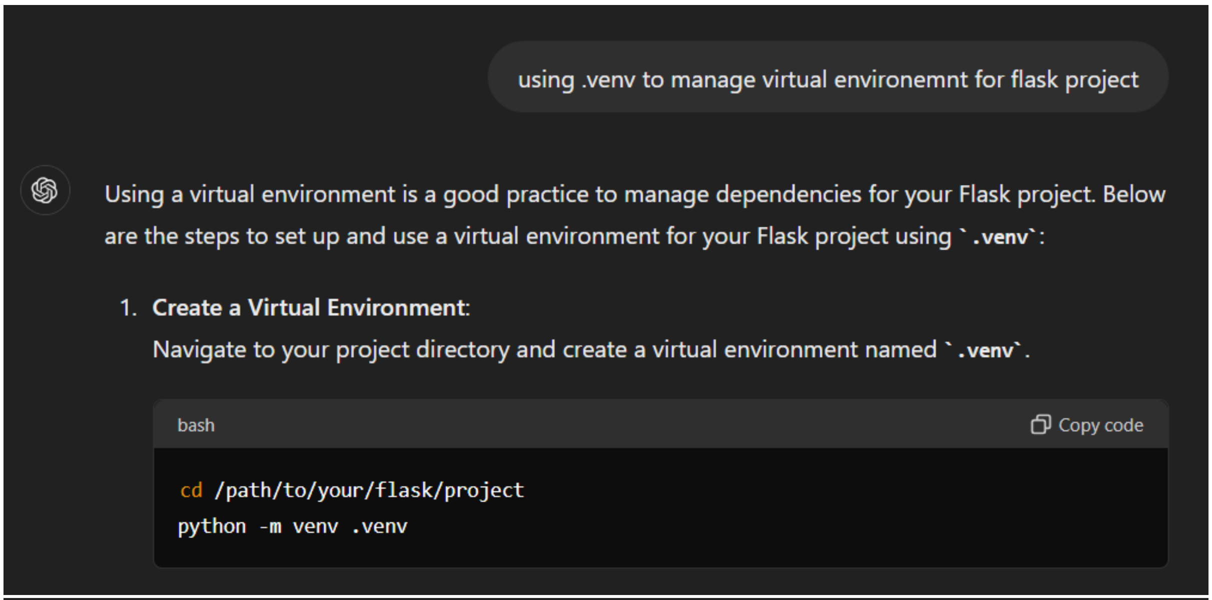
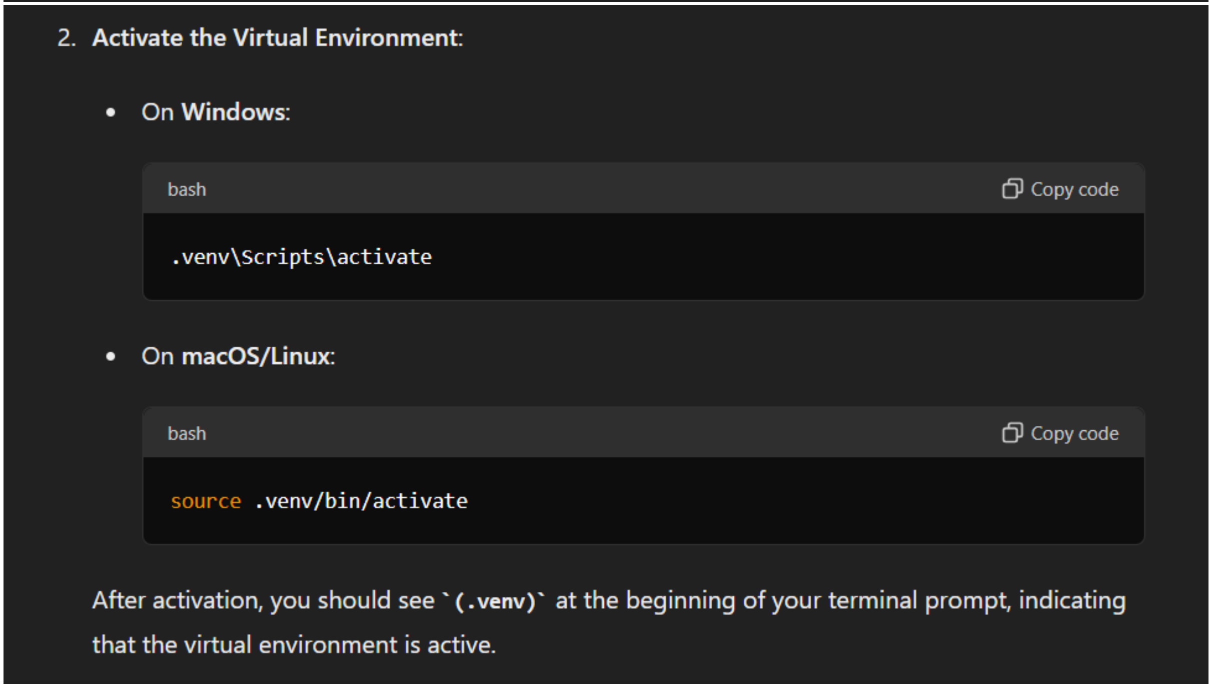

using .venv to manage virtual environemnt for flask project

Using a virtual environment is a good practice to manage dependencies for your Flask project. Below are the steps to set up and use a virtual environment for your Flask project using `.venv`:

1. **Create a Virtual Environment**:
   Navigate to your project directory and create a virtual environment named `.venv`.

   bash

   ```bash
   cd /path/to/your/flask/project
   python -m venv .venv
   ```


2. **Activate the Virtual Environment**:

   - On **Windows**:

     bash

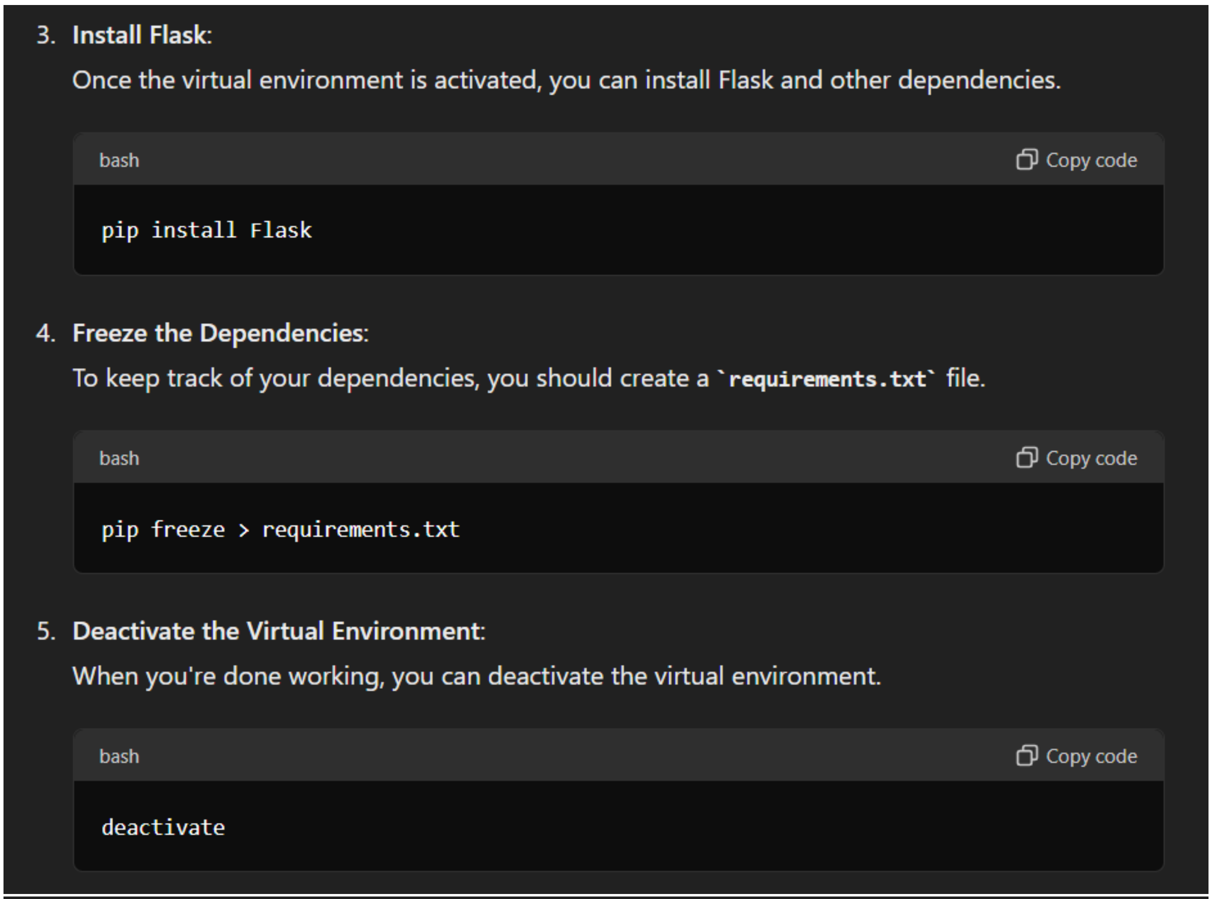
     ```bash
     .venv\Scripts\activate
     ```

   - On **macOS/Linux**:

     bash

     ```bash
     source .venv/bin/activate
     ```

   After activation, you should see `(.venv)` at the beginning of your terminal prompt, indicating that the virtual environment is active.


3. **Install Flask:**
   Once the virtual environment is activated, you can install Flask and other dependencies.

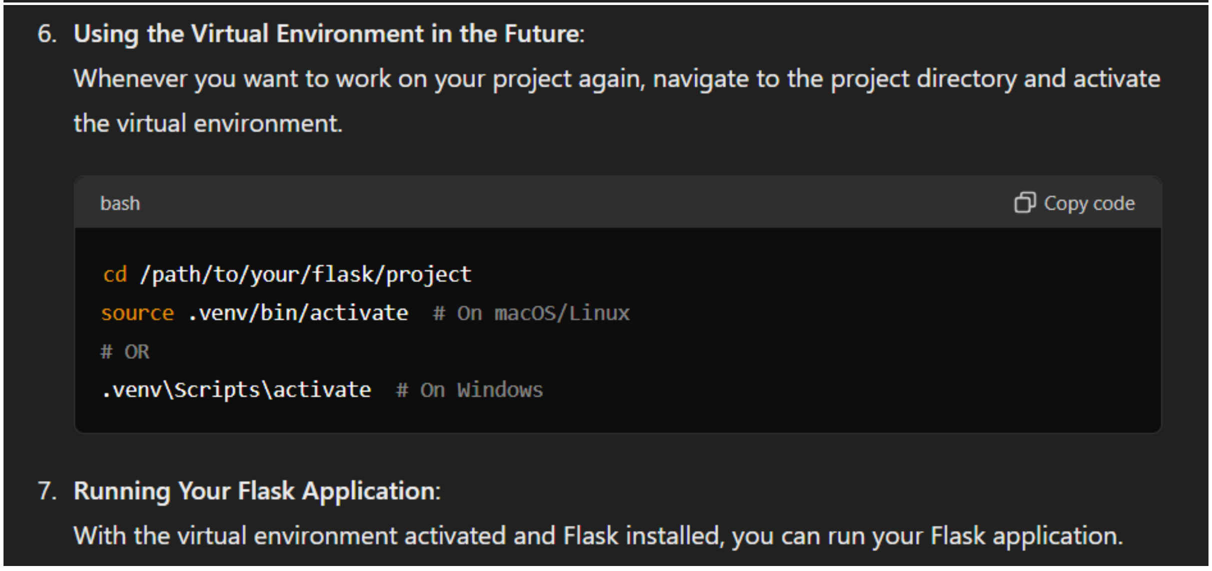
```bash
pip install Flask
```

4. **Freeze the Dependencies:**
   To keep track of your dependencies, you should create a `requirements.txt` file.

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

5. **Deactivate the Virtual Environment:**
   When you're done working, you can deactivate the virtual environment.

```bash
deactivate
```


6. **Using the Virtual Environment in the Future:**
   Whenever you want to work on your project again, navigate to the project directory and activate the virtual environment.

```bash
cd /path/to/your/flask/project
source .venv/bin/activate  # On macOS/Linux
# OR
.venv\Scripts\activate  # On Windows
```

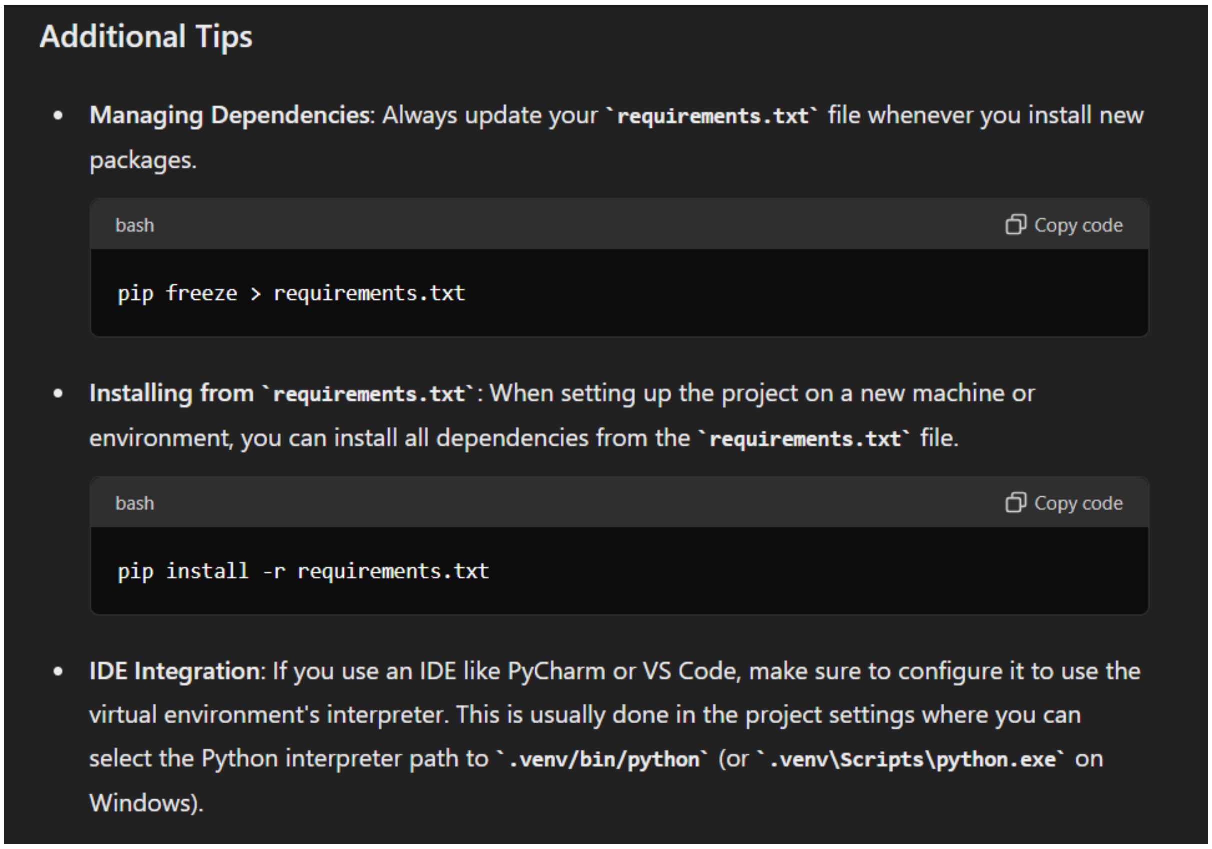
7. **Running Your Flask Application:**
   With the virtual environment activated and Flask installed, you can run your Flask application.

To run the Flask App in the Virtual Environment follow the steps below:

- pip install dotenv
- Create a file called *.env* (Yes it starts with a dot and then the word env for environment)
- In *.env* write:

```
FLASK_APP=index.py
```

Since, in index.py we have imported app from wsgi. More on vercel setup in the Vercel Setup note.



Additional Tips

- **Managing Dependencies**: Always update your `requirements.txt` file whenever you install new packages.

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

- **Installing from `requirements.txt`**: When setting up the project on a new machine or environment, you can install all dependencies from the `requirements.txt` file.

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

- **IDE Integration**: If you use an IDE like PyCharm or VS Code, make sure to configure it to use the virtual environment's interpreter. This is usually done in the project settings where you can select the Python interpreter path to `.venv/bin/python` (or `.venv\Scripts\python.exe` on Windows).