Web based OTT platform

Python Mini Project
Name of students:

- 60004190042 – Hartik Suhagiya
- 60004190057 – Junaid Girkar
- 60004190058 – Kanaad Deshpande

Abstract
The Web-based OTT platform that we have developed - the netflix clone – is a project in which we have used many popular libraries and languages to make an impact in market. On the frontend, we have Reactjs – a popular JavaScript library with HTML for component rendering. CSS and styled components are used for styling our app. For the APIs, Django Rest Framework is used. We have also used swagger docs to display our API endpoints. On the backend, vanilla Django is used to connect to SQLite3 database. The movies and episodes in seasons of a particular series seen in the web page are inserted from Django's admin panel. Some features of this app include authentication (registration, login and forgot password), and video rendering for users to watch movies. A variety of recommended movies and series is shown in carousels for users.

Hardware and Software used:
Hardware:
Processor :Processor Intel(R) Core(TM) i5-3320M CPU @ 2.60GHz, 2601 Mhz, 2 Core(s), 4 Logical Processor(s)
RAM : 8GB
ROM : 256GB

Software:
Code editor: VSCode, JetBrains' PyCharm

Technologies used:
Backend: Django, Django rest framework, Swagger docs with Django Rest Framework
Database: db.sqlite3
Frontend: HTML, CSS, Javascript - React.js, Styled Components (Styling library)

Code on Github:
Frontend: https://github.com/HSJGKD-Python-Mini-Project/Frontend
Backend: https://github.com/HSJGKD-Python-Mini-Project/Backend

Folder Structure
Backend:

```
Folder PATH listing for volume New Volume
Volume serial number is 4A5A-F313
D:.
├   .env
```

```
├   .gitignore
├   db.sqlite3
├   manage.py
├   README.md
├   requirements.txt
├   tree.txt
├
├────.vscode
│       settings.json
│
├────accounts
├   ├   admin.py
├   ├   apps.py
├   ├   models.py
├   ├   serializers.py
├   ├   tests.py
├   ├   urls.py
├   ├   views.py
├   ├   __init__.py
├   ├
├   ├────migrations
├   ├   ├   0001_initial.py
├   ├   ├   __init__.py
├   ├   ├
├   ├   ├────__pycache__
├   ├   │           0001_initial.cpython-37.pyc
├   ├   │           __init__.cpython-37.pyc
├   ├   │
├   ├────__pycache__
│           admin.cpython-37.pyc
│           apps.cpython-37.pyc
│           models.cpython-37.pyc
│           serializers.cpython-37.pyc
│           urls.cpython-37.pyc
│           views.cpython-37.pyc
│           __init__.cpython-37.pyc
│
├────media
├   ├────episodes
├   ├       Episode_1_-_Pilot.mkv
├   ├       Episode_1_-_The_Dundies.mkv
├   ├       Episode_1_-_The_Dundies_ELX2DkI.mkv
├   ├       Episode_2_-_Diversity_Day.mkv
├   ├       Episode_2_-_Sexual_Harassment.mkv
├   ├       Episode_3_-_Health_Care.mkv
```

```
│   │       Episode_4_-_The_Alliance.mkv
│   │       Episode_5_-_Basketball.mkv
│   │       Episode_6_-_Hot_Girl.mkv
│   │       SampleVideo_1280x720_10mb.mp4
│   │       SampleVideo_1280x720_10mb_mFuiuwW.mp4
│   │
│   ├───files
│   │       ACM_101_-_Exploration_1.pdf
│   │       ACM_101_-_Exploration_1_c0LCpJE.pdf
│   │       SampleVideo_1280x720_10mb.mp4
│   │
│   ├───movies
│   │   │   bunny.mp4
│   │   │   bunny_f29zuZM.mp4
│   │   │   bunny_FMkIknb.mp4
│   │   │   bunny_iKwOJFU.mp4
│   │   │   bunny_JTrP9sU.mp4
│   │   │   bunny_w18pCvy.mp4
│   │   │   Joker.2019.720p.BluRay.x264-YTS.LT.mp4
│   │   │   Joker.2019.720p.BluRay.x264-YTS.LT_FoKZZFl.mp4
│   │   │   SampleVideo_1280x720_10mb.mp4
│   │   │   SampleVideo_1280x720_10mb_IyR5uPU.mp4
│   │   │   SampleVideo_1280x720_10mb_obS4pll.mp4
│   │   │   SampleVideo_1280x720_10mb_VvJZt3c.mp4
│   │   │
│   │   └───posters
│   │           101_cover.jpg
│   │           101_cover_2bxElNH.jpg
│   │           101_cover_KQNoTyE.jpg
│   │           101_cover_KzRJy5k.jpg
│   │           small.jpg
│   │           small_DxaRq52.jpg
│   │           small_F3jTizS.jpg
│   │           small_GPAbTox.jpg
│   │           small_kLtbI5Q.jpg
│   │           small_m5o94w9.jpg
│   │
│   └───series
│       └───bgi
│               large.jpg
│               large_1OgXBvM.jpg
│               large_APZhHYE.jpg
│               large_IfzlZeH.jpg
│               large_vD06p4R.jpg
│
```

```
|       ````posters
|               large.jpg
|               small.jpg
|               small_0NBlBvx.jpg
|               small_5OlIYCC.jpg
|               small_H0w6rgm.jpg
|               small_uY4FdWu.jpg
|               small_YAM7hjG.jpg
|
````myapp
|   |   admin.py
|   |   apps.py
|   |   models.py
|   |   recSys.py
|   |   serializers.py
|   |   tests.py
|   |   urls.py
|   |   views.py
|   |   __init__.py
|   |
|   ````migrations
|   |   |   0001_initial.py
|   |   |   0002_auto_20211204_1101.py
|   |   |   0003_auto_20211204_1107.py
|   |   |   0004_series_series_background_image.py
|   |   |   __init__.py
|   |   |
|   |   ````__pycache__
|   |           0001_initial.cpython-37.pyc
|   |           0002_auto_20211204_1101.cpython-37.pyc
|   |           0003_auto_20211204_1107.cpython-37.pyc
|   |           0004_series_series_background_image.cpython-37.pyc
|   |           __init__.cpython-37.pyc
|   |
|   ````__pycache__
|           admin.cpython-37.pyc
|           apps.cpython-37.pyc
|           models.cpython-37.pyc
|           recSys.cpython-37.pyc
|           serializers.cpython-37.pyc
|           urls.cpython-37.pyc
|           views.cpython-37.pyc
|           __init__.cpython-37.pyc
|
````Streaming_Platform
```

```
    ├   asgi.py
    ├   settings.py
    ├   urls.py
    ├   wsgi.py
    ├   __init__.py
    ├
    ├───__pycache__
            settings.cpython-37.pyc
            urls.cpython-37.pyc
            wsgi.cpython-37.pyc
            __init__.cpython-37.pyc
```

Frontend:

```
Folder PATH listing for volume New Volume
Volume serial number is 4A5A-F313
D:.
├   .gitignore
├   package-lock.json
├   package.json
├   README.md
├   tree.txt
├   yarn.lock
├
├───.vscode
│       settings.json
├
├───public
│   ├   index.html
│   ├
│   ├───images
│   ├   ├───films
│   ├   ├   ├   ├───children
│   ├   ├   ├   ├   ├───despicable-me
│   ├   ├   ├   ├   ├       large.jpg
│   ├   ├   ├   ├   ├       small.jpg
│   ├   ├   ├   ├   ├
│   ├   ├   ├   ├   ├───frozen
│   ├   ├   ├   ├   ├       large.jpg
│   ├   ├   ├   ├   ├       small.jpg
│   ├   ├   ├   ├   ├
│   ├   ├   ├   ├   ├───hotel-transylvania
│   ├   ├   ├   ├   ├       large.jpg
│   ├   ├   ├   ├   ├       small.jpg
```

```
                    
                    ├── spirited-away
                    │   ├── large.jpg
                    │   └── small.jpg
                    ├── up
                    │   ├── large.jpg
                    │   └── small.jpg
                    
            ├── drama
            │   ├── fight-club
            │   │   ├── large.jpg
            │   │   └── small.jpg
            │   ├── kings-speech
            │   │   ├── large.jpg
            │   │   └── small.jpg
            │   ├── the-prestige
            │   │   ├── large.jpg
            │   │   └── small.jpg
            │   ├── the-revenant
            │   │   ├── large.jpg
            │   │   └── small.jpg
            │   ├── the-social-network
            │   │   ├── large.jpg
            │   │   └── small.jpg
            
            ├── romance
            │   ├── a-star-is-born
            │   │   ├── large.jpg
            │   │   └── small.jpg
            │   ├── blue-valentine
            │   │   ├── large.jpg
            │   │   └── small.jpg
            │   ├── la-la-land
            │   │   ├── large.jpg
            │   │   └── small.jpg
            │   ├── the-notebook
            │   │   ├── large.jpg
```

```
│   │   │   │   │       small.jpg
│   │   │   │   │
│   │   │   │   ├────titanic
│   │   │   │   │           large.jpg
│   │   │   │   │           small.jpg
│   │   │   │   │
│   │   │   ├────suspense
│   │   │   │   ├────gone-girl
│   │   │   │   │       large.jpg
│   │   │   │   │       small.jpg
│   │   │   │   │
│   │   │   │   ├────prisoners
│   │   │   │   │       large.jpg
│   │   │   │   │       small.jpg
│   │   │   │   │
│   │   │   │   ├────seven
│   │   │   │   │       large.jpg
│   │   │   │   │       small.jpg
│   │   │   │   │
│   │   │   │   ├────shutter-island
│   │   │   │   │       large.jpg
│   │   │   │   │       small.jpg
│   │   │   │   │
│   │   │   │   ├────zodiac
│   │   │   │   │           large.jpg
│   │   │   │   │           small.jpg
│   │   │   │   │
│   │   │   ├────thriller
│   │   │       ├────a-quiet-place
│   │   │       │       large.jpg
│   │   │       │       small.jpg
│   │   │       │
│   │   │       ├────black-swan
│   │   │       │       large.jpg
│   │   │       │       small.jpg
│   │   │       │
│   │   │       ├────joker
│   │   │       │       large.jpg
│   │   │       │       small.jpg
│   │   │       │
│   │   │       ├────nightcrawler
│   │   │       │       large.jpg
│   │   │       │       small.jpg
│   │   │       │
│   │   │       ├────the-silence-of-the-lambs
```

```
│   │   │               large.jpg
│   │   │               small.jpg
│   │   │
│   │   ├───icons
│   │   │       add.png
│   │   │       add.svg
│   │   │       chevron-right.png
│   │   │       chevron-right.svg
│   │   │       close-slim.png
│   │   │       close.png
│   │   │       HomePage.jpg
│   │   │       search.png
│   │   │
│   │   ├───misc
│   │   │       child-friendly.jpg
│   │   │       home-bg.jpg
│   │   │       home-imac.jpg
│   │   │       home-mobile.jpg
│   │   │       home-tv.jpg
│   │   │       joker1.jpg
│   │   │       loading.gif
│   │   │       spinner.png
│   │   │
│   │   └───series
│   │       └───children
│   │           ├───arthur
│   │           │       large.jpg
│   │           │       small.jpg
│   │           │
│   │           ├───dora-the-explorer
│   │           │       large.jpg
│   │           │       small.jpg
│   │           │
│   │           ├───paw-patrol
│   │           │       large.jpg
│   │           │       small.jpg
│   │           │
│   │           ├───peppa-pig
│   │           │       large.jpg
│   │           │       small.jpg
│   │           │
│   │           └───spongebob
│   │                   large.jpg
│   │                   small.jpg
│   │
```

```
├   ├   ├   ├── comedies
├   ├   ├   ├   ├── arrested-development
├   ├   ├   ├   ├       large.jpg
├   ├   ├   ├   ├       small.jpg
├   ├   ├   ├   ├
├   ├   ├   ├   ├── curb-your-enthusiasm
├   ├   ├   ├   ├       large.jpg
├   ├   ├   ├   ├       small.jpg
├   ├   ├   ├   ├
├   ├   ├   ├   ├── family-guy
├   ├   ├   ├   ├       large.jpg
├   ├   ├   ├   ├       small.jpg
├   ├   ├   ├   ├
├   ├   ├   ├   ├── south-park
├   ├   ├   ├   ├       large.jpg
├   ├   ├   ├   ├       small.jpg
├   ├   ├   ├   ├
├   ├   ├   ├   ├── the-office
├   ├   ├   ├           large.jpg
├   ├   ├   ├           small.jpg
├   ├   ├   ├
├   ├   ├   ├── crime
├   ├   ├   ├   ├── long-shot
├   ├   ├   ├   ├       large.jpg
├   ├   ├   ├   ├       small.jpg
├   ├   ├   ├   ├
├   ├   ├   ├   ├── making-a-murderer
├   ├   ├   ├   ├       large.jpg
├   ├   ├   ├   ├       small.jpg
├   ├   ├   ├   ├
├   ├   ├   ├   ├── the-confession-killer
├   ├   ├   ├   ├       large.jpg
├   ├   ├   ├   ├       small.jpg
├   ├   ├   ├   ├
├   ├   ├   ├   ├── the-innocent-man
├   ├   ├   ├   ├       large.jpg
├   ├   ├   ├   ├       small.jpg
├   ├   ├   ├   ├
├   ├   ├   ├   ├── the-staircase
├   ├   ├   ├           large.jpg
├   ├   ├   ├           small.jpg
├   ├   ├   ├
├   ├   ├   ├── documentaries
├   ├   ├   ├   ├── amanda-knox
├   ├   ├   ├   ├       large.jpg
```

```
│   │   │   │   │       small.jpg
│   │   │   │   │
│   │   │   │   ├────citizenfour
│   │   │   │   │       large.jpg
│   │   │   │   │       small.jpg
│   │   │   │   │
│   │   │   │   ├────man-on-wire
│   │   │   │   │       large.jpg
│   │   │   │   │       small.jpg
│   │   │   │   │
│   │   │   │   ├────super-size-me
│   │   │   │   │       large.jpg
│   │   │   │   │       small.jpg
│   │   │   │   │
│   │   │   │   ├────tiger-king
│   │   │   │   │       large.jpg
│   │   │   │   │       small.jpg
│   │   │   │   │
│   │   │   │   ├────feel-good
│   │   │   │   │   ├────forrest-gump
│   │   │   │   │   │       large.jpg
│   │   │   │   │   │       small.jpg
│   │   │   │   │   │
│   │   │   │   │   ├────good-will-hunting
│   │   │   │   │   │       large.jpg
│   │   │   │   │   │       small.jpg
│   │   │   │   │   │
│   │   │   │   │   ├────juno
│   │   │   │   │   │       large.jpg
│   │   │   │   │   │       small.jpg
│   │   │   │   │   │
│   │   │   │   │   ├────midnight-in-paris
│   │   │   │   │   │       large.jpg
│   │   │   │   │   │       small.jpg
│   │   │   │   │   │
│   │   │   │   │   └────school-of-rock
│   │   │   │   │           large.jpg
│   │   │   │   │           small.jpg
│   │   │   │   │
│   │   │   └────users
│   │   │           1.png
│   │   │           2.png
│   │   │           3.png
│   │   │           4.png
│   │   │           5.png
```

```
│   │
│   └───videos
│           bunny.mp4
│
└───src
    │   App.js
    │   globalStyles.js
    │   index.js
    │   logo.svg
    │
    └───components
    │   │   index.js
    │   │
    │   └───accordion
    │   │   │   index.js
    │   │   │
    │   │   └───styles
    │   │           accordion.js
    │   │
    │   └───carousel
    │   │   │   index.js
    │   │   │
    │   │   └───styles
    │   │           carousel.js
    │   │
    │   └───feature
    │   │   │   index.js
    │   │   │
    │   │   └───styles
    │   │           feature.js
    │   │
    │   └───footer
    │   │   │   index.js
    │   │   │
    │   │   └───styles
    │   │           footer.js
    │   │
    │   └───form
    │   │   │   index.js
    │   │   │
    │   │   └───styles
    │   │           form.js
    │   │
    │   └───header
    │   │   │   index.js
```

```
    ?   ?   ?
    ?   ?   ????styles
    ?   ?           header.js
    ?   ?
    ?   ????jumbotron
    ?   ?   ?   index.js
    ?   ?   ?
    ?   ?   ????styles
    ?   ?           jumbotron.js
    ?   ?
    ?   ????movie-container
    ?   ?   ?   index.js
    ?   ?   ?
    ?   ?   ????styles
    ?   ?           movie-container.js
    ?   ?
    ?   ????opt-form
    ?       ?   index.js
    ?       ?
    ?       ????styles
    ?               opt-form.js
    ?
    ????constants
    ?       routes.js
    ?
    ????containers
    ?       browse.js
    ?       faqs.js
    ?       footer.js
    ?       header.js
    ?       jumbotron.js
    ?       movie-container.js
    ?       movies.js
    ?       profiles.js
    ?
    ????fixtures
    ?   ?   faqs.json
    ?   ?   jumbo.json
    ?   ?   movies.json
    ?   ?   series.json
    ?   ?
    ?   ????films
    ?           comingThisWeek.json
    ?           forYou.json
    ?           newReleases.json
```

```
    ?           popularOnNetflix.json
    ?           recentlyWatched.json
    ?           trendingNow.json
    ?
    ????helpers
    ?       routes.js
    ?
    ????hooks
    ?       useToken.js
    ?
    ????pages
            browse.js
            home.js
            index.js
            movie.css
            movie.js
            movies.js
            seasonepisode.js
            seasons.js
            series.js
            signin.js
            signup.js
```

Code:
Backend:
requirements.txt

```
asgiref==3.4.1
certifi==2021.10.8
charset-normalizer==2.0.9
coreapi==2.3.3
coreschema==0.0.4
Django==3.2.9
django-cors-headers==3.10.0
django-rest-swagger==2.2.0
djangorestframework==3.12.4
idna==3.3
itypes==1.2.0
Jinja2==3.0.3
MarkupSafe==2.0.1
openapi-codec==1.3.2
Pillow==8.4.0
pytz==2021.3
requests==2.26.0
simplejson==3.17.6
```

```
sqlparse==0.4.2
uritemplate==4.1.1
urllib3==1.26.7
```

accounts app:
admin.py

```
from django.contrib import admin
from django import forms
from django.contrib.auth.forms import ReadOnlyPasswordHashField
from django.contrib.auth.admin import UserAdmin as BaseUserAdmin

from .models import User

class UserAdminCreationForm(forms.ModelForm):
    """
    A form for creating new users.
    """
    first_name = forms.CharField(max_length=64)
    last_name = forms.CharField(max_length=64)
    password1 = forms.CharField(label='Password', widget=forms.PasswordInput)
    password2 = forms.CharField(label='Password confirmation',
widget=forms.PasswordInput)

    class Meta:
        model = User
        fields = ('email', 'first_name', 'last_name')

    def clean_password2(self):
        # Check that the two password entries match
        password1 = self.cleaned_data.get("password1")
        password2 = self.cleaned_data.get("password2")
        if password1 and password2 and password1 != password2:
            raise forms.ValidationError("Passwords don't match")
        return password2

    def save(self, commit=True):
        # Save the provided password in hashed format
        user = super(UserAdminCreationForm, self).save(commit=False)
        user.set_password(self.cleaned_data["password1"])
        if commit:
            user.save()
        return user
class UserAdminChangeForm(forms.ModelForm):
```

```
    """A form for updating users. Includes all the fields on
    the user, but replaces the password field with admin's
    password hash display field.
    """
    password = ReadOnlyPasswordHashField()

    class Meta:
        model = User
        fields = ('email', 'password', 'first_name', 'last_name', 'active',
'admin')

    def clean_password(self):
        return self.initial["password"]

class UserAdmin(BaseUserAdmin):
    # The forms to add and change user instances
    form = UserAdminChangeForm
    add_form = UserAdminCreationForm

    # The fields to be used in displaying the User model.
    # These override the definitions on the base UserAdmin
    # that reference specific fields on auth.User.
    list_display = ('email', 'first_name', 'last_name', 'admin')
    list_filter = ('admin',)
    fieldsets = (
        (None, {'fields': ('email', 'password')}),
        ('Personal info', {'fields': ('first_name', 'last_name')}),
        ('Permissions', {'fields': ('admin', 'staff', 'active')}),
    )
    # add_fieldsets is not a standard ModelAdmin attribute. UserAdmin
    # overrides get_fieldsets to use this attribute when creating a user.
    add_fieldsets = (
        (None, {
            'classes': ('wide',),
            'fields': ('email', 'first_name', 'last_name', 'password1',
'password2')}
        ),
    )
    search_fields = ('email', 'first_name', 'last_name')
    ordering = ('email',)
    filter_horizontal = ()

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

models.py

```
from django.db import models
from django.conf import settings
from django.db.models.signals import post_save
from django.contrib.auth.models import AbstractBaseUser, BaseUserManager

from rest_framework.authtoken.models import Token

class UserManager(BaseUserManager):
    def create_user(self, email, first_name, last_name, password=None, **kwargs):
        if not email:
            raise ValueError("Users must have an email address")
        user = self.model(
            email=self.normalize_email(email),
            first_name=first_name,
            last_name=last_name,
            **kwargs
        )
        user.set_password(password)
        user.save(using=self._db)
        return user
    def create_staffuser(self, email, first_name, last_name, password=None):
        user = self.create_user(email, first_name, last_name, password)
        user.staff = True
        user.save(using=self._db)
        return user
    def create_superuser(self, email, first_name, last_name, password):
        user = self.create_user(email, first_name, last_name, password)
        user.staff = True
        user.admin = True
        user.save(using=self._db)
        return user

class User(AbstractBaseUser):
    email       = models.EmailField(max_length=255, unique=True, default="")
    first_name  = models.CharField(max_length=64, default="")
    last_name   = models.CharField(max_length=64, default="")

    active      = models.BooleanField(default=True)
    staff       = models.BooleanField(default=False)
    admin       = models.BooleanField(default=False)

    objects     = UserManager()
    USERNAME_FIELD  = 'email'
    REQUIRED_FIELDS = ['first_name', 'last_name']
```

```
    def __str__(self):
        return self.email
    def has_perm(self, perm, obj=None):
        return True
    def has_module_perms(self, app_label):
        return True
    @property
    def is_staff(self):
        return self.staff
    @property
    def is_admin(self):
        return self.admin
    @property
    def is_active(self):
        return self.active

def save_user(instance, created, **kwargs):
    if created:
        Token.objects.create(user=instance)

post_save.connect(save_user, sender=settings.AUTH_USER_MODEL)
```

serializers.py

```
from rest_framework import serializers
from accounts.models import User
from rest_framework.authtoken.models import Token

class UserSerializer(serializers.ModelSerializer):
    class Meta:
        model = User
        fields = [
            "id",
            "email",
            "first_name",
            "last_name",
            "active",
            "staff",
            "admin",
        ]

class LoginSerializer(serializers.ModelSerializer):
    token = serializers.SerializerMethodField()
    def get_token(self, obj):
```

```
        return Token.objects.get(user=obj).key
    class Meta:
        model = User
        fields = ['id', 'token', 'email', 'first_name', 'last_name', 'password']

class SignupSerializer(serializers.ModelSerializer):
    class Meta:
        model = User
        fields = ['email', 'first_name', 'last_name', 'password']
        extra_kwargs = {'password': {'write_only': True}}
    def create(self, validated_data):
        user = User.objects.create_user(**validated_data)
        user.save()
        return user
```

urls.py

```
from django.urls import path
from .views import LoginView, SignupView, ForgotPasswordEmail,
ForgotPasswordReset
urlpatterns = [
    path('login/', LoginView.as_view()),
    path('signup/', SignupView.as_view()),
    path('forgot-password-email/', ForgotPasswordEmail.as_view()),  #Send email
with link to reset password
    path('forgot-password-reset/', ForgotPasswordReset.as_view()),
]
```

views.py

```
from django.shortcuts import render
from django.contrib.auth import authenticate
from django.core.mail import send_mail
from django.conf import settings

from rest_framework.views import APIView
from rest_framework.permissions import AllowAny
from rest_framework.response import Response
from rest_framework import status, serializers
from rest_framework.authtoken.models import Token


from .models import User
from .serializers import LoginSerializer, SignupSerializer
```

```python
# Create your views here.
class LoginView(APIView):
    permission_classes = [AllowAny]
    serializer_class = LoginSerializer

    def post(self, request, format=None):
        user = authenticate(username=request.data['email'],
password=request.data['password'])
        if user is not None:
            serializer = self.serializer_class(user)
            return Response(serializer.data, status=status.HTTP_200_OK)
        return Response({"token": None, "message": "Invalid Credentials"},
status=status.HTTP_401_UNAUTHORIZED)

class SignupView(APIView):
    permission_classes = [AllowAny]
    serializer_class = SignupSerializer
    def post(self, request, format=None):
        serializer = self.serializer_class(data=request.data)
        serializer.is_valid(raise_exception=True)
        user = serializer.save()
        return Response(serializer.data, status=status.HTTP_201_CREATED)

class ForgotPasswordEmail(APIView):
    '''
    Send email with link to reset password
    '''
    permission_classes = [AllowAny]

    def post(self, request, format=None):
        try:
            user = User.objects.get(email=request.data['email'])
            token = Token.objects.get(user=user)
        except:
            return Response("You are not a registered user",
status=status.HTTP_404_NOT_FOUND)

        send_mail(
            subject="Did you forget your password?",
            message="Please use the below link to reset your password \n\n" +
str(settings.DOMAIN_NAME) + "/forgot-password/" + str(token),
            from_email=settings.EMAIL_HOST_USER,
            recipient_list=[request.data['email']]
        )
```

```
        #test will be replaced with appropriate frontend url which will provide a
form with new password
        #send token in message to identify user after they've filled the form
        #reset password and token after form submit
        return Response({}, status=status.HTTP_200_OK)

class ForgotPasswordReset(APIView):
    '''
    Reset password from token (obtained through the link in forgot-password-
email)
    '''
    permission_classes = [AllowAny]

    def post(self, request, format=None):
        token = Token.objects.filter(key=request.data['token'])
        if token.exists():
            token = token.first()
            user = token.user
            user.set_password(request.data['new_password'])
            user.save()
            token.delete()
            token = Token.objects.create(user=user)
            return Response({'success':True, 'message': 'Successfully changed
password', 'token': token.key}, status.HTTP_200_OK)
        return Response({'success': False, 'message': 'Could not password,
invalid token'}, status.HTTP_400_BAD_REQUEST)
```

myapp:
admin.py

```
from django.contrib import admin
from .models import *


class EpisodeAdmin(admin.ModelAdmin):
    list_display = ['id','title', 'uploaded', 'modified', 'video_file']
admin.site.register(Episode, EpisodeAdmin)

class SeasonsAdmin(admin.ModelAdmin):
    list_display = ['id', 'season_number', 'release_year', 'number_of_episodes',
'get_episodes']

    def get_episodes(self, obj):
        return ", ".join([p.title for p in obj.episodes.all()])
admin.site.register(Seasons, SeasonsAdmin)
```

```
class SeriesAdmin(admin.ModelAdmin):
    list_display = ['id','series_poster','name','series_category', 'get_seasons']
    def get_seasons(self, obj):
        return ", ".join([str(p.season_number) for p in obj.seasons.all()])
admin.site.register(Series, SeriesAdmin)

class MovieAdmin(admin.ModelAdmin):
    list_display = ['id', 'movie_name', 'movie_category', 'movie_poster',
'year_released', 'description', 'imdb_rating', 'video_file']
admin.site.register(Movies, MovieAdmin)

class CategoryAdmin(admin.ModelAdmin):
    list_display = ['id', 'name']
admin.site.register(Category, CategoryAdmin)
```

models.py

```
from django.db import models

# Create your models here.
class Category(models.Model):
    name = models.CharField(blank=True, null=True, max_length=255)

    def __str__(self):
        return str(self.id) +'-' +self.name

class Movies(models.Model):
    movie_name = models.CharField(max_length=255, blank=True)
    movie_category = models.ForeignKey(Category, on_delete=models.CASCADE)
    movie_poster = models.ImageField(blank=True, upload_to='movies/posters')
    year_released = models.IntegerField(blank=True)
    description = models.TextField(blank=True)
    imdb_rating = models.FloatField(blank=True)
    video_file = models.FileField(blank=True, upload_to='movies')

    def __str__(self):
        return str(self.id) +'-' +self.movie_name


class Episode(models.Model):
    title = models.CharField(max_length=255, blank=True)
    uploaded = models.DateTimeField(auto_now_add=True)
    modified = models.DateTimeField(auto_now=True)
    video_file = models.FileField(blank=True, upload_to='episodes')
```

```
    def __str__(self):
        return str(self.id) +'-' +self.title

class Seasons(models.Model):
    season_number = models.IntegerField(blank=True)
    release_year = models.IntegerField(blank=True)
    number_of_episodes = models.IntegerField(blank=True)
    episodes = models.ManyToManyField(Episode)

    def __str__(self):
        return str(self.id) +'-' + str(self.season_number)

class Series(models.Model):
    series_poster = models.ImageField(blank=True, upload_to='series/posters')
    series_background_image = models.ImageField(blank=True, null=True,
upload_to="series/bgi")
    name = models.CharField(max_length=255, blank=True)
    series_category = models.ForeignKey(Category, on_delete=models.CASCADE)
    seasons = models.ManyToManyField(Seasons)

    def __str__(self):
        return str(self.id) +'-' +self.name
```

serializers.py

```
from django.db.models import fields
from rest_framework import serializers
from accounts.models import User
from .models import *


class CategorySerializer(serializers.ModelSerializer):
    class Meta:
        model = Category
        fields = ["id", "name"]


class MoviesSerializer(serializers.ModelSerializer):
    movie_category = CategorySerializer()

    class Meta:
        model = Movies
        fields = [
            "id",
            "movie_name",
```

```
            "movie_category",
            "year_released",
            "description",
            "imdb_rating",
            "video_file",
            "movie_poster",
        ]


class EpisodeSerializer(serializers.ModelSerializer):
    class Meta:
        model = Episode
        fields = ["id", "title", "uploaded", "modified", "video_file"]


class SeasonsSerializer(serializers.ModelSerializer):
    episodes = EpisodeSerializer()

    class Meta:
        model = Seasons
        fields = [
            "id",
            "season_number",
            "release_year",
            "number_of_episodes",
            "episodes",
        ]


class SeriesSerializer(serializers.ModelSerializer):
    series_category = CategorySerializer()
    seasons = SeasonsSerializer(many=True)

    class Meta:
        model = Series
        fields = [
            "id",
            "name",
            "series_category",
            "seasons",
            "series_poster",
            "series_background_image",
        ]
```

urls.py

```python
from django.urls import path, include
from rest_framework.routers import DefaultRouter
from myapp import views

router = DefaultRouter()
router.register("categories", views.CategoryViewset, basename="categories")
router.register("movies", views.MovieViewset, basename="movies")
router.register("episodes", views.EpisodeViewset, basename="episodes")
router.register("seasons", views.SeasonsViewset, basename="seasons")
router.register("series", views.SeriesViewset, basename="series")

urlpatterns = [
    path("", include(router.urls)),
    path("rec-sys/<str:movie>", views.recSys),
]
```

views.py

```python
# from django.shortcuts import render
from rest_framework.decorators import api_view
from rest_framework import generics, viewsets, status
from rest_framework.response import Response
from rest_framework.decorators import action
from rest_framework.response import Response
from .recSys import get_sorted_recommendations
from myapp.models import Category, Movies, Episode, Seasons, Series
from myapp.serializers import (
    CategorySerializer,
    MoviesSerializer,
    EpisodeSerializer,
    SeasonsSerializer,
    SeriesSerializer,
)
import json
from random import shuffle


# Create your views here.
class CategoryViewset(viewsets.ModelViewSet):
    queryset = Category.objects.all()
    serializer_class = CategorySerializer
    permission_classes = []

    @action(detail=True, methods=["get"])
    def movies(self, request, pk=None):
```

```
        category = self.get_object()
        movies = Movies.objects.filter(movie_category=category)
        serializer = MoviesSerializer(movies, many=True)
        return Response(serializer.data, status=status.HTTP_200_OK)

    @action(detail=True, methods=["get"])
    def series(self, request, pk=None):
        category = self.get_object()
        series = Series.objects.filter(series_category=category)
        serializer = SeriesSerializer(series, many=True)
        return Response(serializer.data, status=status.HTTP_200_OK)


class MovieViewset(viewsets.ModelViewSet):
    queryset = Movies.objects.all()
    serializer_class = MoviesSerializer

    @action(detail=False, methods=["get"])
    def trending(self, request):
        movies = Movies.objects.all()
        movies = list(movies)
        shuffle(movies)
        serializer = MoviesSerializer(movies[:10], many=True)
        return Response(serializer.data, status=status.HTTP_200_OK)

    @action(detail=False, methods=["get"])
    def popular(self, request, pk=None):
        movies = Movies.objects.all().order_by("-imdb_rating")[:10]
        serializer = MoviesSerializer(movies, many=True)
        return Response(serializer.data, status=status.HTTP_200_OK)


class EpisodeViewset(viewsets.ModelViewSet):
    queryset = Episode.objects.all()
    serializer_class = EpisodeSerializer


class SeasonsViewset(viewsets.ModelViewSet):
    queryset = Seasons.objects.all()
    serializer_class = SeasonsSerializer


class SeriesViewset(viewsets.ModelViewSet):
    queryset = Series.objects.all()
    serializer_class = SeriesSerializer
```

```
    @action(detail=False, methods=["get"])
    def trending(self, request):
        series = Series.objects.all()
        series = list(series)
        shuffle(series)
        serializer = SeriesSerializer(series[:11], many=True)
        return Response(serializer.data, status=status.HTTP_200_OK)


@api_view()
def recSys(request, movie):
    return Response(json.dumps(get_sorted_recommendations(movie)))
```

Streaming_platform project:
settings.py

```
from pathlib import Path
import os
from decouple import config

BASE_DIR = Path(__file__).resolve().parent.parent
SECRET_KEY = 'django-insecure-kkjt8%$jrs_kggr)k&90$4*!5a=*q@26l598nf9op40+4cr_pe'
DEBUG = True
ALLOWED_HOSTS = ['*']

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',

    # MODULES
    'rest_framework',
    'rest_framework.authtoken',
    'corsheaders',
    'rest_framework_swagger',

    # LOCAL APPS
    'myapp',
    'accounts',
]

MIDDLEWARE = [
```

```
    'corsheaders.middleware.CorsMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'Streaming_Platform.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
            'libraries' : {
                'staticfiles': 'django.templatetags.static',
            },
        },
    },
]

WSGI_APPLICATION = 'Streaming_Platform.wsgi.application'

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': BASE_DIR / 'db.sqlite3',
    }
}

AUTH_PASSWORD_VALIDATORS = [
    {'NAME':
'django.contrib.auth.password_validation.UserAttributeSimilarityValidator', },
    {'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator', },
```

```
    {'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',
},
    {'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',
},
]

LANGUAGE_CODE = 'en-us'
TIME_ZONE = 'UTC'
USE_I18N = True
USE_L10N = True
USE_TZ = True
AUTH_USER_MODEL = 'accounts.User'
STATIC_URL = '/staticfiles/'
STATIC_ROOT = os.path.join(BASE_DIR, 'staticfiles')
MEDIA_URL = '/media/'
MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'

REST_FRAMEWORK = {
    'DEFAULT_AUTHENTICATION_CLASSES': [
        'rest_framework.authentication.BasicAuthentication',
        'rest_framework.authentication.SessionAuthentication',
    ],
    'DEFAULT_SCHEMA_CLASS': 'rest_framework.schemas.coreapi.AutoSchema'
}

CORS_ORIGIN_ALLOW_ALL = True

#Email:
EMAIL_USE_TLS = True
EMAIL_HOST = 'smtp.gmail.com'
EMAIL_HOST_USER = config('EMAIL_HOST_USER')
EMAIL_HOST_PASSWORD = config('EMAIL_HOST_PASSWORD')
EMAIL_PORT = 587
DOMAIN_NAME = config('DOMAIN_NAME')
```

urls.py

```
from django.contrib import admin
from django.urls import path, include
from django.conf.urls.static import static
from django.conf import settings
# Swagger Docs
from rest_framework_swagger.views import get_swagger_view
```

```
from rest_framework.schemas import get_schema_view
from rest_framework_swagger.renderers import SwaggerUIRenderer, OpenAPIRenderer

from django.contrib.staticfiles.urls import staticfiles_urlpatterns

# Create our schema's view w/ the get_schema_view() helper method. Pass in the
proper Renderers for swagger
# schema_view = get_schema_view(title='Users API',
renderer_classes=[OpenAPIRenderer, SwaggerUIRenderer])
schema_view = get_swagger_view(title='Pastebin API')

urlpatterns = [
    path('', schema_view, name="docs"),

    path('admin/', admin.site.urls),
    path('myapp/', include('myapp.urls')),
    path('accounts/', include('accounts.urls')),
]
urlpatterns += staticfiles_urlpatterns()
urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

Frontend:
package.json

```
{
  "name": "netflix",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "@testing-library/jest-dom": "^5.11.4",
    "@testing-library/react": "^11.1.0",
    "@testing-library/user-event": "^12.1.10",
    "axios": "^0.24.0",
    "bootstrap": "^5.1.3",
    "firebase": "^9.0.2",
    "fuse.js": "^6.4.6",
    "normalize.css": "^8.0.1",
    "react": "^17.0.2",
    "react-bootstrap": "^2.0.4",
    "react-dom": "^17.0.2",
    "react-router-dom": "^5.3.0",
    "react-scripts": "^4.0.3",
    "styled-components": "^5.3.1",
    "web-vitals": "^1.0.1"
  },
```

```
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject"
  },
  "eslintConfig": {
    "extends": [
      "react-app",
      "react-app/jest"
    ]
  },
  "browserslist": {
    "production": [
      ">0.2%",
      "not dead",
      "not op_mini all"
    ],
    "development": [
      "last 1 chrome version",
      "last 1 firefox version",
      "last 1 safari version"
    ]
  }
}
```

app.js

```
import React from "react";
import { BrowserRouter as Router, Route } from "react-router-dom";
import { Home, Browse, Signin, Signup, Movies, Series, Movie, Seasonepisode} from
"./pages";
import * as ROUTES from "./constants/routes";
import useToken from "./hooks/useToken";

export const TokenContext = React.createContext();

export default function App() {
  const { token, setToken } = useToken();

  return (
    <TokenContext.Provider value={{ token, setToken }}>
      <Router>
        <Route exact path={ROUTES.HOME}>
          <Home />
```

```
        </Route>
        <Route path={ROUTES.BROWSE}>
          {token !== "undefined" ? <Browse /> : <Home />}
        </Route>
        <Route path={ROUTES.SIGN_IN}>
          <Signin />
        </Route>
        <Route path={ROUTES.SIGN_UP}>
          <Signup />
        </Route>
        <Route path={ROUTES.MOVIES}>
          <Movies />
        </Route>
        <Route exact path={ROUTES.SERIES}>
          <Series />
        </Route>
        <Route path={ROUTES.MOVIE}>
          <Movie />
        </Route>
        <Route path={ROUTES.SEASONEPISODE}>
          <Seasonepisode />
        </Route>

      </Router>
    </TokenContext.Provider>
  );
}
```

src/components
carousel styles:

```
import styled from "styled-components/macro";
import { Link as ReachRouterLink } from "react-router-dom";

export const Container = styled.div`
  width: 100%;
  display: flex;
  overflow-y: hidden;
  overflow-x: scroll;
  &::-webkit-scrollbar {
    display: none;
  }
`;
export const Image = styled.img`
```

```
  padding: 20px;
  border-radius: 30px;
  transition: transform 0.3s ease-in;

  &:hover {
    transform: scale(1.1);
  }
`;

export const Title = styled(ReachRouterLink)`
  text-decoration: none;
  cursor: pointer;
  font-size: 28px;
  font-weight: 700;
  color: #fff;
  padding-left: 25px;
`;
```

carousel index.js

```
import React from "react";
import { Container, Image, Title } from "./styles/carousel";

function Carousel({ children, ...restProps }) {
  return <Container {...restProps}>{children}</Container>;
}

Carousel.Image = function CarouselImage({ children, ...restProps }) {
  return <Image {...restProps}>{children}</Image>;
};

Carousel.Title = function CarouselTitle({ children, ...restProps }) {
  return <Title {...restProps}>{children}</Title>;
};

export default Carousel;
```

header.js – styles

```
import styled from "styled-components/macro";
import { Link as ReachRouterLink } from "react-router-dom";

export const Background = styled.div`
  display: flex;
  flex-direction: column;
```

```
  background: linear-gradient(
      to bottom,
      rgba(0, 0, 0, 0.35),
      rgba(0, 0, 0, 0.1),
      rgba(0, 0, 0, 0.35)
    ),
    url(${({ src }) =>
        src ? `../images/misc/${src}.jpg` : "../images/misc/home-bg.jpg"})
      top left / cover no-repeat;
  @media (max-width: 1100px) {
    ${({ dontShowOnSmallViewPort }) =>
      dontShowOnSmallViewPort && `background: none;`}
  }
`;

export const Container = styled.div`
  display: flex;
  margin: 0 56px;
  height: 100px;
  background-color: transparent;
  justify-content: space-between;
  align-items: center;
  a {
    display: flex;
  }
  @media (max-width: 1000px) {
    margin: 0 30px;
  }
`;

export const Link = styled.p`
  color: #fff;
  text-decoration: none;
  margin-right: 30px;
  font-weight: ${({ active }) => (active === "true" ? "700" : "normal")};
  cursor: pointer;
  &:hover {
    font-weight: bold;
  }
  &:last-of-type {
    margin-right: 0;
  }
`;

export const Group = styled.div`
```

```
  display: flex;
  align-items: center;
  background-color: transparent;
`;

export const SearchInput = styled.input`
  background-color: rgba(64, 64, 64, 0.5);
  color: white;
  border: 1px solid white;
  transition: width 0.5s;
  height: 30px;
  font-size: 14px;
  border-radius: 4px;
  margin-left: ${({ active }) => (active === true ? "10px" : "0")};
  padding: ${({ active }) => (active === true ? "0 10px" : "0")};
  opacity: ${({ active }) => (active === true ? "1" : "0")};
  width: ${({ active }) => (active === true ? "200px" : "0px")};
  &:focus {
    background-color: rgba(0, 0, 0, 0.8);
  }
`;

export const Search = styled.div`
  display: flex;
  align-items: center;
  svg {
    color: white;
    cursor: pointer;
  }
  @media (max-width: 700px) {
    display: none;
  }
`;

export const SearchIcon = styled.button`
  cursor: pointer;
  background-color: transparent;
  border: 0;
  outline: 0;
  height: 32px;
  width: 32px;
  padding: 0;
  display: flex;
  align-items: center;
  justify-content: center;
```

```
  img {
    filter: brightness(0) invert(1);
    width: 16px;
  }
`;

export const ButtonLink = styled(ReachRouterLink)`
  display: block;
  background-color: #e50914;
  width: 54px;
  height: fit-content;
  color: white;
  border: 0;
  font-size: 15px;
  border-radius: 3px;
  padding: 8px 17px;
  cursor: pointer;
  text-decoration: none;
  &:hover {
    background: #f40612;
  }
`;

export const Picture = styled.button`
  background: url(${({ src }) => src});
  background-size: contain;
  border: 0;
  width: 32px;
  height: 32px;
  cursor: pointer;
`;

export const Dropdown = styled.div`
  display: none;
  position: absolute;
  background-color: black;
  padding: 10px;
  width: 100px;
  top: 32px;
  right: 10px;
  ${Group}:last-of-type ${Link} {
    cursor: pointer;
  }
  ${Group} {
    margin-bottom: 10px;
```

```
    &:last-of-type {
      margin-bottom: 0;
    }
    ${Link} {
      cursor: pointer;
    }
    ${Picture} {
      cursor: default;
    }
  }
  button {
    margin-right: 10px;
  }
  p {
    font-size: 12px;
    margin-bottom: 0;
    margin-top: 0;
  }
`;

export const Profile = styled.div`
  display: flex;
  align-items: center;
  margin-left: 20px;
  position: relative;
  button {
    cursor: pointer;
  }
  &:hover > ${Dropdown} {
    display: flex;
    flex-direction: column;
  }
`;

export const Feature = styled(Container)`
  background-color: transparent;
  padding: 150px 0 500px 0;
  flex-direction: column;
  align-items: normal;
  width: 50%;
  @media (max-width: 1100px) {
    display: none;
  }
`;
```

```
export const FeatureCallOut = styled.h2`
  color: white;
  background-color: transparent;
  font-size: 50px;
  line-height: normal;
  font-weight: bold;
  text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.45);
  margin: 0;
`;

export const Text = styled.p`
  color: white;
  font-size: 22px;
  background-color: transparent;
  line-height: normal;
  text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.45);
`;

export const Logo = styled.img`
  background-color: transparent;
  height: 36px;
  width: 134px;
  margin-right: 40px;
  @media (min-width: 1449px) {
    height: 45px;
    width: 167px;
  }
`;

export const PlayButton = styled.button`
  box-shadow: 0 0.6vw 1vw -0.4vw rgba(0, 0, 0, 0.35);
  background-color: #e6e6e6;
  color: #000;
  border-width: 0;
  padding: 10px 20px;
  border-radius: 5px;
  max-width: 130px;
  font-weight: bold;
  font-size: 20px;
  margin-top: 10px;
  cursor: pointer;
  transition: background-color 0.5s ease;
  &:hover {
    background-color: #ff1e1e;
    color: white;
```

```
  }
`;
```

header - index.js

```
import React from "react";
import { Link as ReactRouterLink } from "react-router-dom";
import {
  ButtonLink,
  Background,
  Container,
  Logo,
  Feature,
  Group,
  Profile,
  Picture,
  Dropdown,
  Link,
  PlayButton,
  FeatureCallOut,
  Text,
} from "./styles/header";

export default function Header({ bg = true, children, ...restProps }) {
  return bg ? <Background {...restProps}>{children}</Background> : children;
}

Header.Feature = function HeaderFeature({ children, ...restProps }) {
  return <Feature {...restProps}>{children}</Feature>;
};

Header.Group = function HeaderGroup({ children, ...restProps }) {
  return <Group {...restProps}>{children}</Group>;
};

Header.Logo = function HeaderLogo({ to, ...restProps }) {
  return (
    <ReactRouterLink to={to}>
      <Logo {...restProps} />
    </ReactRouterLink>
  );
};

Header.Frame = function HeaderFrame({ children, ...restProps }) {
  return <Container {...restProps}>{children}</Container>;
```

```
};

Header.ButtonLink = function HeaderButtonLink({ children, ...restProps }) {
  return <ButtonLink {...restProps}>{children}</ButtonLink>;
};

Header.Logo = function HeaderLogo({ to, ...restProps }) {
  return (
    <ReactRouterLink to={to}>
      <Logo {...restProps} />
    </ReactRouterLink>
  );
};

Header.Profile = function HeaderProfile({ children, ...restProps }) {
  return <Profile {...restProps}>{children}</Profile>;
};

Header.Feature = function HeaderFeature({ children, ...restProps }) {
  return <Feature>{children}</Feature>;
};

Header.Picture = function HeaderPicture({ src, ...restProps }) {
  return <Picture {...restProps} src={`/images/users/${src}.png`} />;
};

Header.Dropdown = function HeaderDropdown({ children, ...restProps }) {
  return <Dropdown {...restProps}>{children}</Dropdown>;
};

Header.TextLink = function HeaderTextLink({ children, ...restProps }) {
  return <Link {...restProps}>{children}</Link>;
};

Header.PlayButton = function HeaderPlayButton({ children, ...restProps }) {
  return <PlayButton {...restProps}>{children}</PlayButton>;
};

Header.FeatureCallOut = function HeaderFeatureCallOut({
  children,
  ...restProps
}) {
  return <FeatureCallOut {...restProps}>{children}</FeatureCallOut>;
};
```

```
Header.Text = function HeaderText({ children, ...restProps }) {
  return <Text {...restProps}>{children}</Text>;
};

Header.ButtonLink = function HeaderButtonLink({ children, ...restProps }) {
  return <ButtonLink {...restProps}>{children}</ButtonLink>;
};
```

index.js all components

```
export { default as Jumbotron } from "./jumbotron";
export { default as Footer } from "./footer";
export { default as Accordion } from "./accordion";
export { default as OptForm } from "./opt-form";
export { default as Header } from "./header";
export { default as Feature } from "./feature";
export { default as Form } from "./form";
export { default as Carousel } from "./carousel";
export { default as MovieContainer } from "./movie-container";
```

constants/routes.js

```
export const HOME = "/";
export const BROWSE = "/browse";
export const SIGN_UP = "/signup";
export const SIGN_IN = "/signin";
export const SERIES = "/series/:id";
export const MOVIES = "/movies";
export const MOVIE = "/movie/:id";
export const SEASONEPISODE = "/series/episodes/:id";
```

containers

browse.js

```
import React, { useState, useEffect } from "react";
import { Link, useHistory } from "react-router-dom";
import axios from "axios";
import { Header, Carousel } from "../components";
import * as ROUTES from "../constants/routes";
import logo from "../logo.svg";
import FooterContainer from "./footer";

export function BrowseContainer() {
  const history = useHistory();
```

```
const [movieData, setMovieData] = useState([]);
const [seriesData, setSeriesData] = useState([]);

useEffect(() => {
  const fetchData = async () => {
    await axios
      .get("http://localhost:8000/myapp/movies/")
      .then((res) => {
        setMovieData(res.data);
        console.log(movieData);
      })
      .catch((err) => console.log(err));

    await axios
      .get("http://localhost:8000/myapp/series/")
      .then((res) => {
        setSeriesData(res.data);
        console.log("Series data" + seriesData);
      })
      .catch((err) => console.log(err));
  };
  fetchData();
}, []);

return (
  <>
    <Header src="joker1" dontShowOnSmallViewPort>
      <Header.Frame>
        <Header.Group>
          <Header.Logo to={ROUTES.HOME} src={logo} alt="Netflix" />
        </Header.Group>
      </Header.Frame>

      <Header.Feature styles={{ background: "transparent" }}>
        <Header.FeatureCallOut styles={{ background: "transparent" }}>
          Watch Joker Now
        </Header.FeatureCallOut>
        <Header.Text styles={{ background: "transparent" }}>
          Forever alone in a crowd, failed comedian Arthur Fleck seeks
          connection as he walks the streets of Gotham City. Arthur wears two
          masks -- the one he paints for his day job as a clown, and the guise
          he projects in a futile attempt to feel like he's part of the world
          around him.
        </Header.Text>
        <Header.PlayButton
```

```
            styles={{ background: "transparent" }}
            onClick={() => history.push("/movie/6")}
          >
            Play
          </Header.PlayButton>
        </Header.Feature>
      </Header>
      <Carousel.Title to={ROUTES.MOVIES}>Movies</Carousel.Title>
      <Carousel style={{ background: "transparent" }}>
        {movieData.map((item) => {
          return (
            <Link to={`/movie/${item.id}`}>
              {" "}
              <Carousel.Image
                key={`${item.id}`}
                src={`${item.movie_poster}`}
                style={{ background: "transparent" }}
              />
            </Link>
          );
        })}
      </Carousel>
      <Carousel.Title to={ROUTES.MOVIES}>Series</Carousel.Title>
      <Carousel style={{ background: "transparent" }}>
        {seriesData.map((item) => {
          return (
            <Link to={`/series/${item.id}`}>
              <Carousel.Image
                key={`${item.id}`}
                src={`${item.series_poster}`}
                style={{ background: "transparent" }}
              />
            </Link>
          );
        })}
      </Carousel>
      <FooterContainer />
    </>
  );
}
```

footer.js

```
import React from "react";
import { Footer } from "../components";
```

```
function FooterContainer() {
  return (
    <Footer>
      <Footer.Title>Questions? Call 000-800-040-1843</Footer.Title>
      <Footer.Break />
      <Footer.Row>
        <Footer.Column>
          <Footer.Link href="#">FAQ</Footer.Link>
          <Footer.Link href="#">Investor Relations</Footer.Link>
          <Footer.Link href="#">Privacy</Footer.Link>
          <Footer.Link href="#">Speed Test</Footer.Link>
        </Footer.Column>
        <Footer.Column>
          <Footer.Link href="#">Help Center</Footer.Link>
          <Footer.Link href="#">Jobs</Footer.Link>
          <Footer.Link href="#">Cookie Preferences</Footer.Link>
          <Footer.Link href="#">Legal Notices</Footer.Link>
        </Footer.Column>
        <Footer.Column>
          <Footer.Link href="#">Account</Footer.Link>
          <Footer.Link href="#">Ways to Watch</Footer.Link>
          <Footer.Link href="#">Coporate Information</Footer.Link>
          <Footer.Link href="#">Only on Netflix</Footer.Link>
        </Footer.Column>
        <Footer.Column>
          <Footer.Link href="#">Media Center</Footer.Link>
          <Footer.Link href="#">Terms of Use</Footer.Link>
          <Footer.Link href="#">Contact Us</Footer.Link>
        </Footer.Column>
      </Footer.Row>
      <Footer.Break />
      <Footer.Text>Netflix India</Footer.Text>
    </Footer>
  );
}

export default FooterContainer;
```

movie-container.js

```
import React, { useState, useEffect } from "react";
import { useParams } from "react-router-dom";
import axios from "axios";
import FooterContainer from "./footer";
```

```jsx
export function IndividualMovieContainer() {
  const { id } = useParams();

  const [imageInfo, setImageInfo] = useState({
    id: "",
    movie_name: "",
    movie_category: {
      id: "",
      name: "",
    },
    year_released: "",
    description: "",
    imdb_rating: "",
    video_file: "",
    movie_poster: "",
  });
  const getData = async () => {
    try {
      const res = await axios.get(`http://localhost:8000/myapp/movies/${id}`);
      setImageInfo(res.data);
    } catch (error) {
      console.log(error);
    }
  };
  useEffect(() => {
    //alert(id);
    getData();
  }, []);
  return (
    <>
      <div
        style={{
          width: "750px",
          margin: "0 auto 0 auto",
          backgroundColor: "transparent",
          wordSpacing: "2px",
          letterSpacing: "0.5px",
          lineHeight: "30px",
        }}
      >
        <video
          src={imageInfo.video_file}
          width="750"
          height="500"
```

```
          controls
          autoplay
        />
        <h1
          style={{
            backgroundColor: "transparent",
            color: "white",
            textTransform: "uppercase",
          }}
        >
          {imageInfo.movie_name}
        </h1>
        <div
          style={{
            color: "white",
            fontSize: "28px",
            fontFamily: "Forte",
            marginBottom: "20px",
          }}
        >
          {imageInfo.year_released}
        </div>
        <div
          style={{
            color: "white",
            fontSize: "16px",
            marginBottom: "20px",
            backgroundColor: "transparent",
          }}
        >
          {imageInfo.description}
        </div>
        <div
          style={{
            color: "white",
            fontSize: "20px",
            marginBottom: "20px",
            backgroundColor: "transparent",
          }}
        >
          <span
            style={{
              fontWeight: "bold",
              fontSize: "20px",
              backgroundColor: "transparent",
```

```
            }}
          >
            Rating [IMDB]
          </span>
          {" : "}
          {imageInfo.imdb_rating + "  "}
          <input
            type="range"
            value={Number(imageInfo.imdb_rating) * 10}
            min={0}
            max={100}
            disabled
          />
        </div>
        <div
          style={{
            color: "white",
            fontSize: "20px",
            backgroundColor: "transparent",
          }}
        >
          Category : {imageInfo.movie_category.name}
        </div>
      </div>
      <FooterContainer />
    </>
  );
}
```

movies.js

```
import React, { useState, useEffect } from "react";
import { Link } from 'react-router-dom';
import axios from 'axios';
import movieData from "../fixtures/movies.json";
import seriesData from "../fixtures/series.json";
import { Carousel } from "../components";
import FooterContainer from "./footer";

export function MovieContainer() {

  const [trendingMovieeData, setTrendingMovieeData] = useState([]);
  const [popularMovieeData, setPopularMovieeData] = useState([]);

  useEffect(() => {
```

```
  const fetchData = async () => {

    await axios.get('http://localhost:8000/myapp/movies/trending')
      .then(res => { setTrendingMovieeData(res.data);})
      .catch(err => console.log(err))


    await axios.get('http://localhost:8000/myapp/movies/popular/')
      .then(res => { setPopularMovieeData(res.data);})
      .catch(err => console.log(err))

  }
  fetchData();

}, [])

return (
  <>
    <Carousel.Title>Coming This Week</Carousel.Title>
    <Carousel style={{ background: "transparent" }}>
      {movieData.map((item) => {
        return (
          <Carousel.Image
            key={`${item.id}`}
            // alt={`${item.alt}`}
            src={`${item.image}`}
            style={{ background: "transparent" }}
          />
        );
      })}
    </Carousel>
    <Carousel.Title>For You</Carousel.Title>
    <Carousel style={{ background: "transparent" }}>
      {seriesData.map((item) => {
        return (
          <Carousel.Image
            key={`${item.id}`}
            alt={`${item.alt}`}
            src={`${item.image}`}
            style={{ background: "transparent" }}
          />
        );
      })}
    </Carousel>
    <Carousel.Title>Recently Watched</Carousel.Title>
```

```
      <Carousel style={{ background: "transparent" }}>
        {seriesData.map((item) => {
          return (
            <Carousel.Image
              key={`${item.id}`}
              alt={`${item.alt}`}
              src={`${item.image}`}
              style={{ background: "transparent" }}
            />
          );
        })}
      </Carousel>
      <Carousel.Title>Popular on Netflix</Carousel.Title>
      <Carousel style={{ background: "transparent" }}>
        {popularMovieeData.map((item) => {
          return (
            <Link to={`/movie/${item.id}`}> <Carousel.Image
              key={`${item.id}`}
              // alt={`${item.alt}`}
              src={`http://localhost:8000${item.movie_poster}`}
              style={{ background: "transparent" }}
            />
            </Link>
          );
        })}
      </Carousel>
      <Carousel.Title>New Releases</Carousel.Title>
      <Carousel style={{ background: "transparent" }}>
        {seriesData.map((item) => {
          return (
            <Carousel.Image
              key={`${item.id}`}
              alt={`${item.alt}`}
              src={`${item.image}`}
              style={{ background: "transparent" }}
            />
          );
        })}
      </Carousel>
      <Carousel.Title>Trending Now</Carousel.Title>
      <Carousel style={{ background: "transparent" }}>
        {trendingMovieeData.map((item) => {
          return (
            <Link to={`/movie/${item.id}`}> <Carousel.Image
              key={`${item.id}`}
```

```
              // alt={`${item.alt}`}
              src={`http://localhost:8000${item.movie_poster}`}
              style={{ background: "transparent" }}
            />
            </Link>
          );
        })}
      </Carousel>
      <FooterContainer />
    </>
  );
}
```

pages
signin.js

```
import React, { useState, useContext } from "react";
import HeaderContainer from "../containers/header";
import FooterContainer from "../containers/footer";
import { Form } from "../components";

import axios from "axios";
import { TokenContext } from "../App";
import { useHistory } from "react-router-dom";

async function loginUser(credentials) {
  return axios
    .post("http://127.0.0.1:8000/accounts/login/", credentials)
    .then((res) => res.data)
    .catch((err) => {
      console.log(err);
      alert("Invalid Credentials");
      return "invalid login";
    });
}

function Signin() {
  const history = useHistory();
  const { token, setToken } = useContext(TokenContext);

  const [emailAddress, setEmailAddress] = useState('');
  const [password, setPassword] = useState('');
  const [error, setError] = useState("");
  const isInvalid = password === "" || emailAddress === "";
```

```
const handleSignIn = async (event) => {
  event.preventDefault();

  if (emailAddress !== "" && password !== "") {
    const token = await loginUser({
      email: emailAddress,
      password,
    });

    if (token !== "invalid login") {
      setToken(token);
      history.push("/browse");
    }
  }
};

// check form input elements are valid
// email & password

return (
  <>
    <HeaderContainer>
      <Form>
        <Form.Title style={{ background: "transparent" }}>Sign In</Form.Title>
        {error && <Form.Error>{error}</Form.Error>}
        <Form.Base onSubmit={handleSignIn} method="POST">
          <Form.Input
            placeholder="Email Address"
            value={emailAddress}
            onChange={({ target }) => setEmailAddress(target.value)}
          />
          <Form.Input
            type="password"
            autocomplete="off"
            placeholder="Password"
            value={password}
            onChange={({ target }) => setPassword(target.value)}
          />
          <Form.Submit disabled={isInvalid} type="submit">
            Sign In
          </Form.Submit>
        </Form.Base>
        <Form.Text>
          New to Netflix? <Form.Link to="/signup">Sign up now.</Form.Link>
        </Form.Text>
```

```
          <Form.TextSmall>
            This page is protected by Google reCAPTCHA to ensure you're not a
            bot. Learn more.
          </Form.TextSmall>
        </Form>
      </HeaderContainer>
      <FooterContainer></FooterContainer>
    </>
  );
}

export default Signin;
```

signup.js

```
import React, { useState } from "react";
import {useHistory} from 'react-router-dom';
import { Form } from "../components";
import HeaderContainer from "../containers/header";
import FooterContainer from "../containers/footer";
import axios from 'axios';

export default function SignUp() {
  const history = useHistory();
  const [firstName, setFirstName] = useState("");
  const [emailAddress, setEmailAddress] = useState("");
  const [password, setPassword] = useState("");
  const [error, setError] = useState("");

  const isInvalid = firstName === "" || password === "" || emailAddress === "";

  const handleSignup = (event) => {
    event.preventDefault();

    axios.post('http://127.0.0.1:8000/accounts/signup/', {
      email: emailAddress,
      first_name: firstName,
      last_name: firstName,
      password: password
    })
    .then(res=>{alert('successfully registered');
console.log(res);history.push('/signin');})
    .catch(err=>console.log(err))

  };
```

```jsx
return (
  <>
    <HeaderContainer>
      <Form>
        <Form.Title>Sign Up</Form.Title>
        {error && <Form.Error>{error}</Form.Error>}

        <Form.Base onSubmit={handleSignup} method="POST">
          <Form.Input
            placeholder="First name"
            value={firstName}
            onChange={({ target }) => setFirstName(target.value)}
          />
          <Form.Input
            placeholder="Email address"
            value={emailAddress}
            onChange={({ target }) => setEmailAddress(target.value)}
          />
          <Form.Input
            type="password"
            value={password}
            autoComplete="off"
            placeholder="Password"
            onChange={({ target }) => setPassword(target.value)}
          />
          <Form.Submit
            disabled={isInvalid}
            type="submit"
            data-testid="sign-up"
          >
            Sign Up
          </Form.Submit>
        </Form.Base>

        <Form.Text>
          Already a user? <Form.Link to="/signin">Sign in now.</Form.Link>
        </Form.Text>
        <Form.TextSmall>
          This page is protected by Google reCAPTCHA to ensure you're not a
          bot. Learn more.
        </Form.TextSmall>
      </Form>
    </HeaderContainer>
    <FooterContainer />
```

```
    </>
  );
}
```

season.js

```
import React, { useState, useEffect } from 'react';
import axios from 'axios';
import { useParams, Link } from 'react-router-dom';
import { Header, Carousel } from "../components";

const Series = () => {

  // const seriesNumber = useParams(1);
  const seriesNumber = 1;
  const [seasons, setSeasons] = useState([]);
  const [seasonSelected, setSeasonSelected] = useState(false);
  const [episodes, setEpisodes] = useState([]);

  useEffect(async () => {
    await axios.get(`http://localhost:8000/myapp/series/${seriesNumber}/`)
      .then(res => { setSeasons(res.data.seasons);
seasonSelected(seasonSelected=>!seasonSelected)})
      .catch(err => console.log(err))

  }, [])


  const getEpisodes =async ()=>{
    await axios.get(`http://localhost:8000/myapp/episodes/`)
      .then(res => { setEpisodes(res.data); console.log(res.data)})
      .catch(err => console.log(err))
  }

  return (
    <>

    <Carousel.Title>Seasons for Series</Carousel.Title>
    <Carousel style={{ background: "transparent" }}>
      {seasons.map((item) => {
        return (
          <div onClick={getEpisodes}>
            <p style={{color: "white"}}>Season {item.id}</p>
          </div>
        );
```

```
      })}
    </Carousel>


    <Carousel.Title>Episodes for Seasons</Carousel.Title>
    <Carousel style={{ background: "transparent" }}>
        {
        seasonSelected? episodes.map((item) => {
          return (
            // <video src={item.video_file} width="350" height="200" controls
autoplay />
            <p style={{color: "white"}}>Episode {item.id}</p>
          );
        }):<p  style={{color: "white"}}>kfk</p>
      }
      </Carousel>


    </>
  )
}

export default Series
```

seasonepisode.js

```
import React, { useState, useEffect } from "react";
import { useParams } from "react-router-dom";
import axios from "axios";

export default function Seasonepisode() {
  const { id } = useParams();

  const [videoInfo, setVideoInfo] = useState({});

  const getData = async () => {
    try {
      const res = await axios.get(
        `http://localhost:8000/myapp/episodes/${id}/`
      );
      setVideoInfo(res.data);
      console.log(res.data);
    } catch (error) {
      console.log(error);
    }
```

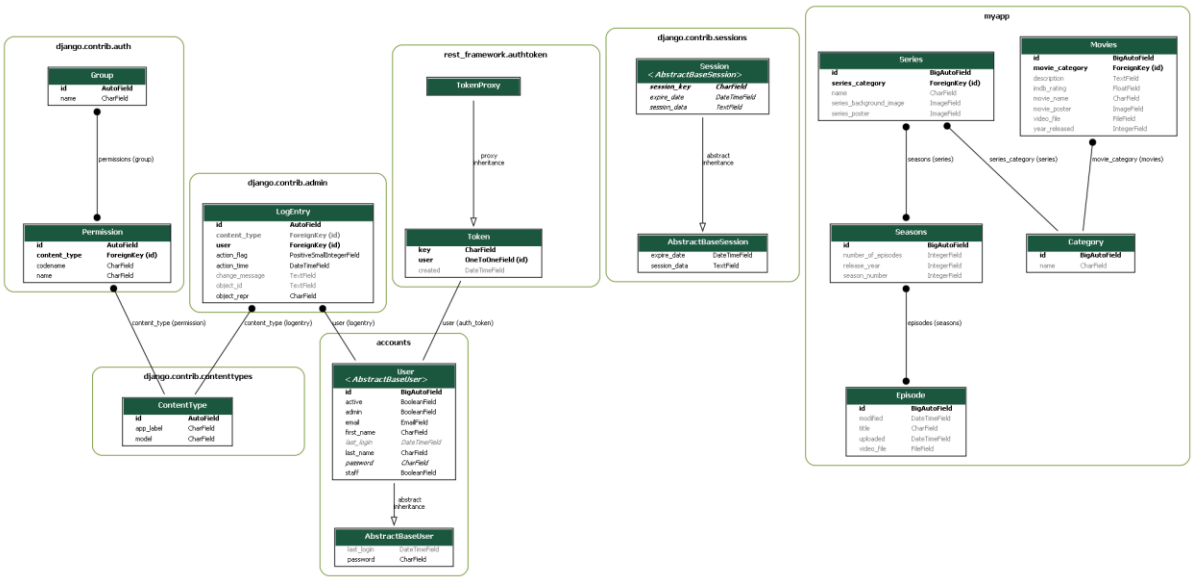
```
};

useEffect(() => {
  //alert(id);
  getData();
}, []);

return (
  <>
    <div
      style={{
        width: "750px",
        height: "100vh",
        margin: "0 auto 0 auto",
        backgroundColor: "transparent",
        wordSpacing: "2px",
        letterSpacing: "0.5px",
        lineHeight: "30px",
        alignItems: "center",
        justifyContent: "center",
      }}
    >
      <video
        src={videoInfo.video_file}
        width="750"
        height="500"
        controls
        autoplay
      />
      <h1
        style={{
          backgroundColor: "transparent",
          color: "white",
          textTransform: "uppercase",
        }}
      >
        {videoInfo.title}
      </h1>
      <div
        style={{
          color: "white",
          fontSize: "28px",
          fontFamily: "Forte",
          marginBottom: "20px",
        }}
```

```
        >
          uploaded on : {videoInfo.uploaded}
        </div>
        <div
          style={{
            color: "white",
            fontSize: "16px",
            marginBottom: "20px",
            backgroundColor: "transparent",
          }}
        >
          Modified on : {videoInfo.modified}
        </div>
      </div>
    </>
  );
}
```

Output:
ER Diagram:



Admin Panel:

Django administration

Site administration

ACCOUNTS		
Users	+ Add	Change

AUTH TOKEN		
Tokens	+ Add	Change

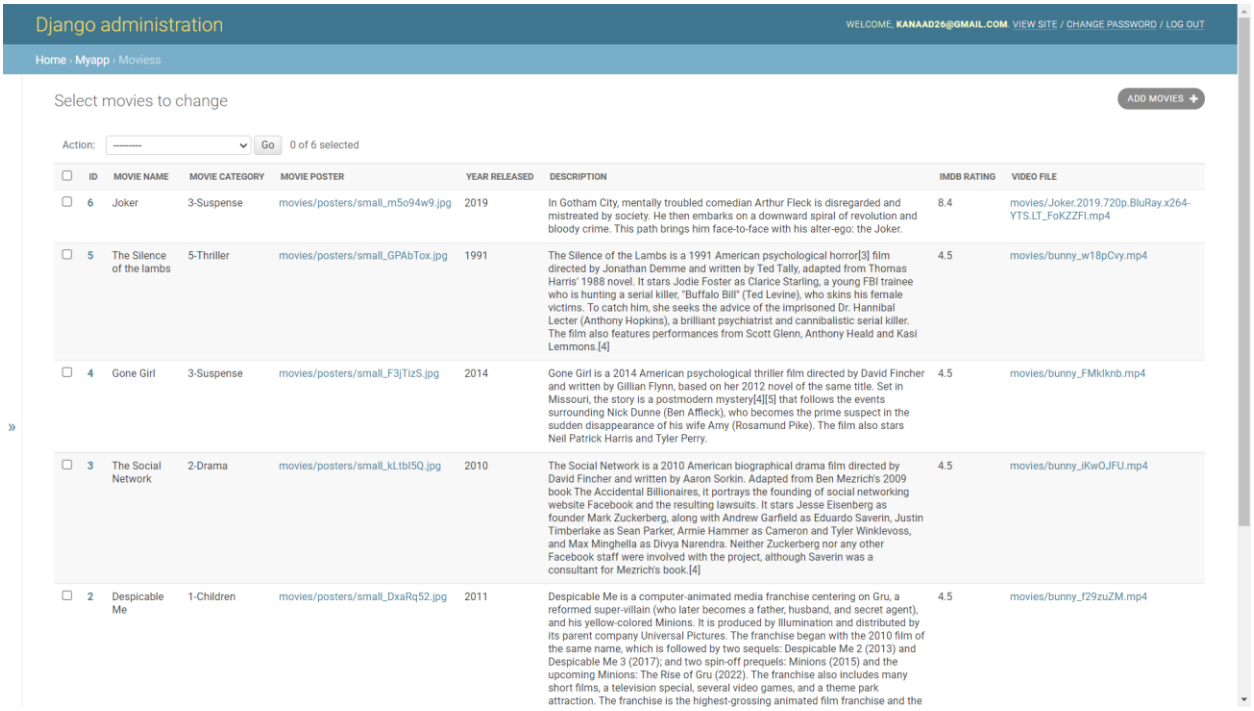
AUTHENTICATION AND AUTHORIZATION		
Groups	+ Add	Change

MYAPP		
Categorys	+ Add	Change
Episodes	+ Add	Change
Moviess	+ Add	Change
Seasonss	+ Add	Change
Seriess	+ Add	Change

Recent actions

My actions

- + 5-Paw Patrol
  Series
- + 4-Spongebob
  Series
- + 3-Forest Gump
  Series
- + 2-The Innocent Man
  Series
- 1-The Office
  Series
- 1-The Office
  Series
- + 3-2
  Seasons
- + 10-Sexual Harrassment
  Episode
- + 9-The Dundies
  Episode
- + 8-The Dundies
  Episode



ID	MOVIE NAME	MOVIE CATEGORY	MOVIE POSTER	YEAR RELEASED	DESCRIPTION	IMDB RATING	VIDEO FILE
6	Joker	3-Suspense	movies/posters/small_m5o94w9.jpg	2019	In Gotham City, mentally troubled comedian Arthur Fleck is disregarded and mistreated by society. He then embarks on a downward spiral of revolution and bloody crime. This path brings him face-to-face with his alter-ego: the Joker.	8.4	movies/Joker.2019.720p.BluRay.x264-YTS.LT_FoKZZFI.mp4
5	The Silence of the lambs	5-Thriller	movies/posters/small_GPAbTox.jpg	1991	The Silence of the Lambs is a 1991 American psychological horror[3] film directed by Jonathan Demme and written by Ted Tally, adapted from Thomas Harris' 1988 novel. It stars Jodie Foster as Clarice Starling, a young FBI trainee who is hunting a serial killer, "Buffalo Bill" (Ted Levine), who skins his female victims. To catch him, she seeks the advice of the imprisoned Dr. Hannibal Lecter (Anthony Hopkins), a brilliant psychiatrist and cannibalistic serial killer. The film also features performances from Scott Glenn, Anthony Heald and Kasi Lemmons.[4]	4.5	movies/bunny_w18pCvy.mp4
4	Gone Girl	3-Suspense	movies/posters/small_F3jTizS.jpg	2014	Gone Girl is a 2014 American psychological thriller film directed by David Fincher and written by Gillian Flynn, based on her 2012 novel of the same title. Set in Missouri, the story is a postmodern mystery[4][5] that follows the events surrounding Nick Dunne (Ben Affleck), who becomes the prime suspect in the sudden disappearance of his wife Amy (Rosamund Pike). The film also stars Neil Patrick Harris and Tyler Perry.	4.5	movies/bunny_FMkIknb.mp4
3	The Social Network	2-Drama	movies/posters/small_kLtbl5Q.jpg	2010	The Social Network is a 2010 American biographical drama film directed by David Fincher and written by Aaron Sorkin. Adapted from Ben Mezrich's 2009 book The Accidental Billionaires, it portrays the founding of social networking website Facebook and the resulting lawsuits. It stars Jesse Eisenberg as founder Mark Zuckerberg, along with Andrew Garfield as Eduardo Saverin, Justin Timberlake as Sean Parker, Armie Hammer as Cameron and Tyler Winklevoss, and Max Minghella as Divya Narendra. Neither Zuckerberg nor any other Facebook staff were involved with the project, although Saverin was a consultant for Mezrich's book.[4]	4.5	movies/bunny_iKwOJFU.mp4
2	Despicable Me	1-Children	movies/posters/small_DxaRq52.jpg	2011	Despicable Me is a computer-animated media franchise centering on Gru, a reformed super-villain (who later becomes a father, husband, and secret agent), and his yellow-colored Minions. It is produced by Illumination and distributed by its parent company Universal Pictures. The franchise began with the 2010 film of the same name, which is followed by two sequels: Despicable Me 2 (2013) and Despicable Me 3 (2017); and two spin-off prequels: Minions (2015) and the upcoming Minions: The Rise of Gru (2022). The franchise also includes many short films, a television special, several video games, and a theme park attraction. The franchise is the highest-grossing animated film franchise and the	4.5	movies/bunny_f29zuZM.mp4

Django administration

WELCOME, KANAAD26@GMAIL.COM. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home › Myapp › Seriess

Select series to change

ADD SERIES +

Action: -------- Go 0 of 5 selected

ID	SERIES POSTER	NAME	SERIES CATEGORY	GET SEASONS
5	series/posters/small_YAM7hjG.jpg	Paw Patrol	1-Children	1
4	series/posters/small_H0w6rgm.jpg	Spongebob	1-Children	1
3	series/posters/small_uY4FdWu.jpg	Forest Gump	9-Feel Good	1
2	series/posters/small_0NBIBvx.jpg	The Innocent Man	3-Suspense	1
1	series/posters/small_5OIIYCC.jpg	The Office	6-Comedy	1

5 seriess

»

Django administration

WELCOME, KANAAD26@GMAIL.COM. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home › Myapp › Categorys

Select category to change

ADD CATEGORY +

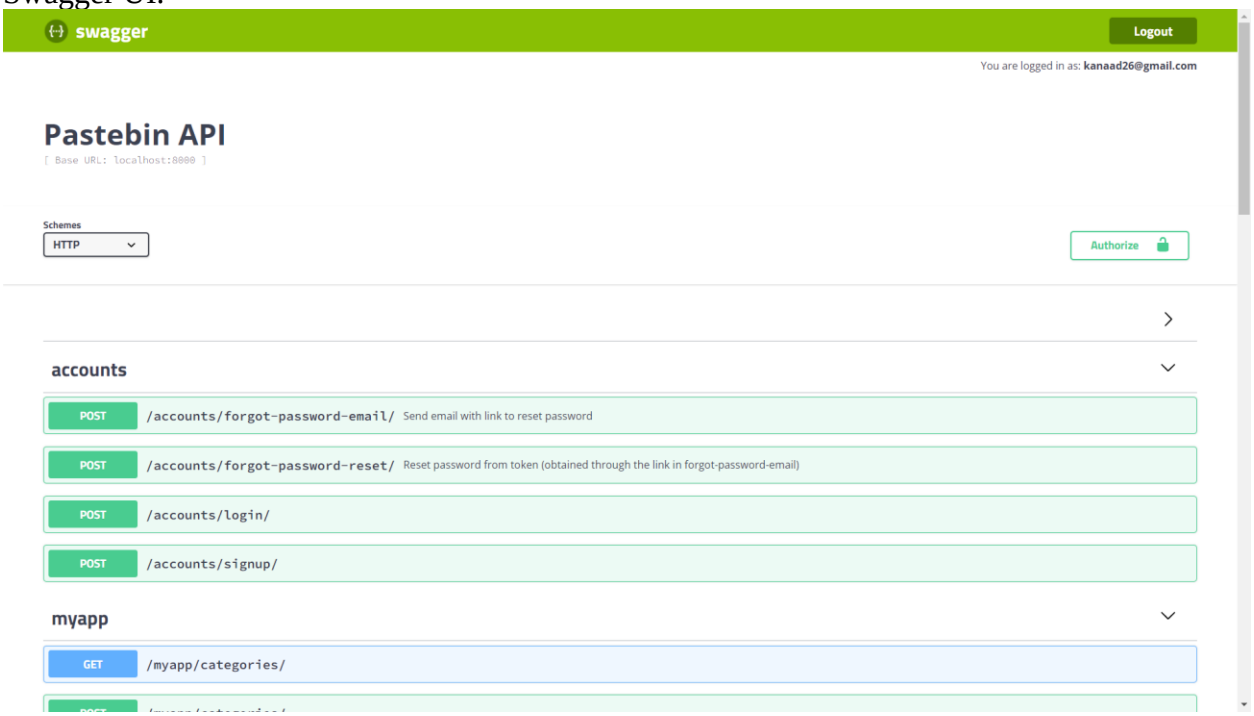
Action: -------- Go 0 of 9 selected

ID	NAME
9	Feel Good
8	Documentary
7	Crime
6	Comedy
5	Thriller
4	Romance
3	Suspense
2	Drama
1	Children

9 categorys

»

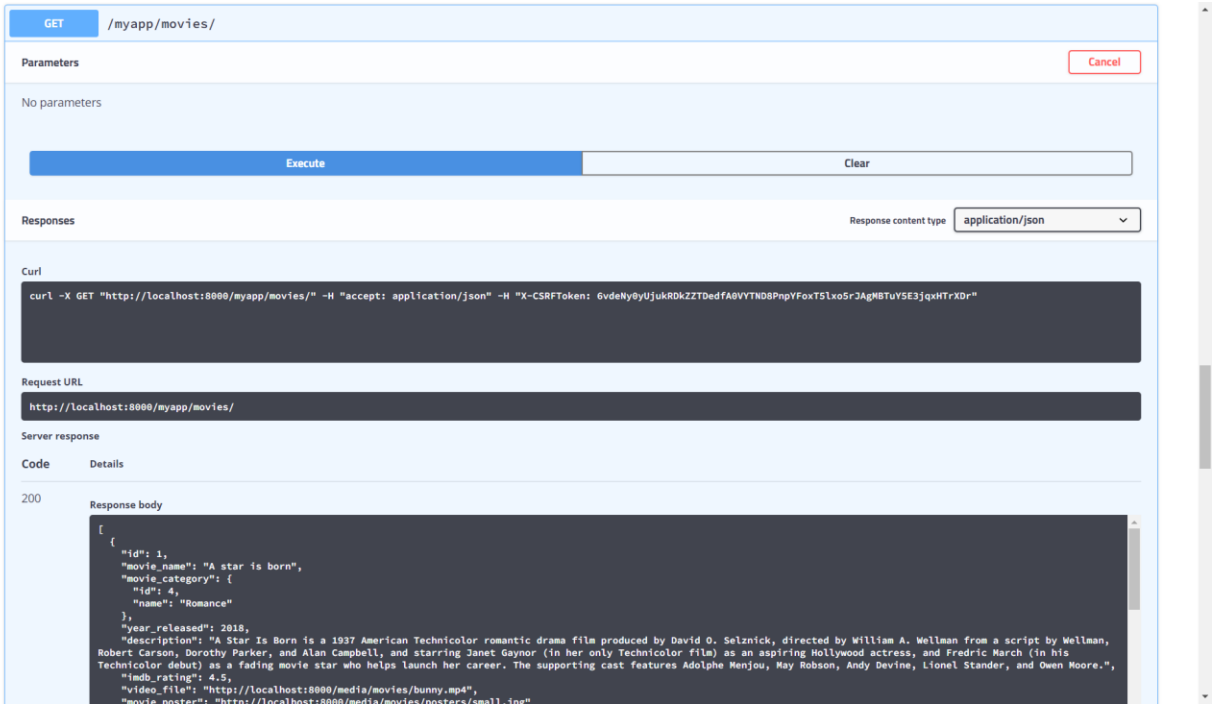
Swagger UI:


swagger
Logout
You are logged in as: kanaad26@gmail.com
Pastebin API
[Base URL: localhost:8000]
Schemes
HTTP
Authorize
accounts
POST /accounts/forgot-password-email/ Send email with link to reset password
POST /accounts/forgot-password-reset/ Reset password from token (obtained through the link in forgot-password-email)
POST /accounts/login/
POST /accounts/signup/
myapp
GET /myapp/categories/

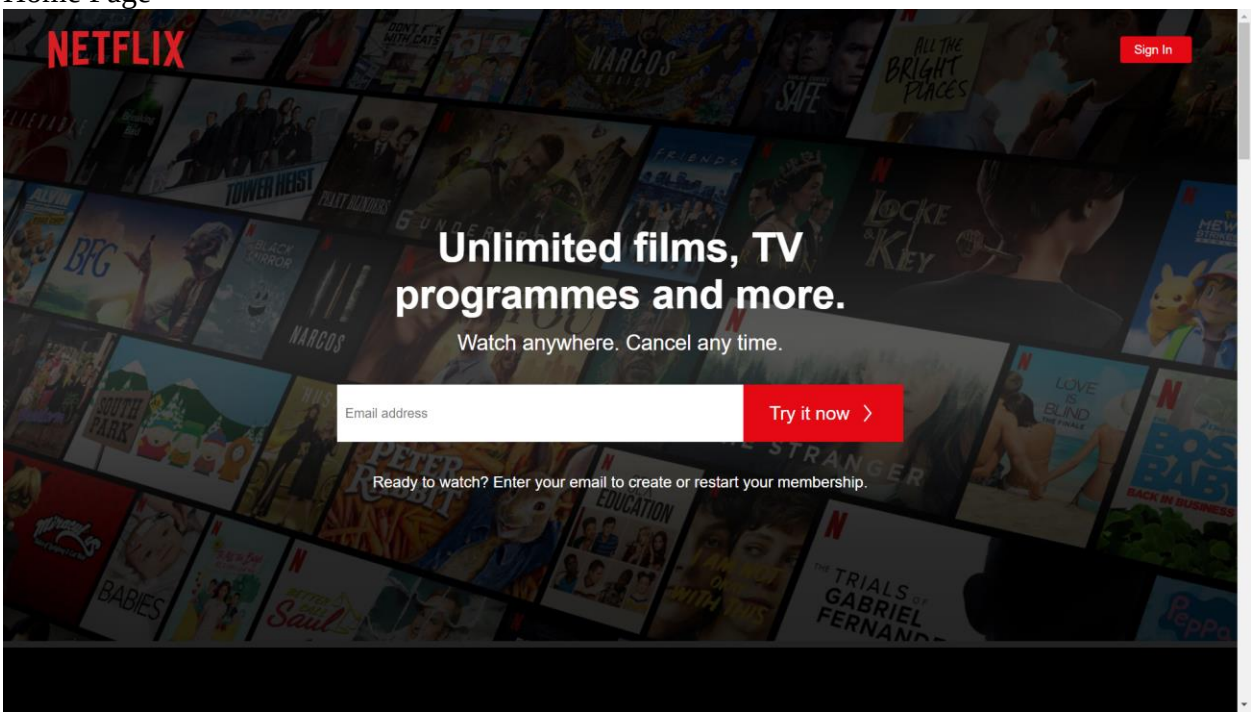
myapp

GET	/myapp/categories/
POST	/myapp/categories/
GET	/myapp/categories/{id}/
PUT	/myapp/categories/{id}/
PATCH	/myapp/categories/{id}/
DELETE	/myapp/categories/{id}/
GET	/myapp/categories/{id}/movies/
GET	/myapp/categories/{id}/series/
GET	/myapp/episodes/
POST	/myapp/episodes/
GET	/myapp/episodes/{id}/
PUT	/myapp/episodes/{id}/
PATCH	/myapp/episodes/{id}/

GET	/myapp/movies/trending/
GET	/myapp/movies/{id}/
PUT	/myapp/movies/{id}/
PATCH	/myapp/movies/{id}/
DELETE	/myapp/movies/{id}/
GET	/myapp/rec-sys/{movie}
GET	/myapp/seasons/
POST	/myapp/seasons/
GET	/myapp/seasons/{id}/
PUT	/myapp/seasons/{id}/
PATCH	/myapp/seasons/{id}/
DELETE	/myapp/seasons/{id}/
GET	/myapp/series/
POST	/myapp/series/


GET
/myapp/movies/
Parameters
Cancel
No parameters
Execute
Clear
Responses
Response content type
application/json
Curl
curl -X GET "http://localhost:8000/myapp/movies/" -H "accept: application/json" -H "X-CSRFToken: 6vdeNy0yUjukRDkZZTDedfA0VYTND8PnpYFoxT5lxo5rJAgMBTuYSE3jqxHTrXDr"
Request URL
http://localhost:8000/myapp/movies/
Server response
Code
Details
200
Response body
[
{
"id": 1,
"movie_name": "A star is born",
"movie_category": {
"id": 4,
"name": "Romance"
},
"year_released": 2018,
"description": "A Star Is Born is a 1937 American Technicolor romantic drama film produced by David O. Selznick, directed by William A. Wellman from a script by Wellman, Robert Carson, Dorothy Parker, and Alan Campbell, and starring Janet Gaynor (in her only Technicolor film) as an aspiring Hollywood actress, and Fredric March (in his Technicolor debut) as a fading movie star who helps launch her career. The supporting cast features Adolphe Menjou, May Robson, Andy Devine, Lionel Stander, and Owen Moore.",
"imdb_rating": 4.5,
"video_file": "http://localhost:8000/media/movies/bunny.mp4",

Home Page


NETFLIX
Sign In
Unlimited films, TV programmes and more.
Watch anywhere. Cancel any time.
Email address
Try it now
Ready to watch? Enter your email to create or restart your membership.

Watch everywhere.

Stream unlimited movies and TV shows on your phone, tablet, laptop, and TV.

Create profiles for children.

Send children on adventures with their favourite characters in a space made just for them—free with your membership.

Frequently Asked Questions

What is Netflix? ×

Netflix is a streaming service that offers a wide variety of award-winning TV shows, movies, anime, documentaries and more – on thousands of internet-connected devices.

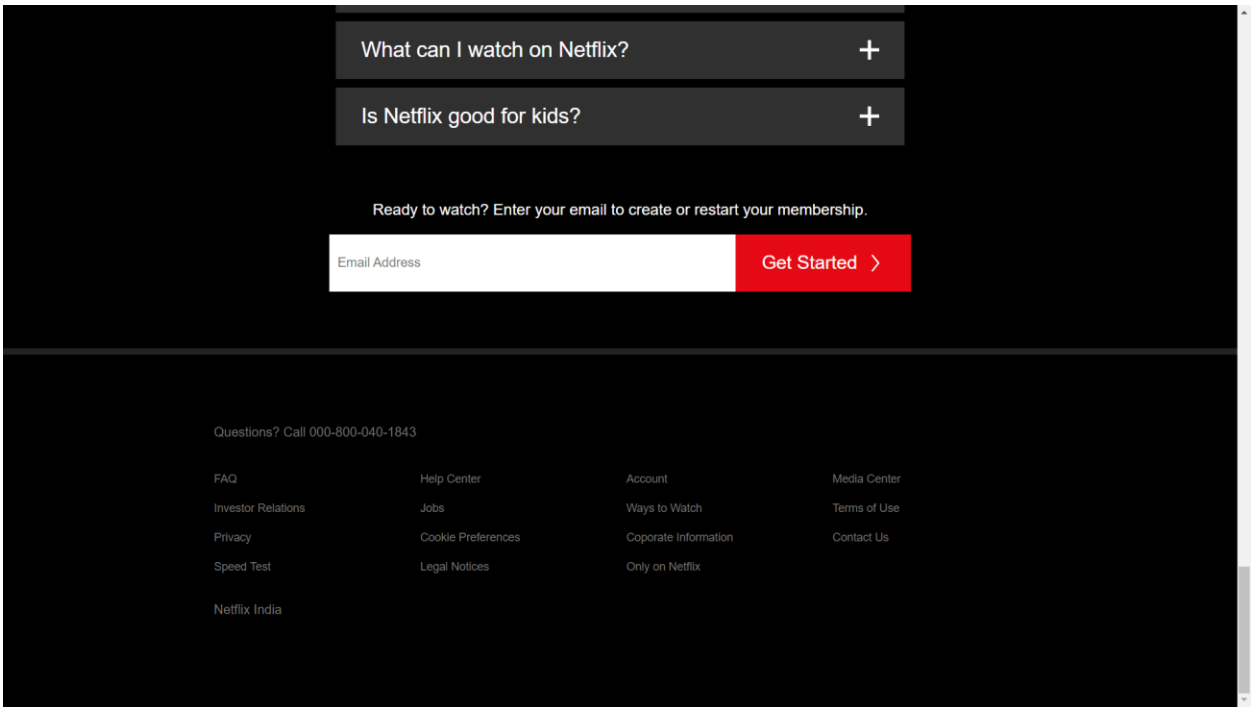
You can watch as much as you want, whenever you want, without a single ad – all for one low monthly price. There's always something new to discover, and new TV shows and movies are added every week!

How much does Netflix cost? +

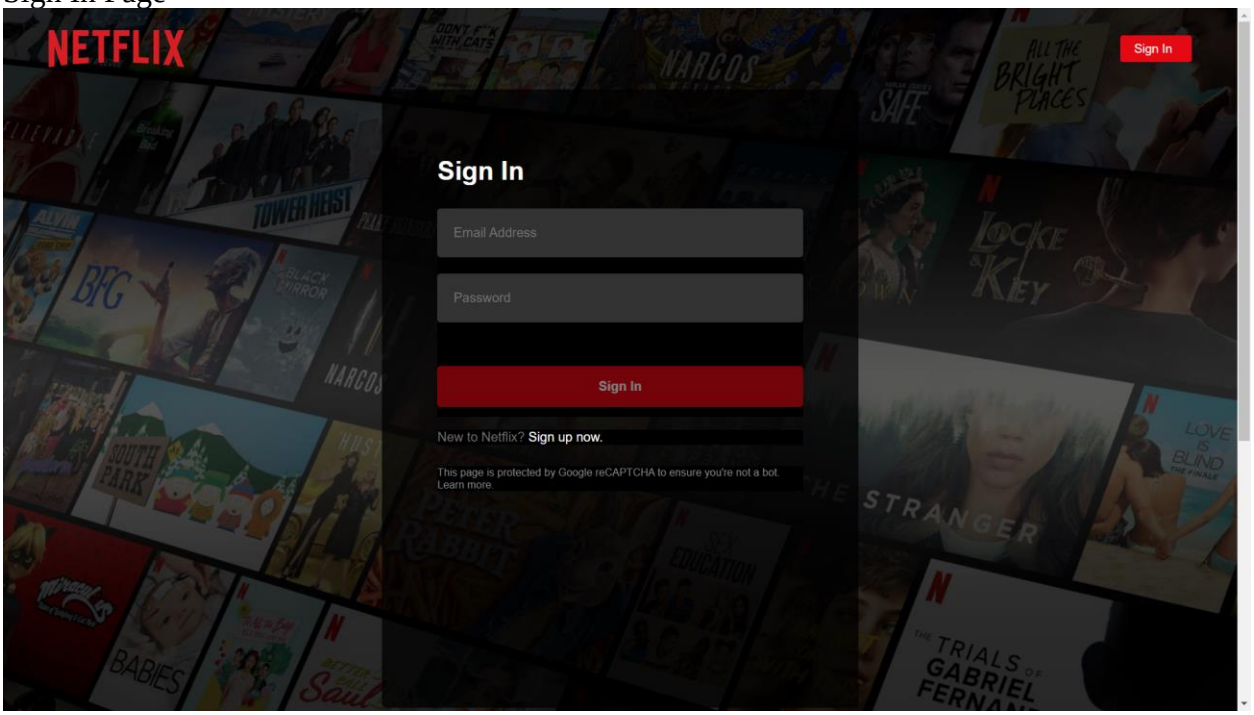
Where can I watch? +

How do I cancel? +

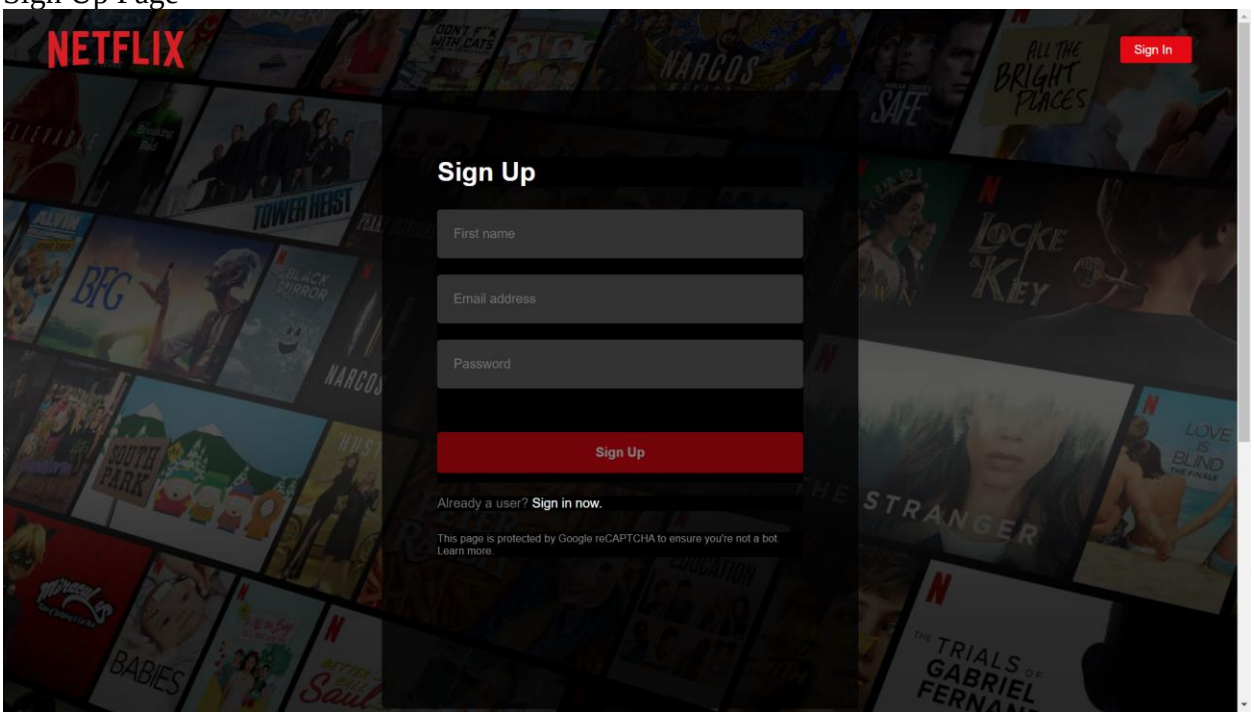
What can I watch on Netflix? +


What can I watch on Netflix?
Is Netflix good for kids?
Ready to watch? Enter your email to create or restart your membership.
Email Address
Get Started
Questions? Call 000-800-040-1843
FAQ
Help Center
Account
Media Center
Investor Relations
Jobs
Ways to Watch
Terms of Use
Privacy
Cookie Preferences
Coporate Information
Contact Us
Speed Test
Legal Notices
Only on Netflix
Netflix India

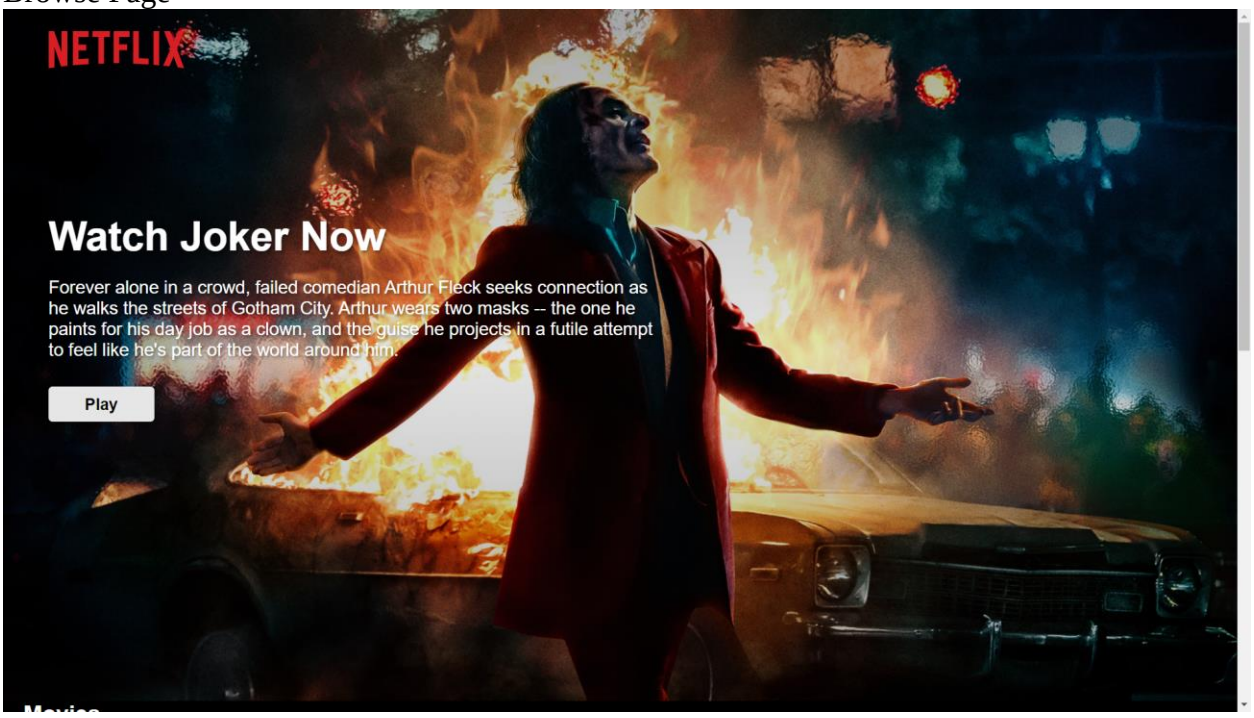
Sign In Page

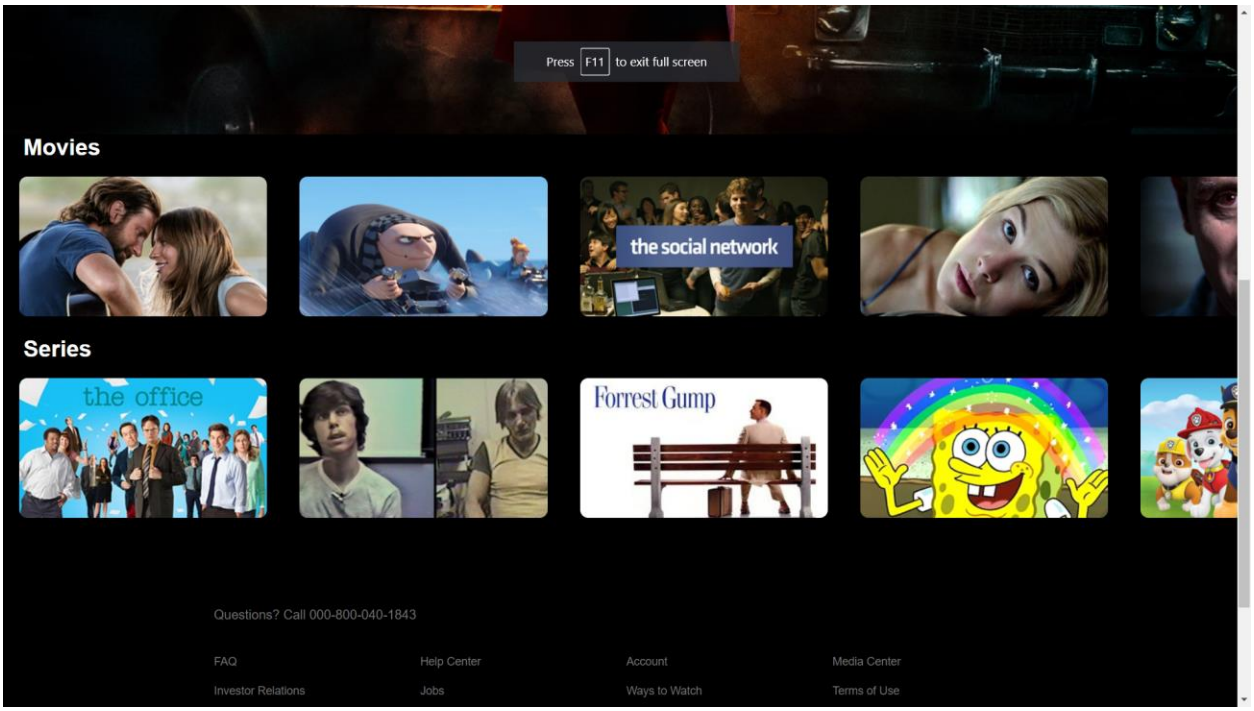

NETFLIX
Sign In
Sign In
Email Address
Password
Sign In
New to Netflix? Sign up now.
This page is protected by Google reCAPTCHA to ensure you're not a bot. Learn more.

Sign Up Page

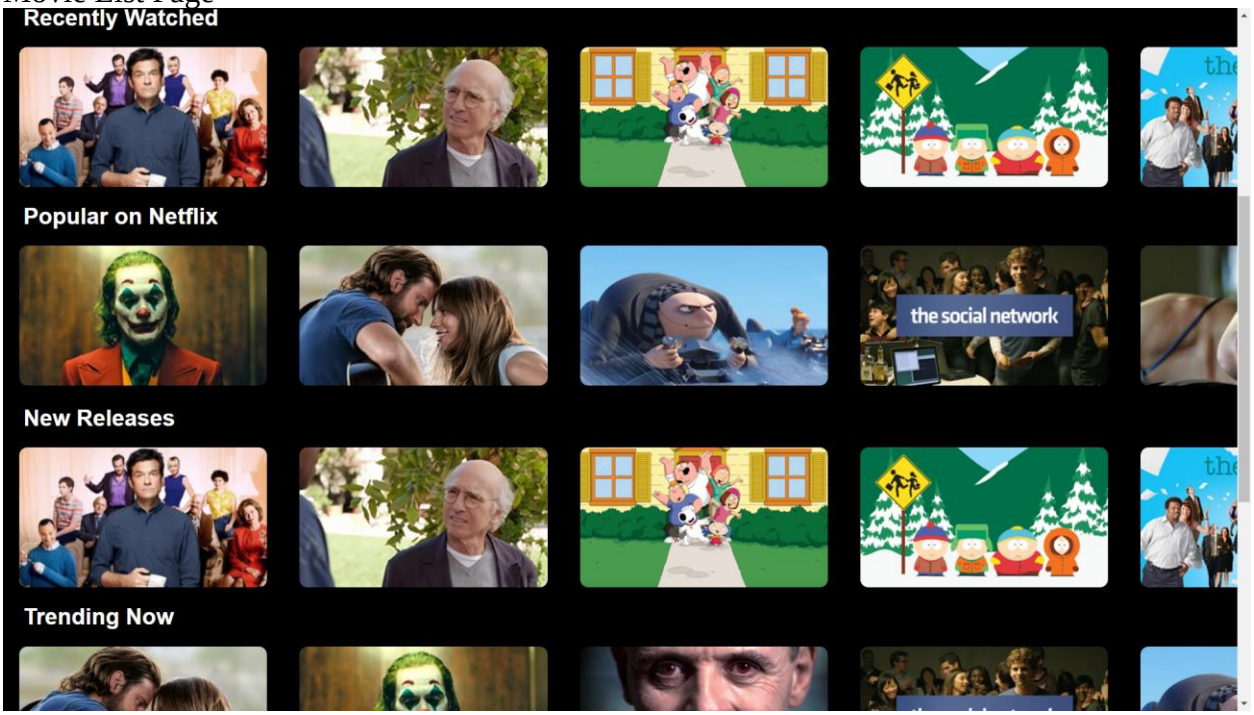


Browse Page

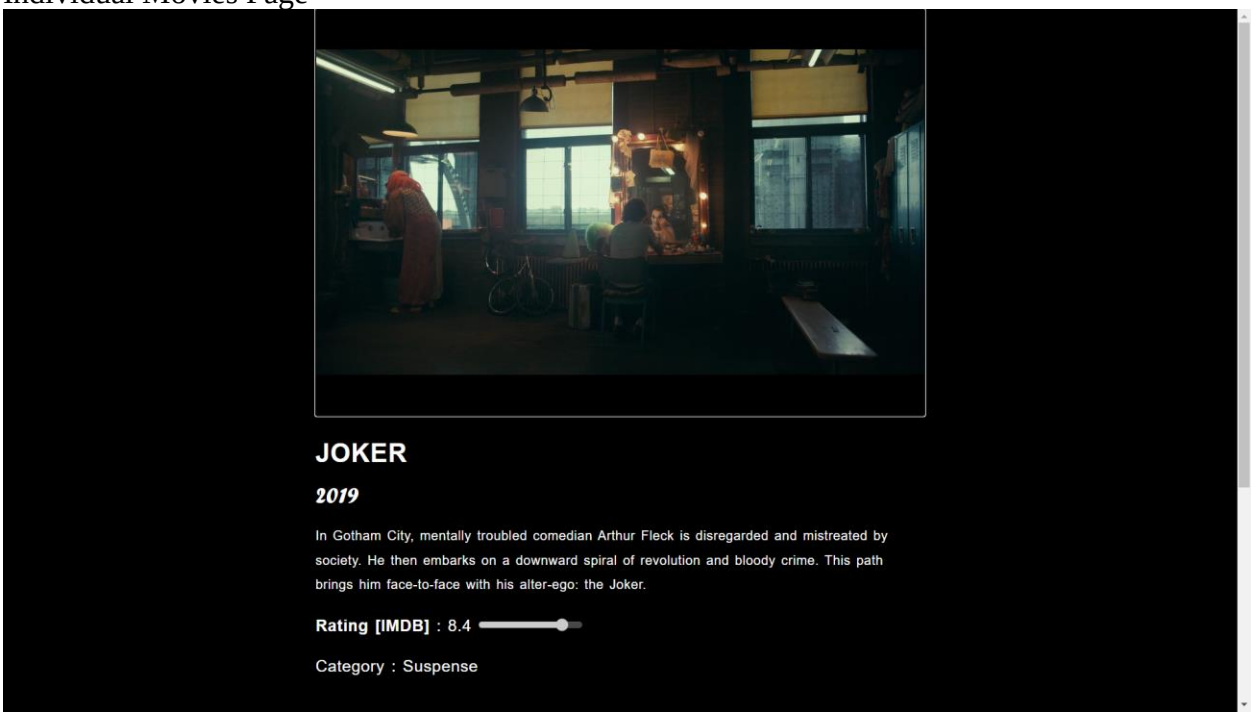




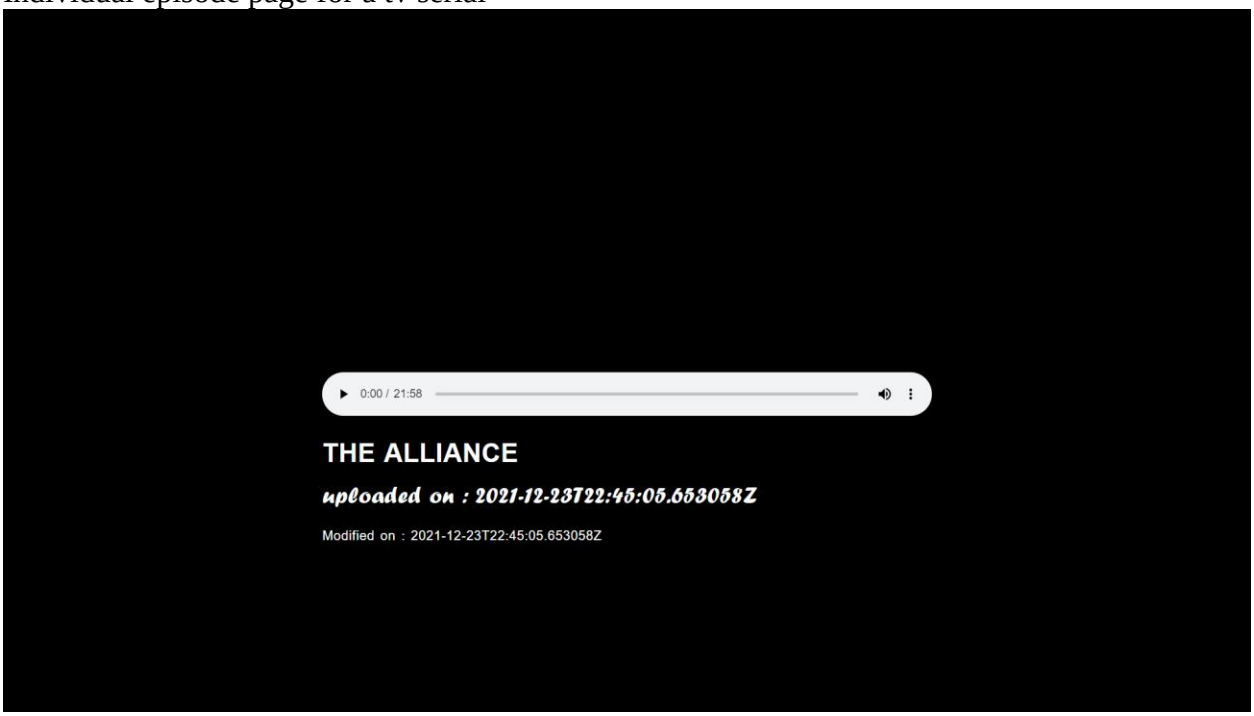
Movie List Page



Individual Movies Page



Individual episode page for a tv serial



Conclusion: This experiment helped us explore the django framework. We learned authentication, form handling, models, databases used with Django, creating APIs with Django rest framework and working with Django on the backend and React.js on the frontend.