Django Vs Flask

Flask follows a **Microservices-based Architecture** or **MVC (Model-View-Controller) Pattern** but in a more **flexible and minimalistic** way compared to Django's **MVT (Model-View-Template)** architecture.

✅ Key Differences: Django (MVT) vs Flask (MVC)

Feature	Django (MVT)	Flask (MVC)
Architecture	MVT (Model-View-Template)	MVC (Model-View-Controller)
Structure	Pre-defined with rigid structure	Minimal, flexible structure
Model	ORM with models.py	Manual DB handling, ORM optional
View	Views handle request-response	Views (routes) process requests
Controller	Implicit (managed by Django)	Explicit (handled manually)
Template	HTML with Django Template Engine	Jinja2 as template engine
Flexibility	Less flexible, more opinionated	Highly flexible, lightweight
Scalability	Better suited for large apps	Suitable for small to medium apps
Admin Panel	Auto-generated	No default admin panel

📝 Flask MVC Architecture Explained:

1. **Model (M)**
    - Handles database interactions.
    - Manually defined using SQLAlchemy or other ORM libraries.
    - Example:

```
from flask_sqlalchemy import SQLAlchemy
db = SQLAlchemy(app)

class User(db.Model):
```

```
    id = db.Column(db.Integer, primary_key=True)
    name = db.Column(db.String(50))
```

2.
3. **View (V)**
    - Responsible for rendering HTML using Jinja2 templates.
    - Views define how data is presented.
    - Example:

```
<!-- templates/index.html -->
<h1>Hello, {{ name }}!</h1>
```

4.
5. **Controller (C)**
    - Routes incoming requests and interacts with models & views.
    - Controllers are defined in Flask routes.
    - Example:

```
from flask import Flask, render_template
app = Flask(__name__)

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

6.

⚡ Basic Flask Project Structure:

```
/flask_app
├── /static          # CSS, JS, Images
├── /templates       # HTML templates
```

```
├── /models.py       # Define models
├── /routes.py       # Define routes
├── /app.py          # Main app entry
└── /config.py       # Configurations
```

🎯 When to Use Flask vs Django?

- **Flask:** Ideal for small to medium projects, APIs, and when flexibility is required.
- **Django:** Best for large-scale applications with built-in features and admin interface.

Let me know if you'd like a project demo for Flask or help with setup! 🚀