Flask Blog Application

Features :

- **User Authentification :**
    - User can register, log in, and log out securely.
    - Passwords are hashed using **Flask-Bcrypt.**
- **Profile Management:**
    - User can update their account details, including uploading a profile picture.
    - Uploaded profile pictures are resized and saved securely.
- **Dynamic Content:**
    - The homepage dynamically displays a list of posts.
    - Posts include metada such as author and date.
- **Session Management:**
    - Implemented with **Flask-Login**, ensuring secure acess to authenticated routes like the account page.
- **Responsive Design:**
    - UI built with **Bootstrap** for responsive and clean design.

Technologies Used :

- **Backend :**
    - Flask: Core web framework.
    - Flask-SQLAlchemy: Database integration.
    - Flask-Bcrypt: Password hashing.
    - Flask-Login: User session management.
- **Frontend:**
    - Bootstrap: Styling and responsiveness.
    - Custom CSS: Additional Styling in **style.css**
- **Database:**
    - SQLite: Lightweight database for development.
- **Other Libraries:**
    - PIL: Used for image processing when resizing profile pictures.

File Structure:

- **run.py** : Entry point for running the Flask App.
- **Flaskblog/**:
    - **__init__.py :** Initializes the app and sets un extensions.
    - **routes.py :** Defines application routes and their logic.
    - **models.py :** Defines database models (**User** and **Post).**
    - **forms.py :** Contains WTForms classes for user input.
    - **static/:**
        - **style.css :** Custom styles for the app.
        - **profile_pics/:** Directory for used-uploaded profile pictures.
    - **templates/:**
        - **base.html:** Base template with shared layout.

- **home.html:** Homepage template displaying posts.
- **account.html:** Account management page.
- **register.html:** Register page.
- **login.html:** Login page
  - **instance/:**
    - **site.db:** SQLite database file storing user and post data.

Installation:

1- Clone this repository:

git clone <repository-url>
cd flask_project

2- Create a virtual environment:

python3 -m venv venv source venv/bin/activate

3- Run the application :

python run.py

Future Enhancements :

Add support for creating, uptading, and deleting posts