	Objectifs	Paramétrage de l'environnement informatique pour CakePHP Création d'un blog avec CakePHP – Partie 1
	Logiciels	XAMPP, phpmyadmin, netbeans
	Ressources	Documents fournis, Internet
	Mots clés	Framework, MVC

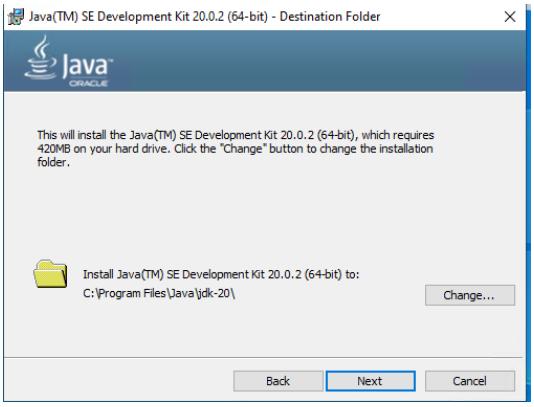
CONSIGNE : SUIVRE LES INSTUCTIONS A LA LETTRE.

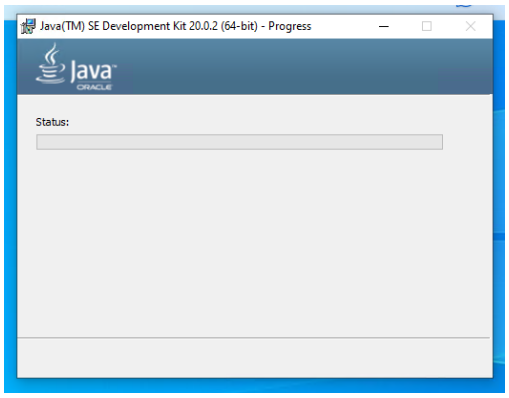
L'objectif de ce TP est de créer un blog de A à Z en utilisant le Framework CakePHP.

Prérequis

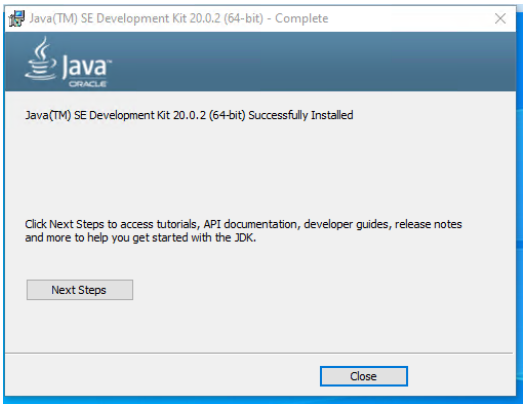
- Installer le dernier XAMPP avec php 8 dans le dossier c\xampp

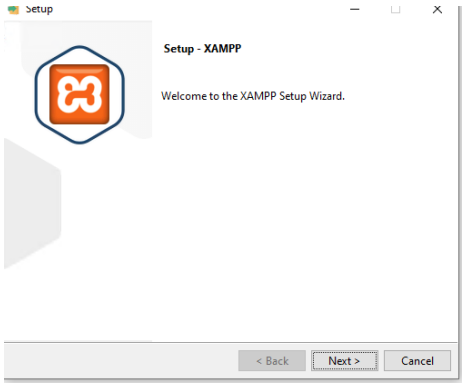
>



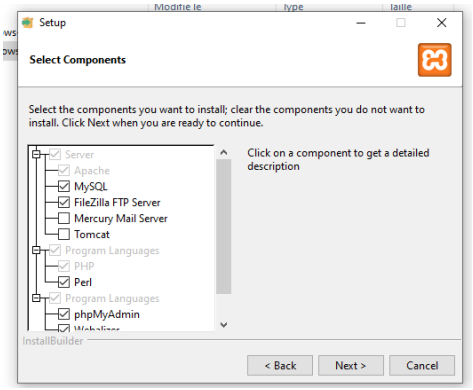


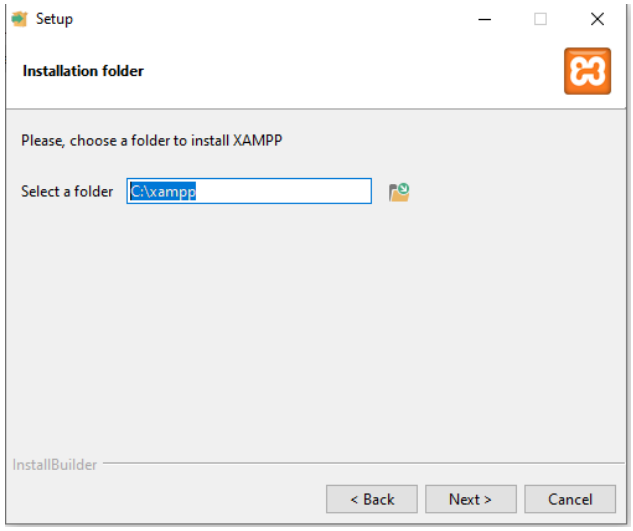
>



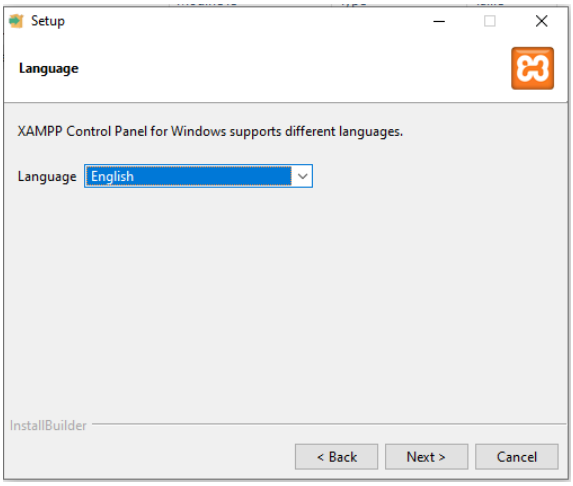

Setup
Setup - XAMPP
Welcome to the XAMPP Setup Wizard.
< Back
Next >
Cancel

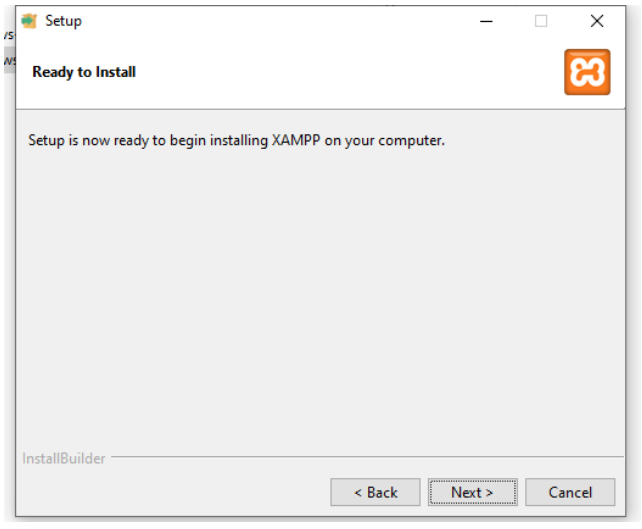
>

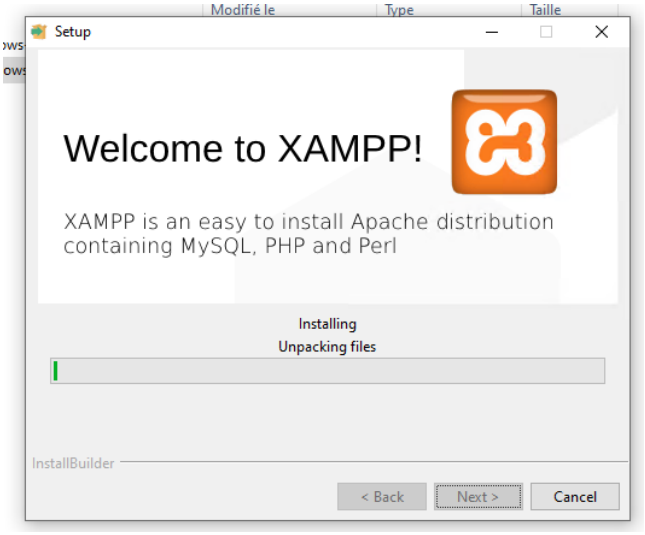

Setup
Select Components
Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.
Click on a component to get a detailed description
Server
Apache
MySQL
FileZilla FTP Server
Mercury Mail Server
Tomcat
Program Languages
PHP
Perl
Program Languages
phpMyAdmin
InstallBuilder
< Back
Next >
Cancel


Setup
Installation folder
Please, choose a folder to install XAMPP
Select a folder
C:\xampp
InstallBuilder
< Back
Next >
Cancel

>

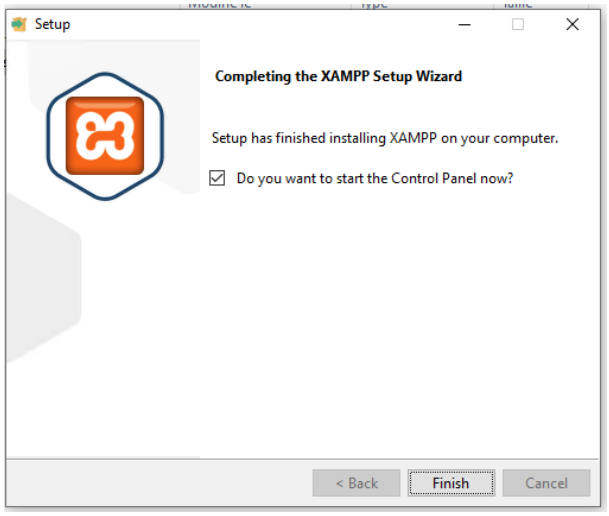

Setup
Language
XAMPP Control Panel for Windows supports different languages.
Language
English
InstallBuilder
< Back
Next >
Cancel


Setup
Ready to Install
Setup is now ready to begin installing XAMPP on your computer.
InstallBuilder
< Back
Next >
Cancel


Modifié le
Type
Taille
Setup
Welcome to XAMPP!
XAMPP is an easy to install Apache distribution containing MySQL, PHP and Perl
Installing
Unpacking files
InstallBuilder
< Back
Next >
Cancel

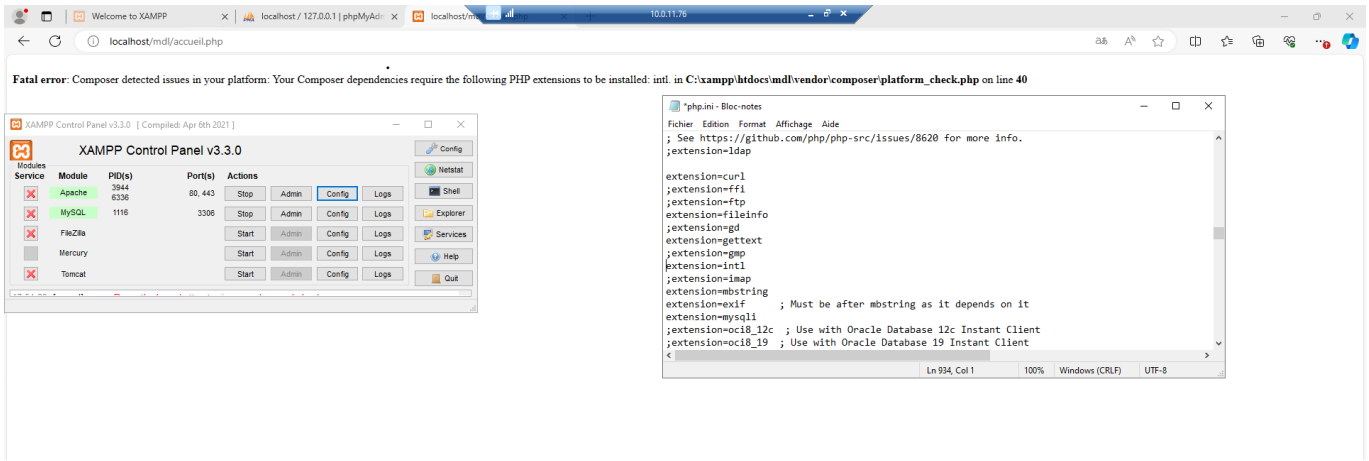
>

- Fin de l'installation



- Vérifiez que les modules intl et openssl sont activés.

- Il faut décommenter l'extension=intl

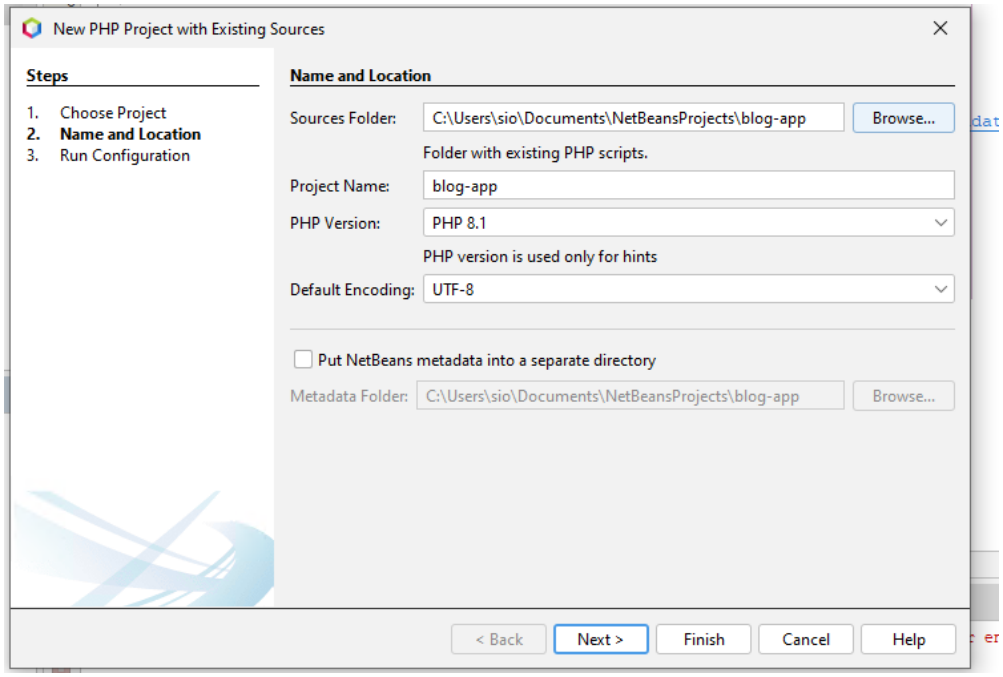


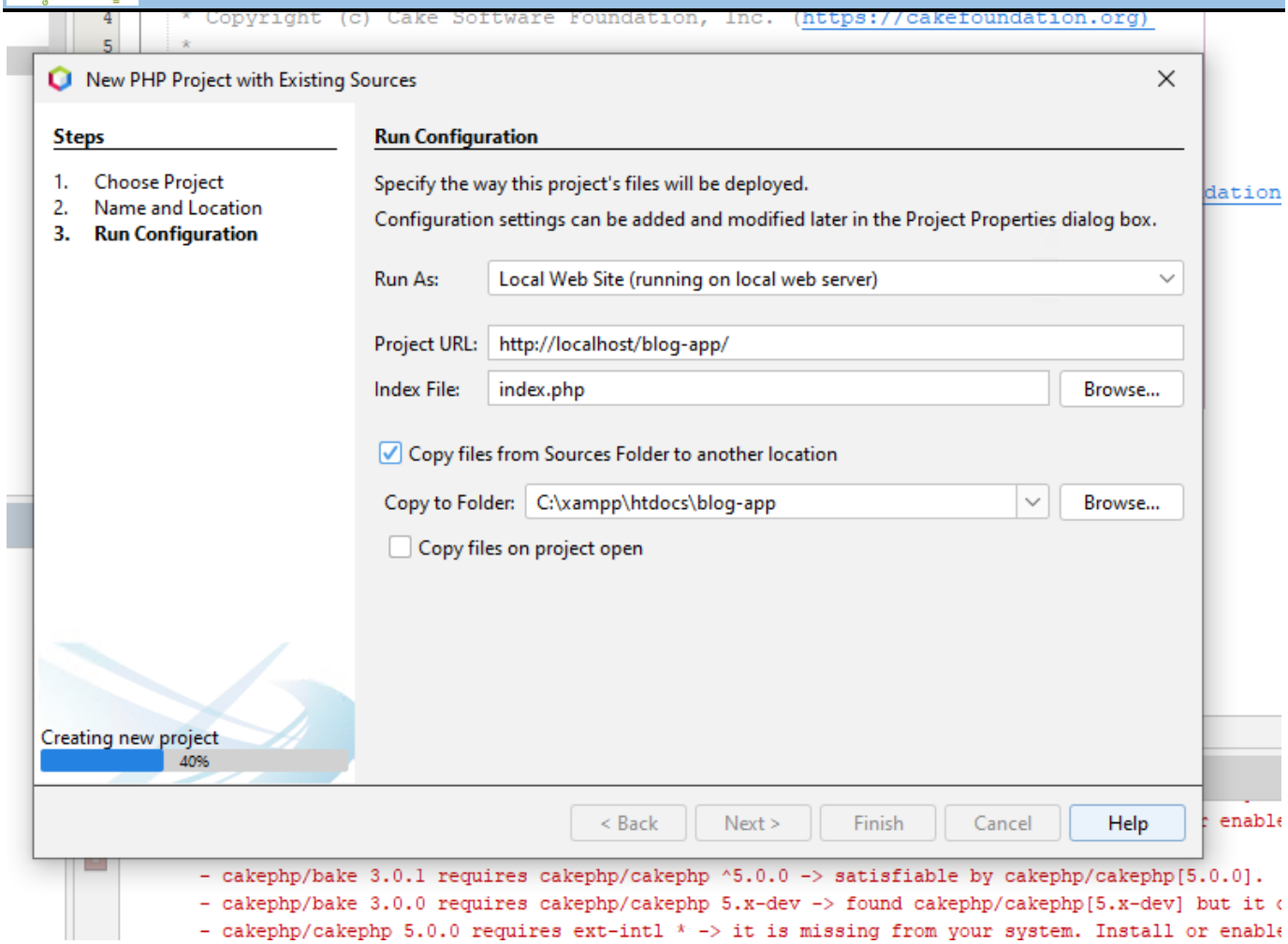
- Restart Apache

Créer le projet blog_app dans Netbeans

- Créez un projet blog_app en utilisant CakePHP en utilisant composer
  composer create-project --prefer-dist cakephp/app:5 blog-app

- Importez le projet dans Netbeans et le copier directement dans htdocs.




* Copyright (c) Cake Software Foundation, Inc. (https://cakefoundation.org)
*
New PHP Project with Existing Sources
Steps
1. Choose Project
2. Name and Location
3. Run Configuration
Run Configuration
Specify the way this project's files will be deployed.
Configuration settings can be added and modified later in the Project Properties dialog box.
Run As: Local Web Site (running on local web server)
Project URL: http://localhost/blog-app/
Index File: index.php
Browse...
Copy files from Sources Folder to another location
Copy to Folder: C:\xampp\htdocs\blog-app
Browse...
Copy files on project open
Creating new project
40%
< Back
Next >
Finish
Cancel
Help
- cakephp/bake 3.0.1 requires cakephp/cakephp ^5.0.0 -> satisfiable by cakephp/cakephp[5.0.0].
- cakephp/bake 3.0.0 requires cakephp/cakephp 5.x-dev -> found cakephp/cakephp[5.x-dev] but it
- cakephp/cakephp 5.0.0 requires ext-intl * -> it is missing from your system. Install or enable

- Lancez votre navigateur à l'adresse localhost/blog_app, quelle erreur y-a-t-il ?
  L'application blog_app n'est pas connecter

Configurer la base de données

- Créez un utilisateur blog_app avec le mot de passe « blog » sur localhost puis créez sa BDD associée en lui donnant tous les droits, encodage utf8_bin en ligne de commande.

Pour créer un utilisateur :

Create user blog_app@localhost identified by «Blog974»

Pour créer une base de donnée :

Grant all privileges on blog_app.* to blog_app@localhost;

Pour créer un nouveau projet :

composer create-project --prefer-dist cakephp/app:5 testp

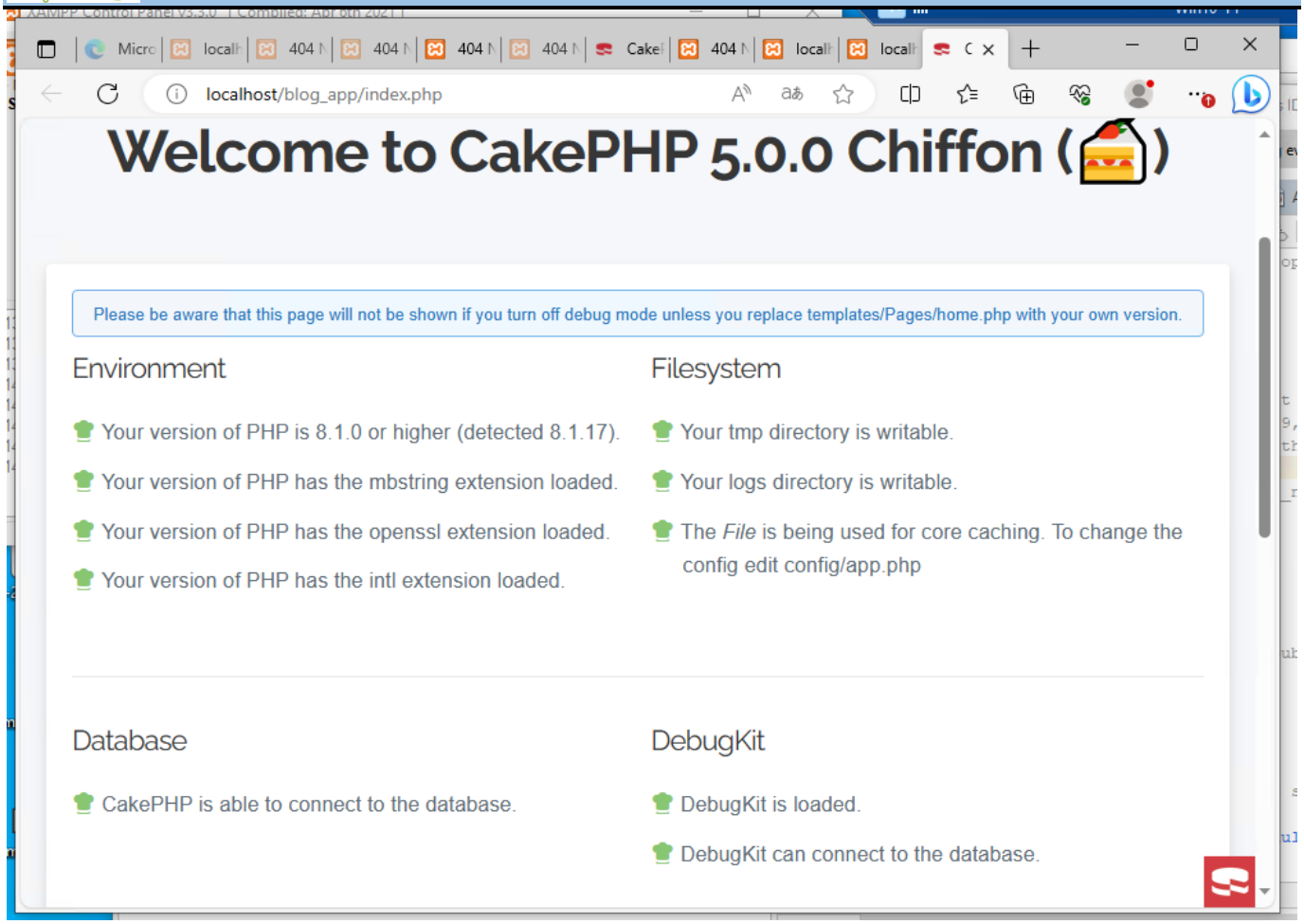
```
sio@WIN10-YY c:\xampp
# mysql -u root
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 11
Server version: 10.4.28-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

On se connecte a la base de donner en tapant mysql -u root

Configurer l'accès à la base de données

- Allez dans le fichier blog_app\config\app_local.php et renseignez les informations concernant votre base de données dans la section Datasources default

Vérifiez que votre application ne contient plus d'erreur.



- Avec l'invite de commande Windows, allez jusque dans le dossier bin de votre application et tapez la commande suivante :

```
cake migrations create InitialMigration
```

```
C:\Users\sio\Documents\NetBeansProjects>cd blog_app\bin

C:\Users\sio\Documents\NetBeansProjects\blog_app\bin>cake migrations create InitialMigration
PHP Warning:  Module "openssl" is already loaded in Unknown on line 0

Warning: Module "openssl" is already loaded in Unknown on line 0
using migration paths
 - C:\Users\sio\Documents\NetBeansProjects\blog_app\config\Migrations
using seed paths
 - C:\Users\sio\Documents\NetBeansProjects\blog_app\config\Seeds
using migration base class Migrations\AbstractMigration
using alternative template C:\Users\sio\Documents\NetBeansProjects\blog_app\vendor\cakephp\migrations\templates\Phinx\cr
eate.php.template
created C:\Users\sio\Documents\NetBeansProjects\blog_app\config\Migrations\20230913135358_initial_migration.php
renaming file in CamelCase to follow CakePHP convention...
renaming file in CamelCase to follow CakePHP convention...
File successfully renamed to C:\Users\sio\Documents\NetBeansProjects\blog_app\config\Migrations\20230913135358_InitialMi
gration.php


C:\Users\sio\Documents\NetBeansProjects\blog_app\bin>
```

- Que s'est-il passé ?
  Un fichier migration a été créer dans le dossier config et un fichier initialMigration à été créer, tous les fichiers vont contenir les modification de la base de donnée.
- Rajoutez ceci dans la méthode change

```
$articles = $this->table('articles');
$articles->create();
```

- Pour appliquer les changements au niveau de la base de données, tapez dans le terminal Windows :

```
cake migrations migrate
```

Résultat :

- Pour savoir quelles sont les migrations effectuées, tapez dans le terminal Windows :

```
cake migrations status
```

Résultat :

- Que s'est-il passé dans la BDD ?

Si jamais la migration posait problème, CakePHP peut revenir en arrière, donc en l'état juste avant de faire la migration, en effectuant un rollback et cela sans avoir créer une méthode qui le fasse. CakePHP se base sur la fonction change pour exécuter automatiquement son inverse.

- Pour effectuer un rollback, tapez dans un terminal Windows :

```
cake migrations rollback
```

Résultat :

L'affichage des status de migration indique que la migration initiale a été annulée :

Résultat :

- Refaite un migrate pour réappliquer la création de la table posts.
- Créez maintenant une migration AlterArticles pour renseigner les champs de notre table articles :

```
cake migrations create AlterArticles
```

- Modifier la fonction change comme ceci :

```
$articles = $this->table('articles');
$articles->addColumn('title', 'string')
->addColumn('content', 'text')
->addColumn('created', 'datetime')
->addColumn('modified', 'datetime')
->save();
```

- Faites un migrate
- Regardez dans la base de données le résultat :

#	Nom	Type	Interclassement	Attributs	Null	Défaut	Extra
1	id	int(11)			Non	*Aucune*	AUTO_INCREMENT
2	title	varchar(255)	utf8_general_ci		Non	*Aucune*	
3	content	text	utf8_general_ci		Non	*Aucune*	
4	created	datetime			Non	*Aucune*	
5	modified	datetime			Non	*Aucune*	

- Quels champs ont été créés ?

- Insérez des articles à la main (au moins 5)

2.4\ Créez le modèle pour Articles

- Créez le modèle ArticlesTable.php dans blog_app\src\Model\Table

```
<?php
    namespace App\Model\Table;
```

```
use Cake\ORM\Table;

class ArticlesTable extends Table{
}
```

2.5\ Créez le contrôleur pour Articles

- Créez le contrôleur ArticlesController.php dans blog_app\src\Controler

```
<?php
     namespace App\Controller;

     class ArticlesController extends AppController
     {
          public function index(){
               //on récupére tous les posts et on les stocke dans $mesArticles
               $mesArticles = $this->Articles->find()->all();
               $this->set('rep', $mesArticles); //envoie à la vue le contenu de $mesArticles dans $rep qui sera utiliseable
          }
     }
```

La fonction set() renvoie à la vue un tableau qui sera utilisable grâce à la variable $rep. Si vous voulez utiliser le même nom, vous pouvez utiliser la fonction compact() pour simplifier l'écriture :

```
$this->set(compact('mesArticles'));
```

- Modifiez le code pour utiliser le compact

- Lancez l'application en appelant le controller articles. Expliquez l'erreur.

2.6\ Créez la vue

- Allez dans le fichier blog_app\src\Template, créez le dossier Articles et créez la vue index.php

```
<h1>Tous les articles du Blog</h1>
<table>
     <tr>
        <th>Id</th>
        <th>Titre</th>
        <th>Date de cr&eacute;ation</th>
    </tr>

    <!-- Ici se trouve l'itération sur l'objet query de notre $mesArticles, l'affichage des infos des articles -->
    <?php foreach ($mesArticles as $article): ?>
    <tr>
        <td><?= $article->id ?></td>
        <td><?= $article->title ?></td>
```

```
        <td><?= $article->created->format(DATE_RFC850) ?></td>
    </tr>
    <?php endforeach; ?>
</table>
```

- Rafraichissez la page.
- CakePHP possède un debuggeur puissant qui permet par exemple de voir le contenu d'une variable. En bas de la page, cliquez sur variable et regardez le contenu de $mesArticles. Quel est son type et que contient-il ?
- Comment récupère-t-on les champs à afficher ?
- Rajouter une 4ème colonne pour afficher la date de modification.

Remarquez la balise de php `<?php ?>` deviennent `<?= ?>` dans la vue lorsque l'on veut faire un `echo`

Prenez l'habitude de supprimer les variables dans les vues avec le code suivant :

```
<?php unset($mesArticles); ?>.
```

- Rajoutez la ligne pour supprimer la variable $mesArticles.

2.7\ Créez un lien pour afficher le contenu dans une nouvelle page

Nous voulons afficher le contenu d'un article dans une page séparée. Pour avoir le contenu de l'article, il faut cliquer sur le titre et son contenu s'affiche dans une autre page. Pour cela :

- Que faut-il créer ?
- Faudra-t-il des paramètres ?

En HTML, les liens se font avec la balise <a> avec un href. Exemple :

<A HREF="intro.txt">Introduction</A>

- Que sera le lien pour voir l'article dont l'id vaut 12 ?
- Que veut-on à la place d'introduction ?

Dans CakePHP il existe des helpers HTML qui permettent d'insérer facilement du code HTML.

Nous allons utiliser :

```
$this->html->link('nom', URL).
```

Il en existe bien d'autres.

- Modifiez la ligne `<td><?= $article->title ?></td>` par :

```
<td>
    <?= $this->html->link($articles->title, [
    'controller' => 'articles',
    'action' => 'detail',
    $article->id]);
    //l'url généré sera de la forme /articles/detail/…
    ?>
```

```
</td>
```

- Testez le résultat et expliquez l'erreur.

Il

2.8\ Rajouter une fonction dans le contrôleur

- Quelle est la signature php de la méthode à créer ?
- Créez cette fonction et mettez ce code à l'intérieur :

```
$leArticle = $this->Articles->get($id);
$this->set(compact('leArticle'));
```

La méthode get() permet de retrouver et retourner l'enregistrement voulu grâce à sa clé primaire.

- Testez le résultat et expliquez l'erreur.

2.9\ Créez une autre vue

- Créez la vue et copiez-collez le code ci-dessous

- Testez le résultat
- Supprimer les variables dans la vue à la fin
- Pourquoi le contenu est affiché sur une ligne ? Comment y remédier ?
- Créez un lien « Retour à la liste des posts » pour rediriger vers la page index.
- Que se passe-t-il si on rentre, à la main, aucun id ou un id qui n'existe pas ?
  L'application plante avec un gros message d'erreur.

Pour gérer les messages d'erreur, vérifiez que le component Flash est loaded :

```
public function initialize()
    {
        parent::initialize();
        $this->loadComponent('Flash');
    }
```

2.10\ Gestion des erreurs

- Cherchez comment fonctionne le try catch en php et expliquez son fonctionnement.
  Le try catch sert a gérer les erreurs : on met le bloc de code qui peut générer des erreurs dans le try et dans le catch on gère les erreurs : on affiche les messages d'erreurs par exemple. Le but du try catch est faire en sorte que l'application ne plante pas.
- Adaptez-le à notre situation.

```php
public function index() {
    //on récupére tous les articles et on les stocke dans $mesArticles
    $mesArticles = $this->Articles->find()->all();
    $this->set(compact('mesArticles'));
}
public function detail($id):void{
    //on recupere l'enregistrement qui correspond a l'ID
    try{
        $leArticle= $this->Articles->get($id);
    } catch (Exception $ex) {

    }

    // on envoie la variable $leArtcicle à la vue
    $this->set (compact('leArticle'));

}
```

- Dans le paramètre de la fonction, rajoutez « = null », cela permettra de gérer l'erreur lorsque l'on appelle l'action « detail » sans paramètres.

try{

$leArticle= $this->Articles->get($id);

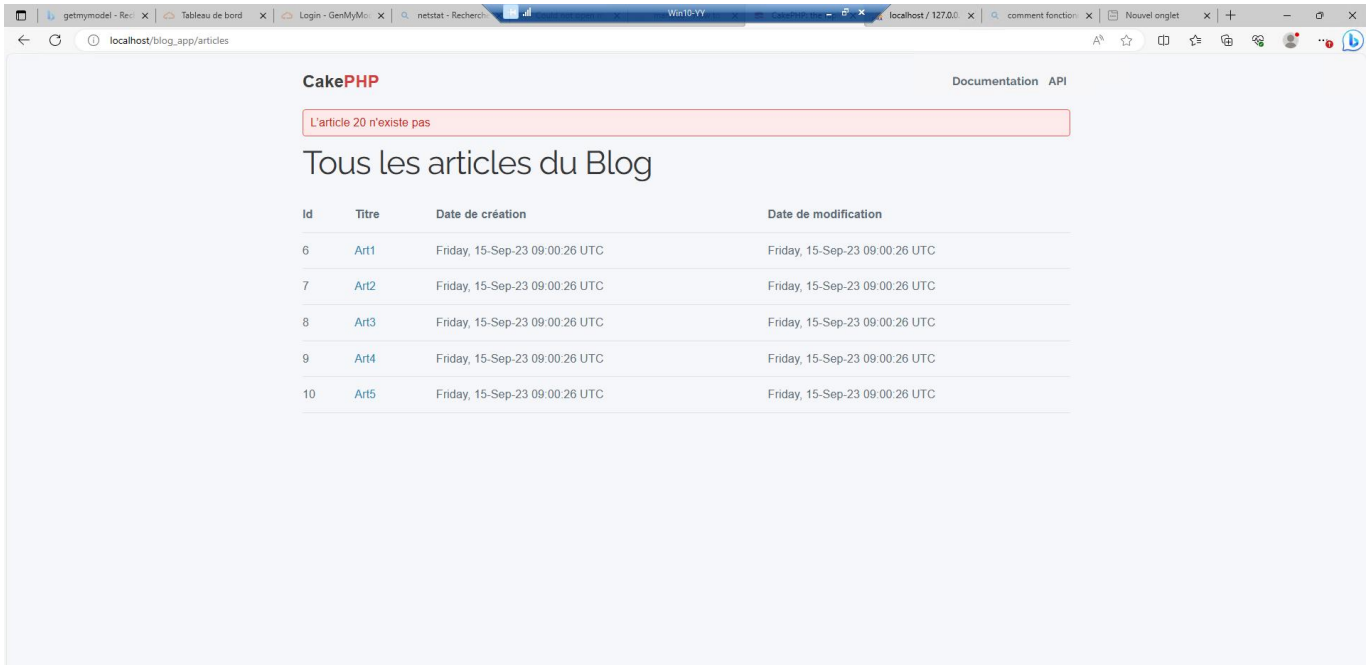
} catch (\Exception $ex) {

Dès qu'une exception est levée, faites ceci :

```
if($id == null){
    $this->Flash->error "L'action detail doit être appelé avec un
identifiant"));
}else {
    $this->Flash->error(__("L'article {0} n'existe pas", $id));
}
return $this->redirect(['action' => 'index']);
```

- Expliquez en français le code
  Si l'utilisateur entre aucun id sur detail on affiche l'action détail doit etre appelé avec un identifiant sinon cela veut dire que l'id n'existe pas on affiche donc le message d'erreur.
- Qu'est-ce que le $id ? Quel est le but de cette ligne avec {0} ? ca correspond a l'article quon veut afficher et ca remplace le dollard acolade le dollard direct sert a rediriger dans la page index pour que lutilisateur choisi le bon detail.
- Expliquez le redirect

- Testez l'action detail en mettant des ids inexistants et sans id pour vérifier que les erreurs sont bien gérées et que le programme continue de tourner.



2.11\ Rajouter des articles

Lire depuis la base de données et afficher les articles est un bon début, mais lançons-nous dans l'ajout de nouveaux articles.

- Quelle action devez-vous créer ?
  L'action add
- Aura-t-elle un paramètre ?
  Non
- Ecrivez la signature de la méthode et insérez le code ci-dessous :

public function add(){
}

```
$leNewPost = $this->Posts->newEntity();
if ($this->request->is('post')) {
     $leNewPost    =    $this->Posts->patchEntity($leNewPost,    $this-
>request->data);
     if ($this->Posts->save($leNewPost)) {
          $this->Flash->success(__("Le post a été sauvegardé."));
          return $this->redirect(['action' => 'index']);
     }
     else $this->Flash->error(__("Impossible d'ajouter votre post."));
}
$this->set(compact('leNewPost'));
```

Remarque : Alors que les objets Table représentent et fournissent un accès à une collection d'objets, les entities représentent des lignes individuelles ou des objets de domaine dans votre application. Les entities contiennent des propriétés et des méthodes persistantes pour manipuler et accéder aux données qu'ils contiennent. Vous n'avez pas besoin de créer des classes entity pour utiliser l'ORM dans CakePHP. Cependant si vous souhaitez avoir de la logique personnalisée dans vos entities, vous devrez créer des classes

- Lancer l'action add. Quelle est l'erreur et comment la réparer ?

2.12\ Créez une autre vue

- Comment s'appelle le fichier à créer ? Créez-le blog_app/article/add
- Insérez le code ci-dessous :

```
<h1>Ajouter un post</h1>
<?php
    echo $this->Form->create($leNewPost);
    echo $this->Form->input('title');
    echo $this->Form->input('content', ['rows' => '3']);
    echo $this->Form->button(__("Sauvegarder le post"));
    echo $this->Form->end();
?>
```

La création d'un formulaire est ultra facilitée par l'utilisation de helpers fournis par CakePHP.

- Comment crée-t-on un formulaire ?
  Grace au helper $this->Form->create et entre parenthèse le nom de la variable qu'on veut créer.
- Ou est gérée l'action du formulaire ?
  L'action est gérer dans le add elle même

Les champs created et modified peuvent être gérés automatiquement en rajoutant une ligne de code dans la classe de ArticlesTable. On n'aura donc pas besoin de s'occuper de ces champs-là. Pour cela, rajouter le code dans la classe PostsTable :

```
public function initialize(array $config) :void
    {
        $this->addBehavior('Timestamp');
    }
```

- Depuis la page add, créez un lien pour retourner vers la page index
- Sur la page index, créer un lien pour ajouter un post.

2.13\ Valider des données

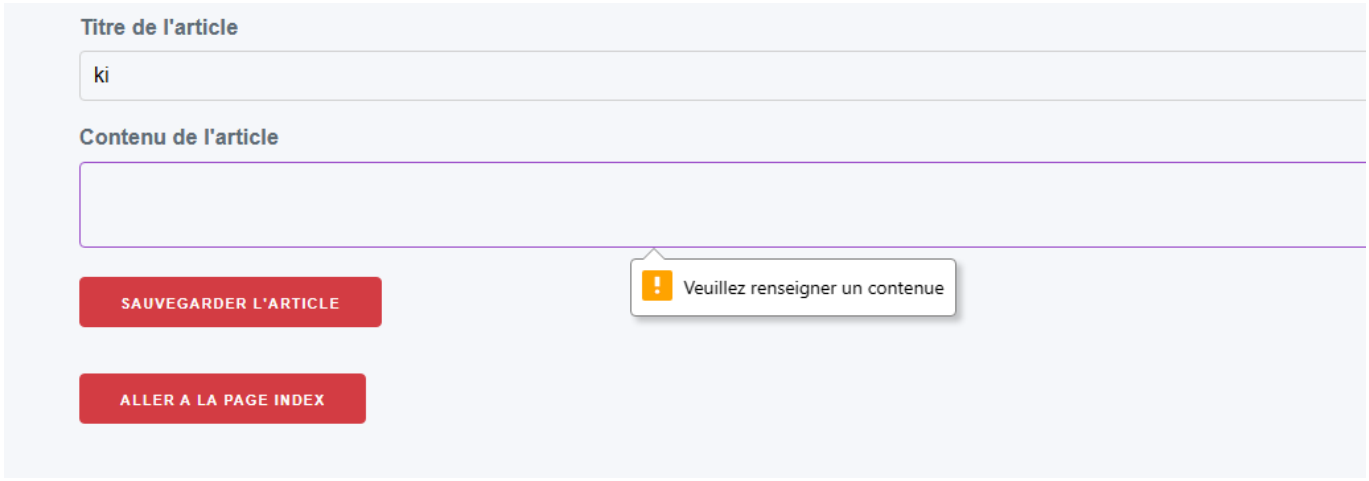
Dans la base de données, le contenu et le titre peuvent être vide « chaine de caractère vide ». Lorsque l'on crée un post, il faudrait mettre un titre et un contenu obligatoirement.

- Comment faire pour indiquer à CakePHP les exigences de validation ? Quel fichier modifier ?
  Il faut aller dans le modele articleTable pour valider les données
- Rajouter dans le fichier :

```
use Cake\Validation\Validator;
public function validationDefault(Validator $validator) {
        $validator
            ->notEmpty('title')
            ->notEmpty('content');

        return $validator;
    }
```

Le méthode validationDefault() indique à CakePHP comment valider vos données lorsque la méthode save() est appelée. Ici, j'ai spécifié que les deux champs "content" et "title" ne doivent pas être vides. Le moteur de validation de CakePHP est puissant, il dispose d'un certain nombre de règles intégrées (code de carte bancaire, adresse emails, etc.)


Ajouter un article
Titre de l'article
Contenu de l'article
SAUVEGARDER L'ARTICLE
Veuillez renseigner un contenue
ALLER A LA PAGE INDEX


Titre de l'article
ki
Contenu de l'article
SAUVEGARDER L'ARTICLE
Veuillez renseigner un contenue
ALLER A LA PAGE INDEX

Maintenant que vos règles de validation sont en place, utilisez l'application pour essayer d'ajouter un article avec un titre et un contenu vide afin de voir comment cela fonctionne. Puisque que nous avons utilisé la méthode Cake\View\Helper\FormHelper::input() du helper "Form" pour créer nos éléments de formulaire, nos messages d'erreurs de validation seront affichés automatiquement.

- Créez un post avec les champs vides pour voir les changements.
- Insérer des posts (5 articles).

2.14\ Editer un post

- Quelle action devez-vous créer ? il faut edit dans le controller articles
- Aura-t-elle un paramètre ? oui il faut
- Ecrivez la signature de la méthode et insérez le code ci-dessous :

```
$lePost = $this->Posts->get($id);
if ($this->request->is(['post', 'put'])) {
     $this->Posts->patchEntity($lePost, $this->request->data);
     if ($this->Posts->save($lePost)) {
          $this->Flash->success(__('Votre post a été mis à jour.'));
          return $this->redirect(['action' => 'index']);
     }
     else $this->Flash->error(__('Impossible de mettre à jour votre post.'));
}
$this->set(compact('lePost'));
```

2.15\ Créez une autre vue

- Comment s'appelle le fichier à créer ? Créez-le
- Insérez le code ci-dessous :

```
<h1>Modifier le post</h1>
<?php
    echo $this->Form->create($lePost);
    echo $this->Form->input('title');
    echo $this->Form->input('content', ['rows' => '3']);
    echo $this->Form->button(__("Mettre à jour le post"));
    echo $this->Form->end();
?>
<br/>
<?php unset($lePost); ?>
```

- Créez un lien retour pour retourner vers la page index

- Dans la vue, modifiez l'affichage de la 1ère ligne pour afficher par exemple :
  Modifier le post "le super titre" (id = 2)
- Gérer le problème d'appeler edit sur aucun id ou sur un id inexistant

2.16\ Modifiez la vue index pour rajouter edit

- Modifiez la vue index pour rajouter le lien edit sur tous les posts

2.17\ Supprimer un post

- Quelle action devez-vous créer ? delete
- Aura-t-elle un paramètre ? oui id de l'article
- Ecrivez la signature de la méthode et insérez le code ci-dessous :
  Public function delete ($id)

```
$this->request->allowMethod(['post', 'delete']);
$lePost = $this->Posts->get($id);
if ($this->Posts->delete($lePost)) {
    $this->Flash->success(__("Le post {0} d' id {1} a bien été supprimé ! ",$lePost->title, $lePost->id));
    return $this->redirect(['action' => 'index']);
}
```

- Faut-il obligatoirement une vue ? Non car quand on supprime on n'a rien afficher
- Modifiez la page index en rajoutant le lien de suppression :

```
<?= $this->Form->postLink(
    __('Supprimer'),
    ['action' => 'delete', $post->id],
    ['confirm' => __("Vraiment supprimer {0} dont l'id vaut {1} ",$post->title, $post->id)])
?>
```

2.18\ Route par défaut

Nous allons faire en sorte que juste taper http://localhost/blog_app redirige directement sur le controlleur posts

- Allez dans **config/routes.php. Commentez la ligne :**

```
$routes->connect('/', ['controller' => 'Pages', 'action' => 'display', 'home']);
```

- Rajouter la route suivante :

```
$routes->connect('/', ['controller' => 'Posts', 'action' => 'index']);
```

- Testez en tapant l'adresse