

## Recent

Sign in to start saving your chats  
Once you're signed in, you can access your recent chats here.  
Sign in

## Conversation with Gemini

prepare 100 mcqs related to django

Here are 100 Multiple Choice Questions (MCQs) related to Django, covering various aspects of the framework.

---

### Django MCQs

**Instructions:** Choose the best answer for each question.

---

**1. What is Django?** a) A content management system b) A programming language c) A high-level Python web framework d) A database management system **Answer: c) A high-level Python web framework**

**2. Which design pattern does Django primarily follow?** a) Model-View-Controller (MVC) b) Model-View-Presenter (MVP) c) Model-Template-View (MTV) d) Model-View-ViewModel (MVVM) **Answer: c) Model-Template-View (MTV)**

**3. What is the default database used by Django for new projects?** a) PostgreSQL b) MySQL c) Oracle d) SQLite **Answer: d) SQLite**

**4. Which command is used to start a new Django project?** a) `django-admin startapp <projectname>` b) `python manage.py startproject <projectname>` c) `django-admin startproject <projectname>` d) `python manage.py createproject <projectname>` **Answer: c) `django-admin startproject <projectname>`**

**5. Which command is used to create a new Django app within a project?** a) `django-admin createapp <appname>` b) `python manage.py newapp <appname>` c) `python manage.py startapp <appname>` d) `django-admin startapp <appname>` **Answer: c) `python manage.py startapp <appname>`**

**6. Which file contains the main settings and configurations for a Django project?** a) `urls.py` b) `views.py` c) `settings.py` d) `models.py` **Answer: c) `settings.py`**

**7. What is the primary purpose of Django's ORM (Object-Relational Mapping)?** a) To render HTML templates b) To handle HTTP requests and responses c) To interact with the database using Python objects instead of raw SQL d) To manage static files **Answer: c) To interact with the database using Python objects instead of raw SQL**

**8. In Django templates, which syntax is used to output a variable?** a) `{% variable %}` b) `{{ variable }}` c) `` `` d) `<?php echo $variable; ?>` **Answer: b) `{{ variable }}`**

**9. Which command is used to start the Django development server?** a) `django-admin runserver` b) `python manage.py startserver` c) `python manage.py runserver` d) `django runserver` **Answer: c) `python manage.py runserver`**

**10. What is the default port used by the Django development server?** a) 5000 b) 3000 c) 8080 d) 8000 **Answer: d) 8000**

**11. Which file is responsible for defining URL patterns in a Django project or app?** a) `views.py` b) `models.py` c) `urls.py` d) `settings.py` **Answer: c) `urls.py`**

**12. What are "migrations" in Django used for?** a) To move data between different databases b) To apply changes made to models to the database schema c) To create new Django projects d) To manage user sessions **Answer: b) To apply changes made to models to the database schema**

**13. Which command creates migration files based on changes in your models?** a) `python manage.py migrate` b) `python manage.py makemigrations` c) `python manage.py createmigrations` d) `python manage.py syncdb` (deprecated) **Answer: b) `python manage.py makemigrations`**

**14. Which command applies the created migration files to the database?** a) `python manage.py applymigrations` b) `python manage.py migrate` c) `python manage.py runmigrations` d) `python manage.py sync` **Answer: b) `python manage.py migrate`**

**15. What is the purpose of the `__str__()` method in a Django model?** a) To define the primary key for the model b) To return a human-readable string representation of the object c) To perform validation on model fields d) To define database relationships **Answer: b) To return a human-readable string representation of the object**

**16. How do you retrieve all objects from a model named `Post`?** a) `Post.objects.get_all()` b) `Post.all()` c) `Post.objects.all()` d) `Post.get_queryset()` **Answer: c) `Post.objects.all()`**

**17. How do you retrieve a single object from a model named `Post` by its primary key (e.g., `id=1`)?** a) `Post.objects.filter(id=1)` b) `Post.objects.get(id=1)` c) `Post.objects.select(id=1)` d) `Post.objects.find(id=1)` **Answer: b) `Post.objects.get(id=1)`**

**18. Which of the following is NOT a valid Django model field type?** a) `CharField` b) `IntegerField` c) `EnumField` d) `DateField` **Answer: c) `EnumField`**

**19. What is the purpose of `INSTALLED_APPS` in `settings.py`?** a) To list external Python packages used in the project b) To list the Django apps that are enabled in the project c) To define database connections d) To configure static file paths **Answer: b) To list the Django apps that are enabled in the project**

**20. How do you include static files (CSS, JavaScript, images) in a Django template?** a) Using `<link href="/static/css/style.css">` b) Using `{% load static %} <link href="{% static 'css/style.css' %}">` c) Using `{{ static 'css/style.css' }}` d) Static files are automatically linked. **Answer: b) {% load static %} <link href="{% static 'css/style.css' %}">**

**21. What is the role of `request.POST` in a Django view?** a) It contains data from GET requests. b) It contains data from form submissions (HTTP POST requests). c) It contains information about the user's browser. d) It is an empty dictionary. **Answer: b) It contains data from form submissions (HTTP POST requests).**

**22. What is the role of `request.GET` in a Django view?** a) It contains data from form submissions (HTTP POST requests). b) It contains data from query parameters in the URL. c) It contains information about the server. d) It is an empty dictionary. **Answer: b) It contains data from query parameters in the URL.**

**23. What is `csrf_token` used for in Django forms?** a) To encrypt user data b) To prevent Cross-Site Request Forgery (CSRF) attacks c) To validate email addresses d) To store user sessions **Answer: b) To prevent Cross-Site Request Forgery (CSRF) attacks**

**24. Which decorator is typically used to restrict a view to authenticated users only?** a) `@login_required` b) `@auth_required` c) `@user_authenticated` d) `@secure_view` **Answer: a) @login\_required**

**25. How do you define a foreign key relationship in a Django model?** a) `my_field = models.ForeignKey('OtherModel')` b) `my_field = models.Relation('OtherModel')` c) `my_field = models.LinkField('OtherModel')` d) `my_field = models.RefField('OtherModel')` **Answer: a) my\_field = models.ForeignKey('OtherModel')**

**26. What is the purpose of `render()` shortcut in Django views?** a) To redirect to another URL b) To send a raw HTTP response c) To load a template, populate it with a context, and return an `HttpResponse` object d) To process form data **Answer: c) To load a template, populate it with a context, and return an `HttpResponse` object**

**27. What is the purpose of `reverse()` in Django?** a) To reverse a string b) To get the URL for a given view name and arguments c) To reverse the order of items in a `QuerySet` d) To reverse database migrations **Answer: b) To get the URL for a given view name and arguments**

**28. How do you include another template within a Django template?** a) `{% import 'header.html' %}` b) `{% include 'header.html' %}` c) `{{ header.html }}` d) `{% load 'header.html' %}` **Answer: b) {% include 'header.html' %}**

**29. What is a "QuerySet" in Django?** a) A database table b) A collection of SQL queries c) A collection of database queries, allowing you to filter, order, and retrieve data d) A form object **Answer: c) A collection of database queries, allowing you to filter, order, and retrieve data**

**30. Which method is used to filter objects from a QuerySet based on specific criteria?** a) `all()` b) `get()` c) `filter()` d) `exclude()` **Answer: c) `filter()`**

**31. What is the difference between `filter()` and `get()` when querying objects?** a) `filter()` returns a QuerySet, `get()` returns a single object. b) `filter()` returns a single object, `get()` returns a QuerySet. c) `filter()` is for exact matches, `get()` is for partial matches. d) There is no difference; they are interchangeable. **Answer: a) `filter()` returns a QuerySet, `get()` returns a single object.**

**32. What happens if `get()` finds more than one object or no object matching the criteria?** a) It returns an empty list. b) It raises an error (`MultipleObjectsReturned` or `DoesNotExist`). c) It returns the first object found. d) It creates a new object. **Answer: b) It raises an error (`MultipleObjectsReturned` or `DoesNotExist`).**

**33. What is the purpose of the `DEBUG = True` setting in `settings.py`?** a) It enables production mode. b) It provides detailed error messages and debugging information. c) It disables logging. d) It optimizes database queries. **Answer: b) It provides detailed error messages and debugging information.**

**34. Which of the following is typically NOT stored in Django's session framework by default?** a) User ID b) Shopping cart contents c) User's IP address d) User's authentication status **Answer: c) User's IP address**

**35. What is the purpose of middleware in Django?** a) To define URL patterns b) To process requests and responses globally, before/after view execution c) To define database models d) To create HTML templates **Answer: b) To process requests and responses globally, before/after view execution**

**36. How can you register a model named `Product` with the Django admin site?** a) Add `Product` to `INSTALLED_APPS` b) `admin.site.register(Product)` in `admin.py` c) Define `admin_register = True` in the `Product` model d) Run `python manage.py register Product` **Answer: b) `admin.site.register(Product)` in `admin.py`**

**37. What is the primary purpose of Django's built-in admin interface?** a) To manage static files b) To provide a web-based interface for managing database content c) To handle URL routing d) To manage user authentication only **Answer: b) To provide a web-based interface for managing database content**

**38. What is the significance of the `wsgi.py` file in a Django project?** a) It defines the database schema. b) It's the entry point for web servers to serve your project. c) It stores URL configurations. d) It manages static files. **Answer: b) It's the entry point for web servers to serve your project.**

**39. What is a "context" in Django templates?** a) A string of HTML code b) A dictionary mapping template variable names to Python objects c) A database query d) A list of installed apps **Answer: b) A dictionary mapping template variable names to Python objects**

**40. Which of the following is used for adding comments in Django templates?** a) `// Comment` b) `` `` c) `{# Comment #}` d) `/** Comment */` **Answer: c) {# Comment #}**

**41. What is the purpose of `HttpResponse` in Django views?** a) To redirect to a new URL. b) To send an HTTP response back to the client, typically containing HTML content. c) To save data to the database. d) To handle form validation. **Answer: b) To send an HTTP response back to the client, typically containing HTML content.**

**42. How can you customize the list display of a model in the Django admin?** a) By defining a `list_display` attribute in the `ModelAdmin` class. b) By overriding the `__str__` method in the model. c) By modifying `settings.py`. d) By using a template tag. **Answer: a) By defining a `list_display` attribute in the `ModelAdmin` class.**

**43. What is the purpose of `ModelForm` in Django?** a) To create forms directly from database models. b) To handle user authentication. c) To define database migrations. d) To render static files. **Answer: a) To create forms directly from database models.**

**44. Which of the following is used for rendering forms in Django templates?** a) `{{ form }}` b) `{% form %}` c) `{% render_form form %}` d) `{{ form.as_p }}` (or `as_ul`, `as_table`) **Answer: d) {{ form.as\_p }} (or as\_ul, as\_table)**

**45. What is the primary function of `manage.py`?** a) To deploy the Django application to a server. b) To interact with your Django project from the command line. c) To manage project dependencies. d) To compile Python code. **Answer: b) To interact with your Django project from the command line.**

**46. Which of the following is NOT a caching strategy supported by Django?** a) File-system caching b) Database caching c) In-memory caching d) Client-side browser caching (directly handled by Django) **Answer: d) Client-side browser caching (directly handled by Django)**

**47. What is the purpose of Django's `select_related()` method on a `QuerySet`?** a) To select specific fields from the database. b) To perform a SQL `JOIN` and retrieve related objects in a single query (for `ForeignKey`/`OneToOneField`). c) To filter related objects. d) To prefetch `ManyToMany` relationships. **Answer: b) To perform a SQL `JOIN` and retrieve related objects in a single query (for `ForeignKey`/`OneToOneField`).**

**48. What is the purpose of Django's `prefetch_related()` method on a `QuerySet`?** a) To perform a SQL `JOIN` and retrieve related objects. b) To retrieve related objects using a separate lookup for each relationship (for `ManyToManyField`/`Reverse ForeignKey`). c) To optimize database writes. d) To pre-populate forms. **Answer: b) To retrieve related objects using a separate lookup for each relationship (for `ManyToManyField`/`Reverse ForeignKey`).**

**49. How do you create a superuser in Django?** a) `python manage.py createsuperuser` b) `python manage.py createadmin` c) `django-admin createsuperuser` d) `python manage.py makesuperuser` **Answer: a) `python manage.py createsuperuser`**

**50. What is a "signal" in Django?** a) A type of database query. b) A mechanism that allows certain senders to notify a set of receivers when certain actions occur. c) A way to manage static files. d) A template tag. **Answer: b) A mechanism that allows certain senders to notify a set of receivers when certain actions occur.**

**51. What is the purpose of Django's messages framework?** a) To send emails to users. b) To display temporary, user-facing notifications. c) To log errors. d) To handle real-time communication. **Answer: b) To display temporary, user-facing notifications.**

**52. Which of the following is a security feature built into Django?** a) Automatic database backup b) CSRF protection c) Client-side input validation d) Automatic code encryption **Answer: b) CSRF protection**

**53. How do you redirect a user to another URL from a Django view?** a) `return HttpResponseRedirect("Redirecting...")` b) `return redirect('/new-url/')` c) `return render(request, 'new_url.html')` d) `return forward('/new-url/')` **Answer: b) `return redirect('/new-url/')`**

**54. What is the purpose of `url()` (or `path()`) in `urls.py`?** a) To define database models. b) To map URLs to view functions or class-based views. c) To configure the web server. d) To manage static files. **Answer: b) To map URLs to view functions or class-based views.**

**55. What is the role of `AUTHENTICATION_BACKENDS` in `settings.py`?** a) To specify which template engines to use. b) To define how users are authenticated (e.g., database, LDAP). c) To configure database connections. d) To list installed apps. **Answer: b) To define how users are authenticated (e.g., database, LDAP).**

**56. Which class-based view is used for displaying a list of objects?** a) `DetailView` b) `CreateView` c) `ListView` d) `TemplateView` **Answer: c) `ListView`**

**57. Which class-based view is used for displaying a single object's details?** a) `ListView` b) `DetailView` c) `FormView` d) `RedirectView` **Answer: b) `DetailView`**

**58. What is the purpose of `MEDIA_ROOT` and `MEDIA_URL` in `settings.py`?** a) To store Django's core files. b) To manage static files like CSS and JS. c) To configure where user-uploaded files are stored and served from. d) To define database media types. **Answer: c) To configure where user-uploaded files are stored and served from.**

**59. What is the difference between `CharField` and `TextField` in Django models?** a) `CharField` stores numbers, `TextField` stores text. b) `CharField` is for small strings, `TextField` is for large amounts of text. c) `CharField` is for foreign keys, `TextField` is for primary keys. d) They are interchangeable. **Answer: b) `CharField` is for small strings, `TextField` is for large amounts of text.**

**60. What is the purpose of `get_absolute_url()` method in a Django model?** a) To retrieve the absolute path to a static file. b) To get the URL for a specific object instance. c)

To get the URL of the current page. d) To define the API endpoint. **Answer: b) To get the URL for a specific object instance.**

**61. Which module provides the `User` and `Group` models for authentication and authorization?** a) `django.contrib.admin` b) `django.contrib.auth` c)

`django.contrib.sessions` d) `django.contrib.messages` **Answer: b)**  
`django.contrib.auth`

**62. What is `urls.py` used for in a Django app?** a) To define database models specific to the app. b) To define view functions for the app. c) To define URL patterns specific to that app. d) To configure the app's settings. **Answer: c) To define URL patterns specific to that app.**

**63. How do you include app-specific URL patterns into the main project's `urls.py`?** a)

`path('blog/', include('blog.urls'))` b) `include_urls('blog.urls')` c)  
`import_urls('blog.urls')` d) `add_urls('blog.urls')` **Answer: a)** `path('blog/', include('blog.urls'))`

**64. What is the purpose of the `DEBUG_PROPAGATE_EXCEPTIONS = True` setting?** a) To catch all exceptions silently. b) To prevent exceptions from being raised. c) To propagate exceptions from middleware to views (or vice-versa) in debug mode. d) To log exceptions to a file. **Answer: c) To propagate exceptions from middleware to views (or vice-versa) in debug mode.**

**65. Which command is used to run unit tests in Django?** a) `python manage.py runtests` b) `python manage.py test` c) `python manage.py check` d) `python manage.py validate` **Answer: b) `python manage.py test`**

**66. What is the purpose of `SessionMiddleware`?** a) To handle user authentication. b) To enable session support in Django. c) To compress HTTP responses. d) To manage static files. **Answer: b) To enable session support in Django.**

**67. What is `HttpRequest` object in Django?** a) An object containing information about the database. b) An object containing metadata about the current request, such as GET/POST data, user info, etc. c) An object used for rendering templates. d) An object representing a database model. **Answer: b) An object containing metadata about the current request, such as GET/POST data, user info, etc.**

**68. What is `HttpResponse` object in Django?** a) An object representing the incoming request from the client. b) An object used for database interactions. c) An object representing the response to be sent back to the client. d) An object for handling user sessions. **Answer: c) An object representing the response to be sent back to the client.**

**69. Which file is generated automatically when you create a new Django project?** a) `views.py` b) `models.py` c) `manage.py` d) `templates.py` **Answer: c) `manage.py`**

**70. What is `TEMPLATE_DIRS` (or `DIRS` within `TEMPLATES` setting) used for?** a) To specify where Django should look for static files. b) To specify where Django should look for HTML templates. c) To specify where Django should look for database files. d) To specify where

Django should look for Python modules. **Answer: b) To specify where Django should look for HTML templates.**

**71. How do you implement a Many-to-Many relationship in Django models?** a)

`my_field = models.ManyToManyField('OtherModel')` b) `my_field = models.ManyToManyField('OtherModel')` c) `my_field = models.RelationalField('OtherModel')` d) `my_field = models.MultipleField('OtherModel')` **Answer: a) `my_field = models.ManyToManyField('OtherModel')`**

**72. What is the purpose of the `form.is_valid()` method?** a) To save the form data to the database. b) To check if the submitted form data is valid according to the form's defined fields and rules. c) To render the form in the template. d) To redirect the user. **Answer: b) To check if the submitted form data is valid according to the form's defined fields and rules.**

**73. Which of the following is used for filtering data using a case-insensitive "contains" lookup?** a) `field__contains` b) `field__icontains` c) `field__exact` d) `field__iregex` **Answer: b) `field__icontains`**

**74. What does `{% extends 'base.html' %}` do in a Django template?** a) It includes the content of `base.html`. b) It defines a block in `base.html`. c) It indicates that the current template inherits from `base.html`. d) It defines a variable named `base.html`. **Answer: c) It indicates that the current template inherits from `base.html`.**

**75. What is the purpose of `{% block content %}` and `{% endblock content %}` in Django templates?** a) To define a reusable component. b) To create a loop. c) To define a section that can be overridden by child templates. d) To include another template. **Answer: c) To define a section that can be overridden by child templates.**

**76. Which of the following is a commonly used third-party package for building REST APIs with Django?** a) Django CMS b) Django REST Framework (DRF) c) Django Oscar d) Django Haystack **Answer: b) Django REST Framework (DRF)**

**77. What is the main benefit of using Class-Based Views (CBVs) over Function-Based Views (FBVs)?** a) CBVs are always faster. b) CBVs promote code reusability and organization through inheritance and mixins. c) CBVs are easier to debug. d) CBVs do not require URL patterns. **Answer: b) CBVs promote code reusability and organization through inheritance and mixins.**

**78. What is the `SITE_ID` setting in Django used for?** a) To identify the current server's IP address. b) To uniquely identify the current Django site in a multi-site setup. c) To set the default language of the site. d) To configure static file serving. **Answer: b) To uniquely identify the current Django site in a multi-site setup.**

**79. How can you define custom template tags and filters in Django?** a) By modifying `settings.py`. b) By creating a `templatetags` directory within an app and defining Python files. c) They are not customizable. d) By adding comments in templates. **Answer: b) By creating a `templatetags` directory within an app and defining Python files.**



**80. What is the purpose of the `Meta` class in a Django model?** a) To define database connections. b) To provide metadata options for the model, such as ordering, table name, and permissions. c) To define view functions. d) To manage forms. **Answer: b) To provide metadata options for the model, such as ordering, table name, and permissions.**

**81. Which command is used to check for potential problems in your Django project without running tests?** a) `python manage.py validate` b) `python manage.py check` c) `python manage.py inspect` d) `python manage.py verify` **Answer: b) `python manage.py check`**

**82. What is the purpose of `APPEND_SLASH = True` in `settings.py`?** a) It automatically appends a slash to URLs if not present. b) It prevents trailing slashes in URLs. c) It is used for URL redirection. d) It defines the root URL. **Answer: a) It automatically appends a slash to URLs if not present.**

**83. What is the role of `STATIC_ROOT` in `settings.py`?** a) The URL from which static files are served. b) The directory where Django collects static files for deployment. c) The directory where user-uploaded media files are stored. d) The directory where templates are stored. **Answer: b) The directory where Django collects static files for deployment.**

**84. What is the role of `STATIC_URL` in `settings.py`?** a) The absolute path to the directory where collected static files are stored. b) The URL prefix for static files when serving them. c) The URL for user-uploaded media files. d) The URL for the admin interface. **Answer: b) The URL prefix for static files when serving them.**

**85. Which filter in Django templates is used to format a date?** a) `|format_date` b) `|date` c) `|to_date` d) `|datetime` **Answer: b) `|date`**

**86. What is the purpose of `DecimalField` in Django models?** a) To store floating-point numbers with high precision. b) To store integers. c) To store text with decimal points. d) To store dates and times. **Answer: a) To store floating-point numbers with high precision.**

**87. How do you handle file uploads in Django forms?** a) By using `CharField` for file fields. b) By using `FileField` or `ImageField` in models and handling the file in the view. c) File uploads are handled automatically by Django. d) Django does not support file uploads. **Answer: b) By using `FileField` or `ImageField` in models and handling the file in the view.**

**88. What is the purpose of `AUTH_USER_MODEL` in `settings.py`?** a) To specify the default authentication backend. b) To specify a custom user model to be used instead of Django's default `User` model. c) To define user permissions. d) To enable user registration. **Answer: b) To specify a custom user model to be used instead of Django's default `User` model.**

**89. What is `makemessages` command used for?** a) To create new Django applications. b) To extract translatable strings from your project for internationalization. c) To send messages to users. d) To generate database migrations. **Answer: b) To extract translatable strings from your project for internationalization.**

**90. What is `compilemessages` command used for?** a) To compile Python code. b) To compile `.po` files (translation files) into `.mo` files for use by Django. c) To compile static

files. d) To compile database queries. **Answer: b) To compile .po files (translation files) into .mo files for use by Django.**

**91. What is the purpose of `slug` in Django models?** a) A unique identifier for database records. b) A short label containing only letters, numbers, underscores or hyphens, typically used in URLs. c) A field for storing large amounts of text. d) A field for storing images. **Answer: b) A short label containing only letters, numbers, underscores or hyphens, typically used in URLs.**

**92. How do you make a model field nullable in the database?** a) `my_field = models.CharField(max_length=255, required=False)` b) `my_field = models.CharField(max_length=255, default=None)` c) `my_field = models.CharField(max_length=255, null=True, blank=True)` d) `my_field = models.CharField(max_length=255, optional=True)` **Answer: c) `my_field = models.CharField(max_length=255, null=True, blank=True)`**

**93. What is the purpose of `SessionStore` in Django?** a) To store data in the browser's local storage. b) To provide an interface for interacting with the session database. c) To manage static files. d) To handle caching. **Answer: b) To provide an interface for interacting with the session database.**

**94. Which type of model inheritance is used when you only want the parent's fields to be used as fields for the child model, but no database table is created for the parent?** a) Multi-table inheritance b) Abstract base classes c) Proxy models d) Concrete inheritance **Answer: b) Abstract base classes**

**95. What is the purpose of `AUTH_PASSWORD_VALIDATORS` in `settings.py`?** a) To validate user email addresses. b) To configure password validation rules for user passwords. c) To validate form submissions. d) To validate database connections. **Answer: b) To configure password validation rules for user passwords.**

**96. What is the common way to handle static files in production with Django?** a) Serve them directly from Django's development server. b) Use `python manage.py runserver --nostatic`. c) Use a dedicated web server (like Nginx or Apache) to serve static files directly. d) Store them in the database. **Answer: c) Use a dedicated web server (like Nginx or Apache) to serve static files directly.**

**97. What is `ADMIN_URL` setting used for (though not a default Django setting)?** a) To define the URL path to the Django admin interface. b) To define the admin user's email. c) To set the default admin theme. d) It's not a standard Django setting. **Answer: a) To define the URL path to the Django admin interface. (While not a default setting, it's a common practice to define this in `urls.py`.)**

**98. How do you handle errors in Django views (e.g., 404 Not Found)?** a) By raising Python exceptions directly. b) By returning `HttpResponseNotFound` or using Django's `Http404` exception. c) Errors are handled automatically by the framework. d) By manually creating a custom error HTML page. **Answer: b) By returning `HttpResponseNotFound` or using Django's `Http404` exception.**

**99. Which of the following is a common reason for a `DoesNotExist` exception?** a) A database table is missing. b) A `get()` query did not find a matching object. c) A form submission failed validation. d) A static file could not be found. **Answer: b) A `get()` query did not find a matching object.**

**100. What is the main purpose of `urls.py` within a Django project's main directory?** a) To define global settings for the project. b) To serve as the root URL configuration for the entire project, dispatching to app-specific URLs. c) To define database models. d) To handle all HTTP requests directly. **Answer: b) To serve as the root URL configuration for the entire project, dispatching to app-specific URLs.**

---

Sources

prepare 100 mcqs related to react

Here are 100 MCQs related to React, covering core concepts, hooks, state management, routing, and common patterns.

---

## React MCQs

**Instructions:** Choose the best answer for each question.

---

**1. What is React.js?** a) A backend framework for building APIs b) A JavaScript library for building user interfaces c) A database management system d) A CSS preprocessor **Answer: b) A JavaScript library for building user interfaces**

**2. Which company developed React?** a) Google b) Microsoft c) Facebook (Meta) d) Amazon **Answer: c) Facebook (Meta)**

**3. What is JSX in React?** a) A JavaScript extension for adding types b) A syntax extension for JavaScript that allows you to write HTML-like code in JavaScript c) A templating language for React d) A JavaScript compiler **Answer: b) A syntax extension for JavaScript that allows you to write HTML-like code in JavaScript**

**4. What is the Virtual DOM in React?** a) A direct representation of the browser's DOM b) A lightweight copy of the actual DOM, kept in memory c) A server-side rendering technology d) A database for storing UI states **Answer: b) A lightweight copy of the actual DOM, kept in memory**

**5. How does React update the actual DOM?** a) By directly manipulating the entire DOM b) By comparing the Virtual DOM with the actual DOM and applying only the necessary changes c) By reloading the entire page d) By using WebAssembly **Answer: b) By comparing the Virtual DOM with the actual DOM and applying only the necessary changes**

**6. What is the purpose of `ReactDOM.render()`?** a) To create a new React component b) To update the state of a component c) To render a React element into the DOM in the supplied container d) To define a functional component **Answer: c) To render a React element into the DOM in the supplied container**

**7. Which method is called immediately after a component is mounted (inserted into the DOM)?** a) `componentWillMount()` (deprecated) b) `render()` c) `componentDidMount()` d) `componentWillUnmount()` **Answer: c) `componentDidMount()`**

**8. Which of the following is NOT a phase in a React component's lifecycle (for class components)?** a) Mounting b) Updating c) Destroying d) Unmounting **Answer: c) Destroying**

**9. What are "props" in React?** a) Internal data of a component that can change over time b) Properties passed from a parent component to a child component c) Events triggered by user interactions d) Methods that manipulate the DOM **Answer: b) Properties passed from a parent component to a child component**

**10. Can child components directly modify the props they receive from a parent component?** a) Yes b) No c) Only if the prop is a function d) Only if the prop is a primitive type **Answer: b) No**

**11. What is "state" in React?** a) Data passed from parent to child components b) Internal data of a component that can change over time and trigger re-renders c) Global application data d) Static data that never changes **Answer: b) Internal data of a component that can change over time and trigger re-renders**

**12. How do you update the state in a class component?** a) `this.state = { new_value: ... }` b) `this.updateState({ new_value: ... })` c) `this.setState({ new_value: ... })` d) `this.changeState({ new_value: ... })` **Answer: c) `this.setState({ new_value: ... })`**

**13. What is the purpose of the `key` prop when rendering lists of elements in React?** a) To apply unique styling b) To identify which items have changed, are added, or are removed, optimizing updates c) To specify the order of elements d) To store additional data about the list item **Answer: b) To identify which items have changed, are added, or are removed, optimizing updates**

**14. What are "Hooks" in React?** a) Functions that allow you to use state and other React features in class components b) Functions that let you "hook into" React state and lifecycle features from functional components c) A way to connect React components to a Redux store d) A tool for debugging React applications **Answer: b) Functions that let you "hook into" React state and lifecycle features from functional components**

**15. Which Hook is used to add state to functional components?** a) `useEffect` b) `useContext` c) `useState` d) `useReducer` **Answer: c) `useState`**

**16. Which Hook is used for performing side effects (like data fetching, subscriptions, or manually changing the DOM) in functional components?** a) `useState` b) `useRef` c) `useEffect` d) `useMemo` **Answer: c) `useEffect`**

**17. What does the dependency array in `useEffect` control?** a) The order in which effects are executed b) When the effect should re-run c) The initial state of the component d) The props passed to the component **Answer: b) When the effect should re-run**

**18. If the dependency array in `useEffect` is empty (`[]`), when will the effect run?** a) On every render b) Only when the component mounts and unmounts c) Never d) Only when state changes **Answer: b) Only when the component mounts and unmounts**

**19. What is `React.memo()` used for?** a) To memoize a function result b) To prevent a functional component from re-rendering if its props have not changed c) To store component state globally d) To create a new component **Answer: b) To prevent a functional component from re-rendering if its props have not changed**

**20. What is a "Higher-Order Component" (HOC)?** a) A component that renders other components b) A function that takes a component as an argument and returns a new, enhanced component c) A component that manages global state d) A component with a higher rendering priority **Answer: b) A function that takes a component as an argument and returns a new, enhanced component**

**21. What is "prop drilling"?** a) The process of accessing props directly from the DOM b) The practice of passing data through multiple layers of nested components c) A method for optimizing component rendering d) A technique for styling components **Answer: b) The practice of passing data through multiple layers of nested components**

**22. Which of the following is a solution to "prop drilling"?** a) Using more props b) Using the Context API or a state management library (e.g., Redux) c) Avoiding nested components d) Using inline styles **Answer: b) Using the Context API or a state management library (e.g., Redux)**

**23. What is the Context API in React used for?** a) To manage local component state b) To provide a way to pass data through the component tree without passing props manually at every level c) To perform asynchronous operations d) To optimize image loading **Answer: b) To provide a way to pass data through the component tree without passing props manually at every level**

**24. How do you consume context in a functional component?** a) `React.createContext()` b) `useContext(MyContext)` c) `this.context` d) `MyContext.Consumer` (for class components, but not the primary for functions) **Answer: b) `useContext(MyContext)`**

**25. What is the main purpose of React Router?** a) To manage global state b) To handle navigation and routing in single-page applications c) To perform network requests d) To build forms **Answer: b) To handle navigation and routing in single-page applications**

**26. Which component from React Router is used to define a route?** a) `<Link>` b) `<Route>` c) `<Router>` d) `<Switch>` (or `<Routes>` in v6) **Answer: b) `<Route>`**

**27. Which component from React Router is used for navigation links?** a) `<Route>` b) `<Navigate>` c) `<Link>` d) `<URL>` **Answer: c) `<Link>`**

**28. What is the `useNavigate` Hook in React Router v6 used for?** a) To get the current URL path b) To programmatically navigate to a different route c) To access route parameters d) To display a navigation bar **Answer: b) To programmatically navigate to a different route**

**29. What is Redux?** a) A database for React applications b) A predictable state container for JavaScript applications c) A styling library for React d) A testing framework for React **Answer: b) A predictable state container for JavaScript applications**

**30. What are the three core principles of Redux?** a) Single source of truth, state is read-only, changes are made with pure functions (reducers) b) Multiple stores, mutable state, direct DOM manipulation c) Component-level state, props drilling, imperative updates d) Virtual DOM, JSX, one-way data flow **Answer: a) Single source of truth, state is read-only, changes are made with pure functions (reducers)**

**31. In Redux, what is a "reducer"?** a) A function that dispatches actions b) A function that transforms data from the server c) A pure function that takes the current state and an action, and returns the new state d) A component that displays data from the store **Answer: c) A pure function that takes the current state and an action, and returns the new state**

**32. What is an "action" in Redux?** a) A JavaScript function b) A plain JavaScript object that describes what happened c) A component that triggers state changes d) A method for asynchronously fetching data **Answer: b) A plain JavaScript object that describes what happened**

**33. What is the purpose of the Redux "store"?** a) To manage component-specific state b) To hold the entire application's state tree c) To render UI components d) To handle routing **Answer: b) To hold the entire application's state tree**

**34. Which React-Redux hook is used to access data from the Redux store?** a) `useDispatch()` b) `useSelector()` c) `useStore()` d) `useReducer()` **Answer: b) `useSelector()`**

**35. Which React-Redux hook is used to dispatch actions?** a) `useSelector()` b) `useDispatch()` c) `useAction()` d) `useStore()` **Answer: b) `useDispatch()`**

**36. What is a "thunk" in Redux (e.g., with Redux Thunk)?** a) A type of reducer b) A function that wraps a Redux action to allow asynchronous logic c) A special type of Redux store d) A method for optimizing rendering **Answer: b) A function that wraps a Redux action to allow asynchronous logic**

**37. What is `PropTypes` used for in React?** a) To validate component state b) To define the type of props that a component expects c) To perform server-side validation d) To create animations **Answer: b) To define the type of props that a component expects**

**38. Which of the following is a correct way to define a functional component?** a) `const MyComponent = () => { return <div>Hello</div>; }` b) `class MyComponent extends React.Component { render() { return <div>Hello</div>; } }` c) `function MyComponent() { return <div>Hello</div>; }` d) Both a) and c) **Answer: d) Both a) and c)**

**39. Which of the following is a correct way to define a class component?** a) `const MyComponent = () => { return <div>Hello</div>; }` b) `class MyComponent extends React.Component { render() { return <div>Hello</div>; } }` c) `function MyComponent() { return <div>Hello</div>; }` d) All of the above **Answer: b) `class MyComponent extends React.Component { render() { return <div>Hello</div>; } }`**

**40. What is "lifting state up" in React?** a) Moving state from a child component to a parent component b) Moving state from a parent component to a child component c) Using global state management d) Optimizing state updates **Answer: a) Moving state from a child component to a parent component**

**41. What is the purpose of `useRef` Hook?** a) To access the previous state b) To create a mutable `ref` object whose `.current` property is initialized to the passed argument c) To perform side effects d) To manage component lifecycle **Answer: b) To create a mutable `ref` object whose `.current` property is initialized to the passed argument**

**42. When should you use `useMemo`?** a) To perform side effects b) To memoize a function definition c) To memoize a computed value, preventing unnecessary re-calculations d) To manage component state **Answer: c) To memoize a computed value, preventing unnecessary re-calculations**

**43. What is the purpose of `useCallback`?** a) To memoize a computed value b) To return a memoized callback function, preventing unnecessary re-creation of functions c) To perform asynchronous operations d) To manage global state **Answer: b) To return a memoized callback function, preventing unnecessary re-creation of functions**

**44. What is `Strict Mode` in React?** a) A mode that prevents any state updates b) A tool for highlighting potential problems in an application, rendering twice in development c) A mode for production deployment only d) A mode that disables all warnings **Answer: b) A tool for highlighting potential problems in an application, rendering twice in development**

**45. Which method is specific to class components and is used for error handling within the component tree?** a) `componentDidCatch()` b) `catchErrors()` c) `handleError()` d) `onError()` **Answer: a) `componentDidCatch()`**

**46. What is a "controlled component" in React forms?** a) A component whose value is managed by the DOM b) A component whose value is controlled by React state c) A component that has no state d) A component that uses `useRef` **Answer: b) A component whose value is controlled by React state**

**47. What is an "uncontrolled component" in React forms?** a) A component whose value is controlled by React state b) A component whose value is managed by the DOM, and you access it using a `ref` c) A component that has no props d) A component that renders only once **Answer: b) A component whose value is managed by the DOM, and you access it using a `ref`**

**48. What is the purpose of `event.preventDefault()` in React?** a) To stop event propagation to parent elements b) To prevent the default action of an event (e.g., form submission, link navigation) c) To log event details to the console d) To attach an event

listener **Answer: b) To prevent the default action of an event (e.g., form submission, link navigation)**

**49. What is `React.Fragment` used for?** a) To group multiple elements without adding an extra node to the DOM b) To create a new component c) To define a reusable piece of UI d) To optimize rendering performance **Answer: a) To group multiple elements without adding an extra node to the DOM**

**50. What is the default port for a React development server created with Create React App?** a) 5000 b) 3000 c) 8080 d) 8000 **Answer: b) 3000**

**51. How do you create a new React app using Create React App?** a) `npm create react-app my-app` b) `npx create-react-app my-app` c) `yarn create react-app my-app` d) Both b) and c) are valid (depending on preferred package manager) **Answer: d) Both b) and c) are valid (depending on preferred package manager)**

**52. What is `ReactDOM.createPortal()` used for?** a) To create a new React component b) To render children into a DOM node that exists outside the DOM hierarchy of the parent component c) To optimize server-side rendering d) To manage global state **Answer: b) To render children into a DOM node that exists outside the DOM hierarchy of the parent component**

**53. What is the purpose of `shouldComponentUpdate()` in class components?** a) To force a component to re-render b) To prevent a component from re-rendering if its props or state have not changed (for performance optimization) c) To update the component's state d) To define initial component props **Answer: b) To prevent a component from re-rendering if its props or state have not changed (for performance optimization)**

**54. Which lifecycle method is called right before a component is removed from the DOM?** a) `componentWillMount()` (deprecated) b) `componentDidUnmount()` c) `componentWillUnmount()` d) `render()` **Answer: c) `componentWillUnmount()`**

**55. What are "memoized" components?** a) Components that store their state in local storage b) Components whose render output is cached and only re-rendered if their props or state dependencies change c) Components that use `useMemo` d) Components that always re-render **Answer: b) Components whose render output is cached and only re-rendered if their props or state dependencies change**

**56. What is the purpose of `children prop`?** a) To pass data from a child component to a parent component b) To render content passed directly between the opening and closing tags of a component c) To define the component's state d) To provide a default value for a prop **Answer: b) To render content passed directly between the opening and closing tags of a component**

**57. How do you typically handle asynchronous operations (e.g., API calls) in functional components?** a) Directly inside the `render` method b) Inside `useState` c) Inside `useEffect` d) Using `useRef` **Answer: c) Inside `useEffect`**

**58. What is `componentDidUpdate()` used for in class components?** a) To perform actions after the component's initial render b) To perform actions after the component updates (due to



prop or state changes) c) To prevent unnecessary re-renders d) To handle errors **Answer: b) To perform actions after the component updates (due to prop or state changes)**

**59. What is `getDerivedStateFromProps()` used for in class components?** a) To update state based on props, just before rendering b) To fetch data asynchronously c) To define initial state d) To handle events **Answer: a) To update state based on props, just before rendering**

**60. What is the main difference between functional and class components BEFORE Hooks?** a) Functional components could not have state or lifecycle methods. b) Class components were faster. c) Functional components were only for styling. d) Class components could not receive props. **Answer: a) Functional components could not have state or lifecycle methods.**

**61. What is the purpose of `Error Boundaries` in React?** a) To catch JavaScript errors anywhere in their child component tree, log those errors, and display a fallback UI b) To define a component's maximum width c) To prevent infinite loops in rendering d) To handle HTTP errors **Answer: a) To catch JavaScript errors anywhere in their child component tree, log those errors, and display a fallback UI**

**62. What is `lazy` and `Suspense` in React used for?** a) To prefetch data before rendering b) For code-splitting and loading components on demand (lazy loading) c) To manage component state asynchronously d) To create animations **Answer: b) For code-splitting and loading components on demand (lazy loading)**

**63. What is the syntax for creating a default export for a component?** a) `export MyComponent;` b) `export default MyComponent;` c) `import MyComponent;` d) `module.exports = MyComponent;` **Answer: b) `export default MyComponent;`**

**64. What is the syntax for a named import of a component?** a) `import MyComponent from './MyComponent';` b) `import { MyComponent } from './MyComponent';` c) `require('MyComponent');` d) `const MyComponent = require('./MyComponent');` **Answer: b) `import { MyComponent } from './MyComponent';`**

**65. What is the purpose of `dangerouslySetInnerHTML`?** a) To safely set HTML content b) To inject raw HTML into the DOM, bypassing React's sanitization c) To prevent XSS attacks d) To set CSS styles dynamically **Answer: b) To inject raw HTML into the DOM, bypassing React's sanitization**

**66. Which tool is commonly used to create React components visually and in isolation?** a) Webpack b) Babel c) Storybook d) ESLint **Answer: c) Storybook**

**67. What is the primary benefit of using `StrictMode` during development?** a) Faster build times b) Improved production performance c) Detection of potential problems and bad practices d) Automatic code formatting **Answer: c) Detection of potential problems and bad practices**

**68. How do you pass data from a child component to a parent component?** a) Directly modify parent's state from child b) Pass a callback function from parent to child, and the child

invokes it c) Use `props.children` d) It's not possible in React's unidirectional data flow

**Answer: b) Pass a callback function from parent to child, and the child invokes it**

**69. What is a "render prop" in React?** a) A prop that is used to define the styling of a component b) A prop whose value is a function that a component calls to render something c) A prop that prevents rendering d) A prop that is only available during server-side rendering

**Answer: b) A prop whose value is a function that a component calls to render something**

**70. What is the role of Babel in a React project?** a) To manage project dependencies b) To transpile JSX and modern JavaScript into browser-compatible JavaScript c) To handle state management d) To perform testing **Answer: b) To transpile JSX and modern JavaScript into browser-compatible JavaScript**

**71. What is the role of Webpack in a React project?** a) To manage database connections b) A module bundler that packs all JavaScript files into a single file c) A testing framework d) A code editor **Answer: b) A module bundler that packs all JavaScript files into a single file**

**72. What is the `setState` (or state updater function in `useState`) callback function used for?** a) To execute code immediately after the state has been updated and the component has re-rendered b) To fetch data asynchronously c) To update multiple state variables simultaneously d) To prevent re-renders **Answer: a) To execute code immediately after the state has been updated and the component has re-rendered**

**73. What is the difference between `React.Fragment` and `<>` (short syntax)?** a) There is no difference; they are identical. b) `<>` allows key prop, `React.Fragment` does not. c) `React.Fragment` allows key prop, `<>` does not. d) `React.Fragment` can only wrap one element. **Answer: c) `React.Fragment` allows key prop, `<>` does not.**

**74. What is the default behavior of `setState` (and `useState`'s setter) when updating state with an object?** a) It merges the new object with the previous state. b) It completely replaces the previous state object. c) It creates a new instance of the component. d) It logs an error. **Answer: a) It merges the new object with the previous state.**

**75. How can you prevent a functional component from re-rendering if its props haven't changed?** a) Using `shouldComponentUpdate` (class component only) b) Using `React.memo()` c) Using `PureComponent` (class component only) d) By not passing any props **Answer: b) Using `React.memo()`**

**76. What is the purpose of `useReducer` Hook?** a) To manage simple state b) To manage complex state logic, often an alternative to `useState` for complex state c) To perform side effects d) To access the DOM directly **Answer: b) To manage complex state logic, often an alternative to `useState` for complex state**

**77. What does the `dispatch` function returned by `useReducer` do?** a) It directly updates the state. b) It triggers a re-render of the component. c) It sends an action to the reducer function to update the state. d) It fetches data. **Answer: c) It sends an action to the reducer function to update the state.**

**78. Which of the following is NOT a benefit of using React?** a) Component-based architecture b) Improved performance with Virtual DOM c) Automatic database integration d) Strong community support **Answer: c) Automatic database integration**

**79. What is the purpose of React Developer Tools?** a) To write React code faster b) A browser extension for debugging React component hierarchies and state/props c) A command-line interface for React d) A tool for deploying React applications **Answer: b) A browser extension for debugging React component hierarchies and state/props**

**80. What is hydration in React?** a) The process of re-rendering a component b) The process of attaching React event listeners and data to server-rendered HTML c) The process of fetching data from an API d) The process of optimizing images **Answer: b) The process of attaching React event listeners and data to server-rendered HTML**

**81. What is the primary difference between BrowserRouter and HashRouter in React Router?** a) BrowserRouter uses hash (#) in the URL, HashRouter uses the HTML5 history API. b) BrowserRouter uses the HTML5 history API, HashRouter uses hash (#) in the URL. c) BrowserRouter is for server-side rendering, HashRouter is for client-side rendering. d) BrowserRouter is for desktop, HashRouter is for mobile. **Answer: b) BrowserRouter uses the HTML5 history API, HashRouter uses hash (#) in the URL.**

**82. What is the useEffect cleanup function used for?** a) To reset the component's state b) To perform actions when the component unmounts or before the next effect runs c) To clear console logs d) To remove event listeners only **Answer: b) To perform actions when the component unmounts or before the next effect runs**

**83. What is the purpose of useState's functional update?** a) To always return a new state object b) To use the previous state value to calculate the new state, ensuring correctness for asynchronous updates c) To update multiple state variables at once d) To prevent unnecessary re-renders **Answer: b) To use the previous state value to calculate the new state, ensuring correctness for asynchronous updates**

**84. When should you use useLayoutEffect instead of useEffect?** a) When performing asynchronous operations b) When you need to read layout from the DOM and synchronously re-render before the browser paints c) When you want to fetch data from an API d) `useEffect` is always preferred. **Answer: b) When you need to read layout from the DOM and synchronously re-render before the browser paints**

**85. What is the concept of "unidirectional data flow" in React?** a) Data flows only from child to parent. b) Data flows only from parent to child. c) Data flows bidirectionally between components. d) Data flows directly to the DOM. **Answer: b) Data flows only from parent to child.**

**86. What is a "controlled input" in React?** a) An input element whose value is managed by the DOM. b) An input element whose value is controlled by React state. c) An input element that only accepts specific types of data. d) An input element that is read-only. **Answer: b) An input element whose value is controlled by React state.**

**87. What is the purpose of React.createElement()?** a) To create a DOM element directly. b) The underlying function called by JSX to create React elements (descriptions of

what to render). c) To create a new component. d) To add an event listener. **Answer: b) The underlying function called by JSX to create React elements (descriptions of what to render).**

**88. How do you import the `useState` Hook?** a) `import useState from 'react';` b) `import { useState } from 'react';` c) `import { Hooks } from 'react';` d) `import React.useState from 'react';` **Answer: b) `import { useState } from 'react';`**

**89. What is the purpose of `strictEquals` in `Redux useSelector`'s equality function?** a) To perform a deep comparison of objects. b) To perform a shallow comparison of objects. c) To check if two values are strictly equal (`===`). d) To prevent any re-renders. **Answer: c) To check if two values are strictly equal (`===`).**

**90. What is a "custom Hook" in React?** a) A built-in React Hook. b) A JavaScript function whose name starts with "use" and that calls other Hooks, used for reusing stateful logic. c) A function component that renders nothing. d) A component that provides context. **Answer: b) A JavaScript function whose name starts with "use" and that calls other Hooks, used for reusing stateful logic.**

**91. What is the recommended way to handle forms in React?** a) Directly manipulating the DOM with `document.getElementById` b) Using controlled components with React state c) Using uncontrolled components exclusively d) Avoiding forms altogether **Answer: b) Using controlled components with React state**

**92. What is `React.Children.map()` used for?** a) To iterate over an array of children and return a new array b) To filter children components c) To get the first child component d) To count the number of children **Answer: a) To iterate over an array of children and return a new array**

**93. What is the `useContext` Hook useful for?** a) Managing component-specific state b) Subscribing to React context without introducing nesting c) Performing asynchronous operations d) Optimizing rendering performance **Answer: b) Subscribing to React context without introducing nesting**

**94. What is the `React.StrictMode` component used for?** a) To enable faster rendering in production. b) To run additional checks and warnings for potential issues in your app during development. c) To apply strict CSS rules. d) To force all components to be class components. **Answer: b) To run additional checks and warnings for potential issues in your app during development.**

**95. What is the purpose of the `displayName` property on React components?** a) To specify the component's default props. b) To provide a name for debugging messages and development tools. c) To define the component's state. d) To set the component's unique key. **Answer: b) To provide a name for debugging messages and development tools.**

**96. What is memoization in the context of React?** a) Storing component state in local storage. b) Optimizing performance by caching results of expensive function calls and returning the cached result when inputs are the same. c) Remembering user preferences. d) Automatic garbage collection. **Answer: b) Optimizing performance by caching results of expensive function calls and returning the cached result when inputs are the same.**

**97. What does the `useRef` hook return?** a) A plain JavaScript object. b) A mutable `ref` object whose `.current` property is initialized to the passed argument. c) A function to update state. d) An array of dependencies. **Answer: b) A mutable `ref` object whose `.current` property is initialized to the passed argument.**

**98. What is the `createRoot()` function in React 18 used for?** a) To replace `ReactDOM.render()` for concurrent mode and new features. b) To create a new component. c) To initialize the Redux store. d) To manage routing. **Answer: a) To replace `ReactDOM.render()` for concurrent mode and new features.**

**99. Which package is required to use React in a web browser environment?** a) `react-native` b) `react-dom` c) `react-router-dom` d) `redux` **Answer: b) `react-dom`**

**100. What is the latest major stable version of React (as of May 2025)?** a) React 16 b) React 17 c) React 18 d) React 19 **Answer: d) React 19 (React 19.1.0 is the latest stable as of May 2025)**

---

WHAT is the difference between `makemigrations` and `migrate` in django

In Django, `makemigrations` and `migrate` are two fundamental commands used to manage changes to your database schema based on your Django models. They work hand-in-hand but serve distinct purposes:

`python manage.py makemigrations`

**What it does:**

- **Detects changes in your models:** `makemigrations` inspects your `models.py` files (and any other files where you define Django models) and compares them to the last set of migrations that were created.
- **Generates migration files:** If it finds any differences (e.g., you added a new field, changed a field's type, created a new model, or deleted a model), it creates new Python files in your app's `migrations` directory.
- **Records database schema changes:** These generated migration files are essentially blueprints or instructions in Python code that describe how to modify the database schema to match your current models. They contain operations like `migrations.AddField`, `migrations.CreateModel`, `migrations.AlterField`, etc.
- **Analogous to a "commit" in version control:** You can think of `makemigrations` as staging your database changes. It records what needs to be done, but doesn't actually do it yet.

**When to use it:**

- **Every time you make a change to your `models.py` file(s).** This includes:
  - Adding new models.
  - Adding new fields to existing models.
  - Changing the type or attributes of existing fields (e.g., `max_length`, `null`, `unique`, `default`).

- Deleting models or fields.
  - Renaming models or fields (Django usually detects this as an add and delete, requiring manual intervention or using `RenameField`).
- Before committing your code to version control (Git, etc.), you should almost always run `makemigrations` and commit the generated migration files along with your model changes.

### Example:

Bash

```
python manage.py makemigrations myapp
```

(This would generate migration files for your 'myapp' application)

```
python manage.py migrate
```

### What it does:

- **Applies pending migrations:** `migrate` reads all the migration files that haven't been applied to your database yet. Django keeps track of applied migrations in a special table in your database called `django_migrations`.
- **Executes database schema changes:** It then executes the SQL commands corresponding to these migration operations against your actual database (e.g., creating tables, adding columns, altering types, etc.).
- **Synchronizes models with the database:** Its ultimate goal is to bring your database schema into sync with the state defined by your migration files.
- **Also applies migrations for built-in Django apps:** When you run `migrate` for the first time on a new project, it applies all the necessary migrations for Django's built-in apps like `auth`, `admin`, `sessions`, etc.

### When to use it:

- **After running `makemigrations`** and generating new migration files.
- **When you deploy your Django project** to a new environment (staging, production) to apply any new migrations from your codebase.
- **When you pull code from a version control repository** and other developers have added new migrations. Django will often tell you if you have unapplied migrations when you start your server.
- **When setting up a new development environment** for an existing project.

### Example:

Bash

```
python manage.py migrate
```

(This would apply all unapplied migrations for all apps)

Bash

```
python manage.py migrate myapp
```

(This would apply unapplied migrations only for your 'myapp' application)

Summary of Differences:

Feature	<code>makemigrations</code>	<code>migrate</code>
Purpose	Creates migration <i>files</i> (blueprints)	Applies those migration <i>files</i> to the database
Input	Changes to <code>models.py</code>	Existing migration files
Output	New <code>.py</code> files in the <code>migrations</code> directory	Modifications to your actual database schema (tables, columns, etc.)
Database Impact	No direct database changes	Modifies the database
Analogy	"Staging" changes / "Committing" database changes	"Applying" changes / "Pushing" changes to the database
Frequency	Run whenever models change	Run after <code>makemigrations</code> or when new migrations are available

Export to Sheets

In essence, `makemigrations` is the **preparatory step** that creates the instructions, and `migrate` is the **execution step** that carries out those instructions on your database. You almost always run `makemigrations` first, then `migrate`.



Gemini can make mistakes, so double-check it