Método de la Ingeniería

Integrantes:

1. Valentina Arias Parra
2. Jose Luis Restrepo
3. Alejandro Saldarriaga

Fase 1: Identificación del Problema.

- **Contexto problemático (Síntomas y necesidades)**

1. El cliente requiere un Software capaz de atender usuarios en las distintas operaciones que realicen.
2. El cliente requiere un programa capaz de manejar su base datos.
3. Se busca implementar con estructuras de datos como Hash table, Queue, Stack, entre otras.
4. Revertir cambios hechos con la información del cliente, para hacerle contra al error humano y poder minimizar las equivocaciones.
5. La solución debe ser eficiente de tal forma que se puedan realizar tareas en el menor tiempo posible

- **Definición del problema**

Un banco necesita implementar un programa para modelar el funcionamiento de una de sus sedes con mayor flujo de personas. Este programa debe ser capaz de solventar todas las necesidades del cliente, entre las cuales se encuentra el proceso de turnos al momento del ingreso (fila para clientes y filas para personas con alguna prioridad), llevar una base de datos de los clientes y atender según la necesidad del usuario del banco. También se necesita visualizar en pantalla las filas para poderlas atender, también es necesario que la base de datos de clientes se muestre. Sus principales operaciones deben ser realizar consignaciones y retirar dinero, cancelar la cuenta bancaria o hacer el pago de tarjeta.

- **Requerimientos funcionales**

RF01- Registrar un usuario nuevo en el banco con los datos necesarios para poder realizar las distintas operaciones.

RF02- Realizar una consignación en la cuenta del usuario ingresando el monto que desea consignar.

RF03- Realizar un retiro en la cuenta de un usuario ingresando el monto que se desea retirar de su cuenta de ahorros.

RF04- Realizar una cancelación de cuenta eliminando el usuario de la lista de clientes y agregándolo a una lista de usuarios retirados.

RF05- Realizar el pago de la tarjeta, sobre el monto utilizado con la tarjeta de crédito hasta el momento.

RF06- Mostrar el estado de las filas en determinado tiempo, mostrando el número de identificación y el orden de atención.

RF07- Atender un usuario, el cual debe ingresar la acción a realizar para proceder a realizar dicha transacción.

RF08- Ordenar la lista de usuarios registrada en el banco por su nombre y mostrar la información ordenada en pantalla.

RF09- Ordenar la lista de usuarios registrada en el banco por su cédula y mostrar la información ordenada en pantalla.

RF10- Ordenar la lista de usuarios registrada en el banco por su tiempo de vinculación y mostrar la información ordenada en pantalla.

RF11- Ordenar la lista de usuarios registrada en el banco por el monto de la cuenta y mostrar la información ordenada en pantalla.

RF12- Generar una fila de preferencia para los clientes con alguna particularidad.

RF13- Deshacer cualquier acción realizada dentro del programa, considerando los errores humanos que se pueden cometer.

- **Fase 2:**

Conceptos importantes para modelar el problema:

Teóricos:

Estructura de Datos
Son una forma de almacenar y ordenar información estructuradamente con el fin de realizar sobre esta distintas operaciones de manera eficiente. Existen distintos tipos de estructuras de datos que son útiles de acuerdo al contexto del problema. Por ejemplo, una HashTable tiene la particularidad de ser una estructura de datos eficiente para la búsqueda de un dato, mientras que una Queue es útil para emular comportamientos relacionados con filas.

Base de datos
Es una herramienta que recopila, organiza y relaciona datos de una organización para poder acceder a ellos de forma eficiente y segura. Las bases de datos son ampliamente usadas en la industria para almacenar todo

tipo de información, desde datos de usuarios, hasta pedidos, cuentas, entre otros. Entre los tipos de bases de datos, se encuentran la base de datos orientadas a objetos, la cual almacena los datos en forma de objetos y clases.

Complejidad Algorítmica

Es un análisis teórico que nos brinda las herramientas necesarias para describir el comportamiento de un algoritmo en términos de su tiempo de ejecución, el tiempo que le toma a dicho algoritmo resolver un problema, y en términos de memoria, la cantidad necesaria que requiere para resolver un problema. Con este análisis teórico, es más sencillo comparar entre varios algoritmos cual posee una eficiencia superior para agilizar el rendimiento global de un programa.

Para poder realizar la comparación entre distintos algoritmos, la complejidad algorítmica de estos se expresa en notación asintótica, la cual se clasifica por órdenes, los cuales son:

$O(1)$	Orden constante
$O(\log n)$	Orden logarítmico
$O(n)$	Orden lineal
$O(n \log n)$	Orden cuasi-lineal
$O(n^2)$	Orden cuadrático
$O(n^3)$	Orden cúbico
$O(n^a)$	Orden polinómico
$O(2^n)$	Orden exponencial
$O(n!)$	Orden factorial

Órdenes de complejidad comunes.

Tabla extraída de Médium, ¿Qué es la complejidad algorítmica y con qué se come? José Guillermo.

Entre más próximo al orden constante se encuentre un algoritmo, mejor es su eficiencia en términos temporales. Por lo tanto, si se comparan varios algoritmos, se tiene en cuenta a qué tipo de orden pertenece su complejidad algorítmica para poder seleccionar aquel que esté más próximo a un comportamiento con menor crecimiento temporal.

Conceptuales:

Cuenta Bancaria:
Es un contrato financiero con una entidad bancaria con el propósito de registrar el balance y los subsiguientes movimientos de dinero del cliente. Una cuenta bancaria puede tener asociada dos tipos de cuentas bancarias en especial:

Cuenta corriente:
Es una cuenta básica en donde el usuario deposita dinero para realizar operaciones sencillas como consignaciones, pagos de servicios u otros productos, transferencias a otra cuenta o retiros. Es una cuenta a la cual se puede acceder al dinero almacenado en cualquier momento sin ningún costo adicional.

Cuenta Crédito:
Es una cuenta en la que se dispone de una cantidad de dinero acordada con la entidad financiera después de un estudio de condiciones económicas del usuario. Dicha cantidad de dinero debe ser pagada bajo las condiciones del banco con cuotas mensuales con una determinada tasa de interés.

Retiro Bancario:
Acción de extraer dinero disponible de una cuenta bancaria, ya sea una cantidad parcial o completa del saldo disponible tanto en una cuenta de ahorros o crédito.

Consignación:
Acción de ingresar dinero a una cuenta bancaria con el propósito de almacenar el dinero para realizar acciones especiales como pago de servicios, transferencias o retiros en el futuro.

Cancelación de cuenta:
Acción con la cual indicamos a un banco que queremos cancelar la cuenta que tenemos, ya sea de crédito o corriente. Por lo general, este tipo de acciones se realizan por medio de una carta, en la cual se especifican las razones por las cual el usuario decide abandonar su responsabilidad como cliente de una entidad bancaria

Pago de Tarjeta:

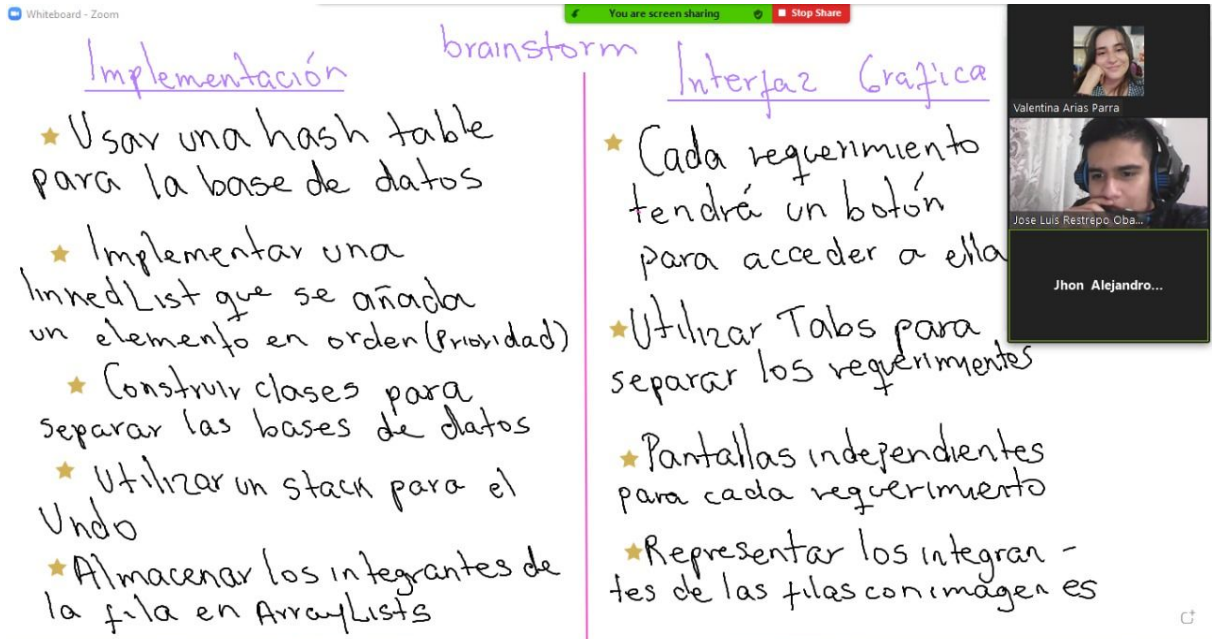
Una tarjeta de crédito es una tarjeta plastificada emitida por una compañía financiera y permite a su propietario la opción de pedir prestado dinero de la entidad bancaria. Este dinero prestado debe ser pagado antes de una fecha límite estipulada por el banco para evitar entrar en mora.

Fuentes (Ahora les pongo bien el formato):

1. https://openwebinars.net/blog/que-son-las-estructuras-de-datos-y-por-que-son-tan-utiles/
2. https://www.ticportal.es/glosario-tic/base-datos-database
3. https://medium.com/@joseguillermo_/qu%C3%A9-es-la-complejidad-algor%C3%ADtmica-y-con-qu%C3%A9-se-come-2638e7fd9e8c
4. https://redautonomos.es/gestion-financiera/poliza-cuenta-credito
5. https://www.iahorro.com/ahorro/educacion-financiera/caracteristicas_cuentas_bancarias.html
6. https://www.scotiabankcolpatria.com/personas/tarjetas-de-credito

- **Fase 3: Búsqueda de soluciones creativas.**

Para esta búsqueda de soluciones creativas hicimos una ***lluvia de ideas*** en la plataforma Zoom en la que todos aportamos nuestros puntos de vista sobre cómo abordar el problema y darle solución a cada requerimiento de este.



En la división del lado izquierdo, se discutió sobre qué estructura de datos era la apropiada para el modelamiento de las filas y las bases de datos, al igual se discutió

la forma en implementar dichas estructuras. En la división del lado derecho, se dieron diferentes ideas y propuestas acerca del diseño para la interfaz teniendo en cuenta que cumpliera con todas las funcionalidades que requiere el programa. Después de estar un largo tiempo discutiendo se llegaron a las siguientes posibles soluciones:

1. **Idea 1:** Realizar un software que tenga como estructura contenedora de usuarios un ArrayList y que mediante este se puedan realizar las transacciones de usuarios y visualización de información.
2. **Idea 2:** Realizar un programa utilizando persistencia para guardar la base de datos del banco, en el cual se escriba y lea la información de los archivos dependiendo de la operación realizada. Para las filas utilizar una LinkedList que se ordene de acuerdo a la prioridad.
3. **Idea 3:** Realizar un software que utilice diversas estructuras de datos para contener la información, pues los usuarios se pueden guardar en una tabla hash al igual que los usuarios retirados, mientras que, para mostrar la información en tablas se puede usar un ArrayList. Por otro lado, las filas se pueden manejar con Queues y Priority Queues de tal forma que permitan realizar las operaciones correspondientes a cada usuario, mientras que, para visualizar dicha información se transfiere la información a ArrayList para ser desplegados en TableViews. Además de esto, con la estructura stack se podría realizar la función undo, la cual que permitirá deshacer y corregir errores.
4. **Idea 4:** Realizar un programa que guarde los usuarios en un ArrayList y maneje las filas con Queue genéricas y de prioridad, de forma que permita visualizar el estado de estas y atender al cliente para realizar las operaciones solicitadas.

Fase 4: Diseños preliminares.

Ideas descartadas:

Opción 1:
Esta opción se descarta ya que, si bien un arrayList puede almacenar a los usuarios y a la vez puede mostrarlos en una TableView, a la hora de buscar alguno usuario, este algoritmo resulta ser muy ineficiente, ya que en el peor de los casos, la complejidad de búsqueda para un ArrayList es O(n). Por lo tanto, se decide usar un Hash Table, ya que la complejidad algorítmica del algoritmo de búsqueda es O(1).

Opción 2:

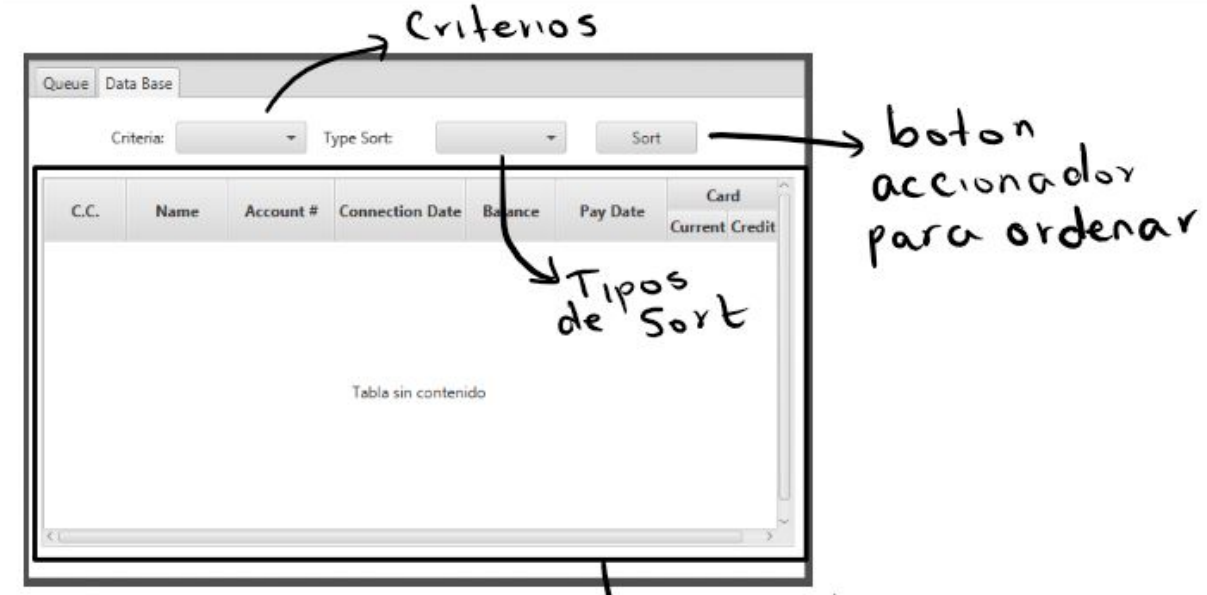
Esta opción se descarta, ya que si bien la persistencia es un buen mecanismo para guardar la información, el cliente no especifica que se implemente esta función en el programa. Por otro lado, modelar las filas con una LinkedList resultaría ineficiente al agregar un nuevo usuario, debido a que se debería recorrer toda la estructura para añadir un nodo al final, por lo tanto, se decide implementar una Queue para modelar las filas, ya que esta estructura posee los punteros, el orden necesario de inserción y eliminación para modelar correctamente una fila.

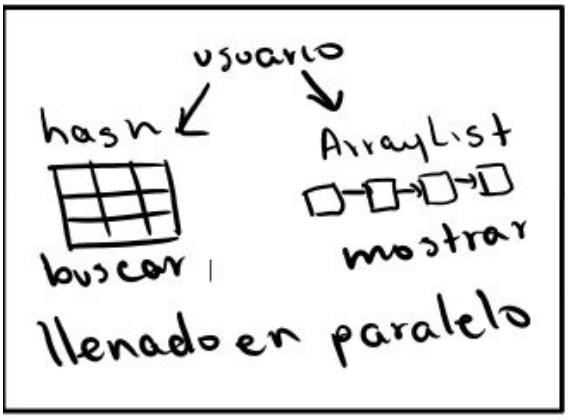
Opción 4:
Al descubrir la existencia de una Tabla Hash, se descartó completamente usar LinkedList o ArrayList para almacenar los usuarios, ya que esta estructura cuenta en su algoritmo de búsqueda con una complejidad, en el mejor de los casos, O(1) lo cual es bastante eficiente para buscar usuarios en una voluminosa cantidad de datos.

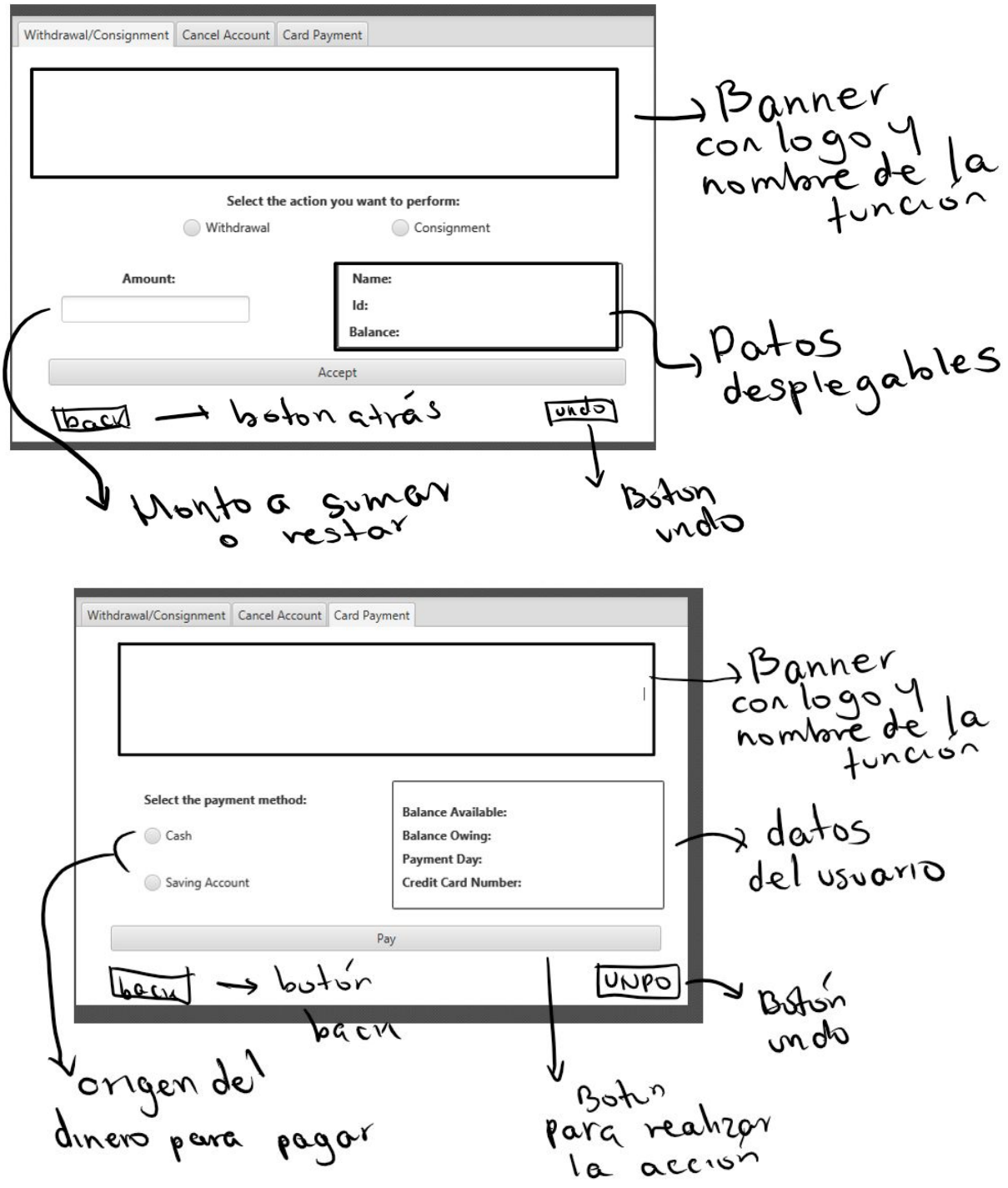
Ideas realizadas:

Opción 3:
De todas las opciones mencionadas, esta opción reunía las estructuras de datos necesarias para poder modelar el problema de forma eficiente, debido a que se debía usar un tipo de estructura de datos para guardar la información y al mismo tiempo, había que usar otra estructura de datos para desplegar la información en pantalla. En ese orden de ideas, para almacenar la información, se decidió modelar las filas comunes y de prioritarios con Queues, ya que esta estructura cuenta con los punteros y el orden de inserción necesarios para poder emular una cola real. Por otro lado, para almacenar todos los usuarios del banco, se optó por usar una HashTable debido a su eficiencia a la hora de buscar un usuario. Finalmente, para implementar la funcionalidad de Undo se decidió usar una Stack debido a que esta estructura tiene el orden y los punteros necesarios para emular una pila de errores.

Sketches de las pantallas.







Fase 5: Evaluación y selección de soluciones.

Criterios de evaluación:

- **A.** Manejo de estructuras eficientes.
- **B.** Cumplimiento de requisitos.
- **C.** Precisión de la solución.
- **D.** Facilidad de implementación.

Escala de calificación:

- **[0]** Deficiente: No cumple con el criterio.
- **[1]** Aceptable: Cumple medianamente con el criterio.
- **[2]** Alto: Cumple el criterio con totalidad.

Evaluación:

Idea 3:

A. (2)
B. (2)
C. (2)
D. (2)

Total: 8 puntos.

Idea 4:

A. (1)
B. (1)
C. (1)
D. (2)

Total: 5 puntos.

La idea que se mejorará y será implementada es la idea 2, ya que tuvo una calificación de 8 contra una calificación de 5.