Tecnológico de Monterrey

Santa Fe Campus

May 10th, 2023

Software Requirements Specification

Software Construction and Decision Making

Santiago Benitez - A01782813

Ian Vázquez-A01027225

Carlos Soto - A01747990

User Stories

Product Owner

1. User Story #01: Purpose of the Videogame Project
2. User Story #02: General focus of the game
3. User Story #03: Added value of the game

Videogame

1. User Story #04: Self-contained story
2. User Story #05: Combat and action elements
3. User Story #06: Sense of progression in every level
4. User Story #07: Display of important information during gameplay.
5. User Story #08 : Main Character Customization
6. User Story #09: Brief on how to play
7. User Story #10: Clear beginning and ending of the game
8. User Story #11: Interesting game mechanics
9. User Story #12: Complete In-Game Awareness

Database

1. User Story #13: Relational database.
2. User Story #14: Relational schema of the database.
3. User Story #15: Data integrity and security.
4. User Story #16: Important user information to be saved.
5. User Story #17: Game Usage Statistics.

Web

1. User Story #18: Website sections
2. User Story #19: database connection
3. User Story #20: basic security measures

Product Backlog

Functional Requirements

<u>Videogame</u>

1. Implement "Create account" screen
2. Implement "Login" screen
3. Implement a unity navmesh to specify navigable areas in the game environment, including areas where the wolf can walk, as well as obstacles
4. Implement wolf movement (pathfinding) behaviour throughout the level using NavMesh Agent (unity).
5. Develop an upgrading weapons system using a base Weapon class and create a screen, that is displayed when an "upgrade weapons" button is clicked, that allows the player to upgrade his weapons.
6. Develop a crafting weapons system using a base CraftingRecipe class and create a screen, that is displayed when an "build weapons" button is clicked, that allows the player to create/build his weapons.
7. Implement the new wolf behaviour through C# scripts when the wolf reaches a checkpoint (position goal).
8. Create main character animations with all weapons using unity animator and link each attack action to the keys "J", "K" and "L" of the keyboard.
9. Develop enemy base class in unity, create 3 instances of enemies and implement animations for all of them.
10. Implement enemy AI player chase and wolf chase.
11. Implement collision detection mechanisms to determine when an attack action occurs on any character and deduct health from the health bar depending on the damage of the weapon used.
12. Create game over screen to be displayed when the player's health bar or the wolf's health is depleted.
13. Design levels that showcase the overall setting, context and vibe of the game's story.
14. Develop a dialogue system to help the player learn about the game
15. Create a canvas game object in the game scene hierarchy and place the following UI elements on the canvas: health bars, health potions, player's tools, inventory, collected items, attack controls, upgrade, build and pause buttons
16. Add descriptive controls to the play screen
17. Implement intuitive game level design using visual cues that help the player distinguish interactable objects.
18. Create pop-up messages or prompts that appear during gameplay to introduce the game mechanics

19. Create a game introduction cutscene with unity timeline.
20. Create a game over cutscene with unity timeline.
21. Add sounds to every action the player does.
22. Implement visual effects to let the player know when he has lost health or caused damage to enemies.
23. Create weapon inventory
24. Implement weapon selection system
25. Implement cooldown effect on weapon inventory

<u>Database</u>

1. Create a relational database schema in MySQL that is in third normal form and has the maximum number of integrity constraints.
2. Design the use case diagrams that represent the functionality of the system and provide a high-level view of the system's behavior.
3. Add different integrity elements present in MySQL such as Primary Key and Foreign Key.
4. Create SQL views to restrict access to sensitive data and ensure data privacy.
5. Create tables that represent the relational schema in a sql script
6. Create sql connection to the website using database credentials
7. Create database connection with unity in the corresponding C# script to store statistics on how players play the game
8. Store the statistics in a "statistics" table in the database

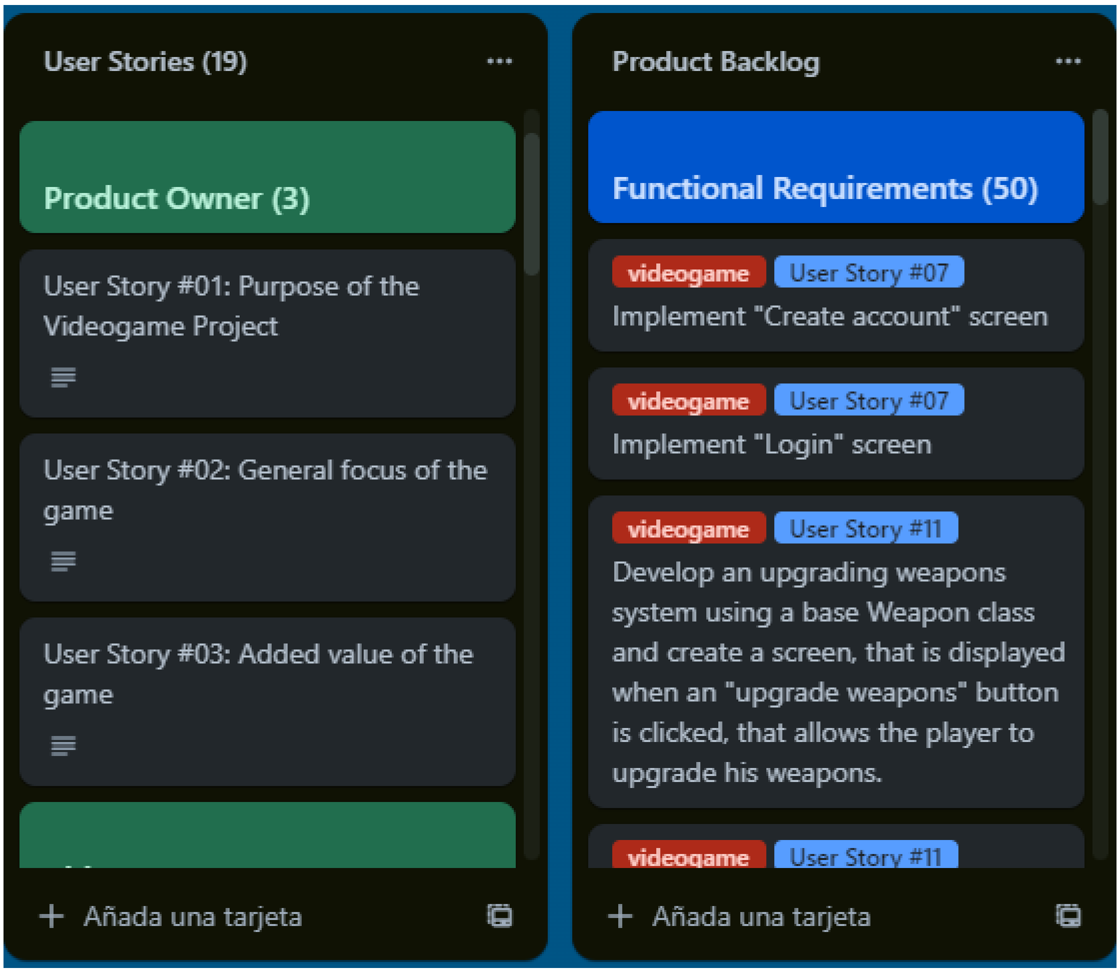
<u>Web</u>

1. Create the application programming interface (api) to allow communication between client and server
2. Create the frontend of the website using html and css
3. Create the backend of the website using Node.js and express.js
4. Host website to Amazon Web Services (AWS)
5. Escape input characters in user forms to avoid SQL injections
6. Add about section to the website that includes information about the game.
7. Add Statistics section to the website where game statistics graphs are displayed.
8. Add manual section to the website to explain how to play the game.
9. Embed the game to the website.
10. Add credits section to the website
11. Add Contact section to the website.
12. Add Footer section with copyright notice, logo and social media icons
13. Design website logo
14. Implement Navbar with links to the sections about, manual, play, statistics, credits.

Non-Functional Requirements

1. Do research on Role Playing Games (RPG)
2. Develop the idea of the game in a Game Design Document (GDD) describing the summary of the game and the gameplay considering an interesting storyline, the setting of the game and a focus on role playing.
3. Develop game mechanics where the player's decisions affect the game experience and explain them in detail in the Game Design Document (GDD)
4. Develop an idea for a game that is innovative and provides the player a feeling of novelty. Write this down in the Game Design Document (GDD)

Link to Trello board: https://trello.com/b/z9AW4AGS/reto-tc2005b



User Stories (19)

Product Owner (3)

User Story #01: Purpose of the Videogame Project

User Story #02: General focus of the game

User Story #03: Added value of the game

Añada una tarjeta

Product Backlog

Functional Requirements (50)

videogame User Story #07
Implement "Create account" screen

videogame User Story #07
Implement "Login" screen

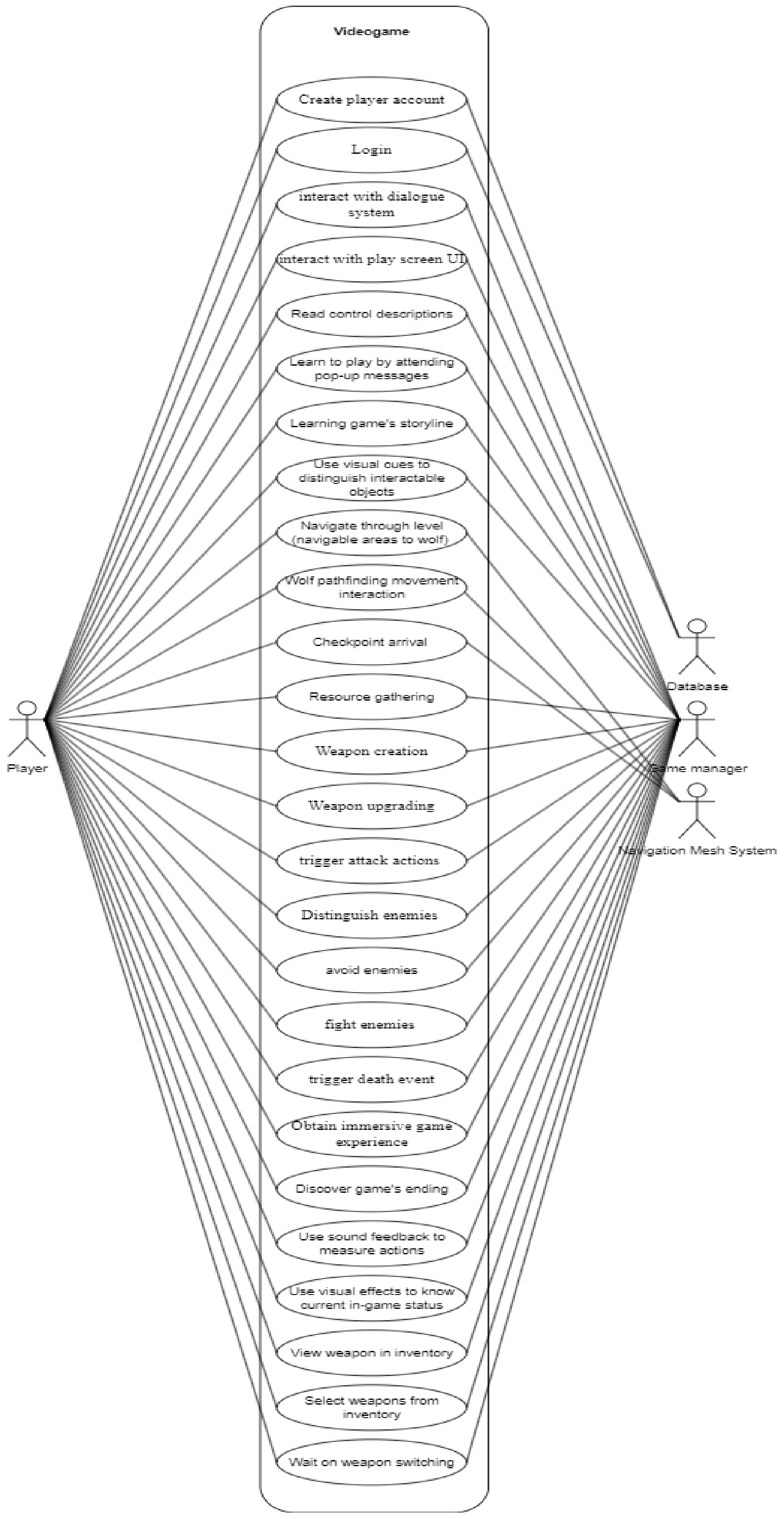
videogame User Story #11
Develop an upgrading weapons system using a base Weapon class and create a screen, that is displayed when an "upgrade weapons" button is clicked, that allows the player to upgrade his weapons.

videogame User Story #11

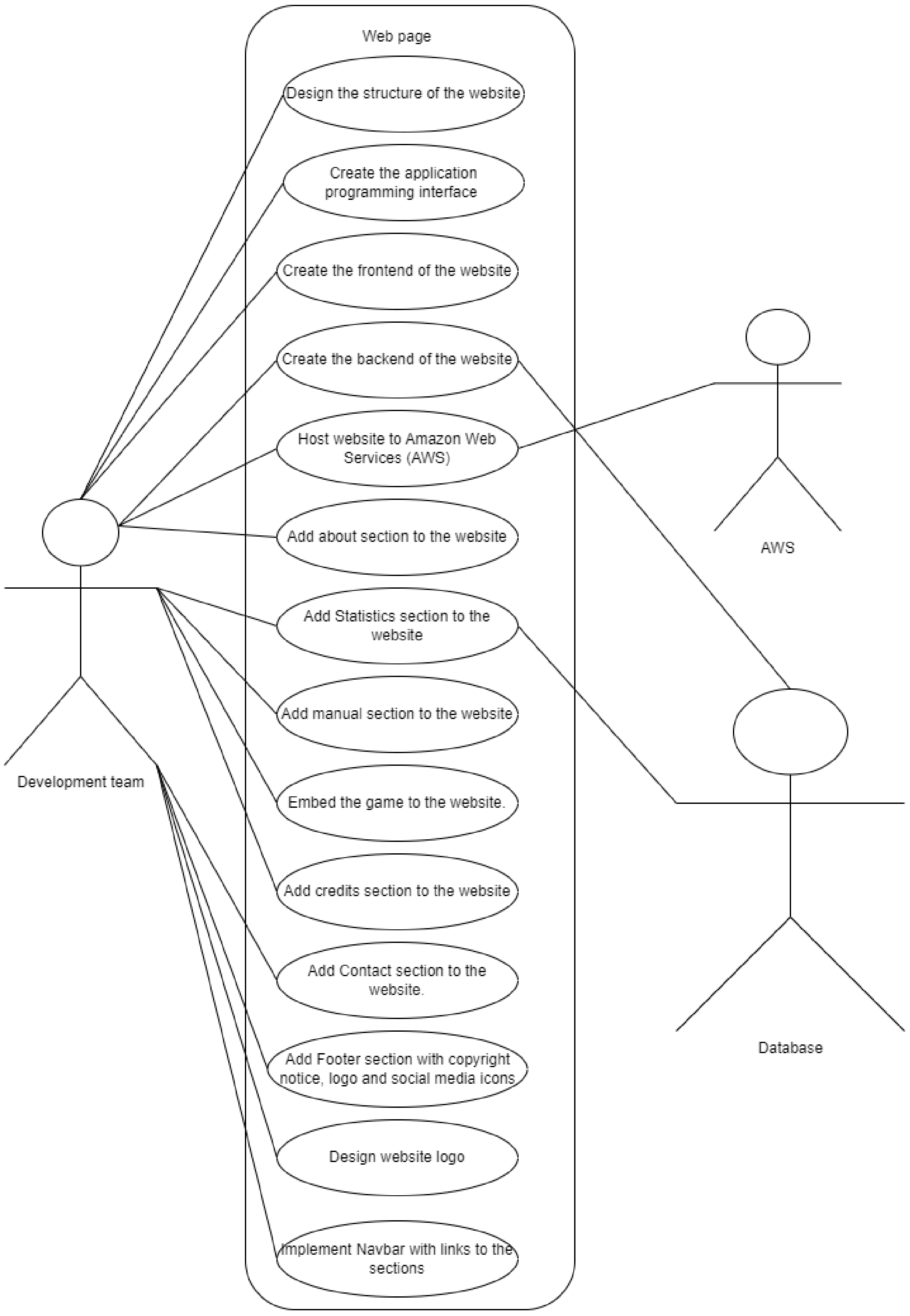
Añada una tarjeta

Use Case Diagrams

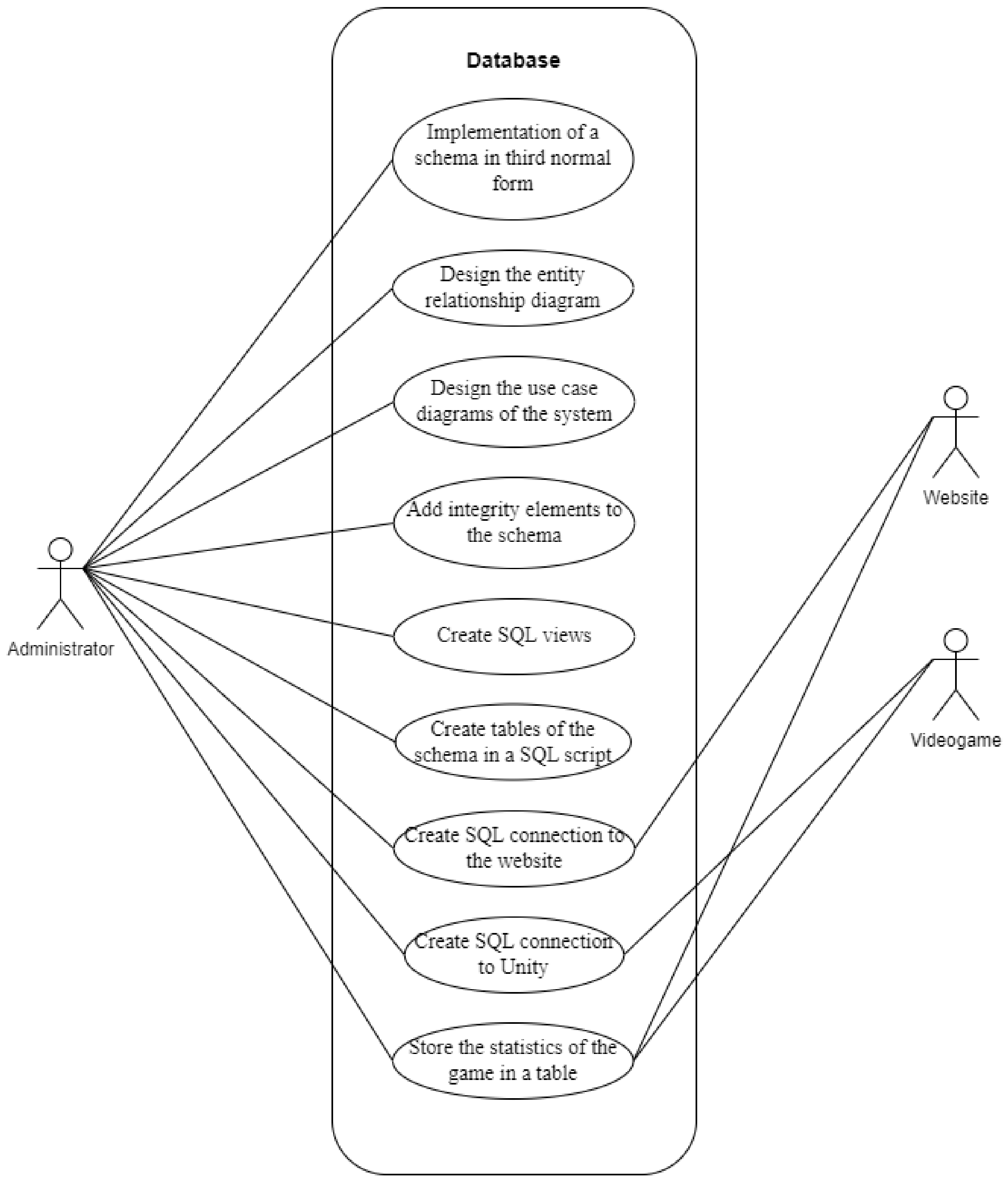
Videogame


Videogame
Create player account
Login
interact with dialogue system
interact with play screen UI
Read control descriptions
Learn to play by attending pop-up messages
Learning game's storyline
Use visual cues to distinguish interactable objects
Navigate through level (navigable areas to wolf)
Wolf pathfinding movement interaction
Checkpoint arrival
Resource gathering
Weapon creation
Weapon upgrading
trigger attack actions
Distinguish enemies
avoid enemies
fight enemies
trigger death event
Obtain immersive game experience
Discover game's ending
Use sound feedback to measure actions
Use visual effects to know current in-game status
View weapon in inventory
Select weapons from inventory
Wait on weapon switching
Player
Database
Game manager
Navigation Mesh System

Web


Web page
Design the structure of the website
Create the application programming interface
Create the frontend of the website
Create the backend of the website
Host website to Amazon Web Services (AWS)
Add about section to the website
Add Statistics section to the website
Add manual section to the website
Embed the game to the website.
Add credits section to the website
Add Contact section to the website.
Add Footer section with copyright notice, logo and social media icons
Design website logo
Implement Navbar with links to the sections
Development team
AWS
Database

Database


Database
Implementation of a schema in third normal form
Design the entity relationship diagram
Design the use case diagrams of the system
Add integrity elements to the schema
Create SQL views
Create tables of the schema in a SQL script
Create SQL connection to the website
Create SQL connection to Unity
Store the statistics of the game in a table
Administrator
Website
Videogame

Description of Use Cases

Videogame

Create Player Account

User Case Name	Create Player Account
Related Requirements	Implement "Create account" screen
Goal in context	Allow the player to create a player account
Preconditions	Click on “Create Account” button
Successful end condition	Player account is successfully created
Failed end condition	Player account is not created
Primary Actors	Player (user)
Secondary Actors	Database
Trigger	Create account event assigned to input form button
Main flow	1. Player clicks on the “Create Account” button. 2. Player fills up the “Create account ” form. 3. Player submits account information. 4. Result of the operation is notified to the player.
Extensions	1. Server Error handling operation

Login

User Case Name	Login
Related Requirements	Implement "Login" screen
Goal in context	Allow the player to login into his/her account
Preconditions	Click on “Login” button
Successful end condition	The player is successfully logged into his/her account
Failed end condition	The player’s login request is rejected.

Primary Actors	Player
Secondary Actors	Database
Trigger	Login event assigned to input form button
Main flow	1. Player clicks on "Login" button. 2. Player uses username and password to authenticate. 3. Player submits credentials. 4. Player is logged into the account.
Extensions	1. Invalid credentials. 2. Database fails to validate credentials.

Resource Gathering

User Case Name	Resource gathering
Related Requirements	Develop a resource gathering system that allows the player to interact with game objects (trees, rocks and healing remedies) using tools (hand axe, pick axe), collect the resources and save them to an inventory of collected items.
Goal in context	Allow the player to gather resources by interacting with the game world
Preconditions	Learn to use initial player tools (hand axe, pick axe)
Successful end condition	The player collects resources and they are saved to an inventory of collected items.
Failed end condition	The resource gathering system fails to allow the player to collect resources.
Primary Actors	Player
Secondary Actors	Game manager
Trigger	The player interacts with game world objects (trees, rocks)
Main flow	1. The player uses keys "X" and "Z" to use the hand axe or the pick axe. 2. The player chops down trees or mines the rocks. 3. The player collects items (wood logs, rocks) 4. The items are saved to the collected items inventory.
Extensions	None

Navigate through level (navigable areas to wolf)

User Case Name	Navigate through level (navigable areas to wolf)
Related Requirements	Implement a unity navmesh to specify navigable areas in the game environment, including areas where the wolf can walk, as well as obstacles
Goal in context	Allow the player to navigate through the wolf's navigable areas.
Preconditions	The player starts following the wolf.
Successful end condition	The player follows the wolf without wandering around the game world.
Failed end condition	The player wanders around the game world without following the wolf.
Primary Actors	Player
Secondary Actors	Navigation mesh system
Trigger	Game introduction cutscene ends.
Main flow	1. The player starts following the wolf through the game world. 2. The player pauses the wolf movement to gather resources. 3. The player resumes wolf movement and starts following it again. 4. The player reaches a navigable checkpoint area.
Extensions	None

Wolf pathfinding movement interaction

User Case Name	Wolf pathfinding movement interaction
Related Requirements	Implement wolf movement (pathfinding) behavior throughout the level using NavMesh Agent (unity).
Goal in context	Allow the player to interact with the wolf agent that moves around the game level randomly to different checkpoints.
Preconditions	The player learns controls to control wolf movement.
Successful end condition	The player can successfully stop or resume the wolf agent movement.
Failed end condition	The player is not able to stop or resume the wolf agent movement.

Primary Actors	Player
Secondary Actors	Navigation mesh system
Trigger	The player presses key “H”
Main flow	1. The player follows the wolf. 2. The player pauses wolf movement with the “H” key. 3. The player resumes wolf movement with the “H” key.
Extensions	None

Checkpoint arrival

User Case Name	Checkpoint arrival
Related Requirements	Implement the new wolf behavior through C# scripts when the wolf reaches a checkpoint (position goal).
Goal in context	Allow the player to identify when a checkpoint is reached
Preconditions	The wolf reaches a checkpoint
Successful end condition	The player is able to determine that a checkpoint has been reached
Failed end condition	The player is unaware that a checkpoint has been reached
Primary Actors	Player
Secondary Actors	Navigation mesh system
Trigger	Wolf reaches a checkpoint and changes movement behavior
Main flow	1. The player follows the wolf. 2. The player keeps track of the wolf position. 3. The wolf changes its movement behavior. 4. The player identifies a checkpoint has been reached.
Extensions	1. Wolf halts movement because of nav mesh error and player does not reach actual checkpoint.

Weapon creation

User Case Name	Weapon creation

Related Requirements	Develop a crafting weapons system using a base CraftingRecipe class and create a screen, that is displayed when an "build weapons" button is clicked, that allows the player to create/build his weapons.
Goal in context	Allow the player to build weapons with gathered resources.
Preconditions	The player collects items to build weapons.
Successful end condition	The player creates weapons.
Failed end condition	The player is not able to create weapons.
Primary Actors	Player
Secondary Actors	Game manager
Trigger	Player clicks on “Build” button
Main flow	1. The player collects items. 2. The player clicks on the “build” button. 3. The player selects a weapon to build. 4. The player builds a weapon.
Extensions	1. The player runs out of available items and cannot build weapons.

Weapon upgrading

User Case Name	Weapon upgrading
Related Requirements	Develop an upgrading weapons system using a base Weapon class and create a screen, that is displayed when an "upgrade weapons" button is clicked, that allows the player to upgrade his weapons.
Goal in context	Allow the player to upgrade weapons.
Preconditions	The player has at least one weapon in the inventory.
Successful end condition	The player successfully upgrades a weapon.
Failed end condition	The player’s weapon is not upgraded.
Primary Actors	Player
Secondary Actors	Game manager
Trigger	Player clicks on “upgrade” button

Main flow	1. Player clicks on “upgrade weapon” 2. Player selects a weapon to upgrade. 3. Player selects the weapon's characteristic to upgrade. 4. Player uses collected items to upgrade the weapon. 5. Player’s items are reduced. 6. Player’s weapon is upgraded.
Extensions	1. The player does not have a weapon to upgrade 2. The player does not have enough items to upgrade a weapon.

trigger attack actions

User Case Name	trigger attack actions
Related Requirements	Create main character animations with all weapons using unity animator and link each attack action to the keys "J", "K" and "L" of the keyboard.
Goal in context	Allow the player to trigger attack actions.
Preconditions	The player has previously created at least one weapon.
Successful end condition	The player uses keys “J”, “K” and “L” to trigger attack actions.
Failed end condition	The player can not execute attack actions.
Primary Actors	Player
Secondary Actors	Game manager
Trigger	Player presses keys. (J, K, L)
Main flow	1. The player presses either key. 2. The player executes attack action as a result.
Extensions	none

Distinguish enemies

User Case Name	Distinguish enemies
Related Requirements	Develop enemy base class in unity, create 3 instances of enemies and implement animations for all of them.

Goal in context	The player is able to distinguish the 3 different types of enemies.
Preconditions	The player reaches a checkpoint.
Successful end condition	The player recognizes grims, impalers and marauders enemies,
Failed end condition	The player does not recognize the 3 enemies.
Primary Actors	Player
Secondary Actors	Game manager
Trigger	The player reaches a checkpoint and enemies start to come out.
Main flow	1. The player reaches a checkpoint 2. The player notices grim enemies. 3. The player notices impaler enemies. 4. The player notices marauder enemies.
Extensions	none

Avoid enemies

User Case Name	avoid enemies
Related Requirements	Implement enemy AI player chase and wolf chase.
Goal in context	The player and wolf are able to avoid/dodge enemies.
Preconditions	Player reaches a checkpoint.
Successful end condition	Enemies start following the player and the wolf.
Failed end condition	The enemies don't follow the player and the wolf in the desired manner.
Primary Actors	Player
Secondary Actors	Game manager
Trigger	Reaching a checkpoint.
Main flow	1. The player reaches a checkpoint 2. The player notices the enemies start following him 3. Enemies start following the wolf 4. The player is able to dodge enemies 5. The wolf avoids/dodges enemies.

Extensions	1. Enemies don't follow the player 2. Enemies don't follow the wolf

Fight enemies

User Case Name	Fight enemies
Related Requirements	Implement collision detection mechanisms to determine when an attack action occurs on any character and deduct health from the health bar depending on the damage of the weapon used.
Goal in context	The player is able to attack enemies, deduct health and enemies can attack player and wolf and deduct health
Preconditions	The player has created at least one weapon.
Successful end condition	The player deducts health from enemies by attacking them. Enemies deduct health from player and wolf by attacking them.
Failed end condition	When an attack is executed by any character (player or enemies), health is not deducted.
Primary Actors	Player, enemies
Secondary Actors	Game manager
Trigger	Player or enemies execute attack action.
Main flow	1. Player uses a weapon to attack enemies. 2. Player deducts health from enemies. 3. Enemies use weapons to attack player and wolf 4. Enemies deduct health from player or wolf.
Extensions	1. When an attack action is executed, no health is deducted.

Trigger death event

User Case Name	trigger death event
Related Requirements	Create a game over screen to be displayed when the player's health bar or the wolf's health is depleted.

Goal in context	To display the game over screen when the player's health bar or the wolf's health is depleted.
Preconditions	Enemies can attack the player and wolf and deduct health from them.
Successful end condition	Game over screen is displayed when the player of the wolf dies.
Failed end condition	Game over screen is not displayed.
Primary Actors	Enemies
Secondary Actors	Game manager
Trigger	The player's health bar or the wolf's health bar is depleted.
Main flow	1. Enemies attack player or wolf 2. Player's health bar is depleted as a result 3. Wolf's health bar is depleted as a result 4. Player dies or the wolf dies. 5. Game over screen is displayed.
Extensions	1. Game over screen is not displayed. 2. Game over screen is displayed early or late.

Obtain immersive game experience

User Case Name	Obtain immersive game experience
Related Requirements	Design levels that showcase the overall setting, context and vibe of the game's story.
Goal in context	Provide the player with an immersive game experience.
Preconditions	Player enters play mode.
Successful end condition	The player has a positive gaming experience.
Failed end condition	The player does not enjoy the gaming experience.
Primary Actors	Player
Secondary Actors	Game manager
Trigger	Player explores the game world.
Main flow	1. Player explores the game world. 2. Player goes through checkpoints 3. Player has good gaming experience.

Extensions	1. Player does not have a good gaming experience.

Interact with dialogue system

User Case Name	Interact with dialogue system
Related Requirements	Develop a dialogue system to help the player learn about the game
Goal in context	Allow the player to get game context from a dialogue system.
Preconditions	The player enters play mode.
Successful end condition	The player learns about the game via a dialogue system.
Failed end condition	The dialogue system fails to help the player learn about the game.
Primary Actors	Player
Secondary Actors	Game manager
Trigger	Player starts to play.
Main flow	1. Player starts to play. 2. Dialogue system lets the player know about the game 3. Player knows what to do as a result.
Extensions	None

Interact with play screen UI

User Case Name	interact with play screen UI
Related Requirements	Create a canvas game object in the game scene hierarchy and place the following UI elements on the canvas: health bars, health potions, player's tools, inventory, collected items, attack controls, upgrade, build and pause buttons
Goal in context	Allow the player to interact with the main play screen,

Preconditions	Game is loaded successfully and play mode is entered.
Successful end condition	Player interacts with UI screen and all buttons are functional.
Failed end condition	Player interacts with UI screen but some or all buttons are not functional.
Primary Actors	Player
Secondary Actors	Game manager
Trigger	Player interacts with any UI element
Main flow	1. Player starts playing 2. Player interacts with any UI element 3. Player is allowed to perform action by clicking on UI elements.
Extensions	1. UI elements don't allow the player to perform actions.

Read control descriptions

User Case Name	Read control descriptions
Related Requirements	Add descriptive controls to the play screen
Goal in context	Allow the player to learn about game controls.
Preconditions	Control descriptions are loaded onto the main play screen
Successful end condition	Player learns about game control via descriptive controls.
Failed end condition	Player does not learn about game controls.
Primary Actors	Player
Secondary Actors	Game manager
Trigger	Player notices game control all over the game screen.
Main flow	1. Player enter play mode 2. Ul game screen is loaded. 3. Descriptive controls are displayed onto game screen. 4. Player reads descriptive controls. 5. Player learns all game controls.
Extensions	None

Use visual cues to distinguish interactable objects

User Case Name	Use visual cues to distinguish interactable objects
Related Requirements	Implement intuitive game level design using visual cues that help the player distinguish interactable objects.
Goal in context	Allow the player to distinguish interactable objects.
Preconditions	Game scene is loaded.
Successful end condition	Player recognizes interactable game objects.
Failed end condition	Player fails to recognize interactable game objects.
Primary Actors	Player
Secondary Actors	Game manager
Trigger	Interactable game objects start shining or shaking.
Main flow	1. Player enters play mode 2. Interactable game objects start shining or shaking. 3. Player identifies interactable game objects. 4. Player interacts with these game objects.
Extensions	1. Player fails to identify interactable game objects because of game error.

Learn to play by attending pop-up messages

User Case Name	Learn to play by attending pop-up messages
Related Requirements	Create pop-up messages or prompts that appear during gameplay to introduce the game mechanics
Goal in context	Allow the player to learn game mechanics by attending in-game pop-up messages
Preconditions	Game scene is loaded
Successful end condition	The player learns the game mechanics by attending pop-up messages.
Failed end condition	Pop-up messages fail to help the player learn the game mechanics
Primary Actors	Player
Secondary Actors	Game manager

Trigger	Player approaches interactable objects and pop-up messages are displayed.
Main flow	1. Player enters play mode. 2. Player recognizes interactable objects. 3. Player approaches interactable objects. 4. Pop-up messages are displayed. 5. Player reads the message. 6. Player learns the related game mechanic.
Extensions	1. Player disregards pop-up message and fails to learn about the game mechanic.

Learning game's storyline

User Case Name	Learning game's storyline
Related Requirements	Create a game introduction cutscene with unity timeline.
Goal in context	Introduce the player to the storyline of the game.
Preconditions	Game is loaded.
Successful end condition	The player has a grasp of the game's storyline.
Failed end condition	The initial cutscene fails to make the player understand the game's storyline.
Primary Actors	Player
Secondary Actors	Game manager
Trigger	Initial cutscene is loaded.
Main flow	1. Game is loaded 2. Player watches game introduction cutscene 3. Player understands the game's storyline
Extensions	1. Player doesn't understand the game's storyline.

Discover game's ending

User Case Name	Discover game's ending
Related Requirements	Create a game over cutscene with unity timeline.

Goal in context	Allow the player to know how the game ends.
Preconditions	Player has passed all levels.
Successful end condition	The player discovers the game ending when he has completed all levels.
Failed end condition	The player never gets to know the game's ending even if he has completed all levels.
Primary Actors	Player
Secondary Actors	Game manager
Trigger	Player kills the final boss in the last level.
Main flow	1. Player completes all levels and reaches the final level. 2. Player engages in battle against the final boss. 3. Player kills final boss 4. Player wins the game 5. Player watches the ending scene 6. Player gets to know how the game's storyline ends
Extensions	None

Use sound feedback to measure actions

User Case Name	Use sound feedback to measure actions
Related Requirements	Add sounds to every action the player does.
Goal in context	Allow the player to be aware of what is happening in his surroundings.
Preconditions	Sounds are properly loaded in the game
Successful end condition	The player is able to identify what is going on in his surroundings by receiving sound feedback. (When he attacks, collects items, etc).
Failed end condition	The player is unaware of what is happening in his surroundings.
Primary Actors	Player
Secondary Actors	Game manager
Trigger	An action performed by the player
Main flow	1. Player enters play mode 2. Player performs an action 3. Sound is reproduced as a result of the action performed

	by the player. 4. Player is aware of what is happening in his surroundings.
Extensions	1. Sound effects don't help the player figure out what is happening.

Use visual effects to know current in-game status

User Case Name	Use visual effects to know current in-game status
Related Requirements	Implement visual effects to let the player know when he has lost health or caused damage to enemies.
Goal in context	Help the player know the consequence of an action with visual effects (If he has attacked an enemy for example, let him know that he successfully attacked the enemy by showing a visual effect)
Preconditions	Visual effects are linked to possible player actions
Successful end condition	The player knows that his actions had an impact.
Failed end condition	The player is unaware that his actions had an impact.
Primary Actors	Player
Secondary Actors	Game manager
Trigger	Player performs an action
Main flow	1. Player enters play mode 2. Player executes an action 3. Visual effects are reproduced as a result of the actions 4. Visual effects let the player know that the action performed had an impact.
Extensions	None

View weapon in inventory

User Case Name	View weapon in inventory
Related Requirements	Create weapon inventory
Goal in context	Let the player view weapons in the inventory

Preconditions	Weapons can be saved to the inventory.
Successful end condition	Player is able to view the weapons in the inventory
Failed end condition	Player cannot view weapons in the inventory
Primary Actors	Player
Secondary Actors	Game manager
Trigger	Player clicks on “inventory” button
Main flow	1. Player enters play mode 2. Player sees main game screen UI 3. Player locates “inventory” button 4. Player clicks on “inventory” button 5. Player views weapons in inventory
Extensions	None

Select weapons from inventory

User Case Name	Select weapons from inventory
Related Requirements	Implement weapon selection system
Goal in context	Allow the player to select weapons from inventory
Preconditions	Weapons can be viewed in the inventory
Successful end condition	Player selects weapons from inventory and can view them in the main play screen
Failed end condition	Player selects or cannot select weapons and they are not shown in the main play screen.
Primary Actors	Player
Secondary Actors	Game manager
Trigger	Player selects a weapon from inventory and clicks on “select” button
Main flow	1. Player opens up inventory 2. Player selects a weapon 3. Player clicks on “select” button 4. Player can view weapon in main play screen.
Extensions	None

Wait on weapon switching

User Case Name	Wait on weapon switching
Related Requirements	Implement cooldown effect on weapon inventory
Goal in context	Let the player change current weapons every 2 minutes
Preconditions	Player has selected weapons.
Successful end condition	Player can change weapons every 2 minutes
Failed end condition	Player can change weapons whenever he wants
Primary Actors	Player
Secondary Actors	Game manager
Trigger	Player selects weapons and cooldown effect on inventory is applied
Main flow	1. Player opens up inventory 2. Player selects weapons 3. Inventory screen is closed and cannot be opened up again for 2 minutes 4. Player waits for 2 minutes and is able to change weapons again.
Extensions	1. Player is able to change weapons repeatedly because of system error.

Database

User Case Name	Implementation of a schema in third normal form.
Related Requirements	Making each field in the database tables depend only on the table's primary key.
Goal in context	Avoid redundancy of the data in the database.
Preconditions	To make all tables of the schema comply with the second normal form and to avoid transitive relations between columns.
Successful end condition	The integrity of data is granted thanks to a well designed data structure, avoiding errors or accidental data elimination.

Failed end condition	There will be redundancy of data as well as errors such as insertion or update anomalies.
Primary Actors	Administrator / Database manager
Secondary Actors	None
Trigger	None
Main flow	1. Identify all statistics to be shown on the website 2. Create tables with the data in order to create the schema 3. Implement second normal form to the tables and avoid transitive relationships between columns
Extensions	None

User Case Name	Design of the entity relationship diagram.
Related Requirements	To clearly describe entities, as well as their attributes, relations and primary keys.
Goal in context	To have a well structured entity relationship diagram with all necessary components
Preconditions	All entities and attributes should be clearly identified within the system.
Successful end condition	The entity relationship diagram explains and describes clearly how the entities within the system interact with each other, as well as their attributes.
Failed end condition	The diagram fails to describe entities and show how they interact.
Primary Actors	Administrator / Database manager
Secondary Actors	None
Trigger	None
Main flow	1. Identify entities within the system 2. Indicate entities attributes. 3. Describe relations between entities 4. Verify the diagram has a clear notation.
Extensions	None

User Case Name	Design the use case diagrams of the system
Related Requirements	To have actors involved in the system identified as well as their interaction with it from the functional requirements previously identified.
Goal in context	To have well structured use diagrams of the system that successfully explain its behavior.
Preconditions	Identify functional requirements of the system from user stories.
Successful end condition	The use case diagrams are easy to understand and modify, explaining precisely and from a user perspective the system's behavior from its functional requirements.
Failed end condition	The use case diagrams fail to explain the system's behavior in an understandable way.
Primary Actors	Administrator / Database manager
Secondary Actors	None
Trigger	None
Main flow	1. Obtain functional requirements from user stories 2. Add functional requirements to the diagram such that they are easy to understand 3. Identify actors involved in the system 4. Show how primary and secondary actors interact with the system.
Extensions	None

User Case Name	Add integrity elements to the schema
Related Requirements	Add primary and foreign keys, as well as restrictions to the schema.
Goal in context	Have a complete and well structured schema for the database so data integrity is granted.
Preconditions	Implement the entity relationship diagram.
Successful end condition	Integrity elements are successfully added to the schema so it is complete and well structured
Failed end condition	Integrity elements are not added to the schema and remains incomplete
Primary Actors	Administrator

Secondary Actors	None
Trigger	None
Main flow	1. Identify the fields that are unique to each entity and can identify them 2. Add primary keys to entities 3. Add foreign keys of entities within other tables
Extensions	None

User Case Name	Create SQL views
Related Requirements	Implement SQL views from existing tables within the schema
Goal in context	Restrict access to sensitive data and ensure data privacy
Preconditions	To have and existing SQL schema with all entities and their data
Successful end condition	SQL views are successfully implemented and data privacy and integrity is granted.
Failed end condition	SQL views fail to ensure data privacy and sensitive data can be accessed.
Primary Actors	Administrator
Secondary Actors	None
Trigger	None
Main flow	1. Create a view within the schema using the command "CREATE VIEW" in SQL 2. Indicate to which tables the different views will go.
Extensions	None

User Case Name	Create tables of the schema in a SQL script
Related Requirements	Add the tables of the schema in a SQL script using the corresponding commands.
Goal in context	Start to formally implement the database in SQL.
Preconditions	To have created the corresponding tables with all their components.
Successful end	The tables are successfully implemented and can be seen in

condition	SQL.
Failed end condition	SQL won't show the tables as they are poorly implemented
Primary Actors	Administrator
Secondary Actors	None
Trigger	None
Main flow	1. Identify the corresponding commands for the creation of tables in SQL 2. Create the tables in SQL with their corresponding data.
Extensions	None

User Case Name	Create SQL connection to the website*
Related Requirements	Use an API to enable connection between the database and the website.
Goal in context	To show relevant data from the database on the website of the game.
Preconditions	To have an existing database containing all the corresponding entities, relations, and tables.
Successful end condition	The connection between the database and the website is successful and selected data is clearly shown to the user.
Failed end condition	The website fails to show data from the database.
Primary Actors	Administrator
Secondary Actors	Website
Trigger	None
Main flow	1. Configure connection with SQL from the website using the corresponding programming language.* 2. Query the database to obtain the data to be shown in the database. 3. Configure the data in HTML format. 4. Insert the data on the website.
Extensions	None

User Case Name	Create SQL connection to Unity.

Related Requirements	Identify the corresponding library to store data to SQL.
Goal in context	Establish connection between the game in Unity and the SQL database.
Preconditions	To have an existing SQL database with all the necessary fields to store the statistics of the game as well as the game.
Successful end condition	A connection is established successfully and the game's data is stored in the SQL database.
Failed end condition	The database fails to connect to the game so no data can be stored.
Primary Actors	Administrator
Secondary Actors	Videogame
Trigger	None
Main flow	1. Configure the connection within Unity using “Unity SQL”.
Extensions	None

User Case Name	Store the statistics of the game in a table
Related Requirements	Use the corresponding commands in SQL to store data in an existing table within the database.
Goal in context	Store statistics from the game in Unity within the database so it can be shown later in the website.
Preconditions	There is a successful connection between the game in Unity and teh SQL database.
Successful end condition	Data is successfully stored within a table in the database so it can be queried later for the website.
Failed end condition	The database fails to store data in a table.
Primary Actors	Administrador.
Secondary Actors	Videogame.
Trigger	None
Main flow	1. Use SQL commands to insert data of the game within a table. 2. In Unity, add functions to call the SQL commands.
Extensions	None

Website

User Case Name	Design the structure of the website
Related Requirements	Logo design requirements
Goal in context	Define the overall structure and layout of the website
Preconditions	-
Successful end condition	The website structure is designed and finalized
Failed end condition	The website structure remains undefined or incomplete
Primary Actors	Development Team
Secondary Actors	-
Trigger	Start of the website development project
Included Cases	-

User Case Name	Create the application programming interface
Related Requirements	Website structure requirements
Goal in context	Develop the API for the website to enable data communication
Preconditions	-
Successful end condition	The API is created and functional
Failed end condition	The API development fails or is incomplete
Primary Actors	Development Team
Secondary Actors	-
Trigger	After the website structure is designed
Included Cases	-

User Case Name	Create the frontend of the website
Related Requirements	API development requirements
Goal in context	Develop the user-facing interface of the website
Preconditions	The website structure is defined and documented
Successful end condition	The frontend is implemented and functional
Failed end condition	The frontend development fails or is incomplete
Primary Actors	Development Team
Secondary Actors	-
Trigger	After the website structure is designed
Included Cases	Design the structure of the website, Create the application programming interface

User Case Name	Create the backend of the website
Related Requirements	Frontend development requirements
Goal in context	Develop the server-side logic and functionality of the website
Preconditions	The website structure is defined and documented
Successful end condition	The backend is implemented and functional, connected to the database
Failed end condition	The backend development fails or is incomplete
Primary Actors	Development Team
Secondary Actors	Database
Trigger	After the website structure is designed
Included Cases	Design the structure of the website, Create the application programming interface

User Case Name	Host website to Amazon Web Services (AWS)

Related Requirements	Backend development requirements
Goal in context	Deploy the website to AWS for hosting
Preconditions	The website development is complete
Successful end condition	The website is successfully hosted on AWS
Failed end condition	The hosting process fails or encounters issues
Primary Actors	Development Team
Secondary Actors	AWS
Trigger	After the website development is complete
Included Cases	-

User Case Name	Add about section to the website
Related Requirements	Website hosting requirements
Goal in context	Include an "About" section on the website with relevant information
Preconditions	The frontend of the website is developed
Successful end condition	The "About" section is added to the website
Failed end condition	The "About" section is not added or contains incomplete information
Primary Actors	Development Team
Secondary Actors	-
Trigger	User requests to view the about section
Included Cases	Design the structure of the website

User Case Name	Add Statistics section to the website
Related Requirements	Statistics section requirements
Goal in context	Include a "Statistics" section on the website to display relevant

	data
Preconditions	The frontend of the website is developed
Successful end condition	The "Statistics" section is added and displays accurate data
Failed end condition	The "Statistics" section is not added or displays incorrect data
Primary Actors	Development Team
Secondary Actors	User requests to view the statistics section
Trigger	User requests to view the statistics section
Included Cases	Design the structure of the website

User Case Name	Add manual section to the website
Related Requirements	Manual section requirements
Goal in context	Include a "Manual" section on the website with instructional content
Preconditions	The frontend of the website is developed
Successful end condition	The "Manual" section is added to the website
Failed end condition	The "Manual" section is not added or contains incomplete content
Primary Actors	Development Team
Secondary Actors	-
Trigger	User requests to view the manual section
Included Cases	Design the structure of the website

User Case Name	Embed the game to the website
Related Requirements	Game embedding requirements
Goal in context	Integrate a game into the website
Preconditions	The frontend of the website is developed

Successful end condition	The game is successfully embedded and playable on the website
Failed end condition	The game embedding process fails or the game is not playable
Primary Actors	Development Team
Secondary Actors	-
Trigger	The user requests to play the embedded game
Included Cases	Design the structure of the website

User Case Name	Add credits section to the website
Related Requirements	Credits section requirements
Goal in context	Include a "Credits" section on the website to acknowledge contributors
Preconditions	The frontend of the website is developed
Successful end condition	The "Credits" section is added to the website
Failed end condition	The "Credits" section is not added or lacks proper acknowledgments
Primary Actors	Development Team
Secondary Actors	-
Trigger	The user requests to access the credits section
Included Cases	Design the structure of the website, Create the frontend of the website

User Case Name	Add Contact section to the website
Related Requirements	Contact section requirements
Goal in context	Include a "Contact" section on the website for user inquiries
Preconditions	The frontend of the website is developed
Successful end condition	The "Contact" section is added and functional

Failed end condition	The "Contact" section is not added or does not work properly
Primary Actors	Development Team
Secondary Actors	-
Trigger	The user requests to access the contact section
Included Cases	Design the structure of the website, Create the frontend of the website

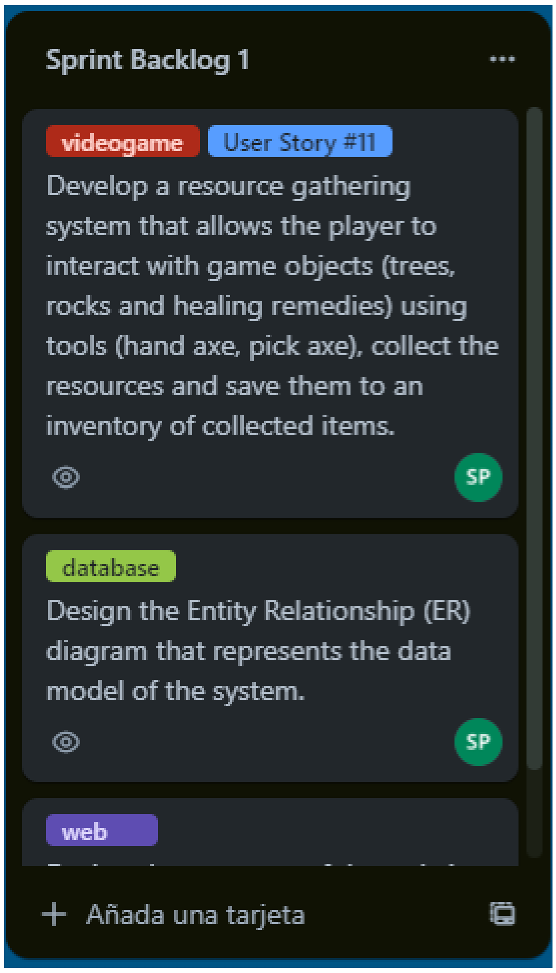
User Case Name	Add Footer section with copyright notice, logo, and social media icons
Related Requirements	Footer section requirements
Goal in context	Include a footer section at the bottom of the website with copyright notice, logo, and social media icons
Preconditions	The website is developed
Successful end condition	The footer section is added and displays the required elements
Failed end condition	The footer section is not added or lacks the required elements
Primary Actors	Development Team
Secondary Actors	-
Trigger	The user interacts with the footer section
Included Cases	Design the structure of the website, Create the frontend of the website

User Case Name	Design website logo
Related Requirements	Logo design requirements
Goal in context	Create a unique and visually appealing
Preconditions	The website structure is designed
Successful end condition	The website logo is designed and finalized
Failed end condition	The logo design process fails or does not meet the requirements

Primary Actors	Development Team (Graphic Designer)
Secondary Actors	-
Trigger	Website design process initialization
Included Cases	-

User Case Name	Implement Navbar with links to the sections
Related Requirements	Navbar functionality requirements
Goal in context	Create a navigation bar with links to different sections of the website
Preconditions	The frontend of the website is developed
Successful end condition	The navbar is implemented and displays links to sections
Failed end condition	The navbar implementation fails or does not display links properly
Primary Actors	Development Team
Secondary Actors	-
Trigger	The user accesses the website or requests navigation
Included Cases	Create the frontend of the website, Design the structure of the website

Sprint Backlog



Sprint Backlog 1

videogame User Story #11
Develop a resource gathering system that allows the player to interact with game objects (trees, rocks and healing remedies) using tools (hand axe, pick axe), collect the resources and save them to an inventory of collected items.

SP

database
Design the Entity Relationship (ER) diagram that represents the data model of the system.

SP

web

Santiago Benitez

Videojuego

1. Develop a resource gathering system that allows the player to interact with game objects (trees, rocks and healing remedies) using tools (hand axe, pick axe), collect the resources and save them to an inventory of collected items.

Database

2. Design the Entity Relationship (ER) diagram that represents the data model of the system.

Web

3. Design the structure of the website