|  |  |
| --- | --- |
|  | **Programmation Web**  Documentation template |

**1. Target user profiles**

*First profile :*

* *Name: Thibaut*
* *Age: 22*
* *Commonly used applications: Facebook, Messenger,Telegram, Instagram, Snapchat.*
* *Back story: A futur ingenior from Enseirb-Matmeca in Bordeaux have a weeks of holydays. He organizes a trip with a group of friends and they need to share their expenses. In fact one person is in charge to buy food and other drinks. They need to share toll and petrol charges. And they have to split rental of accommodation*
* *Overview:*   
  *- loves maths and computer science,*  
  *- he loves skiing and doing parties with his friends without worrying about all the organization,*  
  *- is looking for an app that will help them separate tasks and to share all expenses with precision and simplicity*

*Second profile :*

* *Name: Jennifer*
* *Age: 18*
* *Commonly used applications: TikTok, Messenger, Instagram, Snapchat.*
* *Back story: A student that preparing her A level in literature, her best firend called Emma, who will have her birthday in two weeks. She wants to organize a surprise party for the occasion, after invinting all Emma's friends, she would like to buy a present with all the participants and divided cost of alcohol only with people who are drinking it. Moreover she chooses the best present very expensive for her best friend !*
* *Overview:*   
  *- reading books and doing a lot of TIKTOK,*  
  - she hates maths,  
  *- she wants to do the best party for her friend Emma,*  
  *- she is looking for an app that will help her to calculate how much each person have to give her with simplicity*

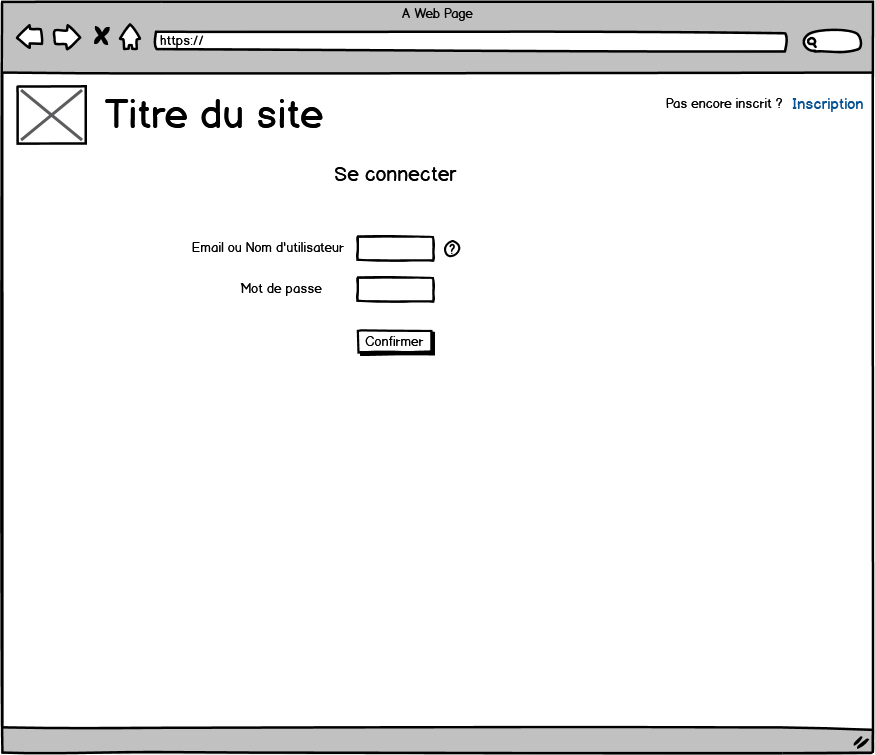
**2. Sitemap**



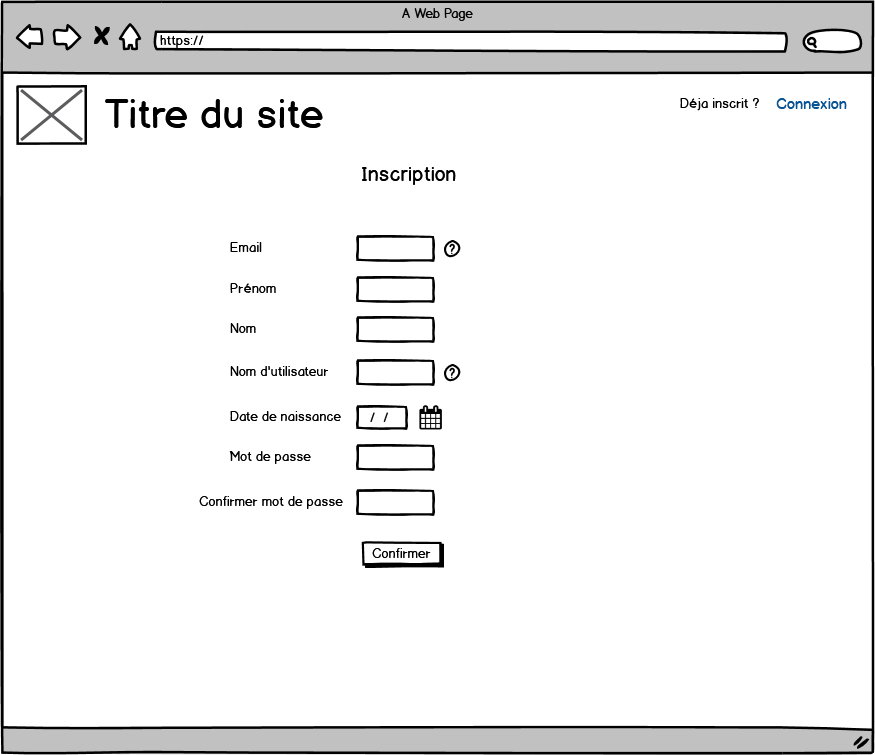
**3. Wireframes**

Les wireframes sont celles créés au début du projet. Le rendu visuel final est différent, en fonction des contraintes d’implémentations.

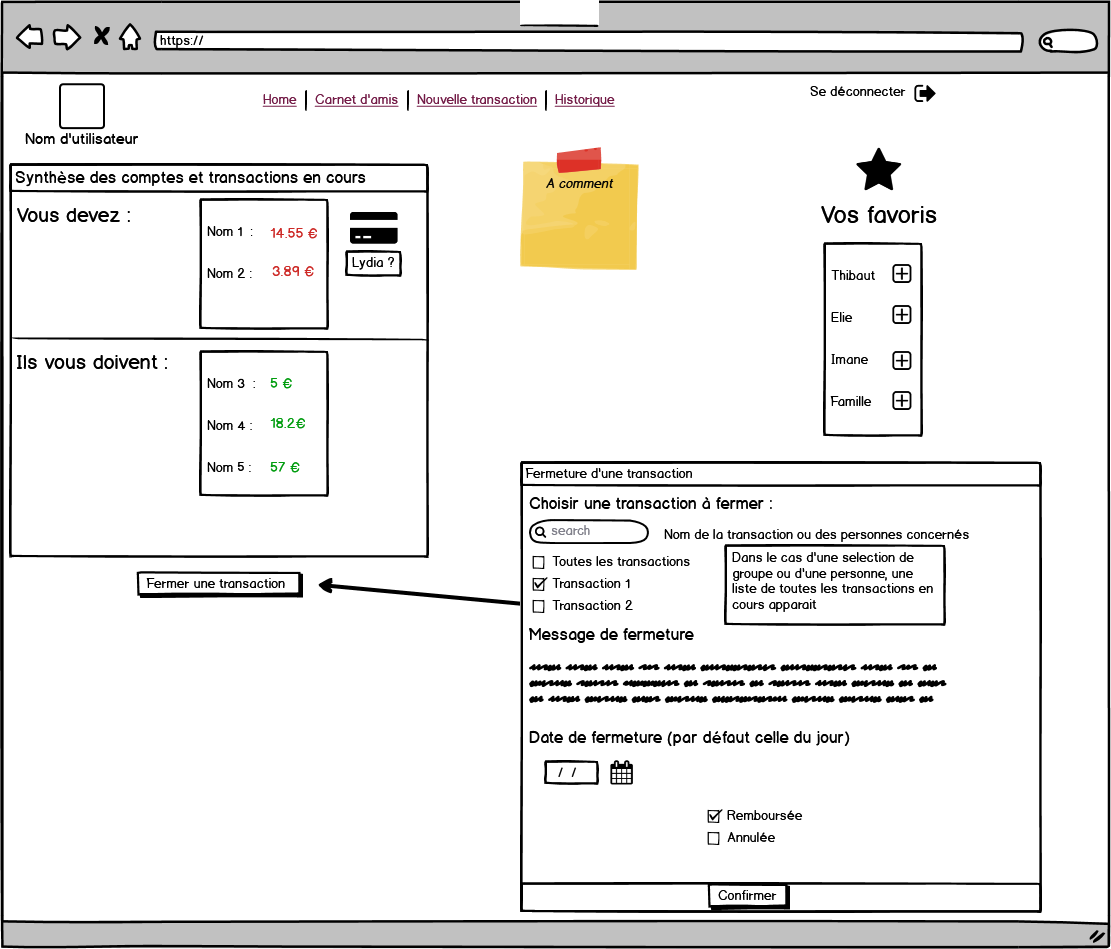
[Frame 1]



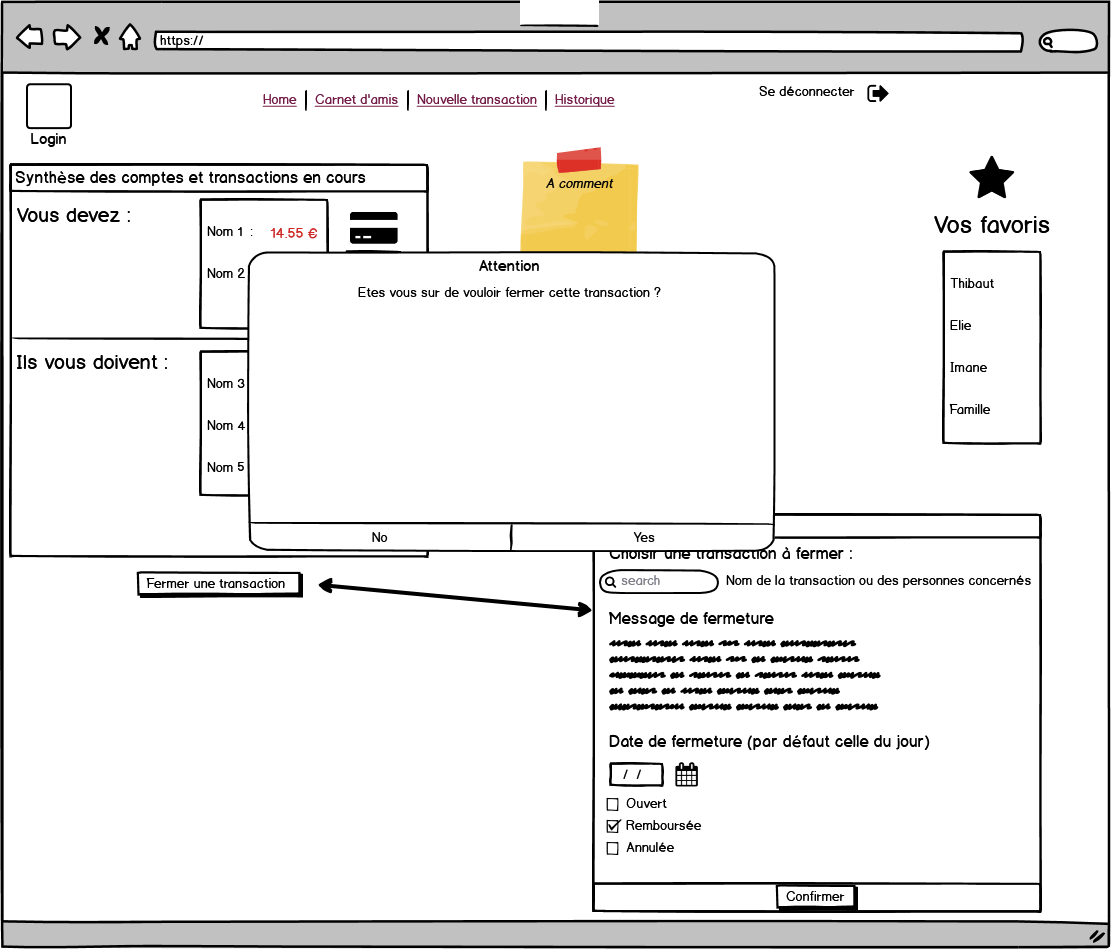
[Frame 2]



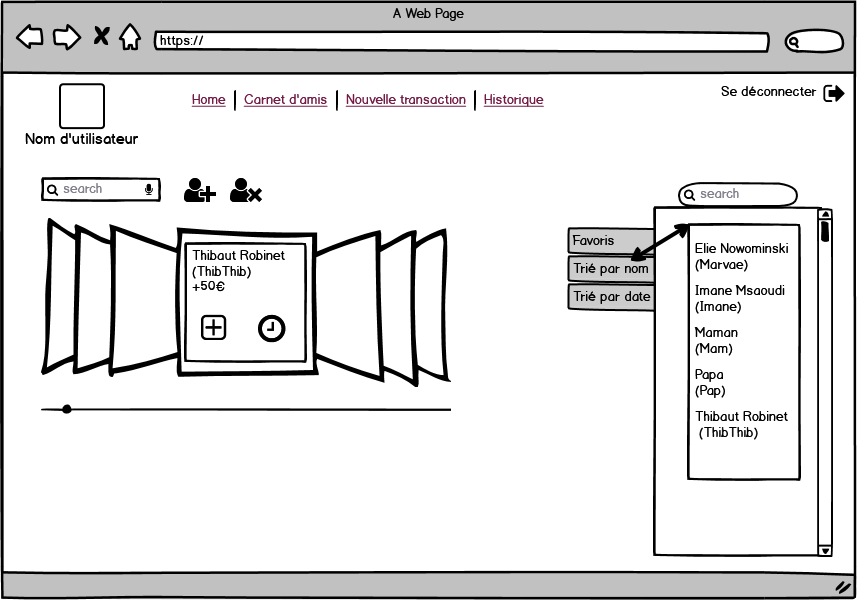
[Frame 3]



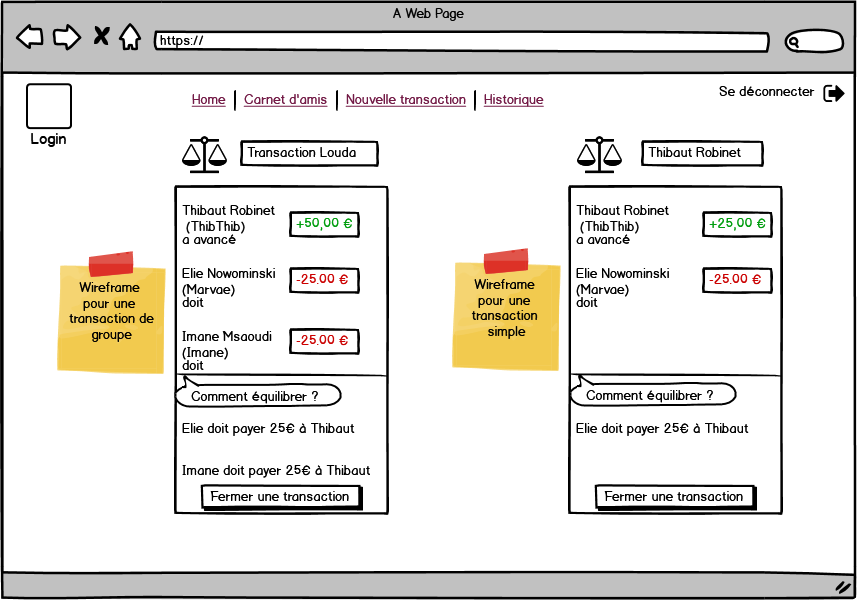
[Frame 4]



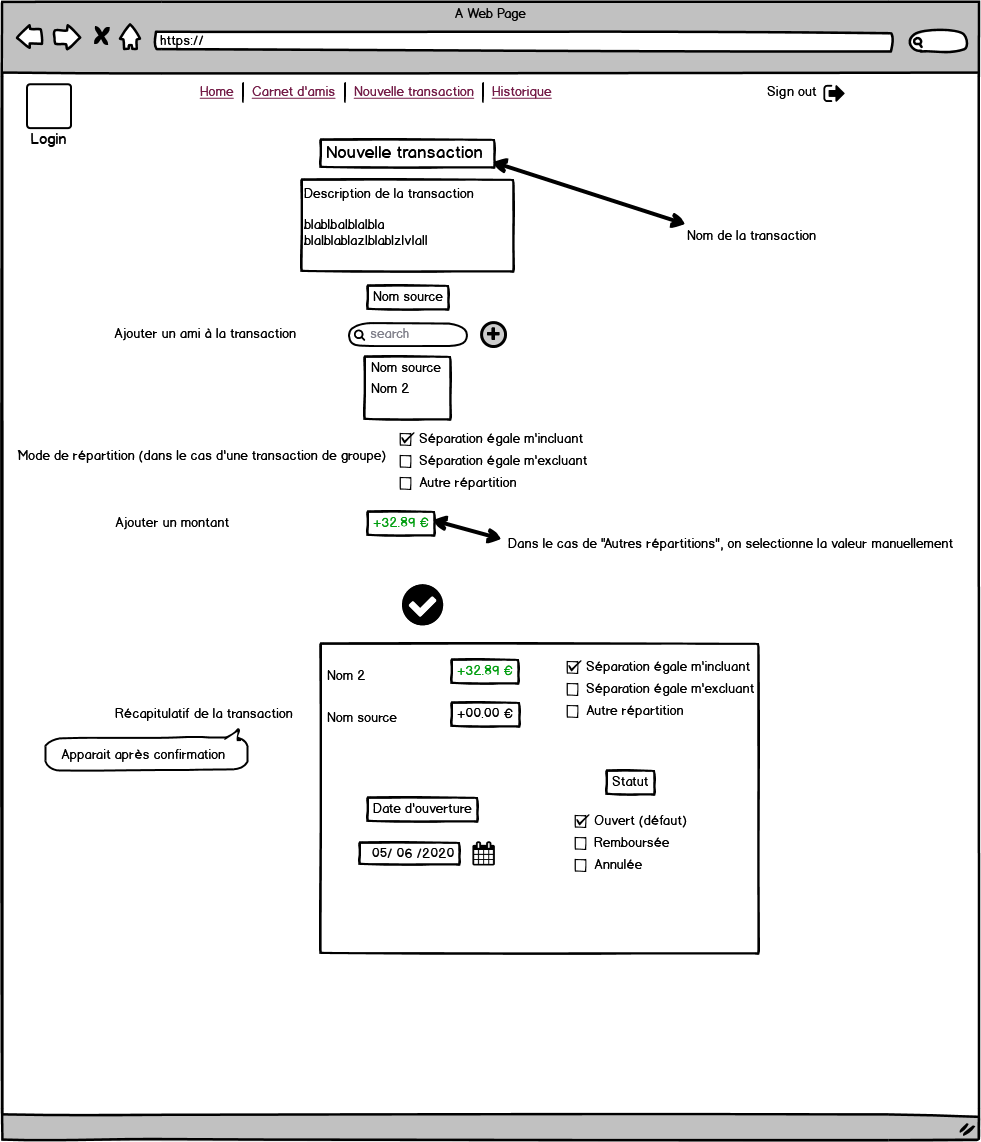
[Frame 5]



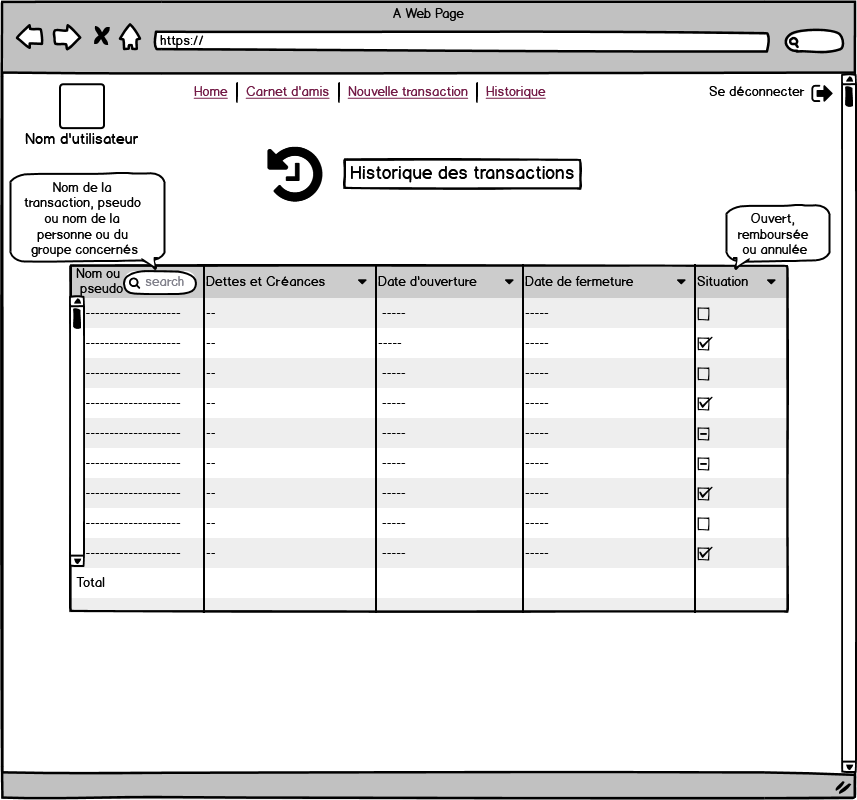
[Frame 6]



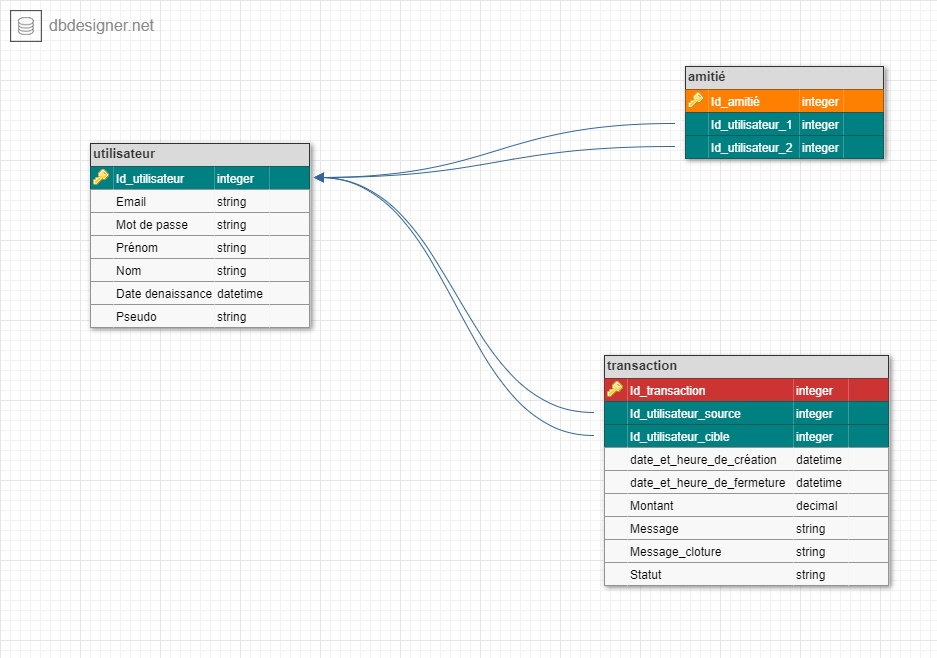
[Frame 7]



[Frame 8]



**4. Database model**



**5. System structure - scripts overview**

For every php script created, please specify the following elements:

* php script name and corresponding Wireframe, if applicable,
* a couple of sentences describing the functionality of a given script ,
* HTTP method call and its parameters,
* type of interaction with the database (Insert/Delete/Update), if applicable,
* dependencies with other scripts, if applicable.

**Inscription :**

-inscription.php : wireframe 2.

- Formulaire d’inscription avec les informations demandées.

- Méthode POST avec comme paramètres : email, Nom d’utilisateur, Prénom, Nom, date de naissance, mot de passe.

- Insert dans la table “Utilisateur ”.

- Renvoie vers connexion\_page.php.

**Connexion :**

-connexion\_page.php : wireframe 1.

- Page de connexion qui permet à un utilisateur déjà enregistré de se connecter.

- Méthode POST avec comme paramètres : email et mot de passe.

- Renvoie vers home\_page.php.

**Déconnexion :**

-déconnexion.php : Toutes les wireframes qui possèdent le bouton « Se déconnecter ».

- Méthode GET avec comme paramètre un « on click » sur le bouton « Se déconnecter ».

- Renvoie vers connexion\_page.php.

**Consultation du carnet des amis :**

- contact\_page.php: wireframe 5.

- Méthode GET avec comme paramètre un « on click » sur le nom d’une personne.

- Afficher le nom et le pseudo de la personne sur la wireframe 5.

**Ajouter un ami :**

-contact\_page.php : wireframe 5.

- Barre de recherche où l’on peut saisir le pseudo ou l’e-mail d’une personne.

- Méthode POST avec comme paramètre le pseudo ou bien l’adresse e-mail.

- Insert dans la table « amitié ».

**Supprimer un ami :**

-contact\_page.php : wireframe 5.

-Méthode GET avec comme paramètre un « on click » sur l’icône de suppression d’un ami.

-Delete de la table « amitié ».

**Consulter les transactions avec un ami :**

- friend\_transaction.php : Wireframe 5.

- Méthode GET avec comme paramètre un « on click » sur le nom de la personne ou du groupe.

- Amène à la wireframe 6.

**Consulter l’historique des transactions avec un ami :**

- friend\_transaction.php : Wireframe 5.

- Méthode GET avec comme paramètre un « on click » sur l’icône d’historique.

- Amène la wireframe 8 déjà chargé sur le nom de l’ami.

**Modifier une transaction :**

- historique\_page.php : Wireframe 8.

- Méthode POST avec comme paramètre le nom ou le pseudo de la personne ou bien le nom de la transaction et avec le statut de la transaction.

- Possibilité de modifier seulement les champs du montant et du message.

- Modifie les champs montant et message d’un objet de la table  « transaction ».

**Ajouter une transaction :**

- create\_transaction\_page.php : Wireframe 3 et 5.

- Méthode avec comme paramètre un « on click » sur le lien « nouvelle transaction ».

- Amène à la Wireframe 7.

- Méthode POST avec comme paramètre la description, le pseudo d’un ami, le montant.

- Ajoute un nouvel objet à la table « transaction ».

**Ajouter une transaction de groupe :**

- create\_group\_transaction.php : Wireframe 3 à 5.

- Méthode GET avec comme paramètre un « on click » sur le lien « nouvelle transaction ».

- Amène à la Wireframe 7.

- Méthode POST avec comme paramètre la description, les pseudos des amis, le montant, le mode de répartition.

- Création des sous transactions individuelles dans la table « transaction »

**Fermer une transaction :**

- Close\_transaction.php : wireframes 3 et 6 .

- Méthode GET avec comme paramètre un « on click » sur le lien « Fermer une transaction ».

- Méthode GET avec comme paramètre le nom ou le pseudo de la personne ou bien le nom de la transaction.

- Amène à la fenêtre de fermeture de la Wireframe 4

- Méthode POST avec comme paramètre le texte saisi par l’utilisateur.

- Méthode GET avec comme paramètre un « on click » sur le bouton confirmer.

- Delete de la table « transaction ».

**Consultation de l’encours :**

- home\_page.php: wireframe 3.

- Méthode GET pour récupérer les montants des dettes et des créances de l’utilisateur dans la table: « transaction ».

- Sommer les créances et soustraire les dettes.

- Afficher le résultat final : « Total » dans la wireframe 8.

**Fermer des transactions par remboursement d’un ami :**

- Close\_transaction : wireframe 3 et 6.

- Méthode GET avec comme paramètre un « on click » sur le lien « Fermer une transaction ».

- Méthode POST avec comme paramètre le nom ou le pseudo de la personne.

- Méthode POST avec comme paramètre le montant et le message saisi par l’utilisateur.

- Méthode GET avec comme paramètre un « on click » sur le bouton confirmer.

- Modifie un maximum de transaction de la table « transaction » en mettant le statut remboursé.

- Peut créer une nouvelle transaction d’ajustement dans la table « transaction » si le montant de remboursement diffère du montant attendu.

**Consultation la liste des dettes et des créances :**

- historique\_page.php: wireframe 8.

- Option de tri créance/dette

- Méthode GET pour récupérer les montants des dettes et des créances de l’utilisateur dans la table : «transaction ».

- Afficher les dettes dans un tableau et les créances dans un autre tableau.

- Les transactions sont triées par rapport au champ montant.

**6. Test strategy**

Test strategy presents the approach to testing taken on a given project. It describes the scope of the tests, the methods used and specification of test cases.

**6A. Test items**

Inscription

Authentification et Déconnexion

Ajout et suppression d’amis

Carnet d’amis

Ajouts de transactions

Consultation des transactions

Fermeture de transactions

Consultation du solde et de l’historique

**6B. Test cases**

There exist different types oftests - functional, integration, etc. - you are free to formalize the test cases that you find appropriate to ensure the right level of quality of the project.

The design of the **functional testing** will focus on the adherence of the system to **user requirements**. Integration testing assures that the associated modules communicate correctly, hence tests of inputs and outputs of the system need to be carried out to verify that they conform to the system specifications.

Each test is executed and the result is either a pass or fail.

Test case ID: #1

Test case Name : Enregistrement dans le système.   
Test case Date :   
Test case Purpose: S’assurer que l’on puisse s’enregistrer en entrant les données nécessaires. Test case Function: Vérifier que les données sont valides .   
Test case Input : Un clic sur confirm.   
Test case expected Output : Pouvoir s’authentifier.

Test case ID: #2  
Test case Name : Authentification   
Test case Date :   
Test case Purpose: S’assurer que l’on puisse s’authentifier.  
Test case Function: Vérifier que le pseudo ou l’e-mail ainsi que le mot de passe sont valides.   
Test case Input : Entrer l’email et le mot de passe puis cliquer sur confirm.   
Test case expected Output : Accéder à la page d’accueil.

Test case ID: #3  
Test case Name : Durée d’authentification   
Test case Date :   
Test case Purpose: S’assurer que l’on reste connecté pendant 1h.  
Test case Function: Sauvegarder l’authentification pendant un laps de temps (1h).  
Test case Input : Quitter la page après s’être authentifié et revenir dessus.  
Test case expected Output : Être encore authentifié si on s’est connecté il y’a moins d’une heure.

Test case ID: #4  
Test case Name : Déconnexion.  
Test case Date :  
Test case Purpose: S’assurer que l’on puisse se déconnecter de l’application.  
Test case Function:.Sortir de la page d’accueil.  
Test case Input : Un clic sur sign out.  
Test case expected Output : Retour à la page d’authentification.

Test case ID: #5  
Test case Name : Ajout d’un ami.   
Test case Date :   
Test case Purpose: S’assurer que l’on puisse ajouter un ami .  
Test case Function: Vérifier que l’on puisse ajouter un ami de la base de donnée.  
Test case Input : Un clic sur l’icône d’ajout.   
Test case expected Output : Voir que l’ami ajouté s’affiche dans le carnet.

Test case ID: #6  
Test case Name : Suppression d’un ami.   
Test case Date :   
Test case Purpose: S’assurer que l’on puisse supprimer un ami .  
Test case Function: Vérifier que l’on puisse supprimer un ami de la base de donnée seulement si son solde est nul.   
Test case Input : Un clic sur l’icône de suppression.   
Test case expected Output : Voir que l’ami supprimé ne s’affiche plus dans le carnet

Test case ID: #7

Test case Name: Consulter les amis.

Test case Date:

Test case Purpose: S’assurer que l’on puisse consulter ses amis

Test case Function: Vérifier que l’on peut récupérer les amis depuis la base de données et afficher le résultat.

Test case Input: Cliquer sur le bouton « Carnet d’amis »

Test case expected Output: être capable voir ses amis à l’écran avec leurs nom, leurs prénom et le solde entre vous et votre ami.

Test case ID: #8

Test case Name: Saisir une transaction simple.

Test case Date:

Test case Purpose: S’assurer que l’on puisse saisir une transaction simple.

Test case Function: Vérifier que l’on créer bien une transaction avec un amis dans la base de donnée.  
Test case Input: Cliquer sur le bouton « Nouvelle transaction» puis saisir les champs demandés.

Test case expected Output: Être capable d’accéder à cette transaction créée.

Test case ID: #9

Test case Name: Saisir une transaction de groupe.

Test case Date:

Test case Purpose: S’assurer que l’on puisse saisir une transaction de groupe.

Test case Function: Vérifier que l’on créer bien plusieurs transactions simple avec des amis dans la base de donnée.  
Test case Input: Cliquer sur le bouton « Nouvelle transaction» puis saisir les champs demandés.

Test case expected Output: Être capable d’accéder à ces transactions créées.

Test case ID: #10  
Test case Name : Consultation du des transactions qui constituent le solde avec un amis  
Test case Date :   
Test case Purpose: S’assurer qu’il n’y a pas de transactions manquantes.  
Test case Function: Vérifier que toutes les transactions sont présentes avec le bon montant.  
Test case Input : Cliquer sur le bouton « historique » à côté du nom d’un ami.  
Test case expected Output : Accéder à toutes les transactions en cours avec un amis.

Test case ID: #11  
Test case Name : Consultation des transactions en cours.  
Test case Purpose: Affichage des transactions en cours.  
Test case Date :  
Test case Function: Vérifier que les transactions s’affichent dans l’ordre décroissant de  
la date de saisie et que les transactions de groupe doivent apparaître en italique.  
Test case Input : Affichage par défaut.  
Test case expected Output : Affichage des transactions après un accès à la page  
d’accueil.

Test case ID: #12

Test case Name: Fermer une transaction avec un ami.

Test case Date:

Test case Purpose: S’assurer que l’on puisse fermer une par remboursement ou annulation.

Test case Function: Vérifier que l’on peut fermer une transaction récupérée depuis la base de donnée.

Test case Input: Sélectionner un ami depuis le carnet d’amis, cliquer sur le bouton « fermer une transaction », mettre le message de fermeture et sélectionner la raison.

Test case expected Output:

La transaction se ferme suite au remboursement ou à l’annulation du montant.

Le solde avec l’ami doit être mis à jour correctement.

Test case ID: #13

Test case Name: Fermer plusieurs transactions avec le même ami.

Test case Date:

Test case Purpose: S’assurer que l’on puisse fermer plusieurs transactions d’un seul coup par remboursement du même ami.

Test case Function: Vérifier que l’on peut fermer plusieurs transactions récupérées depuis la base de données en une seule action.

Test case Input: sélectionner un ami depuis le carnet d’amis, cliquer sur le bouton « fermée plusieurs transactions », rentrer le montant que l’on souhaite rembourser, il rentre un message de cloture, valider

Test case expected Output:

Le maximum de transactions se ferment suite au remboursement du montant.

Si le montant n’est pas juste une nouvelle transaction dit d’équilibrage se crée.  
Toutes les transactions fermées doivent avoir le même message de cloture.

Test case ID: #14

Test case Name: Consulter mes dettes et mes créances

Test case Date:

Test case Purpose: S’assurer que l’on puisse consulter les dettes et les créances

Test case Function: Vérifier que l’on peut récupérer les transaction depuis la base de données et afficher un résultat correcte.

Test case Input: cliquer sur le bouton « historique »

Test case expected Output:

Les dettes et les créances sont dans des tableaux séparés.

Les transactions sont triées dans l’ordre croissant de leur montant.

Le résultat des dettes est affiché et est juste.

Le résultat des créances est affiché et est juste.

**6C. Requirements Traceability Matrix (RTM)**

RTM documents the links between system requirements and test cases that are put in place to ensure an adequate level of testing is achieved. The main purpose of Requirement Traceability Matrix is to see that all requirements have their corresponding test cases so that no functionality is missed out while performing software testing.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **REQUIREMENTS TRACEABILITY MATRIX** | | | | | |
| **Functional Requirement ID#** | **Functional Requirement  / Use Case** | **Test Case ID#** | **Test execution date** | **Execution Status** | **Defect description** |
| FR\_1 | Possibilité de s’inscrire dans le système | TC#001 | 15/05/2020 | Validé |  |
| FR\_2 | En tant qu’utilisateur authentifié je peux me connecter et me déconnecter | TC#002  TC#003 TC#004 | 15/05/2020 | Validé  Validé  Validé |  |
| FR\_3 | Possibilité d’ajouter et supprimer un amis | TC#005 TC#006 | 15/05/2020 | Validé |  |
| FR\_4 | Consulter le carnet d’amis | TC#007 | 15/05/2020 | Validé |  |
| FR\_5 | Création de transaction | TC#008 TC#009 | 15/05/2020 | Validé  Validé |  |
| FR\_6 | Consultations des transactions | TC#010 TC#011 | 15/05/2020 | Validé |  |
| FR\_7 | Possibilité de fermer les transactions | TC#012  TC#013 | 15/05/2020 | Validé | Le bouton “fermer plusieurs transaction” n’est pas accessible depuis le carnet d’amis mais depuis la page home |
| FR\_8 | Consulter mes dettes et créances | TC#014 | 15/05/2020 | Validé | N’a pas lieu sur la page historique mais la page home |