Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Estructuras de Datos
Vacaciones Segundo Semestre de 2023
Proyecto Fase 2

Ingeniero:

- Ing. Edgar Ornelis

Auxiliares:

- Cristian Suy

Tutorías - ECYS

Proyecto Único - FASE I

Ciudad de Guatemala, 18 de diciembre de 2023

Objetivos

Objetivo general:

- Aplicar los conocimientos del curso Estructuras de Datos en el desarrollo de las diferentes estructuras de datos y los diferentes algoritmos de manipulación de la información en ellas.

Objetivos específicos:

- Utilizar el lenguaje Go para implementar estructuras de datos no lineales
- Utilizar la herramienta Graphviz para graficar las estructuras de datos.
- Definir e implementar algoritmos de ordenamiento, búsqueda e inserción en las diferentes estructuras a implementar.
- Aprender la utilización de un cliente-servidor.

Resumen de estructuras a utilizar:

FASE II

- **Árbol B:** Manejo de asignación de cursos de tutores.
- **Tabla Hash:** Manejo de usuarios en el sistema, tutores y estudiantes.
- **Grafos:** Manejo de relación entre cursos.
- **Árbol de Merkle:** Control de libros certificados.
- **Decodificación y Codificación:** Manejo de PDF's de tutores.
- **Encriptación:** Contraseñas de los usuarios.

Definición del problema

ProjectUp es una empresa que se dedica al marketing digital, sin embargo vieron el auge tenía el mundo web y cómo eso impulsaba que pequeñas empresa poder promocionar sus pequeños negocios a través de internet además de solo las redes sociales, por lo que la empresa decidió comenzar a realizar páginas web, sin embargo se dieron cuenta que para entregar una página web de calidad, debe tener un equipo completo, por lo que se realizó una investigación para saber que tipo de empleados debe tener el equipo de desarrollo web. Luego de realizar la selección del equipo completo, la empresa ProjectUp se dio cuenta que ellos necesitan un programa que sea capaz de tener control de las tareas para mantener un orden y entregar los proyectos a tiempo sin complicaciones, por lo que solicita que usted como estudiante en sistemas, cree una plataforma en base a sus conocimientos basicos en el ámbito de las metodologías SCRUM, ellos vieron que ya existen plataformas, como Trello para la gestión de proyectos, pero que también ellos como marketing puedan utilizarlo como un Air Table e ir organizando su tareas de marketing, para tener todo conectado y separado segun el area, para ellos se solicita que realice una aplicación de consola dedicado al equipo de desarrollo web para realizar pruebas y ver la eficiencia de la aplicación.

Descripción de la aplicación

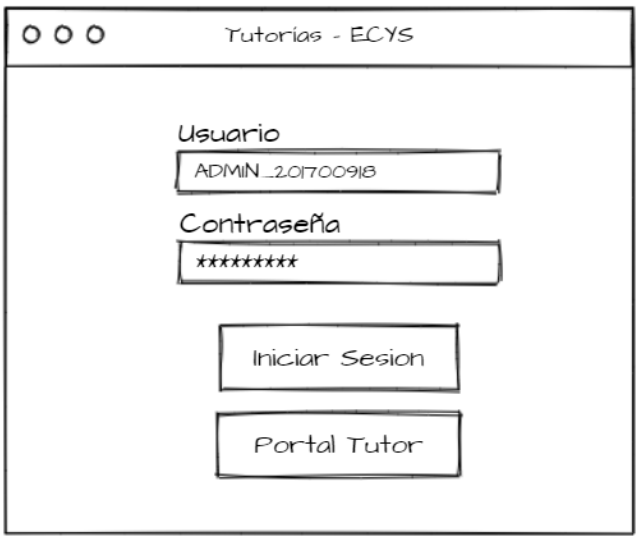
Resumen

La escuela de ciencias y sistemas quedo contento con el prototipo de la aplicación en consola, sin embargo noto algunos inconvenientes con eficiencia, por lo que solicita que usted, como estudiante del curso de Estructura de datos, se le pide que en base a sus conocimientos sobre eficiencia haga cambio de algunas estructuras para que pueda mejorarse el uso de la plataforma aparte de crear una interfaz gráfica interactiva que pueda ser usado para el estudiante y tutores. Por lo que se le da algunas opciones con Go.

- Crear un cliente-servidor con framework: Aquí las estructuras seguirán siendo construidas con Go, con la diferencia que se debe crear un servidor con el mismo, y crear una interfaz gráfica con cualquier framework de su preferencia. (React, Angular, VueJS, etc)
- Crear una interfaz estática con servidor en Go: puede realizar una interfaz gráfica pura en HTML, CSS y mediante Javascript hacer peticiones a Go.

Principal

Login: El sistema contará con un login principal para realizar el inicio de sesión de cada tutor que esté registrado en el sistema o bien los estudiantes registrados, este deberá validar que el usuario y contraseña sean correctos, habrá un usuario especial, el cual será el administrador del sistema este se llamara ADMIN_#Carnet y contraseña Admin, éste será el encargado de realizar la carga masiva de los estudiantes y tutores. En este caso se muestra un ejemplo de la interfaz gráfica, queda a discreción del estudiante si utiliza el mismo login para manejo de estudiantes y tutores, o crear diferentes para cada uno.



Carga Masiva de Tutores:

Aquí se realizará la carga masiva de los tutores que fueron aceptados en la fase anterior, el archivo de entrada contará con 4 parámetros, carnet, nombre, curso y password, como se muestra en la siguiente imagen.

A	B	C	D
carnet	nombre	curso	password
202103105	Luis Daniel Castellanos Betancourt	772	tutor1
202103206	Kewin Maslovy Patzán Tzún	781	tutor1
202103252	Kevin Estuardo Del Cid Quezada	775	tutor1
202106003	Jhonatan Alexander Aguilar Reyes	771	tutor1
202109750	Luis Antonio Castro Padilla	770	tutor1
202110180	Juan Carlos González Valdez	772	tutor1

Estos tutores para un mayor manejo y eficiencia en su búsqueda y no utilizar nuevamente una lista la cual tiene una menor eficiencia, se guardará en un **árbol B de orden 3**, el cual usará el código del curso que está asignado como factor de inserción a dicha estructura.

Carga de Estudiantes dentro del Sistema:

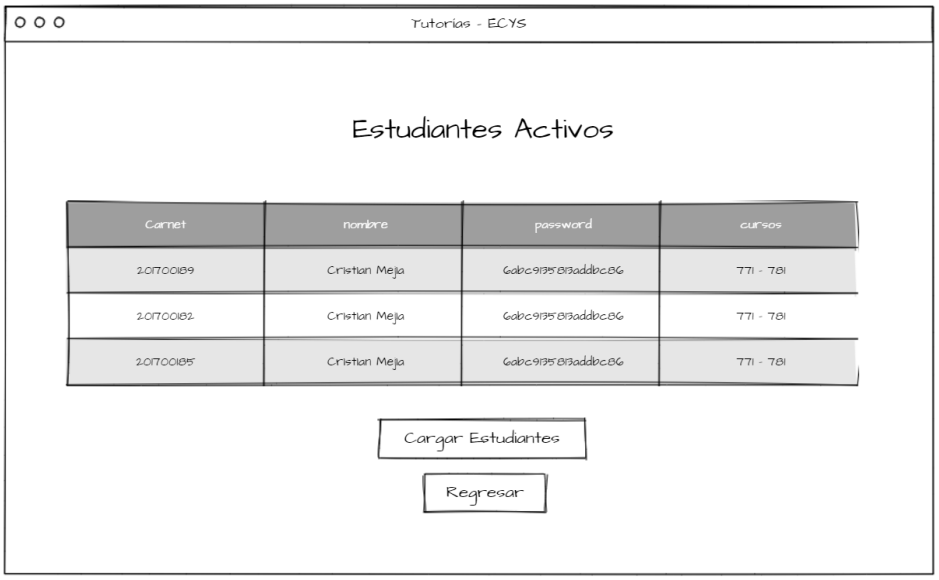
Aqui se realizará la carga masiva de los estudiantes, los datos que se tendrá será carnet, nombre, password, y tendrá 3 cursos por estudiante como máximo, tener en cuenta que para realizar la carga de los estudiantes debe validar que los cursos dentro del sistema ya estén cargados previamente, sino el sistema debe evitar que se cargue los estudiantes para evitar conflictos de cursos inválidos o que no puedan ser llevados, esto se explica más adelante en la sección de Carga de cursos, para los estudiantes se debe utilizar una tabla hash para almacenar, y tomar en cuenta que para el password, al momento de guardarse en la tabla hash, se debe encriptar por el método SHA256 previamente, tu método no reversible.

carnet	nombre	password	Curso 1	Curso 2	Curso 3
202110509	Mario Ernesto Marroquin Perez	2021!@	775	773	781
202110553	Andrea Jazmin Rodriguez Citalan	2021!@	770	980	785
202111563	Jose Pablo Menard Pimentel	2021!@	774	781	285
202111849	Sergio Andrés Larios Fajardo	2021!@	281	286	975
202113580	Andrés Alejandro Agosto Méndez	2021!@	980	280	281

Para la tabla hash de estudiantes comenzará con 7 espacios disponibles, luego de llegar al 70% de utilización, aumentará la capacidad según la serie de Fibonacci iniciando a partir del valor de 5. Cuando llegue al 70% de utilización, el nuevo tamaño de la tabla hash será 8, 13, 21 y así sucesivamente.

Para la función de inserción a la tabla hash será el carnet del estudiante, lo cual necesitará tomar cada carácter del código para convertirlo a su código ASCII y sumarlo de la siguiente manera: si el código es 201700918 daria el siguiente arreglo [50, 48, 49, 55, 48, 48, 57, 49, 56], sumando daría un total de 460, ese número se

pasará como parámetro a una función hash por división y así encontrar el índice en el cual se insertará el nodo. **Para las colisiones dentro de la tabla se utilizará una resolución direccionamiento abierto por salto al cuadrado** el cual tomará el valor hash calculado y lo elevará al cuadrado e intentará insertar el nodo en la nueva posición en caso el nuevo hash sobrepase el arreglo se debe de empezar a recorrer el arreglo desde el inicio con los saltos restantes hasta que se encuentre uno vacío.



Carga de Archivo JSON para cursos:
El Administrador podrá realizar una carga de cursos y sus dependencias, para ello es necesario la implementación de un grafo para poder visualizar las conexiones que hay entre cursos, para ello el archivo tendrá un arreglo de cursos, y cada posición del arreglo tendra el Codigo del curso y un parámetro Post, que será un arreglo con los cursos que pueden llevarse luego de dicho curso. **Para esto se usará el grafo dirigido mediante una lista de adyacencia para su mayor manipulación.**

El archivo JSON tendrá la siguiente estructura:

Aceptar/Rechazar libros de tutores:

El Administrador podrá aceptar o rechazar libros que los tutores carguen, por lo cual debe realizar una interfaz donde pueda ver los libros, una vez que el libro sea aceptado o rechazado, se debe tener un control de seguridad, debido a que los libros deben cumplir con protocolos de calidad, por lo cual sin importar si se rechaza o se acepta, Para guardar la integridad de las acciones y estos no se vean afectados, se utilizara el **árbol de merkle** el cual es una estructura fiable para conservar la integridad de datos, para realizar este árbol se realizara el método SHA-3 para las funciones de hash criptográfico, partirá del siguiente información desde los nodos hojas, sin importar si se **acepta o rechaza,** por lo cual se debe considerar lo siguiente:

Ejemplo:

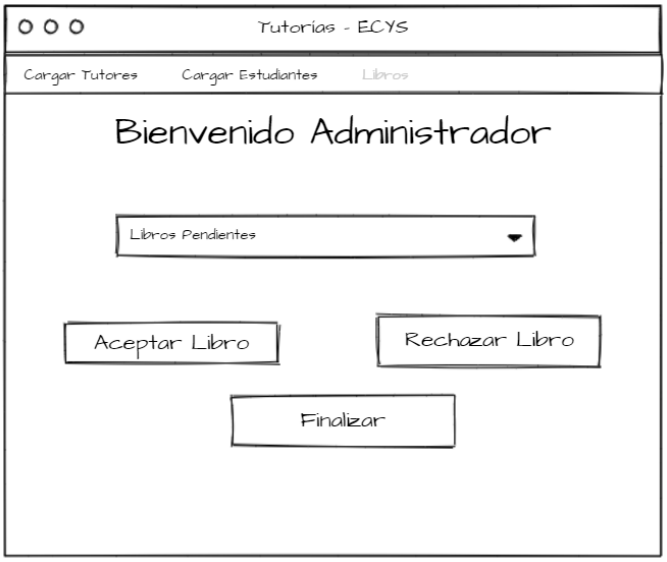
La información que contendrá cada nodo hoja para realizar dicho árbol es la siguiente:

- Fecha y hora de Acción (se tomará la información de la fecha de su computadora, tomando en cuenta el siguiente formato DD-MM-YY::HH:MM:SS)
- Acción realizada (Rechazado o Aceptado)
- Nombre del libro
- Tutor que subio el libro

id = SHA3(“15-10-23::14:02:23” + “Aceptado” + “Libro de introducción a autómatas ” + “201700918”)

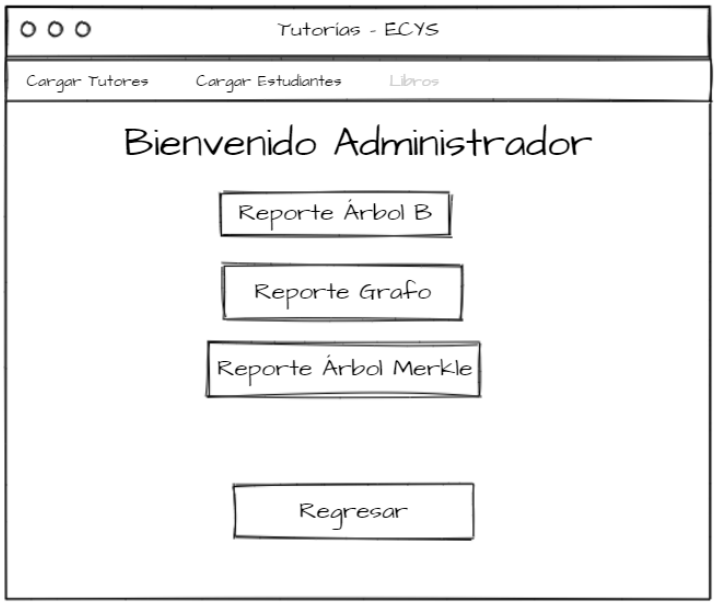
Cuando se tenga el hash de cada bloque de información, se genera un nuevo hash con el mismo método pero utilizando los hash de los anteriores con los nodos internos, ejemplo

id = SHA3(“98366e7eca8….” + “a41602563256284….”)



El botón de finalizar, hará que el sistema cree el árbol de merkle final para su posterior reporte.

Aceptar/Rechazar libros de tutores:
El Administrador podrá hacer reportes principales, para el caso de Tabla Hash ligados los estudiantes, se usará la tabla de esa interfaz como parte de los reportes



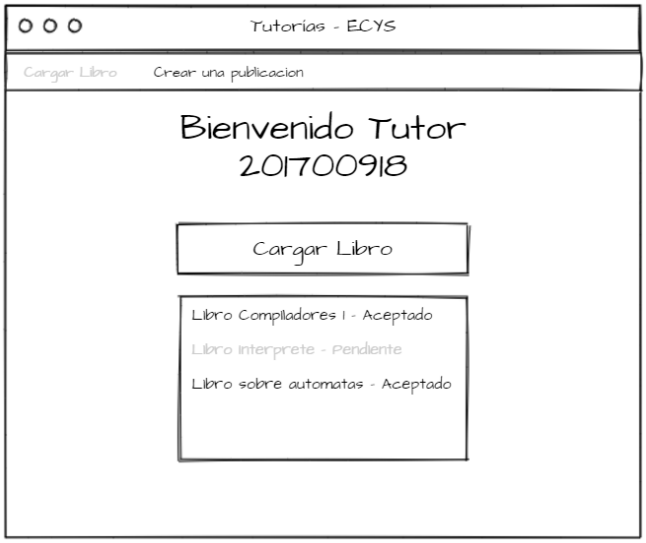
Tutores

Login: El tutor inicia sesión en el mismo login principal, queda a discreción del estudiantes cómo determinar si un usuario es estudiante o tutor, debido a que puede haber estudiantes tutores que también desean sacar tutorías, lo cual se presenta 2 alternativas.

1. Usar el mismo login con un checkbox de Tutor Académico para validar quien entra
2. Crear un login totalmente diferente para los tutores, similar a la implementación de la facultad.

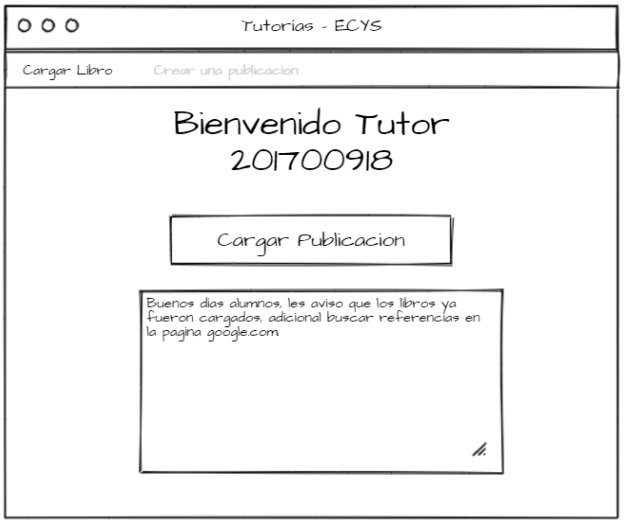
Independientemente de cual opción opte el estudiante, se debe verificar que los datos sean correctos.

Carga de libros y contenido: El tutor académico una vez iniciada la sesión, contará con un apartado donde podrá subir contenido como lecciones, o bien libros certificados y aprobados por el administrador. Para un manejo de PDF más sostenible y que ocupe mayor espacio para una futura implementación a una base de datos, se solicita que usted realice una decodificación y codificación a base64 de los archivos, estos para que los estudiantes puedan visualizarlos desde la interfaz gráfica de ellos. A continuación se muestra una sugerencia de estos apartados.



El tutor puede cargar un libro y se agrega directamente como atributo al árbol B, tomando en cuenta que el libro puede tener 3 estados, Aceptado, Pendiente o Rechazado, para ello el administrador será el único que pueda tomar la decisión.

Para el caso de Contenido, será simplemente una publicación, por lo cual se da la siguiente interfaz, donde el tutor podrá crear una publicación, y este también se irá al árbol B, como un atributo. Será un TextArea, al darle a publicar se irá guardando.

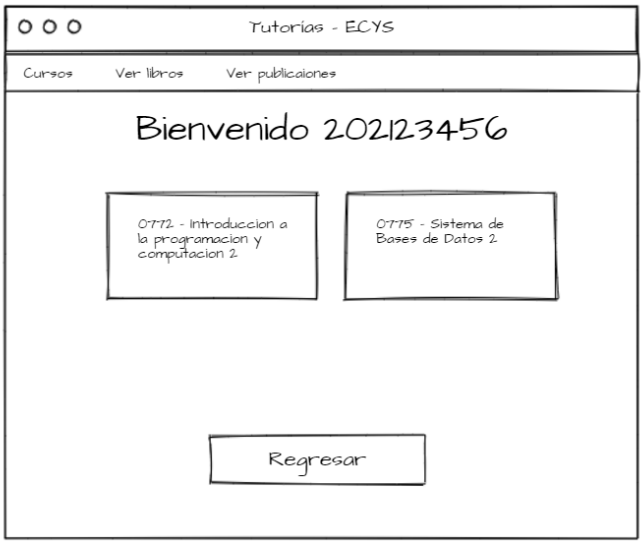


Alumnos

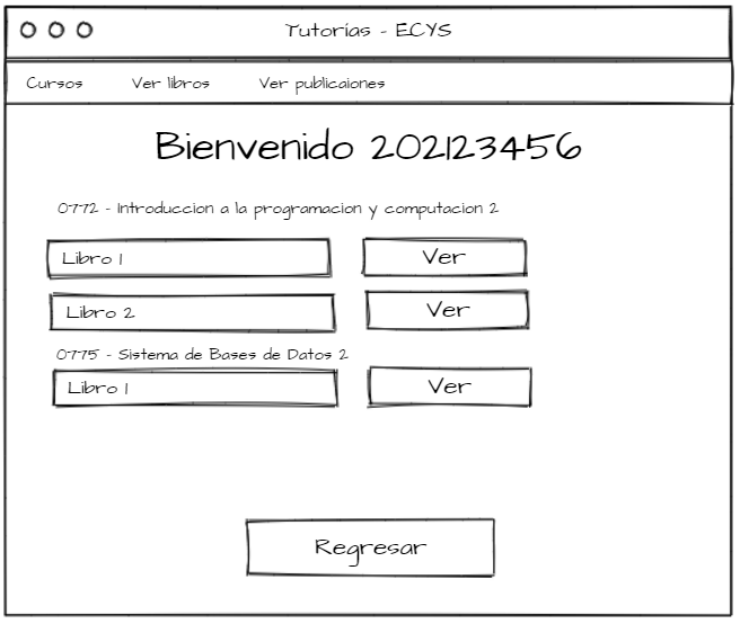
Login: El alumno inicia sesión en el mismo login principal siguiendo los lineamientos descritos en el área de Tutor, diferenciando si es estudiante o tutor.
Se debe verificar que los datos sean correctos.

> **Nota:** El password se debe enviar encriptado antes de llegar al servidor o bien encriptarlo desde el servidor.

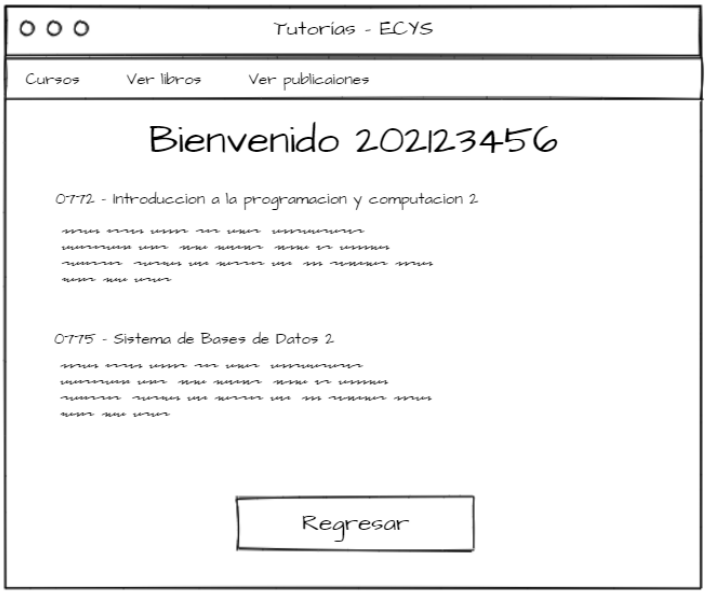
Principal: Aquí el estudiante debe solo ver los cursos que está asignado, de la siguiente manera.



Ver libros: Aquí el estudiante podrá ver los libros que sus tutores han subido y ya fueron aceptados por el administrador. Aquí el estudiante podrá ver cada uno de ellos, al estar codificados en Base64, se recomienda usar un iframe para visualizarlo.



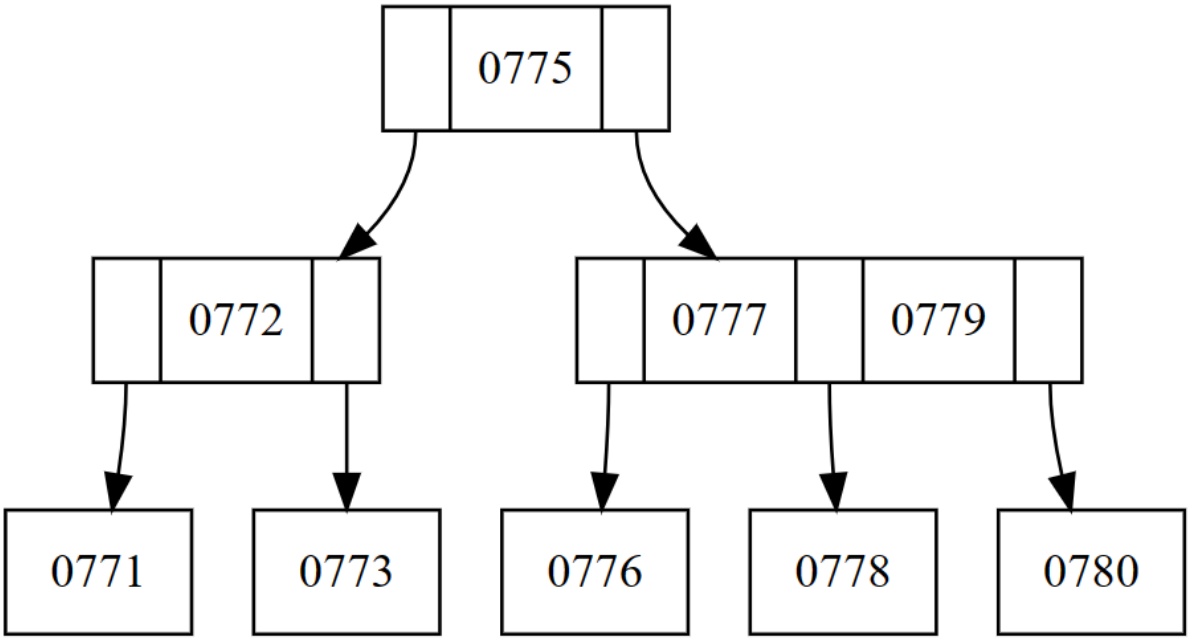
Ver Publicaciones: Aquí el estudiante podrá ver las publicaciones que sus tutores han realizado. Aquí el estudiante podrá ver cada uno de ellos.



Área de Reportes

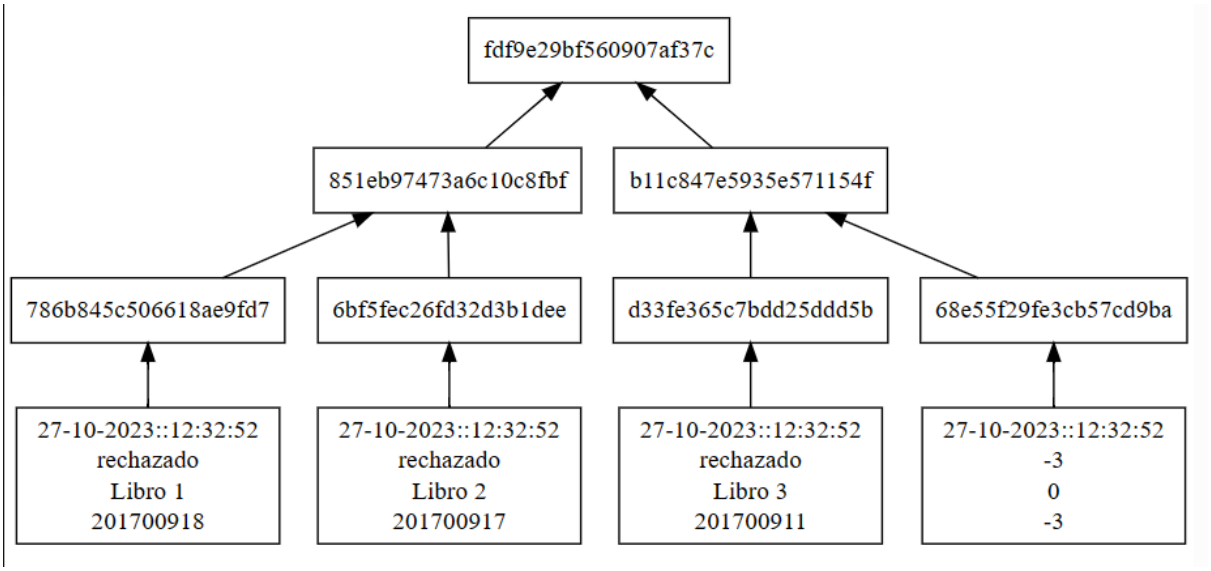
Se contará con un apartado especial en el que se puedan mostrar en forma de grafos las estructuras en tiempo real, generando la imagen utilizando el visualizador de imágenes del sistema operativo. Los reportes sólo serán generados con la herramienta de Graphviz.

Reporte del Árbol B:
Para la realización del reporte se hará de la siguiente manera, mostrando el código del curso para mejor visualización.



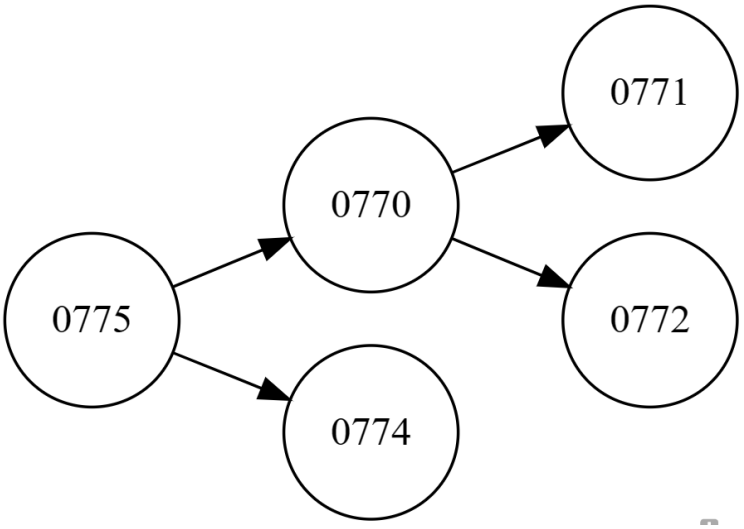
Reporte de Tabla Hash:
Será la tabla de estudiantes de la interfaz, para el usuario administrador. Se deberá de colocar el index que fue generado al momento de la inserción del empleado en una **columna extra de la tabla**.

Reporte de Árbol de Merkle:
Para la realización del reporte se hará de la siguiente manera, mostrando únicamente las cadenas resultantes de las funciones hash a excepción de la última línea que muestra los datos que se utilizó para esas funciones.



Reporte de Grafo:

Para la realización del reporte se usará la lista de adyacencia para partir del grafo, para ello se mostrará el código del curso y su post-requisito de la siguiente manera:



Manuales
(Técnico y Usuario)

Manual de Usuario:
Para el manual de usuario se deberán tomar capturas de cada pantalla que se utilice dentro de la app e indicar cual es la funcionalidad de cada botón o tabla que tenga el acceso al usuario. En otras palabras, una guía para el usuario que utilizará su aplicación, por lo tanto debe de tener los siguientes aspectos:

- Capturas de pantalla de vistas de usuario
- Indicar acción de cada botón de la interfaz
- Enumerar las funcionalidades que posee cada pantalla.

Manual Técnico:
El manual técnico, se deberán de describir aspectos lógicos, librerías y todos los componentes que hacen funcionar su aplicación, por lo que su manual técnico deberá de poseer los siguiente:

- Librerías utilizadas en el desarrollo
- Explicación de las estructuras de dato utilizadas en el proyecto
- Enumerar funciones, métodos y constructores de cada clase con una breve explicación de su funcionamiento.

Tecnologías a utilizar

FASE II:
Realizar la aplicación Servidor en Golang:
Este se realizará por medio de un servidor, todas las estructuras serán realizadas en lenguaje GO y se comunicaran mediante peticiones API-REST con el frontend. Se permite el uso de cualquier package para el manejo de las rutas en Go, la más recomendada para facilidad es fiber se adjunta el link para documentación https://docs.gofiber.io/, en esta pagina se muestra el proceso de instalación. También se permite usar librerías nativas para la parte de codificación, decodificación y encriptación de datos donde corresponda.

Realizar la aplicación Frontend:
Este se realizará en cualquier Framework (React, Angular, VueJS, etc) ya que este solo servirá para la interfaz gráfica de la aplicación, y realizar las peticiones al servidor de Go, se permite también el uso de plantillas o cualquier otro medio para darle estilos a su página o bien HTML/JavaScript y CSS puros igualmente para hacer solo las peticiones y hacer interfaz gráfica.

Realizar los reportes en Graphviz:

Todos los reportes deberán estar realizados en graphviz y deben estar constantemente generando cuando se realice un cambio, para poder observar de forma visual el estado actual de las estructuras.

Restricciones

- Las estructuras deben de ser desarrolladas por los estudiantes **sin el uso de ninguna librería o estructura predefinida en el lenguaje**.
- Se permite el uso de cualquier paquete o repositorio de GO para la parte de blockchain en lo que respecta a codificación, decodificación y encriptación de datos.
- Se permite el uso de paquete para el manejo de la API-REST
- Los reportes son esenciales para verificar si se trabajaron correctamente las estructuras solicitadas, por lo que si no se tiene el reporte de alguna estructura se anularán los puntos que tengan relación tanto al reporte como a la estructura en cuestión.
- Se permite el uso de framework o librerías cómo Angular o React entre otros para el desarrollo del proyecto, solamente para el desarrollo de interfaz gráfica en esta Fase, toda las estructuras deben ser creadas en Go.

Observaciones

- El lenguaje para la fase 2 será Go y todas las salidas y entradas mediante una aplicación web
- Herramienta de desarrollo de reportes solamente **Graphviz** y **HTML** para visualizar imágenes.
- La entrega se realizará por medio de Github, el nombre del repositorio (Fase anterior) debe ser **EDD_VD2S2023_PY_#carnet, dentro de una carpeta llamada Fase 2 y para comodidad y recomendación, separar por carpetas backend y frontend**. Y por medio de **UEDI** se hará entrega del link de su repositorio.
- Recordar tener sus repositorios en privado, para evitar copias o plagios de código.
- Agregar al repositorio como colaborador al Auxiliar, usuario **CristianMejia2198**
- Realizar los **manual de técnico y de usuario** en el README del repositorio o archivos separados.
- Toda duda que se tenga durante el proceso se realizará durante horario de laboratorio**.**
- Fecha de entrega:
    - FASE II: Miércoles 3 de Enero del 2024 a las 20:00 hrs.