Models

Go to Models.py and create a class as Below

**from django.db import models  
class RegisterdUser(models.Model):  
 name = models.CharField(max\_length=100)  
 emailid = models.CharField(max\_length=100)  
 phonenumber = models.CharField(blank=True,null=True,max\_length=13)  
 password = models.CharField(max\_length=30)**

Register this Class in admin.py

**from django.contrib import admin  
from .models import RegisterdUser,CreateTestsuite  
  
admin.site.register(RegisterdUser)**

Go to forms.py create a class with that model

**from django.forms import ModelForm  
from django import forms  
from .models import RegisterdUser,CreateTestsuite  
  
class RegisterForm(ModelForm):  
 password = forms.CharField(widget=forms.PasswordInput)  
  
 class Meta:  
 model = RegisterdUser  
 fields = '\_\_all\_\_'  
 labels = {  
 "name":"Username",  
 "emailid":"Email Id",  
 "phonenumber":"Phone Number",  
 "password":"password"  
 }**

in order to View the html we should use as forms.as\_p

**{% extends "base.html" %}  
  
<body>  
 {% block content %}  
 <form action="" method="post">  
 {% csrf\_token %}  
 {{form.as\_p}}  
 <input type="submit" value="Submit">  
 </form>  
 {% endblock %}  
</body>**

Once Registered user class in Go to Views.py and add the redirect because once sigin is done it should redirect

Import forms in Views.py

from .forms import RegisterdUser

from django.contrib import messages

We need to create a function called Register it should be accessed via post method

It will have two method one for creation one for Getting the data

**python manage.py sqlmigrate IIS\_APP 0001** to Create table in admin

Argument to the redirect is an URL which is in strings

**def registartion(request):**

**if request.method == "POST":**

**form = RegisterForm(request.POST)**

**if form.is\_valid():**

**form.save()**

**messages.success(request,"Account created successful")**

**return redirect("signin")**

**else:**

**form = RegisterForm()**

**user\_info = {'form':form}**

**return render(request,"register.html",user\_info)**

we can get data from registered users using the below commands

python manage.py shell

**>>from groceries\_app.models import RegisteredUser**

**>>RegisteredUser.objects.all()**

we can single data using get command

**RegisteredUser.objects.get(id=1)**

we can get inside data also

**RegisteredUser.objects.get(id=1).id**

In order to get the latest created object we use

**>>> RegisteredUser.objects.last()**

we use filter in oreder to filter out the data

**>>> RegisteredUser.objects.filter(id=2).first()**

we can create users through shell

**>>> post\_data = RegisteredUser(name="alex",emailid="alex@gmail.com",phoneNum="9229292",password="alex")**

**>>> post\_data.save() --> we can create user**

we can interact with the created users by using Views functuion we have to create class with inheratince of List View we need to write three steps

1. which model we need to import

2.Template name that is HTML

**ListView**

3.how much data need to be accessed

views.py

**from django.views.generic import ListView**

**class UserListView(ListView):  
 model = RegisteredUser  
 template\_name = "user\_data.html"  
 context\_object\_name = "alldata"**

the template should like this

**{% extends "base.html" %}  
<body>  
{% block content %}  
{% for user in alldata %}  
 Id:{{user.id}}<br>  
 Username:{{user.name}}<br>  
 Email ID:{{user.emailid}}<br><br>  
 {% endfor %}  
{% endblock %}  
</body>**

we need to add path min Urls.py

urls.py

**path('userList',views.UserListView.as\_view(), name="userList")**

**>>Createjob.objects.all()[0].jobname**

**DetailView:**

List view is used to diplay data of all instances but Detail View is of indivual instances

Views.py

**class UserDetailView(DetailView):  
 model = RegisteredUser**

**HTML**

**{% extends "base.html" %}  
<body>  
{% block content %}  
Id : {{object.id}}<br>  
UserName: {{object.name}}<br>  
Email ID: {{object.emailid}}<br>  
Phone Number: {{object.phoneNum}}<br><br>  
{% endblock %}  
</body>**

Add the path asa below here pk represents Primary key

**path('userdetail/<int:pk>',views.UserDetailView.as\_view(template\_name="user\_details.html"),name='userdetail')**

**In oreder to Access we should use the uRL in Chrome Tab we will be getting the first ID Data**

[**http://127.0.0.1:8000/userdetail/1**](http://127.0.0.1:8000/userdetail/1)

**Create View:**

**class UserCreateView(CreateView):  
 model = RegisteredUser  
 form\_class = RegisterForm**

IN HTML

**{% extends "base.html" %}  
<body>  
 {% block content %}  
 <form method="post">  
 {% csrf\_token %}  
 {{form.as\_p}}  
 <input type="submit" value="Submit">  
 </form>  
 {% endblock %}  
</body>**

**URL**

**path('usercreate/', views.UserCreateView.as\_view(template\_name="user\_create.html"), name='usercreate')**

we can redirect to another userdetails by adding get\_absloutefunction in models

models.py

**from django.db import models  
from django.urls import reverse  
  
class RegisteredUser(models.Model):  
 name = models.CharField(max\_length=100)  
 emailid = models.CharField(max\_length=100)  
 phoneNum = models.CharField(blank=True, null=True, max\_length = 20)  
 password = models.CharField(max\_length=30)  
 def get\_absolute\_url(self):  
 return reverse('userdetail',kwargs={'pk':self.pk})**

UpdateView

**Add the Module**

**class UserUpdateView(UpdateView):  
 model = RegisteredUser  
 form\_class = RegisterForm**

**ADD**

**path('userupdate/<int:pk>',views.UserUpdateView.as\_view(template\_name="user\_create.html"),name='userupdate')**

**redirect to URL:http://127.0.0.1:8000/userupdate/3**

**Authorization**

in order to not give direct access to the users we use

**Views.py**

**from django.contrib.auth.mixins import UserPassesTestMixin**

**class UserUpdateView(UserPassesTestMixin,UpdateView):  
 model = RegisteredUser  
 form\_class = RegisterForm  
 def test\_func(self):  
 if self.request.user.is\_active:  
 return True  
 else:  
 return False**

**user\_create.html**

**{% extends "base.html" %}  
  
<body>  
 {% block content %}  
 <form method="post">  
 {% csrf\_token %}  
 {% if request.user.is\_superuser %}  
 {{form.as\_p}}  
 <button type="submit">Submit data</button>  
 {% else %}  
 the logged user is not a super user  
 {% endif %}  
 </form>  
 {% endblock %}  
</body>**

**Adding Images to Django**

we can upload images to Django first we need to install pillow :**pip install pillow**

**then go to models.py** add the profile code

**class RegisteredUser(models.Model):  
 name = models.CharField(max\_length=100)  
 emailid = models.CharField(max\_length=100)  
 phoneNum = models.CharField(blank=True, null=True, max\_length = 20)  
 password = models.CharField(max\_length=30)  
 profile\_pic = models.ImageField(upload\_to='profile\_pic',default="default.jpg")**

we need to configure image settings in settings py

**import os**

**MEDIA\_ROOT = os.path.join(BASE\_DIR,'media')  
MEDIA\_URL = '/media/'**

we need to add confiragtions in URLS.py

**from django.conf import settings  
from django.conf.urls.static import static**

**if settings.DEBUG:  
 urlpatterns += static(settings.MEDIA\_URL,document\_root=settings.MEDIA\_ROOT)**

**Run the Server**

Getting the uploading image which was in Regesttered user present on models

**def loggedin(request):**

**global usrnme  
 image\_file = RegisteredUser.objects.get(username=usrnme)  
 full\_pic\_path = str(image\_file.profile\_pic)  
 userdetails = {'username': usrnme,'image':full\_pic\_path}  
 return render(request, "loggedin.html", userdetails)**

**Linking Multiple APPS to single project**

**python manage.py startapp products\_app** -> it will Create an app

Add this product app in settings.py on installed APPS  
**INSTALLED\_APPS = [  
 'django.contrib.admin',  
 'django.contrib.auth',  
 'django.contrib.contenttypes',  
 'django.contrib.sessions',  
 'django.contrib.messages',  
 'django.contrib.staticfiles',  
 'groceries\_app',  
 'products\_app'  
]**

add a code in views.py

products\_app/views.py

**from django.shortcuts import render,redirect  
  
def products(request):  
 products = {"dairy":{"milk","yogurt","cheese"},  
 "beverages":{"soda","juice"},  
 "vegtables":{"ginger","lattuce"},  
 "fruits":{"banana","apples"}}  
 return(request,"products\_list.html",products**)

Add Html FILE in products app

**{% extends "base.html" %}  
<style>  
 .column {  
 float: left;  
 width: 33.33%;  
 padding: 10px;  
 height: 300px;  
 }  
  
 .row:after {  
 content: "";  
 display: table;  
 clear: both;  
 }  
</style>  
<body>  
 {% block content %}  
 <form method="get" action="logout">  
 <button type="submit" style="float: right;width: 10%; height: 30px;font-size : 15px">Logout</button>  
 </form>  
 <div class="row">  
 <div class="column">  
 <h2>Dairy</h2>  
 {% for dairy\_prod in dairy %}  
 <input type="checkbox" id={{dairy\_prod}} name="products" value={{dairy\_prod}}>  
 <label for={{dairy\_prod}}>{{dairy\_prod}}</label><br>  
 {% endfor %}  
 </div>  
 <div class="column">  
 <h2>Beverages</h2>  
 {% for beverage\_prod in beverages %}  
 <input type="checkbox" id={{beverage\_prod}} name="products" value={{beverage\_prod}}>  
 <label for={{beverage\_prod}}>{{beverage\_prod}}</label><br>  
 {% endfor %}  
 </div>  
 <div class="column">  
 <h2>Vegetables</h2>  
 {% for veg\_prod in vegetables %}  
 <input type="checkbox" id={{veg\_prod}} name="products" value={{veg\_prod}}>  
 <label for={{veg\_prod}}>{{veg\_prod}}</label><br>  
 {% endfor %}  
 </div>  
 <div class="column">  
 <h2>Fruits</h2>  
 {% for fruit\_prod in fruits %}  
 <input type="checkbox" id={{fruit\_prod}} name="products" value={{fruit\_prod}}>  
 <label for={{fruit\_prod}}>{{fruit\_prod}}</label><br>  
 {% endfor %}  
 </div>  
 </div>  
  
 {% endblock %}  
</body>**

Go to login.html where in grociries app and add the procusts list.html in that

**grociriecs\_app/Loggin.html**

**<img src="{% static 'homepage.jpeg' %}" width="1050" height="700" style="text-align:center;">**

**<div class="container">**

**<p></p>**

**<form method="get" action="products">**

**<button type="submit">View all products</button>**

**</form>**

**</div>**

add the URL in urls.py in grociries app

**from products\_app import views as prod\_views**

**path('products', prod\_views.product\_list, name='products\_list'),**

**Accepting and storing information**

in this app first we will add the code were it will accept the order and display as message

**models.py**

**from django.db import models  
  
# Create your models here.  
  
class OrderList(models.Model):  
 username = models.CharField(max\_length=200)  
 wholeList = models.CharField(max\_length=500,blank=True,null=True)**

we have to register the app

admin.py

**from django.contrib import admin**

**from .models import OrderList**

**admin.site.register(OrderList**)

**add the code in views.py where it can store in models based on check box inputs and grociecs\_app username so that both are integrated rather than usin forms.as\_p**

**def order(request):  
 if request.method =="POST":  
 prod\_list = request.POST.getlist('products')  
 prod\_str = ",".join(prod\_list)  
 order\_data = OrderList(wholelist=prod\_str,username=grocires.usrnme)  
 order\_data.save()  
 messages.success(request,"order created successfully"+prod\_str)  
 return redirect("loggedin")  
 else:  
 return render(request,"products\_list.html")**

add the path in grocires/url.py

**path('order',prod\_views.order,name='order'),**

now we need to add form

**<form method="POST" action="order">**

first Views.py search for method if the method is post it will get the list here action represents in which function it submitted data should go

**python manage.py makemigrations**

since we updated the models and admin

**python manage.py migrate**

**Sending EMAILS:**

Add settings to the in seeting.py in My fristproject app

**# Mail**

**EMAIL\_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'**

**EMAIL\_HOST = 'smtp.gmail.com'**

**EMAIL\_USE\_TLS = True**

**EMAIL\_PORT = 587**

**EMAIL\_HOST\_USER = "bob@loonycorn.com"**

**EMAIL\_HOST\_PASSWORD = "dumber\_password"**

**REST API using Django**

first we need to create a project and install install djangorestframework

django-admin startproject djangoRESTAPI\_project

python manage.py startapp actors\_app

Add the actors\_app in settings.py in installed app along restapi framework

**INSTALLED\_APPS = [  
 'django.contrib.admin',  
 'django.contrib.auth',  
 'django.contrib.contenttypes',  
 'django.contrib.sessions',  
 'django.contrib.messages',  
 'django.contrib.staticfiles',  
 'rest\_framework',  
 'actors\_app'  
]**

Create a model

**from django.db import models  
  
# Create your models here.  
  
class Actor(models.Model):  
 name = models.CharField(max\_length=100,null=True,blank=True,unique=True)  
 age = models.IntegerField()**

Add the Model in admin.py

from django.contrib import admin  
from .models import Actor  
admin.site.register(Actor)

**python manage.py makemigrations**

**python manage.py migrate**

we then create a Seralizers.py file simaliar like forms which we have the data as below

**from rest\_framework import serializers  
from .models import Actor  
  
class ActorSerializers(serializers.SerializerMetaclass):  
 class Meta:  
 model = Actor  
 fields = '\_\_all\_\_'**

in order to display the Seralized model instances we go to Views.py Seralize makes response as HTTP data

from rest\_framework.views import APIView #------------------- it will allow us multiple HTTP methods   
from .serializers import ActorSerializers #  
from .models import Actor  
from rest\_framework.response import Response

**class ActorsLis(APIView):# API this will enable the get function   
  
 def get(self, request):  
 actors = Actor.objects.all()  
 seralize = ActorSerializers(actors,many=True)  
 return Response(seralize.data)**

create URL.py this will enable to response in HTTP GET

from . import views  
from django.urls import path  
  
urlpatterns = [path('',views.ActorsList.as\_view()),]

Go to Orginal Urls.py which is present and add the path

**from django.urls import path,include  
  
urlpatterns = [  
 path('admin/', admin.site.urls),  
 path('',include('actors\_app.urls'))  
]**