Jhon Stiven Arboleda, Sebastián Barrera, Alejandro García

Tarea Integradora 1

1. **Identification of the problem**

A large bank wants to develop software that models the operation of one of its offices with the highest flow of people. Its main need is to be able to fulfill all the client's needs.

The bank needs:

* Manage shifts by entering the lines either the customer line or a line with different priorities.
* Manage data tables with all customer information.
* Allow the customer to perform different operations at the time of serving, that is, when their turn comes.

Problem: The bank has no way to handle these types of actions effectively in a software for a high flow of problems.

1. **Research**

**Functional requirements:**

|  |  |
| --- | --- |
| **Name** | **FR1:** Register a customer |
| **Summary** | The system must register a client at the time of obtaining their turn with their respective data |
| **Input** | **name** of the client and **identification** of the client |
| **Output** | The client has been successfully registered |

|  |  |
| --- | --- |
| **Name** | **FR2:** Assign to a row |
| **Summary** | The system must register a client at the time of obtaining their turn with their respective data |
| **Input** | **name** of the client and **identification** of the client |
| **Output** | The client has been successfully registered |

|  |  |
| --- | --- |
| **Name** | **FR3:** Search the database |
| **Summary** | The person in charge of the attention will be able to look for the client in the database with his identity card before the client arrives at his office |
| **Input** | **Identification card** of the client |
| **Output** | Table with customer data. Such as name, identity card, bank account, debit / credit cards, date of payment of the credit card and date it was incorporated into the bank |

|  |  |
| --- | --- |
| **Name** | **FR4:** Withdrawals/Deposit |
| **Summary** | The client may modify the amount of his savings account when requesting a withdrawal or consignment. |
| **Input** | **Bank account** of the client |
| **Output** | The account has been modified |

|  |  |
| --- | --- |
| **Name** | **FR5:** Cancel account |
| **Summary** | Deletes your information from the customer database and adds them to a database exclusively for those who cancel their accounts at the bank. In any case, both the date and the reason for cancellation will be saved. |
| **Input** | **Bank account** of the client |
| **Output** | The account has been deleted |

|  |  |
| --- | --- |
| **Name** | **FR6:** Card payment |
| **Summary** | The user can pay the amount used with the credit card so far.  You can make the payment in cash or through your savings account. |
| **Input** | **Bank account** of the client |
| **Output** | The amount of money has been paid. |

|  |  |
| --- | --- |
| **Name** | **FR7:** Perform *undo* |
| **Summary** | It will serve to undo mistakes, even after they have been saved |
| **Input** | **----------** |
| **Output** | The action has been undone. |

**Stacks:** Stacks are data structures wherein the last element that enters is the first element that leaves (LIFO, **L**ast **I**n **F**irst **O**ut).

**Queues:** Queues are data structures wherein the first elements that enters is the first element that leaves (FIFO, **F**irst **I**n **F**irst **O**ut)

**Hash Tables:** Hash tables are data structures that map keys to values. In the Java programming language, keys and values can be any object that is not null. The hash tables use a hash function to map the key to the value. This means that to retrieve a value the user must enter the key associated with that value.

A problem that might arise with hash tables are collisions, which happen when a given key has more than 2 values associated with it. These problems are solved with open addressing and chaining. In open addressing, the second value is stored in another address. In chaining, both values are stored in a linked list.

**Bank:** bank service

1. **Creative solutions**

**Brainstorming:**

* 1. Keep a manual record of users.
  2. Hire third-party software that fulfill its functions.
  3. Create a software from scratch that solves your own needs.

1. **Selection of the best solution**

Final decision: