🚀 Project Setup Guide

📂 1. Create Project Folder

```
mkdir my_project
cd my_project
```

🌍 2. Create & Activate Virtual Environment

🔷 Install Virtual Environment

```
pip install virtualenv

python<version> -m venv myenv          # Example: python3 -m venv myenv
```

🔷 Activate Virtual Environment

- **Windows (cmd):**

  ```
  myenv\Scripts\activate
  ```

🔷 Install Dependencies

```
pip install <package_name>
```

🔷 Deactivate Virtual Environment

```
deactivate
```

💡 **Tip:** Always activate your virtual environment before working on your project!

🐙 3. Create a New GitHub Repository

🔷 Initialize Git & Push to GitHub

```
echo "# GITversion" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/Mohamed-mosad-hadia/GITverstion.
git
git push -u origin main
```

📂 4. Recommended Project Structure

```
your-project-root/      ← Main project folder
├── myenv/              ← Virtual environment (DO NOT edit or commit this)
├── src/                ← Your actual project code (Django, Flask, FastAPI, ML, et
c.)
│   ├── backend/        ← Backend API, database, models, etc.
│   ├── frontend/       ← Frontend (React, Angular, Vue, etc.)
│   ├── ml_model/       ← Machine learning models, Jupyter notebooks, etc.
│   ├── tests/          ← Unit and integration tests
│   └── scripts/        ← Utility scripts, automation tools
├── requirements.txt    ← Dependencies (run `pip freeze > requirements.txt
```

```
`)
├── manage.py          ← Django/Flask/FastAPI management script
├── README.md          ← Project documentation
├── .gitignore         ← Exclude `myenv/`, `__pycache__/`, `.env`, etc.
└── .env               ← Environment variables (DO NOT commit this)
```

💡 **Tip:** If you have a frontend, include it in a `Frontend-Template/` folder.

📌 5. Automate Requirements Management

🔷 Before Pushing Code

Always update `requirements.txt`:

```
pip freeze > requirements.txt
```

🔷 When Cloning on a New Machine

Install dependencies with:

```
pip install -r requirements.txt
```

⚙️ 6. Additional Enhancements (Recommended)

🛑 Add a `.gitignore` File

Create a `.gitignore` file and add:

```
# Virtual environment
myenv/
venv/

# Python cache
__pycache__/
*.pyc
*.pyo
```

```
*.pyd

# Environment variables & secrets
.env

# Node.js dependencies (if frontend exists)
node_modules/

# Compiled files
*.log
*.sqlite3
```

💡 **Tip:** This prevents unwanted files from being tracked by Git.

🔐 Store Environment Variables Securely

Create a `.env` file for sensitive credentials:

```
SECRET_KEY=your_secret_key
DEBUG=True
DATABASE_URL=your_database_url
```

Use `python-dotenv` to load variables in Python:

```
pip install python-dotenv
```

```
from dotenv import load_dotenv
import os

load_dotenv()
secret_key = os.getenv("SECRET_KEY")
```

✅ **Never commit `.env` to Git!**

🚀 Git Branching Strategy (For Teams)

- `main` → **Stable production branch**
- `dev` → **Active development branch**
- `feature-branch` → **Develop features separately**

```
git checkout -b dev  # Create & switch to the dev branch
git push origin dev  # Push it to GitHub
```

🐋 Docker Setup (If Needed)

For a **Dockerized setup**, create a `Dockerfile`:

```
FROM python:3.9
WORKDIR /app
COPY . .
RUN pip install -r requirements.txt
CMD ["python", "manage.py", "runserver", "0.0.0.0:8000"]
```

Then, build and run:

```
docker build -t my_project .
docker run -p 8000:8000 my_project
```

📝 Code Formatting & Linting

- **For Python**

  ```
  pip install black
  black .
  ```

- **For JavaScript (Frontend Projects)**

  ```
  npm install --save-dev eslint prettier
  ```

🎯 Final Thoughts

This guide ensures that your project is **structured, maintainable, and scalable** whether you're working on **Frontend, Backend, or Machine Learning**. 🚀

Let me know if you need help customizing it further! 🎯