Table of Contents
1. **Prerequisites**
2. **Phase 1: Python Basics**
3. **Phase 2: Web Fundamentals**
4. **Phase 3: Django Core Concepts**
5. **Phase 4: Intermediate Django**
6. **Phase 5: Advanced Topics**
7. **Phase 6: Deployment & Best Practices**
8. **Projects to Build**
9. **Resources & Communities**

---

1. Prerequisites
- Basic understanding of programming concepts (variables, loops, conditionals).
- Familiarity with using a **command-line interface (CLI)**.
- A code editor (e.g., VS Code, PyCharm).

---

2. Phase 1: Python Basics
(1–2 Weeks)
- **Topics to Learn**:
  - Variables, data types, operators.
  - Control flow (`if-else`, loops).
  - Functions and modules.
  - Basic OOP (classes, objects, inheritance).
  - Working with files and exceptions.
- **Resources**:
  - [Python Crash Course (Book)](https://nostarch.com/pythoncrashcourse2e)
  - [Automate the Boring Stuff (Free Online)](https://automatetheboringstuff.com/)

---

3. Phase 2: Web Fundamentals
(1 Week)
- **Topics to Learn**:
  - How the web works (HTTP, client-server model).
  - Basics of HTML/CSS (no need for mastery).
  - Frontend vs. backend overview.
  - RESTful APIs (conceptual understanding).
- **Resources**:
  - MDN Web Docs: [Introduction to HTTP](https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview)

- FreeCodeCamp: [HTML/CSS Basics](https://www.freecodecamp.org/learn/)

---

4. Phase 3: Django Core Concepts
(2–3 Weeks)
- **Topics to Learn**:
  - **Django Architecture**: MVT (Model-View-Template) pattern.
  - Setting up Django: Installation with `pip`, starting a project.
  - Building a simple app:
    - Models (database tables with `models.py`).
    - Views (logic handling with `views.py`).
    - Templates (HTML rendering with Django syntax).
    - URLs (routing with `urls.py`).
  - Django Admin Panel.
  - Basic forms and user input.
- **Resources**:
  - [Django Official Tutorial](https://docs.djangoproject.com/en/4.2/intro/tutorial01/)
  - Video: [Django for Everybody (Coursera)](https://www.coursera.org/specializations/django)
- **Project**: Build a **Todo App** with CRUD operations.

---

5. Phase 4: Intermediate Django
(2 Weeks)
- **Topics to Learn**:
  - User authentication (login, signup, sessions).
  - Static files (CSS, JS, images).
  - Working with databases (SQLite/PostgreSQL).
  - Django Forms (validation, custom forms).
  - Class-Based Views (CBVs).
  - Middleware and context processors.
- **Project**: Build a **Blog** with user accounts and comments.

---

6. Phase 5: Advanced Topics
(2–3 Weeks)
- **Topics to Learn**:
  - Django REST Framework (building APIs).
  - Testing (unit tests, integration tests).
  - Security best practices (XSS, CSRF, SQL injection).
  - Caching and performance optimization.
  - Custom template tags/filters.
- **Project**: Build an **E-commerce API** with DRF.

---

7. Phase 6: Deployment & Best Practices
(1 Week)
- **Topics to Learn**:
  - Deploying to platforms:
    - Heroku
    - PythonAnywhere
    - AWS/EC2
  - Setting up `gunicorn` and `nginx`.
  - Environment variables with `.env`.
  - Version control with Git/GitHub.
- **Resource**: [Django Deployment Checklist](https://docs.djangoproject.com/en/4.2/howto/deployment/checklist/)

---

8. Projects to Build
- **Beginner**: Todo App, Blog, Weather App.
- **Intermediate**: E-commerce Site, Social Media Clone.
- **Advanced**: Job Board with APIs, Real-time Chat (WebSockets).

---

9. Resources & Communities
- **Books**:
  - *Django for Beginners* by William S. Vincent
  - *Two Scoops of Django* (Best Practices)
- **Courses**:
  - [Django Official Docs](https://docs.djangoproject.com/)
  - [MDN Django Tutorial](https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django)
- **Communities**:
  - Django Forum: forum.djangoproject.com
  - Reddit: r/django
- **Tools**:
  - Django Extensions, Django Debug Toolbar.