# Contexte de l’application : PcParted

Je souhaite créer une application qui me permettrait de facilement calculer le revenu que pourraient générer les diverses cartes graphiques que j’ai amassées au cours du temps. Simplement se dire qu’une carte plus récente sera meilleure pour du minage n’est pas forcément vrai et cela me semble être un sujet intéressant.

Dans cette application, il sera d’obtenir des informations utiles sur des cartes graphiques dans le cadre de minage de cryptomonnaie. PcParted dispose de nombreux filtres pour affiner votre recherche et trouver la carte qui vous correspond.

Vous pourrez ainsi, entre-autres, choisir d’afficher les cartes d’un certain constructeur, d’une certaine série ou de les trier par hashrate (anglicisme signifiant la capacité d’une unité de calcul à « miner » une cryptomonnaie, donné en mégahash par secondes). Chaque carte qui semble répondre à vos besoins peut être ajoutée à une liste à droite de l’écran en la glissant ou en appuyant sur le bouton « ajouter » (mode détaillé).

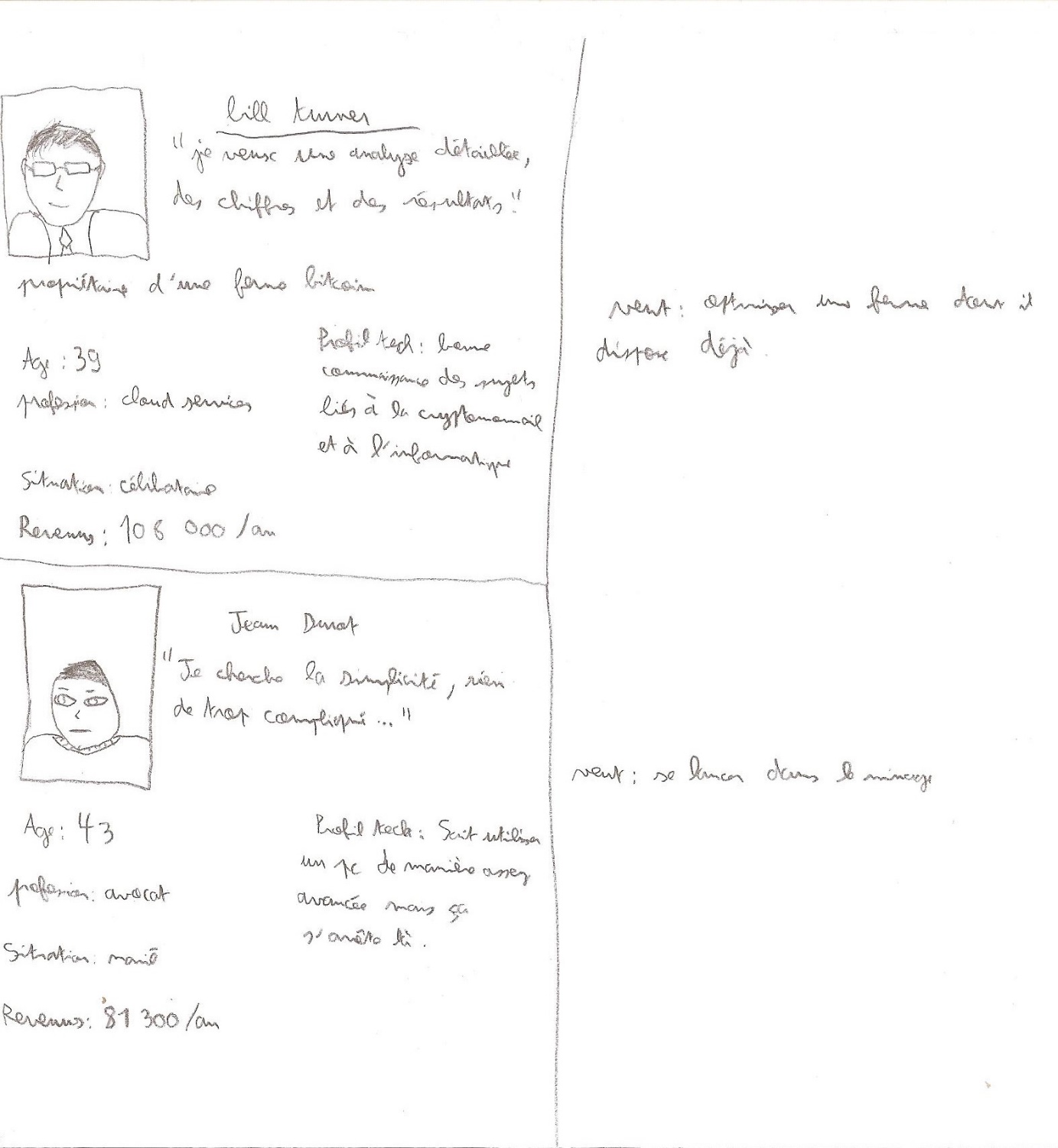
Pour chaque carte affichée il est possible d’obtenir plus de détails en cliquant dessus. Cela lance alors le mode détaillé qui contient toutes les informations techniques dont vous pourriez avoir besoin.

Même si à la base cette application sera dédiée à des cartes graphiques, son mode de fonctionnement sera assez flexible pour permettre d’intégrer tout ce que l’on veut à sa liste. Tant qu’il existe un moyen d’extrapoler une puissance de hachage théorique de l’appareil, ou que l’entrée customisée dispose d’un fichier attaché précisant prix, consommation, et hashrate, l’application saura traiter ces données.

Un mode de prédiction graphique sera disponible, pour mieux visualiser les chiffres et aider dans la prise de décision sur une certaine période.

Enfin, chaque carte ajoutée à la liste de droite sera comptabilisée dans le prix, la consommation et la hashrate totale (avec notation en cas d’incompatibilité ou de blocage de minage par le constructeur).

# Personas



# User stories : Bill Turner

En tant que responsable des services cloud dans une grande entreprise, je recherche une application qui me donne des informations utiles et détaillées, avec une option pour conserver une ou plusieures listes de mineurs afin de faire correspondre les données de l’application avec les mineurs dont je dispose déjà. Etant déjà bien versé dans l’art du minage, les données ne me feront pas peur.

Une inclusion des cartes dédiées spécifiquement au minage (AntMiners) serait une addition appréciable.

# User stories : Jean Durot

En tant que débutant dans le milieu du minage de cryptomonnaies et dans le milieu informatique, je veux une interface simple et intuitive, afin de rendre mon expérience plus facile. J’ai déjà essayé des outils de l’industrie plus complexes et ils m’ont rapidement découragé. Je n’ai pas besoin de grands détails mais d’une mise en évidence de ce qui est important.

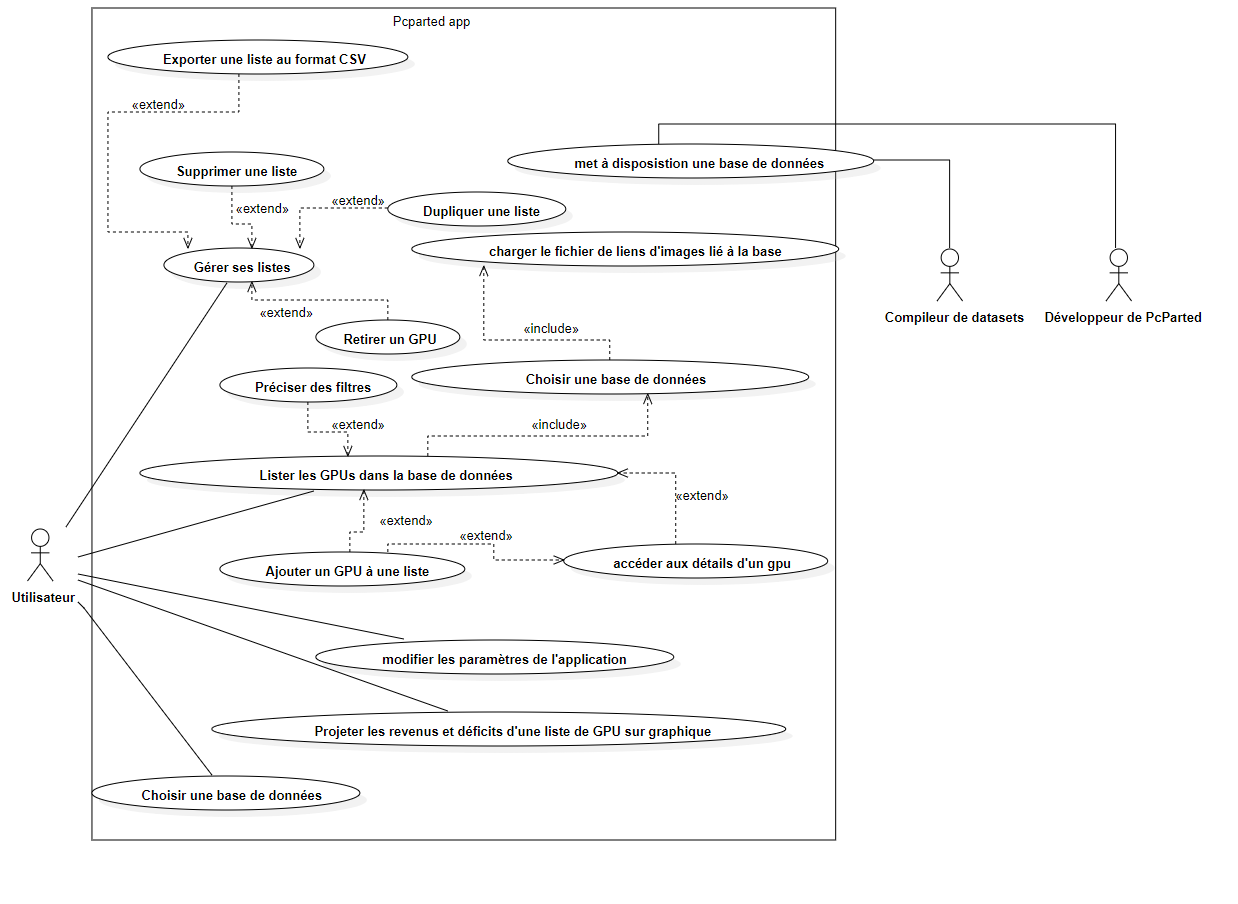
Je veux juste me lancer dans quelque chose de nouveau, et le monde de la crypto me paraît si mystérieux…

# Sketchs/Storyboards (wireframe)



# Diagramme de Flux

# Diagramme de cas d’utilisation



## CAS « Choisir une base de données »

|  |  |
| --- | --- |
| Nom | Choisir une base de données |
| Objectif | Choisir une BDD de cartes graphiques à charger dans l’application |
| Acteurs principaux | Utilisateur |
| Acteurs secondaires | ∅ |

|  |  |
| --- | --- |
| Conditions initiales | -L’utilisateur dispose d’un fichier de Basxe De Données (BDD) compatible avec l’application (.pnm) |
| Scénario d’utilisation | -L’utilisateur précise un chemin dans la boîte de dialogue ou parcourt son disque vers le fichier pnm.  -Le système vérifie l’intégrité du fichier (si corrompu ou incorrect, conditions de fin 2)  -Le système charge le fichier .pem (fichier de liens d’images) indiqué par le fichier .pnm (si absent, conditions de fin 3)  -Le système charge le fichier .pwr (fichier de données de consommation, de prix et de hashrate) (si absent, conditions de fin 4).  -Le système affiche les données chargées (conditions de fin 1) |
| Conditions de fin | 1. Fichiers pnm, pem et pwr corrects et intègres : la BDD est chargée dans l’application.   2) Fichier .pnm corrompu, introuvable ou ne peut pas être ouvert. Le système indique un code erreur et ne charge pas la base de données.   1. Fichier .pem manquant ou incorrect (nombre de lignes différent), la nouvelle base de données sera chargée mais avec des images d’espace réservé [placeholders basés sur la marque de la carte]. 2. Fichier .pwr manquant ou incorrect (nombre de lignes différent), la nouvelle base de données sera chargée mais les données telles que le prix, la hashrate et la consommation seront extrapolées mathématiquement en fonction de la marque, de la date de sortie, du nombre de cœurs, de la fréquence, ect… [un symbole Ligne Noire Icône Pour Des Généralités, La Généralisation Et Le Verre  Illustration de Vecteur - Illustration du logotype, graphisme: 144077947sera présent en haut à gauche des cartes extrapolées avec une tooltip expliquant sa signification]. |

## Cas « Lister les GPUs dans la base de données »

|  |  |
| --- | --- |
| Nom | Lister les GPUs dans la base de données |
| Objectif | Lister les gpus présents dans la base de données actuellement chargée |
| Acteurs principaux | Utilisateur |
| Acteurs secondaires | ∅ |

|  |  |
| --- | --- |
| Conditions initiales | -Au moins une liste existe avec au moins un GPU |
| Scénario d’utilisation | -L’utilisateur ouvre l’onglet « vue générale » ou lance l’application (action par défaut au lancement de l’application) (conditions de fin 1)  -L’utilisateur précise des filtres et / ou effectue une recherche (conditions de fin 2) |
| Conditions de fin | 1. Tous les GPUs de la base de données choisie sont affichés, triés par hasrate par défaut. [Priorité aux non-extrapolés de ~10%] 2. Les GPUs remplissant les conditions des filtres et/ou de la recherche sont affichés, triés de la même façon que dans le cas 1) |

## Cas « Accéder aux détails d’un gpu »

|  |  |
| --- | --- |
| Nom | Accéder aux détails d’un gpu |
| Objectif | Obtenir plus d’informations sur un GPU en particulier, [et] l’ajouter à la liste courante |
| Acteurs principaux | Utilisateur |
| Acteurs secondaires | ∅ |

|  |  |
| --- | --- |
| Conditions initiales | -Au moins une liste existe avec au moins un GPU  -Les GPUs ont été listés comme dans le cas précédent |
| Scénario d’utilisation | -L’utilisateur clique [clic gauche] sur le GPU qui l’intéresse, cela ouvre une version plus grande de la carte avec une description plus complexe contenant les spécifications de cette carte graphique OU l’utilisateur effectue un [clic droit] sur la carte du GPU, ce qui l’ajoute immédiatement à la liste (conditions de fin 1)  -L’utilisateur clique sur le bouton « ajouter » dans la carte agrandie (conditions de fin 1)  -L’utilisateur clique sur le bouton « retour » dans la carte agrandie (conditions de fin 2) |
| Conditions de fin | 1. Le GPU sélectionné est ajouté à la liste courante 2. Retour à la vue précédente, la carte reprend son aspect normal |

## Cas « Gérer ses listes »

|  |  |
| --- | --- |
| Nom | Gérer ses listes |
| Objectif | Dupliquer, modifier, supprimer, exporter une liste existante ou créer une liste vide. |
| Acteurs principaux | Utilisateur |
| Acteurs secondaires | ∅ |

|  |  |
| --- | --- |
| Conditions initiales | -Au moins une liste existe  -L’application a les droits d’écriture dans son dossier courant |
| Scénario d’utilisation | -L’utilisateur sélectionne une liste à traiter dans la comboBox dédiée  -Il peut cliquer sur un des 5 boutons à droite de la comboBox pour respectivement Créer une nouvelle liste vide (conditions de fin 3), dupliquer, modifier, ou supprimer la liste (conditions de fin 1). Il est également possible de l’exporter au format .csv (conditions de fin 4)  -Lorsqu’une liste est choisie, ses GPUs s’affichent dans la dataGrid en-dessous des boutons. En cliquant sur un GPU dans la liste, les détails de ce GPU s’affichent. Il est alors possible de le retirer de la liste (conditions de fin 2) ou de l’exporter dans le presse-papier (conditions de fin 5) |
| Conditions de fin | 1. Les modifications sont apportées à la liste sélectionnée 2. Le GPU sélectionné est supprimé de la liste 3. Une nouvelle liste vide est créée 4. Un fichier nom\_de\_la\_liste.csv est créé dans le dossier ./PcParted/exported/csv/ ou dans le dossier spécifié dans les paramètres 5. Les données du GPU séléctionné sont enregistrées dans le presse-papier (donnés du fichier .pnm + données du fichier .pem [si présent] + données du fichier .pwr [si présent] |

## CAS « Modifier les paramètres de l’application »

|  |  |
| --- | --- |
| Nom | Modifier les paramètres de l’application |
| Objectif | Changer des paramètres divers pour le calcul et la sauvegarde de données |
| Acteurs principaux | Utilisateur |
| Acteurs secondaires | ∅ |

|  |  |
| --- | --- |
| Conditions initiales | -L’application a les droits d’écriture dans son dossier courant |
| Scénario d’utilisation | -L’utilisateur précise des constantes qui serviront aux calculs de l’application, ainsi que d’autres comme le chemin de sauvegarde et de lecture par défaut des bases de données et de listes (à la convenance de l’utilisateur)  -L’utilisateur appuie sur le bouton « appliquer ». Une série de vérifications intégrées se lance. Les champs invalides seront entourés en rouge par le système avec un message aidant à la correction. Tant qu’ils n’auront pas été corrigés, le bouton appliquer sera désactivé. |
| Conditions de fin | 1. Le fichier .ini de l’application enregistre les changements et l’application continue son fonctionnement normal |

## Cas « Projeter les revenus et déficits d'une liste de GPU sur graphique»

|  |  |
| --- | --- |
| Nom | Projeter les revenus et déficits d'une liste de GPU sur graphique |
| Objectif | Prédire les revenus et dépenses sur une durée T indiquée |
| Acteurs principaux | Utilisateur |
| Acteurs secondaires | ∅ |

|  |  |
| --- | --- |
| Conditions initiales | -Au moins une liste existe avec au moins un gpu à l’intérieur. |
| Scénario d’utilisation | -L’utilisateur séléctionne l’onglet « graphe » de l’application.  -Il précise une liste de GPUs à utiliser, et donne la durée souhaitée en jours, mois ou années.( si aucune liste disponible, conditions de fin 2) (si durée incorrecte, conditions de fin 3)  -L’utilisateur  -L’application génère le graphique (conditions de fin 1) |
| Conditions de fin | 1. Le graphe est généré et visible dans l’application, il peut également être exporté en tant qu’image. 2. Sans liste disponible, le bouton « générer » restera désactivé avec une indications « aucune liste disponible ». 3. La textBox de durée sera encadrée en rouge et le bouton « générer » sera indisponible |