

|  |
| --- |
| HoMoMa  Houses Money Manager |



Mbayo Guilain

[guilain.mbayo@cpnv.ch](mailto:guilain.mbayo@cpnv.ch)

SI-CA 2a

07.02.2020



Table des matières

[1 Introduction 3](#__RefHeading___Toc5088_182692148)

[1.1 Cadre, description et motivation 3](#__RefHeading___Toc5090_182692148)

[1.2 Organisation 3](#__RefHeading___Toc5092_182692148)

[1.3 Objectifs 3](#__RefHeading___Toc5094_182692148)

[1.4 Planification initiale 4](#__RefHeading___Toc5096_182692148)

[2 Analyse 5](#__RefHeading___Toc5098_182692148)

[2.1 Maquettes 5](#__RefHeading___Toc5100_182692148)

[2.2 Use cases et scénarios 7](#__RefHeading___Toc5102_182692148)

[2.2.1 Afficher les revenus de la famille 8](#__RefHeading___Toc5104_182692148)

[2.2.2 Afficher les dépenses de la famille 9](#__RefHeading___Toc5106_182692148)

[2.2.3 Créer un nouvel élément 10](#__RefHeading___Toc5108_182692148)

[2.2.4 Modifier un élément 13](#__RefHeading___Toc5110_182692148)

[2.2.5 Afficher un récapitulatif 14](#__RefHeading___Toc5112_182692148)

[2.3 Modèle Conceptuel de Données 14](#__RefHeading___Toc5114_182692148)

[2.4 Stratégie de test 15](#__RefHeading___Toc5116_182692148)

[2.5 Budget 15](#__RefHeading___Toc5118_182692148)

[3 Implémentation 16](#__RefHeading___Toc5120_182692148)

[3.1 Vue d’ensemble 16](#__RefHeading___Toc5122_182692148)

[3.2 Choix techniques 16](#__RefHeading___Toc5124_182692148)

[3.3 Points techniques spécifiques 16](#__RefHeading___Toc5126_182692148)

[3.3.1 Drawer 16](#__RefHeading___Toc5128_182692148)

[3.3.2 Fragments 18](#__RefHeading___Toc5130_182692148)

[3.3.3 Écrire dans le tableau 19](#__RefHeading___Toc5132_182692148)

[3.3.4 Architecture du code 20](#__RefHeading___Toc5134_182692148)

[3.4 Livraisons 21](#__RefHeading___Toc5136_182692148)

[4 Tests 21](#__RefHeading___Toc5138_182692148)

[4.1 Tests effectués 21](#__RefHeading___Toc5140_182692148)

[4.2 Erreurs restantes 22](#__RefHeading___Toc5142_182692148)

[5 Conclusions 22](#__RefHeading___Toc5144_182692148)

[6 Annexes 23](#__RefHeading___Toc5146_182692148)

[6.1 Sources – Bibliographie 23](#__RefHeading___Toc5148_182692148)

[6.2 Journal de bord du projet 23](#__RefHeading___Toc5150_182692148)

# Introduction

## Cadre, description et motivation

Ce projet est réalisé dans le cadre du « Projet (prog/web) » du CPNV, transformé en projet pré-TPI.

Il consistera en la création d’une application android de gestion d’un budget familial.

Cette application permettra aux membre de la famille d’entrer les informations concernant leur revenu global et d’établir un budget clair.

Les données seront stockées de manière à ce que chaque membre de la famille puisse y avoir accès en temps réel (idéalement via google drive) et de manière transparente (il leur suffira de lancer l’application, qui se chargera ensuite du reste).

Ce travail me motive, car il me permet de découvrir une nouvelle technologie (android), un nouveau langage (kotlin) ainsi que de me pencher sur les éléments nécessaires à une bonne gestion d’un budget familial.

J’aurais aussi l’occasion de réfléchir au meilleur moyen de rendre mon application ergonomique et intuitive pour un confort utilisateur maximum.

L’acquisition de ces nouvelles compétences me paraît intéressante autant au niveau professionnel que personnel.

## Organisation

Elève :

Nom : Mbayo Prénom : Guilain

E-mail : [Guilain.MBAYO@cpnv.ch](mailto:Guilain.MBAYO@cpnv.ch) Tél : 079 896 17 45

Chef de projet :

Nom : Virret Prénom : Loïc

E-mail : [Loïc.VIRET@cpnv.ch](mailto:Loïc.VIRET@cpnv.ch) Tél : 079 344 07 35

Le projet sera entièrement réalisé par l’élève. Il indiquera dans ce document toute aide extérieur notable et tiendra un journal de travail afin de pouvoir communiquer régulièrement l’état d’avancement exact du projet au chef de projet.

Le chef de projet contrôlera le bon avancement du travail et pourra apporter son expertise du domaine en cas de problème.

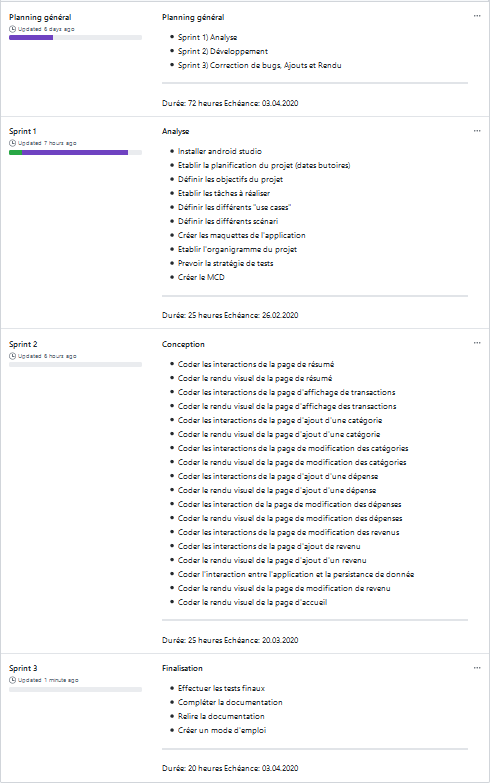
## Objectifs

Un dossier complet et précis du déroulement du projet devra être réaliser d’ici au 3 mars 2020.

L’application HomoMa sera réalisée selon les contraintes fixées dans le cahier des charges. D’ici au 3 mars 2020.

## Planification initiale

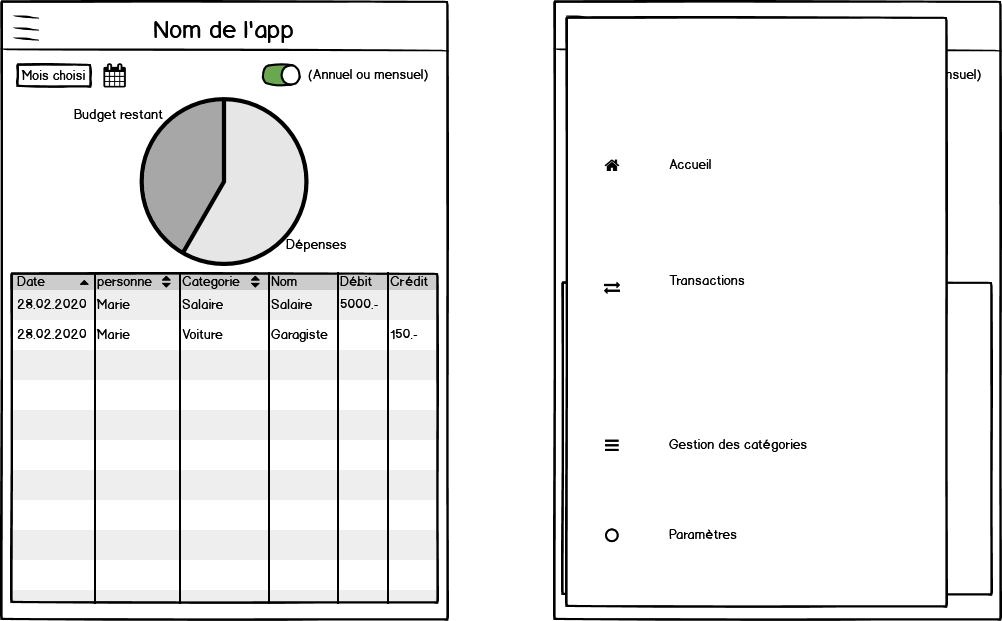
Le projet sera planifié selon la méthode AGILE.

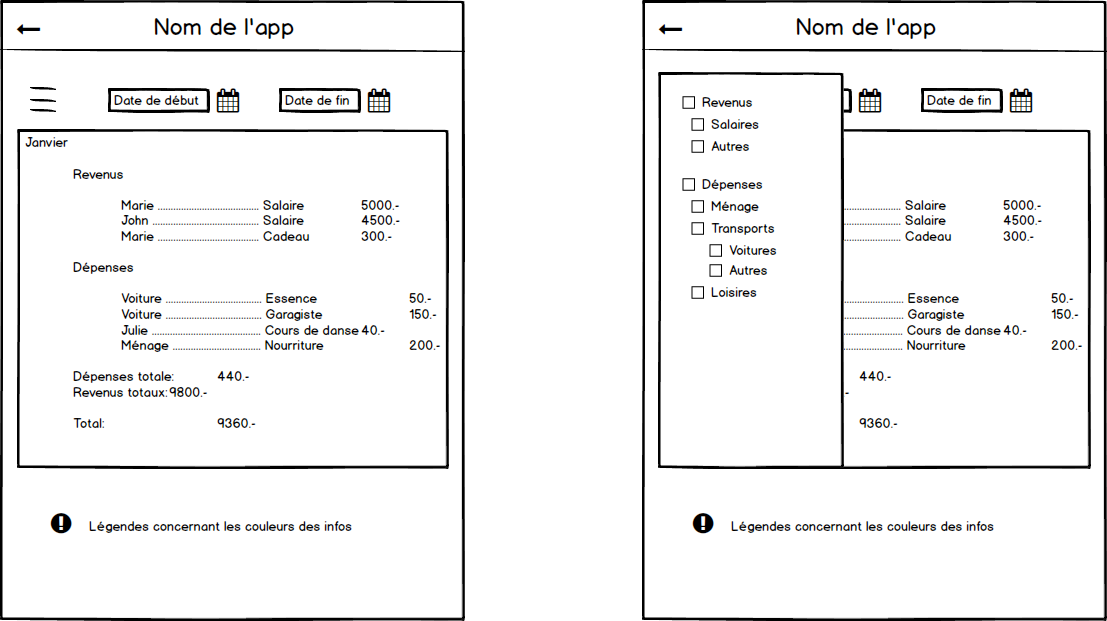


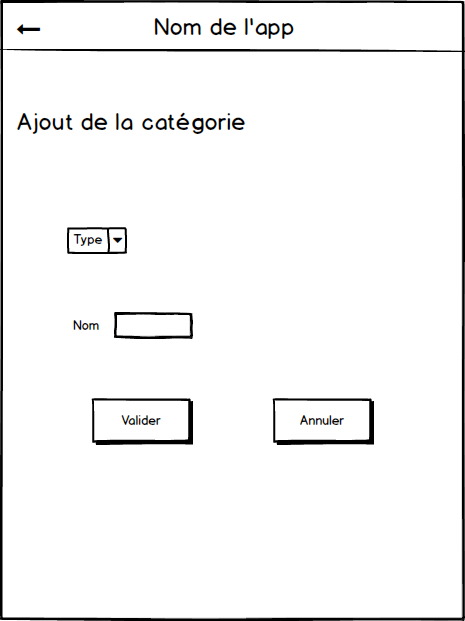
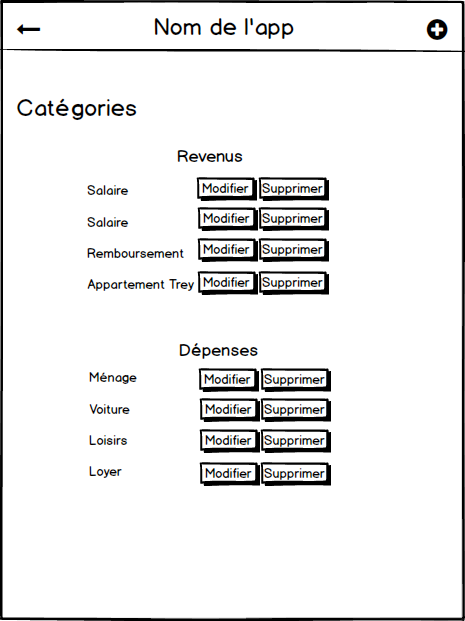
*Lien vers le github : https://github.com/GMbayo/GestionnaireDeBudgetAndroid*

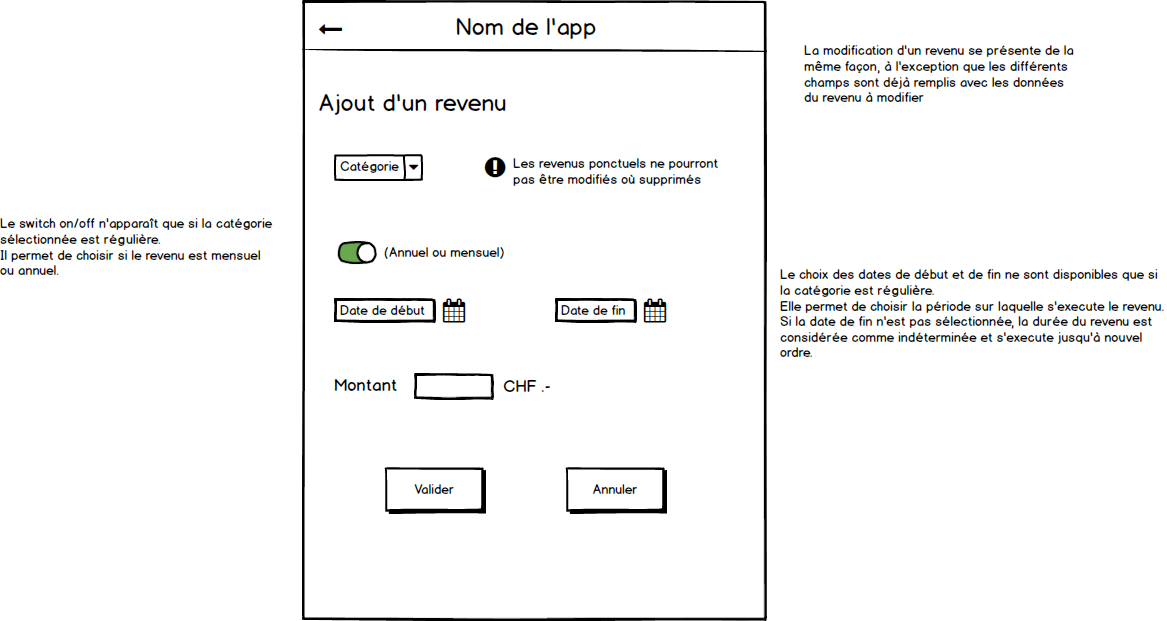
# Analyse

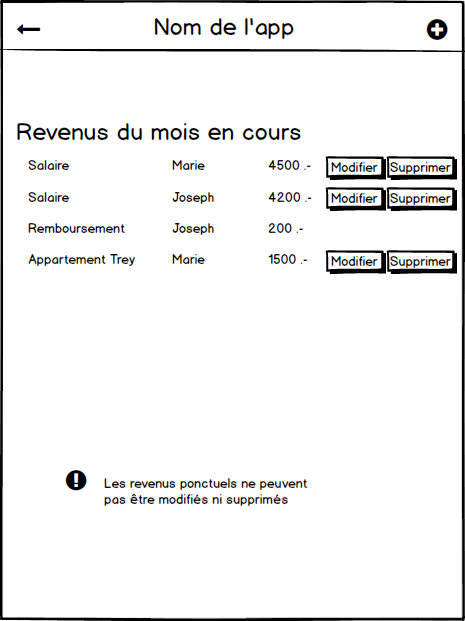
## Maquettes

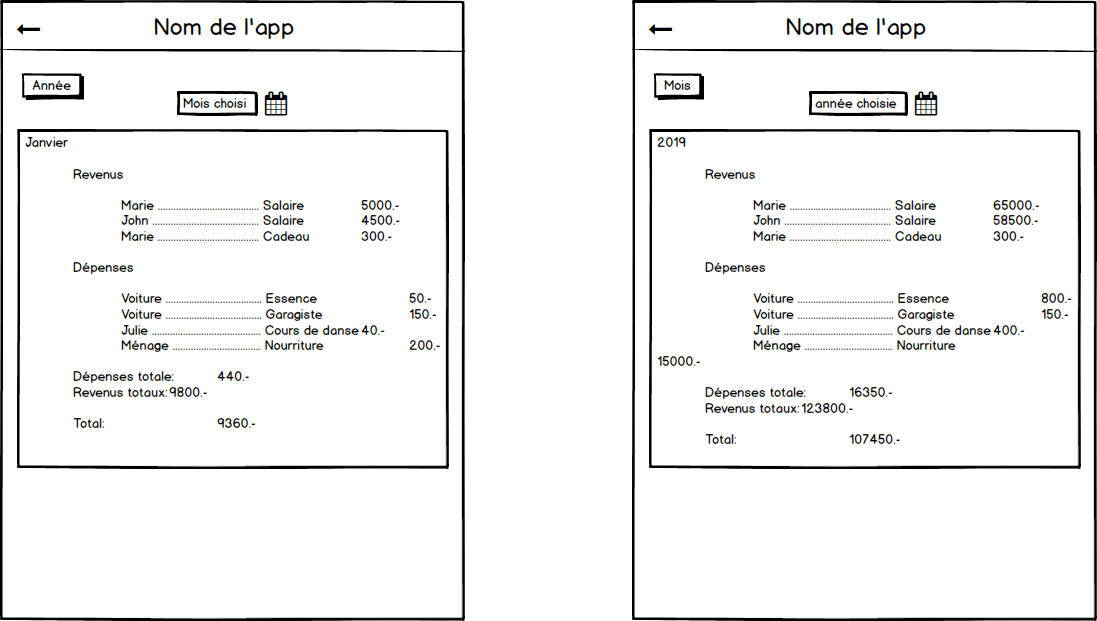












## Use cases et scénarios

**2.2.1 Afficher les revenus de la famille**

*2.2.1.1 Afficher les salaires*

*2.2.1.2 Afficher les autres types de revenus*

*2.2.1.3 Afficher tous les revenus*

*2.2.1.4 Afficher les revenus d’un autre mois*

**2.2.2 Afficher les dépenses de la famille**

*2.2.2.1 Afficher les dépenses du mois courant*

*2.2.2.2 Afficher les dépenses d’un autre mois*

**2.2.3 Créer un nouvel élément**

*2.2.3.1 Ajouter un nouveau revenu régulier*

*2.2.3.2 Ajouter un nouveau revenu ponctuel*

*2.2.3.3 Ajouter une nouvelle dépense régulière*

*2.2.3.4 Ajouter une nouvelle dépense ponctuelle*

*2.2.3.5 Ajouter une nouvelle catégorie de dépense*

*2.2.3.6 Ajouter une nouvelle catégorie de revenu*

**2.2.4 Modifier un élément**

*2.2.4.1 Modifier un revenu*

*2.2.4.2 Modifier une dépense*

*2.2.4.3 Supprimer une catégorie*

### Afficher les revenus de la famille

#### Afficher les salaires

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| J’ouvre l’application |  | La page d’accueil s’affiche |
| J’appuie sur l’icône « Transactions » dans le drawer |  | La page de transaction s’affiche avec par défaut les données du mois en cours |
| Je désélectionne toute les cases à l’exception de « Revenus » et « Salaires » dans le panneau des options (cf maquettes) |  | Le salaires sont affichés |

#### Afficher les autres types de revenus

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| J’ouvre l’application |  | La page d’accueil s’affiche |
| J’appuie sur l’icône « Transactions » dans le drawer |  | La page de transaction s’affiche avec par défaut les données du mois en cours |
| Je désélectionne toute les cases à l’exception de « Revenus » et « Autres » dans le panneau des options (cf maquettes) |  | Le autres types de revenus sont affichés |

#### Afficher tous les revenus

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| J’ouvre l’application |  | La page d’accueil s’affiche |
| J’appuie sur l’icône « Transactions » dans le drawer |  | La page de transaction s’affiche avec par défaut les données du mois en cours |
| Je désélectionne toute les cases à l’exception de « Revenus », « Salaires » et « Autres » dans le panneau des options (cf maquettes) |  | Tous les revenus sont affichés |

#### Afficher les revenus d’un autre mois

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| J’ouvre l’application |  | La page d’accueil s’affiche |
| J’appuie sur l’icône « Transactions » dans le drawer |  | La page de transaction s’affiche avec par défaut les données du mois en cours |
| Je désélectionne toute les cases à l’exception de « Revenus », « Salaires » et « Autres » dans le panneau des options (cf maquettes) |  | Tous les revenus sont affichés |
| Je choisis le mois de départ ainsi que le mois de fin de la sélection à l’aide des menus déroulants (cf maquettes) |  | Les revenus ayant été perçus ou étant prévus au cours des mois sélectionnés sont affichés |

### Afficher les dépenses de la famille

#### Afficher les dépenses du mois courant

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| J’ouvre l’application |  | La page d’accueil s’affiche |
| J’appuie sur l’icône « Transactions » dans le drawer |  | La page de transaction s’affiche avec par défaut les données du mois en cours |
| Je désélectionne toute les cases à l’exception de celles du type « Dépenses » dans le panneau des options (cf maquettes) |  | Les dépenses du mois courant sont affichées |

#### Afficher les dépenses d’un autre mois

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| J’ouvre l’application |  | La page d’accueil s’affiche |
| J’appuie sur l’icône « Transactions » dans le drawer |  | La page de transaction s’affiche avec par défaut les données du mois en cours |
| Je désélectionne toute les cases à l’exception de celles du type « Dépenses » dans le panneau des options (cf maquettes) |  | Les dépenses du mois courant sont affichées |
| Je choisis le mois de départ ainsi que le mois de fin de la sélection à l’aide des menus déroulants (cf maquettes) |  | Les dépenses ayant été faites ou étant prévues au cours des mois sélectionnés sont affichés |

Les autres options d’affichage des revenus et des dépenses se passent selon le même mode opératoire que présenté jusqu’ici.

### Créer un nouvel élément

#### Ajouter un nouveau revenu régulier

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| J’ouvre l’application |  | La page d’accueil s’affiche |
| J’appuie sur l’icône « Transactions » dans le drawer |  | La page de transaction s’affiche avec par défaut les données du mois en cours |
| J’appuie sur l’icône de modification des revenus |  | La page de modification des revenus s’affiche |
| J’appuie sur le bouton « Ajouter un revenu » |  | La page d’ajout d’un revenu s’affiche |
| Je sélectionne une catégorie ayant « régulier » entre parenthèse à côté du nom (par exemple la catégorie « Salaire ») à l’aide du menu déroulant (cf maquettes) |  | Le choix de la catégorie est fait |
| J’écris le montant du revenu dans la zone prévue à cet effet (cf maquettes) |  | Le montant est choisi |
| J’appuie sur le bouton « Valider » |  | Le revenu a été ajouté |

#### Ajouter un nouveau revenu ponctuel

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| J’ouvre l’application |  | La page d’accueil s’affiche |
| J’appuie sur l’icône « Transactions » dans le drawer |  | La page de transaction s’affiche avec par défaut les données du mois en cours |
| J’appuie sur l’icône de modification des revenus |  | La page de modification des revenus s’affiche |
| J’appuie sur le bouton « Ajouter un revenu » |  | La page d’ajout d’un revenu s’affiche |
| Je sélectionne une catégorie n’ayant pas « régulier » entre parenthèse à côté du nom à l’aide du menu déroulant (cf maquettes) |  | Le choix de la catégorie est fait |
| J’écris le montant du revenu dans la zone prévue à cet effet (cf maquettes) |  | Le montant est choisi |
| J’appuie sur le bouton « Valider » |  | Le revenu a été ajouté |

#### Ajouter une nouvelle dépense régulière

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| J’ouvre l’application |  | La page d’accueil s’affiche |
| J’appuie sur l’icône « Transactions » dans le drawer |  | La page de transaction s’affiche avec par défaut les données du mois en cours |
| J’appuie sur l’icône de modification des dépenses |  | La page de modification des revenus s’affiche |
| J’appuie sur le bouton « Ajouter une dépense » |  | La page d’ajout d’une dépense s’affiche |
| Je sélectionne une catégorie ayant « régulier » entre parenthèse à côté du nom à l’aide du menu déroulant (cf maquettes) |  | Le choix de la catégorie est fait |
| J’écris le montant du revenu dans la zone prévue à cet effet (cf maquettes) |  | Le montant est choisi |
| J’appuie sur le bouton « Valider » |  | La dépense a été ajouté |

#### Ajouter une nouvelle dépense ponctuelle

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| J’ouvre l’application |  | La page d’accueil s’affiche |
| J’appuie sur l’icône « Transactions » dans le drawer |  | La page de transaction s’affiche avec par défaut les données du mois en cours |
| J’appuie sur l’icône de modification des dépenses |  | La page de modification des revenus s’affiche |
| J’appuie sur le bouton « Ajouter une dépense » |  | La page d’ajout d’une dépense s’affiche |
| Je sélectionne une catégorie n’ayant pas « régulier » entre parenthèse à côté du nom à l’aide du menu déroulant (cf maquettes) |  | Le choix de la catégorie est fait |
| J’écris le montant du revenu dans la zone prévue à cet effet (cf maquettes) |  | Le montant est choisi |
| J’appuie sur le bouton « Valider » |  | La dépense a été ajouté |

#### Ajouter une nouvelle catégorie de dépense

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| J’ouvre l’application |  | La page d’accueil s’affiche |
| J’appuie sur l’icône « Catégories » dans le drawer |  | La page des catégories s’affiche |
| J’appuie sur l’icône de modification des catégories |  | La page de modification des catégories s’affiche |
| J’appuie sur le bouton « Ajouter une catégorie » |  | La page d’ajout d’une catégorie s’affiche |
| Je sélectionne le type « dépense » à l’aide de la zone de type |  | Le choix du type de la catégorie est fait |
| Je rempli le champs du nom de la catégorie |  | Le nom est choisi |
| J’appuie sur le bouton « Valider » |  | La catégorie a été ajouté |

#### Ajouter une nouvelle catégorie de revenu

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| J’ouvre l’application |  | La page d’accueil s’affiche |
| J’appuie sur l’icône « Catégories » dans le drawer |  | La page des catégories s’affiche |
| J’appuie sur l’icône de modification des catégories |  | La page de modification des catégories s’affiche |
| J’appuie sur le bouton « Ajouter une catégorie » |  | La page d’ajout d’une catégorie s’affiche |
| Je sélectionne le type « revenu » l’aide de la zone de type |  | Le choix du type de la catégorie est fait |
| Je rempli le champs du nom de la catégorie |  | Le nom est choisi |
| J’appuie sur le bouton « Valider » |  | La catégorie a été ajouté |

### Modifier un élément

#### Modifier un revenu

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| J’ouvre l’application |  | La page d’accueil s’affiche |
| J’appuie sur l’icône « Transactions » dans le drawer |  | La page de transaction s’affiche avec par défaut les données du mois en cours |
| J’appuie sur l’icône de modification des revenus |  | La page de modification des revenus s’affiche |
| J’appuie sur le bouton « Modifier » à côté du revenu à modifier |  | La page de modification d’un revenu s’affiche |
| Je modifie les informations nécessaires |  | La modification est effectuée |
| J’appuie sur le bouton « Valider » |  | La modification est enregistrée |

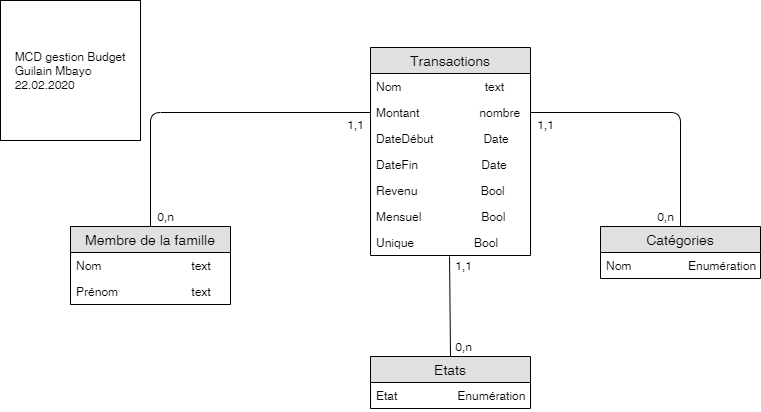
#### Modifier une dépense

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| J’ouvre l’application |  | La page d’accueil s’affiche |
| J’appuie sur l’icône « Transactions » dans le drawer |  | La page de transaction s’affiche avec par défaut les données du mois en cours |
| J’appuie sur l’icône de modification des dépenses |  | La page de modification des dépenses s’affiche |
| J’appuie sur le bouton « Modifier » à côté de la dépense à modifier |  | La page de modification d’une dépense s’affiche |
| Je modifie les informations nécessaires |  | La modification est effectuée |
| J’appuie sur le bouton « Valider » |  | La modification est enregistrée |

#### Supprimer une catégorie

|  |  |  |
| --- | --- | --- |
| **Action** | **Condition particulière** | **Réaction** |
| J’ouvre l’application |  | La page d’accueil s’affiche |
| J’appuie sur l’icône « Catégories » dans le drawer |  | La page des catégories s’affiche |
| J’appuie sur l’icône de modification des catégories |  | La page de modification des catégories s’affiche |
| J’appuie sur le bouton « Supprimer » à côté de la catégorie à supprimer |  | Une demande de confirmation est affichée |
| Je sélectionne « oui » |  | La catégorie est supprimée |

## Modèle Conceptuel de Données



## Stratégie de test

Le gestionnaire de budget sera effectué avec la version 3.6.0.0 d’Android Studio sur une machine Windows 10.

Des tests fonctionnels seront effectué au fur et à mesure de la progression du projet à l’aide d’un émulateur d’Android.

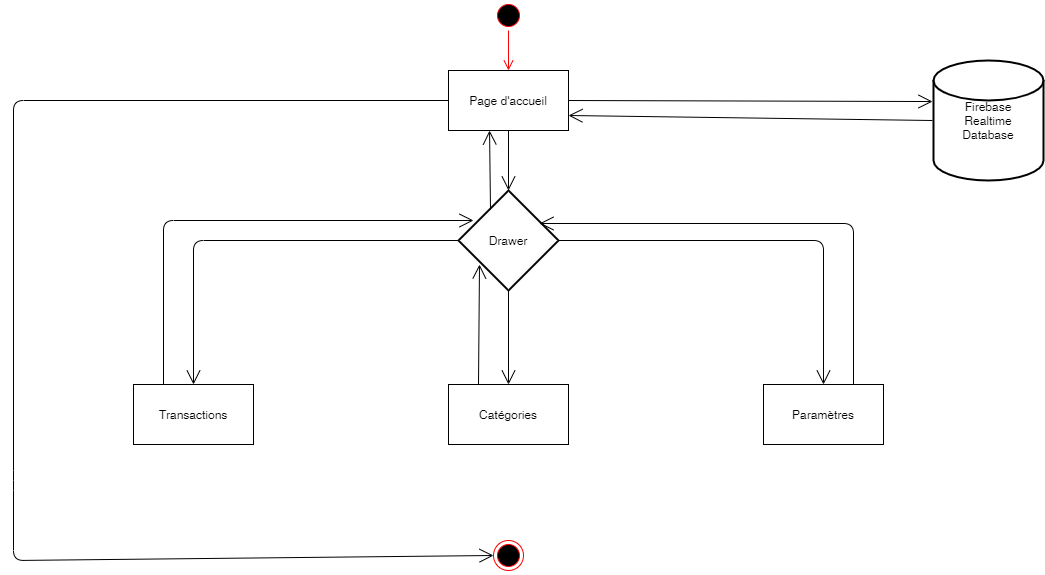
Un ou des camarades, membres de ma famille et moi-même effectuerons également des tests fonctionnels, de robustesse et de performance sur différents smartphones ayant un OS Android à différents moments clé de la réalisation du projet.

## Budget

Aucune dépense nécessaire.

# Implémentation

## Vue d’ensemble



## Choix techniques

Matériel HW : Ordinateur du CPNV

OS : Windows 10

Logiciel : Android Studio

Langage : Kotlin, XML

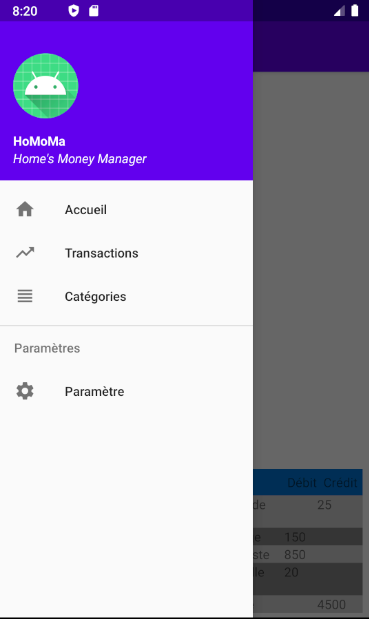
Ces choix ont été fait car il s’agit d’un environnement de développement android connu pour lequel il est possible de trouver de la documentation et des explications relativement détaillées.

Bien que le Kotlin n’ait pas été vu en cours (le xml presque pas non plus), il a été choisi au détriment du Java car c’est un langage plus récent et qui me paraît mieux représenter l’avenir.

## Points techniques spécifiques

### Drawer

Le drawer sert de lien de navigation entre les différentes parties principales de l’application. Bien que la plupart des fonctionnalités n’aient finalement pas été développées, les quatre fragments principaux ont été créés, et sont accessibles via ce drawer, ce qui en fait un élément central de l’UI.

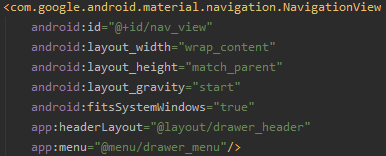


Le drawer est composé de plusieurs parties.

Premièrement, l’entête, qui contient une icône d’application (ici une simple icône android), et le titre de l’application.

Ensuite, le menu, qui liste les différents fragments principaux de l’application (sous la forme d’item).

Ces deux parties sont créée dans deux fichier xml différents (« drawer\_header.xml » et « drawer\_menu.xml ») avant d’être incrustée comme partie du drawer dans le fichier « main\_activity.xml ».



Les interactions déclenchées par chacune des entrées du menu sont ensuite codées dans un fichier kotlin nommé « MainActivity.kt ».

Pour cela, un « listener » est ajouté à notre drawer afin de détecter lorsque l’on clique sur un item. Puis, à l’aide d’une redéfinition de la méthode « onNavigationItemSelected », on indique le fragment à afficher en fonction de l’item sélectionné.



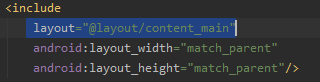
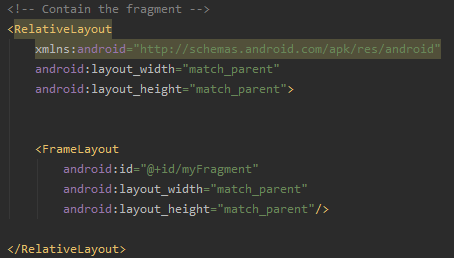
### Fragments

L’utilisation de fragments permet de créer différentes vues et de les appeler au sein d’une seule et même activité, ce qui a pour effet de rendre les changements beaucoup plus fluides. En effet, si l’on tente de créer chaque vue dans une activité différente, l’application sera obligée d’ouvrir cette activité à chaque fois, ce qui n’est pas toujours pertinent. Avec les fragments, seul le contenu de l’activité va changer.

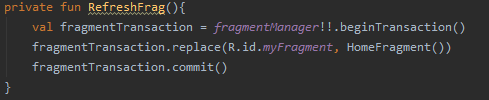
Comme expliqué au point précédant, les différents fragments sont appelés en fonction de l’item sélectionné dans le drawer.

Chaque fragment possède un fichier xml pour la vue et un fichier kotlin pour les interactions.

Ces fragments sont ensuite chargés à l’intérieur de la vue créée par le fichier « content\_main.xml », qui est elle-même chargée dans « main\_activity.xml ».



Dans le fichier « HomeFragment.kt », il était nécessaire de recharger le fragment pour pouvoir afficher toute les infos. Ceci a été fait à l’aide de la fonction suivante :

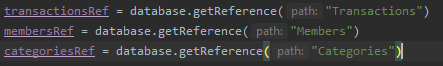


### Écrire dans le tableau

Afin d’écrire les données dans le tableau, il fallait commencer par ouvrir la connexion avec la persistance de données (Dans Android Studio, ce procédé se fait automatiquement à l’aide d’un assistant).



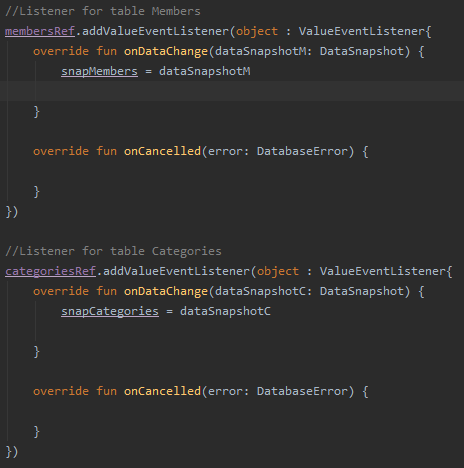
Ensuite, il fallait créer une référence sur la persistance de données.



Comme illustré ci dessus, HoMoMa utilise plusieurs références pointant sur différentes table.

Ce choix a été fais car il fallait pouvoir assigner des eventsListener sur plusieurs tables afin de récupérer leurs données et de pouvoir les ré-afficher en cas de modification des données dans la persistance.

Cela a également permis d’appliquer une requête sur la référence à la table « transaction », dans le but d’appliquer un eventListener non pas à toute la table, mais seulement au résultat de la requête, à savoir les 5 dernières transactions entrées.



L’application se sert du résultat des méthode contenues dans les ValueEventListener afin de récupérer un snapshot des données au premier appel de la méthode ainsi qu’à chacune de leurs modifications.

Il est dès lors possible de récupérer les données souhaitées à l’aide d’une boucle « for », de les associer à des TextView, d’associer ces TextView à une TableRow, et de rajouter cette TableRow au tableau.

La taille de ce tableau sera alors dynamique et dépendra du nombre de données contenues dans la table « Transaction » et allant jusque à un maximum de 5 lignes.

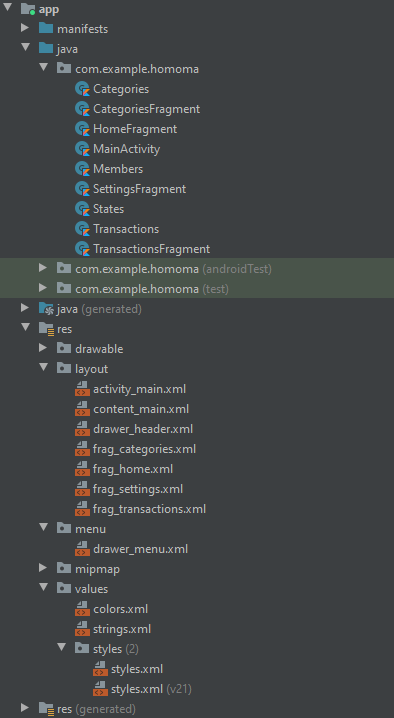
Je n’ai malheureusement pas eu le temps de réaliser les vues permettant de rentrer de nouvelles valeurs dans la persistance, mais le principe aurait été de générer une clé et de lui attribuer la ou les valeurs souhaitée à l’aide de la commande « setValue » (des tests ont été effectués dans ce sens mais n’ont pas pu être implémentés dans HoMoMa).

### Architecture du code

Le code se fait dans deux types de fichiers différents : les fichiers xml et le fichier kotlin.

Les fichiers xml servent à coder la vue.

Les fichiers kotlin servent à effectuer des actions en arrière plan, tel que modifier la vue, lier des vues entre elles, créer des objets, créer des actions spécifiques sur des boutons ou encore lier une base de donnée.



Lors du lancement de l’application, ce sont les fichiers MainActivity.kt et activity\_main.xml qui vont être utilisés.

Ils appellent ensuite différents autres fichiers afin d’afficher le drawer et le fragment contenant la page d’accueil (le tableau). Les fichiers utilisés par la suite sont déterminés par les actions de l’utilisateur.

## Livraisons

Rendu final : 03.04.20

Livrable :

* Rapport
* Cahier des charges
* Projet Android Studio
* Journal de travail
* Journal de bord (intégré dans le rapport)

# Tests

## Tests effectués

Tenant compte du fait que les attentes de bases ne sont pas atteintes, les tests ont été modifiés afin de ne pas avoir 80 % du tableau portant la mention « non implémenté ».

|  |  |
| --- | --- |
| Scénario | Développeur |
| Lancer l’application | Fonctionne |
| Déployer le drawer | Fonctionne |
| Changer de fragment | Fonctionne |
| Ajouter des données dans Firebase | Non implémenté |
| Charger des données depuis Firebase | Fonctionne |
| Les données rajoutées à la base sont ajoutés dynamiquement au tableau de la page d’accueil | Fonctionne |
| Les anciennes données du tableau sont supprimées pour être remplacées par les nouvelles | Ne fonctionne pas |
| L’application peut être installée sur un vrai natel | Fonctionne |

## Erreurs restantes

Lorsqu’une donnée est modifiées dans la persistance de données, le tableau ré-affiche toute les données, mais sans supprimer les anciennes colonnes.

Une solution possible à ce problème serait de créer l’entête du tableau dynamiquement, il serait ainsi possible de totalement le vider et de le remplir à chaque fois.

# Conclusions

Ce projet d’application est à la fois l’un des plus compliqués et l’un des plus intéressant que j’ai eu à réaliser au CPNV. Découvrir une parcelle du monde de la programmation Android m’as donné envie de continuer et de pouvoir réaliser une application aboutie et fonctionnelle.

Cela a cependant également été stressant, car à chaque nouvel élément à réaliser il m’as fallu passer plusieurs heures apprendre à faire en kotlin et en xml des choses qui aurait été faites en 15 minutes en C#. Parfois, certaines informations trouvées sur internet et corroborées par plusieurs sites et vidéos ne fonctionnait pas sur HoMoMa sans que je ne comprenne pourquoi, et c’est seulement après avoir bricolé plusieurs solutions différentes que l’une d’entre elles finissait par marcher.

La compréhension des interactions entre les différents fichiers a elle aussi pris du temps, et je sais qu’il me reste encore beaucoup de choses à comprendre.

L’application en tant que telle est inutilisable et ne rempli pas du tout les attentes fixées en début de projet.

Le temps nécessaire à l’apprentissage de la matière a été sous estimé, ce qui à conduit à un manque de temps sur la fin. La gestion du temps n’a donc pas été optimale.

Une certaine quantité de matière a cependant été acquise, et bien que le résultat final soit frustrant, je suis quand même heureux d’avoir accepté ce défi.

# Annexes

## Sources – Bibliographie

https://stackoverflow.com/questions/2108456/how-can-i-create-a-table-with-borders-in-android

https://www.androidly.net/136/android-tablelayout-using-kotlin

https://technotzz.wordpress.com/2011/11/04/android-dynamically-add-rows-to-table-layout/

https://www.semicolonworld.com/question/44062/how-to-return-datasnapshot-value-as-a-result-of-a-method

https://dzone.com/articles/storing-and-retrieving-data-from-firebase-with-kot

https://www.youtube.com/watch?v=fP\_Tp6ImrDM

https://www.youtube.com/watch?v=wgKDIgDFYto&t=51s

https://www.youtube.com/watch?v=NCU1RE93wy4&t=238s

https://www.androdocs.com/kotlin/implementing-navigation-drawer-in-android-app-using-kotlin.html

Une partie des sources se trouvant encore sur mon PC du CPNV, je me vois malheureusement dans l’incapacité de toute les nommer.

## Journal de bord du projet

|  |  |
| --- | --- |
| **Date** | **Événement** |
| 04.02.2020 | Recherche d’un projet et idée de Loïc sur un gestionnaire de budget android |
| 06.02.2020 | 30 minutes de perdues à cause d'un retard de train |
| 11.02.2020 | Discussion du cahier des charges avec le chef de projet |
| 12.02.2020 | Congé |
| 14.02.2020 | Debriefing avec Loïc |
| 25.02.2020 | Debriefing avec Loïc |
| 27.02.2020 | 40 min perdue à cause du train et d’une présentation du doyen |
| 28.02.2020 | Sprint review |
| 05.03.2020  10.03.2020 | 30 minutes de perdues à cause d'un retard de train  Debriefing avec Loïc |
| 18.03.2020 | Début du travail en confinement les horaires ne seront plus toujours respectés |
| 03.04.2020 | Rendu du projet |