PYTHON + DJANGO INTERVIEW ROADMAP

1️⃣ Core Python (Master the Foundation)

Focus on concepts that are frequently tested in interviews:

✅ Basics

- Variables, data types
- Typecasting
- Input/output
- Conditional statements, loops

✅ Data Structures

- Lists, tuples, sets, dictionaries (operations, comprehension)
- Stack, queue, linked list (basic understanding)

✅ Functions

- Arguments (`*args`, `**kwargs`)
- Lambda, map, filter, reduce
- Recursion
- Decorators

✅ Object-Oriented Programming (Very Important)

- Class, object, constructor
- Inheritance, polymorphism
- Encapsulation, abstraction
- `__str__`, `__repr__`, `__init__`

✅ Exception Handling

- Try, except, finally
- Custom exceptions

✅ File Handling

- Read/write files
- Context managers (`with`)

✅ Modules and Packages

- Creating and importing custom modules
- `__init__.py`

✅ Pythonic Features

- List/dict/set comprehension
- `enumerate()`, `zip()`, `any()`, `all()`
- Generators, iterators

✅ Practice

- **Leetcode, HackerRank, or Coding Ninjas**: easy to medium problems in Python

---

2️⃣ Advanced Python (For Serious Interviews)

- Multithreading vs multiprocessing
- GIL (Global Interpreter Lock)
- Memory management
- `*args` vs `**kwargs` in practice
- `@staticmethod` vs `@classmethod`
- Shallow vs deep copy

3️⃣ Django Framework (The Core Skill)

✅ Basics

- MTV Architecture (Model-Template-View)
- Django project vs app
- `manage.py`, `settings.py`, `urls.py`
- Routing (`path()`, `re_path()`, `include()`)

✅ Models & ORM

- `models.Model`
- Fields, validators
- Querysets: `.filter()`, `.get()`, `.exclude()`, `.aggregate()`
- Relationships: OneToOne, ForeignKey, ManyToMany

✅ Views

- Function-based views (FBV)
- Class-based views (CBV)
- Generic views (ListView, DetailView, CreateView)

✅ Templates

- Template inheritance
- `{{ }}` and `{% %}` syntax
- Template filters and tags

✅ Forms

- `forms.Form` and `forms.ModelForm`
- Validation (`clean()`, `clean_<field>()`)
- Error handling

✅ Admin

- Customizing Django admin
- Registering models
- Inline models, list_display, search_fields

✅ Authentication & Authorization

- `User` model
- Login, logout
- Permissions, groups, decorators

✅ Middleware & Signals

- Custom middleware
- `pre_save`, `post_save`, `post_delete` signals

✅ REST API (with Django REST Framework - DRF)

- Serializers
- Viewsets & routers
- Authentication: JWT, TokenAuth
- CRUD APIs

4️⃣ Project & Deployment

✅ Build Projects (portfolio is key)

- Blog
- E-commerce
- To-do app with authentication
- API backend for mobile app

✅ Deployment (Basic Knowledge)

- Using **Heroku**, **Render**, or **PythonAnywhere**
- Setting `ALLOWED_HOSTS`, static files
- `.env` and environment variables

5️⃣ System Design Basics (Optional but Advantageous)

- How Django handles a request
- Load balancing, database optimization
- Caching (Memcached, Redis)
- Scaling Django app (horizontal/vertical)

🎯 Bonus: Common Interview Questions

✅ Python

- Difference between list and tuple
- What are decorators?
- Mutable vs immutable
- Explain inheritance in Python
- What is `self`?

✅ Django

- How does Django handle a request internally?
- Difference between `null=True` and `blank=True`
- What is the difference between `@login_required` and permission classes in DRF?
- How do you prevent SQL injection in Django?
- What are signals? Use case?

✅ Practical Tasks

- Write a Django model and API for a blog post
- Design a login/logout system
- Create an endpoint that returns JSON response for a query