Step-by-Step Flask Roadmap

1. Basics of Flask

- What is Flask, why use it (vs Django, FastAPI).
- Installing Flask (`pip install flask`).
- First "Hello, Flask" app.
- Flask application structure.

  👉 Practice: Build a simple app with a single route (`/`) returning "Hello World".

---

2. Routing & Views

- `@app.route()` decorator.
- Handling multiple routes (`/about`, `/contact`).
- Using URL parameters (`/user/<name>`).
- HTTP methods (`GET`, `POST`).

  👉 Practice: Create a simple app with a homepage, about page, and dynamic user profile page.

---

3. Templates (Jinja2)

- Template rendering with `render_template`.
- Using variables: `{{ name }}`.
- Control structures: `{% if %}`, `{% for %}`.
- Template inheritance (`base.html`, `child.html`).

  👉 Practice: Build a portfolio website with a base template.

---

4. Forms & User Input

- HTML forms (`<form>`).
- Handling form data in Flask using `request.form`.
- Using `GET` vs `POST`.
- Form validation (basic).

  👉 Practice: Create a feedback/contact form that accepts user input and shows it back.

---

5. Static Files

- CSS, JavaScript, and Images in Flask (`/static/` folder).
- Linking static files inside HTML templates.
  👉 Practice: Add CSS to your portfolio site.

---

6. Flask with Databases

- SQLite integration with `sqlite3` or SQLAlchemy ORM.
- CRUD operations (Create, Read, Update, Delete).
- Models and migrations.
  👉 Practice: Build a **To-Do App** where tasks are stored in a database.

---

7. Flask with REST APIs

- Returning JSON instead of HTML.
- API routes (`/api/data`).
- Flask extensions: `Flask-RESTful` or `Flask-Smorest`.
- Connecting with frontend or Postman.
  👉 Practice: Create an API that returns student data as JSON.

---

8. Flask Extensions

- **Flask-WTF** → advanced forms & validation.
- **Flask-Login** → authentication (login, logout, sessions).
- **Flask-Migrate** → database migrations.
- **Flask-CORS** → enable cross-origin requests (for frontend+backend apps).
  👉 Practice: Add user authentication (register/login) to your To-Do App.

---

9. Deployment

- Running Flask in production with Gunicorn / uWSGI.
- Deploying on **Heroku, PythonAnywhere, or AWS**.
- Using environment variables (`.env`) for secrets.
  👉 Practice: Deploy your portfolio app online.

---

10. Advanced Topics

- Blueprints (modularizing Flask apps).
- Flask Application Factory pattern.
- Middlewares in Flask.
- Flask + Machine Learning model (serve predictions).
  👉 Practice: Deploy a simple ML model (like Titanic survival prediction) as a Flask API.

🎯 Final Project Ideas

- Blog application with authentication & database.
- Student management system (CRUD + authentication).
- REST API for a machine learning project.
- E-commerce mini site (cart + checkout simulation).

⚡ Suggestion for teaching:

- Start with **visual output (HTML pages)** to keep students engaged.
- Then move to **forms + databases**, then **APIs + deployment**.
- End with a **mini-project** combining everything.
