

Université Gustave Eiffel

Master Informatique

Rapport de projet

Java Avancé: Friday



1re Année

Décembre 2021

Auteurs

- Jimmy TEILLARD
- Lorris CREANTOR

Contents

| | | |
|----------|------------------------------------|-----------|
| 1 | Préambule | 3 |
| 2 | Introduction | 4 |
| 3 | Manuel utilisateur | 5 |
| 3.1 | Compilation et lancement | 5 |
| 3.2 | Connexion | 5 |
| 3.3 | Vue générale | 6 |
| 3.4 | Détails des événements | 8 |
| 3.5 | Import d'événements | 9 |
| 4 | Rapport technique | 11 |
| 4.1 | Backend (API) | 11 |
| 4.1.1 | Tables | 11 |
| 4.1.2 | Problème | 11 |

1 Préambule

Ce rapport a pour but principal de présenter le projet d'application réalisé dans le cadre de notre première année de Master informatique à l'IGM. Il présentera l'application résultant dudit projet. Seront ainsi abordés, tant les aspects techniques du projet que le manuel d'utilisation et les fonctionnalités de l'application.

2 Introduction

Le projet **Friday** a pour but d'écrire un assistant de gestion d'agenda. L'application se divise en deux:

- Une application **friday-back** écrite en Java proposant divers services REST pour manipuler les données d'un agenda qui seront stockées en base de données
- Une application **friday-front** écrite en JavaScript affichant visuellement les informations de l'agenda dans un navigateur

Pour la réalisation de l'application, nous nous sommes vu imposer des contraintes quant aux technologies à utiliser.

Notre binôme a ainsi du utiliser les technologies suivantes:

- Micronaut pour la partie server REST
- Micronaut-JPA pour l'ORM
- H2 Database pour la base de données
- Svelte pour la partie graphique de l'application Web
- Tailwind CSS pour gérer le style de l'application Web
- Maven pour la gestion de dépendances
- Google Calendar API pour permettre à l'utilisateur d'importer un calendrier depuis Google Agenda
- BiWeekly pour gérer des calendriers au format iCal

3 Manuel utilisateur

Cette section a pour but de présenter le manuel utilisateur de l'application Friday.

3.1 Compilation et lancement

Le projet nécessite que l'utilisateur ait Java-17 installé sur sa machine. Tout le reste des dépendances sont téléchargées automatiquement au besoin.

Pour compiler un fichier exécutable, il suffit de lancer la commande `./mvnw clean package`. Un fichier `friday-[version].jar` est généré dans un dossier `target/` : c'est le fichier exécutable de Friday.

Pour lancer l'exécutable, il faut lancer la commande `java -jar friday-[version].jar`.

Il est à noter que l'on peut créer un dossier `resources/` à la racine de l'application (c'est à dire **au même niveau que le lancement de l'application, au lancement de l'application, vous recevrez un message "No salt found" si vous avez échoué à cela**). Il peut (devrait) contenir les fichiers suivants :

- `salt.txt` : qui contient une clé de salage de 16 octets qui sert à hasher les mots de passe des utilisateurs (ATTENTION : si vous perdez ce fichier, plus aucun des anciens utilisateurs ne seront accessibles car vous ne pourrez plus hasher les mots de passe avec la même clé)
- `credentials.json` : qui contient les clés d'API de la Google Calendar API

3.2 Connexion

Lors de l'arrivée sur le site l'utilisateur se retrouvera face à un écran de connexion. Il pourra alors soit se connecter soit s'enregistrer sur le site.

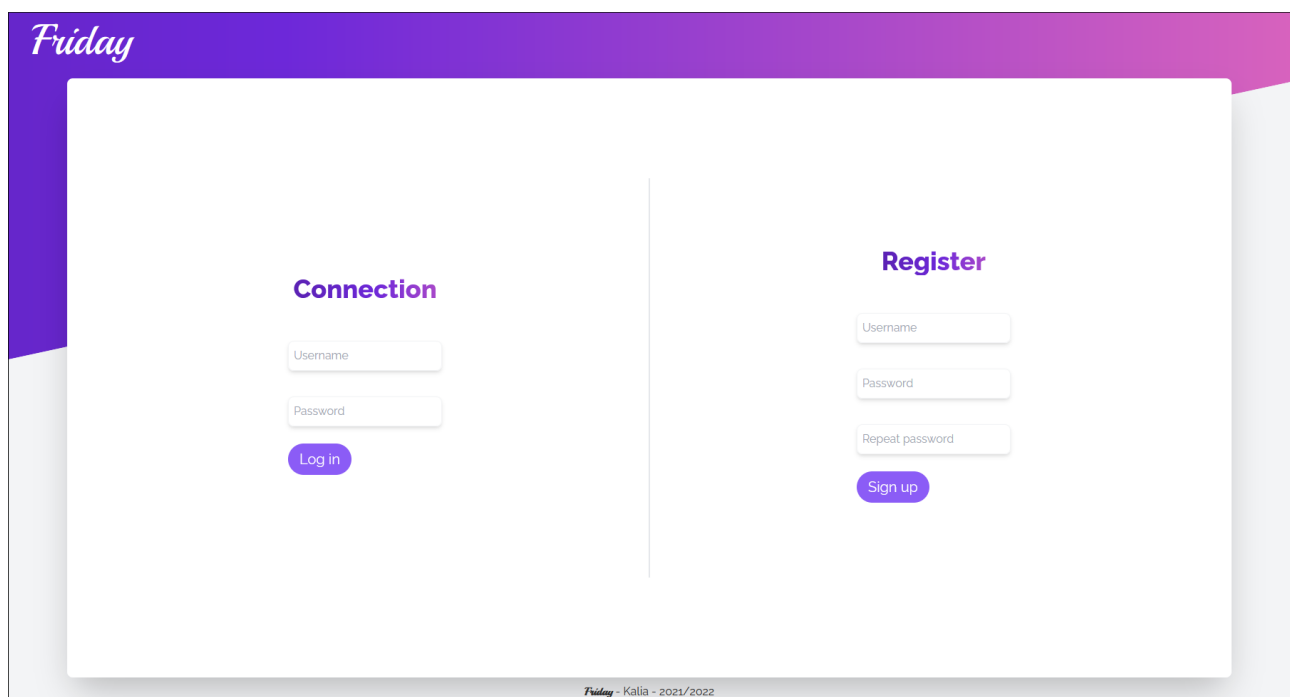


Figure 1: Écran de connexion

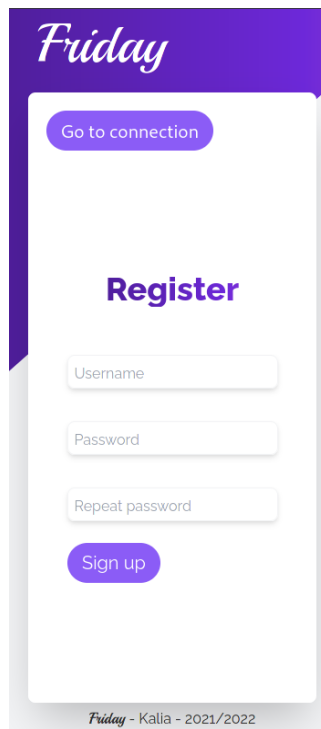


Figure 2: Écran de connexion mobile

Sur la version mobile, un bouton "Go to connection/registration" apparaît pour alterner entre les deux formulaires.

3.3 Vue générale

Une fois connecté, la page de vue générale est présentée à l'utilisateur (la vue mobile est sensiblement similaire):

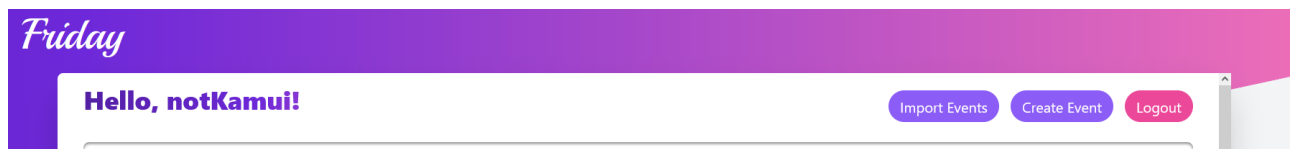


Figure 3: Vue générale

En haut de la page se trouvent trois boutons:

- Import Events : ouvre un formulaire d'import d'événements
- Create Event : ouvre un formulaire de création d'événement
- Logout : permet de se déconnecter

La vue générale se divise en 3 parties :

- Next : Le prochain événement et ses informations

3.4 Détails des événements

Le formulaire de création d'événement se présente comme ceci :

Create an event

Title*

Place

Description

Start Date*

12 / 31 / 2021, 11:39 PM

End Date*

12 / 31 / 2021, 11:39 PM

☐ All Day

Localisation

Latitude

Longitude

Recurrence

Frequency*

None

Until

mm / dd / yyyy, --:-- --

Cancel Create

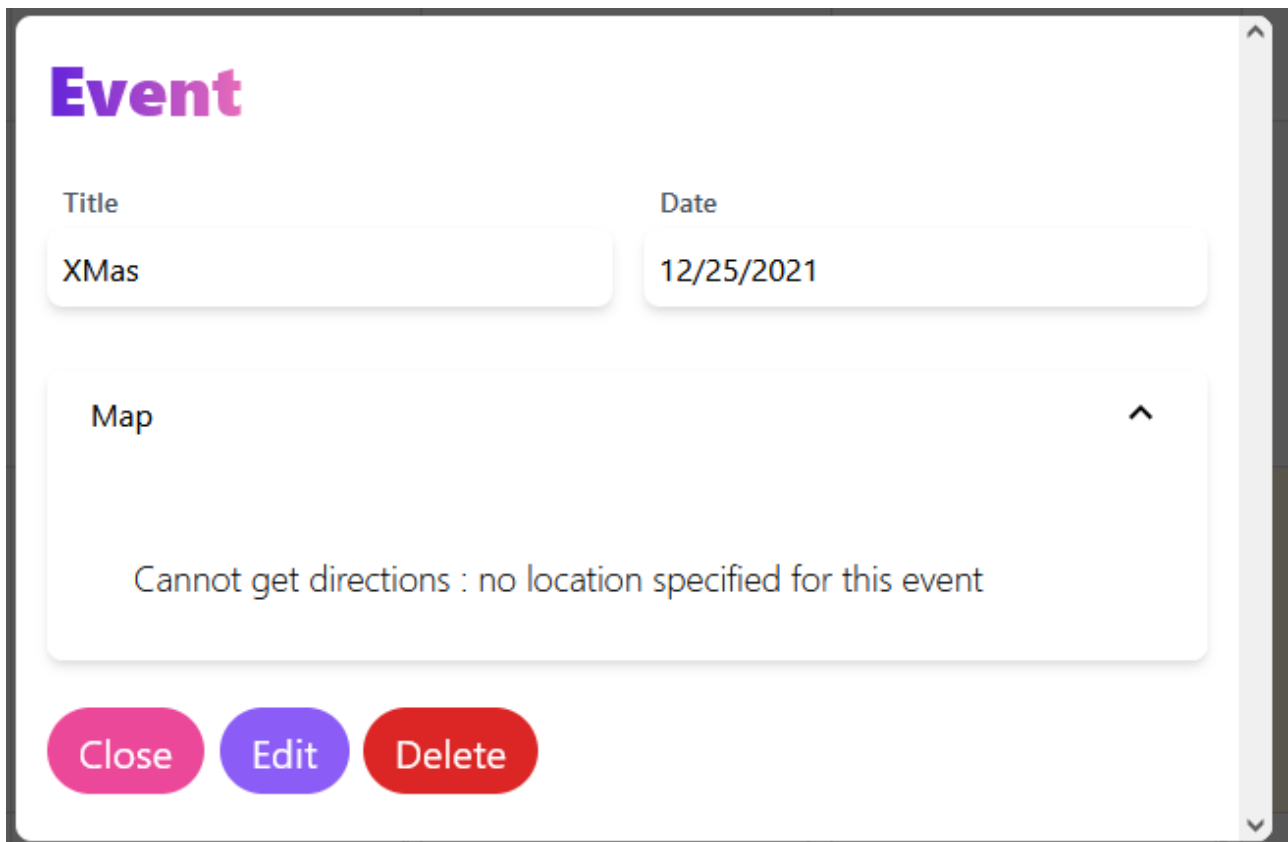
Friday - Kalia - 2021/2022

Figure 7: Création d'événement

Chaque champs est remplissable ; "Title", "Start date", et "End date" (ou "All day") sont obligatoires. Il y a aussi deux menus déroulants pour remplir la géolocalisation de l'événement,

ainsi qu'une règle de récurrence simple.

L'utilisateur peut cliquer sur les événements dans les calendriers pour afficher une vue simplifiée dudit événement :



Event

Title: XMas

Date: 12/25/2021

Map

Cannot get directions : no location specified for this event

Close Edit Delete

Figure 8: Vue d'un événement

Il peut alors aussi supprimer cet événement ("Delete") ou l'éditer ("Update"). Le menu d'édition est sensiblement similaire au menu de création.

3.5 Import d'événements

Le menu d'import d'événement se présente comme ceci :

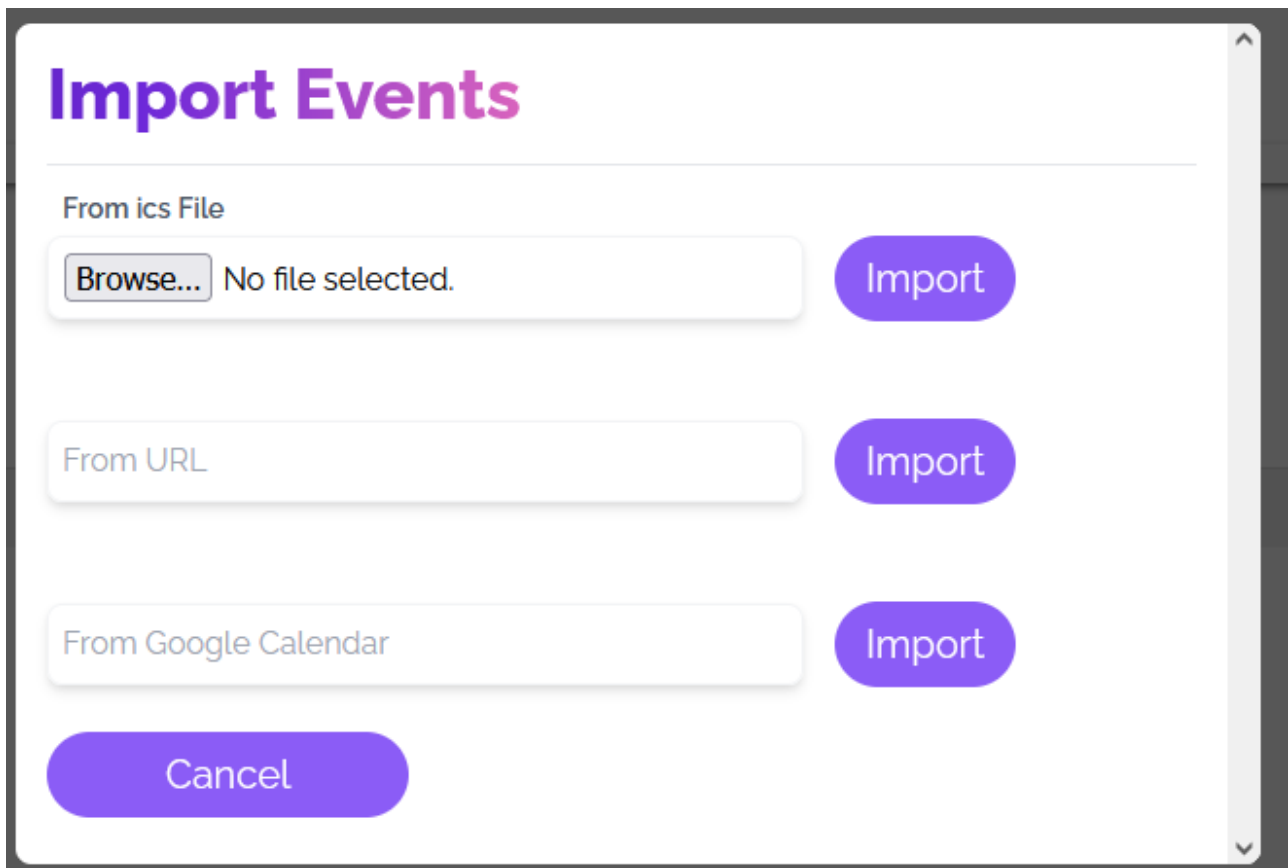
The image shows a web-based dialog box titled "Import Events" in a large, bold, purple font. Below the title, there are three distinct import methods. The first method, "From ics File", includes a text input field with a "Browse..." button and the text "No file selected.", followed by a purple "Import" button. The second method, "From URL", features a text input field and a purple "Import" button. The third method, "From Google Calendar", also has a text input field and a purple "Import" button. At the bottom left of the dialog is a large purple "Cancel" button. The entire interface is set against a light gray background with rounded corners and a subtle shadow.

Figure 9: Import d'événement

L'utilisateur peut importer des événements de trois facons différentes :

- Déposer un fichier iCalendar (ics) valide
- Donner une URL qui pointe vers une source GET de fichier iCalendar
- Donner une adresse GMail (l'API est actuellement en phase de test, et sera validée par Google dans les 6 semaines à venir ; durant ce temps, seules les adresses que nous avons spécifiées sont disponibles ; en voici une contenant un calendrier déjà rempli : **java.friday.test@gmail.com**)

4 Rapport technique

Cette section a pour but de présenter l'architecture Friday ainsi que nos choix d'implémentation.

4.1 Backend (API)

4.1.1 Tables

La base de données est composée des tables suivantes:

- User[@id, username, password]
L'ID est un UUID, tandis que username et password sont des varchar.
- Event[@id, User, title, description, place, recurRuleParts, startDate, endDate, latitude, longitude]
L'ID est un UUID, et une entité appartient à 1 User ; title, description, place et recurRuleParts sont des varchar (ce dernier représente les règles de récurrence de l'événement au format iCalendar), startDate et endDate sont des local-dates, et latitude et longitudes des flottans.
- Login[@(token, User), lastRefresh]
Cette table représente un token de login d'un utilisateur dont il peut se servir pour se connecter à l'application. Une entité appartient à 1 User, et lastRefresh est une local-date représentant la dernière fois qu'il a été utilisé.

(@k: clé primaire, k: clé étrangère, k: unique)

4.1.2 Problème

Il arrive que le calcul du prochain événement soit erroné à cause de l'heure.