Moneythoring

Cahier des charges

HEIG-VD

Projet de semestre

Table des matières

1 Descriptif 2

2 Fonctionnalités 2

2.1 Fonctionnalités de base 2

2.1.1 Création compte utilisateur 2

2.1.2 Connexion sécurisée 2

2.1.3 Compte bancaire 2

2.1.4 Catégorie 3

2.1.5 Devise 3

2.1.6 Transactions 3

2.1.7 Virement compte à compte 3

2.1.8 Dettes 3

2.1.9 Vue globale 3

2.1.10 Budget 4

2.1.11 Budgets partagés 5

2.2 Fonctionnalités optionnelles 5

2.2.1 Vue globale 5

2.2.2 Liste de souhaits 6

2.2.3 Prévisions d’achat 6

2.2.4 Dettes 6

2.2.5 Simulation 6

3 Mockups 6

4 Architecture 6

5 Implémentation 6

# Descriptif

Avec MoneyThoring, nous voulons proposer à l’utilisateur une gestion de ses transactions, que ce soit de simples dépenses ou revenus, des virements entre comptes, des dettes ou encore des factures. Nous voulons aussi lui apporter la possibilité de catégoriser ces transactions, de créer des objectifs de budget pour pouvoir suivre les dépenses dans des domaines précis et ainsi d’avoir une vue la plus informative possible sur l’évolution de son capital.

Pour ce faire, MoneyThoring sera une application proposant à l’utilisateur de créer un compte (fonctionnalité en ligne) ou de travailler uniquement en local (hors ligne). Selon ce choix, certaines fonctionnalités seront ou non accessibles. Bien sûr, même si l’utilisateur possède un compte, s’il venait à manquer d’une connexion internet, il pourrait sans autre continuer son travail hors ligne. Ce dernier étant synchronisé une fois la connexion avec la base de données en ligne rétablie.

Certaines fonctionnalités ne seront disponibles qu’avec la création d’un compte utilisateur et d’une connexion à internet. Ces fonctionnalités concernent effectivement plusieurs utilisateurs et ne peuvent donc pas être gérée localement. Exception faite de ces dernières, tout sera réalisable en local, avec ou sans compte utilisateur.

MoneyThoring proposera un design poussé pour que l’information soit la plus claire et précise et que l’utilisateur n’ait pas à se creuser la tête pour comprendre son utilisation.

# Fonctionnalités

## Fonctionnalités de base

### Création compte utilisateur

Si l’utilisateur ne possède pas de compte à l’ouverture de l’application, il peut choisir d’en créer un ou de continuer sans. En choisissant la première option, qui n’est accessible que si l’utilisateur à une connexion internet, la création du compte se fait à l’aide d’un formulaire qui lui demande son adresse email, son nom d’utilisateur et un mot de passe, qu’il est nécessaire de confirmer une deuxième fois. Une fois le formulaire envoyé, un email de validation contenant un code est envoyé. Une fois le code saisi dans l’application, le compte est activé.

En choisissant la deuxième option, l’utilisateur peut sans autre utiliser l’application mais n’aura pas accès aux fonctionnalités partagées entre utilisateurs.

### Connexion sécurisée

Pour les utilisateurs possédant un compte validé, une connexion est exigée au démarrage de l’application. Cette connexion évite qu’un tiers puisse modifier les informations, même en travaillant en hors ligne (les données étant synchronisées par la suite). Cette demande de connexion est un simple formulaire dans lequel l’utilisateur doit entrer son nom d’utilisateur et son mot de passe. L’application ne nécessitera pas de double authentification.

Le mot de passe sera haché, en y ajoutant du sel, dans la base de données. Il ne sera pas sauvé en clair.

### Compte bancaire

Un utilisateur pourra ajouter un ou plusieurs comptes bancaires. Chaque compte possèdera un nom, un type (courant, épargne), le nom de la banque qui le concerne (optionnel), le montant actuel, un pourcentage d’intérêt éventuel et s’il faut l’utiliser comme compte par défaut lors de transactions. Toutes ces informations, exceptée le solde du compte qui est modifié automatiquement par les transactions, pourront être modifiée par la suite.

Les comptes bancaires pourront également être supprimés. L’utilisateur pourra alors choisir de virer le solde vers un autre compte ou de perdre ce dernier.

### Catégorie

Il sera possible de créer d’autres catégories que celles proposées par défaut. Une catégorie est définie par un nom et une couleur. Toutes les catégories, même celles proposées par défaut, peuvent être modifiées et supprimées.

### Devise

Chaque utilisateur doit, à la création d’un compte, spécifier sa devise principale, qui sera utilisée pour toutes ses transactions. Une conversion automatique se fera lorsque les transactions sont effectuées dans une autre devise, en fonction du taux du jour.

### Transactions

Les transactions regroupent toutes les entrées et sorties d’argent.

L’utilisateur pourra ajouter des revenus ou des dépenses. Chaque transaction est définie par un montant, une catégorie, un compte affecté (par défaut le compte sélectionné par l’utilisateur lors de sa création), une devise utilisée, le taux de change de la journée et son type (revenu ou dépense).

Les transactions pourront également avoir une notion de récurrence (salaire et factures). Celle-ci pourra être annuelle, mensuelle ou plus spécifique (2 semaines, 2 mois, etc.) et s’exécuter à une date précise, par exemple tous les 25 du mois.

Les transactions pourront être supprimées ou modifiées dans le cas d’erreurs.

### Virement compte à compte

Il sera possible d’enregistrer des virements entre les différents comptes d’un utilisateur. Un virement sera pris en compte comme une transaction. Il y aura donc une dépense (transaction sortante) pour le compte à débiter et un revenu (transaction entrante) pour le compte à créditer.

### Dettes

Possibilité de calculer un éventuel intérêt lié à la dette, ou majorer le montant d’une sommation en cas de retard.

Dettes simples (ne concerne que le compte et est traitée entièrement manuellement) ou synchronisées (concerne un autre utilisateur qui devra confirmer les informations)

Simple / Sync : Comme revenu / dépense, avec possibilité de mettre une personne concernée et une description. Comporte, comme les factures, une deadline à respecter (notification à l’approche de cette dernière)

Sync : Concerne un compte existant (ou non, auquel cas spécifier un email) qui doit valider que la dette existe bien (qu’elle soit entrante ou sortante) entre les deux entités.

Une fois la dette acquittée, les deux partis doivent confirmer avant que la dette soit “classée”. Bien sûr, dans le cas où le deuxième parti ne répond pas (ou qu’il n’a pas de compte existant), mettre un délai pour invalider la chose. Une fois le délai écoulé, la dette est de toute façon classée.

### Vue globale

L’utilisateur peut en tout temps suivre l’évolution de son budget, grâce à une vue globale (depuis le début du budget jusqu’à maintenant), annuelle ou mensuelle. Cette vue permet de comparer de manière graphique les différentes dépenses et rentrées d’argent. En plus d’être filtrée chronologiquement, elle peut également l’être selon le mode d’affichage: graphique en courbe, diagramme circulaire, diagramme en barres.

Graphique en courbe

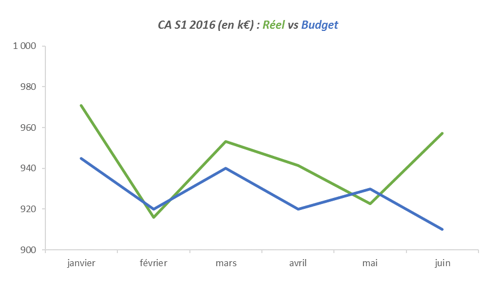
Le graphique en courbe permet la comparaison entre la courbe des dépenses et la courbe des rentrées d’argent selon la chronologie sélectionnée.    

Diagramme circulaire

Le diagramme circulaire affiche les pourcentages de dépenses par catégorie. Une information supplémentaire est toutefois ajoutée à côté de ce graphique, pour signaler le pourcentage de dépenses et rentrées d’argent globales pour la chronologie sélectionnée.



### Budget

Un budget se caractérise en deux types : les budgets mensuels (en fonction des catégories) et les budgets ponctuels (en fonction de deux dates données début/fin). Ces budgets permettent de voir l’évolution des dépenses dans un domaine précis, tout en permettant à l’utilisateur d’être notifié quand il s’approche de la limite fixée. Les budgets seront représentés de manière graphique sur une ligne, avec le budget qu’il lui reste (vert par exemple) et les dépenses déjà effectuées (rouge). L’utilisateur aura également la possibilité de créer des budgets partagés, où les dépenses de chaque utilisateur sont donc communes et le budget évolue en fonction des participants (par exemple un budget de voyage ou de commissions pour la famille).

Ajout d’un budget avec un type, un nom, un montant. Lors de l’ajout du montant, l’utilisateur peut choisir la devise qu’il souhaite mettre.

Si le budget est récurrent (mensuel) ou ponctuel, il faut sélectionner une ou plusieurs catégories, mais s’il est ponctuel, une durée est demandée (date de début et de fin).

Il est possible de supprimer n’importe quel budget. Dans le cas d’un budget partagé, il peut l’être uniquement par le créateur, ce qui affecte tous les membres y participant.

L’utilisateur peut modifier le montant, la devise et la durée du budget et les catégories d’un budget.

### Budgets partagés

Si le budget est partagé, l’utilisateur peut inviter d’autres utilisateurs (via leur nom de compte) à rejoindre un budget. Les personnes concernées recevront une notification pour accepter l’invitation.

Pour les budget partagés, l’utilisateur peut également décider de sortir du budget (sans le supprimer). Dans ce cas-là les dépenses effectuées par l’utilisateur restent enregistrées dans le budget.

Lors de l’ajout d’une dépense, l’utilisateur peut indiquer qu’il souhaite se faire rembourser par certains utilisateurs. Ceux-ci recevraient donc une dette envers celui qui a payé pour les autres.

## Fonctionnalités optionnelles

### Vue globale

Diagramme en barres

Le diagramme en barres permet la comparaison entre les dépenses et rentrées d’argent par mois et par année sous forme de deux barres de couleurs différentes.



Chaque vue est exportable au format PDF.

### Liste de souhaits

L’utilisateur peut créer une liste de souhaits, composée de différents articles qu’il aimerait prochainement acheter. Ces articles sont identifiés par un nom, éventuellement un lien internet et un prix.

Cette liste est mise à jour en fonction du budget actuel. Si l’épargne de l’utilisateur est suffisante, alors le produit est catégorisé comme étant achetable. Si au contraire l’utilisateur n’a pas suffisamment de fonds disponibles, une estimation de temps d’attente avant achat possible est calculée sur la base de l’évolution du budget.

### Prévisions d’achat

Liste de souhaits “prioritaire”, qui correspondrait à “il faudrait (absolument) que j’achète l’article X avant la date Y”, X étant un objet (p.ex un cadeau, un outil, une fourniture, …).

La dépense serait effective une fois que l’utilisateur aurait confirmé avoir effectué cet achat.

L’application envoie un mail la veille de la date limite, pour rappeler à l’utilisateur de faire son achat, ou pour confirmer la dépense s’il l’a déjà effectuée.

### Dettes

Possibilité de scanner les dettes et les documents en lien avec la dette et les lier à celle-ci. Fichier à titre informatif, on ne le lit pas ou autre.

### Simulation

Une simulation est possible en se basant sur les revenus et les dépenses récurrentes (salaire, abonnements, factures, loyer, capitalisation du compte, ...)

Peut-être également prendre en compte les dépenses qui réapparaissent plus ou moins régulièrement sans pour autant être récurrent (notamment les courses qui ne sont pas forcément à un rythme régulier) ?

Un utilisateur peut modifier ses dépenses et revenus en mode simulation, s’il souhaite voir comment évolue son budget sans pour autant sauvegarder les modifications réalisées. L’activation de ce mode est repérée par des indications visuelles, permettant à l’utilisateur de ne pas se tromper de mode.

# Mockups

# Architecture



L’architecture logicielle est conçue pour fonctionner sur le système d’exploitation Windows 10. Le logiciel est découpé en trois couches, chaque couche a une responsabilité précise :

* La couche présentation en bleu permet de présenter les informations à l’utilisateur dans des fenêtres graphiques.
* La couche logique métier en magenta gère toute la logique métier de l’application.
* La couche d’accès aux données en rouge va permettre de gérer la persistance et l’accès aux données.

# Implémentation

Java + JavaFX

## Hibernate

Hibernate est un ORM (Object Relational Mapping) qui permet de développer des applications, qui peuvent aisément gérer et accéder à des bases de données, récupérer, modifier et supprimer des données. Hibernate sera utilisé pour réaliser la couche d’accès aux données internes (SQLite) et la couche d’accès aux données externes (PostgreSQL). Ces couches d’accès aux données sont responsables de la persistance dans un système de gestion de base de données relationnel.

L’utilisation d’Hibernate nous permettra d’écrire du code plus facilement maintenable et compréhensible, de gérer les mises à jour et les changements sur plusieurs relations. Avec Hibernate, concevoir une couche capable de gérer la persistance des données est moins problématique et fastidieux que de le faire entièrement à la main.