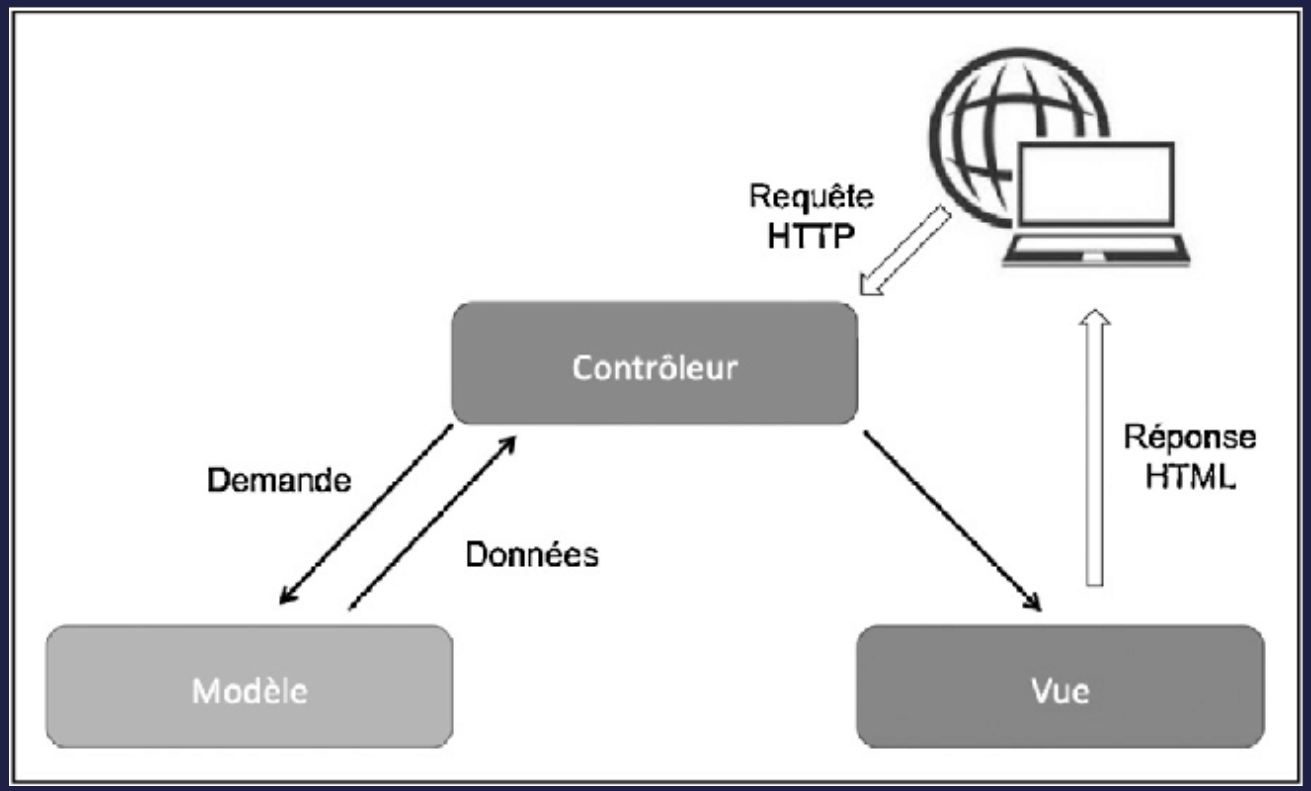
L’architecture MVC (Modèle Vue Contrôleur) est un modèle de conception très utilisé lors de développement de logiciels. C’est une façon d’organiser et structurer le code des interfaces graphiques d’un programme.

L’objectif du modèle MVC est de séparer les aspects suivants :

- Données (Modèle)
- Présentation (Vue)
- Requête et traitement (Contrôleur)

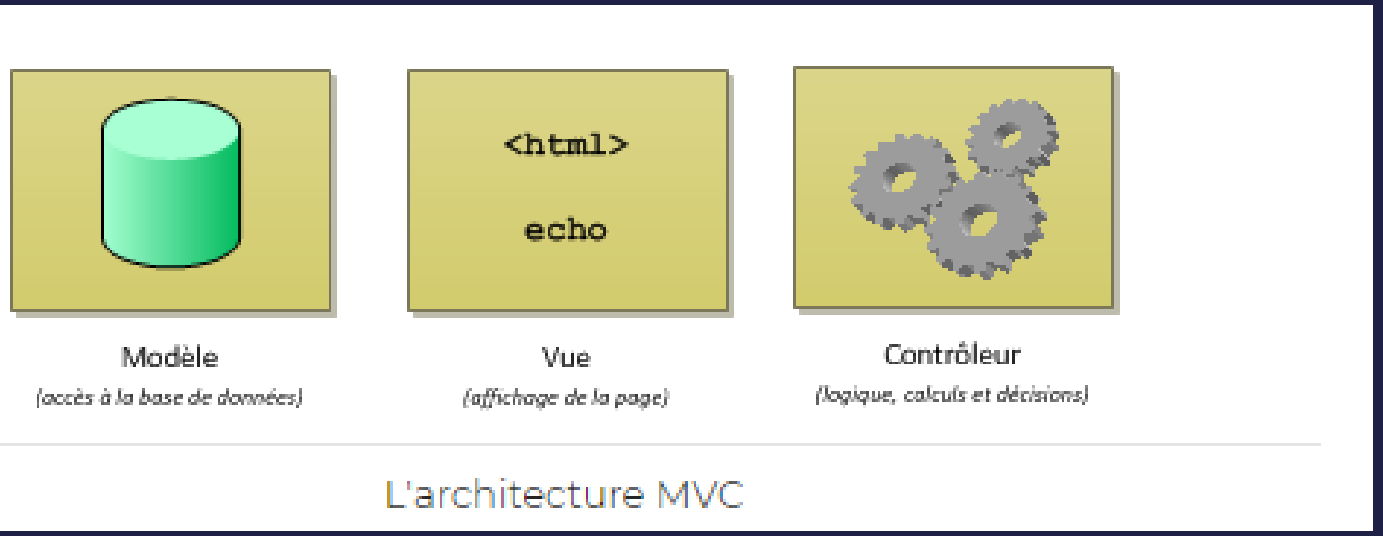
XWNATIS

Architecture MVC



XDNATIS

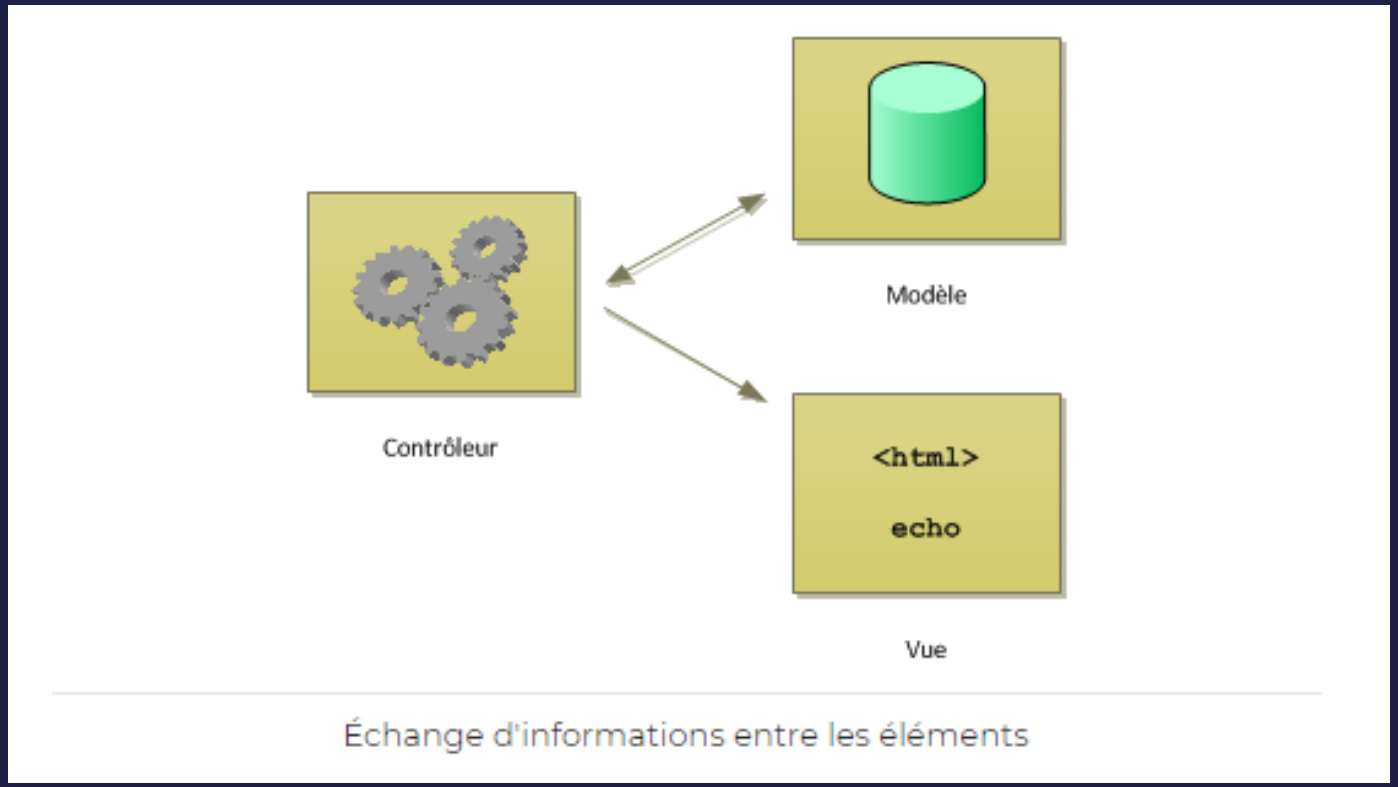
Architecture MVC



L'architecture MVC

XDNATIS

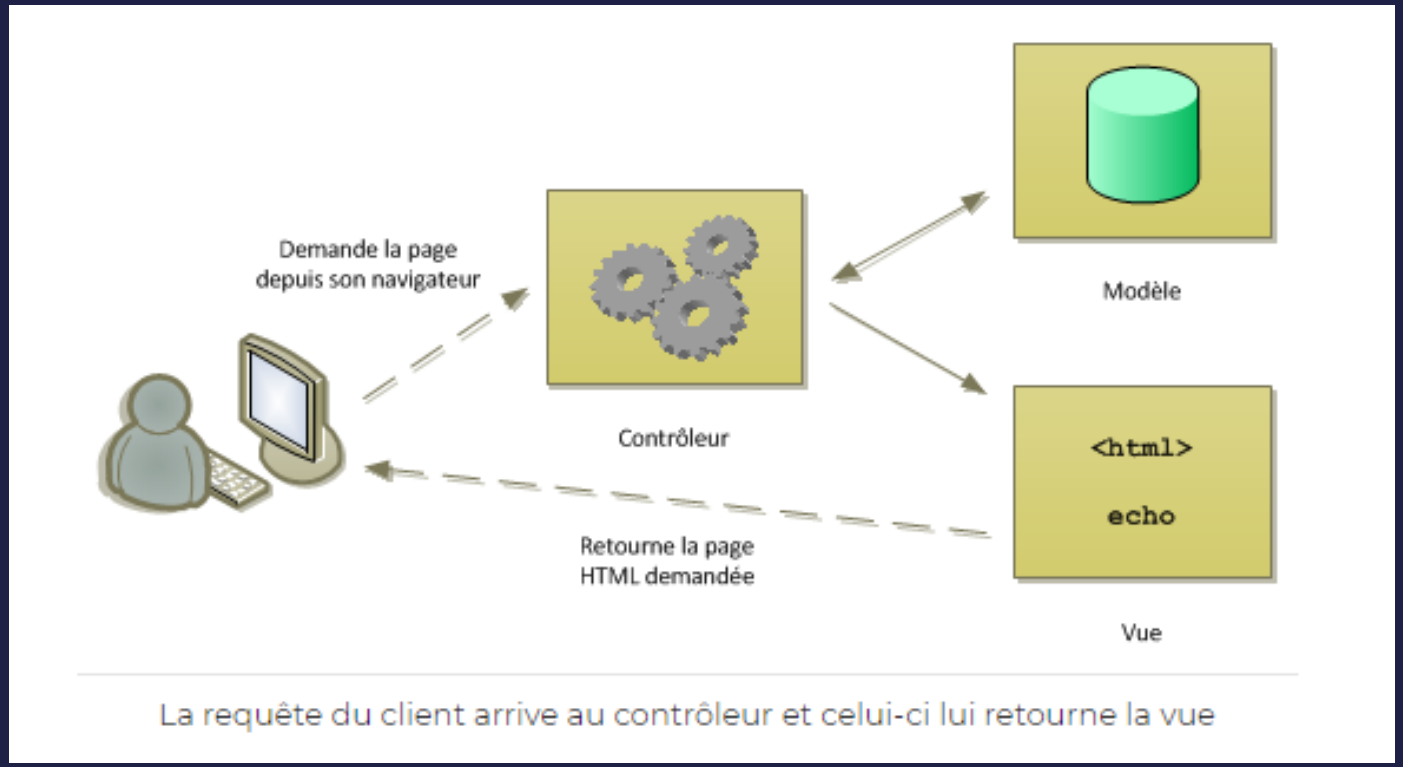
Architecture MVC



Échange d'informations entre les éléments

XNATIS

Architecture MVC



La requête du client arrive au contrôleur et celui-ci lui retourne la vue

XWIKI NATIS

MVC :

Model = Entités

XDNATIS

MVC :

Model = Entités

View = Code pour le visuel

XDNATIS

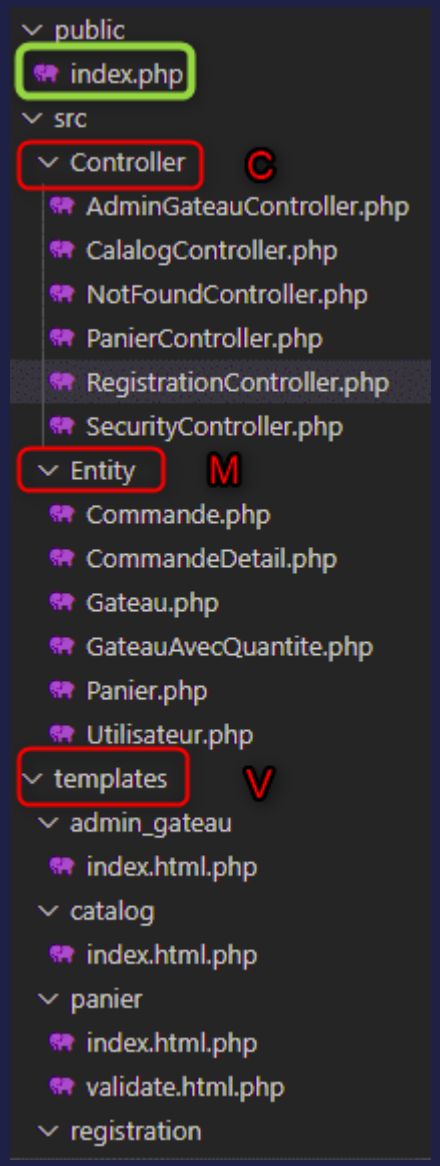
MVC :

Model = Entités

View = Code pour le visuel

Controller = Le reste

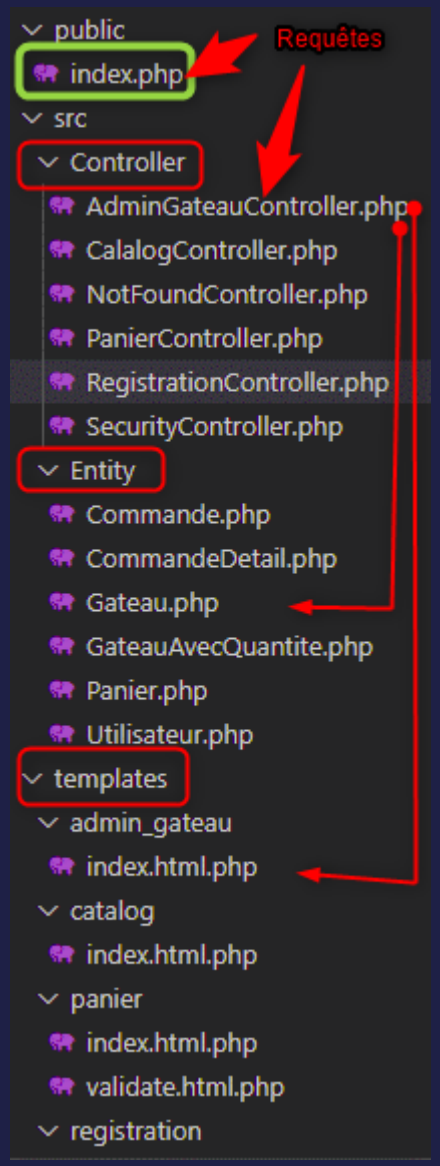
XO NATIS



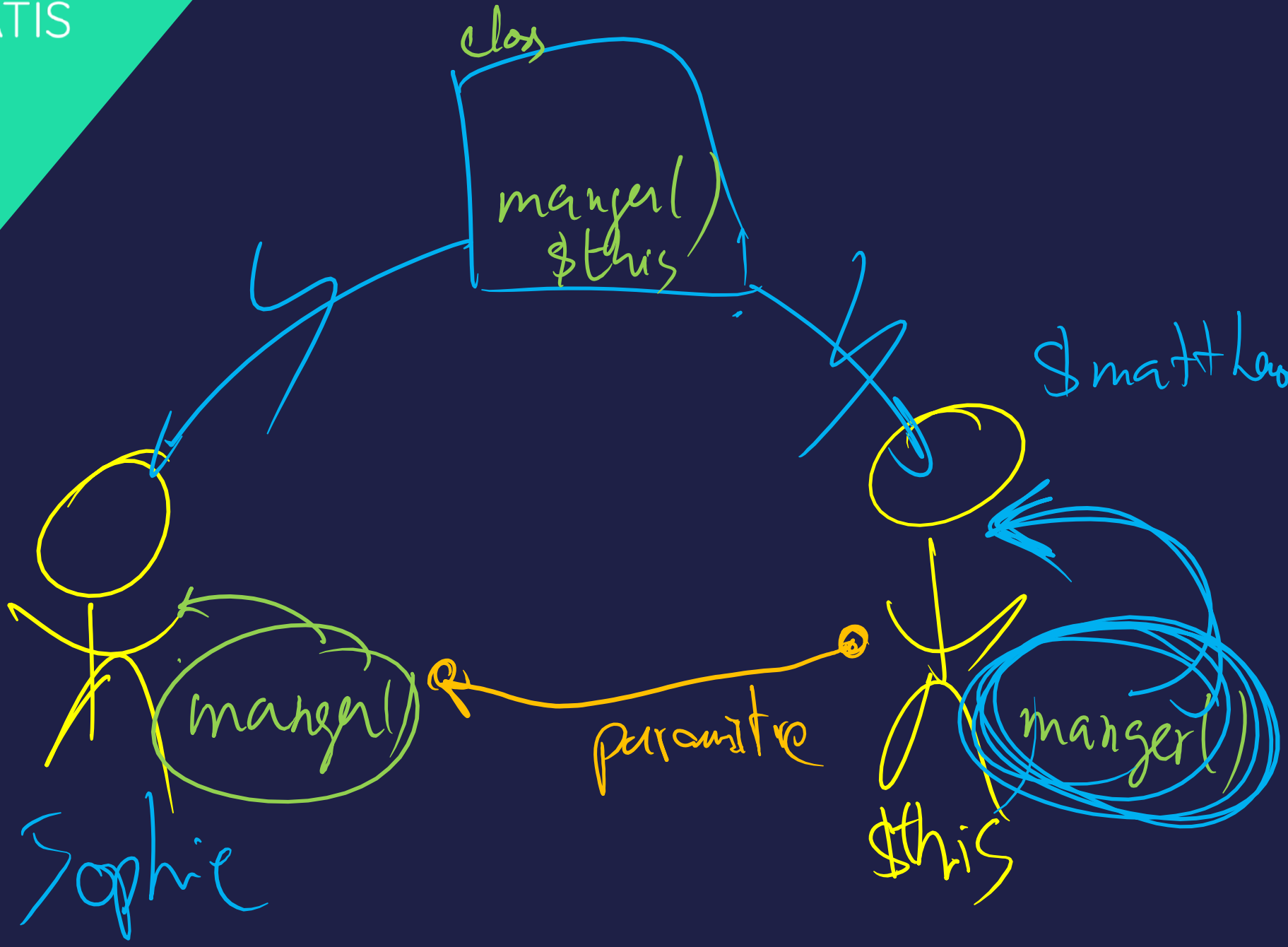
Architecture MVC

XNATIS

Architecture MVC



XDNATIS


class
manger()
$this
$matt
manger()
Sophie
parametre
manger()
$this

XWIKI NATIS

Les controllers peuvent devenir compliqué !

XWIKI NATIS

Les controllers peuvent devenir compliqué !

Isolons la plue-value des controllers dans des services !

XWIKI SAS

XWIKI NATIS

UN SERVICE ?

XWIKI NATIS

maintenance

1. Un bout de code utilisé de manière répétée dans plusieurs controllers
   - ↳ une plus-value -> service
2. Un bout de code qui a une complexité qui va peut-être évoluer
   - ↳ une plus-value -> service

XWIKI SAS

XWIKI SAS

XWIKI NATIS

La couche de service est un modèle architectural, appliqué dans le paradigme de conception orientée service

XWIKI NATIS

Elle vise à rassemble la plus value de l'application en services. Cela permet de faciliter la complexité applicative en simplifiant les controlleurs.

XWIKI NATIS

Elle vise à rassemble la plus value de l'application en services. Cela permet de faciliter la complexité applicative en simplifiant les controlleurs.

XWIKI SAS

XWIKI NATIS

#1

Créer un site internet un système d'administration pour créer des produits en base de données (sans l'ajout d'images)

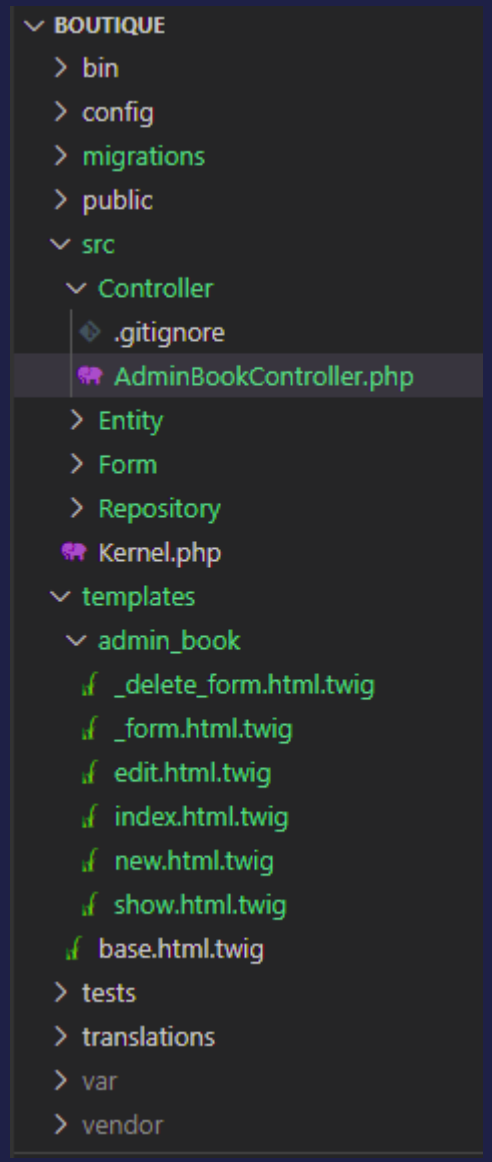
XONATIS

```
symfony new boutique –full
```

XONATIS

```
symfony server:start
```

XDNATIS



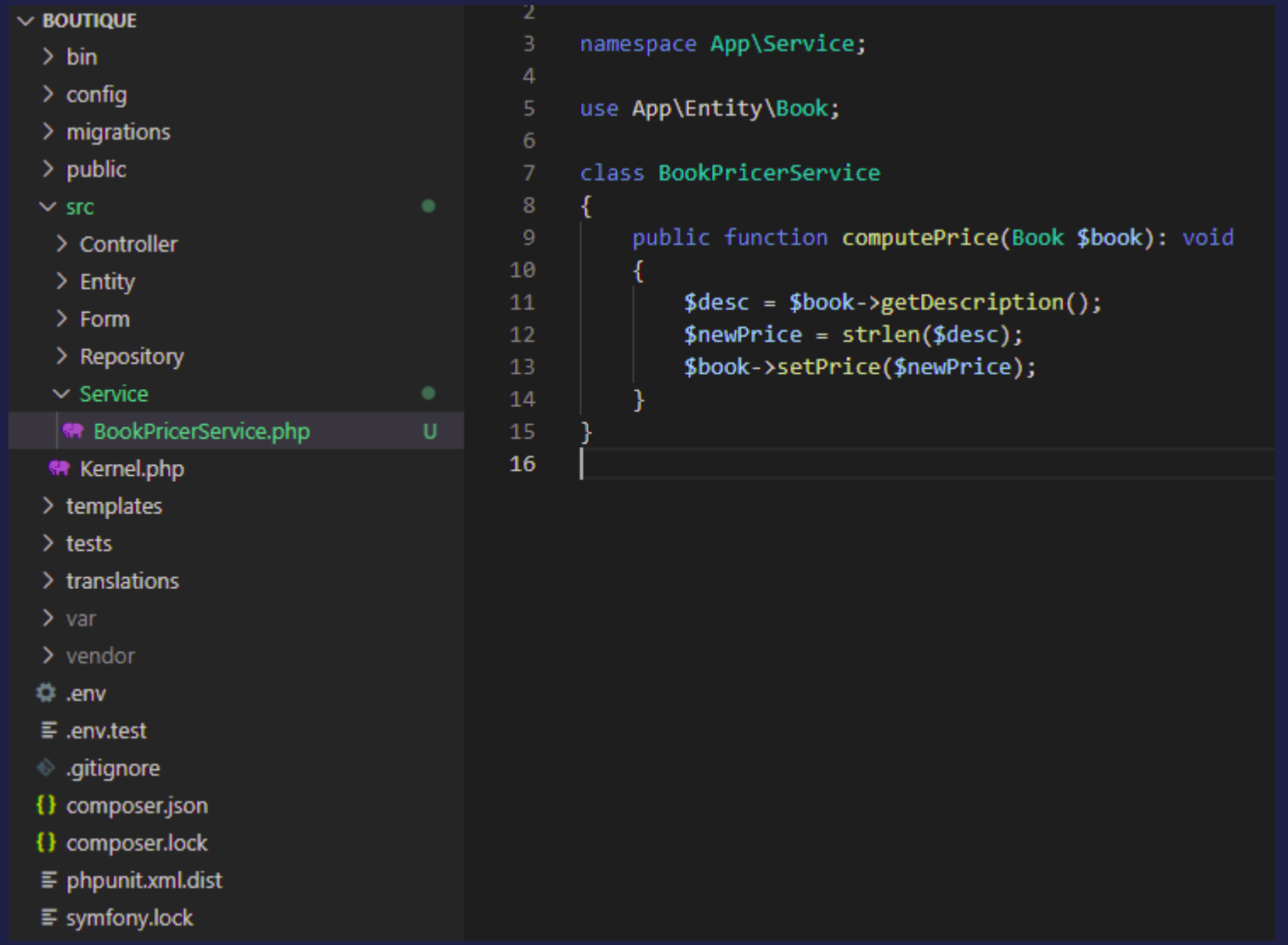

```
php bin/console make:entity Book
```

```
php bin/console make:crud Book
```

XWIKI NATIS

UN SERVICE ?

XDNATIS


BOUTIQUE
bin
config
migrations
public
src
Controller
Entity
Form
Repository
Service
BookPricerService.php
U
Kernel.php
templates
tests
translations
var
vendor
.env
.env.test
.gitignore
composer.json
composer.lock
phpunit.xml.dist
symfony.lock
namespace App\Service;
use App\Entity\Book;
class BookPricerService
{
public function computePrice(Book $book): void
{
$desc = $book->getDescription();
$newPrice = strlen($desc);
$book->setPrice($newPrice);
}
}

XONATIS

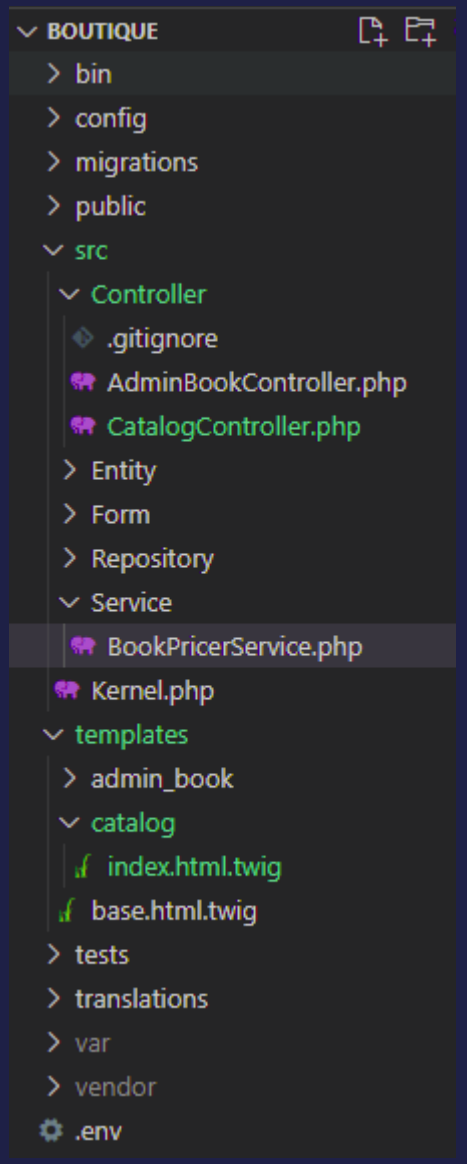
XWIKI NATIS

1. Service de panier
2. Service de paiement
3. Service d’upload de fichiers

XNATIS

PAGE CATALOGUE

XDNATIS




```
php bin/console make:crud Book
```

XWIKI NATIS

PAGE PANIER

XWIKI NATIS

SERVICE PANIER

XDNATIS

```php
public function get()
{
    return $this->sessionInterface->get('cart', [
        'elements' => [],
        'total' => 0.0
    ]);
}
```

```php
public function clear()
{
    $this->sessionInterface->remove('cart');
}
```

XDNATIS

```php
public function add(Book $book)
{
    $cart = $this->get();
    $bookId = $book->getId();

    if (!isset($cart['elements'][$bookId]))
    {
        $cart['elements'][$bookId] = [
            'book' => $book,
            'quantity' => 0
        ];
    }

    $cart['total'] = $cart['total'] + $book->getPrice();
    $cart['elements'][$bookId]['quantity'] = $cart['elements'][$bookId]['quantity'] + 1;

    $this->sessionInterface->set('cart', $cart);
}
```

XDNATIS

```php
public function remove(Book $book)
{
    $cart = $this->get();
    $bookId = $book->getId();

    if (!isset($cart['elements'][$bookId]))
    {
        return;
    }

    $cart['total'] = $cart['total'] - $book->getPrice();
    $cart['elements'][$bookId]['quantity'] = $cart['elements'][$bookId]['quantity'] - 1;

    if ($cart['elements'][$bookId]['quantity'] <= 0)
    {
        unset($cart['elements'][$bookId]);
    }

    $this->sessionInterface->set('cart', $cart);
}
```

XWIKI SAS

XWIKI NATIS

PAGE PAIEMENT

XONATIS

XONATIS

XWIKI NATIS

SERVICE PAIEMENT

XDNATIS

```php
public function __construct(CartService $cartService)
{
    $this->cartService = $cartService;
    $this->stripe = new StripeClient('sk_test_[redacted]');
}
```

XDNATIS

```php
public function create(): string
{
    $cart = $this->cartService->get();
    $items = [];
    foreach ($cart['elements'] as $bookId => $element)
    {
        $items[] = [
            'amount' => $element['book']->getPrice() * 100,
            'quantity' => $element['quantity'],
            'currency' => 'eur',
            'name' => $element['book']->getTitle()
        ];
    }
    $protocol = $_SERVER['HTTPS'] ? 'https' : 'http';
    $successUrl = $protocol . '://' . $_SERVER['SERVER_NAME'] . '/payment/success/{CHECKOUT_SESSION_ID}';
    $cancelUrl = $protocol . '://' . $_SERVER['SERVER_NAME'] . '/payment/failure/{CHECKOUT_SESSION_ID}';

    $session = $this->stripe->checkout->sessions->create([
        'success_url' => $successUrl ,
        'cancel_url' => $cancelUrl,
        'payment_method_types' => [
            'card'
        ],
        'mode' => 'payment',
        'line_items' => $items
    ]);
    return $session->id;
}
```

XONATIS

XDNATIS

SERVICE UPLOAD

XDNATIS

```php
public function __construct($targetDirectory, SluggerInterface $slugger)
{
    $this->targetDirectory = $targetDirectory;
    $this->slugger = $slugger;
}

public function upload(UploadedFile $file)
{
    $originalFilename = pathinfo($file->getClientOriginalName(), PATHINFO_FILENAME);
    $safeFilename = $this->slugger->slug($originalFilename);
    $fileName = $safeFilename.'-'.uniqid().'.'.$file->guessExtension();

    try {
        $file->move($this->getTargetDirectory(), $fileName);
    } catch (FileException $e) {
        // ... handle exception if something happens during file upload
    }

    return $fileName;
}

public function getTargetDirectory()
{
    return $this->targetDirectory;
}
```

XDNATIS

```php
$builder
    ->add('title')
    ->add('price')
    ->add('description')
    ->add('category')
    ->add('image', FileType::class, [
        'mapped' => false
    ])
;
```

```php
if ($form->isSubmitted() && $form->isValid()) {
    $imageFile = $form->get('image')->getData();
    if ($imageFile) {
        $imageFilename = $fileUploader->upload($imageFile);
        $book->setImageFilename($imageFilename);
    }
```

XONATIS

XONATIS

XONATIS