Django Topics and Questions

Django Architecture

Theory

- What is the MVT (Model-View-Template) architecture in Django?
- What is the difference between ASGI and WSGI?
- What is the difference between a proxy and a reverse proxy?
- What is the purpose of WSGI?
- What is the request-response cycle in Django?
- What is the purpose of `urls.py`?
- What is the purpose of `settings.py`?
- What is the purpose of `asgi.py`?
- What is a web server?
- What is Gunicorn?
- What is the purpose of `Root URL_Config`?
- What is a framework?

Practical

- Create a new Django project using the appropriate command.
- Configure `urls.py` for routing.

Models

Theory

- What is the exact use of `models.py`?
- What is the built-in database in Django?
- What is ORM (Object-Relational Mapping)?
- What are the types of model inheritance in Django?
- What is the difference between Abstract and Proxy model inheritance?
- What is the difference between `null=True` and `blank=True`?
- What are the types of relationship fields available in Django (e.g., one-to-one, one-to-many, many-to-many)?
- What is a reverse relationship (`related_name`)?
- What is the difference between `get()` and `filter()`?
- What are `F` and `Q` objects in Django?
- What is `unique_together` in Django?
- What is the difference between `AbstractUser` and `AbstractBaseUser`?

- Is `User` a default model in Django?
- What is the difference between an attribute and a property (e.g., `user.is_authenticated`)?

Practical

- Create models with one-to-one, one-to-many, and many-to-many relationships.
- Write queries using `filter()`, `get()`, `exclude()`, `include()`, `gt`, `lt`, `contains`, and date fields.
- Use `F` and `Q` objects in queries.
- Write queries to find the average salary and fetch employees with salaries greater than a value.
- Perform a bulk create operation.
- Create an instance of a model.
- Revert (unapply) a migration.
- Write an annotation query.
- Filter records starting with a specific letter (e.g., names starting with 'A').
- Find the second largest price in a model.

Views

Theory

- What is the purpose of `views.py`?
- What are class-based views (CBVs) vs function-based views (FBVs)?
- What are generic views in Django?
- What are view function arguments?
- What is the purpose of `request.POST` and `request.GET` in Django?
- What are query parameters and path parameters?
- What is the difference between `reverse()` and `reverse_lazy()`?
- What is the `render()` function?
- What is the `redirect()` function?

Practical

- Create a view to render an HTML page.
- Pass context to a view and display it in an HTML page.
- Implement a class-based view (CBV).
- Read query parameters in a view.
- Configure path parameters in a view.
- Modify the status code of a response.
- Implement a view with a POST method.
- Use `reverse()` and `reverse_lazy()` in views.

Templates

Theory

- What is the Django template language?
- What are Django filters in templates?
- What is Jinja templating?
- What are `extend` and `block` tags in Django templates?

Practical

- Use `extend` and `block` tags in templates.
- Apply Django template filters.

Forms

Theory

- What is the purpose and functionality of Django Forms?
- What is the difference between forms and Django model forms?

Practical

- Create a Django form.
- Create a Django model form.

Middleware

Theory

- What are middlewares in Django?
- What is the order of middleware execution?
- What is a custom middleware?

Practical

- Create a custom middleware.
- Add middleware to a Django project.

Authentication and Authorization

Theory

- What is Django session management?

- What is the difference between authentication and authorization?
- What is the `authenticate` function in Django?
- What is the purpose of the `@login_required` decorator?
- How does Django handle CSRF protection?
- What is a CSRF attack?
- What is a CSRF token?
- What is session-based authentication?
- What are access and refresh tokens?
- What is the structure of a JWT (JSON Web Token)?
- What is the difference between HTTP 401 and 403?

Practical

- Show an authentication error message.
- Redirect a logged-in user from the login page to the home/dashboard.
- Ensure the home page is visible when pressing back while logged out.
- Use the `@login_required` decorator in a view.
- Implement session-based authentication.
- Set cookies expiry in a response.

HTTP and Networking

Theory

- What are HTTP request methods?
- What is the difference between PUT, PATCH, and POST methods?
- What is the purpose of HTTP OPTIONS and preflight requests?
- What are HTTP headers?
- What is the content of a request?
- What are the components of a URL?
- Can we get data using a POST request or send data using a GET request?
- What is `request.header`, its contents, and why is it used?
- What are common HTTP status codes (e.g., 400, 401, 403)?
- What is CORS (Cross-Origin Resource Sharing)?
- What is content negotiation?
- What is DNS resolving?
- What is the default port for HTTPS?
- What is the purpose of a port in a system and URL?
- What is rate limiting?
- What is UDP?
- What is TCP?
- What are OSI layers?

Practical

- Set an HTTP status code in a response.
- Handle query and path parameters.
- Configure CORS in a Django project.

Database and Migrations

Theory

- What are migration commands and how do they work?
- What are the advantagesậc

System: advantages of ORM over raw SQL?

- What is the `raw` method vs the `cursor` method?

Practical

- Use the `makemigrations` and `migrate` commands.
- Revert a migration.
- Perform a raw SQL query using the `raw` method.
- Use the `cursor` method for database queries.

Signals

Theory

- What are Django signals?
- What is the difference between `pre_save` and `post_save` signals?

Practical

- Implement a signal (e.g., `pre_save` or `post_save`).
- Use signals in a Django project.

Serializers

Theory

- What are serializers in Django?

Practical

- Create a serializer for a model.

Static and Media Files

Theory

- What is the use of `MEDIA_ROOT` and `STATIC_ROOT`?
- What is the difference between static and media URLs?
- How does Django handle static files?

Practical

- Configure static files in Django.
- Handle file uploads in Django.

Caching

Theory

- What are the types of cache in Django?
- What is browser cache, and what are browser storage types?
- How do cookies and sessions work in Django?

Practical

- Implement caching in a Django project.
- Set cookies expiry in a response.

Security

Theory

- How does Django use `SECRET_KEY`?
- What is `ALLOWED_HOSTS` in Django?
- What is a CSRF attack?
- What is a CORS header attack?
- What are password storage mechanisms in Django?

Practical

- Configure `SECRET_KEY` and `ALLOWED_HOSTS` in `settings.py`.
- Implement CSRF protection using `csrf_token`.

Admin

Theory

- What is the purpose of `admin.py`?
- What is a superuser in Django?

Practical

- Configure `admin.py` for a Django admin interface.
- Create a superuser.

Miscellaneous

Theory

- What are context processors?
- What are mixins in Django?
- What are viewsets?
- What are the cons of Django?
- What is `__pycache__` folder usage?
- What is the purpose of `__init__.py` in a Django app?
- What is the `user agent` in a request?

Practical

- Create a context processor.
- Use a mixin in a class-based view.
- Implement a viewset.
- Understand and fix the "confirm form resubmission" issue.