SR03 MINI PROJET 01

Trombinoscope

delaunay gregory, KELLER VINCENT, PARTIER XAVIER

P16

Table des matières

[INTRODUCTION 1](#_Toc447470310)

[Répartition des tâches 1](#_Toc447470311)

[I – Site Web 2](#_Toc447470312)

[A- FORMULAIRES 2](#_Toc447470313)

[B- Trombinoscope 3](#_Toc447470314)

[II – Android 4](#_Toc447470315)

[A- Côté serveur 4](#_Toc447470316)

[B- Coté Vue 4](#_Toc447470317)

# INTRODUCTION

L'objectif de ce mini-projet est de développer une application Web qui permet soit par la saisie du nom ou du prénom, soit par le choix d'une structure, d'afficher des informations sur les membres de l'UTC, personnel et étudiant. Afin de développer cette application, nous avons choisi d'utiliser le framework JavaScript AngularJS et le modèle MVC (Modèle-Vue-Contrôleur) que ce framework emploie.

L'application est constituée de deux parties, le formulaire et la grille qui contient les informations liées à la recherche de l'utilisateur.

# Répartition des tâches

Pour ce mini projet, M. Keller Vincent se sera occupé de la partie AngularJS et MVC du projet afin de mettre en place la communication avec le serveur et la séparation des composants du projet. M. Partier Xavier se sera occupé de l’ergonomie et du CSS. M. Delaunay Grégory se sera chargée de réaliser l’application Android de ce même projet et aura apporté son aide sur la partie liée au CSS d’Angular.

# I – Site Web

## Formulaire

Il existe dans cette application deux types de formulaires, le premier permet la saisie de données par l'utilisateur (nom, prénom) tandis que le second lui permet de choisir une structure et une éventuelle sous structure. Afin de gérer les formulaires, plusieurs éléments interagissent au sein de l'application. Tout d'abord, le contrôleur, il s'agit de l'élément central, il reçoit les notifications de la vue en cas de changement ou d'interaction avec l'utilisateur et transmet ensuite les informations au modèle afin que celui-ci récupère les données auprès du Web Service. Ce contrôleur effectue ainsi plusieurs opérations :

* Il initialise les modèles et la vue au lancement de l'application
* Le lien entre les données et la vue est effectuer à travers des champs du contrôleur quand il a récupéré les données du modèle
* Il vérifie qu'au moins un champ est rempli lors l'utilisateur effectue une recherche par nom et/ou prénom
* Il provoque les modifications de données lorsque la vue signale un changement de sélection de structure ou sous-structure
* Il vérifie que le Web Service est disponible avant de demander aux modèles de récupérer les données

Il y a également la vue crée en utilisant le langage HTML étendu par AngularJS mais Angular Material qui permet d'ajouter certains composant aux interfaces des applications utilisant AngularJS. Cette vue se constitue de deux onglets correspondant aux deux types de recherches disponibles, la recherche par nom et la recherche par structure.

Enfin, plusieurs modèles sont utilisés par l'application afin de gérer ce formulaire, le modèle Structure, le modèle SousStructure et enfin Personne. Ces trois modèles permettent au contrôleur d'effectuer diverses opérations :

* Effectuer une requête vers un web service pour obtenir des données
* Ajouter un élément aux données du modèle
* Vider le modèle de ses données

## Trombinoscope

Le trombinoscope est quant à lui constitué d'une grille qui contient, dans chaque case, la photo d’une personne et les informations principales de celles-ci, afin d'obtenir l'intégralité des informations disponibles sur l'utilisateur, une fenêtre s'ouvre en cas de clic sur une case. Cette partie de l'application emploie également le modèle MVC et possède donc une vue sous forme d'HTML et CSS et exploite également le modèle Personne car seules ces données sont utiles.

La vue possède également plusieurs fonctionnalités :

* La pagination
* Le choix du nombre d'élément par page
* Le filtrage des données affichées en fonction d'une chaine pour correspondre au nom / prénom.

Ces fonctionnalités sont permises par l'interaction Vue-Contrôleur car c'est lui qui gère les filtres et qui informe la vue de la quantité d'information qu'elle peut afficher.

# II – Android

La partie Android reprend ce qui a été fait pour l’adapter sur une application mobile. On retrouvera donc les mêmes fonctionnalités à l’exception du filtrage post requête et du choix du nombre d’éléments par page, puisque la vue se construit autrement.

## Côté serveur

La librairie Retrofit est utilisée pour gérer les appels serveurs : L’avantage de cette librairie est qu’elle permet d’instancier très simplement des clients qui iront faire leur requête sur une URL bien précise donnée en appelant la méthode liée à cette URL et de retranscrire le résultat directement en objet Java à la seule condition de lui avoir spécifié le bon modèle de données.

Pour la gestion du chargement et du cache des images, la librairies Glide est utilisée. Pareil, elle permet de simplifier grandement le processus de chargement tout en gardant des performances optimales.

## Coté Vue

Le formulaire contient donc les différents champs permettant de faire la recherche soit par nom/prénom, soit par structure.

Les résultats de la recherche sont affichés au sein d’une RecyclerView. Cet élément permet lors du défilement des résultats, de récupérer les cellules et de simplement changer les valeurs des champs sans avoir à tout recréer à chaque fois, et donc permet un gain en performance majeur. Il fonctionne à l’aide d’un ViewHolder et d’un Adapter : Le ViewHolder permet de définir l’affichage d’une cellule, et l’Adapter fait le lien entre le jeu de données du RecyclerView et le ViewHolder pour faire correspondre un ViewHolder par élément du jeu de données.