INSTITUTE OF TECHNOLOGY,

NIRMA UNIVERSITY

2CSDE86 Application Development Framework

PRACTICAL 6

Name : Jaimik S. Chauhan

Roll No. : 20BCE503

Semester : 7th

Teacher's Signature :

Aim

Continue with the social media application using the Django platform.

Admin module customization hands-on. After hands-on perform following tasks for the social media application.

Design an admin module with dashboard and grant management privileges like admin can update and delete the content of any user.

Preform CRUD operation and connect database with your social media website. Store all the data and retrieve the data according the user.

1. ...Continue from Practical 5

2. Admin Module Customisation on Users

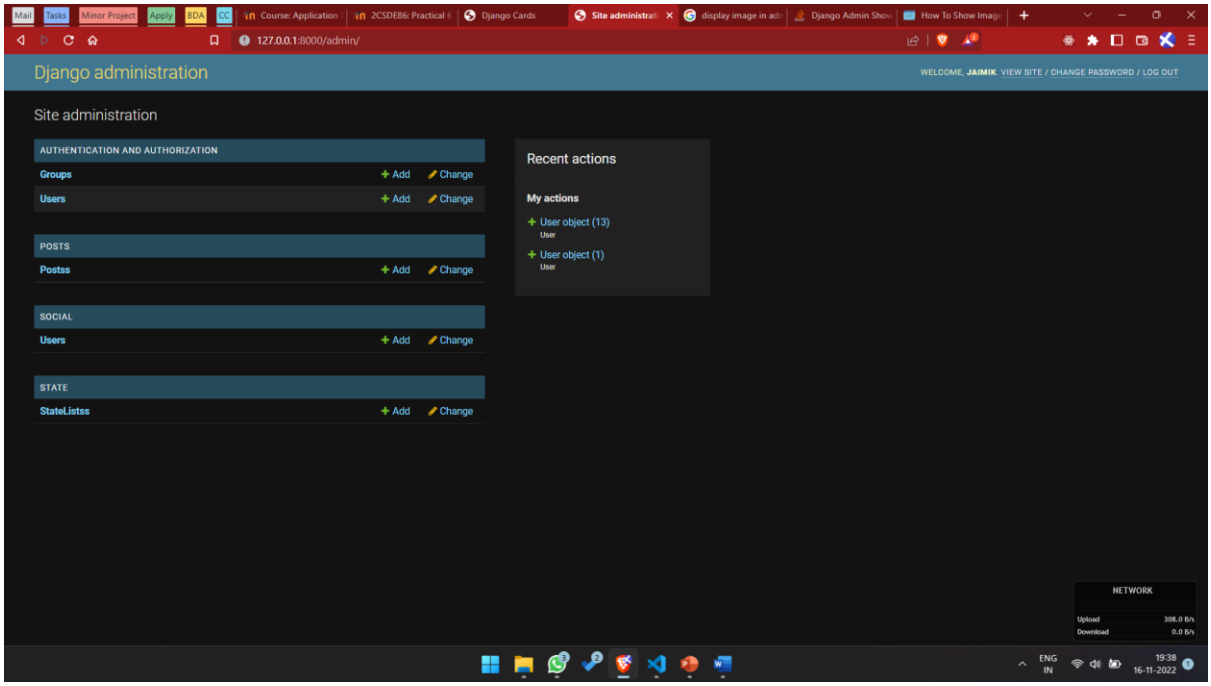
/p3/social/models.py

```
import datetime
from django.db import models

# Create your models here.
class User(models.Model):
    CHOICE = (
        ("Male", "Male"),
        ("Female", "Female")
    )

    name = models.CharField(max_length=250)
    email = models.EmailField()
    password = models.CharField(max_length=18)
    phone = models.CharField(max_length=200)
    username = models.CharField(max_length=20)
    birthdate = models.DateField("Date",default=datetime.date.today())
    gender = models.CharField(choices=CHOICE, max_length=100)
    is_active = models.IntegerField(
        default = 1,
        blank = True,
        null = True,
        help_text='1->active, 0->inactive',
        choices =((1, 'active'), (0,'inactive'))
    )
```

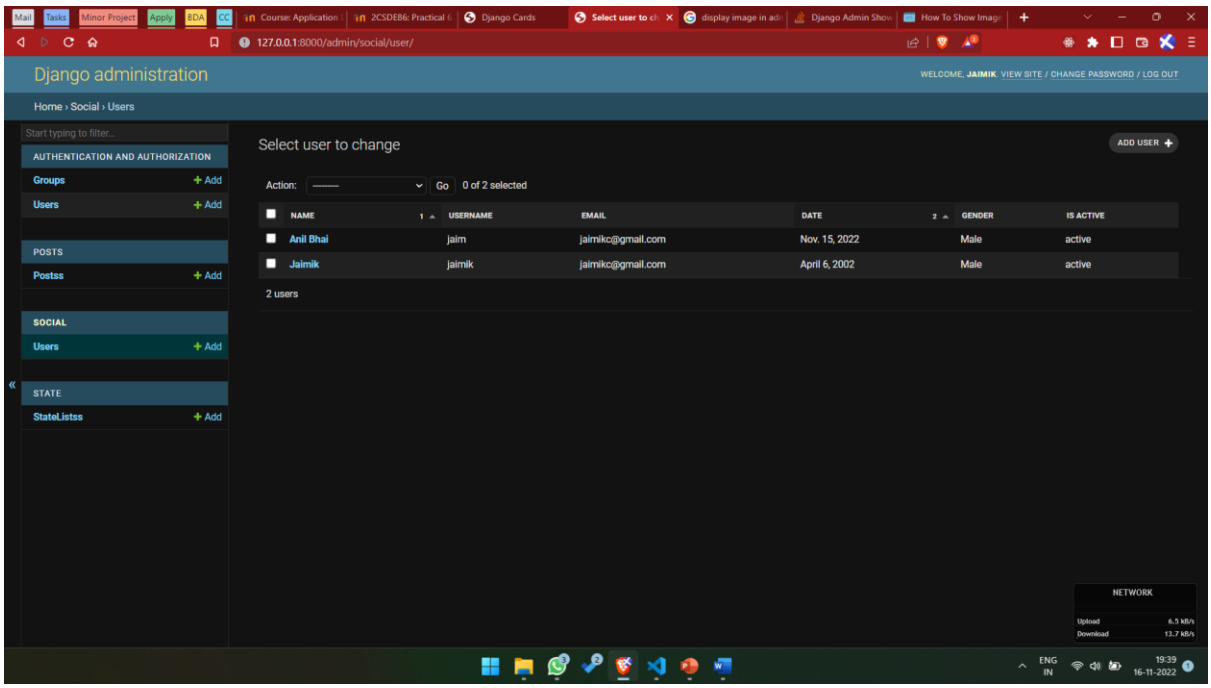
/p3/social/admin.py

```
from django.contrib import messages
from django.contrib import admin

# jaimik : Admin@123
# Register your models here.
from .models import User

class UserAdmin(admin.ModelAdmin):
    list_display = ('name', 'username', 'email', 'birthdate', 'gender',
'is_active')

    ordering = ['name', 'birthdate']

    def active(self, obj):
        return obj.is_active == 1
    active.boolean = True

    def make_active(modeladmin, request, queryset):
        queryset.update(is_active=1)
        messages.success(request, "Selected Record(s) Marked is Active
Succesfully!!!")

    def make_inactive(modeladmin, request, queryset):
        queryset.update(is_active=0)
        messages.success(request, "Selected Record(s) Marked is Inactive
Succesfully!!!")

    admin.site.add_action(make_active,"Make Active")
    admin.site.add_action(make_inactive,"Make Inactive")
    db_table = "New Users"

admin.site.register(User, UserAdmin)
```

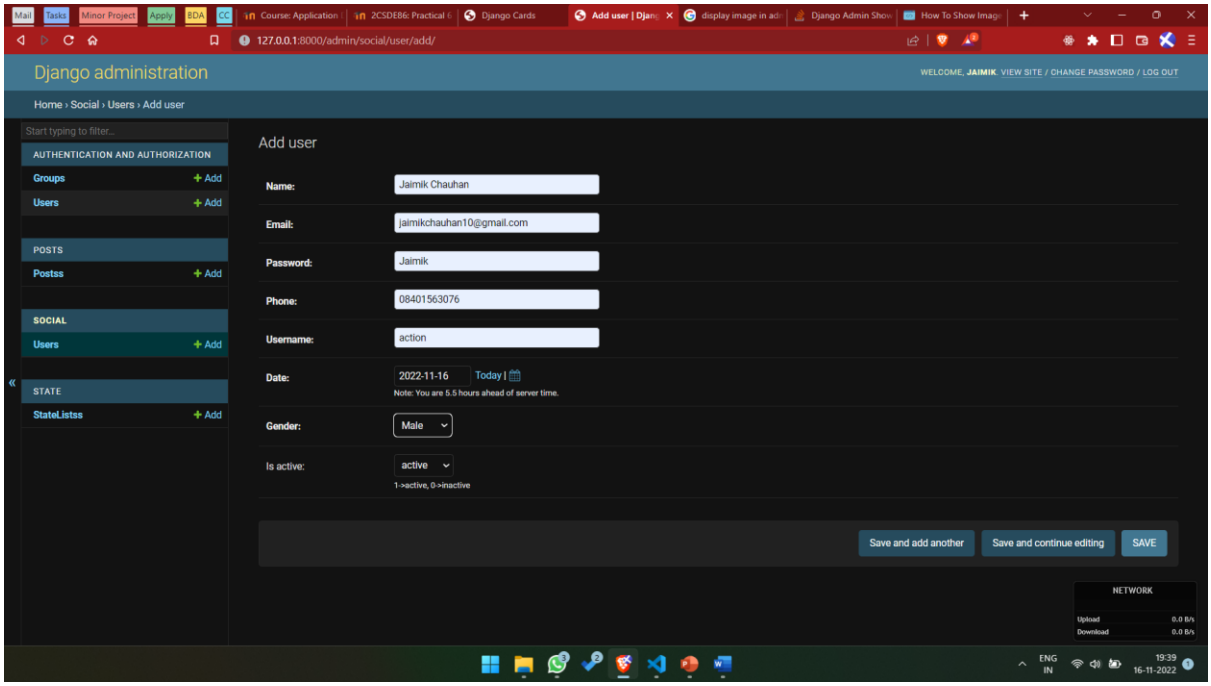
Output :



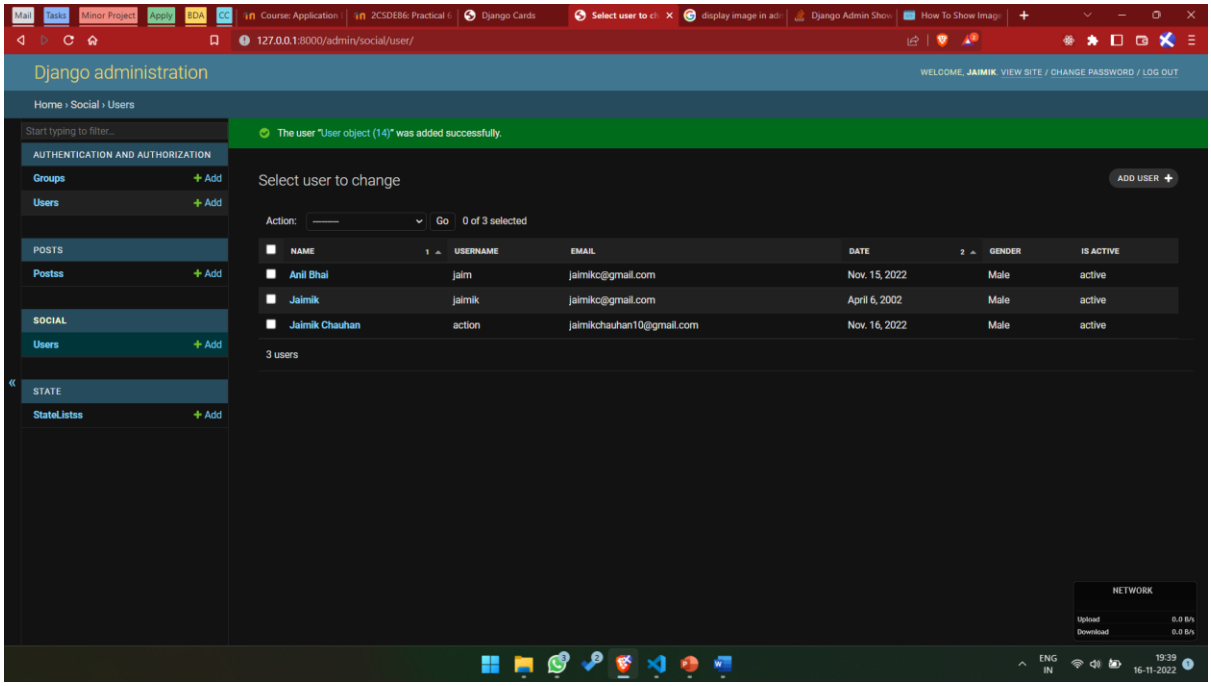
Output 1 Admin Home Page



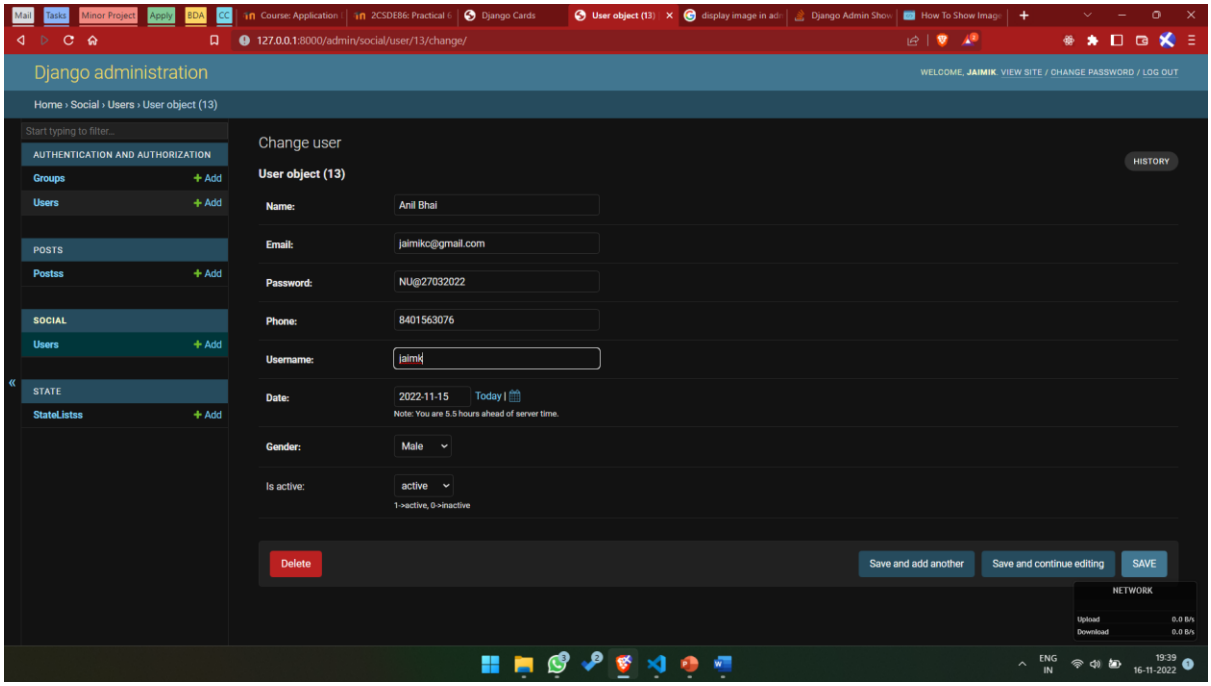
Output 2 List of Users



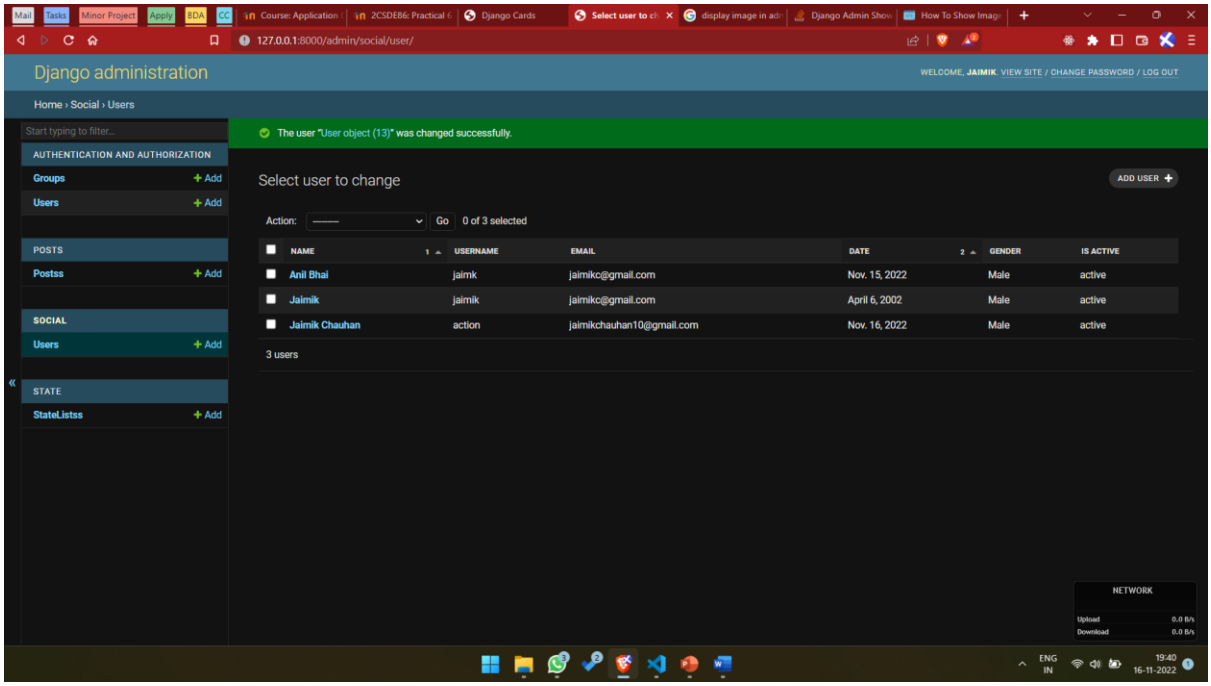
Output 3 Creating a User



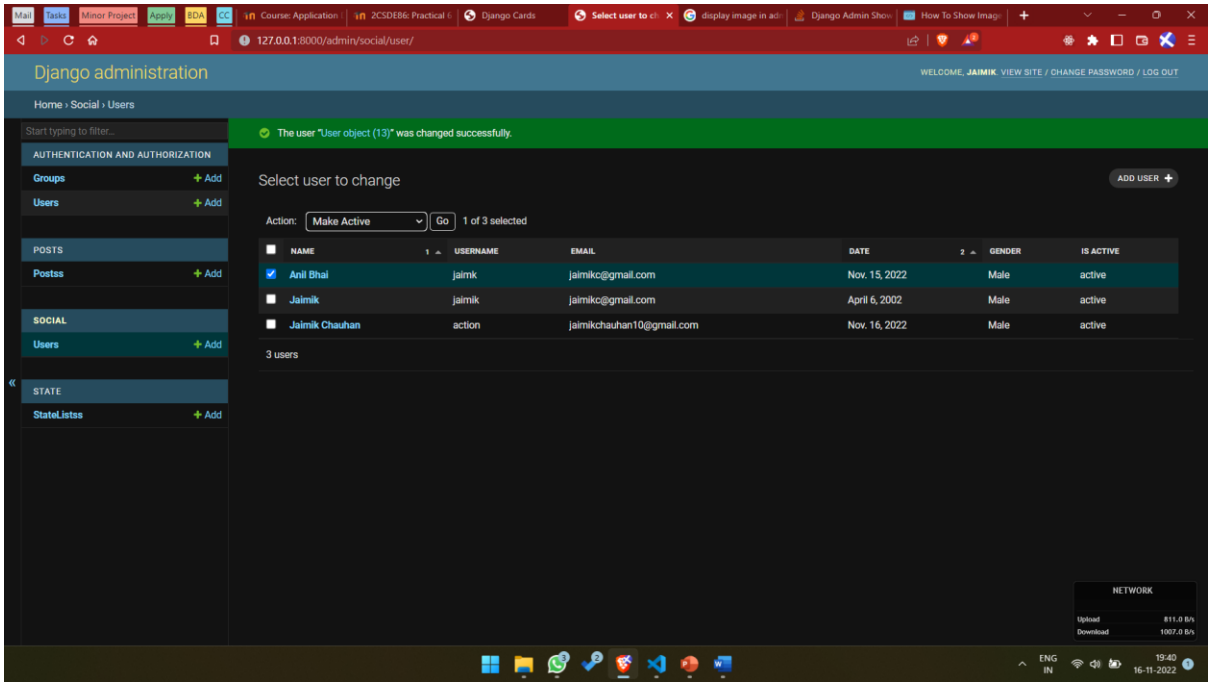
Output 4 After Inserting User



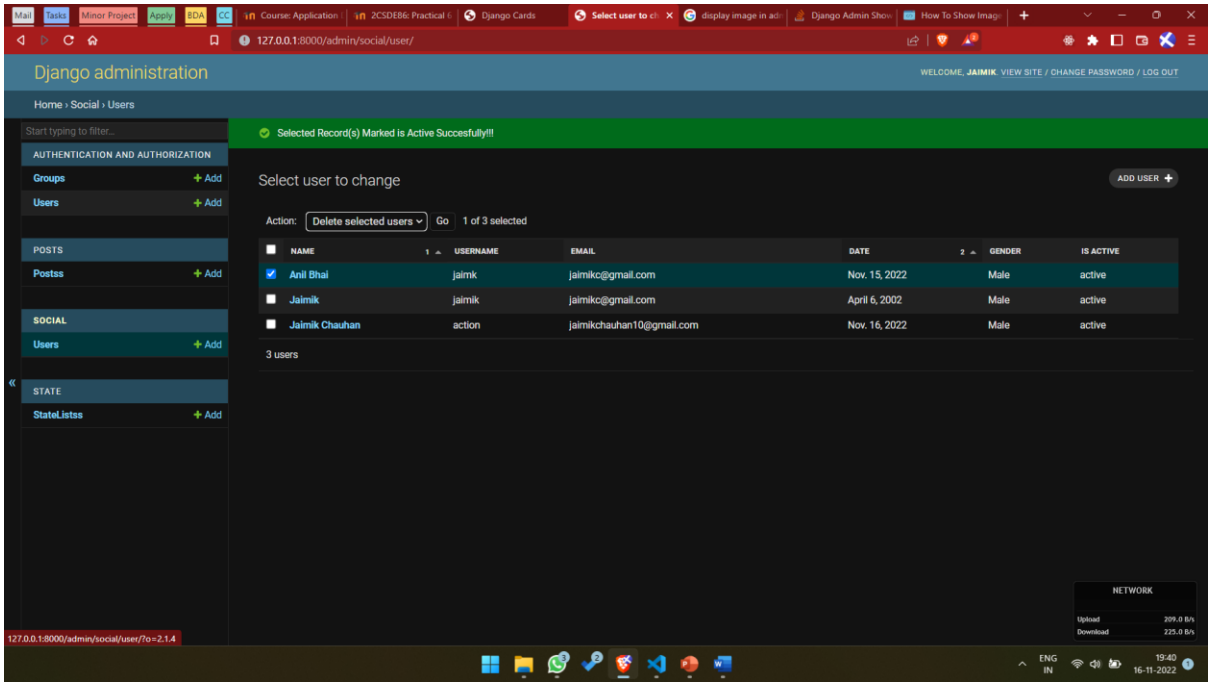
Output 5 Update User Details



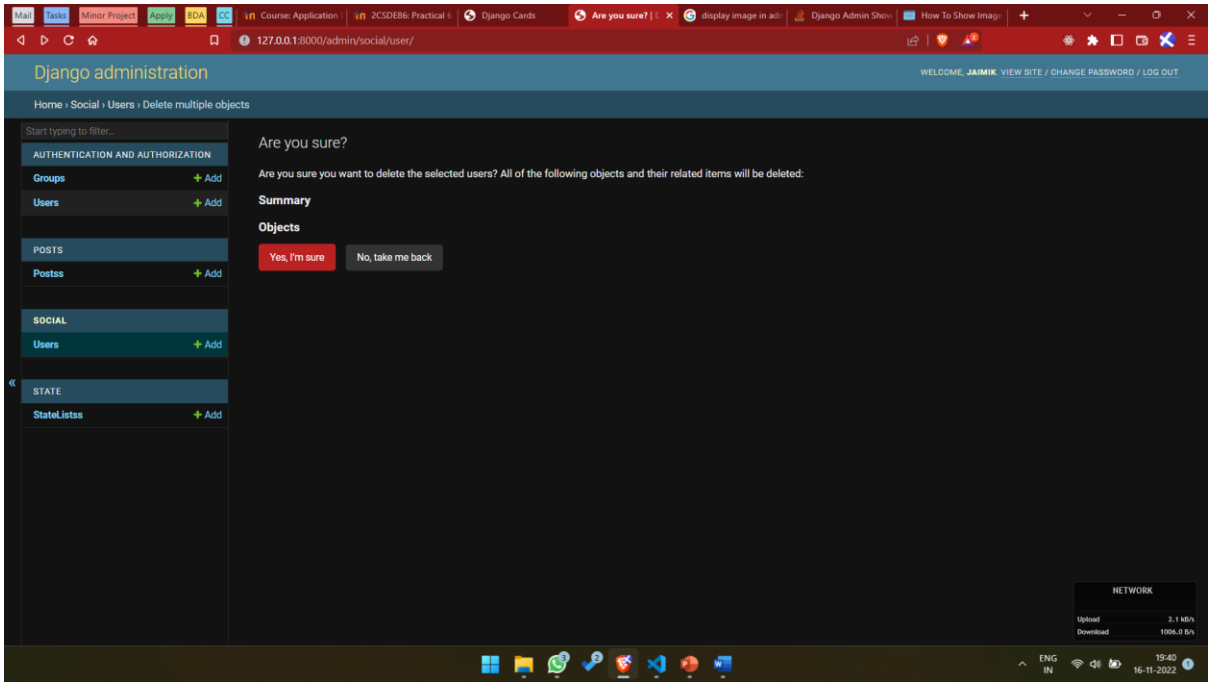
Output 6 User Details Updated



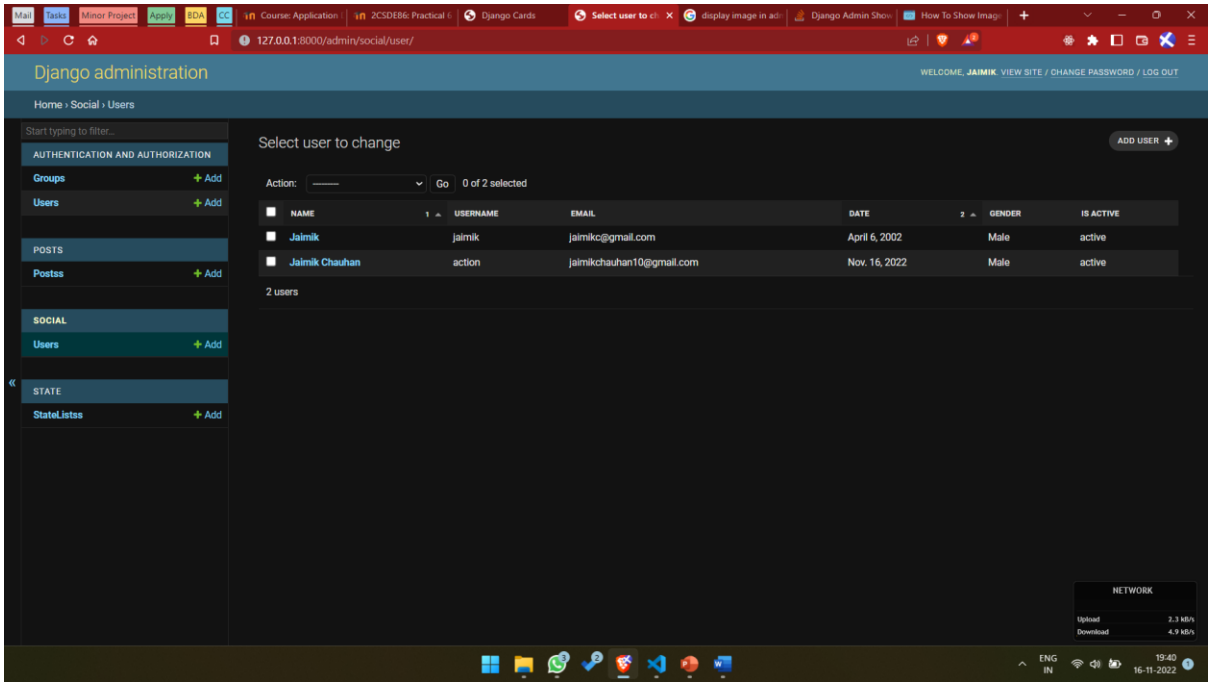
Output 7 Selected Option Make Active User



Output 8 Delete User



Output 9 Confirm Delete User Page



Output 10 Updated List of Users