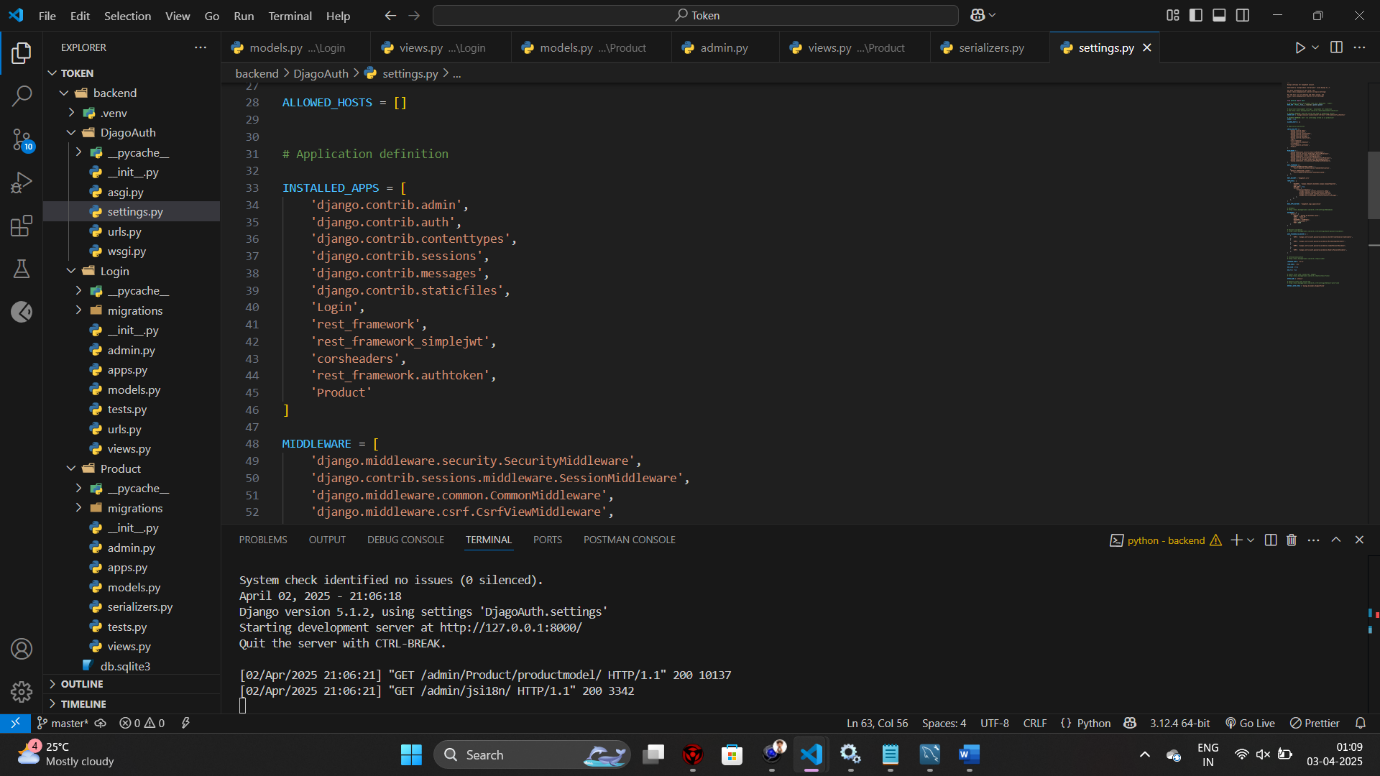
**Token Authentication**

* pip install djangorestframework
* pip install djangorestframework-simplejwt
* pip install django-filter
* pip install django-cors-headers



* **INSTALLED\_APPS**

INSTALLED\_APPS = [

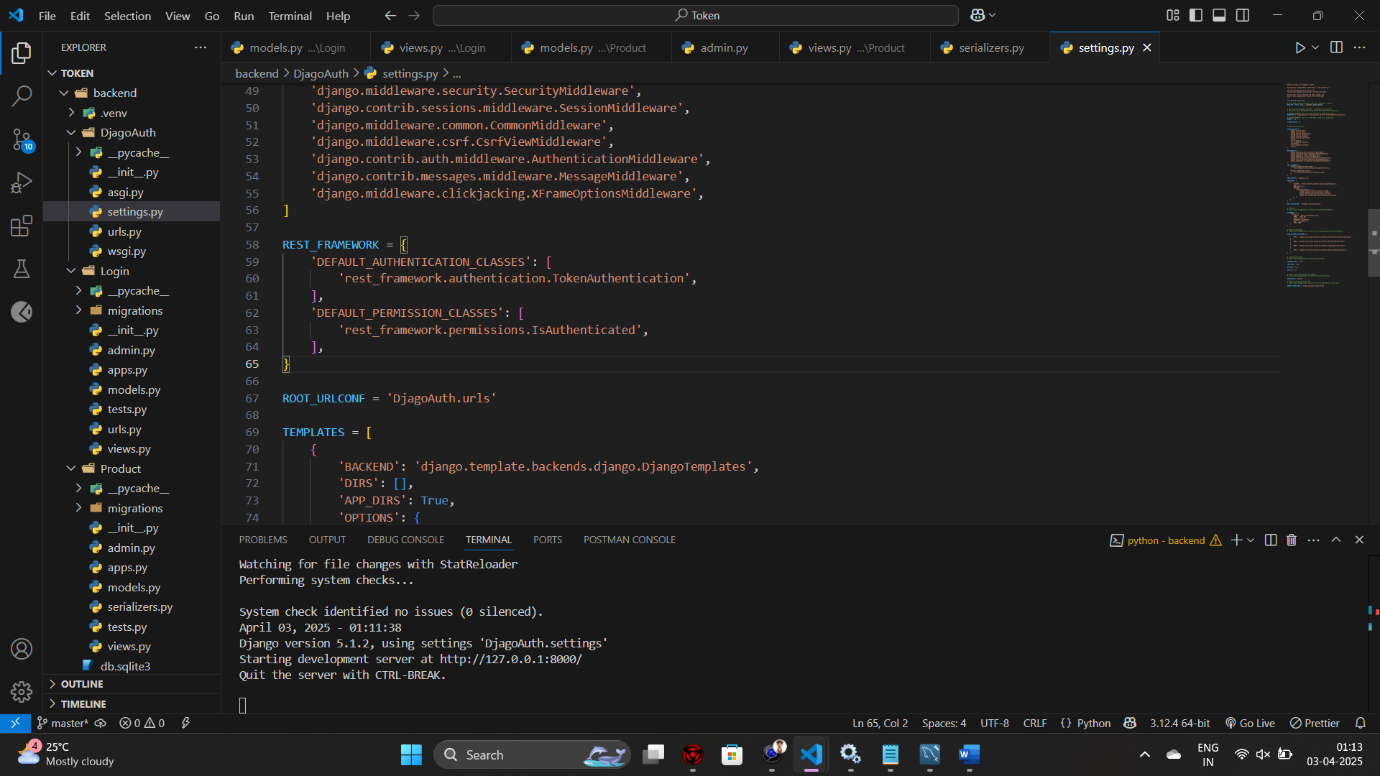
'rest\_framework',

    'rest\_framework\_simplejwt',

    'corsheaders',

    'rest\_framework.authtoken'

]

* python manage.py migrate
* rest\_framework.authentication => BasicAuthentication ,SessionAuthentication ,TokenAuthentication
* rest\_framework.permissions =>IsAuthenticated ,IsAdminUser ,IsAuthenticatedOrReadOnly,AllowAny
* **After the MIDDLEWARE section REST\_FRAMEWORK**

Token authentication is mandatory to use this **middleware**The response is:(Error)  
**Authentication credentials were not provided**

REST\_FRAMEWORK = {

    'DEFAULT\_AUTHENTICATION\_CLASSES': [

        'rest\_framework.authentication.TokenAuthentication',

    ],

    'DEFAULT\_PERMISSION\_CLASSES': [

        'rest\_framework.permissions.IsAuthenticated',

    ],

}

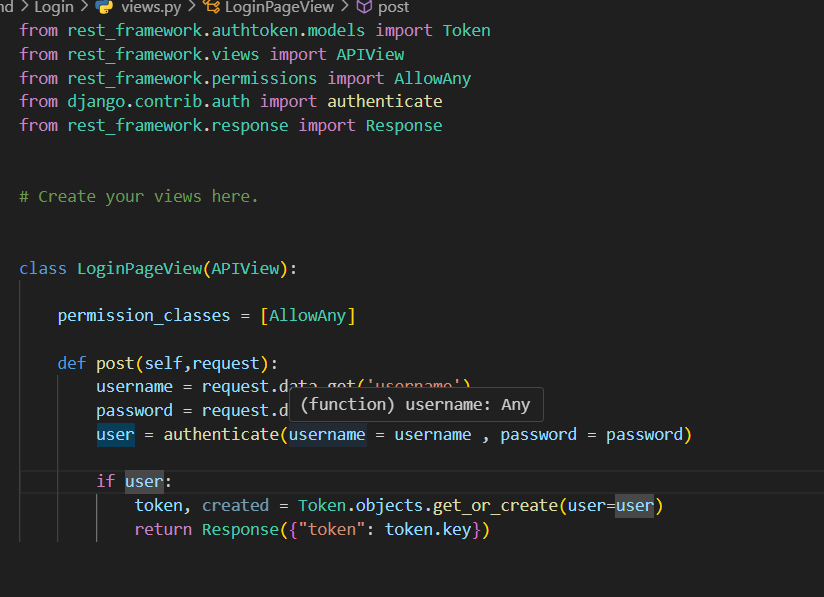
If using the middleware in settings , provide to **permission in class**

* permission\_classes =[AllowAny]

this key term using the all users access details or logic

* **Login page use the** permission\_classes =[AllowAny]
* **Views => APIVview**
* **Permision => AllowAny**
* **Auth => authenticate**
* **Response => response**

The Code is :



class LoginPageView(APIView):

    permission\_classes = [AllowAny]

    def post(self,request):

        username = request.data.get('username')

        password = request.data.get('password')

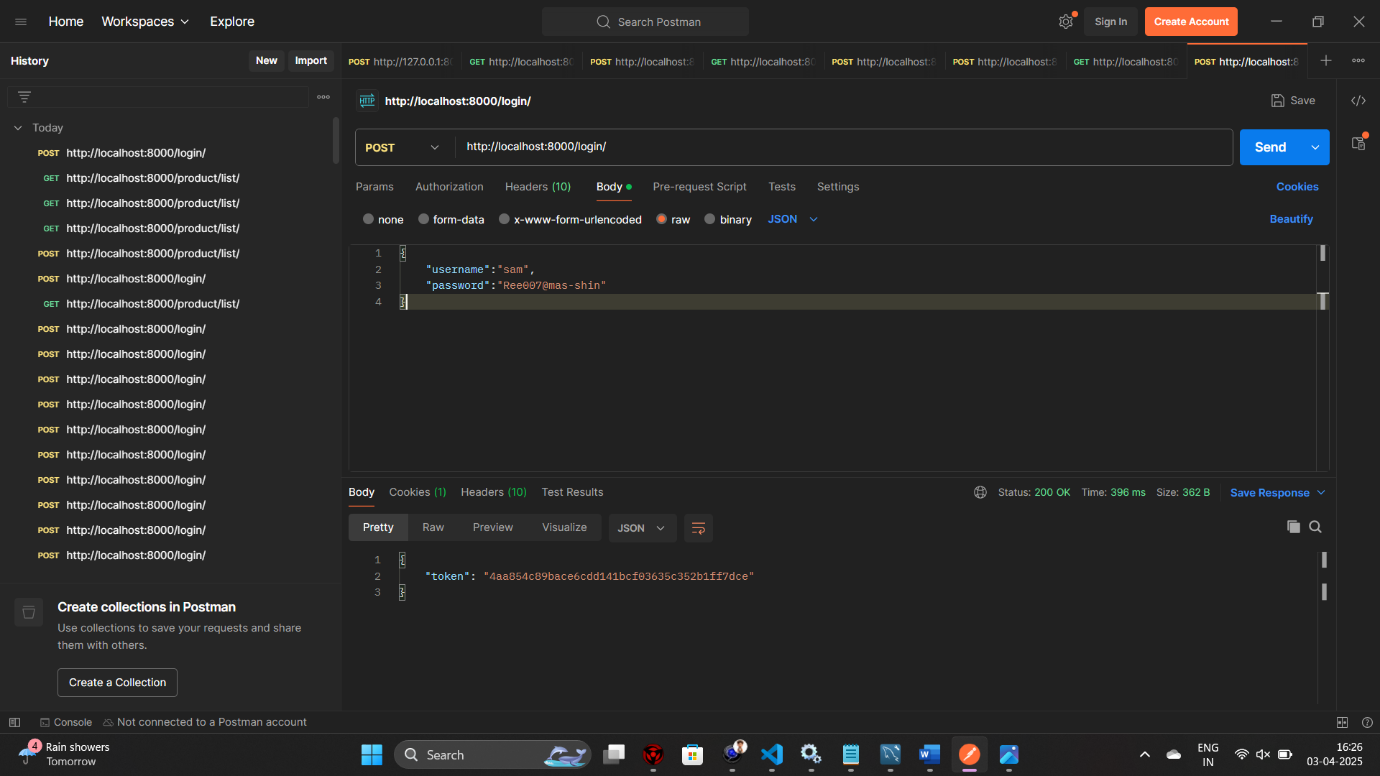
        user = authenticate(username = username , password = password)

        if user:

            token, created = Token.objects.get\_or\_create(user=user)

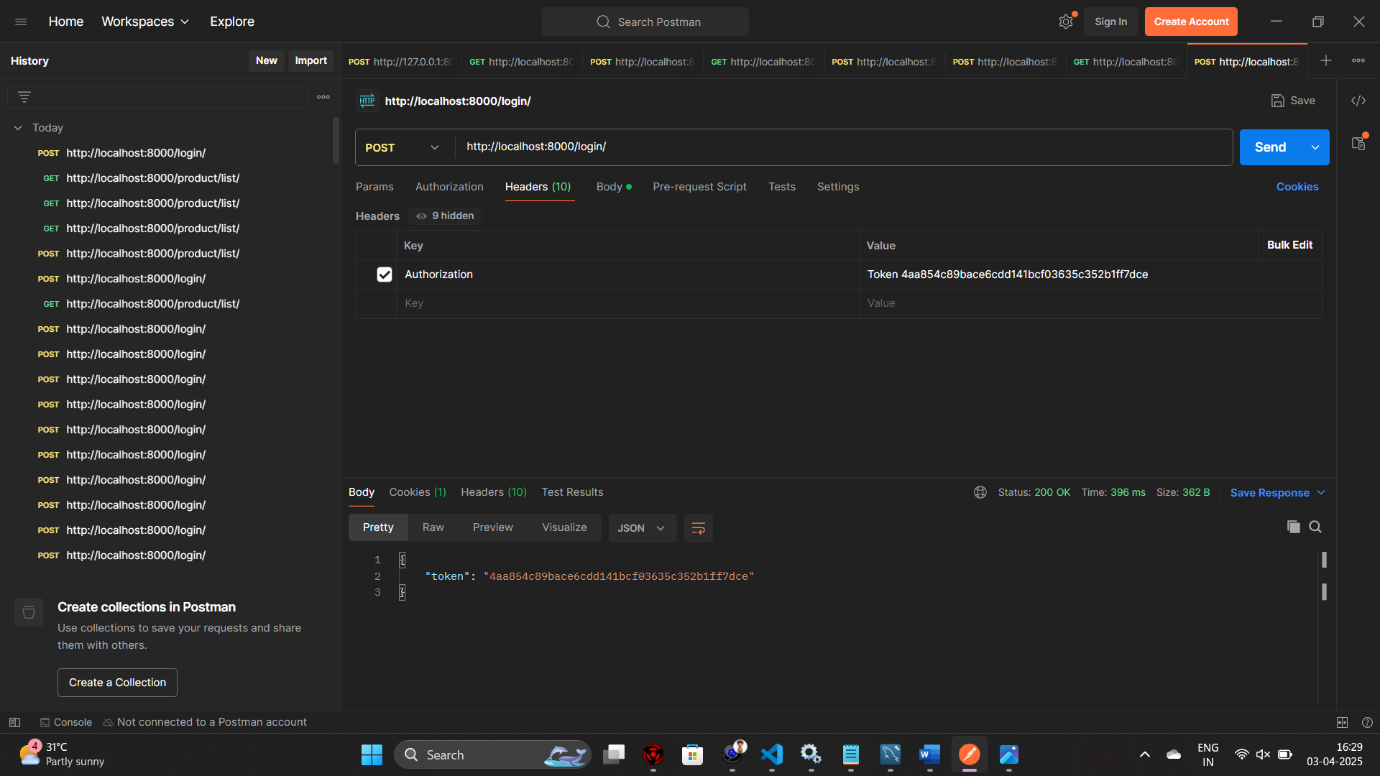
            return Response({"token": token.key})

**Check PostMan to generate Token**

****

**After generate to token check Authentication process**

* Headers to pass the token then use APIs get details

****

For Example ,

Product list show for authentication {This is only after to get token}

**3 way in token authentication:**

* **Generics**

import ListAPIView,CreateAPIView,RetrieveAPIView,UpdateAPIView,DestroyAPIView

* **views** import APIView
* **viewsets** import ModelViewSet

1. **.Using the views:**

from .views import \*

from django.urls import path

urlpatterns = [

    path('list/', ProdutcListView.as\_view()),

]

* **urls**
* **Views**

from django.shortcuts import render

from .serializers import \*

from rest\_framework.views import APIView

from .models import \*

from rest\_framework.response import Response

from rest\_framework.permissions import AllowAny,IsAuthenticated

from rest\_framework.authentication import TokenAuthentication

# Create your views here.

class ProdutcListView(APIView):

    authentication\_classes =[TokenAuthentication]

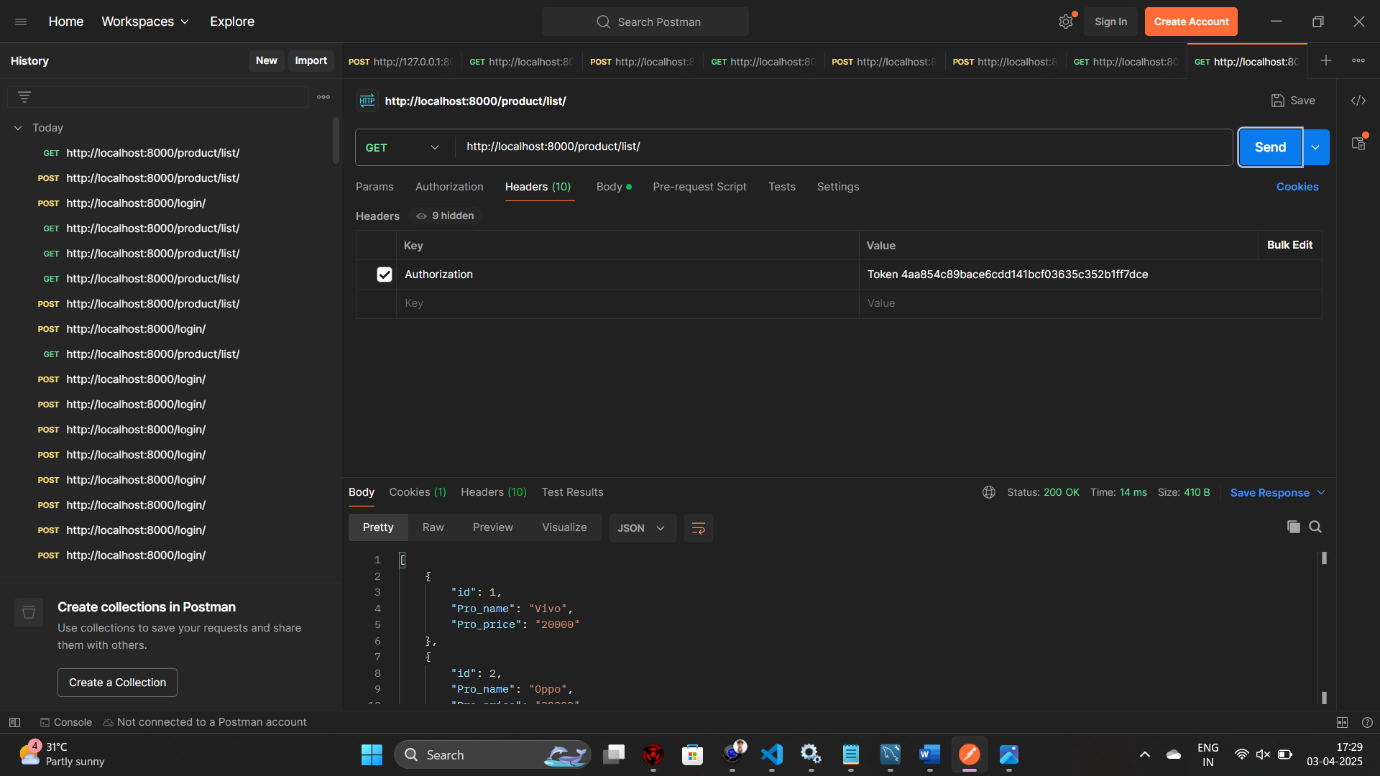
    def get(self, request):

        products = ProductModel.objects.all()

        serializer = ProductSerializer(products, many=True)

        return Response(serializer.data)

* **Postman** 
  + After login to get token use for Authorization person only

****

1. **.Using the generics:**

from django.shortcuts import render

from .serializers import \*

from .models import \*

from rest\_framework.response import Response

from rest\_framework.generics import ListAPIView

from rest\_framework.permissions import AllowAny,IsAuthenticated

from rest\_framework.authentication import TokenAuthentication

# Create your views here.

class ProdutcListView(ListAPIView):

    # permission\_classes =[AllowAny]

    authentication\_classes =[TokenAuthentication]

    queryset = ProductModel.objects.all()

    serializer\_class = ProductSerializer

* **Views**

This also same urls and postman output

**USING FRONTEND FOR REACT**

**THIS IS COMPLETELY PROACTED APP**

**First login logic then set localstorage for token**

* **Get api for post method**
  + **axios.post("http://127.0.0.1:8000/login/", {username,password});**

**The code is** const Login = () => {

  const handleLogin = async (e) => {

    e.preventDefault();

  try {

    const response = await axios.post("http://127.0.0.1:8000/login/", {username,password});

    const token = response.data.token;

    localStorage.setItem("token", token)

    navigate('/product')

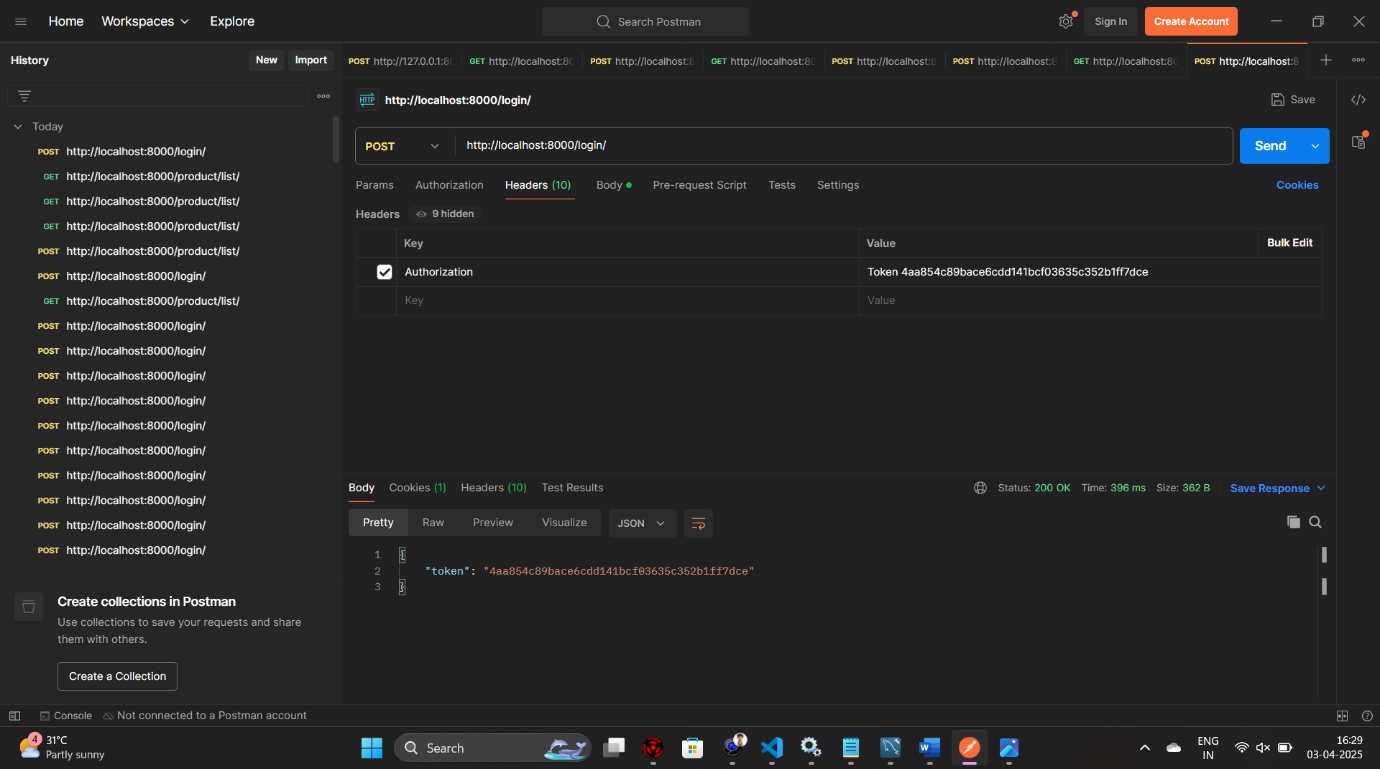
  } catch (error) {

    alert("Login faild");

  }

};

**Product logic**

* **token to get for headers** 
  + - * + [The token is backend logic to,

So frontend logic using for **{headers:{Authorization:`Token ${token}`}}**

]

* **The product code is**

const[product,setProduct] = useState([])

    useEffect(()=>{

        const token = localStorage.getItem("token");

        if(!token){

            navigate('/login')

        }

        fetchData(token)

    },[navigate])

    const fetchData = async(token) =>{

        try{

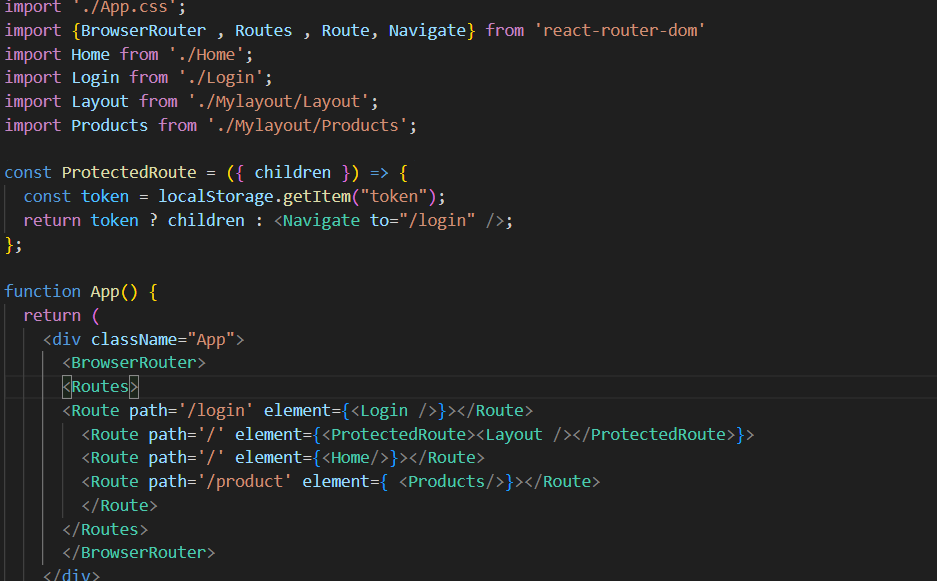
            const response =await axios.get('http://localhost:8000/product/list/',{headers:{Authorization:`Token ${token}`}})

            setProduct(response.data);

        }catch (error) {

           console.log("err");

          }

**App logic**

**Logout logic**

<button onClick={fun}>logout</button>

  const fun = ()=>{

      localStorage.removeItem('token')

      navigate('/login')

  }