**Student Record Documentation**

**Open command prompt**

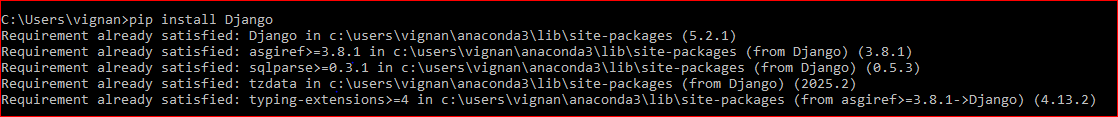
**Python --version**

It displays the **currently installed version of Python**.



**pip install Django**

It installs the **Django** web framework using Python's package manager, pip



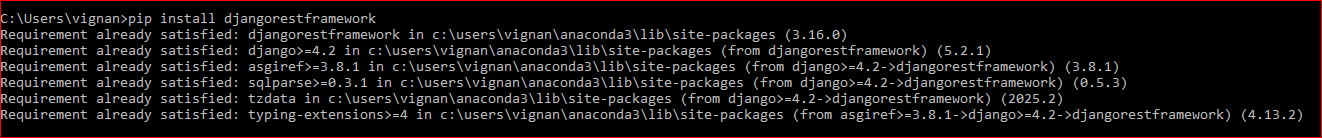
**django-admin –version**

This command checks the currently installed version of Django on your system



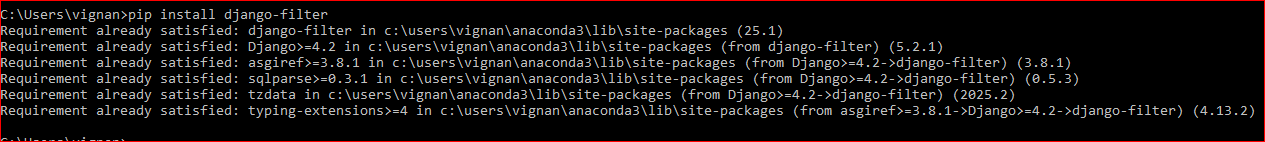
**pip install djangorestframework**

This installs **Django REST Framework (DRF)**, a powerful and flexible toolkit for building Web APIs in Django.



**pip install django-filter**

This installs **django-filter**, a powerful library used with **Django REST Framework** to easily add filtering capabilities to API views.



**django-admin startproject studentrecord**

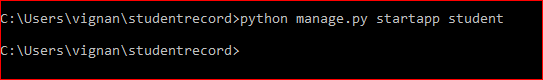
This command **creates a new Django project** named studentrecord.



**cd studentrecord**

It changes the current directory to the newly created Django project folder named studentrecord

**python manage.py startapp tasks**



It creates a new Django **application** named tasks inside your project directory. This is where you'll define the logic and models related to the task management functionality.

**Again add the file studentrecord in pycharm and continue doing project**

**In setting files add this**

INSTALLED\_APPS = [ at last

"rest\_framework",

"django\_filters",

"students",

**In urls.py file add this**

from django.contrib import admin  
from django.urls import path, include  
  
urlpatterns = [  
 path('admin/', admin.site.urls),  
 path('', include('students.urls')),  
]

**Create template new directory and in that create another directory tasks and then add html file codes**

**In settins.py under allowed host add this**

REST\_FRAMEWORK = {

"DEFAULT\_PAGINATION\_CLASS":"rest\_framework.pagination.PageNumberPagination",

"pagesize":2,

"DEFAULT\_FILTER\_BACKEND":[

"django\_filtres.rest\_framework.DjangoFiltreBackend",

"rest\_framework.filtres.orderingFiltre"''

"rest\_framework.filtres.SearchFiltre",

]

"DeFAULT\_VERSIONING\_CLASS":"rest-framework.versioning.NamespaceVersioning",

}

**Under models add this**

from django.db import models  
  
class Student(models.Model):  
 objects = None  
 name = models.CharField(max\_length=100)  
 age = models.IntegerField()  
 student\_class = models.CharField(max\_length=20)  
 roll\_no = models.IntegerField()  
 marks = models.FloatField()  
  
 def \_\_str\_\_(self):  
 return f"{self.name} - {self.roll\_no}"

**create new python file urls.py in app (i.e tasks) add this**  
from django.urls import path  
from . import views  
  
urlpatterns = [  
 path('', views.student\_list, name='student\_list'),  
 path('create/', views.student\_create, name='student\_create'),  
 path('update/<int:pk>/', views.student\_update, name='student\_update'),  
 path('delete/<int:pk>/', views.student\_delete, name='student\_delete'),  
]

**Create new file SERIALZERSSSS**

from rest\_framework import serializers

from .models import Task

class TaskSerializer(serializers.ModelSerializer):

class Meta:

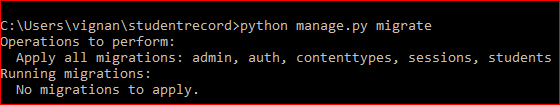
model = Task

fields = '\_\_all\_\_'

**NOW AGAIN RUN commands**

**manage.py migrate**

Applies pending migrations, updates database tables, and syncs the schema with your Django models.



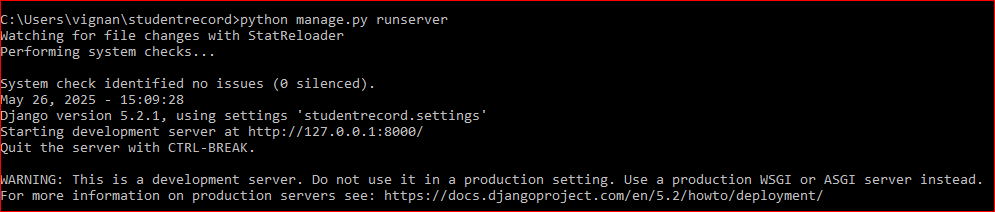
**Python manage.py make migrations**

**creates new migration files** based on changes detected in your Django models.

These migration files describe how to alter the database schema (add/remove/change tables or fields).



**Python manage.py runserver**

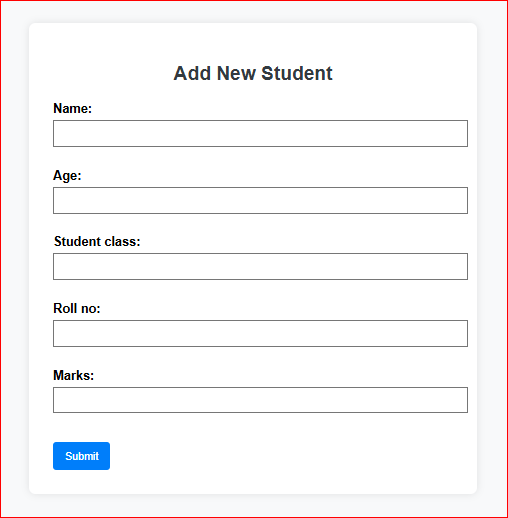
Runs your Django app locally, letting you test it in a browser. The server auto-reloads on code changes so you see updates instantly without restarting.

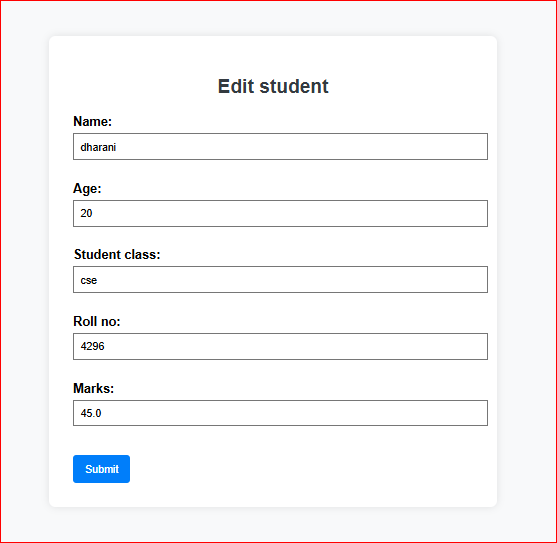
**Now link will be generated**

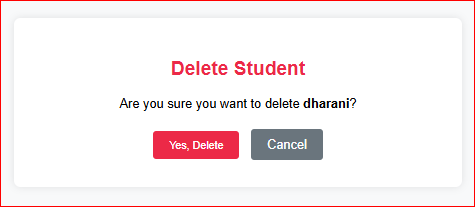
<http://127.0.0.1:8000/>

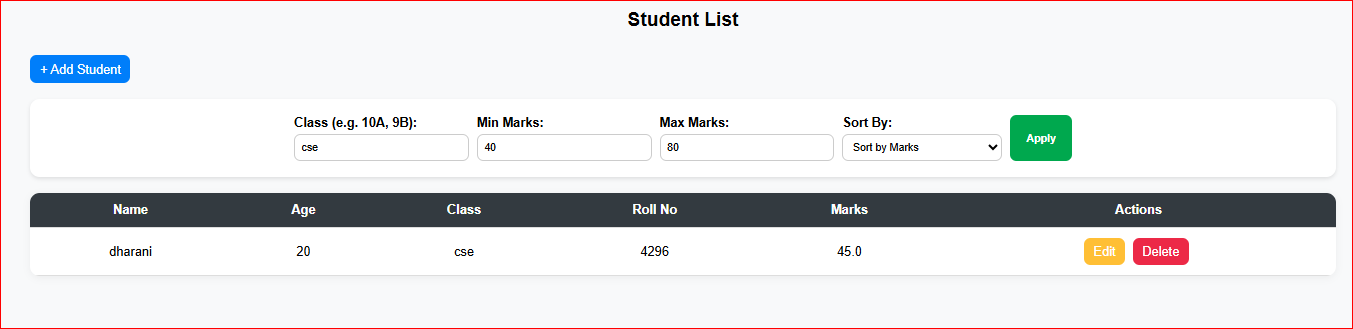
**After opening link you will be able to see project**

**Output:-**









**Structure:**

