

PROYECTO ZUNDADA

INTEGRANTES DEL GRUPO:

Laura Alejandra Páez Daza

Marlon David Pabón Muñoz

María Paula Román Arévalo

 **Uniendo a las personas
para crear el ambiente**





INTRODUCCIÓN //

En la búsqueda de solucionar la organización de eventos en el ámbito universitario, en esta entrega se implemento:

- **Interfaz Gráfica Funcional**
- **Base de datos (Prueba de la Interfaz)**
- **Implementación de metodología de árbol en las funcionalidades**

USUARIOS

1

Organizador

id Organizador Llave principal

Organización, teléfono atributos

id Usuario Llave foránea

2

Cliente

idCliente Llave principal

Cedula atributo

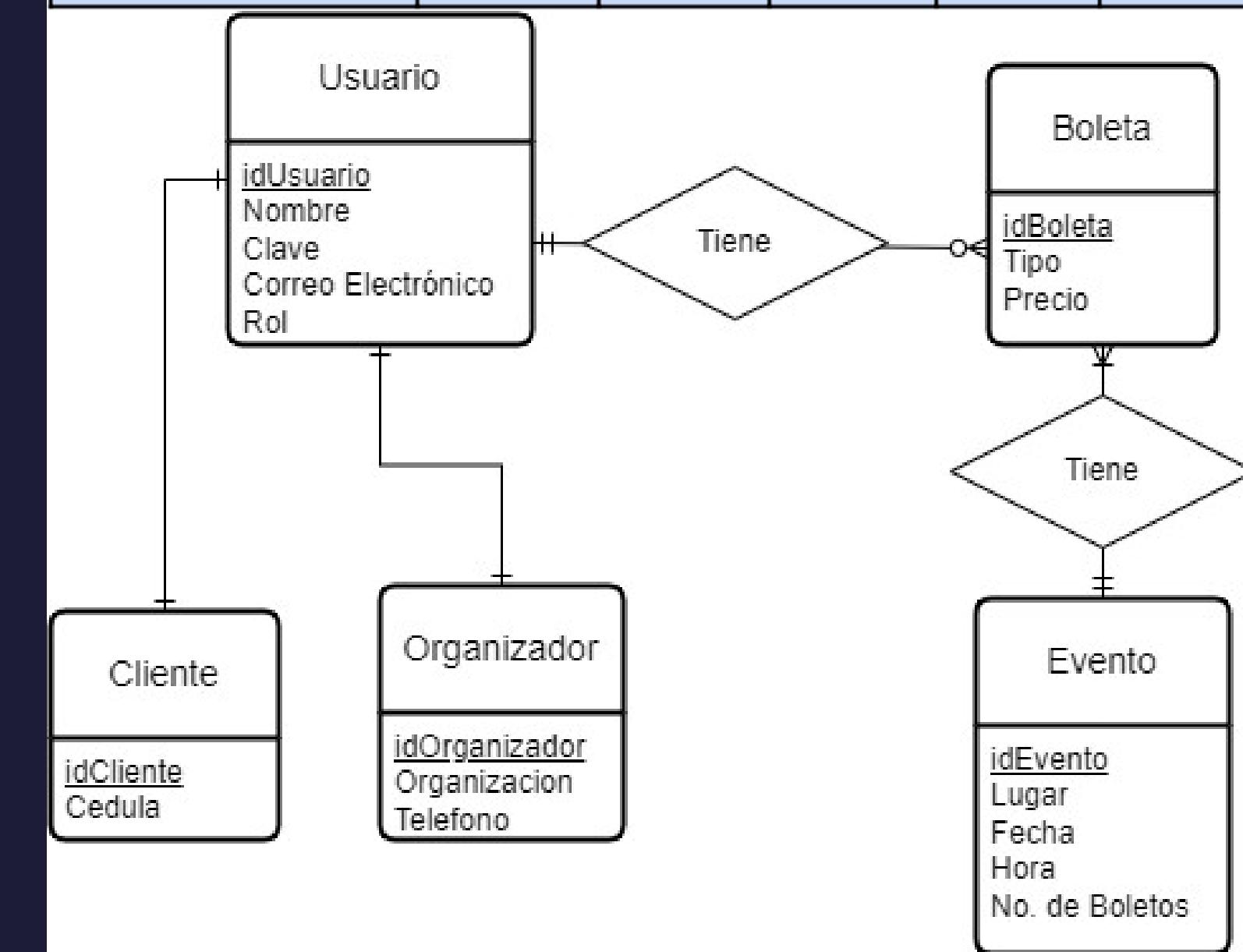
id Usuario Llave foránea

3

Administrador

Usuario con Rol administrador

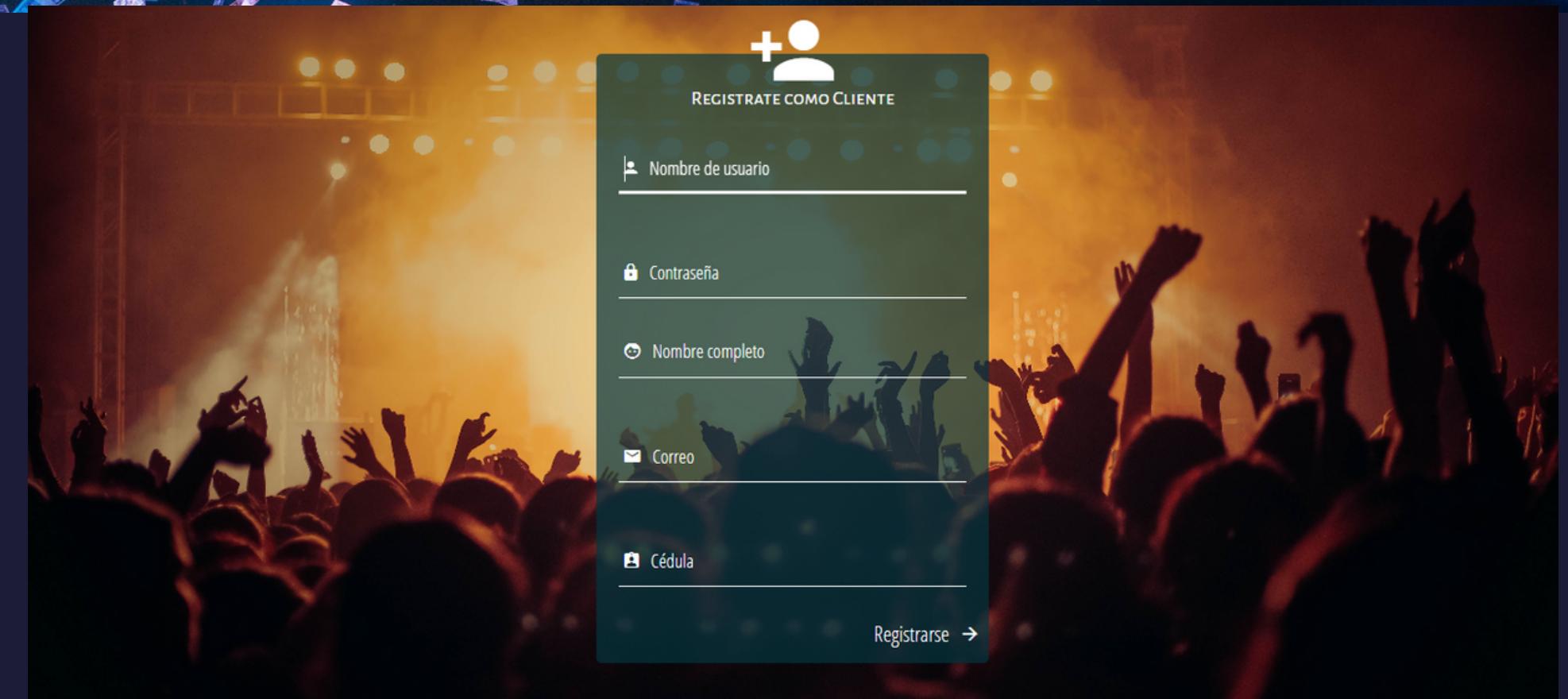
CRUD MATRIZ	Tables Called				
	Usuario	Cliente	Organizador	Boleta	Evento
Calling Object					
Cliente	CRU	R		R	R
Organizador	CRU	R	R	CRUD	CRUD
Administrador	CRUD	CRUD	CRUD	CRUD	RUD



INTERFAZ GRÁFICA

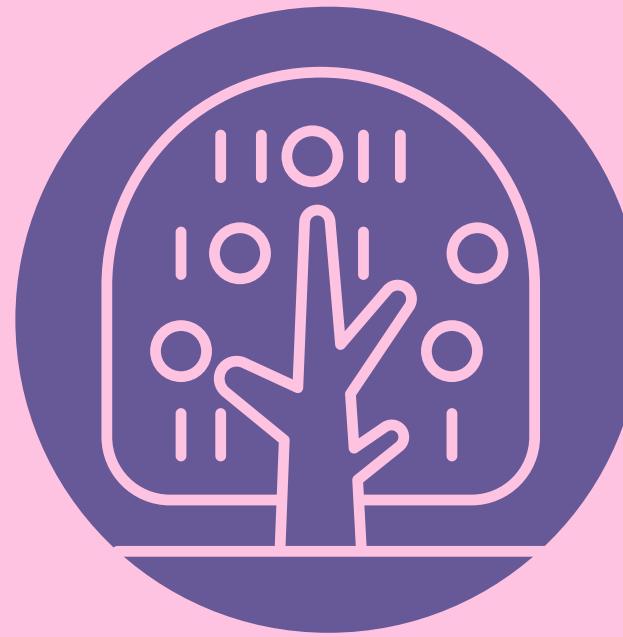
Es una página web funcional que contiene las funcionalidades

The screenshot displays the graphical user interface of the system. On the left, there is a vertical navigation bar with icons and labels: Inicio, Eventos, Clientes, Organizadores, and Administradores. The main area shows a dashboard titled "SISTEMA DE EVENTOS UNIVERSITARIOS INICIO". It features four cards: "EVENTOS" with a red location pin icon, "CLIENTES" with a person icon, "ORGANIZADORES" with a person icon, and "ADMINISTRADORES" with a building icon. At the bottom of the screen, there are two footer links: "ACERCA DE" and "DESARROLLADOR".

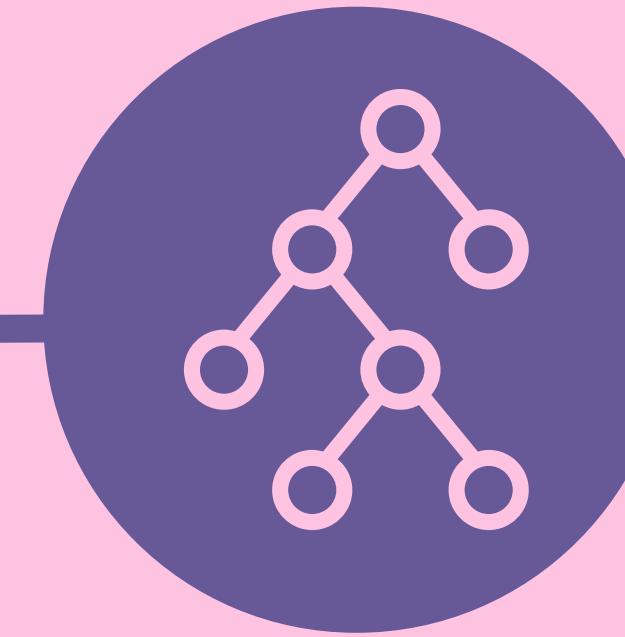


En esta entrega se implementó una mejora visual para que sea comprensible al usuario, una página de inicio según el rol, una página para ver los eventos disponibles, una página registro clientes, y una para ver los eventos disponibles.

ESTRUCTURAS DE DATOS



**Binary Search Tree
(BST) para gestionar y
organizar información
de usuarios**



**MinHeap es una
estructura que
garantiza una mayor
eficiencia**

```
// Implementación de BST para búsqueda por nombre de usuario
class UserBinarySearchTree {
    constructor() {
        this.root = null;
    }

    insert(username, user) {
        this.root = this._insertRec(this.root, username, user);
    }

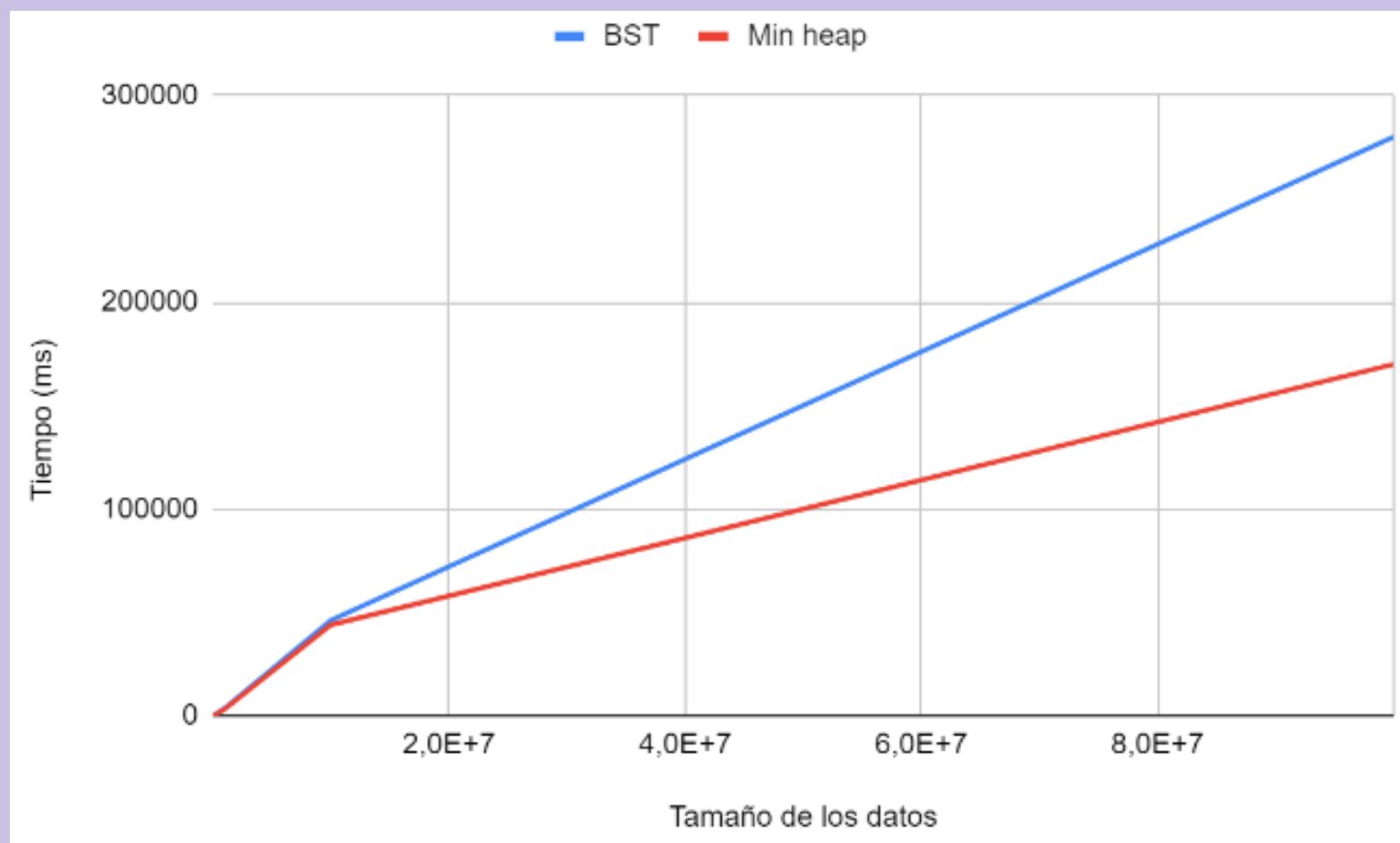
    _insertRec(root, username, user) {
        if (root === null) {
            return new Node(username, user);
        }

        if (username < root.username) {
            root.left = this._insertRec(root.left, username, user);
        } else if (username > root.username) {
            root.right = this._insertRec(root.right, username, user);
        }
    }

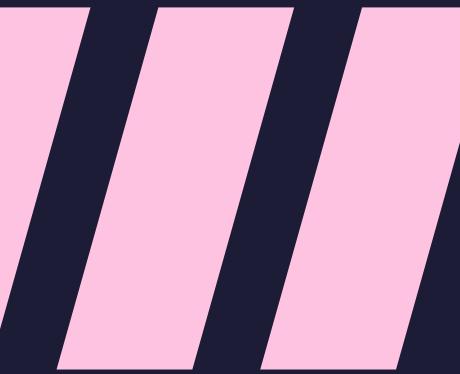
    return root;
}
```

PRUEBA DE TIEMPO

Complejidad de tiempo similar $O(\log n)$



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



MUCHAS GRACIAS
POR LA ATENCIÓN

¿Alguna pregunta respecto al
Zundada o sugerencia?



|||||

"UNIENDO A LAS PERSONAS
PARA CREAR EL AMBIENTE"



ZUNDADA

