

Universidad San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y sistemas
Estructuras de Datos
Ingeniero:

- Ing. Luis Espino

Auxiliar:

- Wilfred Perez



Proyecto 2

"Moviecats"- Almacenamiento dinámico

Índice

Objetivos	3
Objetivo general	3
Objetivos específicos	3
Descripción	4
Página Web	4
Login	¡Error! Marcador no definido.
Home	¡Error! Marcador no definido.
Administrador	¡Error! Marcador no definido.
Vista librerías	¡Error! Marcador no definido.
Vista libros	¡Error! Marcador no definido.
Vista de autores	¡Error! Marcador no definido.
Observaciones	17
Entregables:	17
Restricciones	17
Fecha de Entrega:	17

Objetivos

Objetivo general

- Aplicar los conocimientos del curso Estructuras de Datos en el desarrollo de una aplicación que permita manipular la información de forma óptima.

Objetivos específicos

- Demostrar los conocimientos adquiridos sobre estructuras de datos lineales poniéndolos en práctica en una aplicación de simulación.
- Utilizar el lenguaje de JavaScript para implementar estructuras de datos lineales y visualizarlas en una página Web.
- Utilizar una herramienta que permita graficar las estructuras de datos lineales.
- Definir e implementar algoritmos de búsqueda, recorrido y eliminación.

Descripción

“Moviecats” es una conocida franquicia guatemalteca especializada en alquiler de cine y videojuegos a través de tiendas físicas, servicios por correo y video bajo demanda. La empresa fue fundada en el año 2019, contaba con más de 50 establecimientos a nivel nacional.

“Moviecats” se vio lastrada por la pandemia que vino a restringir el acceso a todas franquicias, haciendo que las ventas empezarán a disminuir haciendo que la empresa se viniera a pique, los directivos de la empresa propusieron transformar su negocio y competir con otros servicios de Streaming como, por ejemplo: Disney+, Netflix, HBO, DAZN. Por lo que se le solicita a usted que ha desarrollado la aplicación de CatsBooks exitosamente, para que pueda implementar esta nueva aplicación Web, con el fin de agilizar, y brindar un nuevo servicio a los clientes. Para ello dicha aplicación web estará desarrollada en el lenguaje de JavaScript.

Página Web

La aplicación que se desarrollará será una versión en la cual se podrá ver la estructura implementada junto con la aplicación de esta, para corroborar que el funcionamiento sea el correcto.

Para esto se tendrán dos vistas, una para el administrador y otra para el usuario, las cuales se detallarán a continuación:

Administrador:

Carga masiva de películas las cuales serán almacenadas en un árbol AVL

Carga masiva de clientes las cuales se almacenarán en una lista simple.

Carga masiva de Actores icónicos de distintas películas, estos serán almacenados en un árbol binario, cada película será única, no tendrá copias.

Carga masiva de Categorías las cuales serán almacenadas en una tabla Hash.

Ver las diferentes estructuras luego de haber cargado cada uno de los archivos de entrada.

Podrá modificar el tiempo de creación de un nuevo bloque en el blockchain

Podrá generar un nuevo bloque pulsando un botón, el cual añadirá un bloque a la blockchain y reiniciará el tiempo de creación.

Ver el **único** árbol de merkle en el apartado de blockchain con el cual se validará la correcta integración del árbol en cada bloque de la blockchain.

Login:

La aplicación contará con un login el cual será la primera vista que se mostrará, no se permitirá entrar a ninguna persona que no esté dentro del sistema de clientes o administrador.

Esta ventana deberá tener:

- Caja de texto en donde se ingresará el usuario.
- Caja de texto donde se ingresa la contraseña.
- Un botón o checkbox que validara si es un usuario o administrador.
- Un botón para ingresar.

Ingresar

User Name:

Password:

☐ Administrador

SIGN IN

Nuevo usuario

El usuario administrador contará con la siguiente información:

dpi: 2354168452525
Nombre completo: Wilfred Perez
Nombre usuario: EDD
Contraseña: 123
Teléfono: +502 (123) 123-4567

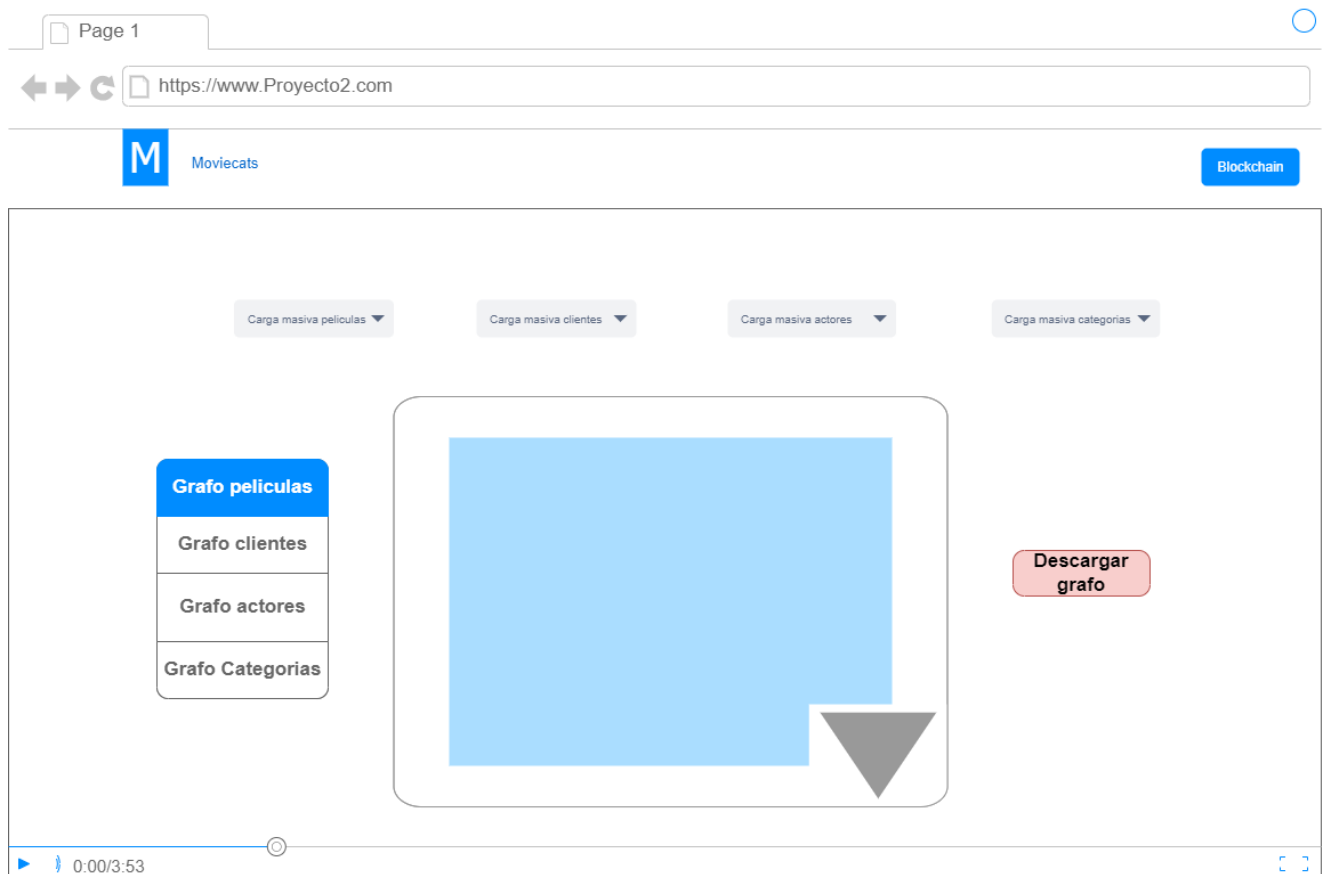
Vista de Administrador

La vista deberá contar con los siguientes elementos:

- Botón para la carga masiva de películas.
- Botón para la carga masiva de clientes.
- Botón para salir de la sesión.
- Botón para la carga masiva de actores.
- Botón para la carga masiva de categorías.
- Botón para descarga de las estructuras.
- Botón para ver el flujo Blockchain
- Un contenedor con scroll donde se pueda apreciar la estructura seleccionada luego de realizar cada una de las estructuras.
- Cada grafo generado tendrá la opción de descargarse con un botón en formato PNG .png

Vistas del administrador:

- Principal



Blockchain

La vista deberá contar con los siguientes elementos:

- Una caja de texto donde se ingresará la cantidad de segundos que se requieren para la creación de un nuevo bloque.
- Un botón con el cual se modifica el tiempo.
- Un botón con el cual se generará un nuevo bloque al ser presionado.
- Dos contenedores o divs en los cuales se mostrarán las estructuras de blockchain y el árbol de merkle.

Prueba de trabajo

Es el proceso por el cual se encuentra un hash que cumpla con la condición de tener un prefijo de n ceros, para este proyecto se utilizaran 2 ceros. Para ello se debe de iterar un entero denominado NONCE hasta encontrar un hash válido para el bloque.

Operaciones de Blockchain

Pasado el tiempo de configuración en la aplicación se genera un nuevo bloque que almacena la sumariación del árbol de merkle anteriormente descrito. El tiempo en que un nuevo bloque se genera será representado por un entero que representa la cantidad de segundo, este puede modificarse y por defecto tendrá el valor de 300 segundos.

Bloque

- **TIMESTAMP:** Es la fecha y hora exacta en la que se creó el bloque. Debe de tener el siguiente formato: (DD-MM-YY-::HH:MM:SS).
- **DATA:** Contendrá todas las acciones de los usuarios cuando realicen el alquiler de una película.
- **NONCE:** Será el número entero que se debe iterar de uno en uno hasta encontrar un hash que cumpla con la prueba de trabado.
- **PREVIOUSHASH:** Es el hash del bloque previo, este es necesario para validar que la cadena de bloques no esté corrupta. En caso del bloque génesis, el hash anterior debe de ser 00.
- **ROOTMERKLE:** En este bloque se almacena el nodo padre del árbol de Merkle. Este árbol de Merkle se forma con los datos del campo DATA, que son las operaciones de alquiler.
- **HASH (bloque actual):** El hash que protege que la data no se ha comprometido, el hash deberá generarse aplicando la función SHA256 a las propiedades: INDEX, TIMESTAMP, PREVIOUSHASH, ROOTMERKLE y NONCE **todas estas propiedades como cadenas concatenadas sin espacios en blanco ni saltos de línea.** Es decir, SHA256(INDEX+TIMESTAMP+PREVIOUSHASH+ROOTMERKLE+NONCE). Para considerar el hash como válido este debe de tener un prefijo de dos ceros. Es decir que un hash valido sería el siguiente:

006282b12041cb5a7bac8ec90f86b654af6b1ac8bfc5ed08092e217235df0229

Árbol Merkle

Cada nodo de este árbol se generará al realizar un alquiler de cualquier película. Cada nodo del árbol deberá almacenar el resultado de aplicarle la función hash SHA256.

Ejemplo:

`id = funcionHash(Data)`

Donde:

- `id`: resultado de la función sha256, este será el valor del nodo del árbol.
- `funcionHash`: Se utilizará la función sha256 para encriptar los datos.
- `Data`: Corresponde a toda la información que generará el hash, debe incluir los datos al realizar el alquiler de la película los cuales son: Nombre cliente y nombre película, la inserción de cada nodo será por cada transacción realizada.

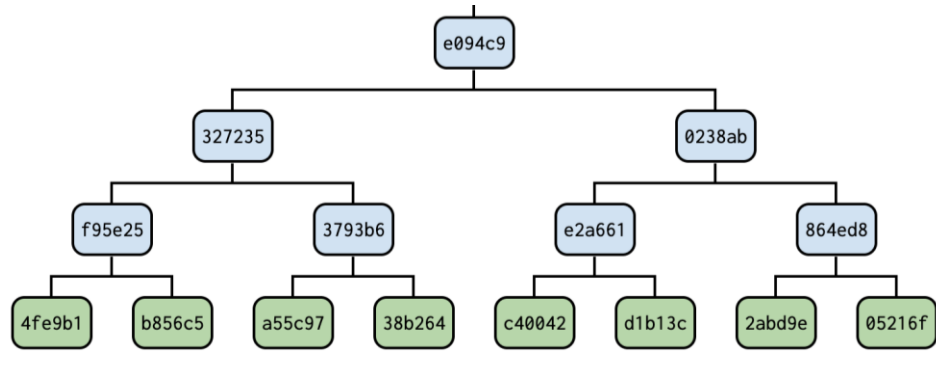
Ejemplo:

```
id = funcionHash256("Paula - A través de mi ventana");  
id= a516ds56fadafd561daf651afd165ad1f5daf5
```

el nodo contendrá este valor.

Se utilizará un árbol de orden exponencial 2, utilizando el algoritmo visto en clase.

Ejemplo de cómo se ve la estructura del árbol de merkle, no coincide con el ejemplo.



Vista usuario

Un usuario podrá realizar las siguientes acciones:

Rentar una película

Calificación de la película basado en estrellas [1-5]

En cada película se podrán dejar comentarios sobre la película, estos se deben almacenar y podrán verse por otros usuarios.

El usuario podrá filtrar el listado de películas.

Vista principal usuario

La siguiente vista contará con las siguientes funciones:

- Un botón con el cual se podrán ordenar la lista de películas por nombre de forma ascendente y descendente, utilizando el nombre del título para ordenarlas.
- Un texto en el cual se verá la descripción de cada película almacenada en la carga masiva de películas.
- Cada película contará con dos botones, uno para ver toda la información de la película y otra para alquilar la película junto con el precio de cada película.
- Mostrar el título en una etiqueta de texto, no es necesario agregar imagen
- Botón para ver las categorías de todas las películas.
- Botón para ver a todos los actores de todas las películas.

Page 1

https://www.Proyecto2.com

M
Moviecats

Bienvenido Usuario

Moviecats

Ordenar

Ver actores

Ver categorias

<div> <div>Titulo</div> <div>pelicula</div> </div>	Descripcion: Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.	<div> <div>?</div> <div>Informacion</div> </div>	<div> <div></div> <div>Alquilar</div> </div>	Q 000.00
<div> <div>Titulo</div> <div>pelicula</div> </div>	Descripcion: Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.	<div> <div>?</div> <div>Informacion</div> </div>	<div> <div></div> <div>Alquilar</div> </div>	Q 000.00
<div> <div>Titulo</div> <div>pelicula</div> </div>	Descripcion: Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.	<div> <div>?</div> <div>Informacion</div> </div>	<div> <div></div> <div>Alquilar</div> </div>	Q 000.00
<div> <div>Titulo</div> <div>pelicula</div> </div>	Descripcion: Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.	<div> <div>?</div> <div>Informacion</div> </div>	<div> <div></div> <div>Alquilar</div> </div>	Q 000.00

0:00/3:53


Vista por película

- Título y descripción de la película
- Un cuadro de texto en el cual se podrá ingresar un nuevo número para la puntuación de la película.
- Un botón con el cual podrá alquilar la película.
- Se contará con una caja de texto y un botón con el cual usted podrá agregar un comentario, este comentario podrá ser visible por todos los clientes que visiten esa película, **para esta funcional podrá utilizar una lista simple, vector, o array para almacenar cada comentario.**

Page 1

https://www.Proyecto2.com


M Moviecats

 Bienvenido Usuario


Titulo película

Descripcion: Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

Modificar puntuacion



★ ★ ★ ★ ☆


Alquilar

Q 000.00

Comentarios


▶ Alex: muy buena pelicula

▶ Alex: recomendada

▶ *ingrese su comentario*

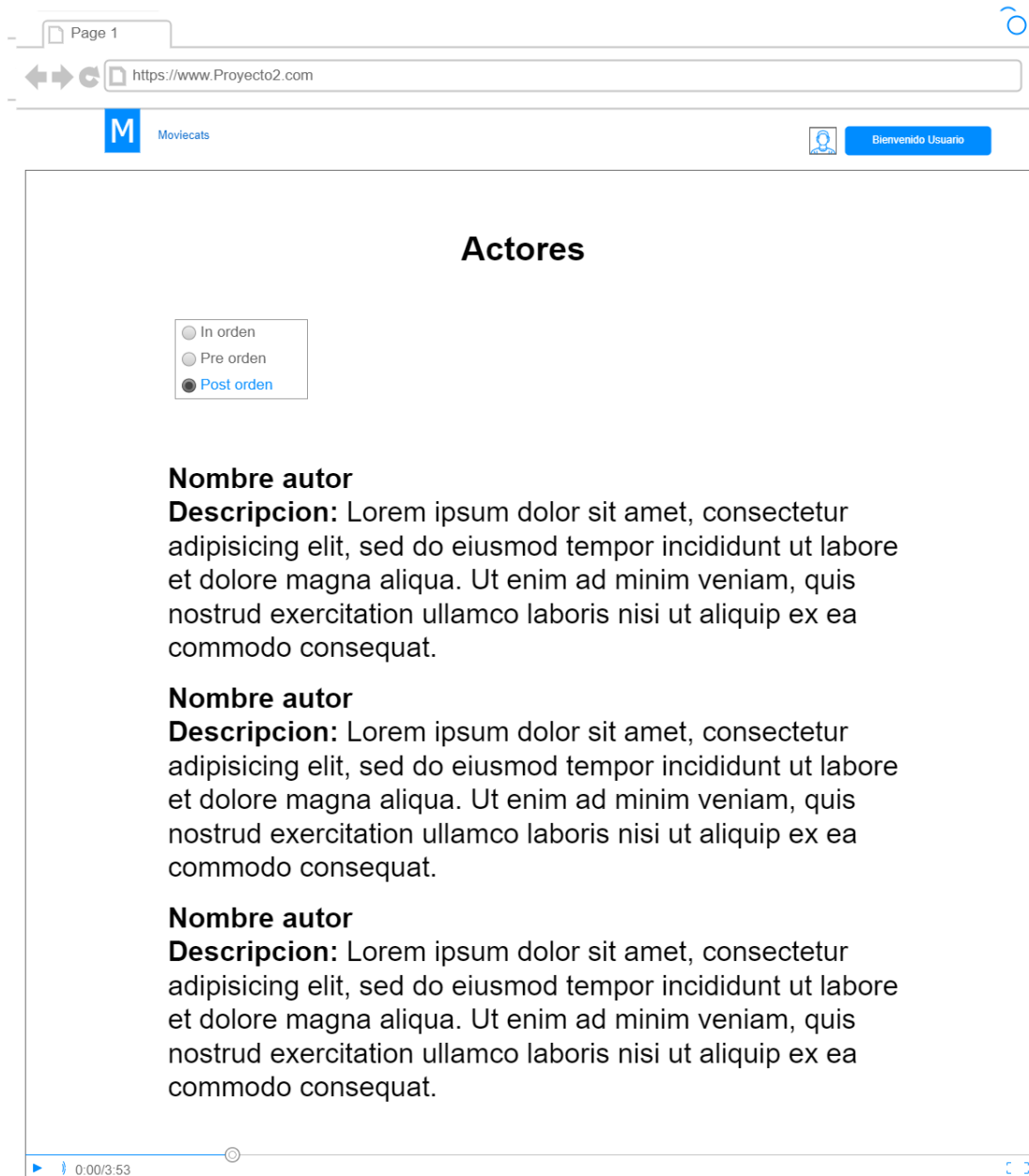
Publicar

▶ 0:00/3:53



Vista de actores

- Un botón con el cual podrán seleccionar el ordenamiento con el cual mostraran a todos los actores ingresados
- Se listarán los actores según el ordenamiento seleccionado.



Vista categorías

- Desglosa un listado de categorías previamente ingresadas.

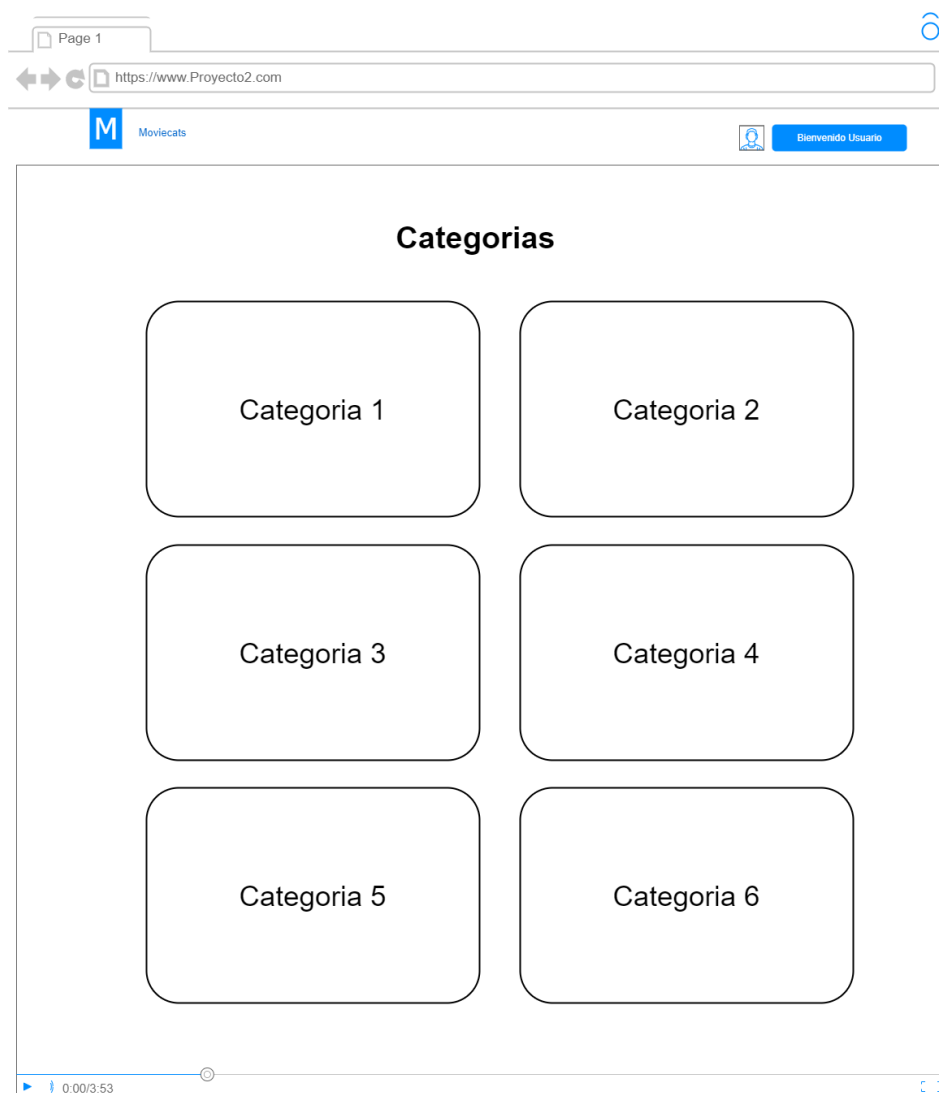
Nota: No tiene funcionalidad.

El tamaño de la tabla hash será de 20 posiciones, al sobrepasar el 75% realizar un rehashing aumentando el tamaño en 5 posiciones por cada ocupación.

Para las colisiones de la tabla Hash se utilizara el método de hashing enlazado, lista de listas.

Ecuación para inserción en la tabla hash:

$$\text{Valor} = \text{id_categoria} \% 20$$



Carga masiva

- Carga masiva de películas las cuales serán almacenadas en un árbol AVL utilizando el id de la película, el cual será único, en dado caso se repita, no insertar en el árbol.

```
[
  {
    "id_pelicula": 3835762654644,
    "nombre_pelicula": "Janice Russo",
    "descripcion": "Fugiat commodo aute ipsum veniam anim voluptate Lorem dolore. Amet magna consectetur eiusmod minim. Veniam consectetur cillum qui est nisi consequat culpa. Excepteur occaecat culpa sit adipisicing. Ea exercitation consequat ipsum irure id exercitation eu adipisicing deserunt reprehenderit. Proident amet sit anim consequat veniam culpa fugiat fugiat occaecat sit commodo. Exercitation laboris Lorem ipsum eu ullamco.\r\n",
    "puntuacion_star": 4,
    "precio_Q": 201
  },
  {
    "id_pelicula": 9935521329222,
    "nombre_pelicula": "Dee Riggs",
    "descripcion": "Nisi ex consectetur nostrud est Lorem voluptate exercitation proident proident irure aliqua voluptate. Exercitation nulla veniam sunt consectetur fugiat proident irure reprehenderit qui mollit id consequat mollit ipsum. Nulla sit mollit veniam quis et nulla adipisicing aliquip aliquip. Nostrud reprehenderit sint nostrud sit fugiat dolore mollit laborum. Tempor qui irure ex ut nisi eiusmod eu id et ut aliquip. Voluptate enim cillum Lorem culpa id magna enim aliqua aliqua ex labore ea amet. Ad magna deserunt eu dolore officia cillum amet eu amet consectetur.\r\n",
    "puntuacion_star": 3,
    "precio_Q": 208
  },
  {
    "id_pelicula": 4718037385694,
    "nombre_pelicula": "Barber Norton",
    "descripcion": "Pariatur aliquip in in excepteur. Commodo ex labore duis esse sit ex sunt cupidatat non. Sunt excepteur excepteur qui sit aute est reprehenderit enim consequat anim pariatur mollit. Commodo Lorem duis aliquip labore minim reprehenderit cillum deserunt reprehenderit cupidatat.\r\n",
    "puntuacion_star": 0,
    "precio_Q": 243
  }
]
```

- Carga masiva de clientes las cuales se almacenarán en una lista simple.

```
[
  {
    "dpi": 3206292060642,
    "nombre_completo": "Riley Shaw",
    "nombre_usuario": "Phillips",
    "correo": "phillipshaw@knowlysis.com",
    "contrasenia": "anim",
    "telefono": "+502 (943) 527-2850"
  },
  {
    "dpi": 2648015604652,
    "nombre_completo": "Estes Nixon",
    "nombre_usuario": "Alma",
    "correo": "almanixon@knowlysis.com",
    "contrasenia": "pariatur",
    "telefono": "+502 (808) 447-2688"
  },
  {
    "dpi": 1566211594505,
    "nombre_completo": "Kate Meadows",
    "nombre_usuario": "Herring",
    "correo": "herringmeadows@knowlysis.com",
    "contrasenia": "quis",
    "telefono": "+502 (894) 557-2178"
  }
]
```

- Carga masiva de Actores icónicos de distintas películas, estos serán almacenados en un árbol binario, este se ordenará por el dni de cada actor.

```
[
  {
    "dni": 878,
    "nombre_actor": "Poole Gaines",
    "correo": "poolegaines@knowlysis.com",
    "descripcion": "amet"
  },
  {
    "dni": 229,
    "nombre_actor": "Riggs Rosario",
    "correo": "riggsrosario@knowlysis.com",
    "descripcion": "labore"
  },
  {
    "dni": 592,
    "nombre_actor": "Alana Barrett",
    "correo": "alanabarrett@knowlysis.com",
    "descripcion": "commodo"
  },
  {
    "dni": 479,
    "nombre_actor": "Oneil Oneal",
    "correo": "oneiloneal@knowlysis.com",
    "descripcion": "tempor"
  }
]
```

- Carga masiva de Categorías las cuales serán almacenadas en una tabla Hash.

```
[
  {
    "id_categoria": 4335,
    "company": "ECLIPSENT"
  },
  {
    "id_categoria": 6867,
    "company": "SULFAX"
  },
  {
    "id_categoria": 4371,
    "company": "ZAJ"
  },
  {
    "id_categoria": 8643,
    "company": "ECSTASIA"
  }
]
```


Observaciones

- Lenguaje de Programación: JAVASCRIPT
- Sistema Operativo: Elección del estudiante.
- El IDE a utilizar queda a discreción del estudiante.
- Librería para graficar las estructuras queda a discreción del estudiante
- La página web deberá verse en GitHub Page
- Los archivos de entrada serán documentos en formato JSON (.json)
- El estudiante debe tener un repositorio privado en github con el nombre [EDD_junio]Proyecto2_#carnet y agregar a su tutor como colaborador al repositorio del proyecto (username: willop).
- Se entregará en UEDI un .rar con los entregables solicitados.
- Las copias tendrán nota de 0 puntos y serán reportadas al catedrático y a la escuela de sistemas.

Entregables:

- Manual de Usuario
- Manual Técnico
- Link a repositorio con el código fuente.
- Link para acceder a la página Web que proporciona GitHub Page
- [EDD_junio]Proyecto2_#carnet.rar

Restricciones

- Las estructuras deben de ser desarrolladas por los estudiantes sin el uso de ninguna librería o estructura predefinida en el lenguaje a utilizar.
- No se permite la modificación de código durante la calificación, únicamente se calificará sobre el commit que el estudiante elija **siempre y cuando esté dentro del horario de entrega establecido.**
- Última fecha para aceptar invitación de GitHub será el viernes 24 11:59 PM.

Fecha de Entrega:

Sábado 2 de julio, a las 11:59 PM.