TimeOffEvents : Gestion de demandes de congés

A. Périmètre fonctionnel

Pour conclure ce cours de programmation fonctionnelle, vous allez implémenter la logique d'un processus de demande et validation de congés.

Pour simplifier le domaine, nous allons poser les postulats suivants :

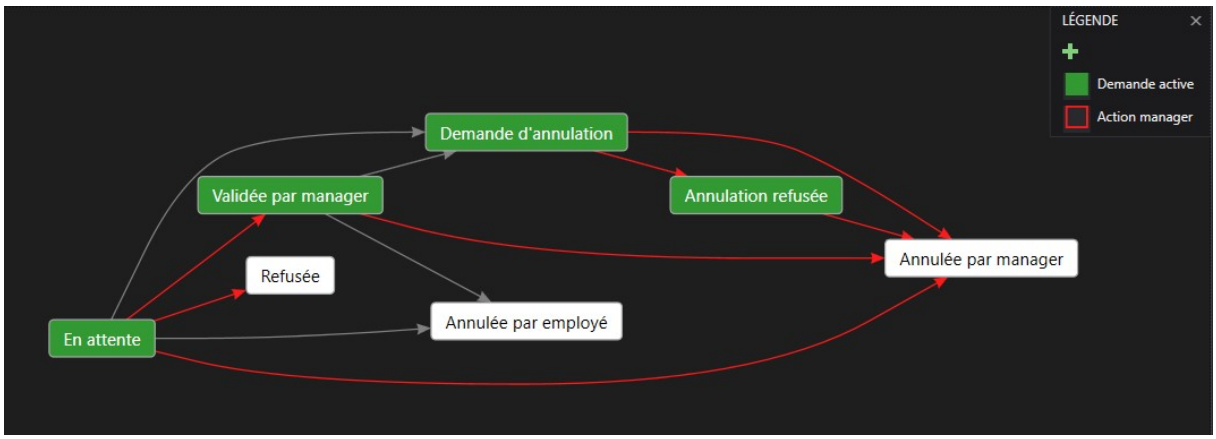
- On ne considère qu'un type de congés. Notamment, on ne fait pas la distinction entre Congés payés et RTT. Il n'est pas non plus demandé de gérer les notions de congés sans solde, évènements familiaux, jours de récupération.
- Tous les employés sont en CDI, on ne gère pas d'autre type de contrat.
- On ne considère que deux rôles, Employé et Manager.
- Le système doit pouvoir gérer plusieurs employés, mais l'identité du manager n'a pas à être gérée.
- On ignore le fait qu'un manager soit aussi un employé et qu'l puisse aussi vouloir demander des congés.
- On ignorera toute notion de fuseau horaire ou de changement d'heure.
- On considérera qu'on dispose d'une fonction qui nous renvoie la liste des jours fériés d'une année (pas de calcul à faire), et on prendra en compte les weekends.
- On considèrera que tous les employés sont en poste depuis le 1er janvier 2018, et qu'il n'est pas nécessaire de gérer des attributions de congés pour des portions de mois.
- On n'envisage pas que les congés attribués puissent être perdus s'ils ne sont pas pris.
- On ne prendra pas en compte l'ancienneté lors de la demande des congés : un employé peut effectuer une demande de congés d'une durée de 3 mois dès son jour d'arrivée. Le manager devra probablement la refuser, mais le système n'empêchera pas de créer la demande.

Le projet s'attache à gérer deux problématiques, détaillées plus bas. Pour chacune de ces problématiques, il est attendu de la part des étudiants de produire le code et les tests qu'ils estiment pertinents.

La plus petite unité de congé sera la demi-journée. Une demande de congés aura donc un début et une fin, inclues dans la période de la demande, et pourra par exemple s'écrire sous cette forme « du 20 décembre matin jusqu'au 22 décembre après-midi ».

1. La gestion du workflow de validation : « le cycle de vie d'une demande de congés »

Les différents états d'une demande de congés peuvent être modélisés ainsi :



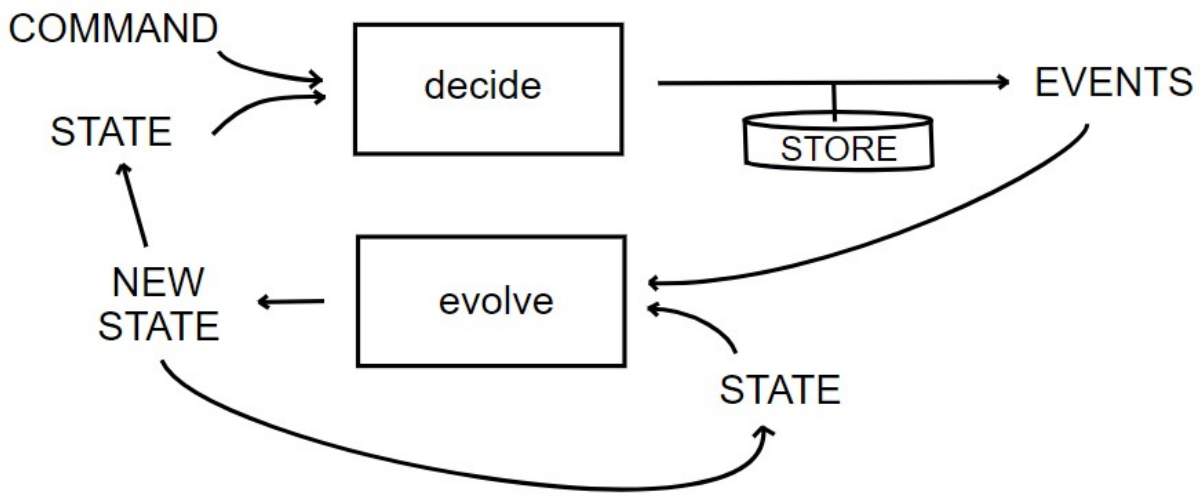
L'état initial d'une demande, lors de sa création, est « En attente ».

Les états colorés en vert sont les états dans lesquels une demande est « active », c'est-à-dire qu'on considère qu'il s'agit d'un congé pris (ou à prendre dans le futur), et qui doit être décompté du solde des congés disponibles (cf. partie 2). On notera qu'avec ce modèle, lorsqu'une demande devient « inactive », il n'existe aucune transition permettant de la réactiver.

Les règles à implémenter et vérifier sont les suivantes :

- Un employé peut uniquement effectuer des demandes qui commencent à une date future (au moins le lendemain de la date à laquelle la demande est effectuée).
- Un employé peut annuler une de ses propres demandes, validée ou non, si la date de début de celle-ci est dans le futur (au moins le lendemain de la date à laquelle la demande est effectuée).
- Un employé peut demander l'annulation d'une de ses demandes de congé, lorsque la date de début de celle-ci est dans le passé (ou qu'il s'agit du jour même).
- Le manager peut annuler n'importe quelle demande active.

L'implémentation devra être effectuée en utilisant les concepts de l'event-sourcing, les intentions des utilisateurs devront dont être représentées sous forme de commandes, et les résultats sous formes de listes d'évènements.



Remarque : pour tester les comportements liés à la date d'une action, il est nécessaire de remplacer les appels à `DateTime.Now` ou `DateTime.Date` par des appels à une source de temps. Une manière commune de procéder consiste à injecter cette source sous la forme d'une interface (`IDateProvider`, `IClock`…). En F#, on peut choisir d'utiliser une interface, ou à la place simplement une fonction getCurrentDate, avec pour signature `unit -> DateTime`.

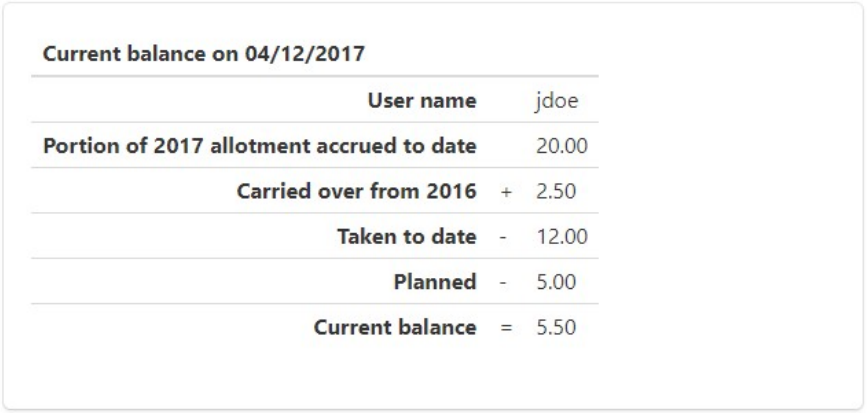
2. **Le calcul du solde de congés disponibles à une date donnée et affichage de l'historique des évènements liés au solde d'un employé**

La seconde problématique est le calcul du solde de congés, ainsi que l'affichage du détail du calcul. L'idée est de produire un modèle permettant d'afficher un écran similaire à celui présenté ci-dessous.

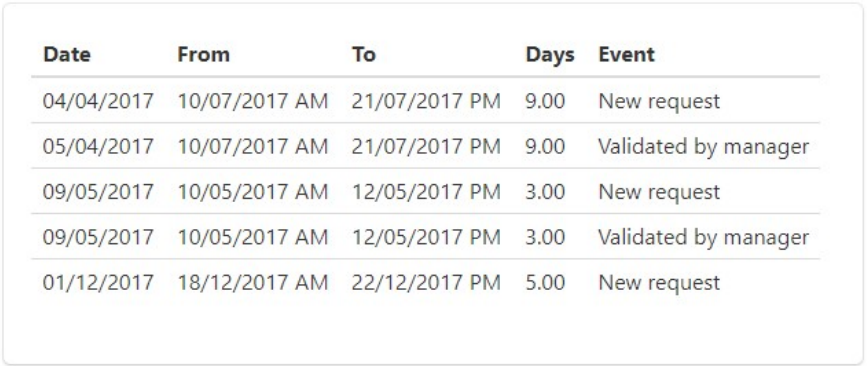
Celui-ci devra pouvoir afficher, pour une date de consultation :

- un récapitulatif de situation des congés :
    - **A** : Le cumul des congés attribués depuis le début de l'année civile (seuls les mois terminés sont considérés attribués, on ne gère pas les attributions pour des portions de mois).
    - **B** : Le report éventuel du solde de congés en fin d'année précédente (qu'il soit positif ou négatif)
    - **C** : Les congés effectifs : le cumul des demandes actives dont la date de début se situe entre le début de l'année et la date de consultation (incluse).
    - **D** : Les congés prévus : le cumul des demandes actives dont la date de début se situe entre le lendemain de la date de consultation et la fin de l'année.
    - **E** : Le solde disponible pour l'année, c'est-à-dire **A + B – (C + D)**
- l'historique des demandes, reprenant chronologiquement les évènements (relatifs aux demandes de l'année considérée) associés aux demandes de l'utilisateur.

Toutes les données nécessaires à cet affichage devront donc être retournées en résultat d'une fonction. Il est demandé de pouvoir afficher une situation à n'importe quelle date (postérieure à l'entrée de l'employé dans l'entreprise).



Current balance on 04/12/2017		
User name		jdoe
Portion of 2017 allotment accrued to date		20.00
Carried over from 2016	+	2.50
Taken to date	-	12.00
Planned	-	5.00
Current balance	=	5.50



Date	From	To	Days	Event
04/04/2017	10/07/2017 AM	21/07/2017 PM	9.00	New request
05/04/2017	10/07/2017 AM	21/07/2017 PM	9.00	Validated by manager
09/05/2017	10/05/2017 AM	12/05/2017 PM	3.00	New request
09/05/2017	10/05/2017 AM	12/05/2017 PM	3.00	Validated by manager
01/12/2017	18/12/2017 AM	22/12/2017 PM	5.00	New request

B. Périmètre technique

La réalisation du projet va s'organiser en 3 étapes :

1. **L'implémentation de la logique métier**

Après avoir étudié en cours les outils nécessaires, nous partirons d'un projet presque vide, pour aboutir sur un squelette d'implémentation de la logique métier en utilisant la technique de l'event-sourcing.

2. **La réalisation d'une API web**

Après avoir étudié en cours les outils nécessaires, nous améliorerons la solution existante pour y ajouter une API au dessus de la logique métier.

3. **La mise en place de l'infrastructure (persistance…)**

Nous mettrons en place l'infrastructure nécessaire pour permettre à l'application de s'exécuter dans un contexte réel, et non uniquement dans le cadre de tests automatisés. Une fois celle-ci en place, nous devrions pouvoir tester notre application au travers d'appels à l'API (avec curl ou Postman par exemple), et persister l'état du système entre deux exécutions.

4. **La réalisation d'un front-end web**

Après avoir étudié en cours les outils nécessaires, nous verrons comment ajouter un front-end utilisant l'API réalisée.

Le projet de base, dont l'implémentation sera réalisée lors d'une des séances de cours, sera ensuite publié dans un un repository github dédié, dont l'adresse sera donnée lors du cours.

Les bases des améliorations suivantes (ajout de l'API, mise en place de l'infrastructure, ajout du front-end), seront disponibles sous la forme de branches sur le même repository, et seront rendues disponibles au fur et à mesure de l'avancement du cours.

Toute question concernant le sujet ou l'implémentation déjà réalisée pourra être posée lors des cours ou sous forme d'issue directement sur le repository github.