Items-Managerをお使い頂くために、必要な開発環境を準備します。
以下の環境構築手順に従って、準備を進めてください。

環境構築手順

<目次>

Pythonの開発環境を準備

---

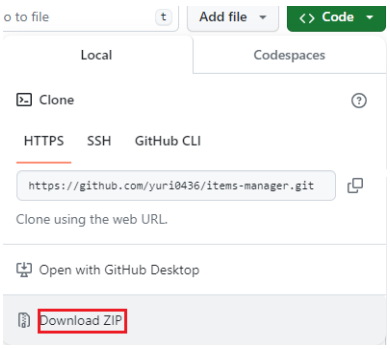
①**Python**のインストール
下記の手順を参考に、Pythonの最新パッケージをインストールしてください。
Pythonのインストール(Windows)
Pythonのインストール(Mac)

②テキストエディタのセットアップ
お使いのテキストエディタをご用意ください。
また、今回作成したWebAPI(items-manager)の開発には、VsCodeを使用しました。
まだインストールされていない方は、下記にVsCodeのセットアップ方法の参考資料を
記載しましたので、ご参照ください。
PythonのためのVisual Studio Codeの始め方

③必須ライブラリをインストール
開発で使用したライブラリをインストールします。
ターミナルで `pip list` コマンドを実行し、<ライブラリー覧>に該当するライブラリが
インストールされていない場合は、別途インストールをお願いします。

<ライブラリー覧>
・alembic
・fastapi
・httpx
・pytest
・pandas
・SQLAlchemy
・uvicorn

④**Github**からソースコード一式をダウンロード
・Github<items-manager>リポジトリから、ソースコードをダウンロードします。



・zipファイルを解凍し任意の場所に配置しましたら、テキストエディタから
対象フォルダを読み込んでください。

データベースを準備

---

①データベースをセットアップ

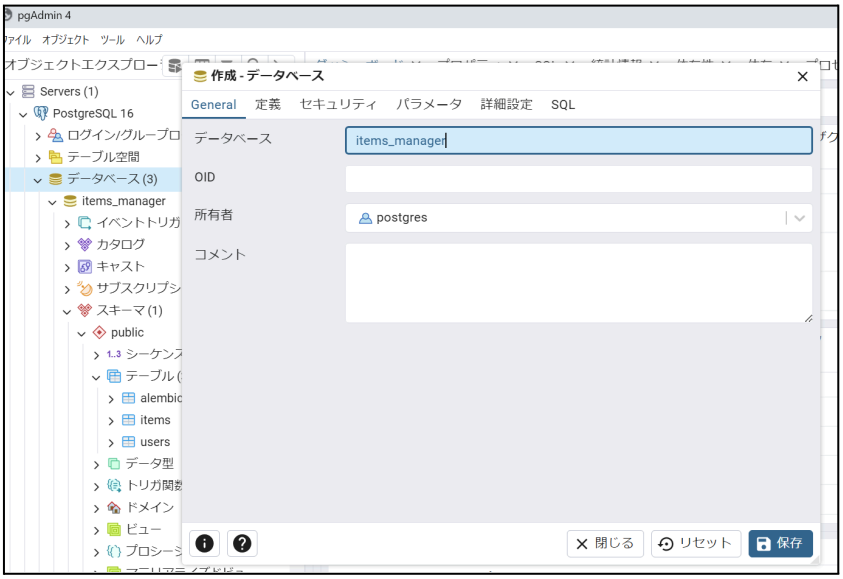
今回作成したWebaAPIは、Postgresqlを使用して開発を行いました。
まだデータベースの開発環境が整っていない方は、参考資料の手順に従い
Postgresqlのセットアップをお願いします。

PostgreSQLをインストール(Windows)
PostgreSQLをインストール(Mac)

設定したパスワードは忘れないように控えておいてください。

②データベース**<items_manager>**を作成する

データベースは、PostgresqlのGUIツールのpgAdminを使って操作していきます。
・pgAdminを立ち上げたら<Servers>をクリックします。
・パスワードが求められるので、先ほど控えたパスワードを入力してください。
・Serversを開いたら<データベース>の上で右クリックします。
作成>データベースを選択し、**<items_manager>**データベースの作成を
行ってください。



③データベースに設定したパスワードを設定ファイルに反映

〇[.env]にパスワードを反映

・データベースに設定したパスワードを、データベースに接続するためのURLに
反映させます。
・データベース接続URLは、リソースの設定ファイル(.env)に記述されています。

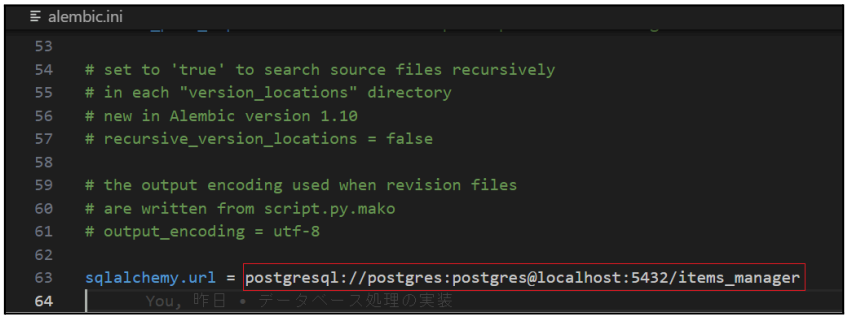
```
.env
#パスワード + SECRET_KEYをハッシュ関数(sha256)を使ってハッシュ化
SECRET_KEY = "c41bbdc76dfb9341a37b6cee828f36780eee629206a8004b5f1d2b0df5c653ff"

#データベース接続URL       "データベース名://ユーザー名:パスワード:@localhost:ポート番号/接続先のデータベース"
SQLALCHEMY_DATABASE_URL = "postgresql://postgres:postgres@localhost:5432/items_manager"
```

・赤枠で囲った部分を、データベースに設定したパスワードに変更してください。

○[alembic.ini]のデータベース接続URLを更新

・SQLALCHEMY_DATABASE_URLに設定したURLをコピーします。

・赤枠の部分のURLを削除し、コピーしたURLを貼り付けます。



④データベースにテーブル設定を反映

・alembicのマイグレーションを実行し、models.pyで定義したテーブル設定を
Postgresqlに反映させます。

・ターミナルから以下のコマンドを実行してください。

＞**alembic upgrade head**

・pgAdminを開き、以下のテーブル設定が反映されていることを確認してください

<itemsテーブル>

・id [int]
・name [String]
・price [int]
・description [String]
・category[String]
・status [String]
・stock [int]
・created_at [datetime]
・updated_at [datetime]
・user_id [int]

<usersテーブル>

・id [int]
・username [String]
・password [String]
・salt [String]
・created_at [datetime]
・updated_at [datetime]

動作検証

---

①サーバーを立ち上げる

・アプリケーションサーバーを立ち上げます。
ターミナルを開き、以下のコマンドを入力して実行してください。
＞**uvicorn main:app –reload**

実行結果が「Application startup complete」となっていれば、
サーバーの起動は成功です。

```
INFO:     Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO:     Started reloader process [8840] using WatchFiles
INFO:     Started server process [6344]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
```

②**Swagger UI**で生成されたドキュメントを表示

・上図の赤枠で囲っているリンクにアクセスしてください。
アクセスに成功すると、{"detail":"Not Found"}と表示されていると思います。

・今度は、リンクの後ろに「**/docs**」を追加してアクセスしてみてください。
「/docs」を追加してアクセスすることによって、
Swagger UIで生成されたドキュメントページを確認することが出来ます。
商品登録などのCRUD処理の操作は全て、このドキュメントを使用して
動作確認を行っていただきます。