**Guide d’utilisation (rapide) V1.0**

1. **Ouvrir l’appli**
   * Dans PowerShell, place-toi dans le dossier du projet :  
     cd "C:\Users\Utilisateur\Documents\edt\_app"
   * Active l’environnement :  
     .\.venv\Scripts\activate
   * Lance :  
     python -m streamlit run app.py
   * L’appli s’ouvre sur http://localhost:8501.
2. **Éditer la semaine**
   * Dans **Édition**, double-clique une cellule → tape le texte (ex : CM Processus).
   * **Astuce** : garde toujours les mêmes mots-clés pour la couleur auto :
     + CM, TD, Cours → Cours
     + BU, Projet → Bibliothèque / projet
     + Musculation
     + Club photo, Court métrage
     + RU → repas
3. **Sauvegarder / Charger**
   * **Nom de la semaine** (barre latérale) : Semaine 1, 2025-09-22, etc.
   * Boutons :
     + **Créer semaine vide** → crée un nouveau fichier dans data/
     + **Charger semaine** → ouvre le CSV existant
     + **Sauvegarder semaine** → enregistre tes modifs
   * Les fichiers sont stockés dans le dossier **data/** au format CSV (un par semaine).
4. **Vue esthétique**
   * La grille du bas colorie automatiquement les cases selon les mots-clés.
   * Si tu vois mal le texte, passe à la **version haute-contraste** ci-dessous.

**UTILISE CE CODE DANS WINDOWS POWERSHELL :**

**cd "C:\Users\Utilisateur\Documents\edt\_app"**

**.\.venv\Scripts\activate**

**pip install -r requirements.txt**

**python -m streamlit run app.py**

ton dossier edt\_app est bien organisé ! Voilà ce que chaque élément correspond :

### 📂 .venv

* C’est ton **environnement virtuel Python**.
* Il contient toutes les librairies installées (streamlit, pandas, etc.).
* ⚠️ **Ne doit jamais aller sur GitHub** → c’est trop lourd et inutile.
* Les dépendances doivent être listées dans requirements.txt à la place.

### 📂 data

* Dossier pour stocker tes données locales (CSV, DB, exports…).
* ⚠️ **À ignorer dans GitHub** si tu as des fichiers persos ou lourds.
* Si tu veux partager des exemples, tu peux mettre un sous-dossier data/examples/.

### 🐍 app (fichier Python, sûrement app.py)

* Ton **script principal Streamlit**.
* C’est ce fichier que Streamlit Cloud doit lancer (Main file path = app.py).

### 📄 guide utilisateur.docx

* Documentation utilisateur (ton mode d’emploi).
* Tu peux le pousser sur GitHub si tu veux partager la doc.
* Sinon, facultatif.

### 📄 README.md

* Fichier d’accueil du repo GitHub (format Markdown).
* Sert à expliquer ton projet : titre, capture d’écran, instructions d’installation.
* Très utile pour présenter ton projet.

### 📄 requirements (fichier texte)

* Ici il s’appelle juste requirements (sans .txt ?).
* Normalement : requirements.txt.
* Contient la liste des librairies Python nécessaires.
* Exemple de contenu :

**V1.1 : Modifier depuis son téléphone + accès et mesures de sécurité**

## 🔹 1. SQLAlchemy → à quoi ça sert ?

* C’est une **librairie Python** pour se connecter et dialoguer avec une base de données.
* Elle te permet d’utiliser **SQLite, PostgreSQL, MySQL…** avec la même syntaxe Python.
* Exemple d’usage typique :

from sqlalchemy import create\_engine

import pandas as pd

# Connexion à un fichier SQLite local

engine = create\_engine("sqlite:///data/forum.db")

# Lire une table SQL dans un DataFrame pandas

df = pd.read\_sql("SELECT \* FROM students", engine)

print(df.head())

* 👉 Utile pour ton projet “Master Query” : construire une **vue consolidée** qui combine plusieurs tables et l’afficher/modifier dans Streamlit.

## 🔹 2. openpyxl → à quoi ça sert ?

* C’est une lib pour **lire et écrire des fichiers Excel (.xlsx)**.
* Tu en auras besoin si tu veux :
  + Exporter tes données vers un fichier Excel.
  + Charger des fichiers Excel d’utilisateurs.
* Exemple simple :

import pandas as pd

df = pd.DataFrame({"Nom": ["Alice", "Bob"], "Note": [15, 12]})

df.to\_excel("data/export.xlsx", index=False, engine="openpyxl")

* 👉 Ça complète bien pandas pour offrir des **exports Excel** (souvent demandés en projets/stages).

## 🔹 3. Créer un .gitignore (recommandé)

C’est un fichier texte nommé exactement **.gitignore** à mettre à la racine de ton projet (edt\_app/).  
Tout ce qui est listé dedans sera **ignoré par Git/GitHub**.

### Étapes sous Windows :

1. Dans ton dossier edt\_app, clic droit → **Nouveau → Document texte**.
2. Nommer le fichier **.gitignore** (sans .txt à la fin → si Windows ajoute .txt, renomme en activant “afficher les extensions de fichiers” dans l’Explorateur).
3. Colle ce contenu dedans :

# Environnement virtuel Python

.venv/

# Données locales (CSV, DB, etc.)

data/

# Secrets Streamlit (mot de passe, clés…)

.streamlit/secrets.toml

# Bases de données

\*.db

\*.accdb

# Fichiers temporaires Python

\_\_pycache\_\_/

\*.pyc

# Fichiers temporaires Windows

Thumbs.db

# Fichiers temporaires Office

~$\*

👉 Après ça, quand tu feras ton **commit/push sur GitHub**, tout ce qui est dans .gitignore ne sera **jamais envoyé**.

**🔹 Axes possibles de différenciation**

**1. 🎨 Expérience utilisateur**

* **Couleurs et thèmes personnalisés** (par exemple une couleur par type d’activité : CM, TD, BU, projet).
* **Mode clair/sombre** (utile sur mobile).
* **Icônes visuelles** (📚 BU, 🎬 Court métrage, 📸 Club photo).

**2. 📱 Accessibilité & mobilité**

* **Version mobile fluide** (tu l’as déjà avec Streamlit Cloud).
* **Export rapide** vers **Google Calendar** ou **Outlook** pour synchro avec le téléphone.
* **QR Code** → scanne et ouvre directement ton emploi du temps sur smartphone.

**3. 📊 Suivi et motivation**

* Ajout de **statistiques personnelles** :
  + Temps passé en cours vs temps en projet.
  + Graphiques hebdomadaires (barres ou camemberts).
* **Objectifs hebdo** → l’appli peut afficher : “Tu voulais 6h de BU, tu en as fait 4h”.
* Mini **gamification** : badges (“🔥 3 semaines consécutives avec toutes les séances BU prévues”).

**4. 🤝 Collaboration / ouverture**

* Permettre à un camarade ou encadrant d’**ajouter un commentaire** sur une séance.
* Exporter le planning en **PDF/Excel** pour le partager.
* Mode “lecture seule” pour partager sans que les autres modifient.

**5. 🔒 Simplicité & sécurité**

* Accès lecture libre + édition avec mot de passe (déjà prévu).
* Journalisation des modifications (date + auteur → pratique si tu présentes ça comme un outil sérieux).

**🔹 Brainstorm rapide “fun & utile”**

* Quand tu ouvres l’app → un petit **message de motivation** (aléatoire, style “Aujourd’hui, avance sur ton projet SQL 💪”).
* Une **timeline animée** → qui défile et montre tes cours/activités de la journée.
* Un bouton “🎲 Planning surprise” → propose un créneau BU libre au hasard dans la semaine pour te motiver à y aller.
* Un **mode révision** → tu coches les chapitres/processus stochastiques étudiés, et ça s’intègre dans le planning.

👉 En résumé :  
Tu n’as pas besoin de rivaliser avec Sciforma en “gros outil”, mais tu peux séduire en faisant un **emploi du temps + tableau de bord étudiant motivant et fun**, facile à utiliser sur téléphone.

Veux-tu que je te fasse une **liste courte de 3 fonctionnalités concrètes à coder dès maintenant** (genre exports, stats perso, motivation) qui donneraient tout de suite cette plus-value ?

**✅ V1.1 (maintenant)**

But : **téléphone + sécurité + petit “wow” utile**.

* Déploiement **Streamlit Cloud** (depuis GitHub).
* 🔒 **Mode édition** protégé par mot de passe (via st.secrets).
* 📤 **Export Excel** de la semaine (openpyxl).
* 📊 **Stats perso** (heures par catégorie CM/TD/BU/Projet).
* 💬 **Message motivant aléatoire** à l’ouverture.
* (Option bonus rapide) 🎛️ **Thème clair/sombre** : st.set\_page\_config + st.theme.

Critère de fin : tu ouvres l’URL sur ton téléphone, tu peux **voir** sans mot de passe et **éditer** avec mot de passe, exporter en Excel, voir un petit graphe.

**V1.1.b (juste après, “polish UX”)**

* 🎨 **Thèmes & couleurs** par activité (palette paramétrable).
* 🖼️ **Icônes** (📚, 🎬, 📸) dans la grille et/ou légende.
* 📱 **QR code** vers l’URL de l’app (lib qrcode) pour ajout rapide sur mobile.

Critère de fin : tu peux changer de palette depuis la sidebar, générer un QR et l’afficher/télécharger.

**V1.2 (mobilité / intégrations sans API)**

* 📅 **Export calendrier** au format **ICS** (pas d’API externe) : bouton “Ajouter au calendrier”.
* 🗂️ **Export PDF** du planning (via pdfkit/weasyprint ou capture HTML → PNG → PDF).
* 💾 **Auto-save** discret (sauvegarde à chaque modif majeure).

Critère de fin : téléchargement d’un .ics importable dans Google Calendar/Outlook, PDF propre.

**V1.3 (motivation & suivi)**

* 🎯 **Objectif hebdo** d’heures BU/Projet + suivi (“6h visées / 4h réalisées”).
* 🏅 **Badges** (“3 semaines consécutives complètes”, “+10h BU/mois”).
* 🔔 **Rappels doux** dans l’UI (si objectif en retard).

Critère de fin : tableau de bord avec objectifs + badges calculés depuis l’historique CSV.

**V1.4 (partage & collaboration légère)**

* 👀 **Lien lecture seule** (paramètre ?readonly=1 ou profil “Viewer”).
* 💬 **Commentaires** par créneau (stockés dans data/notes.csv).
* 🧾 **Journal des modifications** (timestamp, auteur “pc/mobile”).

Critère de fin : on peut partager l’URL en lecture seule, ajouter un commentaire, voir un log.

**V1.5 (socle “Master Query” pour ton forum de stage)**

* 🗄️ **SQLite + SQLAlchemy** (structure tables + VIEW “master\_query”).
* 🔌 **Formulaires CRUD** simples (étudiants, offres, candidatures).
* 🔄 **Export Excel/CSV** de la master\_query.

Critère de fin : base locale fonctionnelle, vue consolidée consultable/exportable depuis l’app.

**Pile & modules par phase (à ajouter au requirements.txt)**

* **V1.1 / 1.b**: streamlit, pandas, openpyxl, qrcode
* **V1.2**: (au choix) ics *ou* génération ICS maison, weasyprint/pdfkit (si PDF)
* **V1.3**: (rien de plus, logique pandas)
* **V1.4**: (rien de plus, stockage CSV)
* **V1.5**: SQLAlchemy, sqlite3 (standard lib)

**Concrètement : la toute prochaine étape**

1. **Finaliser V1.1** (déploiement + mot de passe + export Excel + stats + message).
2. **Tag Git**: crée un tag v1.1.0 quand c’est ok (dans GitHub → Releases).
3. Passer à **V1.1.b** (thèmes/icônes/QR).

Si tu veux, je te fournis **tout de suite** un bloc de code prêt à coller dans app.py pour :

* le **mot de passe d’édition**,
* l’**export Excel**,
* les **stats** (bar chart),
* le **message motivant**,
* et l’option **QR code** (V1.1.b).