# 

# 

# 

# 

# 

# Partie 2 - Document sur les exigences

Nom système: **e-knox**

Presenté par:

Marieme Bousso

Haby Diop

Farma Kane

Abdoulaye SAMBE

# 

# 1.Objectifs du système

Le système bancaire que nous allons concevoir se présente dans un contexte de développement accru des Fintech. Les particuliers, les banques mais aussi leurs clients ont besoin de systèmes fiables et flexibles mais par-dessus tout sécurisés pour assurer la bonne conduite de leurs transactions et des annexes qui pourraient être développés à l’avenir.

Ce projet consiste en la mise en place des éléments du «core banking» axé sur le développement des programmes au cœur du service financier (gestion des transactions – des comptes ...). Les éléments de sécurité sont d’ores et déjà intégrés dès la conception et des premières lignes de codes pour garantir une sécurité inhérente au système.

Nous avons choisi d’implémenter notre système par le biais de Java qui jouit d’un écosystème large et mature mais tout aussi d’une communauté très active. Il est notamment très utilisé dans ce contexte de service financier pour sa performance et sa résilience.

L'objectif de notre système **e-knox** est de permettre à un utilisateur de créer un compte avec un ensemble d’informations permettant de l'identifier telles que son NIN, son nom, son prénom, sa date de naissance… L’ensemble des transactions effectuées par chaque utilisateur devraient être conservées par le système et restitué pour des besoins d’audit ou pour les besoins de l’utilisateur. Les transactions doivent être ACID et signées(notion de signature numérique - confidentialité - intégrité) pour garantir leur intégrité.

# 2. Exigences fonctionnelles

Méthode MoSCoW:

M: Must have this

S: should have this

o: NA

C:Could have this if does not affect anything else

W: won’t have this but would like in the future

Liste des exigences fonctionnelles:

* création de compte
* authentification des utilisateurs
* consulter solde
* consulter historique des transactions
* transfert d’argent
* dépôt d’argent
* retrait
* services de paiement

MSCW

| Exigence fonctionnelle | Type d’utilisateurs | Actifs | Importance | User Story |
| --- | --- | --- | --- | --- |
| Cas de création de compte | Personne disposant d’un numéro de téléphone ayant au moins 18 ans | Le numéro de téléphone, le nom, le prénom, le numéro d’identifiant(ID), date et lieu de naissance, genre. | M | L’utilisateur renseigne les informations nécessaires à la création d’un compte. En effet, sans cette étape, notre application n’existe pas. |
| Authentification des utilisateurs | Toute personne disposant d’un compte e-knox | L’identifiant et mot de passe | M | Les utilisateurs doivent être identifiés de façon unique pour le bon fonctionnement de notre application. Désormais, on peut identifier qui est qui et qui fait quoi? |
| Dépôts d’argent | Un client e-knox et un agent | Le montant à déposer et le numéro de téléphone du concerné | M | L’utilisateur peut déposer et épargner de l’argent sur son compte. |
| Consultation solde | Toute personne disposant d’un compte e-knox | L’argent disponible au niveau du compte | M | Le client vérifie le montant qu’il dispose au niveau de son compte. |
| Transfert d’argent | Toute personne disposant d’un compte e-knox | Le montant à envoyer, le numéro du destinataire et le solde | M | Le client e-knox doit pouvoir transférer de l’argent à n’importe quel client e-knox tant que le solde de son compte le lui permet |
| Retrait d’argent | Tout client e-knox et un agent | Le montant à retirer , le numéro de téléphone du concerné, le solde | M | Le client doit pouvoir retirer de l’argent de son compte tant que le solde le lui permet. |
| Paiement | Tout client e-knox | Le montant à payer | S | L’application doit pouvoir permettre au client de pouvoir effectuer un paiement en achetant ou recevoir un paiement en vendant |

# 3. Analyse des menaces

Menaces sur le système

| **Type d’attaque** | **Motivations** | **Ressources** | **Capacités** | **Attaquant** |
| --- | --- | --- | --- | --- |
| Injection SQL | Récupérer les données utilisateurs | Manipulation de la base de données backend | Connaissances en JS et en SQL | Hacker |
| Usurpation d’identité | Accès à des données personnelles(aux informations des utilisateurs) | identifiants de l’utilisateur | Connaissance en exfiltration de données | Hacker |
| DDos | Exploiter les failles du système | botnet | * Connaissance en réseau * Pouvoir lancer des requêtes vers le serveur | Hacker |
| Chevaux de Troie bancaire | Téléchargement d'applications crackées | Voler les informations d'identification bancaires des utilisateurs | Charger toutes sortes de programmes malveillants sur votre système en faisant office de passerelles | Hacker |
| Attaques par phishing | * Récupérer des données personnelles * Faire de l’argent | * enregistreur de frappes * publicités * adresses mail/numéros SMS | Connaissance en “phishing” | Hacker |
| Transactions non-autorisée | MFA | Accès de l’utilisateur | Limitée | Usurpateur d’identité |

Non-menaces sur le système:

* Les composantes principales de notre système sont destinées à être exécutées sur des serveurs hébergés dans notre propre infrastructure, de même que les données utilisateurs.
* Le code des composants logiciels est rendu inaccessible. ( code propriétaire - code rendu impénétrable)

# 

# 

# 4. Objectifs de sécurité

À la création du compte, la personne a besoin de renseigner des informations telles que son nom,son prénom, son numéro de téléphone, son NIN, date et lieu de naissance. En effet, ces actifs sont des données de type confidentiel. Donc, si ces informations ne sont pas bien protégées, un problème de confidentialité. Avec les attaques par phishing, le hacker peut avoir des informations confidentielles sur le client e-knox.

Ensuite, chaque utilisateur doit être identifié de manière unique . Ceci nous permettra de savoir ce que l'utilisateur a eu à faire. Ainsi, nous pourrons garantir l'intégrité des données savoir si un utilisateur a eu à faire des modifications. Comme exemple, on a un utilisateur victime d’une injection SQL; l’injection SQL remet en question l’intégrité des données.

Nous notons aussi les transactions telles que les dépôts d'argent et les transferts d'argent. Avec ces transactions, le montant nécessaire demeure une information critique. On pourrait être confronté à un problème de disponibilité de service. Car notre système pourrait faire face à des pannes ralentissant ainsi le système dans le cas des attaques DDOS. On peut rencontrer un problème d'intégrité si toutefois une personne externe intercepte les transactions et en fait mauvais usage si on est victime du Chevaux de Troie bancaire ou de transactions non autorisées.

Avec l'option consultation solde, le solde est l'information critique. Car, il n’y a que le concerné qui devrait avoir accès à son solde. Ce qui pose un problème de confidentialité dans le cas des attaques par phishing ou d’usurpation d’identité.

Après toutes ces analyses, nous allons garantir à nos clients l'intégrité et la confidentialité de leurs données pour le respect de la vie privée et la disponibilité des ressources pour un service en temps réel.