Tecnológico de Monterrey

Break Into Valhalla

Software Requirements

Specification Document

Version 2.0

June 16th, 2023

The Einherjar Team

Joaquín Badillo, Pablo Bolio, Shaul Zayat

Table of contents

User Stories and Derived Software Requirements

Organizational User Stories

User Story	**Requirements** **Functional** **Non Functional**	**Related Requirements**	**Priority (highest/medium/ lowest)**
1.1. As a client I want to see a dynamic presentation where the project is pitched.	Create final presentation slides respecting our aesthetic.		Medium
1.2. As a client I want formality.	Practice the presentation Find a set of beta testers and test the game ourselves to detect any unexpected behaviors		Highest
1.3. As a client I would like for the game to be hosted locally and a scalable game.	Separate game behaviors with classes		Lowest
1.4. As a client I want to be informed on the development efforts (programming time) needed to complete the project.	Create a trello board to track the development efforts of each sprint. Create the software requirements specification document.		Highest

Table 1. Organizational Requirements

Database User Stories

User Story	Requirements Functional Non Functional	Related Requirements	Priority
2.1.As a user I want my database to capture my play data.	Create an ER model of the data. Detect what are the different relationships between entities. Add test data that help test CRUD operations for future API implementation.	User Story 2.4. - Implement a schema in MySQL following the third normal form. - Create all the possible restrictions for the tables like PK, FK, Not Null, Index, Unique, Check. - Implement the possible relationships among tables, following the one to one, one to many and many to many cardinalities.	Highest
2.2. As a user I want to have an account to store my game information.	Users can create accounts to store their information. Create database triggers to allow user deletion. Implement database views to create a security barrier.	User Story 2.1. - Create an ER model of the data. 2.3. - Implement CRUD operations over the database. 2.4. - Implement a schema in MySQL following the third normal form. - Create all the possible restrictions for the tables like PK, FK, Not Null, Index, Unique, Check.	Medium
2.3. As a user I want to have control over the elements stored in the database,	Implement CRUD operations over the database.		Highest
2.4. As a client I want a relational	Implement a schema in MySQL following		Highest

database in SQL.	the third normal form. Create all the possible restrictions for the tables like PK, FK, Not Null, Index, Unique, Check. Implement the possible relationships among tables, following the one to one, one to many and many to many cardinalities.		

Table 2. Database Requirements

Web Dev User Stories

User Story	Requirements Functional Non Functional	Related Requirements	Priority
3.1. As a user, I want to have a website that is inspired by the style of my video game.	Create the website's homepage Create an about page Create the info page Create the game page Create a statistics page (with at least 3 visualizations)	User Story 3.4 - Embed the video game in the website.	Medium
3.2. As a user I want my website to have a section for the video game manual.	Create a manual section on the website.		
3.3. As a user, I want to have a section on my website that displays player	Read/consult database stored data through the API	User Story 3.1 - Create a statistics view (with at least 3 visualizations)	

statistics.	Design and implement plots on the website. Create API endpoints to read and update metrics Create API endpoints to create, read and update user data
3.4. As a user I want to access the game through a webpage.	Embed the video game in the website. Deploy the website using a hosting service (Railway).
3.6 As an administrator I would like to have a website view (administrator page) to manage CRUD operations	Create an API using express js. Create an API endpoint for administrators. Create the webpage for the administrators to manage database operations.
3.7. As an API user I want to read, create and update data that allows game data to be persisted	Create API endpoints to Create, Read, Update levels. Create API endpoints to Create, Read, Update game (slot) data. Create API endpoints to Create, Read, Update stats (hp, attack, etc). Create API endpoints to Create,

	Read, Update classes (characters).		

Table 3. Web Dev Requirements

Game Dev User Stories

User Story	Requirements Functional Non Functional	Related Requirements	Priority
4.1. As a user I want a 2D RPG game.	Create the game design document Design and implement the dungeon layout as a graph. Implement the procedural generation algorithm that uses a graph to create the dungeon with static rooms. Create at least 3 variations of each type of room (treasure room, battle room, key room…) Create boss room. Create a death screen. Create enemy Spawner Add trigger events that load rooms when the player advances (similar to doors). Design and implement the UI elements for the title		

	screen Create the title screen Add a loading screen Design and implement the pause menu Create the ending screen and show credits		
4.2. As a user I want to see an attractive UI that displays relevant information for the gameplay	Create a health bar display Add a cooldown display Create an ammo display Create a glow effect for cooldown Create coins graphical display Add stamina bar to the UI		
4.3. As a user I want the game to contain some sort of tutorial.	Create panels to show the resultant behaviors from keyboard inputs.	User Story 3.2. - Create a manual section on the website.	
4.4. As a user I want enemies to do more than just approach players.	Implement A* algorithm for enemy movement Implement functionalities for enemies. Create attack animations and scripts		Lowest

	Create ranged characters	
4.5. As a user I want to know the risks going into the game.	Add NPCs to give relevant information to the user	
4.6. As a user I want the game to be fun and something I would play again.	Create unique experiences through procedural dungeon generation. Create chests.	
4.7. As a user I want an exhilarating boss fight.	Create an enemy with increased stats and difficulty (boss) Create starting boss stage (Normal attacking phase) Create second boss stage and transition: attacking and summoning phase	
4.8. As a user I want to know when my character interacts with any game object, enemy, etc.	Add audio resources related to interaction: hurt sound, slash sound, etc.	
4.9. As a user I want the gameplay to be unique and change based on each decision you make.	Create mechanics for the player to be able to upgrade their stats. Create both melee and ranged combat mechanics. Create the Shop Implement cooldown abilities	User Story 4.6 - Create unique experiences through procedural dungeon generation.

	Create dash mechanic Implement berserker mechanics Implement spellcaster mechanics. Implement archer mechanics. Create health potions Create deflect mechanic	
4.10. I want the game to be balanced but also with different types of gameplay.	Create different variables that modify player behavior (stats). Test said variables and modify them so each class feels balanced and must be played differently. Add a stamina mechanic.	
4.11. As a user I want to create an account in the game and save my progress.	Create the login screen Create an API endpoint to write to the database. Connect Unity with the API to authenticate users in plain text.	
4.12. As a user I want the game to inform me about the story and role of my character.	Use NPCs to contribute to the lore	User Story 4.3 - Add NPCs to give relevant information to the user
4.13. As a user I want the game to	Incorporate all scenes in unity with a	

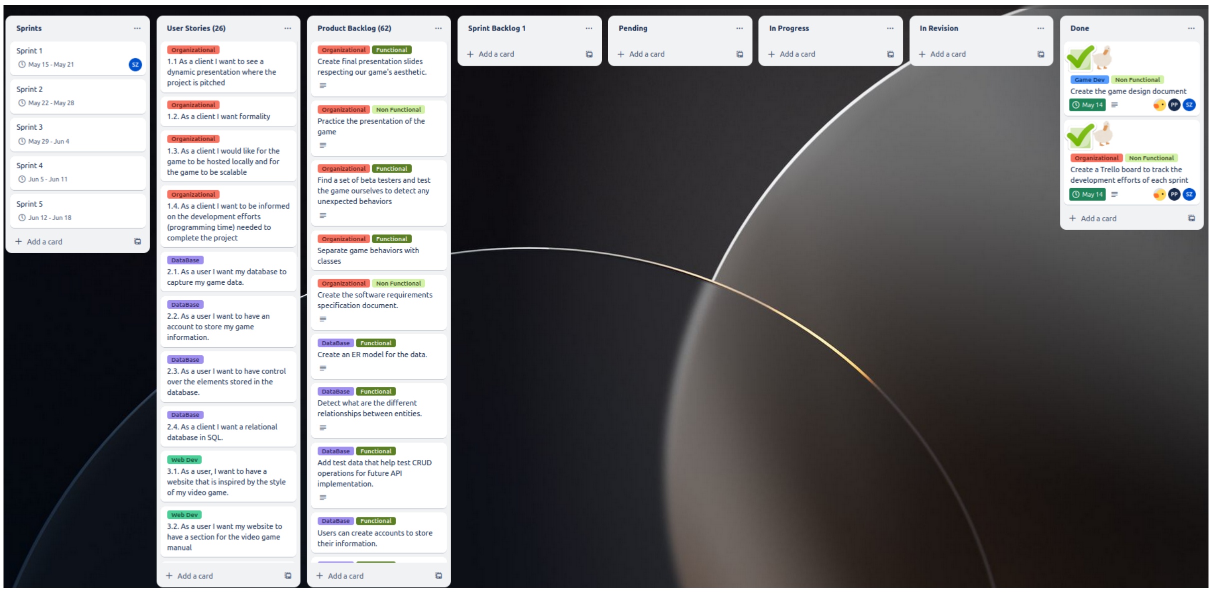
feel as though it has a beginning, ending or to be continued.	consistent timeline. Players must be able to finish the game in "equal conditions", fair enemy stats compared to player's Create the ending screen and fade out to the credits scene. Create the credits scene.	

Table 4. Game Dev Requirements

Restrictions

Use of the UML modeling language.
The project development should be completed by June 2023.
The website must be developed with HTML5 CSS3 and JavaScript technologies.
The video game must be developed with Unity technologies.
The project will be managed using the SCRUM agile methodology.
The video game must be embedded in a website.
The database must be developed in MySQL.
Complete use of github workflow.

Table 5. Software Development Restrictions



Sprints

Sprint 1
May 15 - May 21 SZ

Sprint 2
May 22 - May 28

Sprint 3
May 29 - Jun 4

Sprint 4
Jun 5 - Jun 11

Sprint 5
Jun 12 - Jun 18

+ Add a card

User Stories (26)

Organizational
1.1 As a client I want to see a dynamic presentation where the project is pitched

Organizational
1.2. As a client I want formality

Organizational
1.3. As a client I would like for the game to be hosted locally and for the game to be scalable

Organizational
1.4. As a client I want to be informed on the development efforts (programming time) needed to complete the project

DataBase
2.1. As a user I want my database to capture my game data.

DataBase
2.2. As a user I want to have an account to store my game information.

DataBase
2.3. As a user I want to have control over the elements stored in the database.

DataBase
2.4. As a client I want a relational database in SQL.

Web Dev
3.1. As a user, I want to have a website that is inspired by the style of my video game.

Web Dev
3.2. As a user I want my website to have a section for the video game manual

+ Add a card

Product Backlog (62)

Organizational Functional
Create final presentation slides respecting our game's aesthetic.

Organizational Non Functional
Practice the presentation of the game

Organizational Functional
Find a set of beta testers and test the game ourselves to detect any unexpected behaviors

Organizational Functional
Separate game behaviors with classes

Organizational Non Functional
Create the software requirements specification document.

DataBase Functional
Create an ER model for the data.

DataBase Functional
Detect what are the different relationships between entities.

DataBase Functional
Add test data that help test CRUD operations for future API implementation.

DataBase Functional
Users can create accounts to store their information.

+ Add a card

Sprint Backlog 1

+ Add a card

Pending

+ Add a card

In Progress

+ Add a card

In Revision

+ Add a card

Done

Game Dev Non Functional
Create the game design document
May 14 PP SZ

Organizational Non Functional
Create a Trello board to track the development efforts of each sprint
May 14 PP SZ

+ Add a card

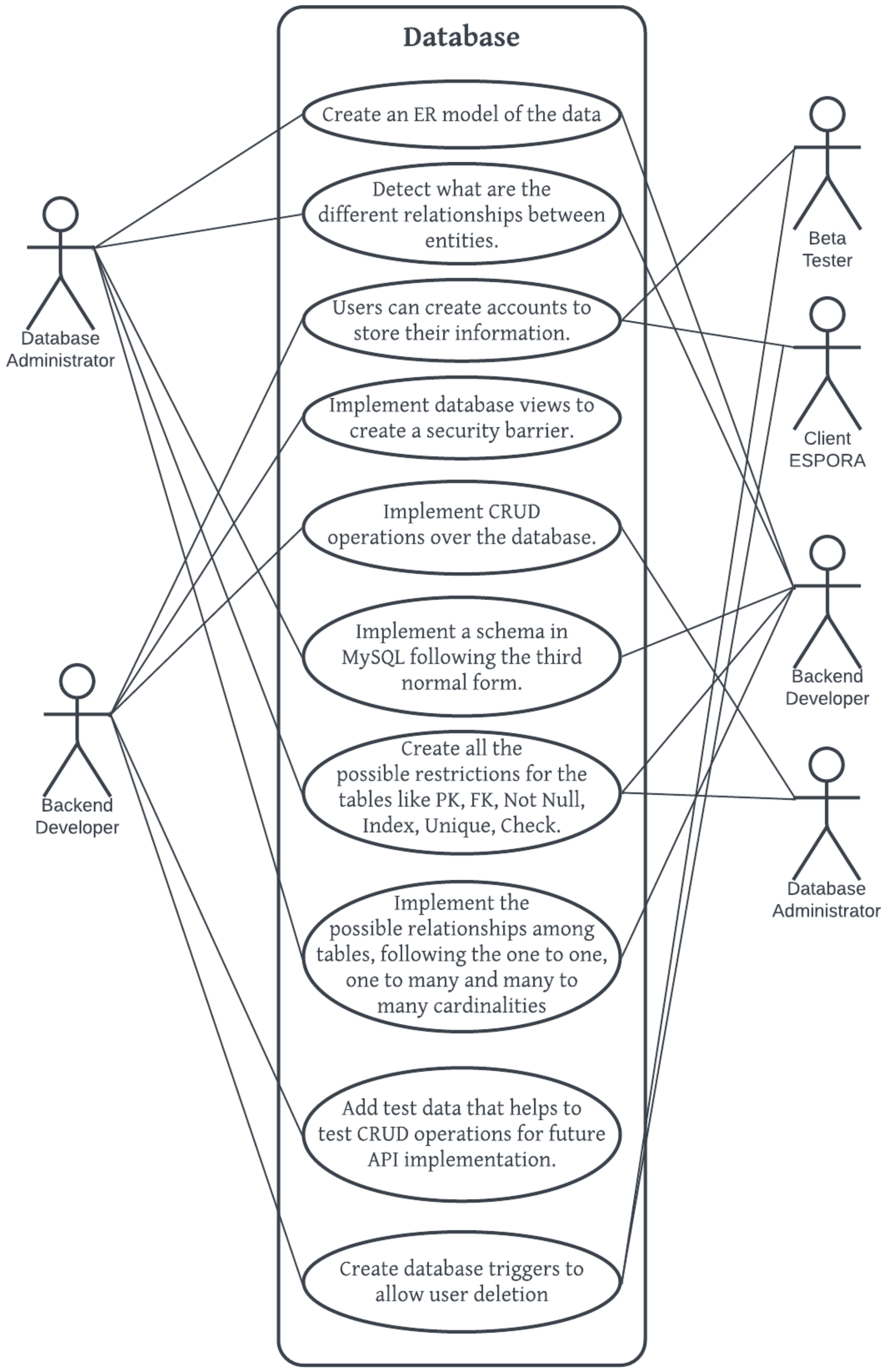
Figure 1. Trello Board

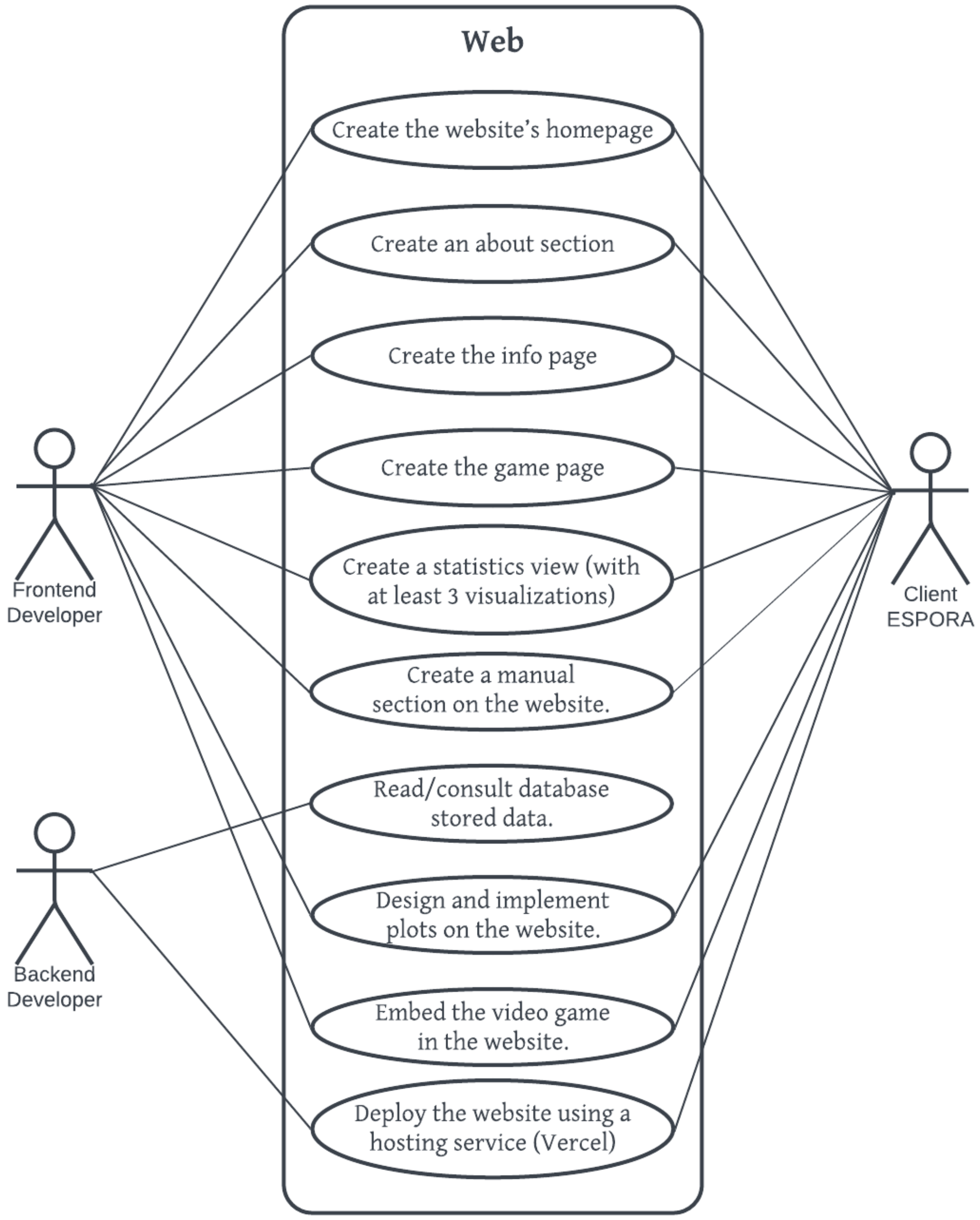
List of Requirements

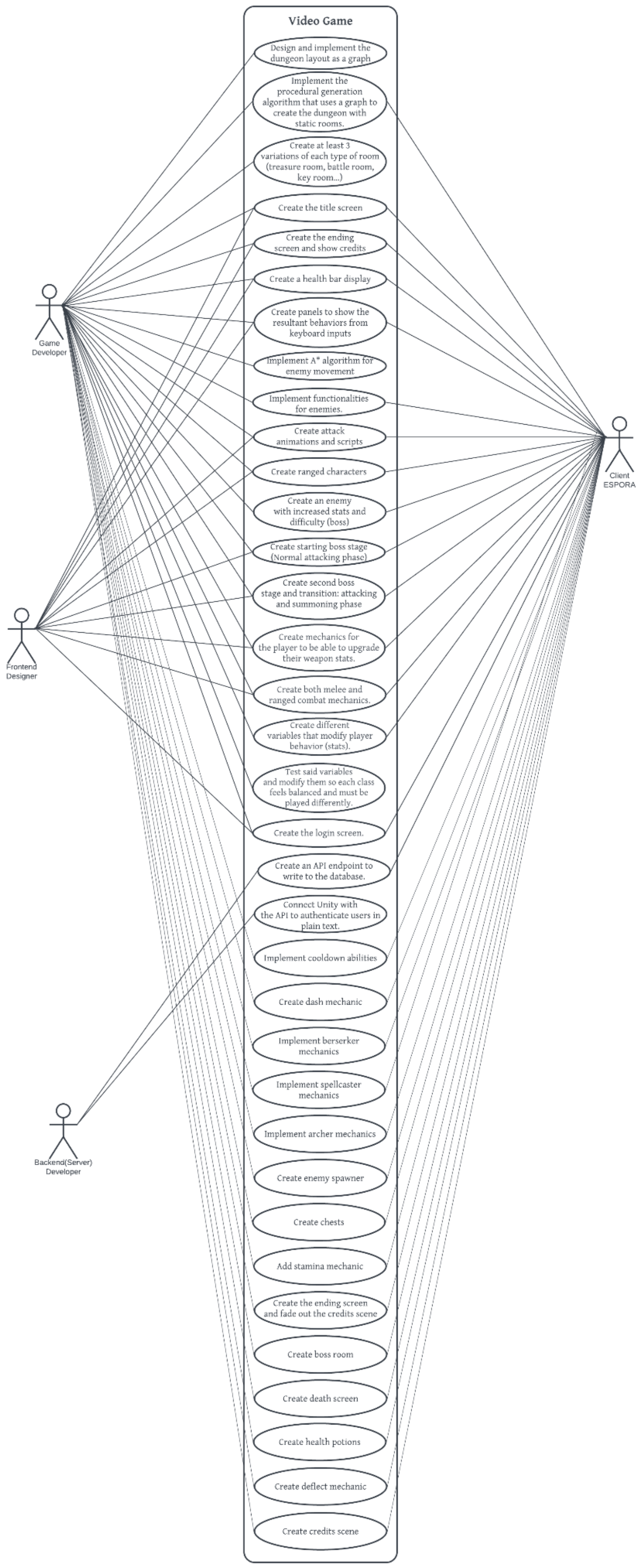
1. Create final presentation slides respecting our aesthetic.
2. Find a set of beta testers and test the game ourselves to detect any unexpected behaviors.
3. Separate game behaviors with classes.
4. Create an ER model of the data.
5. Detect what are the different relationships between entities.
6. Add test data that helps to test CRUD operations for future API implementation.
7. Users can create accounts to store their information.
8. Create database triggers to allow user deletion.
9. Implement database views to create a security barrier and optimize the search for plot data.
10. Implement CRUD operations over the database.
11. Implement a schema in MySQL following the third normal form.
12. Create all the possible restrictions for the tables like PK, FK, Not Null, Index, Unique, Check.
13. Implement the possible relationships among tables, following the one to one, one to many and many to many cardinalities.
14. Create the website's homepage.
15. Create an about page.
16. Create the info page.
17. Create the game page.
18. Create a statistics view (with at least 3 visualizations).
19. Create a manual section on the website.
20. Read/consult database stored data through the API.
21. Design and implement plots on the website.
22. Create an API using express js.
23. Design and implement the dungeon layout as a graph.
24. Implement the procedural generation algorithm that uses a graph to create the

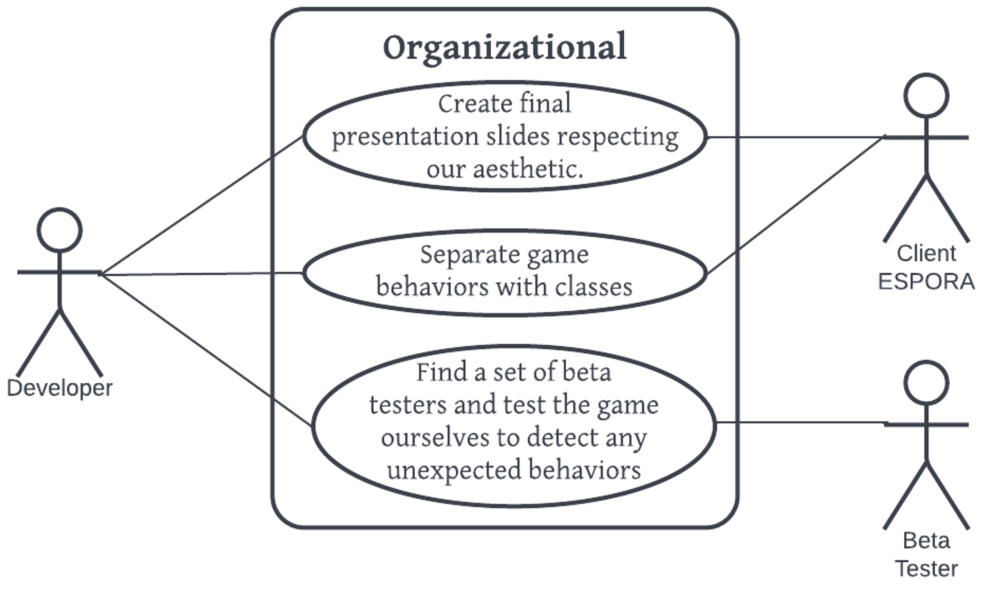
dungeon with static rooms.
25. Create at least 3 variations of each type of room (treasure room, battle room, key room…).
26. Create the title screen.
27. Create a health bar.
28. Implement A* algorithm for enemy movement.
29. Implement functionalities for enemies.
30. Create attack animations and scripts.
31. Create ranged characters.
32. Create an enemy with increased stats and difficulty (boss).
33. Create a starting boss stage (Normal attacking phase).
34. Create a second boss stage and transition: attacking and summoning phase.
35. Create mechanics for the player to be able to upgrade their stats.
36. Create both melee and ranged combat mechanics.
37. Create different variables that modify player behavior (stats).
38. Test said variables and modify them so each class feels balanced and must be played differently.
39. Create the login screen.
40. Create an API endpoint to write to the database.
41. Connect Unity with the API to authenticate users in plain text.
42. Implement cooldown abilities.
43. Create dash mechanic.
44. Implement berserker mechanics.
45. Implement spellcaster mechanics.
46. Implement archer mechanics.
47. Create enemy Spawner.
48. Create chests.
49. Add stamina mechanic.
50. Create the ending screen and fade out to the credits scene.
51. Create boss room.
52. Create death screen.
53. Create health potions.
54. Create deflect mechanic.
55. Create credits scene.

UML Use Case Diagrams


Database
Create an ER model of the data
Detect what are the different relationships between entities.
Users can create accounts to store their information.
Implement database views to create a security barrier.
Implement CRUD operations over the database.
Implement a schema in MySQL following the third normal form.
Create all the possible restrictions for the tables like PK, FK, Not Null, Index, Unique, Check.
Implement the possible relationships among tables, following the one to one, one to many and many to many cardinalities
Add test data that helps to test CRUD operations for future API implementation.
Create database triggers to allow user deletion
Database Administrator
Backend Developer
Beta Tester
Client ESPORA
Backend Developer
Database Administrator


Web
Create the website's homepage
Create an about section
Create the info page
Create the game page
Create a statistics view (with at least 3 visualizations)
Create a manual section on the website.
Read/consult database stored data.
Design and implement plots on the website.
Embed the video game in the website.
Deploy the website using a hosting service (Vercel)
Frontend Developer
Backend Developer
Client ESPORA


Video Game
Design and implement the dungeon layout as a graph
Implement the procedural generation algorithm that uses a graph to create the dungeon with static rooms.
Create at least 3 variations of each type of room (treasure room, battle room, key room...)
Create the title screen
Create the ending screen and show credits
Create a health bar display
Create panels to show the resultant behaviors from keyboard inputs
Implement A* algorithm for enemy movement
Implement functionalities for enemies.
Create attack animations and scripts
Create ranged characters
Create an enemy with increased stats and difficulty (boss)
Create starting boss stage (Normal attacking phase)
Create second boss stage and transition: attacking and summoning phase
Create mechanics for the player to be able to upgrade their weapon stats.
Create both melee and ranged combat mechanics.
Create different variables that modify player behavior (stats).
Test said variables and modify them so each class feels balanced and must be played differently.
Create the login screen.
Create an API endpoint to write to the database.
Connect Unity with the API to authenticate users in plain text.
Implement cooldown abilities
Create dash mechanic
Implement berserker mechanics
Implement spellcaster mechanics
Implement archer mechanics
Create enemy spawner
Create chests
Add stamina mechanic
Create the ending screen and fade out the credits scene
Create boss room
Create death screen
Create health potions
Create deflect mechanic
Create credits scene
Game Developer
Frontend Designer
Backend(Server) Developer
Client ESPORA


Organizational
Create final presentation slides respecting our aesthetic.
Separate game behaviors with classes
Find a set of beta testers and test the game ourselves to detect any unexpected behaviors
Developer
Client ESPORA
Beta Tester

UML Tables

Use Case 1	**Create final presentation slides respecting our aesthetic.**
Related Requirements	
Goal in Context	This presentation is intended for the closing of our work where the finished project is shown.
Preconditions	Define a style that will be the basis of the presentation.
Successful End Condition	The final presentation complies with the proposed style guidelines.
Failed End Condition	The final presentation is not aligned to the proposed style base.
Primary Actors	Developer.
Secondary Actors	Client.
Trigger	The project manager defines the style of the final presentation.
Main Flow	● Choosing the style of our project. ● Build presentation slides.
Extensions	

Use Case 2	**Find a set of beta testers and test the game ourselves to detect any unexpected behaviors.**
Related Requirements	21, 22

Goal in Context	Inviting people to test the game during development to patch any bugs.
Preconditions	Have game scenes built.
Successful End Condition	All bugs (if they exist) were discovered and fixed.
Failed End Condition	Bugs were found that were not corrected or bugs were found that were not discovered during testing.
Primary Actors	Developers.
Secondary Actors	Beta Testers.
Trigger	Beta testers need access to the game to try it out.
Main Flow	● Build playable scenes. ● Invite people to try these parts of the game. ● If errors are found, they are reported to fix them.
Extensions	

Use Case 3	**Separate game behaviors with classes**
Related Requirements	
Goal in Context	Each class in the game is separated by its characteristics and abilities with its own name.
Preconditions	Have our characters with defined characteristics and skills.
Successful End Condition	The classes are easily distinguishable from each other.
Failed End Condition	It is not possible to differentiate between one class and another since they do not have unique characteristics.
Primary Actors	Developers.
Secondary Actors	Client.
Trigger	Propose gameplay diversity.
Main Flow	● Analyze in-game world scalability. ● According to the context of the world, plan future and actual implementations of classes with unique characteristics.
Extensions	

Use Case 4	**Create an ER model of the data.**
Related Requirements	5

Goal in Context	Create an ER model that allows clarity and easy data management.
Preconditions	Define which data is to be stored.
Successful End Condition	The defined entities of the model are appropriate.
Failed End Condition	The defined entities of the model are not adequate.
Primary Actors	Database Administrator.
Secondary Actors	Backend Developer.
Trigger	Important information to be stored is known.
Main Flow	● Know the information to be stored. ● Set up the ER model according to the data.
Extensions	

Use Case 5	**Detect what are the different relationships between entities.**
Related Requirements	4
Goal in Context	The database schema is normalized and the use of database design tools to preserve the accuracy and integrity of queries.
Preconditions	Have the entities defined in the relational database schema.
Successful End Condition	Relationships between entities have a logical meaning and preserve the rules of normalization.
Failed End Condition	Relationships between entities are not specific and there is redundancy between the entities.
Primary Actors	Database Administrator.
Secondary Actors	Backend Developer.
Trigger	ER diagram is proposed.
Main Flow	● Identify all entities that describe our model. ● Identify how the entities interact with each other. ● Specify the relationships between entities. ● Add the relationships to the ER diagram. ● Verify that the model satisfies the 3rd normal form
Extensions	

Use Case 6	**Add test data that helps to test CRUD operations for future API implementation.**

Related Requirements	9
Goal in Context	Populate the database tables temporarily to verify that it is possible to Create, Read, Update and Delete elements from the schema.
Preconditions	ER model of the data.
Successful End Condition	Tables are filled with test data and all CRUD operations are executed correctly.
Failed End Condition	At least one of the CRUD operations is not executed as expected.
Primary Actors	Backend Developer.
Secondary Actors	
Trigger	Create the tables in MySQL.
Main Flow	● Try Create operation by filling tables with test data. ● Create SQL queries to test Reading, Updating and Deleting.
Extensions	

Use Case 7	**Users can create accounts to store their information.**
Related Requirements	49
Goal in Context	Users have a database where information about their progress in the game is stored.
Preconditions	Having a functional database.
Successful End Condition	The user can create an account and store his information.
Failed End Condition	The user cannot create an account.
Primary Actors	Backend Developer.
Secondary Actors	Client. Beta Tester.
Trigger	There is a login or account creation screen.
Main Flow	● Create a username field. ● Create a password field. ● User creates a username and password.
Extensions	

Use Case 8	**Create database triggers to allow user deletion**
Related Requirements	7
Goal in Context	Allow easy user deletion.
Preconditions	Have the schema implemented.
Successful End Condition	All data related to a certain user is easily deleted.
Failed End Condition	Users cannot deleted
Primary Actors	Backend Developer.
Secondary Actors	Client. Beta testers.
Trigger	An account needs to be deleted
Main Flow	● Determine the event that will fire the trigger (before deletion) ● Determine other triggers that need to be fired (e.g. trigger for levels) ● Consider exceptions and raise MySQL signals when necessary ● Use SQL to delete child tables instead of changing foreign keys to null
Extensions	

Use Case 9	**Implement database views to create a security barrier and optimize the search for plot data.**
Related Requirements	18, 20, 21
Goal in Context	Reserved queries that convert into tables that derive from the entire database to get specific and complex results faster.
Preconditions	Have functional relational database schema.
Successful End Condition	Tables obtained from specific queries work accurately and quickly.
Failed End Condition	Tables obtained from specific queries do not yield accurate and efficient results.
Primary Actors	Backend Developer.

Secondary Actors	
Trigger	Need to generate a table with specific data.
Main Flow	• Define query. • Write the query in MySQL (Create View). • Check the view.
Extensions	

Use Case 10	**Implement CRUD operations over the database.**
Related Requirements	6
Goal in Context	The database must be capable of creating, updating, reading and deleting records.
Preconditions	Having a defined relational database schema.
Successful End Condition	The operations performed on the tables work in the correct way. No records are broken and no ambiguities are created.
Failed End Condition	Tables are not modified or operations corrupt records.
Primary Actors	Backend Developer.
Secondary Actors	Database Administrator.
Trigger	You need to modify a table.
Main Flow	• Create tables in the relational database schema. • Define the appropriate restrictions in the fields of the tables so that CRUD operations work correctly.
Extensions	

Use Case 11	**Implement a schema in MySQL following the third normal form.**
Related Requirements	4
Goal in Context	The MySQL database schema is standardized with the third normal form to reduce duplicate data, anomalies and simplify data management.
Preconditions	Have a relational database model.
Successful End Condition	No redundancy or anomalies between tables.
Failed End Condition	Duplicate data, ambiguities and complicated data management are encountered.

Primary Actors	Database Administrator.
Secondary Actors	Backend Developer.
Trigger	Tables are defined within the relational database schema.
Main Flow	● Have tables in the database schema. ● Apply the third normal form between tables. ● Verify that there is no ambiguity or duplicate data.
Extensions	

Use Case 12	**Create all the possible restrictions for the tables like PK, FK, Not Null, Index, Unique, Check.**
Related Requirements	
Goal in Context	Create constraints to ensure data accuracy and integrity.
Preconditions	Have the table fields and their context defined.
Successful End Condition	The integrity of the data is preserved.
Failed End Condition	Data is corrupted or queries have low accuracy.
Primary Actors	Database Administrator
Secondary Actors	Backend Developer. Database Administrator.
Trigger	Create a table.
Main Flow	● Define the fields for the table. ● Identify the context of the data in each field. ● Implement the necessary restrictions according to the context of the data.
Extensions	

Use Case 13	**Implement the possible relationships among tables, following the one to one, one to many and many to many cardinalities.**
Related Requirements	4, 5
Goal in Context	Determine the cardinality of a relationship between entities.
Preconditions	Have a defined ER diagram.
Successful End Condition	The relationships between each entity are visible and their cardinality is logical.

Failed End Condition	The cardinality defined in the relationships between entities is not consistent.
Primary Actors	Database Administrator.
Secondary Actors	Backend Developer.
Trigger	Have the relationships between entities defined.
Main Flow	● Propose the ER model. ● Define the relationship between entities. ● According to the context of the relationship between entities. ● Define cardinality.
Extensions	

Use Case 14	**Create the website's homepage.**
Related Requirements	15, 16, 17, 18, 19
Goal in Context	Have a homepage on the website where the project will be presented.
Preconditions	Design a homepage UI in *Figma*.
Successful End Condition	The homepage is visible on the website.
Failed End Condition	The home page does not exist on the website.
Primary Actors	Frontend Developer.
Secondary Actors	Client.
Trigger	Create the HTML document for development.
Main Flow	● Propose the elements of the homepage. ● Propose an idea of the final result using graphic design tools. ● Create the HTML for the structure. ● Create the CSS for the styles.
Extensions	

Use Case 15	**Create an about page.**
Related Requirements	14, 16, 17, 18, 19
Goal in Context	Display information about the creators of the game in a section of the webpage.

Preconditions	Design an about page UI in *Figma*.
Successful End Condition	The about page is rendered correctly on the devices where the game should be supported.
Failed End Condition	The about page is not displayed as expected on a target device.
Primary Actors	Frontend Developer.
Secondary Actors	Client.
Trigger	Create the static files (HTML, CSS) for the webpage.
Main Flow	● Propose the elements of the about page. ● Propose an idea of the final result using graphic design tools. ● Create the HTML for the structure. ● Create the CSS for the styles.
Extensions	

Use Case 16	**Create the info page.**
Related Requirements	14, 15, 17, 18, 19
Goal in Context	Display the relevant information concerning the game Break Into Valhalla.
Preconditions	Design an info page UI in *Figma*.
Successful End Condition	Any user can access the information of our game in the webpage.
Failed End Condition	No information is displayed about the game.
Primary Actors	Frontend Developer.
Secondary Actors	Client.
Trigger	Create the HTML document for development.
Main Flow	● Propose the elements of the info page. ● Propose an idea of the final result using graphic design tools. ● Create the HTML for the structure. ● Create the CSS for the styles.
Extensions	

Use Case 17	**Create the game page.**
Related Requirements	14, 15, 16, 18, 19
Goal in Context	Display a space reserved for the video game on the web page.

Preconditions	Design a game page UI in *Figma*.
Successful End Condition	The game is shown in its defined space and is playable.
Failed End Condition	The game isn’t shown in its defined space and isn’t playable
Primary Actors	Frontend Developer.
Secondary Actors	Client.
Trigger	Create the HTML document for development.
Main Flow	● Propose the elements of the game page. ● Propose an idea of the final result using graphic design tools. ● Create the HTML for the structure. ● Create the CSS for the styles. ● Embed the game in it.
Extensions	

Use Case 18	**Create a statistics view (with at least 3 visualizations).**
Related Requirements	14, 15, 16, 17, 19
Goal in Context	Display relevant metrics for the players in a webpage.
Preconditions	Determine the statistics to show. Design the plots. Create the statistics section UI in *Figma*.
Successful End Condition	The statistics section is rendered correctly in all target devices, the plots show data that corresponds to the one stored in the database and plots are displayed correctly.
Failed End Condition	If the plots are incorrectly displayed, the data is not consistent with the stored results or the webpage is not rendered correctly.
Primary Actors	Frontend Developer.
Secondary Actors	Client.
Trigger	Create the HTML document for development.
Main Flow	● Propose the elements of the Statistics page. ● Propose an idea of the final result using graphic design tools. ● Create the HTML for the structure. ● Create the CSS for the styles. ● Access the database through API endpoint. ● Display them in a way they are clear and easily readable.
Extensions	

Use Case 19	**Create a manual section on the website.**
Related Requirements	14, 15, 16, 17, 18
Goal in Context	Display the video game manual on the website.
Preconditions	Have a game manual and an HTML document.
Successful End Condition	The game manual is displayed in its defined space and loads correctly.
Failed End Condition	The game manual is displayed in an unwanted space or does not load correctly.
Primary Actors	Frontend Developer.
Secondary Actors	Client.
Trigger	Create the HTML for development and have detailed instructions on how to play the game.
Main Flow	● Propose the elements of the manual section. ● Propose an idea of the final result using graphic design tools. ● Create the HTML for the structure. ● Create the CSS for the styles. ● Redact a game manual.
Extensions	

Use Case 20	**Read/consult database stored data through the API**
Related Requirements	6, 10
Goal in Context	Create API endpoints that allow developers and administrators to obtain data from the database.
Preconditions	Have a database with uncorrupted data that is properly stored.
Successful End Condition	Database records are accessed.
Failed End Condition	Database records are inaccessible.
Primary Actors	Backend Developer.
Secondary Actors	
Trigger	Create an Express.js API.
Main Flow	● Have an API endpoint implemented. ● Make simple queries to test if the read function is working properly. ● Make simple queries to test if the consult function is working properly.

Extensions	

Use Case 21	**Design and implement plots on the website.**
Related Requirements	18
Goal in Context	Display statistical graphs on the website.
Preconditions	Connect the website to the database. Have an HTML document.
Successful End Condition	Website displays plots in the sections they should be for users to read.
Failed End Condition	No plots are displayed.
Primary Actors	Frontend Developer.
Secondary Actors	Client.
Trigger	Have a HTML document and API endpoint connection.
Main Flow	● Have an API endpoint implemented. ● Find a plotting library to create the visualizations. ● Add the HTML elements, CSS styles and JS behaviors to make the plots dynamic with the data read from the database.
Extensions	

Use Case 22	**Create an API using express js.**
Related Requirements	
Goal in Context	Allow Unity to communicate with the database through API endpoints on a node server
Preconditions	Create database schema Fill database with dummy data Define operations that are required by the game flow
Successful End Condition	Endpoints allow relevant CRUD operations on the database
Failed End Condition	At least one endpoints fails to connect to the database or there are missing endpoints
Primary Actors	Backend Developer
Secondary Actors	Game Developer
Trigger	Verify that database works as expected and create screens or utilities in unity that expect data from the server

Main Flow	● Define required operations ● Define the uri of the endpoints ● Implement or find a library for the database connector ● Use SQL queries to manipulate DB as desired
Extensions	

Use Case 23	**Design and implement the dungeon layout as a graph.**
Related Requirements	24, 25
Goal in Context	Abstract the creation of the dungeon to ensure that rooms are accessible in a specific order, yet it can be procedurally generated.
Preconditions	Knowledge of graph functionality. Procedural generation algorithm knowledge.
Successful End Condition	The levels generate at random, multiplying the ways levels are configured and impeding the player from memorizing the levels.
Failed End Condition	The levels are loaded and made like a predetermined map.
Primary Actors	Game Developer.
Secondary Actors	
Trigger	Separate the dungeon design into rooms and the rooms into categories (key room, chest room, etc.)
Main Flow	● Design the abstract layout of the dungeon graphically. ● Create a graph representation in computer memory using C# that is optimized for search operations
Extensions	

Use Case 24	**Implement the procedural generation algorithm that uses a graph to create the dungeon with static rooms.**
Related Requirements	23, 25
Goal in Context	Procedural generation is implemented and levels are generated at random using prefabricated rooms.
Preconditions	Create the graph representation of the dungeon.
Successful End Condition	Random dungeons are created according to the abstract graph representation.
Failed End Condition	Dungeons are not created, they are always the same or some rooms are not accessible.
Primary Actors	Game Developer.

Secondary Actors	Client.
Trigger	Have prefabricated rooms.
Main Flow	● Create a graph representation in computer memory using C# that is optimized for search operations. ● Create a game object that stores room prefabs ● Create a script that selects rooms using random indices
Extensions	

Use Case 25	**Create at least 3 variations of each type of room (treasure room, battle room, key room…).**
Related Requirements	23, 24
Goal in Context	Create sufficient content for the procedural generation to create unique levels.
Preconditions	Create the graph representation of the dungeon. Create or find a tileset to use.
Successful End Condition	At least 3 prefabricated rooms are created for each of the room categories.
Failed End Condition	There is at least one room category that is missing variations.
Primary Actors	Game Developer.
Secondary Actors	
Trigger	Dungeon is represented as a graph and rooms are divided into categories.
Main Flow	● Create a room template to create rooms with the same size (29 by 29 tiles). ● Create different tilemaps to distinguish between different "materials" and walls. ● Design level prefabs.
Extensions	Design or find tilesets.

Use Case 26	**Create the title screen.**
Related Requirements	39, 40
Goal in Context	Have some sort of landing page in the game where players can see a leaderboard.
Preconditions	Both having a database to store statistics saved by different player runs and having beta testers to play and have save data.
Successful End	The game has a title screen that shows a leaderboard and has buttons that

Condition	take you to different screens or the game itself (options, leaderboard, play, and exit).
Failed End Condition	The game doesn't have a title screen and plays automatically.
Primary Actors	Game Developer. Frontend designer.
Secondary Actors	Client.
Trigger	A new scene that is empty.
Main Flow	● Have some mechanics implemented for the game. ● Create a new scene before the game plays that has our designs. ● Make buttons for it to work. ● Implement statistics view. ● Make it start the game if the play button is clicked.
Extensions	

Use Case 27	**Create a health bar.**
Related Requirements	
Goal in Context	Have a visual aid that shows hit points to the user at all times.
Preconditions	The implementation of hitpoints.
Successful End Condition	The health bar not only shows the player's vitality at all times but it also changes color depending on the percentile.
Failed End Condition	Health bar is either inconsistent or not showing at all.
Primary Actors	Game Developer. Frontend designer.
Secondary Actors	Client.
Trigger	Interactive game objects (player, enemies and boss).
Main Flow	● Acquire knowledge of health bar implementation. ● Implement damage dealing functions. ● Implement hit point. ● Design health bar. ● Implement graphics onto the scene and give them behavior.
Extensions	Implement hit points for enemies and make a health bar for Boss (Hel).

Use Case 28	**Implement A* algorithm for enemy movement.**
Related Requirements	

Goal in Context	Make our enemies and boss avoid obstacles while chasing after our player.
Preconditions	Enemies can move in the game.
Successful End Condition	Enemies are able to follow the player avoiding the obstacles that appear in their path.
Failed End Condition	Enemies are unable to avoid obstacles and get stuck while chasing the player.
Primary Actors	Game Developer.
Secondary Actors	
Trigger	The enemy detects the player.
Main Flow	● Understand the A* algorithm. ● Implement the behavior for path calculation and refreshing. ● Make movement change animation states. ● Add scripts to enemies.
Extensions	

Use Case 29	**Implement functionalities for enemies.**
Related Requirements	28, 30.
Goal in Context	Implement enemy behaviors like attacking and dropping loot.
Preconditions	Enemy sprites and animations.
Successful End Condition	Enemies drop loot, attack players and spawn on rooms.
Failed End Condition	At least one of the expected behaviors for enemies (attacking, dropping loot, spawning, etc).
Primary Actors	Game Developer.
Secondary Actors	Client.
Trigger	Implementation of enemy behaviors and animations.
Main Flow	● Create enemy sprites. ● Create C# script for enemy behaviors. ● Create enemy animations (walking, attacking).
Extensions	

Use Case 30	**Create attack animations and scripts.**
Related Requirements	29, 31

Goal in Context	Make the player notice when their attack starts and finishes.
Preconditions	Have the player's frames for his attack movement.
Successful End Condition	The frames create a fluid movement of the character showing an attack animation that works thanks to C# programming.
Failed End Condition	The frames create a motion that is not very fluid and does not respond as expected because of the way it was programmed.
Primary Actors	Game Developer. Frontend designer.
Secondary Actors	Client.
Trigger	Press the key that was assigned to the attacks.
Main Flow	● Design the types of attacks. ● Create the frames to be able to perform the attack animation. ● Use the Unity animator to make the sequence of frames to simulate the movement. ● Use C# to make the scripts that tell the game when these animations should be activated.
Extensions	

Use Case 31	**Create ranged characters.**
Related Requirements	30
Goal in Context	Have characters that can perform actions within a defined distance range.
Preconditions	Define the character's range of reach.
Successful End Condition	Character range actions are executed correctly and are visible.
Failed End Condition	Character range range actions do not respect the defined range or are not visible.
Primary Actors	Game Developer. Frontend designer.
Secondary Actors	Client.
Trigger	Press the key assigned to range actions.
Main Flow	● Define ranged enemy behavior and pathfinding. ● Design projectile graphics. ● Implement shooting mechanics.
Extensions	

Use Case 32	**Create an enemy with increased stats and difficulty (boss).**
Related Requirements	29
Goal in Context	Have a battle that feels completely different than the rest and make it so it feels epic and overwhelming to an extent.
Preconditions	Design or find a sprite for the boss (Hel) Create common enemy behaviors (which will be extended)
Successful End Condition	The boss fight is exhilarating.
Failed End Condition	The boss is either a pushover and is bland or isn't even implemented.
Primary Actors	Game Developer.
Secondary Actors	Client.
Trigger	Clearing other rooms.
Main Flow	● Design the spritesheet for Hel making it so her depiction is accurate. ● Implement A* pathfinding for her. ● Implement the two phases of the battle. ○ Attacking behaviors. ○ Summoning the dead while maintaining the same attack pattern. ● Implement the second phase when Hel's health bar gets to 50%. ● When Hel "dies", instead of going to the win screen, the scene will change and Hel stands up and hits the player so hard they die instantly. ● Implement a winning cutscene of either Odin or Freyja welcome the player into either Valhalla or Fólkvangr.
Extensions	

Use Case 33	**Create a starting boss stage (Normal attacking phase).**
Related Requirements	32, 34
Goal in Context	To make the final battle exciting and challenging, the final boss will have a starting stage with "normal" attacking behaviors.
Preconditions	Design or find a sprite for the boss (Hel). Create common enemy behaviors (which will be extended). Create an enemy with increased stats and difficulty.
Successful End Condition	The boss has stages and starts with normal attacking behaviors
Failed End Condition	The boss fight does not have different phases or does not follow the expected behaviors on each phase.

Primary Actors	Game Developer. Frontend designer.
Secondary Actors	Client.
Trigger	The player reaches the final boss.
Main Flow	● Implement melee attack mechanics with Hel's weapon being a Scythe.
Extensions	

Use Case 34	**Create a second boss stage and transition: attacking and summoning phase.**
Related Requirements	32, 33
Goal in Context	To make the final battle exciting and challenging, the final boss will have a second stage with frantic attacking behaviors and summoning the dead to attack for her.
Preconditions	Design or find a sprite for the boss (Hel). Create common enemy behaviors (which will be extended). Create an enemy with increased stats and difficulty. Create erratic attacking behaviors Create enemy spawner logic
Successful End Condition	The boss has stages and changes to an erratic phase with "minions" when dropped to 50% or below.
Failed End Condition	The boss fight does not have different phases or does not follow the expected behaviors on each phase.
Primary Actors	Game Developer. Frontend designer.
Secondary Actors	Client.
Trigger	The player reaches the final boss.
Main Flow	● Implement melee attack mechanics with Hel's weapon being a Scythe. ● Implement Hel summoning animation. ● Implement summoning mechanics. ● Set the summoning to be melee and ranged Draugr.
Extensions	Create an enemy with increased stats and difficulty (boss).

Use Case 35	**Create mechanics for the player to be able to upgrade their stats.**

Related Requirements	
Goal in Context	Make the player feel they're sort of leveling up by giving the feel of getting stronger.
Preconditions	Weapon sprites and animations. Weapon upgrade item sprite.
Successful End Condition	Players can upgrade their weapons through game items.
Failed End Condition	The player has no weapon upgrade.
Primary Actors	Game Developer. Frontend designer.
Secondary Actors	Client.
Trigger	Store or chest drop.
Main Flow	● Design the weapon upgrades. ● Implement a logic stat bonus for weapons.
Extensions	Design and implement chests.

Use Case 36	**Create both melee and ranged combat mechanics.**
Related Requirements	
Goal in Context	Have unique melee and ranged mechanics making the game more dynamic.
Preconditions	Have characters specifically designed for both melee and ranged mechanics.
Successful End Condition	Characters have melee and ranged combat mechanics.
Failed End Condition	Characters do not possess melee and ranged combat mechanics or do not function correctly.
Primary Actors	Game Developer. Frontend designer.
Secondary Actors	Client.
Trigger	Being in combat.
Main Flow	● Design of combat mechanics. ● Implement the animations for each combat mechanic. ● Define a cooldown for the next attack to be executed.

Extensions	

Use Case 37	**Create different variables that modify player and enemy behavior (stats).**
Related Requirements	
Goal in Context	Control the player behavior completely.
Preconditions	Player designs. Weapon designs.
Successful End Condition	Through the variables the player's behavior and feel is completely different.
Failed End Condition	Player behaves the same despite being a different class.
Primary Actors	Developer.
Secondary Actors	Client.
Trigger	Player behavior implementation.
Main Flow	● Implement the different behaviors in all classes. ● Implement the different behaviors in all enemies. ● Create the 5 variables for player stats (HP, ATK, ATKSPD, DEF, SPD). ● Modify the different scripts for compatibility with stats.
Extensions	

Use Case 38	**Test said variables and modify them so each class feels balanced and must be played differently.**
Related Requirements	37
Goal in Context	Have the modified behaviors be logical and fair.
Preconditions	Implementation of the stats for players and enemies
Successful End Condition	The different stats in both enemies and player classes feel fair and create different experiences.
Failed End Condition	There is no implementation of stats or they are overwhelmingly unfair
Primary Actors	Developers.
Secondary Actors	Client.
Trigger	Player and Enemy implementations
Main Flow	● Test the implemented stats.

	● Correct the stats if they're unfair.
Extensions	

Use Case Name 39	**Add a login screen.**
Related Requirements	40, 41
Goal in Context	There is a visible form with fields for accounts.
Preconditions	The game must exist.
Successful End Condition	When starting the game the login screen appears.
Failed End Condition	When starting the game, the login screen does not appear.
Primary Actors	Game Developer. Frontend designer.
Secondary Actors	Client.
Trigger	The user opens the game.
Main Flow	● The user enters the page to play the video game. ● When the game opens, the login screen appears.
Extensions	

Use Case 40	**Create an API endpoint to write to the database.**
Related Requirements	39, 41
Goal in Context	Allow the game to create new users in the database when they are first registered through the API.
Preconditions	Database tables for users exist and can be modified through create operations.
Successful End Condition	API endpoint is functioning properly, allowing the game to write new data in the database.
Failed End Condition	API endpoint doesn't work and the game doesn't communicate with the database.
Primary Actors	Backend (Server) Developer.
Secondary Actors	Client
Trigger	Player registers or logs into the game.

Main Flow	• Create express.js API • Create functions to write to the database • Expose functions for the game to write to the database
Extensions	

Use Case 41	**Connect Unity with the API to authenticate users in plain text.**
Related Requirements	
Goal in Context	Authenticate players with a username and password within the game.
Preconditions	Players register their accounts.
Successful End Condition	Players can only login to their accounts using the correct username/email and password combination.
Failed End Condition	Players are having problems logging into their account with their username and password, or anyone can access your account despite incorrect passwords (security breach).
Primary Actors	Backend (Server) Developer.
Secondary Actors	
Trigger	Game login page.
Main Flow	• Create login screen elements. ○ Email/username field. ○ Password field. • Create API function to authenticate email/username and password combination. • Connect unity to the API.
Extensions	

Use Case 42	**Implement cooldown abilities**
Related Requirements	30
Goal in Context	Make the player feel more powerful when needed.
Preconditions	Player mechanics.
Successful End Condition	The player is able to make the decision of activating a buff with a consequence.
Failed End Condition	No blessing is added.
Primary Actors	Developers.
Secondary Actors	Client.

Trigger	Player animations.
Main Flow	● Make a function that allows to activate with the Ctrl key something. ● Add its behavior. ● Make it deactivate after a while. ● Design an aura to let the player know. ● Make it pulse with a sine function.
Extensions	

Use Case 43	**Create dash mechanic.**
Related Requirements	30
Goal in Context	Make the player teleport to the direction they wish.
Preconditions	Player mechanics.
Successful End Condition	The player is able to use the dash mechanic to run from enemies in a pinch.
Failed End Condition	No dash is implemented.
Primary Actors	Developers.
Secondary Actors	Client.
Trigger	Player animations.
Main Flow	● Make the player change position instantly. ● Make the activation only after a certain time. ● Give it an afterimage to let the player know they dashed.
Extensions	

Use Case 44	**Implement berserker mechanics**
Related Requirements	30
Goal in Context	Make a class that is close combat.
Preconditions	Player mechanics.
Successful End Condition	The player that chooses this class is able to play with no bugs and has fun with it.

Failed End Condition	There is no berserker class.
Primary Actors	Developers.
Secondary Actors	Client.
Trigger	Player animations.
Main Flow	● Refactor the general code to adapt it to the berserker. ● Balance the class.
Extensions	

Use Case 45	**Implement spellcaster mechanics**
Related Requirements	30
Goal in Context	Make a class that is ranged and terrible at close combat, forcing the player to flee from enemies while attacking them.
Preconditions	Player mechanics.
Successful End Condition	The player that chooses this class is able to play with no bugs and has fun with it.
Failed End Condition	There is no spellcaster class.
Primary Actors	Developers.
Secondary Actors	Client.
Trigger	Player animations.
Main Flow	● Refactor the general code to adapt it to the spellcaster. ● Balance the class.
Extensions	

Use Case 46	**Implement archer mechanics**
Related Requirements	Make a class that is neutral at both ranged and close combat so players can enjoy a hybrid class.
Goal in Context	Player mechanics.
Preconditions	The player that chooses this class is able to play with no bugs and has fun with it.
Successful End Condition	There is no archer class.

Failed End Condition	Developers.
Primary Actors	Client.
Secondary Actors	Player animations.
Trigger	● Refactor the general code to adapt it to the archer. ● Balance the class.
Main Flow	
Extensions	

Use Case 47	**Create enemy spawner**
Related Requirements	29, 30
Goal in Context	Make the enemies spawn once a dungeon is created.
Preconditions	Enemies.
Successful End Condition	The procedurally generated dungeon spawns enemies in all rooms.
Failed End Condition	No enemies are spawned.
Primary Actors	Developers.
Secondary Actors	Client.
Trigger	Dungeon generation.
Main Flow	● Create the locations where enemies are to be spawned. ● Spawn game objects of the two enemies in said locations. ● Spawn them upon starting the game.
Extensions	

Use Case 48	**Create chests**
Related Requirements	
Goal in Context	Make chests for the player to find in certain rooms.
Preconditions	Potion and upgrade implementation.
Successful End Condition	Chests spawn in their respective rooms and are interactable objects.
Failed End Condition	chests don’t spawn

Primary Actors	Developers.
Secondary Actors	Client.
Trigger	Dungeon generation.
Main Flow	● Design the chest sprites. ● Create the function to interact with the chest. ● Make the chest have randomized values with an 80 to 20 ratio of getting a specific item. ● Make them spawn the drop. ● Make the chest destroy itself.
Extensions	

Use Case 49	**Add stamina mechanic.**
Related Requirements	44, 45, 46
Goal in Context	Make the player more aware of the decisions they're making and make them manage their resources.
Preconditions	The different player classes
Successful End Condition	Player has a stamina bar that shows its consumption with secondary attack activation.
Failed End Condition	Stamina mechanic not implemented.
Primary Actors	Developers.
Secondary Actors	Client.
Trigger	Secondary attack.
Main Flow	● Make the player consume a certain amount of stamina with secondary attack. ● Have variables that make it work. ● Have a coroutine that recharges stamina. ● Make a display for it.
Extensions	

Use Case 50	**Create the ending screen and fade out to the credits scene**
Related Requirements	55
Goal in Context	Make the player aware they won and show the developers of our game.
Preconditions	Boss implementation.
Successful End	The scene transition correctly fades out to the other and the player knows

Condition	they've won.
Failed End Condition	The game can't be won.
Primary Actors	Developers.
Secondary Actors	Client.
Trigger	Killing Hel.
Main Flow	● Implement a coroutine that changes the color of the screen to black little by little and then changes the scene. ● Make Hel's dying function start said coroutine.
Extensions	

Use Case 51	**Create boss room**
Related Requirements	32, 33, 34
Goal in Context	Make a room that is exclusively for the boss fight to make it seem like an epic fight.
Preconditions	Sprites for the tiles.
Successful End Condition	The room has a different aura and an eerie feel to it and works perfectly.
Failed End Condition	No boss room is implemented.
Primary Actors	Developers.
Secondary Actors	Client.
Trigger	Player exits the tavern.
Main Flow	● Make the sprite sheet for the tilemap. ● Get ambience assets. ● Organize the room to feel massive and spacious. ● Test its functionality.
Extensions	

Use Case 52	**Create death screen.**
Related Requirements	
Goal in Context	Make the player aware they lost in a stylish way.
Preconditions	Player implementation.
Successful End	Upon death, the scene fades out and a "You died" message is shown.

Condition	
Failed End Condition	The player can't lose.
Primary Actors	Developers.
Secondary Actors	Client.
Trigger	Player death
Main Flow	● Implement a coroutine that slowly fades black and changes the scene. ● Make it activate upon death. ● Return the player to the game scene and start a new game.
Extensions	

Use Case 53	**Create health potions.**
Related Requirements	48
Goal in Context	Allow the player to heal if they gain access to the drop.
Preconditions	Chest implementation.
Successful End Condition	The player is able to heal whenever they interact with a potion game object.
Failed End Condition	There is no implementation of a potion.
Primary Actors	Developers.
Secondary Actors	Client.
Trigger	Interaction with a chest.
Main Flow	● Design the potion sprites and make them fit our game aesthetic. ● Make them modify the player's current health without overflowing their HP. ● Make them destroy themselves.
Extensions	

Use Case 54	**Create deflect mechanic.**
Related Requirements	
Goal in Context	Allow the player to deflect enemy arrows back at them.

Preconditions	Enemy and player implementation
Successful End Condition	The player can deflect arrows back to enemies with the right timing.
Failed End Condition	There is no deflect mechanic.
Primary Actors	Developers.
Secondary Actors	Client.
Trigger	Attacking incoming arrows.
Main Flow	● Make the player destroy enemy arrows if they hit them with melee attacks. ● Spawn a new arrow facing the opposite direction and give it speed.
Extensions	

Use Case 55	**Create the credits scene**
Related Requirements	
Goal in Context	Inform the player about the state of the game, the creators and give a sense of completion
Preconditions	Allow transition to credits scene after beating the final boss.
Successful End Condition	The user can finish the video game and reach the credits scene.
Failed End Condition	The credits do not work as intend or are unreachable by players.
Primary Actors	Game Developers.
Secondary Actors	Client.
Trigger	The user finished the game.
Main Flow	● Create relevant artwork ● Implement the script for credits to scroll and end ● Write the content to display on the credits scene.
Extensions	