Flask is a lightweight **web framework for Python** that lets you build web applications quickly with minimal setup. It's often called a "micro-framework" because it doesn't force you to use specific tools or structures—you choose what you need (database, forms, authentication, etc.).

A good way to think of it:
Flask gives you the basic engine to run a website, and you add parts as needed.

It is built on Python and commonly used for:

- Web apps
- APIs (backend services)
- Prototypes and small-to-medium projects

You can explore it here: Flask

Step-by-step basics of Flask

1. Install Flask

First, make sure Python is installed, then install Flask:

```
pip install flask
```

---

2. Create a simple project

Make a folder, for example:

```
flask_app/
```

Inside it, create a file:

```
app.py
```

---

3. Write your first Flask app

Put this inside `app.py`:

```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def home():
    return "Hello, Flask!"

if __name__ == '__main__':
    app.run(debug=True)
```

---

4. Run the app

In terminal, inside your project folder:

```
python app.py
```

Then open in browser:

```
http://127.0.0.1:5000/
```

You should see:

```
Hello, Flask!
```

5. Understand routing (URLs)

Flask uses `@app.route()` to define pages.

Example:

```
@app.route('/about')
def about():
    return "This is the about page"
```

Now visit:

```
http://127.0.0.1:5000/about
```

6. Return HTML instead of text

You can return HTML directly:

```
@app.route('/')
def home():
    return "<h1>Welcome</h1><p>This is Flask</p>"
```

7. Use templates (recommended)

Create a folder inside your project folder flask-app :

```
templates/
```

Inside it, create:

```
index.html
```

Example `index.html`:

```
<h1>Hello from template</h1>
```

Then update `app.py`:

```
from flask import Flask, render_template

app = Flask(__name__)

@app.route('/')
def home():
    return render_template('index.html')
```

8. Pass data to templates

```
@app.route('/')
def home():
    return render_template('index.html', name="Amit")
```

In HTML:

```
<h1>Hello {{ name }}</h1>
```

9. Accept user input (basic idea)

```
from flask import request

@app.route('/submit', methods=['POST'])
def submit():
    data = request.form['username']
    return f"Hello {data}"
```

Example HTML form index.html file:-

```
<form action="/submit" method="POST">
    <input type="text" name="username">
    <button type="submit">Submit</button>
</form>
```

Finally your full updated code of app.py will be like this :-

app.py index.html

```python
from flask import Flask,render_template,request

app = Flask(__name__)


@app.route('/')
def home():
    return render_template('index.html', name="Amit")

@app.route('/about')
def about():
    return "This is the about page"

@app.route('/submit', methods=['POST'])
def submit():
    data = request.form['username']
    return f"Hello {data}"

if __name__ == '__main__':
    app.run(debug=True)
```

And run it :-python app.py

```
D:\flask-app>python app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
```

1. Importing Flask tools

```
from flask import Flask, render_template, request
```

Meaning:

You are importing 3 important things from Flask:

- **Flask** → creates your web application
- **render_template** → used to load HTML files (templates)
- **request** → used to access data sent by user (forms, JSON, etc.)

 In simple terms:

"Bring Flask tools needed to build a website"

---

2. Creating the app

```
app = Flask(__name__)
```

Meaning:

- You are creating your Flask web application

What is `__name__`?

It tells Flask:

"This file is the main file running the app"

 Think of it like:

"Start building my website from this file"

---

3. Home route (homepage)

```
@app.route('/')
def home():
    return render_template('index.html', name="Amit")
```

Breakdown:

➤ **`@app.route('/')`**

- This defines the **homepage URL**
- Example:

```
http://127.0.0.1:5000/
```

➤ **`def home():`**

- This function runs when someone visits `/`

➤ **`render_template('index.html', name="Amit")`**

- Loads an HTML file called `index.html`
- Sends a variable called `name` with value `"Amit"`

 So inside HTML you can use:

```
<h1>Hello {{ name }}</h1>
```

Output in browser:

```
Hello Amit
```

---

4. About route

```
@app.route('/about')
def about():
    return "This is the about page"
```

Meaning:

- Creates a new page:

```
http://127.0.0.1:5000/about
```

- When user visits it, Flask returns plain text:

```
This is the about page
```

 No HTML file used here, just simple text response.

---

5. Submit route (form handling)

```
@app.route('/submit', methods=['POST'])
def submit():
```

Meaning:

- Creates `/submit` URL
- Only allows **POST requests** (used for sending form data)

 Example use:

- HTML form submission

Next line:

```
data = request.form['username']
```

Meaning:

- Gets data sent from a form field named `"username"`

Example HTML form:

```
<form action="/submit" method="POST">
    <input name="username">
</form>
```

If user types:

```
Amit
```

Then:

```
data = "Amit"
```

Final line:

```
return f"Hello {data}"
```

Meaning:

- Sends response back to browser
- Uses f-string to insert variable

Output:

```
Hello Amit
```

---

6. Running the app

```
if __name__ == '__main__':
    app.run(debug=True)
```

Breakdown:

➤ **`if __name__ == '__main__':`**

Means:

"Run this file directly (not imported from another file)"

➤ **`app.run(debug=True)`**

Starts the Flask server.

What `debug=True` does:

- Auto restarts server when you change code
- Shows detailed errors in browser

FULL FLOW OF YOUR APP

1. User opens:

`/`

→ sees `index.html` with "Amit"

---

2. User opens:

```
/about
```

→ sees text:

```
This is the about page
```

---

3. User submits form:

```
/submit (POST)
```

Flask:

- reads username
- returns:

```
Hello Amit
```

SIMPLE SUMMARY

Part	Meaning
Flask(**name**)	Creates app
@app.route('/')	Defines URL
render_template	Loads HTML file
request.form	Gets form data
POST	Sends data
app.run()	Starts server