PLANIFICACIÓN.

JÁBEGA TEAM

ÍNDICE

Integrantes

Nombre: Iván Díaz García Correo: ivandiazuni@gmail.com
Nombre: Ángel Campos Salido Correo: angelcamsal06@uma.es
Nombre: Adrian Fernandez Vera Correo: adrianfeve@gmail.com
Nombre: Alberto Sánchez Aparicio Correo: albertoalual03@gmail.com
Nombre: Youcef Abi Ruiz Correo: youcefatbi@uma.es
Nombre: Manuel Ruiz Campos Correo: manuelrc@uma.es
Nombre:Antonio Jesús Díaz Plaza Correo:antoniojesusdiazplaza@gmail.com
Nombre: Ángel Tobaruela Baños Correo: angeltoba@uma.es

Repositorio GitHub

https://github.com/AngelUma/Jabega-Team

Espacio de trabajo Trello

https://trello.com/invite/b/HS2u7KpC/ATTI4d4f0dadecc8ae5348a78d7e65f8703f924FDF3A/ingenieria-del-software-jabega-team

INTRODUCCIÓN

Este proyecto el cual será desarrollado por Jábega Team, se enfrenta al desarrollo de un videojuego local de un solo jugador, que consiste en carreras de barcas.

El videojuego titulado como "Dragon Boat Racing", consistirá en varias carreras con fase eliminatoria en la que 4 barcas correrán por obtener la medalla de oro, superando así distintos obstáculos y utilizando mejoras a lo largo del recorrido para llegar a la meta.

A través de la información proporcionada por nuestro cliente, buscamos con nuestro software desarrollar este juego en 2d, con múltiples posibilidades en cada partida, creando así una experiencia fresca y divertida para los usuarios que adquieran este producto.

ROLES

Coordinadores: Ángel Campos Salido e Iván Díaz García.
Programadores: Adrian Fernandez Vera, Alberto Sánchez Aparicio, Ángel Campos Salido y Antonio Jesús Díaz Plaza.
Diseñadores: Manuel Ruiz Campos, Alberto Sánchez Aparicio y Adrian Fernandez Vera.
Analistas: Manuel Ruiz Campos, Iván Díaz García y Antonio Jesús Díaz Plaza.
Testers: Youcef Abi Ruiz, Manuel Ruiz Campos y Ángel Tobaruela Baños.
Consultores: Ángel Tobaruela Baños y Youcef Abi Ruiz.

GESTIÓN DE RIESGOS

Con la intención de conseguir un proyecto sólido y la tranquilidad del cliente en Jábega Team hemos realizado un estudio de todas las posibles circunstancias adversas que puedan golpear tanto al desarrollo del producto como a nuestra organización interna para así minimizar sus efectos sobre el mismo.

Estos riesgos abarcan todos los ámbitos que conciernen al proyecto como pueden ser los relacionados con el personal o lo tecnológico, teniendo en cuenta las entrevistas con el cliente y posibles cambios en los requisitos.

Cabe destacar que todos estos riesgos están en constante supervisión durante la duración de todo el proyecto.

IDENTIFICACIÓN DE RIESGOS

- **Cambio de requisitos por el cliente**: Este riesgo de requisitos consta en que el cliente quiera modificar o añadir requisitos al proyecto, este riesgo es bastante probable. Este causaría un riesgo insignificante si periódicamente se concretan entrevistas con el cliente para ver que el proyecto se está desarrollando correctamente.

- **Baja de algún integrante del proyecto**: Riesgo personal por abandono del proyecto u/o enfermedad, la probabilidad de que ocurra es baja, pero nos crearía un problema serio, A más personas abandonen el grupo, más catastrófico sería el riesgo. ya que perderíamos eficiencia en el grupo y para mitigarlo la única posibilidad es adaptarse a los horarios de entregas y trabajar con menos integrantes del grupo y cubrir sus puestos repartiendo su trabajo entre los integrantes restantes.

- **Dificultades técnicas a la hora de desarrollar el proyecto:** Puede sucederse debido a nuestra poca experiencia de programación a alto nivel, la probabilidad de que pase es alta, y puede generar efectos serios y negativos a la hora de la finalización del proyecto, aunque siempre se puede mitigar investigando en foros, consultas y aprendiendo tecnicas de programacion de alto nivel que nos faciliten la realización del código.

- **Falta de disponibilidad de los integrantes (exámenes u otros eventos importantes):** A diferencia de la baja de algún integrante, este riesgo será

puntual y momentáneo, lo cual lo hace tolerable y se podrá reajustar fácilmente si el grupo entero se pone de acuerdo en cubrir su puesto además de repartir el trabajo del integrante no disponible en ese momento.

- **Indisponibilidad del cliente en entrevistas/revisiones:** Riesgo el cual trata de la poca disponibilidad horaria y compatible del cliente con el grupo, para resolver nuestras dudas o realizar entrevistas. Es un riesgo serio, ya que concretar citas previamente antes de ciertas entregas son claves y sin su disponibilidad son imposibles. Como solución podemos avisarle con la suficiente antelación y dar varias opciones de nuestro horario para poder dejar una fecha escogida que nos sea conveniente para ambos.

- **Pérdida de la información del proyecto:** Riesgo tecnológico muy poco probable, que crearía un efecto serio (pérdida completa de código), es decir, vuelta a empezar desde 0. La forma más fácil de evitarlo es mediante el uso de plataformas como pueden ser GitHub o Drive para subir nuestros archivos día a día y así tener una copia de nuestro avance.

- **Daño en el equipo utilizado para la realización del proyecto:** Este riesgo puede afectar a la eficacia en equipo ya que si X integrantes del equipo se quedan sin su dispositivo, no podrían trabajar e incluso deberian otras personas del grupo cubrir sus puestos mientras intentan conseguir/arreglar su dispositivo. Aunque es poco probable, siempre se calificaría como un riesgo tolerable, porque se puede arreglar fácilmente consiguiendo otro dispositivo dado a que la mayoría contamos con más de un dispositivo a nuestro alcance.

- **Cambio en la fecha de los entregables:** Riesgo de requisitos bastante probable, que afectaría de distintas maneras dependiendo de si dicha fecha es adelantada o atrasada. Si la fecha de entrega es adelantada a una anterior a la prevista, esto causaría una alteración en nuestra distribución de tiempo y trabajo, obligándonos a reestructurarnos para así llegar a tiempo a la nueva fecha, siendo un problema bastante serio. En cambio, si la fecha es atrasada, la inconveniencia será bastante menor, aunque eso también nos haría alterar ligeramente nuestra estructura de trabajo. Para solucionar estos riesgos, podríamos tener como obligación el terminar una tarea varios días antes de su fecha de entrega, así no nos afectaría sin importar si es atrasada o adelantada.

- **Falta de experiencia del personal del proyecto:** Este riesgo de personal puede hacer que el personal tarde bastante más de lo estimado en hacer las tareas que le corresponde debido a la inexperiencia. Es de probabilidad alta, ya que aún estamos aprendiendo y no llevamos el suficiente tiempo como para manejarnos con la soltura necesaria. Esto sería un problema bastante tolerable, ya que su solución sería apoyarse tanto en tus compañeros de trabajo como en guías de internet para resolver las dudas que haya.

- **Estimación inadecuada de los plazos:** Este riesgo de estimación puede hacer que las diferentes tareas del proyecto se vayan acumulando, traduciéndose en un retraso de la entrega final del proyecto al cliente, lo cual sería catastrófico. Es de probabilidad moderada. Para solucionarlo, al igual que cuando se modifican las fechas de entrega, podemos gestionarnos para tener las tareas preparadas antes de su fecha límite, así asegurándonos llegar a tiempo a los plazos y tener cierto margen en caso de que haya que modificar algo.

PLANIFICACIÓN

- ESTRATEGIA APLICADA:

- Usar una estrategia **scrum** es una buena idea para un proyecto de desarrollo de un videojuego indie por varias razones. Una estrategia scrum es una metodología ágil que se basa en la colaboración, la adaptación y la entrega rápida de valor.

- Algunos de los beneficios de usar una estrategia scrum para desarrollar un videojuego de las características solicitadas son:

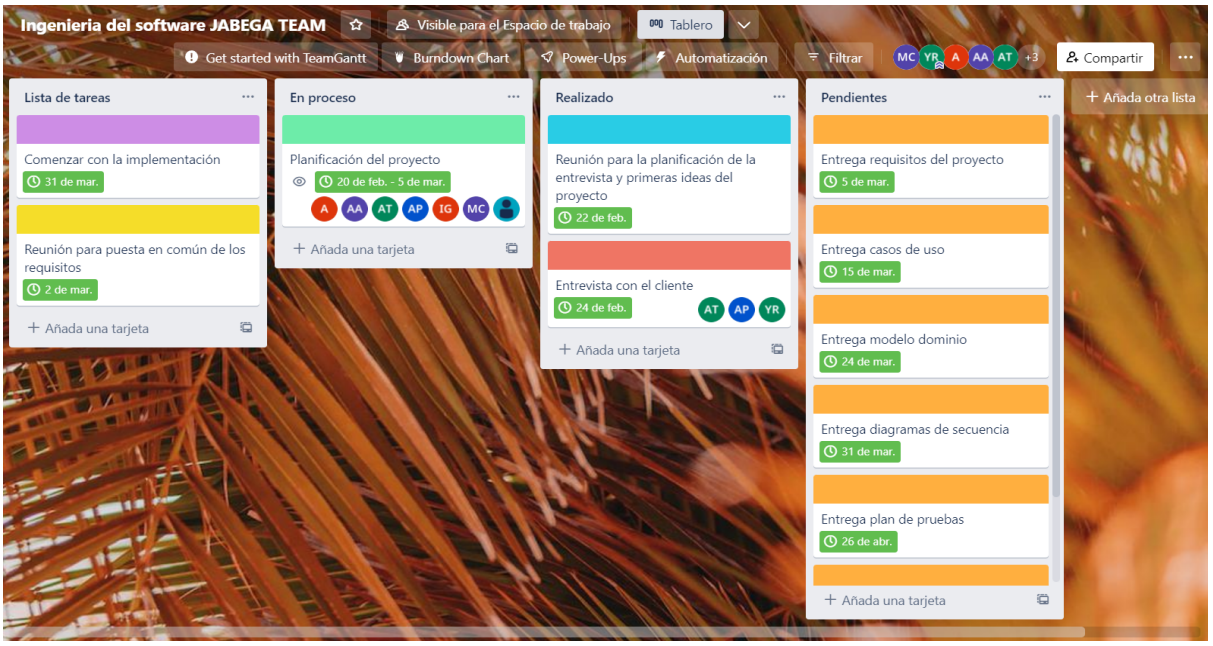
    - Permite definir el alcance y los objetivos del proyecto de forma clara y flexible, adaptándose a las necesidades y preferencias del cliente y a las futuras posibles modificaciones del mismo.
    - Facilita la gestión del equipo de trabajo, asignando roles y responsabilidades claras, fomentando la comunicación y el feedback continuo, y motivando a los miembros a dar lo mejor de sí mismos cumpliendo una serie de entregas marcadas en la empresa.
    - Mejora la calidad del producto final, al realizar entregas parciales e iterativas, que permiten probar y validar el funcionamiento y la satisfacción del videojuego en cada etapa.
    - Reduce los riesgos y los costes del proyecto, al identificar y resolver los problemas o impedimentos que puedan surgir en cada sprint, evitando retrasos o desperdicios innecesarios.

- Por estas razones, usar una estrategia scrum es una buena idea para un proyecto de desarrollo de un videojuego de las características indicadas.

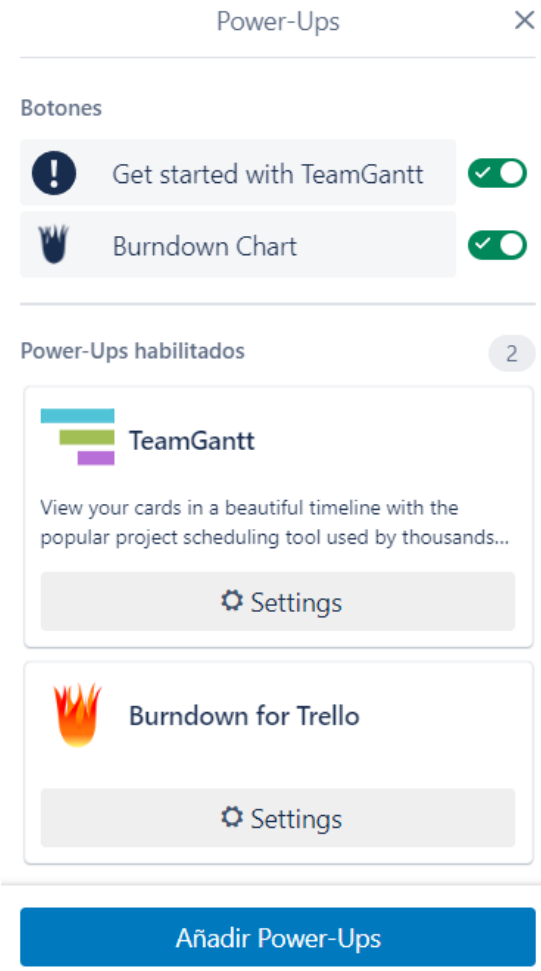
- FECHAS IMPORTANTES:

- Propuesta de proyecto del cliente (fecha límite: 10 de febrero).
- Estudio de viabilidad del proyecto.
- Selección del equipo de proyecto.
- Reunión de proyecto con el cliente para la obtención de requisitos (fecha: 24 de febrero).
- Plan de proyecto (fecha límite: 5 de marzo) .
- Especificación de requisitos del software.
- Entrega de especificación del proyecto al cliente (fecha límite: 12 de marzo).
- Entrega de casos de uso (fecha aproximada: 15 de marzo) .
- Modelo de Dominio (fecha aproximada: 24 de marzo) .
- Entrega de diagramas de secuencia (fecha aproximada: 31 de marzo).
- Plan de Pruebas (fecha aproximada: 26 de abril).
- Entrega de proyecto completo al cliente (fecha : 12 de mayo).
- Presentaciones (fecha aproximada: 15 y 16 de mayo).

Tablero de Trello:



Power-ups usados:
TeamGantt y Burndown for Trello



HERRAMIENTAS SOFTWARE

- GIT: Repositorio para almacenar todos los recursos del proyecto.
- Visual Paradigm: Programa con herramientas CASE para la realización de los requisitos.
- Mockito: Framework de prueba del código abierto para Java.
- Trello: Programa para realizar la planificación del proyecto.
- Discord: Red social para la comunicación del grupo en las reuniones online/semipresenciales por vía audio.
- WhatsApp: Red social que actúa como medio de comunicación del grupo.
- Eclipse: Entorno de desarrollo usado para el desarrollo del código en Java.
- Java Swing: Herramienta de interfaz gráfica de usuario (GUI).
- Visual Studio Code: Entorno de desarrollo de código Java.
- JUnit: Entorno de desarrollo para el código Java.
- LibGDX: Framework de desarrollo.
- Google Drive: Entorno de guardado de datos, en los que podemos utilizar para guardar documentación.