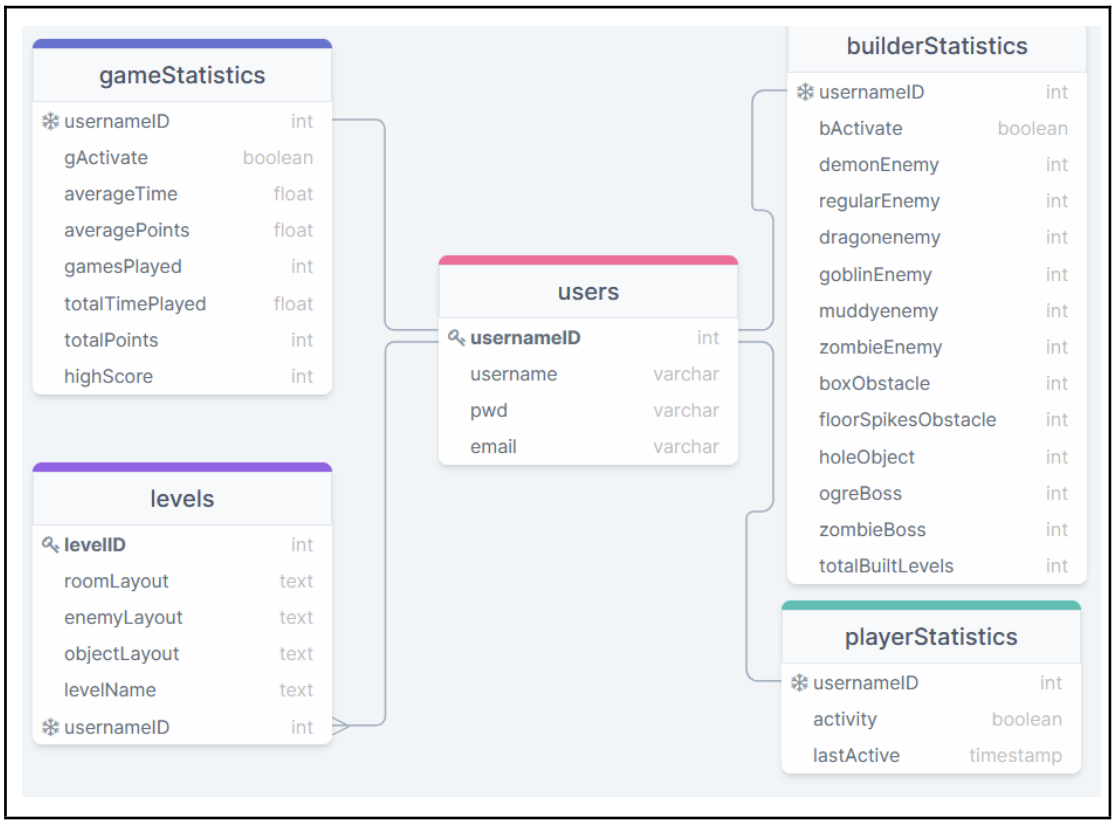
Tecnológico de Monterrey

Database and UML final version

Gerardo Gutiérrez, A01029422
Mateo Herrera, A01751912
Ana Paula Katsuda, A01025303
June 16th, 2022

ER Final Version



Cardinality

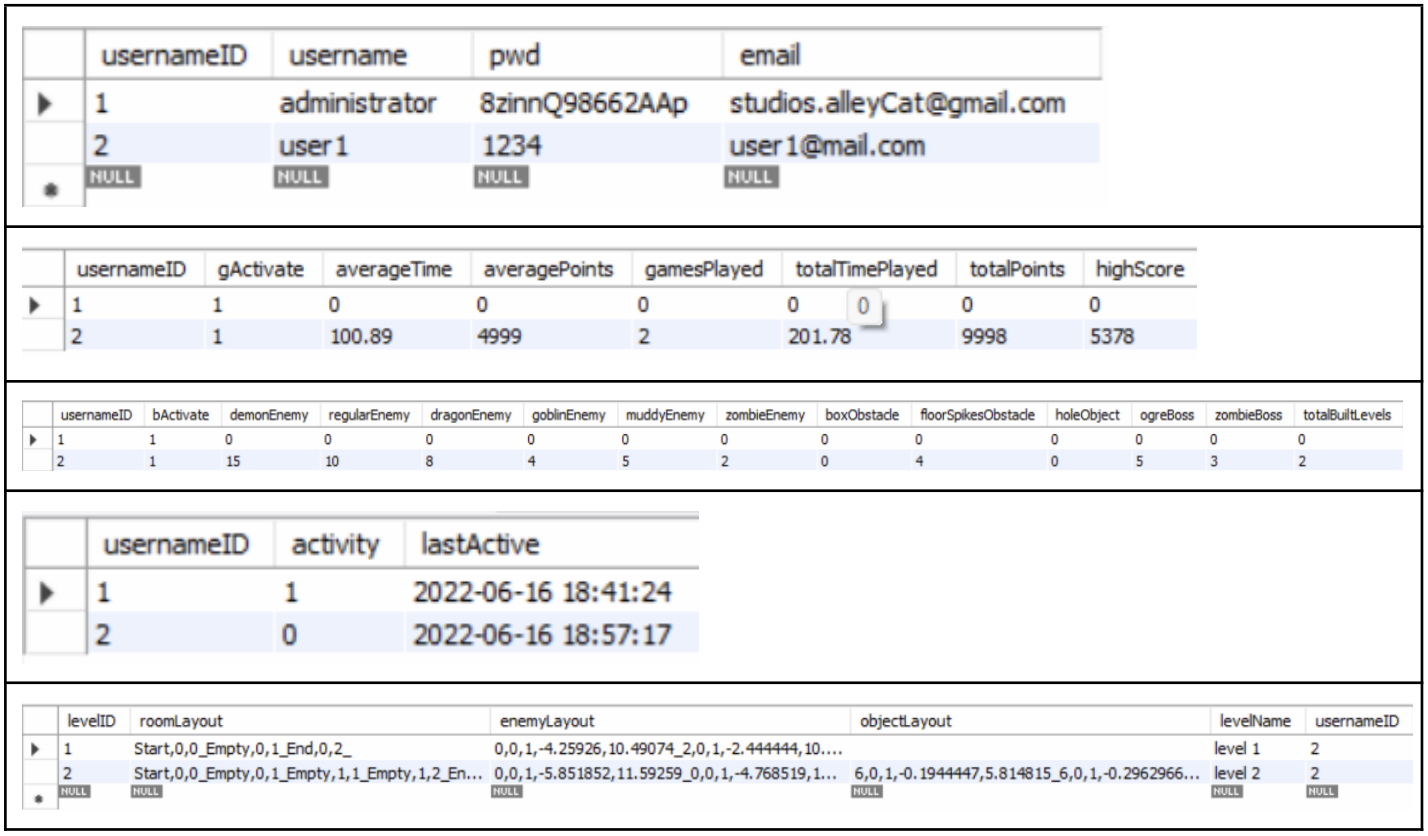
From the diagram shown above, it's possible to determine the following cardinalities between tables:

- The table USERS has a one to one relationship with the PLAYERSTATISTICS table, this is because for each player there'll be a single row of statistics that describes his activity.
- The table USERS has a one to many relationship with the LEVELS table, so one player can create/ have many levels.
- The USERS table has a one to one relationship with the BUILDERSTATISTICS table, this means that one player will have one row of statistics in elements used and levels built.
- The USERS table has a one to one relationship with the GAMESTATISTICS table, because each user will have a single row of statistics describing the results he gets in the play mode.

Restricciones de integridad:

- Required data: Some of the columns in the presented model have restrictions given the type of data that it's needed, meaning that some attributes like the username cannot be null.
- Validity checking: All columns have a domain of specific data.
- Entity integrity: The primary keys of the proposed tables have unique value and cannot be repeated in other columns.
- Referential integrity: The foreign keys in the tables establish the relationship in a correct manner.

MySQL final date schema



usernameID	username	pwd	email
1	administrator	8zinnQ98662AAp	studios.alleyCat@gmail.com
2	user1	1234	user1@mail.com
NULL	NULL	NULL	NULL

usernameID	gActivate	averageTime	averagePoints	gamesPlayed	totalTimePlayed	totalPoints	highScore
1	1	0	0	0	0	0	0
2	1	100.89	4999	2	201.78	9998	5378

usernameID	bActivate	demonEnemy	regularEnemy	dragonEnemy	goblinEnemy	muddyEnemy	zombieEnemy	boxObstacle	floorSpikesObstacle	holeObject	ogreBoss	zombieBoss	totalBuiltLevels
1	1	0	0	0	0	0	0	0	0	0	0	0	0
2	1	15	10	8	4	5	2	0	4	0	5	3	2

usernameID	activity	lastActive
1	1	2022-06-16 18:41:24
2	0	2022-06-16 18:57:17

levelID	roomLayout	enemyLayout	objectLayout	levelName	usernameID
1	Start,0,0_Empty,0,1_End,0,2_	0,0,1,-4.25926,10.49074_2,0,1,-2.444444,10....		level 1	2
2	Start,0,0_Empty,0,1_Empty,1,1_Empty,1,2_En...	0,0,1,-5.851852,11.59259_0,0,1,-4.768519,1...	6,0,1,-0.1944447,5.814815_6,0,1,-0.2962966...	level 2	2
NULL	NULL	NULL	NULL	NULL	NULL

ER normal form justification

Normalization of database

- First normal form: It is possible to notice that for all attributes in all tables, the values are at an atomic level, which means that none of said attributes can be separated or divided into smaller categories. Also, the USERS table has a unique primary key that can't be null, and also auto-increments with each new row created (this avoids any

repetition of any group of values.); this one and only primary key will serve as the sole key for the other tables except for the LEVELS table that has its own primary key but with the same characteristics as that of the USERS table. In addition, it can be observed that non-key values depend on the entirety of their respective key (of each table)(for example, the username value depends entirely on the usernameID).

- Segunda forma normal: Comenzando por que cumple con la primera forma normal argumentada anteriormente, todos los valores en todas las tablas dependen de su llave primaria únicamente. Adicionalmente, esta llave primaria se encuentra en una sola columna de cada tabla, siendo el valor de la llave indivisible, así cumpliendo cada aspecto de la segunda forma normal.
- Second normal form: Starting with the fact that the first normal form is fulfilled and detailed before, every value of each table depends only on the usernameID key and for the LEVELS values on the levelID key. Additionally, each key consists of a single column in each table, therefore the key is indivisible, which in turn completes the second normal form.
- Third normal form: Completing the third normal form, the columns in each table have no transitive dependencies, this recognizes that each value is individual (atomic) and depends solely on the primary key. Going deeper into what was just mentioned, the statistics tables are dependent on the usernameID key, this because the values of the columns of said tables are numbers and values that can not be related to one another making each row from every player quite distinct and diverse. So for the Statistics tables it is virtually impossible for another column that it's not the key to act as one. As for the LEVELS table, the values on the columns can vary exactly like the tables previously explained, this is because each level design can be very distinct as it has a multitude of elements, and none of them can act as a key or guide to the rest of the table values. As for the foreign key used, because the relationship between the tables LEVELS and USERS is many to one, the usernameID column will repeat itself in various rows, leaving the levelID as the only possible unique key that holds the dependency of the rest of the columns in the table.

Case Use Tables

Use case description detail	Create level.

Related Requirements	Requirement 1
Goal In Context	A new or existing user requests the creation of a new level.
Preconditions	The system requires user information in order to access level creation mode, so the user needs to be registered in the database.
Successful End Condition	A new level is created for the user.
Failed End Condition	The creation of the new level is rejected.
Primary Actors	User and game
Secondary Actors	Database
Trigger	The user licks the create level button
Main Flow	**Step and action**
	1º The user enters the main menu
	2º The user clicks the create button
	3º The user drags and drops different elements (rooms, obstacles or enemies)
Extensions	1.1.1º The user isn't able to enter the main menu
	2.1.1º The user clicks the create button but it does not work
	3.1.1º The user tries dragging and dropping different elements but it doesn't work.

Use case description detail	Test game mechanics in level
Related Requirements	Requirement 3
Goal In Context	A user who has created part of or the whole level tests said creation
Preconditions	A level must be in the process of being created by a user
Successful End Condition	The level is able to be play tested by the user
Failed End Condition	The level can not be play tested or the mechanics don't work as intended

Primary Actors	User and game
Secondary Actors	Database
Trigger	The user clicks the test button
Main Flow	**Step and action**
	1° The user creates a new level
	2° The user places various elements and rooms inside the level
	3° The user clicks the test button
	4° The user play tests the level
Extensions	3.1.1° The user clicks the place test button, but it does not work
	3.1.2° The user is able to access test mode but the mechanics do not work as intended

Use case description detail	Save and upload the designed level into the database
Related Requirements	Requirements 4 and 5
Goal In Context	A user who has created a completable level uploads said level into a database
Preconditions	A level must be created and completed during play test
Successful End Condition	The level is able to be uploaded into the database
Failed End Condition	The level can not be uploaded to the database or the level is saved incorrectly
Primary Actors	User and game
Secondary Actors	Database
Trigger	The user clicks the upload level button after the created level is completed.
Main Flow	**Step and action**
	1° The user creates a new level
	2° The user is able to complete said level
	3° The user clicks the upload level button

	4° The level is uploaded to the database with the correct parameters
	5° The database saves the level correctly
Extensions	3.1.1° The user clicks the upload level button, but it does not work and the level isn't uploaded
	4.1.1° The level is uploaded incorrectly or the txt files are corrupted
	4.1.2° The level saves the level incorrectly

Use case description detail	Get Access to user created levels
Related Requirements	Requirement 8
Goal In Context	A user can visualize and access their created levels.
Preconditions	The system requires the user to be logged in, so that they can directly access their created levels.
Successful End Condition	Users can access their created levels.
Failed End Condition	Users can’t access any of their created levels.
Primary Actors	User and database
Secondary Actors	Game and webpage
Trigger	User enters the created games window.
Main Flow	**Step and action**
	1º User enters the main menu
	2º User enters the created levels window
	3º User clicks a level and is able to access it.
Extensions	1.1.1º The user isn’t able to enter main menu
	2.1.1º The user isn’t able to see the created levels window
	3.1.1º The user isn’t able to access their created levels.

Use case description detail	Play level mechanics and objectives from start to finish
Related Requirements	Requirement 2
Goal In Context	Have a game that has a beginning and an end, filled with mechanics and room objectives in between.
Preconditions	A level has to be created and published. the user has to be registered in order to save the collected statistics on the current gaming session.
Successful End Condition	The registered user can play a level from the initial room, all the way to the final room, interacting with the designed mechanics and completing objectives.
Failed End Condition	The user can not play a selected level, no mechanics or objectives are present in the gameplay, and the level can not be completed or end.
Primary Actors	User and game.
Secondary Actors	Webpage
Trigger	the correct reading, loading, and functioning of the level file in the game.
Main Flow	**Step and action**
	1° Select/ click a level to play.
	2° Level loads from database onto the playable web interface
	3° The user plays the game with mouse and keyboard controls.
	4° The game is played from start to finish
Extensions	2.1.1° The gamebuilder reads the correct level file and displays it on a playable interface.
	4.1.1° The player/user encounters in the gameplay the intended mechanics and objectives.

Use case description detail	Search for and get access to published

	levels.
Related Requirements	Requirement 9
Goal In Context	A user is able to search and access different published levels.
Preconditions	The system requires user information to allow them to play any published level.
Successful End Condition	Users are able to access and search for published levels.
Failed End Condition	Users can’t search or access published levels.
Primary Actors	Database and user
Secondary Actors	Game
Trigger	User clicks a published level or the user uses the search bar in the published levels window.
Main Flow	**Step and action**
	1º User enters the main menu.
	2º User enters the levels catalog.
	3º User searches for a level using the search tab.
	4º User accesses the level they want by clicking it.
Extensions	1.1.1º User is unable to enter the main menu.
	2.1.1º User is unable to access the level catalog.
	3.1.1º User is unable to use the search bar.
	4.1.1º User is unable to access a published level.

Use case description detail	Visualize general data and clear information about the videogame in the web page.
Related Requirements	Non-functional requirement 9 and functional requirement 10
Goal In Context	Bring context to the game, its purpose and

	objectives along with some details that might complement said context.
Preconditions	Built Website and tab or section specific for the data that is required to be displayed.
Successful End Condition	The user can access the tab or section within the web page and is able to read the context of the videogame.
Failed End Condition	The user is unable to visualize the general information about the game on the website.
Primary Actors	User and web page.
Secondary Actors	NA
Trigger	Click on the tab or section that contains the general data.
Main Flow	**Step and action**
	1° The user clicks or navigates to the general information tab/section of the website.
	2° The user can visualize the information properly.
Extensions	1.1° The page must load fully in order to display or present the data previously mentioned.

Use case description detail	Display chosen statistics in web page
Related Requirements	Requirement 6
Goal In Context	Different relevant statistics from users are displayed in the web page.
Preconditions	The web page, database and video game must be functional and well designed. Some level must be already created and played to display statistics.
Successful End Condition	The statistics are shown in the web page.
Failed End Condition	The statistics cant be accessed or the information that is displayed is either wrong or corrupted.
Primary Actors	Web page and database

Secondary Actors	User and game
Trigger	The user clicks on the statistics section of the web page
Main Flow	**Step and action**
	1° The user clicks the statistics web page section.
	2° Web page loads section
	3° Web page API requests information
	4° Information is displayed on webpage
Extensions	3.1.1° Web page API can't access information
	4.1.1° Web page does not display correct or complete information.

Use case description detail	Register new users
Related Requirements	Requirement 7
Goal In Context	Register new users with a unique user and password.
Preconditions	Functioning form on the web page, and a connection to a database established.
Successful End Condition	The user is registered successfully on the database and can log in on the web page.
Failed End Condition	The user is not registered at all and can not log in on the web page.
Primary Actors	User and Web page
Secondary Actors	Database
Trigger	filled form on web page
Main Flow	**Step and action**
	1° The user clicks on the register button.
	2° the user fills out new user form (username, password)
	3° The user clicks the send button
	4° The database inserts the information on

	the new user.
Extensions	2.1° The user is bound to use a unique Username and password.

Activity diagrams

https://lucid.app/lucidchart/f1357c2a-8598-4917-b52a-2ce90d86db2e/edit?viewport_loc=-1164%2C-1500%2C5382%2C2959%2C0_0&invitationId=inv_f55f7caf-0822-4e1a-9f6d-1a5883203f63#

One Page

Functional and Non-functional Requirements invested time

Functional Requirements	**Invested Time**	**Non-functional Requirements**	**Invested Time**
1. Create new level	10 hours	1. Use of pixel art sprites	30 minutes
2. Play level mechanics and objectives from start to finish.	25 hours	2. Search level by name	1 hour
3. Test game mechanics at the current level.	10 hours	3. Connection to server through node.js	30 minutes
4. Upload the level once it has been passed in test mode.	5 hours	4. Stylise webpage to match game aesthetics	30 minutes
5. Save and upload the designed level to the database.	10 hours	5. Visualize general data and clear information about the videogame in the webpage	20 minutes
6. Display chosen statistics on the web page.	4 hours	6. Use of MySQLWorkbench to implement relation-entity database.	20 minutes
7. Register new users using a username and a password.	2 hours	7. Protection against SQL injection	10 minutes
8. Get access to user created levels.	3 hours	8. Embed videogame inside webpage	5 hours
9. Search for and get access to published levels.	3 hours	9. Version control through github	35 minutes
10. Website navigation of context, credits, game, and user login/register.	10 hours	10. Include license in webpage	5 minutes

Sprints review

Sprint	General Activities	Functional/non-functional requirements	Roles
Sprint 1	- Frontend significant development - Define entity-relationship diagrams - Start of game development	- Create new level (part 1) - Register user with name and password. - Visualize general data and information about the game on the website.	*Product owner*: teachers *Scrum master*: Ana Paula Katsuda *Software developers*: Gerardo Gutiérrez and Mateo Herrera *Tester*: Gerardo Gutiérrez
Sprint 2	- Construction of both game and game builder - Database script creation	- Play level mechanics and objectives from start to finish. - Test game mechanics at the current level. - Use Pixel Art Sprites - Website navigation of context, credits, and game. - Use of MSQLWorkbench to implement the relation-entity database.	*Product owner*: teachers *Scrum master*: Gerardo Gutiérrez *Software developers*: Mateo Herrera and Ana Paula Katsuda *Tester*: Mateo Herrera
Sprint 3	- Backend connections - Finish frontend - Game and game builder detail handling - Navigate, find and play published levels	- Upload the level once it has been passed in test mode. - Save and upload the designed level to the database. - Play level mechanics and objectives from start to finish - Get access to user created levels. - Connection to server through node.js - Protection against SQL injection	*Product owner*: teachers *Scrum master*: Mateo Herrera *Software developers*: Ana Paula Katsuda and Gerardo Gutiérrez *Tester*: Ana Paula Katsuda
Sprint 4	- Review backend - Functionality tests - Final aesthetic details - Activity Diagrams - Determine general	- Embed videogame into web page - Display chosen statistics in the web page - Search for and get access	*Product owner*: teachers *Scrum master*: Ana Paula Katsuda *Software developers*:

	statistics that will be displayed - Finish webpage general design and information - Navigate and play other users' published levels	to published levels.	Gerardo Gutiérrez and Mateo Herrera *Tester*: Gerardo Gutiérrez
Sprint 5	- Display graphs for statistics - Final detail handling - Final aesthetic modifications - Final product delivery	- Stylise webpage to match game aesthetics.	*Product owner*: teachers *Scrum master*: Gerardo Gutiérrez *Software developers*: Mateo Herrera and Ana Paula Katsuda *Tester*: Mateo Herrera
Total sprints: 5		**General roles:** - Mateo Herrera: Unity and API connection - Ana Paula Katsuda: Unity and Website - Gerardo Gutiérrez: Database and Statistics	

General Overview

Number of requirements per sprint	Functional: - *Sprint 1*: 2 - *Sprint 2*: 3 - *Sprint 3*: 4 - *Sprint 4*: 2 - *Sprint 5*: 0	Non-functional: - *Sprint 1*: 1 - *Sprint 2*: 2 - *Sprint 3*: 2 - *Sprint 4*: 1 - *Sprint 5*: 1
Time spent per project area	***Database***	10 hours
	Web Development	15 hours
	Videogames	60 hours
Total time spent in development	- Sprint 1: 10 hours - Sprint 2: 25 hours - Sprint 3: 20 hours - Sprint 4: 15 hours - Sprint 5: 10 hours	
Total time spent in	- Sprint 1: 5 hours	

documentation per sprint	- Sprint 2: 3 hours - Sprint 3: 1 hour - Sprint 4: 3 hours - Sprint 5: 3 hours
Total time spent in project documentation	15 hours