A picture containing text, clipart

Description automatically generatedLogo, icon

Description automatically generated

**Université Abdelmalek Essaadi**

**Ecole Nationale des Sciences Appliquées d'Al-Hoceima**

**Projet**

**JAVA SE**

**Conception et développement**

**D’une application Desktop de gestion**

**De paiement Bancaire**

**Département d’Informatique**

**G.I.2**

**Réalisé par :**

**Chilouh Faouzi**

**Nafid Amine**

**Chemlal Abdelhakim**

**Encadré par :**

**Pr. A. Bahri**

## **Année Universitaire :**

## **2020/2021**

**Table des matières**

Introduction générale2

**Chapitre 1 :** **Contexte générale du projet** 3

I-Introduction 4

II-Contexte générale4

III-Conclusion 5

**Chapitre2 : Étude conceptuelle** 6

I-Introduction 7

II-Conception de la base de données 8

III-Diagrammes de cas d’utilisation 9

IV-Diagramme d’activité 10

V - Conclusion 11

**Chapitre 3 : Réalisation**12

I-Introduction13

II-Environnement de développement 13

III- Interfaces de l’application15

IV - Conclusion 30

Conclusion Générale31

**Introduction générale :**

L’objectif de ce projet consiste à développer une application Desktop de gestion de paiement bancaire affectés qui permet de payer des factures, et de faire des recharges mobiles.

Cette application est réalisée en JAVA SE en respectant le modèle MVC à l’aide de NetBeans IDE et MySQL.

Dans la première partie on va expliquer le cahier des charges de cette application, ensuite nous allons faire une conception détaillée de notre application en présentant les différents modèles des données. En fin, les outils de développement, ainsi qu’un aperçu sur les résultats obtenus.

*Chapitre 1 :*

*Contexte général*

*Du projet*

1. **Introduction :**

Ce chapitre a pour objectif de situer le projet dans son cadre général, commençant par le cahier de charge de notre application, puis les fonctionnalités de cette dernière.

1. **Contexte générale :**

Le but de ce projet est la création d’une application réalisée en java SE permettant la gestion paiement bancaire online, pour payer des factures et faire des recharges.

* **Cahier des charges :**

L’application permet de faire payer les factures par choisir le type de paiement soit avec compte ou avec carte bancaire, l’utilisateur peut choisir aussi le type de facture à payer, ainsi de faire imprimer les informations de paiement.

* **Besoins fonctionnels :**

Les fonctionnalités ciblées de notre application sont les suivantes :

* Authentification
* Payer avec carte bancaire.
* Payer Avec compte bancaire.

1. **Conclusion :**

Au cours de ce chapitre nous avons étudié les besoins fonctionnels de notre application. Cette étude nous aidera dans l’étape de conception.

*Chapitre 2 :*

*Étude conceptuelle :*

1. **Introduction :**

Dans ce chapitre, nous allons reproduire les différents besoins cités précédemment mais sous forme des vues conceptuelles et des diagrammes UML.

* **Modèle MVC :**

L’**MVC** est une façon d'organiser une interface graphique d'un programme. Elle consiste à distinguer trois entités distinctes qui sont, le modèle, la vue et le contrôleur ayant chacun un rôle précis dans l'interface.

Dans l'architecture MVC, les rôles des trois entités sont les suivants.

* **Modèle :** données (accès et mise à jour)
* **Vue :** interface utilisateur (entrées et sorties)
* **Contrôleur :** gestion des événements et synchronisation

**Diagram

Description automatically generated**

Figure : MVC

1. **Conception de la base de données :**

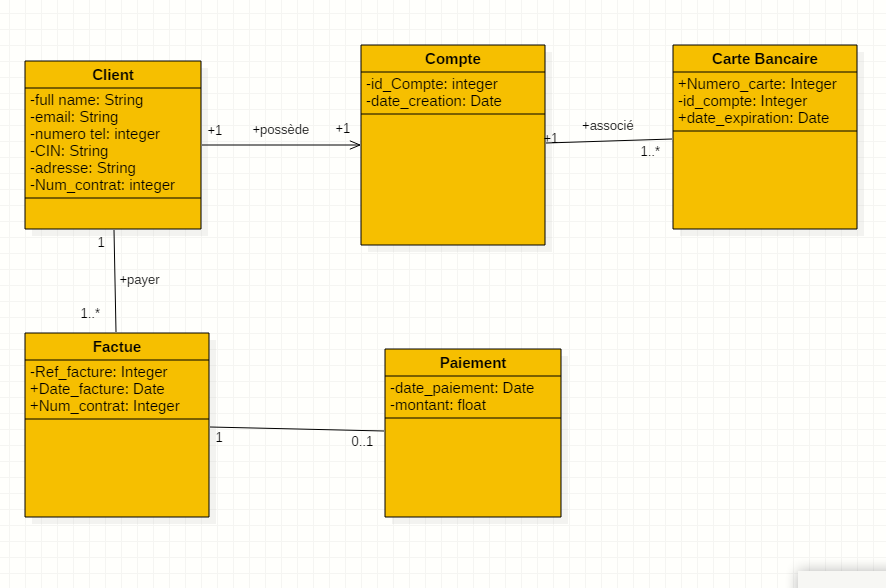
****

Figure 1 : Diagramme De Classe

1. **Diagramme de cas d’utilisation :**
2. **Identification des acteurs :**

L’application dispose de deux types d’utilisateurs :

1. **Diagramme de cas d’utilisation globale :**

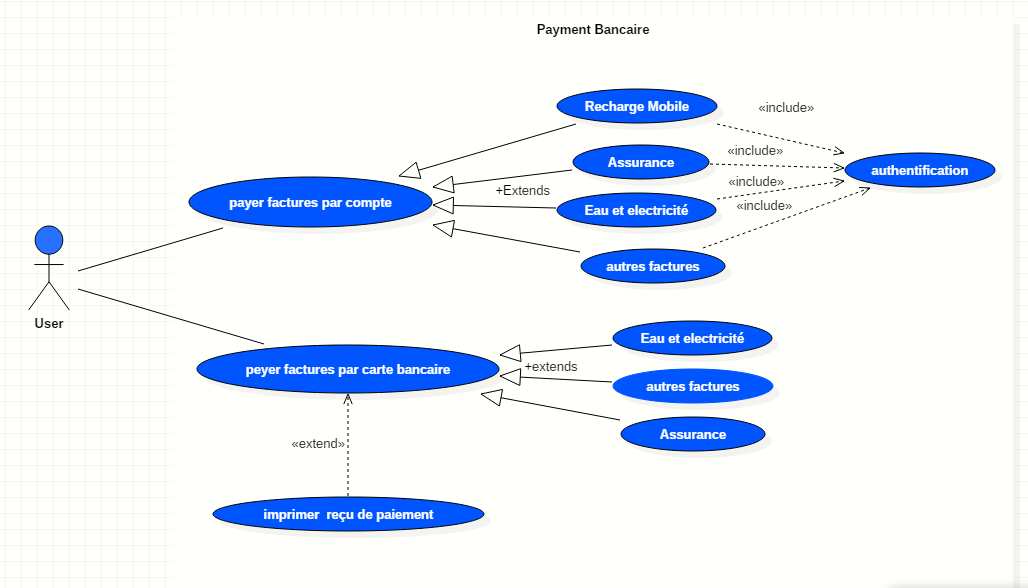


Figure 2 : Diagramme De Cas D’utilisation .

1. **Diagramme d’activité :**

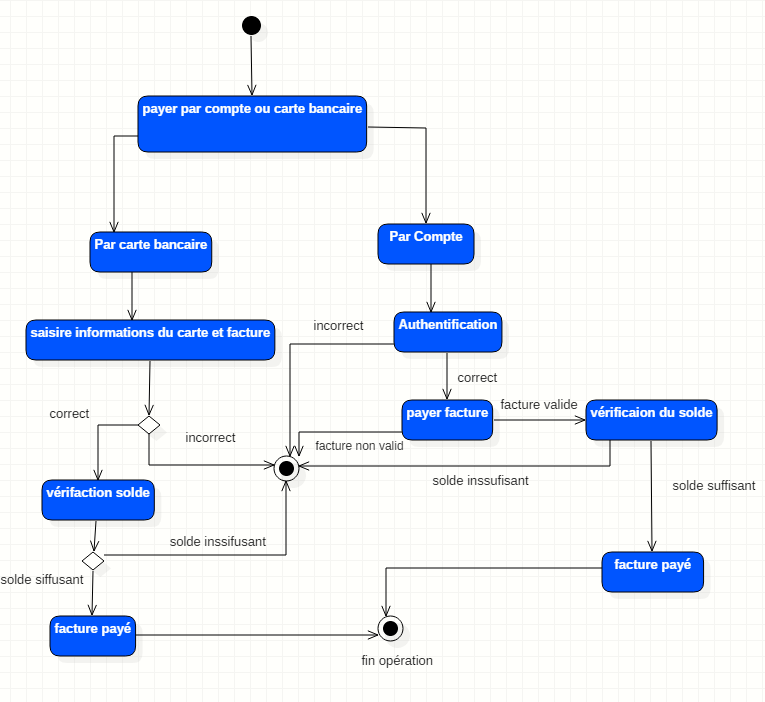
****

Figure 3 : Diagramme D’activité

1. **Conclusion :**

Dans ce chapitre nous avons défini les différents acteurs et leurs fonctionnalités, ainsi que les différentes vues conceptuelles d’application à travers les modèles UML nécessaires, afin de faciliter la phase de la réalisation

*Chapitre 3 :*

*Réalisation :*

1. **Introduction :**

Ce dernier chapitre présente dans un premier temps, les outils de développement de l'application. Ensuite, décrit le travail réalise en détaillant quelques captures d'écrans des fonctionnalités réalises.

1. **Environnement de développement :**

Dans cette partie on va situer les outils et les langages de programmation que nous avons utilisé pour réaliser cette application.

* **Les outils de développement :**
* ***PhpMyAdmin :***

 C’est un système de gestion des bases de données relationnel pour notre application.

* ***NetBeans IDE :***

 Est utilisé pour l'implémentation de notre application avec le langage de programmation java EE.

* ***WampServer :***

 Est utilisé comme un serveur local pour accéder au phpMyAdmin.

* ***StarUML :***

 Est utilisé pour Modéliser les diagrammes UML.

* **Langages de programmation :**

***Java SE :***

est une [spécification](https://fr.wikipedia.org/wiki/Sp%C3%A9cification_(norme_technique)) de la [plate-forme Java](https://fr.wikipedia.org/wiki/Plate-forme_Java) d'[Oracle](https://fr.wikipedia.org/wiki/Oracle_Corporation), destinée typiquement aux applications pour [poste de travail](https://fr.wikipedia.org/wiki/Poste_de_travail#Poste_de_travail_informatique) .

** *JDBC* :**

La technologie JDBC (Java DataBase Connectivity) est une API fournie avec [Java](https://web.maths.unsw.edu.au/~lafaye/CCM/java/javaintro.htm) (depuis sa version 1.1) permettant de se connecter à des bases de données, c'est-à-dire que JDBC constitue un ensemble de [classes](https://web.maths.unsw.edu.au/~lafaye/CCM/java/javaclass.htm) permettant de développer des applications capables de se connecter à des serveurs de [bases de données](https://web.maths.unsw.edu.au/~lafaye/CCM/bdd/bddintro.htm) ([SGBD](https://web.maths.unsw.edu.au/~lafaye/CCM/bdd/bddintro.htm)).

***Swing :***

est une [bibliothèque graphique](https://fr.wikipedia.org/wiki/Biblioth%C3%A8que_graphique) pour le [langage de programmation Java](https://fr.wikipedia.org/wiki/Java_(langage)) , offre la possibilité de créer des [interfaces graphiques](https://fr.wikipedia.org/wiki/Graphical_user_interface) , Il utilise le principe [Modèle-Vue-Contrôleur](https://fr.wikipedia.org/wiki/Mod%C3%A8le-Vue-Contr%C3%B4leur) (MVC, les composants Swing jouent en fait le rôle de la vue au sens du [MVC](https://fr.wikipedia.org/wiki/Mod%C3%A8le-Vue-Contr%C3%B4leur)) et dispose de plusieurs choix d'apparence pour chacun des composants standards.

1. **Interfaces de l’application :**

**1-Fenêtre d’accueil :**

Une fois l’utilisateur entre l’application il va voir la fenêtre d’accueil, qui le donne le choix de payer par compte ou par carte :

**Graphical user interface, application, Word

Description automatically generated**

Figure : Fenêtre d’Accueil

Si l’utilisateur choisi de payer par compte il va entrer sur les fenêtres suivantes :

**2-Payer Par compte :**

* **Login / Registre :**

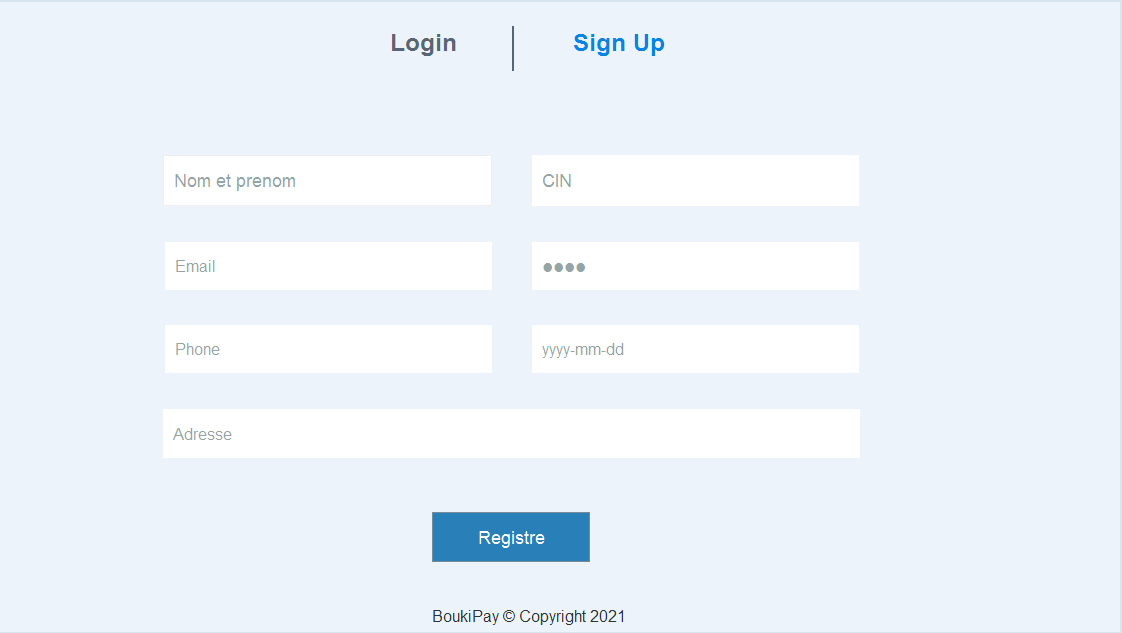
****

Figure  : fenêtre d’inscription

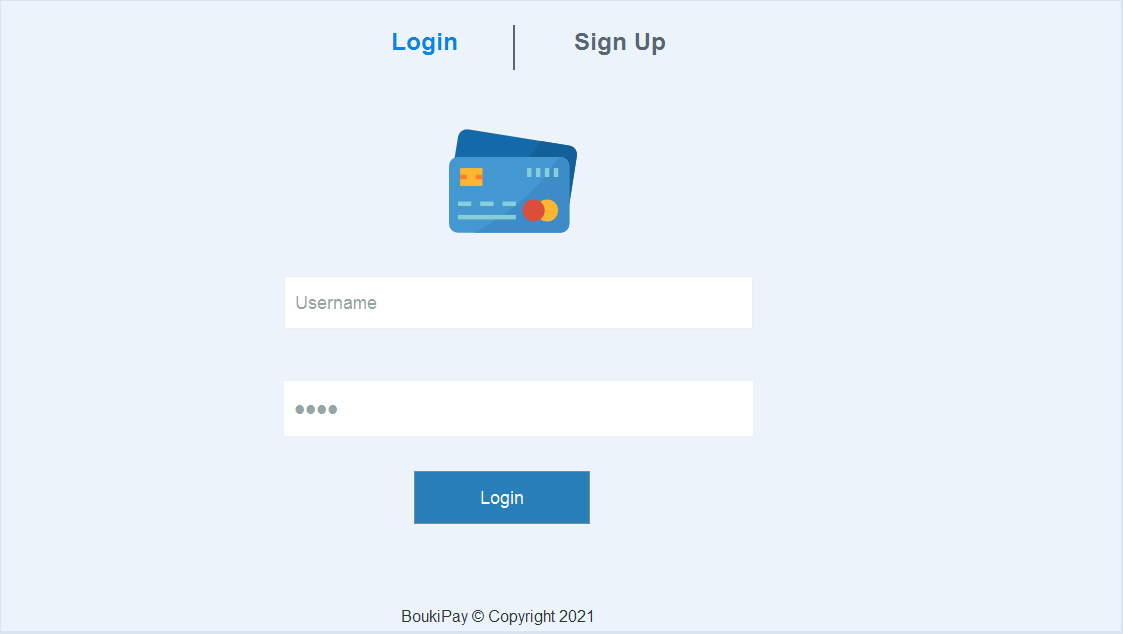


Figure : fenêtre d’authentification

* **Espace User :**

L’utilisateur va accéder à la page ci-dessous après l’authentification.

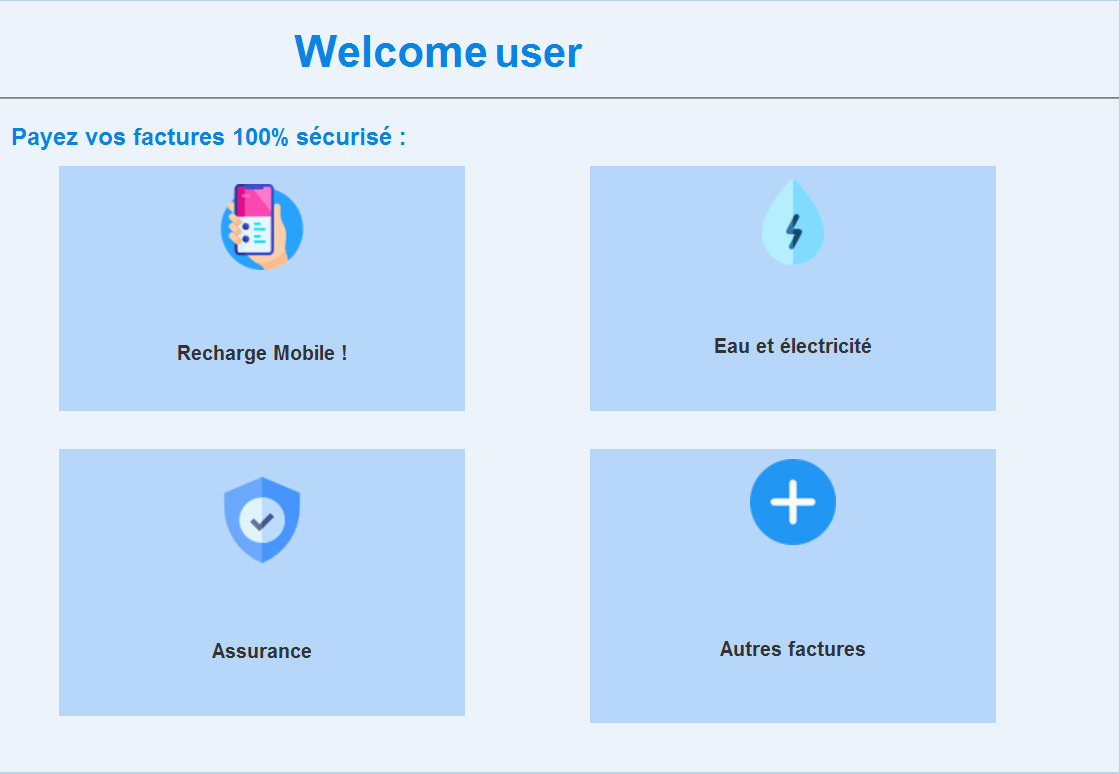


Figure : fenêtre d’accueil d’utilisateur

* **Droit d’utilisateur :**
* **Recharge Mobile :**

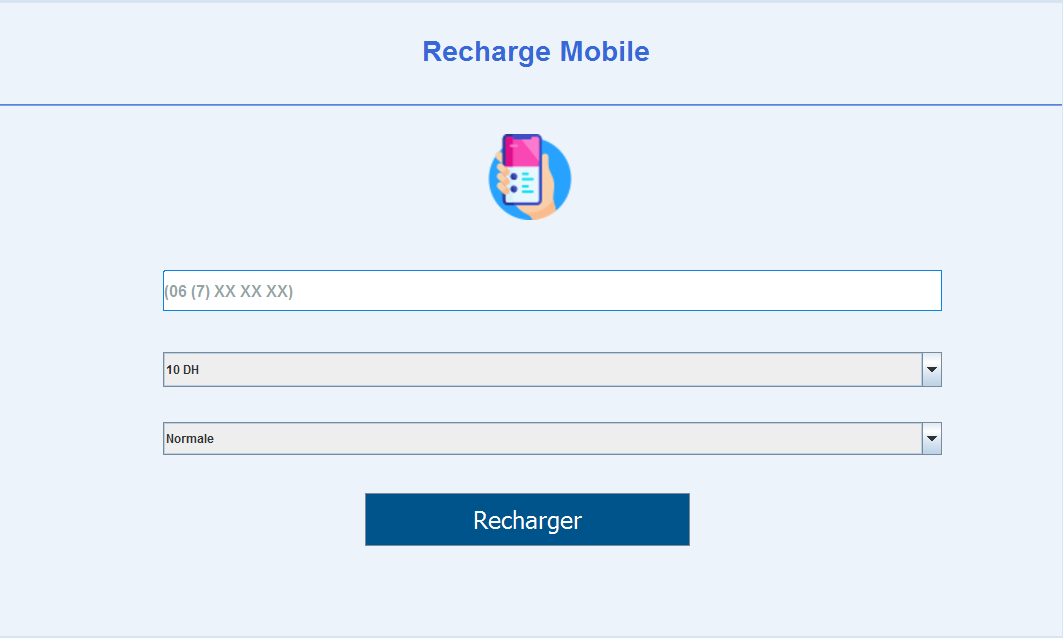
****

Figure : fenêtre de recharge mobile

Après que l’utilisateur fait entrer ses informations personnelles et clique sur le bouton Recharge, il va tomber sur une fenêtre de vérification :

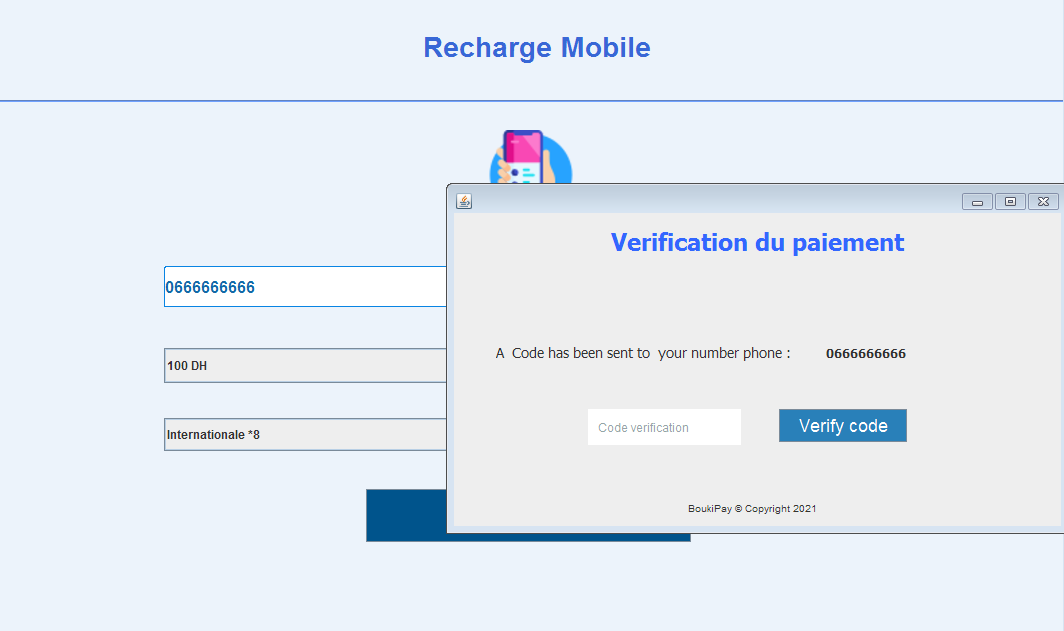


Figure : fenêtre de vérification mobile

Après que l’utilisateur écrit le code de vérification(tkon) qui se trouve dans la console :

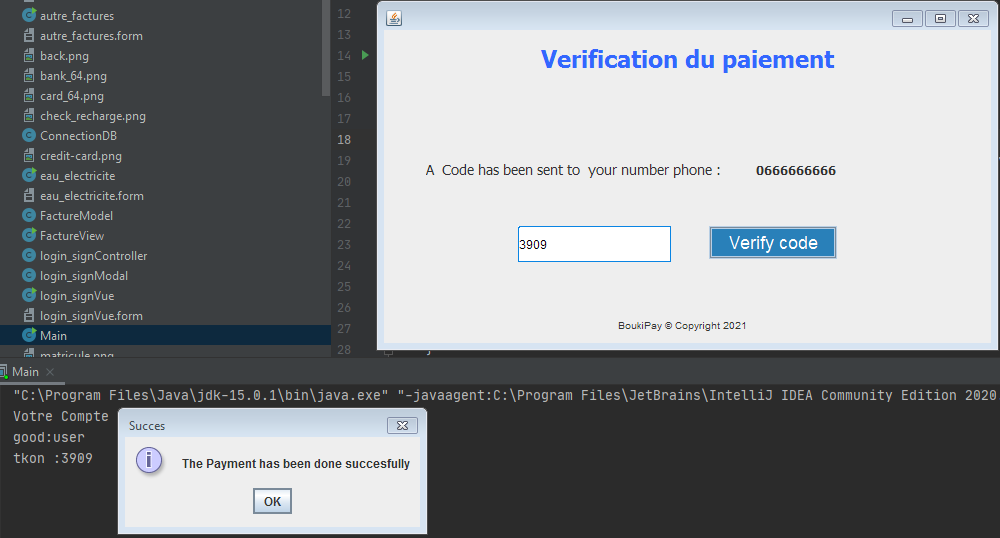


Figure : fenêtre d’insertion du code de vérification

Quand on clique sur le bouton OK, la fenêtre de validation de la recharge va être afficher

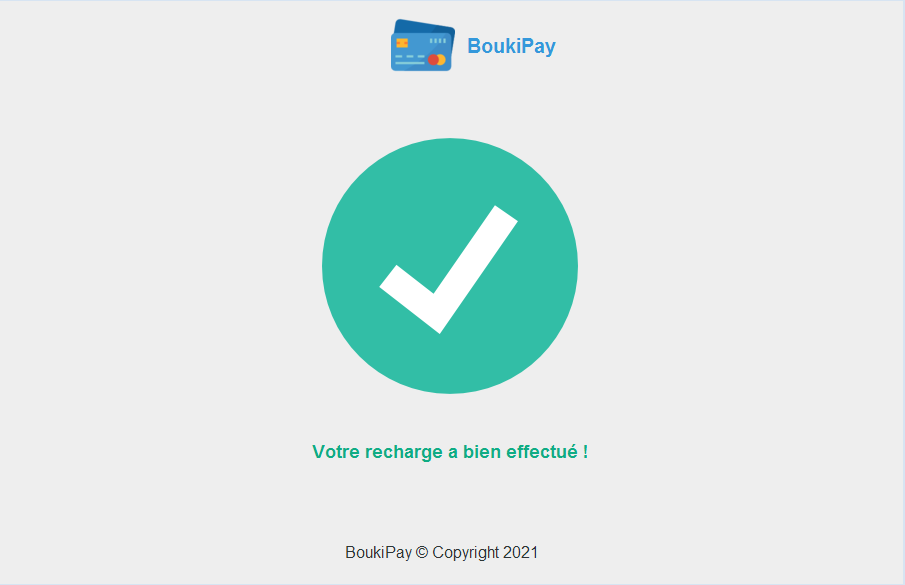


Figure : fenêtre de validation de la recharge

* **Eau et électricité :**

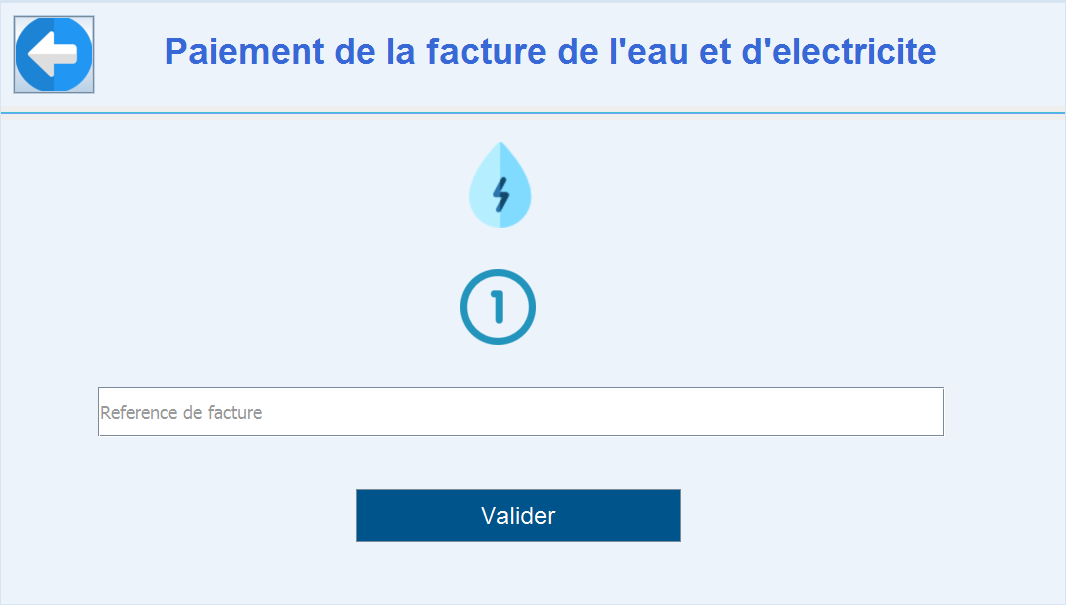
****

Figure : fenêtre de paiement de la facture de l’eau et d’électricité

Après que l’utilisateur fait entrer la référence de facture et clique sur le bouton Valider, il va tomber sur une fenêtre de vérification :

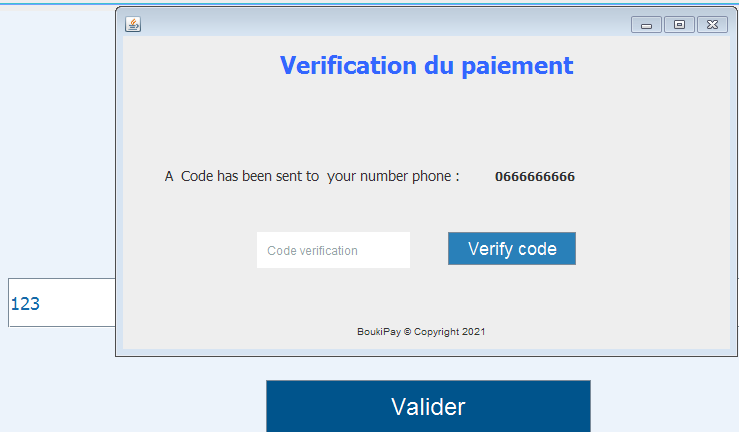
****

Figure : fenêtre de vérification du paiement de la facture de l’eau et d’électricité

Après que l’utilisateur écrit le code de vérification(tkon) qui se trouve dans la console :

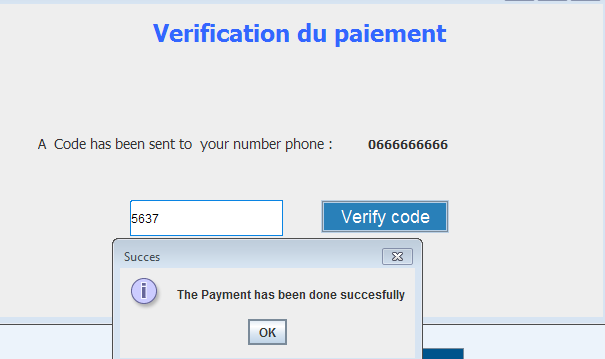


Figure : fenêtre d’insertion du code de vérification

Après que l’utilisateur clique sur le bouton OK, le reçu du paiement de la facture va être afficher :

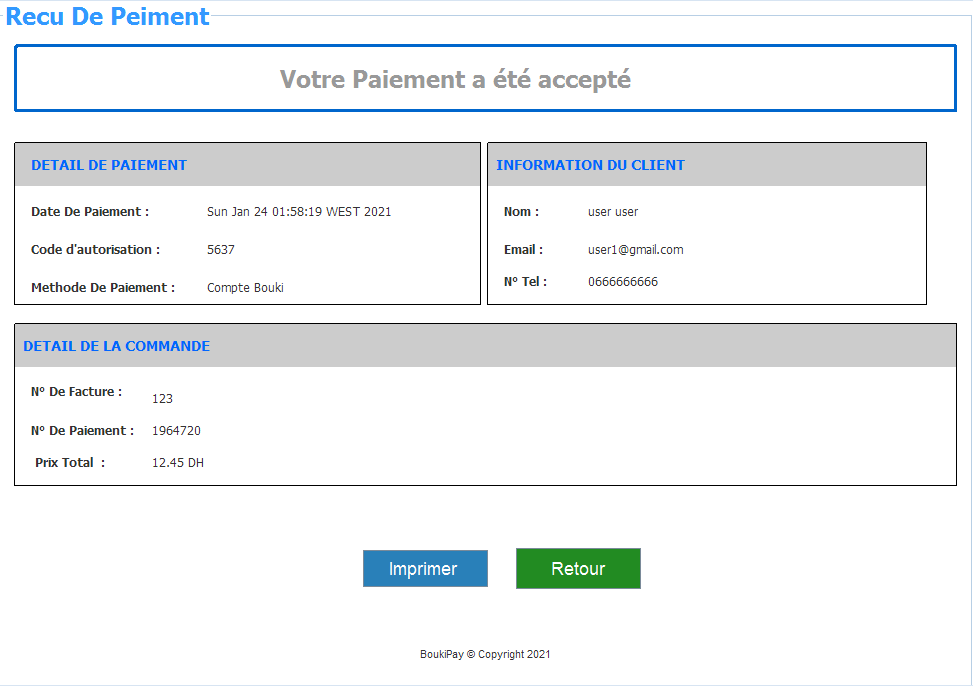


Figure : fenêtre du reçu de paiement

* **Assurance :**

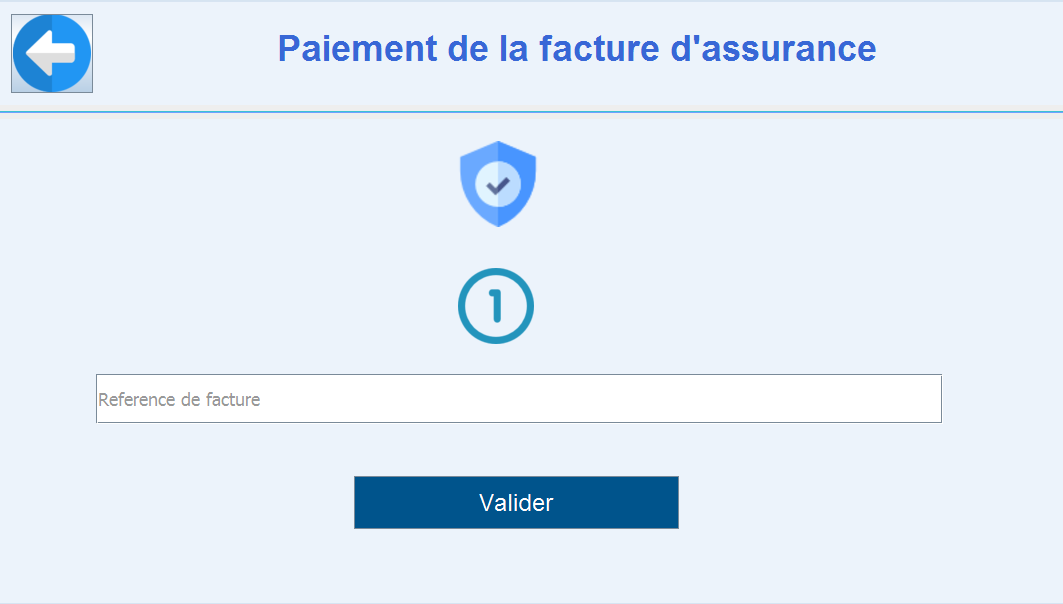


Figure : fenêtre de paiement de la facture d’assurance

Après que l’utilisateur fait entrer la référence de facture et clique sur le bouton Valider, il va tomber sur une fenêtre de vérification :

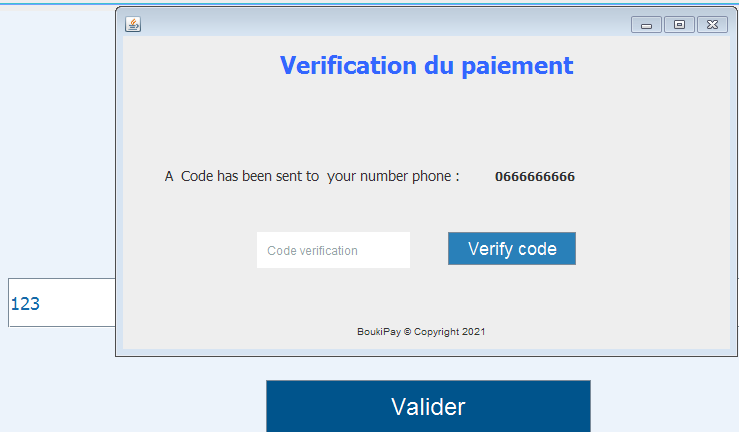
****

Figure : fenêtre de vérification du paiement de la facture d’assurance

Après que l’utilisateur écrit le code de vérification(tkon) qui se trouve dans la console :

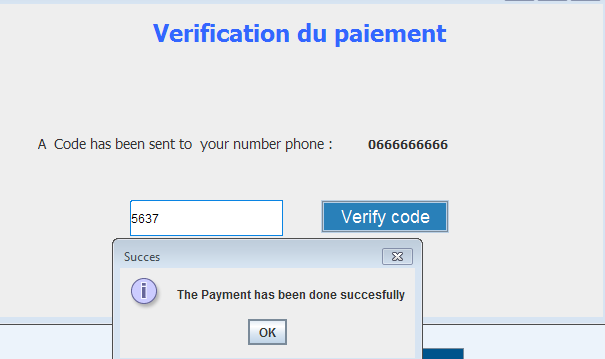


Figure : fenêtre d’insertion du code de vérification

Après que l’utilisateur clique sur le bouton OK, le reçu du paiement d’assurance va être afficher :

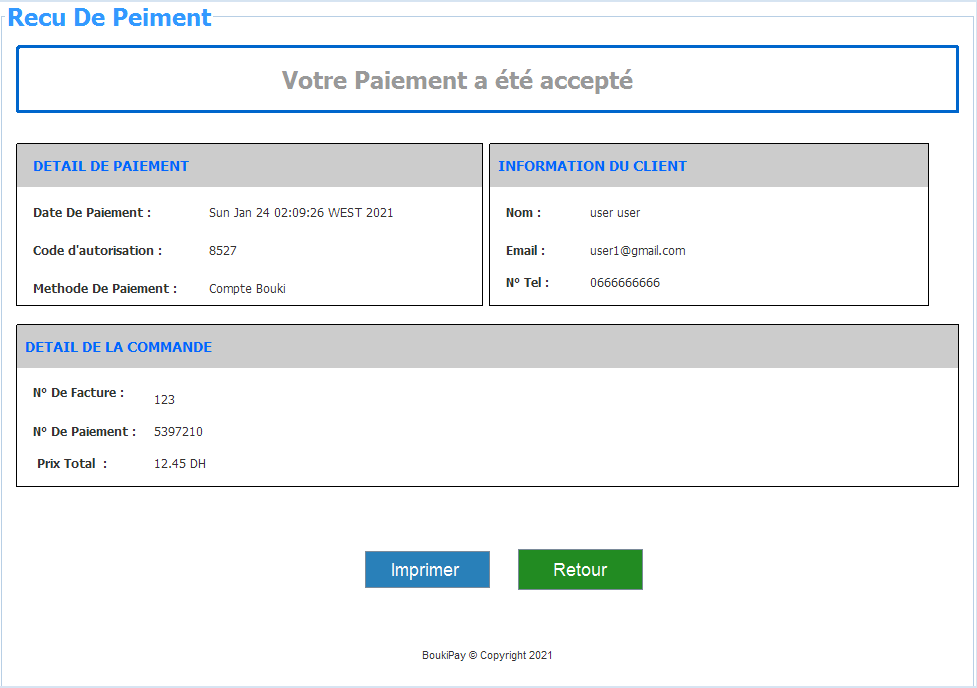


Figure : fenêtre du reçu de paiement

* **Autres factures :**



Figure : fenêtre de paiement des autres types de factures

Après que l’utilisateur fait entrer la référence de facture et clique sur le bouton Valider, il va tomber sur une fenêtre de vérification :

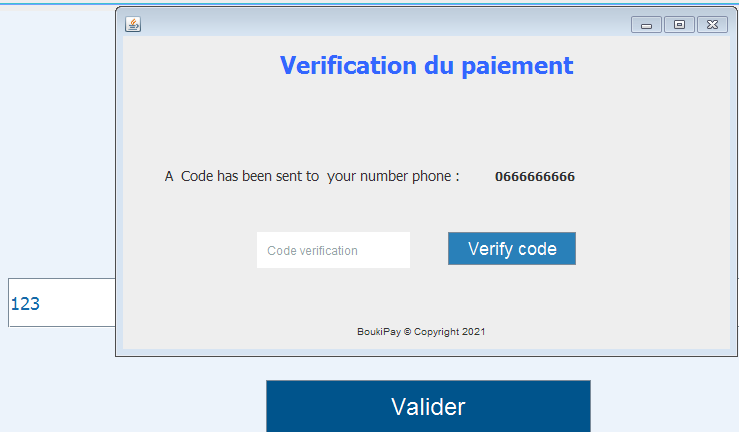
****

Figure : fenêtre de vérification du paiement des autres types de factures

Après que l’utilisateur écrit le code de vérification(tkon) qui se trouve dans la console :

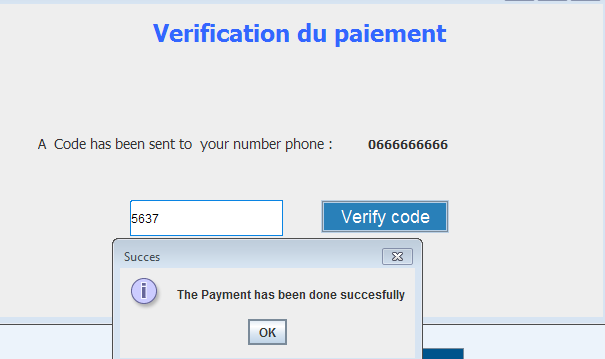


Figure : fenêtre d’insertion du code de vérification

Après que l’utilisateur clique sur le bouton OK , le reçu du paiement des autres types de factures va être afficher :

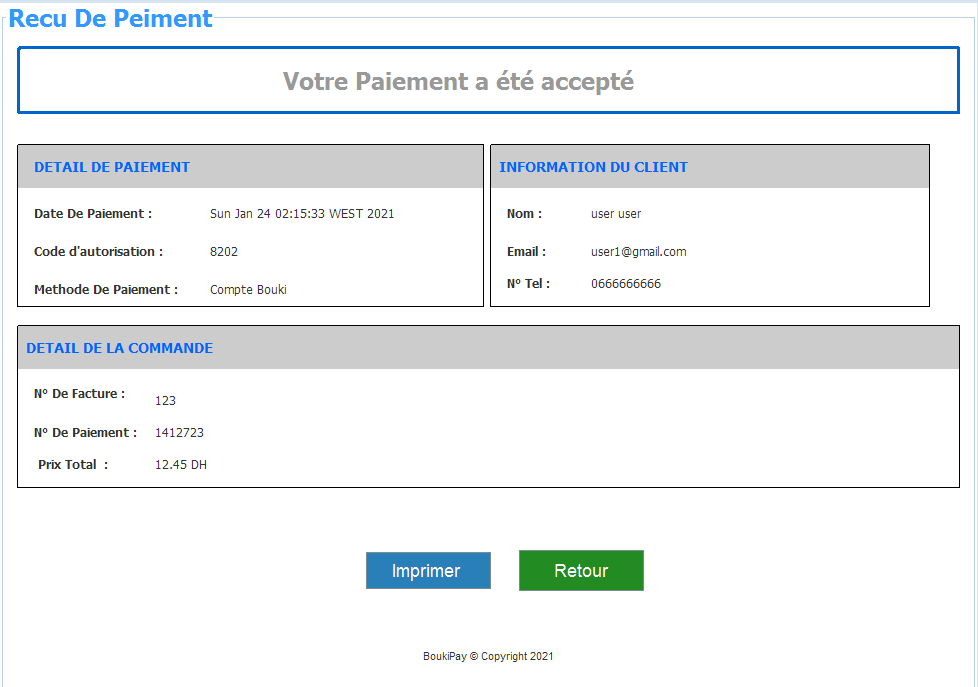


Figure : fenêtre du reçu de paiement

**3-Payer Par Carte bancaire :**

Si l’utilisateur choisi de payer par carte bancaire il va entrer sur les fenêtres suivantes :

* **Informations de carte et de factures :**

Graphical user interface, application

Description automatically generated

Figure : paiement par carte bancaire

Si l’utilisateur a pas remplir les champs des informations valide et correct il va voir une fenêtre de dialogue :

**- Fenêtres des erreurs :**

**Graphical user interface

Description automatically generated** Figure : Carte pas valide

**Graphical user interface

Description automatically generated**

Figure : Reference pas valide

Si l’utilisateur entre des information valides et le solde de la carte est **supérieur au prix de facture** : il va voir la fenêtre de vérification :

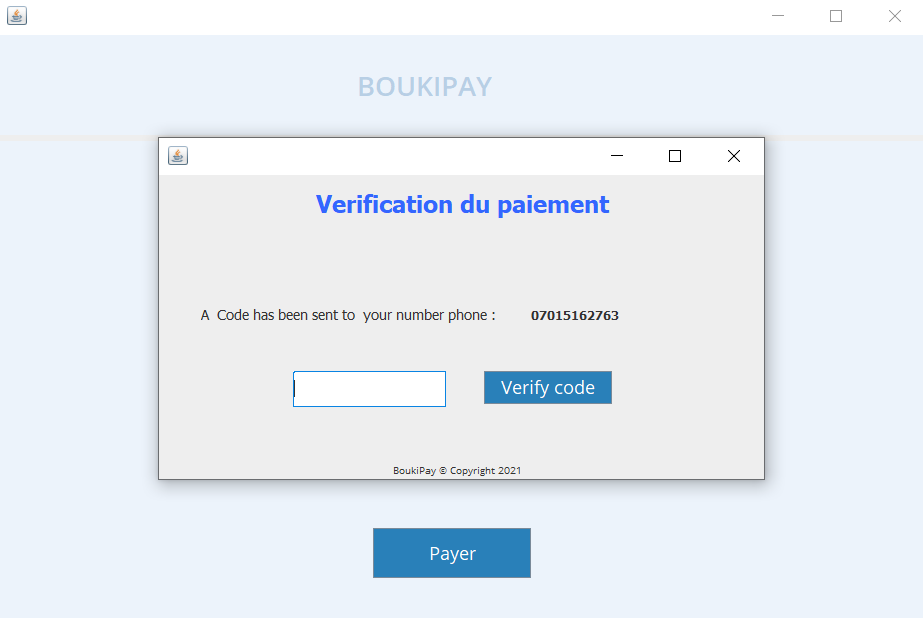


Figure : Fenêtre de vérification

Si l’utilisateur a entre le code envoyer sur leur téléphone, il va passer sur la fenêtre de reçu de paiement :

Graphical user interface, text, application, email

Description automatically generated

Figure : Reçu de paiement

Apre l’utilisateur peut imprimer les reçus ou fait le retour sur la fenêtre d’accueil pour payer une autre facture.

1. **Conclusion :**

Dans ce chapitre, nous avons réalisé une présentation des outils de développement et les Bibliotiques utilisés pour la réalisation de cette application, ainsi que les interfaces de notre application.

**Conclusion Générale**

Ce projet consistait à réaliser et développer une application Desktop de gestion des paiements bancaire dans le but de faciliter les tâches

De paiement online.

a réalisation de ce projet était faite en trois principales phases. La première consiste à une étude générale du projet, dans laquelle nous avons déterminé le cahier de charge et les besoins de l’application .la deuxième phase a été consacré à une étude conceptuelle par une modélisation UML. La dernière consiste à une étude technique en présentant les outils et les contraintes du développement utilisés, qui comprend la mise en œuvre de l’application

Ce projet est une bonne occasion pour renforcer et enrichir nos connaissances et nos compétences dans le domaine de développement java. Il nous a encore donné l'occasion de maîtriser le langage de programmation JAVA SE avec Swing et JDBC.