Lycapp

Lycée-Collège de l'abbaye de St-Maurice

Travail de Maturité

Création d'une application de travail

Natasha Landry

supervisé par
M. Jan Schonbächler Brown

1890 St-Maurice, 23 septembre 2022

Table des matières

Table des figures

1 Résumé

Dans ce travail de maturité, nous allons voir comment créer une application web dans le but d'aider les étudiants du collège dans leur parcours scolaire. Que ce soit pour rassembler leurs documents à un endroit unique ou pour d'autres fonctionnalités complémentaire, nous réfléchirons à ce qui est le plus propice pour les étudiants, comme un agenda, un carnet de note ou un horaire.
Par ce projet, nous aborderons un certain nombre de technologies, notamment l'implémentation d'une base de donnée, ou encore plusieurs languages comme PHP, Laravel, Javascript par l'utilisation de React, dans le but de gérer dynamiquement le contenu d'une page sans pour autant devoir la recharger à chaque clic.
Nous suivrons le déroulement du projet et détaillerons chaques fonctionnalités et leur création. Nous aborderons également les différents problèmes rencontrés lors de la construction de l'application et les diverses solutions trouvées pour les résoudre.
Les fonctionnalités seront toutes détaillées et imagées par le code et le rendu sur le site.
Enfin, nous détaillerons un bilan et élargirons la discussion sur des différentes pistes à explorer pour compléter l'application ou la perfectionner selon les besoins.
Le site est sur ce lien-ci : `www.lycapp.pouleto.ch/`

2 Introduction

2.1 Qu'est ce que Lycapp ?

Pendant la période de covid, les étudiants se sont retrouvés acculés face à du travail non pas inconsidérable, mais plutôt avec une organisation difficile à gérer.

En effet, tous ayant été pris au dépourvu avec la période de confinement, chacun avait sa manière de communiquer les devoirs à faire. Certains utilisaient des applications, d'autres des moyens plus classiques, comme les mails ou Google Classroom. Sans compter les supports diamétralement différents de chaque cours données : vidéos, exercices, etc. Moodle a été créé, mais pas tout le corps professoral utilisait cette application plutôt pratique. J'ai d'abord remis en cause l'organisation du corps professoral, puis je me suis dit que ce devait être également les élèves qui devraient faire quelques efforts.

Aussi me suis-je posée la question : et si les élèves avaient leur propre application pour stocker leurs cours ? Il leur suffirait simplement de tout ranger dans celle-ci par branche et par ordre alphabétique. Ainsi, chacun pourra gérer ses cours et son rangement de manière optimale. Pour augmenter l'efficacité, ajouter d'autres fonctionnalités étaient utiles, pour avoir tout une application complète, comme un agenda, par heures, jours et mois, un horaire des cours, ou encore une section math.

Aussi ai-je décidé de créer Lycapp, une application de travail pour les étudiants.

Pour créer un outil de travail performant, il était obligatoire de le liée à une base de donnée. Ainsi ai-je choisi le langage Laravel avec une base de donnée MySQL. Pour quelques petites fonctionnalités supplémentaires, React a été implémenté.

2.2 Explication des principales technologies utilisées

Laravel est une application web appelée « framework », ce qui indique qu'il créé les fondations d'un site pour que les développeurs puissent innover en ayant une base. Celui-ci est codé entièrement en programmation orienté objet, c'est-à-dire que tout tourne autour de classes, « d'objet » réutilisables.

Laravel breeze est une implémentation de Laravel, ajoutant au projet un système d'authentification, avec un système de réinitialisation du mot de passe, l'enregistrement et la connexion d'un utilisateur. Il y a également la vérification de l'email ainsi qu'une confirmation du mot de passe.

PHP , autrefois étant l'acronyme de « Personal Home Page Tools », désigne désormais « Hypertext Preprocessor ». Il s'agit un langage de script destiné principalement à du langage web, orienté objet.

Wampserver est une plate-forme de développement Web sous Windows pour des applications Web dynamiques à l'aide du serveur Apache2, du langage de scripts PHP et d'une base de données MySQL. Il possède également PHPMyAdmin pour gérer plus facilement les bases de données.

Javascript est le langage le plus populaire au monde pour coder des sites internet. Il est décrit comme facile et intuitif à prendre en main. Il a été édité par Pluralsight team.

React est un framework Javascript qui créé des interfaces utilisateurs interactives avec un système de composants autonomes.

Latex est un langage de balise servant au traitement de texte. Il est un éditeur de texte spécial : il est fait pour mettre en page des documents scientifiques, rendant la mise en page des documents simple et efficace.

le système du MVC , est un acronyme de "Modèle Vue Contrôleur". Il s'agit d'une architecture et une méthode de conception pour le développement d'applications séparant le modèle de la base de donnée. C'est à dire que dans le fichier "model" le reflet de la table de la base doit être explicité, l'interface que verra l'utilisateur et la logique qui liera la base de données à l'intervace, donc la vue. Cette méthode a été mise au point en 1979 par Trygve Reenskaug. impose la séparation entre les données, les traitements et la présentation, ce qui donne trois parties fondamentales dans l'application finale : le modèle, la vue et le contrôleur. C'est ce qui est utilisé pour cette application.

3 Le site

3.1 Ses débuts

Pour créer le site, l'utilisation de l'invite de commande avec les instructions de la documentation de Laravel était selon moi le plus rapide et le plus pratique. Ainsi ai-je simplement créé un dossier avec le nom puis ai-je lancé dans l'invite de commande la création du site de base.

```
laravel new example-app
php artisan serve
composer require laravel/breeze --dev
```

La dernière ligne de code sert à ajouter laravel Breeze, ce qui est décrit plus bas. Comme il me fallait des utilisateurs différents qui ont des pages personnalisées, j'ai installé Laravel Breeze, qui, comme expliqué quelques pages plus haut, créer un système d'authentification. Celui-ci ajoute plusieurs pages automatiquement, donc des vues, ainsi que plusieurs fonctions dans un controller créé également automatiquement. Celui-ci apporte dans la base de donnée une table nommée « User » avec un id, un nom d'utilisateur, un email avec lequel nous pouvons vérifier l'authenticité de l'utilisateur, une colonne pour se rappeler de l'utilisateur ainsi qu'un mot de passe. Pour apporter de la sécurité, le mot de passe est haché par Bcrypt [1], un algorithme utilisé pour hasher les mots de passe dans le langage PHP. Il utilise un « nombre d'itération » défini par le développeur. Plus le nombre est grand, plus le brouillage du mot de passe est poussé, et plus il est difficile pour les hacker de passer outre le hashage.

Pour lier une base de donnée à l'application, il fallait déjà choisir quelle type de base de donnée j'allais utiliser. Après quelques test, j'ai installé « Wampserver » qui semblait bien pratique, avec une base MySQL, étant un système de gestion intuitif dont les bases sont faciles à prendre en main avec un panel s'appelant « PhpMyAdmin » [2], affichant les différentes bases et. La base de données s'organisent en table, puis celles-ci sont distribuées en colonnes. Plusieurs d'entre elles se répètent dans chacune des tables : les ID, afin de ne pas créer de conflit dans la table, et les « users » pour pouvoir trier les données par utilisateurs.

3.2 Fonctionnalités

3.2.1 Page d'accueil et Dashboard

En créant l'application, Laravel nous fournit une page d'accueil que j'ai trouvé élégante, aussi ai-je décidé de m'y inspirer, afin d'expliquer à l'utilisateur l'existence de Lycapp, en expliquant les différentes fonctionnalités.
En enlevant les liens du magasin en ligne de Laravel et de la bibliothèque Laravel, je les ai remplacés par une mention d'un designer pour l'utilisation d'une icône et un lien vers cet écrit.

1. https ://www.cybersecurity-guide.com/bcrypt-un-algorithme-de-hachage-a-utiliser-en-php/
2. https ://www.wampserver.com

Puisqu'il faut que les utilisateurs s'inscrivent pour avoir accès au reste de l'application web, la page d'accueil contient une explication de toutes les fonctionnalités proposés par l'inscription.

Après l'implantation de Breeze, le dashboard est automatiquement ajouté à la page d'accueil, sous deux formes :

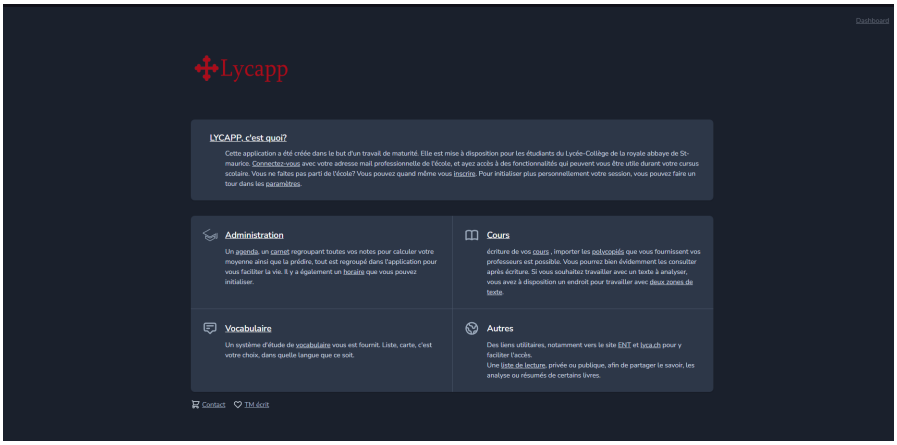

FIGURE 1 – rendu de la page d'accueil

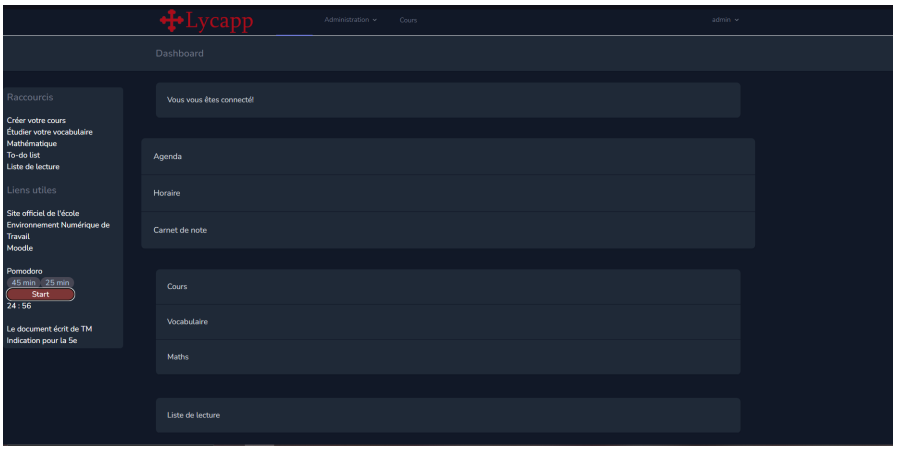

FIGURE 2 – rendu du dashboard

Pour toutes les fonctionnalités auxquelles j'ai réfléchis, j'ai réalisé qu'il me fallait deux endroits pour mettre les liens des pages, aussi ai-je créé deux barres de navigations : une latérale et une au sommet (déjà mise en place par Laravel Breeze) de l'application.
La barre de navigation du haut avait une fonctionnalité très plaisante : lorsque la page se rétrécissait, le contenu de celle-ci devenait de simple flèche pour afficher un menu déroulant. Ne pouvant faire la même chose avec la barre de navigation

du côté, la qualification « Responsive[3] » a été ajoutée, empêchant celle-ci de venir sur le contenu principal. J'ai donc dû adapter pour avoir accès à toutes les fonctionnalités sans la barre latérale, qui contient principalement des raccourcis. Puisque ce n'est que dans un but de rapidité, j'ai pris soin de ne mettre que des liens où on peut y accéder sans. Autrement, si la page prends moins de place et que la barre de navigation latérale disparait, certaines fonctionnalités ne pourront plus être utilisées.
Aussi, dans le cas où la page prend plus de place que l'écran et qu'il faille scroller pour accéder au reste, j'ai ajouté dans le style le type « fixed », pour que la barre de navigation reste figé à un endroit précis de l'application.

3.2.2 la base de donnée

Pour lier une base de donnée à mon site, il faut simplement changer quelques paramètres dans le fichier .env. Celui-ci contient tous les paramètres à remplir :

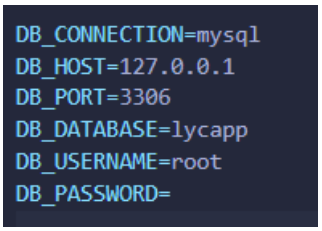

FIGURE 3 – .env

Comme on peut le constater, il suffit de remplir les quelques paramètres « name » et « password » pour lier la base de donnée au site internet. Pour la construire, j'ai d'abord créé un nom sans mot de passe puisque tout ne se trouve que sur mon ordinateur personnel. Ce dernier est donc illusoire jusqu'à ce que je mette en ligne l'application, où je changerai avec un mot de passe sécurisé.

3.2.3 Les cours

Ce fut que qui m'a donné le plus de réflexions. En effet, mon système n'a rien à envier à Microsoft Word, ou d'autres sites. Le plus embêtant était que le retour à la ligne ne pouvait pas être enregistré dans la base de donnée : il fallait donc un symbole à mettre à chaque retour de ligne pour que le rendu le comptabilise comme retour à la ligne dans le code. Le point n'irait pas, puisqu'à la fin de chaque phrase le retour à la ligne s'effectuerait, et il est difficile de savoir quel symbole n'est utilisé par personne. J'ai donc utilisé l'underscore, c'est-à-dire ce symbole-ci : « ».
Autrement, la création de cours est plutôt basique : trois zones d'écritures pour le titre, la branche et le cours en lui-même.
Aussi, l'importation de cours (en pdf) m'a donné matière à réflexion : Où les stocker ? Ce serait donc dans un fichier dans l'application, où dès l'importation du premier PDF un dossier sous le nom de l'utilisateur est créé.

3. Responsive est une fonction permettant au contenu de s'adapter suivant le support (ordinateur portable, téléphone, etc.)

3.2.4 Liste de lecture :

Ceci a été pensé pour les bons nombres de livres que les étudiants lisent en cinq années de travail. Que ce soit pour les branches de français ou d'autres langues, les résumés sont utiles pour la maturité. Ils doivent mettre plusieurs qualificatifs : nom de l'œuvre, l'écrivain, le genre, le résumé et l'avis. Ce sont les en-têtes indiqués, mais libre à chaque utilisateur de préciser, peut-être par chapitre ou autre, suivant les fantaisies de chacun. Aussi, les utilisateurs pourront avoir le choix entre rendre leurs résumés d'œuvres publiques ou privées, selon leurs préférences. Pour faire cela, ce n'est qu'une colonne dans la table de la base de donnée et de trier suivant celui-ci.

3.2.5 Horaire

L'horaire est rempli par l'utilisateur en lui-même. Ainsi les neuf plages horaires et les cinq jours d'école sont proposés en tableau. Si aucun cours n'y est inscrit, la plage indique « temps libre ». Cette fonctionnalité m'a posé quelques soucis : Ma première idée avait été de trier les données reçues de la base de donnée par jour, puis d'utiliser la fonction « foreach » pour éviter la répétition. Mais la constitution de « table » n'était pas propice à cette façon de faire.
J'ai donc pensé à faire par plage horaire. Donc un « foreach » pour tout cours se déroulant entre huit heure quarante jusqu'à neuf heure vingt-cinq, et cætera. Malheureusement, avec cette manière, lorsque je faisais une condition, cela créait à chaque fois une case en plus. Donc, au lieu d'avoir cinq case pour les jours de lundi à vendredi, Il en affichait dix et l'affichage laissait à désirer.
Ne trouvant aucune explication à cela, j'ai dû tout faire à la main chaque horaire de celui-ci.
Voici le code pour une page horaire :

FIGURE 4 – Code de l'horaire

3.2.6 Vocabulaire

Une chose que beaucoup d'étudiants utilisent pour les langues, c'est des sites d'apprentissages de vocabulaire. Aussi cette fonctionnalité a été implémentée. Celle-ci contient pour chaque langue possible un système de liste par tableau. Les utilisateurs peuvent mettre en deux états les différents mots :

1. l'état positif : le mot se teinte de vert. L'utilisateur estime connaitre celui-ci.
2. L'état négatif : le mot tourne cette fois au rouge, indiquant qu'il n'est pas encore su.

Il y a également un état neutre, celui dans lequel se trouve la donnée qui n'a pas encore été traitée.
J'y ai mis les différentes branches étudiées, donc anglais, allemand, italien, espagnol, grec ou latin. J'avais tout d'abord pensé à spécifier suivant les choix spécifiques fait par les étudiants, mais je me suis rendue compte que la page serait bien trop vide. J'ai donc laissé chaque accès à tous.

3.2.7 Le tableau de notes

Celui-ci m'a donné du fil à retordre. En effet, les calculs des branches diffèrent suivant si celle-ci est une option spécifique ou non et je souhaitais que les moyennes soient calculées correctement. En effet, l'idéal aurait été d'afficher le final, comme dans le carnet scolaire, mais il fallait également que l'utilisateur ait accès aux détails, pour qu'il puisse le modifier si une faute s'y était glissée.
Le code était donc particulièrement long, et celui qui calculait les moyennes devait se répéter dans deux fichiers.
Pour tous les calculs, il fallait les mettre dans des balises php dans le fichier « vue ». Aussi, récupérer chaque branche, pour chaque semestre, était un casse-tête pour trouver le bon équilibre.
Finalement, commençant à copier-coller les différents codes créant les tableaux dans une balise php sur la vue affichant le détail des notes, notamment la création des moyennes, j'ai opté pour une fonction créant les moyennes pour celles où j'ai ajouté les « petites feuilles » puis les autres.

Ainsi, je n'ai plus qu'à l'utiliser pour chaque matière et stocker dans une variable ce dont j'ai besoin, c'est-à-dire la moyenne des « petites feuilles » pour certains et la moyenne générale. Ayant déjà écrit une bonne partie de la vue, j'ai simplement lié les variables de la partie "vue"

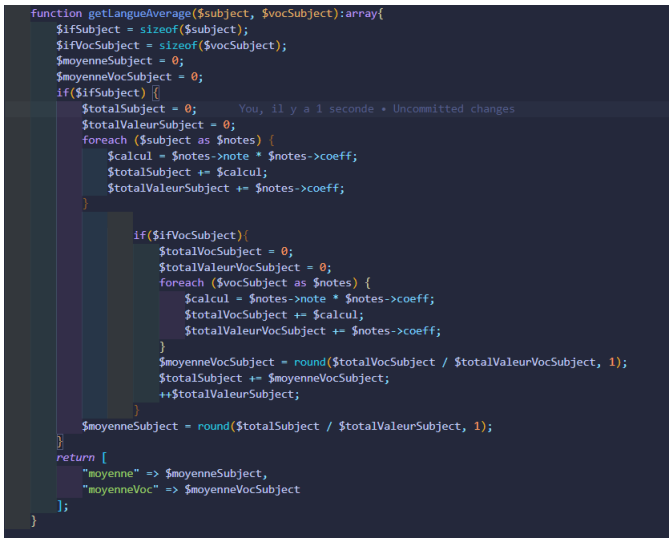

FIGURE 5 – fonction moyenne 1

FIGURE 6 – fonction moyenne 2

3.2.8 Autre

Les fonctions dans les controllers Beaucoup de fonctions sont similaires d'un controller à un autre. Notamment celui pour envoyer, récupérer ainsi pour supprimer les données. Le genre de données varient simplement de par leurs noms et leur type (nombre entier [int], un texte [text], et bien d'autres qui n'ont pas trouvé d'utilités dans l'application). En voici quelques exemples :

Pour beaucoup, le même nom de fonction a été choisi, pour ne pas embrouiller le code par des nominations farfelues, notamment celui pour supprimer une donnée.
Voici le code :

Cette fonction supprime une donnée particulière d'une table. Puisque la table

```php
public function deleting($id){
    allemand::findOrFail($id)->delete();
    return redirect()->back();
}
```

FIGURE 7 – fonction suppression

doit être spécifiée, elle se répète dans différents controllers. Pour ce faire, la fonction doit obtenir l'id de la donnée. La fonction va chercher dans la table sélectionnée (ici allemand) si une donnée a le nombre correspondant. Si c'est le cas, cela la supprimera. Autrement, rien ne se fera. Puisque l'utilisateur ne veut pas changer de page, actionner la fonction fera revenir sur la page d'origine.

Sur l'exemple ci-dessus, la fonction a été prise du controller destiné au vocabulaire de langue allemande, c'est pourquoi la base dans laquelle on fait agir la fonction est "allemand". Dans les autres Controllers, le nom de la base change.

Un autre qui est répété est l'envoi et la récupération des données :

```php
public function vocGetAllemand(){
    return view('allemand', [
        'allemand' => allemand::orderBy('id')->where("username", Auth::user()->name)->paginate(11)
    ]);
}
```

FIGURE 8 – fonction suppression

3.3 Fonctionnalité en React

Pour quelques fonctionnalités, j'ai dû passer par la librairie React. En effet, celle-ci est pratique pour créer des interfaces utilisateurs interactive. Ainsi, le site pourra changer d'affichage sans pour autant devoir recharger la page.
Pour son fonctionnement, React utilise des «composants » dans son code : les classes et les fonctions. Pour toutes mes fonctionnalités, j'ai pu tester les deux types, bien distinctes notamment dans la syntaxe l'une de l'autre. Les fonctions peuvent être utilisées dans les classes, mais non inversement.
Ainsi, il valait mieux pour les petites fonctionnalités d'utiliser une simple fonction au lieu de s'encombrer d'une classe, comme par exemple le Pomodoro, qui va être détaillé plus bas. Pour d'autres, comme l'agenda, c'était plus sage d'utiliser une classe, qui va donc être plus structurée et lisible que si on choisissait la fonction.

3.3.1 Pomodoro

La méthode pomodoro sert à gérer le temps de travail. En effet, il est démontré que le temps est souvent gaspillé dans de la procrastination, ou rien que

de la perte de temps dû à un manque de concentration et la peur de l'échéance [4].
Aussi, cette technique fonctionne ainsi :
Cette méthode fonctionna ainsi : un temps précis de travail, puis un temps de pause défini, souvent pour cinq minutes, puis le retour de la boucle. Dans mon application, le temps de travail est proposé par deux choix à l'utilisateur : une durée de vingt-cinq minutes ou de quarante-cinq.
Historiquement, le pomodoro a été créé par Francesco Cirillo en 1980 [5]. Ce nom est inspiré du minuteur de cuisine en forme de tomate.
Laravel ne permettant pas d'égrener les secondes sur une page sans la recharger, aussi ai-je décidé d'adopter React. Cette bibliothèque Javascript permet de rendre une page réactive sans pour autant devoir la recharger, aussi était-ce parfait pour ce dont je souhaitais faire.
Le code était fondamentalement simple : il me fallait simplement un stockage d'un certain nombre de secondes puis de les égrener une par une, ce qui en fait la seule fonction du code.
Ainsi, voici le code :

```
import React , {useState} from 'react'
import ReactDOM from 'react-dom'

export default function Pomodoro() {
    const [second, setSecond] = useState(1500);
    const [start, setStart] = useState(false);

    if (start === true) {
        setTimeout(() => {
            setSecond(second - 1)
        }, 1000)
    }

    return <div className='pomodoro'>
            <button style={{backgroundColor : "rgb(71, 71, 84)", borderRadius : "15px", width : "70px", color : "rgb 177, 196, 219
            " }} onClick={(e) => setSecond(2700)}>45 min</button>
            <button style={{backgroundColor : "rgb(71, 71, 84)", borderRadius : "15px", width : "70px", color : "rgb 177, 196, 219
            " }} onClick={(e) => setSecond(1500)}>25 min</button>
            {!start ? <button style={{backgroundColor : "rgb(124, 54, 54)", borderRadius : "15px", width : "140px"}} onClick={(e) =>
            setStart(!start)}>Start</button> : <button style={{backgroundColor : "rgb(124, 54, 54)", borderRadius : "15px", width :
            "140px"}} onClick={(e) => setStart(!start)}>Stop</button>}
            <h4>{parseInt(second / 60)} : {second % 60 < 10 ? <span>0{second % 60}</span> : <span>{second % 60}</span>} </h4>
            </div>
}
ReactDOM.render(<Pomodoro />, document.getElementById("pomodoro"))
```

FIGURE 9 – pomodoro

Dans le rendu, j'y ai ajouté des conditions pour passer d'un état de pause à l'état de marche. Étant minime, créer une fonction ne m'a pas semblée nécessaire. Puis le rendu :

4. https ://pomodoro-tracker.com
5. https ://www.selimniederhoffer.com/blog/la-methode-pomodoro-productif

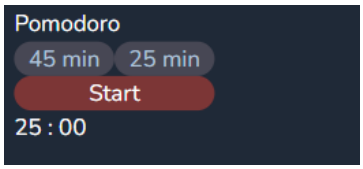

FIGURE 10 – pomodoro

3.3.2 Calendrier

Pour faire un agenda, j'ai choisi d'utiliser fullcalendar.io. Il s'agit d'une extension Javascript créant de magnifiques calendriers. Il est utilisé par les frameworks créant des interfaces utilisateurs, dont React, Vue et Angular. Pour l'application, seule React a été utilisé. Ainsi, pour comprendre, j'ai pris exemple sur la démo-app [6] qui se trouvait sur la librairie fullcalendar.io. En effet, il y avait apparemment des fonctionnalités premium, aussi craigné-je devoir restreindre certaines de mes idées pour le calendrier. Voici son rendu :

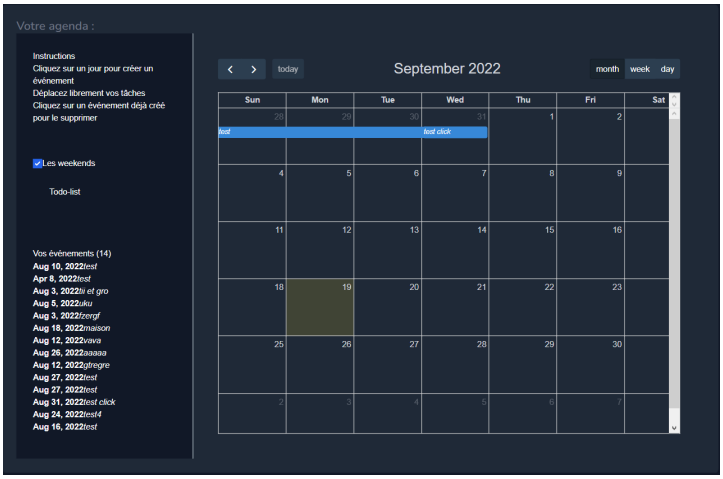

FIGURE 11 – l'agenda

Son fonctionnement est ainsi : cette fonction (figure12) ajoute dans le state [7] si un titre a été écrit. Puis il créé un événement avec la fonction tirée de la bibliothèque de FullCalendar en y ajoutant automatiquement un id, puis comptabilise le ou les jour(s) où l'utilisateur a sélectionné dans l'agenda cf figure11. Pour l'ajouter dans la base de données, il me fallait la fonction que j'ai appelée importEvent (figure 13) qui lance une route web appelé "storageevent", utilisant une fonction dans le controller qui enfin l'insère dans la table de la base de donnée correspondante. Pour rendre les tâches un peu plus lisible, j'ai décidé de créer une branche à part : une todo-list, intégrée dans la barre de navigation

6. https ://fullcalendar.io/demos
7. il s'agit d'un tableau de donnée temporaire qui sera transférée dans la base de donnée

```
handleDateSelect = (selectInfo) => {
  let title = prompt('Veuillez mettre le nom de votre tâche')
  let calendarApi = selectInfo.view.calendar

  calendarApi.unselect() // clear date selection

  if (title) {
    calendarApi.addEvent({
      id: createEventId(),
      title,
      start: selectInfo.startStr,
      end: selectInfo.endStr,
      allDay: selectInfo.allDay,
    })
```

FIGURE 12 – ajouter un événement

```
importEvent = (event) => {
  axios.post('/storageevent', event)
    .catch(error => {
      console.log("ERROR:: ", error.response.data);
    });
}
```

FIGURE 13 – ajouter un événement dans la base de donnée

latérale. Il me suffisait de récupérer les événements ajoutés dans le calendrier et de les mettre en liste trié par ordre des dates de fins des tâches. La fonctionnalité « done » est ajoutée, pour pouvoir l'enlever de la liste à faire.

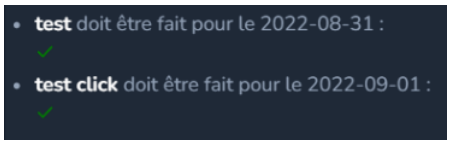

FIGURE 14 – todo

3.3.3 Math

Ayant appris l'utilisation de Latex en cours, le langage de balise et la fonctionnalité « math » avec les balises correspondantes, je me suis tout de suite tournée vers celui-ci pour faire une section où les étudiants pourront stocker leurs formules à apprendre. Aussi l'implantation directe de Latex dans du Laravel n'était pas pratique du fait que déjà le rendu ne s'afficherait qu'avec l'envoi du formulaire, et même si une petite aide est à disposition, ce n'était pas agréable à utiliser. React affiche donc le résultat écrit en temps réel et la rajoute à la liste sans recharger la page.
J'ai souhaité intégrer le Formulaire et Table à disposition, mais la CRN n'a malheureusement pas répondu à ces attente. Il y a cependant une liste des calculatrices autorisées à la maturité disponible.

3.3.4 Problème lié à React

Mon plus gros problème était de lier la base de donnée aux fichiers de Js. En effet, devant passer par des routes APIs, je ne pouvais pas utiliser la manière classique de tri par utilisateur. Aussi ai-je récupéré d'une manière peu sécurisée le nom de l'utilisateur pour pouvoir trier les données avec celui-ci : Dans la barre

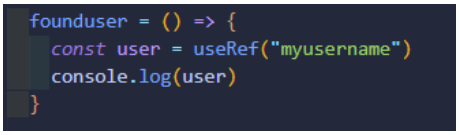

FIGURE 15 – récupération en React du nom de l'utilisateur

de navigation du haut, le nom de l'utilisateur y est affiché. Ajoutant un Id, le récupérer est devenu possible avec « document.getElementById() ». Cependant, dès que je l'utilisais plus d'une fois, une erreur s'affichait.

4 Autres

4.1 Le Graphisme

Pour le style, j'ai principalement utilisé TailwindCSS, pour faire des animations et gérer les couleurs du site, notamment des boutons et des fonds.
C'est pourquoi les balises « div » ont énormément de "class" définissant le style.
J'ai tout d'abord créé un logo basée sur la croix de Saint-Maurice, avec l'application Photoshop, puis ai-je tenté une autre image : une écriture simple, du même type que celui de l'école, teinte en rouge. L'un et l'autre s'accordait, aussi les ai-je mis à côté et ai-je décidé de l'officialiser.

FIGURE 16 – lycapp

Les icônes utilisés proviennent d'un site qui fournit des icônes pour les développeurs informatiques : flavicon.com. Pour ce faire, il faut mentionner le site et le designer.

5 Conclusion et bilan

Pour conclure, pour créer une application, il y a énormément de facteurs à prendre en compte. De la première ligne de code jusqu'à la mise en ligne, il y a eu énormément de contraintes et de difficultés à résoudre. Il y a énormément de raccourcis à savoir rien que pour écrire le code, ou même l'optimiser.
Créer un site internet m'a permis d'obtenir des bases solides dans le Framework Laravel ainsi que d'apprendre quelques bases en javascript. Je n'y connaissais rien, et apprendre tout un langage en peu de temps était un véritable défi. Néanmoins, on se rend compte de l'immensité de connaissances que peu apporter internet si on cherche les bonnes ressources. Youtube, la bibliothèque lié au language utilisé, les cours en ligne, tout est à disposition, le plus souvent gratuitement. Aussi, l'utilisation d'une base de données était toute nouvelle pour moi.
La réflexion pour autrui et la résolution de problèmes m'a appris de penser autrement, d'un point de vue du besoin. Cet exercice intuitif était un bon début sur ce point, étant également étudiante.
Aussi trouver des solutions par moi-même en effectuant des recherches, sur Internet, dans des livres, m'a poussée dans l'indépendance et également de l'autonomie. Les concessions dues aux problèmes non résolus m'ont demandé énormément de mental.
J'ai surtout été restreinte à cause de l'anglais. J'ai un bon niveau oral à ce niveau-là, mais l'écrit était éprouvant à comprendre, car le vocabulaire me manquait cruellement.

6 Sources et bibliographie

https://laravel.com/docs/9.x
https://www.wampserver.com
https://www.php.net
https://www.javascript.com/about
https://www.w3schools.com/js/
https://www.techno-science.net/definition/5331.html
https://www.lemagit.fr/definition/Programmation-orientee-objet
https://www.selimniederhoffer.com/blog/la-methode-pomodoro-productif/
https://pomodoro-tracker.com
https://www.epfl.ch/campus/associations/list/cqfd/studyingmath/latexhelp/
https://www.techtarget.com/whatis/definition/PHP-Hypertext-Preprocessor
https://www.cybersecurity-guide.com/bcrypt-un-algorithme-de-hachage-a-utiliser-en-php/