**DISEÑO EN JAVA EL VIDEOJUEGO SCRABBLE FUNCIONAMIENTO DEL EQUIPO SCRUM**

1. **INTRODUCCIÓN**

Este proyecto, asociado a la asignatura *Ingeniería de Software II* en la Universidad Complutense de Madrid, consiste en la implementación ligeramente simplificada del juego Scrabble, a través de la metodología ágil Scrum.

En este documento se puede encontrar la documentación correspondiente a cada Sprint de dos semanas llevados a cabo durante el cuatrimestre (desde enero hasta mayo de 2022).

El equipo Scrum está formado por Marta Vicente Navarro, Pablo García López, Diego Ostos Arellano, Pablo Navarro Cebrián y Pablo Zamora Morales.

El cliente al que está dirigido es, en general, cualquier persona con un ordenador para poder jugar al juego.

1. **ESTRUCTURA Y FUNCIONAMIENTO DEL EQUIPO SCRUM**

El proyecto ha sido desarrollado por un equipo que se ha adaptado al modelo Scrum. La organización ha sido la siguiente:

*Scrum Master:* Marta Vicente Navarro.

*ProductOwner:* Diego Ostos Arellano.

*Equipo de desarrollo:* Pablo García López, Pablo Navarro Cebrián y Pablo Zamora Morales.

Para organizar el trabajo en equipo el equipo se ha organizado de la siguiente manera. En primer lugar, cada semana el equipo elegía las historias de usuario a implementar en el siguiente sprint indicándolas en el *Sprint Planning.* A lo largo del Sprint se han realizado diferentes reuniones para hablar sobre la evolución del proyecto, comentar las dudas e ideas.

Estas reuniones, conocidas como *Daily Scrums* han tenido especial importancia en la coordinación del trabajo. Repartir las tareas y comentar lo que cada miembro ha podido implementar y no, ha permitido que nuestro trabajo sea uniforme, y que no se solapen las tareas. Estas reuniones se han llevado a cabo entre clases o utilizando la herramienta Google Meet cuando no podían realizarse de forma presencial.

Cabe destacar la importancia de las clases prácticas de la asignatura, que han supuesto un impulso importante a cada *sprint* del proyecto gracias a estar todos juntos trabajando en equipo y a tener la posibilidad de consultar dudas. También ha sido muy importante el trabajo individual de todos los miembros del equipo en casa, sin el que no hubiera sido posible sacar adelante el proyecto.

Para planear la distribución del trabajo entre los diferentes *sprints* y entre los diferentes miembros del equipo en cada sprint, se han tenido en cuenta diferentes factores. Por un lado, cuales son los puntos fuertes de cada uno. Por otro, la carga individual de cada uno, así como la disponibilidad de tiempo en función del trabajo de otras asignaturas.

Para llevar a cabo el proyecto se han usado diferentes herramientas. Por un lado eclipse para la implementación del código. *Github* para llevar a cabo un control de las versiones, compartir el proyecto entre los diferentes miembros y la documentación. También se han usado herramientas como *Modelio* e *IntelliJ* para el diseño UML.

A lo largo de los *sprints* el equipo ha ido evolucionando, siendo cada vez más eficiente y haciendo crecer el proyecto de manera exponencial.

1. **TRABAJO REALIZADO POR CADA MIEMBRO DEL EQUIPO SCRUM**

A lo largo del proyecto, diferentes tareas han sido realizadas por diferentes miembros del equipo. Se ha intentado repartir las tareas de forma homogénea, aunque algunos miembros han destacado especialmente en algunas tareas.

En primer lugar, el equipo de desarrollo.

Pablo García Lopéz ha tenido un papel fundamental en la coordinación y comunicación del grupo. Cabe resaltar su participación activa en las reuniones y su papel de apoyo a todos los miembros del equipo. Destaca su versatilidad puesto que ha participado activamente en el desarrollo del código del proyecto, pero también en el desarrollo de diagramas UML. Podemos mencionar también su papel a la hora de entregar los diferentes Sprints del proyecto. Se ha encargado de realizar las *releases* de cada Sprint así como de las revisiones previas de todo el material asociado al Sprint. Controlando así la evolución del proyecto: los nuevos contenidos de la wiki del proyecto en GitHub, el avance del código y la cohesión de todo el trabajo.

En segundo lugar, Pablo Navarro Cebrián. Importante miembro del equipo de desarrollo. Durante el transcurso de los Sprints su papel en el proyecto ha sido crucial. Cabe destacar su importancia en el desarrollo del código del proyecto donde sin duda ha sido el cabeza de equipo. Por un lado, siendo el principal generador de código, así como la principal ayuda al resto de miembros en este aspecto. Su trabajo ha sido imprescindible para alcanzar los objetivos de cada Sprint. También ha participado en las actividades de documentación de los Sprints, tanto participando en las Daily Scrums como redactando el material que reúne el contenido de las mismas. Así mismo, ha sido una pieza fundamental en la toma de decisiones de qué y cómo implementar el juego. Finalmente, también ha participado en la creación del material UML asociado al código.

//falta pablo Zamora

Por otro lado, Diego Ostos Arellano. Su papel como miembro del equipo es el de *Product Owner*. En esta línea, ha sido el encargado de gestionar el Product Backlog, priorizando las tareas, optimizando el valor del trabajo en equipo y decidiendo qué desarrollar y que no. Destaca la participación activa en las reuniones y sesiones de trabajo. Además, ha sido el principal encargado del desarrollo UML del proyecto, encargándose de la creación de los diagramas de clases y secuencias asociados a las historias de usuario de cada Sprint del proyecto. Su papel en el desarrollo UML ha sido crucial pues también se ha ocupado de buscar herramientas que optimizaran y facilitaran la generación del diseño UML, así como de explicar al resto de miembros como usarlas. Por otro lado, también ha participado como apoyo en la creación del código del proyecto y en la redacción de documentación.

Finalmente, Marta Vicente Navarro. Como Scrum Master se ha encargado de gestionar el proceso Scrum, ayudando a encontrar soluciones a los problemas que pudieran retrasar las entregas o dificultar el funcionamiento del equipo, así como la búsqueda de herramientas que facilitaran el trabajo del mismo. Por otro lado, ha sido la principal coordinadora del grupo, repartiendo las tareas, organizando las reuniones y explicando el contenido de las mismas a quienes no pudieran asistir a alguna de ellas. En cuanto al producto, su principal papel ha sido redactar los Sprint Review, Sprint Retrospectives y Sprint Plannings tras las reuniones correspondientes así como del resto de la documentación de la wiki en GitHub. Por otro lado, ha sido una pieza de apoyo tanto en el diseño UML como en la realización del código del juego.

Cuando hagamos JUNIt y lo q hagamos el resto de Sprints ponérselo a quien los haga

1. **HISTORIAS DE USUARIO**

**DE CADA HISTORIA HAY QUE PONER EN QUE SPRINT FUE PLANEADA, IMPLEMENTADA, MODIFICADA, REFACTORIZADA ETC. ASÍ COMO COMENTARIOS ADICIONALES SI FUERAN NECESARIOS**

* 1. **INICIALIZAR A LOS JUGADORES**

Como usuario, quiero que se implemente una versión simplificada del Scrabble que permita inicializar los jugadores, para reducir la dificultad de juego y diseño.

* Estimación: 4 días.
* Priorización: El proyecto debe incluir esta funcionalidad.
* Terminado: Versión primitiva del juego que inicializa a los jugadores.
  1. **MULTIJUGADOR**

Como usuario, quiero que el juego implementado sea multijugador, para permitir partidas entre 2 y 4 jugadores*.*

* Estimación: 1 hora.
* Priorización: Deber tenerlo.
* Terminado: Cualquier partida permite un flujo de juego entre 2 y 4 jugadores.
  1. **PALABRAS VÁLIDAS**

Como usuario, quiero una versión simplificada del Scrabble en la que se comprueben que las palabras introducidas son correctas (excluyendo tildes y palabras con ñ), para añadir un pequeño nivel de dificultad al juego.

* Estimación: 3 días
* Priorización: Debe tenerlo.
* Terminado: Cualquier palabra introducida que no se encuentre en el diccionario de la RAE no se considera correcta.
  1. **CARGAR/GUARDAR**

Como usuario, quiero que el juego pueda ser cargado desde fichero, y también guardado en cualquier momento, para poder reanudar una partida no finalizada.

* Estimación: 8 horas.
* Priorización: Podría tenerlo.
* Terminado: Las partidas se cargan desde un fichero correctamente y se guardan en otro fichero con el formato establecido.

Durante el sprint 3 el nombre de esta historia de usuario fue modificada. Pasó de llamarse *Entrada/Salida* a llamarse *Cargar/Guardar*.

* 1. **JUGADORES AUTOMÁTICOS CON AL MENOS 3 NIVELES DE DIFICULTAD**

Como usuario, quiero que el juego proporcione la opción de introducir jugadores automáticos con 3 niveles distintos de dificultad

* Estimación: 15 días.
* Priorización: Debe tenerlo.
* Terminado: Los jugadores automáticos sean capaces de completar la partida.
  1. **INTERFAZ GRÁFICA**

Como usuario, quiero que el juego proporcione un entorno visual sencillo para que los usuarios puedan interactuar con facilidad y el juego fluya de forma dinámica.

* Estimación: 8 días.
* Priorización: Debe tenerlo.
* Terminado: La interfaz gráfica tenga todo lo necesario para poder cargar, guardar y jugar partidas de forma sencilla.
  1. **JUEGO EN RED**

Como usuario, quiero que se pueda jugar al juego en línea con jugadores desde distintos terminales.

* Estimación: 10 días.
* Priorización: Debe tenerlo.
* Terminado: Se pueda completar una partida en la que participen jugadores desde diferentes terminales.
  1. **PALABRAS ENCADENADAS**

Como usuario, quiero que las palabras introducidas por un jugador deban cortarse con una palabra ya existente en el tablero (a excepción de la primera, que deberá colocarse en posición central)

* Estimación: 5 horas.
* Priorización: Debe tenerlo.
* Terminado: No se permite introducir una palabra en una posición del tablero que no cumpla los requisitos descritos.

En el Sprint 2 se añadió la historio de usuario “*Palabras encadenadas”.* Durante el Sprint 3 fue eliminada puesto que el Equipo Scrum consideró que debía ser eliminada pues no es una historia de usuario si no una norma.

1. **SPRINT REVIEWS**

**4.1 SPRINT 1**

Durante el desarrollo de este primer sprint, las historias de usuario se desarrollaron de forma óptima acorde a los criterios previamente estipulados por el Equipo Scrum, respetando las priorizaciones y criterios de “Terminado”.

Sin embargo, el mayor inconveniente que hemos enfrentado ha sido la gestión de los tiempos de estimación acordados, pues si bien en un inicio parecían razonables, hemos pasado las últimas horas del sprint a contrarreloj para documentar todo el sprint.

**4.2 SPRINT 2**

Durante el desarrollo de este segundo sprint, hemos logrado implementar todas las historias de usuario que nos habíamos propuesto, y también hemos conseguido completar el diseño UML del código.

Además, durante este sprint no hemos encontrado ningún inconveniente grave, más allá de aprender a usar el software para desarrollo UML *Modelio*, que no ha supuesto ningún retraso ni problema para el Equipo Scrum.

Se espera que para próximos Sprints, esta siga siendo la manera de proceder.

REFACTORIZACIONES Y COMENTARIOS

En este sprint se hizo una refactorización del código del sprint anterior. En primer lugar, se movió la entrada por teclado a la clase *Controller,* para que este sea el único sitio de entrada por teclado. Además, se crearon los comandos de juego y excepciones. Por último, quitamos el *GamePrinter* de *Game* para que sea el controlador el que muestra los mensajes de juego, y así, que en *Game* sólo se maneje la lógica del juego.

**4.3 SPRINT 3**

Durante el desarrollo de este tercer sprint hemos logrado terminar todas las historias planeadas. Además, hemos conseguido completar el diseño UML del código. Hemos refactorizado parte del código asociado a historias de usuarios de Sprints anteriores, haciendo las modificaciones correspondientes en los diagramas UML.

Además, durante el sprint no hemos encontrado ningún inconveniente grave, aunque el Equipo de Desarrollo ha tenido que esforzarse más de lo normal para completar la implementación y diseño del código relacionado con la entrada. Como se comentó en el Sprint *Retrospective*, el Equipo de Desarrollo deberá tener cuidado a la hora de implementar en el último momento historias de usuario ajenas al sprint actual.

Por otro lado, hemos tenido algunos problemas a la hora de trabajar en paralelo en los diagramas UML debido a las limitaciones del software *Modelio*. No obstante, estos problemas no han supuesto un inconveniente grave a la hora de trabajar.

Para los próximos sprints se espera mejorar respecto a estos problemas puesto que han retrasado al equipo en algunos aspectos.

En el Sprint 2 se añadió la historia de usuario *“Palabras encadenadas”*. Durante el Sprint 3 fue eliminada puesto que el Equipo Scrum consideró que debía ser eliminada pues no era una historia de usuario si no una norma.

La historia de usuario *entrada/salida* pasa a llamarse historia de usuario *cargar/guardar.*

En este sprint, a parte de las historias de usuario propias, se ha llevado a cabo una refactorización del código. Por un lado, se ha reunido toda la lógica del juego en la clase *GameLoader* dejando sin lógica la clase *Controller.*

Por otro lado, se ha mejorado la arquitectura del código para hacerlo más sencillo, eficiente y claro. De esta forma, las modificaciones que se hagan en el futuro podrán ser implementadas más fácilmente.

**4.4 SPRINT 4**

Durante el desarrollo de este tercer sprint, hemos logrado terminar todas las historias planeadas. Además, hemos conseguido completar el diseño UML del código. Hemos refactorizado parte del código asociado a historias de usuarios de Sprints anteriores, haciendo las modificaciones correspondientes en los diagramas UML.

Durante este Sprint hemos unificado la forma de hacer el diseño UML. El diagrama de clases de todo el proyecto asociado al Sprint lo hacemos utilizando la herramienta *Modelio*, aplicando un reverse del proyecto. El diagrama de clases asociado a la historia de usuario implementada en el sprint, así como los diagramas de secuencias los hacemos mediante el plugging de eclipse *Plant* UML.

En cuanto al código, de cara la siguiente sprint donde implementaremos la historia de usuario de la interfaz gráfica, se han llevado a cabo importantes refactorizaciones. El objetivo principal es facilitar la implementación de la historia de usuario del al interfaz gráfica.

**4.5 SPRINT 5**

1. **SPRINT RETROSPECTIVE** 
   1. **SPRINT 1**

**1.1.1 Start doing**

El equipo está de acuerdo en que debe haber mayor comunicación entre todos, para ser conscientes de todos los cambios producidos. Por ejemplo, alguna vez ha ocurrido que alguien no ha podido hacer un “commit” porque no había un “pull” de código escrito anteriormente por alguien que no había avisado de ello.

**1.1.2 More of**

Algo que ha funcionado bien son las *DailyScrum.* Han sido especialmente útiles para asignar el trabajo y coordinar a los diferentes miembros del equipo.

**1.1.3. Keep doing**

La implementación del código ha sido especialmente eficiente, y se espera que así siga en los próximos sprints.

**1.1.4. Less of**

De cara a los siguientes sprints, el Equipo Scrum está de acuerdo en que se debe establecer un formato general para la documentación del proyecto de manera que esta tenga apariencia uniforme.

**1.1.5 Stop Doing**

Un fallo grave del Equipo Scrum, y en particular del Equipo de Desarrollo, ha sido implementar partes del producto sin consultar previamente las historias de usuario del Sprint *Backlog*, y teniendo que modificar este para poder entregar un incremento coherente. También se han encontrado varios defectos gramaticales y de coherencia en la documentación.

* 1. **SPRINT 2**
     1. **Startdoing**

A la hora de programar, deberíamos emplear comentarios para describir qué hace el código en puntos complejos de la lógica, y el por qué se ha realizado de esa determinada forma si hubiera más opciones para ello.

* + 1. **More of**

Es necesario mantener una mayor comunicación entre el equipo para asegurarnos de que la documentación e implementación del proyecto no contienen fallos como duplicación de contenidos.

**1.1.3. Keep doing**

Las historias de usuario escogidas han resultado convenientes para continuar programando el código del proyecto.

**1.1.4. Less of**

Nuestro reparto de roles resulta algo heterogéneo. Aunque todos trabajamos por igual en cuanto a cantidad y esfuerzo, todos deberíamos trabajar temporalmente en las distintas partes del proyecto (programación, documentación….)

**1.1.5 Stop Doing**

El Equipo de Desarrollo debe tener cuidado con la organización del código (arquitectura). Antes de programar, hay que pensar en esto, para evitar costosas refactorizaciones.

* 1. **SPRINT 3**
     1. **Start doing**

Tenemos que empezar a explotar toda la funcionalidad del software *Modelio*. En concreto, debemos aprender a generar los diagramas de clases directamente a partir de código de GitHub, y no manualmente como hacíamos hasta ahora, para ser más eficientes.

* + 1. **More of**

Teniendo en cuenta las mejoras del Sprint 2 respecto del Sprint 1, tenemos que seguir haciendo hincapié en una buena comunicación entre los diferentes miembros del equipo. Además también aprovechamos al máximo las clases prácticas, tanto para trabajar como para preguntar al profesor.

**1.1.3. Keep doing**

Como equipo Scrum, tenemos un buen ritmo de trabajo. El código avanza y mejora constantemente. Además, lo hace de forma coherente con el avance de los programas UML.

**1.1.4. Less of**

En línea con los aspectos a mejorar de semanas anteriores, tenemos que seguir haciendo hincapíe en la organización del código (arquitectura) para evitar refactorizaciones evitables.

**1.1.5 Stop Doing**

Tenemos que dejar de hacer refactorizaciones erróneas que supongan nuevos fallos y errores en el código. De cara a siguientes sprints hay que mejorar la forma de hacer estas modificaciones en el código. Para ello, hemos de pensar bien qué aspectos queremos mejorar, cómo vamos a hacerlo y, siendo cuidadosos con no perjudicar el resto de códigos.

Un error grave por parte del Equipo de Desarrollo ha sido la implementación en el proyecto de parte de una historia de usuario planeada para sprints posteriores (jugadores automáticos), lo que ha provocado que por falta de tiempo no se finalice, y no se pudiese garantizar al final del sprint un producto funcional para el cliente.

* 1. **SPRINT 4**
     1. **Start doing**

A lo largo del proyecto, el Equipo Scrum ha mejorado en muchos aspectos. Llegados a este punto, el equipo trabaja de forma eficiente y con un objetivo común. De esta forma, no consideramos que haya nuevos aspectos a trabajar, si no aspectos a mejorar como equipo.

* + 1. **More of**

Un aspecto en el que sigue habiendo cierto margen de mejora es en la organización e implicación de los miembros del equipo en los *DailyScrum*. Darles la prioridad e importancia que merecen y optimizar al máximo el tiempo invertido en ellos.

**1.1.3. Keep doing**

Los aspectos que mejor han funcionado como Equipo Scrum en este Sprint son la coordinación entre los diferentes miembros del equipo y el reparto de las tareas. Como equipo Scrum consideramos que tenemos que seguir trabajando de esa forma que impulsa el equipo hacia un mejor proyecto.

**1.1.4. Less of**

Algo que tenemos que mejorar como equipo es la coordinación. Ha mejorado desde sprints anteriores pero todavía hay algunos aspectos que se solapan a veces por falta de comunicación. Por ejemplo, a veces varios miembros hacen el diseño UML de una misma clase.

**1.1.5 Stop Doing**

En este sprint ha habido una mejora significativa respecto a otros sprints anteriores. Por tanto, pese a que si hay margen de mejora y aspectos que cambiar como equipo, no consideramos que haya algún aspecto que haya que dejar de hacer, si no sólo mejorar.

1. **SPRINT PLANNING**
   1. **SPRINT 1**

**Historias de usuario planeadas para el Sprint actual:**

1. Como usuario, quiero que se implemente una versión simplificada del Scrabble que permita inicializar los jugadores, para reducir la dificultad de juego y diseño.

* Estimación: 4 días.
* Priorización: El proyecto debe incluir esta funcionalidad.
* Terminado: versión primitiva del juego que inicializa a los jugadores.

Detalles: La aplicación pregunta al usuario el número de jugadores de la partida, y pregunta por los nombres de ellos. También establece, de forma aleatoria, el orden de juego de la partida, y se informa a los jugadores. La aplicación carga además las fichas e inicializa las casillas del tablero.

1. Como usuario, quiero que el juego implementado sea multijugador, para permitir partidas entre 2 y 4 jugadores

* Estimación: 1 hora.
* Priorización: Debe tenerlo.
* Terminado: Cualquier partida permite un flujo de juego entre 2 y 4 jugadores.

Detalles: La aplicación se asegura de que el número de jugadores es un número entre 2 y 4 (inclusive), y también se asegura de que ninguno repite nombre.

Para llevar a cabo el trabajo necesario para entregar este incremento, los diferentes miembros del equipo trabajarán un mínimo de una hora diaria en el proyecto además de una participación activa en los *DailyScrum.*

Adicionalmente, se harán puestas en común y se avanzará con el trabajo en las horas de clase, especialmente las de laboratorio. De cara a mejorar en los siguientes sprints también se llevarán a cabo un Sprint *Review* y Sprint *Retrospective.*

* 1. **SPRINT 2**

**Historias de usuario planeadas para el Sprint actual:**

1. Como usuario, quiero una versión simplificada del Scrabble en la que se compruebe que las palabras introducidas son correctas (excluyendo tildes y palabras con ñ), para añadir un pequeño nivel de dificultad al juego.

* Estimación: 3 días.
* Priorización: Debe tenerlo.
* Terminado: Cualquier palabra introducida que no se encuentre en el diccionario de la RAE no se considera correcta.

Detalles: Las palabras válidas descritas se guardan en un fichero de texto. Es importante tener en cuenta que todas las palabras con tildes o con la letra ñ fueron eliminadas de este fichero, luego si un usuario introduce una palabra con este estilo, se considerará no válida. El Equipo de Desarrollo se ha servido de la búsqueda binaria en un array (ordenado), para verificar si realmente la palabra existe o no.

1. Como usuario, quiero que las palabras introducidas por un jugador deban cortarse con una palabra ya existente en el tablero (a excepción de la primera que deberá colocarse en la posición central)

* Estimación: 5 horas.
* Priorización: Debe tenerlo.
* Terminado: No se permite introducir una palabra en una posición del tablero que no cumple los requisitos descritos.

Detalles: Con esta historia de usuario, el Equipo Scrum se asegura que la lógica del juego va incrementando su complejidad, y se va acercando más a como realmente se juega al Scrabble. La impementación se ha llevado a cabo mediante expresiones booleanas.

Con respecto al trabajo necesario para entregar este incremento, el Equipo de Desarrollo empezará a buscar por internet ficheros que contengan las palabras del diccionario español y tratarán dicho fichero para que se ajuste a la historia 1. El Equipo de Desarrollo mantendrá alguna que otra reunión breve para aclarar detalles de implementación. Además se empezará a desarrollar los modelos UML de la aplicación (Diagrama de clases y Diagrama de secuencias) a través del software *Modelio*.

* 1. **SPRINT 3**

**Historias de usuario planeadas para el Sprint actual:**

1. Como usuario, quiero que el juego pueda ser cargado desde fichero y también guardado en cualquier momento para poder reanudar una partida no finalizada.

* Estimación: 10 horas.
* Priorización: Debe tenerlo.
* Terminado: La partidas se cargan desde un fichero correctamente y se guardan en otro fichero con el formato establecido.

Detalles: Para el comienzo de una partida, para la carga de una existente o para guardar la actual, se sigue el formato JSON. Además, se emplearán factorías para crear los objetos en la aplicación y dos clases específicas encargadas de cargar y guardar partidas. En cualquier momento del juego se pueden usar comandos para guardar o cargar una partida.

Para la implementación de esta historia de usuario, el Equipo de Desarrollo se apoyará en proyectos anteriores para manejar dicho formato. Será necesaria la creación de un fichero con formato JSON para la carga de nuevas partidas.

* 1. **SPRINT 4**

**Historias de usuario planeadas para el Sprint actual:**

1. Como usuario, quiero que el juego proporcione la opción de introducir jugadores automáticos con 3 niveles distintos de dificultad.

* Estimación:.15 días.
* Priorización: Debe tenerlo.
* Terminado: Los jugadores automáticos sean capaces de completar la partida.

Detalles: Se introducirán jugadores automáticos con tres niveles de dificultad diferentes: EASY, MEDIUM y HARD. Los jugadores automáticos podrán jugara partidas entre ellos o con usuarios humanos.

Para la historia de usuario elegida para este sprint, el Equipo de Desarrollo deberá llevar a cabo ciertas modificaciones del código para adaptarlo a distintos niveles de dificultad. Además, se deben crear tres niveles distintos de dificultad para los jugadores automáticos, y con ello, las distintas estrategias automáticas para estos jugadores.

1. **PRODUCT BACKLOG**

AÑADIR POR CADA HISTORIA DE USUARIO, CUANDO FUE IMPLEMENTADA CUANDO FUE MODIFICADO LO IMPLEMENTADO EN ESA HISTORIA ETC. MODIFICACIONES DE CADA UNA Y PORQUÉ CUANDO ENTRARON Y SALIERON (CUANDO SE CREARON Y SE PUSIERON EN TO DO, CUANDO PASARON A IN PROGRESS Y CUANDO CLOSE)

* 1. **INICIALIZAR A LOS JUGADORES**

Como usuario, quiero que se implemente una versión simplificada del Scrabble que permita inicializar los jugadores, para reducir la dificultad de juego y diseño.

* Estimación: 4 días.
* Priorización: El proyecto debe incluir esta funcionalidad.
* Terminado: Versión primitiva del juego que inicializa a los jugadores.
  1. **MULTIJUGADOR**

Como usuario, quiero que el juego implementado sea multijugador, para permitir partidas entre 2 y 4 jugadores*.*

* Estimación: 1 hora.
* Priorización: Deber tenerlo.
* Terminado: Cualquier partida permite un flujo de juego entre 2 y 4 jugadores.
  1. **PALABRAS VÁLIDAS**

Como usuario, quiero una versión simplificada del Scrabble en la que se comprueben que las palabras introducidas son correctas (excluyendo tildes y palabras con ñ), para añadir un pequeño nivel de dificultad al juego.

* Estimación: 3 días
* Priorización: Debe tenerlo.
* Terminado: Cualquier palabra introducida que no se encuentre en el diccionario de la RAE no se considera correcta.
  1. **CARGAR/GUARDAR**

Como usuario, quiero que el juego pueda ser cargado desde fichero, y también guardado en cualquier momento, para poder reanudar una partida no finalizada.

* Estimación: 8 horas.
* Priorización: Podría tenerlo.
* Terminado: Las partidas se cargan desde un fichero correctamente y se guardan en otro fichero con el formato establecido.

Durante el sprint 3 el nombre de esta historia de usuario fue modificada. Pasó de llamarse *Entrada/Salida* a llamarse *Cargar/Guardar*.

* 1. **JUGADORES AUTOMÁTICOS CON AL MENOS 3 NIVELES DE DIFICULTAD**

Como usuario, quiero que el juego proporcione la opción de introducir jugadores automáticos con 3 niveles distintos de dificultad

* Estimación: 15 días.
* Priorización: Debe tenerlo.
* Terminado: Los jugadores automáticos sean capaces de completar la partida.
  1. **INTERFAZ GRÁFICA**

Como usuario, quiero que el juego proporcione un entorno visual sencillo para que los usuarios puedan interactuar con facilidad y el juego fluya de forma dinámica.

* Estimación: 8 días.
* Priorización: Debe tenerlo.
* Terminado: La interfaz gráfica tenga todo lo necesario para poder cargar, guardar y jugar partidas de forma sencilla.
  1. **JUEGO EN RED**

Como usuario, quiero que se pueda jugar al juego en línea con jugadores desde distintos terminales.

* Estimación: 10 días.
* Priorización: Debe tenerlo.
* Terminado: Se pueda completar una partida en la que participen jugadores desde diferentes terminales.

1. **SPRINT BACKLOG**