• What are common field types in Django models? (CharField, IntegerField, DateTimeField, etc.)

Here are **common field types** used in **Django models**, along with brief explanations:

1. **CharField**
   * Stores short text or strings.
   * Requires a max\_length parameter.
   * Example: name = models.CharField(max\_length=100)
2. **TextField**
   * Stores long text.
   * No max\_length required.
   * Example: description = models.TextField()
3. **IntegerField**
   * Stores whole numbers.
   * Example: age = models.IntegerField()
4. **FloatField**
   * Stores floating-point numbers.
   * Example: rating = models.FloatField()
5. **DecimalField**
   * Stores decimal numbers with fixed precision.
   * Requires max\_digits and decimal\_places.
   * Example: price = models.DecimalField(max\_digits=5, decimal\_places=2)
6. **BooleanField**
   * Stores True or False.
   * Example: is\_active = models.BooleanField(default=True)
7. **DateField**
   * Stores a date (YYYY-MM-DD).
   * Example: birth\_date = models.DateField()
8. **DateTimeField**
   * Stores both date and time (YYYY-MM-DD HH:MM:SS).
   * Example: created\_at = models.DateTimeField(auto\_now\_add=True)
9. **TimeField**
   * Stores only time.
   * Example: start\_time = models.TimeField()
10. **EmailField**
    * Validates and stores an email address.
    * Example: email = models.EmailField()
11. **URLField**
    * Stores a valid URL.
    * Example: website = models.URLField()
12. **SlugField**
    * Stores short labels (typically used in URLs).
    * Example: slug = models.SlugField(unique=True)
13. **FileField**
    * Uploads and stores files.
    * Requires upload\_to parameter.
    * Example: document = models.FileField(upload\_to='docs/')
14. **ImageField**
    * Like FileField, but validates uploaded image files.
    * Example: photo = models.ImageField(upload\_to='images/')
15. **ForeignKey**
    * Creates a many-to-one relationship.
    * Example: author = models.ForeignKey(User, on\_delete=models.CASCADE)
16. **ManyToManyField**
    * Creates a many-to-many relationship.
    * Example: categories = models.ManyToManyField(Category)
17. **OneToOneField**
    * Creates a one-to-one relationship.
    * Example: profile = models.OneToOneField(User, on\_delete=models.CASCADE)