**Mes notes de formation laravel 8 suivant le lien :** <https://m.youtube.com/watch?v=EaSgftRyvAM&list=PLeeuvNW2FHVj4vHJRj9UDeDsXshHlnHJk>

* Composer est le gestionnaire des librairies php. Il permet de gérer les dépendances lors de l’installation des différentes librairies.
* Une route peut renvoyer une chaine de caractère qui sera « parsée » en réponse http
* Une route « closure » retourne une fonction sans nom dont la valeur de retour peut être une vue, une chaine de caractère, une réponse json (sous la forme clé=> valeur), …
* Les vues peuvent et devraient être retournées depuis un contrôleur pour plus de lisibilité du code. Pour retourner une vue depuis le contrôleur, il faut créer le contrôleur (soit manuellement, en prenant soin de mettre le namespace (namespace App\Htpp\Controllers ;) et de faire hériter ce contrôleur de la class Controller, soit en ligne de commande via php artisan make : controller).
* Pour retourner des données à travers une vue, on utilise la fonction ‘compact’ sous forme de chaine de caractère (compact(‘nom\_variable’)). Dans les vues, les variables passées depuis le contrôleur sont utilisées entre double accolades {{ }}. On peut également utiliser la fonction with qui prend en paramètre une clé (la clé étant le nom de la variable sous forme de chaine de caractère, donc entre quotes et sa valeur étant la variable en elle-même. Ex : ->with(‘title’, $title). Lorsqu’il y a plusieurs variables à retourner, on peut utiliser compact ou alors retourner un tableau de clé-valeur

Ex : return view(‘article’, [

‘title’=>$title,

‘title2’=>$title2

]) ;

Généralement, les données passées aux peuvent être des tableaux. Pour utiliser les éléments du tableau dans la vue, on indexe la variable à notre convenance.

* Les « blade » sont des générateurs de templates (gère les vues html en les raccordant avec les contrôleurs)
* Pour passer des paramètres aux routes, on les met entre accolades et comme argument lors de l’appel dans la vue concernée
* Le nom donné aux routes(->name(‘nomroute’)) est avantageux dans la mesure où on peut changer l’arborescence de la route sans avoir à toucher aux fichiers concernés par cette route