**Explain the role of settings.py in Django.**

In Django, the settings.py file plays a **central role in configuring your project**. It acts like the **control center** for your Django application, where all project-level configurations are defined.

**One-Line Answer (for interviews):**

settings.py contains all the configuration settings for a Django project, including database, installed apps, middleware, static files, templates, and more.

**Key Roles of settings.py:**

* **Database Configuration**: Defines which database engine to use (e.g., SQLite, PostgreSQL, MySQL) and connection details.
  + DATABASES = {
  + 'default': {
  + 'ENGINE': 'django.db.backends.sqlite3',
  + 'NAME': BASE\_DIR / "db.sqlite3",
  + }
  + }
* **Installed Apps**: Lists all apps enabled in the project, including built-in, third-party, and custom apps.
  + INSTALLED\_APPS = [
  + 'django.contrib.admin',
  + 'django.contrib.auth',
  + 'myapp', # your custom app
  + ]
* **Middleware**: Specifies the middleware stack used for request/response processing.
  + MIDDLEWARE = [
  + 'django.middleware.security.SecurityMiddleware',
  + 'django.contrib.sessions.middleware.SessionMiddleware',
  + ...
  + ]
* **Static and Media Files**: Sets paths for serving static files (like CSS/JS) and media files (like uploads).
  + STATIC\_URL = '/static/'
  + MEDIA\_URL = '/media/'
* **Templates Configuration**: Tells Django where to look for HTML templates.
  + TEMPLATES = [
  + {
  + 'BACKEND': 'django.template.backends.django.DjangoTemplates',
  + 'DIRS': [BASE\_DIR / 'templates'],
  + ...
  + },
  + ]
* **Security Settings**: Includes SECRET\_KEY, debug mode, allowed hosts, etc.
  + SECRET\_KEY = 'your-secret-key'
  + DEBUG = True
  + ALLOWED\_HOSTS = []