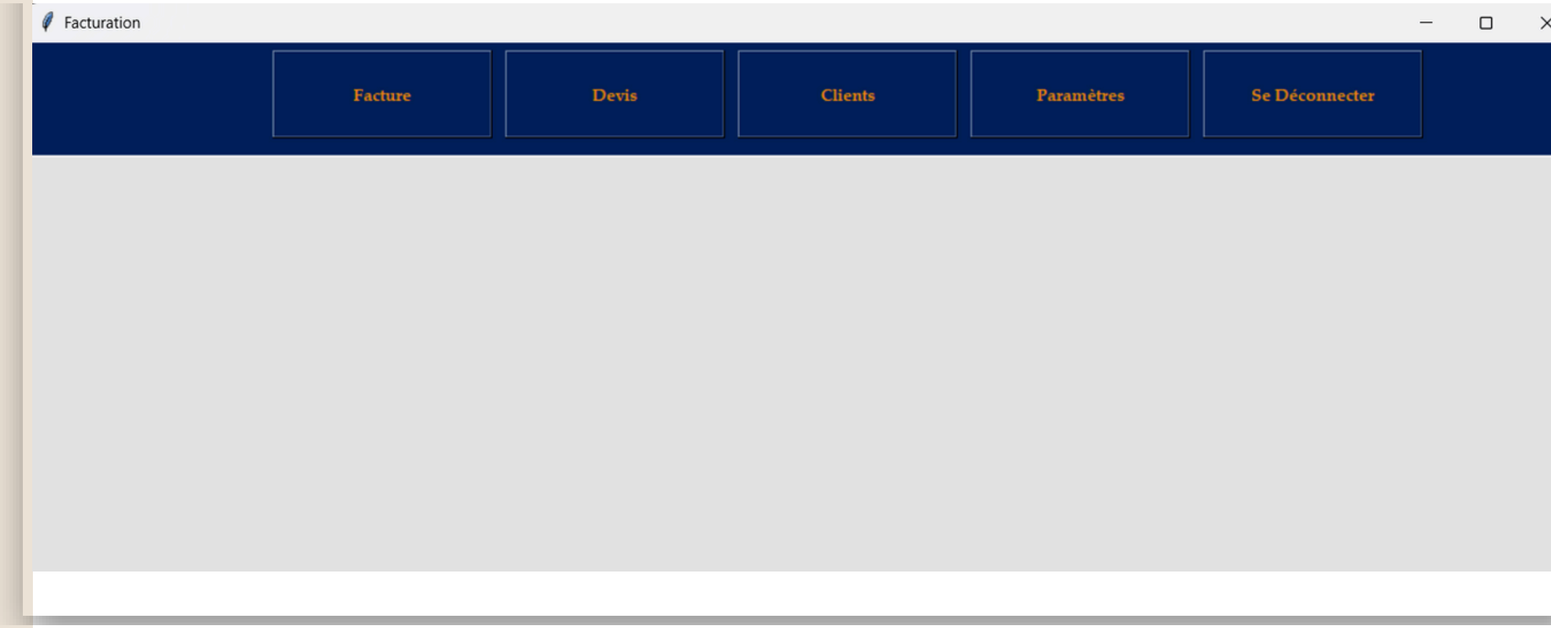
Logiciel De Facturation

Développement d'applications de gestion



Présentés par :

- Saleh BA-ZIGHIFAN
- Abdulaziz AL KHULAQI
- Ryan Khlaief Benmiled

Introduction

BIENVENUE À TOUS !

Présentation d’une solution de facturation intuitive pour artisans et indépendants.

🔹 Problématique

- Les professionnels ont peu de temps pour la gestion administrative.
- Les tâches comme la facturation et la gestion des clients sont chronophages.

🔹 Solution proposée

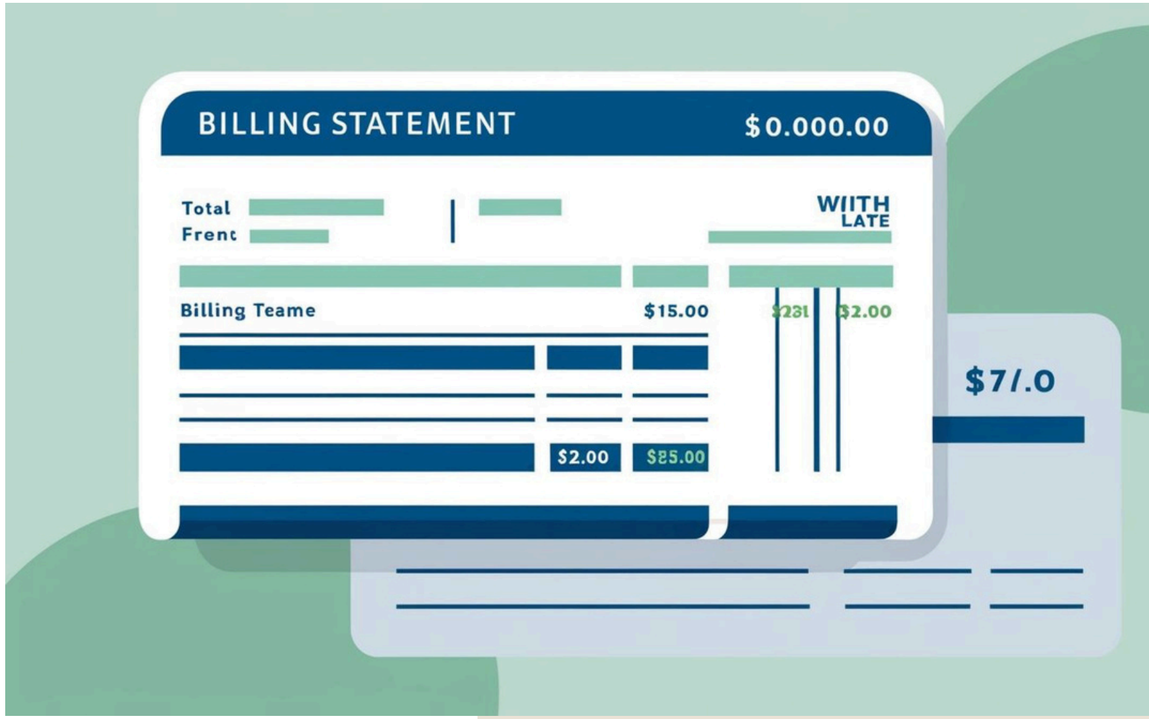
- Une application simple et efficace.
- Gestion des clients, édition de devis et génération de factures en quelques clics.
- Interface intuitive et accessible à tous.

🔹 Objectif principal

- Permettre aux artisans de se concentrer sur leur métier sans stress administratif.

🔹 Public cible

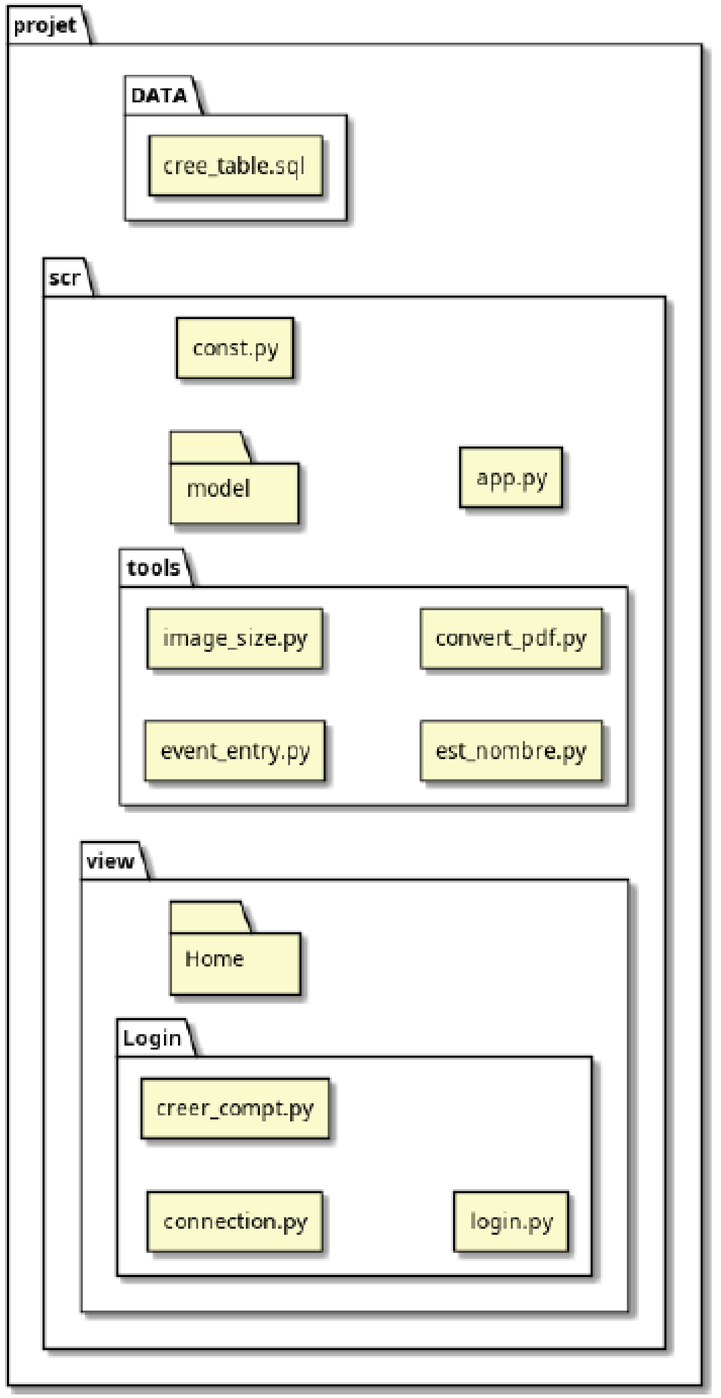
- Électriciens, plombiers et autres professionnels du secteur.
- Toute personne ayant besoin d’un outil fiable pour la facturation.



Organisation des fichiers du projet

Structure générale

- DATA : Contient les fichiers SQL pour la base de données.
- scr : Contient les fichiers principaux du projet.
    a. model : Gère la logique métier et l'accès aux données.
    b. app.py : Fichier principal de l'application.
    c. const.py : Fichier de constantes.
    d. tools : Contient des outils spécifiques (conversion, validation, etc.).
    e. view : Gère l'affichage des pages.



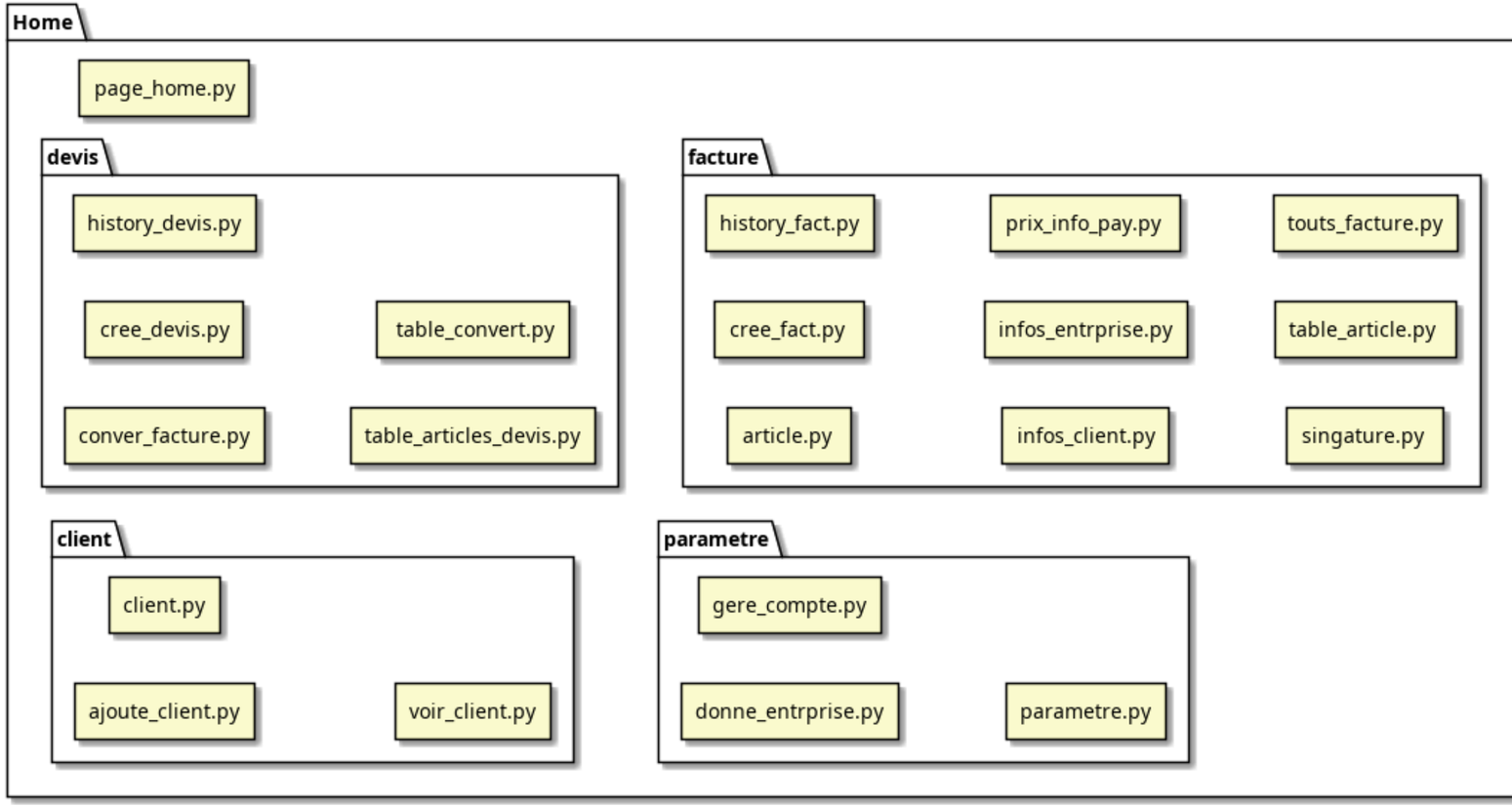
Organisation des fichiers (Home)

Home : Page principale de l’application.

- **Par example**

Un dossier spécial client contenant plusieurs fichiers :

- client.py : Classe principale.
- ajoute_client.py et voir_client.py : Fonctions pour ajouter et afficher un client.
- La classe Client regroupe plusieurs fonctions (def) qui interagissent grâce à self.



Avantages

- Code organisé et facile à maintenir.
- Séparation claire des responsabilités.
- Meilleure collaboration entre les fichiers.

view > home > client > client.py

```python
import tkinter as tk
from tkinter import messagebox

from const import *
from tools.event_entry import effacer_indicatif
from view.home.client.ajoute_client import AjouteClient
from view.home.client.voir_client import VoirClient


class Client():
    def __init__(self, root, frame_button, BDD, id_utilisateur):
        self.root = root
        self.frame_button = frame_button
        self.BDD = BDD
        self.id_utilisateur = id_utilisateur

        self.canvas = None
        self.root.after(10, self.initialisation)
        self.root.bind("<Configure>", self.on_configure)



    def initialisation(self):

        x = self.root.winfo_width()
        y = self.root.winfo_height()
        self.canvas = tk.Canvas(self.root, width=x, height=y,bg=COULEUR_PRINCIPALE)
        self.canvas.place(x=0, y=(y//11.42))



        self.recherhe_client = tk.Entry(self.canvas,width=35,fg="gray")
```

Le compte avec le quel nous avons fait la présentation
email : QQQ
mot de passe : QQQ