COVER

Table of Contents

1. Introduction

This section serves as a small introduction to our project, which is video-game called "Dragon Boat Racing" as specified by our dear client Mr. Javier Cámara. Dragon Boat Racing is a 2D, single-player game that will be developed for the PC platform using the Java programming language.

For the gameplay aspects, Dragon Boat Racing is an adventure game where the player will experience the rush of competition, the excitement of power-ups, the satisfaction of overcoming obstacles and most importantly glory after winning a race. It will consist of different levels, each one with increased difficulty and having a unique scenery. Players will have the ability to customize their boat's appearance and choose from boats with different stats to match their play-style.

Every race is an adrenaline-fueled test of skill and strategy as it takes great skill for a player to control their boat in the lane and defeat their opponents. The game's mechanics are basic yet very exciting. Leaving the lane penalizes you making it harder for you to win the race and colliding with obstacles eventually results in the sinking of your boat. It is here that the game will give less skilled players a chance to play a simple "Simon-says" mini-game to repair and revive their boat. The different power-ups make the experience more exciting, enjoyable, and dynamic.

1.2 Software Tools

Every project and team needs different tools. Some tools are more suitable for the job than others. Here we will list the tools that we have chosen for our Dragon Boat Racing project after carefully looking at all the options, consulting with different team members and of course making sure that they are the best tools for the job.

1.2.1 Communication software

This section outlines the tools we utilize for seamless communication within the team. Effective communication software is crucial for real-time collaboration, swift issue resolution, and maintaining a cohesive team environment. These tools facilitate instant messaging, and videoconferencing ensuring that team members stay connected regardless of geographical locations.

- **Discord** will serve as our main communication platform. The app is great for videoconferencing between our team members, has a modern interface and mobile applications to make sure our team is always connected.
- **Whatsapp** will serve as another communication platform mainly for its convenience, and its instant messaging capabilities as anything critical can be discussed via Whatsapp between our team members for instant response.

1.2.2 Project planning and productivity

This section enumerates the tools we employ for project planning, task management, and enhancing overall productivity. These tools are essential for organizing workflows, setting project milestones, allocating resources, tracking progress, and managing timelines efficiently. By utilizing such tools, we ensure transparency, accountability, and alignment with project objectives, ultimately optimizing team performance and project outcomes.

- **Trello** will be used exclusively as it is very well known in the industry and of course preferred by our project managers. It will make it easy for us to assign tasks and distribute work between our members thanks to its excellent, simple yet powerful web interface.

1.2.3 Development related

This section delineates the software tools integral to the development process, including integrated development environments (IDEs), and version control systems. These tools facilitate code creation, debugging, version tracking, and testing. By leveraging these tools, we enhance code quality, accelerate development cycles, and foster collaboration among our team's programmers.

- **IntelliJ IDEA** is an IDE very well known in the java development space. It is modern, simple, powerful and above all our team of developers is very comfortable using it. It has support for different build tools like ***maven*** which we will use for our Dragon Boat Racing game.
- **Git** is an excellent version control system and is an industry leader. It is fully free and open source, and will give us access to powerful version controlling. As for a git server, we will use ***Github*** simply because it doesn't require us self-hosting our git server, is easy to use and also provides an excellent desktop client for us to use.

1.2.4 Design

This section highlights the tools utilized for graphical design, prototyping, and user interface (UI) development. These tools enable our graphic designers to conceptualize, iterate, and refine design elements, ensuring intuitive user experiences and visually appealing interfaces. From wireframing and mockup creation to asset management and prototyping, these tools play a pivotal role in translating design concepts into tangible product features, aligning with our client Mr Cámara's expectations.

- **Visual Paradigm** is versatile tool for UML diagramming and requirements definition. It offers a user-friendly interface and a wide range of features for creating diagrams like use case, class, and sequence diagrams. Facilitating collaborative design and analysis, it ensures efficient communication and alignment of design concepts between us and our client, Mr Cámara. It is simply put ideal in our case of agile development.
- **Canva** is an excellent modern graphic design tool which serves for creating simple mock-ups as well as different marketing and mock-up material when working in our team.

- **Adobe Creative Cloud** is a comprehensive suite of creative tools for graphic design, photo editing, video production, and web development. Adobe Creative Cloud includes industry-standard software such as **Photoshop** and **Illustrator** which will be used by our professional graphic designers to design different graphical elements of the game be it the simple game menu or complex boat designs.

1.2.5 Documentation purposes

This section identifies the tools employed for documenting project requirements, specifications, codebase, and user documentation. Effective documentation tools facilitate knowledge sharing, maintain project transparency, and serve as invaluable references for our team and are important for our client Mr. Cámara.

- **Google Docs** is a a cloud-based document collaboration platform offering real-time editing and sharing capabilities. Google Docs enables our team members to work concurrently on documents, facilitating seamless collaboration regardless of geographical locations.
- **Microsoft Office** is a suite of productivity tools including Word, Excel, and PowerPoint, providing robust capabilities for document creation, data analysis, and presentation. Microsoft Office offers a familiar interface to our team members and is an industry standard.

2. Planning

In this section, we adopt the SCRUM methodology and agile practices to foster iterative development, adaptability, and continuous improvement throughout the project lifecycle. SCRUM emphasizes teamwork, collaboration, and delivering incremental value to stakeholders through short, time-boxed iterations called sprints. Our approach to project planning involves defining and prioritizing tasks, setting achievable goals, and allocating resources effectively to maximize productivity and project success. This robust planning will help us achieve our goals and provide make sure that we satisfy the needs of our client, Mr Cámara.

• ***(March 8 - March 13)***: Dedicated to discussions regarding the video-game implementation, creating a list of questions to present to the client on March 13. These questions cover aspects such as boat movements, obstacle generation, boat skins, and levels. Additionally, communication channels through Discord were established, along with task management using Trello and GitHub integration.

• ***(March 13 - March 22):*** With a more refined concept, we implemented the requirement diagram, risk management, role assignment, and future planning. Several meetings were conducted via Discord to oversee the progress of these tasks and their respective reviews.

• ***(March 23 - April 1):*** Break for Holy week.

• ***(April 2 - April 7):*** Devoted to meetings between programmers and testers, consulting the proposed Java library and necessary documentation, as well as defining and developing initial programs and functions. Graphic designers convened to sketch and specify the number of boat, obstacle, and level skins. A general meeting was also held to consolidate ideas and review progress across the different sectors.

• ***(April 7 - April 21):*** Beta versions of the game were archived, featuring boat movement, obstacles, and basic skins for boats, obstacles, and levels

• ***(April 22 - May 6):*** Improved versions of the game incorporating physics collisions with obstacles and boats, diverse attributes for boats such as handling or speed, even the increase of difficulty across levels, random object appearances, and mini-games. Graphic developers expanded the repertoire of boat and level skins.

• ***(May 6 - May 13):*** Final versions from developers with the latest implementations and improvements, potentially including the integration of power-ups during gameplay or enhanced developer-friendly controls. Graphic designers introduced a degraded version of boats to depict damage accumulation, upgrading previous iterations as necessary.

• ***(May 13 - May 31):*** Dedicated to testing the final game, with testers assuming responsibility for this phase. Programmers focused on bug fixes in the code, without introducing additional mechanics. Delivering the project to the client and ask for their feedback.

Section 2: Roles.

Indicate the roles each team member has. Each one must have two roles, and each role must be covered by at least two colleagues. [Remember that having a role means being responsible for that task, but all team members must be knowledgeable about all tasks.]

Proposed roles:

PROGRAMMER

-Angel Escaño

-Pablo Hormigo

-Sultan

-Fran

TESTER

-Ricardo

-Diego

-Juan Torres

-Angel Bayon

PROJECT MANAGER

-Manu

-Diego

-Pablo Hormigo

SPOKESMAN

-Angel

-Fran

-Ricardo

GRAPHIC DESIGNER

-Angel Bayon

-Sultan

-Manu

-Juan Torres

Section 3: Risk Management.

It is necessary to identify the project's risks and, for each one, indicate: type of risk, description of the risk, probability (very low, low, moderate, high, or very high), effects of the risk (catastrophic, serious, tolerable, or insignificant) and strategy to mitigate it.

Section 4: Planning.

The chosen software process model must be indicated (and why it was chosen) and paste screenshots of the Trello boards with the identified tasks and their organization. If Power-Ups have been used, also paste screenshots of what they show (Gantt Diagrams will be positively valued).

Section 5: Software tools

Tools used during the project's realization. Indicate the software tools that have been used to develop the project to date. Any type of tool: communication, collaborative work, document elaboration, etc.

Section 6: Requirements

Add the requirements as explained at the end of the slides in the practical requirements. That is, identify with colored cards the FRs and NFRs (both mandatory and optional) in the form of user stories, and it will be positively valued if the requirements have associated acceptance tests. Each user story must have a unique identifier (for example, RF1) and a title.

FR1: INITIALIZE THE GAME

As a player I want the game to start whenever I want to play it

FR2: BOAT CONTROL

As a player I want to be able to configure my own keybinds and I want the controls to be responsive with low latency, in order to have a more personalized and pleasant experience.

FR3: POWER UPS

As a player I want to have powerups that modify the way the game is played, in order to have as different an experience as possible in each run.

FR4: LEVELS

As a player I want to have different levels in order to get a sense of progress while I´m playing and also to raise the stakes whilst I advance in the game.

FR5: BOAT SPRITE

As a player I want to be able to modify the way my boat looks to avoid a repetitive and dull gameplay.

FR6: DIFFICULTY

As a player not having an incrementally more difficult gameplay would make the gaming experience bland and boring. This goes together with the levels functional requirement

FR7: OBSTACLES

As a player I want to have different obstacles positioned in different places in order for the game to be challenging and engaging.

FR8: SCENERY

As a player I want to experience a visual change once I advance through the levels, to have a notion of progress and risk.

FR9: AFTERLIFE

As a player I would like to have an opportunity to come back if I die as it would make it more engaging and fun to play.

FR10: BOAT LANES

As a player I want to have a boat lane in which I must stay during the race in order to reduce chaos if there are multiple boats on the screen.

FR11: RIVALS

As a player I want to be competing against other alleged players, in order for the game to have a little of competition.

FR12: PENALTY

As a player I want to be punished and rewarded accordingly, for the game to have a bit of edge.

FR13: EXITTING THE GAME

As a player I want to be able to move fast between screens and to exit the game seamlessly in order to avoid annoyance.

FR14: BOAT STATS

As a player I want to be able to choose boats with different characteristics that add depth to the way the game is played.

FR15: SIMON SAYS MINIGAME

As a player I want to have a minigame inside the main game in order to have a more complete experience.

FR16: LOADING SCREENS

As a player I want to have an ejoyable experience while I wait for the game to load.

VOI

As a player I would like to have a fair crash system that stuns me the right amount of time in order to have a challenging but not impossible difficulty.

NFR1: IMAGE REFRESH RATE

As a game developer I want the game to execute at a flawless 30 fps in order for the experience to be more pleasant.

NFR2: LOW LATENCY AND KEYBINDS

As a game developer I want the controls to be responsive and accurate for the player to have a smooth experience.

NFR3: AVOIDING SYSTEM ERRORS

As a developer I want the code to be robust and free of errors in order to provide the player a satisfactory experience.

NFR4: DIFFICULTY AND RANDOMNESS

As a developer I want the experience to be easy and simple at first but harder as the player advances through the levels in order to have an entertaining game for all publics but a challenging one for the adventurous.

NFR5: JAVA LANGUAGE

As a developer I want to use the language in which all of my team is experienced at.

NFR6: LIBRARIES

As a developer I want to use the same libraries that have been used for similar projects that have been successful

NFR7: COLLISIONS

As a developer I want the game to have fair hit boxes in order for the game not be frustrating.

NFR8: SYSTEM FAILURES
As a developer, I don’t want to deliver a final version of the project that could contain fatal errors making the game not playable.

NFR9: AI
As a developer I want to implement an ai in order to make competition between different players and make the game more interesting.

Section 7: Requirements diagram

An image of the requirements diagram made in Visual Paradigm must be pasted (this diagram does not indicate the text of the user stories, only the identifier and title). The inclusion of relationships between requirements will be valued, as well as a brief explanatory text indicating why those relationships have been included. If the diagram is too large to be legible in the document, it is recommended to include an overview of it, and then the inclusion of several "detail" images with different parts of the diagram that are legible, always indicating which part of the diagram they correspond to.