Programmation Web

⌘ Documentation template

DOCUMENT TEMPLATE: DUBNATION

Phase de conception

1. Target user profiles

Premier Profil

- *Nom: Victor*
- *Age: 20*
- *Applications utilisées: Réseaux sociaux (Facebook, Instagram…)*
- *Contexte d'utilisation: étudiant en école supérieur part en vacances à Varsovie avec 5 de ses amis afin de profiter du pays. Une cagnotte commune est réalisée. Il a donc besoin d'une application permettant à lui et ses amis d'être égaux financièrement malgré leurs nombreuses dépenses.*

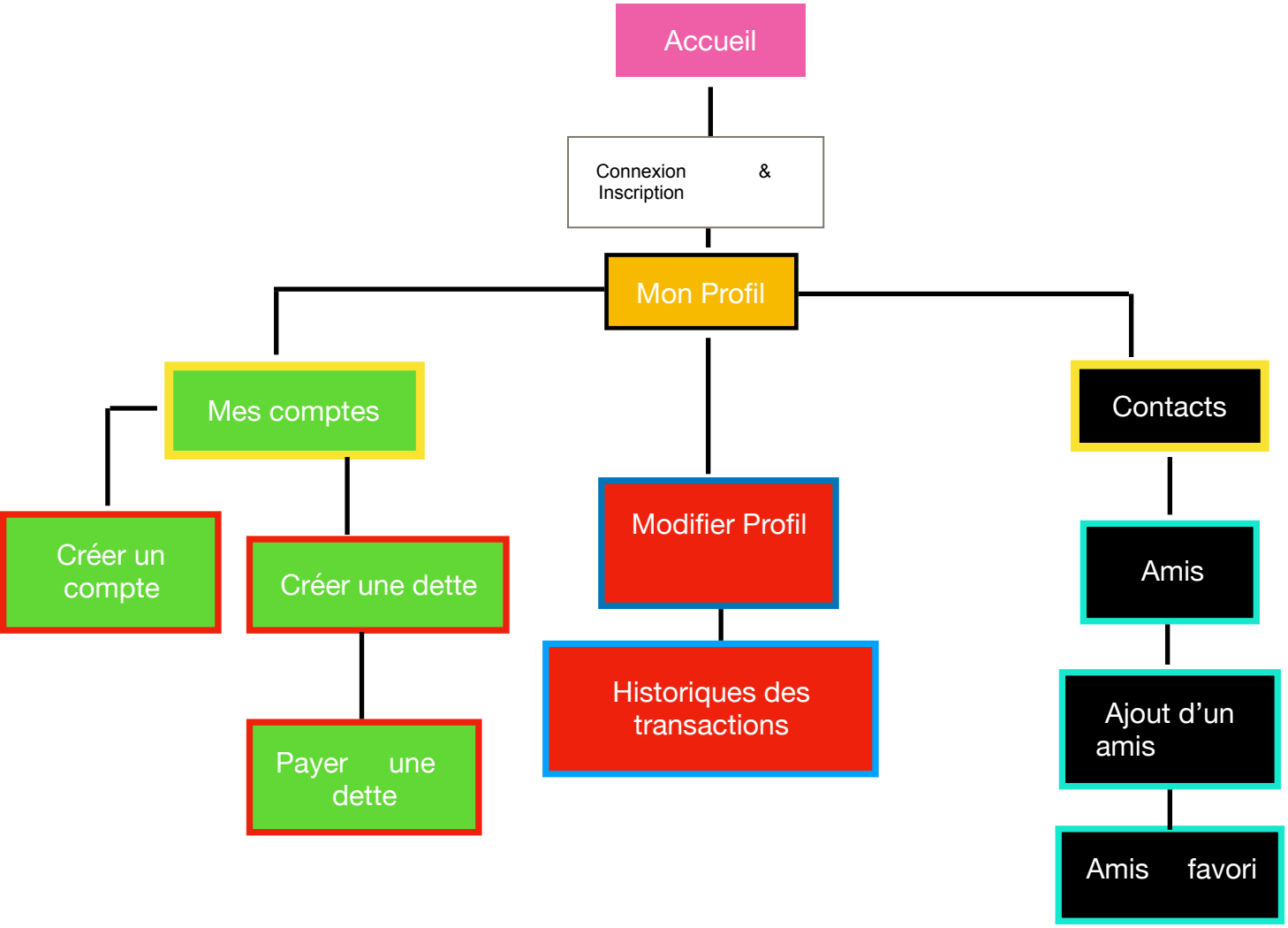
Second Profil

- *Nom: Louis*
- *Age: 33*
- *Applications utilisées: Uber, LinkedIn*
- *Contexte d'utilisation: travailleur dans le centre de Paris, en collocation avec une conjointe. Un budget pour cette collocation est mise en place afin de gérer les dépenses du quotidien parisien; pour les deux personnes.*

Troisième Profil

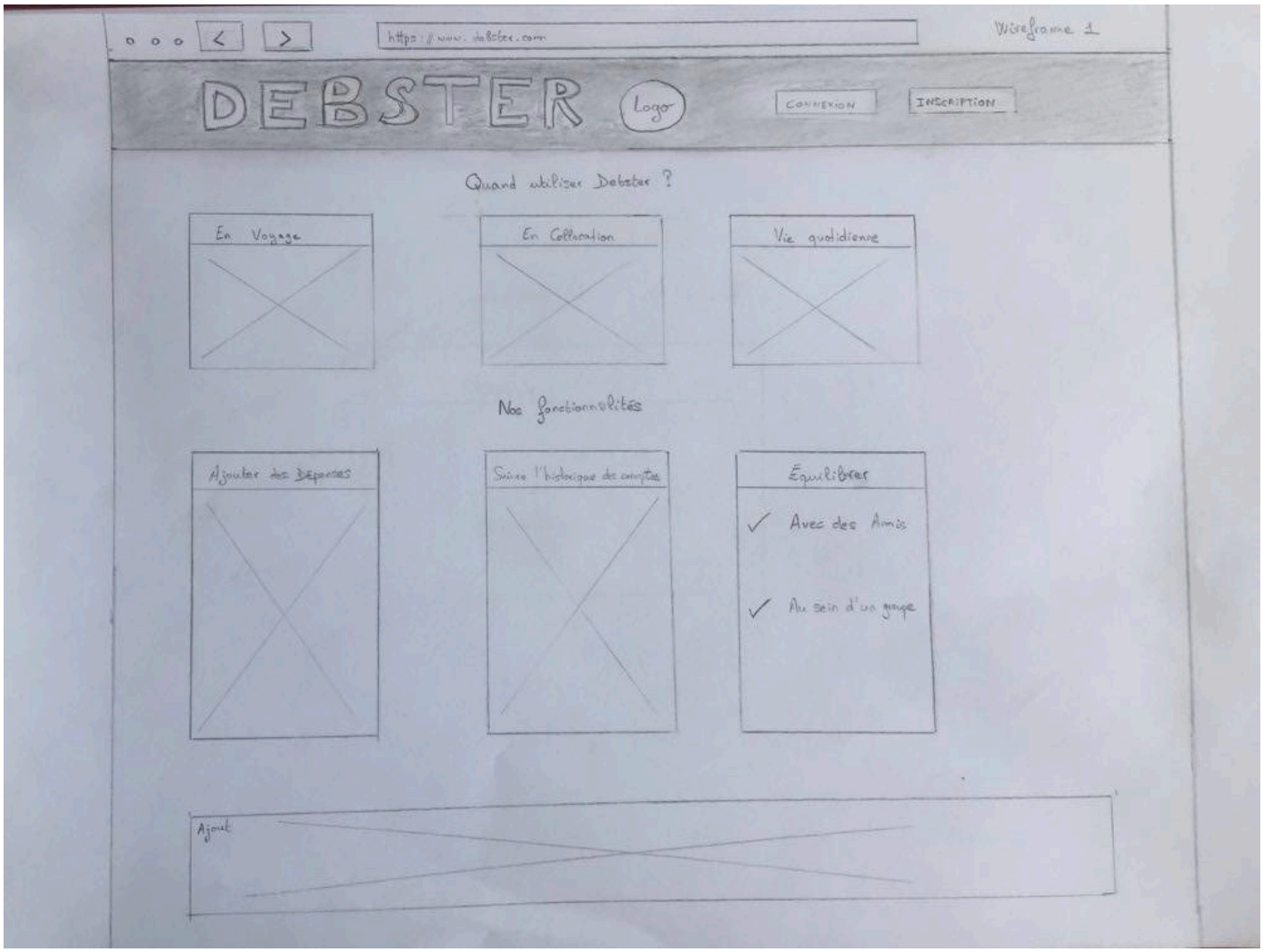
- *Nom: Delphine*
- *Age: 51*
- *Applications utilisées: /*
- *Contexte d'utilisation: Mère de famille seule a besoin d'une application lui permettant de gérer ses propres dépenses ne réalisant pas tout ce qu'elle paye au fil des jours. L'application et le site doivent être ergonomiques n'étant pas habituée à l'usage des nouvelles technologies..*

2. Sitemap

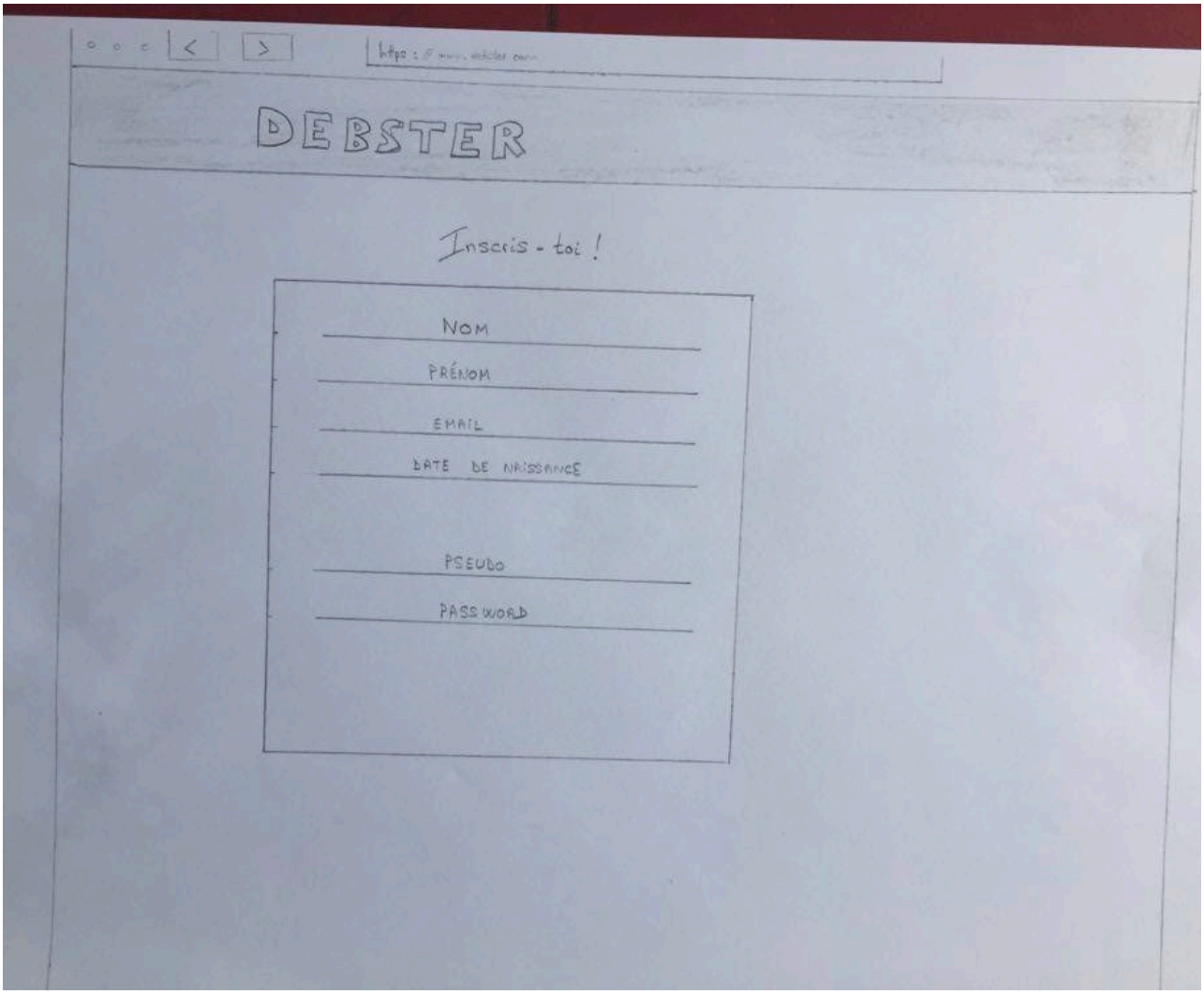


Les transactions se font dans « payer une dette » et sont liées aux amis et aux groupes via les foreign keys présentes dans les tables de données.

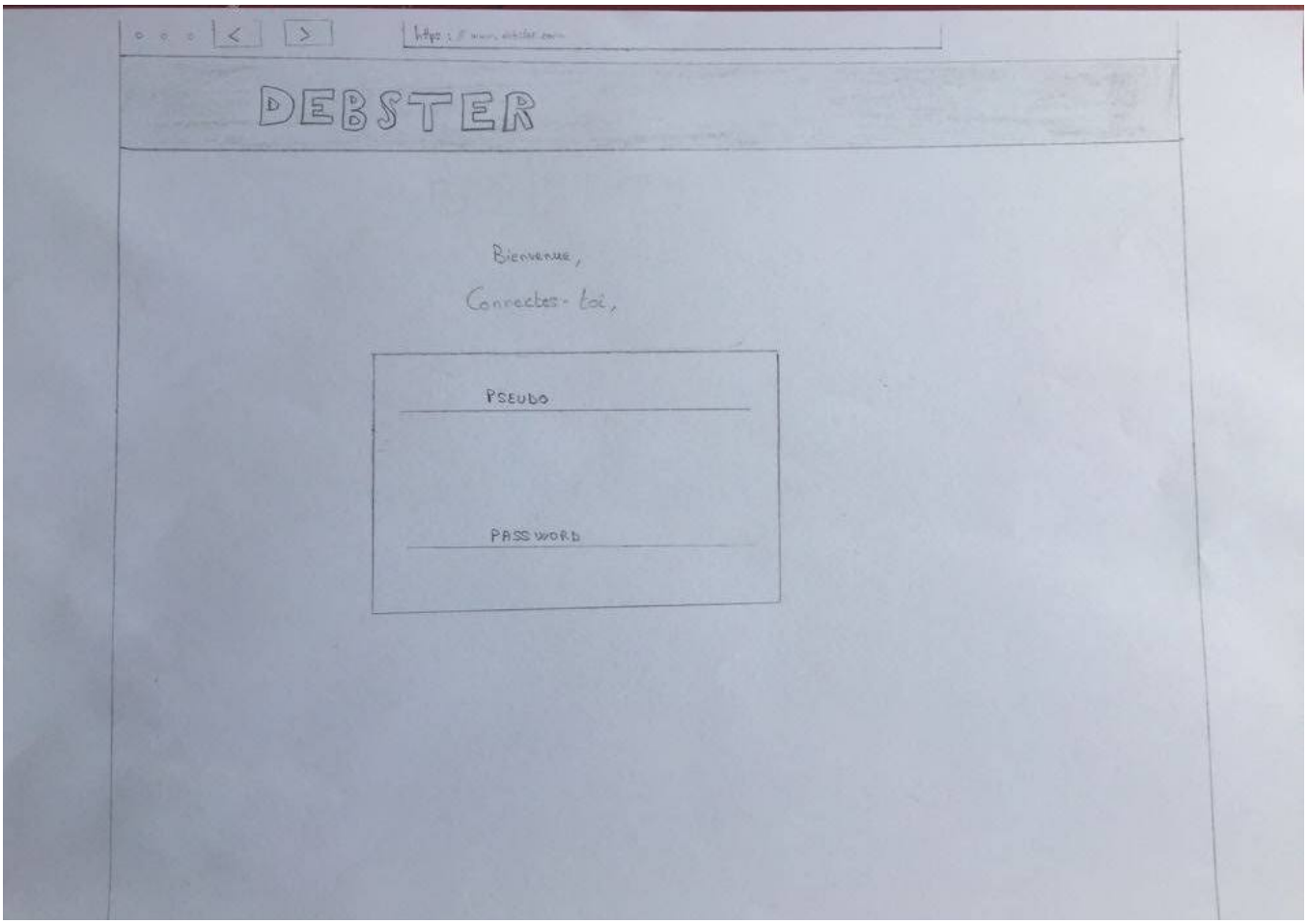
3. Wireframes



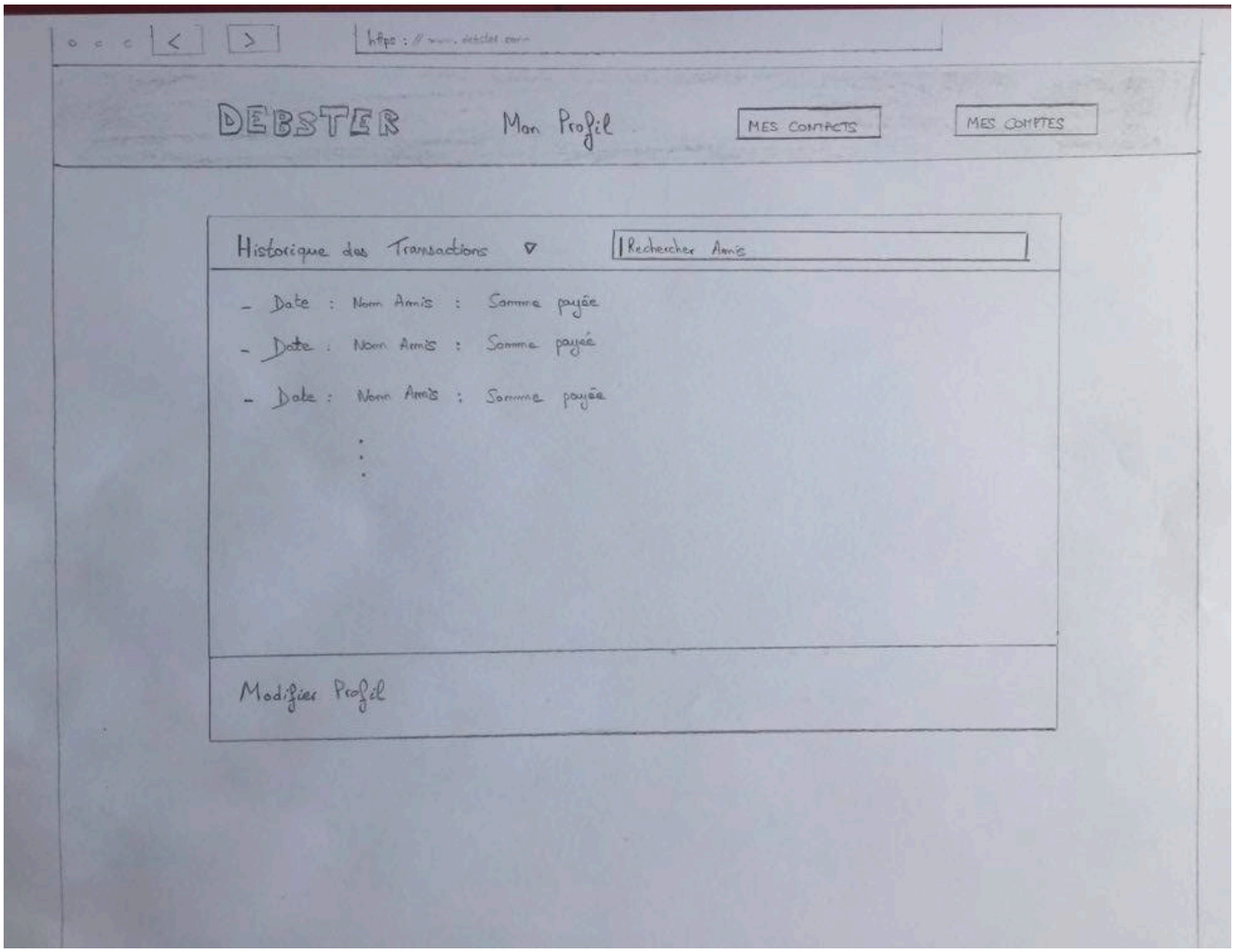
WIREFRAME 1: ACCUEIL, PAGE MAJORITAIREMENT DESCRIPTIVE PERMETTANT DE SE CONNECTER OU DE D'INSCRIRE



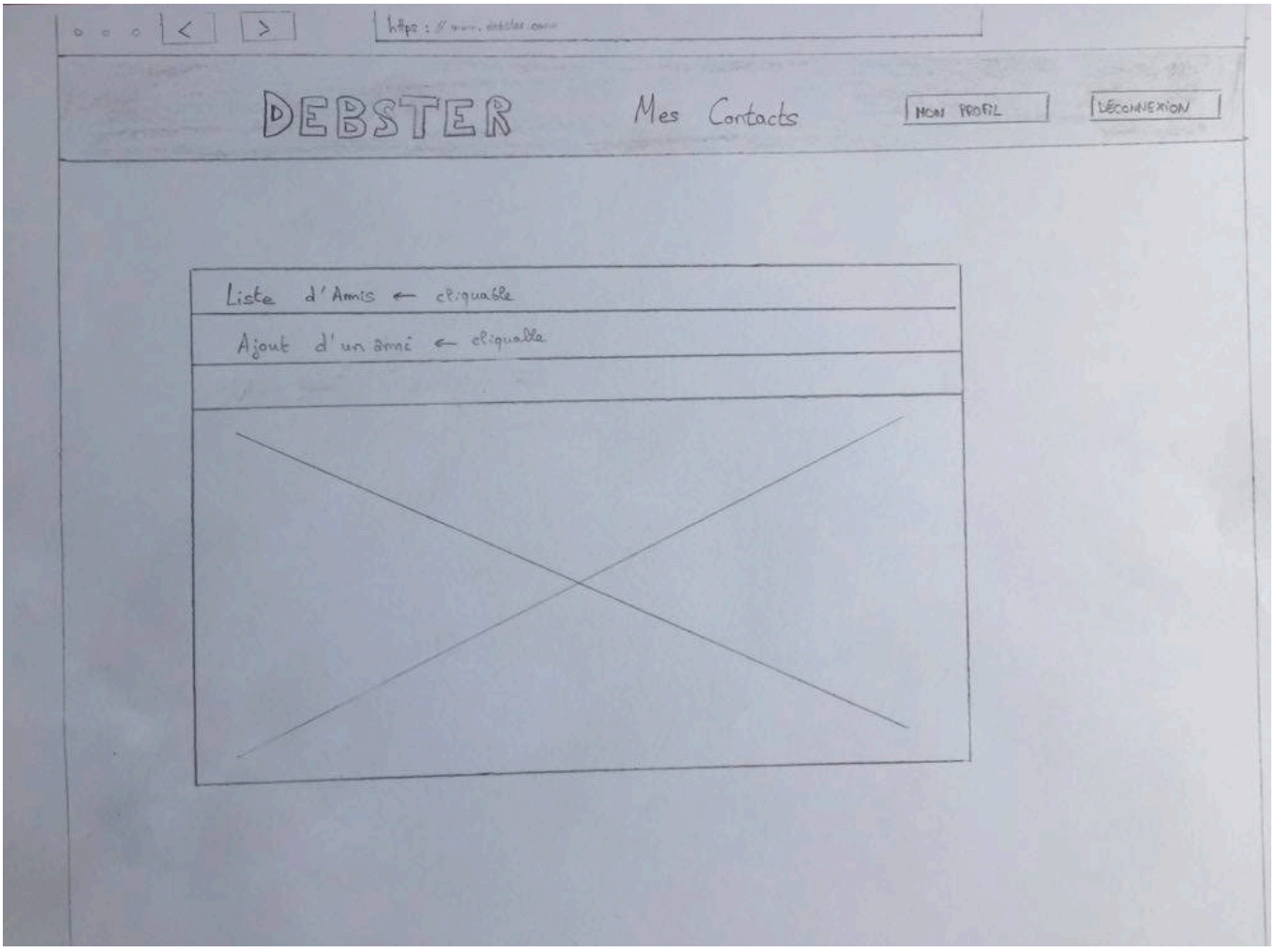
WIREFRAME 2: INSCRIPTION



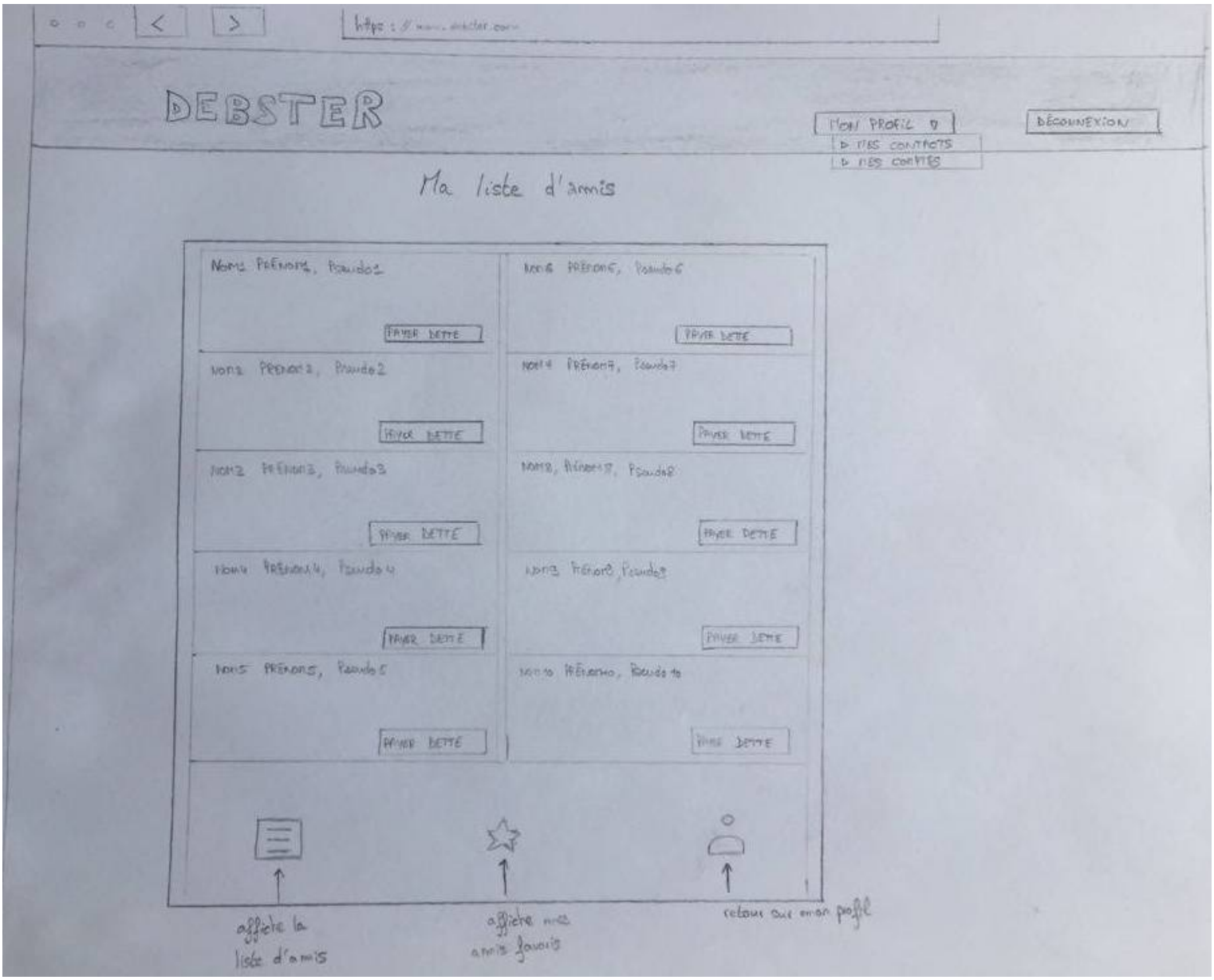
WIREFRAME 3: CONNEXION



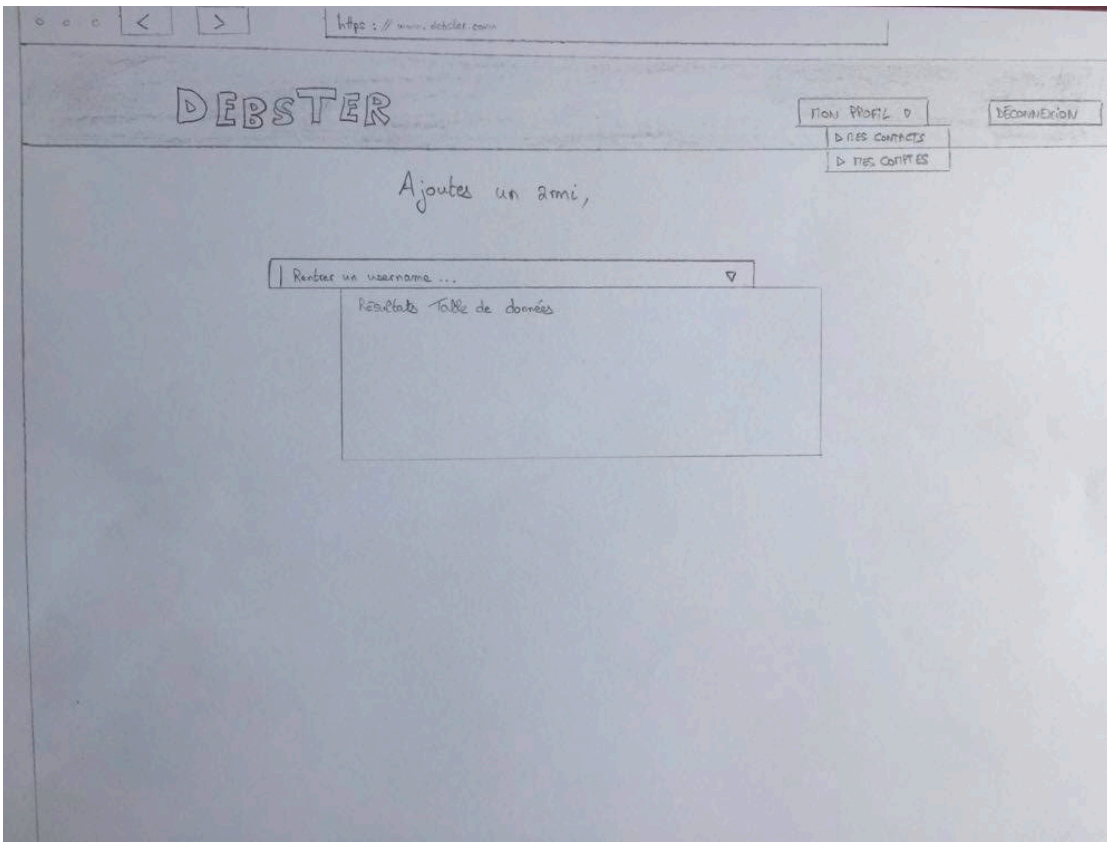
WIREFRAME 4: PAGE « MON PROFIL »



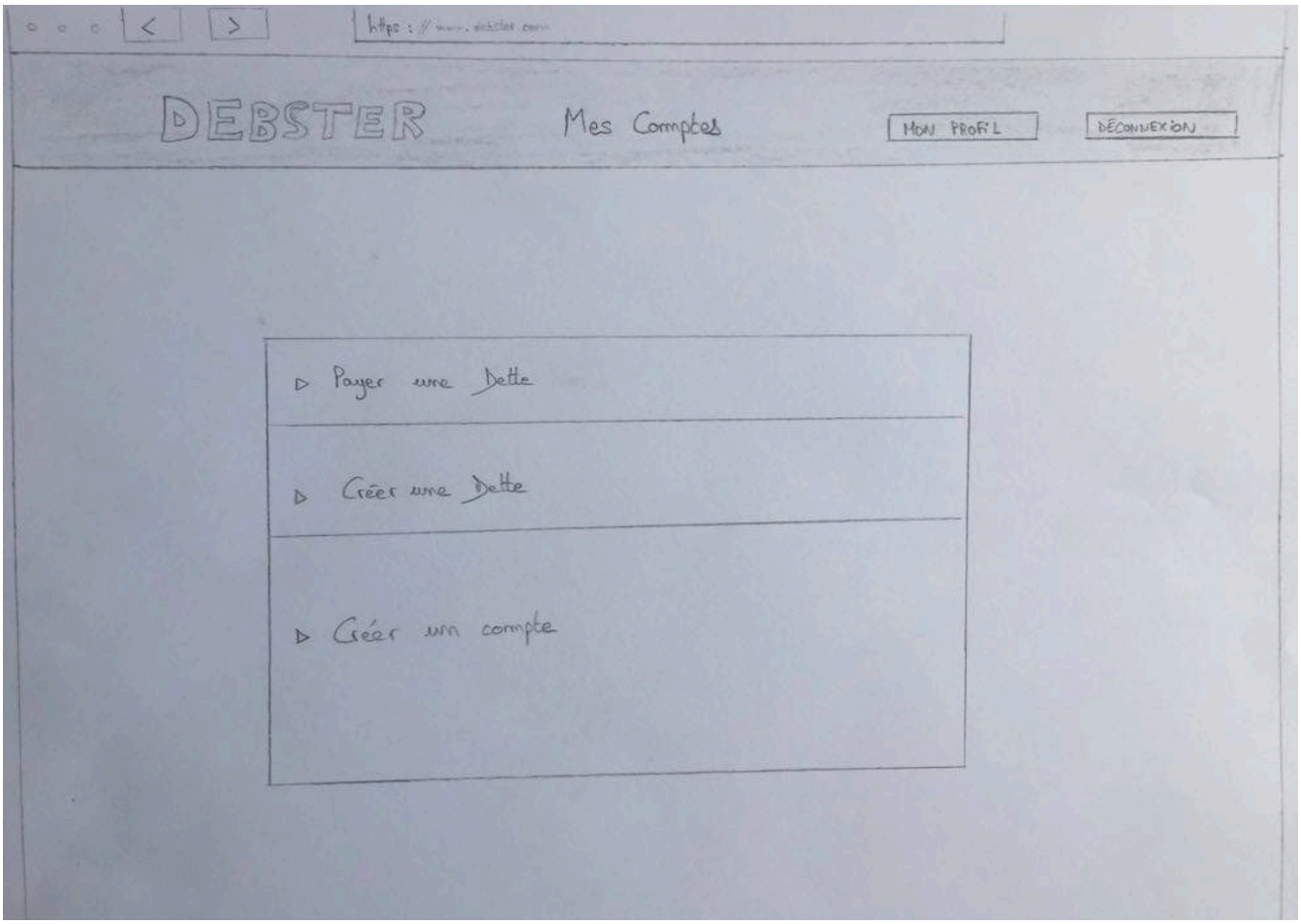
WIREFRAME 5: PAGE « MES CONTACTS »



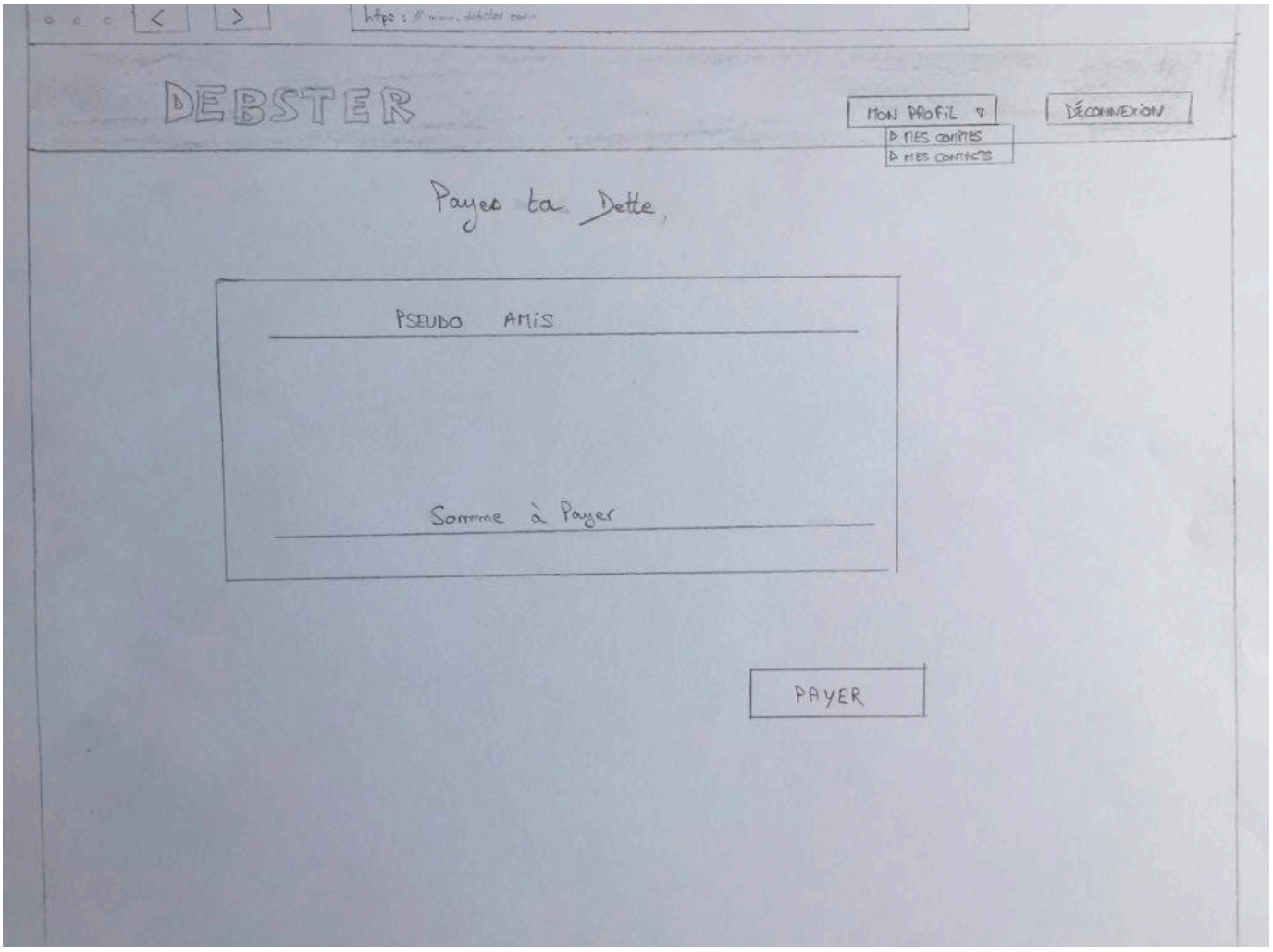
WIREFRAME 6: MA LISTE D'AMIS



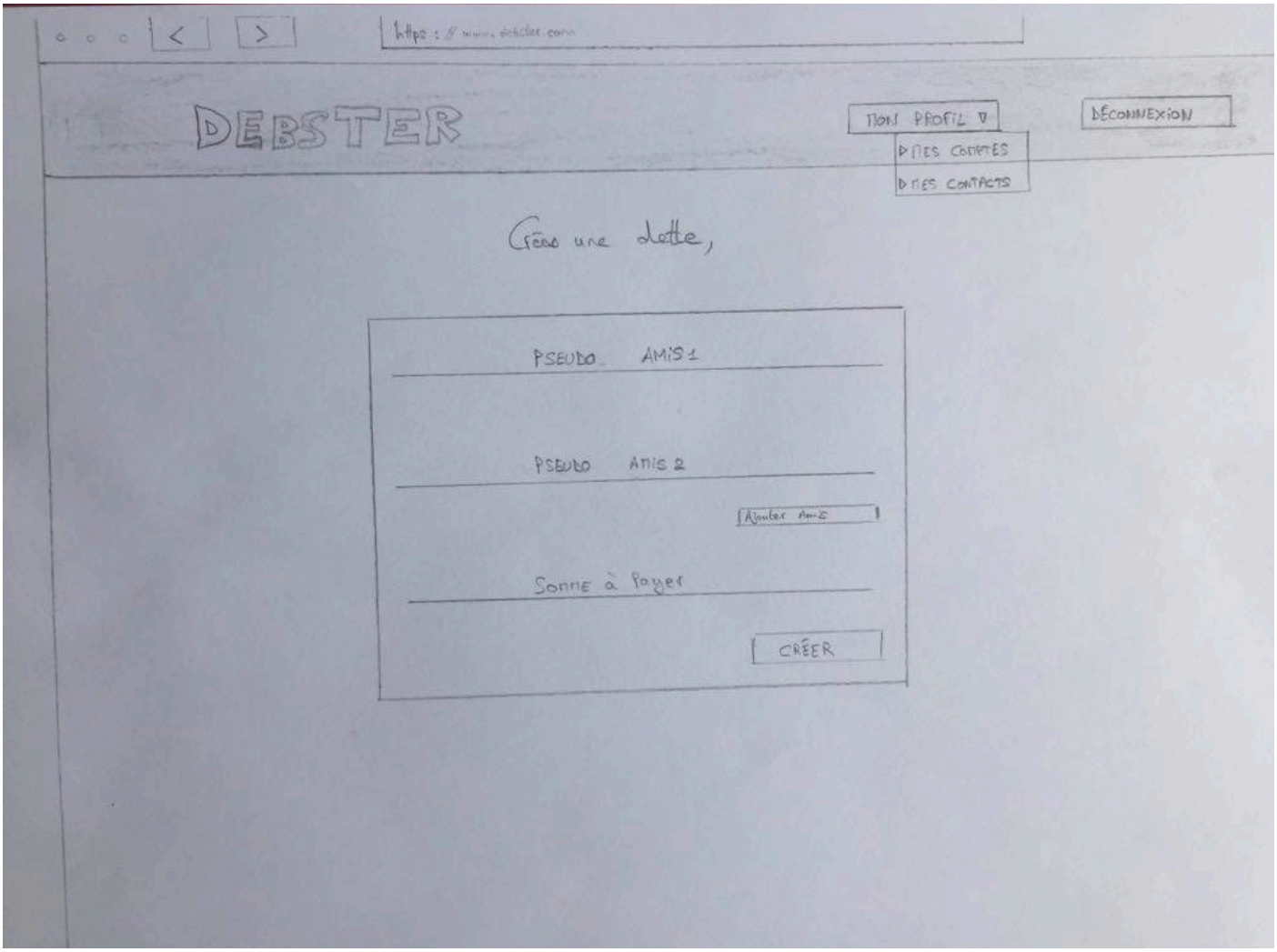
WIREFRAME 7: AJOUT D'UN AMI



WIREFRAME 8: PAGE « MES COMPTES »



WIREFRAME 9: PAIEMENT DETTE



WIREFRAME 10: CRÉATION DE DETTE

4. Database model

Table User		
ID	Int	PK
Prénom	Varchar	
Nom	Varchar	
Email	Varchar	
Mot_de_passe	Varchar	
Date de naissance		
Pseudo	Varchar	

Table Group		
ID	Int	PK
Nom	Varchar	
ID_leader	Int	FK Table User
Date_de_creation	Varchar	

Table Reach_my_friend		
ID_user_en_cours	Int	FK Table User
ID_user_ami	Int	FK Table User
Status	Varchar	

Table Reach_my_group		
ID_group_en_cours	Int	FK Table Group
ID_user	Int	FK Table User
Status	Varchar	

Table Transaction_Ami		
ID	Int	PK
ID_user_avance	Int	FK Reach_my_friend
ID_user_in_dept	Int	FK Reach_my_friend
Status	Varchar	
Date		
Contexte	Texte	
Somme_a_payer	Int	

PK: Primary Key: permet d'identifier chaque enregistrement dans une table de base de données

FK: Foreign Key: permet de « crosser » des colonnes entre tables

Note: *Nous ne sommes pas sûrs (même après recherche sur internet) que, pour utiliser une Foreign Key, les colonnes doivent avoir les mêmes noms. L'idée est bien sur d'utiliser une colonne déjà existante (par exemple dans la Table User) au sein d'une autre table. Peut-être que c'est une autre clé qu'il faut utiliser ?*

5. System structure - scripts overview

Connexion
- sign_in.php, wireframe 3
- Page de connection à l'application, formulaire simple.
- Méthode POST avec paramètres email, password vers connexion.php

Inscription
- sign_up.php, wireframe 2
- Page d'inscription à l'application, formulaire demandant toutes les informations
- Méthode POST avec paramètres de la table user
- Insert à la table "user"

Accueil
- accueil.php, wireframe 1 -> combinaison avec un autre document .css qu'on insérera

-Un fichier index.php afin de réaliser le squelette du site
-Un document pour l'ensemble des requêtes SQL

Mon profil
-my_user.php, Wireframe 4, Tables User & Transaction Ami à INSERT (pour gérer l'historique des transactions)

Mes contacts

-mes_contacts.php, wireframe 5, Tables Reach_my_friend & User à INSERT

-liste_amis.php, wireframe 6, Tables Reach_my_friend & User à INSERT
-ajout_amis.php, wireframe 7, Tables Reach_my_friend & User à INSERT

—> Indépendamment de ces fichiers, il faudra d'autres fichiers pour insérer les bases de données pour configurer chaque rubrique.

Mes comptes

-mes_comptes.php, wireframe 8

-transaction_ami.php, wireframe 9, Tables Reach_my_friend, User & Transaction_Ami à INSERT

-creation_dette.php, wireframe 10, Tables Reach_my_friend, User & Transaction_Ami à INSERT

6. Test strategy

6A. Test items

- Authentification de l'utilisateur
- Liste d'amis
- Payer/Créer une dette
- Créer/Rejoindre un groupe
- Agir sur un groupe choisi (ajouter des gens, quitter le groupe)
- Consulter l'historique des transactions

6B. Test cases

Test case ID: #1
Test case Name: *Inscription au site*
Test case Date:
Test case Purpose: *Valider l'inscription d'un utilisateur au site*
Test case Function: *Vérifier que tous les champs de l'utilisateur inscrit sont correctement remplis*
Test case Input: *Email valide, mot de passe, Prénom et Nom, Date de naissance numérique valide, pseudo*
Test case expected Output: *Un utilisateur peut s'enregistrer dans le système avec les informations correctes, le site réagit en cas d'invalidité dans les informations entrées*

Test case ID: #2
Test case Name: *Authentification de l'utilisateur et déconnexion*
Test case Date:
Test case Purpose: *Vérifier que seuls les utilisateurs inscrits correctement peuvent se connecter, et que la déconnection est possible ensuite*
Test case Function: *Seul un bon pseudo, lié au mot de passe correspondant, peut connecter l'utilisateur pendant 1h maximum, la déconnection est possible et fonctionne bien*
Test case Input: *Un pseudo et un mot de passe*
Test case expected Output: *Lorsque l'utilisateur rentre son pseudo et son mot de passe correctement, il est identifié et peut accéder aux fonctionnalités. Il peut ensuite se déconnecter, ou sinon il sera automatiquement déconnecté au bout d'1h. Si le mot de passe ou le pseudo ne sont pas bien entrés par l'utilisateur, un message d'erreur apparaît sur la page de connexion.*

Test case ID: #3
Test case Name: *Liste d'amis*
Test case Date:
Test case Purpose: *Vérifier le fonctionnement de la liste d'amis*
Test case Function: *Affichage de la liste d'amis de l'utilisateur, ajout d'un ami, suppression d'un ami*
Test case Input: *Consultation de la liste, pseudo ou adresse mail d'un autre utilisateur que l'on souhaite ajouter, ami que l'on souhaite supprimer*
Test case expected Output: *Le carnet d'amis, lorsqu'il est consulté, montre le prénom, le nom, le pseudo et le solde du compte des amis de l'utilisateur. L'utilisateur ajoute un ami : si le pseudo et l'adresse mail ne sont pas dans la base de données il apparaît un message d'erreur, sinon le carnet se rafraîchit correctement avec les infos du nouvel ami. Si l'utilisateur souhaite supprimer un ami, sa suppression n'est possible que si l'ami a un solde égal à 0, et dans ce cas le carnet se rafraîchit correctement sans l'ami supprimé.*

Test case ID: #4
Test case Name: *Consultation/Modification des Transactions*
Test case Date:
Test case Purpose: *Vérifier le fonctionnement des transactions avec un ami*
Test case Function: *Consulter les transactions, consulter l'historique, modifier les transactions*
Test case Input: *Consultation des transactions avec un ami*
Test case expected Output: *Les transactions ouvertes avec un ami sont affichées, dans l'ordre correct, les transactions de groupe apparaissent en italique. L'utilisateur peut choisir d'afficher les transactions réglées et donc de voir l'historique avec un ami. Les transactions non réglées peuvent être modifiées : le montant et la courte description.*

Test case ID: #5
Test case Name: *Saisir une transaction simple*
Test case Date:
Test case Purpose: *Valider le fonctionnement de la saisie d'une nouvelle transaction simple*
Test case Function: *Saisie d'une transaction simple*
Test case Input: *Une transaction simple*
Test case expected Output: *Les champs descriptifs d'une transaction simple sont correctement remplis, les valeurs par défauts sont correctes. Les propositions d'utilisateurs correspondent bien à la liste d'amis ou à des utilisateurs appartenant à la base de données. Bonne lecture de la dette.*

Test case ID: #6
Test case Name: *Saisir une transaction de groupe*
Test case Date:
Test case Purpose: *Valider le fonctionnement de la saisie d'une nouvelle transaction de groupe*

Test case Function: *Saisie d'une transaction de groupe*
Test case Input: *Une transaction de groupe*
Test case expected Output: *Les champs descriptifs d'une transaction de groupe sont correctement remplis, les valeurs par défauts sont correctes. Les propositions d'utilisateurs sont maintenant multiples et correspondent bien à la liste d'amis ou à des utilisateurs appartenant à la base de données. L'utilisateur peut choisir la répartition de la transaction. La base de données est correctement mise à jour.*

Test case ID: #7
Test case Name: *Annuler/Terminer une transaction*
Test case Date:
Test case Purpose: *Vérifier que l'utilisateur peut fermer ou terminer une transaction en cours*
Test case Function: *Fermeture d'une transaction*
Test case Input: *Annulation ou remboursement de la dette*
Test case expected Output: *L'utilisateur termine la transaction par annulation simple ou suite au remboursement de la dette. La base de données est correctement mise à jour, et l'utilisateur est amené à saisir un message de fermeture.*

Test case ID: #8
Test case Name: *Consultation de l'encours*
Test case Date:
Test case Purpose: *Vérifier la consultation de l'encours de l'utilisateur*
Test case Function: *Consultation de l'encours*
Test case Input: *Encours d'un utilisateur*
Test case expected Output: *L'utilisateur consulte son encours, qui affiche la somme de ses dettes et créances. Les données affichées sont cohérentes et se mettent à jour régulièrement.*