**RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE**

**MINISTÈRE DE L’ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE**

**UNIVERSITE FERHAT ABBES de SETIF 1**



**Faculté des Sciences**

**Département d'Informatique**

**Projet de Fin d'études**

**En vue d’obtention du diplôme de   
« LICENCE académique en informatique****»**

**THEME :**

Développement d’une application web E-Learning.

**Présenté par : Encadré par :**

* SEFIANNE RAYHANA. Mr. BOUAAMARI.
* GUERRAS AICHA.

# Résumé:

L’application des Technologies de l’Information et de la Communication (TIC) au domaine de la formation a conduit à la naissance de la « formation à distance » appelé aussi « E-Learning ». Ce dernier est décrit comme la naissance du multimédia (texte, son, image, vidéo) et les moyens de communication (diffusion en ligne, discussion instantanée,…). L’enseignement à distance, devenu désormais un atout important pour la formation, passe en particulier par la mise en place d’une plate-forme d’apprentissage à distance.

Dans le cadre de notre projet, nous allons réaliser une plate-forme en ligne et interactive pour les professeurs et étudiants pour la formation et l’apprentissage ; où ses étudiants et enseignants peuvent s’inscrire sur cette plateforme. L’étudiant pourra suivre des cours que les enseignants auront préalablement publiés, et évaluer ses connaissances en faisant des travaux pratiques et examens. L’administrateur gère la maintenance de la plateforme et gère aussi les accès et les droits des utilisateurs.

* **Outils et méthodes :**

Laravel, Bootstrap, Editeur code PHP, Outil d’administration de bases de données.

# 

# Table des matières :

………………………………………………………………………………..2

…………………………………………………………………...3

……………………………………………………………………...5

Introduction général:…………………………………………………………………6

Chapitre 1 : Le domaine « E-Learning ».....................................................................7

…………………………………………………..

………………………………………………………………….

……………………

……………………………………

…………………………………

……………………….

………………………………………………….

……………………………………………………….

…………………………………………………………….

……………………………………………….

……………………………………………..

………………………………………………..

……………………………

………………………………………………………………..

Chapitre 2 : Le Framework Laravel.........................................................................11

…………………………………………………

……………………………………………….

………………………………………………………

………………………………………

…………………………………………………………………….

……………………………………….

……………………………………………………..

……………………………………..

…………………………………………………….

……………….

……………………….

……………………………………

……………………………………………

………………………………………………………….

………………………………………………………………

………………………………………………

…………………………………………………

……………………………………………………….

………………………………………………….

………………………………………………………

………………………………………

…………………………….

………….

………………..

…………………………………………………………………..

………………………………………………………………

…………………………………………………………………..

……………………………………………………….

…………………………….

…………

…………..

………………………………………………………………..

**…………………………………………**

…………………………………………………

…………………………………………

………………………………………………………

……………………………………………..

…………………………………………

………………………………………………………………

**………………………………………………………………..**

…………………………………………………………………….

# 

# Liste des figures :

# 1:Figure1 : Organisation de Laravel .

# Introduction général:

L’explosion d’internet et son accès au plus grand nombre permet d’imaginer aujourd’hui des formations accessibles dans de multiple domaine que ce soit d’un point de vue universitaire ou encore dans le cadre de la formation continue, L’utilisateur peut profiter d'un accès pratiquement illimité à des documents éducatifs ou amusants et à des services qui permettent de communiquer avec des personnes d'autres pays et d'autres cultures pour s'enrichir.

De plus l'arrivée de l'ordinateur multimédia et des réseaux informatiques a ouvert la porte à de nombreuses possibilités d'apprentissage. Leur intégration dans le milieu scolaire permet d’adapter l’enseignement aux besoins des élèves pour passer de la méthode dite traditionnelle à des activités d’apprentissage centrés sur eux.

La formation à distance ou bien le E-Learning pourra jouer un rôle accompagnateur pour assurer des formations à moindre coût par rapport aux formes classiques basées sur la présence des participants, dans ce cadre, notre projet a pour but de réaliser une application web e-Learning interactive pour les professeurs et les étudiants pour la formation à distance. Concernant le développement cette application devrait être accessible via des clients web et développée à l’aide du Framework Laravel (programmation orienté objet PHP), base de donnée relationnelle (MySQL).

Le travail que nous allons présenter sera organisé en 4 chapitres :

* Dans le **1er chapitre** «Le domaine « E-Learning » » nous accordons une place particulière pour citer et définir les différentes notions qui concernent le E-Learning.
* Dans le **2ème chapitre** « le Framework Laravel» nous essayons de définir les différentes notions du Framework Laravel.
* Dans le **3ème chapitre** « Analyse et Conception».
* Dans le 4**ème chapitre** « Réalisation ».

# 

# **Chapitre 1 :** Le domaine « E-Learning » :

## **1.1 Introduction du chapitre :**

L’utilisation d’Internet à des fins d’enseignement sous de multiples formes se développe actuellement dans tous les secteurs de l’éducation et particulièrement dans le monde universitaire. Cette nouvelle modalité d’enseignement offre des avantages dont celui notamment de faciliter l’enseignement à distance.

Le E-Learning découle de l’application de l’internet et des Nouvelles Technologies de l’Information et de la Communication (NTIC) au domaine de la formation. Il est basé sur un apprentissage en ligne centré sur le développement de compétences et organisé dans le cadre d’une interaction entre l’encadrant et les participants à la formation.

## **1.2 Définition :**

L’e-Learning est un terme anglais qui veut dire « apprentissage par des moyens électroniques ». Il se réfère à l’utilisation du web et des nouvelles applications technologiques d’apprentissage distribué pour améliorer la qualité de l’apprentissage en facilitant d’une part l’accès à des ressources et à des services, d’autre part les échanges et la collaboration à distance. Le e-Learning est utilisé dans les programmes d’éducation nationale, les programmes de l’enseignement supérieur, les programmes de formation de l’entreprise, et les programmes de formation continue.

## **1.3 Les avantages et les inconvénients du « E-Learning » :**

### **1.3.1 Les avantages du e-Learning :**

* l’e-Learning facilite l’accès à la connaissance, aux ressources multimédia, et logiciels didactiques qui existent sur Internet.
* La formation est ouverte à toute personne, quels qu’en soient son âge, son niveau d’instruction, sa catégorie socioprofessionnelle, etc.
* Il permet a l’apprenant de : s’informer, s’exprimer, communiquer, et échanger.
* l’e-Learning fournit une flexibilité aux apprenants qui désormais peuvent choisir le temps et le lieu de leur apprentissage, ainsi que le rythme et la cadence de leur de leur progrès.
* l’e-Learning permet le déploiement rapide des stratégies de formation dans les entreprises multinationales.
* l’e-Learning réduit le coût de formation et permet à l’État de prendre en charge les besoins éducatifs des populations éloignées des centres urbains. Ces populations se trouvent souvent exclues du système de l’éducation nationale, et ce compte tenu des coûts investissements requis par l’enseignement présentiel.

### **1.3.2 Les inconvénients du e-Learning :**

* Manque d’accès internet ou les problèmes technique (perturbation du réseau de communications, pannes des ordinateurs, terminaux ou serveurs, attaques des documents électroniques de cours par des virus ou des pirates, etc.)
* L’absence physique de l’enseignant
* L'e-Learning nécessite une maîtrise suffisante des outils informatiques et d'Internet pour pouvoir suivre la formation.

## **1.4 Techniques de communication dans E-Learning :**

On distingue trois modes de communication ou diffusion du E-Learning :

### **1.4.1 Asynchrone :**

Les activités pédagogiques ne se font pas en temps réel mais elle est différée. L’enseignant planifie les activités pédagogiques et conçoit le site à l’avance ; ensuite, il donne aux apprenants une période de temps pour qu’ils accomplissent leurs devoirs et tâches pédagogiques. Les apprenants inscrits au cours forment une cohorte de classe virtuelle et avancent en groupe.

### **1.4.2 Synchrone :**

Méthode de communication en temps réel basé sur la parole et l’écoute. Les apprenants peuvent poser des questions et recevoir des réponses. Le dialogue entre les formateurs et les apprenants peut se faire par : Chat, Webcam, micro, partage d’application, fonctionnalités de prise en main à distance, tableau blanc et téléphone.

### **1.4.3 Mixte :**

C’est une méthode qui combine les deux méthodes précédentes : synchrone et asynchrone, il peut utiliser pour homogénéiser les savoir avant une présentation, une conférence, une intervention ou une formation.

## **1.5 L’objectif de E-Learning :**

L’objectif principal de e-Learning est d’améliorer la qualité de l’apprentissage et non de se substituer aux modes traditionnels d’enseignement. Les moyens pour atteindre cet objectif sont multiples, complémentaires et indépendants tout en gardant l’autonomie de ces différents moyens : accès à des ressources variées, offres de services de tutorat à distance, outils de communication, résolution d’exercices, échanges, et collaboration à distance.

## **1.6 Le Principe du E-Learning :**

Le principe de E-Learning étant de pouvoir accéder à des cours depuis un poste distant, les lieux nécessaires au suivi d’une formation (établissements, classes..) n’existent plus physiquement, ils sont remplacés par le Système de Gestion des Cours ou S.G.C (plate-forme). Le SGC est un lien entre les apprenants, les cursus, les tuteurs, les ressources et les contenus présents dans le système.

L’apprenant, via une plate-forme, se verra attribuer un certain nombre de modules de cours, d’exercices, d’évaluations qu’il devra effectuer selon un programme établi. Le rôle du tuteur (ou formateur) est de gérer les apprenants qu’il doit suivre, ainsi il pourra leur affecter des ressources à consulter, des cours à étudier, des évaluations afin de se rendre compte de la bonne assimilation des contenus proposés. La communication entre tous ces acteurs se fait via internet.

## **1.7 Type des projets E-Learning:**

On peut classer les projets e-Learning en quatre catégories :

* des outils d’aide aux formateurs qui utilisent le réseau pour préparer et suivre un cours présentiel. Cela concerne les multiples cours et exercices mis en ligne en format statique HTML ;
* des moyens pour mettre à disposition des savoirs par des portails donnant accès à des documents pédagogiques. Ils jouent le rôle de complément de formation en libre service. Ce sont, dans la plupart des cas, des systèmes de gestion de contenu pédagogique capable de générer des cours et des exercices à la demande ;
* des guides à distance avec définition de parcours pédagogiques. Ce sont des systèmes et plates-formes plus complexes qui concentrent des modules de création de contenu, de suivi et d’évaluation des résultats et définissent pour certaines réalisations le parcours de l'apprenant en fonction de ses propres résultats ;
* des formations intégrées aux dispositifs de gestion des compétences et des connaissances où le e-Learning vient en support à la gestion, à la mise à disposition du savoir de l’entreprise et vient en support également au développement des compétences. Ce type de systèmes caractérise une formation dite d'entreprise.

## **1.8 Les besoins des différents types d’utilisateurs :**

Le système e-Learning regroupe des outils nécessaires aux principaux utilisateurs, à savoir l'enseignant, l'étudiant et l'administrateur.

Les besoins de l'enseignant consistent en : la création d'un contenu pédagogique multimédia, la création des parcours pédagogiques types ou individualisés et, finalement des moyens de suivi des activités des étudiants. Les enseignants intervenant dans une matière ont besoin d'échanger des objets pédagogiques pour la création en collaboration d'un contenu mieux adapté aux besoins d'un certain public ou d’une formation particulière.

L'étudiant de son côté a besoin de consulter en ligne ou de télécharger les contenus pédagogiques qui lui sont recommandés, d'effectuer les exercices qui lui sont soumis, de répondre à des questionnaires proposés et d'obtenir une évaluation de son parcours individualisé.

Les enseignants et les étudiants ont des besoins communs de communication mutuelle ou en groupe par des thèmes de discussion et de collaboration à des documents communs.

L’administrateur d'un système e-Learning installe et assure la maintenance de la plate-forme, gère les accès et les droits des uns et des autres, crée des liens avec les systèmes d’information externes (scolarité, catalogues, ressources pédagogiques, etc.).

## **1.9 Conclusion :**

# 

# **Chapitre 2 :** Le Framework Laravel :

## **Introduction du chapitre :**

Laravel, créé par Taylor Otwel, initie une nouvelle façon de concevoir un framework en utilisant ce qui existe de mieux pour chaque fonctionnalité.

Par exemple toute application web a besoin d’un système qui gère les requêtes HTTP. Plutôt que de réinventer quelque chose, le concepteur de Laravel a tout simplement utilisé celui de Symfony en l’étendant pour créer un système de routage efficace.

De la même manière, l’envoi des emails se fait avec la bibliothèque SwiftMailer. En quelque sorte Otwel a fait son marché parmi toutes les bibliothèques disponibles. Mais Laravel ce n’est pas seulement le regroupement de bibliothèque**s** existantes, c’est aussi de nombreux composants originaux et surtout une orchestration de tout ça.

Vous allez trouver dans Laravel :

* un système de routage (RESTFul et ressources),
* un créateur de requêtes SQL et un ORM,
* un moteur de Template,
* un système d’authentification pour les connexions,
* un système de validation,
* un système de pagination,
* un système de migration pour les bases de données,
* un système d’envoi d’emails,
* un système de cache,
* un système de gestion des événements,
* un système d’autorisations,
* une gestion des sessions,
* un système de localisation,
* un système de notifications…

**1.3 Installation et organisation :**

Le Framework Laravel a quelques exigences système. Toutes ces exigences sont satisfaites par la machine virtuelle Laravel Homestead, il est donc fortement recommandé d'utiliser Homestead comme environnement de développement Laravel local.

Cependant, si vous n'utilisez pas Homestead, vous devrez vous assurer que votre serveur répond aux exigences suivantes:

* PHP >= 7.2.5
* BCMath PHP Extension
* Ctype PHP Extension
* Fileinfo PHP extension
* JSON PHP Extension
* Mbstring PHP Extension
* OpenSSL PHP Extension
* PDO PHP Extension
* Tokenizer PHP Extension
* XML PHP Extension

### **1.3.1 INSTALLATION:**

Laravel utilise Composer pour gérer ses dépendances. Donc, avant d'utiliser Laravel, assurez-vous que Composer est installé sur votre machine.

#### 1.3.1.1 Composer :

Composer est un outil de gestion des dépendances pour PHP (**Dependancy Manager**). Il permet de déclarer les bibliothèques, desquelles dépend un projet en PHP, et il va les gérer (installer / mettre à jour).

Pour télécharger et installer Composer : [**https://getcomposer.org/download/**](https://getcomposer.org/download/)

#### 1.3.1.2 Laravel :

**a. Installation avec Laravel Installer :**

Tout d'abord, téléchargez le programme d'installation de Laravel à l'aide de Composer:

composer global require laravel/installer

Il faut ensuite informer la variable d’environnement path de l’emplacement du dossier …/composer/vendor/bin.

Une fois installée, la commande laravel new créera une nouvelle installation Laravel dans le répertoire que vous spécifiez. Par exemple, laravel new blog créera un répertoire nommé blog contenant une nouvelle installation de Laravel avec toutes les dépendances de Laravel déjà installées.

**b. Installation avec Composer Create-Project:**

Alternativement, vous pouvez également installer Laravel en émettant la commande Composer create-project dans votre terminal:

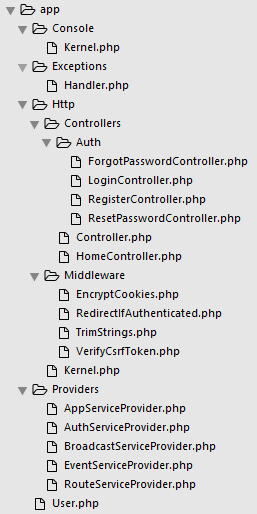
composer create-project --prefer-dist laravel/laravel blog

Si PHP est installé localement et que vous souhaitez utiliser le serveur de développement intégré de PHP pour servir votre application, vous pouvez utiliser la commande php artisan serve. Cette commande démarrera un serveur de développement à **http://localhost:8000**.

### **1.3.2 Organisation de Laravel :**

Maintenant qu’on a un Laravel tout neuf et qui fonctionne voyons un peu ce qu’il contient :

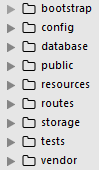
- Dossier app qui contient les éléments essentiels de l’application :



#### FIGURE 1 : Organisation du dossier app.

* **Console**: toutes les commandes en mode console,
* **Http**: tout ce qui concerne la communication : contrôleurs, middlewares (il y a 4 middlewares de base qui servent à filtrer les requêtes HTTP) et le kernel,
* **Providers**: tous les fournisseurs de services (providers), il y en a déjà 5 au départ. Les providers servent à initialiser les composants.
* **User:** un modèle qui concerne les utilisateurs pour la base de données.
* Évidemment tout cela doit vous paraître assez nébuleux pour le moment mais nous verrons en détail ces éléments au fil du cours. Et on verra d’ailleurs que seront créés bien d’autres dossiers selon nos besoins.

-Autre dossiers:

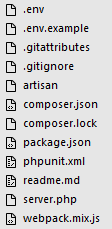


#### FIGURE 2 : Autre dossiers d’une application Laravel.

-Voici une description du contenu des autres dossiers :

* **bootstrap**: scripts d’initialisation de Laravel pour le chargement automatique des classes, la fixation de l’environnement et des chemins, et pour le démarrage de l’application,
* **public**: tout ce qui doit apparaître dans le dossier public du site : images, CSS, scripts…
* **config:** toutes les configurations: application, authentification, cache, base de données, espaces de noms, emails, systèmes de fichier, session…
* **database**: migrations et populations,
* **resources**: vues, fichiers de langage et assets (par exemple les fichiers Sass),
* **routes**: la gestion des urls d’entrée de l’application,
* **storage**: données temporaires de l’application : vues compilées, caches, clés de session…
* **tests**: fichiers de tests,
* **vendor**: tous les composants de Laravel et de ses dépendances (créé par composer).

-Fichiers de la Racine:



**FIGURE 3 : Organisation des fichiers de la racine d’une application Laravel.**

-Il y a un certain nombre de fichiers dans la racine dont voici les principaux :

* **artisan:** outil en ligne de Laravel pour des tâches de gestion,
* **composer.json:** fichier de référence de composer,
* **package.json**: fichier de référence de npm pour les assets,
* **phpunit.xml:** fichier de configuration de phpunit (pour les tests unitaires),
* **.env**: fichier pour spécifier l’environnement d’exécution.

## **1.4 MVC:**

****

#### FIGURE 4 : L’architecture MVC.

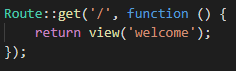
Laravel est un framework web open-source écrit en PHP respectant le principe modèle-vue-contrôleur et entièrement développé en programmation orientée objet.

### **1.4.1 Le routage(Routing) :**

Toutes les Routes de Laravel sont définies dans le dossier **Routes\**. Ces fichiers sont chargés automatiquement par le Framework depuis ce dossier.

Pour la majorité des applications Laravel, les Routes sont définies dans le fichier **routes\web.php**. Ce fichier contient les Routes de l’interface Web de l’application.

Voici la Route défini par défaut par Laravel :

****

#### FIGURE 5 : La route définie par défaut par Laravel.

#### -Cette Route :

* prend en charge une requête GET (**GET request**),
* est utilisée pour la page d’accueil (**Home : ‘/’**),
* renvoie la **View** (page) nommée **welcome**.

-Il est indispensable de connaître les principales méthodes du HTTP :

Route::get ($uri, $callback);

Route::post ($uri, $callback);

Route::put ($uri, $callback);

Route::delete ($uri, $callback);

* **GET:** c’est la plus courante, on demande une ressource qui ne change jamais, on peut mémoriser la requête, on est sûr d’obtenir toujours la même ressource,
* **POST:** elle est aussi très courante, là la requête modifie ou ajoute une ressource, le cas le plus classique est la soumission d’un formulaire (souvent utilisé à tort à la place de PUT),
* **PUT**: on ajoute ou remplace complètement une ressource,
* **DELETE**: on supprime une ressource.

### **1.4.2 Controller :**

Au lieu de définir la logique de réponse à différentes requêtes dans les fichiers du Routing, il est recommandé d’organiser ces comportements dans des **Controllers**.

Les Controllers permettent de regrouper les logiques de réponse aux requêtes http reliées dans la même classe.

On trouve les Controllers dans le dossier **App\Http\Controllers**. La création des Controller peut être faite manuellement, cependant, l’utilisation de la console intégrée de Laravel : **Artisan** s’avère une meilleure solution. Donc, Dans la console entrez cette commande : php artisan make:controller Name\_Controller.

### **1.4.3 Migrations et modèles :**

#### 1.4.3.1 Les migrations :

-Une migration permet de créer et de mettre à jour un schéma de base de données. Autrement dit, vous pouvez créer des tables, des colonnes dans ces tables, en supprimer, créer des index… Tout ce qui concerne la maintenance de vos tables peut être pris en charge par cet outil. Vous avez ainsi un suivi de vos modifications.

Pour créer une migration nous allons utiliser Artisan. Dans la console entrez cette commande :php artisan make:migration create\_table\_name

-Les Migrations sont stockées dans le dossier **database/migrations**.

-L’exécution d’une Migration permet de créer la table dans la BD avec le schéma défini dans la Migration. La commande permettant d’exécuter les Migrations est :

php artisan migrate

-Il existe une commande qui permet de créer une Migration avec le Model :

php artisan make:model --migration NomModel

-Ou bien : php artisan make:model -m NomModel.

#### 1.4.3.2 Les modèles :

**-**L'**ORM Eloquent** inclus avec Laravel fournit un outil simple pour travailler avec la BD.

-ORM (acronyme de Object-relationnel Mapping ou en bon Français un mappage objet-relationnel) est très performant, où tous les éléments de la base de données ont une représentation sous forme d'objets manipulables.

-Avec Eloquent une table est représentée par une classe qui étend la classe Model.

-Tous les **Models** de l’application Laravel sont stockés dans le dossier **App\**. Pour créer un model il suffit d’exécuter la commande: php artisan make:model NomModel.

-Par convention le nom du Model est le singulier du nom de la table qi lui correspond dans la BD. Si le nom de la table est différent, il faut le spécifier dans le model :

protected $table = 'my\_clients';

-Eloquent suppose également que chaque table possède une colonne de clé primaire appelée **id**. Vous pouvez définir une propriété **$primaryKey** pour remplacer cette convention : public $primaryKey = ‘ncin’ ;

-De plus, Eloquent suppose que la clé primaire est une valeur entière incrémentée, si vous souhaitez utiliser une clé primaire non incrémentée ou non numérique, vous devez définir la propriété publique **$incrementing** sur votre modèle sur **false :**

public $incrementing = false ;

-Par défaut, Eloquent s'attend à ce que les colonnes **created\_at** et **updated\_at** existent sur vos tables. Si vous ne souhaitez pas que ces colonnes soient gérées automatiquement par Eloquent, définissez la propriété **$timestamps** sur votre modèle sur **false**: public $timestamps = false;

### **1.4.4 Les vues :**

-Le Framework Laravel intègre un **Templating engine** (moteur de génération de Templates) simple et très puissant : **Blade**.

-Contrairement aux autres Templating engines PHP populaires (Twig, Mustache, …) Blade n’empêche pas l’utilisation du code PHP dans les Views. En fait, toutes les Views Blade sont compilées dans du code PHP pur, et mises en cache jusqu’à ce qu’elles soient modifiées, ce qui signifie que Blade ajoute zéro charge à votre application.

Les Views créées avec Blade utilisent l’extension **.blade.php** et sont, typiquement, enregistrées dans le dossier **resources/views**.

-On va créer, dans le dossier dédié aux Views par défaut, un dossier nommé **layouts**. Puis, il faut créer dans ce dossier un fichier Blade nommé **myapp**. Et enfin, on va définir dans ce fichier un layout qui doit être utilisé par les autres Views.

- L’idée est de mettre dans ce fichier **myapp.blade.php** le contenu qui sera utilisé dans toutes pages (le layout général). Il suffit ensuite d’intégrer les sections dynamiques avec ce layout dans les Views.

- L’instruction **@yield** est utilisée dans les Views pour afficher le contenu d’une section définie ailleurs.

<body>

@yield(‘content’)

</body>

- Pour permettre à une View d’utiliser une autre, il faut utiliser l’instruction Blade **@extends**. Cette instruction permettra de spécifier quel layout la View doit **hériter**.

Les Views qui héritent un layout Blade peuvent en injecter du contenu en utilisant l’instruction **@section**.

@extends(‘layouts.app’)

@section(‘content’)

{{-- contenu à injecter --}}

@endsection

- Laravel intègre par défaut **Laravel Mix** qui fournit une API fluide pour définir les étapes de construction du Webpack de votre application en utilisant plusieurs préprocesseurs CSS et JS.

- Laravel est livré aussi avec un autre fichier de style qui contient tous les styles prédéfinis (y compris **Twitter Bootstrap**). Pour que nos pages puissent utiliser ces styles précompilés, on doit ajouter le lien nécessaire dans le layout :

<link rel="stylesheet" href="{{asset(‘css/app.css’)}}">

Ce lien permettra de référencer le fichier **public/css/app.css** qui est une compilation des fichiers **SASS** (le préprocesseur CSS intégré avec Laravel), et qui se trouve dans le dossier **resources/assets/sass**. Pour pouvoir utiliser les modules importés par le fichier **app.scss** (y compris Bootstrap), et qui sont déjà définis dans le fichier **package.json**, on a besoin d’exécuter la commande suivante (**npm** doit être installé auparavant) : npm install. Cette commande permettra d’installer tous les modules complémentaires définis comme des dépendances à l’application Laravel.

Le **npm** (**Node Package Manager**) est un gestionnaire de packages pour JavaScript, et peut être installé via **Node.js** en visitant cette adresse :

[**https://nodejs.org/en/download/**](https://nodejs.org/en/download/)

- Pour modifier la mise en forme du style prédéfini, il faut modifier dans le fichier **ressources/assets/sass/\_variables.scss** et non pas dans le fichier **public/css/app.css**, car ce dernier est la version compilée du premier.

Toute modification doit être suivi de recompilation des fichiers source en exécutant la commande : npm run dev.

## **1.5 Récupération et insertion des données avec Eloquent :**

### **1.5.1 Générer un Controller avec ressources :**

-On peut assigner les Routes du typique **CRUD (Create, Read, Update & Delete)** à un Controller avec une seule ligne commande grâce au Routing de ressources.

Pour créer un Controller avec ressources on utilise la commande suivante :

php artisan make:controller ControllerNom --resource

index()create()store()show() edit()update()destroy()

- Pour référencer ces ressources dans le fichier des Routes, on n’a pas besoin d’écrire un Route pour chaque méthode, on peut écrire une seule ligne permettant de générer les Routes pour toutes les ressources à la fois :

Route::resource(‘photos, ‘PhotosController’) ;

### **1.5.2 Récupération des données :**

**1.5.2.1 La méthode index () :**

La première étape est d’indiquer au Controller quel Model il va utiliser. Ensuite, dans la méthode **index()**, ajouter la méthode prédéfinie **all()**, et Eloquent va se charger de récupérer toutes les lignes de la table.

Voici un exemple :

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\Flight;

class FlightsController extends Controller

{

public function index()

{

$flights = Flight::all();

return $flights;

}

- La méthode **all()** permet ainsi de récupérer tous les enregistrements de la table en question.

- Pour afficher les résultats de la requête, il suffit de parcourir la variable qui héberge le résultat :

foreach ($flights as $flight) {

echo $flight->name;

}

#### 1.5.2.2 La méthode show () :

Eloquent nous permet également de récupérer des enregistrements individuels en utilisant **find()** ou **first()**. Au lieu de renvoyer une collection de modèles, ces méthodes renvoient une seule instance de modèle comme le montre les exemples suivants :

// Récupérer un Model par sa clé primaire...

$flight = App\Flight::find($id);

// Récupérer le premier Model correspondant à la contrainte de la requête...

$flight = App\Flight::where('active', 1)->first();

### **1.5.3 insertion de données :**

#### 1.5.3.1 La méthode store () :

-Cette méthode consiste en fait à la création d’un nouvel enregistrement au niveau de la table de la BD.

-Pour ce faire, on doit tout simplement créer une nouvelle instance de notre **Model**, affecter les valeurs récupérées depuis le formulaire aux attributs de l’objet, puis appeler la méthode **save()**.

-Voici un exemple illustrant les étapes décrites ci-dessus :

public function store(Request $request)

{

$flight = new Flight;

$flight->name = $request->name;

$flight->save();

}

#### 1.5.3.2 Les messages d’alerte :

L’ajout des messages d’alertes dans Laravel est aussi simple. Laravel contient une classe nommé **MessageBag** qui contient une variété de méthodes pratiques pour utiliser les messages d’erreur.

Une variable prédéfinie **$errors**, qui est une instance de cette classe, est automatiquement instancié et mise à la disposition dans toutes les **Views**.

## **1.6 Authentification :**

-Le fichier de configuration d'authentification se trouve dans **config/auth.php.**

-Une simple commande est utilisée pour construire les Routes nécessaires à ce processus : php artisan make:auth.

-Cette commande devrait être utilisée sur de applications nouvellement créées, et installera une View de Layout, des Views pour l’inscription et la connexion, ainsi que des Routes pour tous les Requests d'authentification. Un **HomeController** sera également généré pour gérer les demandes post-connexion au tableau de bord de votre application.

## **1.7 Conclusion :**

# **Chapitre 3 : Analyse et conception** :

## **3.1 Introduction du chapitre :**

Comme n’importe quel type de projet, un projet informatique nécessite une phase d’analyse, suivi d’une étape de conception. La phase d’analyse est une étape essentielle pour spécifier de façon détaillée les besoins des utilisateurs de la plateforme avec une modélisation par les cas d’utilisation en utilisant le langage UML. On montre aussi dans cette étape l’interaction entre les utilisateurs et la plateforme. Après validation de notre compréhension du besoin, nous imaginons la solution. C’est la partie analyse de la solution. La phase de conception est consacré à concevoir l’aspect fonctionnel de la solution qui répond aux spécifications fonctionnelle que nous avons parlé précédemment. Dans ce sens, le diagramme UML présenté est le diagramme de classe.

## **3.2 Analyse des besoins :**

### **3.2.1 Les besoins fonctionnels :**

### **3.2.2 Les besoins non fonctionnels :**

## **3.3 Les cas d’utilisation :**

### **3.3.1 Identification des acteurs :**

### **3.3.2 Identification des cas d’utilisation :**

### **3.3.3 Présentation des diagrammes des cas d’utilisations :**

#### 3.3.3.1 Diagramme de cas d’utilisation générale :

#### 3.3.3.2 Diagramme de cas d’utilisation détaillé « Authentification » :

#### 3.3.3.3 Diagramme de cas détaillé pour «Gestion des ressources par l’enseignant » :

#### 3.3.3.4 Diagramme de cas détaillé pour «Gestion des comptes par l’administrateur » :

#### 3.3.3.5 Diagramme de cas détaillé pour «Consulter la liste des cours par l’étudiant» :

## **3.4 Diagramme de classe :**

### **3.4.1 Présentation du diagramme de classe :**

#### 3.4.1.1 Diagramme de classe de conception de E-Learning :

#### 3.4.1.2 Description des classes d’objet et d’association :

## **3.5 Conclusion :**

# Chapitre 4: Réalisation de l’application:

## **3.1 Introduction du chapitre :**

## **3.2 Les outils de développements :**

### **3.2.1 Ressources matérielles :**

### **Ressources logicielles :**

* **Système d’exploitation** : Windows.
* **Plateforme Logicielles** :
* **Visual Studio Code** :
* **XampServer** :
* **PHP My Admin:**
* **Visual Paradigm :**
* **Mozilla Firefox, Google Chrome** et **Opera :**
* **Langages, Framework et librairies utilisées :**
* **Laravel :**
* **Html :**
* **Css :**
* **Sql :**

## **3.3 La base de données :**

### **3.3.1 Tables et attributs :**

Notre base de données contient 11 tables :

* **Table Administrateur :**
* **Table Enseignant :**
* **Table Etudiant :**
* **Table Cour :**
* **Table Leçon :**
* **Table Examen :**
* **Table Exercice :**
* **Table Solution :**
* **Table Evaluation :**
* **Table Inscriptions :**

**3.4 Les interfaces de l’application :**

## **3.5 Conclusion :**

# Conclusion général:

# Bibliographie:

[1] :« Conception et réalisation d’une plateforme web pour la formation à distance», **RAHIL Hassna,SARRI Sadia.**, mémoire Master 2.

[2] « Guide pratique du e-Learning : Statégie, pédagogie et conception avec le logiciel Moodle», Sid Ahmed BENRAOUANE.

[3]Fascicule de l’atelier Framework côté serveur : Le Framework Laravel. Licence, Houcem Hedhly , Tunisie. 2018. ‌hal-02049177‌.

**Webographie:**

[4]https://www.procomptable.com/papier\_recherche/mmbf.htm

[5]<https://openclassrooms.com/fr/courses/3613341-decouvrez-le-framework-php-laravel>