IKNAE TECHNO

https://iknaetechno.fr

RAPPORT D'ACTIVITÉS

Joy Huré

Stage du 01/05/2024 au 18/06/2024

TABLE DES MATIERES

PRESENTATION

L’entreprise

IKNAE TECHNO est une entreprise spécialisée dans **les bases de données**, fondée par des experts reconnus de ce domaine. Elle se consacre à l'amélioration de la gestion des bases de données pour les entreprises, en offrant une stratégie complète de gestion, d’optimisation et de performances.
IKNAE TECHNO est partenaire des acteurs majeurs du marché des bases de données.

Organisation du travail

Cette entreprise est dirigée par trois associés, dont Alexandra Champavert, qui en est la présidente. Le mode de travail au sein de l’entreprise est décentralisé : chaque associé travaille à distance, disposant d’un bureau personnel dans une localisation différente et se déplaçant ponctuellement chez les clients selon les projets en cours.
Ce fonctionnement flexible permet à l’entreprise de rester réactive et de maintenir une proximité avec ses clients pour répondre efficacement à leurs besoins.
Durant mon stage, j'ai principalement travaillé à distance, en suivant le mode de fonctionnement décentralisé de l'entreprise.

MISSION INITIALE

Contexte

Ma mission initiale chez IKNAE TECHNO consistait à mettre en place un environnement de développement adapté, dans l'optique de monter progressivement en compétences sur **les bases de données** et **les technologies web**.

À la demande de ma tutrice, l'objectif était notamment de me former à l'utilisation de **PostgreSQL** et d'explorer les bases d'un projet plus ambitieux : la création d'un service applicatif en mode **SaaS** (Software as a Service).

Ce **SaaS** avait pour vocation de contrôler l'accès des clients aux outils d'expertise et d'optimisation de bases de données proposés par l'entreprise. Bien que le projet complet ait dépassé mon niveau de compétences à ce stade de la formation, j'ai orienté mes apprentissages en ce sens.

À la fin de la période, j'ai pu présenter un prototype simplifié illustrant les fondements de ce **SaaS** : installation et configuration de l'environnement, création d'une base **PostgreSQL**, mise en place d'une **authentification** utilisateur et conception d'une **interface web** rudimentaire.

Prérequis

Ce projet nécessitait la création d'un environnement de développement dont ma tutrice m'a précisé les exigences : sous **Linux**, avec des outils spécifiques adaptés à la gestion de bases de données, tels que **PostgreSQL** pour le stockage des données et **pgAdmin** pour leur administration. Dans le cadre de ce projet, j'étais également amené à utiliser le langage **Pascal** pour le développement de certaines fonctionnalités du **SaaS**.

Objectif

L'objectif global de cette mission était de définir les bases techniques de cette solution afin de permettre un suivi automatisé des comptes abonnés, en contrôlant l'accès et les droits des utilisateurs de façon sécurisée. Il m'a été confié la mission d'installer et de configurer cet environnement de travail de façon indépendante, sous la supervision de ma tutrice Alexandra Champavert.

Installation des outils de développement

Déploiement d'une VM Ubuntu

Avantages de l'environnement

Pour créer un environnement de développement isolé, adapté à la gestion de bases de données et en conformité avec les consignes de ma tutrice, j'ai déployé **une machine virtuelle sous Ubuntu**.

Le choix d'**Ubuntu** repose sur plusieurs raisons qui en font un système particulièrement adapté dans ce contexte.
Tout d'abord, **Ubuntu est une distribution Linux open-source** largement utilisée dans les environnements de développement et les entreprises pour ses performances, sa sécurité, et sa stabilité. Elle est adaptée aux tâches de gestion de bases de données grâce à sa légèreté et à sa capacité à fonctionner efficacement même avec des ressources limitées.
Ensuite, Ubuntu offre **une large compatibilité avec les outils de développement nécessaires au projet**, notamment **PostgreSQL**, et dispose d'un vaste catalogue de packages et d'outils disponibles dans ses dépôts. Cette compatibilité garantissait que tous les logiciels requis pour le SaaS pourront être installés et fonctionner correctement.
Enfin, **Ubuntu** est régulièrement mis à jour et bénéficie d'une large communauté de développeurs qui fournit **une documentation abondante, des ressources, et un support** en cas de besoin. Cela rend la résolution de problèmes plus accessible et permet de travailler dans un environnement stable et maintenu.

Configuration de la VM selon les ressources de l'hôte

Pour garantir des performances optimales, j'ai configuré la VM selon les caractéristiques de ma machine physique :

- **8 Go de RAM**
- **6 cœurs** du processeur
- **Stockage** sur un **SSD de 400 Go** (disque D) dont **100 Go** est alloué à cette VM, pour accélérer les accès au disque

Spécifications de l'appareil

Nom de l'appareil	JoyO_Len
Processeur	12th Gen Intel(R) Core(TM) i9-12900HX 2.30 GHz
Mémoire RAM installée	32,0 Go (31,7 Go utilisable)
ID de périphérique	4EAC0C2F-E081-49AC-AE55-C66C464AF6CB
ID de produit	00330-50000-00000-AAOEM
Type du système	Système d'exploitation 64 bits, processeur x64
Stylet et fonction tactile	La fonctionnalité d'entrée tactile ou avec un stylet n'est pas disponible sur cet écran

Ressources de l'ordinateur hôte

VM-UBUNTU-GENERAL01

Power on this virtual machine
Edit virtual machine settings

Devices

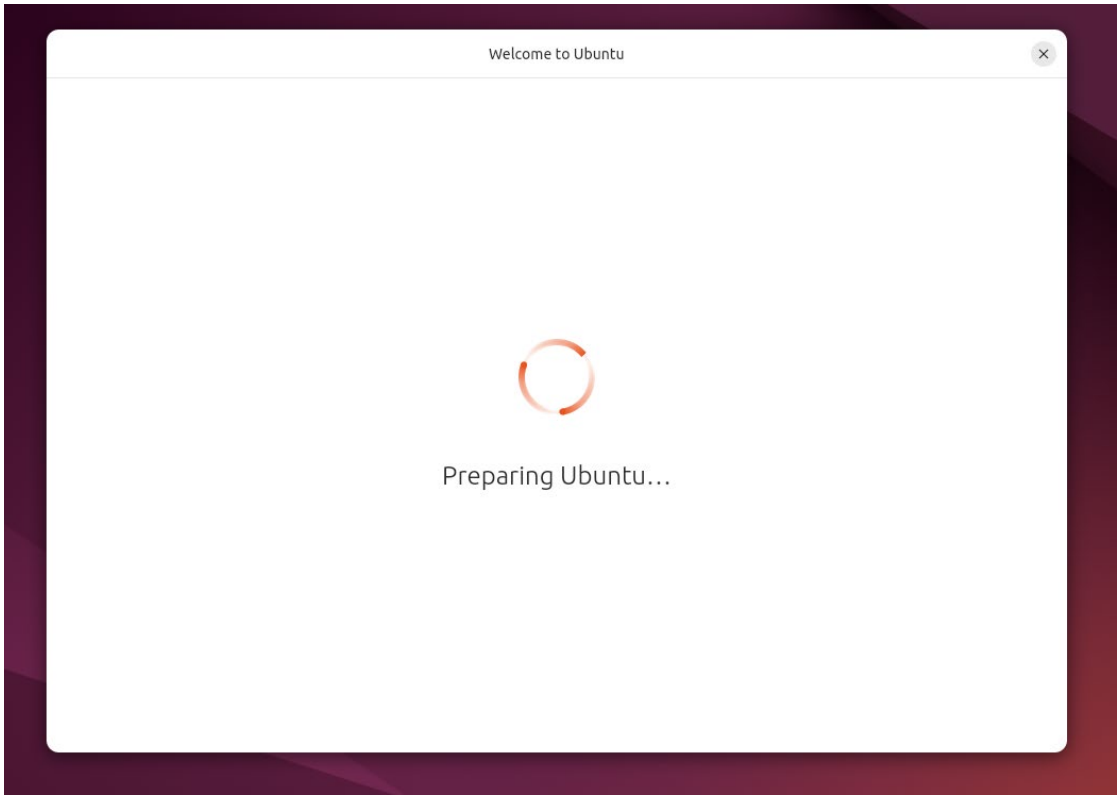
Memory	8 GB
Processors	6
Hard Disk (SCSI)	100 GB
CD/DVD 2 (SATA)	Using file D:\VM...
CD/DVD (SATA)	Using file autoin...
Floppy	Using file autoin...
Network Adapter	NAT
USB Controller	Present
Sound Card	Auto detect
Display	Auto detect

Description

Environnement de développement Linux Ubuntu.

Configuration de la VM

Installation du système



Installation d'Ubuntu en cours

Fin de l'installation

Mises à jour

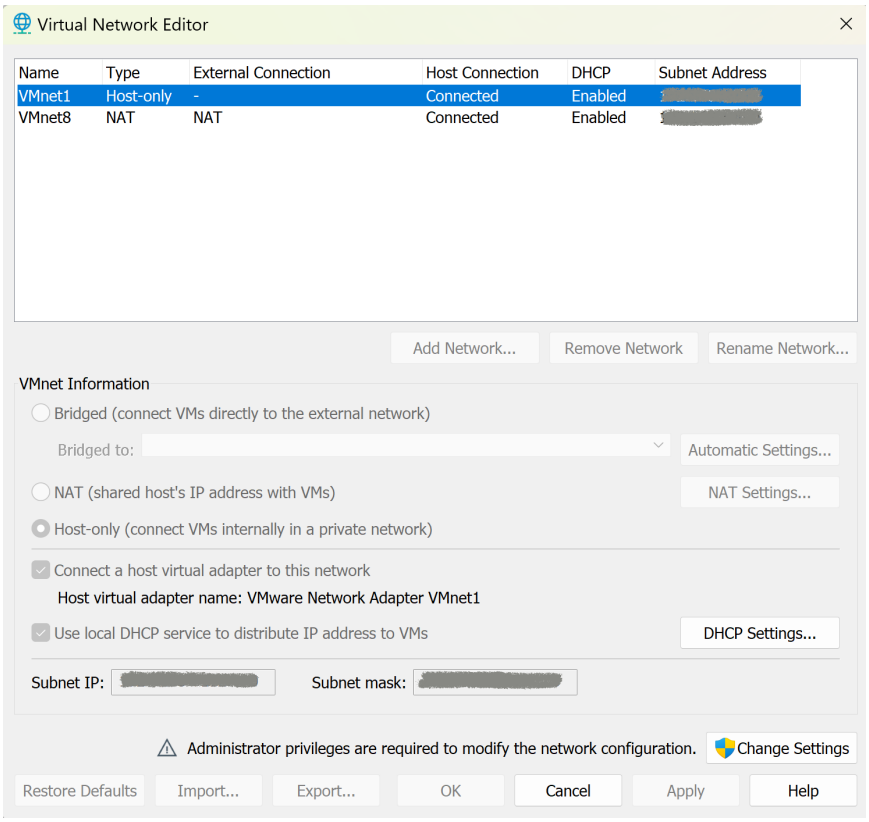
Une fois l'installation d'Ubuntu terminée, il est essentiel de **mettre à jour le système** pour s'assurer que tous les paquets sont à jour, sécurisés, et optimisés. Cela inclut l'installation des dernières mises à jour de sécurité et des versions des logiciels.

```
joy@joy-VM-UBUNTU01:~$ sudo apt update && sudo apt upgrade && sudo apt full-upgr
ade -y
```

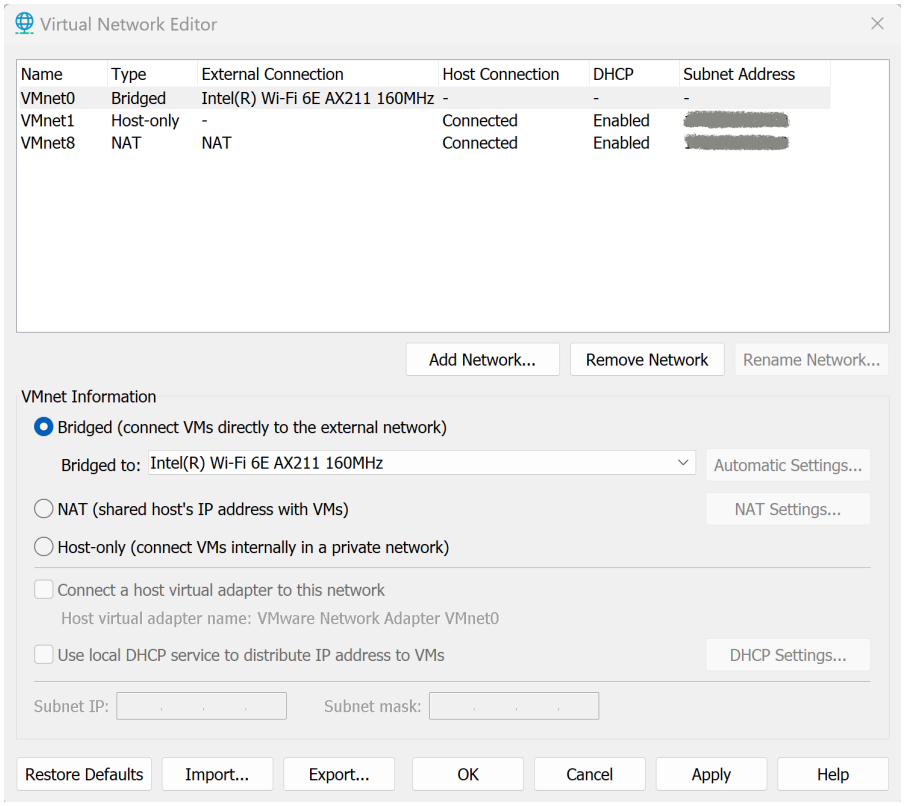
Commande de Mises à jour

Configuration du réseau en « bridge »

Le choix du mode **bridge** s'est imposé comme la solution la plus adaptée en offrant **une flexibilité réseau maximale**, **une communication facilitée** avec les services, et une configuration proche des standards utilisés en production. Ce paramétrage constitue une base solide pour les futures étapes du projet.



Configuration initiale en NAT



Paramétrage du bridge

```
joy@joy-VM-UBUNTU01:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=116 time=21.6 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=116 time=21.0 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=116 time=19.5 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=116 time=21.9 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=116 time=51.3 ms
^C
--- 8.8.8.8 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4010ms
rtt min/avg/max/mdev = 19.534/27.070/51.287/12.135 ms
```

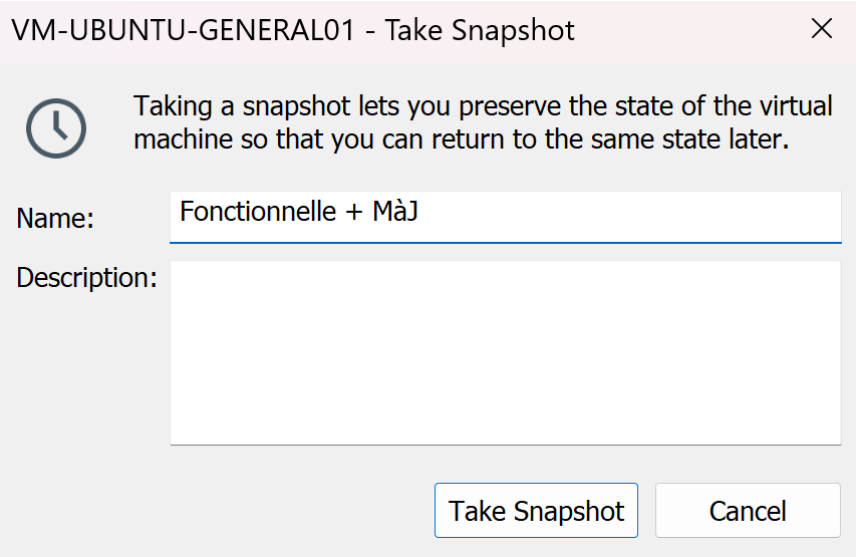
Ping de google pour tester la connexion

```
PS C:\Users\joyou> ping [redacted]

Envoi d'une requête 'Ping'  [redacted] avec 32 octets de données :
Réponse de [redacted] : octets=32 temps<1ms TTL=64
Réponse de [redacted] : octets=32 temps<1ms TTL=64
Réponse de [redacted] : octets=32 temps=1 ms TTL=64
Réponse de [redacted] : octets=32 temps<1ms TTL=64

Statistiques Ping pour [redacted]:
    Paquets : envoyés = 4, reçus = 4, perdus = 0 (perte 0%),
Durée approximative des boucles en millisecondes :
    Minimum = 0ms, Maximum = 1ms, Moyenne = 0ms
```

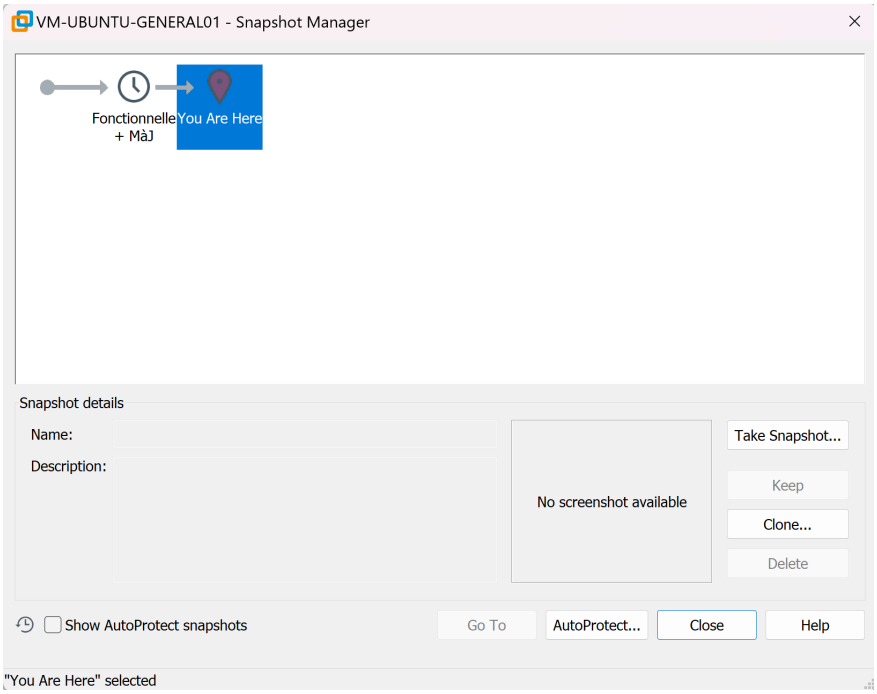
Ping de la VM depuis l'hôte pour vérifier la communication

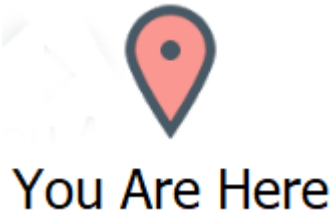
Snapshot

Après avoir configuré l'environnement de développement et effectué les mises à jour, il est conseillé de prendre un **snapshot** (instantané) de la VM. Un **snapshot** capture **l'état actuel de la machine virtuelle** à un instant donné, ce qui permet de revenir rapidement à cet état en cas de problème ou de besoin de restauration. Cela offre une sécurité supplémentaire lors de tests ou d'installations de nouveaux logiciels, en permettant de restaurer facilement le système à un état stable antérieur.



Snapshot après installation d'Ubuntu et mise à jour du système



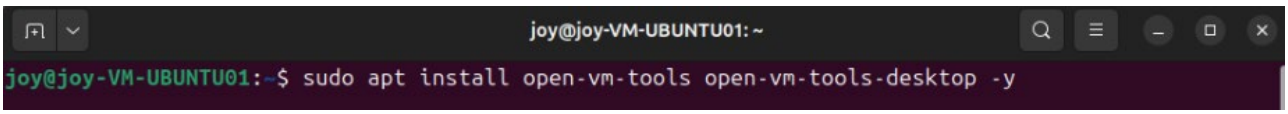


Visualisation des snapshots dans VMWare Workstation

Installation des VMWare Tools

Les VMware Tools permettent d'améliorer l'intégration entre l'hôte et la VM, notamment pour obtenir le **copier-coller** entre systèmes, le **glisser-déposer** et une meilleure **gestion de l'affichage**.

L'installation a été réalisée en ligne de commande avec les packages **open-vm-tools** et **open-vm-tools-desktop**.



Installation des VMWare Tools en utilisant la ligne de commande

Résolution d'un problème d'affichage

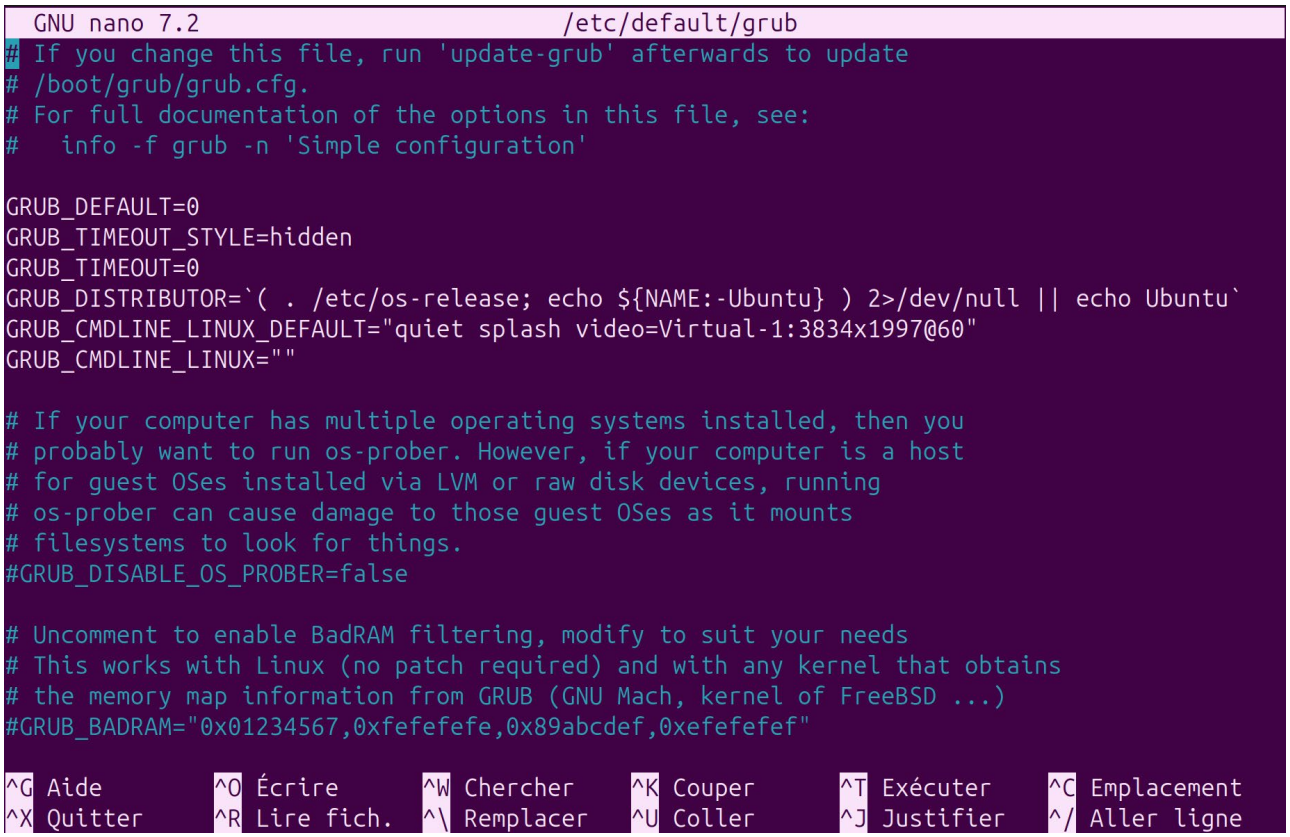
Lors de l'installation et de la configuration de la VM Ubuntu, un problème d'affichage a été rencontré concernant **l'activation d'une résolution 4K** (3840 x 2160). Plusieurs approches ont été explorées pour résoudre cette problématique, notamment :

- Réinstallation des VMware Tools
- Utilisation de **xrandr** et de modes personnalisés : Des commandes comme **gtf** et **xrandr** ont été utilisées pour créer des résolutions personnalisées, mais ces manipulations ont échoué avec des erreurs telles que **BadValue** et **BadMatch**. Ces erreurs ont révélé une incompatibilité entre les paramètres de la carte graphique virtuelle et les options disponibles.
- Exploration des journaux et diagnostics système : Des outils comme **journalctl** et **lshw** ont permis de diagnostiquer les configurations de la carte graphique et de confirmer les limitations liées aux pilotes utilisés.

Après plusieurs essais infructueux, une solution durable a été mise en œuvre.

```
92  sudo lshw -c video
93  sudo nano /etc/default/grub
94  sudo update-grub
95  sudo reboot
```

Commandes ayant solutionné le problème



Ajout de « video=Virtual-1:3834x1997@60 » à la ligne GRUB_CMDLINE_LINUX_DEFAULT

Bilan des Étapes Initiales

À ce stade, j'ai préparé un environnement de développement **isolé** et **performant** grâce au déploiement d'**une machine virtuelle sous Ubuntu**. La configuration de cet environnement a été réalisée en tenant compte **des ressources de l'hôte** pour **optimiser les performances**, tout en assurant une compatibilité avec les outils nécessaires au projet à venir.

J'ai également mis en œuvre des étapes clés, telles que **l'installation d'Ubuntu**, **la mise à jour du système**, et la réalisation de **snapshots** à chaque étape pour garantir la stabilité et la sécurité de l'environnement.

L'ajout des **VMware Tools** a permis **d'améliorer l'interaction entre l'hôte et la machine virtuelle**, renforçant ainsi l'efficacité du travail dans cet environnement.

La prochaine étape consistera à installer et configurer les outils de développement nécessaires, tels que **PostgreSQL**, **pgAdmin** et **Visual Studio Code**, afin de préparer l'environnement pour le SaaS.

Compétences mobilisées

Gérer le patrimoine informatique

- **Recenser et identifier les ressources numériques** : La configuration de la VM (8 Go RAM, 6 cœurs, SSD) implique un recensement des ressources disponibles sur l'hôte pour les allouer à la VM.
- **Exploiter des référentiels, normes et standards adoptés par le prestataire informatique** : L'utilisation d'Ubuntu, un système d'exploitation standard dans l'industrie, et des outils open-source (comme VMware Tools) montre la prise en compte de normes et standards reconnus.
- **Vérifier les conditions de continuité d'un service informatique** : La création d'un snapshot pour sécuriser les configurations de la VM assure la continuité et la possibilité de restaurer l'état stable de l'environnement.

Mettre à disposition des utilisateurs un service informatique

- **Réaliser les tests d'intégration et d'acceptation d'un service** : Les tests de connectivité réseau (mode bridge et NAT, ping de Google) confirment que l'environnement est fonctionnel et prêt pour l'utilisation.
- **Déployer un service** : L'installation d'Ubuntu et sa configuration initiale (mises à jour, gestion réseau, VMware Tools) constituent un déploiement complet d'un environnement de service.
- **Accompagner les utilisateurs dans la mise en place d'un service** : La documentation des étapes (commandes utilisées, diagnostic des problèmes) constitue un support d'accompagnement indirect.

Organiser son développement professionnel

- **Mettre en place son environnement d'apprentissage personnel** : La mise en place de la VM Ubuntu crée un environnement isolé propice au développement, expérimentation et apprentissage.

Installation de postgreSQL et pgadmin

Utiliser PostgreSQL a été un choix de l'entreprise qui l'utilise intensivement dans ses projets professionnels.

Avantages de postgreSQL

PostgreSQL est reconnu pour sa **robustesse**, ses **performances élevées**, et sa **compatibilité** avec de nombreux outils de développement.
Les avantages de **postgreSQL** résident principalement dans sa puissance et sa **flexibilité**, puisqu'il peut être utilisé aussi bien pour des requêtes simples que pour des tâches complexes d'administration de bases de données.
Sa simplicité en fait un outil accessible, permettant une gestion directe des bases sans nécessiter d'interface graphique. De plus, **postgreSQL** se distingue par sa compatibilité avec des scripts, ce qui le rend particulièrement adapté à l'**automatisation** des tâches récurrentes.

postgreSQL

```
1  sudo apt update && sudo apt upgrade -y
2  sudo apt install postgresql postgresql-contrib
3  sudo systemctl status postgresql
4  sudo -i -u postgres
```

Mise à jour des paquets, installation, vérification de l'état du service postgreSQL puis connexion au shell postgreSQL

```
postgres@joy-VM-UBUNTU01:~$ createdb test
postgres@joy-VM-UBUNTU01:~$ createuser -P joy
Enter password for new role:
Enter it again:
postgres@joy-VM-UBUNTU01:~$ psql
psql (16.4 (Ubuntu 16.4-0ubuntu0.24.04.2))
Type "help" for help.
```

Création d'une BDD « test » et de l'utilisateur « joy »

```
postgres=# GRANT ALL PRIVILEGES ON DATABASE test TO joy;
GRANT
postgres=# \q
postgres@joy-VM-UBUNTU01:~$ exit
déconnexion
```

Attribution des droits sur la base de données puis déconnexion

pgadmin

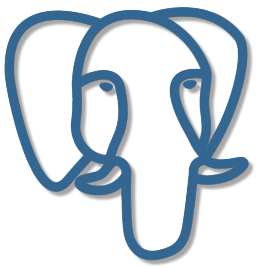
pgAdmin est **un outil graphique** qui facilite la gestion des bases de données **PostgreSQL**. Il permet une interaction visuelle et intuitive avec les bases, notamment pour exécuter des requêtes, créer ou modifier des schémas, et gérer les utilisateurs.

```
1  sudo apt  install curl  # version 8.5.0-2ubuntu10.4
```

Installation de l'utilitaire curl pour la gestion des téléchargements et des clés GPG

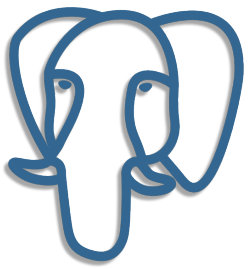
```
    3  curl https://www.pgadmin.org/static/packages_pgadmin_org.pub | gpg
--dearmor | sudo tee /usr/share/keyrings/pgadmin-keyring.gpg > /dev/null
    4  echo "deb [signed-by=/usr/share/keyrings/pgadmin-keyring.gpg] https
://ftp.postgresql.org/pub/pgadmin/pgadmin4/apt/$(lsb_release -cs) pgadmin4
 main" | sudo tee /etc/apt/sources.list.d/pgadmin4.list
```

Ajout de la clé GPG et configuration du dépôt pgAdmin4



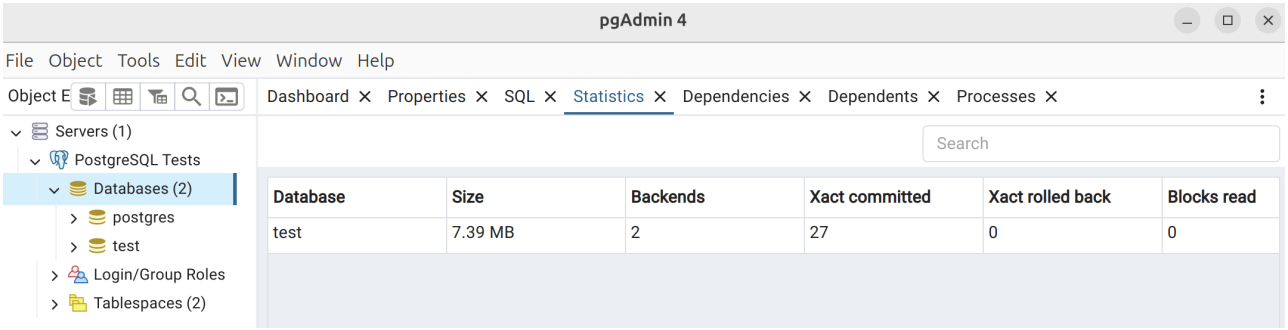
```
 7  sudo apt upgrade pgadmin4
```

Mise à jour de pgAdmin vers la dernière version



```
 9  sudo snap install pgadmin4
10  pgadmin4
```

Installation via Snap puis lancement de pgAdmin



Configuration et test de pgadmin pour accéder à la BDD « test » crée précédemment en ligne de commande

Installation de PHP

```
109  sudo apt install php php-cli php-pgsql php-json php-curl php-mbstring php-xml -y
110  sudo apt update && sudo apt upgrade -y
111  php -v
```

Installation, mise à jour puis vérification de la version PHP

Installation de VSCode

Pour mes besoins de développement, j'ai choisi d'utiliser **Visual Studio Code** (VS Code), un éditeur de code léger, **polyvalent** et adapté à de nombreux langages de programmation, dont PHP.
Ce choix est avant tout personnel, motivé par son **interface intuitive**, ses nombreuses **extensions** disponibles, et sa **compatibilité** avec les outils modernes de développement. Il offre également des fonctionnalités avancées, comme **l'autocomplétion**, **le débogage intégré**, et la prise en charge native de **Git**, ce qui en fait un outil idéal pour optimiser ma productivité et la qualité de mon code.

```
joy@joy-VM-UBUNTU01:~$ history
    1  wget -qO- https://packages.microsoft.com/keys/microsoft.asc | gpg --dearmor >
 microsoft.gpg
    2  sudo install -o root -g root -m 644 microsoft.gpg /usr/share/keyrings/
    3  rm microsoft.gpg
```

Télécharger et installer la clé de dépôt Microsoft

```
    4  echo "deb [arch=amd64 signed-by=/usr/share/keyrings/microsoft.gpg] https://packages
.microsoft.com/repos/vscode stable main" | sudo tee /etc/apt/sources.list.d/vscode.list
    5  sudo apt update
    6  sudo apt upgrade
    7  sudo apt install code -y
    8  code
```

Ajout du dépôt, mise à jour de la source, installation puis lancement de VSCode

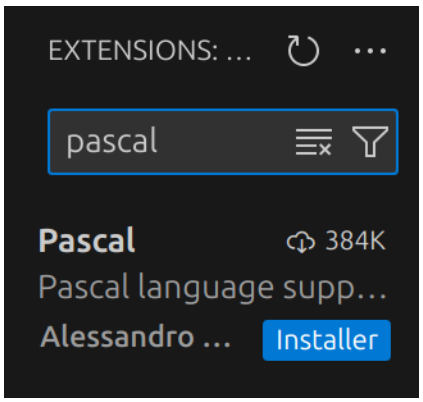
Visual Studio Code a été installé avec succès sur la machine virtuelle. La configuration et **l'ajout d'extensions** seront réalisés progressivement, en fonction des besoins spécifiques rencontrés lors des différents projets et développements effectués durant la suite de mon stage et de ma formation.

Installation des outils nécessaires à l'utilisation du langage Pascal

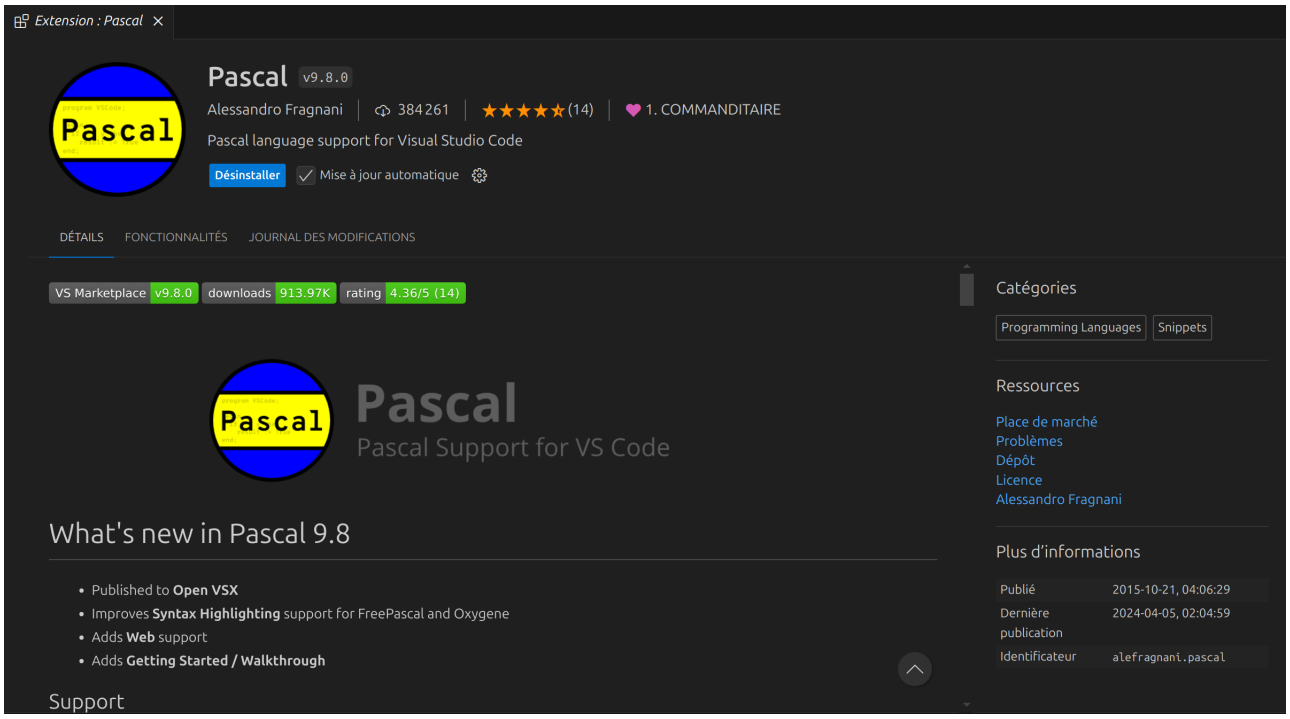
Le choix d'utiliser **Pascal** avec **Visual Studio Code** (VS Code) (et non avec Lazarus) repose sur plusieurs considérations pratiques et techniques.
Pascal, bien qu'un langage moins courant aujourd'hui, reste pertinent pour des applications nécessitant une gestion stricte des types, une syntaxe claire, et une structure pédagogique, particulièrement dans des contextes éducatifs ou des projets nécessitant une base logicielle robuste.
Visual Studio Code, en revanche, est un éditeur de texte moderne, extensible et largement adopté dans le monde du développement. Grâce à sa flexibilité et à sa richesse d'extensions, il permet d'intégrer efficacement des langages plus anciens comme Pascal dans un environnement de travail contemporain.
En utilisant **VS Code** avec le **Free Pascal Compiler (FPC)** et l'extension **Pascal**, il est possible de bénéficier des fonctionnalités avancées de l'éditeur tout en travaillant avec un langage classique.

```
1  sudo apt install fpc
2  sudo apt update
3  fpc -v
```

Installation de fpc



Recherche de l'extension Pascal dans VS Code



L'extension Pascal a été installée avec succès

Installation de git

Git est un système de **gestion de versions** distribué largement utilisé dans le développement logiciel. Il permet de suivre les modifications apportées au code source, de **collaborer** efficacement avec d'autres développeurs et de gérer différentes **versions** d'un projet. Sa flexibilité, sa rapidité et son efficacité en font un outil incontournable pour tout développeur.

```
2  sudo apt-get install git
```

Téléchargement et préparation du paquet Git avec APT

```
4  sudo add-apt-repository ppa:git-core/ppa
5  sudo apt update
6  sudo apt upgrade -y
7  sudo apt install git
```

Ajout du dépôt et installation de Git

Compétences Mobilisées

Gérer le patrimoine informatique

- **Recenser et identifier les ressources numériques** : L'installation et la configuration de PostgreSQL, pgAdmin, PHP, Visual Studio Code (VSCode) et Git impliquent une analyse des outils nécessaires au projet. L'utilisation des gestionnaires de paquets **apt** et **snap**, reposant sur des dépôts officiels, m'a permis d'exploiter les catalogues de logiciels disponibles pour installer les outils nécessaires, conformément aux standards en vigueur.
- **Exploiter des référentiels, normes et standards adoptés par le prestataire informatique** : L'utilisation de VSCode, compatible avec divers langages, et Git, standard de gestion de versions, illustre une adoption de pratiques standardisées dans l'industrie.

Mettre à disposition des utilisateurs un service informatique

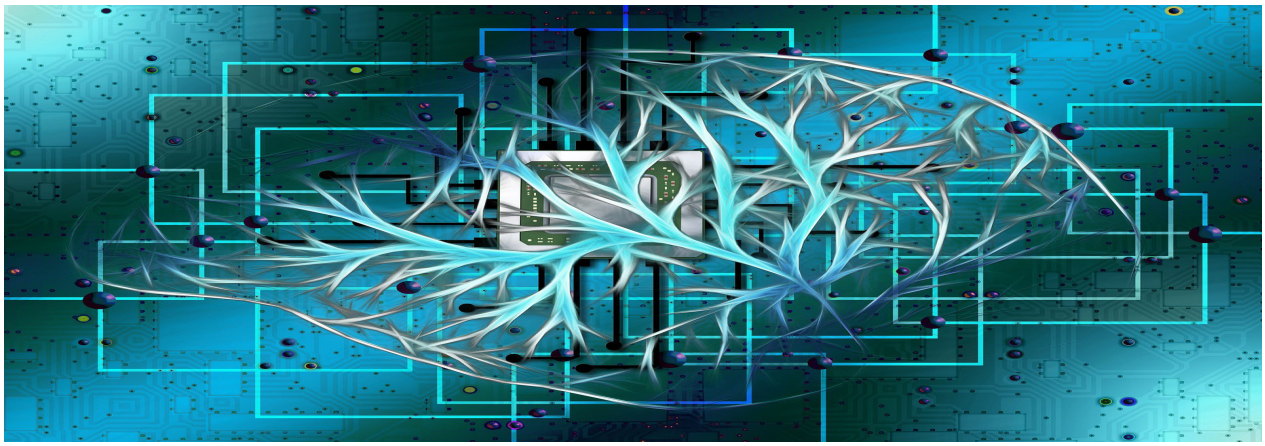
- **Réaliser les tests d'intégration et d'acceptation d'un service** : Les tests réalisés sur PostgreSQL (connexion à la base, création d'une base de données et d'un utilisateur) et sur pgAdmin (connexion à la base depuis une interface graphique) valident cette compétence.
- **Déployer un service** : L'installation et la configuration des outils de développement (PostgreSQL, pgAdmin, PHP, etc.) constituent des déploiements réussis.
- **Accompagner les utilisateurs dans la mise en place d'un service** : La documentation des étapes d'installation et de configuration pourrait être utilisée pour accompagner des utilisateurs futurs.

Organiser son développement professionnel

- **Mettre en place son environnement d'apprentissage personnel** : La configuration de VSCode et l'ajout d'extensions (comme pour Pascal) démontrent ma capacité à construire un environnement de travail adapté.
- **Mettre en œuvre des outils et stratégies de veille informationnelle** : Les choix techniques pour l'installation (par exemple, choix de VSCode pour sa flexibilité ou Git pour la gestion des versions) ont été guidé par une recherche active d'outils adaptés au contexte professionnel.

Bilan des installations

La première étape de ce projet a permis de poser les bases techniques nécessaires pour aborder les futures tâches avec un environnement stable, sécurisé et bien configuré. Grâce au déploiement d'une machine virtuelle sous Ubuntu, à l'installation des logiciels requis, et à la mise en place d'outils adaptés comme PostgreSQL, Visual Studio Code et Git, un cadre de travail performant et optimisé a été construit.
Ce processus rigoureux garantit une efficacité et une flexibilité maximales pour les prochaines étapes du stage, tout en respectant les standards professionnels. La configuration a été pensée pour évoluer au gré des besoins, assurant ainsi une adaptation continue aux exigences des projets.



Développement d'un prototype de mini SaaS client avec authentification

Après avoir préparé mon environnement de développement, j'ai amorcé un projet de connexion, pour les clients abonnés, à une page leur donnant accès à des outils d'expertise et d'analyse de bases de données.
Cette interface a été développée en PHP et PostgreSQL.
Ce mini-tableau de bord connecte les utilisateurs abonnés, leur donnant accès à des services.

Base de données

J'ai conçu une base PostgreSQL iknae contenant une table **clients** avec les champs suivants : identifiant, prénom, nom, entreprise, email, date d'inscription, statut (activé/suspendu). L'objectif était de simuler une base de gestion des comptes clients que l'entreprise pourrait utiliser pour contrôler les accès à ses services.

Création de la BDD

J'ai commencé par créer **la base de données iknae** appartenant à l'utilisateur joy.

```
joy@joy-VM-UBUNTU01:~$ sudo -u postgres psql
psql (16.8 (Ubuntu 16.8-0ubuntu0.24.04.1))
Type "help" for help.

postgres=# \dt
Did not find any relations.
postgres=# \du
                             List of roles
 Role name |                         Attributes
-----------+------------------------------------------------------------
 joy       |
 postgres  | Superuser, Create role, Create DB, Replication, Bypass RLS

postgres=# CREATE DATABASE iknae;
CREATE DATABASE
postgres=# GRANT ALL PRIVILEGES ON DATABASE iknae TO joy;
GRANT
postgres=# \q
```

Création et attribution de la BDD iknae

Création de la table et insertion de données

J'ai créé un schéma de données simple permettant de stocker les informations de chaque client, incluant prénom, nom, entreprise, email, date d'inscription, et un statut d'activation de l'abonnement.

J'ai ensuite inséré un jeu de données fictives représentant des clients afin de valider la structure et d'effectuer des tests de requêtes.

```
joy@joy-VM-UBUNTU01:~$ psql -U joy -d iknae;
psql (16.8 (Ubuntu 16.8-0ubuntu0.24.04.1))
Type "help" for help.

iknae=> CREATE TABLE clients (
  id SERIAL PRIMARY KEY,
  firstname VARCHAR(100),
  lastname VARCHAR(100),
  email VARCHAR(100) UNIQUE NOT NULL,
  password VARCHAR(100) NOT NULL,
  date_inscription DATE DEFAULT CURRENT_DATE,
  statut BOOLEAN DEFAULT TRUE
);
CREATE TABLE
iknae=> INSERT INTO clients (firstname, lastname, email, password, date_inscription, statut) VALUES
('Alice', 'Durand', 'alice.durand@example.com', 'azerty123', '2024-01-15', TRUE),
('Bob', 'Martin', 'bob.martin@example.com', 'motdepasse', '2023-12-10', FALSE),
('Claire', 'Petit', 'claire.petit@example.com', 'mdpClaire!', '2024-03-01', TRUE),
('David', 'Lemoine', 'david.lemoine@example.com', 'david123', '2024-04-20', FALSE),
('Emma', 'Thomas', 'emma.thomas@example.com', 'emmaTop42', DEFAULT, TRUE);
INSERT 0 5
iknae=> SELECT * FROM clients;
```

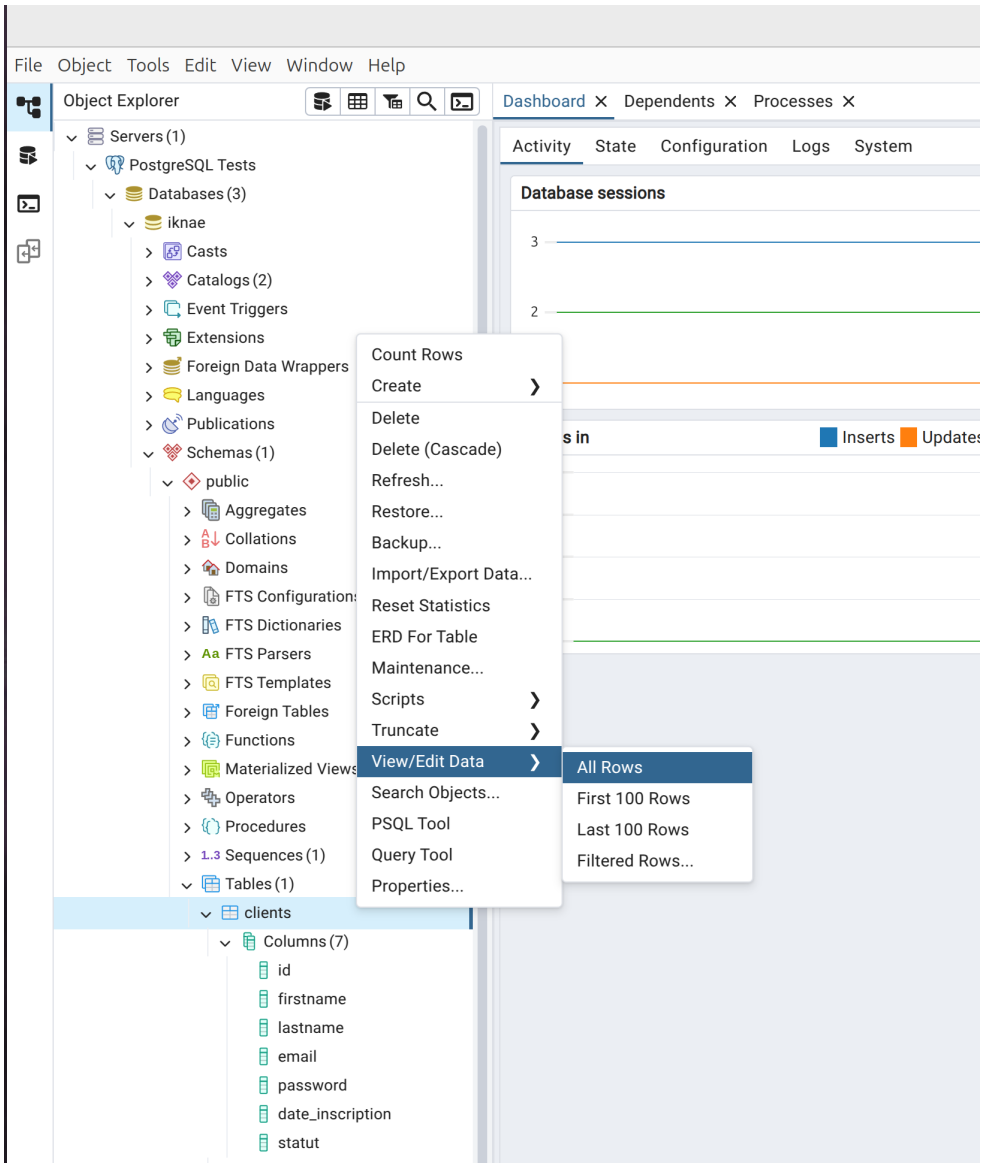
Création de la table clients puis insertion de données fictives

```
 id | firstname | lastname |          email          |  password  | date_inscription | statut
----+-----------+----------+-------------------------+------------+------------------+--------
  1 | Alice     | Durand   | alice.durand@example.com | azerty123  | 2024-01-15       | t
  2 | Bob       | Martin   | bob.martin@example.com   | motdepasse | 2023-12-10       | f
  3 | Claire    | Petit    | claire.petit@example.com | mdpClaire! | 2024-03-01       | t
  4 | David     | Lemoine  | david.lemoine@example.com | david123   | 2024-04-20       | f
  5 | Emma      | Thomas   | emma.thomas@example.com  | emmaTop42  | 2025-05-05       | t
(5 rows)
```

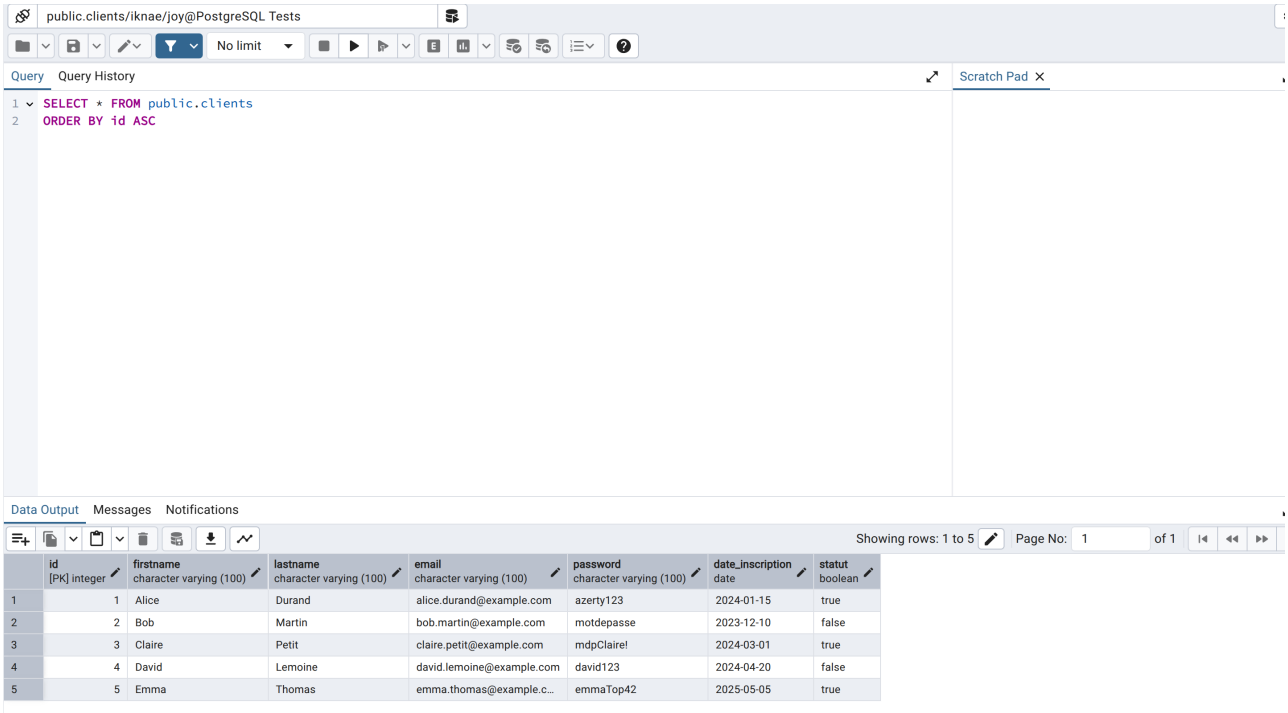
Contenu de la table clients après création et insertion des données

L'interface graphique de pgadmin

Ces données sont aussi visibles graphiquement dans pgadmin.



Accès à l'affichage de la table clients



	id [PK] integer	firstname character varying (100)	lastname character varying (100)	email character varying (100)	password character varying (100)	date_inscription date	statut boolean
1	1	Alice	Durand	alice.durand@example.com	azerty123	2024-01-15	true
2	2	Bob	Martin	bob.martin@example.com	motdepasse	2023-12-10	false
3	3	Claire	Petit	claire.petit@example.com	mdpClaire!	2024-03-01	true
4	4	David	Lemoine	david.lemoine@example.com	david123	2024-04-20	false
5	5	Emma	Thomas	emma.thomas@example.c...	emmaTop42	2025-05-05	true

Affichage de la table clients dans pgadmin

Interface web

Une fois la base de données et la table **clients** opérationnelles, j'ai conçu et développé une interface web en **html**, **CSS** et **PHP** permettant la gestion de comptes clients, intégrant un système d'authentification simple, une interface de visualisation, et un contrôle d'accès basé sur le statut de l'abonnement.

Connexion à la base de données PostgreSQL

Un fichier **db.php** a été créé pour assurer la connexion à la base **PostgreSQL** via **PDO (PHP Data Objects)**.
L'utilisation de **PDO** présente plusieurs avantages :

- une sécurité renforcée car PDO permet l'utilisation de requêtes préparées, ce qui protège efficacement contre les attaques de type injection SQL.
- une gestion fine des erreurs avec les exceptions. PDO permet de capturer et de traiter proprement les erreurs de connexion ou d'exécution.
- l'interface est orientée objet ce qui améliore la structure et la lisibilité du code, notamment pour les projets amenés à évoluer.

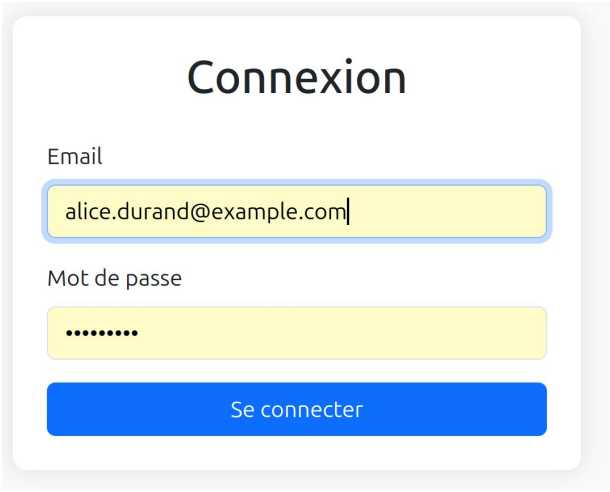
La gestion des erreurs a été prévue dès la connexion, afin de garantir une stabilité du service.

Fichier db .php et arborescence des fichiers

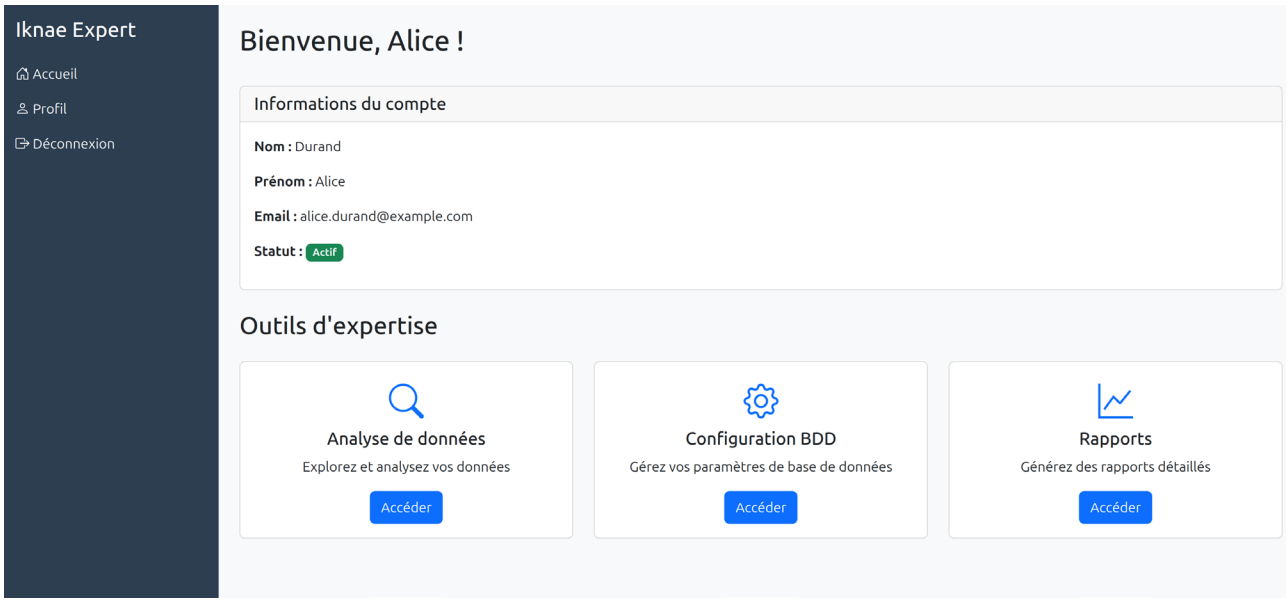
Authentification utilisateur

L'écran d'accueil du site présente une interface de connexion. Lors de la soumission, les informations saisies sont vérifiées depuis la table **clients**. Si le compte existe et que le mot de passe est correct :

- un statut de client actif autorise l'accès

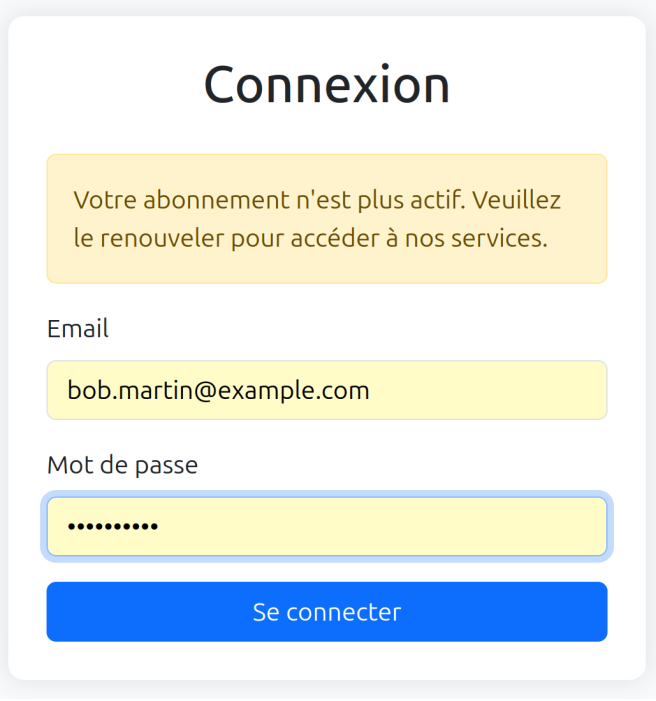


Connexion d'un statut d'abonnement actif



Affichage de l'accueil pour l'utilisateur connecté

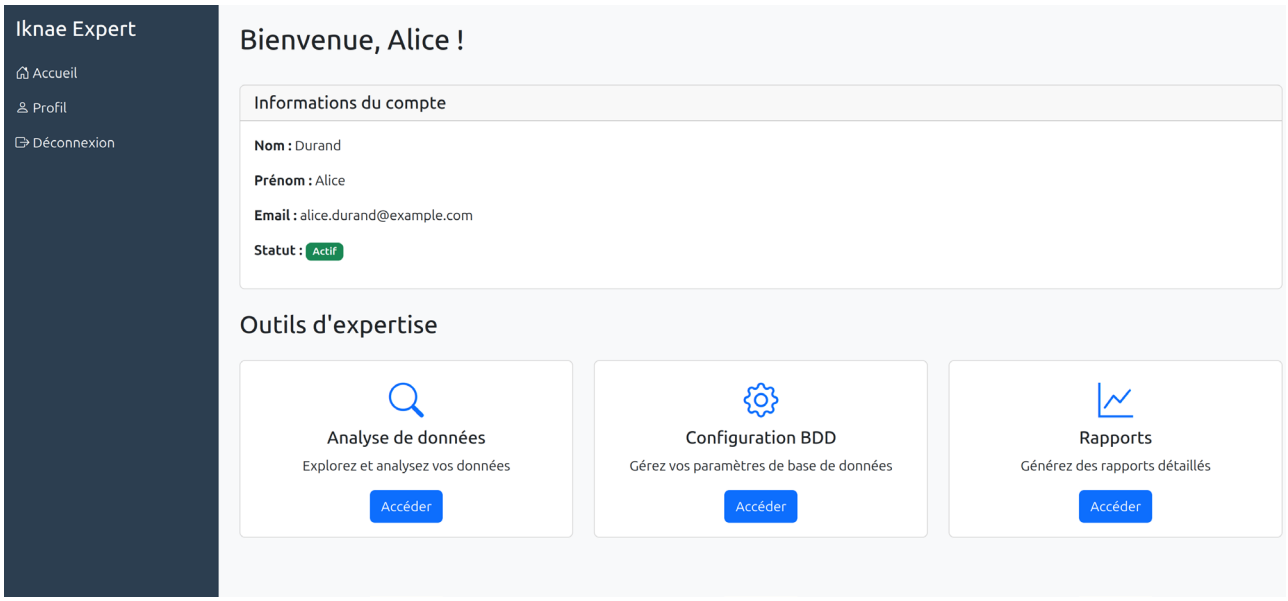
- un statut inactif refuse l'accès et incite l'utilisateur à s'abonner



Message affiché en cas d'abonnement expiré

Interface du tableau de bord

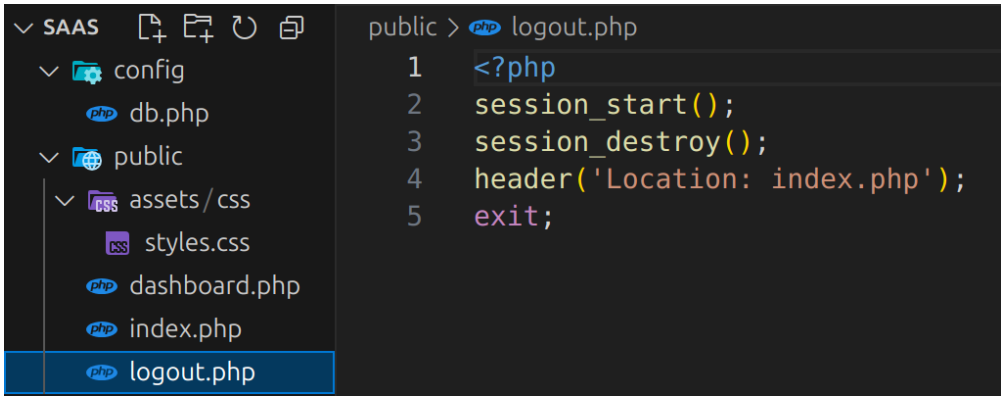
Une fois connecté, l'utilisateur est redirigé vers sa page d'accueil personnalisée qui affiche les informations du compte, le statut d'abonnement, des liens vers de futures fonctionnalités d'analyse ou de configuration.



Affichage de l'accueil pour l'utilisateur connecté

Déconnexion sécurisée

Un bouton "Déconnexion" permet de détruire la session et revenir à l'écran de login.



Fichier logout.php pour la déconnexion

Environnement de test et de développement

Afin de développer et tester l'application en toute autonomie, j'ai mis en place un environnement de test local sur ma machine virtuelle Ubuntu. Celui-ci intègre :

- **PHP 8.3**, utilisé avec le serveur de développement intégré (**php -S**) pour exécuter l'application web sans serveur Apache externe.
- **PostgreSQL 16**, configuré localement pour stocker les données clients, avec des accès contrôlés via un utilisateur dédié (**joy**) et une base nommée **iknae**.
- **pgAdmin**, utilisé pour visualiser la structure des tables, vérifier les requêtes SQL et manipuler les données de manière graphique pendant le développement.
- **Visual Studio Code** comme éditeur principal, avec une organisation du code en dossiers **(/config, /public, /assets**), facilitant la lisibilité et la modularité.

L'application a été lancée à l'adresse locale suivante :

http://localhost:8000

```
joy@joy-VM-UBUNTU01:~/Projets/Iknae/Saas$ cd public
joy@joy-VM-UBUNTU01:~/Projets/Iknae/Saas/public$ php -S localhost:8000
[Mon May  5 19:58:08 2025] PHP 8.3.6 Development Server (http://localhost:8000) started
```

Ce choix d'un environnement local m'a permis de tester rapidement les fonctionnalités sans dépendre d'un serveur de production, de corriger facilement les erreurs PHP grâce aux messages en direct, de manipuler les sessions et la base PostgreSQL de manière sécurisée, de capturer les interfaces et l'activité SQL à des fins de documentation et de démonstration.

Intégration de GitHub pour le versionnage et la centralisation du projet

Dans une logique de professionnalisation et de bonnes pratiques de développement, j'ai intégré **Git** comme système de gestion de versions et **GitHub** comme plateforme d'hébergement distant du code source. Cette démarche répondait à plusieurs objectifs concrets :

- Assurer une sauvegarde sécurisée et continue de l'ensemble du projet.
- Partager facilement mon travail avec ma tutrice.
- Structurer le développement grâce à l'utilisation de branches, de commits et d'un historique détaillé.
- Me familiariser avec des outils largement utilisés en entreprise pour le travail collaboratif.

Le choix d'un dépôt privé sur GitHub permettait de conserver le projet confidentiel, tout en autorisant un accès ciblé à des personnes de confiance (comme ma tutrice), grâce à un système d'invitation sécurisé.

Initialisation du dépôt Git local

Une fois Git installé, je me suis positionnée dans le dossier du projet et j'ai initialisé un dépôt Git local.

```
^Cjoy@joy-VM-UBUNTU01:~/Projets/Iknae/Saas/public$ cd ..
joy@joy-VM-UBUNTU01:~/Projets/Iknae/Saas$ git init
astuce : Utilisation de 'master' comme nom de la branche initiale. Le nom de la branche
astuce : par défaut peut changer. Pour configurer le nom de la branche initiale
astuce : pour tous les nouveaux dépôts, et supprimer cet avertissement, lancez :
astuce :
astuce :       git config --global init.defaultBranch <nom>
astuce :
astuce : Les noms les plus utilisés à la place de 'master' sont 'main', 'trunk' et
astuce : 'development'. La branche nouvellement créée peut être rénommée avec :
astuce :
astuce :       git branch -m <nom>
Dépôt Git vide initialisé dans /home/joy/Projets/Iknae/Saas/.git/
```

Initialisation du dépôt git local

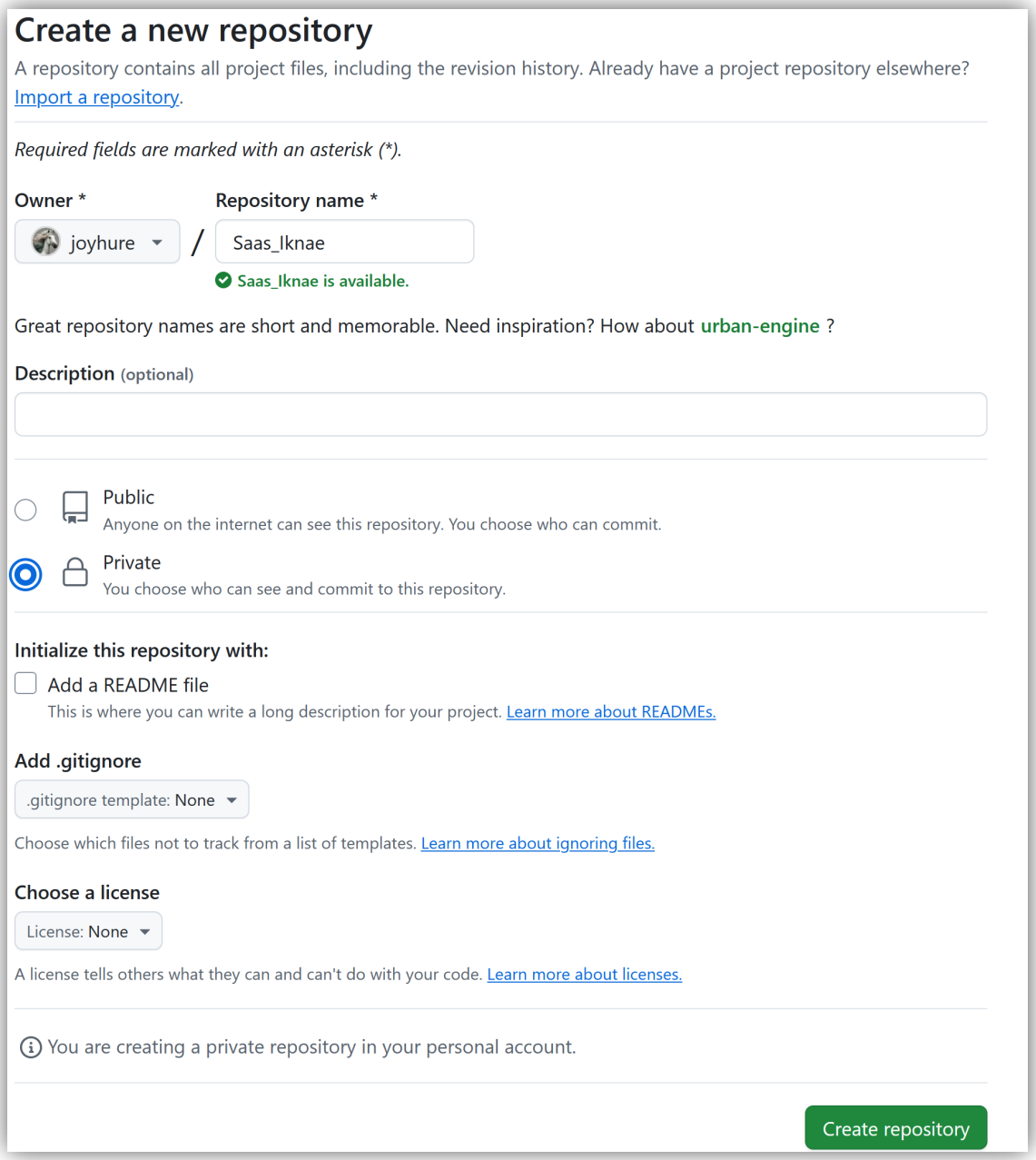
J'ai ensuite ajouté tous les fichiers existants et réalisé un premier commit.

```
joy@joy-VM-UBUNTU01:~/Projets/Iknae/Saas$ git add .
joy@joy-VM-UBUNTU01:~/Projets/Iknae/Saas$ git commit -m "Initialisation du projet Saas"
[master (commit racine) 048efd2] Initialisation du projet Saas
 6 files changed, 223 insertions(+)
 create mode 100644 Ressources/mdpid
 create mode 100644 config/db.php
 create mode 100644 public/assets/css/styles.css
 create mode 100644 public/dashboard.php
 create mode 100644 public/index.php
 create mode 100644 public/logout.php
```

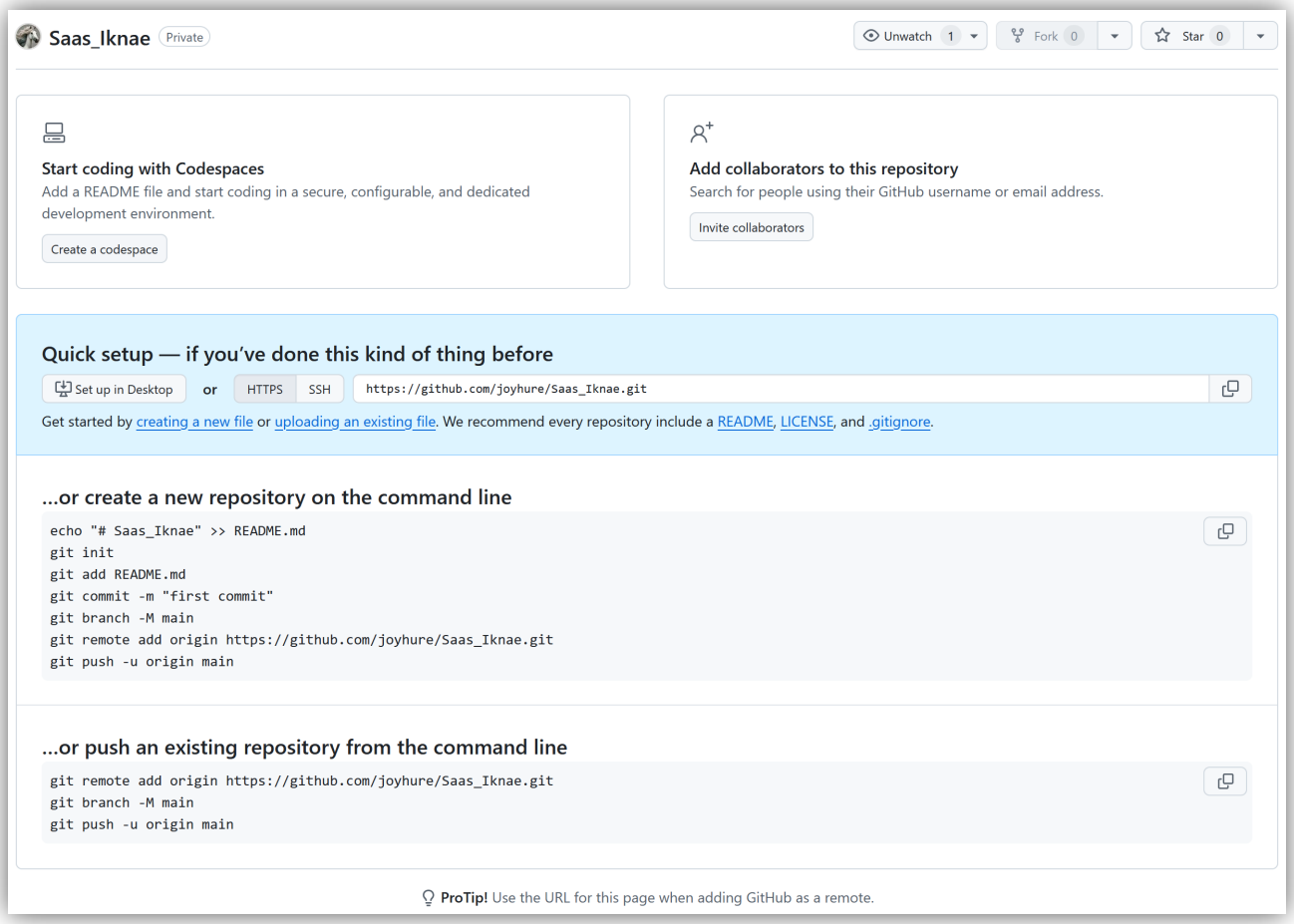
Premier commit « Initialisation du projet Saas »

Création et configuration du dépôt distant sur GitHub

Depuis l'interface web de GitHub, j'ai créé un dépôt privé nommé **Saas_Iknae**. Ce dépôt a pour but d'héberger l'ensemble du code source, de pouvoir y accéder depuis n'importe quel poste, et de le partager avec ma tutrice.



Création du dépôt distant



Le dépôt distant a bien été crée

Connexion du dépôt local à GitHub

Une fois le dépôt distant créé, j'ai connecté mon projet local à ce dépôt puis j'ai exécuté un **push** pour envoyer l'historique de mon projet sur GitHub, et créer en ligne cette branche initiale.

```
joy@joy-VM-UBUNTU01:~/Projets/Iknae/Saas$ git remote add origin https://github.com/joyhure/Saas_Iknae.git
joy@joy-VM-UBUNTU01:~/Projets/Iknae/Saas$ git branch -M master
joy@joy-VM-UBUNTU01:~/Projets/Iknae/Saas$ git push -u origin master
Énumération des objets: 13, fait.
Décompte des objets: 100% (13/13), fait.
Compression par delta en utilisant jusqu'à 6 fils d'exécution
Compression des objets: 100% (8/8), fait.
Écriture des objets: 100% (13/13), 3.47 Kio | 1.16 Mio/s, fait.
Total 13 (delta 0), réutilisés 0 (delta 0), réutilisés du paquet 0 (depuis 0)
To https://github.com/joyhure/Saas_Iknae.git
 * [new branch]      master -> master
la branche 'master' est paramétrée pour suivre 'origin/master'.
```

Connexion au dépôt local et push

J'ai ensuite renommé la branche principale locale en **main** (convention actuelle) et effectué un nouveau **push** vers GitHub.

```
joy@joy-VM-UBUNTU01:~/Projets/Iknae/Saas$ git branch -M main
joy@joy-VM-UBUNTU01:~/Projets/Iknae/Saas$ git push -u origin main
Total 0 (delta 0), réutilisés 0 (delta 0), réutilisés du paquet 0 (depuis 0)
remote:
remote: Create a pull request for 'main' on GitHub by visiting:
remote:      https://github.com/joyhure/Saas_Iknae/pull/new/main
remote:
To https://github.com/joyhure/Saas_Iknae.git
 * [new branch]      main -> main
la branche 'main' est paramétrée pour suivre 'origin/main'.
```

Branche renommée en « main » et push

Compétences mobilisées

Gérer le patrimoine informatique

- **Recenser et identifier les ressources numériques** : La base de données PostgreSQL a été structurée de manière à représenter un modèle réel de gestion des clients abonnés avec des champs pertinents (statut, date d'inscription, entreprise…).
- **Exploiter des référentiels, normes et standards adoptés par le prestataire informatique** : Utilisation de PostgreSQL, pgAdmin et PHP avec PDO pour la gestion de la BDD et de l'interface web, selon des pratiques standardisées du secteur.
- **Vérifier les conditions de continuité d'un service informatique** : L'environnement de test local permet de valider la stabilité de l'application avant un éventuel déploiement réel.

Travailler en mode projet

- **Analyser les objectifs et les modalités d'organisation d'un projet** : À partir du besoin exprimé par la tutrice (suivi d'accès des clients abonnés), j'ai identifié les fonctionnalités prioritaires (connexion, affichage selon le statut, interface simple), puis défini un périmètre technique réaliste compatible avec mon niveau à ce moment du stage.

- **Planifier les activités** : J'ai découpé le projet en étapes de développement distinctes : conception de la base de données, réalisation de l'authentification, affichage conditionnel, et tests. Chaque étape a été réalisée dans un ordre logique, validée avant de passer à la suivante.

Mettre à disposition des utilisateurs un service informatique

- **Réaliser les tests d'intégration et d'acceptation d'un service** : L'application a été testée en local (connexion, authentification, affichage dynamique) avec un serveur intégré PHP et une base PostgreSQL locale.
- **Accompagner les utilisateurs dans la mise en place d'un service** : L'interface développée est pensée pour une prise en main simple des clients, avec messages d'erreurs clairs et gestion d'état (actif/inactif).

Bilan

Le développement de ce prototype m'a permis d'apprendre, de me documenter en autonomie et de mettre en pratique plusieurs compétences clés du métier de développeur en environnement professionnel.

J'ai appris à structurer une base de données exploitable, à développer une interface web complète et fonctionnelle, et à intégrer les problématiques de gestion des utilisateurs et d'accès sécurisé aux services. Le projet, bien que simplifié par rapport à un produit final, représente une base solide pour une future évolution vers un vrai SaaS complet.

Ce travail m'a également permis de renforcer ma capacité à travailler de manière autonome, à documenter mes actions, et à utiliser des outils professionnels comme GitHub et PostgreSQL dans un contexte réel.