UN Campesino:

Sistema de Gestión de datos para UN Campesino

Samuel David Sanchez Cardenas, Valentina Muñoz Rodriguez, Cristian Javier Medina Barrios, Yesid Camilo Ojeda Hernandez.

Equipo de trabajo: D

I. Introducción

La agricultura es un factor intrínseco a la identidad cultural colombiana, en dónde los pequeños productores son una pieza fundamental. Sin embargo, este grupo de individuos, aquellos que incentivan el campo, quienes entran a este mundo de producción, enfrentan desafíos significativos gracias a la falta de herramientas. Como lo son: el acceso limitado a información crucial para una gestión efectiva en sus cultivos, en su ganado; carencias en la administración de información a lo largo de sus procesos productivos; y la falta de acceso a tecnologías especializadas.

En consecuencia a la problemática expuesta, el equipo de UN Campesino se ha propuesto ofrecer una solución integral mediante un sistema de gestión agrícola, basado en la implementación de diversas estructuras de datos. Pues, la importancia y el deseo de abordar este desafío radica en el impacto directo que se puede generar en la calidad de producción y el conocimiento de los pequeños agricultores, así como en el fortalecimiento del sector agrícola colombiano en su conjunto.

Visto de esta forma, el presente documento busca orientar una visión detallada del funcionamiento del programa y describe la etapa inicial del proyecto. Se abordarán aspectos como: la problemática a trabajar, el público objetivo (usuarios), las funcionalidades y el funcionamiento del programa, una visión de las estructuras de datos a implementar, un acercamiento a la interfaz del sistema de gestión de datos, entre otros.

Debe señalarse, entonces, que esta primera etapa proporciona una visión de las estructuras de datos secuenciales implementadas, como lo son las listas encadenadas, pilas y colas. Además, de un reporte reportan las funciones de cada integrante del equipo y sus respectivas responsabilidades y aportes en este primer avance.

II. Descripción del Problema a Resolver

El proyecto UN Campesino nace a partir de las necesidades para la administración y gestión de datos que poseen los pequeños productores del sector del agro en Colombia. Por ello el proyecto se centra en abordar los desafíos de organización que enfrentan los agricultores en la gestión de sus actividades agrícolas, tales como:

- Seguimiento Ineficiente de Cultivos y Ganado: La falta de un sistema organizado para monitorear el estado, rendimiento y necesidades específicas de cultivos y animales.
- Gestión de Horarios y Calendarios Desorganizada: Sin una herramienta adecuada para planificar las actividades agrícolas, los campesinos a menudo enfrentan desafíos en la organización del tiempo necesario para adquirir insumos o llevar a cabo tareas esenciales para el mantenimiento de cultivos y animales.
- Manejo Deficiente de Inventarios: La ausencia de un sistema de inventario claro y accesible complica el seguimiento de los insumos disponibles y los productos generados, impactando la eficiencia en la gestión de recursos.
- Acceso Limitado a Tecnología Especializada: La brecha tecnológica es un obstáculo considerable. La disponibilidad de aplicaciones móviles u otras herramientas digitales que se ajusten específicamente a las necesidades de los pequeños productores es escasa, y su adaptabilidad a contextos locales es frecuentemente limitada. Además de los escasos recursos tecnológicos que llega a tener el campo.

UN Campesino se propone superar estos obstáculos mediante el desarrollo de un sistema integral de gestión agrícola que emplea estructuras de datos para optimizar la organización y seguimiento de las actividades de los campesinos. La meta es ofrecer una solución que no solo facilite la gestión diaria de las granjas sino que también sea intuitiva y eficaz, mejorando la calidad de producción y ampliando el conocimiento y las capacidades de los pequeños agricultores. Así, se pretende dotar a los campesinos de las herramientas necesarias para mejorar la toma de decisiones, aumentar la productividad y fomentar el desarrollo sostenible de sus actividades agrícolas.

III. Usuarios del Producto de Software

Para garantizar la efectividad y accesibilidad del sistema UN Campesino, es de gran importancia identificar y entender las características y necesidades de los usuarios que interactúan con el producto final. A continuación, se detallan los principales usuario que acceden al producto y las funcionalidades accesibles para cada uno:

- UN Campesino (Propietario o Encargado de la granja): el producto final tiene como objetivo ayudar y facilitar la gestión del seguimiento de los productos y tareas que hay en una granja, por lo cual este usuario será el centro del sistema UN Campesino. Los privilegios de este usuario es el acceso al manejo

de toda la información y funcionalidades de la aplicación.
- Trabajadores de la granja: estos usuarios son los empleados que trabajan en la granja. Cada trabajador puede tener distintas tareas y responsabilidades por lo que no requieren de un acceso directo a todas las funcionalidades o información de la aplicación, sino que requiere acceso a información específica para llevar a cabo sus responsabilidades. Según el rol que el granjero le indique al empleado, éste tendrá acceso a ciertas funcionalidades.
- Consultores (veterinarios, agricultores, etc.): Estos usuarios son personas externas a la granja que en algún momento requerirán de tener la información de seguimiento específica de algún insumo, bien y/o producto, ya sea para la gestión de la granja o la salud de los animales.

IV. Requerimientos Funcionales del Software

La presente sección es el núcleo del proyecto y detalla las funcionalidades específicas que UN Campesino proporcionará a sus usuarios. Al definir claramente cómo el software debe actuar en diferentes escenarios y qué acciones inician ciertos comportamientos, se establece una ruta para el desarrollo y la implementación del sistema. Aquí, nos enfocamos en las capacidades de autenticación, registro y gestión de información, asegurando que UN Campesino cumpla con las necesidades de sus usuarios.

- **Ingreso a la plataforma por medio de un usuario y contraseña**

  Descripción: Todos los usuarios deben autenticarse mediante un sistema seguro de usuario y contraseña para, así, acceder a la plataforma.

  Flujo del programa y salidas esperadas:
  1. Inicio de sesión: Se presentará una interfaz gráfica donde el usuario podrá ingresar sus datos y dar click a un botón “login”.
  2. Validación: el sistema verificará la información del ingresada con la que tenga almacenada.
  3. Acceso: Si los datos coinciden con un usuario creado se dará acceso a la plataforma. Por el contrario, si no se encuentra la información del usuario ingresado en la base de datos, se dará un aviso y no obtendrá acceso a la plataforma.

  Requerimientos funcionales:
  - Autenticación e Ingreso a la plataforma si y sólo si existe un usuario y contraseñas ya creados. Para lo cual el sistema tendrá que hacer una búsqueda parcial de la información.
  - Redireccionar hacia la página de inicio de la plataforma.

- **Registro como persona externa en la plataforma por medio de un usuario y contraseña**

  Descripción: El usuario que no es campesino o trabajador, es decir, los usuarios externos podrá crear una cuenta con la cual acceder a la plataforma mediante el login. Así, se permite a consultores externos crear una cuenta para acceder a información específica.

  Flujo del programa y salidas esperadas:
  1. Selección de Rol: al oprimir el botón de “ingresar como externo” se le preguntará al usuario qué tipo de consultor es:
     - Veterinario
     - Agricultor
     - Comprador
  2. Registro: se completará un formulario de registro con el nombre de usuario y su contraseña de ingreso.
  3. Verificación: adicionalmente, se le pide una contraseña de acceso que sólo tiene el campesino para poder completar y validar el registro.
  4. Confirmación: si la información es la correcta, se hace la creación de la cuenta, almacenando los datos, y se redirecciona a la página principal de la aplicación.

  Requerimientos funcionales:
  - Registro a la plataforma si y sólo si, no existe el usuario y se tiene la contraseña de acceso por parte del propietario. Para lo cual el sistema tendrá que realizar una búsqueda parcial de la información para verificar que no existe ese usuario y una búsqueda adicional para validar la contraseña de acceso.
  - Redireccionar hacia la página de ingreso de la plataforma.

- **Registro como trabajador en la plataforma por medio de un usuario y contraseña**

  Descripción: El usuario que sea trabajador será registrado por el campesino.

  Flujo del programa y salidas esperadas:
  1. Registro: El campesino dará click a un apartado “Registro trabajadores”.
  2. Gestión de Información: Se despliega una interfaz donde podrá ingresar los datos del trabajador así como el nombre de usuario y contraseña para el ingreso.

  Requerimientos funcionales:
  - Registro a la plataforma si y sólo si, no existe el usuario. El sistema tendrá que realizar una búsqueda en las listas de trabajadores para confirmar la información.

- Confirmación de creación de usuario trabajador.

- **Gestión de información: Campesino**

Descripción: Medio por el cual el campesino tiene acceso a la información de los apartados definidos:
- Animales
- Cultivos
- Trabajadores
- Productos

Para su gestión basada en el CRUD a la cual el campesino, al tener un rol de administrador, tendrá acceso a total.

Flujo del programa y salidas esperadas:
1. Selección de Apartado: Se dispone una interfaz donde el campesino podrá elegir hacer click en el botón que lo redirija al apartado que requiera gestionar.
2. Operaciones CRUD: Al hacer click y ser redireccionado se mostrará una interfaz que contiene los elementos existentes del apartado seleccionado, también botones "agregar", "buscar", "actualizar" y "borrar" para realizar las respectivas modificaciones.
3. Visualización de Datos: Cada botón generará una pantalla donde se ingresarán los datos necesarios para dicha modificación.

Requerimientos funcionales:
- El usuario ingresado debe ser el del campesino, de lo contrario, no tendrían acceso a la información especificada.
- El sistema ocupará de una interfaz intuitiva para la gestión y registro de información.
- El sistema buscará, agregara, actualizará o borrará la información que el campesino requiera y disponga en la interfaz.

- **Gestión de información: Trabajador**

Descripción: Medio por el cual el trabajador tiene acceso a la información de los apartados definidos:
- Animales
- Cultivos
- Productos

Para su gestión basada en el CRUD. El trabajador podrá consultar la información de todos los apartados y únicamente podrá modificar la información de las listas de tareas. Si el trabajador tiene un permiso concedido por el campesino podrá modificar, también, la información de los apartados.

Flujo del programa y salidas esperadas:
1. Consulta de Información: Se dispone una interfaz donde el campesino podrá elegir hacer click en el botón que lo redirija al apartado que requiera gestionar.
2. Modificación de Tareas: Al hacer click y ser redireccionado se mostrará una interfaz que contiene los elementos existentes del apartado seleccionado, en caso de tener el permiso del campesino dispondrá de botones "agregar", "actualizar" y "borrar" para realizar las respectivas modificaciones. Si no solo podrá seleccionar la lista de tareas y modificarla.
3. Ingreso de Información: Cada botón generará una pantalla donde se ingresarán los datos necesarios para dicha modificación.

Requerimientos funcionales:

- Haber tenido acceso a la plataforma por medio del usuario de trabajador.
- El sistema deberá mostrar las interfaces adecuadas según los permisos de visualización del usuario de clase trabajador.

- **Gestión de información: Consultor**

Descripción: Medio por el cual dependiendo del tipo de consultor tendrá acceso a la información de los apartados definidos:
- Animales
- Cultivos
- Productos

El consultor podrá acceder a visualizar la información del apartado definido por su tipo. Adicionalmente tendrá permiso para actualizar la información del apartado definido por su tipo.

Flujo del programa y salidas esperadas:
Existen tres tipos de consultores:

- Veterinario: Si el externo es veterinario tendrá en la página de inicio la información a la que se redirige a un usuario cuando se selecciona el apartado de "Animales"

- Agricultor: Si es agricultor, tendrá en la página de inicio la información a la que se redirige un usuario cuando selecciona el apartado de "cultivos".

- Comprador: Si es un comprador, tendrá en su página de inicio la información a la que se redirige a un usuario cuando se selecciona el apartado de "Productos". Allí encontrará información de los productos disponibles que puede proveer la granja en tiempo real. Cada producto tiene la opción de "seleccionar y apartar".

Por o cual cada uno de los consultores pasarán por el siguiente proceso:

1. Visualización de Datos: Acceso inmediato a la información relevante según el rol del consultor.
2. Actualización de Información: capacidad de modificar datos específicos.

Requerimientos funcionales:

- Haber tenido acceso a la plataforma por medio del usuario de consultor.
- El sistema tendrá que mostrar en pantalla las interfaces correspondientes al rol del consultor.
- El sistema tendrá que actualizar la información que el consultor requiera y/o disponga.

V. Descripción de la Interfaz de Usuario Preliminar

En esta creación de boceto y primer avance de la interfaz de Un Campesino se enfocó en darle a los usuarios una experiencia visual que permita un uso de la aplicación fácil e intuitivo.

Se selecciona una paleta de colores:

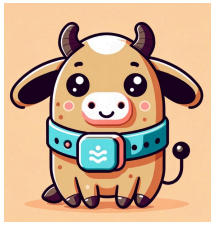
- #FDD3A9
- #CA7050
- #7EA19A
- #FCF5E7

Fuentes:

- Alice
- Alef
- Glacial indifference

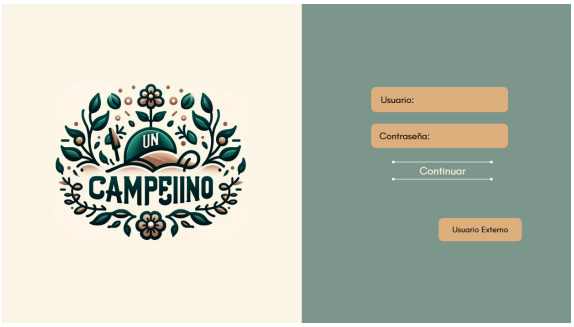
Logos e imágenes

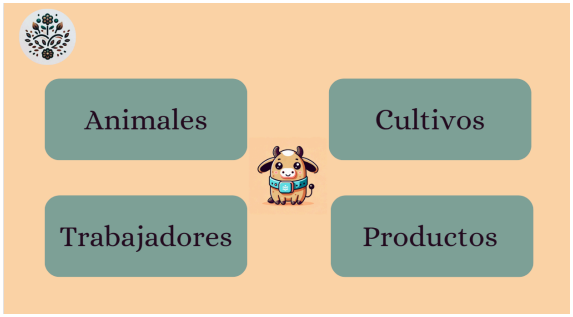
Mascotas:

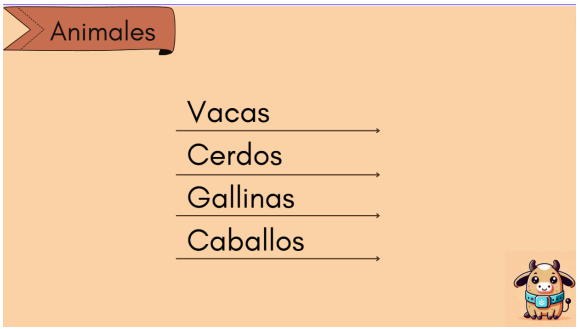


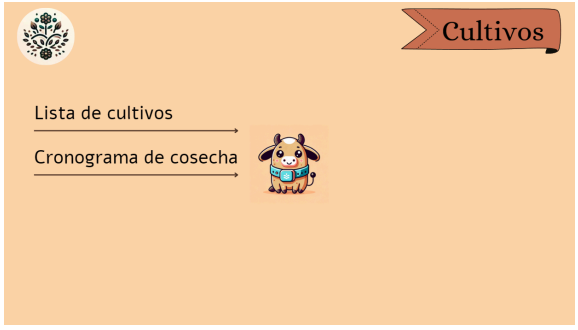
Se definieron como mascotas de la aplicación a este torito pequeño y una yuca, los cuales representan el acompañamiento que se quiere lograr al usar la aplicación.

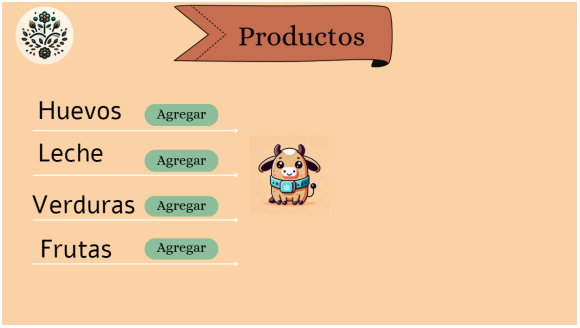
A continuación se muestra los primeros bocetos de la plataforma en su inicio, combinación de los colores cómodos y agradables











Como nos podemos dar cuenta, se opta por dar un estilo informal al diseño de la aplicación, se busca que esta sea agradable y cómoda de usar, con colores suaves pero llamativos.

VI. Entornos de Desarrollo y de Operación

El desarrollo de UN Campesino se llevará a cabo utilizando herramientas y plataformas que faciliten la colaboración y el control de versiones, como Git para repositorio local y GitHub para repositorio remoto. La aplicación será desarrollada en Java utilizando el entorno de NetBeans y estará disponible para el sistema operativo Windows, garantizando así su accesibilidad para todos los usuarios en el campo agrícola.

A. Prototipo de Software Inicial

En el siguiente link se encuentra el repositorio en GitHub del proyecto: https://github.com/CristianMed25/UN_campesino

VII. Diseño, Implementación y Aplicación de las Estructuras de Datos

En esta entrega, se ha desarrollado en Java el manejo de datos mediante el uso de arreglos dinámicos, listas doblemente enlazadas y colas. Se han implementado métodos específicos para cada tipo de estructura de datos, con el objetivo de facilitar su manipulación y optimizar el tratamiento de la información.

Los métodos implementados por cada estructura fueron:

DynamicArrayList:
- pushFront
- popFront
- find
- resize
- length
- empty
- print

DoublyLinkedList:
- pushBack
- topFront
- topBack
- find
- length
- delete
- print

Queue:
- enqueue
- dequeue
- peek

La utilización de cada estructura en el programa fue realizada de la siguiente manera.

DynamicArrayList: El objetivo es que los datos que se almacenan en esta estructura no sean modificados. Sin embargo, se proporcionan opciones para hacerlo en caso de que sea necesario. En este contexto, se almacenan las necesidades de los animales y de los cultivos, donde cada elemento representa un objeto de la clase Tarea.

Estas tareas se encolan para su posterior procesamiento, y luego se agregan las colas de tareas correspondientes a los trabajadores encargados. Cada tarea puede ser identificada de manera única mediante su ID, lo que facilita la búsqueda específica de elementos. Se ofrece la posibilidad de eliminar la tarea más recientemente añadida en caso de que sea necesario deshacer una acción.

Queue: En el programa, la cola se ha utilizado para almacenar objetos tanto de la clase Tarea como de la clase Producto, cada uno siguiendo una lógica diferente. Como se mencionó previamente, la lista de necesidades de animales o cultivos se encola y se asigna a un trabajador, mientras que para la clase Producto, su objetivo es permitir que el usuario comprador pueda apartar un producto en específico en el que esté interesado.

Se ha implementado la capacidad de consultar el elemento en el frente de la cola y editarlo según sea necesario. Además, se proporcionan opciones para encolar y desencolar los datos, lo que permite una gestión dinámica de la información almacenada en la cola.

El principio FIFO (First In, First Out) de las colas se ajusta perfectamente a la lógica del programa, donde se requiere un tratamiento secuencial de los datos. Esto asegura que las necesidades de animales, cultivos y productos sean gestionadas de manera ordenada y justa, garantizando un flujo eficiente de las operaciones en el programa.

DoublyLinkedList: La estructura de datos utilizada para almacenar la información de cada animal, cultivo y producto proporciona la mayor libertad para la modificación de los datos. Cada elemento almacenado en esta estructura puede ser consultado por su ID, modificado y eliminado según sea necesario. Además, se ofrece la posibilidad de agregar nuevos elementos de manera dinámica.

Esta estructura se eligió principalmente por su facilidad y rapidez de acceso dinámico a los datos. Al permitir consultas, modificaciones y eliminaciones eficientes basadas en el ID de cada elemento, se facilita la gestión y manipulación de la información almacenada.

Finalmente, cabe resaltar que a pesar de contar con la implementación de las pilas. Estas no se incluyeron en el desarrollo del programa. Más, sin embargo, sí se incluyeron en las pruebas de control y analizar a continuación.

VIII. Pruebas del Prototipo y Análisis Comparativo

Tablas de datos:

Para realizar el análisis en el tiempo se creó un paquete apartado llamado tests, en el que se implementó una clase que a través de un loop creaba la cantidad necesaria de objetos y los probaba en cada una de las funcionalidades del programa.

Insertar:

Datos	DLL (ms)	DAL (ms)	Pila (ms)	Cola (ms)
10000	6	118	5	7
100000	19	526	20	20
200000	56	1336	65	64
300000	57	2623	64	79
400000	915	4051	129	165
500000	94	6973	145	470
600000	1105	8079	36	42
700000	136	10153	502	645
800000	229	20727	182	282
900000	356	23026	53	54
1000000	251	25632	244	930

Eliminar:

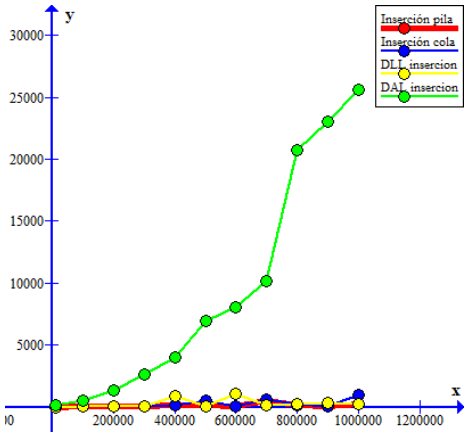
Datos	DLL (ms)	DAL (ms)	Pila (ms)	Cola (ms)
10000	78	97	2	1
100000	498	442	4	3
200000	3383	1385	4	2
300000	1430	2395	3	1
400000	2116	3040	2	3
500000	3751	4201	3	3
600000	6504	5548	7	4
700000	14678	10684	7	4
800000	19889	12551	5	4
900000	26867	14758	11	5
1000000	33580	16743	10	7

Búsqueda:

Datos	DLL (ms)	DAL (ms)
10000	85	1
100000	505	3
200000	3402	4
300000	1429	0
400000	2257	1
500000	3731	0
600000	6489	1
700000	14591	0
800000	20037	1
900000	26264	1
1000000	33703	0

Análisis gráfico

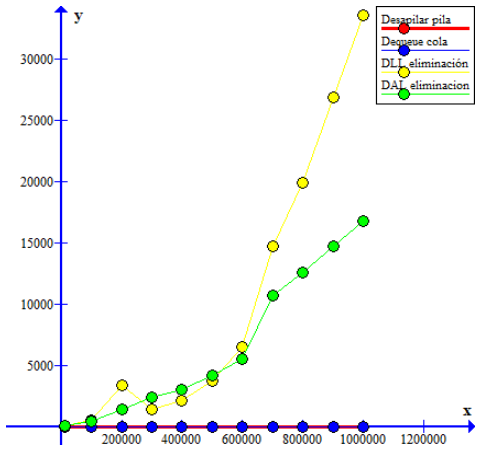
Insertar:



En este caso se ve que la cola, fila y double linked list tienen una clara ventaja con respecto a la dynamic array list, la cual

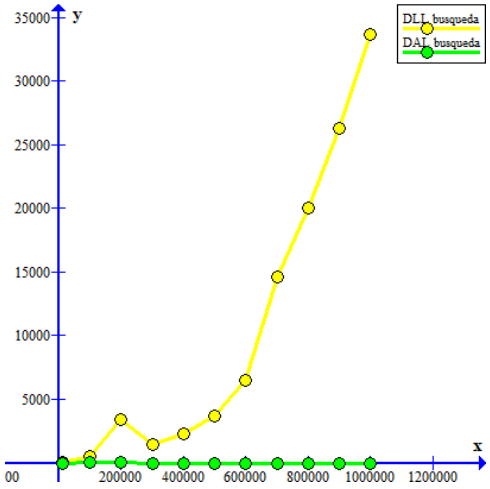
tiene un orden de O(n) o más frente a el O(1) de las demás, esto es contando claramente que se ingresen al final de la lista, pues en otro caso la complejidad aumenta enormemente.

Eliminar:



Se puede ver que la lista doblemente enlazada pierde su ventaja frente al análisis de inserción, aumentando su complejidad a ser incluso mayor que la de la dynamic array list, mientras que la pila y cola mantienen su complejidad constante.

Búsqueda:



Para este caso se observa la clara ventaja de la dynamic array list, dado que tiene puede mantener sus ventajas de ser un array al costo de tomar más tiempo al insertar datos y ser menos eficiente en memoria ubicada, en este caso no se tomó en cuenta la cola y pila por la misma naturaleza de funcionamiento de estas.

En general y realísticamente se iban a acabar considerando dos estructuras para manejar la mayoria de trabajo de almacenamiento de datos, estas siendo la lista doblemente enlazada y el array dinámico, y se acabo usando la lista doblemente enlazada por su eficiencia en la inserción de datos y por su conveniencia a la hora de apartar espacios grandes en memoria.

Análisis asintótico:

Para insertar:

- Pila: O(1).
- Cola: O(1).
- Array dinámico: O(n)
- Lista doblemente enlazada: O(1).

Para eliminar:

- Pila: O(1).
- Cola: O(1).
- Array dinámico: O(n^2)
- Lista doblemente enlazada: O(n). (Pendiente Alta)

Para búsqueda:

- Array dinámico: O(1)
- Lista doblemente enlazada: O(n^2).

IX. Información de Acceso al Video Demostrativo del Prototipo de Software

En el siguiente link se encuentra el video demostrativo de la primera versión del software:
https://drive.google.com/file/d/1msd_xWkE_2yrUeMK2mz4xrGIOPIxXPMV/view?usp=sharing

X. Roles y Actividades

INTEGRANTE	ROL(ES)	ACTIVIDADES REALIZADAS
Samuel Sanchez	Líder	Dirigir al equipo, estableciendo los objetivos y plazos de cada avance para esta primera entrega
	Observador	Comunicar en cada momento el avance del proyecto para así poder seguir dando tareas a los integrantes
		Ayudó a identificar problemas o posibles mejoras en las implementaciones.
Valentina Muñoz	Coordinadora	Asignación de tareas y responsabilidades después de las reuniones
		Programar las reuniones, dando horarios y mandando siempre los enlaces para entrar a las reuniones
	Secretaria	Toma de notas y observaciones durante las reuniones.
		Seguimiento del documento en el cual se organizó el planteamiento del problema e información que el equipo usó como bitácora.
	Animadora	Mantener un ambiente tranquilo de trabajo, motivando al equipo. Esto dando el reconocimiento y felicitación a cada integrante al tener avances. Recordar las metas de cada avance y no dejar que el grupo se desmotive o estrese durante el trabajo del proyecto.
Cristian Medina	Experto y líder de procedimiento de código	Apoyo en organización de tareas y asignación de responsabilidades durante el desarrollo de la primera entrega
		Fue el guía del equipo, el cual sabiendo cómo organizar las tareas y el desarrollo del programa, proporcionó un orden y un camino a seguir al ir programando.
	Secretario	Asignación de tareas y responsabilidades después de las reuniones

Yesid Ojeda	Investigador	Proporcionar información de uso de las plataformas que se usaron.
		Recopilar información para mejoras de en las implementaciones y desarrollo del código
	Técnico	Hacer pruebas de programa, junto a análisis de implementaciones
		Dar soluciones que optimicen el rendimiento y eficiencia del código.

En esta tabla tenemos algunas de las tareas hechas por cada integrante con su papel otorgado al inicio del proyecto para esta primera entrega.
Pero es importante recalcar que cada integrante fue partícipe del desarrollo de la creación del software, e idealización del modelo y proyección que tiene UN Campesino. Cada integrante fue partícipe de la redacción de la documentación entregable.

XI. Dificultades y Lecciones Aprendidas

- Es de gran importancia la organización del tiempo, pues este es un proyecto que no puede ser desarrollado de la noche a la mañana por lo que es necesaria una mejor organización de las tareas y sus plazos para no correr con las siguientes entregas.

XII. Referencias Bibliográficas

[1] Streib, J. T., and Soma, T., "Guide to Data Structures: Guide to Data Structures: A Concise Introduction Using Java." Springer Case Structure, 2017.

[2] Diaz, D. (2023, 29 octubre). *DaniDiazTech — Programming and Tech*. DaniDiazTech. https://danidiaztech.com/

[3] Diaz, D. (2023a, agosto 28). *How to Implement Stack in C++? Amortized analysis*. DaniDiazTech. https://danidiaztech.com/implement-stack-in-cpp/