Rapport de Transaction Bancaire



**Présentés par :**

Khlifi Ayoub

Zouaghi Firas

Belaazi Intissar

Azzouzi Ahmed

Sommaire

Objectifs……………………………………………………...……3

Description générale du projet…………………………..3

Spécifications fonctionnelles……………………………..3

Spécifications techniques…………………………………..4

Réalisations…………………………………………………………5

Difficulté et Solutions…………………………………………..7

Apport …………………………………………………………………8

Objectif

Pour appliquer les connaissances acquises pendant le cours de programmation système et réseau (socket, multithreading, synchronisation), nous sommes amenés à réaliser un projet informatique qui gère les transactions bancaires.

Description générale du projet

Pour répondre à cet objectif, nous avons développés une petite application graphique codée en python coté client et un serveur qui va agir à la suite des demandes des clients relié à une base de données.

Spécifications fonctionnelles

Le projet porte sur trois volets :

•Profil client

•Profil serveur (qui joue l’intermédiaire entre le client et la base de données)

•Base de données

Profil client

A travers cette application, le client est capable de :

•S’authentifier

• Faire des transactions bancaires (ajout, retrait)

• Recevoir une facture à la fin de chaque transaction.

Profil serveur

Le serveur est capable de :

•S’authentifier

• Consulter la liste des comptes

• Consulter la facture d’un compte

• Consulter l’historique des transactions

Base de données

Accédé uniquement par le serveur afin de récupérer les données nécessaires pour traiter les requêtes provenues du client et sachant que le serveur n’a pas accès direct et total à la base de données.

Spécifications non fonctionnelles

Les besoins non fonctionnels décrivent toutes les contraintes techniques, ergonomiques et esthétiques auxquelles est soumis le système pour sa réalisation et pour son bon fonctionnement. Et ce qui concerne notre application, nous avons dégagé les besoins suivants :

- La disponibilité : l’application doit être disponible pour être utilisé par n’importe quel utilisateur.

- La sécurité de l’accès aux informations critiques : nous devons prendre en considération la confidentialité des données de clients surtout au niveau de l’authentification. Pour cela nous devons restreindre l’accès à ces informations à l’administrateur.

- La fiabilité : les données fournies par l’application doivent être fiables.

- La convivialité de l’interface graphique : l’application doit fournir une interface conviviale et simple pour tout type d’utilisateur car elle présente le premier contact de l’utilisateur avec l’application et par le biais de celle-ci on découvrira ses fonctionnalités.

- Une solution ouverte et évoluée : l’application peut être améliorée par l’ajout d’autres modules pour garantir la souplesse, l’évolutivité et l’ouverture de la solution.

- La possibilité de retourner au menu principal de l’application à partir de n’importe quelle fenêtre de celle-ci.

Réalisation

Cette partie constitue le dernier volet de ce rapport. Après avoir terminé la phase de spécification et « conception », la solution étant déjà choisie et étudiée, il nous reste que de se décider dans quel environnement nous allons travailler, exposer les choix techniques utilisés et le langage adopté, et présenter l’implémentation et les tests réalisés.

Choix Technique

•choix du langage de programmation

Python est non seulement un langage de programmation puissant conçu pour être sûr, inter plateformes et international, mais aussi un environnement de développement qui est continuellement étendu pour fournir des nouvelles caractéristiques et des bibliothèques permettant de gérer de manière élégante des problèmes traditionnellement complexes dans les langages de programmation classiques tels que le langage c , tels que le multithreading, les accès aux bases des données, la programmation réseau, l’informatique répartie.

Aussi dans ce projet, Kivy est fortement choisi grâce à sa réputation et sa souplesse dans l’utilisation. Kivy est un Framework accéléré OpenGL ES 2 pour la création de nouvelles interfaces utilisateur en python est fortement choisi.

•Choix de l’architecture de l’application

Client/Serveur

Dans l’architecture à trois niveaux, les applications au niveau serveur sont délocalisées, c’est- à-dire que chaque serveur est spécialisé dans une tâche (serveur web/ serveur de base de données par exemple).

Il permet :

- une plus grande flexibilité/souplesse.

- une sécurité accrue car la sécurité peur être définie indépendamment pour chaque service, et à chaque niveau.

- de meilleures performances, étant donné le partage des tâches entre les différents serveurs.

Protocole de communication : TCP

L'un des principaux avantages du protocole TCP/IP est son très haut niveau de fiabilité lors du transfert de données. Même si un paquet d'informations est perdu en suivant un canal particulier, le protocole garanti qu'il arrivera à destination en empruntant un autre chemin

•système de gestion de base de données

SQLite est un système de base de données qui a la particularité de fonctionner sans serveur, on dit aussi "standalone" ou "base de données embarquée". On peut l'utiliser avec beaucoup de langages : PHP, Python, C# (.NET), Java, C/C++, Delphi, Ruby...

L'intérêt c'est que c'est très léger et rapide à mettre en place, on peut s'en servir aussi bien pour stocker des données dans une vraie base de données sur une application pour smartphone (iPhone ou Android), pour une application Windows, ou sur un serveur web.

Une base de données SQLite est bien plus performante et facile à utiliser que de stocker les données dans des fichiers XML ou binaires, d'ailleurs ces performances sont même comparables aux autres SGBD fonctionnant avec un serveur comme MySQL, Microsoft SQL Server ou PostgreSQL

Organisation du travail

Au premier lieu, le travail est subdivisé en 3 sous tâches :

• création de la base de données et les fonctions d’insertion, de mise à jour et select.

• création de l’interface graphique c à d interface du client

• création d’un server

Puis,

•faire la liaison entre ces trois derniers

Difficultés et Solutions

Nous avons rencontré plusieurs difficultés au sein de ce projet.

A titre d’exemple :

Comment faire passer d’une fenêtre à un autre ?

Solution : utiliser screenmanager

Comment utiliser le même socket pour enchainer la communication avec le serveur sans recréer un autre socket car le code est divisé en plusieurs classes selon le nombre de fenêtres et de traitement.

Solution : faire passer le même socket d’une classe vers un autre classe en utilisant un getter.

J’ai nommé une colonne sous le nom ‘transaction’ dans une table dans la base de données SQLite3 ?

Solution : Ne jamais nommer une colonne sous le nom transaction car est un mot réservé dans sqlite3.

Apport

Au terme de ce rapport, nous pouvons conclure que ce projet nous a donné une occasion opportune nous permettant de confronter l’acquis théorique à l’environnement pratique.

Ce travail de conception et de développement d’une application de transactions bancaire nous a été bénéfique sur plusieurs plans : il nous a permis de perfectionner nos connaissances acquises en programmation système et en conception

Tout au long de l’élaboration du projet, nous avons rencontré plusieurs difficultés tant au niveau conceptuel qu’au niveau de la réalisation. Tout de même, nous avons réussi à les surpasser pour présenter en fin de compte une application opérationnelle.