Justin Lagüe, Étienne Dubé,

Christophe Duchesne, Étienne Cloutier

Projet d’intégration en Sciences informatiques et mathématiques

420-204-RE, gr.00001

Dossier de conception

**CREDIGIT par Bank-era Corp.**

Travail présenté à

M. Walid Boulabiar

Département d'informatique

Cégep Limoilou

Le 6 février 2020

Table des matières

[Description du projet￼ ￼](bookmark://_Toc31531429#_Toc31531429)

[Objectifs 1](#_Toc31531430)

[Description détaillée 1](#_Toc31531431)

[Concepteurs, Rôles et justifications 6](#_Toc31531432)

[Répartition des tâches 6](#_Toc31531433)

[Répartition des rôles 7](#_Toc31531434)

[Type d’application 8](#_Toc31531435)

[Langages utilisés, patrons de conception et cadre d’application 8](#_Toc31531436)

[Technologies impliquées 8](#_Toc31531437)

[Références et documentation 8](#_Toc31531438)

[Prototypes de l’application 10](#_Toc31531439)

[Prototypes 10](#_Toc31531440)

[Principaux objets 10](#_Toc31531441)

[Scénarios ou user stories 10](#_Toc31531442)

[Diagramme de classes 14](#_Toc31531443)

[Échéancier 14](#_Toc31531444)

# Description du projet

## Objectifs

Notre projet vise à pallier les enjeux de sécurité et de logistique reliés aux transactions par cartes. Dans ce sens, nous proposons un nouveau mode de paiement basé sur la lecture d’empreintes digitales plus sécurisé que les systèmes actuels. En effet, les modes de paiement utilisés présentement peuvent facilement être fraudés; les cartes peuvent êtres perdues ou volées et les mots de passe (NIP) peuvent être oubliés ou trouvés. C’est là que notre solution prend les devants en palliant tous les problèmes énumérés plus haut. Notre système permettra d’éradiquer les oublis tout en augmentant la sécurité puisque le client n’aura qu’à utiliser son doigt pour effectuer toutes ses transactions et les empreintes digitales, étant uniques, rend le tout très sécuritaire.

## Description détaillée

Notre projet consiste à concevoir un système de paiement par carte de même type que VISA et MASTERCARD. Cependant, ce système ne fonctionnera pas avec une carte, mais avec une empreinte digitale; celle-ci permettra d’associer une personne à un compte. Étant unique, l’empreinte permet d’enlever l’étape du NIP tout en gardant une sécurité plus accrue tout en permettant de faciliter les transactions. Il est bien important de garder en tête que ce système constitue une solution que les commerces et les institutions bancaires pourront implémenter et non un système bancaire en ligne. Puisque ce système est unique, nous devons offrir une solution aux commerçants afin qu’ils puissent utiliser notre produit. C’est dans cet objectif que nous allons construire, en plus d’un système contenant les informations du client, un terminal de paiement et un point de vente (une caisse enregistreuse).

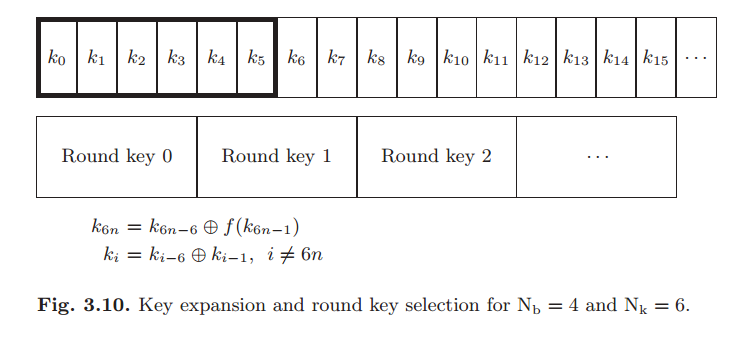
Bien que l’objectif soit d’offrir une expérience de paiement fluide et simple pour les clients et les commerçants, la réalisation de ce projet est loin d’être simple.

Tout d’abord, nous devrons créer un programme permettant l’inscription de nouveaux utilisateurs à notre service. Pour ce faire, nous devrons créer un formulaire d’inscription enregistrant toutes les informations sur l’utilisateur ainsi que son empreinte digitale sous forme d’un *array* de *bytes.* Ces informations une fois collectées, nous devrons ensuite utiliser un système d’encryptage (Voir section sur l’AES) afin de protéger ces données. Par la suite, nous devrons stocker ces données dans une base de données pour que celles-ci soient accessibles n’importe où.

Ensuite, afin d’assurer une sécurité optimale des données personnelles stockées sur la base de données, celles-ci seront encryptées selon l’Advanced Encryption Standard (AES). Ce standard s’appuie sur l’algorithme de Rijndael. Cet algorithme est l’un des plus sécuritaires dans l’encryptage et est couramment utilisé (notamment par le gouvernement américain); il sera donc choisi pour notre projet. Cet algorithme utilise des notions mathématiques très complexes : le changement de base, les opérations matricielles, les espaces vectoriels et la théorie de Galois. Donc, les mathématiques auront un apport considérable dans notre projet, voire dans la conception de cet algorithme. Cet algorithme sera implémenté par nous. Celui-ci sera décrit dans les grandes lignes ici-bas. Pour en connaître davantage sur les technicités de cet algorithme, il est fortement conseillé de se référer aux documents de référence.

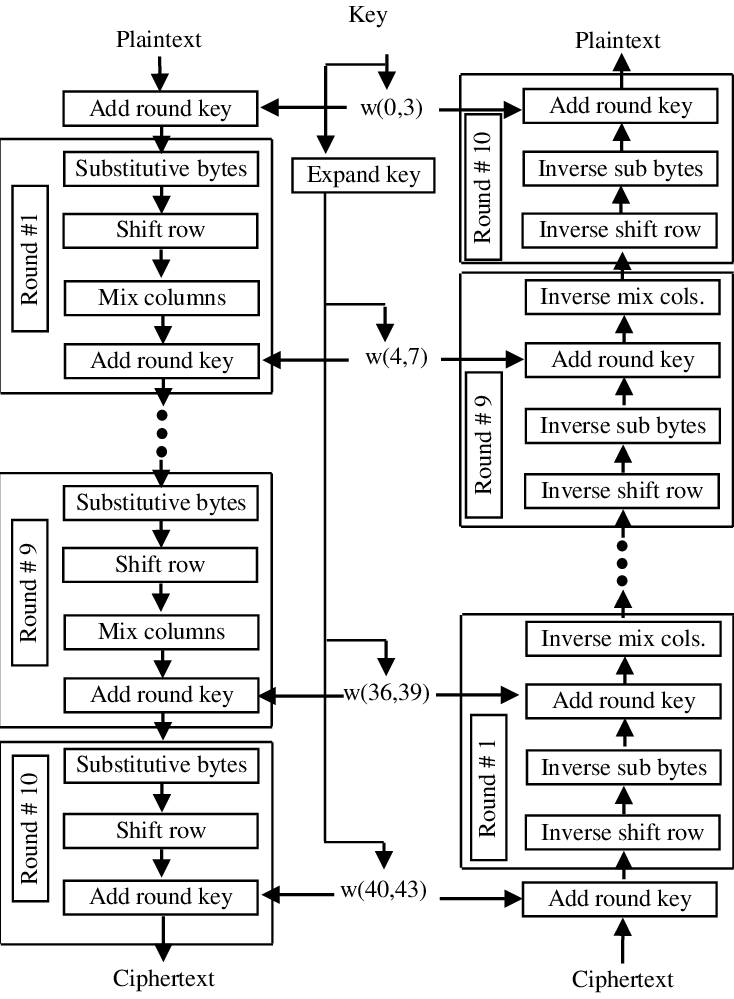
Cet algorithme fonctionne avec des blocs de chiffrement de 128, 192 ou 256 bits. Il y a deux types de blocs: les blocs de texte à encrypter et les blocs de clé. Ces blocs peuvent être visualisés comme des matrices de 4 rangées. Le nombre de colonne correspond à la longueur de bits du bloc divisée par 32. Chaque position de la matrice contient un octet de 8 bits. Chaque octet est rentré dans la matrice selon un ordre prédéterminé.

Sachant cela, la clé initiale doit être allongés à l’aide d’un algorithme fournit dans la documentation. À partir de ce point, des clés différentes peuvent être trouvées. En effet, l’algorithme nécessite une clé différente pour toutes les boucles de l’algorithme ce qui augmente la sécurité de ce système.



**Image 1: Expansion de la clé originale**

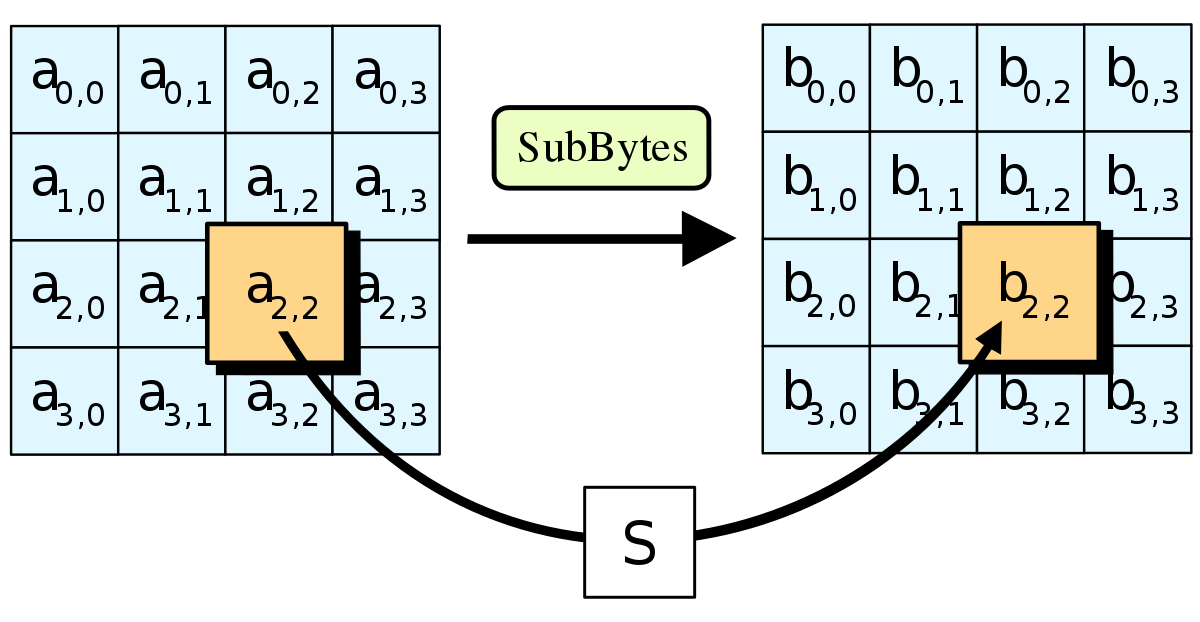
Par la suite, une boucle d’étapes se répète un certain nombre de fois selon la longueur du bloc d’encryptage et celui de la clé. Dans notre cas, puisque des blocs de 128 bits seront utilisés, l’algorithme fera 10 boucles. Toutes les boucles sont similaires sauf la dernière qui est légèrement différente.



**Image 2: L’algorithme de l’AES**

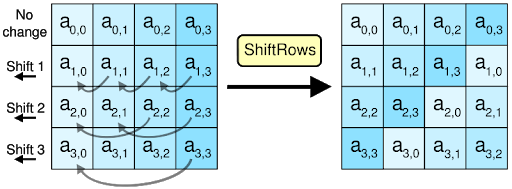
Chaque boucle contient généralement les opérations suivantes : *ByteSub*, *ShiftRow*, *MixColumn* et *AddRoundKey*.

L’opération *ByteSub* est constituée de deux fonctions mathématiques. La première inverse l’octet. La seconde multiplie chaque octet par une matrice prédéfinie. On parle ici d’une multiplication matricielle.



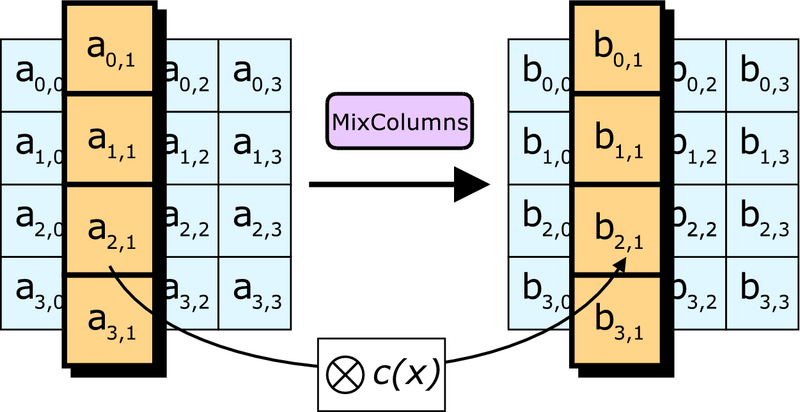
**Image 3: L’opération *SubByte***

L’opération *ShiftRow* mélange les rangées de 0, 1, 2 ou 3 cases vers la gauche.



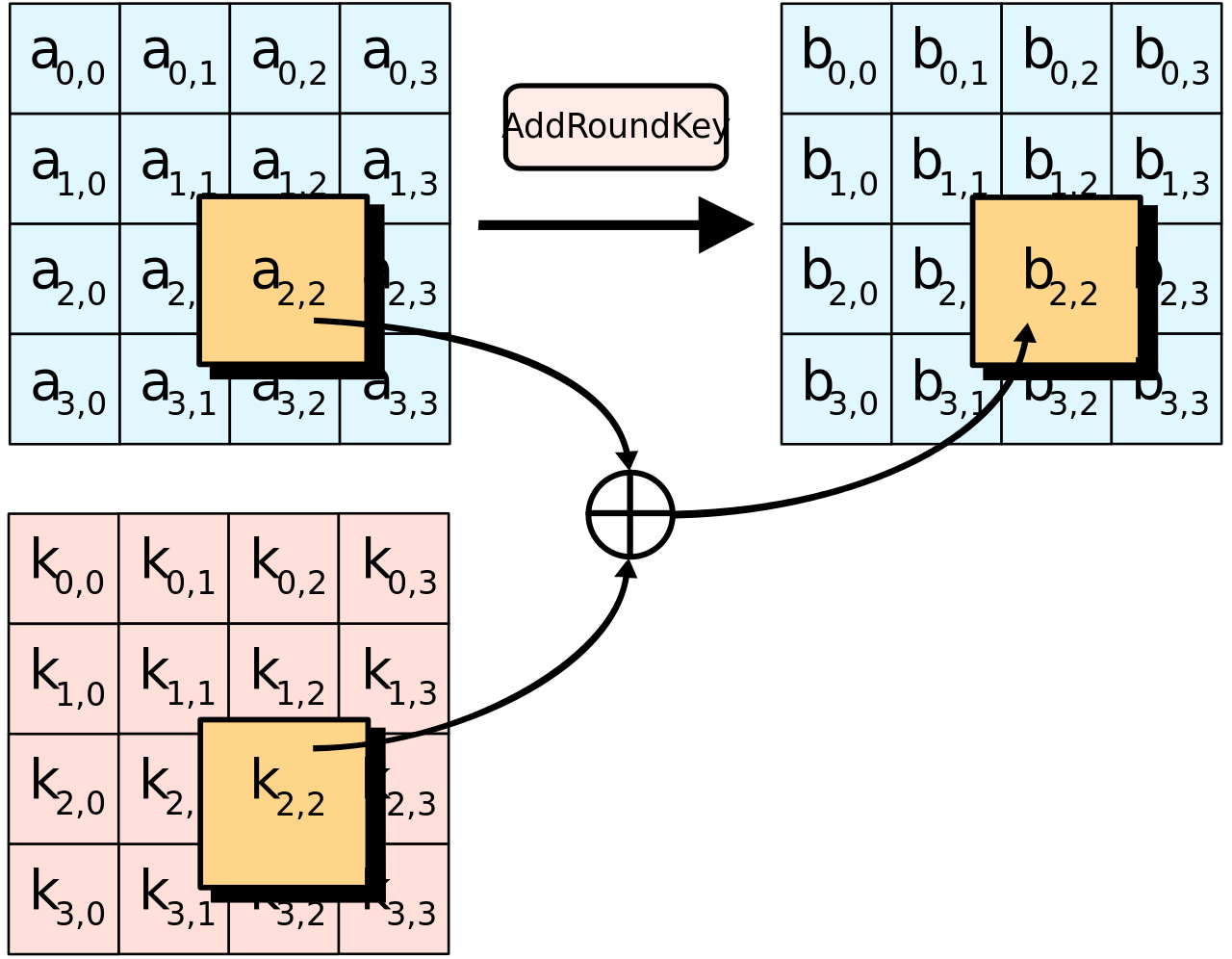
**Image 4: L’opération *ShiftRow***

L’opération *MixColumn* multiplie chaque colonne par une matrice prédéfinie. On parle ici d’une multiplication matricielle.



**Image 5: L’opération *MixColumn***

L’opération *AddRoundRey* constitue un XOR de la matrice résultante avec la clé correspondante à la boucle où l’algorithme est rendu.



**Image 6: L’opération *AddRoundKey***

Pour décrypter l’AES, il suffit d’appliquer les opérations dans l’ordre inverse. Il est recommandé de se référer à la documentation pour s’assurer de la validité du décryptage.

Nous devrons ensuite monter un point de vente (POS) pour les commerçants désireux d’implémenter les paiements par empreintes dans leurs établissements. En effet, notre projet étant un service unique, les POS actuels ne sont pas compatibles avec notre offre et c’est pour cette raison que nous devrons en créer un. Un POS consiste en un programme permettant à un commerce de scanner ou sélectionner des articles et les charger à un client. Pour ce faire, un commis doit être en mesure d’ajouter ou supprimer des produits dans une commande, en faire le total, y ajouter les taxes et charger le coût total au client. Nous devrons donc implémenter toutes ces fonctions ainsi que d’autres telles que la possibilité de créer ou supprimer des produits, entrer des informations manuellement, choisir le pourcentage de taxes, créer des caissiers et créer des rapports.

Une autre étape de la conception de ce système consiste à construire un terminal de paiement afin d’accepter les transactions. Ce terminal est la pièce maitresse de ce projet puisque la majorité des opérations seront effectuées dessus. En effet, celui-ci doit afficher le total de la transaction au client, scanner l’empreinte du client, accéder à la base de données et retrouver le compte du client, décrypter et encrypter les données, déterminer si la transaction peut être effectuée, effectuer la transaction, transmettre le statut de la transaction au POS et envoyer une facture par courriel au client. Bien qu’une partie de ces opérations peuvent être effectuées via le POS, nous préférons les effectuer sur le terminal pour augmenter la sécurité en gardant le plus d’informations possibles inaccessibles aux organisations tierces. Afin de réaliser cette pièce, nous allons utiliser un Raspberry Pi 3, un écran tactile et un scanneur d’empreintes digitales. Notre choix s’est arrêté sur le Raspberry Pi 3, car celui-ci sera assez puissant pour effectuer toutes les opérations énumérées plus haut.

Voilà qui englobe les grandes lignes de ce projet. Toutes ces différentes parties communiqueront ensembles afin d’offrir l’expérience promise aux clients et aux commerçants. Cette communication est quelque chose que nous n’avons jamais fait et que nous devons apprendre. Finalement, la majorité des opérations requises pour l’AES sont nouvelles pour nous et nécessitent aussi leur apprentissage.

## Concepteurs, Rôles et justifications

### Répartition des tâches

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tâche/Nom** | **Christophe** | **Justin** | **Étienne D.** | **Étienne C.** |
| **Conception de l’interface** | X |  |  |  |
| **Construction du terminal** |  | X | X |  |
| **Développement du «back-end» du terminal** |  |  | X |  |
| **Encryptage des données** |  | X |  | X |
| **Conception du point de vente (POS)** | X | X | X | X |
| **Conception du fichier d’inscription** | X |  |  | X |
| **Recherchiste** | -Base de données, système de facturation et interface | -Construction du terminal | -Conception du «back-end» | -Encryptage des données |
| **Gestion du temps et des travaux** |  |  |  | X |
| **Conception du UML** | POS et Inscription | AES | «Back-end» du terminal | AES |
| **Conception du plan des composantes électroniques** |  | X | X |  |
| **Conception et organisation de la base de données** | X |  |  |  |
| **Système de facturation** | X |  | X |  |
| **Conception des tests de l’application** | X | X | X | X |
| **Rédaction des rapports** | X | X | X | X |

### Répartition des rôles

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Rôle/Nom** | **Christophe** | **Justin** | **Étienne D.** | **Étienne C.** |
| **Chef de la communication et de l’animation** | X |  |  |  |
| **Chef de la planification et de la direction** |  |  |  | X |
| **Contrôle de la qualité (Relecture du code)** |  | X |  |  |
| **Contrôle de la qualité (Tests du code)** |  |  | X |  |
| **Secrétaire des réunions et des documents** | X |  |  |  |

### Justification de la répartition des tâches et des rôles

Justin Lagüe va principalement s’occuper du terminal, car il a de très bonnes connaissances en ce qui attrait à l’électronique. En effet, à l’université, il aimerait continuer son cheminement scolaire en génie informatique. Cela sera donc une bonne expérience dans le domaine qu’il souhaite poursuivre. De plus, il est très minutieux dans son travail et est un excellent programmeur. Cela lui permettra de bien relire le code pour s’assurer que tout soit présent et conforme aux normes.

Christophe Duchesne va principalement s’occuper à ce qui attrait à l’interface et aux bases de données. En effet, dans ses temps libres, il code une application de réseaux sociaux pour mobile. Celle-ci nécessite la conception de nombreuses interfaces et le stockage de données sur des bases de données. Ces connaissances lui seront donc un atout. Aussi, il aime bien parler avec les autres et il est assidus. C’est pour cela qu’il sera chef de la communication et secrétaire aux réunions.

Étienne Dubé s’occupera en grande partie du terminal et de la conception du back-end de celui-ci. Tout comme Justin, il a un grand intérêt pour ce genre de travail et a de nombreuses connaissances dans ce domaine. Aussi, il est très méthodique dans son travail et un programmeur averti; cela lui permettra de réviser les tests pour s’assurer que ceux-ci soient valables.

Étienne Cloutier s’occupera de l’AES et de la conception des différentes interfaces. En effet, il est très bon en mathématiques; cela lui permettra de mieux comprendre l’AES qui est très compliqué sur le plan mathématique. Aussi, il possède de bonnes bases en programmations qui seront mises de l’avant dans la conception des interfaces. De plus, il est très organisé et est capable d’estimer la durée de tâches futures. C’est pour cela qu’il sera le chef de la planification et de la direction.

## Type d’application

Puisque notre projet est divisé en trois sections : inscription, terminal et POS, nous allons effectuer trois applications différentes. Les trois applications seront des applications de bureau. En effet, le terminal possèdera un système d’exploitation et supportera donc aussi les applications de bureau. Conserver le même type d’applications pour les différentes parties permettra une meilleure communication entre celles-ci.

## Langages utilisés, patrons de conception et cadre d’application

Pour ce projet, nous allons utiliser Java comme langage de programmation. Nous avons décidé d’utiliser ce langage, car celui-ci est supporté sur toutes les plateformes que nous allons utiliser et cela nous permettra ainsi d’éviter de coder en double. Nous allons utiliser le modèle MVC (modèle, vue, contrôleur), car ce patron de conception s’applique bien pour des applications de bureau. Nous allons aussi utiliser différentes librairies pour notre projet dont trois différentes librairies pour connecter nos applications à notre base de données. Ces trois librairies sont fournies par *MongoDB,* le service que nous allons utiliser pour notre base de données. De plus, nous allons employer une librairie disponible sur *Github* nommée «java-fingerprint-sensor» pour convertir les empreintes lues sur le scanneur en *array* de *bytes*.

## Technologies impliquées

Pour mener à bien notre tâche, nous ferons appel à différentes technologies. Pour coder, nous allons utiliser comme IDE *Eclipse* puisque c’est celui avec lequel nous avons travaillé depuis le début du cégep. Nous ferons aussi appel à *Scene Builder* afin de concevoir nos interfaces graphiques. Par la suite, nous allons utiliser un Raspberry Pi afin de construire notre terminal. En résumé, un Raspberry Pi est essentiellement un petit ordinateur qui fonctionne sous Linux. Combiné à ce petit ordinateur, nous allons utiliser un lecteur d’empreintes digitales et un petit écran tactile.

## Références et documentation

**Base de données**

1. Katsov, I. (2012), *NoSQL data modeling techniques.* Repéré à <https://highlyscalable.wordpress.com/2012/03/01/nosql-data-modeling-techniques/>
2. MongoDB (2020), *NoSQL databases explained.* Repéré à <https://www.mongodb.com/nosql-explained>
3. MongoDB (2020), *MongoDB Docs.* Repéré à <https://docs.mongodb.com/>

**Système de facturation**

1. Irby, L. (2019), *7 Things to Know About Your Credit Card Billing Statement.* Repéré à <https://www.thebalance.com/credit-card-billing-statement-959999>
2. Irby, L. (2019), *How to Understand Your Credit Card Billing Statement.* Repéré à <https://www.thebalance.com/how-to-understand-your-credit-card-billing-statement-960246>
3. Tutorialspoint, *Java – Sending Email.* Repéré à <https://www.tutorialspoint.com/java/java_sending_email.htm>

**Algorithme AES**

1. Benvenuto, C.J. (2012). *Galois Field in Cryptography*. Repéré à <https://sites.math.washington.edu/~morrow/336_12/papers/juan.pdf>
2. Cartrysse, K., van der Lubbe, J.C.A. (2004). *The Advanced Encryption Standard: Rijndael*. Repéré à <https://math.boisestate.edu/~liljanab/Rijndael-version2.pdf>
3. Computerphile. (22 novembre 2019). AES Explained (Advanced Encryption Standard) - Computerphile [Youtube]. Repéré le 26 janvier 2019 à <https://www.youtube.com/watch?v=O4xNJsjtN6E&feature=share>
4. Daemen, J., Rijmen, V. (2002). *The Design of Rijndael*. Repéré à <https://autonome-antifa.org/IMG/pdf/Rijndael.pdf>
5. Daemen, J., Rijmen, V. (1999). *The Rijndael Block Cipher*. Repéré à <https://csrc.nist.gov/csrc/media/projects/cryptographic-standards-and-guidelines/documents/aes-development/rijndael-ammended.pdf>
6. Federal Information Processing Standards Publications. (2001). Announcing the Advanced Encryption Standard (AES) (Publication no FIPS PUB 197). Repéré à <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf>

**Construction du terminal**

1. Kuman Tech. (2017). 3.5inch LCD Module Documentation. Repéré à <https://mega.nz/#F!HUFQFDwL!fhM7RPrhbMJLsZ7ISLEr0A>
2. Raspberry Pi Tutorials. (2017). *How to use a Raspberry Pi Fingerprint Sensor for Authentication*. Repéré à <https://tutorials-raspberrypi.com/how-to-use-raspberry-pi-fingerprint-sensor-authentication/>

**Conception du «Back-end»**

1. Fabian, M. (2017). *Java library for interfacing with Adafruit fingerprint sensor*. Repéré à <https://github.com/milan-fabian/java-fingerprint-sensor>
2. Raspberry Pi. (2019). Pi4J (Version 1.2) [Librairie Java]. Repéré à <https://www.pi4j.com/1.2/index.html>

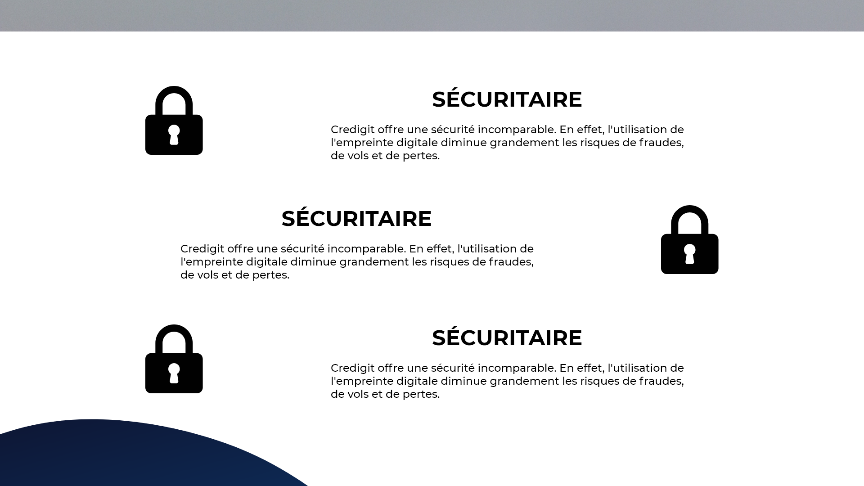
# Prototypes de l’application

## Prototypes

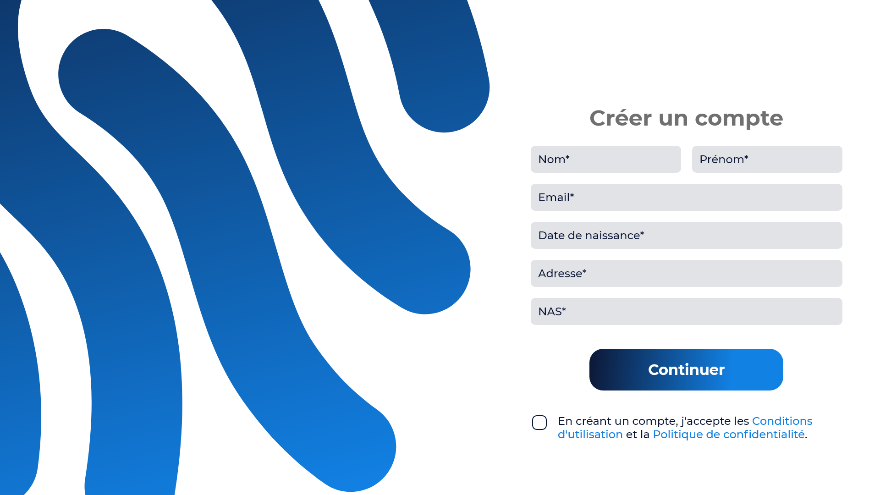
Vues de l’inscription :



L’image ci-dessus représente la page d’accueil que le client verra lorsqu’il arrivera sur un ordinateur d’un point de service pour s’inscrire.



L’image ci-dessus représente une explication brève des forces de notre service afin de convaincre le client de l’adopter.



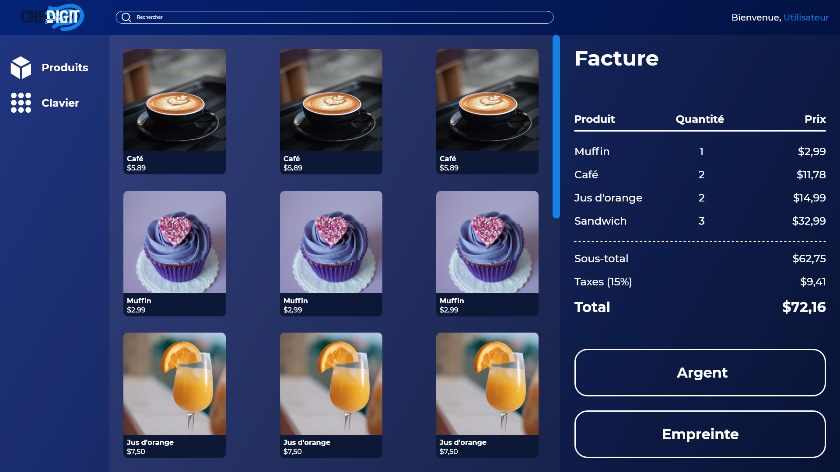
L’image ci-dessus représente la fiche d’inscription d’un nouveau client afin que celui-ci puisse se créer un compte.

Les 3 images précédentes représentent une seule vue qui « scroll ».

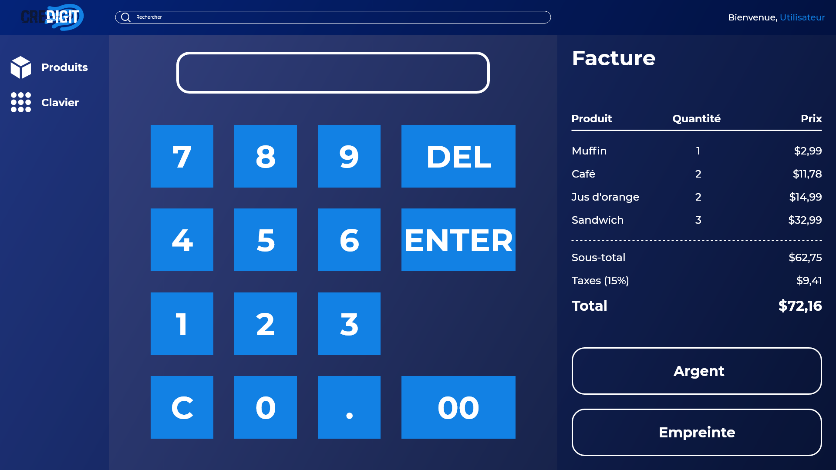
Vues du POS (Caisse enregistreuse) :



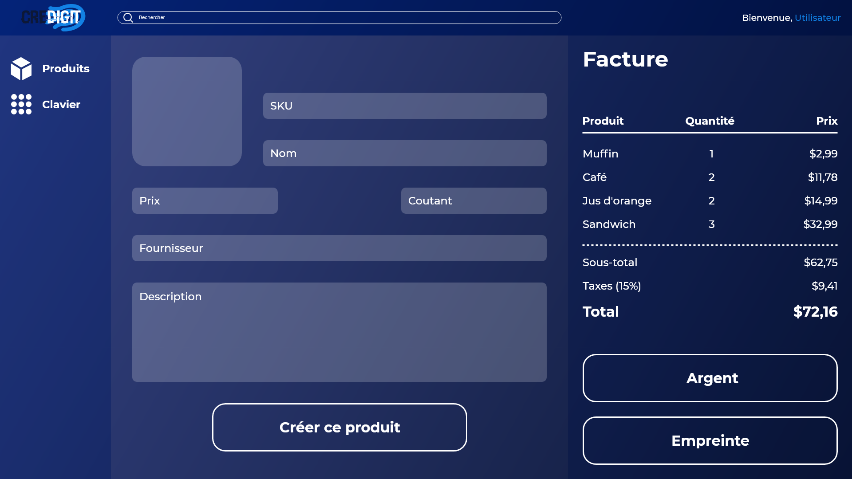
L’image ci-dessus représente la page d’accueil du POS. C’est sur cette page que le caissier va pouvoir se connecter en débutant son quart de travail.



L’image ci-dessus représente le tableau de bord du POS. Le commis peut, de cette vue, sélectionner des produits et faire payer le client.



L’image ci-dessus représente une autre version du tableau de bord du POS. Cette version présente un clavier afin d’effectuer des opérations plus précises avec la caisse.



L’image ci-dessus représente l’interface qui permet d’ajouter un nouveau produit dans la liste par le commerçant.

Vues du terminal :



L’image ci-dessus représente ce que le client va voir sur le terminal lorsque ses articles sont ajoutés à sa commande.



L’image ci-dessus représente ce que le client va voir sur le terminal lorsque celui-ci doit procéder au paiement.



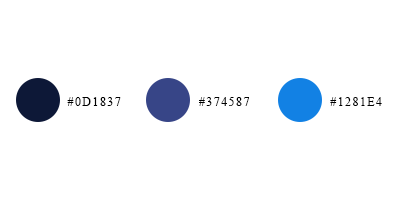
L’image ci-dessus représente ce que le client va voir sur le terminal lorsque la transaction est effectuée.

## Principaux objets

Logo :



Couleurs principales :



# Scénarios ou user stories

*\*\*Les scénarios sont placés dans l’ordre chronologique de leur réalisation\*\**

|  |  |
| --- | --- |
| **1** | |
| Acteur ou rôle : | Le client |
| Scénario ou story : | En tant que client je veux pouvoir me créer un compte à l’un des bureaux de Bank-era afin de pouvoir utiliser le service. |
| Détail ou description : | 1. Le client entre ses informations personnelles 2. L’empreinte digitale du client est ajoutée à son dossier 3. Le client répond à deux questions de sécurité 4. Le tout est ajouté à la base de données 5. Le client reçoit un courriel de confirmation avec son numéro de compte |
| Tests d’acceptation : | Un client potentiel réussit à se créer un compte, il reçoit un courriel de confirmation/bienvenue et toutes ses informations sont bien dans la base de données. |
| Complexité : | 5 |
| Effort : | 7j/homme |
| Commentaires : |  |

|  |  |
| --- | --- |
| **2** | |
| Acteur ou rôle : | Client |
| Scénario ou story : | En tant que client je veux pouvoir désactiver mon compte afin de garder le contrôle de quand ma carte est active. |
| Détail ou description : | 1. Formulaire de désactivation 2. Vérification du solde |
| Tests d’acceptation : | Un client désactive son compte et nous pouvons voir le changement dans la base de données. |
| Complexité : | 1/2 |
| Effort : | 1j/homme |
| Commentaires : | Formulaire similaire à l’inscription, mais avec moins de champs à remplir. |

|  |  |
| --- | --- |
| **3** | |
| Acteur ou rôle : | Vendeur |
| Scénario ou story : | En tant que vendeur, je veux pouvoir ajouter ou enlever des produits de la commande d’un client afin de faciliter le calcul de la facture |
| Détail ou description : | 1. Ajouter des produits à la transaction 2. Enlever des produits de la transaction 3. Annuler la transaction 4. Calculer le sous-total 5. Calculer les taxes 6. Calculer le total de la vente |
| Tests d’acceptation : | Un vendeur peut ajouter un produit, l’ôter, et annuler la transaction. Le calcul des taxes et du total se fait sans erreurs. |
| Complexité : | 2 |
| Effort : | 4j/homme |
| Commentaires : |  |

|  |  |
| --- | --- |
| **4** | |
| Acteur ou rôle : | Le vendeur |
| Scénario ou story : | En tant que vendeur je veux disposer d’une interface graphique afin de faciliter l’utilisation du point de service |
| Détail ou description : | 1. Le vendeur peut ajouter des items à la transaction grâce à l’interface graphique 2. Le vendeur est aidé par les instructions sur l’interface graphique |
| Tests d’acceptation : | Le vendeur réussit à utiliser l’interface graphique pour aller modifier une (nouvelle) facture. |
| Complexité : | 2 |
| Effort : | 6j/homme |
| Commentaires : |  |

|  |  |
| --- | --- |
| **5** | |
| Acteur ou rôle : | Vendeur |
| Scénario ou story : | En tant que vendeur, je veux ajouter ou supprimer des produits afin d’avoir une liste actualisée des produits que je dispose en magasin. |
| Détail ou description : | 1. Formulaire de création de produit 2. Accès à la liste des produits 3. Option de supprimer un produit |
| Tests d’acceptation : | Le vendeur crée un nouveau produit et peut le voir dans la liste de ses produits. Il peut ensuite le supprimer de la liste. |
| Complexité : | 2 |
| Effort : | 5j/homme |
| Commentaires : | On parle ici d’un inventaire des différents produits offerts et non d’inventaire des quantités disponibles en magasin. |

|  |  |
| --- | --- |
| **6** | |
| Acteur ou rôle : | Vendeur |
| Scénario ou story : | En tant que vendeur, je veux avoir accès à un clavier affiché à l’écran afin de pouvoir entrer des montants ou des codes de produits. |
| Détail ou description : | 1. Interface avec clavier 2. Espace affichant le texte 3. Déterminer si le nombre entré est un montant ou un code |
| Tests d’acceptation : | Le vendeur tente d’ajouter un produit à la commande à l’aide du clavier et le produit est bel et bien ajouté et le vendeur entre le montant donné par le client pour payer. |
| Complexité : | 1 |
| Effort : | 2j/homme |
| Commentaires : |  |

|  |  |
| --- | --- |
| **7** | |
| Acteur ou rôle : | Vendeur |
| Scénario ou story : | En tant que vendeur, je veux avoir un système de connexion afin que seul mes employés puissent utiliser la caisse. |
| Détail ou description : | 1. Interface de connexion 2. Champs pour nom d’utilisateur 3. Champs pour mot de passe 4. Sauvegarder les utilisateurs dans une base de données |
| Tests d’acceptation : | On crée un nouveau compte caissier et on doit être ensuite en mesure de se connecter avec ce compte. |
| Complexité : | 5 |
| Effort : | 6j/homme |
| Commentaires : |  |

|  |  |
| --- | --- |
| **8** | |
| Acteur ou rôle : | Client |
| Scénario ou story : | En tant que client, je veux payer avec mon empreinte digitale afin d’éviter de transporter une carte de crédit sur moi en tout temps. |
| Détail ou description : | 1. L’empreinte digitale est enregistrée lors de la création du compte du client 2. L’empreinte digitale est lue lors de la transaction 3. L’empreinte digitale est utilisée pour trouver le compte du client et effectuer la transaction 4. Le terminal doit être construit |
| Tests d’acceptation : | Le client peut effectuer une transaction à l’aide de son empreinte digitale. |
| Complexité : | 5 |
| Effort : | 6j/homme |
| Commentaires : |  |

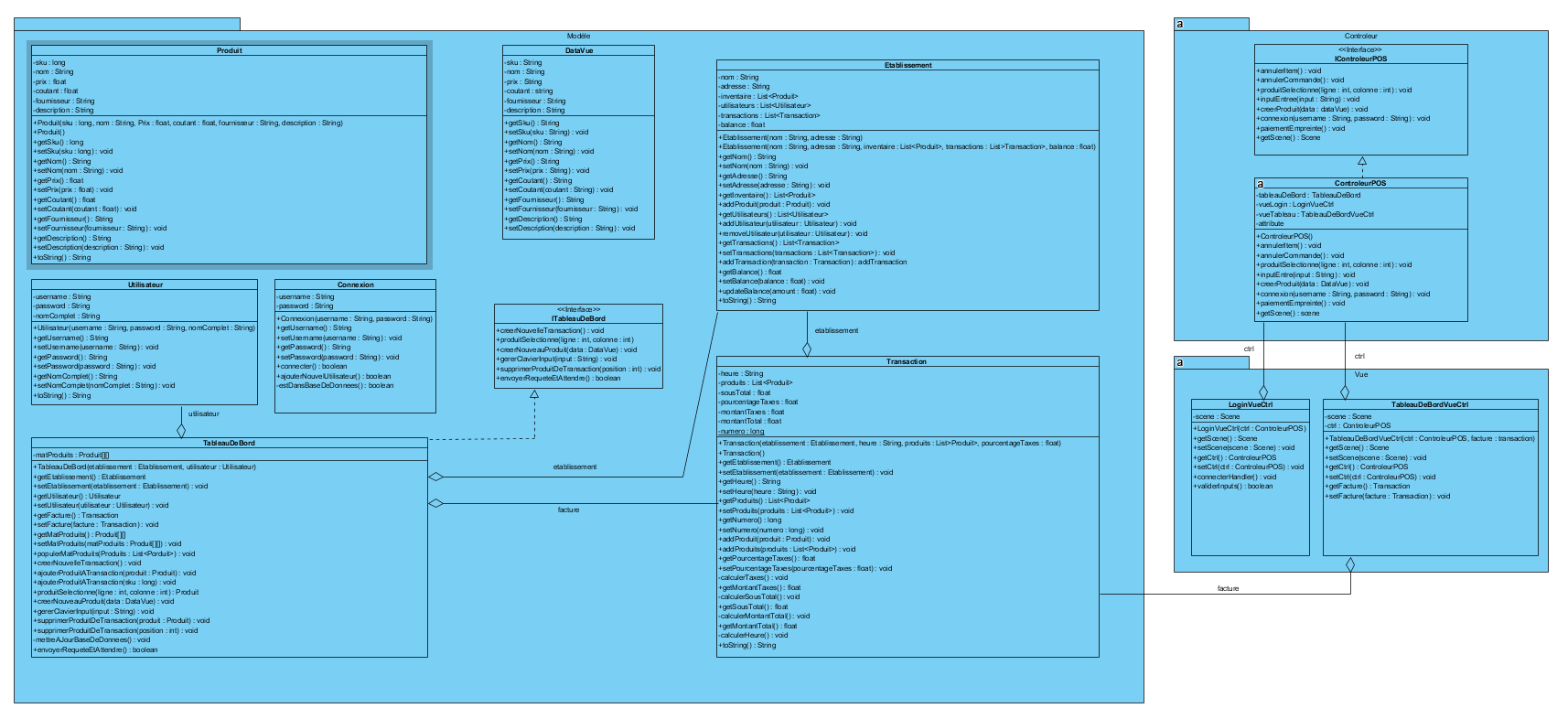
|  |  |
| --- | --- |
| **9** | |
| Acteur ou rôle : | Client |
| Scénario ou story : | En tant que client, je veux être guidé par une interface graphique lors du paiement afin d’effectuer la transaction |
| Détail ou description : | 1. Afficher les produits de la transaction 2. Afficher le total de la transaction 3. Indiquer le moment où il faut payer 4. Indiquer le statut de la transaction |
| Tests d’acceptation : | Une interface graphique est affichée lors de la transaction et affiche les bonnes informations. |
| Complexité : | 1 |
| Effort : | 4j/homme |
| Commentaires : |  |

|  |  |
| --- | --- |
| **10** | |
| Acteur ou rôle : | Client |
| Scénario ou story : | En tant que client je veux recevoir une facture suivant chaque achat afin de garder une trace des produits que j’ai acheté ainsi que le coût de la transaction. |
| Détail ou description : | Suite à son achat le client reçoit une facture contenant les informations suivantes :   1. Le lieux et l’heure de la transaction 2. Les informations du vendeur qui gère le point de service 3. Une liste des produits achetés 4. Le sous-total, les taxes, ainsi que le total de la transaction. |
| Tests d’acceptation : | Un client fait un achat et l’on vérifie que ce client reçoit bien un courriel avec les informations pertinentes. |
| Complexité : | 3 |
| Effort : | 3j/homme |
| Commentaires : |  |

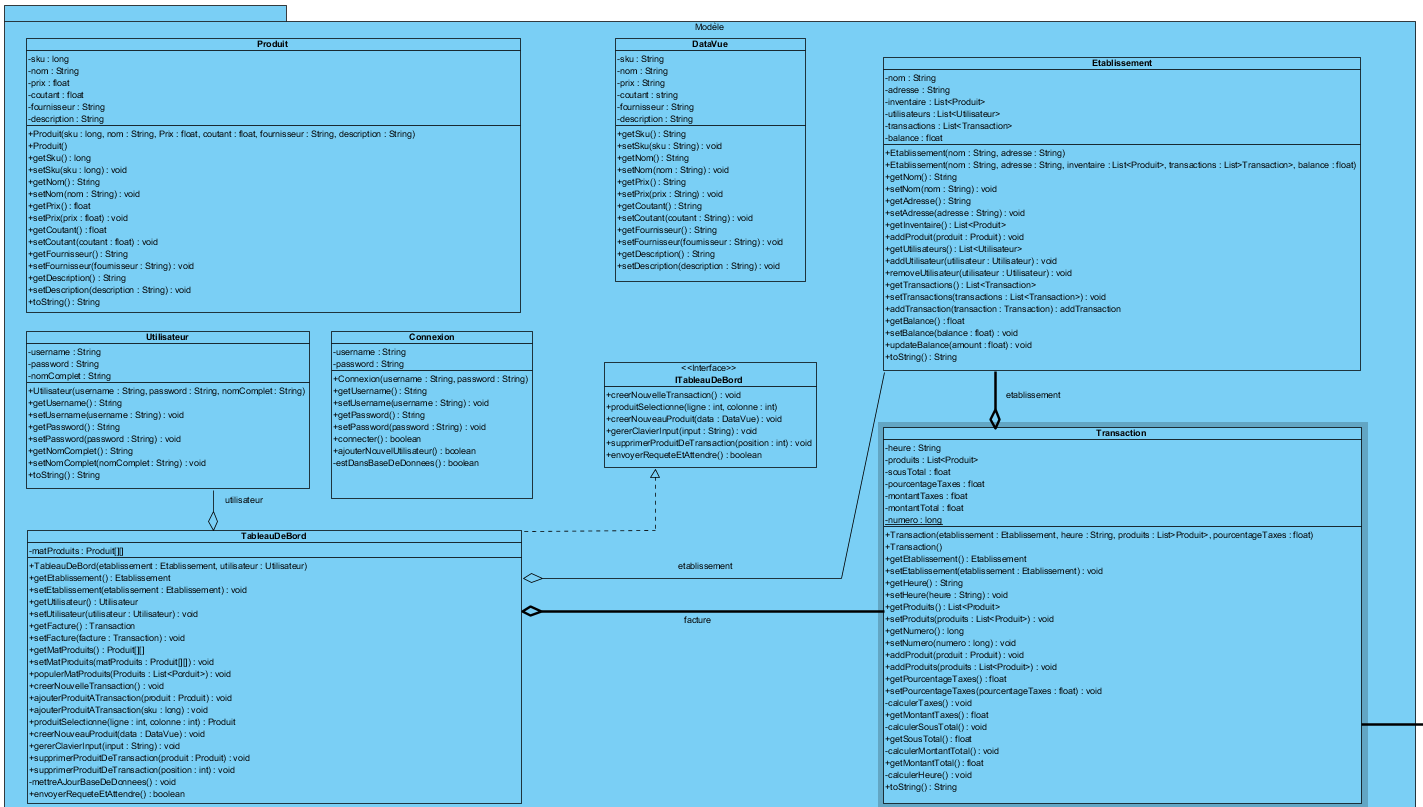
|  |  |
| --- | --- |
| **11** | |
| Acteur ou rôle : | Client |
| Scénario ou story : | En tant que client, je veux avoir une confirmation sonore et visuelle que le terminal a bien pris mon empreinte afin de savoir si le paiement a été accepté/refusé. |
| Détail ou description : | 1. Le terminal émet un son selon l’état de la transaction 2. La couleur d’une lumière sur le terminal change selon l’état de la transaction. |
| Tests d’acceptation : | Le client met son doigt sur le capteur et une confirmation est vue/entendue. |
| Complexité : | 1 |
| Effort : | 2j/homme |
| Commentaires : |  |

|  |  |
| --- | --- |
| **12** | |
| Acteur ou rôle : | Client |
| Scénario ou story : | En tant que client, je veux que mes données soient encryptées afin que je ne puisse pas me les faire voler. |
| Détail ou description : | 1. Générer une clé 2. Encrypter les données (Informations du compte et transactions) avec AES 3. Stocker les données encryptées sur la base de données |
| Tests d’acceptation : | Les informations des clients et les transactions dans la base de données sont indéchiffrables mais peuvent être utilisées par le client lors d’une transaction. |
| Complexité : | 10 |
| Effort : | 14j/homme |
| Commentaires : |  |

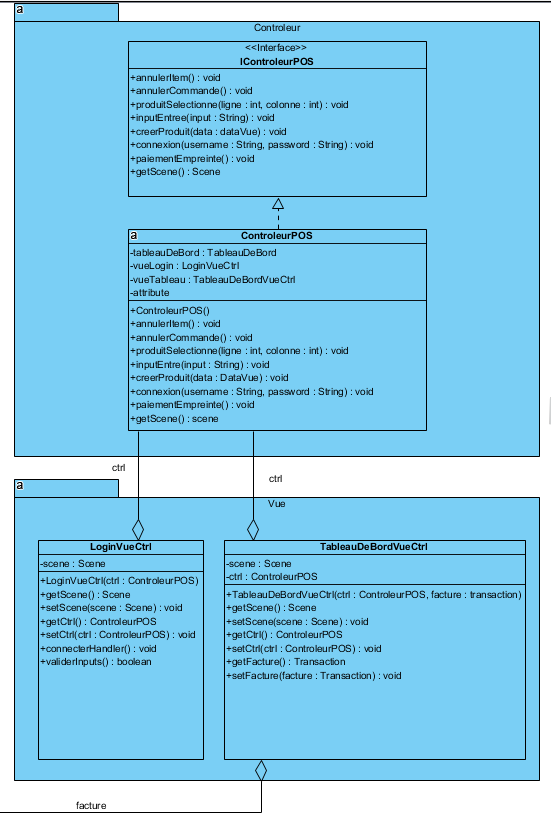
# Diagramme de classes

UML du POS:

*L’image ci-dessus représente la vue globale de l’UML du POS*

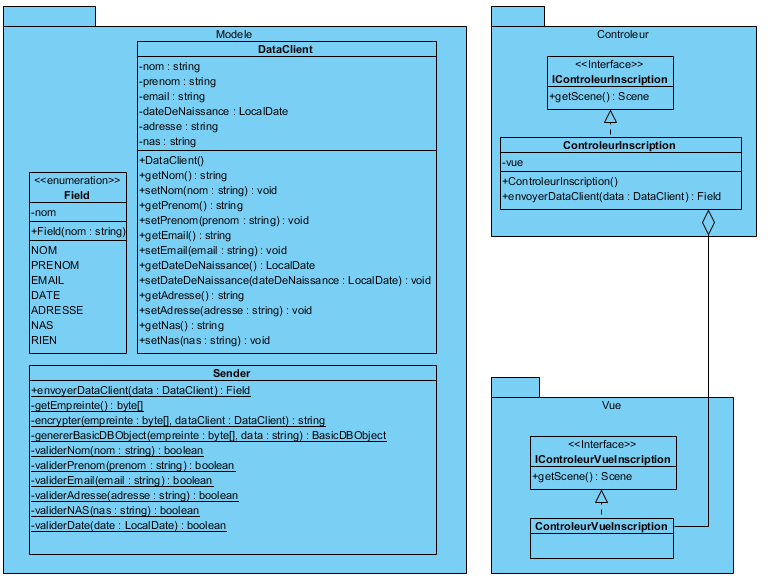


*L’image ci-dessus représente le package Modèle du POS*



*L’image ci-dessus représente les packages Vue et Contrôleur du POS*

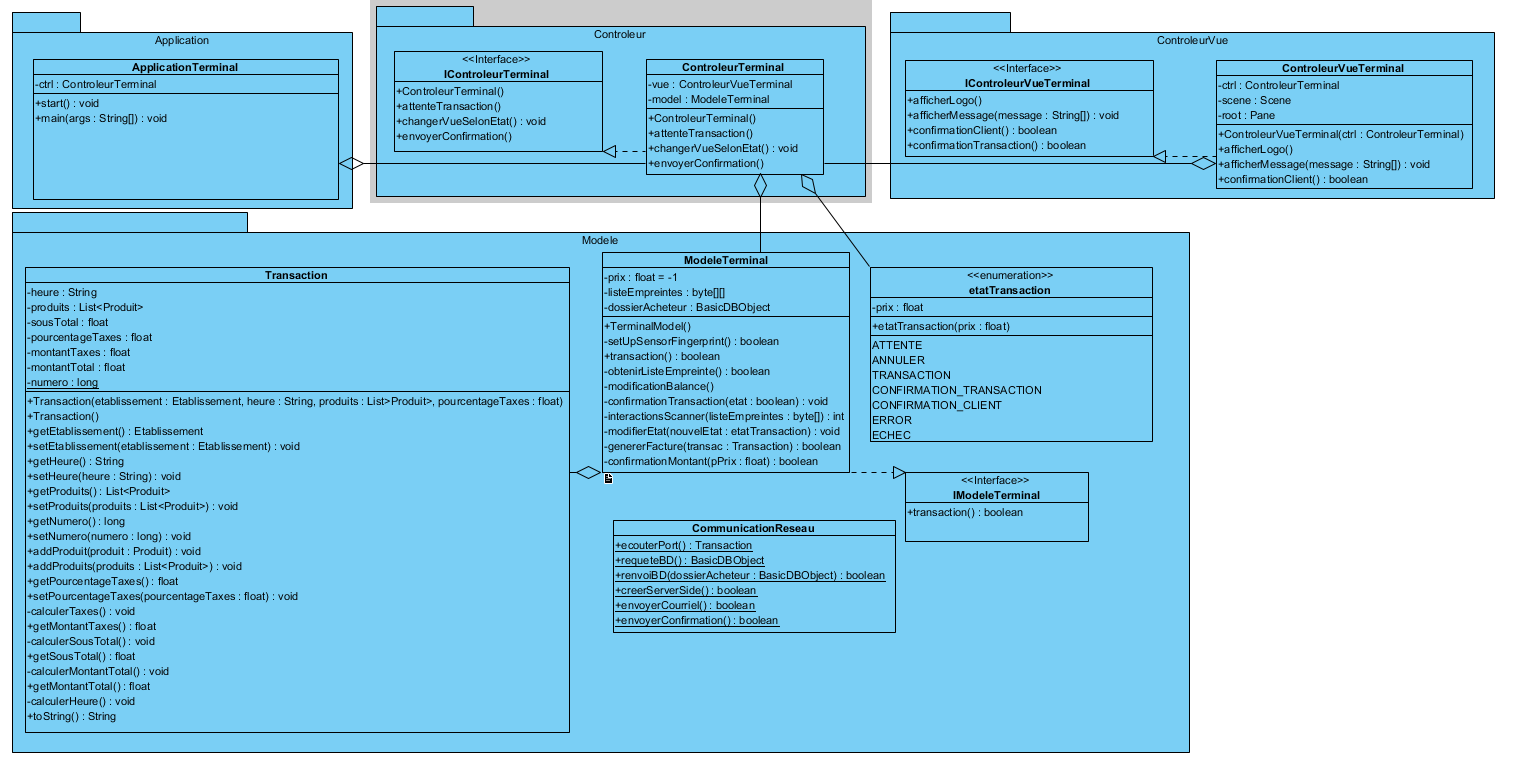
UML du formulaire d’inscription:



*L’image ci-dessus représente le modèle MVC qui servira à l’inscription des clients au service.*

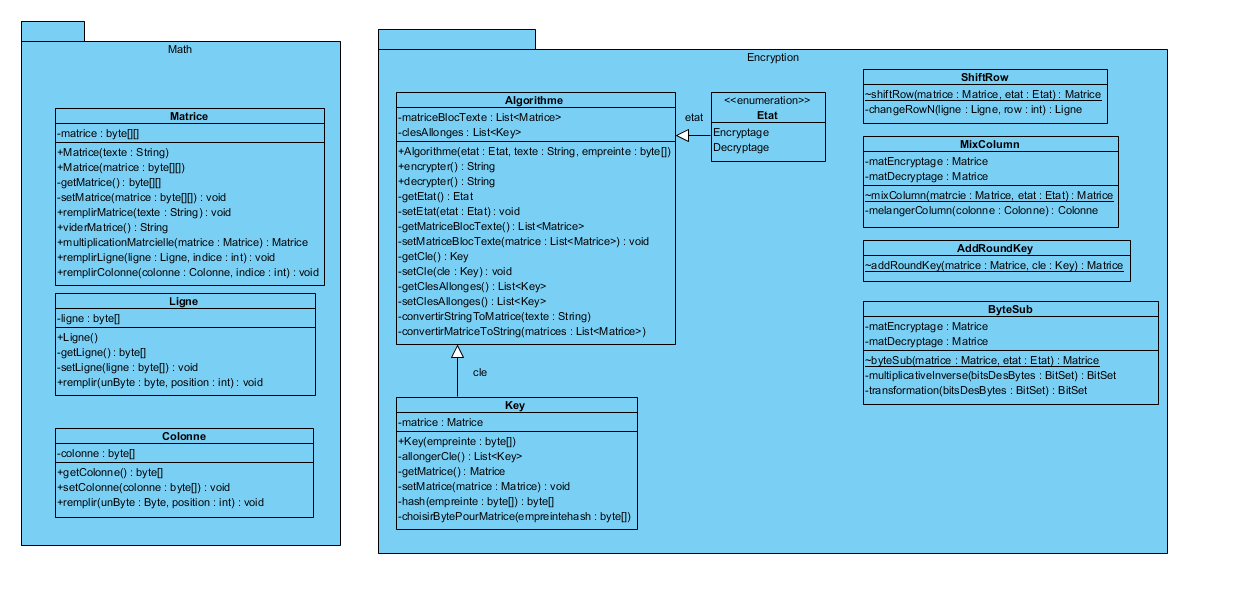
*¸*

UML du terminal:



*L’image ci-dessus représente le modèle MVC du terminal de paiement.*

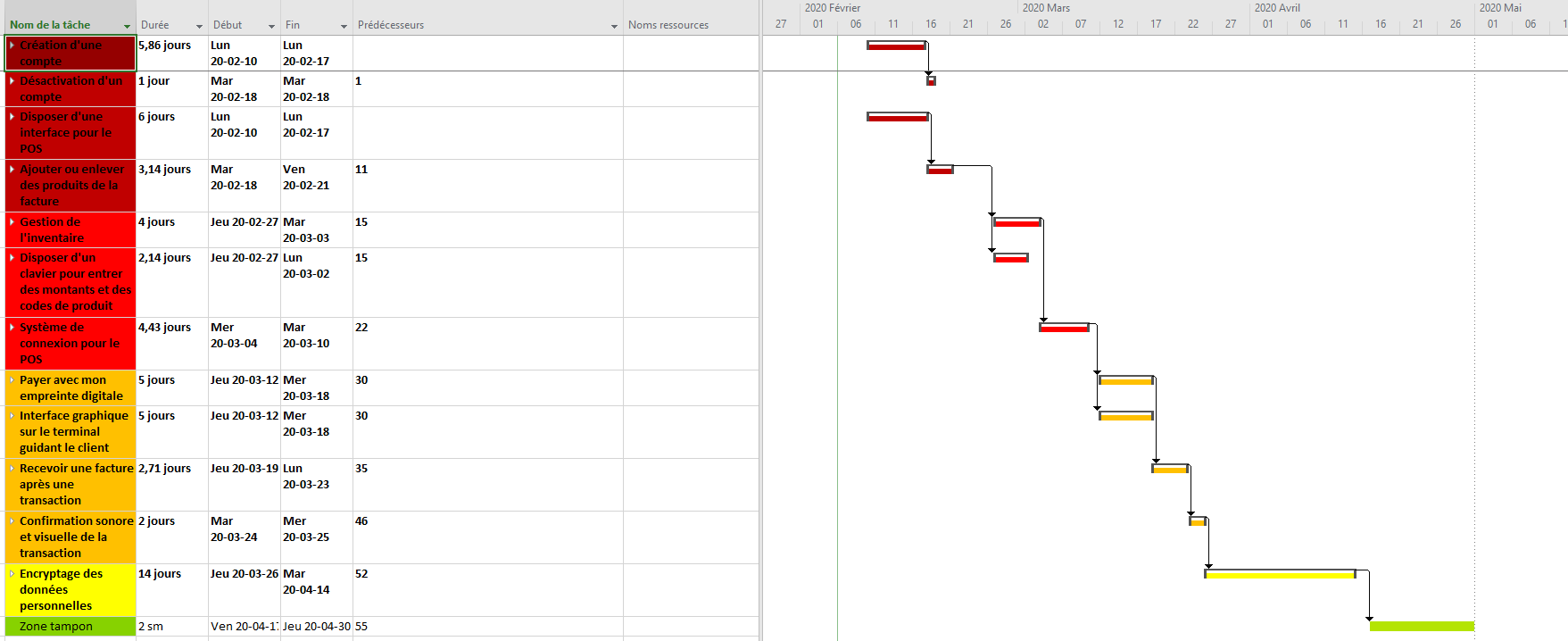
UML de l’AES:



*L’image ci-dessus représente le modèle MVC de l’AES.*

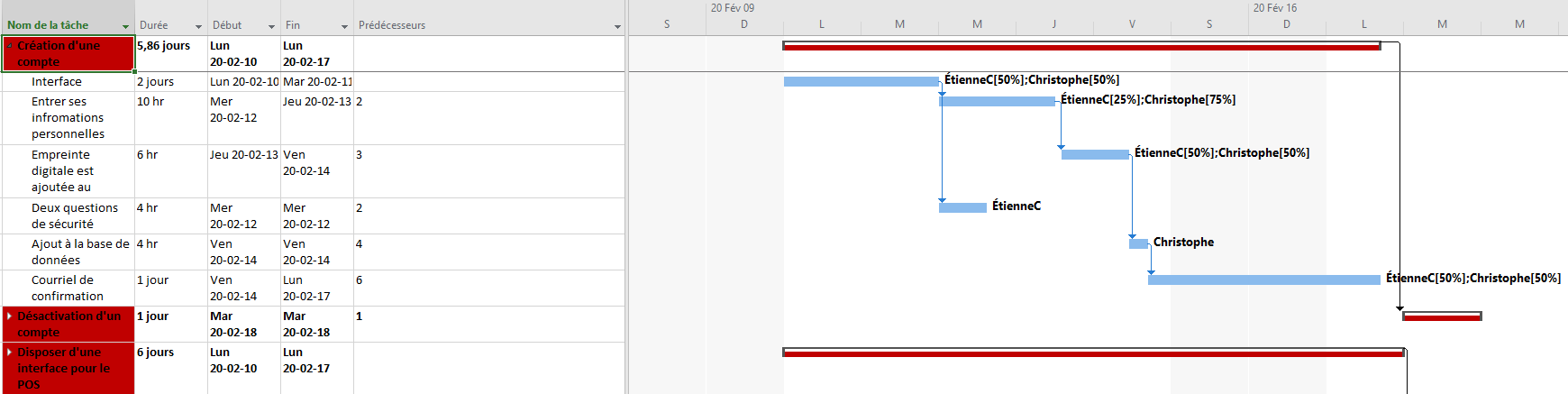
# Échéancier

Échéancier complet:



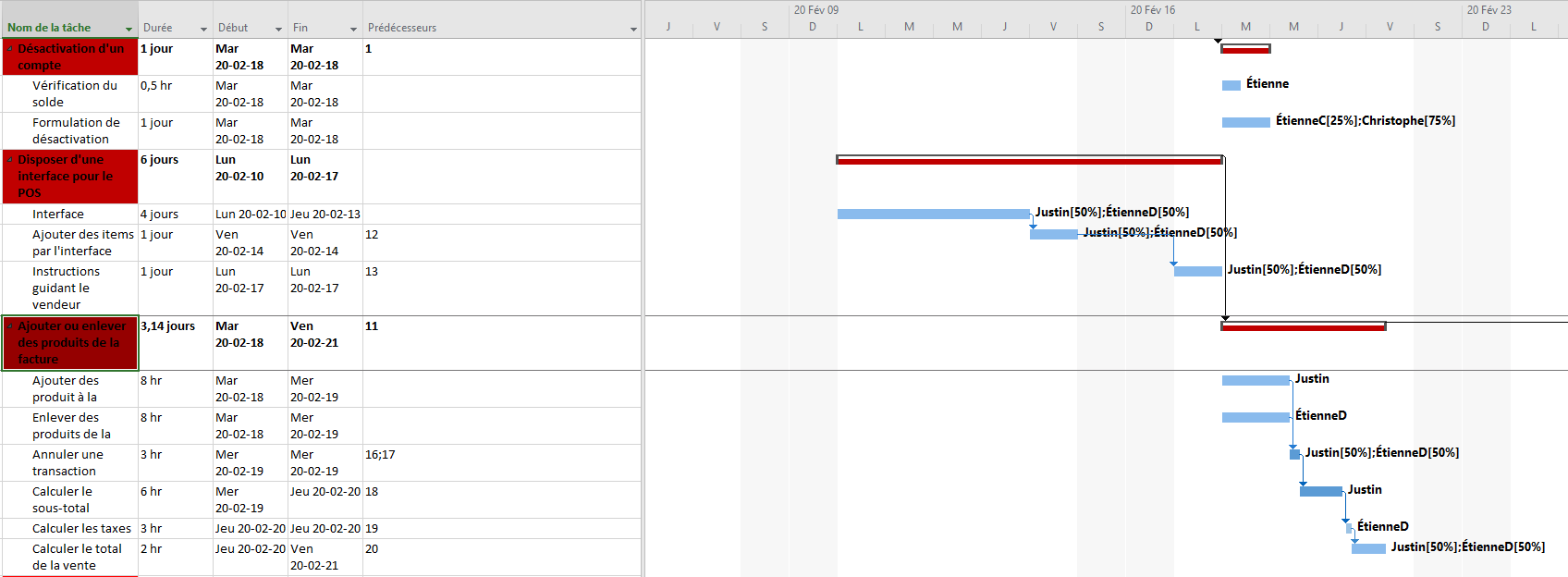
*Cette image représente l’horaire jusqu’à la fin du projet*

Échéancier du scénario 1:



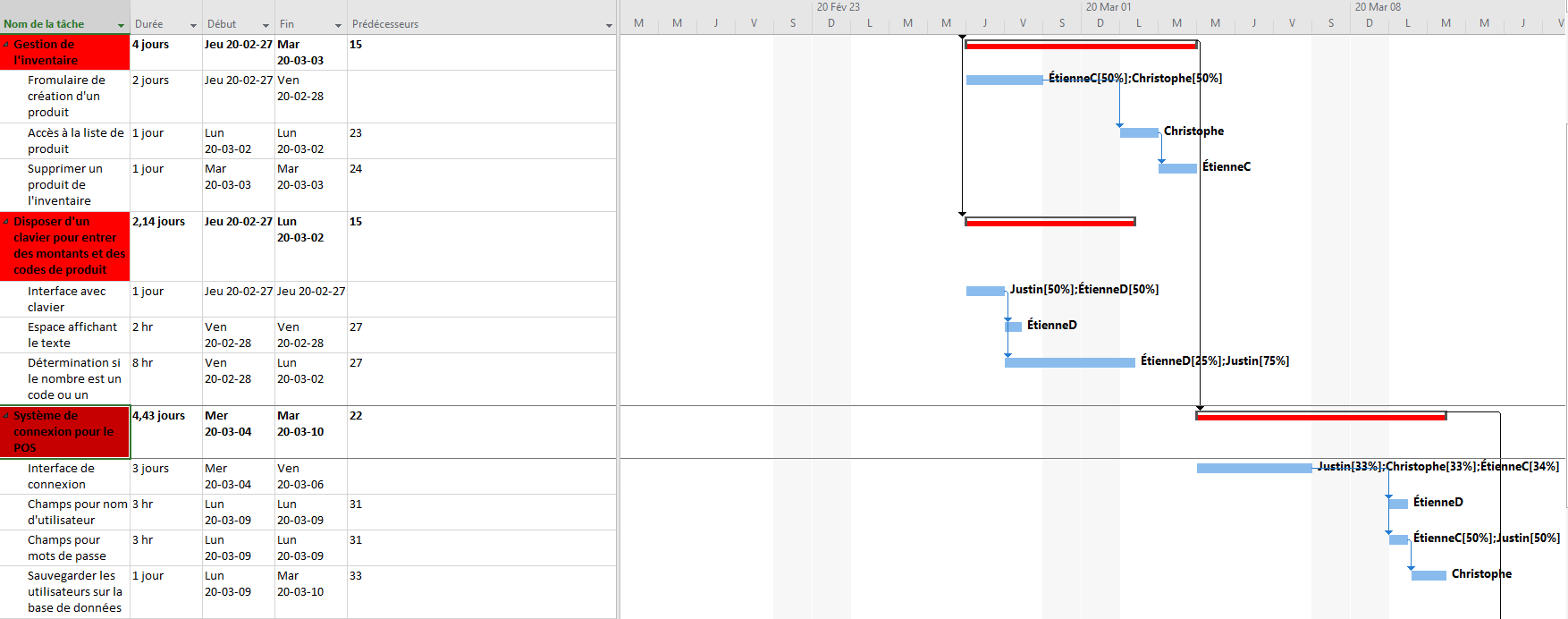
*Cette image focus sur le 1er scénario.*

Échéancier des scénarios 2, 3 et 4:



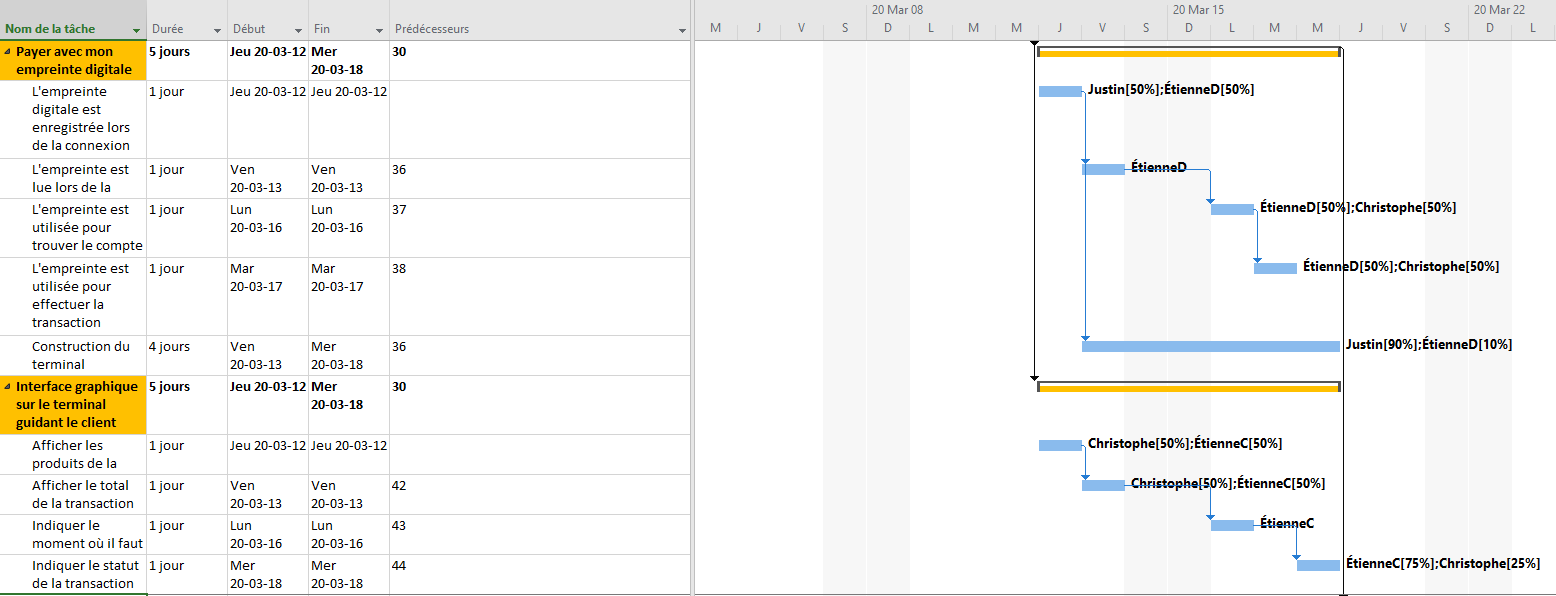
*Cette image focus sur les scénarios 2, 3 et 4 ce qui représente la fin du 1er sprint.*

Échéancier des scénarios 5, 6 et 7:



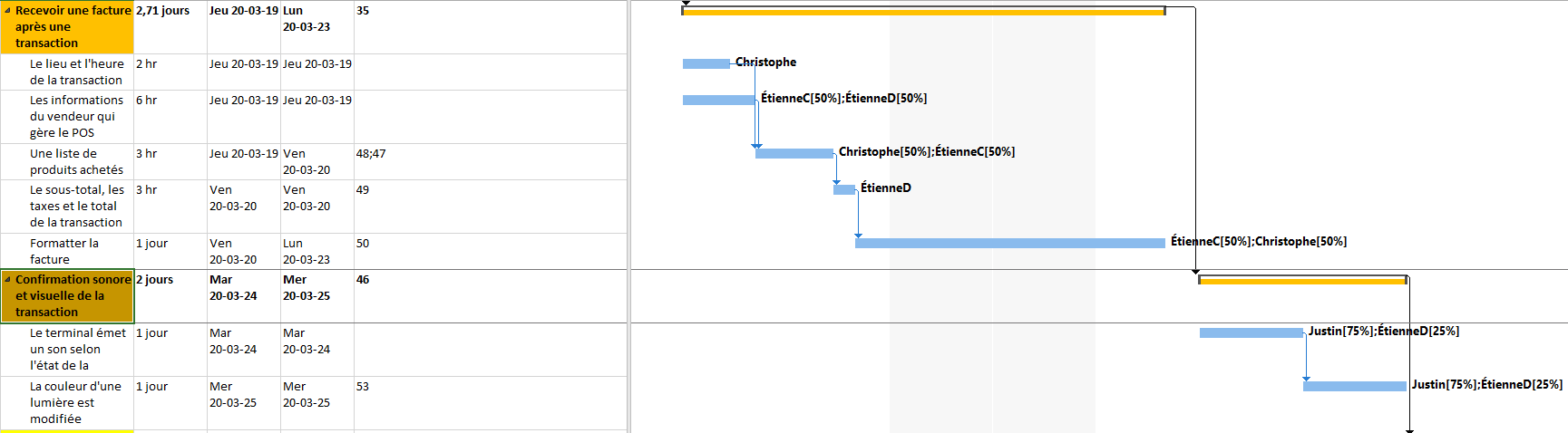
*Cette image focus sur les scénarios 5, 6 et 7 ce qui représente le 2e sprint.*

Échéancier des scénarios 8 et 9:



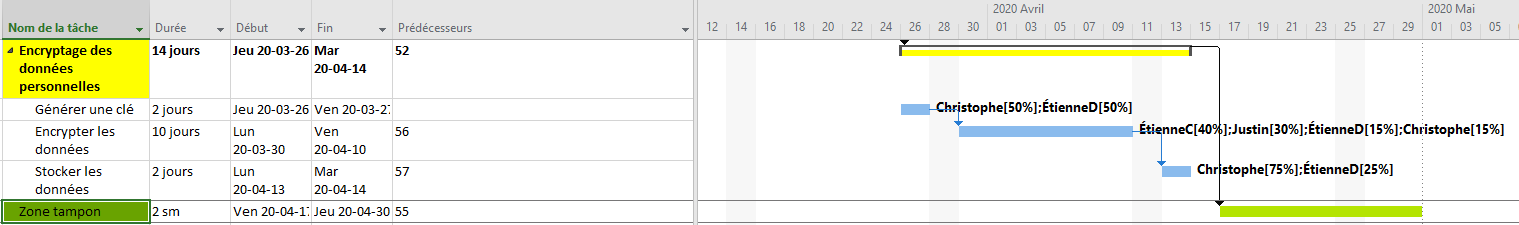
*Cette image focus sur les scénarios 8 et 9.*

Échéancier des scénarios 10 et 11:



*Cette image focus sur les scénarios 10 et 11 ce qui marque la fin du 3e sprint.*

Échéancier des scénarios 12 et 13:



*Cette image focus sur les scénarios 12 et 13 ce qui marque les sprints 4 et 5.*