Django Questions

can you give me Django also

Absolutely! Since you’re diving into Django, I can provide you with structured content similar to how we’ve done for Python and algorithms. Here's what I can offer:

✅ Django Study Plan

Let me know what you'd like to focus on:

1. **100 Django Interview Questions (Topic-wise)**
2. **Django Daily Challenges**
3. **Django Project Ideas (Beginner to Advanced)**
4. **Core Concepts with Code Examples**
5. **Django + REST Framework Study**
6. **Django ORM Deep Dive**
7. **Django with Frontend (HTML/CSS/JS Integration)**
8. **Django Deployment (Heroku, Docker, etc.)**

For starters, would you like me to begin with **100 topic-wise Django interview questions** (like we’re doing with Python), or something else from the list above?

Also, would you like **a daily Django challenge** added to your routine mix of interview prep questions?

100 topic-wise Django interview questions in each topic 50 question give me proper question

Perfect! Here's how I'll break it down:

✅ Django Interview Questions (Topic-Wise)

Each topic will have **50 well-structured interview questions**, covering both conceptual and practical aspects.

Topic 1: Django Basics (50 Questions)

1. What is Django and why is it used?
2. How does Django follow the MVT architecture?

3. What are the major features of Django?
4. Explain the difference between Django and Flask.
5. What is a Django project and a Django app?
6. How do you create a Django project?
7. How do you create a Django app?
8. What does the `manage.py` file do?
9. How does Django's development server work?
10. What is the role of `settings.py`?
11. Explain the structure of a default Django project.
12. How does URL routing work in Django?
13. What is the use of `urls.py`?
14. How do you map a view to a URL in Django?
15. What are views in Django?
16. Difference between function-based views and class-based views.
17. How do you return an HTTP response from a view?
18. How does Django handle requests internally?
19. How can you pass data from views to templates?
20. What is the role of templates in Django?
21. How does Django render HTML from templates?
22. What is the use of `render()` function?
23. What is context in Django templates?
24. What are template tags and filters?
25. What is the difference between `include` and `extends` in templates?
26. How can you serve static files in Django?
27. How can you serve media files in development?
28. What is the `STATIC_URL` setting used for?
29. How do you configure static and media file settings?
30. What are models in Django?
31. How do you define a model in Django?
32. What are model fields? Name a few common ones.
33. How does Django ORM work?
34. What is the role of `makemigrations` and `migrate`?
35. How do you run raw SQL in Django?
36. How do you access data from a model?
37. What is the admin interface in Django?

38. How do you enable a model in Django admin?
39. What is the purpose of `` `@admin.register()` ``?
40. How do you customize Django admin?
41. What is the difference between `` `null=True` `` and `` `blank=True` `` in models?
42. What are `` `Meta` `` options in a Django model?
43. How do you define model relationships (OneToOne, ForeignKey, ManyToMany)?
44. What are querysets in Django?
45. How do you filter/query data in Django ORM?
46. What is the difference between `` `get()` `` and `` `filter()` ``?
47. What is `` `select_related()` `` and `` `prefetch_related()` ``?
48. How do you handle 404 errors in Django?
49. What is the use of `` `get_object_or_404()` ``?
50. How do you run the development server and access your Django app?

Let me know if you'd like me to continue with **Topic 2: Django Models and ORM (50 Questions)** next, or if you'd like a different topic!

Topic 2: Django Models and ORM (50 Questions)

Awesome! Here's the next set:

✅ Topic 2: Django Models and ORM (50 Interview Questions)

1. What is a model in Django?
2. How do you create a model in Django?
3. What is Django ORM?
4. How does Django map models to database tables?
5. What are model fields in Django?
6. What are the different field types in Django models?
7. What is the default primary key in Django models?
8. How do you override the default primary key?
9. How do you define a field with a default value?
10. How do you make a field optional or required?
11. What is the difference between `` `null=True` `` and `` `blank=True` ``?
12. What is a model method?
13. What is a model manager in Django?

14. How do you create a custom model manager?
15. What is the difference between `` `objects.all()` `` and `` `objects.filter()` ``?
16. What are querysets in Django?
17. Are querysets lazy or evaluated immediately? Explain.
18. How do you filter data using Django ORM?
19. How do you use `` `exclude()` `` in a queryset?
20. What is `` `get()` `` used for in Django models?
21. What’s the difference between `` `get()` `` and `` `filter()` ``?
22. What is the `` `values()` `` method in Django ORM?
23. How is `` `values_list()` `` different from `` `values()` ``?
24. What does `` `annotate()` `` do in Django?
25. What is `` `aggregate()` `` used for in Django ORM?
26. What are F expressions in Django?
27. What are Q objects in Django?
28. How do you create relationships between models?
29. Explain `` `ForeignKey` ``, `` `OneToOneField` ``, and `` `ManyToManyField` ``.
30. How do you perform a join query in Django?
31. What is `` `select_related()` `` and when should you use it?
32. What is `` `prefetch_related()` `` and how does it differ from `` `select_related()` ``?
33. How do you save data to a model?
34. What is the use of `` `save()` `` method?
35. How do you delete a model instance?
36. How do you update a model instance?
37. What is `` `bulk_create()` `` and when is it useful?
38. How do you run raw SQL queries in Django?
39. What does `` `db_table` `` inside `` `Meta` `` class do?
40. How can you order query results?
41. What is the `` `unique=True` `` field option used for?
42. How can you create a model form from a model?
43. How can you auto-set the current date/time on create/update?
44. What does `` `auto_now` `` and `` `auto_now_add` `` do?
45. What is `` `choices` `` used for in Django fields?
46. How do you create model inheritance in Django?
47. What are proxy models in Django?
48. What is the `` `verbose_name` `` and `` `verbose_name_plural` `` in Meta class?

49. How do you check if an object exists in the database?
50. How do you use `get_or_create()` and `update_or_create()`?

Let me know if you'd like to continue with:

👉 **Topic 3: Django Views and URL Routing (50 Questions)**
Or suggest another area you'd like to explore next!

Topic 3: Django Views and URL Routing (50 Questions)

Perfect! Here's the next set:

✅ Topic 3: Django Views and URL Routing (50 Interview Questions)

A. Views (Function-Based and Class-Based)

1. What is a view in Django?
2. How do views work in Django’s MVT architecture?
3. What is the difference between function-based views (FBVs) and class-based views (CBVs)?
4. How do you define a function-based view?
5. How do you return an HTML response from a view?
6. What is the `HttpResponse` class used for?
7. How do you pass data from views to templates?
8. What is the use of the `render()` function?
9. What is `redirect()` used for in views?
10. How do you handle POST and GET in a view?
11. How do you access form data in a view?
12. What is the purpose of `request.method`?
13. What is `request.GET` and `request.POST`?
14. How do you raise a 404 error in a view?
15. What does `get_object_or_404()` do?
16. What are generic class-based views (CBVs)?
17. How do you use `ListView` in Django?
18. How do you use `DetailView` in Django?
19. What is `CreateView` and how do you use it?
20. How do `UpdateView` and `DeleteView` work?
21. What is the role of `model` and `template_name` in CBVs?

22. How do you add custom logic in a CBV?
23. What is `get_context_data()` used for?
24. How do you override `form_valid()` in CBVs?
25. How do mixins work in CBVs?

B. URL Routing and Configuration

26. What is URL routing in Django?
27. How does Django process URLs internally?
28. What is the purpose of `urls.py`?
29. How do you map a view to a URL?
30. What is `path()` function used for in URLconf?
31. How do you capture URL parameters?
32. What are path converters in Django (`int:`, `str:`, etc.)?
33. What is `re_path()` and when is it used?
34. How do you include app-specific URLs in the main project?
35. What is the purpose of `include()` in URLconf?
36. How do you assign names to URLs?
37. Why is `name=` important in URL mapping?
38. How do you reverse a URL by its name?
39. What is `reverse_lazy()` used for?
40. How do you access URL parameters in views?
41. What happens if a URL is not matched in `urls.py`?
42. How do you create a custom 404 page?
43. How do you serve static files via URLs?
44. How do you redirect from one URL to another?
45. What is `HttpResponseRedirect`?
46. How can you restrict access to certain URLs?
47. How do you implement login-required URLs?
48. What is a `slug` and how is it used in URLs?
49. How do you use regex in URL patterns?
50. How do you organize URLs in a large Django project?

Would you like to continue with:
👉 **Topic 4: Django Forms and Validation (50 Questions)**
Or a different topic of your choice?

Topic 4: Django Forms and Validation (50 Questions)

Great choice! Here's the next full set:

✅ Topic 4: Django Forms and Validation (50 Interview Questions)

A. Django Forms Basics

1. What is a form in Django?
2. What is the difference between `forms.Form` and `forms.ModelForm`?
3. How do you create a basic Django form?
4. How do you render a form in a template?
5. What is the `{{ form.as_p }}`, `{{ form.as_table }}`, and `{{ form.as_ul }}` used for?
6. How do you manually create form fields in a Django form?
7. What are some common field types used in Django forms?
8. How do you handle form submissions in a view?
9. What does `request.POST` contain?
10. What is the purpose of `form.is_valid()`?
11. How do you access cleaned data from a form?
12. What is `form.cleaned_data` and how is it used?
13. What happens when `form.is_valid()` returns `False`?
14. How do you display form validation errors in a template?
15. How do you prepopulate a form with initial values?

B. Django ModelForms

16. What is a `ModelForm`?
17. How do you create a `ModelForm` from a model?
18. What is the purpose of the `Meta` class in `ModelForm`?
19. How do you include or exclude fields in a `ModelForm`?
20. How do you customize field labels in a `ModelForm`?
21. How do you add widgets to form fields?
22. How do you override form field behavior in a `ModelForm`?
23. What happens when you save a `ModelForm`?
24. How do you save a form without committing to the database?

25. How do you update an existing model instance using a form?

C. Form Validation

26. How does Django perform form validation?
27. What are `clean()` and `clean_<field>()` methods used for?
28. When is the `clean()` method called?
29. How do you perform custom validation for a single field?
30. How do you perform validation across multiple fields?
31. How do you raise a validation error in Django?
32. What is `ValidationError` and how is it used?
33. How do you display custom validation messages?
34. How do you handle non-field errors in Django forms?
35. What is `form.non_field_errors()`?

D. Form Features & Customization

36. How do you add custom widgets to a form field?
37. How do you disable a field in a Django form?
38. How do you make a field read-only in a Django form?
39. How do you style forms with CSS or Bootstrap?
40. What are formsets in Django?
41. How do you create a formset in Django?
42. What is the difference between `formset_factory` and `modelformset_factory`?
43. How do you validate a formset?
44. How do you save data from a formset?
45. How do you use hidden fields in Django forms?

E. Advanced Concepts

46. How do you dynamically add fields to a Django form?
47. How can you create reusable form components?
48. How do you handle file uploads with Django forms?
49. What is `enctype="multipart/form-data"` and why is it needed?
50. How do you protect forms from CSRF attacks in Django?

Would you like to move on to:
👉 **Topic 5: Django Templates and Static Files (50 Questions)**

Or any specific area you're more excited about next?

Topic 5: Django Templates and Static Files (50 Questions)

Awesome! Here's your next complete set:

✅ Topic 5: Django Templates and Static Files (50 Interview Questions)

A. Django Template System (Basics)

1. What is a template in Django?
2. What is the purpose of the Django template engine?
3. How do you create and use templates in a Django project?
4. Where should template files be stored?
5. How do you configure template directories in `settings.py`?
6. How do you render a template from a view?
7. What is the purpose of the `render()` function?
8. How do you pass data from a view to a template?
9. What are template context variables?
10. How do you use variables inside templates?

B. Template Tags and Filters

11. What are template tags in Django?
12. How do you use `{% if %}`, `{% else %}`, and `{% elif %}` in templates?
13. How do you use `{% for %}` loops in templates?
14. What does `{% include %}` do in Django templates?
15. What does `{% extends %}` do?
16. What is the purpose of `{% block %}` and `{% endblock %}`?
17. What is a base template and how is it used?
18. How do you display dynamic data inside templates?
19. What are template filters in Django?
20. Name some built-in filters and their usage (e.g., `date`, `length`, `lower`, `upper`, `default`).

C. Template Inheritance and Structure

21. How does template inheritance work in Django?

22. What is the benefit of using base templates?
23. How do you define multiple blocks in a base template?
24. What happens if a child template doesn't override a block?
25. Can you include one template into another? How?

D. Custom Template Tags and Filters

26. How do you create a custom template tag?
27. How do you create a custom filter in Django?
28. What is the purpose of `` `@register.filter` ``?
29. What file should custom tags and filters be placed in?
30. How do you load a custom tag/filter in a template?

E. Template Best Practices and Security

31. How do you prevent XSS (Cross-site scripting) in templates?
32. What is auto-escaping in Django templates?
33. How can you disable auto-escaping?
34. What is the purpose of the `` `safe` `` filter?
35. How do you debug template variables?

F. Static Files

36. What are static files in Django?
37. How do you configure static files in `` `settings.py` ``?
38. What is the role of `` `STATIC_URL` `` and `` `STATICFILES_DIRS` ``?
39. Where should you store CSS, JS, and image files?
40. How do you use the `` `{% static %}` `` template tag?
41. How do you load static files in a template?
42. How do you serve static files during development?
43. How are static files served in production?
44. What is `` `collectstatic` `` and how is it used?
45. What is the difference between `` `STATICFILES_DIRS` `` and `` `STATIC_ROOT` ``?

G. Media Files

46. What are media files in Django?
47. How do you handle file uploads in Django?

48. How do you configure `MEDIA_URL` and `MEDIA_ROOT`?
49. How do you serve media files during development?
50. What’s the best practice for handling media files in production?

Next up:
👉 **Topic 6: Django Admin and Customization (50 Questions)**
Let me know if you want to continue or switch focus!

Topic 6: Django Admin and Customization (50 Questions)

Here you go!

✅ Topic 6: Django Admin and Customization (50 Interview Questions)

A. Django Admin Basics

1. What is the Django admin site?
2. How do you access the Django admin panel?
3. How do you create a superuser?
4. How do you register a model with the admin site?
5. What is `admin.site.register()` used for?
6. What does the `@admin.register()` decorator do?
7. How do you enable multiple models in the admin panel?
8. How do you customize the display of a model in the admin?
9. What is the `list_display` attribute?
10. How do you show foreign key fields in `list_display`?

B. Filtering, Searching, and Ordering

11. What is the `list_filter` attribute used for?
12. How do you implement a search bar in the Django admin?
13. What is `search_fields` in the admin?
14. How do you make the list view sortable?
15. What is `ordering` in Django admin?

C. Admin Field Customization

16. What is `` `readonly_fields` `` used for?
17. How do you customize the form layout in admin?
18. What is the `` `fieldsets` `` attribute in Django admin?
19. How do you customize form field widgets in admin?
20. What is `` `exclude` `` in admin, and how is it used?

D. Inlines and Related Models

21. What are inlines in Django admin?
22. What is `` `TabularInline` `` vs `` `StackedInline` ``?
23. How do you display related models inline?
24. How do you add or remove inline formsets?
25. How do you save related models using inlines?

E. Custom Admin Classes and Methods

26. How do you define a custom admin class?
27. What is `` `get_queryset()` `` in admin?
28. How do you create custom columns in `` `list_display` ``?
29. How do you make a custom column sortable?
30. How do you use `` `@admin.display()` ``?

F. Permissions and Access Control

31. How do you restrict admin access by user permissions?
32. What are the default permissions Django creates?
33. How do you add or remove permissions for a model?
34. How do you customize access to admin actions?
35. What is `` `has_add_permission()` `` used for?

G. Admin Actions

36. What are admin actions in Django?
37. How do you define a custom admin action?
38. How do you register a custom action?
39. How do you add confirmation or prompts to an action?
40. How do you restrict actions based on user roles?

H. Appearance and Branding

41. How do you customize the admin site title?
42. How do you change the Django admin site header?
43. How do you add custom CSS or JS to the admin panel?
44. What is `Media` class in admin customization?
45. How do you override admin templates?

I. Advanced Admin Features

46. How do you handle file/image uploads in admin?
47. How do you use autocomplete fields in admin?
48. What are `formfield_overrides` used for?
49. How do you integrate a rich text editor (like CKEditor) in admin?
50. How do you log admin changes or track history?

Next up: 👉 **Topic 7: Django Authentication and Authorization (50 Questions)**
Ready to dive into that?

Topic 7: Django Authentication and Authorization (50 Questions)
how many topic are remaining

You're making great progress!

✅ Topic 7: Django Authentication and Authorization (50 Interview Questions)

A. Authentication Basics

1. What is the difference between authentication and authorization?
2. How does Django handle user authentication?
3. What is `django.contrib.auth`?
4. How do you create a user in Django?
5. What is the default `User` model in Django?
6. How do you authenticate a user in a view?
7. What does `authenticate()` do?
8. How do you log a user in?
9. What does `login()` do in Django?
10. How do you log a user out?

B. Built-in Views and URLs

11. What are the built-in authentication views in Django?
12. What is the purpose of `` `LoginView` ``?
13. How do you use `` `LogoutView` ``?
14. What is `` `LOGIN_URL` `` and `` `LOGIN_REDIRECT_URL` ``?
15. How do you use `` `@login_required` `` decorator?
16. How do you protect class-based views with login?
17. How do you redirect users after login/logout?
18. What is `` `next` `` parameter in login redirection?
19. How do you create a custom login view?
20. How do you create a logout confirmation page?

C. Password Management

21. How are passwords stored in Django?
22. What hashing algorithm does Django use?
23. How do you change a user's password?
24. How do you implement password reset functionality?
25. What views and URLs are required for password reset?
26. How do you customize the password reset email?
27. How do you enforce password strength or complexity?
28. What is `` `PasswordChangeView` `` and how is it used?
29. How do you add password confirmation in a form?
30. How can users change their password after login?

D. Permissions and Groups

31. What are permissions in Django?
32. What are the default permissions Django provides?
33. How do you create custom permissions?
34. How do you check if a user has a specific permission?
35. What are groups and how are they used?
36. How do you assign users to groups?
37. How do you assign permissions to a group?
38. How do you check if a user is in a group?

39. How do you restrict access using permissions in views?
40. How do you manage permissions in the admin panel?

E. Custom User Models

41. How do you create a custom user model?
42. What is `` `AbstractUser` `` vs `` `AbstractBaseUser` ``?
43. When should you use a custom user model?
44. How do you set a custom user model in Django?
45. What changes are required in forms when using a custom user model?

F. Advanced Access Control

46. What is the `` `user_passes_test` `` decorator?
47. How do you restrict views based on staff or superuser status?
48. How do you show/hide content in templates based on user roles?
49. How do you implement role-based access control in Django?
50. How do you handle unauthorized access (e.g., 403 errors)?

🧮 You've Completed 7 Topics So Far!

✅ Topics Covered:

1. Django Basics & Project Structure
2. Django Models and ORM
3. Django Views and URL Routing
4. Django Forms and Validation
5. Django Templates and Static Files
6. Django Admin and Customization
7. Django Authentication and Authorization

📌 Topics Remaining (out of 12 total):

8. Django Middleware
9. Django REST Framework (DRF)
10. Django Signals
11. Django Testing and Debugging
12. Django Deployment and Performance Optimization

Would you like to move on to **Topic 8: Django Middleware (50 Questions)**?

Topic 8: Django Middleware (50 Questions)

Absolutely! Here's your full set for the next section:

✅ Topic 8: Django Middleware (50 Interview Questions)

A. Middleware Basics

1. What is middleware in Django?
2. How does middleware work in Django's request/response cycle?
3. Where do you define middleware in a Django project?
4. What is the order of middleware execution?
5. What is the role of `` `MIDDLEWARE` `` setting in `` `settings.py` ``?
6. What are some default middleware components Django provides?
7. How does middleware differ from views and templates?
8. What is the difference between process request and response in middleware?
9. Can middleware modify the request before it reaches the view?
10. Can middleware modify the response before it returns to the client?

B. Creating Custom Middleware

11. How do you create custom middleware in Django?
12. What are the required methods in a custom middleware class?
13. What is the purpose of `` `__init__()` `` in middleware?
14. What does `` `__call__()` `` do in custom middleware?
15. What is `` `process_view()` `` used for?
16. What is `` `process_exception()` `` and when is it called?
17. How can middleware short-circuit the request-response cycle?
18. How do you log requests in a custom middleware?
19. How do you write middleware to restrict access based on IP address?
20. How do you measure the time taken by a request using middleware?

C. Middleware Use Cases

21. How can middleware be used for authentication?

22. How can you use middleware for rate limiting?
23. How can middleware be used for localization?
24. How can middleware handle logging and auditing?
25. How can middleware be used for header injection or modification?
26. How would you use middleware for API versioning?
27. Can middleware be used to add custom request attributes?
28. Can middleware encrypt or decrypt request/response bodies?
29. How can middleware be used to validate API tokens?
30. Can middleware block certain user agents or bots?

D. Middleware in Depth

31. What is the difference between `process_request()` and `__call__()`?
32. How do you control the order of middleware execution?
33. Can middleware interact with the session or user object?
34. How does Django handle exceptions raised in middleware?
35. What happens if middleware returns `None`?
36. Can middleware return a custom `HttpResponse`?
37. What is the performance impact of using middleware?
38. How do you debug middleware issues?
39. Can middleware be conditionally applied (per view)?
40. How do you test middleware logic?

E. Middleware Configuration and Best Practices

41. How do you enable/disable middleware components?
42. What happens if a middleware is incorrectly ordered?
43. How do you avoid middleware conflicts in large projects?
44. What’s the best way to structure custom middleware in a reusable app?
45. Should middleware be used for business logic?
46. What’s the difference between middleware and decorators?
47. How does Django’s `CommonMiddleware` work?
48. What does `SecurityMiddleware` do in Django?
49. What is the role of `CsrfViewMiddleware`?
50. When should you use a decorator vs. middleware?

Want to jump into the next one?

👉 **Topic 9: Django REST Framework (DRF) – 50 Questions**
Let me know if you're ready to hit the APIs!

Topic 9: Django REST Framework (DRF) – 50 Questions

You're on a roll! Here’s a complete list of **Topic 9: Django REST Framework (DRF) – 50 Interview Questions**:

✅ Topic 9: Django REST Framework (DRF) – 50 Interview Questions

---

A. DRF Basics

1. What is Django REST Framework (DRF)?
2. Why use DRF instead of Django alone for APIs?
3. How do you install DRF in a Django project?
4. What changes are needed in `` `INSTALLED_APPS` `` for DRF?
5. How do you create a basic API view in DRF?
6. What is `` `APIView` `` in DRF?
7. What is the difference between Django views and DRF views?
8. How do you return JSON data using DRF?
9. What is a `` `Response` `` object in DRF?
10. What is the role of `` `@api_view` `` decorator?

B. Serializers

11. What is a serializer in DRF?
12. What is the difference between serializers and Django forms?
13. How do you create a basic serializer?
14. What fields can a serializer include?
15. What is the purpose of `` `serializer.is_valid()` ``?
16. How do you access validated data in a serializer?
17. What is `` `create()` `` and `` `update()` `` in serializers?
18. How do you handle nested serialization?
19. What is `` `ModelSerializer` `` in DRF?
20. How do you exclude or include specific fields in a `` `ModelSerializer` ``?

C. Views and ViewSets

21. What is the difference between `APIView` and `GenericAPIView`?
22. What are mixins in DRF?
23. What are `ListModelMixin` and `CreateModelMixin`?
24. What is a `ViewSet` in DRF?
25. What are the types of ViewSets in DRF?
26. How do `ModelViewSet` and `ReadOnlyModelViewSet` differ?
27. How do you register a ViewSet with a router?
28. What is the purpose of DRF routers?
29. How do you define custom actions in a ViewSet?
30. What is the `@action` decorator used for?

D. Routers and URL Routing

31. How do DRF routers simplify URL routing?
32. What is the difference between `SimpleRouter` and `DefaultRouter`?
33. How do you include a router in your URLConf?
34. Can you manually define URLs for a ViewSet?
35. How do you handle nested routes in DRF?

E. Permissions, Authentication, and Throttling

36. What are permissions in DRF?
37. What are some built-in permissions in DRF?
38. How do you write custom permission classes?
39. What are authentication classes in DRF?
40. How do you enable token authentication?
41. What is session authentication in DRF?
42. How does DRF handle throttling?
43. What are throttle classes and why are they used?
44. How do you create custom throttling policies?
45. How can you apply different permissions to different actions?

F. Advanced DRF Concepts

46. What is pagination in DRF and how is it configured?
47. What is the difference between limit-offset and page-number pagination?

48. How do you validate serializer fields with custom logic?
49. How do you upload files using DRF?
50. How can you test DRF endpoints using Django’s test client or tools like Postman?

Would you like to move to the next one?

👉 **Topic 10: Django Signals – 50 Interview Questions**
Or circle back to review anything?

Topic 10: Django Signals – 50 Interview Questions

You're nearing the finish line! Here's a complete set of:

✅ Topic 10: Django Signals – 50 Interview Questions

A. Signals Basics

1. What are signals in Django?
2. Why are signals used in Django applications?
3. When should you use a signal instead of overriding model methods?
4. How do signals help in decoupling logic?
5. Where are signals typically defined in a Django app?

B. Built-in Django Signals

6. What is the `` `pre_save` `` signal used for?
7. What does the `` `post_save` `` signal do?
8. What is `` `pre_delete` `` used for?
9. What is the `` `post_delete` `` signal used for?
10. What is the difference between `` `m2m_changed` `` and `` `post_save` ``?

C. Connecting and Disconnecting Signals

11. How do you connect a signal to a handler?
12. What is a signal receiver function?
13. What does the `` `@receiver` `` decorator do?
14. How do you pass additional arguments to a signal?
15. How do you disconnect a signal?

D. Signal Handler Behavior

16. What are the arguments passed to a signal handler?
17. What does `` `sender` `` mean in the context of a signal?
18. What is `` `instance` `` in a signal receiver?
19. What does the `` `created` `` flag in `` `post_save` `` indicate?
20. Can a signal handler prevent saving or deleting an object?

E. Custom Signals

21. How do you define a custom signal?
22. How do you connect a custom signal?
23. How do you trigger (send) a custom signal?
24. What are use cases for custom signals?
25. What is the difference between a built-in and a custom signal?

F. Signal Use Cases

26. How do you send a welcome email after a user registers?
27. How do you update a profile automatically after saving a user?
28. How do you log delete actions using `` `post_delete` ``?
29. How do you track audit logs with signals?
30. How can you invalidate a cache on model changes?

G. Best Practices

31. Should signals be placed in `` `models.py` `` or `` `signals.py` ``?
32. How do you ensure signals are loaded properly at runtime?
33. How do you avoid circular import issues with signals?
34. What are the disadvantages of using signals?
35. When should you prefer signals over view logic?

H. Testing and Debugging Signals

36. How do you test signal handlers?
37. How do you mock signals during unit tests?
38. How can you debug if a signal is not firing?
39. What is `` `dispatch_uid` `` and why is it important?

40. How can you prevent multiple signal firings?

I. Advanced Signal Patterns

41. How can you use signals with Django's built-in User model?
42. How can you chain multiple signal receivers?
43. Can you define signal receivers in different apps?
44. How do signals interact with transaction management?
45. Can signals handle asynchronous tasks?

J. Real-world Scenarios

46. How do you trigger an action only on updates (not create)?
47. How do you stop a post-save signal from triggering on bulk_create?
48. How do signals help with domain-driven design?
49. Can you use signals to integrate third-party services?
50. How do signals relate to event-driven architecture?

Next up:
👉 **Topic 11: Django Testing and Debugging – 50 Questions**

Want to keep pushing forward or take a recap break?

Topic 11: Django Testing and Debugging – 50 Questions

Awesome! Here's your complete set for:

✅ Topic 11: Django Testing and Debugging – 50 Interview Questions

---

A. Django Testing Basics

1. What is the purpose of testing in Django?
2. What types of tests does Django support?
3. What is the default testing framework in Django?
4. Where should you write your tests in a Django app?
5. How do you run tests in Django?

B. Unit Testing with `TestCase`

6. What is `` `django.test.TestCase` ``?
7. How is `` `TestCase` `` different from Python’s `` `unittest.TestCase` ``?
8. How do you write a simple unit test for a model?
9. How do you test a view function?
10. How do you test the status code of a response?

C. Testing Models

11. How do you test a model's `` `__str__()` `` method?
12. How do you test custom model methods?
13. How do you set up test data using `` `setUp()` ``?
14. How do you test model relationships (ForeignKey, ManyToMany)?
15. How do you test signal-related behaviors?

D. Testing Views and URLs

16. How do you test a URL route?
17. How do you test redirect behavior?
18. How do you test view templates?
19. How do you test context data in a view?
20. How do you test form submissions in views?

E. Testing Forms and Validation

21. How do you test Django forms?
22. How do you check form validation in tests?
23. How do you simulate POST data for form testing?
24. How do you test form error messages?
25. How do you test custom form validation methods?

F. Testing APIs with Django REST Framework

26. How do you test DRF API views?
27. What is `` `APIClient` `` in DRF testing?
28. How do you test GET, POST, PUT, DELETE methods in DRF?
29. How do you authenticate users in API tests?
30. How do you test JSON responses?

G. Fixtures and Mocking

31. What are fixtures in Django tests?
32. How do you load a fixture for a test?
33. How do you mock external API calls in Django tests?
34. What is `` `patch()` `` used for in mocking?
35. How do you use `` `setUpTestData()` `` for test optimization?

H. Debugging Tools and Techniques

36. How do you debug Django applications?
37. What is `` `pdb` `` and how is it used in Django?
38. What is Django Debug Toolbar?
39. How do you enable detailed error pages in development?
40. How do you inspect SQL queries for performance debugging?

I. Test Coverage and Performance

41. How do you measure test coverage in Django?
42. What tools are used for coverage reporting?
43. How do you run only specific tests?
44. How do you optimize slow test suites?
45. What are the pros and cons of using `` `TransactionTestCase` ``?

J. Best Practices for Django Testing

46. How do you structure test files in large projects?
47. What naming conventions should you follow for tests?
48. How do you avoid data leakage between tests?
49. When should you use mocks vs actual objects?
50. Why is automated testing important in CI/CD pipelines?

Next up:

👉 **Topic 12: Django Deployment and Performance Optimization – 50 Questions**

Let me know if you're ready to finish strong with the final topic!

Topic 12: Django Deployment and Performance Optimization – 50 Questions

You're almost at the finish line — let’s crush this final one!

✅ Topic 12: Django Deployment and Performance Optimization – 50 Interview Questions

---

A. Deployment Basics

1. What are the general steps to deploy a Django application?
2. What is the role of `WSGI` and `ASGI` in deployment?
3. How does `Gunicorn` work with Django?
4. Why is `Nginx` often used in Django deployments?
5. How do you configure static files for production?

B. Production Settings

6. What is `DEBUG = False` and why is it important in production?
7. Why is `ALLOWED_HOSTS` critical in a production environment?
8. How do you handle secret keys in production?
9. What is `SECURE_SSL_REDIRECT` and when should it be used?
10. How do you configure logging for production?

C. Environment Management

11. How do you use `.env` files in Django?
12. What is `django-environ` and how is it useful?
13. How do you set environment variables in a production server?
14. How do you manage different settings for development and production?
15. What is the best way to store and manage credentials?

D. Database Configuration

16. How do you connect Django to a PostgreSQL database?
17. What are the best practices for production database settings?
18. How do you handle database migrations in deployment?
19. How can you backup and restore PostgreSQL databases?
20. How do you optimize database queries in Django?

E. Security Best Practices

21. How do you prevent SQL injection in Django?
22. What is `SECURE_BROWSER_XSS_FILTER`?
23. What is `X_FRAME_OPTIONS` and why is it important?
24. How do you protect against CSRF attacks?
25. What is `SECURE_HSTS_SECONDS` and what does it do?

F. Static and Media Files

26. How do you serve static files in production?
27. How do you serve user-uploaded media files?
28. What is `collectstatic` and when do you run it?
29. How do you configure AWS S3 or other cloud storage with Django?
30. How do you manage static assets with versioning?

G. Performance Optimization

31. How do you enable query optimization in Django?
32. What is `select_related` and `prefetch_related`?
33. How do you use caching in Django?
34. What are cache backends supported by Django?
35. How do you cache entire views?

H. Asynchronous and Scalability

36. What is the difference between synchronous and asynchronous Django views?
37. How does Django support ASGI and async views?
38. What is Celery and how is it used with Django?
39. How do you run periodic tasks in production?
40. What is a message broker and how does it work with Celery?

I. Monitoring and Debugging in Production

41. How do you monitor a Django app in production?
42. What tools can be used to log errors (e.g. Sentry)?
43. How do you configure email notifications for errors?
44. How do you profile a Django app's performance?
45. What metrics should you track in production?

J. Deployment Platforms and CI/CD

46. How do you deploy Django on Heroku?
47. How do you deploy Django using Docker?
48. What are the advantages of containerizing Django?
49. How does a typical CI/CD pipeline work for Django?
50. What are some common pitfalls in Django deployment?

🎉 **Congrats! You've now completed all 12 Django Interview Topics — that's 600 high-quality questions!**

Would you like:

- A **PDF export** of all questions?
- Or to move into **mock interviews**, **topic quizzes**, or maybe **real-world Django projects**?

Let me know how you'd like to continue leveling up!

mock interviews, topic quizzes, or maybe real-world Django projects?

Love your energy — let's break down all three options so you can pick what works best for your next step:

🎯 1. Mock Interviews (Django)

We simulate a **real interview** experience. I'll:

- Ask you **topic-based questions** (multiple rounds or levels).
- Expect short or detailed answers (your call).
- Provide feedback, tips, or corrections immediately.
- Cover **core**, **advanced**, or **full-stack Django** (backend + DRF + deployment).

🧪 You can choose:

- *Technical round only*
- *HR + Tech mix*
- *Live coding (within chat)*
- *Rapid-fire mode (for time pressure)*

📚 2. Topic Quizzes

Each quiz will:

- Cover **one Django topic** (like Models, Middleware, DRF, etc.)
- Be **10–20 questions** each.
- Include MCQs, True/False, Fill-in-the-blanks, Short Code Snippets.
- Provide **instant answers + explanations**.

🔁 Perfect for revision & confidence boosts.

🛠️ 3. Real-World Django Projects

I guide you through full projects like:

- 🛍️ **E-commerce Store** (cart, payments, admin dashboard)
- 🧑‍💼 **Job Board** (user auth, role-based access, resumes)
- 📊 **Analytics Dashboard** (data viz + DRF APIs)
- 🗣️ **Blog with Comments** (Rich Text Editor, moderation, likes)
- 📧 **Email Newsletter App** (Celery, templates, scheduling)

Each project includes:

- Features roadmap
- Model & URL design
- Templates/forms
- Advanced: DRF + async tasks + deployment

👉 So… what's your pick?

- **Mock interview**
- **Topic quiz** (tell me which topic)
- **Real-world project** (choose a type or I can suggest one based on your interests)

Let's go 💻🔥