

# Sistema de Información Microempresarial

Daniel Melo (*cod. 1010074452*)<sup>1</sup>, Diego Fernández (*cod. 1032450568*)<sup>2</sup>,  
Nikolai Cáceres (*cod. 1032478839*)<sup>3</sup>

{<sup>1</sup> dmelo, <sup>2</sup> diafernandezso, <sup>3</sup> nacaceresp}@unal.edu.co

<sup>1,2,3</sup>:Universidad Nacional de Colombia, Facultad de Ingeniería  
Estructuras de Datos

## I. INTRODUCCIÓN

En el presente documento se describe la segunda etapa de desarrollo de un sistema de información del estado de una microempresa. En primer lugar se describe la población hacia la que se enfoca el proyecto así como el problema identificado en la misma. Posteriormente se describen los diferentes requerimientos funcionales con que debe contar el sistema así como algunas posibles interfaces del mismo.

Se tiene como principal objetivo plasmar todos los conocimientos adquiridos durante el transcurso del semestre. En esta medida se busca, manejar elevados volúmenes de datos que permitan forzar el funcionamiento del proyecto y encontrar posibles vulnerabilidades futuras en el mismo.

## II. DESCRIPCIÓN DEL PROBLEMA A RESOLVER

En Colombia de acuerdo con Confecámaras, a finales de 2017 había cerca de 2,4 millones de microempresas en Colombia, lo que representa cerca del 88 % del tejido empresarial en el país.[1] Comenzar un emprendimiento de este tipo es una labor bastante compleja, de hecho tan sólo el 29,7 % de los emprendimientos nuevos en el país sobreviven y el 70 % de las empresas fracasan en los primeros cinco años de existencia, afirma Confecámaras.[2]

Ahora bien, existen muchos motivos por los cuales una microempresa fracasa, pero sin duda alguna, el primer paso para evitar que esto suceda es tal y como lo afirma IARA generar un análisis objetivo sobre la trayectoria reciente de la empresa y, a partir de ahí, implementar mejoras e innovaciones en el producto, servicio o proceso que se realice.[2]

Estos análisis son implementados por algunas de estas microempresas que cuentan con conocimientos en finanzas, pero muchas otras, tal vez más informales, en ningún momento implementan estos análisis dada la complejidad de los mismos. En este orden de ideas, el presente proyecto consiste en desarrollar un programa simple y de fácil entendimiento, para una persona sin conocimiento alguno de finanzas, que le permita al microempresario saber el estado de su negocio, es decir, si le es rentable o no, a través de

diferentes reportes.

## III. USUARIOS DEL PRODUCTO DE SOFTWARE

El principal público objetivo del proyecto son las microempresas, que por lo general están constituidas por seis o máximo diez personas en su estructura, es decir, los posibles usuarios del software son los administradores de dichas empresa o, en su debido caso, los trabajadores encargados de tareas muy específicas dentro de la actividad económica. Ahora bien, los administradores tienen acceso libre a toda la información y los procesos dentro del programa mientras que los trabajadores tienen un acceso, limitado por los administradores, hacia ciertos aspectos del software según sean necesarias las herramientas que necesite para desarrollar adecuadamente su labor.

## IV. REQUERIMIENTOS FUNCIONALES DEL SOFTWARE

Generalmente un software dedicado a la organización de una empresa debe cubrir tres pilares o aspectos fundamentales: las finanzas, la producción y los recursos humanos. Sin embargo para el cumplimiento de los plazos y según la dificultad del proyecto nos centraremos en las finanzas y la producción dentro del desarrollo de nuestro software.

En cuanto al módulo financiero la propuesta es manejar la contabilidad de la empresa a través de una serie de reportes financieros que el usuario puede generar; ya que el proyecto está enfocado a beneficiar micro empresas, el usuario debe como primera medida ingresar las inversiones iniciales, bien sea a modo de pasivos o activos, además de las materias primas o inventario inicial disponible según el tipo de negocio. Estos datos serán ingresados a un árbol AVL como nodos dotados de: un nombre del producto/inversión o una descripción corta (cadena de caracteres) y un valor numérico (double). El análisis de cada una de las estructuras usadas se muestra en la siguiente sección.

Durante la ejecución del negocio se ingresarán tanto el valor de las ventas (double), como los costos y gastos consecuentes de la actividad económica (double). El programa estará en la capacidad de manejar estos flujos en intervalos de tiempo

fijos según lo requiera el usuario (días, meses, años, etc).

Si el usuario lo desea, puede ingresar datos de proyecciones de ventas (double) para que el programa genere indicadores financieros útiles (como lo pueden ser los TIR, TMAR o puntos de equilibrio) para evaluar la viabilidad del negocio y así contribuir con la toma de decisiones.

El programa con los datos ingresados estará en la capacidad de generar un reporte financiero donde se muestre el estado de la empresa, lo que permite a sus administradores ejecutar acciones rápidas que puedan evitar que la empresa llegue a una bancarrota y que logre avanzar hasta convertirse en una mediana empresa.

Otra gran funcionalidad que pertenece al pilar de producción y que es fundamental para el desarrollo económico de la empresa es como mantener control de los bienes de la empresa en todo momento mediante el manejo de inventarios, el establecimiento de un control de las diferentes fases de producción (para las empresas que apliquen), el control de los diferentes clientes y proveedores involucrados dentro del ciclo económico y el control de las compras y ventas realizadas.

Estas funcionalidades permiten que las empresas que producen algún producto mantengan un total control sobre sus bienes en todo momento y puedan detectar qué proceso no está generando tanto beneficio como el esperado o que productos no están siendo comprados por los clientes.

Para esta segunda entrega tenemos ya implementadas 4 funcionalidades, que se encargan de gestionar el acceso de usuarios al sistema:

- Carga de usuarios:

Descripción: Inicialmente la base de datos de usuarios tendrá únicamente el/los datos de el/los superusuario/s. Esta base de datos esta soportada en base a JSON. Al abrir la aplicación el programa automáticamente lee este archivo JSON a través de las librerías JSON.simple y JSON.simple.parser y los guarda como un array de objetos. Luego a través de la función insert() propia del AVL se crea el árbol con los usuarios.

Acciones iniciadoras y comportamiento esperado: Justo al iniciar la aplicación, el método main() llama a la función chargeusers(). Se espera que esta lea la base de datos en JSON y la organice en un AVL en memoria. El usuario no vera cambios en la interfaz ni en la consola. Inicialmente habrán en la base de datos los superusuarios, estos deberán estar escritos correctamente en formato JSON para que no haya errores durante la ejecución del código.

- Registro y creación de usuario:

Descripción: Nuestra interfaz posee un apartado para el registro de nuevos usuarios. Estos poseen un rango default, ya que dentro de la base de datos ya se encuentra el superusuario o root, el cual se encargara de ejercer control sobre los roles de los usuarios nuevos, y los promoverá según su cargo dentro de la empresa. Al ingresar los datos del nuevo usuario(nickname, email y contraseña) estos se guardan en el JSON ya antes dispuesto a modo de base de datos de la siguiente manera: Como primera medida se compara que la contraseña y la confirmación de la contraseña sean iguales. Al estar cargado el AVL en memoria se hace la búsqueda de el nickname ingresado; si hay existencia de ese mismo nickname, el programa mostrara error; si no; guardara este usuario como objeto en el árbol. Este se ordenará lexicográficamente y se auto balanceará (a través de las diferentes funciones rotate()). Luego el programa mostrara un mensaje de registro satisfactorio.

Acciones iniciadoras y comportamiento esperado:

Al abrir la aplicación aparecerá la interfaz de login. El usuario deberá clicar en "crear nuevo usuario", se abrirá la interfaz de creación donde el usuario deberá registrarse. para este punto la base de datos ya estará cargada en memoria a través del método chargeusers() y disponible para recibir el nuevo registro. Por el momento los campos de contraseña y usuario reciben cualquier tipo de carácter, así que no habrá restricción o error al momento de validar. El programa llamara el método find() para buscar coincidencias y el método insert() para ingresar el nuevo usuario al AVL. A continuación llamará al método saveusers() para guardar nuevamente los datos en memoria en la base de datos.

- Guardado de datos de los usuarios:

Descripción: En el momento que un usuario nuevo se registre, un usuario loguee, o sencillamente no logre entrar a la plataforma, la aplicación procederá a guardar los usuarios organizados en memoria (árbol AVL) a través del método InOrder en una queue, para luego a través de las mismas librerías guardar la información en el JSON.

Acciones iniciadoras y comportamiento esperado:

En cualquiera de los casos mencionados el programa llamara el método saveusers() para guardar nuevamente los datos previamente cargados en memoria en la base de datos.

- Ingreso de usuario a la plataforma:

Descripción: Luego de la carga de los usuarios, en la pantalla de login aparecen los campos para ingresar el nickname y la contraseña, la aplicación procederá a buscar en el árbol a través del método find() el nickname ingresado; si encuentra coincidencia entre el nickname

y la contraseña, la aplicación ingresara a la pantalla principal. En caso contrario la aplicación mostrara mensaje de error.

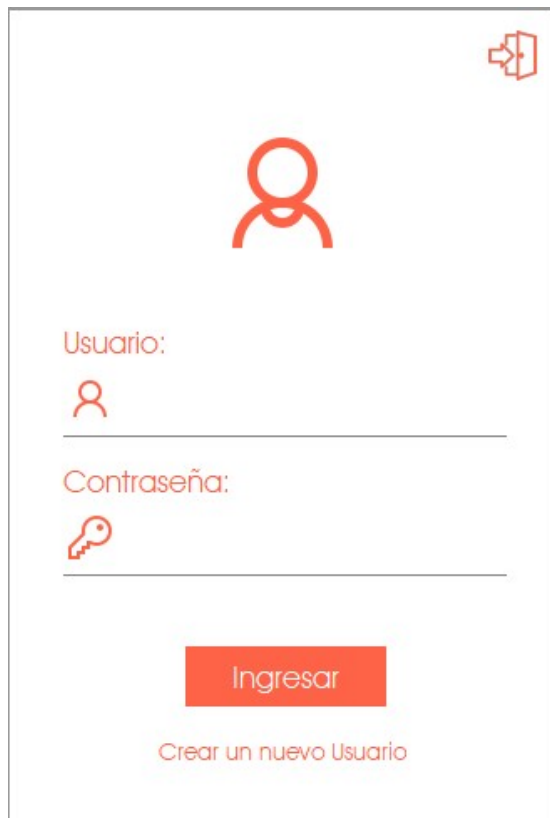
Acciones iniciadoras y comportamiento esperado:

Justo al iniciar la aplicación, el método main() llama a la función chargeusers(). Se espera que esta lea la base de datos en JSON y la organice en un AVL en memoria. El usuario luego de ingresar los datos de login en los respectivos campos y clickear en Ingresar, el programa llamara el metodo find() para buscar coincidencia. Si la hay, la aplicación pasara a la pantalla principal y llamara el metodo saveusers(). Si no, la aplicación mostrara mensaje de error Usuario ingresado no existe o fue escrito incorrectamente. En este caso el usuario debera corregir los datos ingresados para continuar o salir de la aplicación. En ambos casos el programa llamara al metodo saveusers() para guardar la base de datos desplegada en memoria.

#### V. AVANCE EN LA IMPLEMENTACIÓN DE LA INTERFAZ DE USUARIO

En primer lugar, una vez abierto el programa, se presenta una interfaz de login, donde el administrador debe ingresar su usuario y contraseña para poder acceder al sistema.

Figura 1: Interfaz de Login.

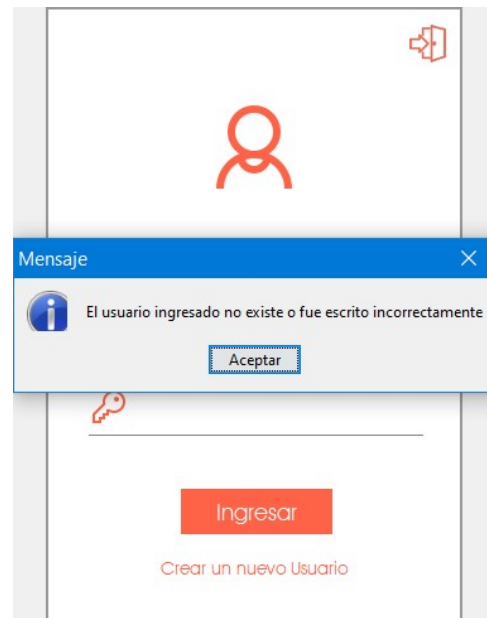


La interfaz de login muestra un icono de usuario en la parte superior. Debajo, hay dos campos de entrada: 'Usuario:' con un icono de persona y 'Contraseña:' con un icono de llave. Al final, hay un botón rojo 'Ingresar' y un enlace 'Crear un nuevo Usuario'.

Si el usuario ingresa un nickname o contraseña errada, se le muestra una ventana emergente indicándole que la

información ingresada no es correcta, tal y como se puede observar en la figura 2.

Figura 2: Usuario no encontrado.



Se muestra una ventana emergente con el título 'Mensaje'. El contenido indica: 'El usuario ingresado no existe o fue escrito incorrectamente'. Hay un botón 'Aceptar'. Debajo de la ventana, se ve la interfaz de login con el botón 'Ingresar' y el enlace 'Crear un nuevo Usuario'.

Una vez el usuario haga click en el botón de aceptar, este es redireccionado a la sección de login, sección donde debe hacer un nuevo intento. En el caso de que el usuario no se encuentre registrado, este debe hacer click en el botón 'crear un nuevo usuario'. Hecho esto, se le redirecciona al panel mostrado en la figura 3, donde debe ingresar su nickname, contraseña e email.

Figura 3: Crear usuario.

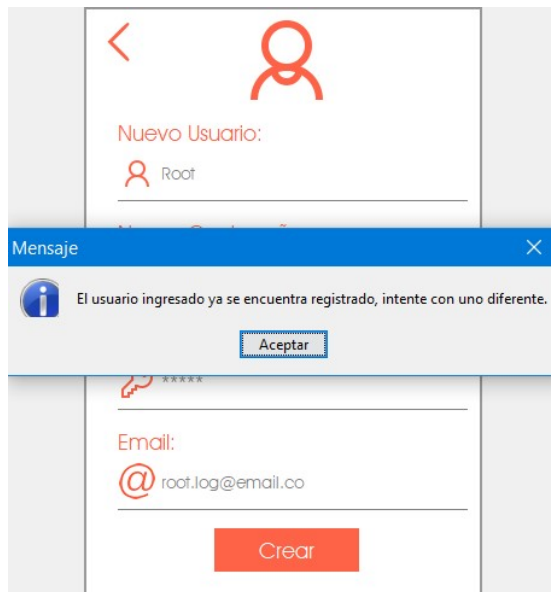


La interfaz de 'Crear usuario' muestra un icono de usuario en la parte superior. Debajo, hay cuatro campos de entrada: 'Nuevo Usuario:' con un icono de persona, 'Nueva Contraseña:' con un icono de llave, 'Confirmar Contraseña:' con un icono de llave y 'Email:' con un icono de '@'. Al final, hay un botón rojo 'Crear'.

Si se ingresa un nickname ya existente, se le muestra al

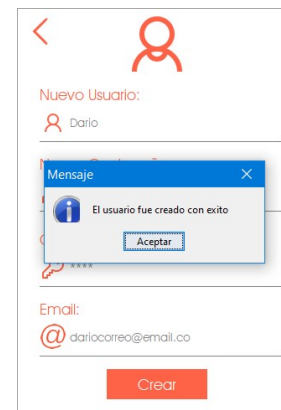
usuario una ventana indicándole este hecho.

Figura 4: Usuario existente.



operación ejecutada.

Figura 6: Usuario creado con éxito.



Posteriormente se le redirecciona, al usuario, a la ventana de login, en la cual debe ingresar con los datos creados. A continuación se muestra la interfaz principal del programa en la figura 6.

Si las contraseñas puestas no coinciden, se informa del mismo modo la irregularidad.

Figura 5: Las contraseñas no coinciden.

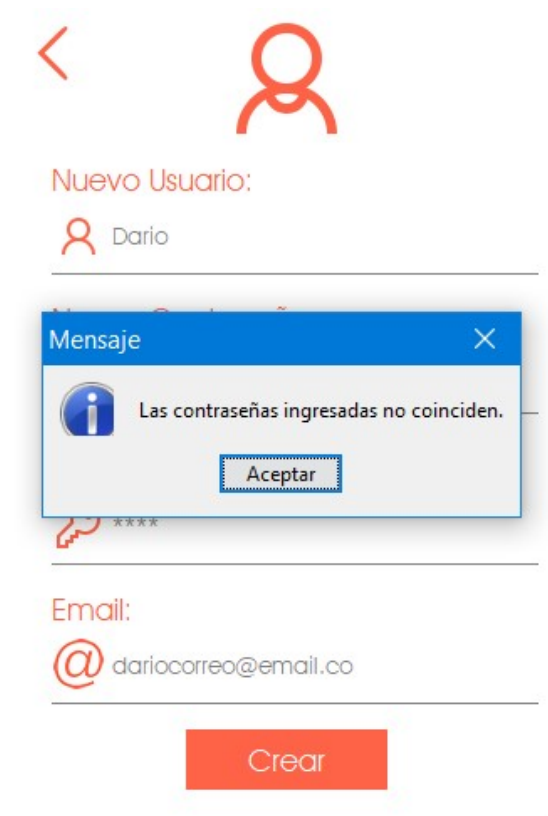
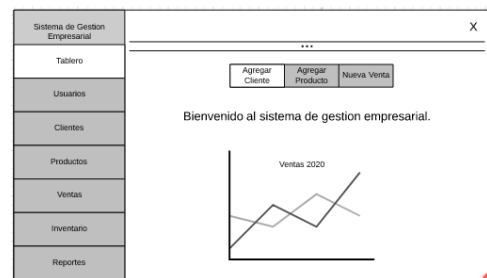


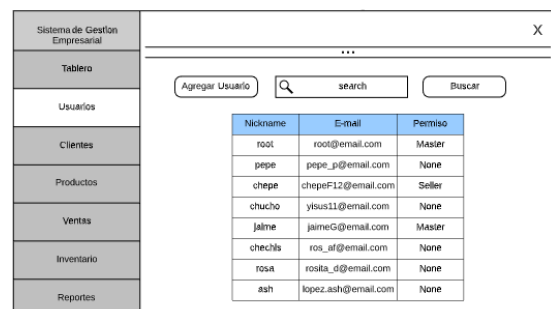
Figura 7: Interfaz principal.



En esta, se presentan los botones principales de agregar Cliente, agregar producto y realizar una nueva venta. También, es posible visualizar una gráfica que representa las ventas tenidas por la empresa mensualmente.

El usuario puede navegar a través de cada una de las pestañas dispuestas. Al dar click en la opción "Usuarios" se muestra la siguiente interfaz, en donde es posible agregar un nuevo usuario y buscar un usuario.

Figura 8: Usuarios.



Una vez ingresados todos los datos de manera correcta, se muestra un mensaje donde se le indica el éxito de la

Al dar click en la opción Clientes”, se despliega una tabla con todos los clientes del sistema, donde se hace posible ver toda la información principal de los mismos, así como el medio por el cual estos realizan sus compras y la fecha en que fueron agregados al sistema.

Figura 9: Clientes.

Sistema de Gestion Empresarial						
Tablero	Nueva venta <input type="text"/> search <input type="button" value="Buscar"/>					
Usuarios	Debe <input type="button" value="▼"/>					
Clientes	1 > 2 > 3					
Productos						
Ventas						
Inventario						
Reportes						
#	Nombre	Email	Origen	Creado	Compras	
1	Pepe Pipo	pepe_p@email.com	Online	10/12/20	48.000	
2	Pepe Pipo	pepe_p@email.com	Store	10/12/20	1'020.000	
3	Pepe Pipo	pepe_p@email.com	Online	10/12/20	48.000	
4	Pepe Pipo	pepe_p@email.com	Online	10/12/20	48.000	
5	Pepe Pipo	pepe_p@email.com	Online	10/12/20	1'020.000	
6	Pepe Pipo	pepe_p@email.com	Online	10/12/20	1'020.000	
7	Pepe Pipo	pepe_p@email.com	Online	10/12/20	48.000	
8	Pepe Pipo	pepe_p@email.com	Online	10/12/20	1'020.000	

Así mismo, la ventana de productos permite visualizar cada uno de los items dispuestos, con sus respectivos precios y cantidades.

Figura 10: Productos.

Sistema de Gestion Empresarial						
Tablero	Acciones de Productos <input type="button" value="▼"/> search <input type="button" value="Buscar"/>					
Usuarios	1 > 2 > 3					
Clientes						
Productos						
Ventas						
Inventario						
Reportes						
<input checked="" type="checkbox"/>	Item	Codigo	Cantidad	Precio		
<input type="checkbox"/>	00000	Store	1	15.000		
<input type="checkbox"/>	00001	Online	1	15.000		
<input type="checkbox"/>	00002	Store	1	15.000		
<input type="checkbox"/>	00003	Online	1	15.000		
<input type="checkbox"/>	00004	Store	1	15.000		
<input type="checkbox"/>	00005	Online	1	15.000		
<input type="checkbox"/>	00006	Store	1	15.000		
<input type="checkbox"/>	00007	Store	1	15.000		

Para las ventas se destaca el canal de venta, el cliente comprador y el medio de pago de la misma.

Figura 11: Ventas.

Sistema de Gestion Empresarial						
Tablero	Nueva venta <input type="text"/> search <input type="button" value="Buscar"/>					
Usuarios	Debe <input type="button" value="▼"/>					
Clientes	1 > 2 > 3					
Productos						
Ventas						
Inventario						
Reportes						
<input checked="" type="checkbox"/>	Numero de Venta	Canal de Venta	Cliente	Pago	Fecha	Total
<input type="checkbox"/>	00000	Store	Pepe Pipo	Efectivo	12/02/20	10.0005
<input type="checkbox"/>	00001	Online	Chepe Fortuna	Tarjeta	12/02/20	10.0005
<input type="checkbox"/>	00002	Store	Pablo Jaramillo	Efectivo	12/02/20	10.0005
<input type="checkbox"/>	00003	Online	Pablo Jaramillo	Efectivo	12/02/20	10.0005
<input type="checkbox"/>	00004	Store	Chepe Fortuna	Tarjeta	12/02/20	10.0005
<input type="checkbox"/>	00005	Online	Chepe Fortuna	Efectivo	12/02/20	10.0005
<input type="checkbox"/>	00006	Store	Chepe Fortuna	Tarjeta	12/02/20	10.0005
<input type="checkbox"/>	00007	Store	Pepe Pipo	Tarjeta	12/02/20	10.0005

En los inventarios se dispone de la ubicación del artículo, como aspecto a destacar.

Figura 12: Inventarios.

Sistema de Gestion Empresarial						
Tablero	Agregar Artículo <input type="text"/> search <input type="button" value="Buscar"/>					
Usuarios	Accion <input type="button" value="▼"/>					
Clientes	1 > 2 > 3					
Productos						
Ventas						
Inventario						
Reportes						
#	Item	Código	Cantidad	Ubicacion	Precio	
1	Lapiz	000	1	A	10.000	
2	SSD	200	3	B	1.000	
3	Mouse	000	2	C	1.000	
4	Keyboard	000	1	C	1.000	
5	Termo	000	2	B	1.000	
6	Resma	000	3	A	1.000	
7	Sillas	000	2	A	1.000	
8	Repisas	000	1	B	1.000	

Ya por ultimo, en la sección de Reportes se presenta un informe que muestra las ganancias obtenidas por cada producto.

Figura 13: Reportes.

Sistema de Gestion Empresarial						
Tablero	<input type="button" value="PDF"/> <input type="button" value="Crear Pdf"/>					
Usuarios						
Clientes						
Productos						
Ventas						
Inventario						
Reportes						
#	Nombre	Cantidad	Ventas	Impuestos	Ventas + Impuestos	Ganancias
1	Lapiz	40	40.00	8.000	48.000	10.000
2	SSD	200	1'000.000	20.000	1'020.000	100.000
3	Mouse	15	40.00	8.000	48.000	100.000
4	Keyboard	3	40.00	8.000	48.000	100.000
5	Termo	1	1'000.000	20.000	1'020.000	100.000
6	Resma	1	1'000.000	20.000	1'020.000	100.000
7	Sillas	1	40.00	8.000	48.000	100.000
8	Repisas	1	1'000.000	20.000	1'020.000	100.000

## VI. ENTORNOS DE DESARROLLO Y DE OPERACIÓN

El entorno en el que se desarrollara el software es Java ya que nos permite exportar nuestro proyecto a diferentes sistemas operativo al ser un lenguaje compilado. El entorno en el que se operará será inicialmente en windows pero a futuro se espera llegar a manejar la aplicación en dispositivos móviles.

## VII. DESCRIPCIÓN DEL PROTOTIPO DE SOFTWARE

Se crea el repositorio correspondiente de software Github con link: [https://github.com/AlejoM1908/Sistema\\_de\\_gestion\\_empresarial.git](https://github.com/AlejoM1908/Sistema_de_gestion_empresarial.git), organizado de la forma estructurada prevista. En este se almacenan las estructuras de datos:

- Arbol AVL
- Pila
- Cola
- Arreglo

Todas estas con sus respectivos soportes de las operaciones funcionales indicadas.

### VIII. DIFICULTADES Y LECCIONES APRENDIDAS

Normalmente un proyecto de esta magnitud requiere de recursos suficientes, por ejemplo de tiempo, ya que requiere diferentes fases desde la planeación hasta la implementación.

Por lo general este tipo de software es desarrollado por grandes y medianas compañías, es decir, la reducida cantidad de personal y recursos económicos juegan un papel importante; sin mencionar la situación actual de pandemia que en cierta manera dificulta el trabajo en grupo.

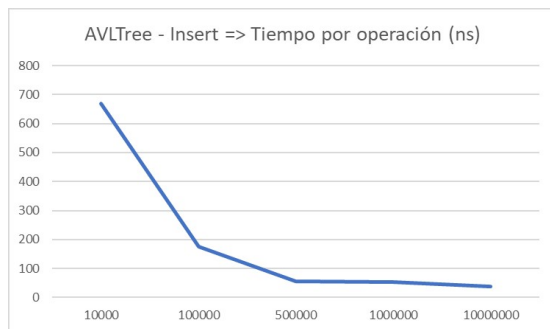
### IX. PRUEBAS DEL PROTOTIPO

En las siguientes tablas se muestran las distintas pruebas a las funcionalidades mas importantes. Se usaron bases de datos con las cantidades recomendadas: 10 mil, 100 mil, 500 mil, 1 millón y 10 millones.

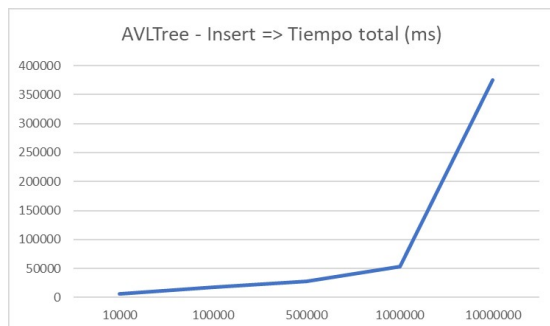
**TABLA I**  
TIEMPOS DE EJECUCIÓN PARA LA FUNCIÓN INSERT()

AVLTree - Insert		
# de Operaciones	Tiempo por operación (ns)	Tiempo total (ms)
10000	670	6706,5
100000	175	17562,5
500000	56	28220,62
1000000	52	53124,58
10000000	37	374725,86

*Figura 14: Tiempo por operación para la función insert en un árbol AVL.*



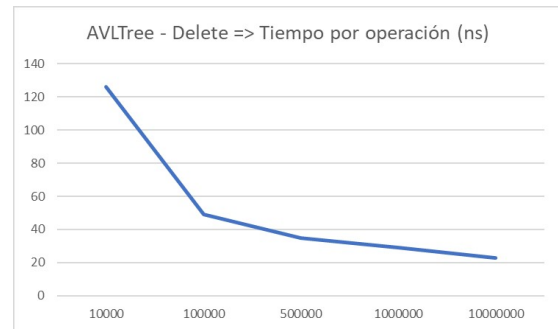
*Figura 15: Tiempo total para la función insert en un árbol AVL.*



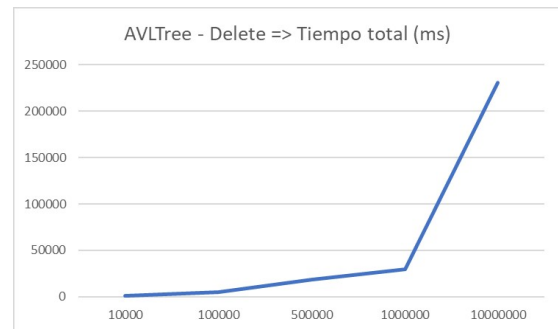
**TABLA II**  
TIEMPOS DE EJECUCIÓN PARA LA FUNCIÓN DELETE()

AVLTree - Delete		
# de Operaciones	Tiempo por operación (ns)	Tiempo total (ms)
10000	126	1262
100000	49	4918,12
500000	35	18163,58
1000000	29	29858,9
10000000	23	231085,62

*Figura 16: Tiempo por operación para la función delete en un árbol AVL.*



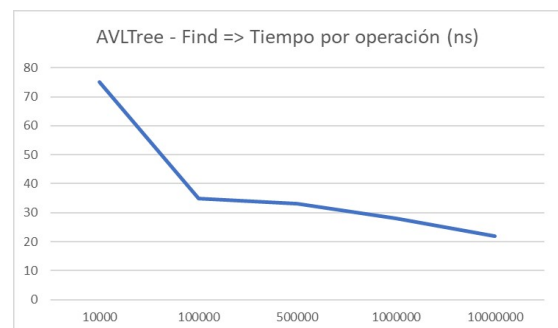
*Figura 17: Tiempo total para la función delete en un árbol AVL.*

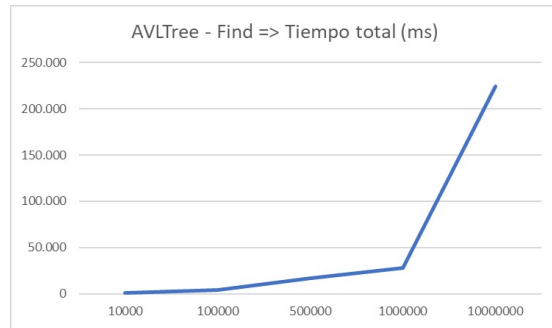
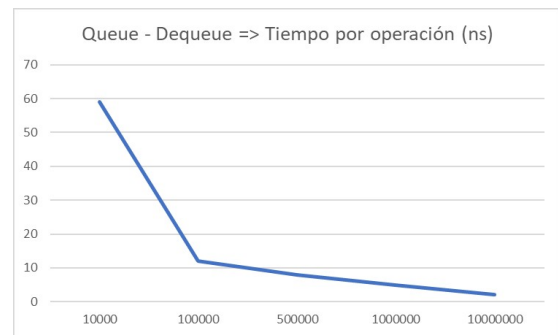


**TABLA III**  
TIEMPOS DE EJECUCIÓN PARA LA FUNCIÓN FIND()

AVLTree - Find		
# de Operaciones	Tiempo por operación (ns)	Tiempo total (ms)
10000	75	757
100000	35	3637,2
500000	33	16760,64
1000000	28	28226,9
10000000	22	224192,08

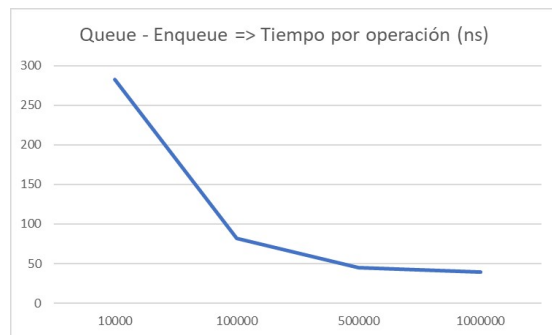
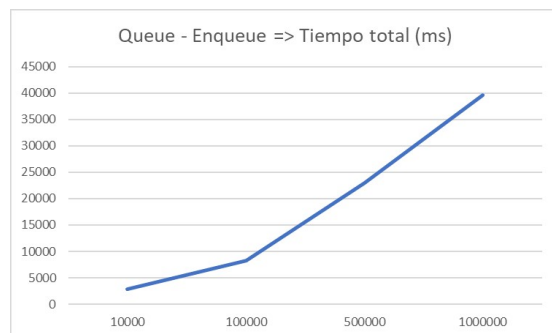
*Figura 18: Tiempo por operación para la función find en un árbol AVL.*



*Figura 19: Tiempo total para la función find en un árbol AVL.**Figura 22: Tiempo por operación para la función dequeue en una pila.*

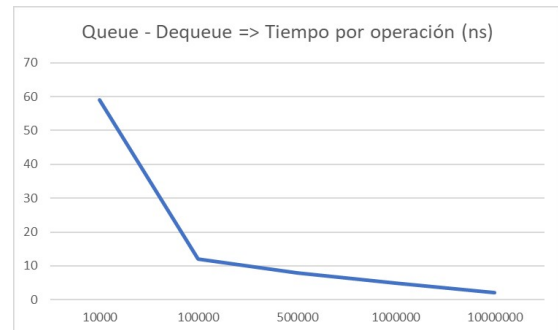
**TABLA IV**  
**TIEMPOS DE EJECUCIÓN PARA LA FUNCIÓN ENQUEUE()**

Queue - Enqueue		
# de Operaciones	Tiempo por operación (ns)	Tiempo total (ms)
10000	283	2833,74
100000	82	8303,72
500000	45	22983,46
1000000	39	39629,48
10000000	-	-

*Figura 20: Tiempo por operación para la función enqueue en una pila.**Figura 21: Tiempo total para la función enqueue en una pila.*

**TABLA V**  
**TIEMPOS DE EJECUCIÓN PARA LA FUNCIÓN DEQUEUE()**

Queue - Dequeue		
# de Operaciones	Tiempo por operación (ns)	Tiempo total (ms)
10000	59	593,18
100000	12	1282,48
500000	8	4582,38
1000000	5	5889,12
10000000	2	30472,02

*Figura 23: Tiempo total para la función dequeue en una pila.*

## REFERENCIAS

- [1] D. Londoño, "Financiamiento de las microempresas en Colombia," May 2018. [En línea]. Disponible en: <https://bancadelas oportunidades.gov.co/es/blogs/blog-de-bdo/financiamiento-de-las-microempresas-en-colombia>
- [2] R. Nacional, "¿Por qué el 70 % de las empresas en Colombia fracasan en los primeros cinco años?" Oct 2018. [En línea]. Disponible en: <https://www.elespectador.com/economia/por-que-el-70-de-las-empresas-en-colombia-fracasan-en-los-primeros-5-anos-articulo-820897>
- [3] Universidad Nacional de Colombia sede Bogotá, Profesor asociado, Cristian Daniel Cordoba Aguirre. (2020) Notas de clase.