📂 Why is `__pycache__` created when we import a file?

When you do:

```python
from hello_chai import chai
```

Python does **two things**:

1. ✅ It **loads** the `chai` function so you can use it.
2. 💾 It **compiles** the `hello_chai.py` file into **bytecode** — a faster, low-level version that Python can run quickly.

🧠 What is bytecode?

- Bytecode is an optimized version of your `.py` file.
- It ends in `.pyc` (Python compiled).
- It's stored inside a folder called `__pycache__`.

Example:

```markdown
__pycache__/
    hello_chai.cpython-311.pyc
```

This means:

- It's a **cached** version of `hello_chai.py`
- Compiled using **CPython** (Python's standard interpreter)
- For version **3.11** (or whatever Python version you're using)

🚀 Why does Python do this?

To **improve performance**.

- Next time you run or import `hello_chai.py`, Python can skip parsing and compiling the `.py` file again.
- It just loads the `.pyc` file from `__pycache__`, which is faster.

🧹 Can you delete `__pycache__`?

Yes, you can safely delete the `__pycache__` folder.
Python will simply recreate it the next time you run or import that module.

✅ Summary

Thing	Explanation
`__pycache__`	Folder where compiled `.pyc` files are stored
Why it's created	To speed up future imports and execution
Can you delete it?	Yes, Python will regenerate it automatically