Farm Data

Daniel Ahumada, Germán Bernal, Laura Camila Guacaneme Melo

No. de equipo de trabajo: {*A*}

I. INTRODUCCIÓN

Hoy en día existen una gran cantidad de problemas en el sector agropecuario colombiano, por un lado, el abandono del gobierno y por otro, una falta de innovación tecnológica en el desarrollo de los procesos agropecuarios, los cuales pueden ser resueltos por medio de las nuevas técnicas y herramientas de manejo de información, donde; el futuro del sector agropecuario se encuentra en los sistemas información.

Así, uno de los problemas más notorios es el manejo incorrecto de la información, los productores llevan sus registros de ventas, compras, ganancias, etc, de forma manual o desorganizada en libros de Excel, lo que ocasiona un uso deficiente de recursos.

Por esto, la solución se puede encontrar a través de la ingeniería de sistemas y computación, con la creación de varias funcionalidades, de las se piensa implementar un "CRUD", un generador de notificaciones para estar al tanto de cualquier novedad que esté sucediendo en la granja, un registro de pedidos para estar al tanto de los pedidos que van entrando y saliendo, un gestor de transporte en las granjas para estar al tanto de cualquier traslado que se esté realizando , Entre otros. Esto con dar una solución eficiente e innovadora, usando las estructuras de datos en Python a partir de las que ya están definidas, Así, esta información tendrá unas persistencias y un fácil acceso además de una fácil manipulación.

II. DESCRIPCIÓN DEL PROBLEMA A RESOLVER

Muchas veces los dueños de las granjas no tienen gran afinidad con la tecnología cosa que en la época en la que vivimos es una gran desventaja, al no tener un registro claro de las actividades que se están haciendo y una adecuado manejo de insumos se puede perder mucha inversión, ya que las ganancias se verán afectadas, sin el uso de las tecnología tendrían que hacer esto mediante libros contables pero a la hora de calcular la rentabilidad o cuando quieran hacer operaciones con los datos que tengan almacenados tendrían que contratar personas para hacer ese tipo de tareas, la versatilidad que traería tener todas estas funcionalidades seria mucha ya que se podría de una manera fácil llevar registro, y a su vez poder hacer operaciones entre ellos para tener información sobre lo que necesiten.

III. USUARIOS DEL PRODUCTO DE SOFTWARE

Los usuarios a los cuales está destinado el producto de software es para todos los productores de alimentos animales de los sectores agropecuario, ganaderos, avícolas, entre otros, que deseen tener un control y manejo de los animales que constituyen su granja, así como los implementos relacionados con estos.

Los usuarios pueden dividirse en dos principales grupos. Aquellos que sólo desean implementar el CRUD básico para el control de los datos en su finca, y los usuarios que desean implementar las funcionalidades más especializadas que brinda la aplicación.

Aún así, a pesar de esta división, el software está dirigido para personas con un nivel muy básico de experiencia en aspectos tecnológicos, de forma que sea muy sencilla y simple de utilizar.

IV. REQUERIMIENTOS FUNCIONALES DEL SOFTWARE

- CRUD
- Descripción:
    - Crear, leer, actualizar y borrar los principales datos que requiera el productor
    - Ingresar los datos y que el programa los almacene, poder leer los datos que se han ido introduciendo, y eliminar los datos que ya no se necesiten
    - esta funcionalidad debe recibir los datos que requiera el productor, se utiliza a manera de inventario para tener control sobre cualquier vaca que se desee ingresar, los datos de la vaca van a ser el número de identificación, su estado de gestación el cual se representa con 0 si no está gestando y 1 si está gestando, en el caso en el que se ponga un valor que no sea 0 o 1 el sistema no ingresara el dato hasta que se pongo el dato correctamente, y por último se tendrá el peso de la vaca.

- Gestor de pedidos
- Descripción
    - Generar y almacenar los pedidos de animales, en forma de cola
    - Ingresar un pedido y que el sistema lo tome y por orden ir resolviendo los pedidos
    - esta funcionalidad debe estar al tanto de los pedidos que se vayan generando e ir revisando su estado a medida que van entrando para hacer su respectiva entrega.

- Gestor de notificaciones
- Descripción:
    - Generar notificaciones sobre necesidades y cuidados que ocurran al pasar los días en forma de lista
    - Consultar el estado general de la granja y notificar cualquier alteración que se dé dentro de la misma
    - esta funcionalidad debe recorrer toda la lista de vacas y revisar el estado de cada una, debe notificar si hay alguna vaca que se encuentre en estado de gestación, para poder dar la alerta e indicar detalladamente la situación de esa vaca

- Gestor de transporte en la granja.
- Descripción:
    - Hacer seguimiento de las vacas que estén siendo transportadas y revisar su estado en todo momento.
    - Consultar el estado de las vacas que estén en transporte e ir actualizando el estado de dicho transporte
    - Esta funcionalidad debe revisar si se deben transportar vacas y almacenar el estado de las mismas, informar del estado del transporte y registrarlas en el nuevo espacio en el que van a estar.

V. DESCRIPCIÓN DE LA INTERFAZ DE USUARIO PRELIMINAR

En primer lugar, la aplicación contará con un Login sencillo, donde los usuarios pueden iniciar sesión con un correo y contraseña o registrarse si aún no se encuentran en la plataforma.

Ya logueados en la plataforma, el usuario encuentra como primera vista la gestión de animales, en donde puede ver las opciones de CRUD básicas para la administración de los animales. En todas las vistas se encuentra la cabecera, donde se puede hallar el nombre de la aplicación y los botones que dirigen a las demás vistas.

En la vista de pedidos se observan las opciones de agregar pedido y ver lista de pedidos.

Por último, en la vista de notificaciones encontramos todas las notificaciones almacenadas con un botón para actualizarlas. Cada una de las visitas mencionadas cuentan con un diseño simple e intuitivo que permite al usuario navegar con facilidad.

En el repositorio del código se puede encontrar un prototipo semi-funcional de la navegación de las vistas de los mockups mostrados.

VI. ENTORNOS DE DESARROLLO Y DE OPERACIÓN

El software se desarrollará en tres entornos principales: Spyder, PyCharm y Visual Studio Code. Estas herramientas facilitan el desarrollo, generación y depuración de código, estando los dos primeros entornos enfocados en el lenguaje de programación Python.

Para la integración de código y trabajo colaborativo se hará uso de la plataforma GitHub, que permite la creación de repositorios, aprovechando el sistema de control de versiones de Git y favoreciendo la organización del proyecto.

Ya en operación el software está diseñado para ejecutarse en el sistema operativo Windows, en hardware como computadores portátiles y de mesa.

VII. PROTOTIPO DE SOFTWARE INICIAL

Este link lleva al proyecto el cual implementa las estructuras abstractas por medio de listas enlazadas

https://github.com/GermanB7/FarmData

Por otro lado, se envia otro repositorio el cual hace uso de las listas doblemente enlazadas

https://github.com/GermanB7/FarmData2

VIII. IMPLEMENTACIÓN Y APLICACIÓN DE LAS ESTRUCTURAS DE DATOS

- Lista enlazada: Se crea mediante una clase de nodo el cual contiene un dato y un apuntador que señala al siguiente nodo o a un valor nulo, si es un nodo va a comprender la misma estructura de contar con un apuntador, así hasta que toda la lista sea recorrida.
  En nuestro proyecto cumple la labor de la funcionalidad CRUD, ya que nos funciona para el manejo de nuestros datos, también nos ayuda para consultar el estado de gestación de las vacas recorriendo toda la lista y dar una notificación si alguna vaca necesita atención.
- Pila: Heredando de la lista enlazada cuenta con la funcionalidad el CRUD, además cuenta con una funcionalidad que permite tener vigilado cualquier transporte que se haga de las vacas, ya que nos informa si algún grupo de vacas se encuentra dentro de un camión para un traslado, se hace uso de la pila ya que cuando el grupo de vacas llegue a su destino al depositarlas en ese nuevo espacio queremos que la

última vaca que añadimos al camión sea la primera que tenga un nuevo registro.

- Cola: Heredando de la lista enlazada cuenta con la funcionalidad el CRUD, además cuenta con una funcionalidad que es un gestor de pedidos, ya que nos mantiene al tanto de qué pedidos se están realizando y bajo el principio de "First in First out" queremos que el primer pedido que entre sea el primer pedido que esté lista para entregar.

IX. PRUEBAS DEL PROTOTIPO Y ANÁLISIS COMPARATIVO

Listas Enlazadas:

Funcionalidad: Notificaciones

#Datos	Tiempo(S)
10000	0.055409431
100000	0.534392452
1000000	5.850321388
10000000	69.49904027

Tabla 1. Tiempo en relación a los datos: notificaciones implementado con listas enlazadas simples.

Figura 1. Tiempo en función a los datos: notificaciones implementadas con listas enlazadas simples.

Al realizar la suma de operaciones se obtiene un resultado de:

```
Tp + 4Ta + n( 21Tc + 25Ta + 9Tr) + Tr
```

Cuando se simplifican las operaciones que tienen tiempos constantes se obtiene una ecuación de la forma:

```
A + n * B
```

Esta fórmula es lineal, como también se comprueba en la *Figura 1* y se puede observar que tiene una complejidad de O(N).

Funcionalidad: Apilador

#Datos	Tiempo(S)
10000	0.011601973
100000	0.25584507
1000000	3.001123953
10000000	22.87179208

Tabla 2. Tiempo en relación a los datos: apilador implementado con listas enlazadas simples.

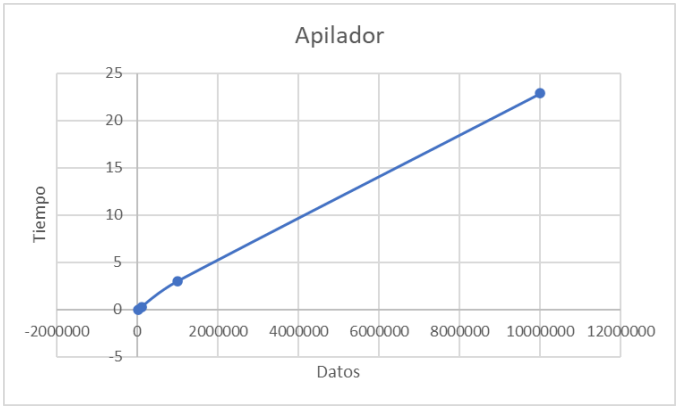

Figura 2. Tiempo en relación a los datos: apilador implementado con listas enlazadas simples.

Al realizar la suma de operaciones se obtiene un resultado de:

```
6Ta + 3Td + n * (14Tc + 8Ta + 2Tr + 2To
+ 12Td)
```

Al simplificar las operaciones que tienen tiempos constantes se obtiene una ecuación de la forma

```
A + n * B
```

Esta fórmula es lineal, como también se comprueba en la *Figura 2* y se puede observar que tiene una complejidad de O(N).

Funcionalidad: Pedidos

#Datos	Tiempo(S)
10000	0.013601971
100000	0.316155195
1000000	2.594153261
10000000	21.85211391

Tabla 3. Tiempo en relación a los datos: control de pedidos implementado con listas enlazadas simples.

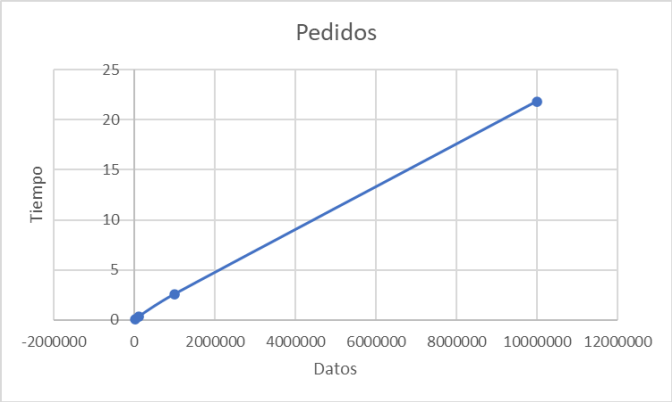

Figura 3. Tiempo en relación a los datos: control de pedidos implementado con listas enlazadas simples.

Al realizar la suma de operaciones se obtiene un resultado de:

```
6Td + 8Ta + 2To + Tc + n * (4Tc + 5Td + 22Ta + 3To + 3Tr)
```

Al simplificar las operaciones que tienen tiempos constantes se obtiene una ecuación de la forma

```
A + n * B
```

Esta fórmula es lineal, como también se comprueba en la *Figura 3* y se puede observar que tiene una complejidad de O(N).

Lista doblemente enlazada

Funcionalidad: Notificaciones

Datos	Tiempo(S)
10000	0.074013042
100000	0.885154152
1000000	8.332854795
10000000	285.1822284

Tabla 4. Tiempo en relación a los datos: notificaciones implementadas con listas doblemente enlazadas.

Figura 4. Tiempo en relación a los datos: notificaciones implementadas con listas doblemente enlazadas.

Al realizar la suma de operaciones se obtiene un resultado de:

```
10nTr + Tr + 8nTd + 12Td + 25nTa + Ta + 15nTc
```

Al simplificar las operaciones que tienen tiempos constantes se obtiene una ecuación de la forma

```
A + n * B
```

Esta fórmula es lineal, como también se comprueba en la *Figura 4* y se puede observar que tiene una complejidad de O(N).

Funcionalidad: Pedido

Datos	Tiempo(S)
10000	0.009501
100000	0.332558
1000000	3.294375
10000000	24.91695

Tabla 5. Tiempo en relación a los datos: control de pedidos implementado con listas doblemente enlazadas.

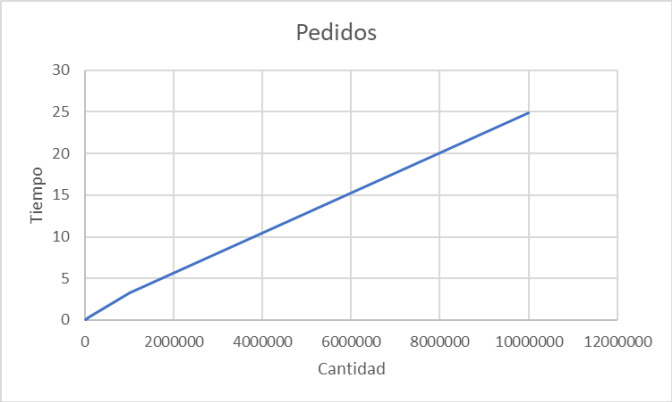

Figura 5. Tiempo en relación a los datos: control de pedidos implementado con listas doblemente enlazadas.

Al realizar la suma de operaciones se obtiene un resultado de:

```
A + B + Tc + nB + 3Td + 5Ta
```

Al simplificar las operaciones que tienen tiempos constantes se obtiene una ecuación de la forma

```
A + n * B
```

Esta fórmula es lineal, como también se comprueba en la *Figura 5* y se puede observar que tiene una complejidad de O(N).

Las pruebas de 100 millones de datos, no se pudieron realizar, ya que al momento de tratar de cargar los datos a una de nuestras estructuras implementadas siempre se llenando las listas hasta generar un error de memoria, así mismo, sucedió con 50 millones de datos por ellos se omitió cualquier prueba mayor a 10 millones de datos

Por otro lado, se omite la segunda implementación de las colas ya que según las diapositivas del curso existen tres maneras de implementarlas por medio de listas enlazadas simples, arrays y java collections, y python no hace uso de arrays estáticos ni tampoco tiene las java collections, además, hacer la implementación que se mencionó en el curso sería exactamente a la implementación de listas enlazadas que ya se habría realizado en el primer proyecto provocando que se haga un comparación con dos implementaciones iguales.

X. INFORMACIÓN DE ACCESO AL VIDEO DEMOSTRATIVO DEL PROTOTIPO DE SOFTWARE

https://www.youtube.com/watch?v=cW2W0a_dYaU&ab_channel=GermánBernal

XI. ROLES Y ACTIVIDADES

Daniel Felipe Ahumada Hernández	Líder	Diseño e implementación de la lista enlazada..
		Diseño de la funcionalidad de Notificaciones.
	Coordinador	Análisis de los tiempos del manejo de datos.
		Diseño de funciones tipo Pila.
Germán Camilo Bernal Ladino	Investigador	Diseño e implementación de la lista doblemente enlazada.
		Diseño de funciones tipo Cola.
	Técnico	Manejo de GitHub.
		Realización y edición del video demostrativo.
Laura Camila Guacaneme Melo	Experta	Creación e implementación de Mockups data.
		Implementación de la serialización como método de almacenamiento de datos.
	Observadora	Diseño de la funcionalidad de realizar pedidos.
		Diseño del bosquejo de la interfaz gráfica.

XII. DIFICULTADES Y LECCIONES APRENDIDAS

En este proyecto principalmente hubo dificultades a la hora de reunirnos y ponernos de acuerdo con los roles del proyecto, además de que se nos dificulta la tarea de unir nuestros avances y hacer que todo funcionara junto, pero a medida que avanzábamos nuestra comunicación fue mejorando haciendo que el trabajo en grupo sea más dinámico.
Otra dificultad que tuvimos fue el manejo de las estructuras, ya que, al ser nuevos en este tema, no sabíamos como hacer muchas cosas y nos enfrascamos en un pequeño error por mucho tiempo.

XIII. REFERENCIAS BIBLIOGRÁFICAS

.

[1] Campos Laclaustra, J.: *Apuntes de Estructuras de Datos y Algoritmos*, segunda edición, 2018. (En línea.)
[2] Martí Oliet, N., Ortega Mallén, Y., Verdejo López, J.A.: Estructuras de datos y métodos algorítmicos: 213 ejercicios resueltos. 2ª Edición, Ed. Garceta, 2013.
[3] Joyanes, L., Zahonero, I., Fernández, M. y Sánchez, L.: Estructura de datos. Libro de problemas, McGraw Hill, 1999.
[4] Diario Veterinario. (2021, Junio 3). La gestión de datos en granjas porcinas puede mejorar la salud animal. [Online]. Available:https://www.diarioveterinario.com/t/1441668/gestion-datos-granjas-porcinas-puede-mejorar-salud-animal