# https://lh6.googleusercontent.com/JOlFMiIkATRLw4AVmD5a-Vav00tP11-j2YZpiqaSK3hpG4gCiu4XnO_xbLZ79ujN81_qbk7DRcR85vMLKP0g921x1sBfwoJp19NKcn-UDW2oISL6X70MMXKJTFPoHwN4nnwK3k9q Método de la Ingeniería

**Aplicación para la Solución de un Problema**

**Miguel Angel Romero Rosas, Juan David Lopez y Emanuel Suluaga**

**Contexto Problemático**

Una empresa de EpicGames esta evaluando la posibilidad de implementar un ranking en donde pueda categorizar a los jugadores según su destreza para el juego (skill).Por otra parte, la empresa quiere se puedan realizar partidas en modo “Plataforma”, lo cual permite limitar los jugadores que se unirán a la partida de acuerdo a la plataforma que estén utilizando. Por último, requiere la creación de variantes del juego para celebrar ocasiones especiales (como lo son navidad, halloween, pascua, etc)

**Desarrollo de la Solución**

**Paso 1. Identificación del Problema**

**Identificación de necesidades y síntomas**

·inconvenientes que los usuarios han experimentado con respecto al gameplay de sus partidas.

· La empresa no cuenta  con un sistema de clasificación que permita a los usuarios ser rankeados, que no afecte el tiempo promedio de búsqueda de una nueva partida entre n usuarios con ranking similar.

· La empresa no cuenta con una opción en donde se asegure que todas las personas en una partida tenga un latencia muy cercana. Es decir, que la desviación estándar del ping de los jugadores de una partida sea el mínimo posible (para asegurar la igualdad de condiciones en la partida).

·  La empresa quiere proponer una implementación para el modo plataforma.

· La empresa quiere proponer una implementación del modo de juego que será lanzado para San Valentín

· La solución del problema debe de ser eficiente ya que esto evitaría que no se afecte el promedio de búsqueda a la hora de encontrar una nueva partida

**Paso 2. Requerimientos Funcionales:**

A Continuación se presentan los 3 principales requerimientos del programa (Ranking, Plataforma, Invariantes) seguidos de otras necesidades fundamentales para el correcto funcionamiento del programa.

***Ranking (Skill):***

Se necesitan encontrar alternativas de algoritmos o estructuras que nos permitan crear de manera acertada y eficiente  un ranking ordenado según la habilidad de cada jugador. El valor “Skill” estará representado por un número decimal.

***Plataforma:***

Se necesita encontrar alternativas de algoritmos o estructuras que nos permitan separar y administrar los jugadores dependiendo de la plataforma que se encuentre. En cada una de las plataformas deben encontrarse jugadores con la misma o parecida destreza. Cada plataforma estará representada por una cadena la cual describe una cualidad de su modo.

***Invariantes:***

Se necesita encontrar alternativas de algoritmos o estructuras que nos permitan manejar los axiomas postulados y requeridos en el problema. Para esta ocasión en especial (San Valentín), se toma como invariantes:

* Solamente se puede utilizar la última arma que se levanta.
* No hay un límite de armas para ser llevadas.
* Solo se puede cambiar el arma encontrando una nueva o agotando las municiones del arma actual; cuando esto sucede automáticamente se equipara la última arma que se levantó antes de la agotada.
* La primera arma que se equipa por defecto es el hacha.
* El hacha no tiene municiones por ende no se puede descartar.
* La interfaz gráfica debe ofrecer una opción mediante la cual la persona pueda crear el usuario que va a ser ingresado a la plataforma.
* . El programa debe de permitir realizar partidas en modo plataforma, en donde habrá un número límite de usuarios.
* .

**Paso 3. Recopilación de Información**

**Definiciones**

**FortNite:**

**​**es un videojuego desarrollado por las empresas Epic Games y People Can Fly que, a pesar de ser anunciado en 2011, no fue lanzado hasta el 25 de julio de 2017. Se trata de un juego de zombis en el que supervivientes controlados por humanos cooperarán online para mantener sus fortalezas a salvo de los ataques en oleadas de los muertos vivientes, que están intentando conquistar la tierra. El modo Fortnite Battle Royale, basado en el género de Batalla Real, fue lanzado de forma independiente en septiembre de 2017 para Microsoft Windows, macOS, PlayStation 4 y Xbox One. En este modo de juego participan hasta 100 jugadores y luchan todos entre si hasta que solo queda uno vivo: el ganador.

**Latencia:**

La latencia es el tiempo que tarda en transmitirse un paquete dentro de la red, y es un factor clave en las conexiones a Internet. En función de la conexión que tengamos, esta latencia será mayor o menor. La latencia influye, por ejemplo, en el tiempo que tarda en cargar una web.

**Ping:**

Para medir la latencia, se utiliza el ping, muy conocido sobre todo por aquellos que jueguen a videojuegos online. El ping, medido en milisegundos (o ms) mide el tiempo que tardan en comunicarse una conexión local con un equipo remoto en la red IP. Por ejemplo, en un videojuego de disparos, cuanto menor sea el ping, menos tiempo tardará en registrar el servidor que tú has disparado y que la bala ha alcanzado su objetivo.

**Mod:**

es una extensión del software que modifica un videojuego original proporcionando nuevas posibilidades, ambientaciones, personajes, diálogos, objetos, etc. En la actualidad, prácticamente todos los videojuegos importantes para PC incorporan herramientas y manuales para que exista la posibilidad de modificarlo al gusto del jugador.

**Tipo abstracto de datos: (TAD)**

1. Un conjunto   de valores   y operaciones asociadas
2. especificados de manera precisa  e independiente de la implementación

**Notación:**

1. El estado de un Tad viene dado por el número de operaciones dado sobre el.
2. La definición de las operaciones suele darse mediante axiomas de reglas lógicas

Un tipo de dato abstracto (TDA) o Tipo abstracto de datos (TAD) es un modelo matemático compuesto por una colección de operaciones definidas sobre un conjunto de datos para el modelo. Algunos ejemplos de utilización de TDA´s en programación son:

**Conjuntos:** Implementación de conjuntos con sus operaciones básicas (unión, intersección y diferencia), operaciones de inserción, borrado, búsqueda...

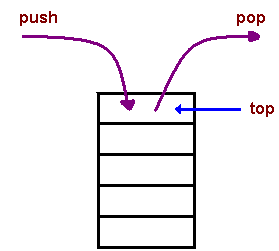
**Árboles Binarios de Búsqueda:** Implementación de árboles de elementos, utilizados para la representación interna de datos complejos. Aunque siempre se los toma como un TDA separado son parte de la familia de los grafos.

**Pilas y Colas:** Implementación de los algoritmos FIFO y LIFO.

**Grafos:** Implementación de grafos; una serie de vértices unidos mediante una serie de arcos o aristas.

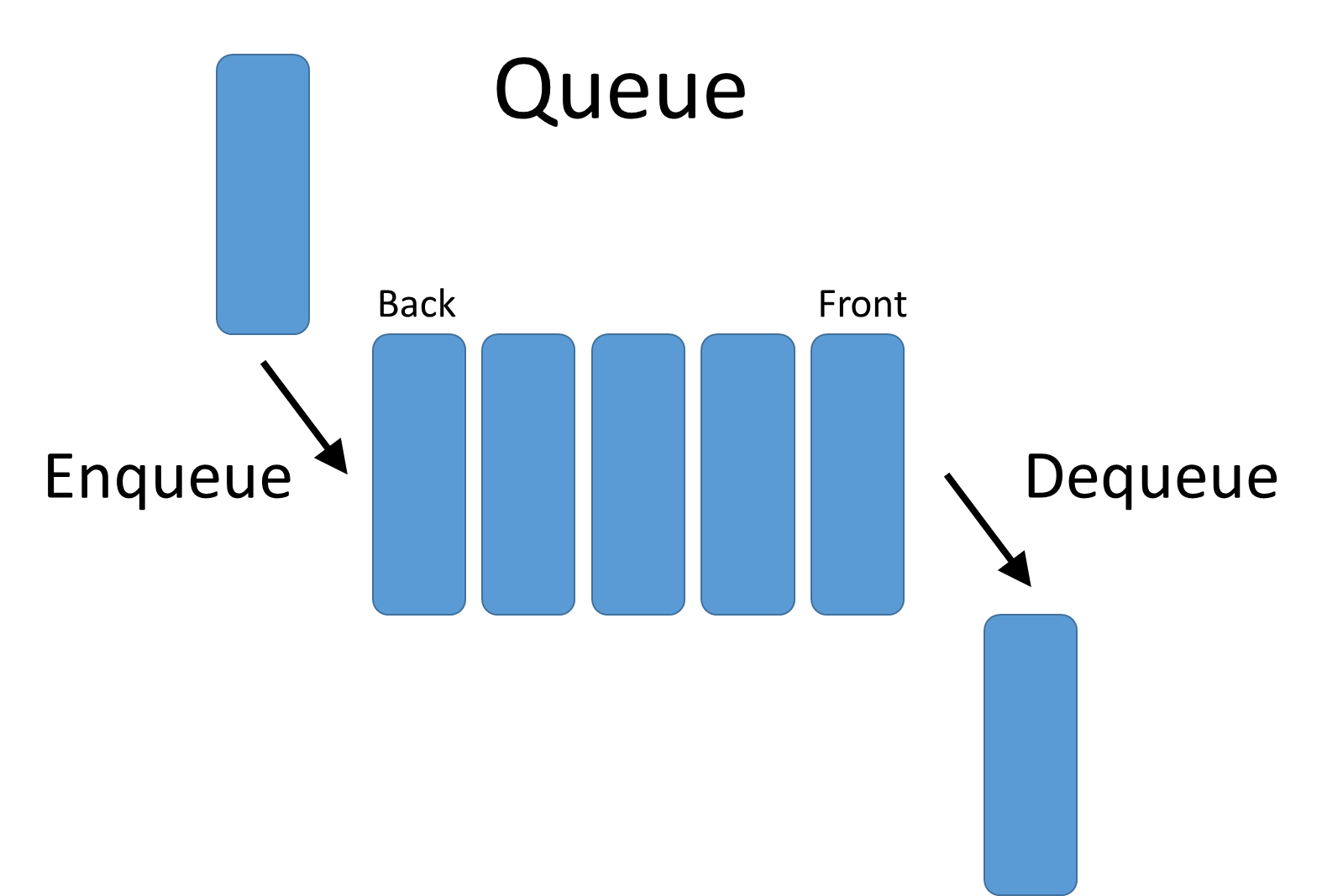
Pila

Es una lista ordenada o [estructura de datos](https://es.wikipedia.org/wiki/Estructura_de_datos) que permite almacenar y recuperar datos, el modo de acceso a sus elementos es de tipo [LIFO](https://es.wikipedia.org/wiki/LIFO) (del inglés *Last In, First Out*, «último en entrar, primero en salir»).



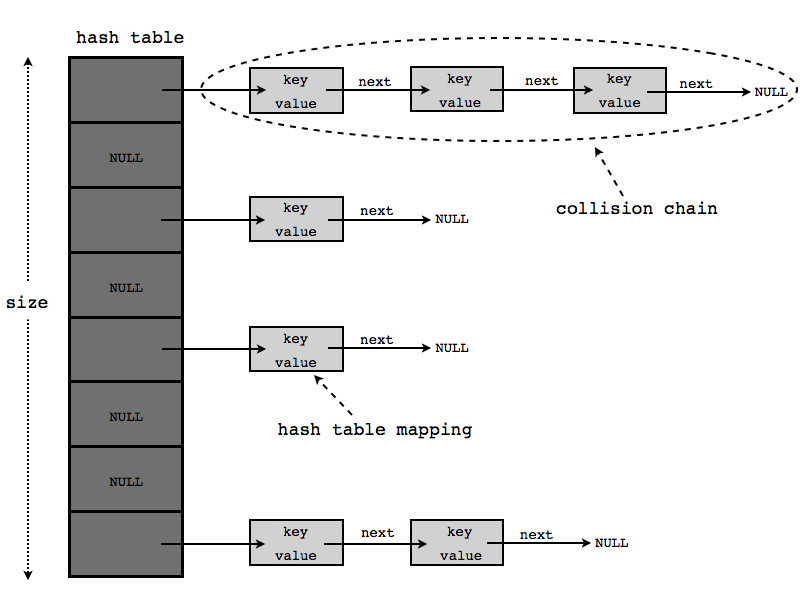
Cola

Es una [estructura de datos](https://es.wikipedia.org/wiki/Estructura_de_datos), caracterizada por ser una secuencia de elementos en la que la operación de inserción *push* se realiza por un extremo y la operación de extracción *pop* por el otro. También se le llama estructura [FIFO](https://es.wikipedia.org/wiki/FIFO) (del inglés *First In First Out*), debido a que el primer elemento en entrar será también el primero en salir.



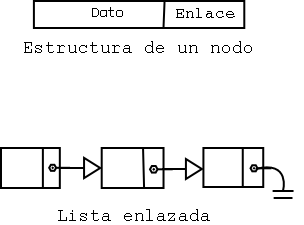
Tablas Hash

Es una [estructura de datos](https://es.wikipedia.org/wiki/Estructura_de_datos) que asocia *llaves* o *claves* con *valores*. La operación principal que soporta de manera eficiente es la *búsqueda*: permite el acceso a los elementos (teléfono y dirección, por ejemplo) almacenados a partir de una clave generada (usando el nombre o número de cuenta, por ejemplo). Funciona transformando la clave con una [**función hash**](https://es.wikipedia.org/wiki/Funci%C3%B3n_hash) en un [*hash*](https://es.wikipedia.org/wiki/Hash), un número que identifica la posición (*casilla* o *cubeta*) donde la tabla hash localiza el valor deseado.



Listas Enlazadas

Consiste en una secuencia de [nodos](https://es.wikipedia.org/wiki/Nodo_(inform%C3%A1tica)), en los que se guardan campos de datos arbitrarios y una o dos referencias, enlaces o [punteros](https://es.wikipedia.org/wiki/Puntero_(inform%C3%A1tica)) al nodo anterior o posterior.



**Paso 4. Alternativas Creativas:**

**Pila para el Ranking:**

En este proyecto utilizaremos varias alternativas creativas, entre varias de ellas encontramos el utilizamiento de los tads, colas y pilas. Tendremos que utilizar varios tads, tanto para obtener el ranking de los usuarios en donde se pueda categorizar a los jugadores según su destreza para el juego.

**Pila para el ingreso del Usuario a la Plataforma:**

En este caso lo que vamos a hacer es crear una pila de Usuarios los cuales serán ingresados a la plataforma lo cual permitirá limitar los jugadores que se unirán a la partida de acuerdo a la plataforma que estén utilizando.

**Pila para agregar arma:**

Actualmente la versión que aprobaron para el día de San Valentín es un modo de juego donde solamente puedes utilizar la última arma que levantaste. En este modo de juego no tienes límite de armas para llevar y solamente podrás cambiar de arma encontrando una nueva o agotando las municiones de tu arma actual, lo cual hará desaparecer el arma actual y te equipará con la última arma que levantaste antes que esa.R

# Paso 3. Definición de los Tads

* **TAD para Pilas**

|  |
| --- |
| **Nombre: Stack (pila)** |
| **Dibujo:** |
| **Invariante:**  Tamaño= 1 |
| **Operaciones Primitivas:**  createStack = Stack  top = Stack x Element Weapon  pop = Stack x Weapon Stack  push = Stack x Element Stack  peek = Stack Element  isEmpty = Stack Boolean |

|  |
| --- |
| **createStack()** |
| “construye un Stack vacio”  {pre:}  {post: stack sta=0} |

|  |
| --- |
| **Top()** |
| “Debe devolver el elemento que se encuentran en el topa de la pila actualmente”  {pre: ture}  {post: Weapon ={name=” ”}} |

|  |
| --- |
| **Pop()** |
| “Elimina el elemento que se encuentra en el tope de la pila actualmente”  {pre: ture}  {post: Weapon ={name=” ”}} |

|  |
| --- |
| **Push()** |
| “Ingresa el elemento a la posición tope de la pila y este devuelve el elemento agregado”  {pre: ture}  {post: Weapon ={name=” ”}} |

|  |
| --- |
| **Peek()** |
| “Recupera el valor del elemento que se encuentra en el tope de la pila”  {pre: Stack s !=0 es decir, s =< a1,a2,a3,…,an}  {post: Element an} |

|  |
| --- |
| **isEmpty()** |
| “Informa, si la pila esta o no vacia ”  {pre: t: Stack s}  {post: true if s==0, false if l=0} |

* **TAD para HashTable**

|  |
| --- |
| **Nombre: HashTable (Players)** |
| **Dibujo:** |
| **Invariante:**  Inv: función hash |
| **Operaciones Primitivas:**   * create Table = HashTable * tableInsert = HashTable x Element HashTable * tableDelate = HashTable x SearchKey Element * isEmpty = HashTable x boolean * tableLength = HashTable int |

|  |
| --- |
| **createTable()** |
| **“**Crea un hashTable completamente vacio” |
| {pre: }  {post: HashTable h !=0} |

|  |
| --- |
| **tableInsert()** |
| **“**Agrega a la TablaHash h un nuevo elemento x” |
| {pre: el nuevo elemento por agregar no debe estar en la hashTable}  {post: el nuevo elemento se agrega con exito} |

|  |
| --- |
| **tableInsert()** |
| **“**Agrega a la TablaHash h un nuevo elemento x” |
| {pre: el nuevo elemento por agregar no debe estar en la hashTable}  {post: el nuevo elemento se agrega con exito} |

|  |
| --- |
| **tableDelate()** |
| **“**elimina un elemento según su código” |
| {pre: el elemento existe en la hashTable}  {post: el elemento se elmina de la hashTable con exito} |

|  |
| --- |
| **tableDelate()** |
| **“**elimina un elemento según su código” |
| {pre: el elemento existe en la hashTable}  {post: el elemento se elmina de la hashTable con exito} |

|  |
| --- |
| **isEmpty()** |
| **“**verifica si esta vacio” |
| {pre: debe existir la tabla hash}  {post: ejecuta la verificación correctamente} |

|  |
| --- |
| **isEmpty()** |
| **“**verifica si esta vacio” |
| {pre: debe existir la tabla hash}  {post: ejecuta la verificación correctamente} |

|  |
| --- |
| **tableLength()** |
| **“**tamaño de la tabla hash” |
| {pre: debe existir la tabla hash y debe tener un tamaño definido}  {post: retorna el tamaño de la tabla} |

**Paso 4. Elementos relacionados entre las Alternativas creativas escogidas**

Una de las relaciones que Podemos encontrar entre las soluciones creativas es que todas tratan de organizar los elementos por medio de tads. Si bien, cada uno tienen una manera más eficiente para resolver el problema en el peor de los casos, todas ellas están pensadas para satisfacer la necesidad de ordenar una entrada de elementos ya sean la cantidad de municiones del arma, cuantas armas tiene el personaje, cuantos usuarios estarán disponibles en la plataforma, y el ranking del usuario. Las practicas que se realizan en cada una varían demasiado entre cada alternativa. Pero algo en común es que se pueden aplicar a pilas, colas, tablas hash, y montículos de forma lineal. La otra característica es la facilidad en implementación algorítmica: Las soluciones encontradas son compatibles con las operaciones básicas que tendrá que hacer el usuario a la hora de ingresar al juego, hacienda el problema no complejo de tratar, ya que las soluciones escogidas son posibles en todos sus sentidos.

**Fase 4. ALTERNATIVAS DE SOLUCIÓN CREATIVAS:**

1**) Alternative 1:**

El usuario tiene la opción de ingresar las entradas por medios manuales. Existen dos posibilidades: por medio de cuadros emergentes y por archivos de texto plano. el usuario verá un cuadro donde podrá crear el usuario que será ingresado a la plataforma. También, puede elegir la ruta donde tiene su archivo de texto plano en donde tiene la partida guardada, en donde se encontrará el Ranking, el nivel, el total de armas, etc. Estos números, se agregarán ya sea en las pilas, colas, tabla hash o montículos. Finalmente, se mostrarán en una JList la lista de forma ordenada todos los elementos del usuario. Cabe resaltar que en esta opción el usuario tendrá la Interfaz dividida en dos, la primera parte será donde el tendrá que ingresar sus datos para poder conectarlo con el sistema y la otra es en donde este tendrá la oportunidad de crear un nuevo usuario con todos los elementos descritos anteriormente.

**2) Alternative 2:**

El usuario ingresará los datos por medio de un campo de texto. En él, el usuario deberá registrar el usuario o ya sea ingresar como un usuario ya existente. Una vez ingresados, se guardarán en una pila. Cuando el usuario de la orden de guardar los elementos, el campo de texto se vaciará. El programa deberá controlar todo con respecto a si el campo de texto es vacío o se ingresan datos diferentes a los correspondientes. Luego, cuando el usuario decida crear su usuario o agregar una nueva arma, se llamarán a los métodos para crear ya sea una pila o una cola, dependiendo de lo que el usuario vaya a guardar. Finalmente, se mostrará el arreglo ordenado en un campo de texto y en una lista. Además, mostrará todas las características del usuario, tanto sus armas, como el ranking que tiene este con respecto a su desempeño en el juego. Por otra parte, el programa contara con una opción de disparar en esta opción el podrá disparar las balas con las armas que tiene hasta el momento, una vez las balas se le acaben obtendrá la ultima arma que tome antes de que se acabara la otra.

**3) Alternative 3:**

El programa le mostrara dos opciones en las cuales el usuario tendrá que escoger, dependiendo que opción quiera, ya sea crear un nuevo usuario o Iniciar sesión. Si el usuario escoge la opción de iniciar sesión el usuario deberá ingresar los datos de este por medio de un archivo de texto plano. El programa tendrá un campo donde el usuario podrá ingresar la ruta de su archivo y por medio de un botón se leerá, así serán cargados todas las características del usuario al programa. Si no se encuentra ningún error una vez terminado el proceso de leer el archivo, se almacenarán los datos en el programa, los datos que ya habían sido guardados, como el nombre, apellido, nombre de usuario, correo electrónico, la contraseña, los cuales serán guardados en una tabla hash. Las armas que este posee serán guardadas por medio de pilas, el número de municiones será guardado por medio de una cola y el ranking que este tiene también. Por otro Lado, cuando el usuario ingrese al sistema, tendrá dos opciones, las cuales serán; Iniciar en una partida o crear una partida. En este proyecto solo trabajaremos con la primera opción, la cual es iniciar en una partida. Una vez el Usuario escoja esta opción, el programa se encargará de ingresarlo a una partida en donde el ranking de los usuarios sea similar, para que no haya desventaja entre estos.

Finalmente, se mostrará por medio de un cuadro de texto todos los usuarios que están actualmente en la partida escogida, las armas que tienen, las municiones y el ranking de cada uno.

**4) Alternative 4:**

El usuario ingresará los datos por medio de un campo emergente. Dicho cuadro, aparecerá cuando el usuario de la orden de crear el usuario o iniciar como un usuario existente. Los datos serán almacenados en un Arreglo dinámico. Una vez que el usuario haya creado el usuario o haya ingresado al sistema como usuario ya existente, el programa deberá mostrarle las opciones al usuario de ingresar a una partida, dependiendo de el ranking de este usuario, el cual será implementado como una tabla hash. Una vez el usuario haya ingresado a la plataforma tendrá las opciones de coger una nueva arma, la cual será mostrada por el sistema preguntándole que nueva arma le gustaría escoger, por medio de un JList estarán las armas que el usuario puede obtener, las cuales estarán guardadas al mismo tiempo en una cola o tabla hash, y las armas que el usuario tiene serán guardadas en una pila, ya que se basa en el criterio Last in, First Out. Así el usuario tendrá la opción de usar la última arma que recogió (San Valentine Day). Finalmente, se exporta un archivo de texto plano con las características del usuario, cuantas armas tiene, cuantas, las municiones de cada una de sus armas, el ranking de este, etc.

**Paso 5. Fuentes:**

[**https://jarroba.com/tablas-hash-o-tabla-de-dispersion/**](https://jarroba.com/tablas-hash-o-tabla-de-dispersion/)

[**https://www.monografias.com/trabajos92/arboles-binario/arboles-binario.shtml**](https://www.monografias.com/trabajos92/arboles-binario/arboles-binario.shtml)

[**https://estructuradedatosunivia.wordpress.com/2014/11/18/metodos-de-ordenamiento-mergesort/**](https://estructuradedatosunivia.wordpress.com/2014/11/18/metodos-de-ordenamiento-mergesort/)

[**http://ocw.upm.es/lenguajes-y-sistemas-informaticos/estructuras-de-datos/contenidos/tema2/TADs.pdf**](http://ocw.upm.es/lenguajes-y-sistemas-informaticos/estructuras-de-datos/contenidos/tema2/TADs.pdf)

[**https://www.aprenderaprogramar.com/index.php?option=com\_content&view=article&id=608:la-estructura-de-datos-pila-en-java-clase-stack-del-api-java-ejemplo-simple-y-ejercicios-resueltos-cu00923c&catid=58&Itemid=180**](https://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=608:la-estructura-de-datos-pila-en-java-clase-stack-del-api-java-ejemplo-simple-y-ejercicios-resueltos-cu00923c&catid=58&Itemid=180)

[**http://michelletorres.mx/colas-en-java/**](http://michelletorres.mx/colas-en-java/)

**http://www.ciberaula.com/articulo/colas\_en\_java**

**Fase 6. Búsqueda de Soluciones Creativas**

**Paso 1: método generador de ideas:**

Para desarrollar las alternativas creativas usamosel método de lluvia de ideas. Primero, dividimos el problema en tres categorías: entradas, procesos, salidas. Luego, en cada categoría, enlistamos ideas abstractas que nos sirven como sustantivos para la siguiente etapa. Categorías como “procesos” tiene ya predefinidas las alternativas investigadas en el paso anterior de recolección de información. Con las categorías de entrada y salida lo que buscamos es buscar componentes que nos permitan darnos una idea de cómo recibir y mostrar la información, como por ejemplo en la categoría de procesos tenemos las opciones de Pilas, colas, tablas hash, montículos. A continuación, cada componente enlistado en cada categoría es indexado. Por ejemplo, las ideas que salieron en la categoría de entrada están enumeradas; las categorías en proceso están ordenados alfabéticamente y en la categoría de salidas están enlistadas con números romanos.

|  |  |  |
| --- | --- | --- |
| Entradas | Procesos | Salidas |
| 1. Cuadro de Texto | 1. Pilas | 1. JList |
| 1. JTextField | 1. Colas | 1. Campo de Texto |
| 1. Archivo Plano | 1. Montículos | 1. Cuadro Emergente |
| 1. Cuadro Emergente | 1. Tablas Hash | 1. Etiquetas |
|  |  | 1. Archivo |
|  |  | 1. Barra de progreso |
|  |  |  |

**Soluciones:**

1. **1-A-B-C-ii**
2. **2-B-D-C-iii**

Una vez teniendo llena esta tabla, cada uno procede a escribir combinaciones tomando componentes de cada categoría. Decidimos poner dos condiciones, la primera es solo tomar un componente de la categoría de entrada, y varios diferentes, de la categoría de procesos. De esta forma, nos aseguramos tener un solo componente de entrada y los tres métodos de ordenamiento requeridos. Con respecto a la salida decidimos seleccionar más de uno pues nos daría una general, o alternativas, para mostrar el resultado.

Un ejemplo del procedimiento es el siguiente:

**1,a,b,c,delta,teta** esto quiere decir que la idea se arma con el sustantivo 1 de la categoría de entrada, primero, segundo y tercero de la categoría de proceso y el componente nombrado delta y el componente nombrado teta de la categoría de salida.

Con este formato, procedemos a armar oraciones que representen una idea para solucionar nuestro problema.

Continuamos seleccionado las ideas postuladas por cada uno, las unimos para mejorarlas y sacamos las ideas que serán postuladas para el siguiente proceso el cual es analizarlos.

**Fase 6. En este paso procedemos a** definir los criterios que nos permitirán evaluar las alternativas de las soluciones anteriormente propuestas y con base en este resultado elegir la solución que mejor satisface las necesidades del problema planteado. Los criterios que escogimos en este caso son los que enumeramos a continuación. Al lado de cada uno se ha establecido un valor numérico con el objetivo de establecer un peso que indique cuáles de los valores posibles de cada criterio tienen más

**Criterio 1: En el criterio 1 evaluaremos la precisión de la solución que se va a escoger, definiéndola en una escala numérica.**

**A)** La solución es Exacta—Valor: 2

B) La solución es Aproximada –Valor: 1

**Criterio 2: En el criterio 2 evaluaremos la facilidad que tendrá el usuario al momento de ingresar los datos.**

A) Fácil ----) Valor: 3

B) Media ---) Valor: 2

C) Algo Complicada ---) Valor:1

**Criterio 4: En el criterio 4 evaluaremos el tiempo utilizado por cada algoritmo.**

1. Constante --) Valor: 1
2. Logarítmica –) Valor: 6
3. Lineal --) Valor: 5
4. Cuadrática --) Valor: 4
5. Cubica --) Valor: 3
6. Exponencial --) Valor: 2

**Criterio 5: En el criterio 5 evaluaremos el espacio en memoria utilizado por cada algoritmo.**

1. Constante --) Valor: 1
2. Logarítmica –) Valor: 6
3. Lineal --) Valor: 5
4. Cuadrática --) Valor: 4
5. Cubica --) Valor: 3
6. Exponencial --) Valor: 2

**Criterio 6: En el criterio 6 evaluaremos la facilidad con la que se muestran los datos en el programa.**

A) Fácil ----) Valor: 3

B) Media ---) Valor: 2

C) Algo Complicada ---) Valor:1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Alternativas** | **Criterio 1** | **Criterio 2** | **Criterio 3** | **Criterio 4** | **Criterio 5** | **Criterio 6** |
|  | **Precisión de la solución** | **Facilidad de ingreso de datos** | **Facilidad de implementación del mundo** | **Tiempo Utilizado por cada Algoritmo** | **Espacio de Memoria Utilizado por cada Algoritmo** | **Facilidad de implementación del algoritmo** |
| **Alternativa 1** | **Medio: 2** | **Cuadro Emergente: 3**  **Archivos de texto Plano: 2** | **Medio: 2** | **Pilas** **O(n): 5**  **Colas O(n): 5**  **Tablas Hash: O(1): 1** | **Pilas O(n) : 5**  **Colas** **O(1):** 1  **Tablas Hash: O(c): 1** | **Pilas 2**  **Colas: 3**  **33**  **Tablas Hash: 1** |
| **Alternativa 2** | **Alto: 3** | **Campo de texto: 3** | **Medio 2** | **Arbol Binario:** **O(log n): 6**  **Colas O(n): 5**  **Tablas Hash: O(1): 1** | **ArbolBinario: 3**  **Colas** O(1): 1  **Tablas Hash: O(c): 1** | **ArbolBinario: 3**  **Colas: 3**  **Tablas Hash: 1 32** |
| **Alternativa 3** | **Media: 2** | **Campo Emergente: 3** | **Medio : 2** | **Pilas** **O(n): 5**  **Monticulos O(nlog n). 5**  **Tablas Hash: O(1): 1** | **Pilas O(n) : 5**  **Monticulos:O( n): 1**  **Tablas Hash: O(c): 1** | **Pilas 2**  **Monticulos: 3**  **Tablas Hash: 1**  **31** |
| **Alternativa 4** | **Media : 2** | **Archivo de texto plano: 2** | **Medio: 2** | **Pilas** **O(n): 5**  **Colas O(n): 5**  **Tablas Hash: O(1): 1** | **Pilas O(n) : 5**  **Colas** O(1): 1  **Tablas Hash: O(c): 1** | **Pilas 2**  **Colas: 3**  **30**  **Tablas Hash: 1** |

**Fase 7. Evaluación y selección de la mejor solución**

**Alternativa1:**

Categoria1(2)+Categoria2(3+2)+Categoria3(2)+Categroria4(5+5+1)+Categoria5(5+1+1)+Categoria6(2+3+1)+=33

**Alternativa 2:**

Categoria1(3)+Categoría 2(3)+Categoria3(2)+Categoría 4(6+5+1)+ Categoria5(3+1+1)+Categoría 6(3+3+1)+=32

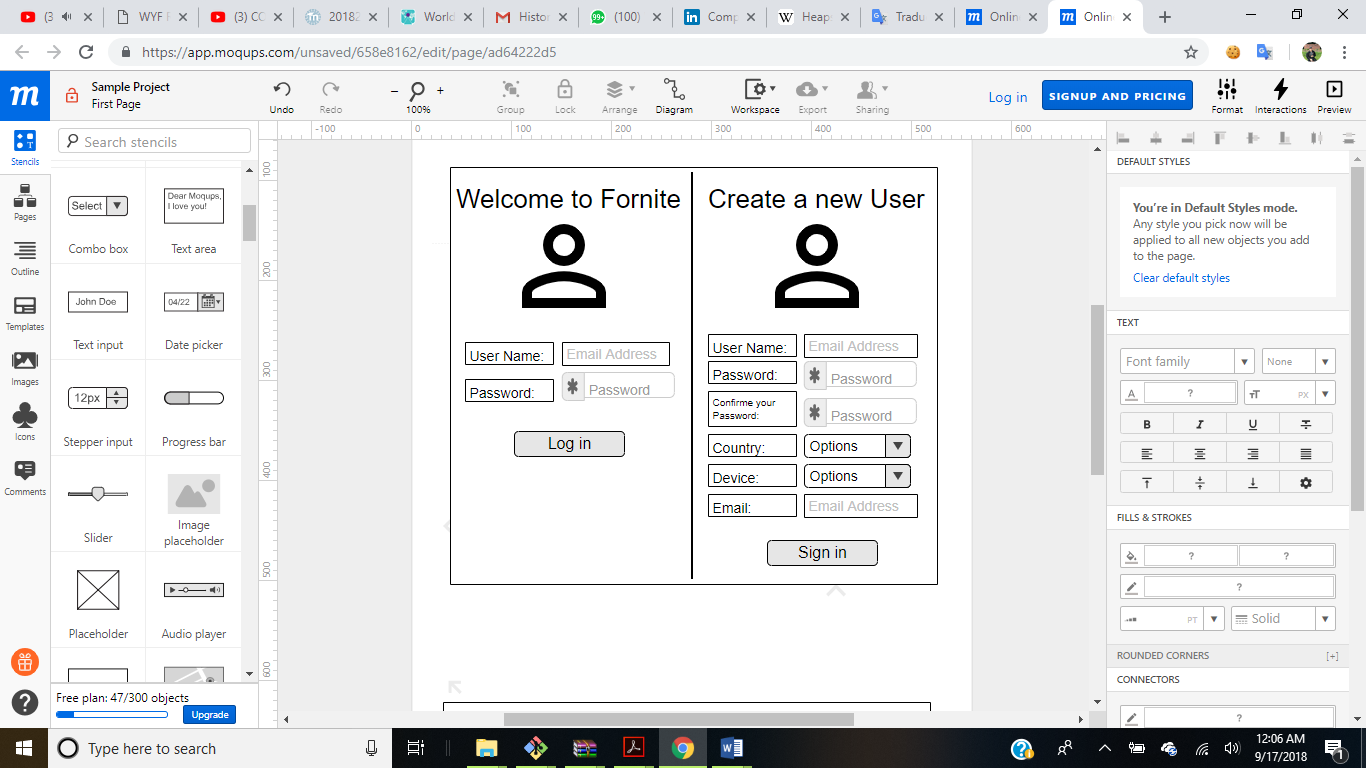
**Alternativa 3:**

Categoria1(2)+Categoría 2(3)+Categoria3(2)+Categoría 4(5+5+1)+ Categoria5(5+1+1)+Categoría 6(2+3+1)+=31

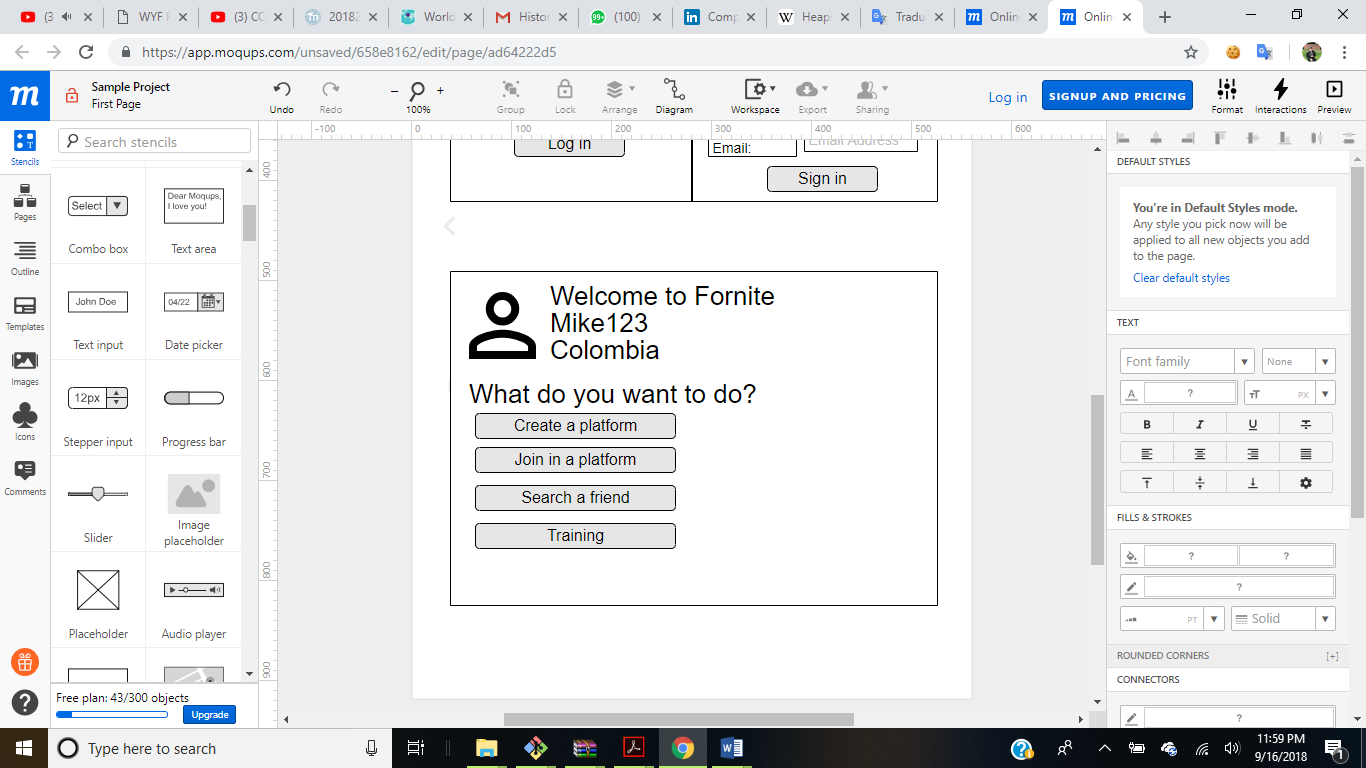
En esta fase lo que procedimos a hacer fue evaluar la mejor solución, teniendo en cuenda la tabla de criterios anteriormente definida y descartar las ideas que no alcanzaran el puntaje promedio para pasar. Al final de la evaluación encontramos que las mejores soluciones son la Alternativa 1, 2 y 3, descartando así las Alternativas 4. Con base en el puntaje obtenido lograremos escoger la mejor solución entre las tres restantes y así poder implementar con todos sus requerimientos correspondientes.

**Fase 8. Diseño preliminar de cada idea no descartada: (Modelos de Simulación)**

**Alternativa 1:**



En esta solución encontramos que el usuario tendrá dos opciones para ingresar en el juego, la primera es ingresar al sistema teniendo ya un usuario que este dentro del programa. Para esto, el usuario tendrá que ingresar su Username y su contraseña para poder ingresar al juego. Por otra parte, el usuario también tendrá la oportunidad de crear un usuario nuevo, en el cual tendrá que ingresar sus datos personales, en este caso el nombre de usuario que le gustaría ingresar, la contraseña, el país desde el cual se conecta, el dispositivo por el cual se esta conectado y el correo electrónico.



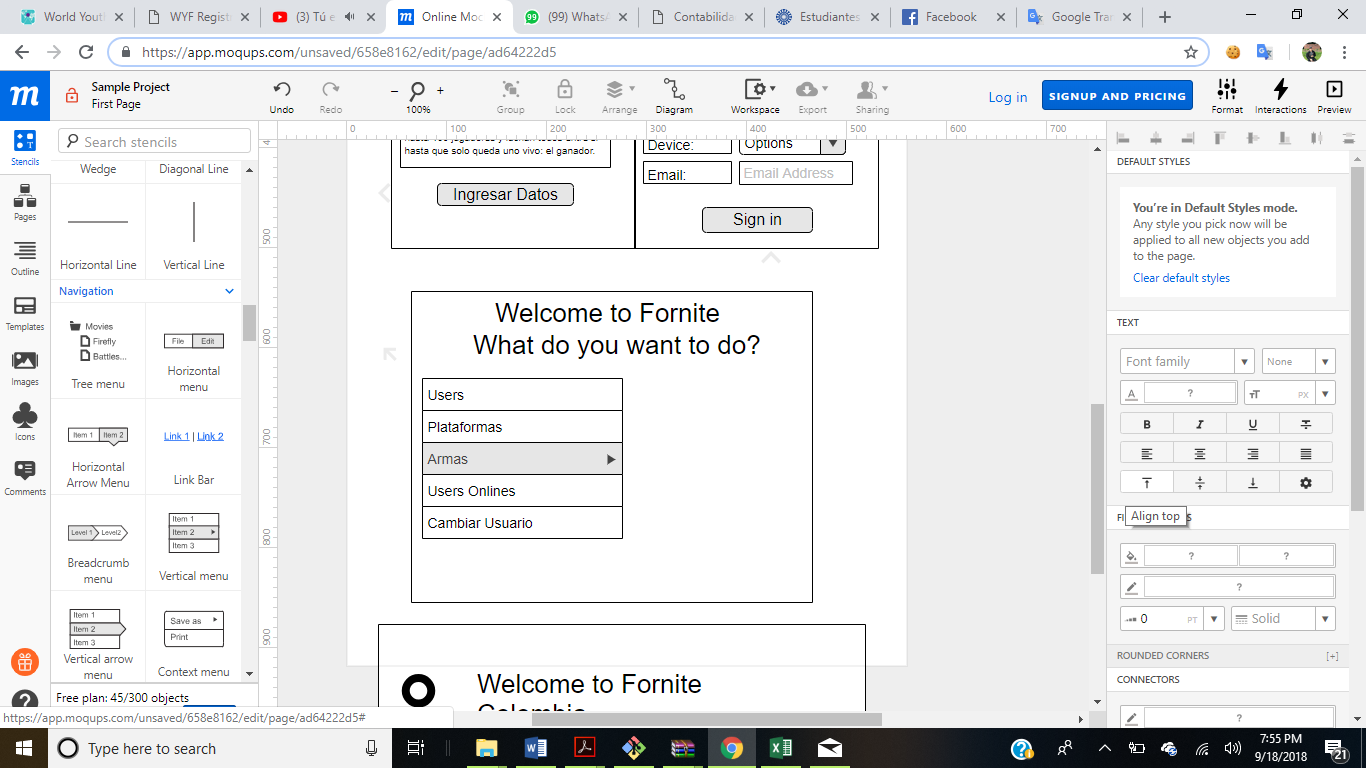
Una vez el usuario ingrese al juego, este le preguntara que le gustaría hacer, ya sea conectarse a una plataforma en donde podrá jugar con otros usuarios, por medio de su ranking de juego, crear una plataforma en done pondrá invitar a sus amigos, podrá buscar un amigo para saber si está conectado o podrá tomar un training, el cual consiste en prepararse antes de ingresar a la plataforma.

**Alternativa 2:**

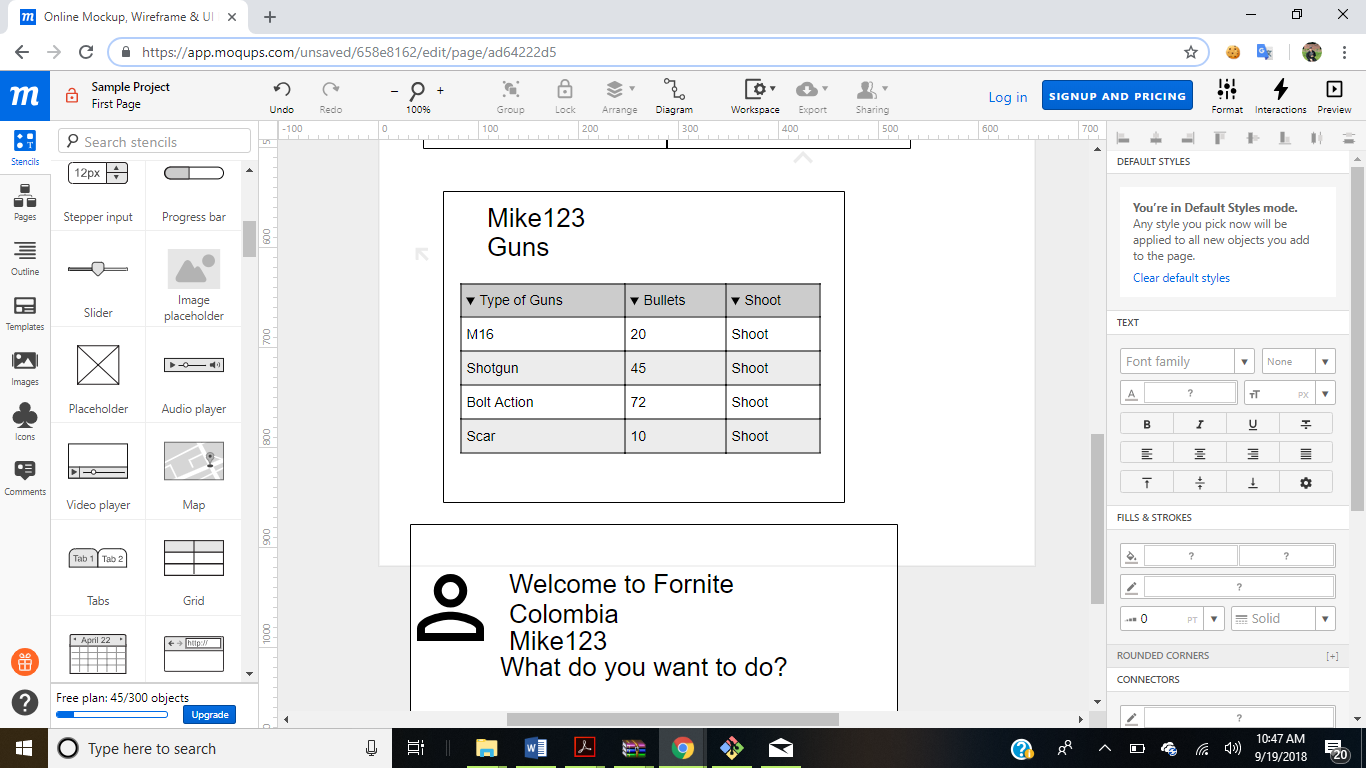
En esta alternativa el usuario tendrá una opción en la cual el sistema le mostrará una manera de ingresar los datos por archivos planos. En esta interfaz el usuario podrá ver una descripción del juego Fornite por medio de un campo de texto. Una vez el usuario ingrese los datos por medio del archivo plano, el sistema le mostrará otra opción en la cual este habrá ingresado al sistema de una manera exitosa, pero esto dependerá de la lectura del archivo. Una vez el usuario haya ingresado de una manera exitosa, en el sistema aparecerá otra ventana en la cual este tendrá diferentes opciones.



Por otra parte, el usuario también contara con la opción de crear un nuevo usuario. Para ello, el usuario tendrá que escoger la opción de Sign in y llenar los datos requeridos para poder ingresar un nuevo usuario al sistema. Los campos requeridos para esto es el Nombre del usuario, contraseña, el país, el dispositivo por el cual se esta conectando y el correo de la persona. Estos son los campos requeridos para poder ingresar un nuevo usuario al sistema, una vez creado el usuario este tendrá la oportunidad de encontrar varias opciones dentro del sistema.



Una vez el usuario este dentro del sistema este le mostrara varias opciones tales como; cantidad de usuarios que hay en ese momento jugando, el número de plataformas que hay disponibles hasta el momento, con su nombre respectivo y el número de balas que tiene hasta el momento cada una de ellas, cuantas armas tiene el usuario actualmente, y la opción de cambiar de usuario. Dependiendo de la opción que este escoja el sistema le mostrara diferentes pestanas del juego.



Una vez el usuario escoja la opción de armas, el sistema le mostrara un panel en el que este tendrá la visibilidad de las armas y tendrá la oportunidad de disparar con el arma que el quiera. Una vez las balas se le acaban en un arma, esta desaparecerá y se pondrá en la lista la ultima arma antes de la que había escogido y así sucesivamente hasta que quede solamente el hacha en la lista de armas, la cual es el arma por defecto.

**Alternative 3:**

**A screenshot of a social media post

Description generated with very high confidence**En esta alternativa nos encontramos con que el usuario tendrá la posibilidad de agregar un nuevo Usuario a la plataforma. La interfaz estará dividida en dos partes, una parte en la cual se irán agregando todos los usuarios disponibles en ese momento en el sistema y en la otra parte el usuario tendrá tres opciones, una la cual es la de agregar un nuevo usuario, la otra es la de eliminar a uno y al final tendremos la opción de agregar a un usuario a una plataforma dada dependiendo del ping de cada uno y la habilidad de este. El sistema al final ira mostrando por medio de un JList todos los usuarios en la plataforma, la habilidad de cada uno, el dispositivo que esta usando para conectarse a la plataforma.

**A screenshot of a cell phone

Description generated with very high confidence**Por otra parte, el sistema le mostrara al usuario las opciones para que este se pueda registrar como usuario nuevo en el juego. Una vez el usuario decida agregarlo tendrá que llenar los campos requeridos para esto y son el nombre, el tipo de habilidad que este posee, el ping que tiene el usuario, la localización en la cual se esta conectado al sistema. Una vez todos los campos estén llenos, el usuario procederá a agregar el nuevo ítem al sistema. SI los campos no están llenos correctamente, el programa le mostrara al usuario una alerta en donde le hace saber que algo anda mal en la información suministrada, y que vuelva a examinarla.

A screenshot of a cell phone

Description generated with very high confidence

Por otra parte, el programa contara con la opción de mostrar los usuarios actualmente en el sistema con sus características respectivas. En este caso hablamos del ping, el nombre del jugador, la plataforma en al cual estará conectado, de que país es y de que dispositivo se estará conectando la persona que va a ingresar al juego.

**Adaptado de :**

**https://www.geeksforgeeks.org/radix-sort/**

**https://stackoverflow.com/questions/42303108/how-can-i-use-radix-sort-for-an-array-of-float-numbers**

[**https://stackoverflow.com/questions/14619653/how-to-convert-a-float-into-a-byte-array-and-vice-versa/14619742**](https://stackoverflow.com/questions/14619653/how-to-convert-a-float-into-a-byte-array-and-vice-versa/14619742)

[**https://www.theguardian.com/games/2018/mar/07/fortnite-battle-royale-parents-guide-video-game-multiplayer-shooter**](https://www.theguardian.com/games/2018/mar/07/fortnite-battle-royale-parents-guide-video-game-multiplayer-shooter)

[**https://www.slideshare.net/diwal10/pilas-colas-y-listas-estructura-de-datosg**](https://www.slideshare.net/diwal10/pilas-colas-y-listas-estructura-de-datosg)

**….**