Projet de Java avancé Master 1

Friday

GHAMRI Samy
OSAJ Xhavit

Table des matières

Introduction

Le projet Friday consiste en la création d'un assistant gérant un agenda d'un utilisateur. Ce projet est dit fullstack car nous avons une partie frontend, qui permet f'afficher les informations de l'agenda, et d'une partie backend permettant de manipuler les données d'un agenda à l'aide de services REST.

L'application permet à l'utilisateur de créer, modifier ou supprimer des évènements, de récupérer des données au format iCalendar, d'extraire des données d'un calendrier Google Calendar et de donner des informations sur les prochain rendez-vous, les conditions de trafic,…

Technologies utilisées

Nous avons utilisé l'IDE **IntelliJ** pour écrire notre application.

L'outil de build utilisé pour l'application est ***Maven***.

Nous avons utilisé ***Gitlab*** pour héberger notre code.

Pour la partie Backend

Pour cette application, notre binôme a utilisé la version 2.5.5 de ***Spring Boot*** pour implanter les services REST nécessaires à l'application.

Nous avons utilisé ***Spring Data – JPA*** pour utiliser les lignes d'une table de la base de données comme un objet Java.

Pour la partie Frontend

Pour la partie frontent, nous avons utiliser la version 3 du framework graphique ***Vuejs*** pour afficher visuellement l'agenda.

Pour le style de l'application, nous avons utilisé la version 5.1.1 de ***Bootstrap***.

Pour la base de données

Nous avons utilisé la version 2.6.0 de la base de données ***HyperSQL*** pour stocker les données des utilisateurs.

Partie utilisateur

Dans cette partie, nous allons expliquer à l'utilisateur comment utiliser l'application et faire la liste de toutes les fonctionnalités de celle-ci.

Compilation de l'application

Cette partie vous permet de connaître la marche à suivre si vous voulez compiler le projet par vous même dans le cas où vous avez supprimé, sans le vouloir, le jar fournit.

Pour compiler, vous allez avoir besoin d'un terminal et de ***Maven***.

Compilation sous Unix

Dans votre terminal, placez vous à la **racine** du dossier du projet.

Vous avez deux possibilités :

- Utiliser la commande **`mvn clean package`**
- Utiliser le fichier *mvnw* et la commande **`./mvnw clean package`**

Compilation sous Windows

Dans votre terminal, placez vous à la racine du dossier du projet.

Vous avez deux possibilités :

- Utiliser la commande **`mvn clean package`**
- Utiliser le fichier *mvnw.cmd* et la commande **`./mvnw.cmd clean package`**

Après la compilation, vous devriez avoir dans le dossier ***target***, un fichier sous le nom ***FridayApp.jar***. Si ce n'est pas le cas, vous avez surement mal fait une des étapes, on vous invite donc à refaire les étapes.

Exécution de l'application

Dans cette partie, vous allez apprendre à lancer le jar qui est dans le dossier *target*.

Pour cela, dans votre terminal, placez vous à la racine du dossier du projet.

Entrez ensuite la commande suivant : **`java -jar target/FridayApp.jar`**

Vous devriez alors voir le serveur se lancer.

Accès à l’application sur un navigateur web

Pour ouvrir l’application, ouvrez **Google Chrome** (nous avons des bugs avec les autres navigateurs, nous pensons que c’est lié à notre utilisation des routers) et entrez l’URL suivante (vous pouvez aussi cliquer dessus directement) :

http://localhost:8080

Cela devrait vous envoyer vers la page de login

Fonctionnalités de l’application

Se connecter

Pour vous connecter, il va de soi que vous devez être déjà inscrit sur l’application. Si ce n’est pas le cas, veuillez lire la partie **S’inscrire** pour vous enregistrer.

Vous devez entrer votre nom d’utilisateur dans le champs *Username* de la page et votre mot de passe dans le champs *Password.* Votre **nom d’utilisateur ne doit pas contenir d’espace** car cela créerait un beug. Si les informations entrées ne correspondent pas à un utilisateur existant, vous allez voir un *pop-up* disant que l’un des champs n’est pas le bon. On ne précise pas lequel pour des raisons de sécurité. Vous pouvez vérifier si vous n’avez pas fait de faute de frappe en entrant le bon mot de passe en cochant la case *ShowPassword.*

Après la connexion, vous allez être redirigé vers votre agenda.

Bug

Lors de la première utilisation de l’application, vous allez devoir vous connecter deux fois car nous avons un bug avec la première connexion, elle renvoie vers la même page.

S’inscrire

Pour vous inscrire, vous allez devoir entré un nom d’utilisateur et un mot de passe que vous allez devoir confirmer en l’entrant une deuxième fois.

Si vous entrez un nom d’utilisateur déjà existant, un *pop-up* va vous avertir de cela.

Si votre mot de passe ne correspond pas à la confirmation, un *pop-up* va vous avertir de cela.

Vous pouvez vérifier si vous n’avez pas fait de faute de frappe en entrant le bon mot de passe en cochant la case *ShowPassword.*

Après vous être enregistré, vous allez être redirigé vers la page de connexion.

Bug

Lors de la première utilisation de l’application, vous allez devoir vous enregistrer deux fois car nous avons un bug avec le premier enregistrement, elle renvoie vers la même page.

L’agenda

Sur cette page, vous allez voir des informations sur votre prochain rendez-vous, la liste de tous vos évènements de la journée et un calendrier du mois regroupant vos évènements à venir. Dans le calendrier, une barre horizontale sur un jour veut dire que vous avez un évènements ce jour-là.

Ajouter un évènement

En dessous du calendrier du mois, vous avez un bouton *Add Event* vous permettant d’ajouter un évènement. Lorsque vous cliquer dessus, un formulaire apparaît :

Entrez votre description dans le champs *Description*, votre location dans le champs *Location*, sélectionnez le jour et l’heure de l’évènement et en cliquant sur le bouton *All day*, vous spécifiez que l’évènement se déroule sur toute la journée. Appuyer ensuite sur *Add Event* pour sauvegarder votre évènement.

Modifier un évènement

Cela permet de modifier un évènement.

Cette fonctionnalité n’a pas encore été implanter et sera disponible dans les prochaines mise à jour de l’application.

Supprimer un évènement

Cela permet de supprimer un évènement.

Cette fonctionnalité n’a pas encore été implanter et sera disponible dans les prochaines mise à jour de l’application.

Créer un nouveau calendrier

Le bouton N*ew Calendar* est disponible en cliquant sur le menu *Files* situé en haut à droite de l’écran. Il permet de créer un agenda vierge. Attention, en cliquant sur ce bouton, vous effacerai **complètement** votre ancien agenda.

Importer un calendrier depuis un fichier

Le bouton *Load a calendar from exising file* est disponible en cliquant sur le menu *Files* situé en haut à droite de l'écran. Vous devez entre le chemin absolu de votre fichier au format ics dans la zone prévu à cet effet et vos évènements contenus dans ce fichier seront chargés.

Exporter votre calendrier

Le bouton *Export my calendar* est disponible en cliquant sur le menu *Files* situé en haut à droite de l'écran. Ce bouton permet de créer un fichier au format ics sous le nom *username-Calendar.ics* où *username* correspond à votre nom d'utilisateur. Ce fichier retranscrit votre agenda.

Deconnexion

Pour vous deconnecter, appuyez sur le bouton *Deconnexion* en haut à droite de l'écran. Vous serez alors redirigé vers la page de connexion.

Arrêter le serveur

Quand vous avez fini d'utiliser l'application, retournez dans le terminal dans lequel votre serveur est en marche et appuyé sur Ctrl + C pour arrêter le serveur. On sait que c'est pas bien de faire comme ça mais ça marche bien.

Partie développeur

Cette partie est plus technique et est destinée aux développeurs.

Dans cette partie, nous allons expliquer comment nous avons créer l'application et les problèmes rencontrés.

Création de l'application

Nous avons commencé par nous familiariser avec les différentes technologies. Nous avons lu les différentes documentations, lu des articles disponibles sur internet décrivant ses technologies et regarder des vidéos sur Youtube qui expliquait comment elles fonctionnent.

Partie Backend

Nous avons ensuite crée notre application Spring Boot à l'aide de *Spring Initializr* présent sur *IntelliJ*.

Après avoir lancé une première fois le serveur Spring Boot, nous avons essayé de connecter notre base de données à notre projet. Nous avons donc écrit le code nécessaire pour cela dans le fichier *application.properties* présent dans le dossier *ressources*. Nous avons aussi créer un fichier *ical4j.properties* pour régler un warning lors du lancement du serveur.

Après avoir fait cela, nous avons crée les tables de la base de données. On a fait cela dans le dossier en*tity.* On a créé une table User et une table Event.

Table User

Cette table contient les champs id_user, correspondant à l'ID de l'utilisateur, un champs username correspondant à son nom d'utilisateur et un champs password correspondant à son mot de passe.

La classe User contient les getters de chaque champs.

Table Event

Cette table contient les champs idCal, correspondant à l'ID du calendrier, un champs calendar correspondant au calendrier, qui est vide initialement et un champs user qui est une clé étrangère correspondant à l'utilisateur propriétaire de ce calendrier.

La classe Event contient les getters de chaque champs et une méthode permettant d'ajouter un évènement au calendrier de l'utilisateur.

Repository

UserRepository et EventRepository sont des interfaces qui extend l'interface JpaRepository et qui permettent d'avoir des méthodes qui simulent le comportements d'une requête SQL et qui renvoient les données sous forme d'objet Java.

DTO

Ce module permet de définir les classes permettant de définir comment une donnée va être transférée selon l'action qu'elle fait.

Les records EventResponseDTO,et UserResponseDTO, permettent de créer un objet que l'on va envoyer à l'application web en enlevant les éléments sensibles comme le mot de passe d'un utilisateur ou les éléments inutiles comme l'utilisateur à qui appartient le calendrier car on le sait déjà.

Les records EventSaveDTO,et UserSaveDTO, permettent de créer l'objet que l'on va envoyer à nos méthodes pour créer un User, donc son username et son password, et un Event, sa description, sa location, sa date, le User correspondant et le fait que l'évènement dure toute la journée ou pas.

Ces records permettent de renforcer la sécurité et la robustesse des informations échangées entre le front et le back.

Converter

Les converter permettent de convertir les données récupérées à l'aide des DTO. Par exemple, on convertit un objet EventSaveDTO en Event donc à partir des informations entrées par l'utilisateur on crée l'Event associé.

Controller

Les controller permettent de fournir des méthodes que l'on appèlent en utilisant des liens. Notre application web enverra des données à ces lien et récupèrera des résultats grâce à ces méthodes.

Par exemple, pour sauvegarder un nouvel utilisateur, le front va envoyer une requête de type POST avec des données JSON, l'username et le password, à l'adresse localhost:8080/user/save, la méthode save de UserController récupère les données, les converties et crée un utilisateur à l'aide de ses données. Nous avons aussi la méthode check qui vérifie si la combinaison username, password correspond à un utilisateur et d'autre méthodes.

Dans la classe EventController, nous avons aussi la méthode save à l'adresse localhost:8080/event/save et d'autre méthodes.

Partie Frontend

Router

Nous avons utilisé des router pour naviguer entre chaque page de notre application. Les router nous ont causé pas mal de problèmes. Au début, il ne fonctionnait pas bien. Maintenant ils fonctionnent presque bien, en effet lors de la première utilisation de l'application, nous avons un bug qui renvoie l'utilisateur sur la même page après une connexion ou un enregistrement.

Nous avons malheureusement manqué de temps pour régler ce problème qui, selon nous, n'aurait pas été difficile à régler.

Les différents fichiers vue

App.vue

Ce fichier représente la page principale sur laquelle les différentes vues vont être affichées. Nous avons mis une image avec le nom de l'application en haut de la page.

Lorsqu'on lance l'application, le rendu de la page login est affiché et l'utilisateur peut se connecter

FridayLogin.vue

Ce fichier contient le code la page de connexion. Il y a deux champs pour la connexion et un bouton permettant de passer à la page d'inscription si l'utilisateur veut s'inscrire.

Lorsque l'utilisateur appuie sur le bouton Sign in, une vérification des informations entrées est faite à l'aide d'un fetch sur la méthode de UserController check. Si les données sont bonnes et que l'utilisateur existe alors il est redirigé vers la page calendar sur laquelle il aura accès à son calendrier. Sinon, un pop-up lui dire qu'il s'est trompé et il sera invité à entrer des informations valides.

Comme dit précédemment, lors du lancement, l'utilisateur devra se connecter deux fois car la première fois le router beug et ne connecte pas l'utilisateur.

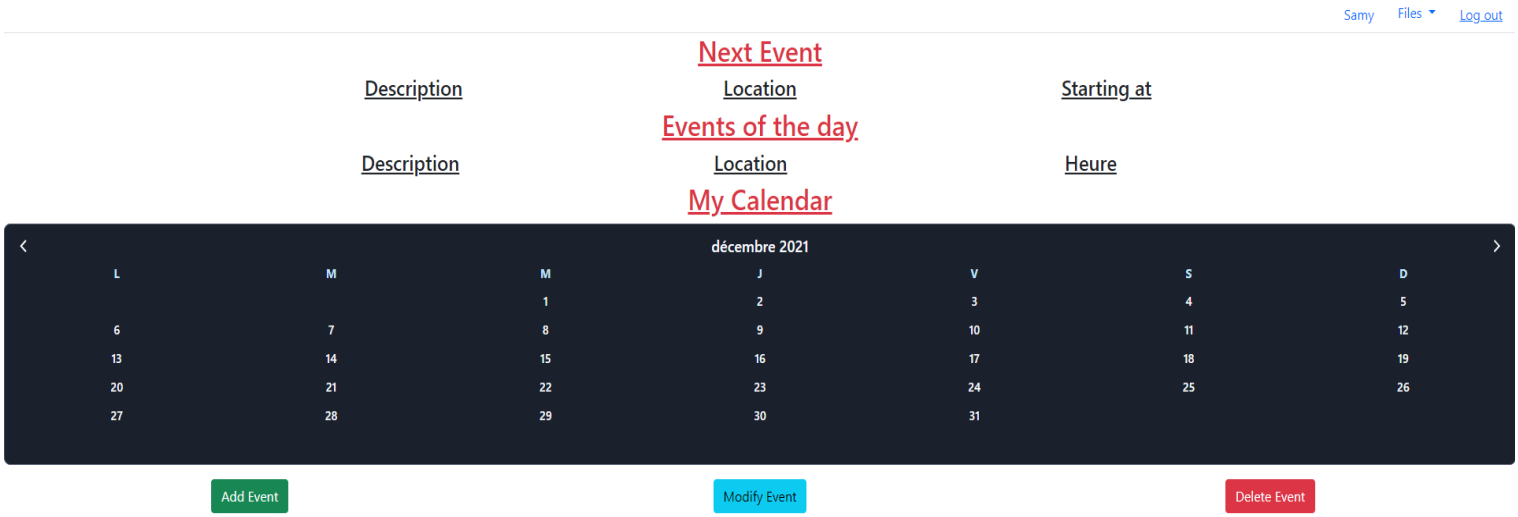
FridayRegistration.vue

Ce fichier contient le code de la page d'inscription. Il y a 3 champs pour l'inscription et un bouton permettant de passer à la page login si l'utilisateur a déjà un compte.

L'utilisateur doit entrer un nom et un mot de passe qu'il doit confirmer en le retapant. Lorsque l'utilisateur appuie sur le bouton Sign up, à l'aide de deux fetch, un vérifiant si le nom est déjà pris et l'autre sauvegardant l'utilisateur si toutes les informations sont bonnes. Si l'utilisateur entre un nom déjà utilisé, un pop-up le prévient et l'inscription n'est pas faite, de même si l'utilisateur n'entre pas le même mot de passe lors de la vérification. Si toutes les informations sont bonnes, il sera redirigé vers la page de login et il sera invité à se connecter avec son compte fraichement créé.

CalendarManager.vue

Ce fichier contient le code servant à afficher la page principale de notre application. En effet, c'est ici que toute les données relatives au calendar de notre utilisateur se trouve.



Au début, chaque nouvel utilisateur aura donc cette affichage puisque son calendrier sera vide.

Sur cette page se trouve un menu déroulant lorsque l'on clique sur le bouton «Files» qui correspond aux fonctionnalités associées. Malheuresement, seulement la fonction «New Calendar» fonctionne. Néanmoins, vous pouvez voir nos essais avec les méthodes setCalendarFromFile et exportCalendar.

Ensuite, le «Next Event» n'affichera pas le bon prochain événement. Nous n'avons en effet pas réussi à trouver comment définir la prochaine date la plus proche entre deux dates. Encore une fois, nos essais sont visibles au niveau de la méthode starter (cela correspond à la partie en commentaire).

Ensuite, les event du jour sont recupérés lors de l'appel à la méthode starter dans une liste nommées «todaysTODO». Cette fonctionnalité fonctionne correctement. Elle affiche ligne par ligne les events du jour (par ordre d'entré et non pas chronologique).

Enfin, l'utilisateur peut voir son calendrier comportant les events mois par mois. Les dates comportant un trait honrizontal bleu indiquent qu'un event s'y trouve.

Des trois boutons implémentés, seul le bouton «Add Event» fonctionne. Ce bouton et son formulaire sont liés à la méthode «addEvent» qui permet de mettre à jour notre page ainsi que la base de données.

Conclusion

Ce projet aura été le plus dur de notre (très jeune) carrière de développeurs. N'étant pas habitués à des projets de cette ampleur, nous n'avons malheureusement pas réussi à boucler ce projet proprement. Cela nous laisse un petit goût d'amertume car nous ne sommes pas habitués à rendre un devoir ou projet incomplet, cependant nous devons admettre que la marche était peut-être un peu trop haute pour nous.

La plupart de ces technologies nous étaient inconnues; le temps que nous avons pris pour les prendre en main nous aura fait défaut ici. Nous restons convaincus qu'une ou deux semaines nous auraient été suffisante afin de terminer ce projet. Mais nous n'avons pas réussi à jongler entre la masse de travail qui était déjà présente et le projet.

Néanmoins, nous avons tout de même pris beaucoup de plaisir à travailler sur ce dernier. Notre novicité aura été aussi un avantage puisque nous avons pu découvrir énormément de nouvelles choses, que ce soit les technologies, la manière de travailler ou encore le thème du projet en lui-même étaient d'autant plus de raison d'apprécier de travailler sur ce dernier.

De plus, nous avons enfin pu découvrir à quoi un vrai projet style «entreprise» ressemblait.

La répartition des tâches s'est faite de manière équitable. Nous avons essayé de tout les deux toucher à tout de manière à en apprendre le maximum. Nous travaillions aussi souvent ensemble, que ce soit à la fac ou sur discord, afin de bien se tenir informé des avancés du projet, et de pouvoir s'aider rapidement en cas de problème.