Models.py

1)from django.db import models

:-Models are Python classes that represent database tables, and each attribute of the model class corresponds to a field in the table. By using these models, you can interact with the database in a high-level, object-oriented manner, making it easier to perform database operations.

2)from django.contrib.auth.models import User

:- Inherit from models.Model: To create a model, we define a Python class that inherits from models.Model. This base class provides the essential functionality to represent a database model.

# Create your models here.

3)class Profile(models.Model):

i)username=models.OneToOneField(User,on\_delete=models.CASCADE)

:- This creates a one-to-one relationship between the Profile model and the built-in User model. It means that each Profile instance is associated with exactly one User instance, and vice versa. The on\_delete=models.CASCADE parameter indicates that if the referenced User is deleted, then the associated Profile should also be deleted.

ii)address=models.CharField(max\_length=100)

:- This defines a character field named address with a maximum length of 100 characters.

iii) profile\_pic=models.ImageField()

:- This defines an ImageField named profile\_pic, indicating that it will store an image file.

Admin.py

1)from django.contrib import admin

:- Imports the admin module from django.contrib, which provides the Django admin functionality.

2)from app.models import \*

:-Imports all models from the models module within the app package. The \* is a wildcard that imports all names defined in the module.

# Register your models here.

3)admin.site.register(Profile)

:- Registers the Profile model with the Django admin site. This means that the Profile model will be accessible and manageable through the Django admin interface.

Forms.py

1)from django import forms

:- Imports the forms module from Django, which contains classes and utilities for working with HTML forms.

2)from app.models import \*

:- Imports all models from the models module within the app package. The \* is a wildcard that imports all names defined in the module.

3)class UserMF(forms.ModelForm):

:- indicating that they are Django ModelForms. ModelForms are a high-level abstraction that allows us to create HTML forms based on Django models with minimal code.

1)class Meta:

:-block inside each ModelForm is used to provide metadata about the form, specifically how it should be constructed based on the associated model.

i)model=User

:-Specifies that the UserMF form is based on the User model.

ii)fields=['username','email','password']

:- Specifies the fields from the User model that should be included in the form.

iii)widges={'password':forms.PasswordInput}

:- Specifies that the widget for the password field should be a forms.PasswordInput, which is used to render password input fields in a secure manner.

iv)help\_texts={'username':''}

:- Provides an empty help text for the username field. This text can be displayed alongside the form field to provide additional guidance to the user.

4)class ProfileMF(forms.ModelForm):

:- indicating that they are Django ModelForms. ModelForms are a high-level abstraction that allows us to create HTML forms based on Django models with minimal code.

5) class Meta:

:- :-block inside each ModelForm is used to provide metadata about the form, specifically how it should be constructed based on the associated model.

i)model=Profile

:- Specifies that the ProfileMF form is based on the Profile model.

ii)fields=['address','profile\_pic']

:- Specifies the fields from the Profile model that should be included in the form.