ADDITIONAL READING

Setting Up A Virtual Environment

Visit our website

WELCOME TO THE VIRTUAL ENVIRONMENT WALKTHROUGH GUIDE!

In this guide, we walk through the process of setting up a virtual environment for your Python project using either the built-in `venv` module or the external `virtualenv` package. Virtual environments isolate your project's dependencies, providing a clean and self-contained environment.

We'll explore the generation of a requirements.txt file to document and manage your project's dependencies. Keeping this file up-to-date is crucial for ensuring a reproducible development environment. By following these steps, you are well-equipped to initiate and maintain a Python project with best practices in mind, fostering readability, consistency, and ease of collaboration.

SET UP A VIRTUAL ENVIRONMENT

A virtual environment is a self-contained space in a computer system that isolates software, managing dependencies independently. Widely used in software development, it ensures consistent and reproducible application behaviour across different computing environments.

Using `venv` (built-in in Python 3.3 and newer) or `python -m venv`:

Open a Terminal or Command Prompt:

- Open a terminal or command prompt on your computer or IDE (VS CODE).
- Navigate to Your Project Directory (File):
- Use the `cd` command to navigate to the directory where you want to create your virtual environment.

```
cd your_path/to/your/project
```

Create a Virtual Environment:
Run the following command to create a virtual environment. Replace the second `venv` with the name you want to give your virtual environment.

```
python -m venv venv
```

If you are using Python 3.3 or newer, you can use:

```
python3 -m venv venv
```

Activate the Virtual Environment:
Activate the virtual environment. On Windows, you might use:

```
venv\Scripts\activate.bat
```

On Linux or MacOS, you might use:

```
source venv/bin/activate
```

After activation, your terminal prompt should change to indicate that you are now in the virtual environment. You should see the following:

```
(.venv) C:\Users\your_path\to\your\project>
```

Install Dependencies:
Now, you can use `pip` to install Python packages such as bcrypt within your virtual environment.

```
pip install hashlib
pip instal bcrypt
Pip install cryptography
```

Deactivate the Virtual Environment:
When you're done working in your virtual environment, you can deactivate it.

On Windows, you might use:

```
venv\Scripts\deactivate.bat
```

On Linux or MacOS, you might use:

```
source venv/bin/deactivate
```

Using `virtualenv` (an external package):
If you prefer using `virtualenv`, you can install it globally and then use it to create virtual environments.

Install `virtualenv` using `pip`. This is a once-off step.

```
pip install virtualenv
```

Navigate to your project directory and create a virtual environment.

```
virtualenv venv
```

Activate the virtual environment and install dependencies as described above.

```
source venv/bin/activate  # On Linux/Mac
venv\Scripts\activate.bat     # On Windows
```

When you're done, deactivate the virtual environment.

```
source venv/bin/deactivate  # On Linux/Mac
venv\Scripts\deactivate.bat     # On Windows
```

Remember to activate your virtual environment whenever you work on your project. This ensures that the dependencies you install are specific to your project and don't interfere with other projects or the system-wide Python installation.

GENERATING A REQUIREMENTS FILE

A `requirements.txt` file in a project serves as a crucial document outlining the specific Python packages and their corresponding versions required for the project to run successfully. This file captures the dependencies, allowing for easy replication of the project's environment on different systems. To generate a `requirements.txt` file for your Python project, you can use the `pip freeze` command. This command lists all installed packages and their versions. Here's how you can create a **`requirements.txt`** file:

Activate your virtual environment (if you're using one):

```
source venv/bin/activate  # On Linux/Mac
venv\Scripts\activate     # On Windows
```

Run the following command to generate the requirements.txt file:

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

This command creates a **`requirements.txt`** file containing the names and versions of all installed packages.

Here's an example requirements.txt file:

```
cryptography==42.0.2
bcrypt==2.26.0
```

You can customise the **`requirements.txt`** file based on your project's dependencies. It's good practice to regularly update this file as you add or remove dependencies in your project.

If you want to include only the packages needed for your project (excluding development dependencies), you can use the **`pip freeze --exclude-editable`** option:

```
pip freeze --exclude-editable > requirements.txt
```

This command will exclude packages installed in editable mode (-e option) from the **requirements.txt** file. Editable installations are often used during development with packages installed in "editable" mode, e.g., using the command: `pip install -e .`

The `.` at the end of the command refers to the current directory, indicating that the package in editable mode should be installed from the current project directory. Excluding it ensures that only production dependencies are listed.

In conclusion, this comprehensive guide equips you with the fundamental knowledge to kickstart your Python project development in a structured and efficient manner. By embracing virtual environments and the generation of a **`requirements.txt`** file, you establish a solid foundation for building scalable, maintainable, and organised applications. Incorporating these best practices not only enhances the clarity and readability of your code but also streamlines collaboration and ensures a smooth development experience. As you continue on your coding journey, the skills and techniques outlined here will contribute to the success of your Python projects, fostering a robust and sustainable development workflow. Remember that if you get stuck, you can contact your mentor for help.