# ***Sequência projeto cadastro***

1. Crie um novo diretório para o projeto: Abra o terminal ou prompt de comando e execute:

**mkdir (Nome do projeto)**

**cd (Nome do projeto)**

1. Crie um ambiente virtual (opcional, mas recomendado):

**python -m venv venv**

Ative o ambiente virtual:

* No Windows:

**venv\Scripts\activate**

**.\venv\Scripts\Activate.ps1**

1. Instale as dependências necessárias:

**pip install customtkinter**

**pip install Pillow ou python -m pip install Pillow**

**pip install --upgrade customtkinter Pillow**

1. Criar um arquivo chamado **app.py** no diretório do projeto.
2. Copie e cole o código que forneci anteriormente no arquivo **app.py.**
3. Execute o programa: **python app.py**

# ***Por que usar ambiente virtual?***

Recomendo o uso de um ambiente virtual por várias razões importantes:

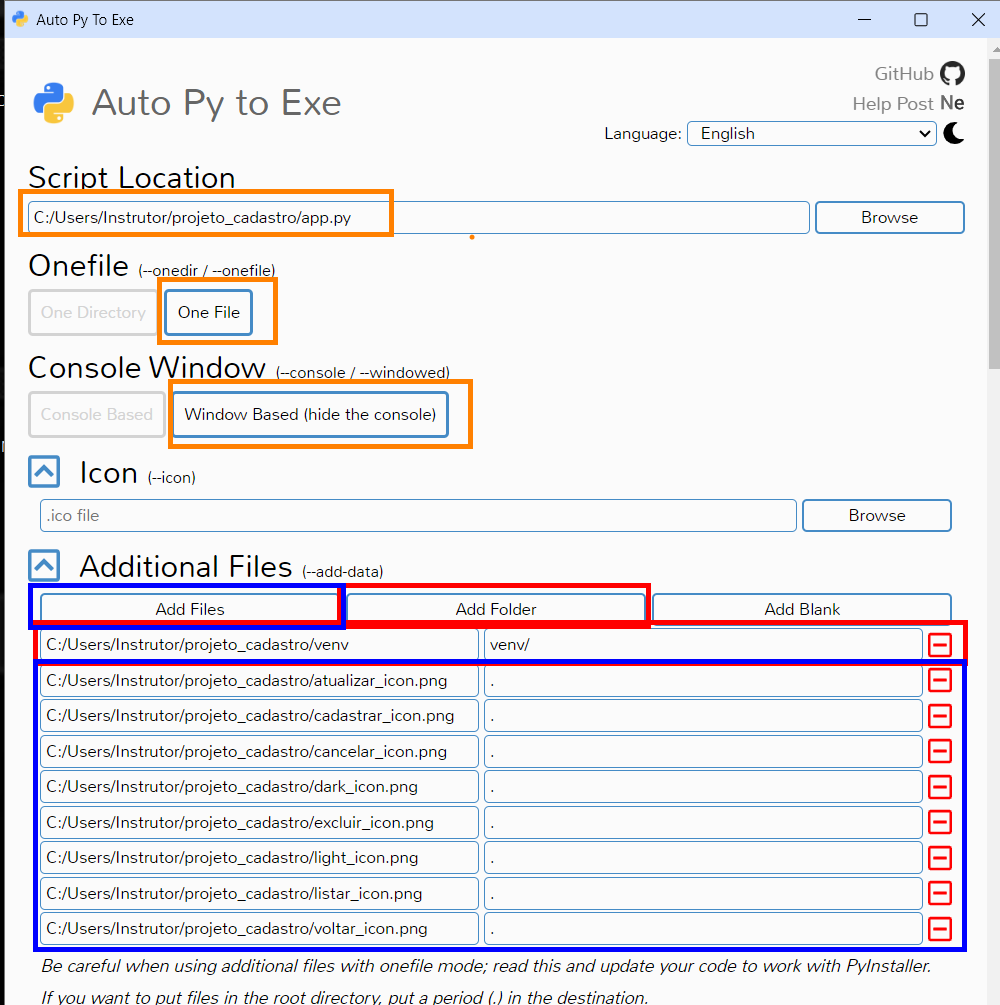
1. Isolamento de dependências: Cada projeto pode ter suas próprias versões de bibliotecas, evitando conflitos entre projetos diferentes.
2. Reprodutibilidade: Facilita a recriação exata do ambiente de desenvolvimento em outras máquinas ou em produção.
3. Limpeza do sistema: Evita a poluição do Python global do sistema com múltiplas bibliotecas.
4. Segurança: Reduz riscos de segurança ao isolar as dependências de cada projeto.
5. Facilidade de gerenciamento: Permite gerenciar facilmente as dependências específicas de cada projeto.
6. Portabilidade: Torna o projeto mais portátil entre diferentes sistemas e desenvolvedores.
7. Testes de diferentes versões: Permite testar o projeto com diferentes versões de Python ou bibliotecas.
8. Facilita a colaboração: Outros desenvolvedores podem replicar facilmente o ambiente do projeto.
9. Melhores práticas: É considerado uma boa prática no desenvolvimento Python profissional.
10. Fácil limpeza: Se algo der errado, você pode simplesmente deletar o ambiente virtual e criar um novo, sem afetar o resto do sistema.

Usar um ambiente virtual é especialmente útil em projetos maiores ou quando você trabalha em múltiplos projetos com diferentes requisitos. Para projetos pequenos ou experimentos rápidos, pode não ser estritamente necessário, mas ainda é uma boa prática a se adotar.

***Gerar exe usando auto-py-to-exe***

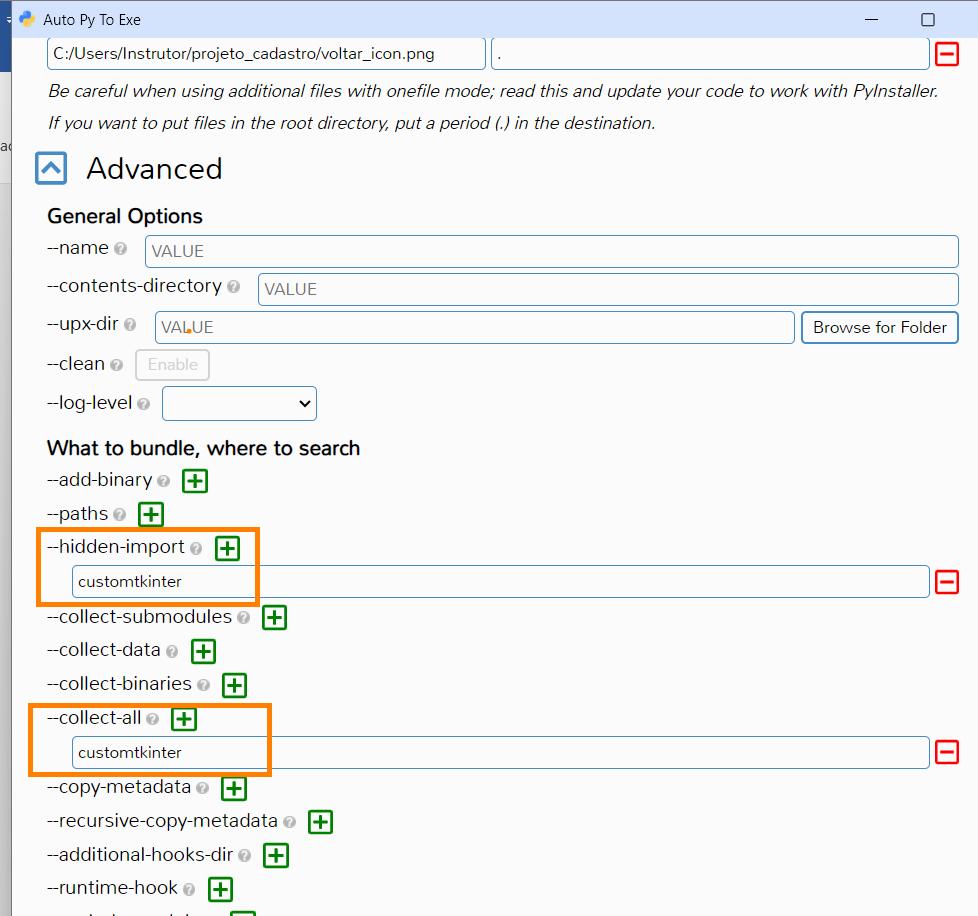
1. **Rodar (Interface):** **auto-py-to-exe**

**Obs:** Se necessário seguir todos os passos de: Gerar executável de programa Python.docx



a) Na seção "Additional Files", Para --hidden-import, adicione apenas: **customtkinter**

b) Para --collect-all, adicione em uma linha separada: **customtkinter**



1. Tente gerar o executável novamente.