



USAC
TRICENTENARIA
Universidad de San Carlos de Guatemala

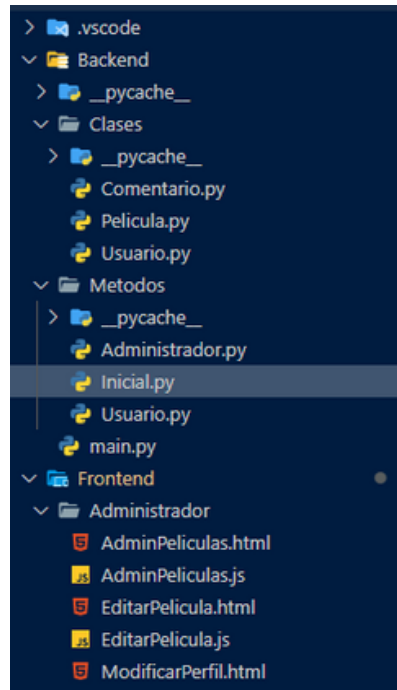
PROYECTO 2

MANUAL TÉCNICO

Introducción a la programación y computación 1

Miguel Adrian Tubac Agustin
202101927

Descripción de la solución



El programa resuelve el problema de un sitio de vistas de películas en donde se almacenan y se cargan películas, además se realiza el registro de diferentes usuarios al sistema. El sistema cuenta con diferentes opciones, resaltado el almacenamiento masivas de películas.

Requerimientos mínimos del entorno de desarrollo



Los requerimientos necesarios para la edición y ejecución del programa de gestión de envíos deberán ser los siguientes para poder garantizar la correcta depuración del mismo:

- Instalar el lenguaje de programación Python
- Modelo del procesador i5-6300U
- Procesador Intel® Core™ i5 de la sexta generación
- Capacidad de disco duro 250 GB
- Memoria RAM 8 GB
- Sistema Operativo Windows 10
- Conexión a una red de internet
- Contar con el acceso a un navegador web

Diccionario de clases

```
class Comentario:
    def __init__(self, pelicula, autor, mensaje):
        self.pelicula = pelicula
        self.autor = autor
        self.mensaje = mensaje

    def toDict(self):
        return {
            'pelicula': self.pelicula,
            'autor': self.autor,
            'mensaje': self.mensaje
        }
```

Pelicula: en esta clase se almacenan los datos correspondientes a las películas ingresadas.

```
class Pelicula:
    def __init__(self, nombre, genero, clasificacion, anio, duracion, link):
        self.nombre = nombre
        self.genero = genero
        self.clasificacion = clasificacion
        self.anio = anio
        self.duracion = duracion
        self.link = link

    def toDict(self):
        return {
            'nombre': self.nombre,
            'genero': self.genero,
            'clasificacion': self.clasificacion,
            'anio': self.anio,
            'duracion': self.duracion,
            'link': self.link
        }
```

```
class Usuario:
    def __init__(self, nombre, apellido, nombre_usuario, contrasena, tipo):
        self.nombre = nombre
        self.apellido = apellido
        self.nombre_usuario = nombre_usuario
        self.contrasena = contrasena
        self.tipo = tipo
```

Comentario: en esta clase se almacenan los datos correspondientes a los comentarios de los usuarios a las películas.

Usuario: en esta clase se almacenan los datos correspondientes a los diferentes usuarios que se ingresan al sistema.

Diccionario de métodos

Administrador: en esta parte se realiza la actualización de los usuarios que ingresan al sistema, esto debido a que unos son administradores.

```
You, hace 20 horas | 1 author (You)
from Clases.Pelicula import Pelicula

def CargarPelículas(datos):
    películas = []
    texto = datos['texto']

    # separacion del texto en líneas
    líneas = texto.split("\n")

    cont = 0
```

```
You, hace 20 horas | 1 author (You)
from Clases.Usuario import Usuario

def RegistrarUsuario(datos, usuarios):
    nombre = datos['nombre']
    apellido = datos['apellido']
    nombre_usuario = datos['nombre_usuario']
    contrasenia = datos['contrasenia']
    tipo = int(datos['tipo'])

    for usuario in usuarios:
        # si ya existe ese nombre de usuario
```

Inicial: en esta parte se efectúa la agregación de los datos personales de los usuarios, así mismo el estatus que el mismo representa.

Usuario: en esta parte se realiza la actualización de los usuarios que se encuentren en línea y su respectivo estatus dentro del sistema.

```
from Clases.Usuario import Usuario

def GetUsuarioEnSesion(usuarios, usuario):
    usuario = {}
    status = 400
    if(usuarioEnSesion != -1):
        usuario = usuarios[usuarioEnSesion]
        status = 200

    # si no existe el usuario retornar
```

```

from flask import Flask, request
from flask_cors import CORS

from Clases.Usuario import Usuario
from Clases.Comentario import Comentario
from Metodos.Inicial import *
from Metodos.Usuario import *
from Metodos.Administrador import *

usuarios = []

```

main: en esta parte se corre el programa, además es donde se inicia todo el programa.

Diccionario de Frontend

AdminPelicula: en esta parte se muestran las películas y se almacenan en el sistema.

```

AdminPeliculas.html x AdminPeliculas.js
Frontend > Administrador > AdminPeliculas.html >
You, hace 20 horas | 1 author (You)
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <title>Administrador: Peliculas</title>
6   <!--Script para la funcionalidad-->
7   <script src="AdminPeliculas.js"></script>
8   <!--Para poder usar bootstrap-->
9   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet">

```

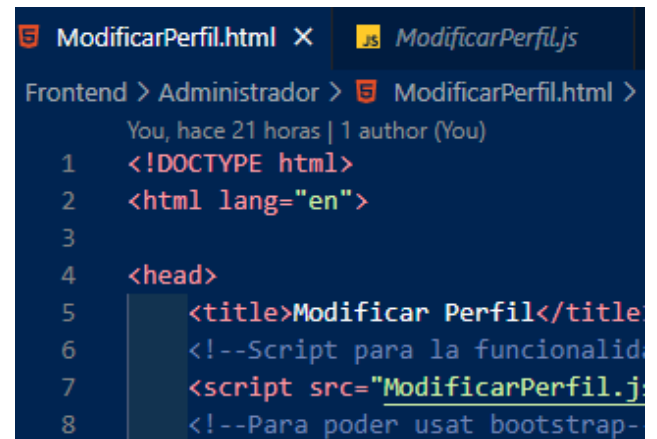
```

EditarPelicula.html EditarPelicula.js x
Frontend > Administrador > EditarPelicula.js >
You, hace 20 horas | 1 author (You)
1 var objeto = { 'nombrePelicula': 'S'
2
3 fetch(`http://localhost:3000/getPe
4   method: 'POST',
5   body: JSON.stringify(objeto),
6   headers: {
7     'Content-Type': 'applicati

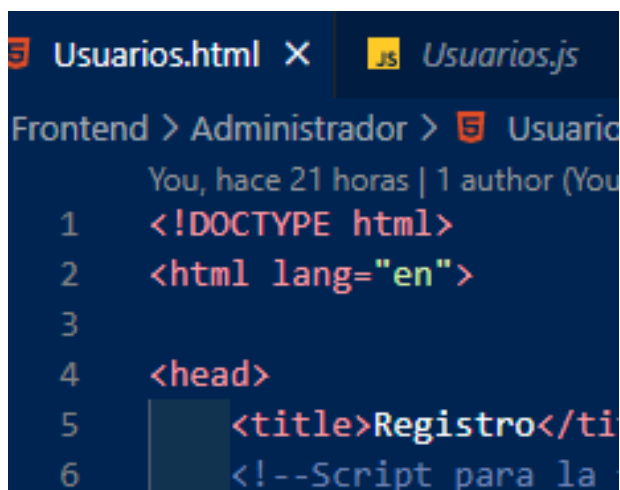
```

EditarPelicula: en esta parte se efectúa la edición de los datos que tenga la película y al mismo tiempo se almacena de forma permanente.

ModificarPerfil: en esta parte se modifica la información del personal que ingrese al sistema, tomando en cuenta únicamente al administrador.



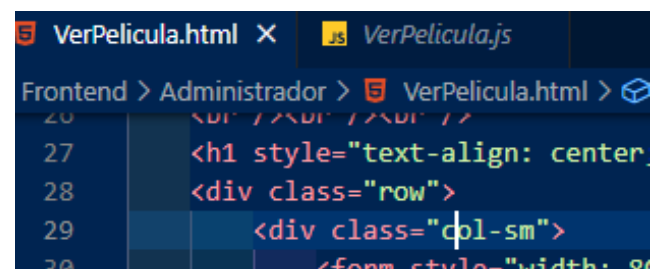
```
ModificarPerfil.html X JS ModificarPerfil.js
Frontend > Administrador > ModificarPerfil.html >
You, hace 21 horas | 1 author (You)
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <title>Modificar Perfil</title>
6   <!--Script para la funcionalidad-->
7   <script src="ModificarPerfil.js">
8   <!--Para poder usar bootstrap-->
```



```
Usuarios.html X JS Usuarios.js
Frontend > Administrador > Usuarios.html >
You, hace 21 horas | 1 author (You)
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <title>Registro</title>
6   <!--Script para la funcionalidad-->
```

Usuarios: en esta parte se muestran los usuarios que estén registrados en el sistema y la posibilidad de eliminarlos definitivamente del sistema.

VerPelicula: en esta parte se cuenta con la opción de ver la película y comprender el contenido del mismo.



```
VerPelicula.html X JS VerPelicula.js
Frontend > Administrador > VerPelicula.html >
20 <!--Script para la funcionalidad-->
27 <h1 style="text-align: center;">Ver Pelicula</h1>
28 <div class="row">
29   <div class="col-sm">
30   <form style="width: 80%;">
```



```
Frontend > Inicial > Login.html > html
You, hace 21 horas | 1 author (You)
1 <!DOCTYPE html>
2 <html lang="en">
3 |
4 <head>
5 |
6 <title>Login</title>
7 <!--Script para la fun
8 <script src="Login.js"
9 <!--Para poder usat bo
10 <link href="https://cd
```

Login: en esta parte se corre el inicio de sesión y la opción de recuperación de contraseña que se encuentra en el sistema.

RecuperarContrasenia: en esta parte se muestran la opción de recuperar la contraseña con únicamente el usuario que se registró al sistema.

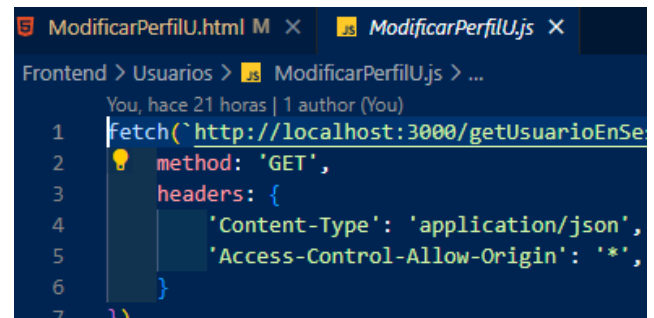
```
Frontend > Inicial > RecuperarContrasenia.html > html > html
You, hace 21 horas | 1 author (You)
1 <!DOCTYPE html>
2 <html lang="en">
3 |
4 <head>
5 |
6 <title>Recuperar Contraseña</title>
7 <!--Script para la funcionalidad-->
8 <script src="RecuperarContrasenia.js"></script>
```

```
Frontend > Inicial > RegistrarCliente.html > html
You, hace 21 horas | 1 author (You)
1 <!DOCTYPE html>
2 <html lang="en">
3 |
4 <head>
5 |
6 <title>Registro</title>
7 <!--Script para la funcionalidad
```

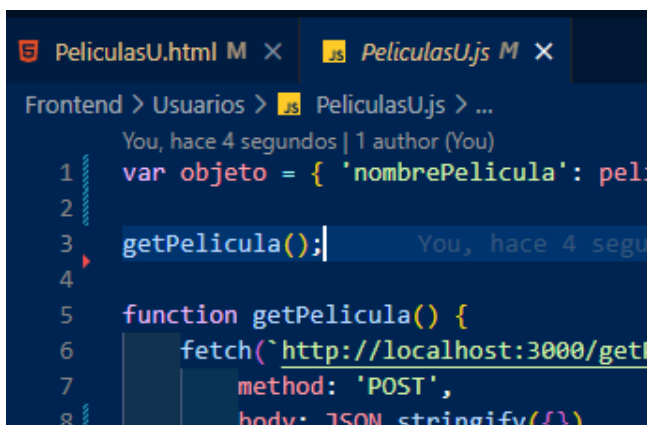
RegistrarCliente: en esta parte se efectúa el registro de clientes o usuarios comunes y al mismo tiempo se almacena en la base de datos del sistema.



ModificarPerfilU: en esta parte se modifica la información del usuario que ingrese al sistema y se almacena en la base de datos.



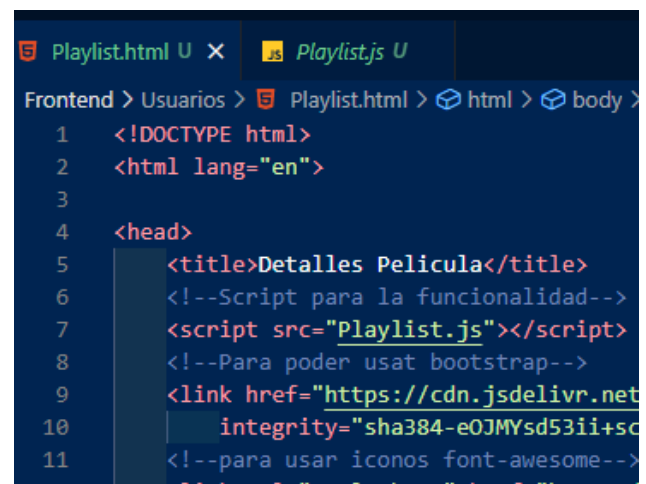
```
1 fetch('http://localhost:3000/getUsuarioEnSe')
2   .method: 'GET',
3   .headers: {
4     'Content-Type': 'application/json',
5     'Access-Control-Allow-Origin': '*',
6   }
7 }
```



```
1 var objeto = { 'nombrePelicula': peli
2
3 getPelicula();
4
5 function getPelicula() {
6   fetch('http://localhost:3000/getP
7     .method: 'POST',
8     .body: JSON.stringify({}),
```

PeliculasU: en esta parte se muestran las películas al público en general y se cuenta con las diferentes opciones del código.

Playlist: en esta parte se almacenan las películas que el cliente prefiere y se muestran al público.



```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <title>Detalles Pelicula</title>
6   <!--Script para la funcionalidad-->
7   <script src="Playlist.js"></script>
8   <!--Para poder usar bootstrap-->
9   <link href="https://cdn.jsdelivr.net
10     integrity="sha384-eOJMYsd53ii+sc
11   <!--para usar iconos font-awesome-->
12   <link rel="stylesheet" href="https://
```