Epoka University

Faculty of Engineering and Architecture

Department of Computer Engineering

CEN 302 – Software Engineering

Worlds Around™
Requirements Specification

April 16, 2023

Worlds Around™

Requirements Specification

First Draft

April 13, 2023

Prepared by:

Arbin Bici

Elvjo Fejzo

Emis Reka

Enri Shtjefni

Eralba Korbi

Eva Veli

Marvin Hoxha

Terens Tare

Received by: M.Sc. Ari Gjerazi

Table of Contents

1. Executive Summary

1.1 Project Overview

In our ever-evolving world, computer games serve as the main form of entertainment available. Their popularity has increased ever since the release of the original Pong game. This increase in popularity ignited a rivalry in the game development industry, which forced developers to improve their games on every possible aspect to increase the sales rate. Hardware limits were often passed as if they didn't exist in the first place, with the focus of these optimizations being the visuals offered and making the game playable on limited (usually last-generation or older) hardware.

Nowadays, the hardware available can handle a wide range of games, even on low-end devices, thus easing the process of developing games that do not contain a huge number of objects in them. AI and physics-related code are an exception to this statement, where research is still undergoing on implementing more efficient algorithms or improving existing ones. There are still other things that need to be taken into consideration when developing a game. On this document, we present the process of developing a game, describing the requirements of said game, the problems and shortcomings encountered, as well as the solutions proposed to address them.

The game that will be described is a platformer-style game, a choice made based on two main factors: first, the genre is popular on both AAA studios and indie developers, thus allowing for both points of inspiration for game mechanics and UI design, as well as for comparison of these aspects as well, using existing games as a "standard". Second, the game incorporates many aspects of game development, such as UI design (different in-game menus), animation (player, objects in the world, etc.), physics (jumping, colliding with objects, etc.), AI (for enemies), serialization (saving/loading data), thus making it suitable to showcase different aspects of game development. We will further expand on each of these aspects, and others such as player controls and interaction with the surroundings, describing approaches we have considered, with the benefits and shortcomings of each. Along with it, we will provide sketches of the game's design on different stages of the game to make visualization easier.

The game's main objective will be to provide the player with a memorable experience. One of our goals as developers is to build the gameplay such that it allows for simple, yet configurable controls. To further help players, the game will demonstrate new mechanics and concepts to players as needed, by effectively providing built-in tutorials for such mechanics.

1.2 Purpose and Scope of this Specification

The purpose of this specification is to define the requirements for a platformer-style game that provides players with an engaging and entertaining experience. This document outlines the game's objectives, mechanics, and features, as well as the technical specifications required for its development. The specification serves as a guide for the game development team to ensure that the game meets the

desired quality standards, is delivered within the specified budget, and is completed within the agreed timeline.

The scope of this specification is to define the requirements for developing a platformer-style game that includes multiple levels, characters, enemies, and obstacles. The game is designed to provide a fun and challenging experience for players, with various game mechanics and features that keep the players engaged. This specification covers the game's objectives, mechanics, and features, including player controls, user interface, graphics, sound, music, and game progression. The specification will also define the technical specifications, such as the platforms, programming languages, software tools, and hardware requirements necessary for game development. However, this specification does not cover the marketing, distribution, or monetization of the game.

2. Product/Service Description

The platformer-style game we are developing is an action-packed, adventure game that challenges players to overcome obstacles, defeat enemies, and explore new worlds. The game is designed to provide players with an immersive experience that keeps them engaged and entertained for hours.

The game will include multiple levels, each with its own unique challenges, enemies, and obstacles. The player will control a character who has the ability to run, jump, and attack. The player will face various challenges such as jumping across platforms, dodging obstacles, and defeating enemies. The game will also include power-ups and collectibles that will help the player progress through the game.

In addition to the core gameplay, the game will feature an engaging storyline that takes the player on an adventure through various worlds. The player will encounter a variety of characters, each with their own unique story and personality. The game will feature high-quality graphics and sound effects to enhance the player's experience. The user interface will be intuitive and easy to use, allowing players to quickly navigate through the game and access different features.

The platformer-style game we are developing is a thrilling adventure that combines challenging gameplay with an engaging storyline, high-quality graphics, and an intuitive user interface. It is designed to provide players with a fun and entertaining experience that they will want to come back to again and again.

2.1 Product Context

Market and Industry Landscape: Platformer games are a popular genre, with many successful titles available on various platforms, including consoles, PCs, and mobile devices. Our game must compete with existing titles and meet or exceed their quality and entertainment value to attract players.

Technological Advancements: The gaming industry is continually advancing, and our game must leverage the latest technologies to create a visually stunning and immersive experience for players. This includes the use of game engines, graphics rendering, animation, physics simulations, and other tools to create a polished and engaging game.

Target Audience Expectations: We will conduct market research to gain insight into the preferences and expectations of platformer game players. This research will inform our decisions on game mechanics, art styles, level design, and other features to ensure our game meets the needs of our target audience.

Cross-Platform Compatibility: To reach the broadest audience possible, our game will be designed to run on multiple platforms, including PC and Mac. This will require careful consideration of the technical requirements and limitations of each platform.

Cultural and Social Context: Our game will be released in a global market, and we must consider the cultural and social context in which it will be played. This includes language localization, cultural sensitivities, and accessibility features to ensure all players can enjoy the game regardless of their background or abilities.

2.2 User Characteristics

There is only one user that will interact with the game:

Player:

- ***Can move:*** it has the ability to control the character's movement using the keyboard arrows or some other particular keys which are displayed at the menu page. They are able to move the character left and right, jump and progress through the game.
- ***Can use superpowers:*** The player can use one of the two options to give advantage in game. The first superpower makes the user run faster, while the second one gives the player the possibility to become invisible.
- ***Can customize the settings:*** The player can select one of the powers. It can also customize the background music of tha game and also the volume of the character movement sounds.
- ***Can create and delete profiles:*** The player has the possibility to create a profile in the game. It also has the option to delete a particular profile and add another new profile.
- ***Can pause the game:*** The player has the ability to pause the game at any time, providing them the control and flexibility during gameplay. This feature allows the player to take a break, and then resume the gameplay at their convenience without losing the progress.
- ***Can quit the game:*** The player basically is available to quit the game anytime providing them the option to exit the game in a controlled and safe manner. This feature enables the user to end their gameplay session in a particular level.
- ***Can restart the game:*** This option provides the option to again begin their gameplay session.This feature allows the user to reset their progress and start over, either to improve their previous performance or to experience the game again from the beginning.
- ***User can receive accomplishment:*** The ability to play the game can also allow the user to experience the achievement as they overcome obstacles and reach new milestones as they progress through

different levels. By taking reward the user can better indicate their progress and performance encouraging them to continue playing.

2.3 Assumptions

User:

- It is assumed that the game is a 2D platformer with a main character that can jump,run, interact with the environment.
- It is assumed that the game is designed for a computer platform.
- It is assumed that the user has access to a computer or device that meets the minimum system requirements for running the game.
- It is assumend that the game is free of bugs that could hinder the user's ability to play or complete the game.
- It is assumed that the game will have sound effects and background music to enhance the gaming experience.
- It is assumend that the game will be played using the keyboard inputs for controlling the main character , with arrow keys for movements, or some other characters visible at the main page and the spacebar for jumping.
- It is assumed that each level of the game becomes progressively more challenging, with a gradual increase in difficulty by adding more difficult obstacles.
- It is assumed that the game has different levels with varying degerees of difficulty.
- It is assumed that the game has a scoring system that rewards the player for completing levels and achieving specific objectives.
- It is assumed that the game includes obstacles that the main character needs to overcome to progress through the game.
- It is assummed that the game has two power-ups that can improve the main character's abilities and enhance the gaming experience.

- It is assumed that the game has a well-designed user interface, including the main page, pause and the settings.

- It is assumed that the game engine is optmized for performance to ensure smooth gameplay on the target hardware and software configuration.

- It is assumed that the game's controls and mechanisms are intuitive and easy to understand, allowing the user to quickly learn how to play the game.

2.4 Constraints

- The game should be designed to run on any set of hardware and software configurations, such the operation system, processor and graphics card.

- The game development process should be completed within a specific resource allocation, including hardware, software and human resources.

- The game development process should be completed within a specific timeframe to meet the project deadlines.

- The game should be designed to meet specific design requirements and constraints, such as the game's theme, styling, colors and other design specifications.

- The user interface need to be designed to meet specific design requirements , such as the user interface's appearance, layout and functionality.

- The sound effects should match with the movements of the main character and together with the background music should have the right quality for a better gameplay.

- The game is designed to fit within a certain screen size or aspect ratio, limiting the amount of visual information that can be presented at once.

- The game may be limited by the audio capabilities of the device, which could affect the quality and complexity of the sound effects and music.

2.5. Dependencies

- Pygame library: Pygame is a set of Python modules that enable developers to create games and multimedia applications. It provides access to a range of functionality for creating 2D games, including graphics, sound, input handling, and event handling.

- Python programming language: Python is a high-level, interpreted programming language that is widely used for developing games and other applications. It provides a simple syntax and a wide

range of libraries and tools that make it easy to build complex applications.

- Documentation: Documentation is used to generate documentation for the game platformer. It mostly includes description of different classes responsibilities, the methods and attributes it provides, making it easier for developers and users to understand the code and its functionality.

- Graphics and sound assets: The game platformer will require graphics and sound assets, such as images, animations, and sound effects, to create a visually appealing and engaging game.

- Development environment: The game platformer will require a development environment for coding and testing. The development environment chosein is Visual Studio Code.

- Version control system: A version control system is a software tool that tracks changes to code over time, allowing developers to collaborate on a project and keep track of different versions of the code. The version control system choose is Git.

3. Requirements

3.1 Functional Requirements

Req#	Requirement	Comments	Priority	Date	Reviewed/ Approved
FR_01	The game should have different views for different menus.	The view for main menu, settings and pause menuwill be different.			
FR_02	The game should be able to take user input from mouse and keyboard.	To be able to move the character the game should be able to take inputs.			
FR_03	The user should be able to explore the in-game world.	The user will be able to move around the world freely.			
FR_04	The user should be able to interact to object in the environment.	The game should have interactive objects that the player can use to solve puzzles or progress through the levels.			

FR_05	The user should be able interact with the enemies.	The game should include different types of enemies that the player must avoid or defeat.			
FR_06	The game should have appropriate sound effects and background music that enhance the player's experience.	The user should be able to control the music volume.			
FR_07	The user should be able to pause the game and restart it.	The user will be able to pause the game or restart it in every moment.			
FR_08	The game should be able to save the player's progress at specific points in the game.	The will be able to save the game in any point and the game will autosave when level is finished.			
FR_09	The user should be able to exit the game in every moment.	The game can be exited without loosing the progress of the player.			
FR_10	The game should include multiple levels with increasing difficulty.	Each level should be designed to challenge the player's skills and offer a unique experience.			
FR_11	The game should have various power-ups that can be collected by the player.	These power-ups should provide the player with temporary abilities such as invincibility, speed boost, or extra lives.			
FR_12	The game should have a scoring system that tracks the player's progress and performance.	The score should be based on the number of enemies defeated, and time taken to complete each level.			

FR_13	The game should have different difficulty levels.	The player can choose from various difficulty levels such as easy, medium, and hard.			
FR_14	The game should have collectibles items through the levels	The player can find and collect items throughout the levels, such as power-ups and points.			
FR_15	The game should have a time limit for each level.	The time limit will be to add an element of urgency and challenge to the gameplay.			
FR_16	The game should include accessibility options for players with disabilities.	Every user can change things to be able to play the game, such as adjustable difficulty or alternative control schemes.			
FR_17	The game should include a tutorial mode.	The tutorial mode should be able to teach the player how to play the game and use its various mechanics.			
FR_18	The game should allow the player to control the camera.	The user can change the camera to get a better view of the level or environment.			
FR_19	The game should have achievements or trophies.	The player can get achievements that he can earn by completing specific challenges or objectives.			

FR_20	The game will have a lives system.	This will allow the player to earn extra lives by collecting coins or defeating enemies.			
FR_21	The game should have platforming challenges.	This will require the player to use their jumping and timing skills to navigate obstacles.			

3.2 Non-Functional Requirements

General Non-Functional Requirements :

Req#	Requirement	Comments	Priority	Date	Reviewed/ Approved
NFR_01	Performance.	The game should be responsive and fast, with no noticeable lag or delay in gameplay.			
NFR_02	Scalability.	The game should be scalable to different hardware platforms, screen sizes, and resolutions.			
NFR_03	Compatibility.	The game should be compatible with different operating systems, browsers, and devices.			
NFR_04	Reliability.	The game should be reliable and stable, with minimal crashes or bugs.			
NFR_05	Usability.	The game should be easy to use and navigate, with clear and concise instructions.			
NFR_06	Maintainability.	The game should be easy to maintain and update, with clear and organized code structure and documentation.			
NFR_07	Compliance.	The game should comply with relevant laws, regulations, and industry standards, such as copyright or data protection laws.			

3.2.1 Product Requirements

3.2.1.1 User Interface Requirements

The user interface for the game is a critical aspect that can make or break the player's experience. To ensure the best possible experience, the interface should be easy to use and navigate, providing players with an intuitive way to interact with the game. Furthermore, it should be compatible with different devices and operating systems, allowing players to enjoy the game regardless of the platform they are using. These factors can greatly enhance the overall enjoyment of the game, creating a positive user experience that keeps players engaged and coming back for more.

In addition to outlining the necessary functions, it is important to describe the characteristics of each interface supported by the game's sketches. This provides valuable insight into how the user interacts with the game and how the game responds to user input. By analyzing these characteristics, we can identify areas for improvement and make adjustments to create a more seamless and enjoyable user experience. The main interfaces are:

Main Menu

The Main Menu interface of World Around will be the first screen that the player sees upon launching the game. The screen will have a dark-colored background image that evokes the adventurous theme of the game. At the top of the screen, the game name, "Worlds Around," will be displayed in large and bold white letters using a clear and easily readable font, such as Arcade Classic. The game logo, which may feature elements such as the world map or an adventure-themed icon, will be placed next to the game name on the left-hand side of the screen.

Below the game name and logo, three buttons will be displayed in a row - Play, Settings, and Quit. These buttons will be rectangular in shape with rounded corners and have a slightly raised appearance. The text on each button will be written in a clear and legible font, such as Arcade Classic, and will be presented in a large size to make it easy for players to read. The Play button will have a green color scheme with white text, the Settings button will have a gray color scheme with white text, and the Quit button will have a red color scheme with white text.

Finally, on the far right of the screen, a small square button with an info logo will be displayed, which players can click to access information about the game, such as the developer, game version and the game

about. This button will have a light blue color scheme with white text and a transparent background.

After clicking the Play button on the Main Menu interface, the player will be taken to a new screen that displays the profiles available in the game. This screen will feature a white background with a simple and minimalist design that emphasizes the profiles.

The profiles screen will have three rectangular boxes aligned horizontally, each representing a different profile. The profile number (either one, two, or three) will be displayed prominently on the top of each box using a clear and legible font, such as Arcade Classic.

On the bottom of each profile box, there will be a Delete button with a red background, which players can click to delete the corresponding profile. When the Delete button is clicked, a warning message will appear, asking the player to confirm if they really want to delete the profile. The warning message will be displayed in a prominent red color to draw the player's attention and ensure that they do not accidentally delete the profile. There will be two buttons here, yes or no, to confirm it or not.

When a player selects a profile, they will be taken to the last point where they left the game previously or they will continue the game from the beginning if this profile has never been used.

Pause

The pause tab is essential that it is visually appealing, intuitive, and consistent with the main menu and should have a similar background image as the main menu and buttons with similar characteristics. The buttons should be clearly labeled with text and have a border to distinguish them from the background. The font style should be chosen to complement the overall design, and it should be easy to read. A sans-serif font like Arcade Classic is a good choice, as it is simple and legible. The font size should be large enough to be easily readable.

The resume, restart, settings, and quit buttons should be aligned in a logical order and be visually distinguishable. When the player clicks the resume button, the game should immediately resume from where it was paused. Clicking the restart button should reset the game to where he started. The settings button should take the player to the settings tab, and clicking the quit button should bring the player back to the main menu, and the game should be saved where it was left in the corresponding profile.

The color scheme for the pause menu should be consistent with the game's overall aesthetic. The buttons should be a different color from the background to make them stand out, but the contrast shouldn't be too strong to avoid eye strain. A light-colored button with a dark-colored text can work well. The text on the buttons should be descriptive and easy to understand, and the buttons should be large enough to be easy

to click.

Finally, when the player clicks on a button, there should be a clear visual indication that the action has been registered. This could be a brief animation, a sound effect, or a color change of the button.

Settings

The Settings tab will have a similar background image to the Main Menu and Pause tabs, providing a consistent visual theme throughout the game. In the middle of the tab, there will be a large text header that reads "Settings" in a clear, easy-to-read font, with a font size that is large enough to be easily visible.

At the top left corner, there will be a back button with an arrow pointing to the left and the text "Back" below it. The button should have a different color from the other buttons on the tab to make it stand out and should be easy to click. When you click it, it will take you to the main menu if you clicked the settings button from the main menu tab or the pause tab if you clicked the settings button from the pause tab.

Below the Settings header, there will be the control buttons for the game. There will be four texts on the left side, one below the other, which read "Enter Door", "Left", "Down", and "Right". On the right side of each of these four texts, there will be four buttons that are "W", "A", "S" and "D" respectively, corresponding to the four texts on their left.

At the right side of these four buttons, there will be three texts one below the other, that are "Jump", "Power1", and "Power2". On the right of each of these three texts, there will be three buttons that are "Space", "1", and "2" respectively, corresponding to the three texts on their left. The buttons should have a clear, easy-to-read font and a font size that is large enough to be easily visible.

When the user clicks on these buttons, they can change the controls to their preferences and the new control scheme will be saved. Below these texts and buttons in the middle, there will be a reset button that the user can click if they want to reset the controls to the default.

At the end of the tab, there will be two small sound images, one for the background music and the other for the game sound. These images will be one below each other and will have two straight lines and a point for each line that will be on these lines. This will show how much the volume is for the background music and the game sound. If it is at the beginning of the line, it tells that there will be no sound or music for the game, and there will be a red cross on the image to show that this sound is muted.

The colors of the buttons and text should be chosen to provide good contrast against the background image and should be visually appealing. The buttons should be large enough to be easily clicked and the text should be easy to read.

3.2.1.2 Learnability

If you are working on a platformer game, here are some learnability requirements to consider:

- **User Interface Design:** The user interface of the game should be designed to be intuitive and easy to navigate. All the game mechanics, menus, and settings should be accessible through the interface.
- **Consistency:** Consistency is key when it comes to learnability. The game mechanics, controls, and visual feedback should be consistent throughout the game. The player should be able to predict the outcome of their actions based on their experience from previous levels.
- **Onboarding:** The game should have a clear and concise onboarding process that introduces the player to the game mechanics and controls. It is best to introduce new mechanics one at a time, rather than overwhelming the player with too much information at once.
- **Feedback:** The game should provide clear and immediate feedback to the player. This can be done through visual or audio cues. The player should know what action they need to take and understand the consequences of their actions.
- **Difficulty Curve:** The game should have a gradual difficulty curve that increases as the player progresses through the levels. It is important to balance the challenge with the player's skill level to keep them engaged and motivated.
- **Playtesting:** Playtesting is an essential part of the game development process. You should regularly test the game with real users to get feedback on the game mechanics, controls, and difficulty level. This feedback can be used to make adjustments to the game to improve its learnability.

3.2.1.3 Accessibility

When designing a platformer game , it is essential to consider accessibility requirements to ensure that the game is playable by a wider audience. Here are some accessibility requirements to consider:

- **Input Options:** Provide different input options for players, such as keyboard, mouse and gamepad. Ensure that the game can be played with a single hand, as some players may have a disability that affects one hand.
- **Customizable Controls:** Allow players to customize the game controls to their liking. This feature is essential for players with disabilities that may not be able to use the default controls.
- **Text Size and Font:** Provide an option to adjust the text size and font of the game. This feature is important for players with visual impairments.
- **Colorblind Mode**: Provide a colorblind mode that adjusts the colors of the game to make it easier for colorblind players to distinguish between different objects.
- **Sound Options:** Provide different sound options, such as subtitles or closed captions for players who are deaf or hard of hearing.
- **Difficulty Options:** Provide different difficulty options, such as easy or hard mode, to accommodate players with varying skill levels.
- **Clear Visual Feedback:** Ensure that all the game mechanics, controls, and visual feedback are easy to understand and distinguish.

- **Avoid Time-Based Challenges:** Avoid time-based challenges that may be difficult for players with disabilities that affect their reaction times.

3.2.1.4 Efficiency

Efficiency requirements for a platformer game are related to the performance of the game, its load times, and its ability to run smoothly on different hardware configurations. Here are some efficiency requirements to consider:

- **Optimization:** Optimize the game code and assets to reduce load times and improve performance. Use efficient algorithms, data structures, and rendering techniques to minimize the CPU and GPU usage.
- **Level Design:** Design the levels with efficiency in mind, avoiding large areas or overly complex structures that may affect the game's performance. Use level streaming or loading techniques to keep the memory usage under control.
- **Asset Management:** Manage the game assets efficiently, using compressed formats and reducing the number of textures or models to improve load times and reduce memory usage.
- **Platform Support:** Ensure that the game can run smoothly on different hardware configurations, including low-end and high-end systems. Optimize the game for different platforms, such as PC, console, or mobile, to ensure good performance on each platform.
- **Memory Management:** Implement efficient memory management techniques, such as object pooling or garbage collection, to reduce memory fragmentation and improve performance.
- **Testing:** Test the game on different hardware configurations and use profiling tools to identify performance bottlenecks. Address any performance issues identified during testing to improve the game's efficiency.

3.2.1.5 Memorability

Memorability requirements for a platformer game are related to the ability of the player to remember how to play the game after a break or when returning to the game after some time. Here are some memorability requirements to consider:

- **Tutorial and Onboarding:** Include a tutorial and onboarding process that is easy to understand and remember. This will help the player remember the game mechanics and controls when returning to the game after a break.
- **Consistency:** Ensure that the game mechanics, controls, and visual feedback are consistent throughout the game. This will make it easier for the player to remember how to play the game when returning to it.
- **Clear Objective:** Provide a clear objective or goal for the player to remember. This will help the player understand what they need to do when returning to the game after a break.
- **Save Progress:** Include a save progress feature that allows the player to pick up where they left off. This will help the player remember where they were in the game and what they need to do next.
- **Reminders:** Include reminders or hints that help the player remember how to play the game. This could be in the form of pop-ups or visual cues that highlight important game mechanics or controls.
- **Feedback:** Provide clear and immediate feedback to the player. This will help the player remember the consequences of their actions and what they need to do next.

3.2.1.6 Errors

- The error rate is lower than the current error rate.

- If an error occurs it can be edited and corrected immediately.

3.2.1.7 Satisfaction

Satisfaction requirements for a platformer game are related to the player's emotional response to the game and their overall satisfaction with the game experience. Here are some satisfaction requirements to consider:

- **Engaging Gameplay:** Provide engaging gameplay that is challenging but not frustrating. This will keep the player motivated and interested in playing the game.
- **Interesting Storyline:** Include an interesting storyline or narrative that captures the player's attention and provides motivation to progress through the game.
- **Reward System:** Implement a reward system that incentivizes the player to complete objectives and progress through the game. This could include unlocking new levels, power-ups, or cosmetic items.
- **Replayability:** Provide replayability features, such as different difficulty modes or randomized levels, to encourage the player to play the game again and again.
- **Visual Appeal:** Create visually appealing graphics and animations that enhance the player's immersion in the game world.
- **Audio Design:** Create an immersive audio design that includes sound effects and music that enhances the player's emotional response to the game.
- **Accessibility:** Ensure that the game is accessible to a wide range of players, including those with disabilities, to provide a more inclusive and satisfying gaming experience.

3.2.1.8 Capacity

Capacity requirements for a platformer game are related to the storage and processing resources required to develop and run the game. Here are some capacity requirements to consider:

- **Storage Capacity:** Estimate the amount of storage capacity required for game assets such as levels, characters, textures, sound effects, and music. Ensure that there is sufficient storage capacity available for the development and distribution of the game.
- **Processing Power**: Estimate the amount of processing power required to render the game graphics and animations, calculate physics, and handle game logic. Ensure that there is sufficient processing power available for the game to run smoothly and efficiently.
- **Cloud Infrastructure:** Consider using cloud infrastructure services such as Amazon Web Services or Microsoft Azure to scale up the capacity of the game. This can help reduce costs and improve scalability.
- **Hardware Compatibility:** Ensure that the game is compatible with a wide range of hardware configurations, including low-end and high-end systems. Optimize the game for different platforms, such as PC, console to ensure good performance on each platform.

3.2.2 Organizational Requirements

3.2.2.1 Availability

Availability requirements for a platformer game are related to ensuring that the game is available and accessible to players at all times. Here are some availability requirements to consider**:**

- **Uptime:** Ensure that the game is available to players for the majority of the time. This means minimizing server downtime and addressing any technical issues that could impact the game's

availability.

- **Accessibility:** Make the game accessible to a wide range of players, including those with disabilities. Ensure that the game is compatible with assistive technologies and that it meets accessibility guidelines.
- **Responsiveness:** Ensure that the game responds quickly to player input and provides a smooth and lag-free gameplay experience. This means optimizing game code, minimizing network latency, and using efficient server infrastructure.
- **Scalability:** Design the game architecture to be scalable, so that it can handle increases in player traffic and usage without impacting availability or performance.
- **Disaster Recovery:** Have a disaster recovery plan in place in case of unexpected events that could impact the availability of the game, such as server crashes or natural disasters.

3.2.2.2 Latency

Latency requirements for a platformer game are related to ensuring that the game responds quickly and smoothly to player inputs, providing a seamless and enjoyable gameplay experience. Here are some latency requirements to consider:

- **Input Responsiveness:** Ensure that the game responds quickly and accurately to the player's input, with minimal delay or lag. This is particularly important for a platformer game where precise timing and movement are crucial.
- **Game Engine Optimization:** Optimize the game engine to reduce processing time and minimize input lag. This can be achieved by using efficient algorithms, reducing unnecessary computations, and optimizing the game engine's architecture.
- **Hardware Optimization:** Optimize the game for different hardware configurations to minimize input lag. This can be achieved by optimizing the game engine for different CPU and GPU architectures, and testing the game on a wide range of hardware configurations.
- **Player Feedback:** Provide immediate and clear feedback to the player when they perform an action. This can include visual and audio cues, such as animation or sound effects, to indicate that the game has registered the player's input.

3.2.2.3 Monitoring

Monitoring requirements for a platformer game project are related to the measurement and management of various aspects of the game during development and after release. Here are some monitoring requirements to consider:

- **Performance Monitoring:** Monitor the performance of the game, including frame rate, loading times, and resource utilization, to identify and diagnose performance issues.
- **Bug Monitoring:** Implement a bug tracking system to monitor and track reported bugs and issues. This can help prioritize and manage bug fixes.
- **User Behavior Monitoring:** Monitor user behavior, including playtime, progression, and retention, to understand how users interact with the game and identify areas for improvement.
- **Server Monitoring:** Monitor game servers to ensure that they are running smoothly and efficiently, and to identify and diagnose issues that may impact gameplay.
- **Security Monitoring:** Implement security monitoring to detect and prevent cheating, hacking, and other security issues that may impact the integrity of the game.
- **Analytics:** Implement analytics tools to measure and analyze user behavior, engagement, and other

metrics that can inform game development and optimization.

3.2.2.4 Maintenance

3.2.2.5 Operations

3.2.2.6 Standards Compliance

3.2.2.7 Portability

3.2.3 External Requirements

3.2.3.1 Security

3.2.3.2 Protection

3.2.3.3 Authorization and Authentication

3.3 Domain requirements

Domain requirements for a platformer game refer to the specific features, mechanics, and content that are required to create a game that fits within the platformer genre. Here are some domain requirements to consider:

- **Character Movement:** Platformer games require precise and responsive character movement mechanics, such as jumping, running, and sliding, to navigate through levels and avoid obstacles.
- **Level Design:** Platformer games require well-designed levels that provide interesting challenges for players to overcome, such as jumping puzzles, enemy encounters, and environmental hazards.
- **Collectibles:** Platformer games often feature collectibles, such as coins, power-ups, or other rewards, which incentivize exploration and mastery of levels.
- **Enemies and Bosses:** Platformer games often include enemies and bosses that players must defeat to progress through levels and the game.
- **Platform Types:** Platformer games feature different types of platforms that the player must navigate, such as moving platforms, disappearing platforms, and platforms that crumble or break.
- **Power-Ups:** Platformer games often feature power-ups that grant the player temporary abilities or enhancements, such as invincibility or increased speed.
- **Music and Sound Effects:** Platformer games often feature catchy and upbeat music, as well as sound effects that enhance the player's experience and create a sense of immersion.

4 Software Design / Diagrams

4.1 Requirements Analysis

4.1.1 User Scenarios

4.1.1.1 User Scenarios List

4.1.1.2 User Scenarios Extended

4.1.2 User Cases

4.2 Behavioral Diagrams

4.2.1 Use Case Diagrams

4.2.2 Activity Diagrams

4.2.3 State Diagrams

4.2.4 Sequence Diagrams

4.2.5 Collaboration Diagrams

4.3 Data Flow Diagrams

4.4 Entry Relation

4.4.1 Database Schema Design

4.4.2 Entity Relation Diagram

4.5 Structural Diagrams

4.5.1 Class Diagram

4.5.2 Object Diagrams

4.5.3 Component Diagrams