Escuela Superior Politécnica del Litoral

Administración de Sistemas y Servicios en Red

Aplicación web para streaming de video usando API-REST

Paralelo 1 Grupo 1

Autores:
Daniela García
Aarón Nicolalde
Jhostin Ocampo
Romulo Coronel

Septiembre 2020

1 Descripción del Problema

Nuestra página web ofrece un servicio que distribuye digitalmente contenido multimedia a través de internet. El concepto streaming nos permite ver un archivo de audio o video, sin descargarlo directamente a nuestra computadora, solo se ven los videos subidos a una base de datos y presentados en un servidor web, que a través de un buffer de datos va almacenando el flujo de descarga de los datos multimedia, mientras se va reproduciendo. Nuestro proyecto ofrece una solución para lo anteriormente descrito, y así evitar las descargas directas a nuestro equipo, lo que nos da un mayor nivel de seguridad ya que una transmisión en streaming no se almacena la información en el equipo, y así evitamos que se capturen o se pierdan datos de audio o video.

2 Diagramas de diseño del proyecto

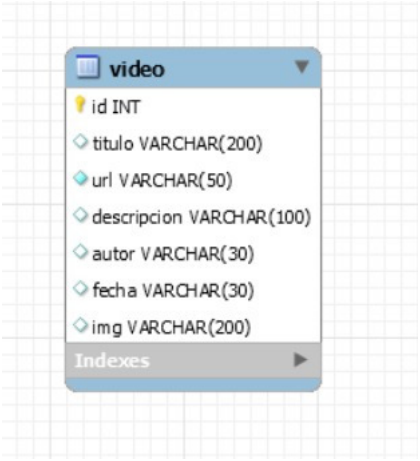

Figure 1: Diagrama entidad-relación de la base de datos de ESPOLflix

Como backend se tiene una base de datos creada en MySQL Workbench dónde se aloja los videos con los atributos id, titulo, foto, url y descripción , como API Rest se utiliza el framework express que está alojado en el entorno de ejecución de NodeJS , express se encarga de realizar la comunicación a la base de datos y al frontend .

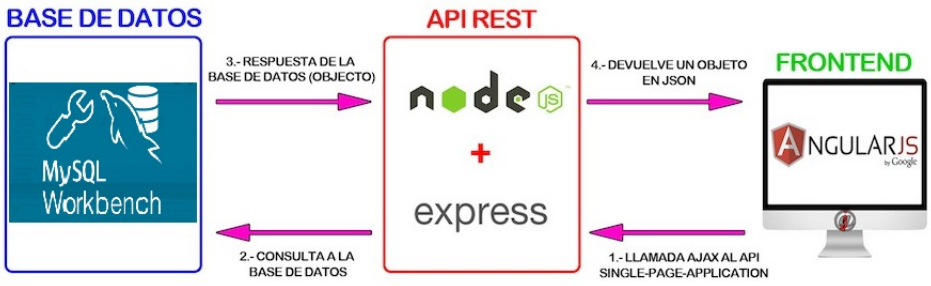

Figure 2: Diagrama de diseño

El framework para diseñar el sitio web es Angular dónde en la página principal estará las fotos de los distintos videos así como un título y la descripción , al hacer clic en la foto del video se hace una llamada al api quién realiza la consulta a la Base de Datos según el id del video , MySQL devuelve todos los campos pertenecientes a ese id , el API lo devuelve en formato JSON y en Angular se toma la url y se proyecta el video en el sitio web .

3 Recursos de Hardware

Requerimientos de Hardware AngularJs, NodeJs , Express se alojarán usando el entorno de desarrollo integrado de Visual Code el cuál tiene como requisitos :

- Procesador de 1,6 GHz o superior.
- 1 GB de RAM (1,5 GB si se ejecuta en una máquina virtual)
- 600 MB de espacio disponible en el disco duro.
- Unidad de disco duro de 5400 rpm.
- Tarjeta de vídeo compatible con DirectX 9 con una resolución de pantalla de 1024 x 768 o superior.

Para MySQL Workbench se tiene como requisitos de hardware :

- Procesador doble núcleo de 2 Ghz (4 núcleos recomendado)
- 4 gigabytes de RAM (6 gigabytes recomendado).
- Pantalla con una resolución mínima de 1024768 píxeles (12801024 recomendado).

4 Recursos de software

- **Node JS:** Node.js es un entorno de tiempo de ejecución de JavaScript, que nos ayudó a modelar nuestra página web.

- **MySQL:** Para manejar nuestra base de datos utilizamos MySQL Workbench, que nos permitió crear y almacenar las tablas donde tendríamos los videos que se iban a presentar en la página web. Aquí pudimos definir diferentes campos de los videos, como título, url, una breve descripción, autor, fechas, y la imagen de miniatura.
- **Angular:** Se usó el framework de Angular para el frontend y el backend de nuestra aplicación. La ventaja de usar angular es que ofrece una forma ordenada y eficaz, gracias a su modelado vista-controlador que permite modificación y actualizaciones.
- **Express:** Esta herramienta nos ayudó en la creación del API. Express es una aplicación web de node js.

5 Funcionamiento

El diseño de la página web queda de la siguiente manera como se observa en la figura 3. Nuestra página web Espolflix muestra en la pantalla principal las cartillas con los videos que se obtienen de la base de datos, se puede observar que aparece el nombre del autor, el título del video y la imagen de portada

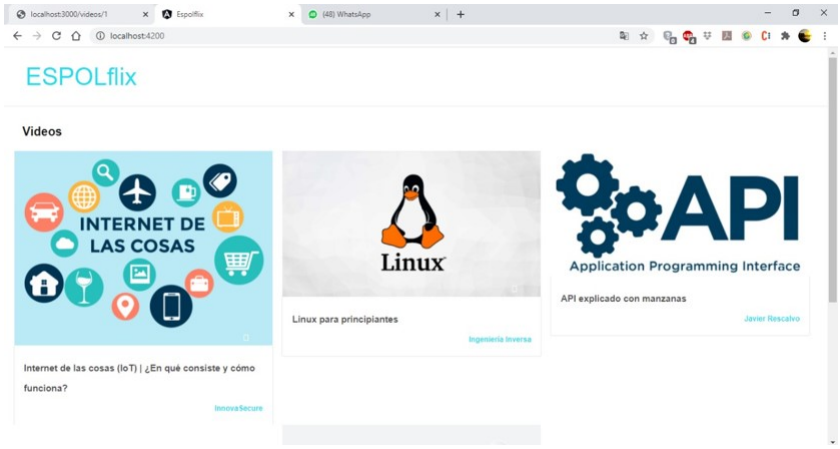

Figure 3: Página principal de ESPOLflix

Al dar click en un video, nos dirigimos a la página de presentación del video , en esta se puede observar también el título del video , la fecha de creación del mismo , una descripción y el video.

Figure 4: Página secundaria de ESPOLflix

6 Explicación de la solución

El funcionamiento de ESPOLflix se divide en dos partes, el frontend y el backend. Tanto el frontend como el backend fueron realizados mediante el editor de codigo visual studio code. Para el backend se utilizó un api-rest con node js, express y mysql. Para empezar se deben instalar las dependencias mediante el uso del comando "npm install". Las dependencias serán instaladas en el package.json como se visualiza en la figura 5.

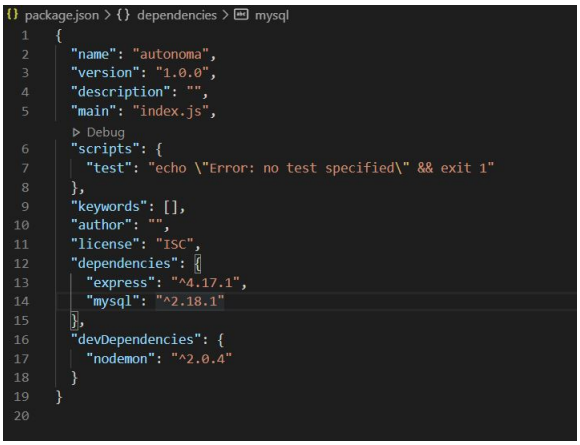

Figure 5: Package.json del api-rest de ESPOLflix

A partir de estas se crea una página inicial con extensión js que nombramos como index.js la cual se muestra en la figura 6. Se utilizaron métodos use y get para determinar las rutas y los datos a mostrar, los cuales requerirán de una

ruta a la carpeta routes en donde se encontrará el archivo videos.js. Además se declara un método listen para escuchar al puerto 3000 en este caso.

Figure 6: index.js del api-rest de ESPOLflix

El archivo video.js obtiene las funciones getVideos y getUrlVideo del archivo video.controller.js localizado en la carpeta controller como se muestra en la figura 7.

Figure 7: video.js del api-rest de ESPOLflix

En el archivo video.controller.js mostrado en la figura 8, se encuentran declaradas las funciones con sus respectivos querys de consulta a la base mysql, para esto se requiere de una conexión a la base de datos del proyecto la cual se encuentra en la ruta "../config/database.js".

Figure 8: video.controller.js del api-rest de ESPOLflix

Para finalizar se llama al archivo database.js en el cual se realiza la conexión a nuestra base de datos local del proyecto en donde se aloja la tabla de videos mostrada en la figura 1. El código necesario para realizar la conexión se muestra a continuación en la figura 9. Cabe recalcar que es de suma importancia colocar la línea de código "insecureAuth:true" debido a que de no hacerlo no se permitirá el acceso a la base de datos por lo cual no se mostraría la información en el api-rest.

Figure 9: database.js del api-rest de ESPOLflix

Para finalizar se corré el api-rest mediante el comando "node index.js" y se visualizan en las rutas "localhost:3000/videos" mediante la función getVideos() los videos pertenecientes a la tabla videos, por otro lado en la ruta "localhost:3000/videos/:id" en donde el id mostrara la información correspondiente a la tupla con el id indiciado. Se muestran los resultados en la figura 10 y figura

11 respectivamente.

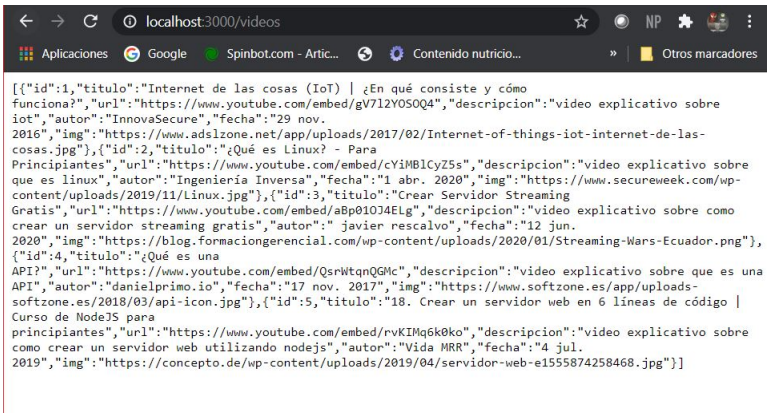

Figure 10: Información de la tabla videos obtenida mediante el api-rest

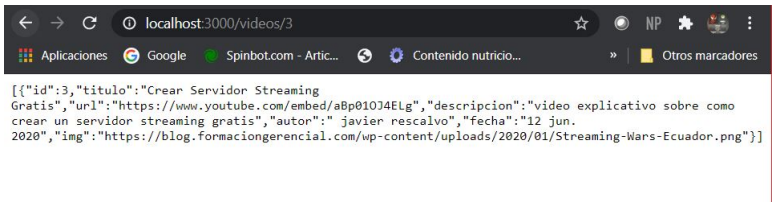

Figure 11: Información de una tupla de la tabla videos obtenida mediante el api-rest

El frontend fue desarrollado con angular, para la creación de la interfaz de usuario fue necesario que nuestro frontend consuma al api-rest previamente creado, por lo cual en el archivo video-service.service se coloca el url de la api rest en este caso "localhost:3000/videos" el cual tiene extension ts ya que es un formato para almacenar y transmitir datos de audio y video como se muestra en la figura 12.

Figure 12: Código del archivo video-service.service.ts para consumir al api-rest

En este momento en nuestro frontend ya se pueden invocar a los atributos o directamente al objeto video mediante su id. Para mostrar nuestra página de inicio utilizamos un archivo con formato html el cual se muestra en la figura 13.

Figure 13: Código de la cabecera inicial de la página ESPOLflix

Para mostrar las diferentes opciones de videos para su visualización de manera streaming, se configuró el archivo dashboard en donde colocamos los objetos de nuestra base datos y de esta forma mostrarlo como se muestra en la figura 14.

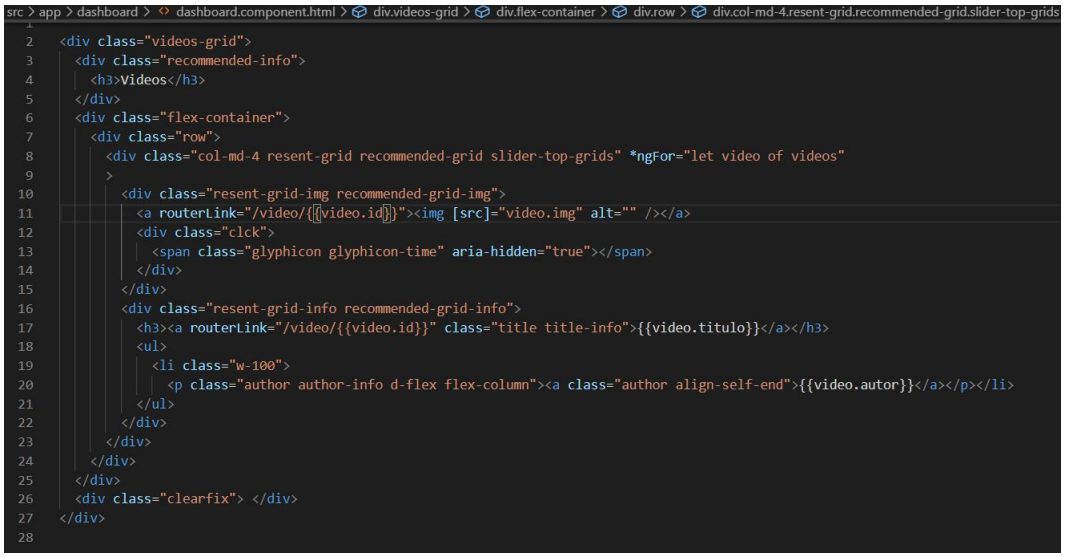

Figure 14: Código del dashboard de la página ESPOLflix

Para reproducir cada video se colocan mediante el id en el archivo video.component.html en donde se obtienen los atributos de cada tupla mediante el id seleccionado en la pagina principal. Se muestra el código para obtener dichos atributos en la figura 15.

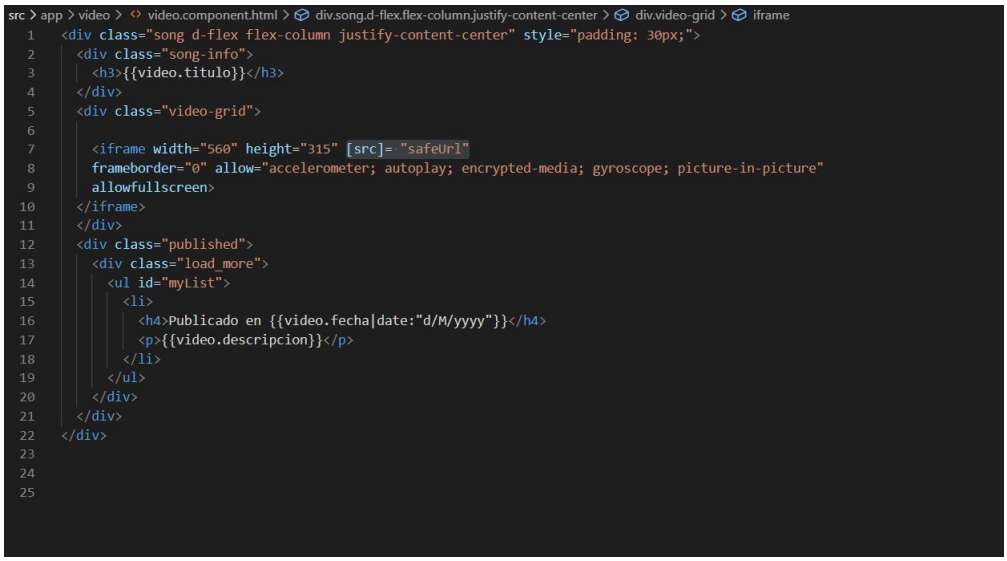

Figure 15: Código del video.component de la página ESPOLflix