# Cours Symfony 4

Mohamed CHEMINGUI

Institut Supérieur des Arts Multimédia de la Manouba

Symfony

isamm

# Table des matières

# 1. Introduction

Quand on pense au développement web, outre HTML, CSS et JavaScript, le langage PHP est l'un des noms qui viennent à l'esprit.
Les programmeurs PHP se tournent souvent vers un framework PHP pour composer leur code. Découvrons ce que sont les frameworks PHP et pourquoi ils sont utilisés ?

## 1.1. *Qu'est-ce qu'un framework PHP ?*

Un framework PHP est une plate-forme (ensemble d'outils) permettant de créer des applications web en PHP.
Les frameworks PHP fournissent des bibliothèques de code pour les fonctions les plus courantes.

## 1.2. *Pourquoi utiliser un framework PHP ?*

L'objectifs d'un framework c'est d'apporter des composants de bases qui constituent toutes les applications afin de faciliter le processus de développement d'application/site web.
Il existe de nombreuses bonnes raisons d'utiliser des frameworks PHP plutôt que de coder à partir de zéro.

**Un développement plus rapide / Moins de code à écrire**
Comme les frameworks PHP disposent de bibliothèques et d'outils intégrés, le temps nécessaire au développement est moindre.

L'utilisation de fonctions intégrées au framework permet de ne pas avoir à écrire autant de code original.

**Bibliothèques pour les tâches communes**
De nombreuses tâches que les développeurs devront effectuer dans les applications web sont courantes.
Il s'agit par exemple de la validation des formulaires, du nettoyage des données et des opérations CRUD (Create, Read, Update, and Delete).
Plutôt que d'avoir à écrire vos propres fonctions pour ces tâches, vous pouvez simplement utiliser celles qui font partie du framework.

**Suivre les bonnes pratiques de codage**

Les frameworks PHP suivent généralement les meilleures pratiques de codage.
Par exemple, ils divisent le code en plusieurs répertoires selon la fonction.
Structure du répertoire par défaut de Symfony
Ils vous obligent à organiser le code d'une manière plus propre, plus nette et plus facile à maintenir.

**Plus sûr que d'écrire vos propres applications**

Il existe de nombreuses menaces à la sécurité de PHP, notamment les scripts cross-sites, les attaques par injection SQL ….
Si vous ne prenez pas les bonnes mesures pour sécuriser votre code, vos applications web PHP seront vulnérables.

L'utilisation d'un framework PHP ne remplace pas l'écriture d'un code sécurisé, mais elle minimise les risques d'exploitation par des pirates.

**Un meilleur travail d'équipe**

Les projets avec plusieurs développeurs peuvent se tromper s'il n'y a pas de clarté au sujet de :

- La documentation
- Les décisions du design
- Les normes du code

L'utilisation d'un framework :

- Définit des règles de base (conventions) claires pour votre projet.
- Favorise la collaboration entre les membres de l'équipe.

**Plus facile à entretenir**

Vous n'avez pas non plus à vous soucier de la maintenance du cœur du framework, puisque les développeurs le font pour vous.

## 1.3. *Prérequis utilisation framework PHP*

La première chose que vous devez connaître avant d'utiliser un framework PHP est PHP lui-même !
Si vous n'avez pas une bonne maîtrise du langage, vous aurez du mal à vous choisir un framework.

Ensuite, vous devez avoir construit vos propres applications PHP, afin de bien comprendre ce qui est nécessaire sur l'interface publique et l'administration.

Il est également indispensable de connaître le **PHP orienté objet**, car la plupart des frameworks PHP modernes sont orientés objet.
Assurez-vous de comprendre les concepts tels que les classes, les objets, l'héritage et les méthodes…

Comme de nombreuses applications web se connectent à une base de données, vous devez connaître les bases de données et la syntaxe SQL.
Chaque framework PHP a sa propre liste de bases de données supportées.

Il est utile de comprendre un modèle de cartographie relationnelle des objets (**Object-Relational-Mapping** ou **ORM**).
L'ORM est une méthode d'accès aux données d'une base de données utilisant une syntaxe orientée objet au lieu d'utiliser le langage SQL. Cela signifie que vous pouvez écrire vos requêtes de base de données dans un langage PHP familier, bien qu'il puisse arriver que vous souhaitiez utiliser SQL.

De nombreux frameworks PHP ont leur propre ORM intégré.

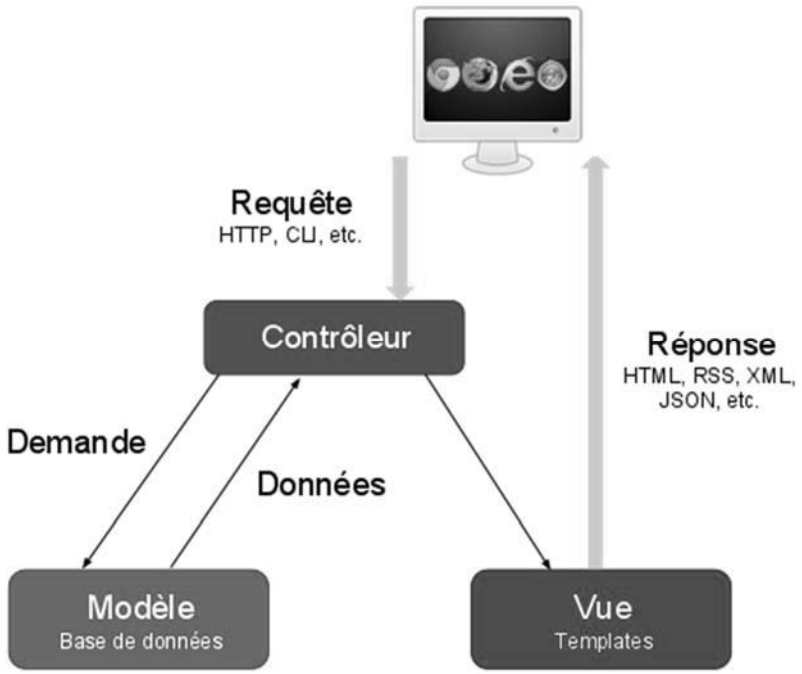
Il est utile de comprendre comment fonctionnent les serveurs web comme Apache.

Vous ferez probablement une grande partie de votre développement au niveau local, vous devez donc aussi connaître **localhost**.

## 1.4. *Architecture du contrôleur de vue de modèle*

Les frameworks PHP suivent généralement le modèle de conception du contrôleur de vue de modèle (Model View Controller ou MVC).

Ce concept sépare la manipulation des données de leur présentation.



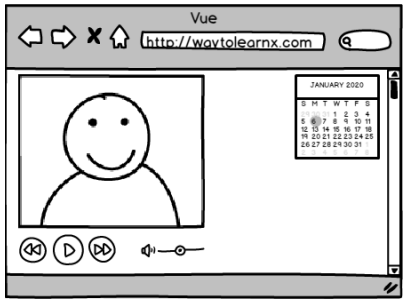

Le modèle MVC va nous aider à séparer les requêtes de la base de données (Model) de la logique relative au traitement des demandes (Controller) et au rendu de la présentation (View).

Les trois parties du MVC sont interconnectées :

- Le contrôleur accepte les entrées de l'utilisateur et met à jour le modèle et la vue.
- La vue affiche le modèle pour l'utilisateur.
-

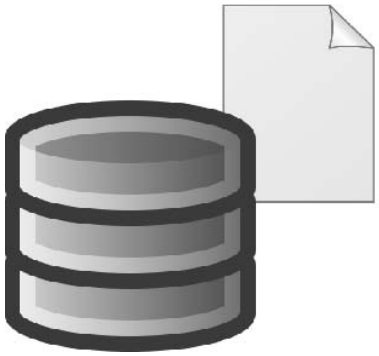
Bien que le MVC ne soit pas requis dans la conception des applications, de nombreux langages de programmation supportent l'architecture MVC, ce qui en fait un choix pour les développeurs.

### 1.4.1. Vue



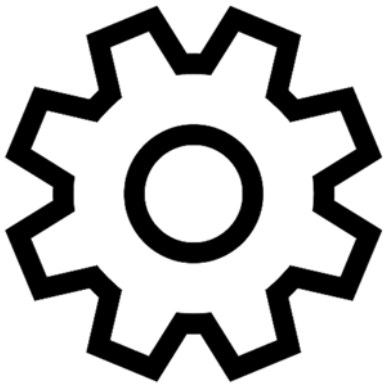

Une vue est un moyen d'afficher des objets dans une application. Par exemple, l'affichage d'une fenêtre ou des boutons ou d'un texte dans une fenêtre. Il comprend tout ce que l'utilisateur peut voir.
La vue est l'interface utilisateur. La vue permet à l'utilisateur d'afficher les données à l'aide d'un modèle et lui permet également de modifier les données.

### 1.4.2. Modèle



Un modèle contient les données utilisées par un programme. Il peut s'agir d'une base de données, d'un fichier ou d'un simple objet. Par exemple, un objet Client récupérera les informations de la base de données, les manipulera et mettra à jour ses données dans la base de données.

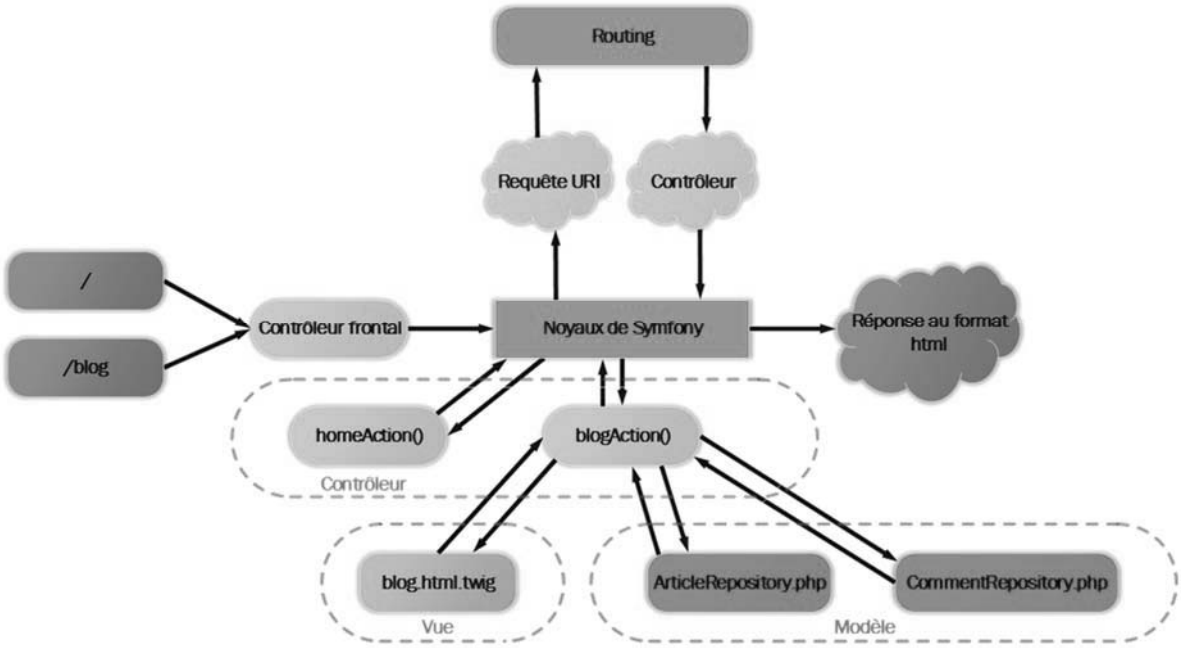
### 1.4.3. Contrôleur



Les contrôleurs agissent comme une interface entre le modèle et la vue, pour traiter toute le logique métier et les requêtes entrantes, manipuler les données à l'aide du composant Modèle et interagir avec les Vues pour rendre le résultat final.
Par exemple, le contrôleur « Client » va traiter toutes les interactions et les entrées de la Vue « Client » et mettre à jour la base de données en utilisant le Modèle « Client ». Le même contrôleur sera utilisé pour visualiser les données du client.

### 1.5. *Symfony ?*

- Symfony est un framework Web.
- Symfony est un ensemble de composants PHP.
- Il est basé sur l'architecture (modèle) MVC.
- Il est libre écrit en PHP.
- Il permet de réaliser des applications web
- Il fonctionne avec d'autres composants PHP comme symfony/yaml, symfony/twig-bundle



### *1.5.1. Spécifications rapides*

Lancement : Octobre 2005

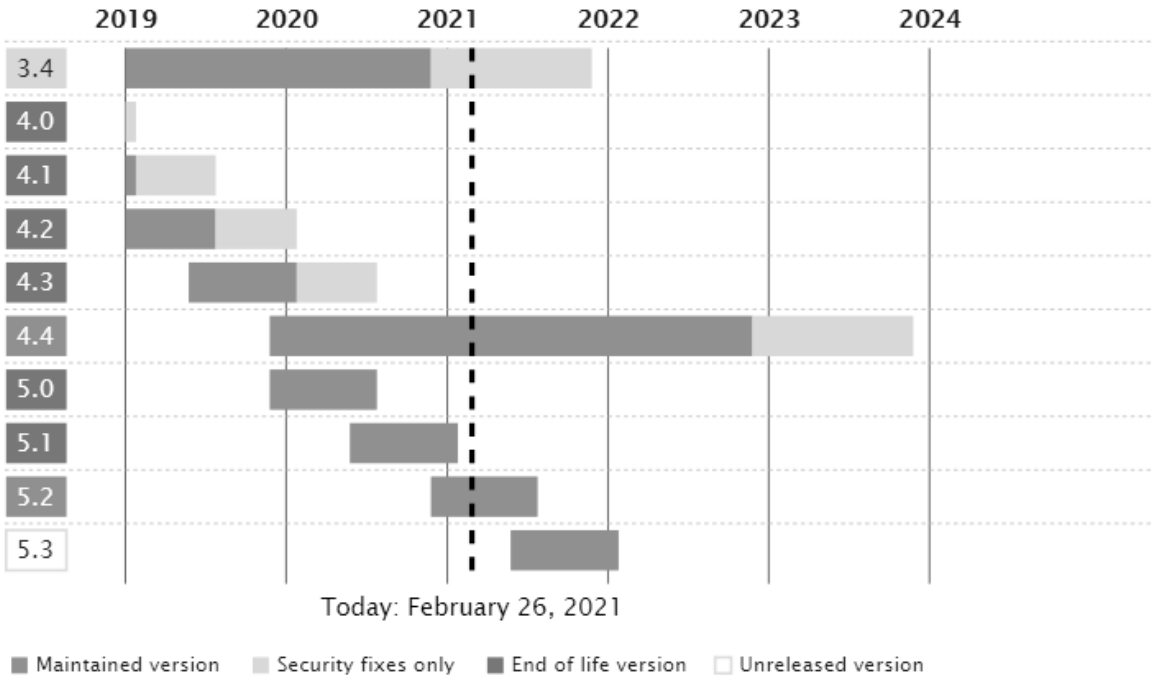
Latest Stable Release

5.2.3

Latest Long-Term Support Release

4.4.19

Symfony Releases Calendar




Symfony 4.4 Release

| Status: | Released: | End of bug fixes: | End of security fixes: |
|---|---|---|---|
| Maintained | November 2019 | November 2022 | November 2023 |

Latest version: 4.4.19 Code Documentation Changelog New features

Exigences techniques :

PHP >= 7.1

Composer installé

Git installé

### *1.5.2. Les avantages de Symfony*

Symfony est un excellent choix pour les sites web et les applications qui doivent être évolutifs.
Son système de composants modulaires est très flexible et vous permet de choisir les composants dont vous avez besoin pour votre projet.

Symfony supporte la plupart des **bases de données** des frameworks PHP populaires :

- MySQL
- Oracle
- PostgreSQL
- SAP Sybase SQL Anywhere
- SQLite
- SQLServer

La meilleure façon d'interagir avec vos bases de données est de passer par **l'ORM Doctrine**.
Symfony utilise des mappeurs de données pour faire correspondre les objets à la base de données.
Cela permet de garder votre modèle d'objet et le schéma de la base de données séparés, ce qui signifie que si vous modifiez une colonne de la base de données, vous n'avez pas besoin de faire beaucoup de changements dans votre base de données.

Le **débogage** des projets Symfony est simple grâce à la barre d'outils intégrée.


Symfony utilise le moteur de templating **Twig**, qui est facile à apprendre, rapide et sûr.

Packagist liste plus de 4 000 paquets Symfony que vous pouvez télécharger et utiliser.

Symfony bénéficie du support commercial de Sensio Labs. Cela signifie qu'il y a un support professionnel disponible, contrairement à la plupart des autres frameworks PHP. Il dispose également de versions de support à long terme qui ont 3 années complètes de support.

Les développeurs de Symfony peuvent se former et obtenir de l'aide par de multiples canaux :

- **Documentation complète**
- **Université Sensio Labs**, la plateforme d'apprentissage en ligne Symfony **SymfonyCasts**
- **Certification Symfony**
- Conférences Symfony

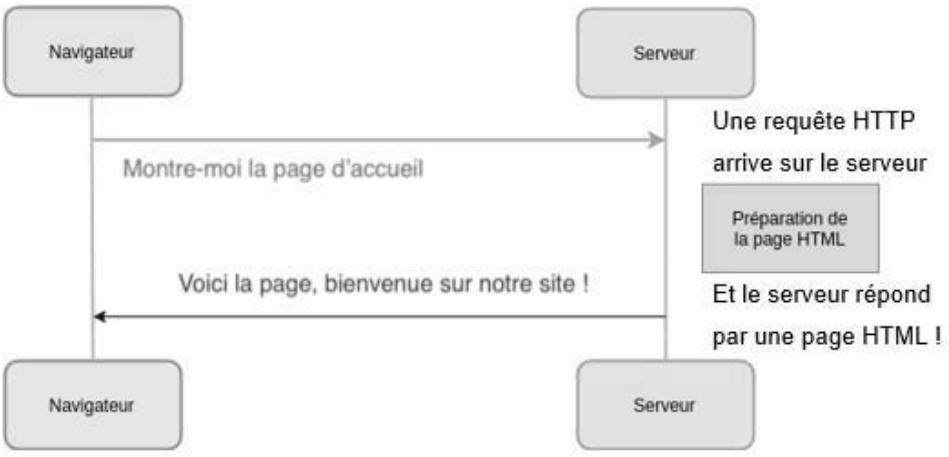
En plus, la **communauté Symfony** est énorme, avec plus de 600 000 développeurs activement impliqués.

## 1.6. *Web et Symfony ?*

Nous l'avons vu en introduction, Symfony est un framework Web.
Il est basé sur le protocole HTTP requête/réponse.
Mais comment cela fonctionne-t-il ?

### *1.6.1. Requêtes et réponses (HTTP)*

L'Hypertext Transfer Protocol (**HTTP**, littéralement «protocole de transfert hypertexte») est un protocole de communication client-serveur développé pour le World Wide Web. C'est un ensemble de règles qui permettent à 2 machines de communiquer sur le réseau internet. Le schéma suivant explique ce qu'il se passe au niveau du réseau quand un utilisateur accède à un lien à l'aide d'un navigateur web :



### 1.6.2. Requêtes et réponses en Symfony

Le framework Symfony, et notamment son composant **HttpFoundation**, apporte une couche d'abstraction pour la requête et la réponse, simple à utiliser et à manipuler.

#### La classe Request

La classe Request permet de centraliser l'accès à toutes les super variables de PHP en une seule classe utilitaire. ($_GET, $_POST,$_COOKIE, $_SESSION, …)

#### La classe Response

La classe Response permet de retourner une réponse à l'utilisateur valide en termes de langage http.

### 1.6.3. Plate-forme de développement Web

Pour développer des applications avec Symfony, vous devez avoir un serveur Web, un interpréteur PHP.
Nous pouvons installer ça dans un package tout prêt contenant (PHP et d'une base de données, exemple WAMP) et utiliser le serveur Web fourni par Symfony.

## 1.7. Composer

Composer est un outil de gestion de dépendances PHP.
Il va nous permettre de déclarer les librairies dont notre projet dépend et s'occuper de les gérer (installation/mis à jour) pour nous.

Quand vous travaillez sur un grand projet PHP, il arrive pour éviter de recréer la roue et aller plus vite dans le développement de votre application, d'utiliser des librairies externes comme pour la gestion de vos utilisateurs ou un système d'envoi de mails, Composer va donc vous permettre de gérer toutes vos dépendances sans trop se casser la tête.

## 1.8. Résumé

Si vous souhaitez réduire le temps passé à développer vos applications web PHP, l'utilisation d'un framework est un choix judicieux.

Le framework que vous choisirez dépendra du type d'application que vous créerez.

# 2. Installation de Symfony :

## 2.1. *Prérequis*

Avant tout, vérifiez que vous avez un environnement de développement web avec un serveur web (Apache par exemple), un moteur de base de données (Mysql, PostgreSQL, SQLite), et PHP.

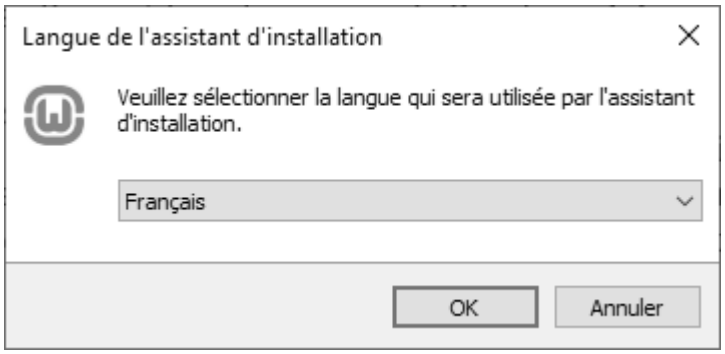
Nous allons utiliser la version de Symfony 4.4.19

Pour installer Symfony, nous pouvons soit utiliser composer ou l'installateur Symfony, nous allons utiliser ici composer et vous devez aussi le privilégié par rapport à l'installateur Symfony. Entrer donc cette ligne commande en vous plaçant dans le dossier dans lequel vous voulez créer votre projet :

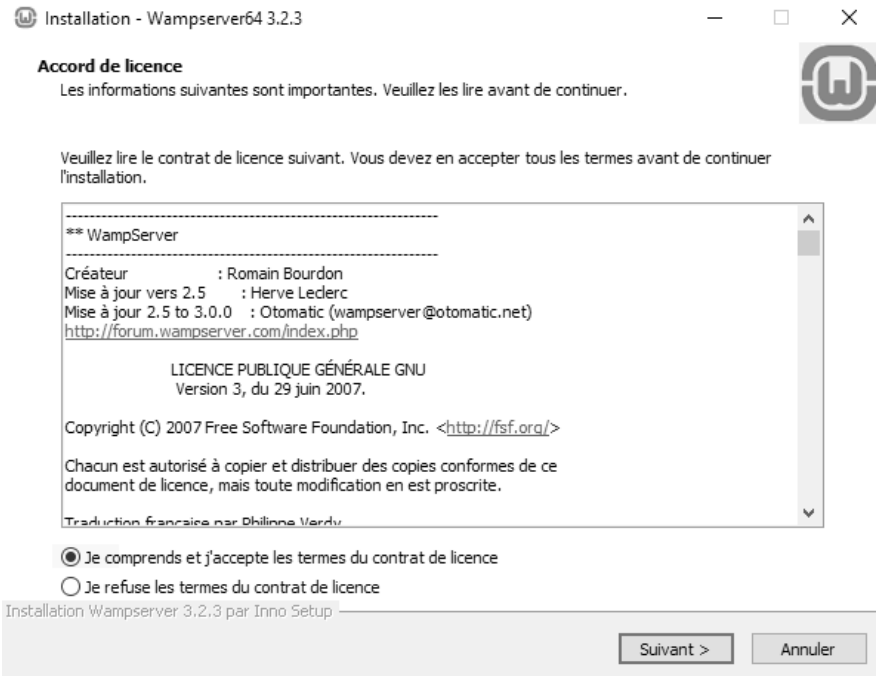
## 2.2. *Installation WAMP*

Lien téléchargement « Wamp » : https://www.wampserver.com/

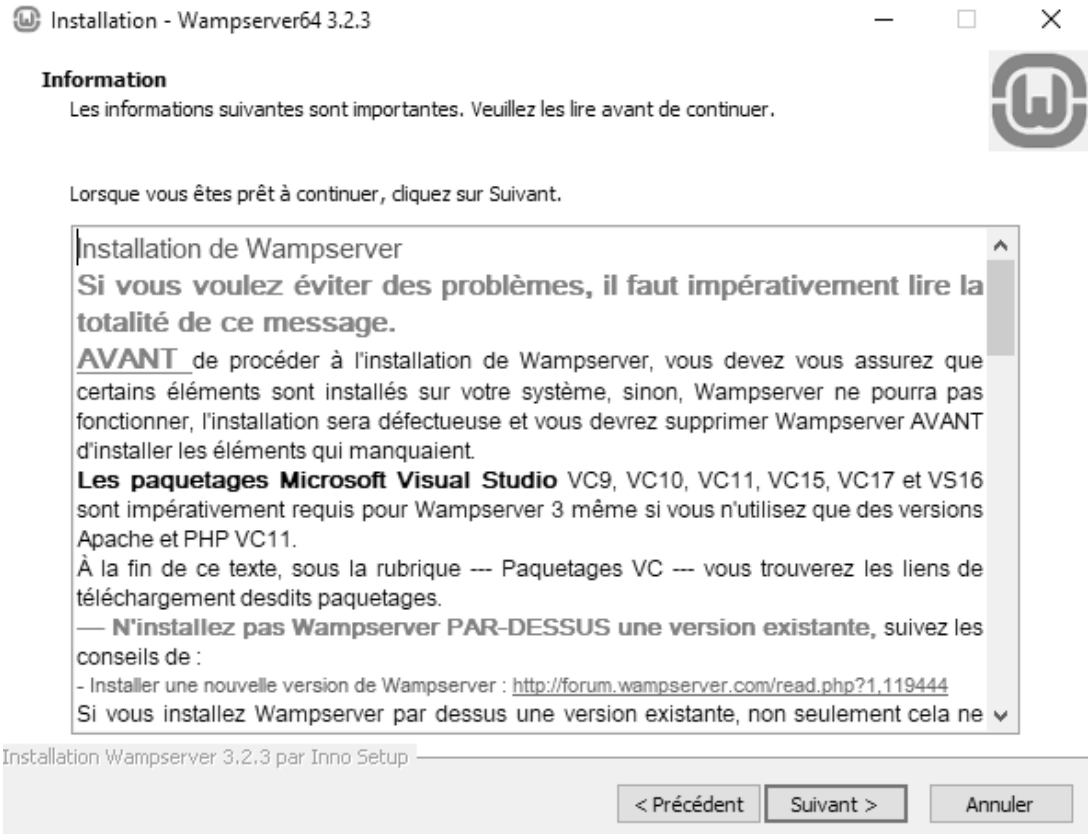
1. Fermez SKYPE pour libérer le port 80 ou changez du l'apache
2. Exécutez le fichier « wampserver3.2.3_x64.exe »



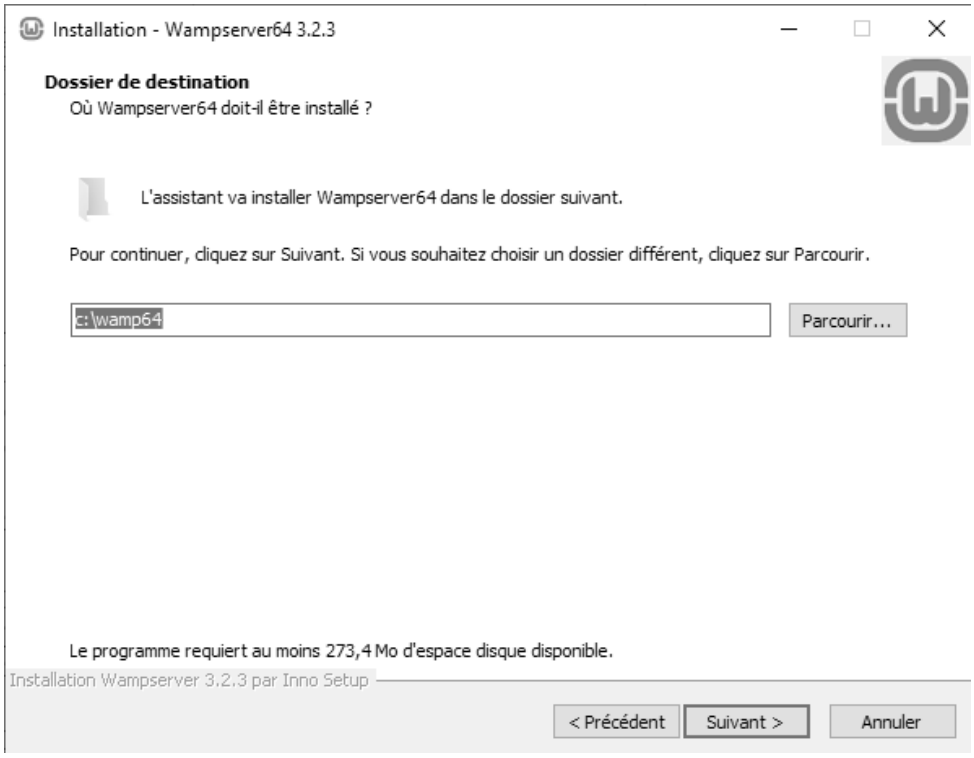
3. Cochez «je Comprends et j'accepte les termes du contrat de licence »



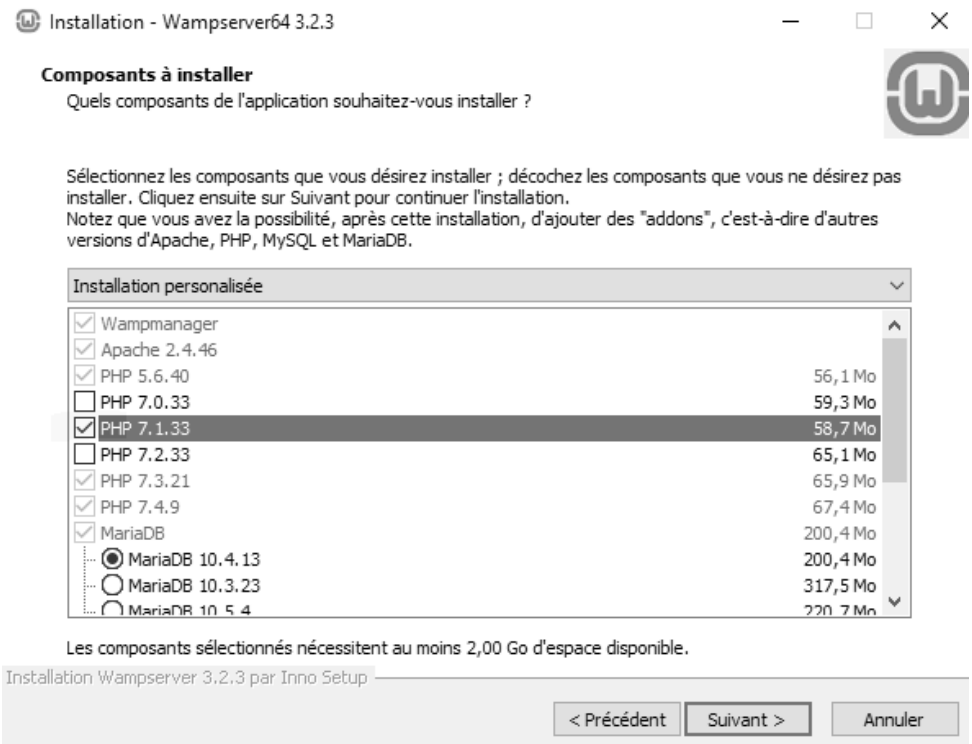

4. Cliquez sur « Suivant »



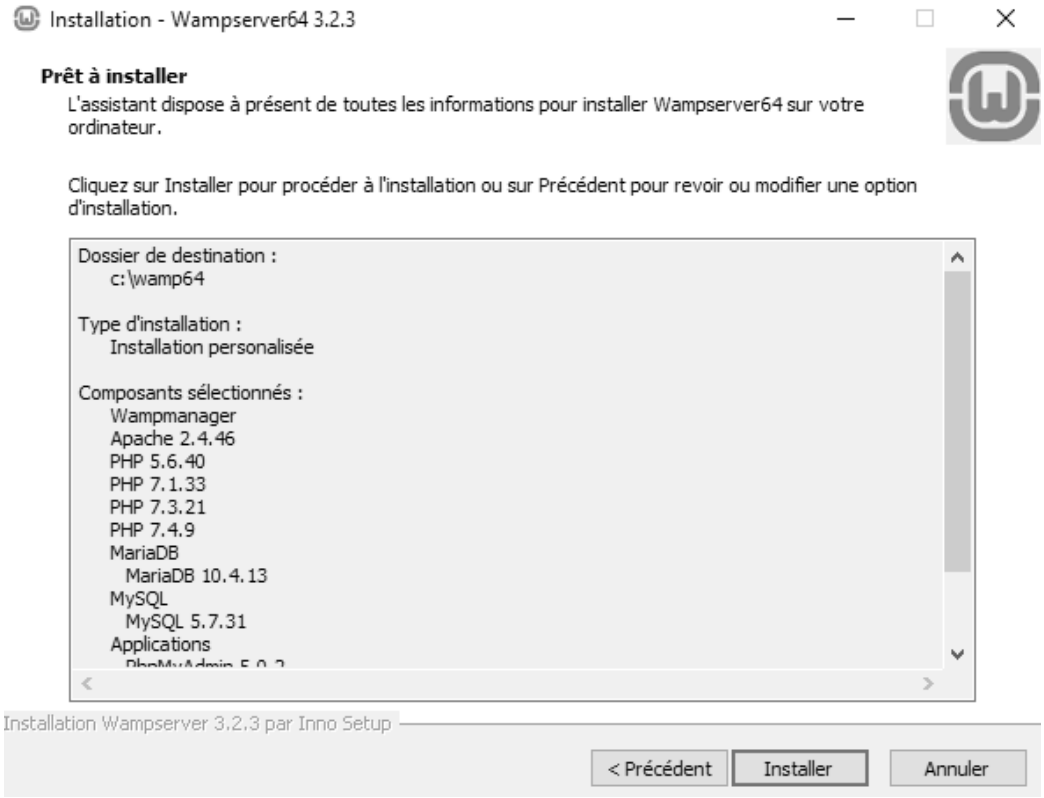
5. Cliquez sur « Suivant »



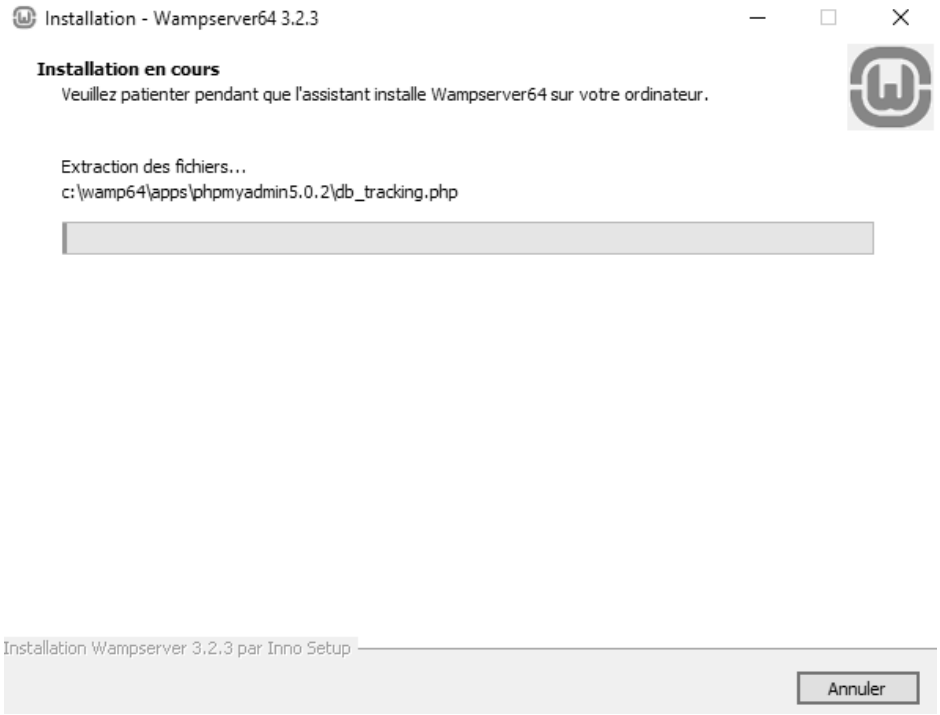

6. Cochez « PHP 7.1.33 » et Cliquez sur « Suivant »



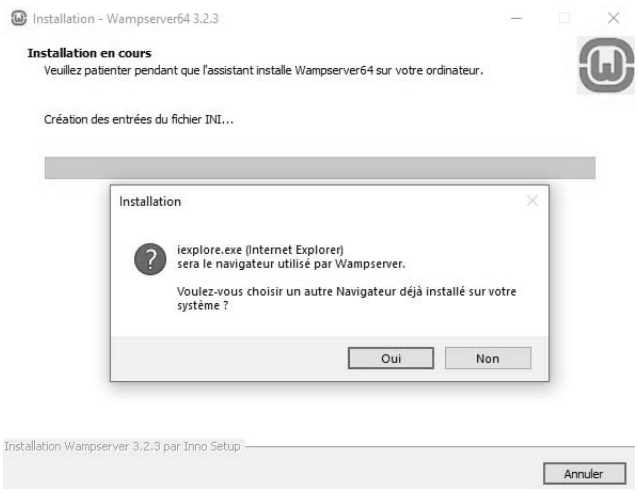
7. Cliquez sur « Installer »

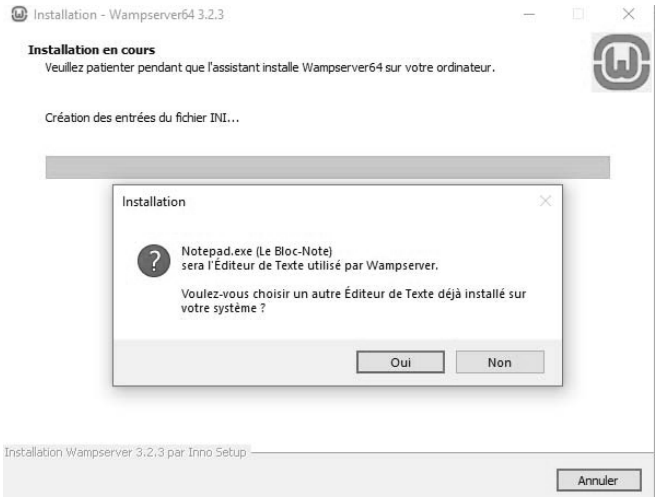



Patientez

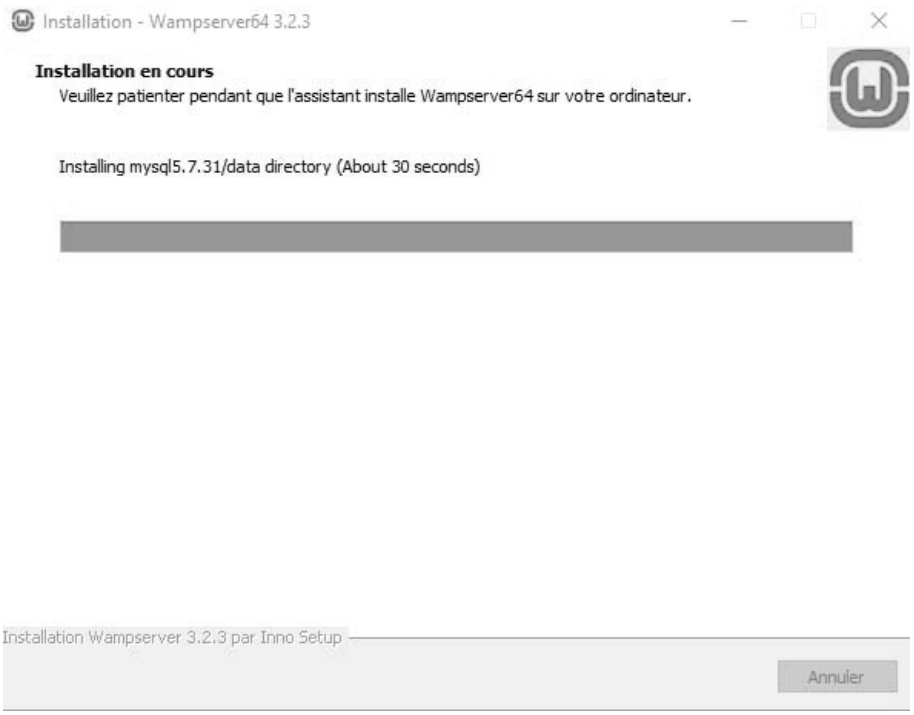


8. Cliquez sur oui ou non à vous de choisir

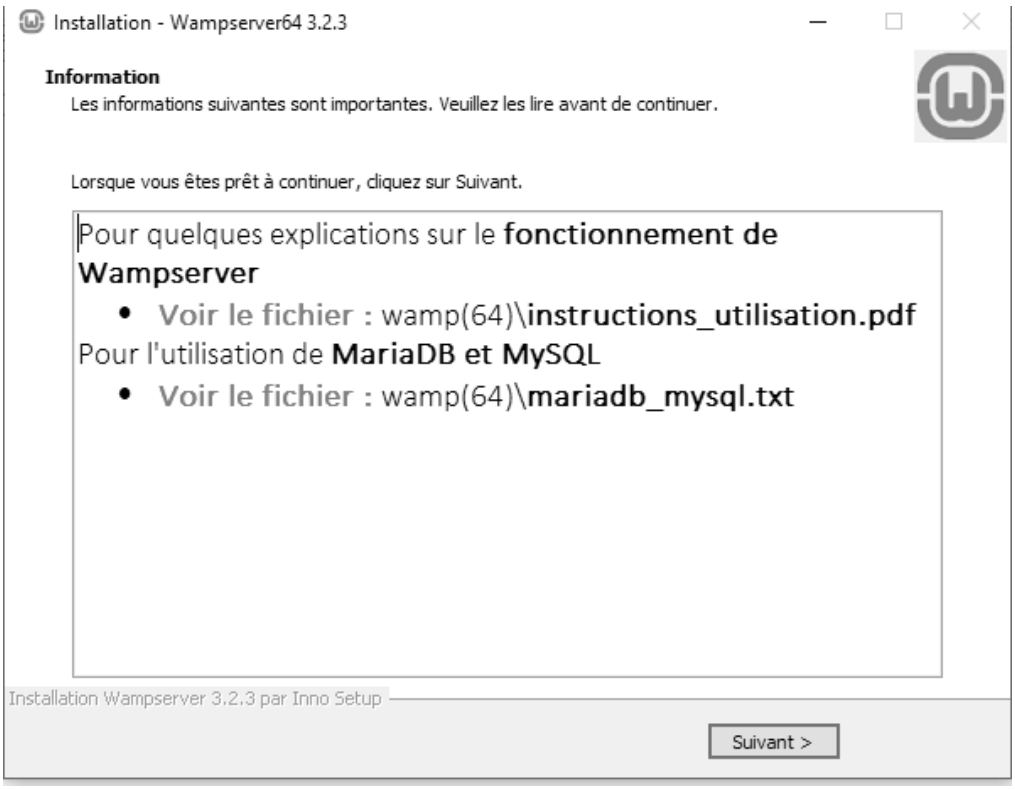


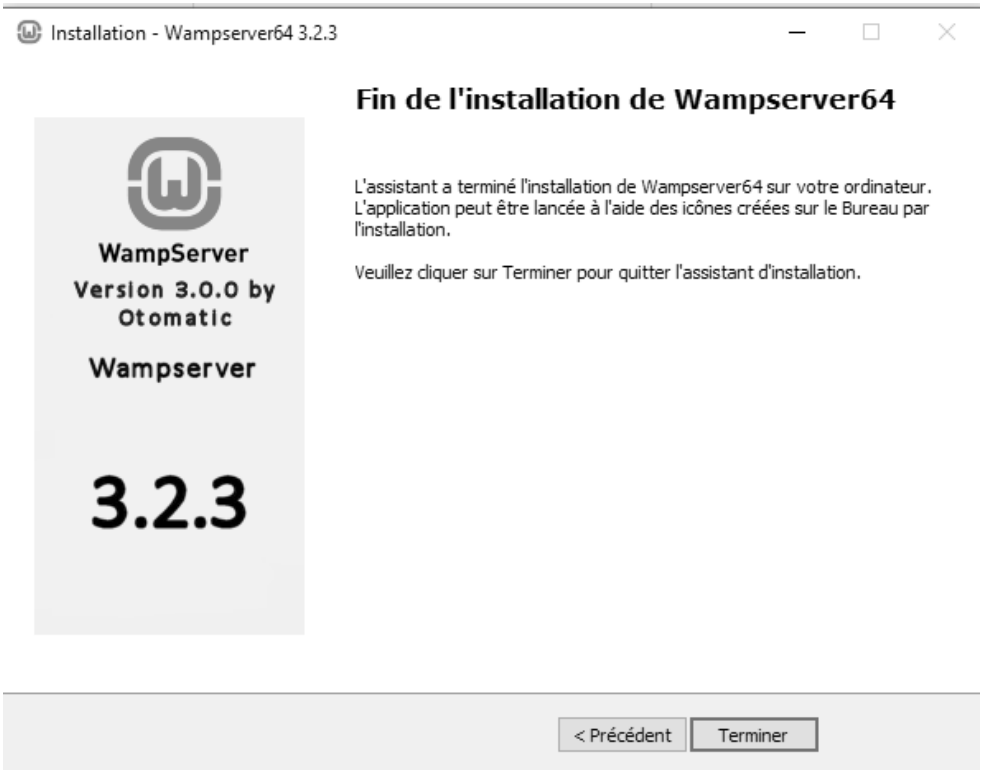

Patientez



9. Optionnel




10. Cliquez sur « Terminer »



11. Ajoutez C:\wamp64\bin\php\php7.1.33 dans la variable PATH d'environnement Windows 7/8/10.

Pour y accéder rapidement, vous pouvez taper « path » dans la barre de recherche et cliquez sur « Modifier les variables d'environnement système ».

12. Redémarrez votre ordinateur
13. Lancez Wamp et vérifier les statuts des services

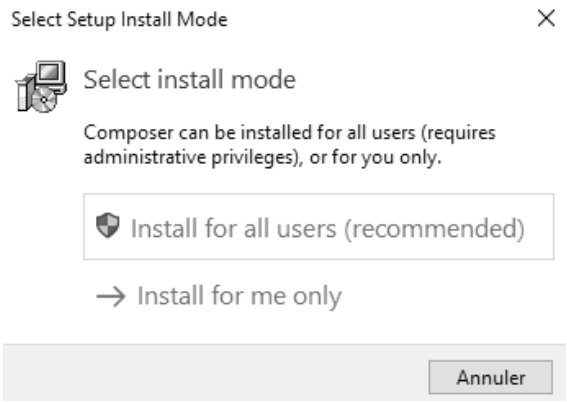

Vous pouvez vérifier le bon fonctionnement de PHP en tapant « php –v » dans un terminal.

```
C:\Users\hp>php -v
PHP 7.1.33 (cli) (built: Oct 23 2019 09:24:14) ( ZTS MSVC14 (Visual C++ 2015) x64 )
Copyright (c) 1997-2018 The PHP Group
Zend Engine v3.1.0, Copyright (c) 1998-2018 Zend Technologies
```

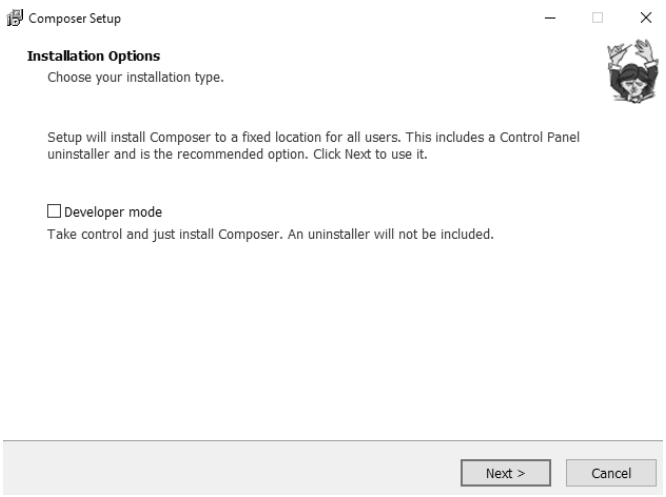
### 2.3. *Installation « Composer »*

Lien téléchargement « Comoser » : https://getcomposer.org/download/

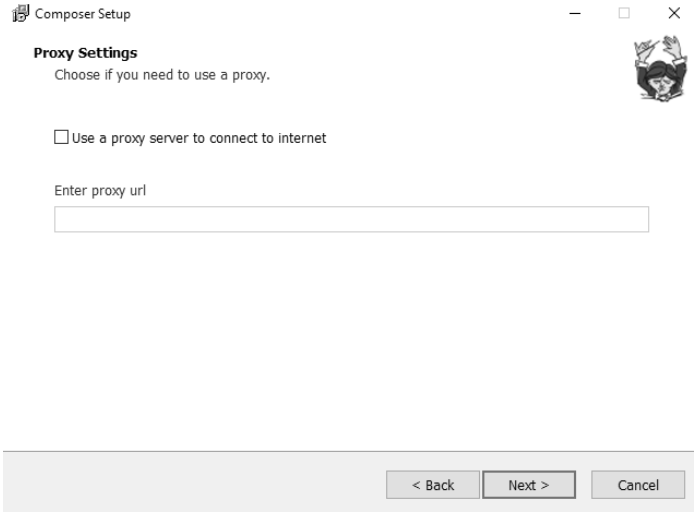
1. Lancez « Composer-Setup » pour Installer « Composer »



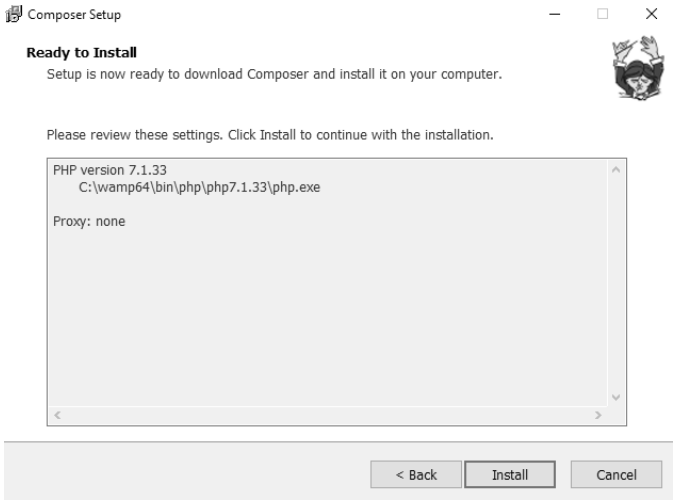

2. Cliquez sur « Next »



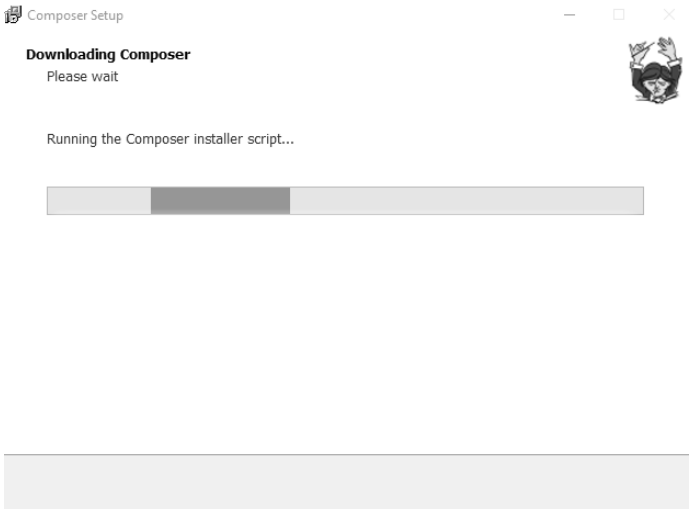

3.



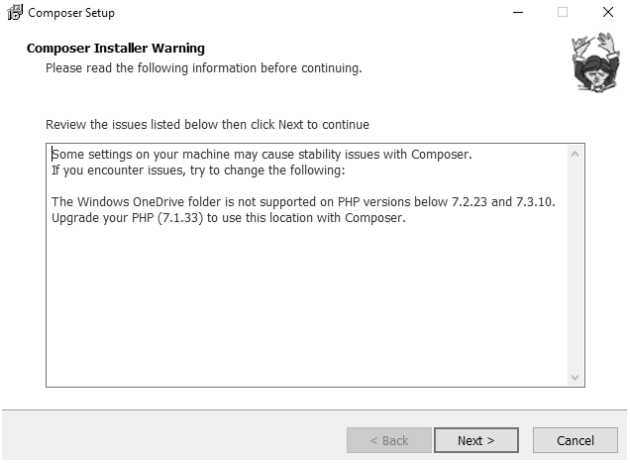

4. Cliquez sur « Install »

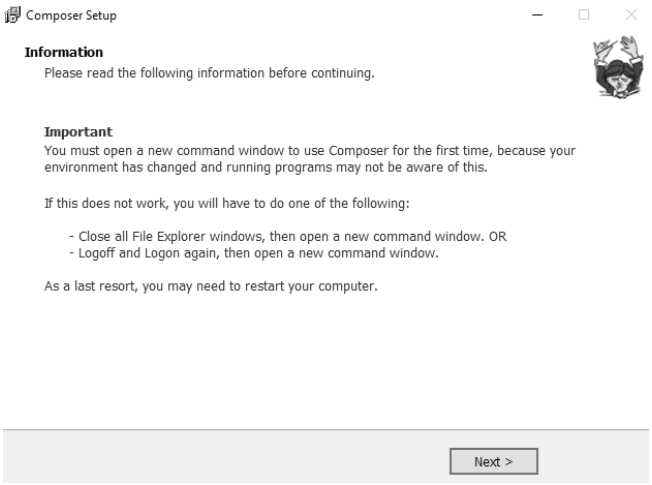



Patientez




5. Cliquez sur « Next »

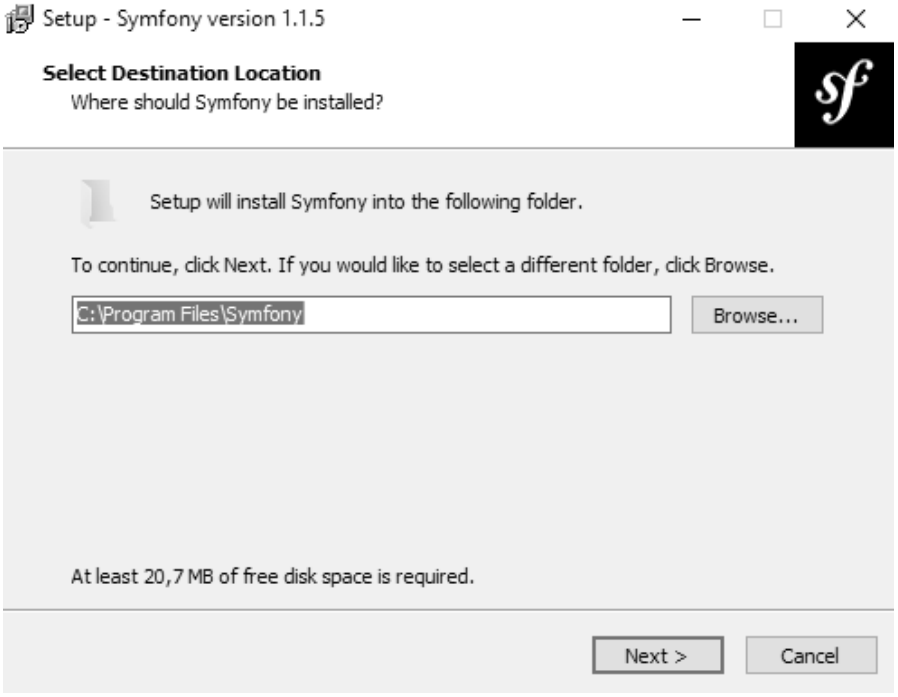


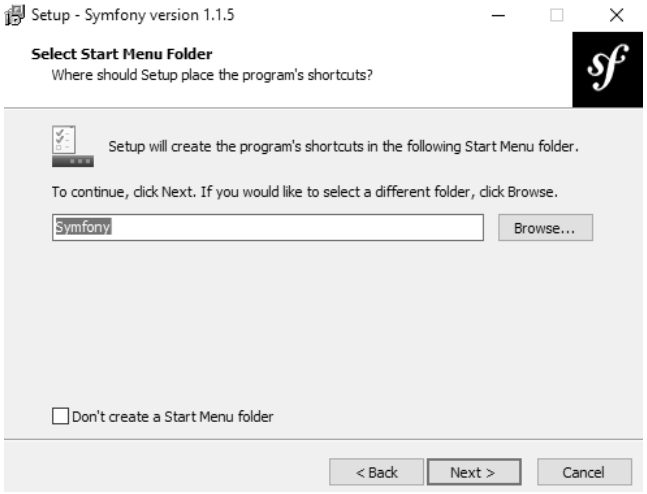

6. Cliquez sur « Finish »

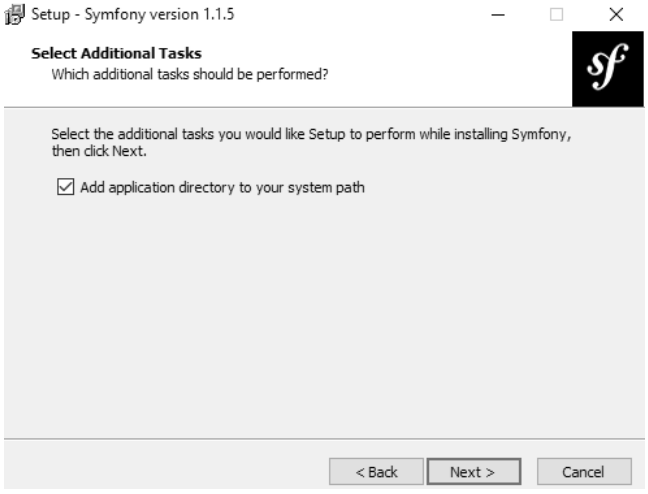

## 2.4. Installation « *Symfony* »

1. Lancez « Smfony.exe »
2. Cliquez sur « Next »

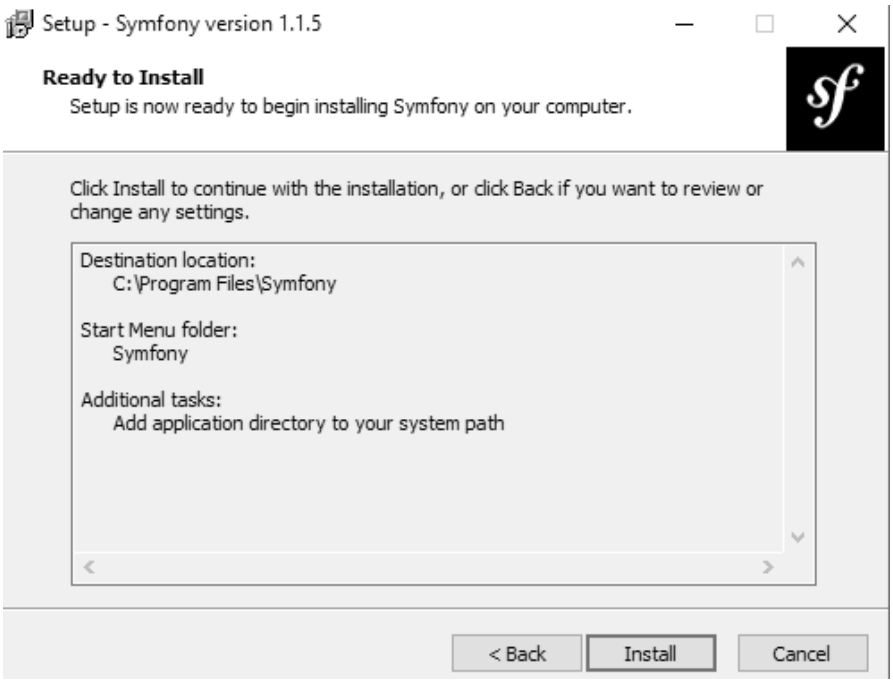



3. .Cliquez sur « Next »

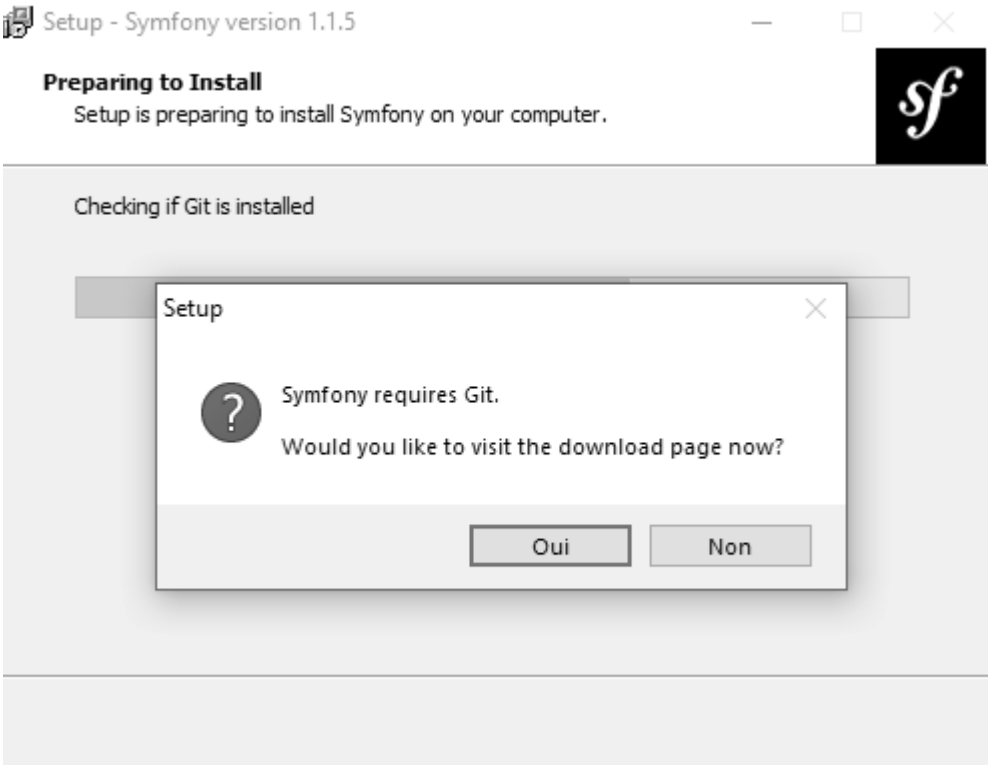


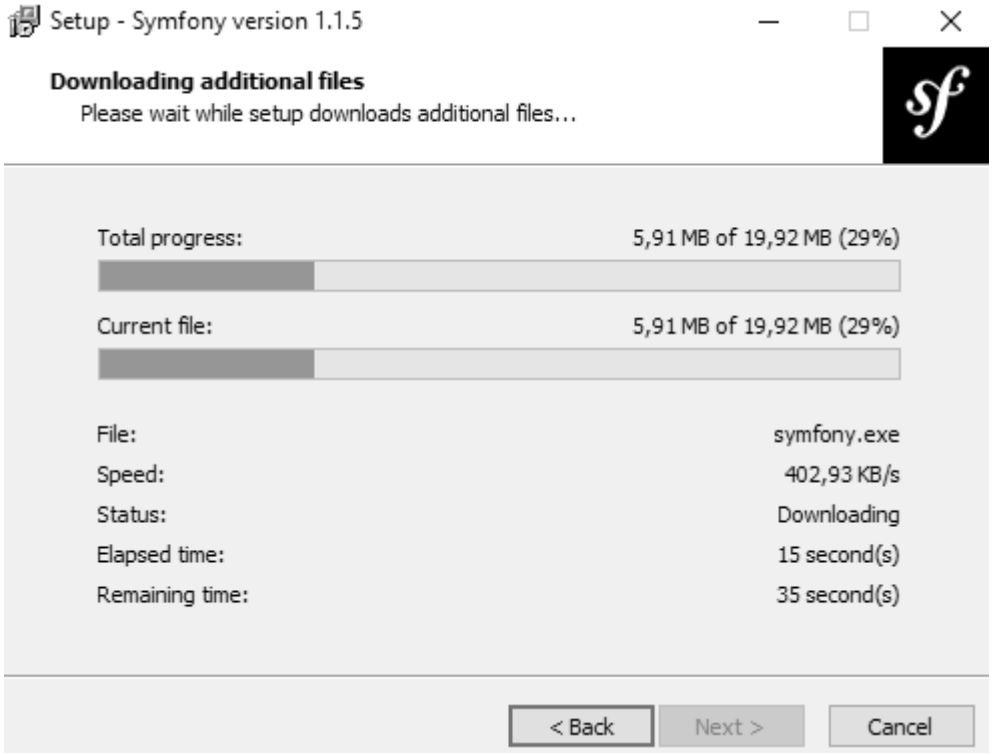

4. Cliquez sur « Install »




5. Cliquez sur « Non », nous allons faire l'installation dans la prochaine section



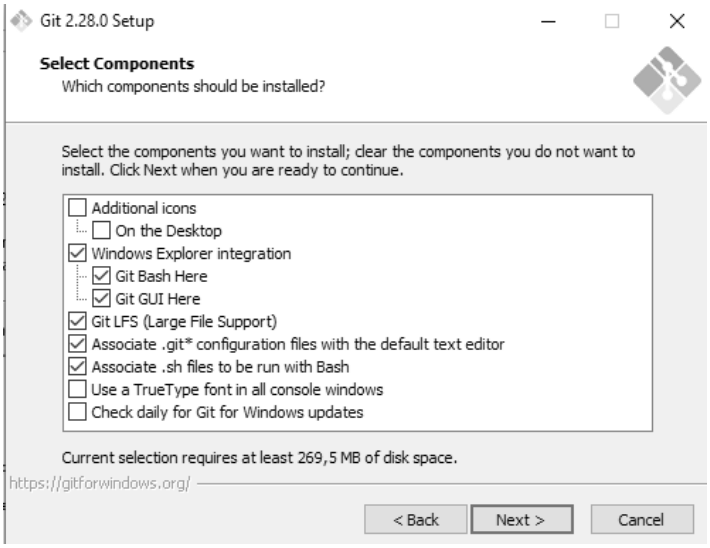

Patientez



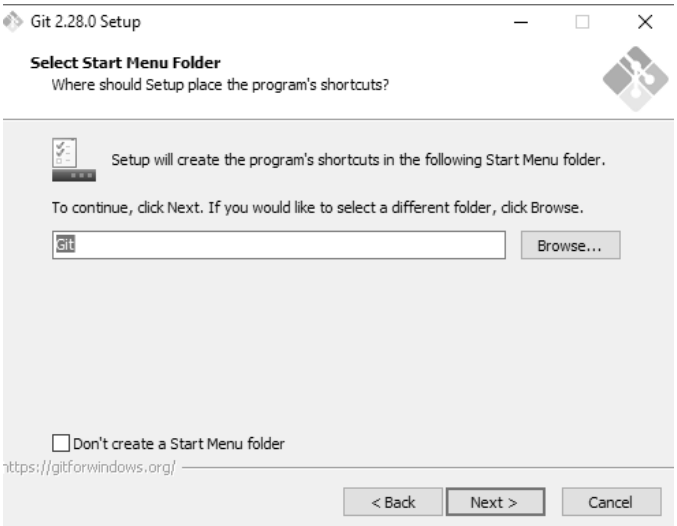

6. Cliquez sur « Finish »

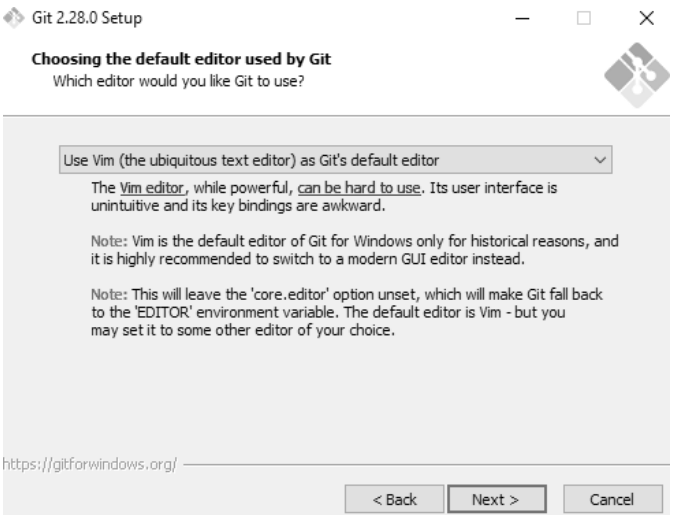
## 2.5. *Installation « Git »*

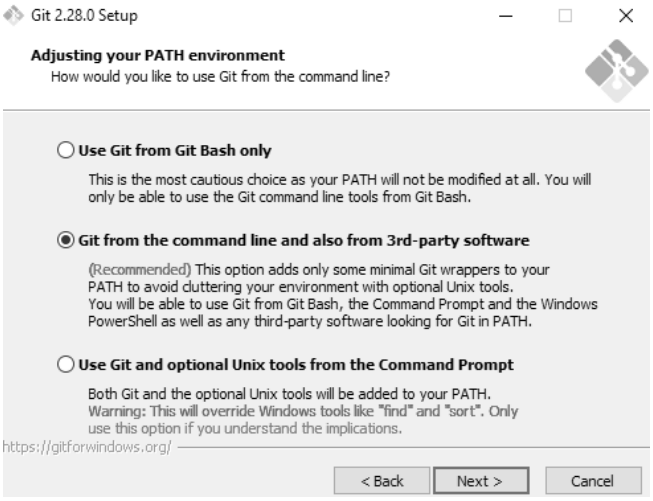
1. Lancez « Git.exe »
2. Cliquez sur « Next »

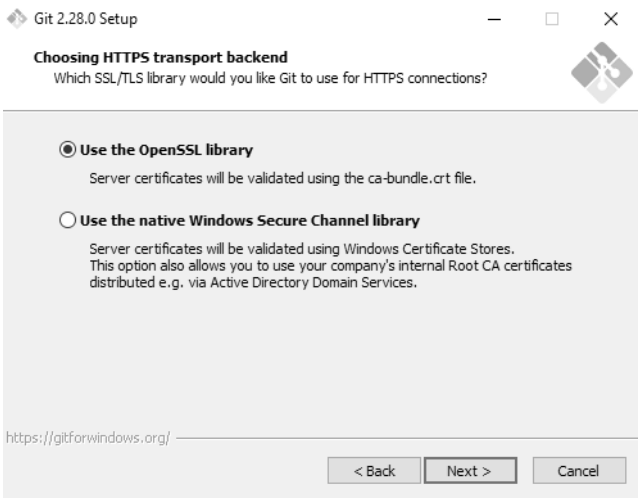




3. Cliquez sur « Next »

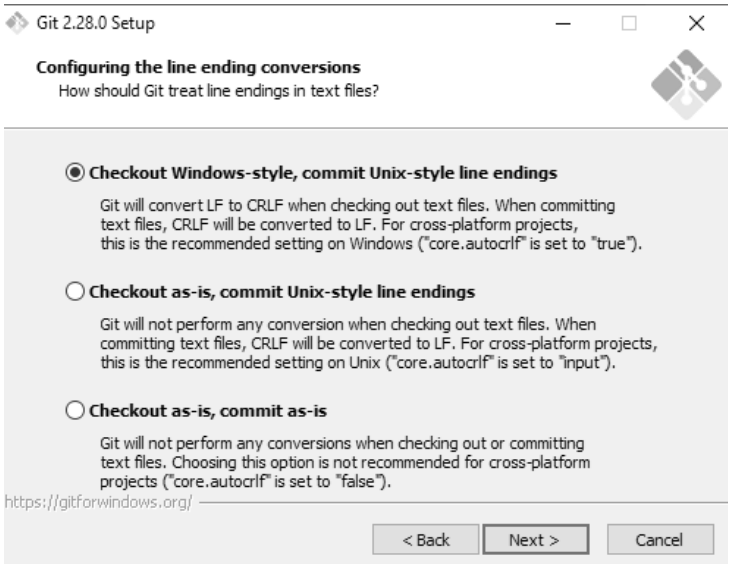


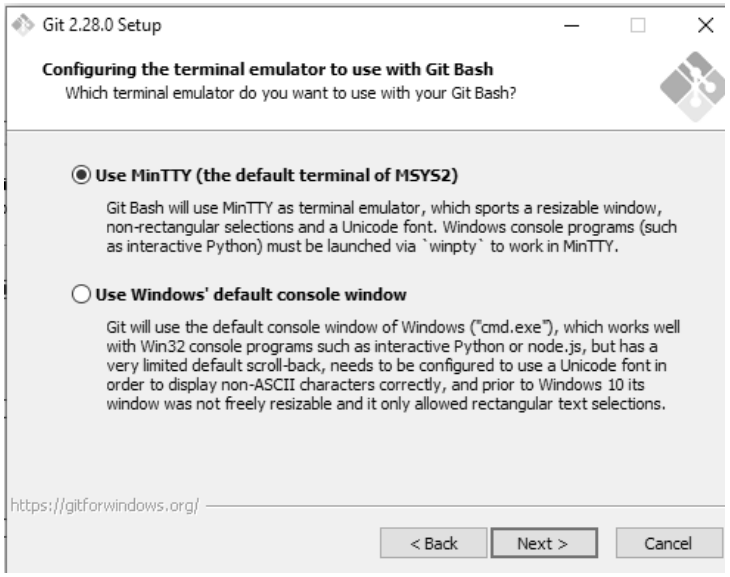



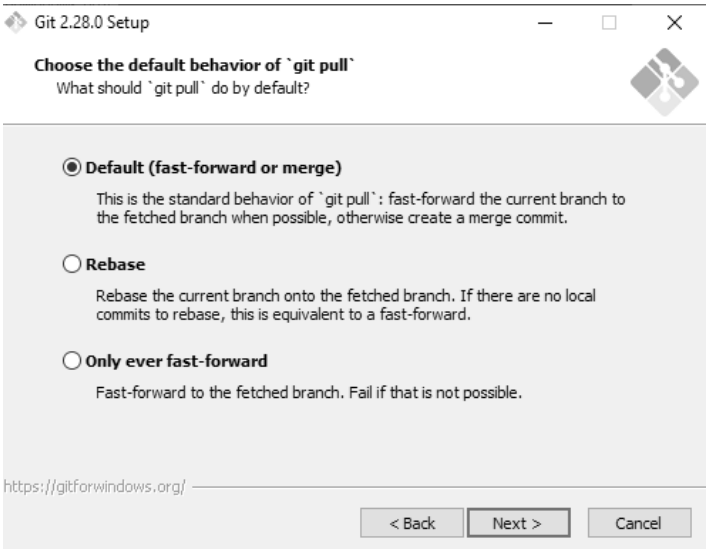



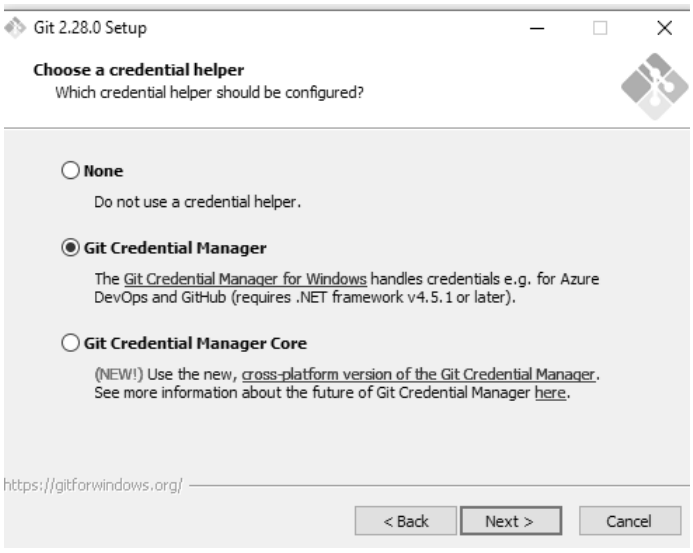



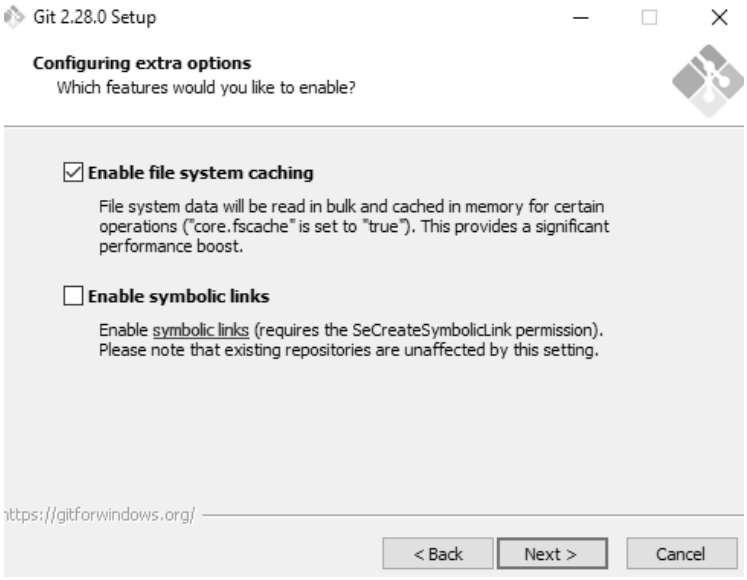
4. Cliquez sur « Next »



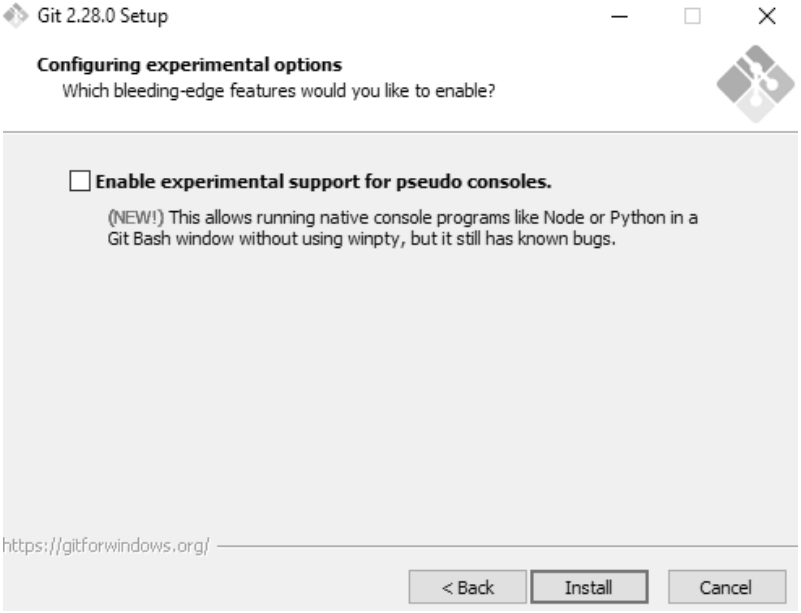



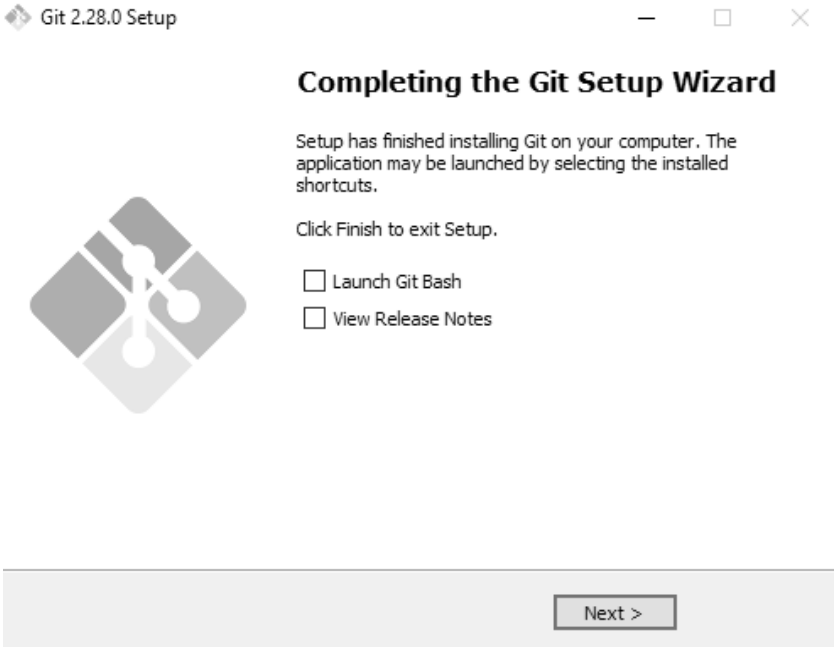



5. Cliquez sur « Install »



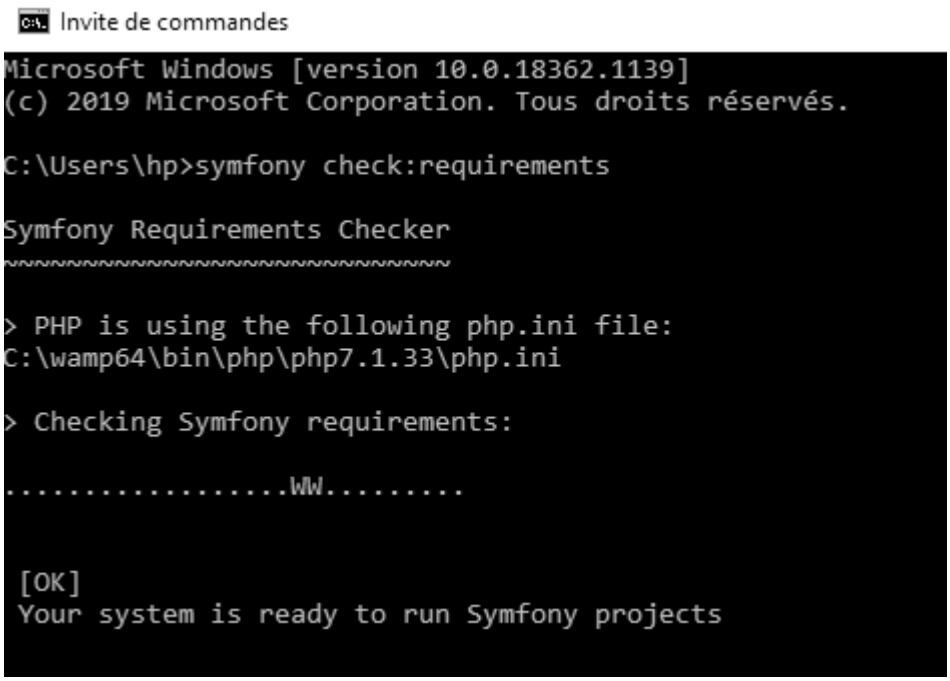

6. Cliquez sur « Finish »




7. Ajoutez les deux chemins suivants C:\Program Files\Git\bin\ et C:\Program Files\Git\cmd\
8. Redémarrez Votre Ordinateur
9. Exécutez la commande suivante « git config --global user.email you@example.com »

## 2.6. *Welcome to Symfony*

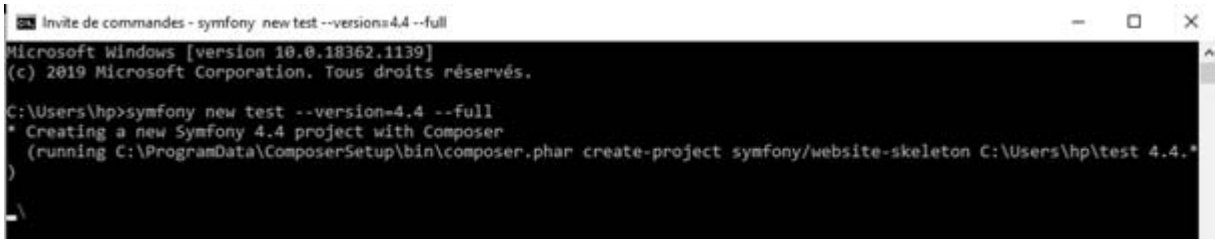
1. Lancez la commande « symfony check:requirements »




2. Créez un nouveau dossier nommé « projet » sous le C :, puis naviguez vers le dossier via l'invite de commande avec la commande « cd C:\projet »

3. Lancez la commande « symfony new test14 --version=4.4 --full »

NB :test c'est le nom du projet ; --full pour installer tous les packages




Si vous rencontrerez l'erreur suivant "Could not find package symfony/web-server-bundle" merci de lancer la commande en ci-dessous

"composer require symfony/web-server-bundle --dev ^4.4.*"

Patientez

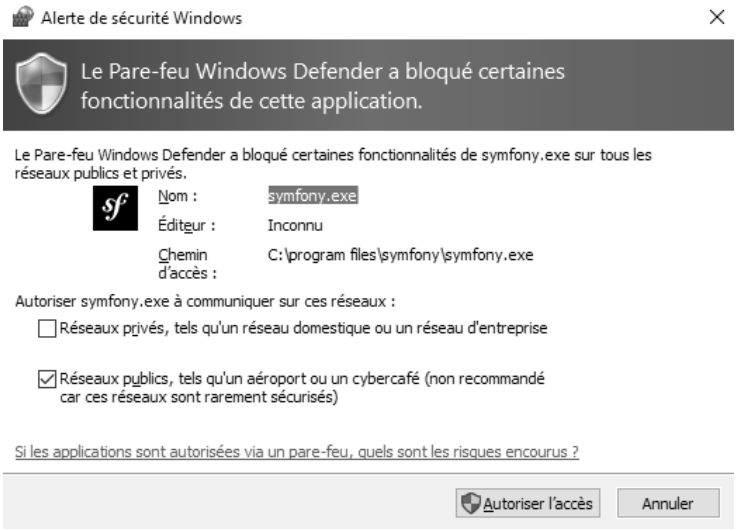
```
C:\project>symfony new test --version=4.4 --full
* Creating a new Symfony 4.4 project with Composer
  (running C:\ProgramData\ComposerSetup\bin\composer.phar create-project symfony/website-skeleton C:\project\test 4.4.*)


* Setting up the project under Git version control
  (running git init C:\project\test)

[OK] Your project is now ready in C:\project\test
```

4. Naviguer vers le dossier « test » Cd test

5. Démarrez le serveur web Symfony avec la commande «symfony server:start »

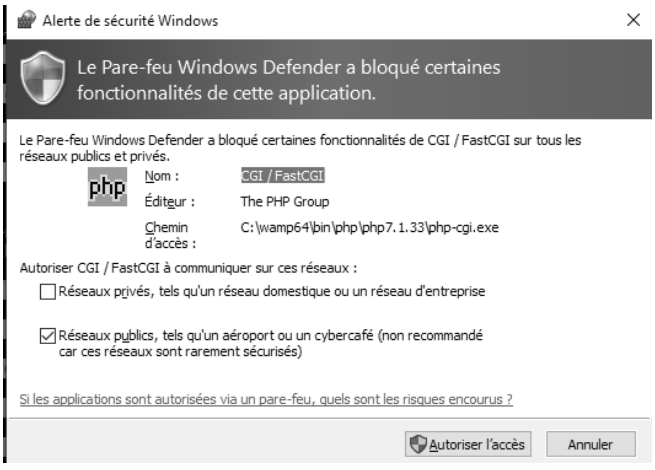



```
C:\project>symfony server:start

[WARNING] run "symfony.exe server:ca:install" first if you want to run the web server with TLS support, or use "--no-
tls" to avoid this warning


Tailing Web Server log file (C:\Users\hp\.symfony\log\736ce3a0499dd4b4a6f44d7c33e26d92b038fc41.log)

[OK] Web server listening
     http://127.0.0.1:8001
```



6. Lancer la page http://localhost:8000/ ➔Finalement « Welcome page »


## 2.7. *Architecture des dossiers*

- **bin**: c'est ici que se trouve les exécutables de Symfony, comme le fichier console. Les fichiers dans ce dossier seront donc exécutables en ligne de commande.

- **config**: comme son nom l'indique, 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**: C'est ici que la magie s'opère, c'est ici que vous écrirez vos fichiers PHP. Les contrôleurs, entités, migrations, … sont dans ce dossier.

- **templates**: Les vues de notre application sont ici, toutes les pages qui vont être affichées à l'écran vont être ici.

- **tests**: C'est ici que vous allez mettre les différents tests pour tester vos fonctionnalités.

- **var:** Dans ce dossier Symfony va mettre les caches et logs

- **vendor**: Ce dossier contient toutes les dépendances créées avec composer.

- **env**: Ce fichier définit les variables d'environnement de notre application, il définit l'environnement dans lequel nous sommes, développement ou production, les informations de connexion à la BDD, ...