

INFORME ITERACIÓN 2

**Dementei E., Deniz L.
Grupo 2 - BadCoders
Ingeniería del Software 1
Facultad de Informática
Universidad del País Vasco**

ACTUALIZADO 21/MAR/2020

Indice

Indice	2
Avance del proyecto	3
Interfaces	4
Interfaz de cliente	4
Interfaz de Admin	4
Casos de uso	5
Editar/borrar evento.....	5
Editar/borrar pregunta.....	5
Editar/borrar respuesta.....	6
Ver/editar perfil.....	7
Cargar monedero.....	7
Movimientos de cuenta global.....	8
Crear usuario admin.....	8
Definir resultados de eventos.....	9
Anular apuesta.....	9
Dedicación horaria	10
Diseño UML	11
Diagrama de Clases de Dominio (DCD)	11
Diagrama de Casos de Uso (DCU).....	12
Flujo de eventos	13
Consideraciones	19

Avance del proyecto

Los casos de uso con los que partía el sistema eran:

- Crear pregunta
- Ver respuestas

Durante la primer iteración generamos los siguientes casos de uso exigidos por requisito:

- Login
- Registro
- Crear evento (Vista admin)
- Crear respuestas (Vista admin)
- Logout

Además, complementamos la iteración desarrollando dos casos de uso más:

- Apostar (Vista cliente)
- Histórico de apuestas por cliente (Vista cliente)

Durante la segunda iteración, además de analizar y optimizar los casos de uso anteriores, hemos desarrollado los siguientes casos de uso:

- Editar/borrar Evento (Vista admin)
- Editar/borrar pregunta (Vista admin)
- Editar/borrar respuesta (Vista admin)
- Ver/editar Perfil (Vista cliente)
- Cargar monedero (Vista cliente)
- Movimientos de cuenta global (Vista admin)
- Crear usuario admin (Vista admin)
- Definir resultados de eventos (Vista admin)
- Anular apuesta (Vista cliente)

Interfaces

Interfaz de cliente



Interfaz de Admin



Casos de uso

Los casos de uso de esta iteración se describen a continuación, así como un esbozo general de la solución brindada. La dedicación horaria estará en otro apartado.

Editar/borrar evento.

En el desarrollo de este caso de uso, el sistema permite elegir el evento a través del `jCalendar`. Se habilita un `JTextFrame` que copia el título del evento seleccionado. Cuando se presiona el botón **Modificar evento**, el sistema lo actualiza en el objeto y lo persiste en la base de datos. Se refresca el `JComboBox` de eventos para mostrar el evento con el nuevo título. Para eliminar el evento, se presiona una vez el botón **Eliminar evento**, que dará lugar a otro botón de **Confirmar eliminar?**, de presionarse este, el objeto es borrado de la base de datos y el `JComboBox` actualizado. Cualquier apuesta que esté en estado Pendiente pasa a ser anulada por el administrador (código 2), acreditando el dinero apostado a la cuenta del cliente, debitándolo de la cuenta de saldo global del sistema, anulando cada una de sus preguntas y sus respuestas, dejando la pregunta y todas sus respuestas en estado inactiva.

The screenshot shows a window titled "Editar evento". It contains a date selection interface with a calendar for April 2020. The calendar shows the 28th of April is selected. To the right of the calendar is a dropdown menu labeled "Seleccionar evento" with "Atlético-Athletic" selected. Below that is a text input field labeled "Pregunta a modificar" containing the text "Atlético-Athletic". At the bottom of the window are three buttons: "Cancelar", "Eliminar evento", and "Modificar evento".

Editar/borrar pregunta.

En el desarrollo de este caso de uso, el sistema permite elegir el evento a través del `jCalendar`. De seleccionar una fecha que contiene eventos, se habilita un `JComboBox` con los eventos de ese día. Si se selecciona un evento del `JComboBox`, se habilita el siguiente `JComboBox` de Preguntas sobre ese evento. Se habilita un `JTextFrame` que copia el título de la pregunta seleccionada y otro `JTextFrame` con la apuesta mínima sobre esa pregunta. Cuando se presiona el botón **Modificar pregunta**, el sistema la actualiza en el objeto y la persiste en la base de datos. Se refresca el `JComboBox` de preguntas para mostrar la pregunta con el título modificado. Para eliminar la pregunta, se presiona una vez el botón **Eliminar pregunta**, que dará lugar a otro botón de **Confirmar eliminar?**, de presionarse este, el objeto es borrado de la base de datos y el `JComboBox` de preguntas actualizado. Cualquier apuesta que esté en estado Pendiente pasa a ser anulada por el administrador (código 2), acreditando el dinero apostado a la cuenta del

cliente, debitándolo de la cuenta de saldo global del sistema, anulando cada una de sus respuestas y dejando la pregunta y todas sus respuestas en estado inactiva.

Editar pregunta

Fecha evento

abril 2020

	lun	mar	mié	jue	vie	sáb	do...
14			1	2	3	4	5
15	6	7	8	9	10	11	12
16	13	14	15	16	17	18	19
17	20	21	22	23	24	25	26
18	27	28	29	30			

Seleccionar evento: Atlético-Athletic

Seleccionar pregunta: ¿Quién ganará el partido?

Pregunta a modificar: ¿Quién ganará el partido?

Apuesta mínima: 1.0

Cancelar Eliminar pregunta Modificar pregunta

Editar/borrar respuesta.

En el desarrollo de este caso de uso, el sistema permite elegir el evento a través del jCalendar. De seleccionar una fecha que contiene eventos, se habilita un JComboBox con los eventos de ese día. Si se selecciona un evento del JComboBox, se habilita el siguiente JComboBox de Preguntas sobre ese evento. Si se selecciona una pregunta del JComboBox, se habilita el siguiente JComboBox de Respuestas sobre esa pregunta. Se habilita un JTextFrame que copia el texto de la respuesta seleccionada y otro JTextFrame que trae el coeficiente de ganancia de la respuesta. Cuando se presiona el botón **Modificar respuesta**, el sistema la actualiza en el objeto y la persiste en la base de datos. Se refresca el JComboBox de respuestas para mostrar la respuesta con el texto modificado. Para eliminar la respuesta, se presiona una vez el botón **Eliminar respuesta**, que dará lugar a otro botón de **Confirmar eliminar?**, de presionarse este, el objeto es borrado de la base de datos y el JComboBox de respuestas actualizado. Cualquier apuesta que esté en estado Pendiente pasa a ser anulada por el administrador (código 2), acreditando el dinero apostado a la cuenta del cliente, debitándolo de la cuenta de saldo global del sistema, y dejando la respuesta en estado inactiva.

Editar respuesta

Fecha evento

abril 2020

	lun	mar	mié	jue	vie	sáb	do...
14			1	2	3	4	5
15	6	7	8	9	10	11	12
16	13	14	15	16	17	18	19
17	20	21	22	23	24	25	26
18	27	28	29	30			

Seleccionar evento: Atlético-Athletic

Seleccionar pregunta: ¿Quién ganará el partido?

Seleccionar Respuesta: Sin goles

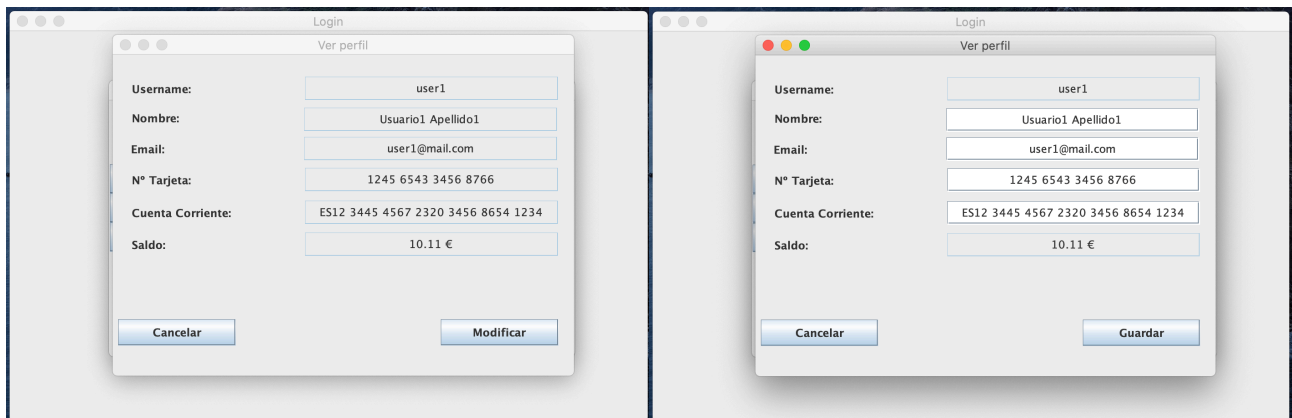
Respuesta a modificar: Sin goles

Coeficiente de ganancia: 2.5

Cancelar Eliminar respuesta Modificar respuesta

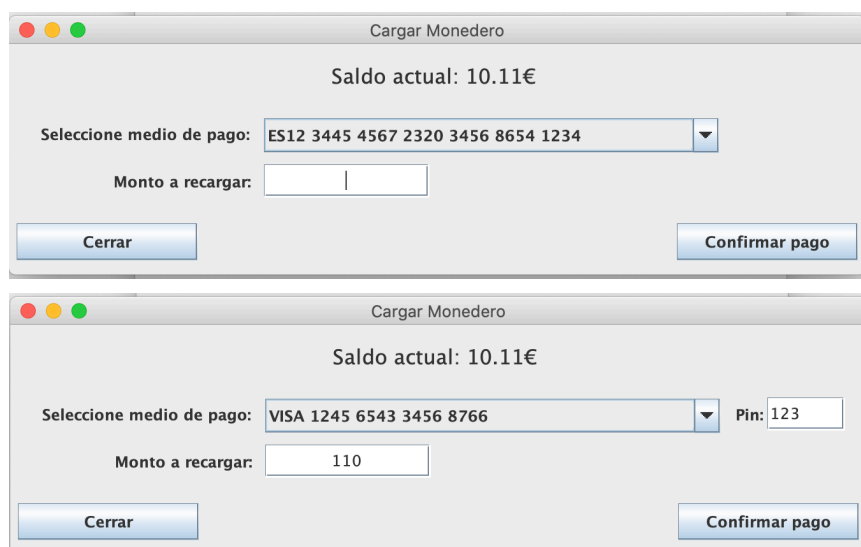
Ver/editar perfil.

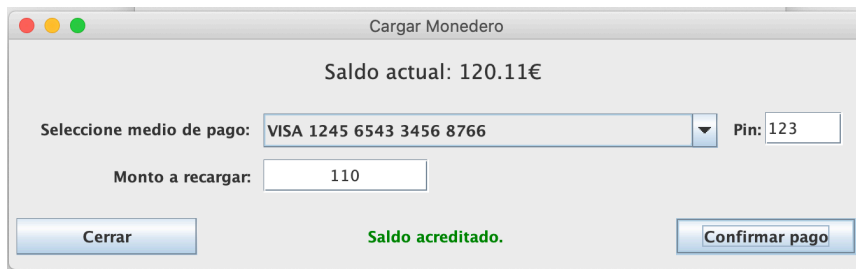
La GUI recibe por parámetro el objeto del cliente que se logueó, trae de la base de datos la información actualizada del cliente y la muestra en el JTextFrame que le corresponde a cada atributo. Si se presiona el botón Modificar, los JTextFrame se ponen en setEditable(true) permitiendo su edición y cambia el botón por Guardar. Al presionarlo, lleva el objeto a DataAccess y lo persiste, volviendo a poner los JTextFrame en setEditable(false).



Cargar monedero.

La GUI recibe por parámetro el objeto del cliente que se logueó, trae de la base de datos la información actualizada del cliente, mostrando su saldo en cuenta, su número de cuenta bancaria y número de tarjeta de crédito, ambas pueden ingresarse desde el caso de uso Ver/Editar Perfil. El cliente elige el medio de pago desde el JComboBox e ingresa el monto a recargar. Si elige tarjeta de crédito, se habilita un JTextFrame para ingresar el CVC de la tarjeta. Cuando se presiona **Confirmar pago**, se envía la información al DataAccess quién haría la conexión con el broker correspondiente. A los efectos del sistema, el CVC por defecto es 123, que se escribe como como texto de inicio, pudiendo así validar la tarjeta. El objeto cliente no guarda el CVC por seguridad. El saldo es acreditado y se actualiza el saldo total del cliente en la parte superior.





Cargar Monedero

Saldo actual: 120.11€

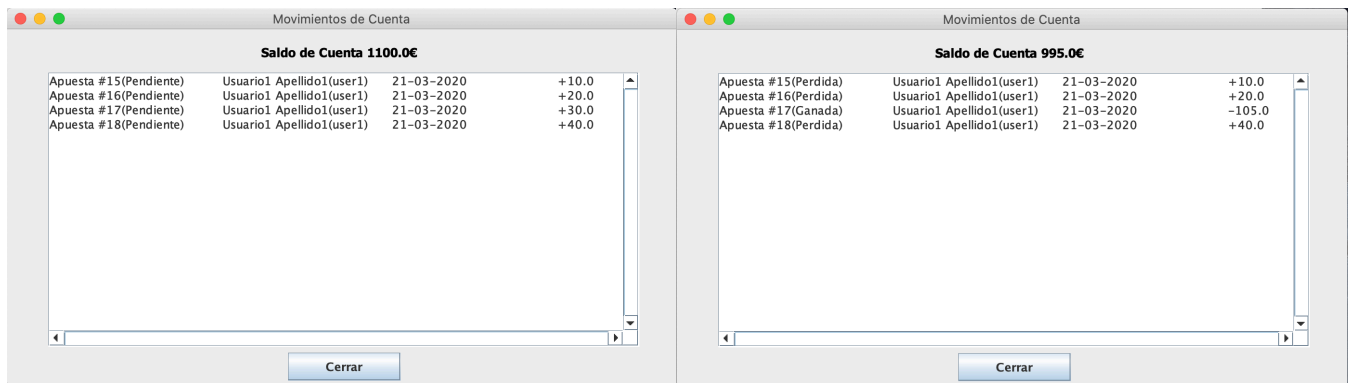
Seleccione medio de pago: VISA 1245 6543 3456 8766 Pin: 123

Monto a recargar: 110

Cerrar Saldo acreditado. Confirmar pago

Movimientos de cuenta global.

Consulta al único objeto de la clase CuentaGlobal que mantiene la información del saldo del sistema y una colección de todas las apuestas realizadas. Se lista cada una de ellas con su ID, el estado en el que se encuentra, el usuario que apostó, cuándo lo hizo y el flujo de dinero en cuenta. La imagen de la izquierda muestra el estado con las apuestas pendientes, la de la derecha el estado cuando se ha definido la respuesta correcta del evento.



Movimientos de Cuenta

Saldo de Cuenta 1100.0€

Apuesta #15(Pendiente)	Usuario1 Apellido1(user1)	21-03-2020	+10.0
Apuesta #16(Pendiente)	Usuario1 Apellido1(user1)	21-03-2020	+20.0
Apuesta #17(Pendiente)	Usuario1 Apellido1(user1)	21-03-2020	+30.0
Apuesta #18(Pendiente)	Usuario1 Apellido1(user1)	21-03-2020	+40.0

Cerrar

Movimientos de Cuenta

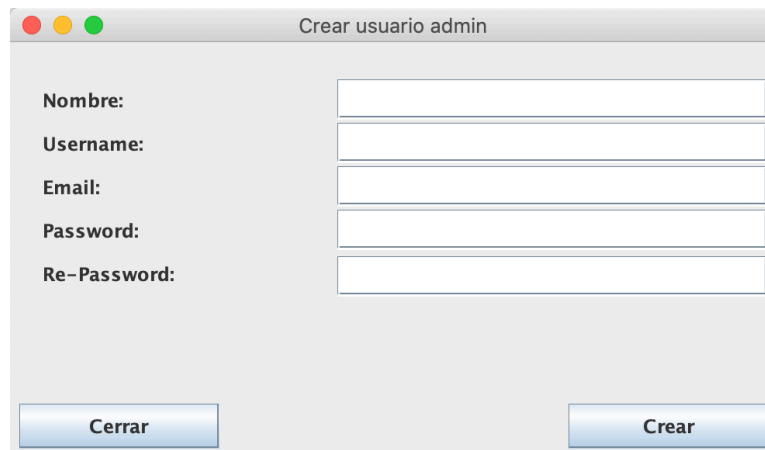
Saldo de Cuenta 995.0€

Apuesta #15(Perdida)	Usuario1 Apellido1(user1)	21-03-2020	+10.0
Apuesta #16(Perdida)	Usuario1 Apellido1(user1)	21-03-2020	+20.0
Apuesta #17(Ganada)	Usuario1 Apellido1(user1)	21-03-2020	-105.0
Apuesta #18(Perdida)	Usuario1 Apellido1(user1)	21-03-2020	+40.0

Cerrar

Crear usuario admin.

Un usuario admin es creado por otro usuario admin. El sistema verifica que Password y Re-Password sean iguales, que Email coincida una expresión regular para email y que Username no esté ingresado previamente en la base de datos.



Crear usuario admin

Nombre:

Username:

Email:

Password:

Re-Password:

Cerrar Crear

Definir resultados de eventos.

La interfaz permite elegir el evento por la fecha a través de un *jCalendar*, y se van desplegando en cascada los *JComboBox* de evento, pregunta y respuesta conforme se van eligiendo. Una vez que se selecciona la respuesta correcta a la pregunta de ese evento, se hace click en el botón **Definir resultados**. Se envía la información al *DataAccess*, quien buscará la respuesta elegida en la base de datos y la marcará como respuesta ganadora. Luego traerá de la base de datos la pregunta a la que correspondía esa respuesta, buscará todas las respuestas asociadas y de cada una traerá todas las apuestas sobre ellas, seteando el estado de la apuesta a perdida (código 1) o ganada (código 3), dependiendo si la respuesta apostada es la correcta; luego seteará la pregunta como resuelta (*pregunta.result=true*). Finalmente, consulta si el evento tiene más preguntas sin resolver, de no haberlas, da por cerrado el evento, seteándolo a *evento.result=true*.

Anular apuesta.

El sistema trae de la base de datos y lista todas las apuestas en estado pendiente (código 0) de ese cliente. Carga el *JComboBox* con el texto de la apuesta (ID/FECHA/PREGUNTA/RESPUESTA), y anula la que se haya seleccionado. Acto seguido, el *JComboBox* es actualizado con las nuevas apuestas pendientes. La apuesta se setea a código 4 (anulado por el usuario), debitándose el monto apostado de la cuenta global del sistema y acreditándole el mismo monto a la cuenta del cliente.

Dedicación horaria

A continuación presentamos los datos de dedicación horaria aproximada por actividad y miembro del equipo.

Actividad	Deniz, L	Dementei, E	Iglesias, J
Editar/borrar evento	4	1	
Editar/borrar pregunta	2	10	
Editar/borrar respuesta	2	10	
Ver/editar perfil	3		4
Cargar monedero	2	6	
Movimientos de cuenta global	5		
Crear usuario admin		2	4
Definir resultados de eventos	18		3
Anular apuesta	5		
Apostar (optimizar)	10		
Histórico de apuestas (optimizar)	4		8
Corrección de bugs	3	10	3
Documentación	6	6	2
Diseño UML		7	2
Ajustes de código	6	8	4
TOTAL	60	60	30

Diseño UML

Diagrama de Clases de Dominio (DCD)

A continuación presentamos el DCD correspondiente a la segunda iteración que, si bien mantiene la base del primero, sufre modificaciones en atributos que permiten generar los nuevos casos de uso.

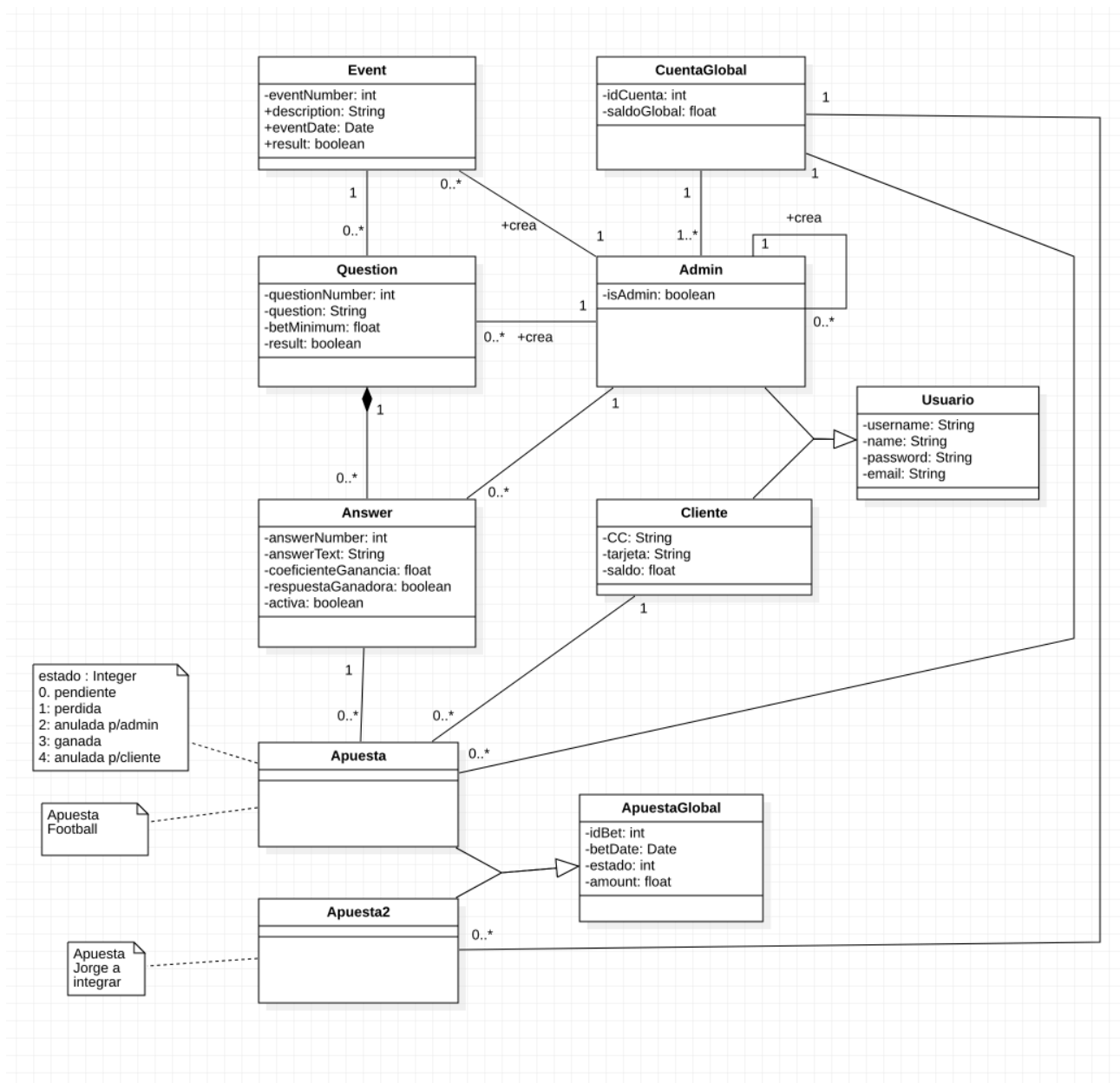
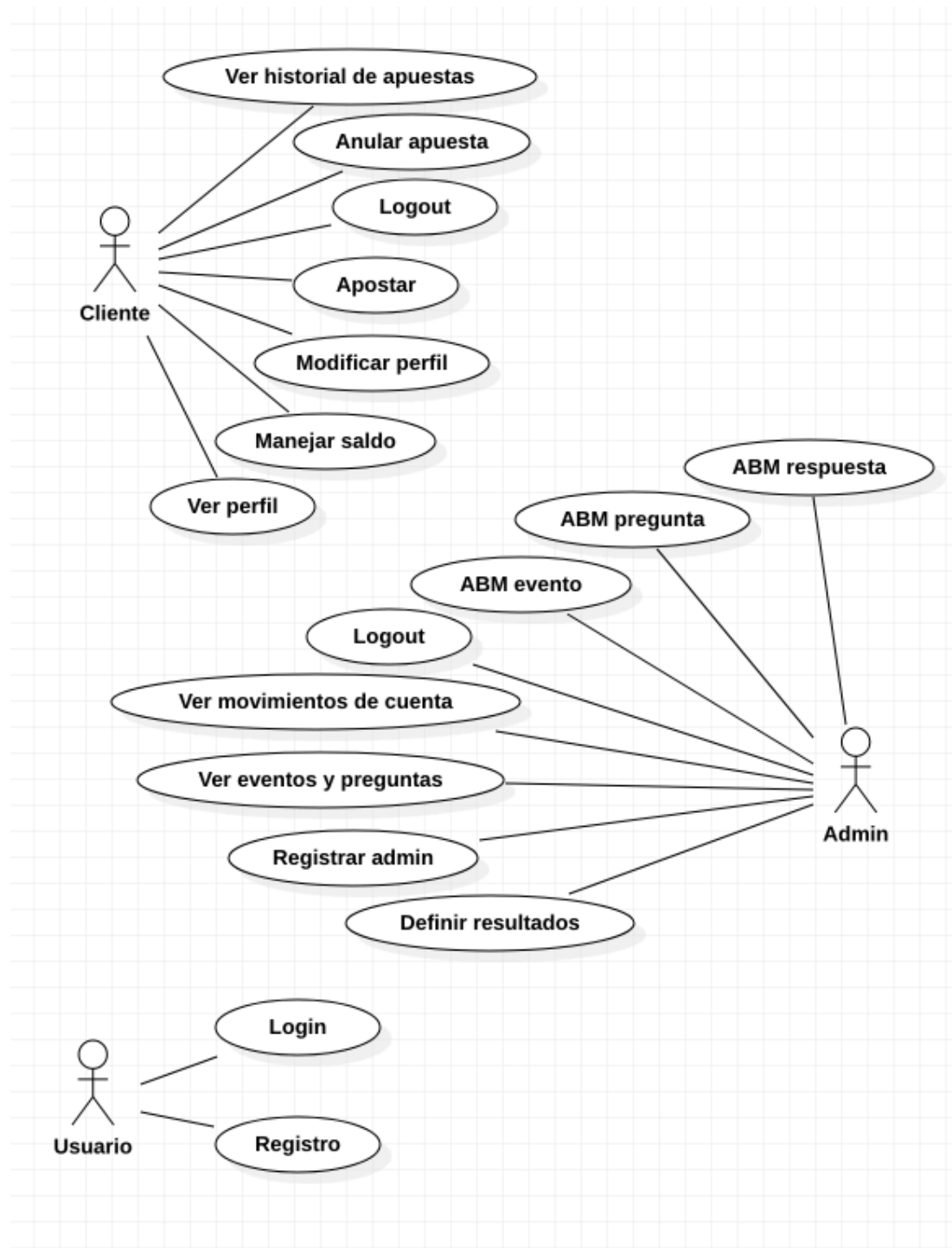


Diagrama de Casos de Uso (DCU)



Flujo de eventos

Nombre	Ver historial de apuestas
Actores	Cliente
Precondiciones	—
Flujo normal	<ol style="list-style-type: none"> 1. El usuario hace login y se vuelve Cliente. 2. Hace click sobre el botón 'Ver historial de apuestas' que le abrirá una GUI. 3. El sistema consulta todas las apuestas que tiene el cliente y muestra alguno de sus campos.
Flujo alterno	—
Postcondiciones	Se muestra el historial de apuestas del cliente logueado y el estado en el que se encuentra cada una de ellas.

Nombre	Apostar
Actores	Cliente
Precondiciones	El usuario está registrado. Existe un evento con al menos una pregunta y una respuesta posible.
Flujo normal	<ol style="list-style-type: none"> 1. El cliente elige el evento, la pregunta y la respuesta sobre la cual apostar. 2. Ingresar el monto a apostar. 3. El sistema verifica que el cliente tenga el saldo suficiente para realizar la apuesta y que el monto a apostar sea mayor al mínimo de la apuesta. 4. Guarda en la base de datos el objeto apuesta y lo agrega a los vectores de Respuesta y Cliente para acceder directamente a las apuestas.
Flujo alterno	<ol style="list-style-type: none"> 3.1. El monto apostado es menor al mínimo, el sistema no permite apostar. 3.2. El monto apostado es superior al saldo de la cuenta.
Postcondiciones	Se guarda la apuesta.

Nombre	Ver/editar perfil
Actores	Cliente
Precondiciones	El cliente existe en la base de datos y está logueado.
Flujo normal	<ol style="list-style-type: none"> 1. El cliente hace click en el botón de 'Ver/Editar Perfil'. 2. Se muestra una GUI con los datos del usuario en un JFrameText cada uno en modo enable false. 3. El usuario hace click en el botón Modificar y los campos se ponen en enable true. 4. El usuario presiona el botón 'Guardar' y los valores son persistidos en la base de datos.
Flujo alterno	—
Postcondiciones	Los datos del usuario fueron editados y guardados.

Nombre	Cargar monedero
Actores	Cliente
Precondiciones	El cliente existe en la base de datos y está logueado como tal.
Flujo normal	<ol style="list-style-type: none"> 1. El sistema trae la información actualizada del saldo, número de cuenta bancaria y número de tarjeta de crédito del cliente. 2. El cliente elige el medio de pago del combobox. 3. Al presionar el botón 'Confirmar pago', el sistema se comunica con el broker a quien se le enviará los datos de pago para que realice la transacción. 4. La transacción es aceptada y el saldo es acreditado a la cuenta del cliente. 5. La GUI actualiza el saldo total disponible del cliente en la parte superior.
Flujo alterno	<ol style="list-style-type: none"> 2.1. Si el cliente elige pagar con tarjeta, el sistema le pedirá el CVC. 4.1. La transacción es rechazada y emite un mensaje de error.
Postcondiciones	El saldo es acreditado en la cuenta del cliente.

Nombre	Ver movimientos de cuenta (global)
Actores	Admin
Precondiciones	El usuario admin está registrado e hizo login.
Flujo normal	<ol style="list-style-type: none"> 1. El admin hace click en 'Ver movimientos de cuenta'. 2. El sistema consulta el saldo de cuentaGlobal y su vector de apuestas. 3. Se lista la información de todas las apuestas realizadas, con su estado y el saldo que se acreditó o se debitó de la cuenta global.
Flujo alterno	—
Postcondiciones	El sistema muestra el flujo de caja que hay en el sistema de apuestas.

Nombre	Registrar admin
Actores	Admin
Precondiciones	El usuario admin está registrado e hizo login.
Flujo normal	<ol style="list-style-type: none"> 1. El admin hace click en 'registrar admin'. 2. El sistema abre una GUI con un formulario para ingreso de datos. 3. Se valida que el correo electrónico se apegue a una expresión regular, que la contraseña y la verificación de contraseña coincida. 4. Se envía al DataAccess la información y se ingresa el nuevo objeto de tipo Admin.
Flujo alterno	4.1. Si el usuario ya está registrado envía un mensaje de error al Admin y queda a la espera de modificar el username.
Postcondiciones	El nuevo admin es registrado en el sistema.

Nombre	Definir resultados
Actores	Admin
Precondiciones	El evento está creado y existe al menos una pregunta y una respuesta. El usuario admin existe en la base de datos e hizo login.
Flujo normal	<ol style="list-style-type: none"> 1. El admin selecciona el evento, luego la pregunta y luego la respuesta ganadora. 2. Presiona sobre el botón 'Definir resultado' y se envía la información al DataAccess. 3. El sistema verifica todas las apuestas sobre esa respuesta y las marca como ganadas. 4. Luego busca todas las respuestas sobre esa pregunta y marca el estado de todas sus apuestas como perdidas. 5. Define la pregunta como resuelta. 6. Si el evento no tiene más preguntas, lo define como cerrado.
Flujo alternativo	—
Postcondiciones	El resultado fue ingresado y las apuestas resueltas.

Nombre	Anular apuesta
Actores	Cliente
Precondiciones	Existe una apuesta ingresada para ese cliente.
Flujo normal	<ol style="list-style-type: none"> 1. El cliente selecciona la apuesta del combobox de apuestas que solamente muestra aquellas que están en estado pendiente (código 0). 2. Presiona el botón Anular apuesta. 3. El sistema busca la apuesta en la base de datos, debita el monto de la cuenta global del sistema, lo acredita en la cuenta de saldo del cliente y marca la apuesta en estado Anulada por cliente (código 4). 4. Se actualiza el combobox de apuestas pendientes para que el cliente pueda anular otra.
Flujo alternativo	—
Postcondiciones	Se anuló la apuesta.

Nombre	Editar evento
Actores	Admin
Precondiciones	Existe el evento en la base de datos.
Flujo normal	<ol style="list-style-type: none"> 1. El admin elige la fecha desde el jCalendar. 2. El JComboBox se llena con los eventos de la fecha seleccionada. 3. Se habilita un campo de texto debajo que copia el nombre del evento. 4. Se modifica el nombre y se da click en guardar.
Flujo alternativo	—
Postcondiciones	Los datos del evento fueron editados y persistidos.

Nombre	Borrar evento
Actores	Admin
Precondiciones	Existe el evento en la base de datos.
Flujo normal	<ol style="list-style-type: none"> 1. El admin elige la fecha desde el jCalendar. 2. El JComboBox se llena con los eventos de la fecha seleccionada. 3. Se habilita el botón 'Eliminar evento'. 4. Cuando se hace click sobre él, da paso a otro botón 'Confirmar eliminar'. 5. Cuando se hace click a este último, se envía el evento al Data Access. 6. Se buscan todas las preguntas sobre ese evento. 7. Se buscan todas las respuestas sobre cada una de las preguntas. 8. Se buscan todas las apuestas sobre cada una de las respuestas y se anulan marcándolas como Anulada por Admin (código 2), debitando el monto apostado de la cuenta global del sistema y acreditándolo a la cuenta del cliente. 9. Se marcan todas las respuestas como activa=false. 10. Se marcan todas las preguntas como result=true. 11. Se marca el evento como result=true.
Flujo alternativo	—
Postcondiciones	El evento fue eliminado.

Nombre	Editar pregunta
Actores	Admin
Precondiciones	Existe el evento y la pregunta en la base de datos.
Flujo normal	<ol style="list-style-type: none"> 1. El admin elige la fecha desde el jCalendar. 2. El JComboBox se llena con los eventos de la fecha seleccionada. 3. Si hay preguntas sobre ese evento, se habilita un JComboBox con el listado de preguntas disponibles. 4. Se habilita un campo de texto debajo que copia el título de la pregunta y otro copia la información de la apuesta mínima. 5. Se modifican los datos y se da click en guardar.
Flujo alternativo	—
Postcondiciones	Los datos de la pregunta fueron editados y persistidos.

Nombre	Borrar pregunta
Actores	Admin
Precondiciones	Existe el evento y la pregunta en la base de datos.
Flujo normal	<ol style="list-style-type: none"> 1. El admin elige la fecha desde el jCalendar. 2. El JComboBox se llena con los eventos de la fecha seleccionada. 3. Se habilita otro JComboBox que se llena con las preguntas sobre ese evento. 4. Se habilita el botón 'Eliminar pregunta'. 5. Cuando se hace click sobre él, da paso a otro botón 'Confirmar eliminar'. 6. Cuando se hace click a este último, se envía la pregunta al Data Access. 7. Se buscan todas las respuestas sobre la pregunta. 8. Se buscan todas las apuestas sobre cada una de las respuestas y se anulan marcándolas como Anulada por Admin (código 2), debitando el monto apostado de la cuenta global del sistema y acreditándolo a la cuenta del cliente. 9. Se marcan todas las respuestas como activa=false. 10. Se marcan la pregunta como result=true.
Flujo alternativo	—
Postcondiciones	La pregunta fue eliminada.

Nombre	Editar respuesta
Actores	Admin
Precondiciones	Existe el evento, la pregunta y la respuesta en la base de datos.
Flujo normal	<ol style="list-style-type: none"> 1. El admin elige la fecha desde el jCalendar. 2. El JComboBox se llena con los eventos de la fecha seleccionada. 3. Si hay preguntas sobre ese evento, se habilita un JComboBox con el listado de preguntas disponibles. 4. Si hay respuestas disponibles en esa pregunta, se habilita un JComboBox con el listado de respuestas disponibles. 5. Se habilita un campo de texto debajo que copia el título de la respuesta y otro copia la información del coeficiente de ganancia. 6. Se modifican los datos y se da click en guardar.
Flujo alternativo	—
Postcondiciones	Los datos de la respuesta fueron editados y persistidos.

Nombre	Borrar respuesta
Actores	Admin
Precondiciones	Existe el evento, la pregunta y la respuesta en la base de datos.
Flujo normal	<ol style="list-style-type: none">1. El admin elige la fecha desde el jCalendar.2. El JComboBox se llena con los eventos de la fecha seleccionada.3. Se habilita otro JComboBox que se llena con las preguntas sobre ese evento.4. Se habilita otro JComboBox que se llena con las respuestas sobre esa pregunta.5. Se habilita el botón 'Eliminar respuesta'.6. Cuando se hace click sobre él, da paso a otro botón 'Confirmar eliminar'.7. Cuando se hace click a este último, se envía la respuesta al Data Access.8. Se buscan todas las apuestas sobre esa respuesta y se anulan marcándolas como Anulada por Admin (código 2), debitando el monto apostado de la cuenta global del sistema y acreditándolo a la cuenta del cliente.9. Se marcan todas la respuesta como activa=false.
Flujo alternativo	—
Postcondiciones	La respuesta fue eliminada.

Consideraciones

En primer lugar, la integración del equipo al inicio de la iteración era la siguiente: Emanuel Dementei, Jorge Iglesias y Leroy Deniz.

La distribución de tareas y desarrollo de los casos de uso, fue acorde a los casos de uso realizado en la primer iteración, a fin de continuar con a línea de desarrollo de cada uno. Sin embargo, el pasaje de Jorge a evaluación global en medio de la iteración, devino en la reestructura de la planificación. Pasamos de un equipo de cuatro integrantes a tres, y luego a dos.

No obstante, Jorge trabajó en la iteración y, por tanto, corresponde hacer referencia de las horas que alcanzó a estar en el apartado de Dedicación horaria.

La estructura del DCD plantea dos tipos de apuestas, la clase Apuesta es con la que trabajamos para el desarrollo del proyecto Bets original sobre partidos de fútbol, sin embargo, hemos dispuesto la creación de un segundo tipo, en principio su clase es Apuestas2, para la integración en el corto plazo del tipo de apuestas del sistema de Jorge sobre F1.

Habiendo aclarado esto, es importante destacar que la dedicación horaria no se corresponde con xp-dev, ya que hemos tenido inconvenientes para configurar git en algunas versiones de eclipse, lo que llevó a trabajar cada uno en sus versiones y, con cada hito alcanzado, hacer el merge antes de subirlo una sola persona. Para la próxima iteración, esto quedará resuelto.