**Juan Esteban López – 202021147**

**Nicolas Lozano – 202021428**

**Manuel Pérez – 202021826**

**Entrega 1 - Proyecto 1**

**Diseño y Programación Orientada a Objetos**

**John Casallas**

***Tabla de contenido***

**0. Descripción del análisis.**

**1. Diagrama UML3**

**2. Requerimientos Funcionales3**

2.1 Usuarios3

2.2 Administrador6

**3. Restricciones del Proyecto**

**0. Descripción del análisis de proyecto.**

Para la primera entrega del proyecto de diseño y programación orientada a objetos se hizo un análisis del problema planteado en el enunciado. Con el análisis del problema se pudo identificar los elementos importantes del contexto y así poder definir un dominio para el programa. Luego de la abstracción se identificaron posibles candidatos a ser clases, clases abstractas, interfaces, etc. Con esta información se pudo hacer un primer diagrama en el que se mostraban las relaciones de cada una de las clases, sin embargo, como este es un proceso iterativo donde se discute y se acuerdan ideas entre todos los integrantes del grupo, al final logramos tener el diagrama UML con el que se va a iniciar la siguiente fase.

Después de realizar el diagrama de clases se especificaron los requerimientos funcionales identificados a partir del contexto dado. Para esta parte se utilizaron historias de usuario para mayor claridad respecto al objetivo del requerimiento. En este proceso surgieron diferentes restricciones las cuales se tuvieron en cuenta para la última parte del documento en la que se mencionan los requerimientos no funcionales y las restricciones tanto de la aplicación como del uso, las cuales van dirigidas a los usuarios.

Todo lo mencionado anteriormente se puede revisar con mayor detalle a lo largo del documento.

1. Diagrama UML

Diagrama, Esquemático

Descripción generada automáticamente

1. ***Requerimientos Funcionales***
   1. **Usuarios:** 
      1. **Registro de Usuario:** Para los nuevos usuarios debe existir la opción de registrarse al abrir la aplicación. Para ello el usuario debe ingresar algunos datos básico para hacer el registro en el sistema. Todo usuario debe ingresar un nombre de usuario y una contraseña. El sistema debe permitir que el usuario escriba estos datos en consola y almacenarlos para cuando el usuario vuelva a entrar en la aplicación.
      2. **Ingreso a la aplicación usuario**: Todo usuario debe poder ingresar a la aplicación con su nombre de usuario y contraseña previamente registrado. La aplicación debe permitir que el usuario escriba su nombre de usuario y contraseña en consola, y consultar que el nombre de usuario exista y que la contraseña digitada coincida con la almacenada.
      3. **Crear equipo**: Cada usuario de la plataforma podrá crear diferentes equipos a lo largo de la temporada. Sin embargo, el usuario solo puede tener un equipo al tiempo participando en el juego; en caso de querer cambiar el equipo durante la temporada, se reiniciará el puntaje y se eliminará el equipo anterior. Cada equipo deberá tener exactamente 15 jugadores, 2 arqueros, 5 defensores, 5 mediocampistas y 3 delanteros. Todos los jugadores deben hacer parte de un equipo que pertenezca a la liga real. Cada equipo se debe armar con un presupuesto inicial cuyo valor se define al inicio de la temporada. El sistema debe permitir que el usuario escoja a sus jugadores de la nómina ingresada por el usuario administrativo, debe validar que el equipo cumpla con los requisitos y que no se sobrepase el presupuesto. Una vez creado el quipo debe almacenarlo.
      4. **Seleccionar alineación:** Previo al inicio de cada jornada (hasta la medianoche anterior al primer partido), el usuario podrá cambiar su alineación titular (jugadores que suman puntos). Esto con el fin de que no haya jugadores lesionados o suspendidos en la alineación titular. En caso de que el usuario no cambie su formación, su alineación titular será la misma que utilizó la jornada anterior. La alineación titular debe estar compuesta por 11 jugadores: 1 arquero, 4 defensas, 4 mediocampistas y 2 delanteros. Además, debe poder seleccionar un capitán. El sistema debe permitir que el usuario seleccione su alineación, sin embargo, debe verificar que la fecha no pase la permitida y además asegurarse que la formación coincida con las determinadas. Finalmente, el sistema debe actualizar el archivo con la formación de la plantilla seleccionada.
      5. **Compra y venta de jugadores:** El usuario puede realizar compras y ventas de jugadores (teniendo en cuenta su presupuesto). Las ventas tendrán un descuento del 3%, sin embargo, las compras si se deberán hacer por el valor completo del jugador. El usuario en ningún momento de la temporada podrá tener ni más ni menos de 15 jugadores en su equipo. La aplicación debe permitir la venta y compra de jugadores. Debe validar que siempre haya 15 jugadores por equipo y que se sigan respetando la distribución de posiciones determinada. Así mismo debe calcular el valor de venta y actualizar los saldos del presupuesto de cada equipo. Finalmente, actualizar la plantilla del equipo con las actualizaciones hechas.
      6. **Ranking equipos de fantasía:** El usuario podrá ver el ranking de equipos de fantasía hasta la fecha. La aplicación debe poder calcular está información con todos los reportes suministrados hasta la última fecha.
      7. **Rankings jugadores:** El usuario podrá ver el ranking de jugadores que más puntos han sumado hasta la fecha. La aplicación debe poder calcular está información con todos los reportes suministrados hasta la última fecha.
      8. **Mejor equipo de la fecha:** El usuario podrá consultar el mejor equipo de fantasía de una jornada. La aplicación debe poder calcular está información con el reporte suministrado después de cada fecha.
      9. **Mejor jugador de la fecha:** El usuario podrá consultar el jugador que más puntos sumó en una jornada. La aplicación debe poder calcular está información con el reporte suministrado después de cada fecha.
      10. **Consultar puntaje del equipo de fantasía:** El usuario debe poder consultar el puntaje del equipo de fantasía y, además, consultar el puntaje de los jugadores por separado. Para esto la aplicación debe tener la capacidad de actualizar el puntaje de cada jugador real en el momento en que se acabe cada partido, teniendo en cuenta las reglas para asignar puntos.
   2. **Administrador:**
      1. **Ingreso a la aplicación administrador**: El administrador debe poder ingresar a la aplicación con su nombre de usuario y contraseña previamente establecidos. Para esto, la aplicación debe poder guardar la información del administrador, y así estar en la capacidad de validar las credenciales.
      2. **Agregar datos del partido y jugadores:** Al finalizar cada partido de la temporada, el administrador deberá registrar el resultado de dicho partido, así como el puntaje de cada uno de los jugadores. La aplicación debe ser capaz de leer el reporte que sube el administrador con los datos de cada fecha donde se incluye la información de cada partido y cada jugador.
      3. **Configuración temporada**: El administrador debe poder configurar los aspectos de la temporada como cantidad de fechas, programación de los partidos y la nómina de los equipos reales. La nómina de los equipos reales se mantiene constante durante toda la temporada. Para esto la aplicación le debe dar la opción de ingresar los datos y además retener estos datos durante toda la temporada para así poder manipularlos y que la programación que lleva la aplicación sea acorde a la programación de la temporada real.
2. ***Restricciones y requerimientos no funcionales del proyecto***
   1. **Restricciones:**
      1. **Lectura de Archivos:** La aplicación es la única que podrá leer y escribir los archivos guardados en la carpeta compartida. Ningún usuario puede acceder ni modificar a la información almacenada.
      2. **Almacenamiento de la información:** La información y datos de los equipos y usuarios debe estar en una carpeta/archivo diferente del archivo en el que se encuentra el código fuente de la aplicación.
      3. **Actualización en Git:** La aplicación y los usuarios deben hacer uso de un cliente Git para actualizar la información. La aplicación no se encarga de esto directamente.
      4. **Lectura de archivos:** La aplicación puede leer archivos JSON y CSV. En estos formatos será cargada la información de los juegos de cada fecha, así como almacenada la información de los equipos y usuarios.
      5. **Zona horaria:** La zona horaria es la misma para todos los usuarios, no importa en que parte del mundo estén ubicados. Esto mismo aplica para todos los partidos de la temporada.
      6. **Lenguaje de desarrollo:** La aplicación será desarrollada en su totalidad en el lenguaje de programación Java.
      7. **Interacción con el usuario:** La interacción del usuario con la aplicación será basada en consola.
   2. **Requerimientos no Funcionales:**
      1. **Persistencia:** La aplicación debe ser persistente. Las listas de equipos y jugadores, la conformación de los equipos de fantasía, los resultados de los partidos, el desempeño de los jugadores, los puntos asignados a cada equipo, etc. Deben ser persistentes en el tiempo y toda la información debe poder se accedida.
      2. **Usabilidad:** La aplicación será la misma para los usuarios y para el usuario administrativo. Sin embargo, según el tipo de usuario tendrá una visibilidad diferente, así como acceso a diferentes opciones e información.
      3. **Seguridad:** La aplicación debe garantizar la seguridad de los datos personales de los usuarios, así como la información privada de la compañía.