esprit
Se former autrement
HONORIS UNITED UNIVERSITIES

Introduction au Framework Symfony

UP Web

AU: 2022/2023

United Nations Educational, Scientific and Cultural Organization
UNESCO Chair "Project-based learning" ESPRIT School of engineering, Tunisia
uniTwin
EUR-ACE® Délivrée par la Commission des Titres d'Ingénieur
CONFERENCE DES GRANDES ÉCOLES
cdio

Plan

- Introduction
- Framework
- Framework et CMS
- Symfony 5 : Introduction
- Symfony 5 : Prise en main
  - L'architecture MVC
  - Mise en place de l'environnement
  - Déroulement d'une application Symfony 5
  - Créer un Controleur
  - Créer une route

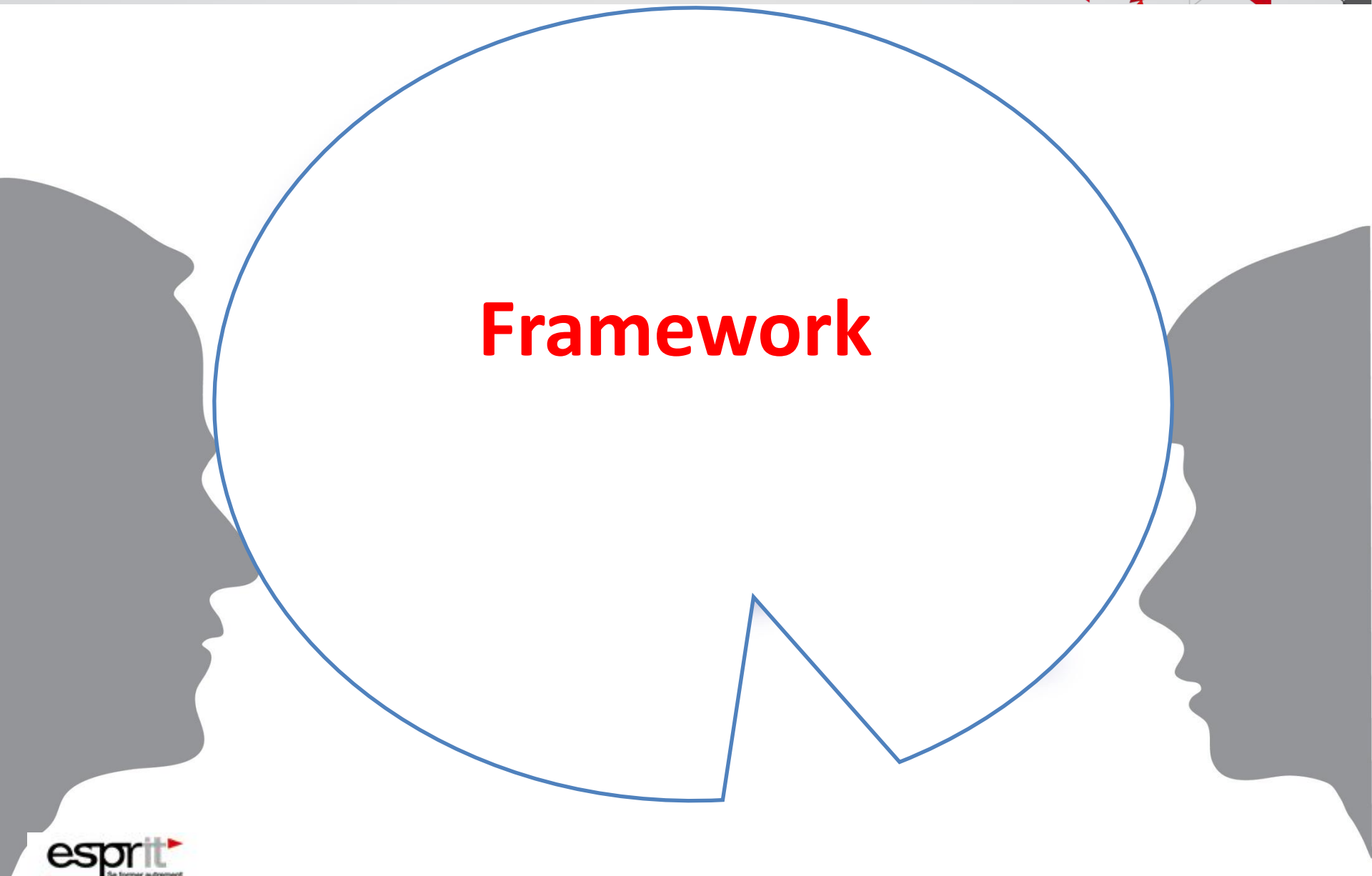
Objectifs du cours

- A la validation de ce chapitre l'étudiant sera capable de :

✔ Introduire les fondements théoriques du framework Symfony 5.

✔ Démarrer un projet avec Symfony 5

Introduction



C’est quoi un Framework ?

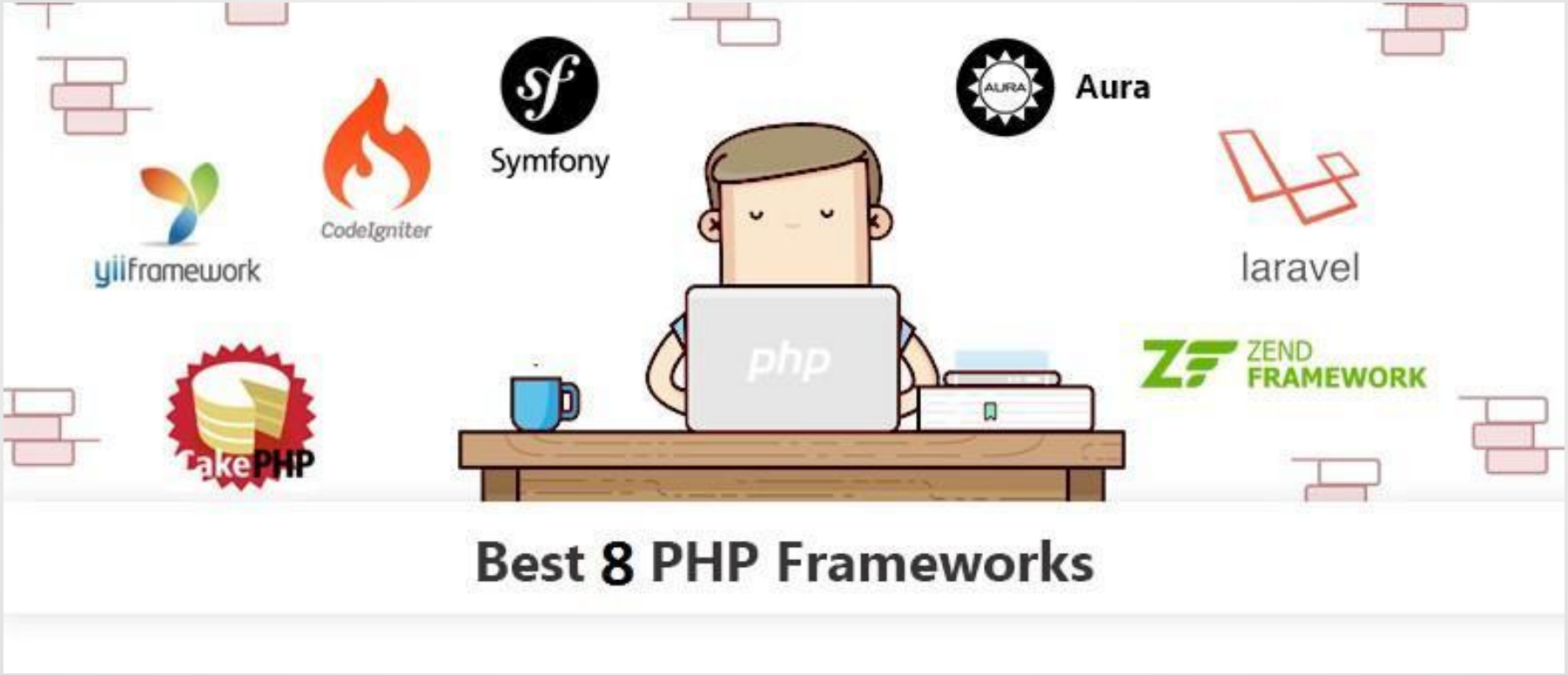
Framework

Cadre de travail

- Structurer votre projet
- Apporter un ensemble d'éléments qui définissent le squelette d'une application
- Offrir des composants et bibliothèques réutilisables
- Offrir une liberté dans la réalisation technique de l’application

C’est quoi un Framework ?

Panorama Framework PHP



C’est quoi un CMS ?

- **CMS/ Content Management System**

- Système de gestion de contenu, permettant à un ou plusieurs utilisateurs de créer, gérer le contenu d'un site Web

- **Composé** d’un « noyau » avec les fonctionnalités de base (pages, gestion des utilisateurs, etc.) sur lequel se greffent ***des modules*** (ou extensions) spécifiques comme par exemple un forum, une newsletter, un annuaire, un glossaire, des actualités…

C’est quoi un CMS ?

CMS PHP

CMS et Framework

Le choix entre un CMS et un Framework dépend du

- Besoin du développeur

CMS

Fonctionnalités standards:

- Gestion de panier
- Gestion du back office

Framework

- Fonctionnalités sur mesure
- Pas d'architecture globale du site Web
- Pas de fonctionnalités standards
- Liberté de créer ses propres fonctionnalités

Pourquoi utiliser un Framewok?

- **Portabilité** : L'abstraction de la base de données et du cache permet à **votre application d'être utilisée sur de nombreuses configurations de serveurs différents**

- **Temps de développement plus court** : Puisque vous n'êtes pas obligé de réécrire le code sur la gestion des utilisateurs, et même de l'authentification, de l'accès à la base de données et aux formulaires, **le temps de développement ce réduit considérablement**

- **Sécurité des applications** : Les fonctions de sécurité comme l'authentification et les autorisations sont gérées par le *framework*

Pourquoi utiliser un Framewok?

- **Soutenue par la communauté** : Les *frameworks* ont des forums, des listes de diffusion et des canaux *IRC (Internet Realy Chat)* pour les soutenir

- **Plugins et modules** : **Un bon nombre de membres de la communauté développent des plugins et des modules** que vous pouvez télécharger et utiliser dans votre application

- **Règles de codage stricts** : La plupart des *frameworks* vous forcent à suivre des principes de codage, notamment le modèle MVC

Critères de comparaison

- **Taille des projets** : les petits projets peuvent être développer avec un framework ultralight
- **Documentation** : une documentation complète est un réel plus !
- **Performance** : certaines frameworks sont trop gourmands. Même le moindre « Hello World » peut nécessiter l'appel à plus de 100 fichiers différents
- **Communauté** : un forum actif sera synonymes de personne prête à vous aider en cas de problème
- **Évolutivité** : de quand date la dernière mise à jour

Et pourquoi Symfony et pas un autre ?

Comparaison des Frameworks

	Adapté au développement d'applications rapides *Ses principales caractéristiques :* ▪Pas de configuration nécessaire ▪Licence MIT ▪Entièrement Orienté Objet (OO)
	Est simple à utiliser, performant et d'une vitesse d'exécution remarquable. *Ses principales caractéristiques :* ▪Simplicité de prise en main ▪Nombreuses librairies ▪Complètement orienté objet
	Laravel est le framework PHP qui est principalement connu pour son temps de développement réduit avec une approche de codage simple. *Ses principales caractéristiques :* ▪Bonne documentation ▪Services postaux intégrés ▪Mise sur le marché plus rapide

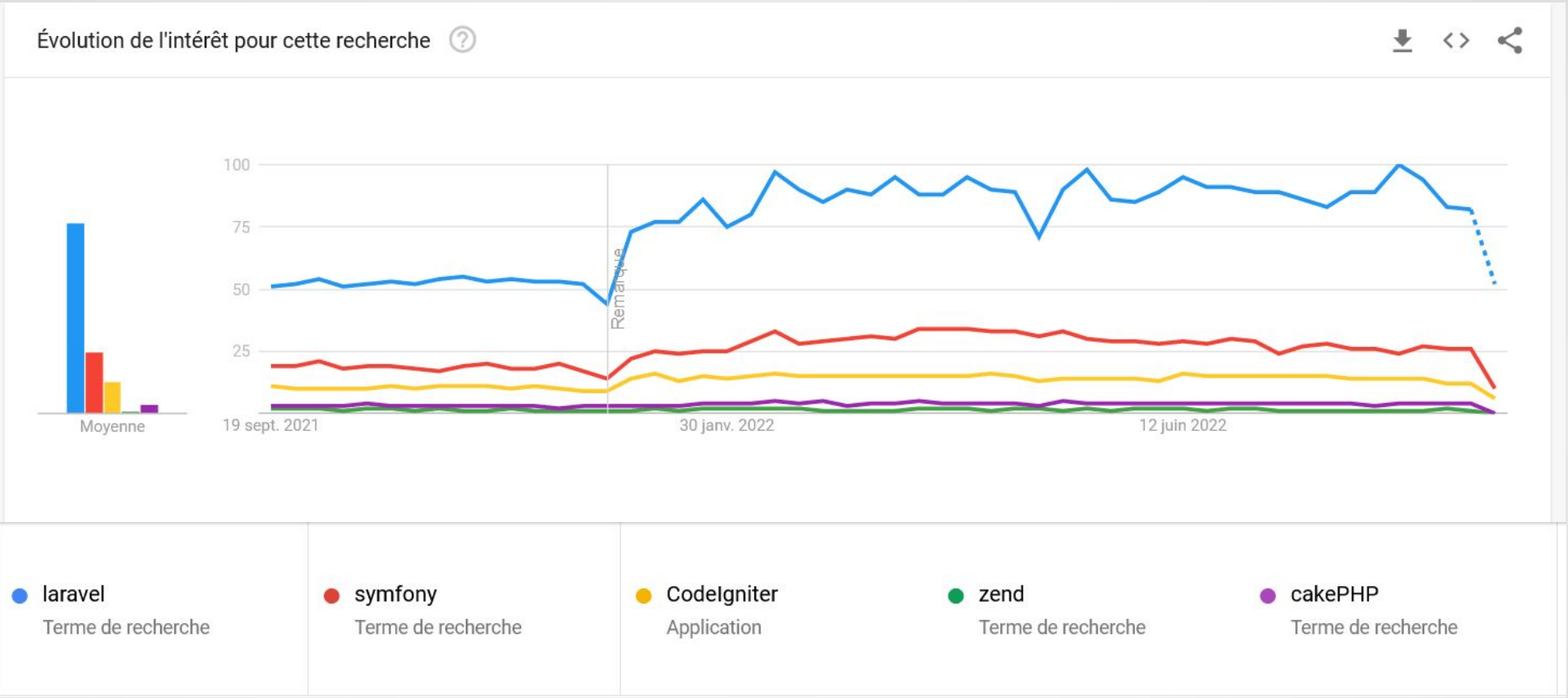
Et pourquoi Symfony et pas un autre ?

Comparaison des Frameworks

	Considéré comme une grosse bibliothèque de fonctionnalités plutôt qu'un framework Il n'est pas très simple à comprendre *Ses principales caractéristiques :* ▪Conception totalement orientée objet ▪Puissant, extensible et modulaire ▪Fortement adapté à l'environnement professionnel ▪Alliés et contributeurs hors normes : Microsoft, Google…
Symfony	Il est spécialement dédié à la conception d'applications moyennes à très lourdes *Ses principales caractéristiques :* ▪Licence **MIT** ▪Extensible et modulaire ▪Supporte **Ajax**

Et pourquoi Symfony et pas un autre ?

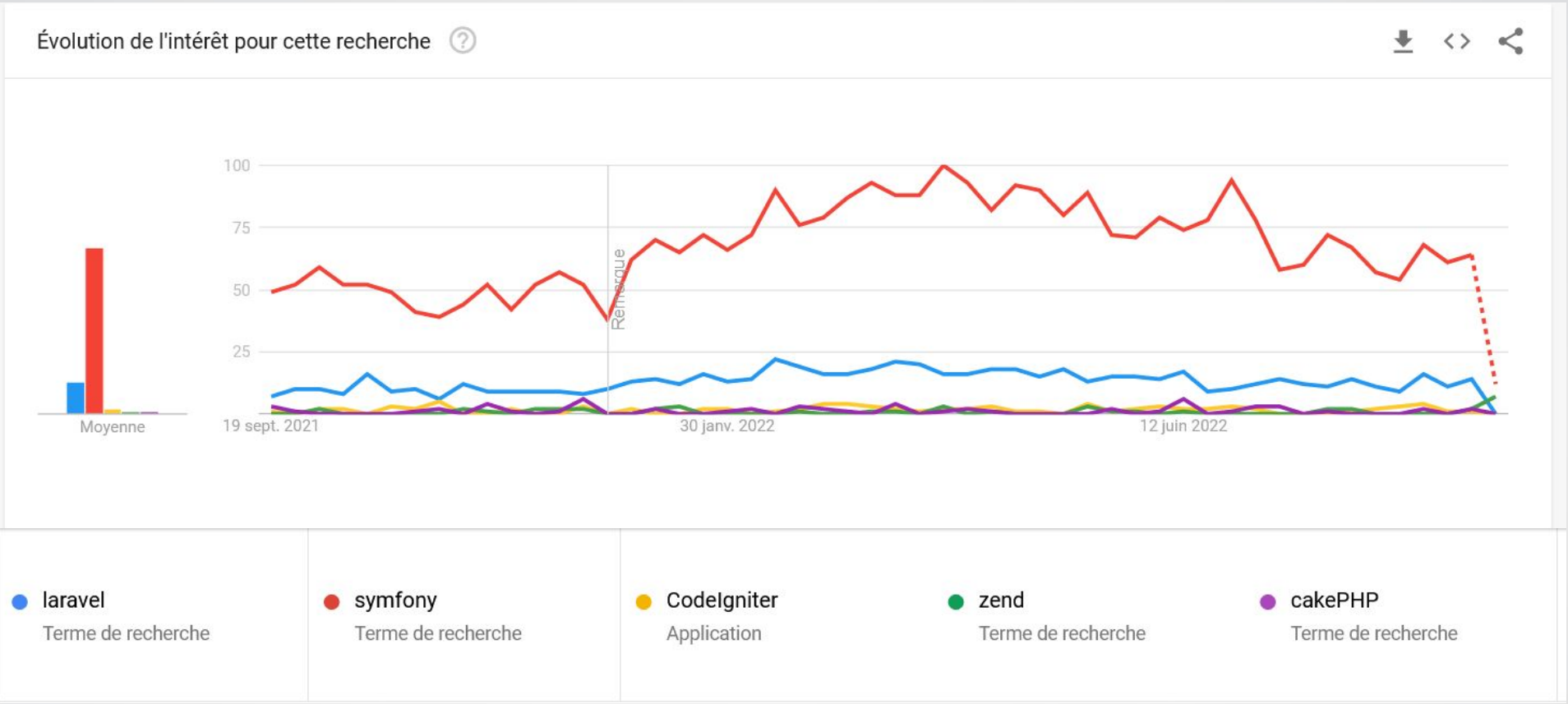
Comparaison des Frameworks (International)



Comparatif des statistiques Google sur les recherches mondiales ayant pour mot clé d'un framework PHP

Et pourquoi Symfony et pas un autre ?

Comparaison des Frameworks (France)



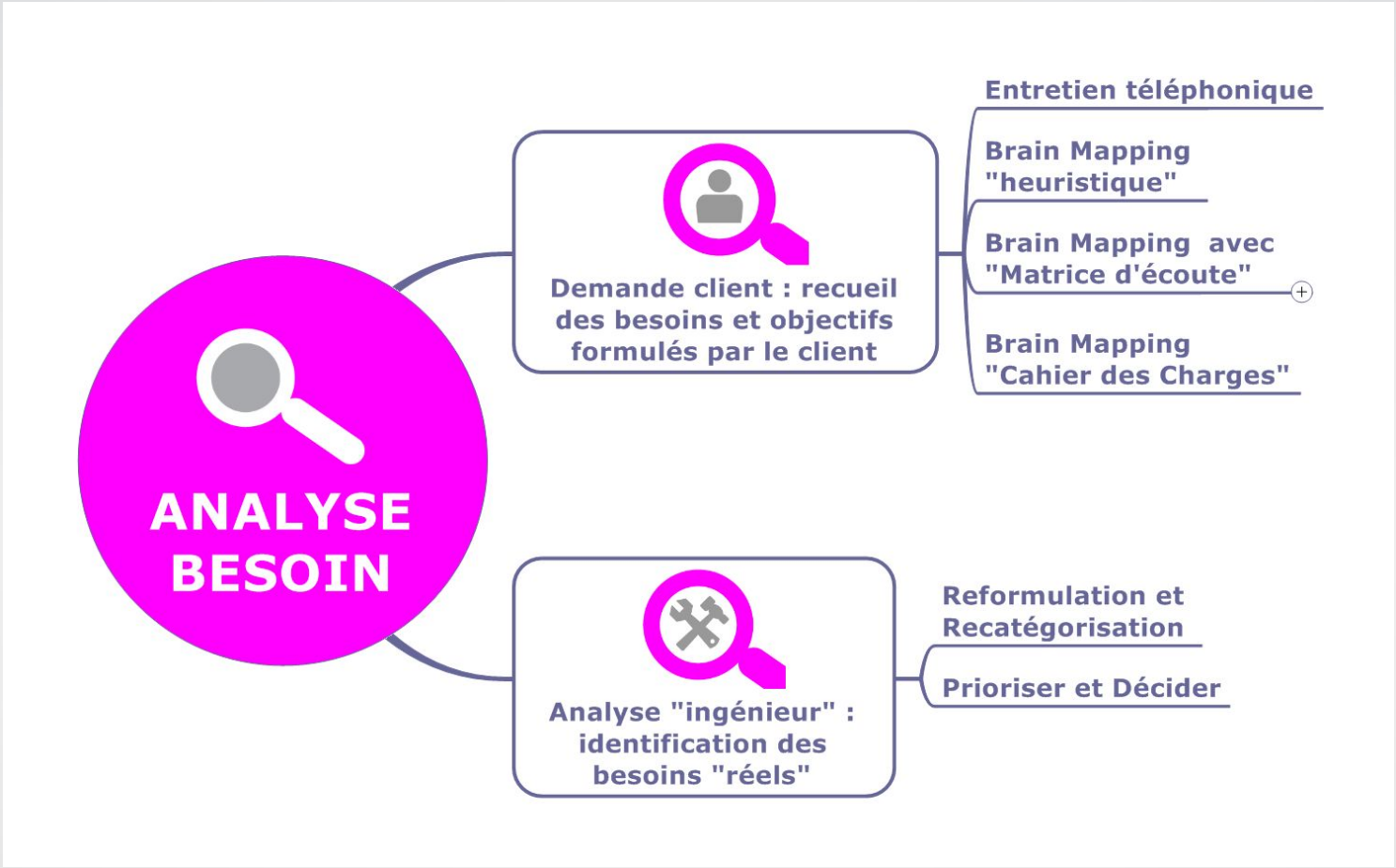
Comparatif des statistiques Google sur les recherches mondiales ayant pour mot clé d'un framework PHP

Le meilleur des Framework ?

Quel est le meilleur des framework?

La mauvaise question

La bonne question !



Quel est le framework le mieux adapté à mon besoin ?

Symfony : Présentation

Qui est derrière tout ça ?

SensioLabs
Créateur de Symfony

Fabien Potencier

Une grande réputation

- Lancé en 2005, Symfony est aujourd'hui un framework stable connu et reconnu à l'international.
- Symfony dispose aussi une communauté active de développeurs, intégrateurs, utilisateurs et d'autres contributeurs qui participent à l'enrichissement continu de l'outil

Qui utilise Symfony ?

De nombreux sites et applications de toutes tailles et de tous les types !

C'est par exemple le cas de Yahoo! , Dailymotion, Drupal8

Symfony : Présentation

De la version 1.x à la version 2.x à la version 3.x à la version 4.x à la version 5.x

Version Symfony	Version de php utilisée	Utilisation d'un système de Template TWIG	Utilisation des bundles	Avantages
Symfony 1.x	**PHP** ≥ 5.2.4	Non	Non	-ayant connu un succès notable
Symfony 2.x	**PHP** ≥ 5.3.3	Oui	Oui	-Grâce à la nouvelle gestion des namespaces, Symfony2 a gagné de la rapidité. -Nouveautés dans le fichier security.yml -Utilisation des annotations.
Symfony 3.x	**PHP** ≥ 5.5.9	Oui	Oui	-Plus standard -Plus découplé et plus que jamais réutilisable
Symfony 4.x /5.x	**PHP** ≥ 7.4.9	Oui	Oui	-Version minimaliste et plus légère de Symfony 3 -Libre d'installer des packages supplémentaires. -Plus besoin de configurer ses bundles -Déploiement rapide et simple. -Installation via composer (Composer va faire le nécessaire pour créer un dossier du projet et télécharger les dépendances associées) -Génération de migration -Symfony Flex devient obligatoire sur un projet Symfony 4.

Composer

Composer est un outil de gestion de dépendances PHP, il permet de :

- Installer des dépendances
- Mettre à jour des dépendances

Un fichier intitulé “composer.json” contient la liste des dépendances du projet.

- Lien d’installation:https://getcomposer.org/download/

CLI: Commande Ligne Interface

- Le CLI de Symfony est un outil de développement qui vous aide à créer, exécuter et gérer vos applications Symfony directement à partir de votre terminal.

- Il est Open-Source, fonctionne sur macOS, Windows et Linux, et vous n'avez qu'à l'installer une seule fois dans votre système.

CLI: Commande Ligne Interface

Pour l'installer :

1- Il faut installer Scoop (installeur en ligne de commande pour Windows) via la commande

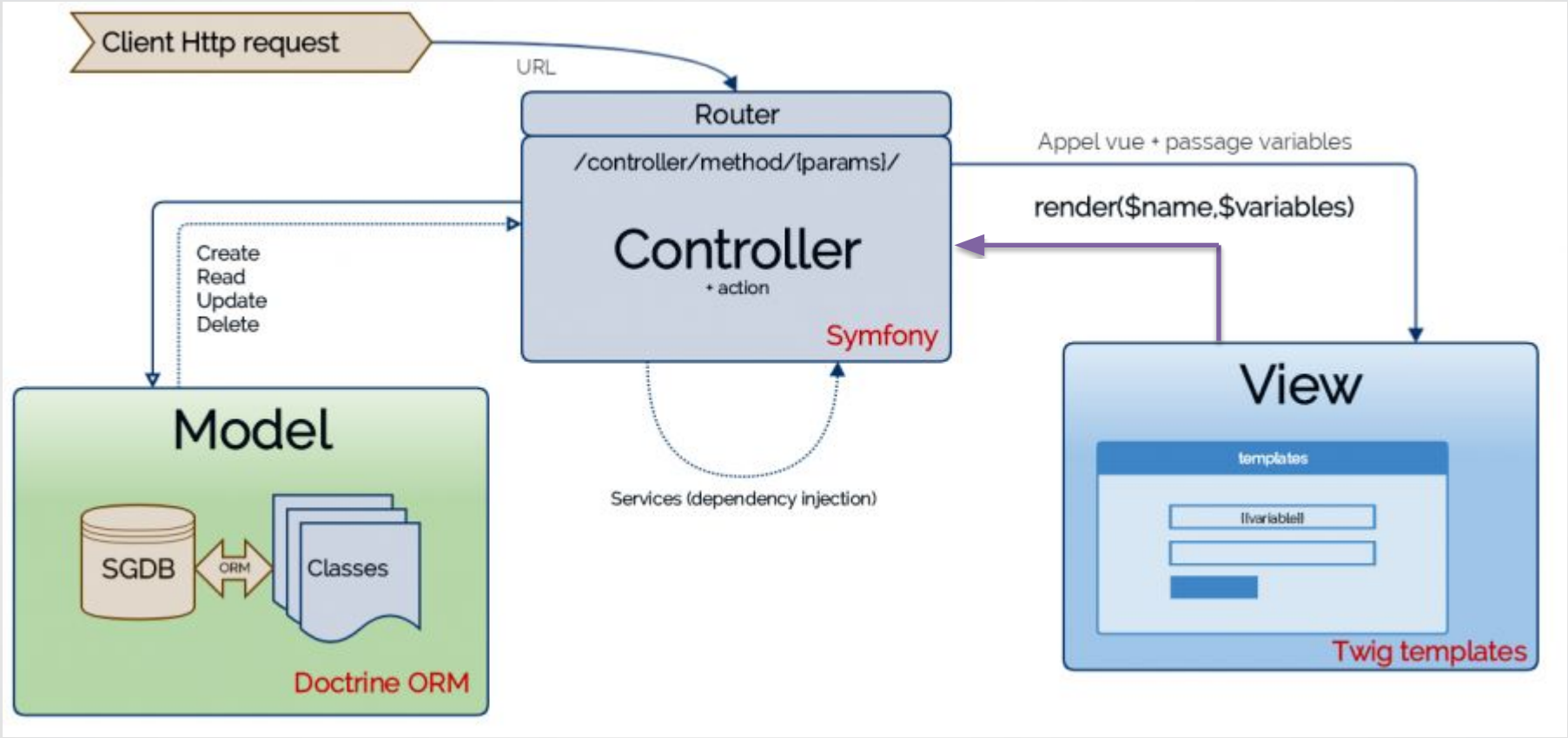
irm get.scoop.sh | iex ou bien

iex (new-object net.webclient).downloadstring('https://get.scoop.sh')

2- Taper la commande : **scoop install symfony-cli**

L'architecture MVC

Symfony est un framework basé sur le modèle MVC (Model View Controller).



L’architecture MVC

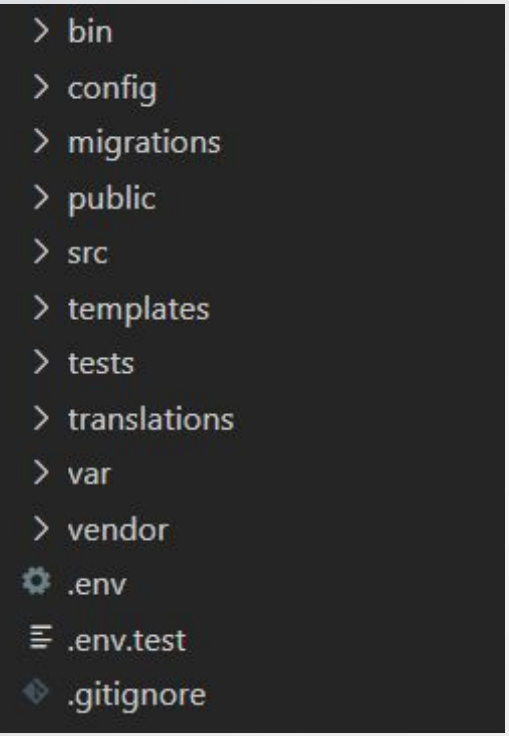
- Le contrôleur : Permet de traiter la requête de l’utilisateur et de générer une réponse
- Le modèle : Permet de gérer les données
- La vue : Permet d'afficher les données qu'elle a récupérées auprès du modèle.

 le modèle MVC va nous aider à séparer les requêtes de la base de données (Modèle), de la logique relative au traitement des demandes (Contrôleur), et à l’affichage (Vue).

Atelier 2.1:

Mise en place de l'environnement de Symfony 5 et création du premier projet

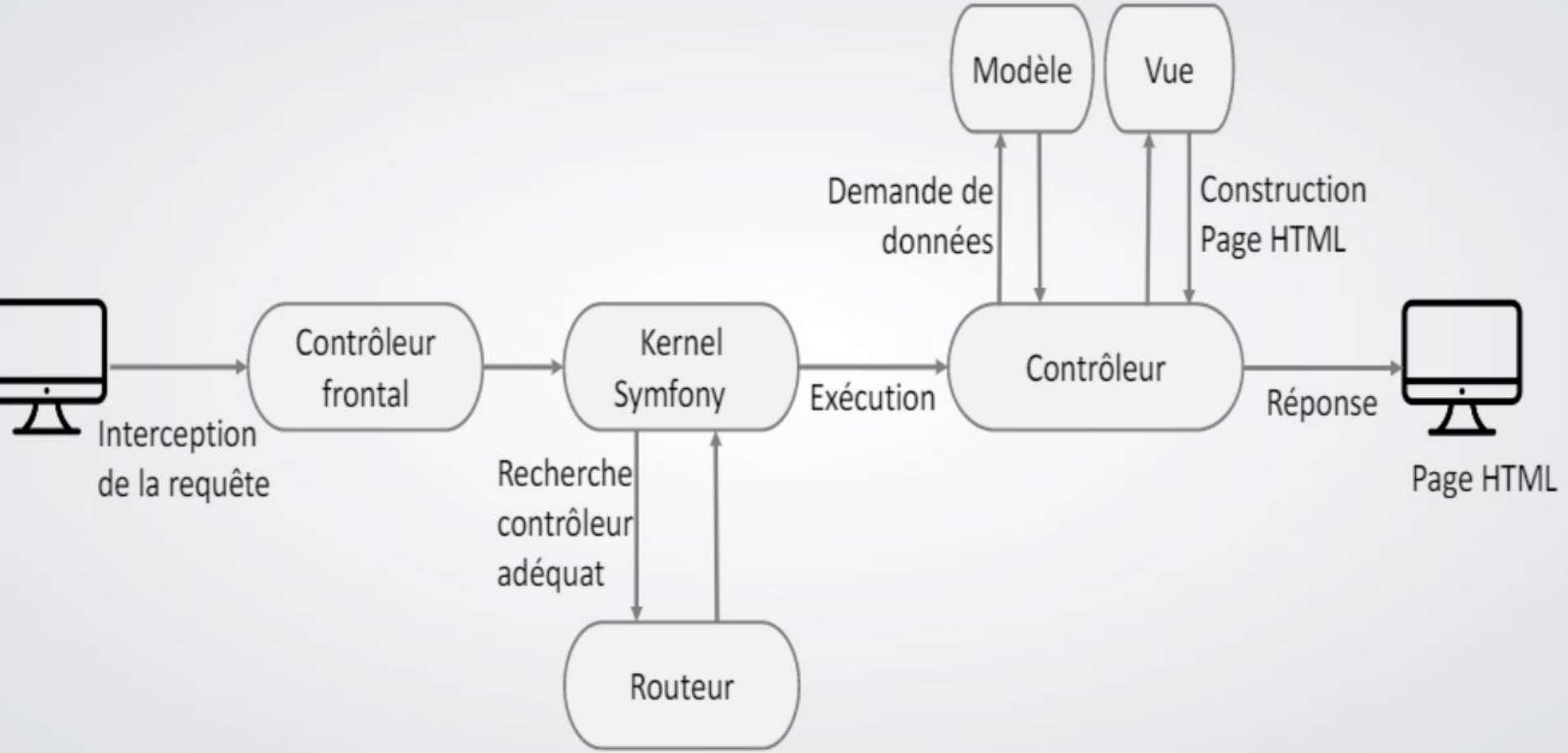
L’architecture d’un projet Symfony 5



bin	Contient les fichiers executables (exemple bin/console)
config	Ce dossier va contenir la configuration de notre application Symfony. C’est ici que nous allons configurer les routes, les services, …
public	Ceci est le dossier d’entré de notre application, mais aussi le dossier public, nous allons y mettre tout les fichiers accessibles au public. Il contient notamment le contrôleur frontal de Symfony.
src	Les fichiers contenant toute la logique de notre projet
templates	Toutes les pages qui vont être affichées à l'écran vont être ici.
test	Contient les tests automatiques (exemple Unit tests)
vendor	Le dossier contenant les dépendances nécessaire au fonctionnement de notre projet.
.env	Fichier contenant la configuration de l'environnement d'exécution de notre code

Principe de fonctionnement d'une application Symfony 5

Symfony suit un schéma simple et identique pour toutes les requêtes



Flux applicatif d'une requête

Soit un visiteur qui veut accéder à la page « /student » pour afficher la liste des étudiants

1. Le visiteur demande une URL qui est la page « /student »
2. Cette URL est traitée par le « Contrôleur frontal » qui va charger le « Kernel » et la lui transmet.
3. Le « Kernel » appel le « routeur » afin de lui indiquer quel « contrôleur » exécuter pour l'URL « /student ».
4. Le « Routeur » cherche le « contrôleur » correspondant et envoie au « Kernel » qu'il faut exécuter le « ControllerStudent ».
5. Le « Kernel » appel le « ContrôleurStudent »
6. Ce « Contrôleur » demande au « Modèle » la liste des étudiants, puis la donne à la « Vue » pour qu'elle construise la page HTML.
7. Le contrôleur transmet la page HTML au visiteur .

Contrôleur frontal

- C’est le point d’entrée unique de l’application

 C'est le fichier par lequel passent toutes vos pages.

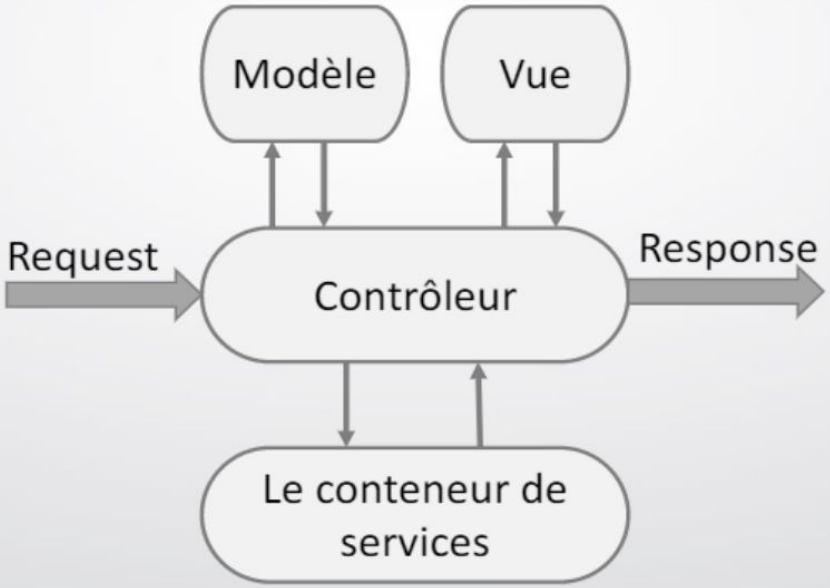
- Le contrôleur frontal se situe dans le répertoire «public », on l’appelle aussi « index.php » .

- Le contrôleur frontal s'occupe de l'envoi de la demande (requête) en appelant le noyau (kernel) de Symfony pour avoir finalement la réponse.

Contrôleur

- Un contrôleur est un élément indispensable de l'architecture MVC.

- Il reçoit une requête et il interagit avec les différents composants d'une application Symfony pour renvoyer une réponse (un objet Response Symfony).



Contrôleur

- « Request » correspond aux données de la requête utilisateur
- « Response » correspond à la réponse retournée par le contrôleur.
- La réponse peut être :

✔ une page HTML (Twig).
✔ un document XML
✔ un tableau JSON sérialisé
✔ etc…

Contrôleur

- Un contrôleur est une classe PHP qui hérite de la classe AbstractController
- Les Fonctions de base de la classe AbstractController

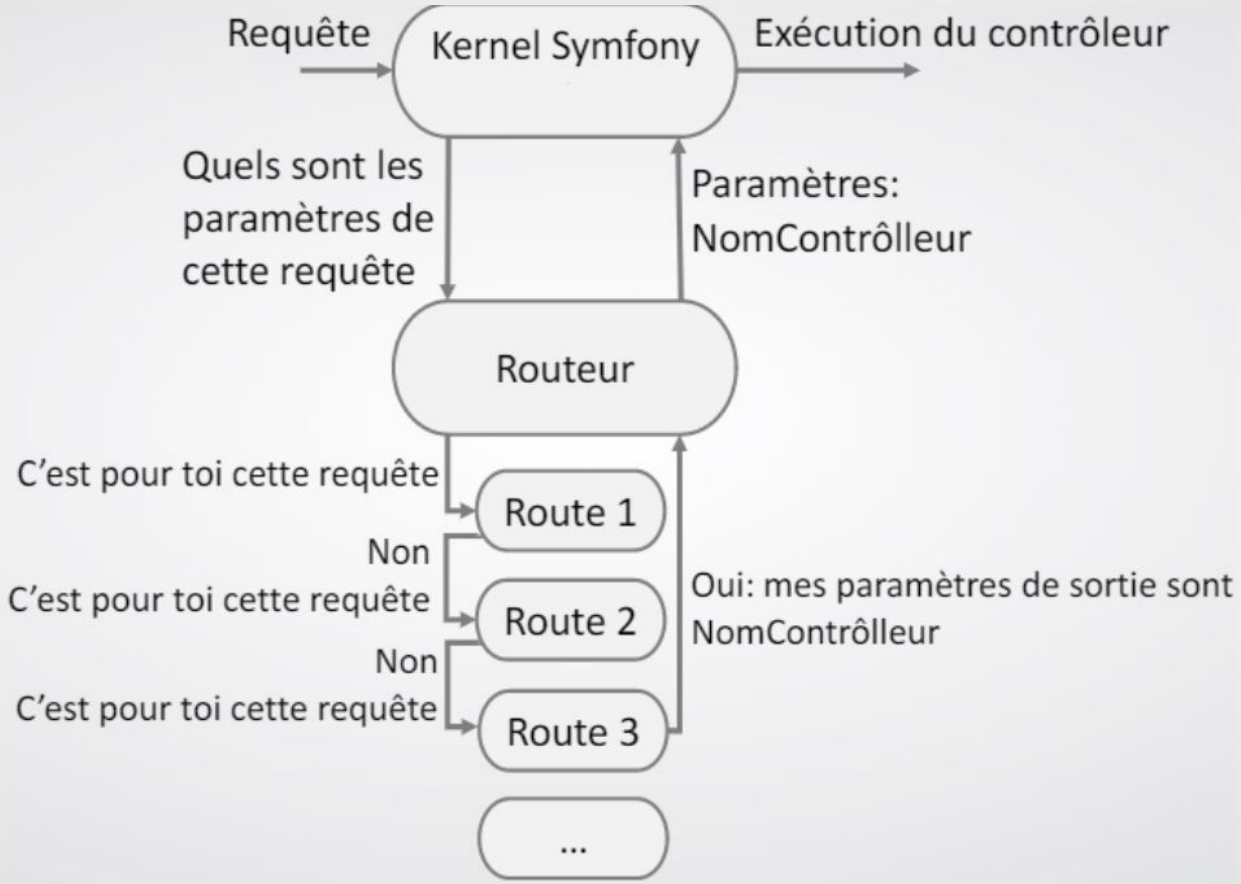
Méthode	**Fonctionnalité**	**Valeur de retour**
generateUrl()	Génère une URL à partir de la route	String
forward(String Action,array ()$parameters)	Forward la requête vers un autre contrôleur	Response
Redirect(string $url, int $statut	Redirige vers une url	RedirectResponse
RedirectToRoute(string $route, array $parameters)	Redirige vers une route	Rsponse
Render(string $view,array $parameters)	Affiche une vue	Response
Get (string $id)	Retourne un service à travers son id	objet
createNotFoundException (String $message)	Retourne une NotFoundException	NotFoundException

Routage

- La route permet de faire correspondre une URL et une action d’un contrôleur
- L’objectif du routeur est de chercher la route qui correspond à l'URL appelée, et de retourner les paramètres de cette route.

Routage

Principe de fonctionnement du routage



Symfony Flex

- Symfony Flex est un plugin Composer.
- Il permet d'exécuter des tâches sur certaines commandes Composer (require, update et remove).
- Il permet d’automatiser l’installation et la suppression des dépendances en fournissant une configuration par défaut

  - sans avoir à aller lire le fichier pour trouver quelle configuration écrire ou autre tâche à effectuer

Symfony Flex

Exemple: Nous souhaitons installer une API REST
Il faut lancer donc la commande «composer require api»

- l'application envoie une requête au serveur Symfony Flex avant d'installer le paquetage avec Composer.

- « Symfony Flex » va se charger lui-même:

✔ D'enregistrer le bundle dans le « Kernel »
✔ De fournir une configuration par défaut
✔ De charger les routes nécessaires
✔ etc.

 La commande **«composer require api»** suffi pour mettre en place une API REST fonctionnelle

Symfony Flex

- Flex laisse une trace des recettes qu'il a installées dans un fichier appelé « **symfony.lock** ».

- Ces recettes contiennent des instructions pour indiquer à Flex ce qu'il doit faire pour chaque paquet.

- Une recette définit plusieurs informations à savoir les alias.

Symfony Flex

- Les « alias » permettent à Flex d'installer des paquetages en utilisant des nom courts et facile à retenir.
- Si on revient à l'exemple précédent:

composer require api

- Le paquetage «api» n'existe pas pour composer. Dans ce cas, Flex intervient pour détecter si le nom du paquetage correspondre à un allias

 Si oui, le composer installe le paquetage correspondant

Exemple d'allias:

✔ Admin
✔ Log
✔ Orm
✔ etc…

Symfony Flex

Symfony Flex permet de:

- Configurer automatiquement le paquetage dans le projet.
- Supprimer toutes les configurations relatives au paquetage, quand vous le supprimez

MakerBundle

- Permet de créer des commandes vides, des contrôleurs, des classes de formulaire, des tests etc…
- Ce bundle existe par défaut, pas besoin de l'installer
- S'il n'existe pas, la commande suivante permet de l'installer:

Composer require maker

MakerBundle

- Quelques exemples de l’utilisation du makerBundle:

- make:contoller pour créer un nouveau contrôleur
- make:entity pour créer une nouvelle entité
- Etc…

Créer un Contrôleur

Pour générer un contrôleur via la console, vous pouvez utiliser le MakerBundle

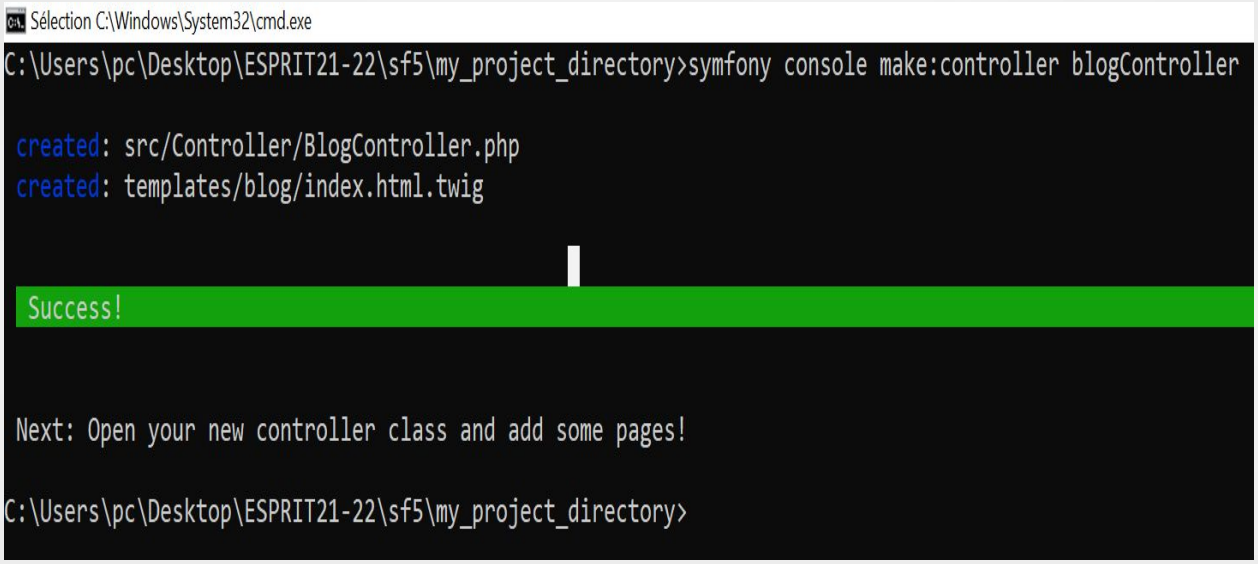
php bin/console make:controller NomController

ou

symfony console make:controller NomController

Créer un Contrôleur

Exemple: Nous demandons à Symfony de nous créer le contrôleur BlogController



2 fichiers ont été créés:

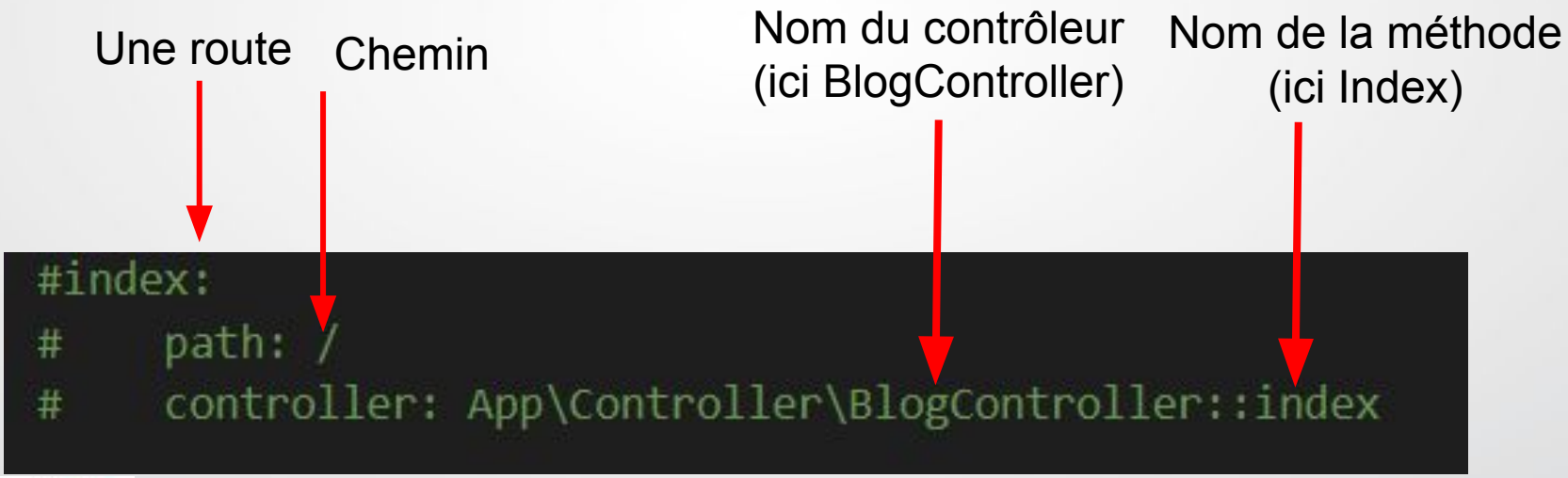
- BlogController.php dans src/Controller
- index.html.twig dans templates/blog,

Créer une Route

- Il existe deux méthodes pour créer une route:

1ère méthode: YML

- Nous créons une route qui s'appelle index dont le chemin est «/» et qui a comme contrôleur la méthode «index» qui se trouve dans la classe « BlogController ».



Créer une Route

2ème méthode: Annotation

Au lieu de placer toutes les routes de l'application dans un seul fichier, il peut-être plus souple d'utiliser les annotations dans le même code du contrôleur

Chemin de la route Nom de la route

Créer une Route

3ème méthode: Annotation

Chemin de la route

Nom de la route

```
#[Route('/blog', name: 'app_blog')]
public function index(): Response
{
    return $this->render('blog/index.html.twig', [
        'controller_name' => 'BlogController',
    ]);
}
```

Créer une Route

Le routage avec paramètres

- Nous utilisons une annotation avec un paramètre variable « id » comme ci-dessous

```php
/**
 * @Route("/view/{id}", name="view")
 */
public function view($id)
{
    // $id vaut 5 si l'URL appelée est /view/5
    return new Response("Affichage d'id : ".$id);
}
```

- "{id}" est la variable qui permettra d'avoir une route dynamique.
- La méthode view() prend en paramètre la variable « id» pour l'afficher

Atelier 2.2:
création d’un contrôleur + routing

Références

http://www.finalclap.com/faq/422-php-comparatif-framework

https://www.codeur.com/blog/developpement-web-meilleurs-frameworks/

http://blog.nicolashachet.com/technologies/php/pourquoi-utiliser-un-framework-php/

http://symfony.com/

http://symfony.com/doc/current/quick_tour/the_architecture.html

https://symfony.com/doc/current/introduction/http_fundamentals.html

http://www.lafermeduweb.net/tutorial/symfony-mvc-les-modeles-p34.html

http://www.javavillage.in/view-topic.php?tag=difference-between-mvc1-mvc2

Merci pour votre attention