|  |  |
| --- | --- |
| C:\Users\Esmeralda\Documents\Tecnologias para la construccion\5 en linea\logo4.png  Documento de Especificación de requerimientos | Mariana Cadena Romero  Esmeralda Jimenez Ramos  **Tecnologías para la construcción de software** |

Tabla de contenido

[1. Introducción 2](#_Toc503091398)

[1.1. Propósito 2](#_Toc503091399)

[1.2. Ámbito del sistema 2](#_Toc503091400)

[1.3. Definiciones, acrónimos y abreviaturas 2](#_Toc503091401)

[2. Descripción general 3](#_Toc503091402)

[2.1. Perspectiva del producto 3](#_Toc503091403)

[2.2. Funciones del producto 3](#_Toc503091404)

[2.3. Características de los usuarios 4](#_Toc503091405)

[2.4. Restricciones 4](#_Toc503091406)

[2.5. Suposiciones y dependencias 4](#_Toc503091407)

[3. Requerimientos específicos 4](#_Toc503091408)

[3.1. Modelo de dominio 5](#_Toc503091409)

[3.2. Requisitos funcionales 5](#_Toc503091410)

[3.3. Restricciones de Diseño 20](#_Toc503091411)

[3.4. Atributos del Sistema 20](#_Toc503091412)

[4. Información adicional del proyecto 21](#_Toc503091413)

[4.1. Modelo de datos 21](#_Toc503091414)

[4.2. Diagrama de despliegue 21](#_Toc503091420)

[4.3. Modelo Entidad-Relación 22](#_Toc503091421)

[4.4. Frameworks seleccionados 22](#_Toc503091422)

[4.5. Estándar de codificación 23](#_Toc503091423)

[4.6. Conclusiones y lecciones aprendidas 25](#_Toc503091424)

# Introducción

El análisis de requisitos es una de las tareas más fundamentales en el ciclo de vida del desarrollo de software, puesto que en ella se determinan los “planos” de la nueva aplicación.

Este documento de especificación de requerimientos de software (ERS) del videojuego “5 en línea” hablará del propósito, ámbito del sistema, la perspectiva, funciones del producto, características de los usuarios, restricciones, suposiciones, dependencias, requerimientos futuros y se mostrarán las interfaces, se hará una descripción de los casos de uso y los atributos del Sistema. Esta especificación se ha estructurado basándose en las directrices dadas por el estándar IEEE Práctica Recomendada para Especificaciones de Requerimientos de Software ANSI IEEE 830, 1998.

## Propósito

El presente documento tiene como propósito definir los requerimientos funcionales y no funcionales para el desarrollo de un videojuego el cual contará con soporte para multijugadores y reporte de las mejores puntuaciones.

## Ámbito del sistema

En este documento hablaremos a profundidad del videojuego en línea “5 en línea”, este juego clásico estará a la mano de todas las personas que cuenten con conexión a internet, contará con separación por jugador para que puedan existir partidas multijugador.

## Definiciones, acrónimos y abreviaturas

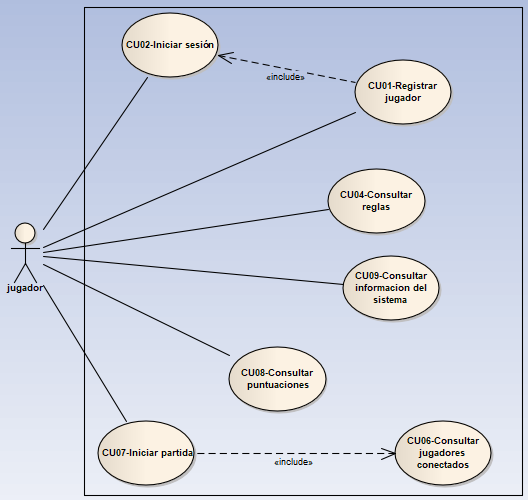
|  |  |
| --- | --- |
| **Nombre** | **Descripción** |
| **Usuario** | Persona que usará el sistema para gestionar procesos |
| **ERS** | Especificación de Requerimientos de Software |
| **RQ** | Requerimiento |
| **CU** | Caso de uso |

# Descripción general

## Perspectiva del producto

El producto a desarrollar se clasifica como videojuego y está diseñado para soportar el modo multijugador en línea. Es un software independiente, no tiene relación con otros programas externos. Provee una interfaz amigable, con un enfoque infantil, pero que cualquier tipo de usuario puede utilizar.

## Funciones del producto



## Características de los usuarios

|  |  |  |
| --- | --- | --- |
| **Tipo** | **Descripción** | **Rol dentro del sistema** |
| General | Usuario sin rango de edad con poca o mucha experiencia en el uso de tecnologías. En cualquiera de los casos, se favorece de interfaces intuitivas y ambientes centrados en el usuario. | Jugador |

## Restricciones

* Sistema para correr en un entorno en línea.
* Requiere una conexión de área local a un servidor propio.
* Lenguajes y tecnologías en uso: MySQL, JAVA FX
* Los servidores deberán ser capaces de atender consultar concurrentemente
* El sistema será un diseño “sencillo” y didáctico que sea de suma interés para un usuario de cualquier edad.

## Suposiciones y dependencias

* Se asume que los requerimientos descritos en este documento son estables, en total acuerdo y no cambiarán.
* Los equipos donde se ejecutará el sistema cumplen con los requerimientos descritos para mantener una buena y efectiva ejecución.
* Se asume que las personas o niños que jugaran el juego, saben cómo “manejarlo”.

# Requerimientos específicos

**RQ01- Registrar jugador**

Un nuevo jugador podrá registrar sus datos

**RQ02 – Iniciar sesión**

El jugador podrá iniciar sesión para ingresar al sistema

**RQ04 – Consultar reglas**

El jugador podrá ver las reglas del juego desde la pantalla principal o en la pantalla de juego

**RQ05 - Consultar jugadores conectados**

El jugador podrá ver a todos los jugadores que estén en línea

**RQ06 – Iniciar partida**

El jugador podrá iniciar una partida nueva

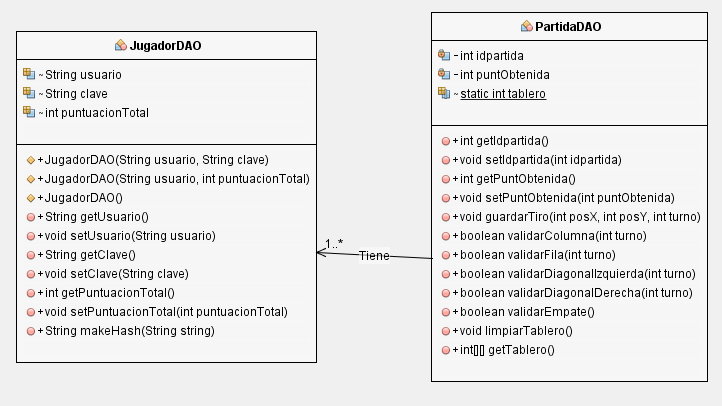
**RQ07 - Consultar puntuaciones**

El jugador podrá ver cuál de los jugadores tiene más partidas ganadas

**RQ08-Consultar información del sistema**

El jugador podrá ver la información de los desarrolladores y de la versión del sistema

## Modelo de dominio



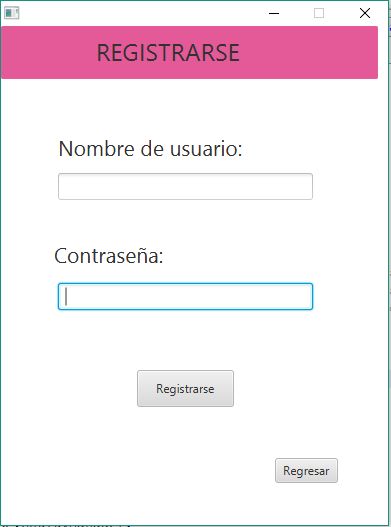
1.. 2

## Requisitos funcionales

A continuación, se detallan las descripciones de los casos de uso, en donde se podrá consultar el identificador único, nombre, descripción, precondiciones a la función, flujo normal, flujos alternos (en caso de existir), excepciones al flujo, Poscondiciones al terminar la función, entradas requeridas por la función, salidas de la función, prioridad y prototipo. Para mayor claridad se podrá consultar el diagrama de casos de uso

**CU01. Registrar jugador**

|  |  |
| --- | --- |
| **ID:** | CU01 |
| **Nombre:** | Registrar jugador |
| **Autor(es):** | Esmeralda Jimenez Ramos |
| **Fecha de creación:** | 25/sep./2017 |
| **Fecha de actualización:** |  |
| **Actor(es):** | Jugador |
| **Descripción:** | Permite a un nuevo jugador registrar sus datos en la base de datos |
| **Precondiciones:** |  |
| **Flujo normal:** | 1. El sistema muestra una pantalla los botones Registrarse y Regresar y cuadros de texto para el nombre de usuario y la contraseña 2. El nuevo jugador rellena los cuadros de texto con la información necesaria y selecciona el botón Registrarse 3. El sistema valida que la contraseña sea mayor a 8 caracteres y que el usuario no exista y guarda la información en la base de datos 4. Incluye el CU02 5. Fin del caso de uso |
| **Flujos alternos:** | 2A. El jugador selecciona Regresar  2A1. Fin del caso de uso  2B. El jugador ingresa una contraseña menos a 8 caracteres o ingresa un usuario existente  2B1. El sistema valida la información y manda un mensaje en pantalla “Información incorrecta” |
| **Excepciones:** | 3. El sistema pierde conexión con la base de datos y lanza el mensaje “No se pudo conectar con la base de datos, intenta más tarde” |
| **Poscondiciones:** | Los datos del nuevo jugador quedan registrados en la base de datos |
| **Include:** | CU02 - Iniciar sesión |
| **Extend:** |  |
| **Prioridad:** | Alta |



**CU02. Iniciar sesión**

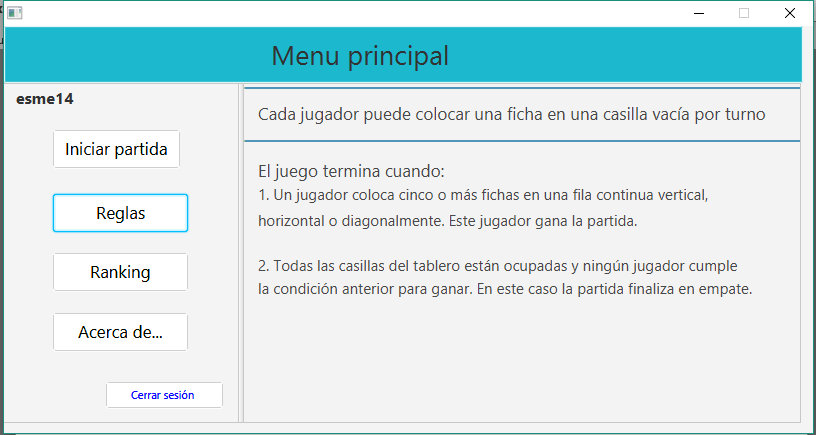
|  |  |
| --- | --- |
| **ID:** | CU02 |
| **Nombre:** | Iniciar sesión |
| **Autor(es):** | Mariana Cadena Romero |
| **Fecha de creación:** | 25/sep./2017 |
| **Fecha de actualización:** |  |
| **Actor(es):** | Jugador |
| **Descripción:** | Permite a un jugador previamente registrado acceder a su cuenta del sistema y ver las opciones disponibles. |
| **Precondiciones:** | El jugador debe haber creado previamente una cuenta en el sistema. |
| **Flujo normal:** | 1. El sistema muestra una pantalla cuyo título es “Iniciar sesión” que contiene un cuadro de texto con la etiqueta “Nombre de usuario” y un cuadro de clave con la etiqueta “Contraseña”, así como los botones “Ingresar” y “Registrarse”. 2. El jugador ingresa su usuario y contraseña en los campos correspondientes y da clic en el botón “acceder”. 3. El sistema consulta en la base de datos si existe el usuario introducido por el jugador y verifica que la contraseña corresponda al mismo. Luego muestra la ventana “Menú principal” 4. Termina el caso de uso |
| **Flujos alternos:** | 3   1. El sistema no encuentra el usuario en la base de datos, entonces muestra el mensaje “Usuario inválido” 2. Termina el caso de uso   3   1. El sistema detecta que la contraseña no corresponde con el usuario, entonces muestra el mensaje “Contraseña incorrecta”. 2. Termina el caso de uso   3E) El jugador da clic en el botón “Salir”   1. Termina el caso de uso |
| **Excepciones:** | **3**   1. El sistema pierde la conexión con la base de datos, por lo tanto, genera un cuadro de diálogo con el mensaje “Se ha perdido la conexión, intente más tarde” y el botón “aceptar”. 2. El jugador da clic en el botón aceptar 3. Termina el caso de uso |
| **Poscondiciones:** | Ninguna, el estado del sistema no cambia |
| **Include:** | Ninguno |
| **Extend:** |  |
| **Prioridad:** | Alta |



**CU04. Consultar reglas**

|  |  |
| --- | --- |
| **ID:** | CU04 |
| **Nombre:** | Consultar reglas |
| **Autor(es):** | Mariana Cadena Romero |
| **Fecha de creación:** | 25/sep./2017 |
| **Fecha de actualización:** |  |
| **Actor(es):** | Jugador |
| **Descripción:** | Permite al jugador ver las reglas del juego. |
| **Precondiciones:** | 1. El jugador debe haber ingresado al sistema. 2. El jugador debe haber dado clic en el botón “Reglas” del menú principal. |
| **Flujo normal:** | 1. El sistema muestra en pantalla la ventana “Menú principal” donde muestra varios botones “Iniciar partida”, “Reglas”, “Ranking”, “Acerca de…” junto al logo del juego. 2. El jugador da clic en el botón “Reglas”. 3. El sistema despliega las “Reglas” en donde se describen las reglas del juego. 4. El jugador consulta las reglas. 5. El caso de uso finaliza. |
| **Flujos alternos:** | 4   1. El jugador da clic en “Regresar” 2. El sistema retorna al menú principal. 3. El caso de uso finaliza. |
| **Excepciones:** |  |
| **Poscondiciones:** | Ninguna, el estado del sistema no cambia |
| **Include:** | Ninguno |
| **Extend:** |  |
| **Prioridad:** | Media |

Pantalla 1 Menú principal



**CU06. Consultar jugadores conectados**

|  |  |
| --- | --- |
| **ID:** | CU05 |
| **Nombre:** | Consultar jugadores conectados |
| **Autor(es):** | Mariana Cadena Romero |
| **Fecha de creación:** | 25/sep./2017 |
| **Fecha de actualización:** |  |
| **Actor(es):** | Jugador |
| **Descripción:** | Permite al jugador visualizar los jugadores conectados y así iniciar una nueva partida. |
| **Precondiciones:** | 1. El jugador debe haber ingresado al sistema. 2. El jugador debe haber dado clic en el botón “Iniciar partida” del menú principal. |
| **Flujo normal:** | 1. El sistema muestra en pantalla la ventana “Menú principal” donde muestra varios botones “Iniciar partida”, “Reglas”, “Ranking”, “Acerca de…”, junto al logo del juego. 2. El jugador da clic en el botón “Iniciar partida”. 3. El sistema muestra en pantalla la ventana “Jugadores conectados” en donde se muestran un cuadro de texto para ingresar el usuario del jugador con quien se quiere iniciar una partida y los botones para “Iniciar partida” y el botón “Regresar” 4. El jugador escribe el nombre de usuario con quien desea jugar y da clic en el botón “Iniciar partida” 5. Al jugador invitado a jugar se le muestra en pantalla un mensaje “El jugador (usuario que invita) quiere iniciar una partida contigo” con dos botones “Aceptar” y “Cancelar” 6. El jugador invitado da clic en “Aceptar” e Incluye el CU07. 7. El caso de uso finaliza. |
| **Flujos alternos:** | 9   1. El jugador da clic en “Regresar” 2. El sistema retorna al menú principal. 3. El caso de uso finaliza.   10   1. El jugador da clic en “Aceptar” y el sistema retorna a la ventana “Consultar jugadores conectados”   3.1 Si el jugador da clic en “Regresar al menú principal” el sistema retorna a la ventana del menú principal.  3.2 El caso de uso finaliza. |
| **Excepciones:** |  |
| **Poscondiciones:** | Inicia la partida con el jugador seleccionado. |
| **Include:** | CU07-Iniciar partida |
| **Extend:** |  |
| **Prioridad:** | Media |

\*Pantalla 1 Menú principal



\*Pantalla 2 Consultar jugadores conectados

**CU07. Iniciar partida**

|  |  |
| --- | --- |
| **ID:** | CU06 |
| **Nombre:** | Iniciar partida |
| **Autor(es):** | Mariana Cadena Romero |
| **Fecha de creación:** | 25/sep./2017 |
| **Fecha de actualización:** |  |
| **Actor(es):** | Jugador |
| **Descripción:** | Permite al jugador iniciar una partida nueva. |
| **Precondiciones:** | 1. El jugador debe haber ingresado al sistema. 2. El jugador debe haber enviado la invitación para iniciar partida 3. El jugador invitado debe haber aceptado la solicitud. |
| **Flujo normal:** | 1. El sistema muestra en pantalla la ventana de la partida la cual muestra el tablero el turno del jugador al que le toca tirar. 2. Los jugadores realizan los tiros necesarios. 3. Cuando uno de los dos jugadores llega a 5 en línea el sistema muestra en pantalla un mensaje “Ganaste” con el botón “Aceptar” en la pantalla del jugador victorioso en caso de la pantalla del perdedor muestra el mensaje “Felicidades, perdiste” con el botón “Aceptar”. 4. Los jugadores dan clic en Aceptar. 5. El caso de uso finaliza. |
| **Flujos alternos:** |  |
| **Excepciones:** |  |
| **Poscondiciones:** | Se registra en la base de datos los puntajes, 10 puntos para el ganador, 0 para el perdedor. |
| **Include:** |  |
| **Extend:** |  |
| **Prioridad:** | Media |



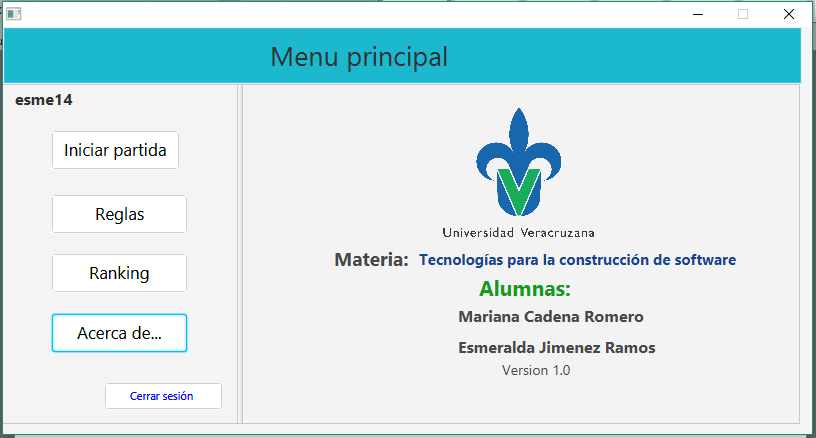
**CU08. Consultar puntuaciones**

|  |  |
| --- | --- |
| **ID:** | CU07 |
| **Nombre:** | Consultar puntuaciones |
| **Autor(es):** | Esmeralda Jimenez Ramos |
| **Fecha de creación:** | 25/sep./2017 |
| **Fecha de actualización:** |  |
| **Actor(es):** | Jugador |
| **Descripción:** | Le permite al jugador ver la lista de jugadores con el puntaje más alto |
| **Precondiciones:** | El jugador debe estar registrado en la base de datos  El jugador debe iniciar sesión |
| **Flujo normal:** | 1. El sistema muestra la pantalla principal con los botones iniciar partida, ranking, reglas y acerca de… así mismo las opciones cerrar sesión y eliminar cuenta 2. El jugador selecciona la opción de ranking 3. El sistema despliega en la pantalla una tabla con los 5 jugadores que tiene el puntaje más alto de mayor a menor 4. El jugador consulta la lista 5. Fin del caso de uso |
| **Flujos alternos:** | 1A. El jugador selecciona cualquier otra opción de la pantalla |
| **Excepciones:** | **3. El sistema pierde la conexión con la base de datos y muestra el mensaje en pantalla “Se perdió la conexión con la base de datos, intente más tarde”** |
| **Poscondiciones:** | El jugador consulta la lista de puntajes |
| **Include:** |  |
| **Extend:** |  |
| **Prioridad:** | Alta |



**CU09. Consultar información del sistema**

|  |  |
| --- | --- |
| **ID:** | CU08 |
| **Nombre:** | Consultar información del sistema |
| **Autor(es):** | Esmeralda Jimenez Ramos |
| **Fecha de creación:** | 25/sep./2017 |
| **Fecha de actualización:** |  |
| **Actor(es):** | Jugador |
| **Descripción:** | Permite al jugador ver información sobre los desarrolladores y la actual versión |
| **Precondiciones:** | El jugador debe estar registrado en la base de datos  El jugador debe iniciar sesión |
| **Flujo normal:** | 1. El sistema muestra la pantalla principal con los botones iniciar partida, ranking, reglas y acerca de… así mismo las opciones cerrar sesión y eliminar cuenta 2. El jugador selecciona la opción acerca de... 3. El sistema despliega la información del sistema 4. Fin del caso de uso |
| **Flujos alternos:** | 1A. El jugador selecciona alguna otra opción de la pantalla principal |
| **Excepciones:** |  |
| **Poscondiciones:** | El jugador consulta la información del sistema |
| **Include:** |  |
| **Extend:** |  |
| **Prioridad:** | Alta |



## Restricciones de Diseño

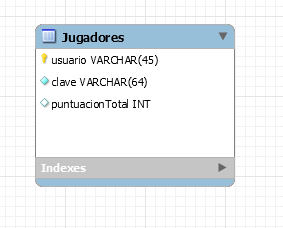
El análisis y diseño de la aplicación se hace bajo los principios del paradigma Orientado a objetos, por lo tanto, lo más conveniente es usar un lenguaje de programación enfocado a su manejo: Java. Entonces, se hará una conexión entre un IDE que utilice JAVA y la base de datos usada: MySQL.

## Atributos del Sistema

* **Desempeño:**
  + Se garantiza que el diseño de las consultas u otro proceso no afecte el desempeño de la base de datos, ni considerablemente el tráfico de la red.
* **Seguridad:**
  + Cada usuario deberá establecer una contraseña con las siguientes características:
    - Longitud mínima de 8 caracteres.
  + Se garantiza la confiabilidad, la seguridad y el desempeño del sistema informático a los diferentes usuarios. En este sentido la información almacenada o registros realizados podrán ser consultados y actualizados permanente y simultáneamente, sin que se afecte el tiempo de respuesta.
* **Rapidez:**
  + El sistema estará conformado de manera que la realización de las actividades se logre de una manera rápida y eficaz
* **Uso:**
  + El sistema contará con interfaces intuitivas para que los usuarios puedan utilizarlo con facilidad.

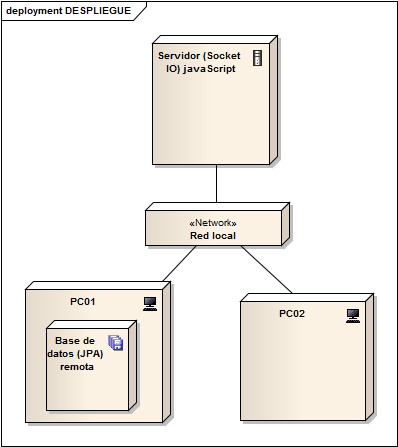
# Información adicional del proyecto

## Modelo de datos

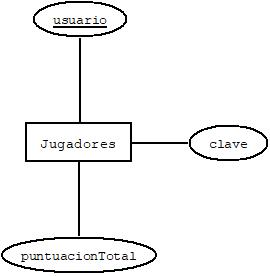




## Diagrama de despliegue



## Modelo Entidad-Relación



## Frameworks seleccionados

**Java Persistence API(JPA):**

JPA es un framework que permite un desarrollo más rápido y fluido ya que se trabaja con la base de datos a través de entidades en lugar de Querys, ofrece un paradigma enteramente orientado a objetos y limita los errores en tiempo de ejecución permitiéndonos ahorrar tiempo y esfuerzo evitando definir de manera manual la capa de conexión y el mapeo de la información de la base de datos a la aplicación.

JPA esta compuesto por:

**Unidad de persistencia**

Define un modelo relacional de objeto que correlaciona las clases Java (entidades + estructuras de soporte) con una base de datos relacional.

**EntityManagerFactory**

Se utiliza para crear un EntityManager para las interacciones con la base de datos, esta puede actualizar, recuperar, eliminar o aplicar la persistencia de objetos de una base de datos (entidades).

**EntityManager**

Gestor de recursos que mantiene la colección activa de objetos de entidad que está utilizando la aplicación. EntityManager maneja la interacción y metadatos de bases de datos para las correlaciones relacionales de objetos. Una instancia de EntityManager representa un contexto de persistencia.

**Objetos de entidad**

Clase Java simple que representa una fila en una tabla de base de datos con su formato más sencillo. Estas contienen “Querys” que permiten realizar las consultas con la base de datos.

**JUnit:**

JUnit es un framework java que permite la realización de la ejecución de clases de manera controlada, esto para poder corroborar que los métodos realizan su función de forma correcta. También sirve como herramienta para realizar las pruebas de regresión, que realizaremos cuando una parte del código ha sido modificada y sea necesario comprobar que se sigue cumpliendo con los requisitos.

**Node.js**

Es un entorno en tiempo de ejecución multiplataforma, de código abierto, que utiliza una entrada y salida de datos orientada a eventos. El uso de este framework nos permitirá comunicar mediante el uso de eventos el cliente de la aplicación y el servidor de Java Script de la aplicación. Este framework permite no preocuparse por los bloqueos de procesos ya que esto no existe.  Casi ninguna función en Node realiza E/S directamente, así que el proceso nunca se bloquea. Debido a que no hay bloqueo es muy razonable desarrollar sistemas escalables en Node.

Estándar de codificación

**Estándar de codificación para los lenguajes de programación Java y JavaScrip**

|  |  |
| --- | --- |
| Propósito | Guiar la implementación en los programas de Java y JavaScript |
| Cabeceras de programa | Comenzar todos los programas con una cabecera descriptiva. |
| Formato de la cabecera | /\*    Nombre del proyecto:  Nombres de los desarrolladores:    Fecha en la que se inició el programa:    Descripción:  \*/ |
| Identificadores | Usar nombres descriptivos para todas las variables, nombres de funciones, constantes y otros identificadores. Evitar abreviaciones o nombres de variables con una sola letra.  Nota: JavaScript es un lenguaje de tipificación dinámica, por lo que o requiere establecer el tipo de dato del identificador. |
| Ejemplo de identificadores | int numeroEstudiantes; /\* Bien \*/  Float: x4, j, ftave; /\* Mal\*/ |
| Comentarios | * Documentar el código para que el lector pueda entender la operación. * Los comentarios deben explicar el propósito y el comportamiento del código. |
| Buen comentario | /\*\*  \* Descripción  \* @author  \* @param  \* @return  \*@exception  \*/ |  |
| Mal comentario | If(record\_count > limit) /\* Determina si el contador de grabaciones excedió el límite\*/  If (record\_count > limit) /\* ¿Han sido procesadas todas las grabaciones? \*/ |  |
| Secciones principales | Preceder las secciones principales del programa con un comentario en bloque que describa el procesamiento que se realiza en la siguiente sección. |  |
| Ejemplo | /\*\*  \* Método que guarda el tiro de cada jugador en una matriz que representa al  \* tablero  \*  \* @param posX Representa el identificador de la posición horizontal en el  \* tablero  \* @param posY Representa el identificador de la posición vertical en el  \* tablero  \* @param turno Nos dice de quien es el tiro que se está guardando  \*/  public void guardarTiro(int posX, int posY, int turno) {  } |  |
| Espacios en blanco | * Escribir programas con suficiente espacio, de esta manera no estarán amontonados. * Separar cada bloque del programa con al menos un espacio. |  |
| Identación | * Cada vez que se abre un nuevo bloque la sangría aumenta por cuatro espacios. Cuando el bloque termina, la sangría vuelve al nivel de sangría anterior.      * Las llaves abiertas deben estar en renglón de la función, ciclo o condición a la que correspondan y las cerradas deben estar en líneas por sí mismas y alineadas con el elemento que cierra. |  |
| Ejemplo de identación | while (missDistance > threshold) {  successCode = moveRobot (target Location);  if (successCode == MOVE\_FAILED) {  System.Out.PrintLn(“The robot move has failed.\n”);  }  } |  |
| Capitalización | * Lower Camel Case para todos los identificadores y letras minúsculas para las palabras reservadas * Los mensajes para el usuario pueden usar mayúsculas y minúsculas |  |
| Ejemplo de capitalización | String correoElectronico;  var correoElectronico;  System.Out.Println(“ADVERTENCIA: Error de conexión con la base de datos”); |  |

## Conclusiones y lecciones aprendidas

***Conclusiones***

**Mariana Cadena Romero:**

Al desarrollar este proyecto pude reafirmar los conceptos vistos a lo largo del semestre como la conexión en red, la importancia de la internacionalización y el uso de frameworks que gracias a todo esto permite construir un producto de software seguro y completo. También la gran importancia que tiene realizar un análisis estático de código que gracias a este se puede entregar o finalizar un programa con buenas practicas de programación.

**Esmeralda Jimenez Ramos**

A lo largo de la realización de este proyecto pude adquirir mucho conocimiento sobre desarrollo, como es el uso de frameworks, los cuales facilitan el manejo de bases de datos y la comunicación en red para así poder hacer productos de software más seguros, eficientes y distribuidos, del mismo modo el uso del análisis estático que nos ayuda a identificar problemas en el código que normalmente no notamos, como las malas prácticas o posibles bugs mejorando la calidad de nuestros productos.

***Lecciones aprendidas***

**Mariana Cadena Romero:**

* El uso de frameworks como JPA, Node js el primero permite acceder a la informacion en la base de datos de manera segura y fácil y el segundo frameworks también fue “fácil” entender y desarrollar ya que mediante eventos mandados a llamar en el programa permitiendo una comunicación bidireccional.
* El análisis estático de código como lo mencione en mi conclusión aprendí que es de suma importancia ya que permite encontrar errores, falsos positivos, bugs antes de entregar el producto y así entregar un producto efectivo.
* La Internacionalización. En cuanto a este aspecto aprendí que hay diferentes formas de internacionalizar un producto, ya sea “Manualmente” donde tu le des al “cliente” elegir en que idioma quiere ver correr todo el programa o que lo adaptes “localización” (país e idioma de la pc) del cliente.

**Esmeralda Jimenez Ramos**

* El uso de frameworks puede llegar a ser de gran utilidad si se tienen en cuenta diferentes aspectos que pueden afectar el desarrollo, como lo es el tiempo que se utilizara para el aprendizaje del mismo y el impacto que tiene en el presupuesto del proyecto
* El análisis estático de código es de mucha utilidad para identificar errores que se cometen de manera frecuente y así corregirlos a tiempo mejorando la calidad de nuestro proyecto
* La internacionalización y la programación en red son temas que han ido y seguirán creciendo a lo largo del tiempo de la informática por lo tanto son temas muy importantes para nosotros ingenieros de software ya que desarrollamos software que puede o debe ser accesible para el mayor número de personas posible