Laboratorio 2

Con este laboratorio se espera que el estudiante adquiera competencias en la implementación de operaciones básicas, el uso básico del lenguaje **ruby**, así como reafirmar conceptos presentados en el curso hasta el momento.
También se espera que el estudiante recurra al material disponible en el grupo **Forge 2015** del curso y que recurra a Internet con espíritu crítico, identificando y corroborando fuentes confiables de información.

Letra

Según las ventas recibidas sobre la aplicación elaborada en el laboratorio 1, se pudo constatar que hay una alta demanda del producto, esto provocó que los usuarios nos hicieran llegar sus comentarios acerca del mismo.

Teniendo en cuenta la devolución de los usuarios se pensó en mejorar la aplicación para poder brindar algunas nuevas funcionalidades y mejorar las que ya se estaban brindando.

La principal mejora sería ofrecer una visualización a nivel gráfico, con lo cual se podrá administrar de manera más amigable y con un mayor nivel de usabilidad, las funcionalidades ya existentes y las nuevas que se detallarán a continuación.

Funcionalidades realizadas

Las funcionalidades realizadas en el laboratorio 1 se deben administrar mediante la interfaz gráfica (UI).

Jugadores

Los jugadores serán agregados a la aplicación con su fecha de nacimiento.
Tendremos cuatro tipo de Jugadores:

- Arquero
- Defensa
- Volante
- Delantero

Debe haber al menos un jugador de cada tipo en el equipo y entre ellos solo un arquero para que el equipo se considere completo.

Para cada tipo de jugador, se quiere que el sistema lleve registro de diferentes estadísticas:

De los arqueros interesa saber la cantidad de goles que le anotaron, y cuántos de esos goles fueron en el área chica. De los defensas se quiere registrar cuantos pases del rival interceptó.
De los volantes y delanteros interesa saber la cantidad de goles anotados.
Además en el caso de los delanteros también se quiere llevar registro del porcentaje de efectividad al intentar generar un gol:

% efectividad = (goles / cantidad de tiros al arco) * 100

Y por último para los volantes interesa porcentaje de pases correctos

%pases correctos = (cantidad de pases correctos / cantidad de pases) * 100

Transcurso de un partido

Una de las funcionalidades nuevas requeridas por los usuarios, fue poder ingresar con más detalle lo que sucede en el transcurso del partido (en lugar de ingresar simplemente el resultado como en la anterior versión). Por eso el nuevo sistema cuenta con una pantalla específica para el partido en juego.

Durante el transcurso del partido se podrán registrar eventos relacionados al mismo. Al registrar un nuevo evento (o acción de juego) siempre hay un jugador involucrado.

Las diferentes acciones de un partido que pueden ocurrir son las típicas de un partido de fútbol:

- Tiro al arco
- Gol
- Gol en area chica
- Pase correcto
- Pase incorrecto
- Pase interceptado
- Amonestación con Tarjeta Amarilla
- Amonestación con Tarjeta Roja

Si un jugador recibe dos amonestaciones con tarjetas amarillas, automáticamente recibirá luego una amonestación con la tarjeta roja.

Un jugador amonestado con tarjeta roja no podrá participar de ninguna nueva incidencia de juego.

Tabla de Anuncios

Otro de los requerimientos muy solicitado es el de poder tener una tabla de anuncios donde se visualizan los distintos eventos que van ocurriendo en el campeonato.

Esta deberá dar información sobre:

- Inicio: “El encuentro Nacional vs Danubia ha comenzado.”
- Amonestaciones:
    - “Luis Suárez (Nacional) ha sido amonestado con una tarjeta amarilla.”
    - “Luis Suárez (Nacional) ha sido amonestado con una tarjeta roja.”
    - “Luis Suárez (Nacional) ha sido amonestado con una tarjeta roja por acumulacion de amarillas.”
    -
- Goles:
    - “Luis Suárez (Nacional) ha convertido un gol”
    - “Luis Suárez (Nacional) ha convertido un gol en el área chica”
- Pases:
    - “Nicolás Lodeiro (Nacional) ha realizado un pase correcto”
    - “Nicolás Lodeiro (Nacional) ha generado un pase que no llega a destino”
    - “Diego Lugano (Nacional) ha interceptado un pase”
- Otras incidencias:
    - “Nicolas Lodeiro (Nacional) ha rematado al arco”
- Fin: “El encuentro Nacional vs Danubio ha finalizado. Resultado 2 - 0.”

Observacion. También la pantalla de “Partido en juego” tiene una versión reducida de la tabla de anuncios, donde solo se muestran los anuncios del partido en juego.

Especificacion (program.rb)

initialize

Parámetros:

- teams_size: Cantidad de jugadores por equipo. (Valor por defecto: 5)

Resultado esperado:

- Se crea una instancia de la clase programa.

set_championship

Parámetros:

- name: Nombre del campeonato.
- teams_size: Cantidad de jugadores por equipo. (Valor por defecto: 5)

Resultado esperado:

- Modifica el nombre y la cantidad de jugadores por equipo de un campeonato.

championship_can_be_played
Parámetros:

- No tiene

Resultado esperado:

- Debe retornar un string si no hay equipos para disputar el campeonato.
- Debe retornar un string si la cantidad de equipos disponibles es impar.
- Debe retornar un string si hay equipos que tienen menos de la cantidad de jugadores requeridos.
- Debe retornar nil en caso de que el campeonato se pueda jugar.

championship_start
Parámetros: No recibe
Resultado esperado:

- Comienza en campeonato en caso de que el campeonato no haya sido comenzado.

championship_name
Parámetros: No recibe
Resultado esperado:

- Debe retornar el nombre del campeonato.

championship_started?
Parámetros: No recibe
Resultado esperado:

- Debe retornar true si el campeonato fue iniciado, false en caso contrario.

add_team
Parámetros:

- team_name: Nombre del equipo.

Resultado esperado:

- Agrega un equipo al campeonato.
- Debe retornar un string si ya existe un equipo con ese nombre.

add_player
Parámetros:

- ci: Cédula de identidad del jugador.
- name: Nombre del jugador.
- birthdate: Fecha de nacimiento del jugador.
- position: Posición del jugador

Resultado esperado:

- Agrega un jugador al campeonato.
- Debe retornar un string si ya existe un jugador con esa cédula de identidad

add_player_to_team

Parámetros:

- team_name: Nombre del equipo.
- player_id: Cédula de identidad del jugador.

Resultado esperado:

- Debe retornar nil si se agregó el jugador al equipo.
- Debe retornar un string si se quiere agregar un arquero cuando ya existe uno para ese equipo.
- Debe retornar un string si el equipo ya está completo.
- *(Recordar que un equipo completo es aquel que tiene al menos un tipo de cada jugador y que solo debe de haber un arquero y solo uno en el equipo.)*

player_list

Parámetros:

- No tiene.

Resultado esperado:

- Debe retornar una lista (Array) con todos los jugadores del campeonato.

team_list

Parámetros:

- No tiene.

Resultado esperado:

- Debe retornar una lista (Array) con todos los equipos del campeonato.

matches_list

Parámetros: No recibe

Resultado esperado:

Debe retornar una lista (Array) con la información de cada uno de los partidos a disputarse en el campeonato. Cada elemento de esta lista debe ser un hash con las siguientes claves:

id: identificador único del partido

description: String indicando los equipos que juegan el partido (ej. “Nacional vs River”)

result: String indicando el resultado del partido (ej. “2-1”)

state: Simbolo indicando el estado del partido (:pending, :in_progress, :ended)

Ejemplo:

[

{ id: “Nacional_River”, description: “Nacional vs River”, result: “”, state: :pending }

{ id: “Nacional_Rampla”, description: “Nacional vs Rampla”, result: “0-2”, state: :ended }

{ id: “River_Rampla”, description: “River vs Rampla”, result: “1-2”, state: :in_progress}
]

get_match
Parámetros:

- match_id: Identificador del partido

Resultado esperado:
Debe retornar un hash con la información del partido identificado con el identificador *match_id*.

El hash debe tener las siguientes claves:
state: Simbolo indicando el estado del partido (:pending, :in_progress, :ended)
description: String indicando los equipos que juegan el partido (ej. “Nacional vs River”)
result: String indicando el resultado del partido (ej. “2-1”)
team_a: String indicando el nombre del primer equipo
team_b: String indicando el nombre del segundo equipo
news: Array de Strings. Cada elemento es la descripcion de una acción del juego ocurrida durante el partido.

Ejemplo:

```
{
        state: :in_progress,
        description: “River vs Rampla”,
        result: “1-1”,
        team_a: “River”,
        team_b: “Rampla”,
        news: [
           “Luis Suárez (River) ha convertido un gol”,
           “Luis Suarez (River) ha realizado un pase correcto”,
           “Diego Forlan (Rampla) ha convertido un gol en el area chica”
        ]
}
```

start_match
Parámetros:

- match_id: Identificador del partido

Resultado esperado:
Debe cambiar el estado del partido identificado con el identificador *match_id* a “en juego”

end_match

Parámetros:

- match_id: Identificador del partido

Resultado esperado:

Debe cambiar el estado del partido identificado con el identificador *match_id* a “finalizado”

available_players_list_for_match

Parámetros:

- match_id: Identificador del partido
- team_name: Nombre del equipo

Resultado esperado:

Debe retornar un Array con información de todos los jugadores del equipo con nombre *team_name* que aún se encuentran disponibles para participar del partido con identificador *match_id* (es decir que si un jugador fue expulsado no debe formar parte de esta lista).

Cada elemento de esta lista debe ser un hash con las claves:
id: identificador del jugador (cédula)
name: nombre del jugador

Ejemplo

```
[
        {id: “11111111”, name: “Luis Suarez”},
        {id: “22222222”, name: “Nicolas Lodeiro”}
]
```

add_match_action

Parámetros:

- match_id: Identificador del partido
- team_name: Nombre del equipo involucrado en la acción de juego
- player_id: Id del de jugador involucrado en la acción de juego
- action: Simbolo indicando cual fue la acción ocurrida.
  Los posibles valores son:

```
'Tiro al arco' => :shoot,
'Gol' => :goal,
'Gol en area chica' => :small_box_goal,
'Pase correcto' => :good_pass,
'Pase incorrecto' => :wrong_pass,
'Pase interceptado' => :intercepted_pass,
'Tarjeta Amarilla' => :yellow_card,
'Tarjeta Roja' => :red_card
```

Resultado esperado:

- Debe registrar esta nueva acción relacionada al partido

- Debe actualizar la información del sistema teniendo en cuenta la acción se está registrando. Por ejemplo, actualizar resultado del partido, actualizar las estadísticas de los jugadores, etc.

get_table_data
Parámetros: No recibe
Resultado esperado:

- Debe retornar un Array conteniendo un elemento por cada equipo.
- Esta lista debe estar ordenada por puntos y en caso de empate en puntos, por diferencia de goles.
- Cada elemento de esta lista debe ser un hash que contenga la información necesaria para armar la tabla de posiciones, es decir las siguientes claves.
  team_name: nombre del equipo
  played_matches: cantidad de partidos jugados,
  won_matches: cantidad de partidos ganados,
  drawn_matches: cantidad de partidos empatados,
  lost_matches: cantidad de partidos perdidos,
  goals_difference: diferencia de goles,
  points: puntos

Ejemplo:

```
[
        {
           team_name: “River”,
           played_matches: 2,
           won_matches: 1,
           drawn_matches: 1,
           lost_matches: 0,
           goals_difference: 2,
           points: 4
        },
        {
           team_name: “Rampla”,
           played_matches: 1,
           won_matches: 0,
           drawn_matches: 1,
           lost_matches: 0,
           goals_difference: 0,
           points: 1
        }
]
```

get_news_data
Parámetros: No recibe
Resultado esperado:
Debe retornar un Array de Strings representando los distintos eventos que han ocurrido a lo largo del campeonato (leer sección Tabla de Anuncios de la letra).

Ejemplo:

```
[
  “El encuentro River vs Rampla ha comenzado”,
  “Luis Suárez (River) ha convertido un gol”,
  “Luis Suarez (River) ha realizado un pase correcto”,
  “Diego Forlan (Rampla) ha convertido un gol en el area chica”,
  “El encuentro River vs Rampla ha finalizado. Resultado 1-1.”
]
```

Forma de Entrega

La entrega deber ser mediante un correo electrónico incluyendo el link hacia el repositorio que deberá contener:

- Todos los archivos que generó y/o modificó para completar el laboratorio.
- Un archivo llamado *modelo.pdf* con el modelo de dominio generado para realizar el laboratorio.
- Un documento de texto llamado *integrantes.txt* conteniendo el nombre de cada integrante del grupo:
  - Luis Perez - Grupo Lunes
  - Laura López - Grupo Lunes

Fecha de Entrega

Los trabajos deberán ser entregados antes del domingo 1 de noviembre a las 23:30 horas. forge2015@moove-it.com con el asunto ‘[Forge2015] - Lab2’