RETROSPECTIVA VAE VICTIS

Análisis retrospectivo del sprint. Reflexión.

En general todos los miembros del equipo sentimos una sensación de plenitud, hemos trabajado de manera muy intensa este último sprint y eso ha tenido sus frutos. Hemos avanzado en muchas partes del proyecto que se quedaron rezagadas durante el último sprint. Hemos dedicado buena parte del tiempo de trabajo a la documentación, un aspecto a bastante a mejorar durante el tercer sprint. Con esto hemos podido desarrollar profundamente los diagramas clases, dominio y capas. También pudimos refinar aún más las historias de usuarios, y se redactaron por completo los patrones aplicados y las decisiones de diseño.

Por otra parte, el grupo en general se siente bastante satisfecho con el trabajo realizado individualmente, todos hemos podido alcanzar la meta que nos impusimos al principio del sprint. Se ha desarrollado por completo el panel de administración de un administrador, el bloque de estadísticas y logros. Además se ha creado la entidad “Card” con la que se generan las cartas que durante la partida son repartidas y que pueden ser usadas por un jugador durante su turno. Además, aunque no se haya alcanzado la totalidad de los turnos hemos realizado un gran avance pudiendo, cumplidas determinadas características, avanzar los turnos entre los jugadores incluso hasta terminar la partida.

Es cierto, que por otra parte el grupo ha sentido mucha presión durante este último sprint por la gran cantidad de trabajo pendiente y aunque no se ha podido terminar en su plenitud debido a la inmensa amplitud del proyecto seleccionado. Esto como es de esperar ha provocado también sensación de tristeza pero aun así satisfechos con cómo hemos podido resolver este proyecto.

La forma de trabajar del equipo ha sido impecable y aunque hubiesen disputas de cómo hacer distintas clases,servicios o métodos, siempre hemos mantenido una gran comunicación entre todos los miembros y hemos utilizado el tablero de kanban propio de GitHub para la designación de tareas. Otro aspecto positivo de nuestro equipo es la implicación del mismo sobre las tareas individuales o por parejas designadas, siempre dispuestos a terminarlas e incluso a ayudar al resto del equipo en otras tareas.

En general respecto a la sensación de todos los miembros del equipo durante todo el proyecto acordamos que deberíamos, todos, haber dedicado más tiempo al proyecto sabiendo aún más que aspiramos a MH, haciendo 2 módulos extras. Aún así, el cómputo total de horas empleadas son 894 horas y con eso queremos plasmar toda la dedicación, esfuerzo y trabajo que conlleva este proyecto.

A continuación adjuntamos una tabla que registra el cómputo de horas empleadas este último sprint, y posteriormente una tabla con el cómputo de todas las horas:

Fecha	Gonzalo Martínez	Álvaro Vázquez	Luis Rodríguez	Álvaro Miranda	Daniel Díaz	Pedro Parrilla
Módulo Estadísticas y logros	20	-	-	-		-
Administración admin	15	-	-	35	-	-
Patrones	-	-	-	-	-	40
Documentación	10	5	5	10	5	5
Acciones y eventos in-game	-	40	15	-	-	-
Turnos	-	-	-	-	40	-
Cartas	-	-	25	-	-	-
Total:	**45**	**45**	**45**	**45**	**45**	**45**

La unidad de medida son las horas de trabajo.

Fecha	Gonzalo Martínez	Álvaro Vázquez	Luis Rodríguez	Álvaro Miranda	Daniel Díaz	Pedro Parrilla
Sprint 1	19	19	19	19	19	19
Sprint 2	40	40	40	40	40	40
Sprint 3	45	45	45	45	45	45
Sprint 4	45	45	45	45	45	45
Total:	**149**	**149**	**149**	**149**	**149**	**149**

La unidad de medida son las horas de trabajo.

ANEXO 1.
RETROSPECTIVA SPRINT 1

Conseguir los requisitos del nivel señalado. Justificación.

El grupo ha trabajado conjuntamente en el principio del sprint para unificar puntos de vista y organizar la forma de trabajo. Cuando teníamos claro cómo distribuir la carga del proyecto, empezamos a trabajar en paralelo, manteniendo siempre un alto nivel de comunicación tanto para cosas puntuales vía mensajería de texto, o reuniones de grupo y trabajos conjuntos en parejas vía videollamadas.

La descripción del proyecto y los roles los pensamos y desarrollamos completamente en grupo. La pareja formada por Álvaro Vázquez y Gonzalo empezaron especificando las primeras historias de usuarios, completándose a continuación por la pareja de Luis y Álvaro Miranda para terminar con el resto de historias de usuario.

El modelo de datos se empezó planificando en grupo en la propia clase de prácticas. La pareja de Pedro y Daniel, continuaron diseñándolo; tras varias consultas con nuestro tutor del proyecto, se corrigió y terminó por la pareja de Pedro y Álvaro Miranda, con ayuda adicional de Daniel.

La primera iteración de trabajo en las reglas de negocio fue realizada por Daniel y Gonzalo, más tarde se terminaron por Luis y Álvaro Vázquez. Los mockups fueron realizados primeramente por Álvaro Vázquez y Gonzalo, con especial comunicación con Luis y Álvaro Miranda, quienes realizaban las historias de usuario que se iban a diseñar en los mockups. Se añadieron las últimas pantallas por Gonzalo y Álvaro Miranda. La entidad, servicio y controlador se implementaron por Luis y Daniel.

Finalmente, todos reunidos revisamos, debatimos y perfeccionamos cada apartado y aspecto.

Análisis retrospectivo del sprint. Reflexión.

General

El sprint comenzó con una organización ciertamente caótica, con la acumulación de trabajos de otras asignaturas era difícil coordinar en primera instancia a todo el grupo. Luego nos subdividimos en grupos de 2 intentando dividir la carga equitativamente utilizando las herramientas de diagrama de Gantt. Esta sección del sprint ha sido más organizada, aunque atropellada ya que no disponíamos de todo el tiempo que queríamos para terminar todo por la carga de trabajo.

El sprint ha supuesto en líneas generales mucho esfuerzo por parte de todos los integrantes, al elegir el juego “complejo” era de esperar, pero el tiempo dedicado a pensar, diseñar y revisar ha supuesto mucha carga para todos.

De todas formas, esta complejidad y el hecho de ver cómo van tomando forma todos los apartados del proyecto nos motiva cada vez más a continuar con el mismo.

Álvaro Vázquez - Gonzalo Martínez: Mockups.

Para empezar, creamos un proyecto de MarvelApp. Posteriormente, buscamos recursos que nos pudieran servir para el diseño del juego, como pergaminos, fichas del juego, marcos, cartas, etc. Paralelamente mientras se diseñaban las historias de usuario, trabajábamos conforme a lo detallado. Hubo problemas con esto, ya que si se cambiaban las historias de usuario, tendríamos que cambiar nosotros nuestro diseño. A medida que se van detallando las historias de usuario, realizamos la versión final del mockup. Por último hicimos las capturas de cada pantalla y las integramos en el documento.

Luis Rodríguez - Álvaro Miranda: Historias de Usuario.

Al principio, íbamos desarrollando las historias de usuario de forma lenta, ya que debíamos debatir primero cómo queríamos implementar cada funcionalidad, hablarlo con Gonzalo y Álvaro Vázquez para contemplar el diseño del mockup, y finalmente, redactar la historia de usuario. Posteriormente obtuvimos feedback de nuestro tutor de proyecto, que nos comentó lo que debíamos cambiar y nos aconsejó cómo continuar, como por ejemplo, ser más concreto en los datos introducidos por el usuario en cada escenario.

Una vez corregidas las historias de usuario realizadas previamente, ya cogimos soltura con las historias, nos comunicábamos constantemente con Gonzalo y Álvaro para terminarlas. Consideramos que la comunicación, tanto entre nosotros como con Álvaro y Gonzalo ha sido la clave del desarrollo y lo que nos ha permitido avanzar, por lo que estamos bastante satisfechos.

Daniel Díaz - Pedro Parrilla: Modelo Conceptual.

Empezamos sin una plantilla o modelo a seguir de forma que íbamos un poco “a tientas” al principio de forma que confundimos el modelo de datos con el diagrama de clases y en base a nuestra experiencia anterior, comenzamos modelando como se hace para BBDD en vez de como se hace en Java con Spring.

Gracias a los comentarios del profesor sobre los prototipos del diagrama, logramos reconducir el modelo hasta una versión final satisfactoria. La complejidad del proyecto y el nivel de abstracción de este primer sprint han supuesto la dificultad añadida de profundizar lo máximo posible para generar un modelo relativamente fiable a lo que vamos a construir. Esta profundización era complicada de alcanzar debido a que, como no son elementos que necesitamos implementar en el proyecto inmediatamente sino en un futuro, dificulta bastante la especificación de los atributos, entidades y relaciones de las mismas.

Luis Rodríguez - Daniel Díaz: Entidad, repositorio, servicio y vistas.

Al plantear esta actividad donde teníamos que crear una lista de una entidad y poder editar los elementos, optamos por usar la entidad User, ya que en un primer momento parecía la entidad que más nos podía interesar listar de nuestro sistema.

Siguiendo los vídeos de EV y tomando cómo referencia la clase Owner fue muy sencillo implementar los cambios, sin embargo hubieron un par de dificultades:
En un primer lugar queríamos editar la clase User de Spring para que fuese idéntica a la de nuestro planteamiento inicial, pero no pudo ser posible porque Spring Security espera que la clase User tenga exactamente 3 parámetros, por lo que al cambiarlo no dejaría hacer login con los usuarios.

El otro inconveniente vino al intentar editar los nombres de los usuarios, ya que la clase User dada por Spring no hereda de BaseEntity por lo que no tiene id y username es su primarykey. Finalmente sólo fue posible editar la contraseña, y se planteó como posible solución crear una clase heredada de User como clase principal para el segundo Spring.
Al hacer las ediciones sobre User se cambió además el formulario del Login, que ahora registra correctamente a un User con su respectivo Auth con la cadena "player", sin crear un Owner asociado.

Y finalmente se implementó un buscador junto a la lista de Users, y la posibilidad de editar usuarios de la lista sólo está disponible para un login con credenciales de Admin.

Fecha	Gonzalo Martínez	Álvaro Vázquez	Luis Rodríguez	Álvaro Miranda	Daniel Díaz	Pedro Parrilla
15/10/2021	2	2	2	2	2	2
19/10/2021	2	2	-	-	1	1
20/10/2021	3	3	3	2	2	2
24/10/2021	2	2	4	5	5	5
26/10/2021	7	7	3	3	4	4
27/10/2021	1	1	1	1	1	1
28/10/2021	2	2	6	6	4	4
Total:	**19**	**19**	**19**	**19**	**19**	**19**

La unidad de medida son las horas de trabajo.

ANEXO 2.

RETROSPECTIVA SPRINT 2

Conseguir los requisitos del nivel señalado. Justificación.

La planificación de esta entrega se recoge todas las historias de usuario que permiten realizar al user-player todas las acciones previas a una partida:

- Registro, inicio sesión y bases.
- Perfiles de usuario, invitación de amigo y gestión de invitaciones.
- Crear un lobby previo al matchmaking, invitación de cualquier usuario al lobby, invitación de amigos al lobby y gestión de invitaciones al lobby.
- Matchmaking con websocket que creara *Búsquedas de lobby* o *LobbySearch* donde los diferentes lobbies de diferentes usuarios puedan combinarse de manera condicionada.
- Creación de jugadores desde cada usuario del lobby, y creación de partida
- Creación de tableros de Acción, Guerra y Estados del a ciudad

Registro, inicio de sesión y bases.

Antes de trabajar cada vista individual, se decidió realizar un diseño base que se compartiera entre todas estas y que permitiera tener componentes comunes que luego ayudaran con el responsive o a la reutilización. Esta parte incluye la vista welcome, que según el usuario hubiera iniciado sesión o no, mostraría botones para iniciar sesión/registrarse, o empezar partida o entrar en las futuras estadísticas del perfil. También se trabajó en la estructura de vista común, que ha sido utilizada por en todo el proyecto por las diferentes vistas de cada grupo de entidad-controlador-vista-repositorio.

La funcionalidad de inicio de sesión se basa en el funcionamiento que trae el ejemplo de Petclinic de Spring. Declarado en el archivo de configuración de Spring, se permite el uso del controlador de inicio que usa los token csrf. Esta parte se estudió por parte de un subgrupo para conseguir integrarlo en una vista no predeterminada por spring, pero tuvo que cesar esa parte del trabajo debido a la gran inversión de tiempo que estaba conllevando. El registro se realiza en el modelo de usuarios. Se crea un nuevo controlador, y se añaden nuevos métodos al servicio que utilizan el repositorio para guardar y recoger los usuarios. Además se realiza una vista customizada de formulario de registro.

Perfiles de usuario, invitación de amigo y gestión de invitaciones

Los perfiles de usuario utilizan la base que venía con el ejemplo de spring, pero se modifican sus funcionalidades, la vista y se integran nuevos métodos para que un usuario que entra en un perfil que no es suyo, pueda enviar petición de amistad o aceptarla. También se tiene gestión de invitaciones a partida o de amistades desde el menú lateral.

Crear un lobby previo al matchmaking, invitación de cualquier usuario al lobby, invitación de amigos al lobby y gestión de invitaciones al lobby

Para empezar una partida, se requiere que un grupo de jugadores esté reunido. Se ofrece el modelo de Lobby para esta tarea. Cuando un usuario hace click en crear partida, realmente crea un nuevo lobby donde puede invitar a cualquier usuario o usuarios amigos desde el menú lateral. Un usuario puede aceptar invitación de lobby desde el mismo menú lateral, o desde la pantalla de welcome habiendo iniciado sesión.

Matchmaking con websocket que creará *Búsquedas de lobby* o *LobbySearch* donde los diferentes lobbies de diferentes usuarios puedan combinarse de manera condicionada

Una vez que desde el lobby se busca partida, se crea en memoria una búsqueda de lobby (*LobbySearch*) que empieza partida si está *satisfecha* o busca combinarse con otros lobbies si se encuentran dentro de su misma condición. Todos los usuarios dentro de lobby están escuchando el canal del websocket para escuchar y recoger las modificaciones de su propio lobby. Una partida se considera satisfecha (*satisfy*) si la cantidad de usuarios restantes menos la cantidad de usuarios que ya hay en la búsqueda es igual a 0. Dado este caso, se crea una lista de jugadores (*Players*) por cada usuario (User) del lobby, y se crea una partida que relaciona estos jugadores y los diferentes tableros.

Creación de jugadores desde cada usuario del lobby, y creación de partida

Tal y como se detalla en el anterior punto, es el matchmaking el encargado de crear una partida con los jugadores. No era objetivo de esta planificación integrar las funcionalidades dentro de la partida con los tableros, turnos y jugadores, pero se deja creada y lista para que puedan realizar las acciones correspondientes de las partidas y del resto de historia de usuario en la partida.

Creación de tableros de Acción, Guerra y Estados de la ciudad

Cuando se crea la partida, se crean los 3 tableros de la partida. Estos van ligados directamente a la partida e individualmente tienen una interfaz, algunos en canvas, otros

directamente en HTML y Javascript, dentro de una estructura común realizada dentro de un tag de layout exclusivo para las partidas.

Análisis retrospectivo del sprint. Reflexión.

General

La mayor dificultad encontrada en este segundo sprint la hemos encontrado en trabajar en Spring, que nos ponía ciertas limitaciones con algunas funcionalidades, haciéndonos emplear algo de tiempo vanamente.
A pesar de seguir teniendo problemas en este ámbito, hemos mejorado nuestra rendimiento con respecto al primer sprint, con la organización por parejas jugando un factor clave en esto.
Finalmente consideramos que hemos conseguido un resultado satisfactorio, hemos estado a la altura de nuestra planificación, y nos encontramos motivados a seguir con la planificación del tercer sprint.

Álvaro Vázquez - Gonzalo Martínez: Pantalla de inicio, Log In, registrarse y tablero de guerra.

En el inicio del sprint nos dedicamos ambos a diseñar en css y jsp la pantalla de inicio. Trás partir del diseño base, lo adaptamos a la pantalla de iniciar sesión y registrarse siendo ambos responsive para todo tipo de dispositivos. Esto nos dio bastantes problemas, los cuales aún persisten. Investigar cómo funciona el procesamiento de iniciar sesión en spring añadiendo una vista personalizada, nos llevó demasiado tiempo y no acabó de funcionar.

Tras acudir a la revisión el viernes, se nos indicó que no era importante, así que a partir de ese momento rectificamos, priorizamos tareas y nos dedicamos a la parte del tablero de guerra. Para ello, creamos controladores, las entidades, servicios y repositorios pertinentes así como la vista del propio siendo también responsive para todos dispositivos. Esta última parte fué la más costosa a implementar debido a dos problemas que surgieron durante la programación de los mismos, el primero fue el orden temporal en el que se ejecutaban las funciones definidas en los ".tag" lo que provocó que o bien el tablero o las fichas del mismo no se imprimiesen en su debido orden (se dibujaban en el orden en el que se cargasen las imágenes en el navegador). El segundo problema que surgió fue la mezcla de lenguajes (JSP y JavaScript) durante la programación de la integración de las fichas en el tablero, ya que los bucles se comportaban de manera independiente en cada lenguaje.

Luis Rodríguez - Pedro Parrilla: Módulo social, panel de navegación lateral y aceptación de partidas.

Primeramente, nos dedicamos a la creación de la tabla "Amigos" y a desarrollar el servicio asociado (sin controlador debido a que los amigos son mostrados en el panel lateral y en los

detalles de los usuarios). El principal problema era a la hora de diseñar la tabla ya que las relaciones de amistad son bidireccionales (X es amigo de Y tanto como Y es amigo de X) pero las peticiones no (la petición que envía X a Y, no es una petición de Y a X) de forma que teníamos que decidir entre unificar toda la “Amistad” en una tabla o tablas distintas de amistad y petición. Optamos por solo una tabla de “Amistad” en la que con un parámetro booleano detectamos si es una petición de amistad y por otro lado según quién envía la petición de amistad, se detecta si es el origen o el destino de la petición. Una vez aceptada, el parámetro booleano cambia y los usuarios pertenecientes a esta relación siguen siendo “Origen” y “Destino”, aunque ya no importe el orden.

Las querys eran más complicadas en esta tabla debido a la bidireccionalidad pero una vez que se definieron bien, no hubo mayor problema.

Para la barra lateral, no hubo mayor problema, simplemente los módulos que se habían ido añadiendo y añadir un menú de navegación para hacer todo más cómodo ya que principalmente para movernos por la aplicación usábamos directamente la uri.

Daniel Díaz - Álvaro Miranda: Lobby, invitaciones de partida, matchmaking y tablero de acciones.

En primer lugar, nos pusimos con el lobby. Empezamos enlazándolo con el botón de jugar del “welcome”, y a partir de ahí, fuimos construyéndolo. Configuramos las opciones por defecto y añadimos el panel para modificarlas, (el cual solo podía editarlo el administrador del lobby) y una lista de los jugadores del lobby.

Cuando terminamos la base, implementamos la opción de invitar a otros usuarios. Al trabajar en paralelo, teníamos que esperar a que se terminara el módulo social para probar las invitaciones a usuarios, por lo que probamos primero a invitar a un usuario cualquiera escribiendo su nombre, y una vez se terminó el módulo social, implementamos las invitaciones a amigos. Mientras Álvaro implementó dichas invitaciones, Daniel desarrolló el matchmaking para poder empezar partida una vez se llegara a los jugadores configurados. Finalmente, Daniel desarrolló el prototipo y funcionalidades básicas del tablero de acciones, siendo este la interfaz al comenzar partida.

Tipo	Gonzalo Martínez	Álvaro Vázquez	Luis Rodríguez	Álvaro Miranda	Daniel Díaz	Pedro Parrilla
Registro y login, pantalla inicio	23	23	6	-	-	-
Lobby	-	-	-	22	12	-
Amistad	-	-	18	15	-	19
Usuarios	-	-	16	-	-	13
Panel social	-	-	-	3	-	8
Matchmaking	-	-	-	-	19	-
Tableros	17	17	-	-	9	-
Total:	**40**	**40**	**40**	**40**	**40**	**40**

La unidad de medida son las horas de trabajo.

ANEXO 3.
RETROSPECTIVA SPRINT 3

Conseguir los requisitos del nivel señalado. Justificación.

La planificación de esta entrega recoge:

- Primera mitad de una partida (primera fase completa, segunda y tercera en desarrollo)
- Módulo de estadísticas
- Funciones del administrador
- Implementación de patrones de diseño
- Depuración de código y correcciones varias

Análisis retrospectivo del sprint. Reflexión.

General

En este tercer sprint, nos hemos dedicado a avanzar en gran medida el desarrollo de una partida, creando un sistema de turnos funcional y su conexión entre el juego en directo y en la base de datos; así como completar aspectos del entorno de la aplicación como el módulo estadístico y la funcionalidad de los administradores. Además, se fue implementando paulatinamente el patrón de diseño “estados”.

Generalmente, hemos sabido organizarnos pese a la carga del resto de asignaturas y a estar en periodo de Navidades. Destacamos el tiempo empleado en solucionar errores que no comprendíamos y depurar el código existente para poder avanzar.

Con la dinámica de las nuevas implementaciones, se busca terminar de probar y desarrollar las últimas historias de usuario para la entrega final.

Daniel Díaz - Pedro Parrilla: Turnos con fases

Dado que ya teníamos implementada una primera versión de interfaz con websockets y servidor, y con la sugerencia de meter patrón estado a las fases del turno, creamos dos líneas de desarrollo conjuntas buscando poder seguir avanzar con las fases del turno al mismo tiempo que se diseñan las mismas fases con la estructura de patrón de estado.

Dentro de las diferentes fases, se diseñan las diferentes interfaces que van llamándose para cada fase del turno. Estas mandan la información de cada turno en específico y el servidor

lo interpreta para registrar, procesar y devolver la petición de nueva fase al jugador. La información de los movimientos también es interpretada para mostrar los movimientos en fichas y actualizar los diferentes tableros.

Al mismo tiempo que esto funciona dentro del canal de turnos, se añade un nuevo canal que es llamado cuando un cliente actualiza la página o entra por primera vez la partida, mande la información de la partida actual para que pueda sincronizarse al turno. Una vez queda sincronizada la partida, se cierra la conexión con este canal por el cliente. Esto formula dos canales (turn y gameload) por los que el servidor contacta con el cliente, y el mismo cliente trata de actualizar su interfaz de la misma manera.

El patrón estado se ha adaptado hasta la fase en la que se encuentra la rama, por lo que lo primero que se trabajará es implementarlo para continuar desarrollando las diferentes opciones de fase restantes y restringir el uso de los turnos de cara a la entrega final.

La versión presentada no incluye el patrón ya que no incluía todas las fases que se habían desarrollado hasta el momento. Todo debido a que tuvimos que trabajar en paralelo para no estancarnos ambos en algún error que no nos dejase avanzar, como por ejemplo que la tecnología hibernate no admite interfaces de Java como forma de tipado, no como en Java vanilla que se puede abstraer un tipo como instanciación de un objeto que implemente la interfaz.
Esto hizo que el tratamiento de las fases se hiciera en memoria y no se guardara en el servidor, provocando así diferencias con la implementación sin el patrón.
Por otro lado, las fases del juego no están ordenadas en ciertos momentos sino que quedan a elección del jugador de forma que el patrón estado no puede saber qué estado será el siguiente, esto genera ciertos problemas y diferencias con la implementación habitual del patrón de diseño las cuáles estamos trabajando para solventar de la mejor manera posible.

Álvaro Vázquez - Luis Rodríguez: Tableros, fichas (movimiento tanto en servidor como en interfaz), modificación de entidades como GameBoard y GamePiece, interacciones de monedas, panel de errores y sus respectivas pruebas.

En el inicio del sprint nos dedicamos a hacer que las fichas y tableros se visualizarán correctamente en la interfaz, la primera idea era hacer que la base de datos estuviese perfectamente conectado a la interfaz del juego, a lo largo del desarrollo, vimos que era inviable, pues el modelo que se recibe en la vista solo se carga al cargar la página. Por ello no se podía tener en cuenta en la interfaz los datos en tiempo real. Planteamos una solución y fue que por websocket se mandasen los datos, tanto en el servidor, para que se modificase la base de datos, como en la interfaz, en la que se iban a cargar los datos y modificar de manera local en el dispositivo de cada usuario. También se ha hecho limpieza en todo lo que se guardaba en la base de datos, que con la nueva estructura no hacía falta almacenar (por ejemplo las listas de posiciones X e Y de las fichas). Añadimos también las interacciones de las monedas en el juego con sus respectivas restricciones por lo que fue necesario modificar cómo funcionaban las fases. Una vez terminamos estas funciones añadimos un notificador de errores en las vistas de register y edición de usuarios.

Gonzalo Martínez - Álvaro Miranda: Módulo de estadísticas, funciones del administrador.

Al inicio del sprint, discutimos la apariencia de la vista del módulo de estadísticas con el resto compañeros a la vez que las estadísticas a calcular; tras la aprobación de todos, Gonzalo empezó a programar la vista en jsp, mientras que Álvaro empezó con las funciones para calcular las estadísticas. Tras varios errores que se fueron solucionando, ambos conseguimos terminar con la vista del módulo pero la parte funcional no pudimos terminarla por completo por falta de tiempo, quedando así únicamente sin ser funcional el apartado de logros.

Para implementar las funciones de administración de la aplicación, modificamos la interfaz y añadimos funcionalidades: la opción de crear un usuario nuevo en la pantalla de buscar usuarios, editar un usuario accediendo a su perfil (en nuestro caso, modificar la contraseña) y la posibilidad de borrar el usuario, también desde su perfil. El borrado de usuarios nos llevó bastante tiempo y fue un quebradero de cabeza porque salían bastantes errores y no dejaba borrarlos. Tras profundizar e ir arreglando errores, tuvimos que reestructurar algunas entidades para permitir el borrado en cascada al eliminar un usuario.

Tipo	Gonzalo Martínez	Álvaro Vázquez	Luis Rodríguez	Álvaro Miranda	Daniel Díaz	Pedro Parrilla
Tableros y fichas (interfaz en funcionamiento)	10	29	17	5	15	1
Monedas	-	8	19	-	-	-
Estadísticas	25	2	-	14	-	1
Administración	6	-	3	20	-	-
Turnos	-	-	-	-	30	38
Corrección de errores	4	6	6	6	-	5
Total:	**45**	**45**	**45**	**45**	**45**	**45**

La unidad de medida son las horas de trabajo.