Super Valis X Project

What was done

I chose to develop a Phaser game “Super Valis X”, which is a retro themed dark fantasy roguelike game inspired by SNES Super Valis IV. In the game you kill enemies in maps and eventually fight the end boss of each map. There are three maps, which each contain four stages. Each map has a unique set of enemies and level designs and each end boss is unique with their abilities and phases. Map difficulty level increases linearly. The game uses some SNES Super Valis IV assets and various SNES soundtracks to make it look and sound more retro and original and also to give me more time to develop the actual game. The game is responsive, but the game itself is only playable on desktop.

Due to Phaser's physics engine cannot handle changing hitboxes very well (or my skill issue), so the player’s attack animations do not work as intended, because when hit box changes, the game does not recognize that there is a collider between the player and the platform, which makes the player fall through the floor and I had to implement workaround for this. I could have fixed this by making all the player assets the same size, but then the player’s hitbox would have been so massive and also removed the point of crouch ability. The best bet is to use jumping attack, if you want to see the player’s attack animations work.

What tools were used

The game is built with the Phaser game engine, which works in JavaScript. I made some assets by myself with GIMP and used some old Super Valis IV assets from https://www.spriters-resource.com/snes/supervalis4/ and cleaned them with GIMP. Music and sounds are from khinsider.com https://downloads.khinsider.com/game-soundtracks/album/super-valis-iv-snes .

Points generated from the course project

Feature	Reasoning	Points
Well written PDF report	-	3

Application is responsive and can be used on both desktop and mobile environment	The game is responsive, but is only playable on desktop so minus two points	2
Application works on Firefox, Safari, Edge and Chrome	Tested on Edge, Chrome and Firefox, so it probably works on Safari too	3
The application has clear directory structure and everything is organized well	Everything except the html file has their own folder, so everything is organized well	2
There is a clear plot in the game. It has a start and end.	Player starts clearing maps and finishes when the final boss is killed	3
There are different (more than 1) objects to collect	Hearts (health pickups) and potions (attack boost pickups)	2
There are more than one map	Three maps and each has four stages	3
Gamer needs to use both keyboard and mouse to meaningfully control the player character	Player only uses the mouse to attack so minus 1 point for that. The game is a retro arcade/roguelike game, so I wanted to keep the game theme and controls as original as it can be	2
Game uses physics engine, so that there are falling parts / enemies / players	There are a lot of projectiles and falling items	2
There are enemies that can hurt the player	At least five enemy types and three bosses.	3
There is music and sound effects when player shoots/jumps or anything like that	Each map has its own soundtrack and there is a map clear sound and game over sound.	3
Player, Enemy and Boss animations	This required cleaning assets, fixing the size of the images to the same size, so that player and enemies don't fall through the floor due to the physics engine and also making the animation functions with these animations properly.	4
Damage indicators	When a player or enemy takes damage, they turn red for a second, which shows when the player or enemy is taking damage. The player also gets pushed back a bit	2
Health bar	Zelda themed health bar, that shows hit points as hearts and when the player takes damage, then health bar updates according to damage taken	3

Unique boss fights	Each boss has unique abilities and phases with animations	4
Main menu, options, level selection, info	Player can navigate in the main menu and select if he wants to start a new game, select a certain level, change music options, check controls or read info about the game.	4
Game over screen	When Player dies, he has an option to restart the map again, go back to the main menu or change options. There is also a classic game over sound. The game knows which level you want to restart, so you don't have to restart the whole game when you die.	3
Ending screen	When the player kills the last boss, the ending screen appears and the player is congratulated for his/her achievement and also the ending music starts. The player can choose to restart the game from map 1 or go to the main menu.	2
Custom font	The game uses a custom font, which makes the game feel more like a retro arcade game.	1
Uses OOP	The game design mainly uses object oriented programming paradigm to reduce duplication and also makes the repository structure more clearer. The game was originally made with procedural programming paradigm, but refactored to object oriented programming paradigm to make it more easier to handle and read.	2