# Método de la Ingeniería

**Aplicación para la Solución de un Problema**

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

**Fase 5. Análisis de cada idea propuesta, resaltando los Pros y Contra de cada solución.**

**Ventajas Y Desventajas:**

**Alternativa 1:**

Esta alternativa tiene una ventaja principal que tanto muestra los datos principales en el que la aplicación funciona y expone la funcionalidad necesaria para procesar los datos a través de interfaz de usuario, y dependiendo de la opcion que el usuario ingrese, se habilitaran mecanismos como barras de menús, barras de herramientas, etc. Esta alternativa a la hora de ingresar los datos es muy buena ya que por medio de los cuadros emergentes mostrara información específica a los usuarios, recopilar a información de los usuarios, y también puede mostrar y recopilar información. Una desventaja de usar un cuadro emergente es que la información del usuario en el juego no será mostrada correctamente. Por otra parte, una ventaja de usar el texto plano como mecanismo de ingreso de datos es que el usuario no tendrá que invertir tiempo a la hora de poner datos del usuario que desea ingresar, ya que estos estarán previamente listos en el archivo te texto plano. Una desventaja del archivo de texto plano es que tomara un tiempo estipulado a la hora de leer el archivo. Puede haber errores de lectura que el usuario por no tener conocimiento de cómo deberían de ser ingresados los datos no pueda avanzar.

Una vez el usuario ingrese los datos este será ingresado al sistema en donde tendrá la opción de ingresar a una plataforma, eliminar un usuario existente o mirar las armas que este posee. Para esto se utilizaron pilas y colas. Una de las desventajas que podemos encontrar en algunos de esos algoritmos es la implementación en la fase del mundo, ya que pueden llegar a ser un poco complejos de procesar en el sistema. A la hora de la salida los elementos serán mostrador por medio de un JList, en este caso estamos hablando de los usuarios existentes en la plataforma.

**Alternativa 2:**

En esta alternativa podemos encontrar que el usuario ingresara los datos o elementos por medio de un cuadro de texto, esto le facilitara al usuario en lo relacionado con el tiempo de ingreso. Por otra parte, cuando hablamos de los métodos que vamos a utilizar, tanto los montículos, como las Tablash hash tienen una facilidad de implementación en comparación con otros algoritmos, pero una o varias desventajas que podríamos encontrar con este, es a la hora de entradas grandes, en este caso estamos hablando de las tablas hash, ya que esto permitiría que hubieran colisiones dentro de esta, el margen de error de que esto suceda es grande, por ende esto podría volverse algo tedioso.

Una de las ventajas de esta alternativa a la hora de mostrar los usuarios o las plataformas es que esta los estará mostrará por medio de un cuadro de texto y no se vería muy bien todos los elementos.

**Alternativa 3:**

En esta alternativa podemos encontrar que el usuario ingresara los datos por medio de campos emergentes. Una de las ventajas que podríamos encontrar a la hora de implementar esta solución es que el usuario podrá ingresar los datos de una manera concisa y sin gasto de tiempo. También, por medio del campo de texto se le hará saber al usuario como ingresar los datos de una manera correcta reduciendo así el nivel de error. Una vez los elementos sean ingresados estos harán el llamado a los métodos que se encargaran de crear ya se la pila, cola, o montículos. Estos métodos tienen una ventaja ya que estos son muy eficientes en cuestión de tiempo y retornan una respuesta eficaz, pero a la hora de la implementación se vuelve un poco complicado de tratar. Por otra parte, también tenemos la opción de crear un árbol binario con los elementos ingresados por el usuario, la ventaja de esta opción es que ya nos estaría organizando los elementos en la lista, y esto nos facilitaría a la hora de buscar los elementos en ella. Una ventaja de utilizar un árbol es la facilidad de implementación por el medio recursivo.

Finalmente, los datos ordenados serán exportados como un archivo plano. Una de las ventajas de exportar los elementos como un archivo plano es que el usuario tendrá facilidad de consultar los datos ingresados, los usuarios que están en ese momento en el sistema, el número de plataformas que hay, el numero de armas que este tiene, y el numero de balas que tiene cada arma.

**Alternativa 4:**

**Evita que el usuario ingrese datos de forma manual:**

Ya que el usuario ingresará los datos por medio de un archivo plano, se beneficiará de la función de generación de datos aleatorios que ofrecerá el programa. Al mismo tiempo, el software no tendrá que esperar entradas inesperadas que el usuario quiera ingresar.

**Facilita mostrar datos pequeños:**

Al implementar una JList, los datos pequeños serán muy fáciles de visualizar. El usuario podrá arrastrar el scroll y ver la lista de datos ordenados, en este caso las armas, las balas, los usuarios y las plataformas.

**Contras:**

**Implementa algoritmos con coliciones:**

Tablas Hash algoritmos complicados a la hora de ingresar demasiados elementos y aumentar el índice de colisiones en estos.

**Vista para entradas grandes difíciles de mostrar:**

La JList puede que no soporte entradas grandes de datos y colapse. En caso de que soporte la entrada, el usuario no tendrá claridad en la lista que está viendo pues será una lista vertical inmensa.

**Definicion de los Tads**

* **TAD para Pilas**

|  |
| --- |
| **Nombre: Stack (pila)** |
| **Dibujo:**  **Sin título** |
| **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:**  **images** |
| **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} |

**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:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **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

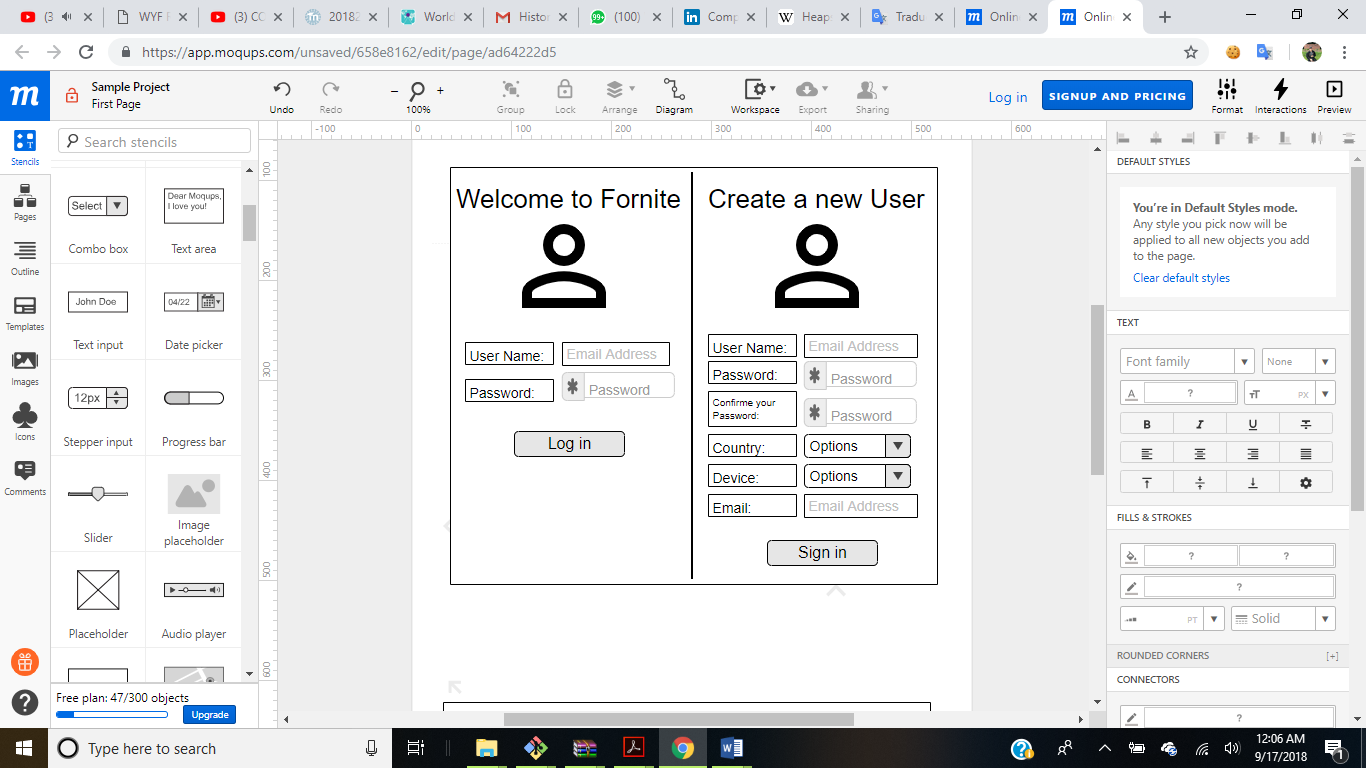
**Alternativa 3:**

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

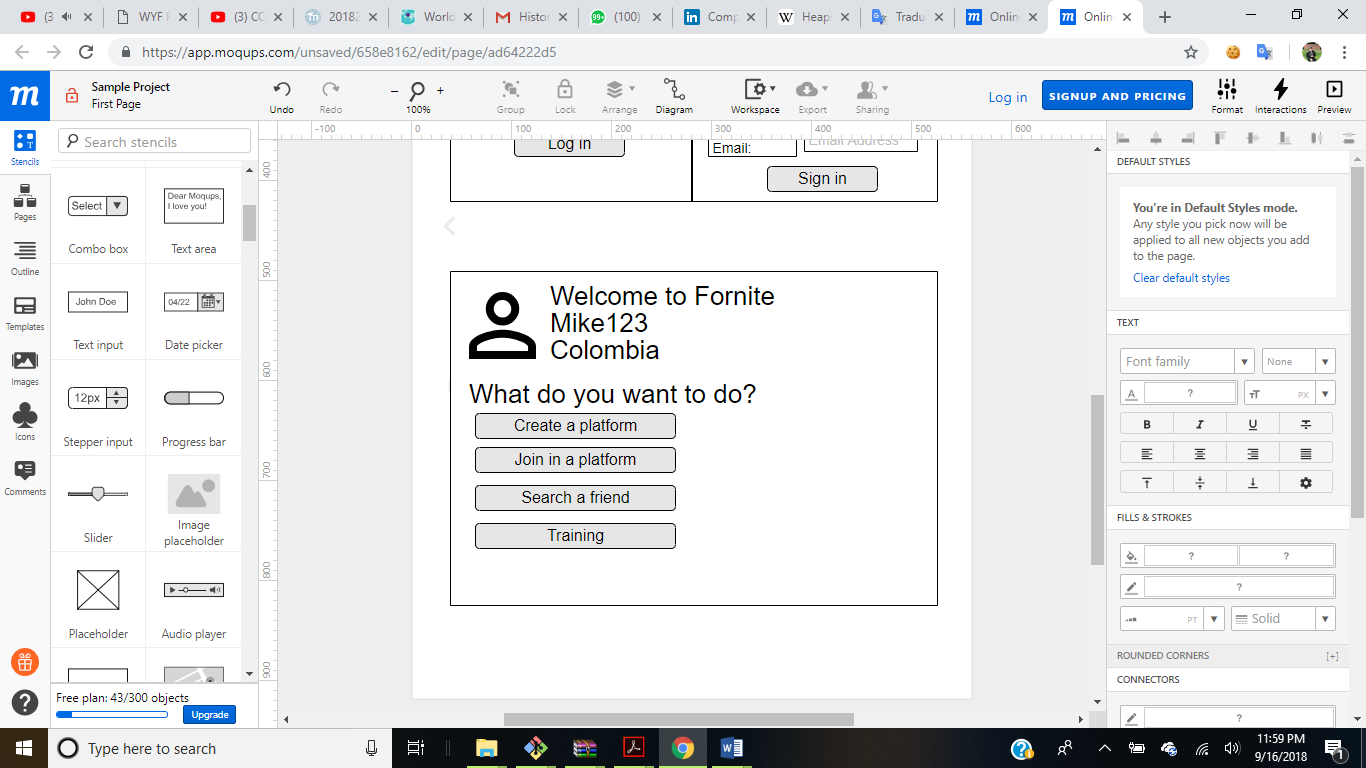
En esta fase lo que procedimos a hacer fue evaluar la mejor solución, teniendo en cuenda la tabla de criterios anteriormente definida y descartar las ideas que no alcanzaran el puntaje promedio para pasar. Al final de la evaluación encontramos que las mejores soluciones son la Alternativa 1, 2 y 3, descartando 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.



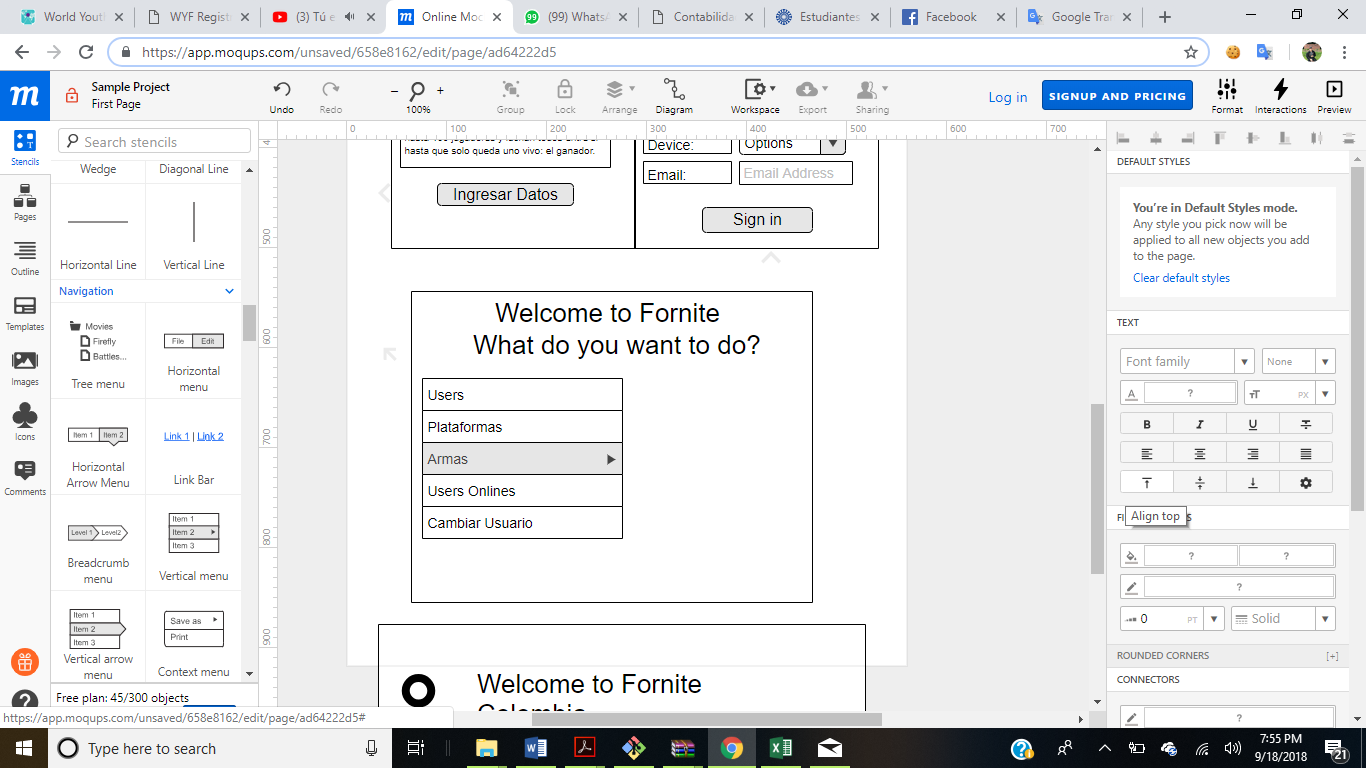
Una vez el usuario ingrese al juego, este le preguntara que le gustaría hacer, ya sea conectarse a una plataforma en donde podrá jugar con otros usuarios, por medio de su ranking de juego, crear una plataforma en done pondrá invitar a sus amigos, podrá buscar un amigo para saber si está conectado o podrá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

**Alternativa 2:**

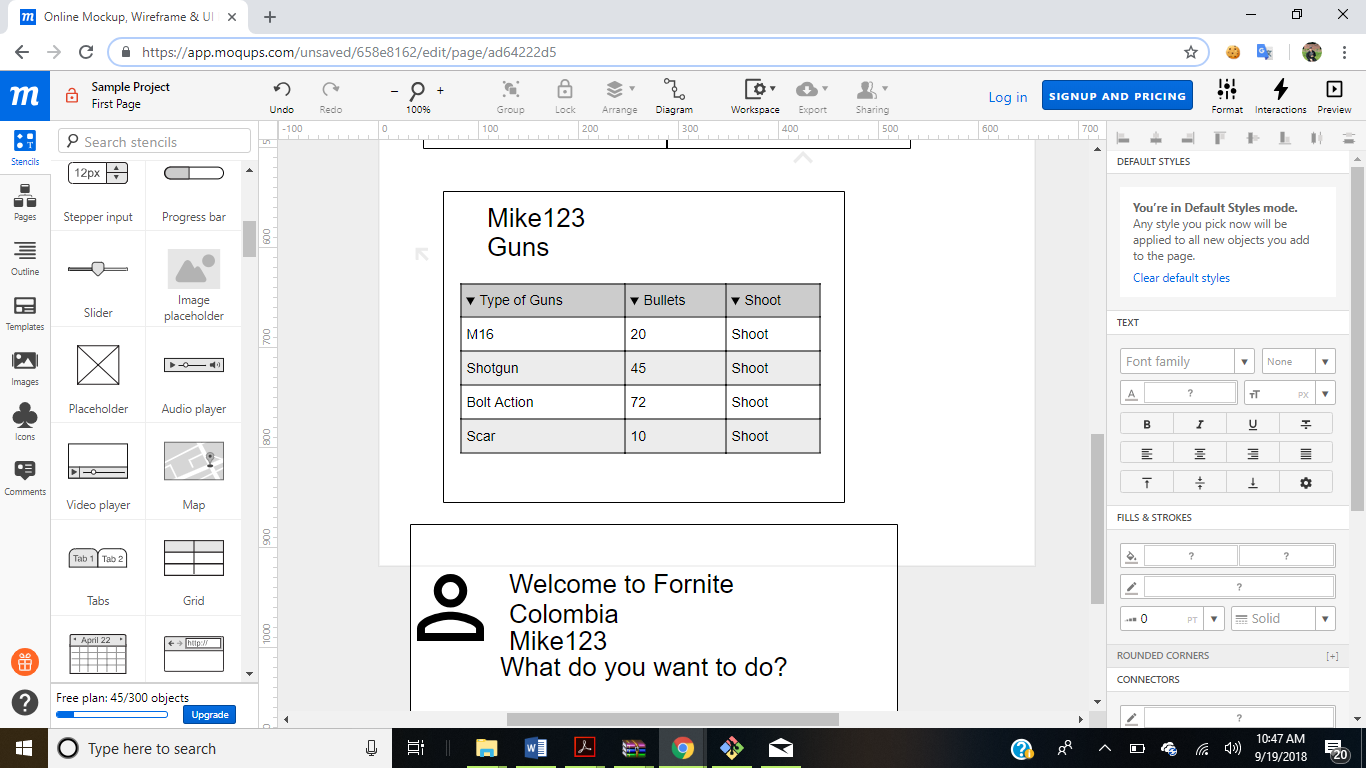
En esta alternativa el usuario tendrá una opción en la cual el sistema le mostrará una manera de ingresar los datos por archivos planos. En esta interfaz el usuario podrá ver una descripción del juego Fornite por medio de un campo de texto. Una vez el usuario ingrese los datos por medio del archivo plano, 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 está conectando 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

**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)

**….**