# Chapter:2, basic API project

## Refarance project project\_a

## Example\_1,single apps

### Setting .py

INSTALLED\_APPS = [

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'app1',

    'rest\_framework',

]

### Pro/url

from django.contrib import admin

from django.urls import path,include

urlpatterns = [

    path('admin/', admin.site.urls),

    path('',include('app1.urls'))

]

### Urls.py

from django.urls import path

from.views import one,two

urlpatterns = [

    path('',one.as\_view(),name='home'),

    path('api/',two.as\_view(),name='home'),

]

### Views.py

#from django.shortcuts import render

from django.views.generic import ListView

from.models import cool

from rest\_framework import generics

from .serializers import sos

class one(ListView):

    template\_name='home.html'

    model=cool

class two(generics.ListAPIView):

    queryset = cool.objects.all()

    serializer\_class = sos

### Models.py

from django.db import models

class cool(models.Model):

    title=models.TextField()

    roll=models.IntegerField()

    def \_\_str\_\_(self):

        return self.title

### Admin.py

from django.contrib import admin

from.models import cool

admin.site.register(cool)

### Serializer.py

from rest\_framework import serializers

from app1.models import cool

class sos(serializers.ModelSerializer):

    class Meta:

        model = cool

        fields = ('title', 'roll')

### Home.html

{% for cool in object\_list %}

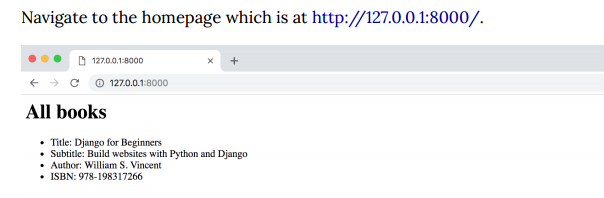
<li>{{cool.title}}</li>

<li>{{cool.roll}}</li>

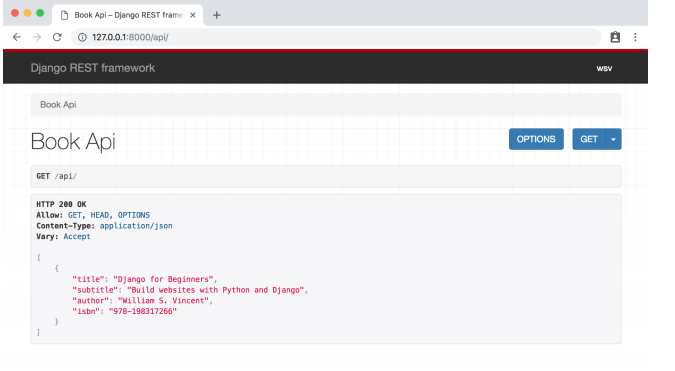
{% endfor %}

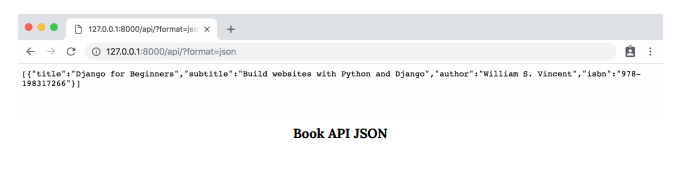
## Example\_2,multiple apps with api

Before api using



after api using rest and json





### General setting

Create a app=books

#### Setting.py

INSTALLED\_APPS = [

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'books',

]

#### url.py

from django.contrib import admin

from django.urls import path, include

urlpatterns = [

    path('admin/', admin.site.urls),

    path('', include('books.urls')),

]

#### urls.py

from django.urls import path

from .views import BookListView

urlpatterns = [

    path('', BookListView.as\_view(), name='home'),

]

#### Views.py

from django.shortcuts import render

# Create your views here.

from django.views.generic import ListView

from .models import Book

class BookListView(ListView):

    model = Book

    template\_name = 'book\_list.html'

Models.py setting

#### Models.py

from django.db import models

from django.db import models

class Book(models.Model):

    title = models.CharField(max\_length=250)

    subtitle = models.CharField(max\_length=250)

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

    isbn = models.CharField(max\_length=13)

    def \_\_str\_\_(self):

        return self.title

#### Admin.py

from django.contrib import admin

from .models import Book

admin.site.register(Book)

#### Book\_list.html

{% for book in object\_list %}

<ul>

<li>Title: {{ book.title }}</li>

<li>Subtitle: {{ book.subtitle }}</li>

<li>Author: {{ book.author }}</li>

<li>ISBN: {{ book.isbn }}</li>

</ul>

{% endfor %}

### Rest api setting

Pip install djangorestframework

Create new app name api

#### Setting.py

INSTALLED\_APPS = [

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'books',

    'rest\_framework', # new

    'api',

]

#### Pro/url

from django.contrib import admin

from django.urls import path, include

urlpatterns = [

    path('admin/', admin.site.urls),

    path('', include('books.urls')),

    path('api/', include('api.urls')), # new

]

#### Api/ url.py

from django.urls import path

from .views import BookAPIView

urlpatterns = [

    path('', BookAPIView.as\_view()),

]

#### Api /views.py

from django.shortcuts import render

from rest\_framework import generics

from books.models import Book

from .serializers import BookSerializer

class BookAPIView(generics.ListAPIView):

    queryset = Book.objects.all()

    serializer\_class = BookSerializer

#### api/serializers.py

from rest\_framework import serializers

from books.models import Book

class BookSerializer(serializers.ModelSerializer):

    class Meta:

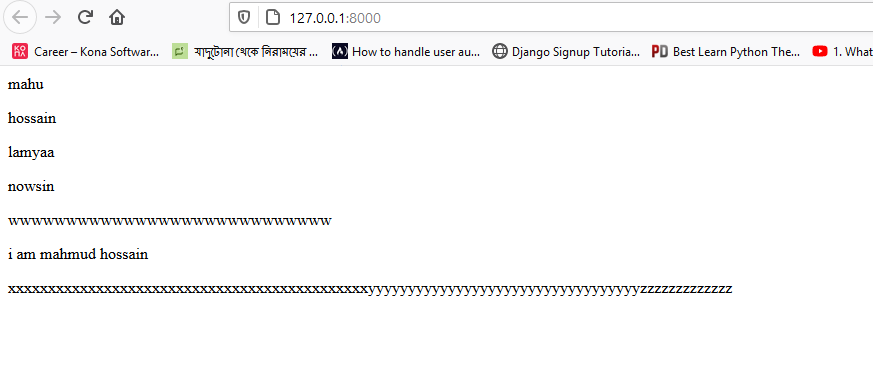
        model = Book

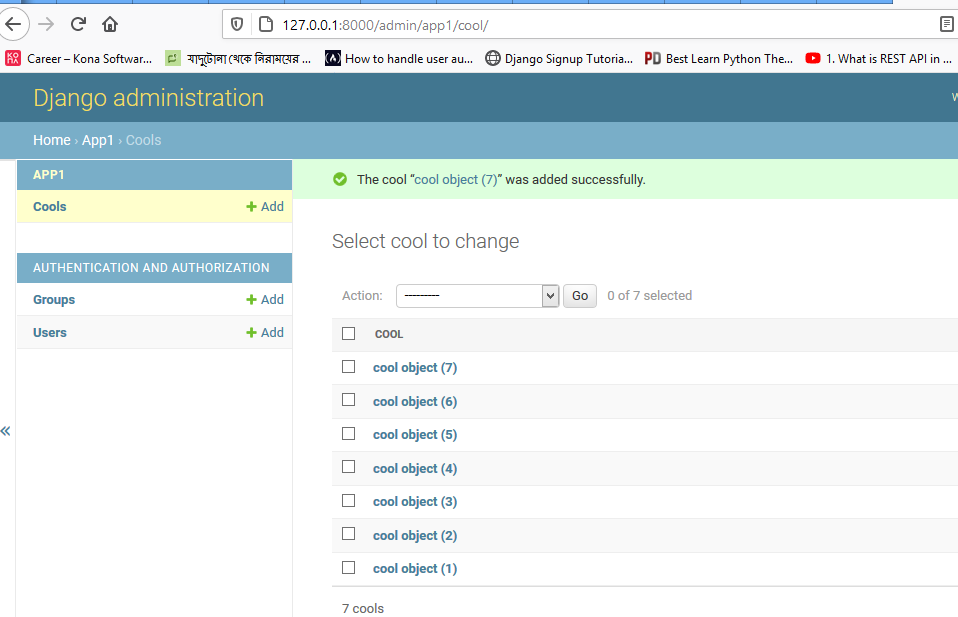
        fields = ('title', 'subtitle', 'author', 'isbn')

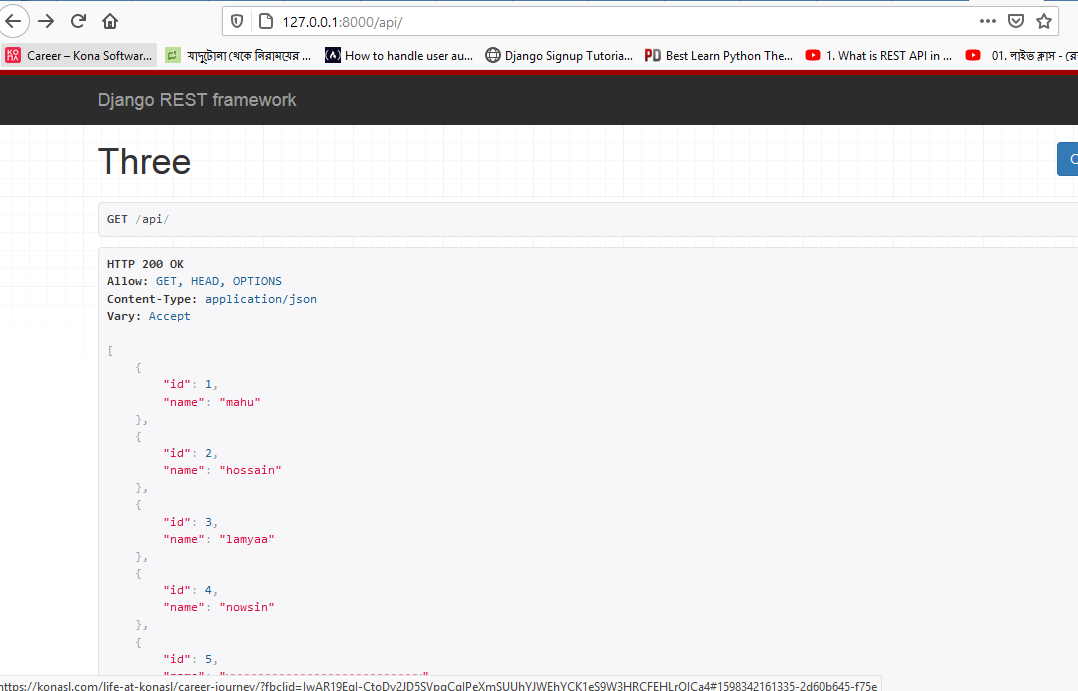
# Chapter\_3 combine, rest api with detailview template/ Pk page

## Remember chapter\_3 and chapter\_4 is combine

## Refarance project project\_b







Ey project a extra hisebe ListView (home.html) er kaj kora hoyese,eta na korlew hobe

### Install pakages:

Pip install install djangorestframework

install django-cors-headers

### Setting.py

INSTALLED\_APPS = [

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'app1',

    'rest\_framework',

]

### Pro/url

from django.contrib import admin

from django.urls import path,include

urlpatterns = [

    path('admin/', admin.site.urls),

    path('',include('app1.urls'))

]

### Urls.py

from django.urls import path

from.views import one,two,three

urlpatterns = [

    path('',one.as\_view(),name='home'),

    path('api/',three.as\_view(),name='base'),

    path('api/<int:pk>/',two.as\_view(),name='base'),

]

### Views.py

from django.views.generic import DetailView,ListView

from rest\_framework import generics

from .models import cool

from.serializers import sos

class one(ListView):

    template\_name='home.html'

    model=cool

class three(generics.ListAPIView):

    template\_name='base.html'

    queryset = cool.objects.all()

    serializer\_class = sos

class two(generics.RetrieveAPIView):

    queryset = cool.objects.all()

    serializer\_class = sos

    template\_name='base.html’

### Model.py

from django.db import models

class cool(models.Model):

    name=models.TextField()

### Admin.py

from django.contrib import admin

from.models import cool

admin.site.register(cool)

### Serializers.py

from rest\_framework import serializers

from app1.models import cool

class sos(serializers.ModelSerializer):

    class Meta:

        model = cool

        fields = '\_\_all\_\_'

### Home.html

{% for cool in object\_list %}

<li>{{cool.name}}</li>

{% endfor %}

### detail.html

{{cool.name}}

## CORS/port change sytem

CORS

There’s one last step we need to do and that’s deal with Cross-Origin Resource

Sharing (CORS). Whenever a client interacts with an API hosted on a different domain

(mysite.com vs yoursite.com) or port (localhost:3000 vs localhost:8000) there are

potential security issues. Specifically, CORS requires the server to include specific HTTP headers that allow for

the client to determine if and when cross-domain requests should be allowed.

Our Django API back-end will communicate with a dedicated front-end that is located

on a different port for local development and on a different domain once deployed. The easiest way to handle this–-and the one recommended by Django REST Framework–-is to use middleware that will automatically include the appropriate HTTP

headers based on our settings.

The package we will use is django-cors-headers which can be easily added to our

existing project.

[mane amra Jodi local host port 3000 a api search korte chi]

Pip install django-cors-headers

### Setting.py

INSTALLED\_APPS = [

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'app1',

    'rest\_framework',

    'corsheaders', #new

]

MIDDLEWARE = [

    'django.middleware.security.SecurityMiddleware',

    'django.contrib.sessions.middleware.SessionMiddleware',

    'corsheaders.middleware.CorsMiddleware', # new

    'django.middleware.common.CommonMiddleware',#new

    'django.middleware.csrf.CsrfViewMiddleware',

    'django.contrib.auth.middleware.AuthenticationMiddleware',

    'django.contrib.messages.middleware.MessageMiddleware',

    'django.middleware.clickjacking.XFrameOptionsMiddleware',

]

CORS\_ORIGIN\_WHITELIST = (

'http://localhost:3000',

#'http://localhost:8000',

)

It’s very important that corsheaders.middleware.CorsMiddleware appears in the proper

location. That is above django.middleware.common.CommonMiddleware in the MIDDLEWARE

setting since middlewares are loaded top-to-bottom. Also note that we’ve whitelisted

two domains: localhost:3000 and localhost:8000. The former is the default port for

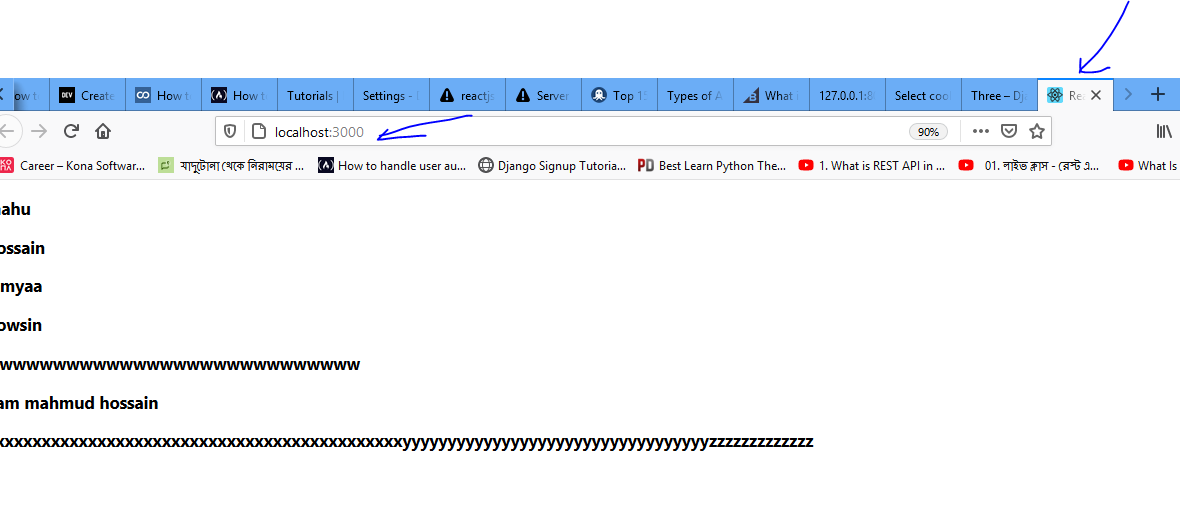
React, which we will use for our front-end in the next chapter. The latter is the default

Django port.

# Chapter\_4 React fontend

Reference project: project\_b

Above chapter\_3 all photo +

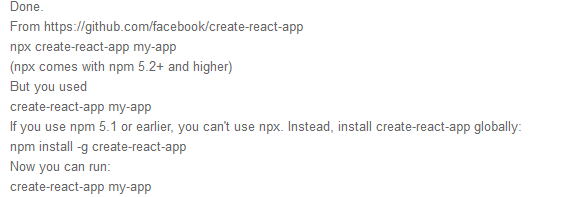


Chapter \_3 er sathe add hobe ,mone rakte hobe react app ‘localhost:3000’ port a run hoy so above all step need first the start now

1. First install node software form node.js website
2. After install check by CDM: node -v and npm –v
3. From CMD install : npm install –g create-react-app #eyta install korle react er all liabrary install hoy
4. From CMD create a react app : create-react-app Mahmud\_react #here is folder name, eta issa moto diya ji
5. Dir/ls command dile 2ta folder show korbe ,ekta django project er ,oporta react er
6. Cd Mahmud\_react #goto the react folder
7. Type npm start to run react page
8. Mone rakte hobe 2ta terminal open rakte hobe (cmd power shell for django page running and virtual stdio for react app er jonne)
9. Django runn takbe 8000 port a and react run hobe 3000 port a
10. CMD : npm install axios
11. Goto Mahmud\_react /src/APP.Js , open APP.JS file and edit , eta niche add kora hoyse

### Pakage install problem fixed

<https://html.developreference.com/article/10421668/create-react-app+command+aborting+installation+Why%3F>



### APP.JS(traial)

Eta manually rest file er data copy kore react app.js file a past kore npm start diya show kora

import React, { Component } from 'react';

const list = [

  {

    "id": 1,

    "name": "mahu"

},

{

    "id": 2,

    "name": "hossain"

},

{

    "id": 3,

    "name": "lamyaa"

},

{

    "id": 4,

    "name": "nowsin"

},

{

    "id": 5,

    "name": "wwwwwwwwwwwwwwwwwwwwwwwwwwww"

},

{

  "id": 6,

  "name": "i am mahmud hossain"

}

]

class App extends Component {

  constructor(props) {

  super(props);

  this.state = { list };

  }

  render() {

    return (

      <div>

      {this.state.list.map(item => (

      <div key={item.id}>

      <h1>{item.name}</h1>

      </div>

      ))}

      </div>

      );

      }

      }

      export default App;

### APP.JS(final)

Eta main format ,ekahane sudo ekbar configure kore model artribut add kore react page refresh ban pm start korley react font show korbe

import React, { Component } from 'react';

import axios from 'axios'; // new

class App extends Component {

//default

state = {

//app1 is app name

app1: []

};

// new

componentDidMount() {

//cool is model name

this.getcool();

}

// new

getcool() {

axios

//rest api page link

.get('http://127.0.0.1:8000/api/')

//jason api page link

//.get('http://127.0.0.1:8000/api/?format=json')

.then(res => {

this.setState({app1: res.data });

})

.catch(err => {

console.log(err);

});

}

render() {

return (

  <div>

{this.state.app1.map(item => (

//default dite hobe

<div key={item.id}>

<h3>{item.name}</h3>

</div>

))}

</div>

);

}

}

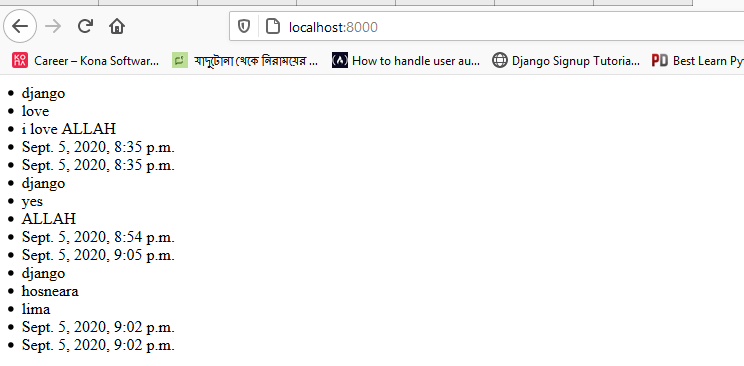
export default App;

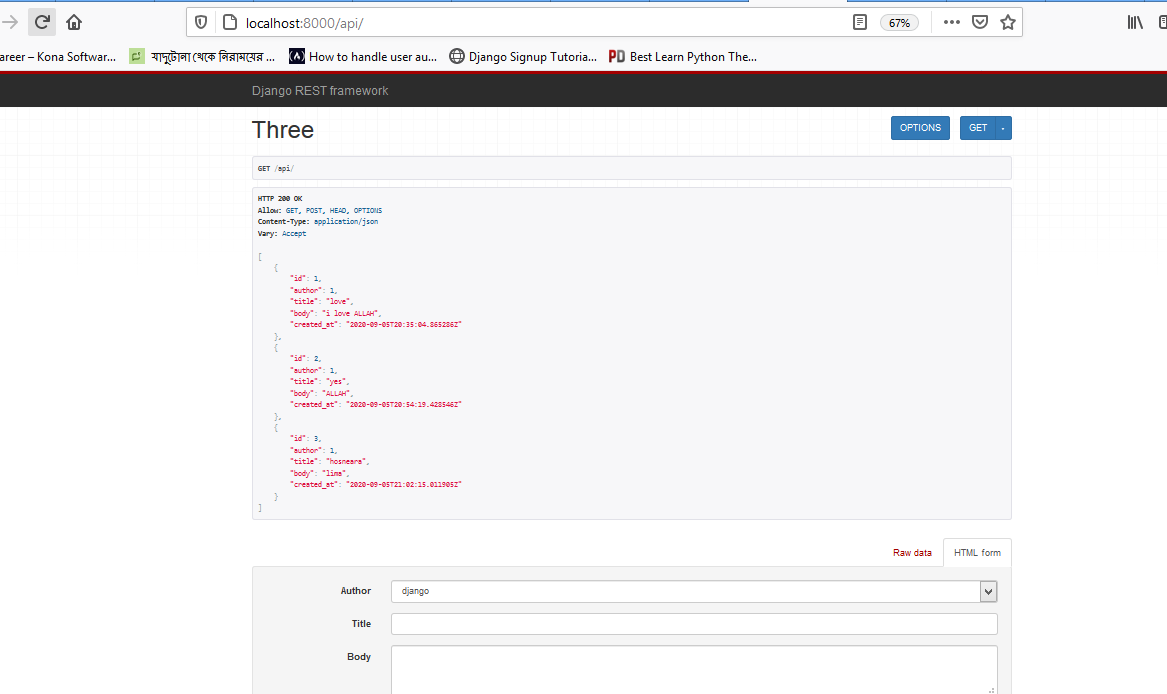
# Chapter\_5 blog api/CRUD project api

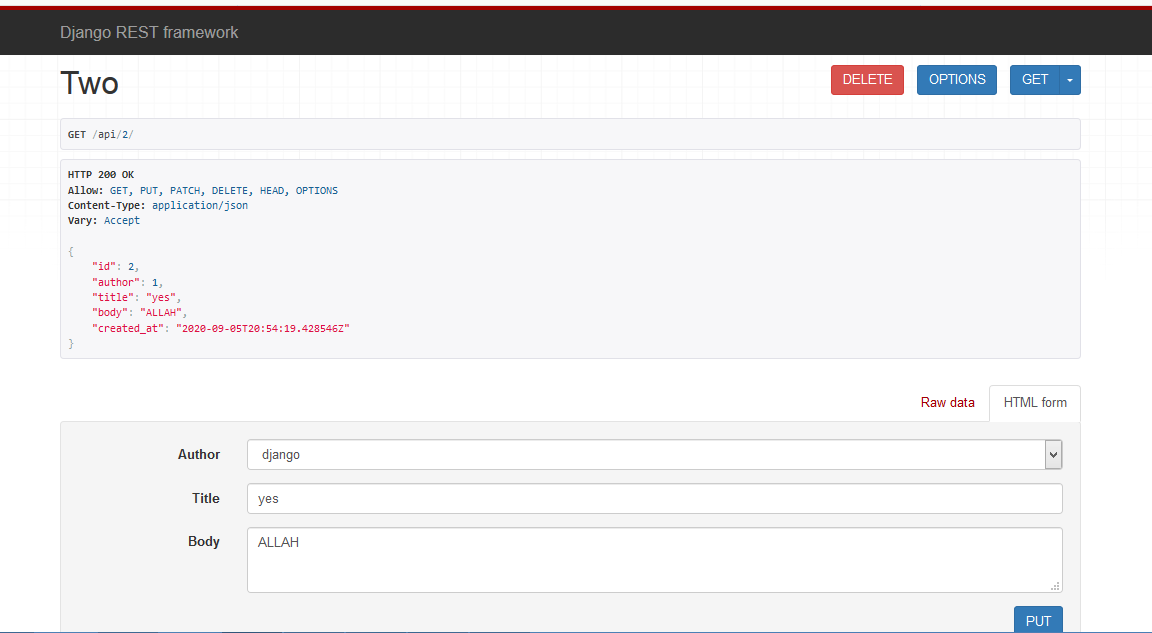
## Example\_1

by django CreateApiView and UpdateDeleteView

## Refarance project project\_d







### Setting.py

INSTALLED\_APPS = [

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'app1',

    'rest\_framework',

]

### Pro/url

from django.contrib import admin

from django.urls import path,include

urlpatterns = [

    path('admin/', admin.site.urls),

    path('',include('app1.urls'))

]

### Urls.py

from django.urls import path

from.views import one,two,three,four

urlpatterns = [

    path('',one.as\_view(),name='home'), #for show model data in html page

    path('api/',two.as\_view(),name='base'),  #for show model data in rest format

    path('api/new/',three.as\_view(),name='base'), #for create/add new data

    path('api/<int:pk>/',four.as\_view(),name='base'), #for red,update,delete data

]

]

### Views.py

from django.shortcuts import render

from django.views.generic import ListView #use for listview purpose

from.models import cool     #include models.py

from rest\_framework import generics #for rest

from .serializers import sos #include serializes.py

class one(ListView):

    template\_name='home.html'

    model=cool

#for api view

class two(generics.ListAPIView): #'ListAPIView'  all data in list array

    queryset = cool.objects.all()

    serializer\_class = sos

    template\_name='base.html'

class three(generics.ListCreateAPIView): # 'ListCreateAPIView' create/add new date

    queryset = cool.objects.all().order\_by('id')  #show all list data in accensing, (1.2.3.4.....)

    #queryset = cool.objects.all().order\_by('-id') #show all data in deaccensing, (.......4.3.2.1)

    #queryset = cool.objects.all().order\_by('-id')[:1] #show only last added data

    serializer\_class = sos

    template\_name='base.html'

#for detail view

class four(generics.RetrieveUpdateDestroyAPIView): #genarelly write 'RetrieveAPIView' but here RetrieveUpdateDestroyAPIView

    queryset = cool.objects.all()                 #cause of red, update and delete post purpose

    serializer\_class = sos

    template\_name='base.html'

### Models.py

from django.db import models

from django.contrib.auth.models import User #it use for ForeingKey purpose otherwise no need

class cool(models.Model):

    author = models.ForeignKey(User, on\_delete=models.CASCADE)

    title = models.CharField(max\_length=50)

    body = models.TextField()

    created\_at = models.DateTimeField(auto\_now\_add=True)

    updated\_at = models.DateTimeField(auto\_now=True)

    def \_\_str\_\_(self):

        return self.title

### Admin.py

from django.contrib import admin

from .models import cool

admin.site.register(cool)

### Serializes.py

from rest\_framework import serializers #need for rest

from .models import cool

class sos(serializers.ModelSerializer):

    class Meta:

        fields = ('id', 'author', 'title', 'body', 'created\_at',)

        model = cool

### Home.html

{% for cool in object\_list %}

 <li>{{cool.author}}</li>

 <li>{{cool.title}}</li>

 <li>{{cool.body}}</li>

 <li>{{cool.created\_at}}</li>

 <li>{{cool.updated\_at}}</li>

{% endfor %}

### Base.html

<li>{{cool.author}}</li>

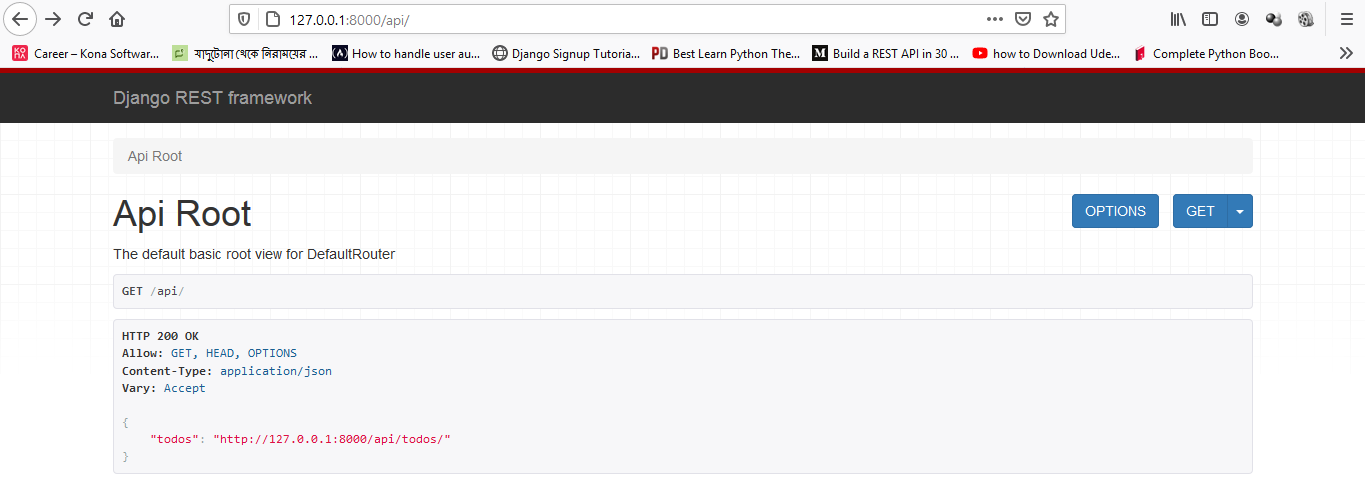
 <li>{{cool.title}}</li>

 <li>{{cool.body}}</li>

 <li>{{cool.created\_at}}</li>

 <li>{{cool.updated\_at}}</li>

## Example\_2



Link: <https://scotch.io/tutorials/build-a-to-do-application-using-django-and-react>

Only viewset

Install above all pakage

### Setting.py

INSTALLED\_APPS = [

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'corsheaders',            *# add this*

    'rest\_framework',         *# add this*

    'todo',

]

MIDDLEWARE = [

    'corsheaders.middleware.CorsMiddleware',    *# add this*

    'django.middleware.security.SecurityMiddleware',

    'django.contrib.sessions.middleware.SessionMiddleware',

    'django.middleware.common.CommonMiddleware',

    'django.middleware.csrf.CsrfViewMiddleware',

    'django.contrib.auth.middleware.AuthenticationMiddleware',

    'django.contrib.messages.middleware.MessageMiddleware',

    'django.middleware.clickjacking.XFrameOptionsMiddleware',

]

CORS\_ORIGIN\_WHITELIST = (

'http://localhost:3000',

*#'http://localhost:8000',*

)

### Pro/url

from django.contrib import admin

from django.urls import path, include                 *# add this*

from rest\_framework import routers                    *# add this*

from todo import views                            *# add this*

router = routers.DefaultRouter()                      *# add this*

router.register(r'todos', views.TodoView, 'todo')     *# add this*

urlpatterns = [

    path('admin/', admin.site.urls),

    path('api/', include(router.urls))                *# add this*

]

### Urls.py

No need

### Views.py

from django.shortcuts import render

from rest\_framework import viewsets          *# add this*

from .serializers import TodoSerializer      *# add this*

from .models import Todo                     *# add this*

class **TodoView**(**viewsets**.**ModelViewSet**):       *# add this*

    serializer\_class = TodoSerializer          *# add this*

    queryset = Todo.objects.all()

### Model.py

from django.db import models

*# Create your models here.*

*# add this*

class **Todo**(**models**.**Model**):

    title = models.CharField(max\_length=120)

    description = models.TextField()

    completed = models.BooleanField(default=False)

    def **\_str\_**(self):

        return self.title

### Admin.py

from django.contrib import admin

from .models import Todo *# add this*

class **TodoAdmin**(**admin**.**ModelAdmin**):  *# add this*

    list\_display = ('title', 'description', 'completed') *# add this*

*# Register your models here.*

admin.site.register(Todo, TodoAdmin) *# add this*

### Serializers.py

from rest\_framework import serializers

from .models import Todo

class **TodoSerializer**(**serializers**.**ModelSerializer**):

    class **Meta**:

        model = Todo

        fields = ('id', 'title', 'description', 'completed')

# Chapter\_6 permission

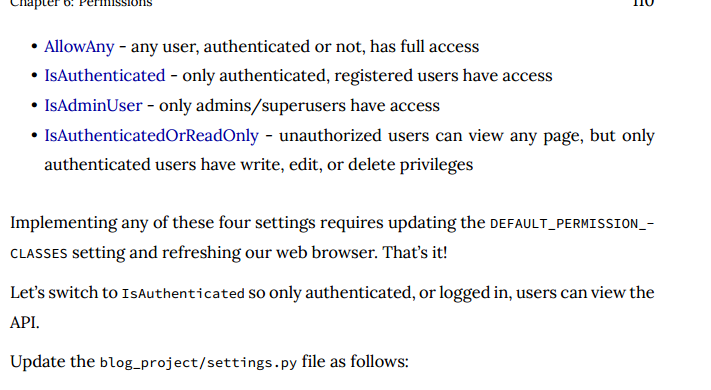
Sample project, project\_e

### Permission classification

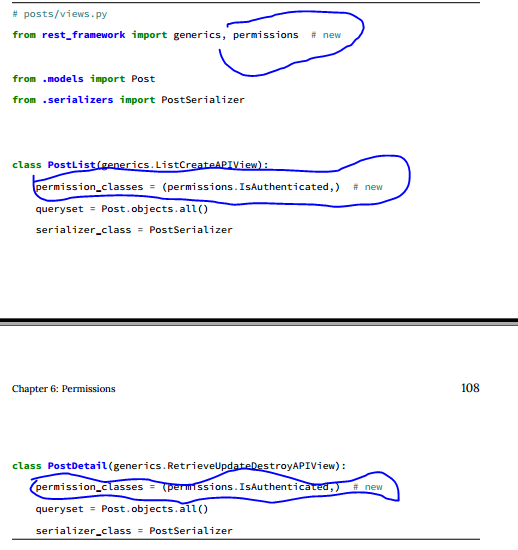
In rest api permission are three lavel permission . othat 3 vabe permission create kora ji

### Project lavel permission: eta dite hoy setting.py file teke

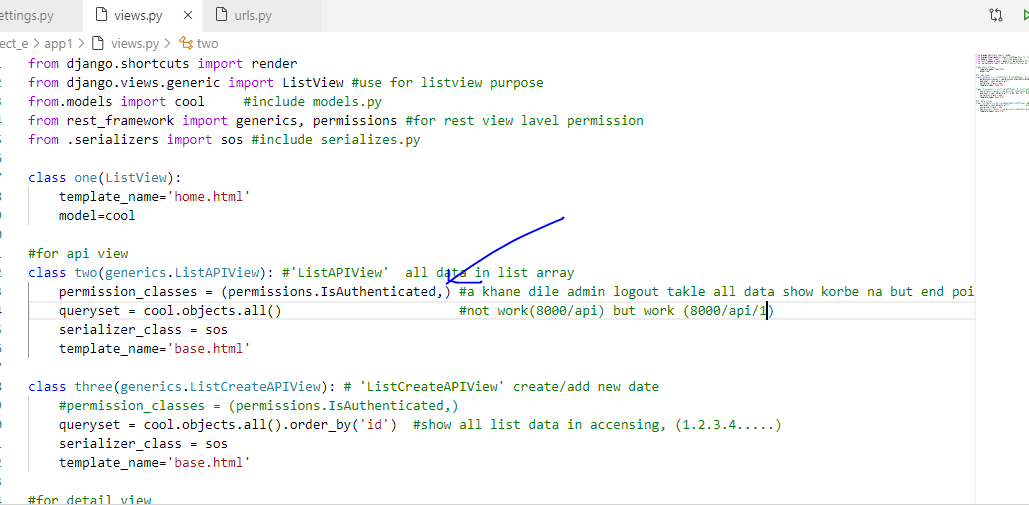
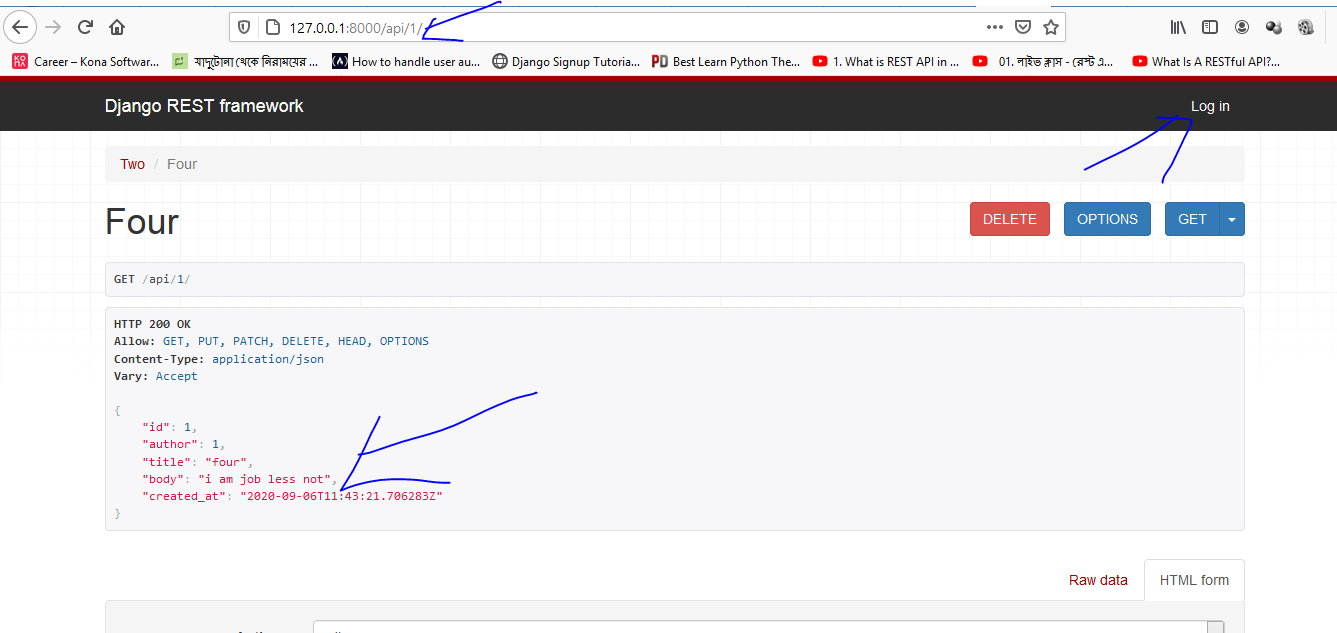
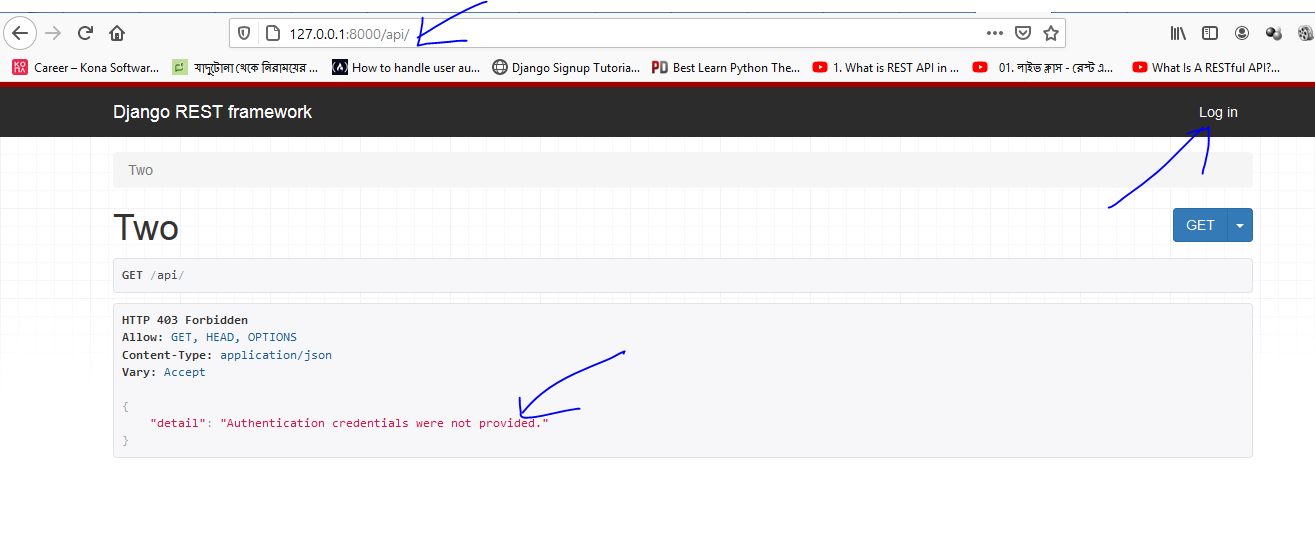
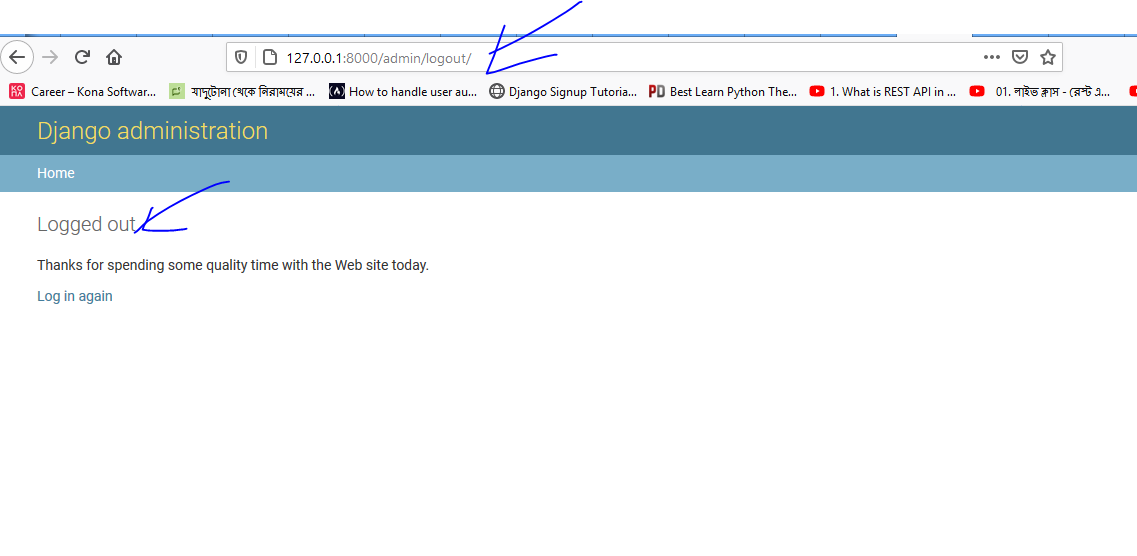




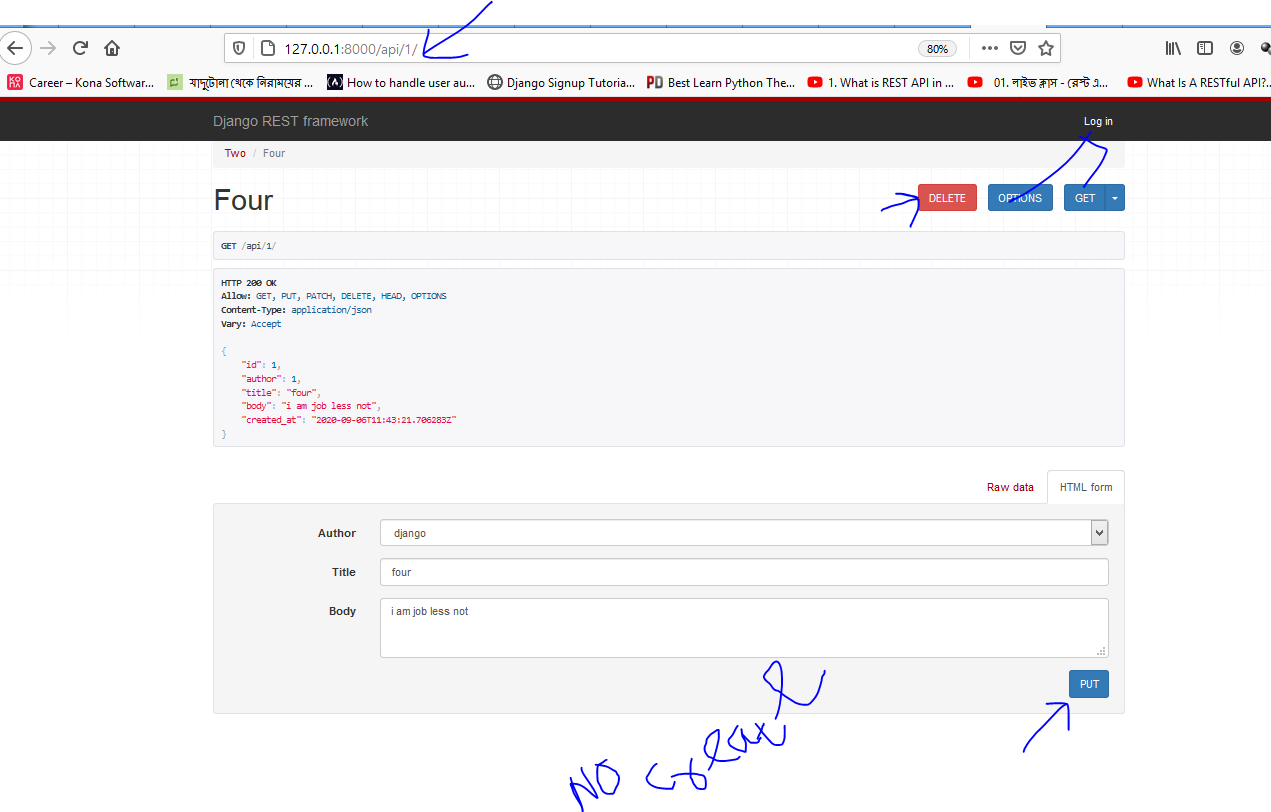
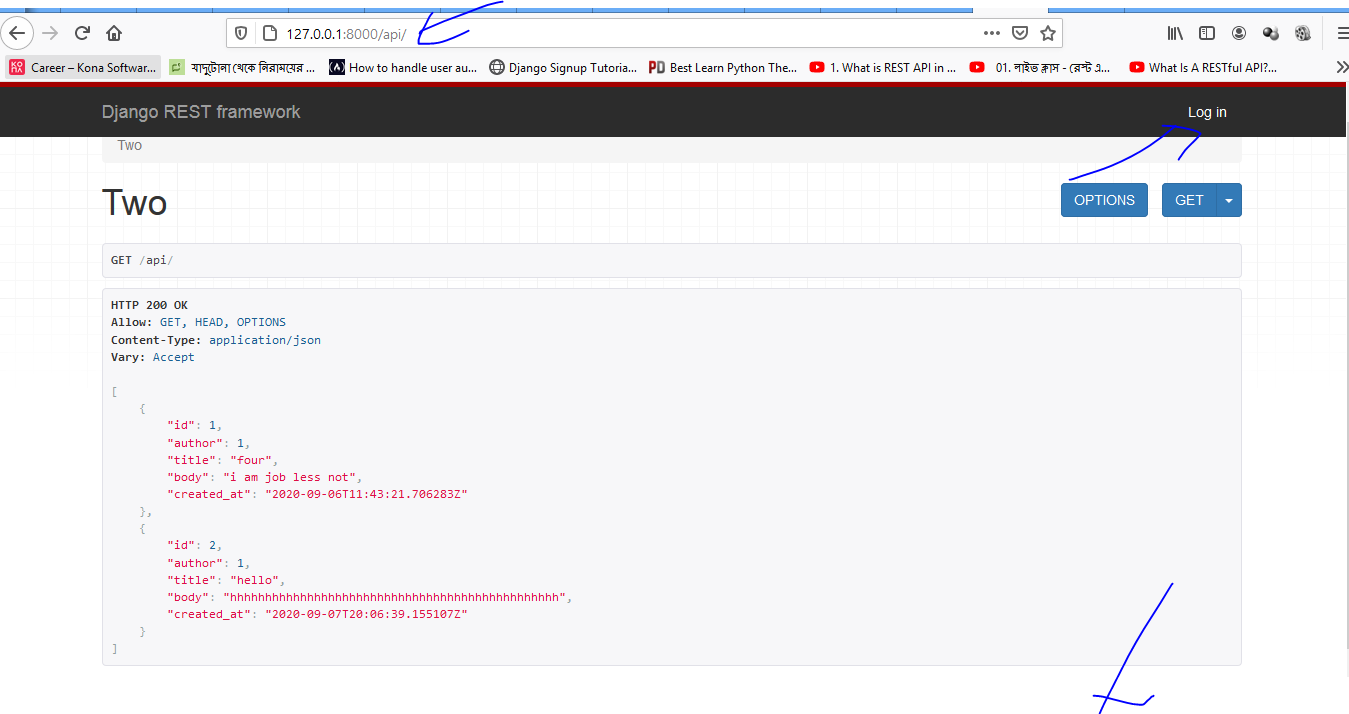
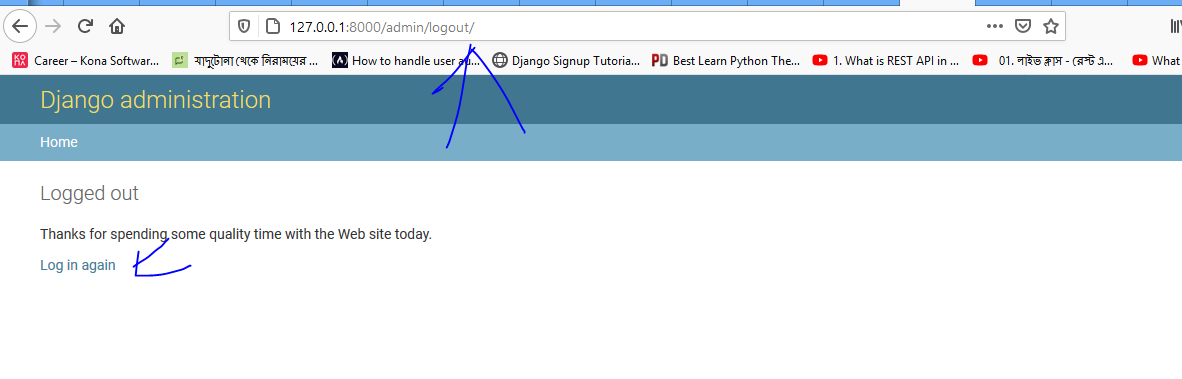
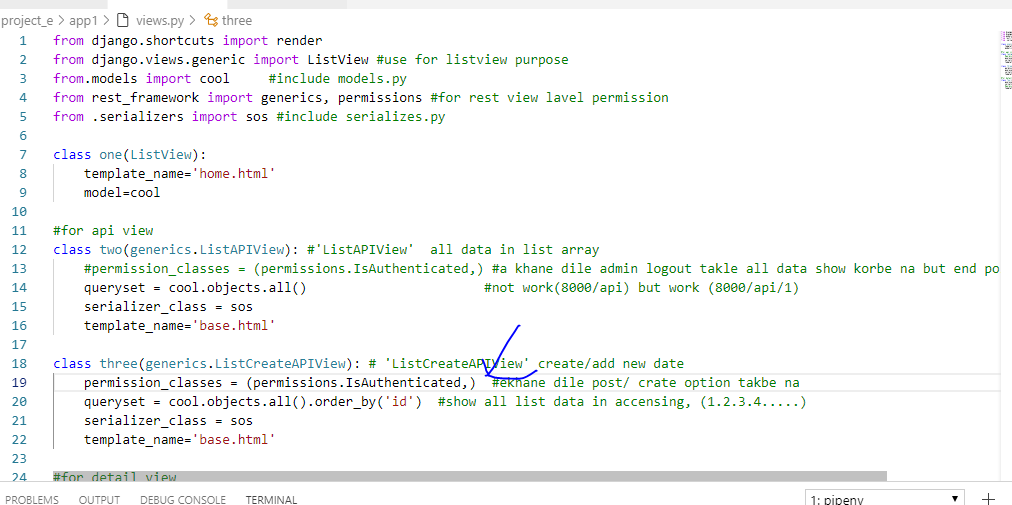
### View lavel permission: eta dite hoy views.py file teke, Etar jonne view.py file a add korte hoy



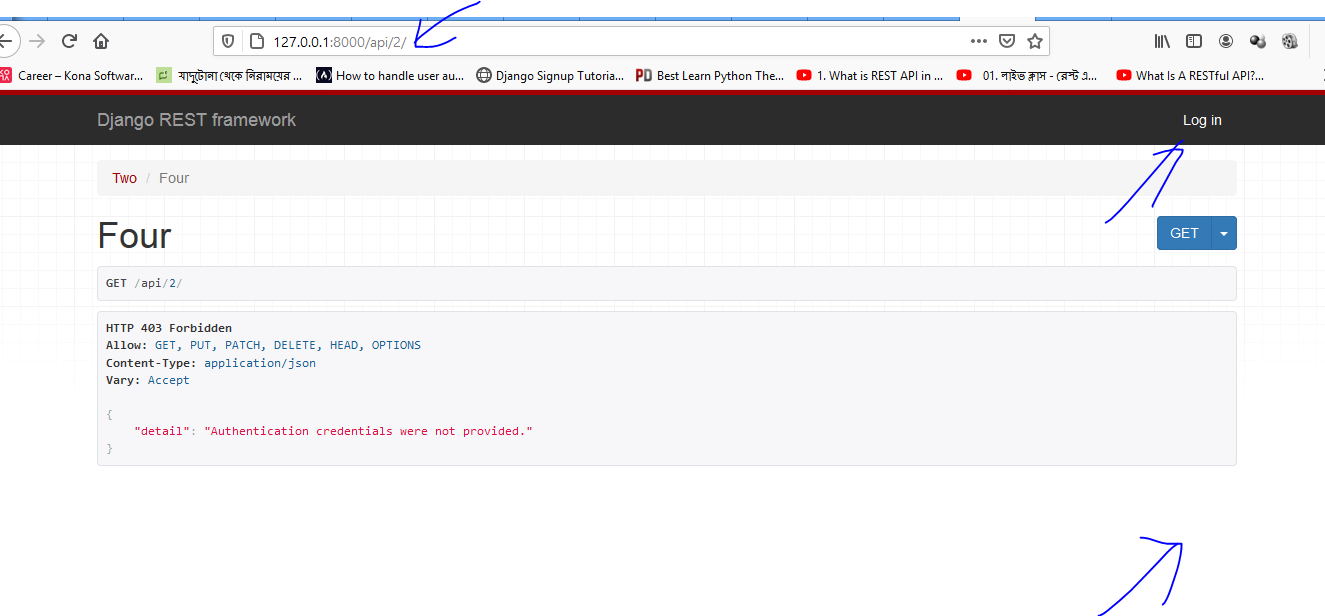
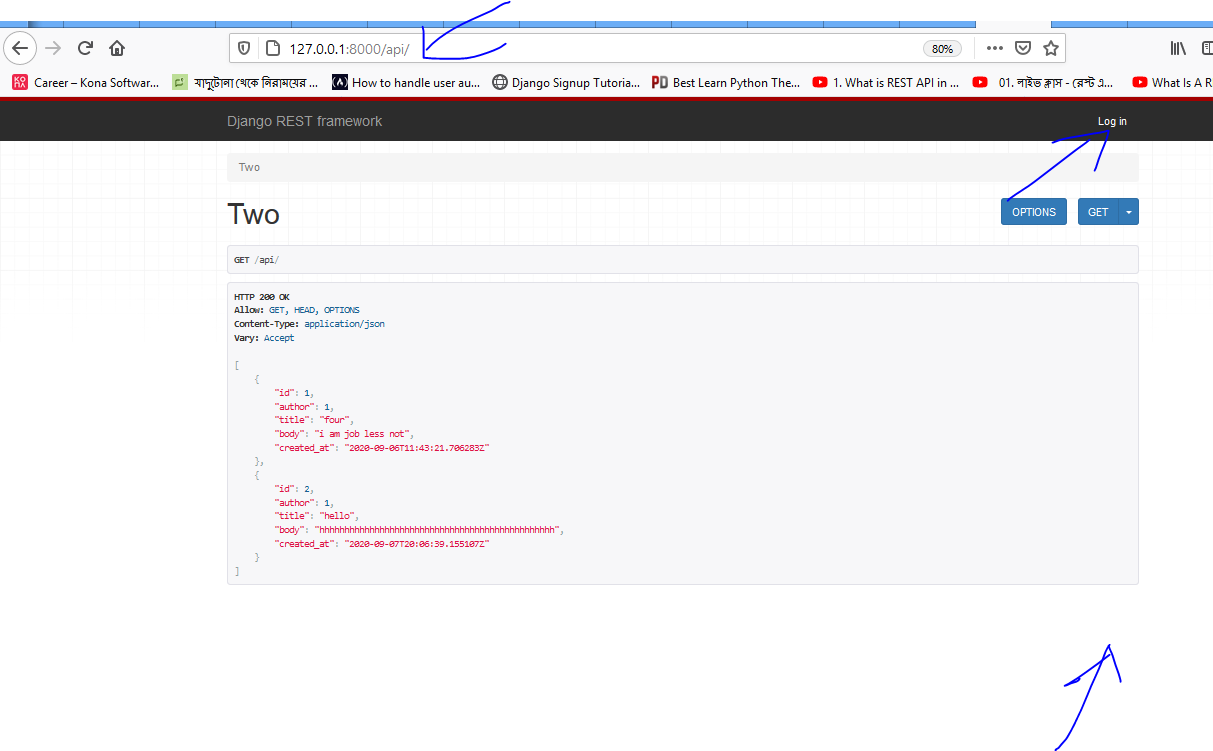
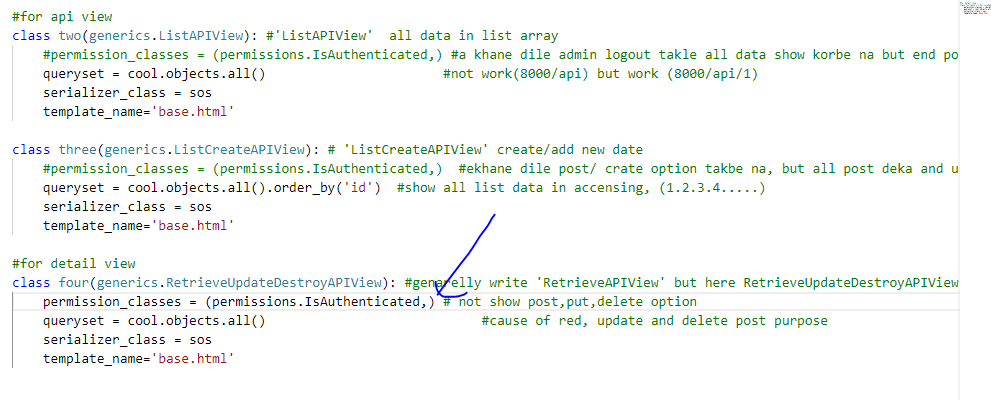
#### When give permission in ListAPIView



#### When give permission in ListCreateAPIView

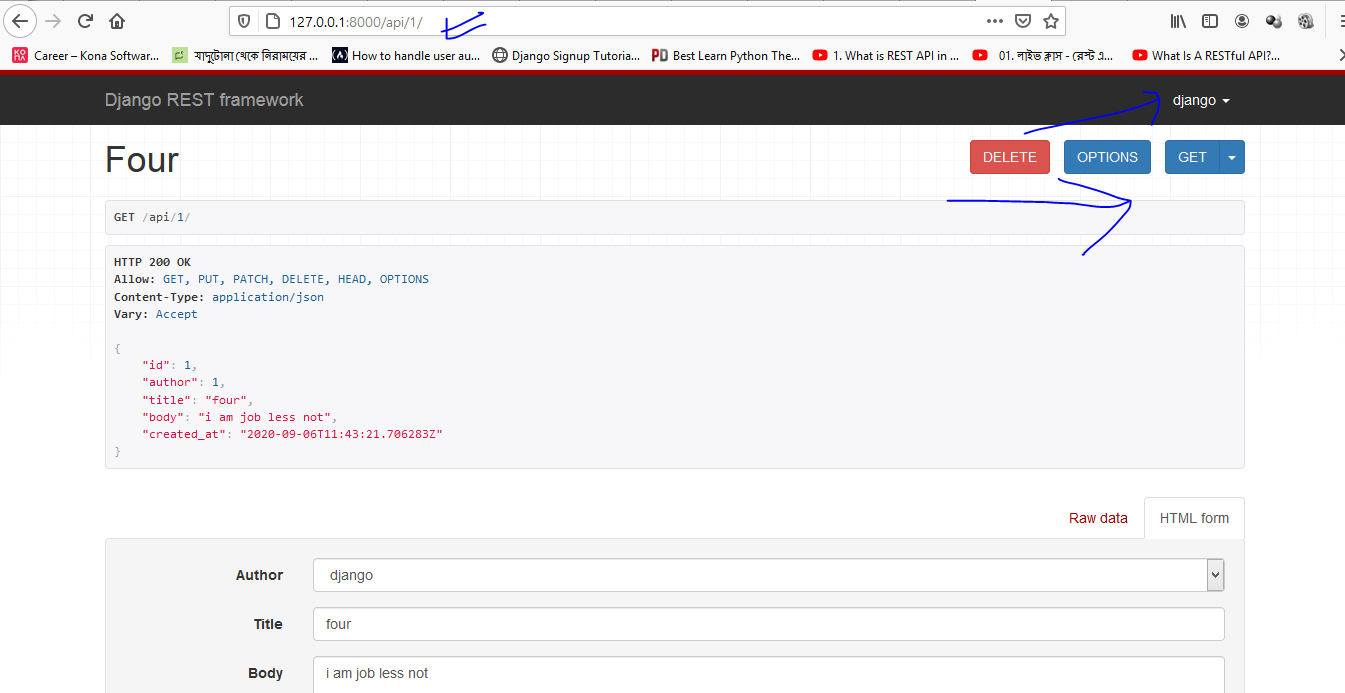


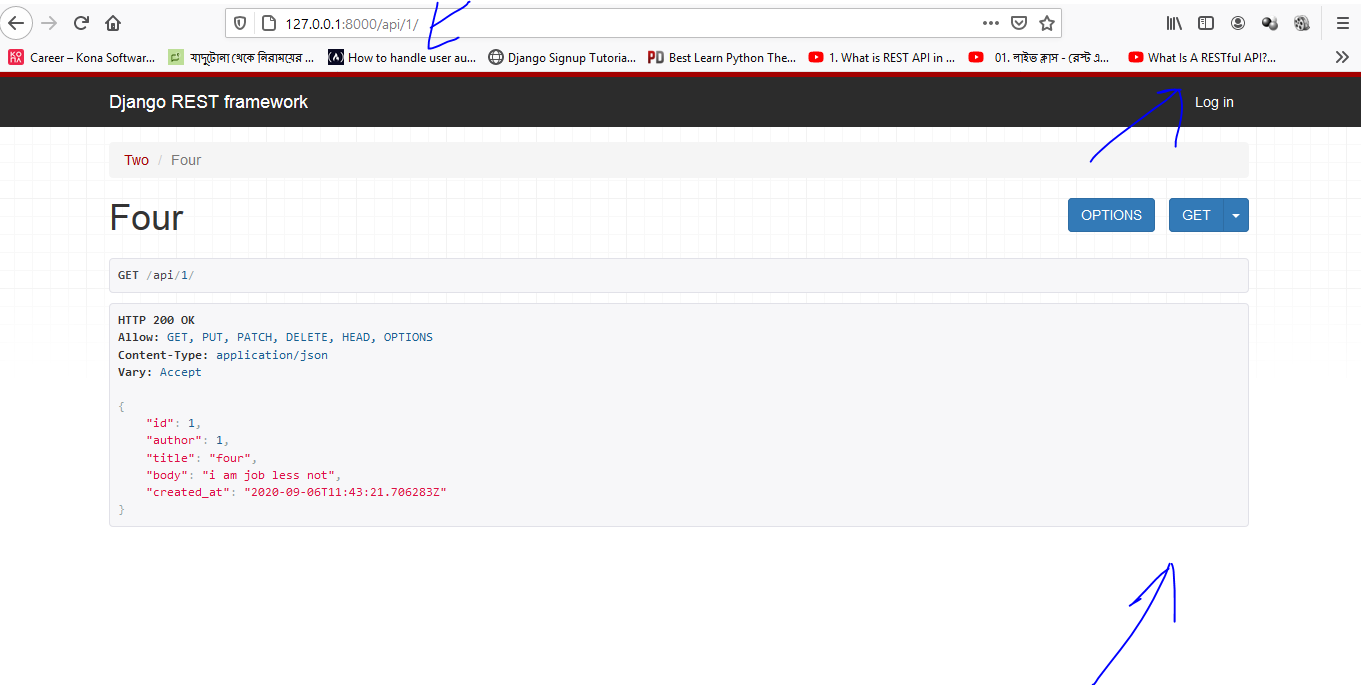
#### When give permission in RetriveUpdateDeleteAPIView

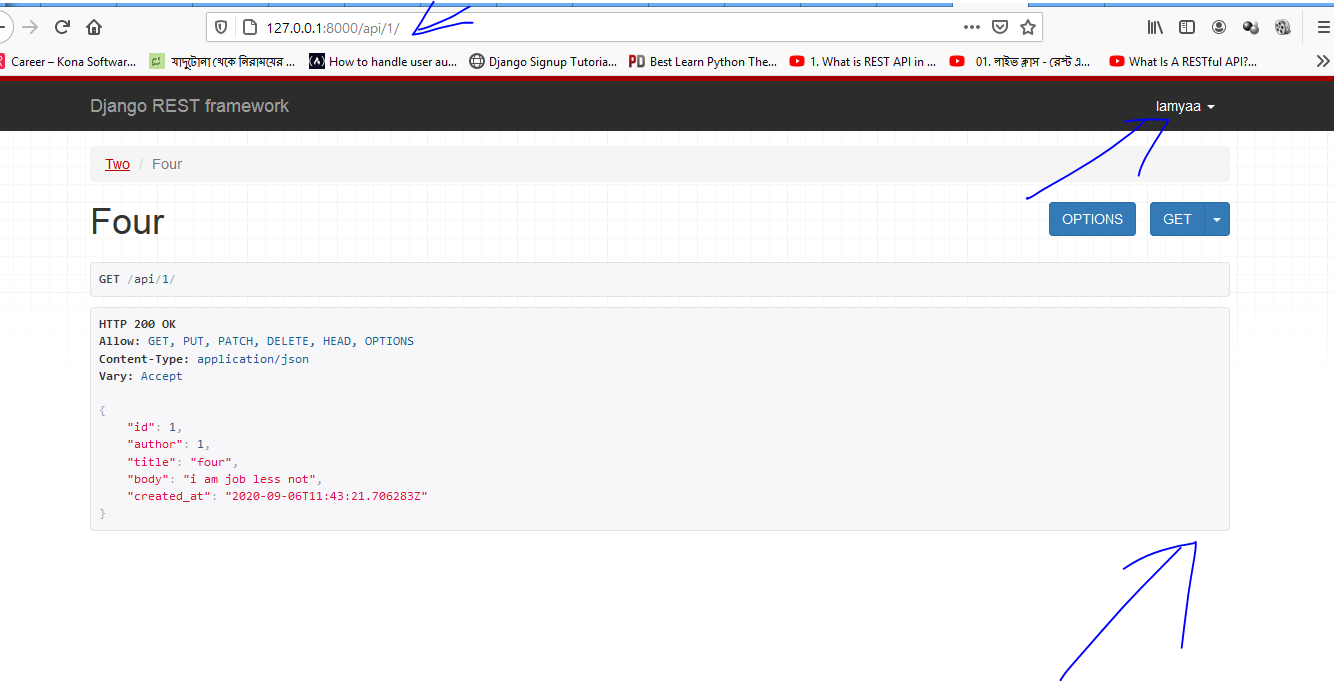


### Object/customize permission: eta dite hoy permission.py file teke

Eta muloto, superuser login korle ga post,put,delete option pabe ta sara onno sub user login talke sudo show korte parbe all data and end point data both







### Setting.py

INSTALLED\_APPS = [

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'app1',

    'rest\_framework',

]

REST\_FRAMEWORK = {   #this is rest pase authenticate pannel

'DEFAULT\_PERMISSION\_CLASSES': [

#'rest\_framework.permissions.AllowAny', #any user, authenticated or not, has full access

#'rest\_framework.permissions.IsAuthenticated',  #only authenticated, registered users have access

#'rest\_framework.permissions.IsAdminUser', #only admins/superusers have access

'rest\_framework.permissions.IsAuthenticatedOrReadOnly', #unauthorized users can view any page, but only

                                                           #authenticated users have write, edit, or delete privileges

]

}

### Pro/url

from django.contrib import admin

from django.urls import path,include

urlpatterns = [

    path('admin/', admin.site.urls),

    path('',include('app1.urls')),

    path('', include('rest\_framework.urls')), #for many to one field

]

### Urls.py

from django.urls import path

from.views import one,two,three,four

urlpatterns = [

    path('',one.as\_view(),name='home'), #for show model data in html page

    path('api/',two.as\_view(),name='base'),  #for show model data in rest format

    path('api/new/',three.as\_view(),name='base'), #for create/add new data

    path('api/<int:pk>/',four.as\_view(),name='base'), #for red,update,delete data

]

### Views.py

from django.shortcuts import render

from django.views.generic import ListView #use for listview purpose

from.models import cool     #include models.py

from rest\_framework import generics#, permissions #for rest view lavel permission

from .serializers import sos #include serializes.py

from .permissions import IsAuthorOrReadOnly # new

class one(ListView):

    template\_name='home.html'

    model=cool

#for api view

class two(generics.ListAPIView): #'ListAPIView'  all data in list array

    #permission\_classes = (permissions.IsAuthenticated,) #a khane dile admin logout takle all data show korbe na but end point show korbe

    queryset = cool.objects.all()                       #not work(8000/api) but work (8000/api/1)

    serializer\_class = sos

    template\_name='base.html'

class three(generics.ListCreateAPIView): # 'ListCreateAPIView' create/add new date

    #permission\_classes = (permissions.IsAuthenticated,)  #ekhane dile post/ crate option takbe na, but all post deka and update,delete kora jabe

    queryset = cool.objects.all().order\_by('id')  #show all list data in accensing, (1.2.3.4.....)

    serializer\_class = sos

    template\_name='base.html'

#for detail view

class four(generics.RetrieveUpdateDestroyAPIView): #genarelly write 'RetrieveAPIView' but here RetrieveUpdateDestroyAPIView

    #permission\_classes = (permissions.IsAuthenticated,) # not show post,put,delete option

    queryset = cool.objects.all()                            #cause of red, update and delete post purpose

    serializer\_class = sos

    template\_name='base.html'

    permission\_classes = (IsAuthorOrReadOnly,) # new

### Model.py

from django.db import models

from django.contrib.auth.models import User #it use for ForeingKey purpose otherwise no need

class cool(models.Model):

    author = models.ForeignKey(User, on\_delete=models.CASCADE)

    title = models.CharField(max\_length=50)

    body = models.TextField()

    created\_at = models.DateTimeField(auto\_now\_add=True)

    updated\_at = models.DateTimeField(auto\_now=True)

    def \_\_str\_\_(self):

        return self.title

### Admin.py

from django.contrib import admin

from .models import cool

admin.site.register(cool)

### Serializers.py

from rest\_framework import serializers #need for rest

from .models import cool

class sos(serializers.ModelSerializer):

    class Meta:

        fields = ('id', 'author', 'title', 'body', 'created\_at',)

        model = cool

### Home.html

{% for cool in object\_list %}

 <li>{{cool.author}}</li>

 <li>{{cool.title}}</li>

 <li>{{cool.body}}</li>

 <li>{{cool.created\_at}}</li>

 <li>{{cool.updated\_at}}</li>

{% endfor %}

### Base.html

<li>{{cool.author}}</li>

 <li>{{cool.title}}</li>

 <li>{{cool.body}}</li>

 <li>{{cool.created\_at}}</li>

 <li>{{cool.updated\_at}}</li>

### Permissions .py

from rest\_framework import permissions

class IsAuthorOrReadOnly(permissions.BasePermission):

    def has\_object\_permission(self, request, view, obj):

# Read-only permissions are allowed for any request

        if request.method in permissions.SAFE\_METHODS:

            return True

# Write permissions are only allowed to the author of a post

        return obj.author == request.user

# Chapter\_7 authentication /login/logout

Reference project: project\_f

### Django rest frame work authentication classification

More authentication have in DRF.(basic, session, token, default, …..)

# Example\_project:1

Ey project a eksathe django,reast and react use kora hoyse.

### Install pakage:

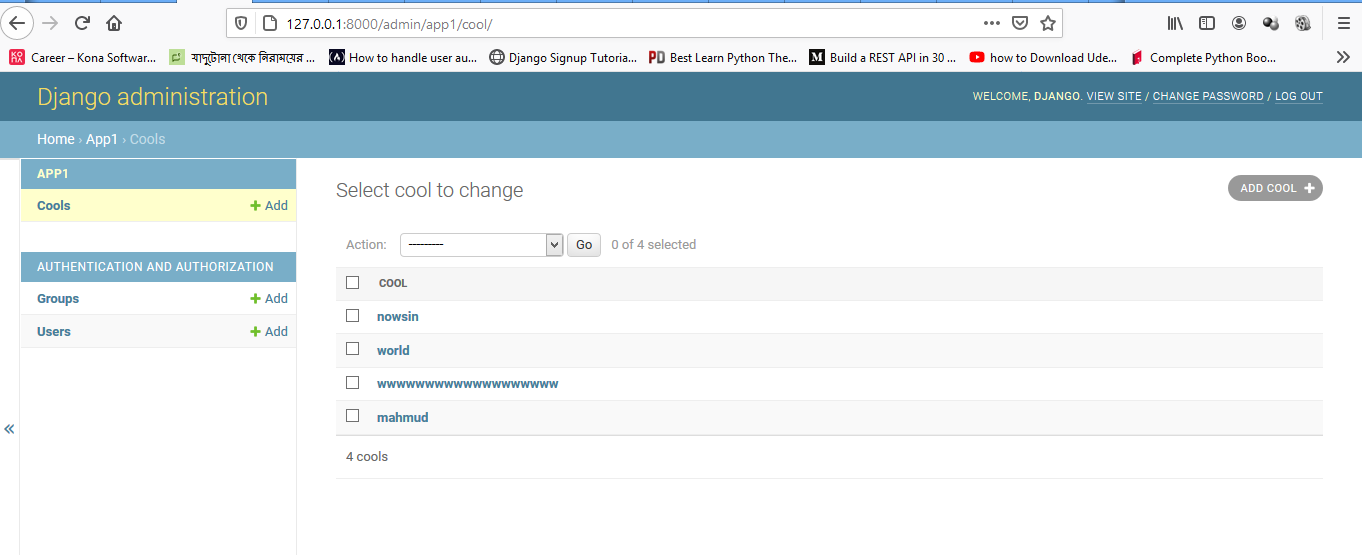
Above prject \_b, project\_d combine this project

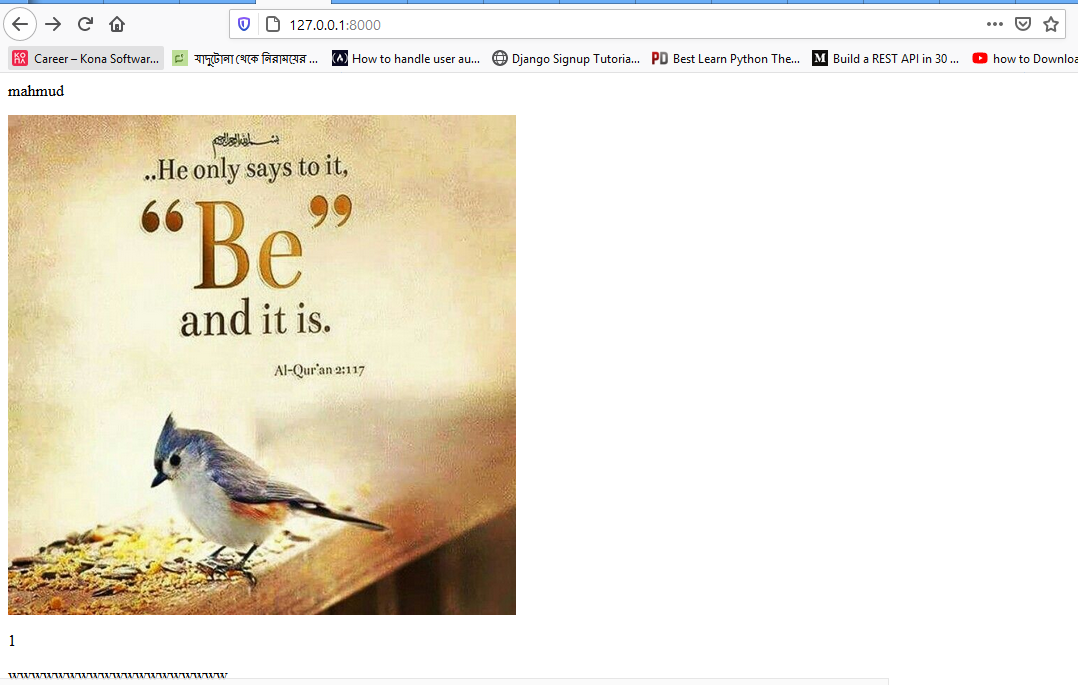
Pip install pillow # for image liabrary purpose

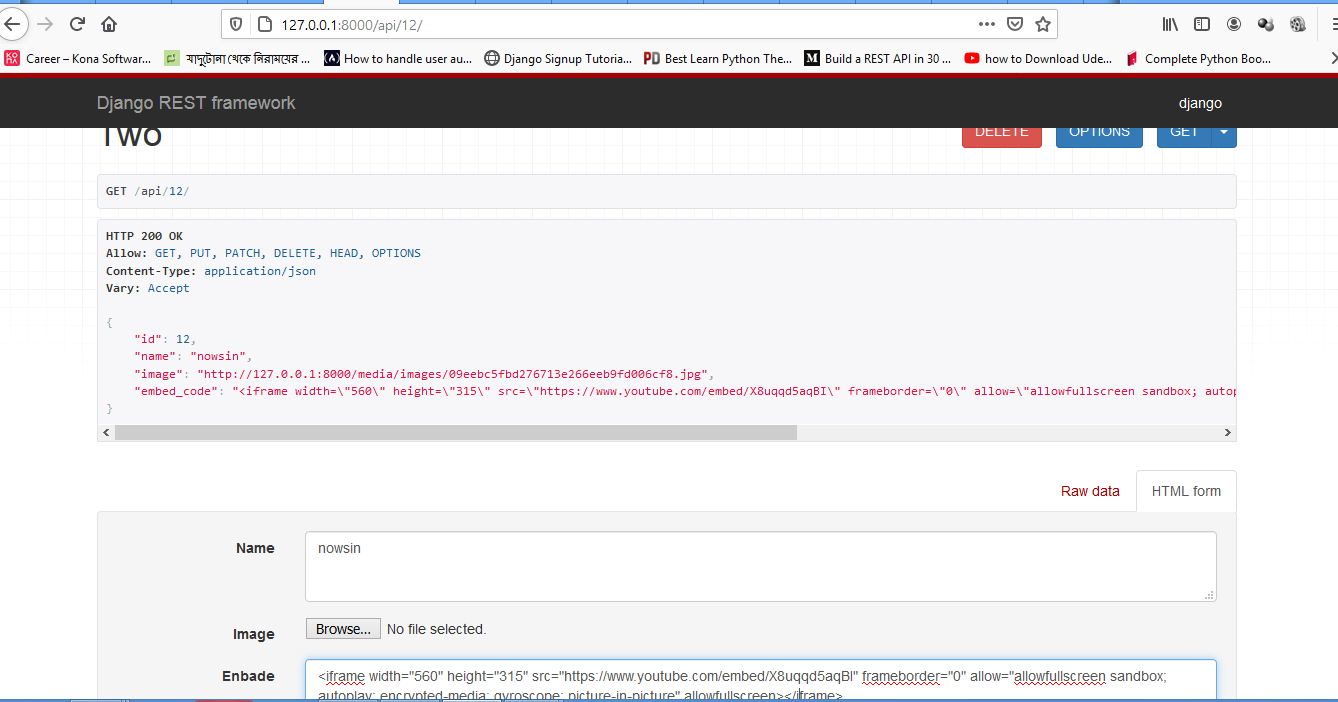
Pip install install djangorestframework

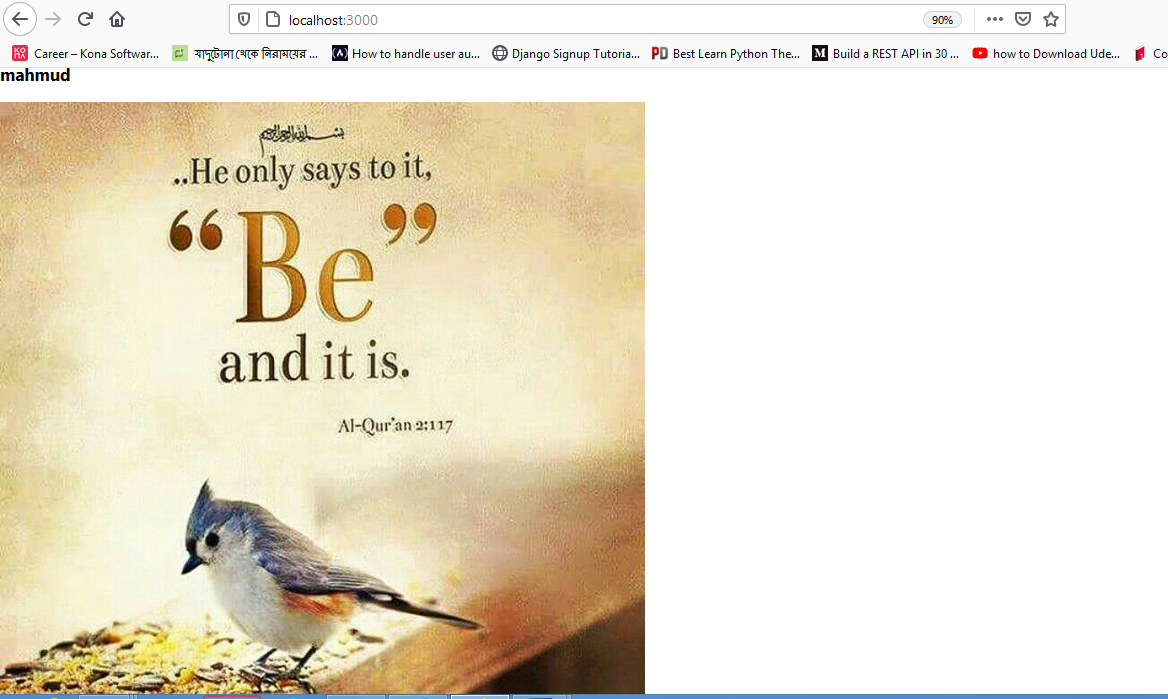
install django-cors-headers

1. First install node software form node.js website
2. After install check by CDM: node -v and npm –v
3. From CMD install : npm install –g create-react-app #eyta install korle react er all liabrary install hoy
4. From CMD create a react app : create-react-app Mahmud\_react #here is folder name, eta issa moto diya ji
5. Dir/ls command dile 2ta folder show korbe ,ekta django project er ,oporta react er
6. Cd Mahmud\_react #goto the react folder
7. Type npm start to run react page
8. Mone rakte hobe 2ta terminal open rakte hobe (cmd power shell for django page running and virtual stdio for react app er jonne)
9. Django runn takbe 8000 port a and react run hobe 3000 port a
10. CMD : npm install axios
11. Goto Mahmud\_react /src/APP.Js , open APP.JS file and edit , eta niche add kora hoyse









### Setting.py

INSTALLED\_APPS = [

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'app1',

    'rest\_framework',

    'corsheaders', #new

]

MIDDLEWARE = [

    'django.middleware.security.SecurityMiddleware',

    'django.contrib.sessions.middleware.SessionMiddleware',

    'corsheaders.middleware.CorsMiddleware', # new

    'django.middleware.common.CommonMiddleware',#new

    'django.middleware.csrf.CsrfViewMiddleware',

    'django.contrib.auth.middleware.AuthenticationMiddleware',

    'django.contrib.messages.middleware.MessageMiddleware',

    'django.middleware.clickjacking.XFrameOptionsMiddleware',

]

CORS\_ORIGIN\_WHITELIST = (

'http://localhost:3000',

#'http://localhost:8000',

)

STATIC\_URL = '/static/'

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

MEDIA\_URL = '/media/'

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

### Pro/url

from django.contrib import admin

from django.urls import path,include

from django.conf import settings # new

from django.conf.urls.static import static # new

urlpatterns = [

    path('admin/', admin.site.urls),

    path('',include('app1.urls'))

]

if settings.DEBUG:

    #urlpatterns += static(settings.STATIC\_URL, document\_root=settings.STATIC\_ROOT)

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

### Urls.py

from django.urls import path

from.views import one,two,three

urlpatterns = [

    path('',one.as\_view(),*name*='home'),

    path('api/',three.as\_view(),*name*='base'),

    path('api/<int:pk>/',two.as\_view(),*name*='base'),

]

### Views.py

from django.views.generic import DetailView,ListView

from rest\_framework import generics

from .models import cool

from.serializers import sos

*class* one(*ListView*):

    template\_name='home.html'

    model=cool

*class* three(*generics*.*ListCreateAPIView*):

    template\_name='base.html'

    queryset = cool.objects.all()

    serializer\_class = sos

*class* two(*generics*.*RetrieveUpdateDestroyAPIView*):

    queryset = cool.objects.all()

    serializer\_class = sos

    template\_name='base.html'

# Create your views here.

### Model.py

from django.db import models

*class* cool(*models*.*Model*):

    name=models.TextField()

    image = models.ImageField(*upload\_to*='images/')

    embed\_code = models.TextField("enbade")

*def* \_\_str\_\_(*self*):

        return self.name

# Create your models here.

### Admin.py

from django.contrib import admin

from.models import cool

admin.site.register(cool)

# Register your models here.

### Serializers.py

from rest\_framework import serializers

from app1.models import cool

*class* sos(*serializers*.*ModelSerializer*):

*class* Meta:

        model = cool

        fields = '\_\_all\_\_'

### Home.html

{% for cool in object\_list %}

<p>{{cool.name}}</p>

<img src="{{ cool.image.url }}" alt="mahmud"style="max-width: 1000px; max-height: 500px;">

<p>{{ cool.embed\_code | safe }}</p>

{% endfor %}

### Detail.html

{{cool.name}}

<img src="{{ cool.image.url }}" alt="mahmud"style="max-width: 1000px; max-height: 500px;"></img>

{{ cool.embed\_code | safe }}

### Mahud\_react/APP.JS

import React, { Component } from 'react';

import axios from 'axios'; // new

*class* App extends *Component* {

//default

state = {

//app1 is app name

app1: []

};

// new

componentDidMount() {

//cool is model name

this.getcool();

}

// new

getcool() {

axios

//rest api page link

.get('http://127.0.0.1:8000/api/')

//jason api page link

//.get('http://127.0.0.1:8000/api/?format=json')

.then(*res* *=>* {

this.setState({app1: res.data });

})

.catch(*err* *=>* {

*console*.log(err);

});

}

render() {

return (

  <div>

{this.state.app1.map(*item* *=>* (

//default dite hobe

<div key={item.id}>

<h3>{item.name}</h3>

<p><img src={ item.image } alt="mahmud"></img></p>

//<p>{ item.embed\_code}</p> //not work

</div>

))}

</div>

);

}

}

export default App;