**1-create folder for project**

**2-select folder in CMD**

C:\Users\kuwar>cd/my django projects

C:\my django projects>cd myproject

C:\my django projects\myproject>code .

C:\my django projects\myproject>pip install virtualenv

C:\my django projects\myproject>virtualenv env

**3-activate env**

C:\my django projects\myproject>.\env\Scripts\activate

(env) C:\my django projects\myproject>

**4-install django**

(env) C:\my django projects\myproject>pip install django

Installing collected packages: tzdata, sqlparse, asgiref, django

Successfully installed asgiref-3.6.0 django-4.1.7 sqlparse-0.4.3 tzdata-2022.7

**5-create project**

(env) C:\my django projects\myproject>django-admin startproject core .

**6-create app**

(env) C:\my django projects\myproject>python manage.py startapp accounts

**7-go direct project folde**

(env) C:\my django projects\myproject>code .

**1-project settings.py**

from pathlib import Path

import os #stating in settings.py file

# SECURITY WARNING: don't run with debug turned on in production!

DEBUG = True

ALLOWED\_HOSTS = []

AUTH\_USER\_MODEL = 'accounts.Account' #its compulsory register here

INSTALLED\_APPS = [

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'accounts',

]

**2-template set in settings.py**

TEMPLATES = [

    {

        'BACKEND': 'django.template.backends.django.DjangoTemplates',

        'DIRS': [os.path.join(BASE\_DIR,'templates')],

        'APP\_DIRS': True,

        'OPTIONS': {

            'context\_processors': [

                'django.template.context\_processors.debug',

                'django.template.context\_processors.request',

                'django.contrib.auth.context\_processors.auth',

                'django.contrib.messages.context\_processors.messages',

            ],

        },

    },

]

**2.2. Create folder in project, templates >>>create index.html file**

{% extends 'base.html' %}

{% load static %}

{% block content %}

<h1> index </h1>

{% endblock %}

**3- set the time zone in settings.py**

TIME\_ZONE = 'Asia/Kolkata'

**4-**

STATIC\_URL = 'static/'

STATIC\_ROOT = 'staticfiles'

STATICFILES\_DIRS = [os.path.join(BASE\_DIR,'static')]

MEDIA\_URL = '/media/'

MEDIA\_ROOT = [os.path.join(BASE\_DIR,'media')]

**4.2. create folder in project, static >>> assets >>> css >>> style.css**

**5-project’s urls.py**

from django.contrib import admin

from django.urls import path, include

urlpatterns = [

    path('admin/', admin.site.urls),

    path('', include('accounts.urls')),

]

**6-create file in app(accounts) for urls.py**

from django.urls import path, include

from accounts import views

urlpatterns = [

    path('', views.index, name='index'),

    path('contact/', views.contact, name='contact'),

    path('base/', views.base, name='base')

]

**7-admin.py file register**

from django.contrib import admin

from accounts.models import Account

#Register your models here.

admin.site.register(Account)

**8-in accounts views.py file.**

from django.shortcuts import render

# Create your views here.

def index(request):

    return render(request, 'index.html')

def contact(request):

    return render(request, 'contact.html')

def base(request):

    return render(request, 'base.html')

**9-accounts models.py file code here.**

from django.db import models

from django.contrib.auth.models import AbstractBaseUser,BaseUserManager

# Create your models here.

class MyAccountManager(BaseUserManager):

    def create\_user(self,email,username, password = None):

        if not email:

            raise ValueError("user must have an email")

        if not username:

            raise ValueError("user must have user name")

        user = self.model(

            email = self.normalize\_email(email),

            username = username,

        )

        user.set\_password(password)

        user.save(using=self.\_db)

        return user

    def create\_superuser(self,email,username,password = None):

        user = self.create\_user(email,username,password)

        user.is\_staff = True

        user.is\_superuser = True

        user.save(using = self.\_db)

class Account(AbstractBaseUser):

    name = models.CharField(max\_length=25)

    email = models.EmailField(unique=True)

    username = models.CharField(max\_length=20,unique = True)

    mobile = models.CharField(max\_length= 12)

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

    pincode = models.CharField(max\_length=10)

    date\_join = models.DateTimeField(auto\_now\_add= True)

    last\_login = models.DateTimeField(auto\_now\_add= True)

    is\_active = models.BooleanField(default=True)

    is\_staff = models.BooleanField(default=False)

    is\_superuser = models.BooleanField(default=False)

    objects = MyAccountManager()

    USERNAME\_FIELD = 'email'

    REQUIRED\_FIELDS = ['username']

    def has\_perm(self,perm,obj= None):

        return self.is\_staff

    def has\_module\_perms(self,app\_label):

        return True

**10- create or Update Models.**

(env) C:\my django projects\myproject>python manage.py makemigrations

(env) C:\my django projects\myproject>python manage.py migrate

**11-Create superuser or admin.**

python manage.py createsuperuser

Email: admin@gmail.com

Username: admin

Password:

Password (again):

The password is too similar to the username.

This password is too short. It must contain at least 8 characters.

This password is too common.

Bypass password validation and create user anyway? [y/N]: y

Superuser created successfully.