Zundada

Laura Alejandra Páez Daza, Marlon David Pabón Muñoz, María Paula Román Arévalo

No. de Equipo Trabajo: 9

I. INTRODUCCIÓN

En esta segunda entrega del proyecto “Zundada”, cómo solución al problema de la organización de eventos universitarios, se implementa una interfaz gráfica funcional. Está interfaz está diseñada por medio de una página web, que se espera que su contenido sea entendible a los usuarios, y por ende sea fácil de aprender e intuitiva de usar. Respecto al prototipo anterior, el diseño se ha modificado para que sea más agradable, y se han agregado nuevas páginas que incluyen eventos, registro, entre otras opciones que tienen los usuarios al estar en la página de “Zundada”. Además, se implementó el desarrollo de una base de datos, para identificar las relaciones y las restricciones que tienen los diferentes tipos de usuarios respecto a la información que contiene el proyecto. De esta manera, se permite definir un límite a los usuarios en la navegación web por la página, y se realizan pruebas de ingreso a la web y a su funcionamiento con los usuarios registrados en la base de datos. Finalmente, las funcionalidades recibieron una mejora a través de la implementación de árboles y otras nuevas estructuras de datos, y se prueba la eficacia a través de pruebas de tiempo. El objetivo de la entrega es ofrecer una página web funcional y aumentar la eficacia del programa del proyecto “Zundada”.

II. DESCRIPCIÓN DEL PROBLEMA A RESOLVER

El ámbito universitario es conocido por ser un espacio lleno de actividades y eventos que van desde conferencias académicas hasta actividades sociales y culturales. Sin embargo, la organización de estos eventos puede resultar desafiante y caótica para las personas que organizan estos. Problemas como la falta de una plataforma centralizada para la planificación y gestión de eventos, dificultades en la comunicación con los asistentes, limitaciones en la difusión de eventos y la falta de herramientas eficientes para el seguimiento y control de las actividades planificadas son recurrentes.

Este proceso de organización de eventos dentro de la universidad puede llevar a situaciones como eventos sobreexplotados o subutilizados, problemas con la confirmación de asistencia, falta de información actualizada sobre eventos futuros, entre otros. Este desafío afecta tanto a los organizadores de eventos como a los asistentes, dificultando la participación y la creación de una comunidad universitaria más cohesionada.

El proyecto "Zundada" tiene como objetivo general resolver el problema de la organización de eventos en la Universidad Nacional a través de una plataforma web. El propósito principal es crear un ecosistema en línea que simplifique y mejore la planificación, promoción y gestión de eventos en el ámbito universitario, comenzando con un enfoque en la organización de fiestas y actividades sociales.

Se basará en el uso eficiente de estructuras de datos y algoritmos para gestionar la información relacionada con los eventos y proporcionar una experiencia óptima para los usuarios. Al lograr estos objetivos, se espera crear un entorno más colaborativo y participativo en la Universidad Nacional, donde los eventos se organicen y disfruten de manera más efectiva, contribuyendo así a fortalecer la vida universitaria y fomentar los ámbitos sociales y culturales.

III. USUARIOS DEL PRODUCTO DE SOFTWARE

Para la descripción de los usuarios se utiliza una descripción mediante los accesos y las vistas que tiene permitido el usuario, en la base de datos y en la página web. También se implementa un diagrama entidad relación para ver su interacción.

Administradores
Es el perfil que tiene acceso a todas las funcionalidades, debido a que administra la página. La única a la que no se le permite acceso es a crear un evento, puesto que es una tarea exclusiva de los organizadores del evento. Sin embargo tiene la posibilidad de eliminar eventos, actualizarlos y acceder a ellos.

Debido a que son los encargados de controlar cada detalle de la plataforma, cuentan con los privilegios de acceso y seguridad a cualquier información relacionada con la página y su estructura.

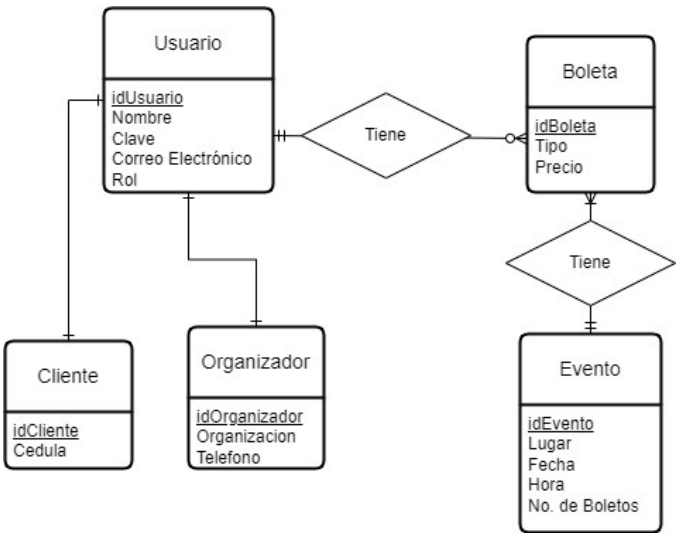
Clientes
El cliente tiene la posibilidad de leer su información cómo cliente, de ver la información de una boleta si la compra, y de acceder a los datos sobre los eventos disponibles. No puede acceder a la información de un organizador.

Organizador
Debido a que organiza los eventos, puede crear, leer, actualizar y eliminar los eventos que esté organizando. Además, tiene las mismas posibilidades para las boletas de sus eventos. Puede leer su información de organizador, y la información de su usuario puede editarla. También accede únicamente a leer la información del cliente si debe contactarse por algún incidente con el evento o con la boleta.

Matriz CRUD
En la siguiente tabla se puede ver los usuarios, cliente, organizador y administrador, y su relación con las diferentes tablas. Es importante mencionar que las letras CRUD significan las funcionalidades, crear, leer, actualizar y eliminar en inglés.

CRUD MATRIZ	Tables Called				
Calling Object	Usuario	Cliente	Organizador	Boleta	Evento
Cliente	CRU	R		R	R
Organizador	CRU	R	R	CRUD	CRUD
Administrador	CRUD	CRUD	CRUD	CRUD	RUD

Diagrama Entidad Relación
Es un diagrama que muestra las llaves principales de la clase, y la relación que tienen (uno a muchos, uno a ninguno o muchos, uno a uno, entre otras)



IV. REQUERIMIENTOS FUNCIONALES DEL SOFTWARE

Para el módulo dedicado a la gestión de eventos se deben poder realizar las siguientes funciones:

- **Crear evento:**

La funcionalidad deberá crear un nuevo evento y almacenarlo en la estructura de datos más adecuada.
Únicamente los administradores podrán acceder a esta funcionalidad después de rellenar un formulario con los datos de la fecha, ubicación y número de boletas disponibles del evento.
Requerimientos funcionales:

- Verificar que toda la información cumpla con el tipo de dato aceptado y todos los campos estén completos, en caso contrario mandará una advertencia.
- Comprobar que el evento no se haya registrado anteriormente.

- **Comprar boleta:**

Para todos aquellos usuarios que quieran asistir a un evento, podrán apartar su boleta por medio de esta funcionalidad, donde por medio de una nueva pestaña confirmara el número de boletas y lo redirecciona hacia el chat directo con el organizador de ese evento.
Requerimientos funcionales:

- Permitir seleccionar el número de boletas que se quieran comprar.
- La funcionalidad redirige al usuario a un chat con el o los organizadores solicitando la compra de las boletas.

- **Consulta de eventos registrados:**

En una ventana nueva el programa deberá permitir consultar los eventos vigentes registrados.
Todos los roles de usuarios pueden acceder a esta función.

Requerimientos funcionales:

- Ordenamiento de eventos por el orden de llegada, el número de boletas disponibles o por la proximidad de la fecha.

- **Eliminar evento:**

Saca de la estructura de datos de almacenamiento un evento en específico.
Los administradores seleccionan el evento a eliminar de una lista tipo consulta.
Requerimientos funcionales:

- Si el evento seguía vigente, se envía un mensaje con la cancelación del evento a las personas que tenían comprada su boleta y la devolución de dinero.

- **Modificación datos asociados a los eventos:**

Permite editar alguno de los datos del evento.
Solo los administradores pueden acceder al formulario de modificación.

Requerimientos funcionales:

- Verificar que los datos cambiados cumplan con el tipo de dato aceptado y todos los campos estén completos, en caso contrario mandará una advertencia.

- **Resumen de usuarios que compraron entradas para el evento.**

Para cada evento deberá permitir consultar los usuarios y la cantidad total de personas que compraron entradas para un evento.
Los administradores acceden a esta función consultando primero el evento y luego las personas registradas para el mismo.
Requerimientos funcionales:

- Permitir consultas completas o parciales de la cantidad de personas que compraron su boleta para el evento.

Para el módulo de usuarios se deben tener las siguientes funcionalidades, se aclara que a pesar de que se tienen 3 roles de usuarios, las funcionalidades permanecen igual, lo único que cambia es el objeto(Cliente, Administrador o Organizador) que se almacena en las estructuras de datos.

- **Crear nuevo usuario:**

La funcionalidad deberá crear un nuevo usuario y almacenarlo en una estructura de datos.
Los visitantes deberán llenar los datos completos de un formulario para registrarse.
Requerimientos funcionales:

- El inicio de la aplicación tendrá un botón de “Sign up”, donde cada visitante nuevo podrá crear su propio usuario.
- Verificar que toda la información cumpla con el tipo de dato aceptado y todos los campos estén completos, en caso contrario mandará una advertencia.
- Comprobar que el usuario no se haya registrado anteriormente para evitar la duplicación de datos.

- **Modificar datos del usuario:**

Modificar la información de los datos del usuario asociado. Cada usuario podrá actualizar sus datos personales por medio de un formulario similar al de registrarse.

Requerimientos funcionales:
-Verificar que los datos cambiados cumplan con el tipo de dato aceptado y todos los campos estén completos, en caso contrario mandará una advertencia.

- **Eliminar usuario:**

Los usuarios podrán eliminar su cuenta en cualquier momento por medio del botón de configuración y eliminar cuenta.

Requerimientos funcionales:

- Arrojar un mensaje de advertencia que pida la confirmación de eliminar la cuenta.

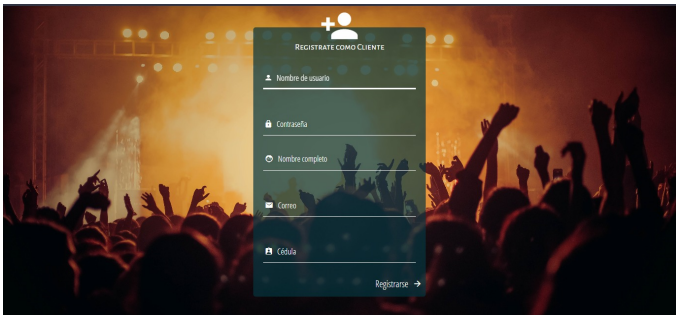
- **Consultar de entradas compradas:**

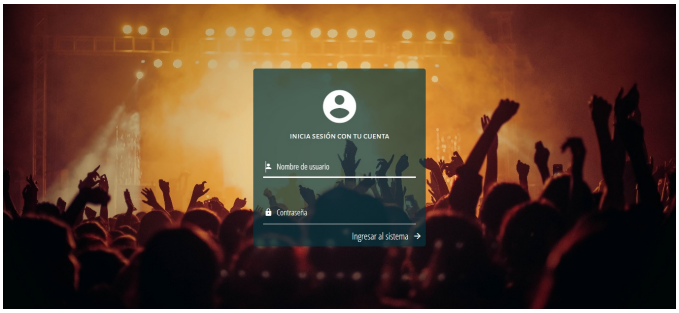
Por medio de un menú, los usuarios consultan toda la información de los eventos a los cuales compraron entradas.
Requerimientos funcionales:

- Ordenamiento por orden de eventos más recientes o por fecha de realización.

V. DESCRIPCIÓN DE LA INTERFAZ DE USUARIO PRELIMINAR

La interfaz de usuario ha presentado una mejora en la parte visual, tiene las partes mencionadas anteriormente pero con mejores detalles y más comprensible para los usuarios algunas son: la sección de eventos destacados, la pantalla al entrar a la página para que cada usuario se identifique por su rol, la pantalla de inicio de sesión, la pantalla de registro de clientes, entre otras.





VI. ENTORNOS DE DESARROLLO Y DE OPERACIÓN

El entorno de desarrollo de "Zundada" estará configurado de la siguiente manera, el software se desarrollará principalmente en TypeScript, aprovechando su fuerte tipado y ventajas en el desarrollo web.2. Visual Studio Code será la principal herramienta de desarrollo, proporcionando un entorno altamente configurable y optimizado para TypeScript., además, GitHub será la plataforma central para el control de versiones y colaboración en equipo, permitiendo un seguimiento preciso de los cambios y una gestión efectiva del proyecto.

El software "Zundada" estará diseñado para funcionar en un entorno de producción en línea, pero no requerirá una base de datos, ya que se basará en el almacenamiento de datos en tiempo real u otras soluciones de almacenamiento de datos sin bases de datos tradicionales. Es importante mencionar que la base de datos creada para está entrega ha sido únicamente para visualizar mejor las relaciones y restricciones en los usuarios, y principalmente para la prueba del ingreso a la página web y la interacción del usuario con ella.

VII. PROTOTIPO DE SOFTWARE INICIAL

Este prototipo se compone se maneja por medio de la consola y presenta dos módulos, la gestión de los usuarios y la gestión básica de eventos.

Link GitHub: https://github.com/Lapd1103/Zundada

VIII. IMPLEMENTACIÓN Y APLICACIÓN DE LAS ESTRUCTURAS DE DATOS

En esta entrega se implementaron las siguientes estructuras:

- **BST**

El Binary Search Tree (BST) implementado en el prototipo de software es una estructura de datos que se utiliza para organizar y gestionar la información de los usuarios. Esta estructura de datos contribuye a la funcionalidad del prototipo de software al permitir la realización de las siguientes operaciones:

- ❖ Creación de Usuarios: El BST permite crear y almacenar nuevos usuarios junto con sus datos, como nombres de usuario, correos electrónicos y claves.
- ❖ Inserción de Datos: A través del método correspondiente, se insertan los datos de un usuario en el BST, manteniendo una estructura organizada.
- ❖ Actualización de Datos: Los datos de un usuario específico pueden ser actualizados en el BST mediante el método adecuado.
- ❖ Eliminación de Datos: El BST facilita la eliminación de la información de un usuario específico, manteniendo la integridad de la estructura.
- ❖ Búsqueda de Datos: Es posible buscar un usuario por su nombre de usuario utilizando las capacidades de búsqueda del BST, lo que permite una recuperación eficiente de la información.
- ❖ Consulta de Todos los Datos: Se pueden consultar todos los usuarios almacenados en el BST, brindando una visión completa de los datos de usuarios registrados en el sistema.
- ❖ Almacenamiento de Datos: El BST almacena de manera eficiente y persistente la información de los usuarios, asegurando su disponibilidad y consistencia dentro del prototipo de software.

- **MinHeap**

El MinHeap implementado en el prototipo de software es una estructura de datos que se utiliza para organizar y gestionar la información de los usuarios de manera eficiente. Esta estructura de datos contribuye a la funcionalidad del prototipo de software al permitir la realización de las siguientes operaciones:

- ❖ Creación de Usuarios: El MinHeap permite crear y almacenar nuevos usuarios junto con sus datos, como nombres de usuario, correos electrónicos y claves.
- ❖ Inserción de Datos: A través del método correspondiente, se insertan los datos de un usuario en el MinHeap, manteniendo una estructura organizada y de prioridad mínima.
- ❖ Actualización de Datos: Los datos de un usuario específico pueden ser actualizados en el MinHeap mediante el método adecuado.
- ❖ Eliminación de Datos: El MinHeap facilita la eliminación de la información de un usuario específico, manteniendo la integridad de la estructura de prioridad mínima.
- ❖ Búsqueda de Datos: Es posible buscar un usuario por su nombre de usuario utilizando las capacidades de búsqueda del MinHeap, permitiendo una recuperación eficiente de la información según la prioridad.
- ❖ Consulta de Todos los Datos: Se pueden consultar todos los usuarios almacenados en el MinHeap, brindando una visión completa de los datos de usuarios registrados en el sistema organizados por prioridad.
- ❖ Almacenamiento de Datos: El MinHeap almacena de manera eficiente y persistente la información de los usuarios, asegurando su disponibilidad y consistencia dentro del prototipo de software.

IX. PRUEBAS DEL PROTOTIPO Y ANÁLISIS COMPARATIVO

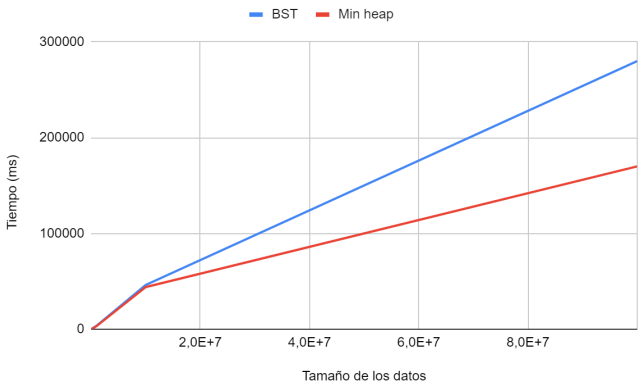
- Añadir evento: Añadir un evento a un BST generalmente tiene una complejidad de tiempo promedio de O(log n), donde 'n' es el número de nodos en el árbol. Esto significa que el tiempo requerido para añadir un evento en un BST aumenta de forma logarítmica a medida que se agregan más eventos. En un BST equilibrado, la operación será eficiente, pero en un BST desequilibrado, podría acercarse a O(n), lo que es menos eficiente. Mientras que añadir un evento a un MinHeap tiene una complejidad de tiempo de O(log n), donde 'n' es el número de elementos en el heap. Añadir un elemento es una operación de "burbujeo" que consiste en comparar y mover el elemento hacia arriba en el heap

hasta que se restablezcan las propiedades del MinHeap. Esto se realiza en tiempo logarítmico.

En general, ambas estructuras de datos tienen una complejidad de tiempo similar para la operación de añadir un elemento, que es O(log n). Sin embargo, hay una diferencia clave:

- El BST está diseñado para admitir búsquedas más eficientes (O(log n)), pero puede volverse ineficiente en caso de desequilibrio.
- El MinHeap está diseñado específicamente para mantener el elemento mínimo en la parte superior, lo que hace que las operaciones de extracción del elemento mínimo sean muy eficientes (O(1)).

FUNCIONALIDAD : Añadir Evento		**FUNCIONALIDAD :Añadir Evento**	
Implementación con: BST		**Implementación con Min Heap**	
Tamaño de Datos	**Tiempo (ms)**	**Tamaño de Datos**	**Tiempo (ms)**
1	0,314	1	0,04
10	0,832	10	0,62
100	0,783	100	0,169
1000	1,8113	1000	0,368
10000	55,81	10000	37,75
100000	250,46	100000	165,78
1000000	3600,34	1000000	3200,16
10000000	46142	10000000	44000
100000000	280000	100000000	170000



X. ROLES Y ACTIVIDADES

INTEGRANTE	**ROL(ES)**	**ACTIVIDADES REALIZADAS**
Laura Alejandra Paez Daza	Líder Coordinadora	-Planeación de las reuniones del equipo. -Distribuir las actividades pendientes. -Gestionar la distribución de documentos de la primera entrega -Encargada principal de la interfaz gráfica
Marlon David Pabon Muñoz	Animador Técnico	-Apoyo emocional al equipo. -Coordinador de pruebas técnicas del código. -Encargado de las pruebas de tiempo y de la implementación de cambios en las funcionalidades
María Paula Román Arévalo	Investigador Observador	- Identificación de los usuarios que utilizan la plataforma - Crear la base de datos en mysql con unos datos de prueba para aplicar en interfaz gráfica -Realización del diagrama entidad relación y la matriz CRUD para definir las restricciones y relaciones de los usuarios

XI. DIFICULTADES Y LECCIONES APRENDIDAS

Mencione las dificultades encontradas durante el desarrollo del proyecto. Además, haga alusión a las principales lecciones aprendidas durante el proceso.

- Identificación de las relaciones y restricciones entre los usuarios y los datos, puesto que al inicio se pensó en cliente y en organizador como una subclase, pero luego se percató que ellos tienen su llave propia.
- En la ejecución de las pruebas de tiempo fue difícil encontrar una manera de realizar las pruebas en el caso de funcionalidades como registrar cliente, donde se debía rellenar celdas de campo. Pero se encontró una manera de crear datos aleatorios, y llevar a cabo las pruebas.