1. Project 3: HTML5 game

Use Phaser and build an HTML5 game. You can do the game you wish, but it needs to have at least several maps, movable player, collectable items and probably also enemies – or flowers or whatever you wish to do in the game! Build your own Crystal Caves, tower defense or tank game!

2. Documentation

About the game: I made a HTML 5 game in this project. The game is titled "Asteroid Demonetizer". The game is about destroying asteroids to win, and the goal is to get as high of a score as possible while completing this objective. There are 4 asteroids that the player must destroy in order to win. If the player wins they can insert their score. Only the top 10 players get a spot in the scoreboard. Players get score by completing objectives, destroying enemies, and gathering collectibles. The player can collect stars (from enemies) for points, small shields for health (from enemies), and shield powerups from the asteroids. All the drops fall with gravity (with there being a floor to make it seem like you're on the surface of a moon/planet), the reason for this wasn't due to being unable to take away gravity (enemies have none for example), it was because I wanted the player to struggle more than just shooting down a horde of enemies while staying still and then collecting everything when it's safe. This pushes the player to be more risky for a higher score. I also made the bullets not decay in the end off-screen, because it lets the player hit asteroids, even if off-screen. Why this isn't a problem is because the quicker you destroy the less time you have to get even more score. If I wanted to limit the range I'd have to introduce new game design elements such as how to communicate the range to the player visually or with audio. I couldn't visually, since the game would be too claustrophobic to play by limiting the projectile distance inside the screen (in view).

About the code: The game was made on Phaser 3.80.1. While the video tutorials introduced a different Phaser version, I thought it would be appropriate to use the more up to date version of Phaser. The game making was made similarly to the teacher which what I understood was that the game is mainly in the javascript form, and we didn't need to use CSS. The game was implemented inside one javascript file named game.js. Assets are all in a separate folder (just like the javascript file is). I've kept the javascript file well organized with comments and sections which are dedicated to certain functions specifically. I used multiple helper functions, but decided to not section them off separately, since they were already well organized in their own sections of the code. I did, however, separate the multiple screens (called scenes in Phaser 3) that the game had to make it much easier to modify, and view instead of leaving it all in one main scene.

I took advantage of multiple methods and ideas that were inspired by either the teacher or by skimming through the Phaser tutorial sections. I mainly wanted to make my own code, though, so I attempted to code on my own. I took advantage of the physics engine, Phaser's intuitive control checking, and math functions.

The MainMenu scene is there to preload everything, and to then create the main menu itself. From there you can access PlayGame or the ScoreBoard scenes via their respective buttons (play game, and view scores).

The PlayGame scene has all the main functions for the game, everything related to the gameplay, essentially. The code starts off with CREATE AND SETUP LOGIC which just initializes and makes everything (including UI and colliders). The collision logic and UI logic is separate in their own sections within the same scene. Later on there come sections about game timer, then the update section (which just checks for player movement/rotation and UI updates), then the player movement and rotation section, then the actual UI section, then shooting logic, then spawning enemy logic, collision logic, and finally we have the shield powerup and all the spawnable collectibles and how to collect them.

When you finish a game you either trigger endGame or winGame functions. Which is used in the scoreboard section or to skip adding your score if you lost. This takes us to the ScoreBoard scene in which it's a simple approach of initializing the data, creating everything, and handling the highscore entry (to check if it is needed or not), key input (to enter if needed to enter), submitting the score, and finally showing the player highscores no matter what in the end. From there you can directly just go back to the MainMenu scene.

3. Points:

Feature	**Max points**	**How this feature was fulfilled**
Well written PDF report	3	I believe I have a PDF report that conforms to the instructions. I'm assuming this is how I was supposed to do the grading part. The document might seem to have too much text, but that's because we weren't specified what to write about, so it's always safer to tell about everything (mostly).
Application is responsive	2	The application works without failure.
Application works on Firefox, Safari, Edge, and Chrome	3	The application can be run on anything that can run Phaser 3, as that was what we were given during the course to do project 3, so, yes, it can.
The application has clear directory structure and everything is organized well	2	The application has 1 html called index.html, 1 javascript that is in the javascript folder with the appropriate name of game.js. All assets are placed in the assets folder. Since there a are very little assets I didn't separate the audio, png, and spritesheets.

There is a clear plot in the game. It has a start and end.	3	You have a timer to complete the task. The task is to destroy 4 asteroids. Failure to do so will result in a bad game over.
User can get their name in the scoreboard	3	You get score by destroying asteroids, destroying enemies, and collecting stars (that drop from enemies). The scoreboard can be accessed via main menu, and after completion of the game. The game fetches the score from local storage to simulate an arcade highscore system.
There are different (more than 1) objects to collect	2	You can collect stars dropped by roughly every 6th enemy, a shield powerup from destroyed asteroids, and health (blue shield) from roughly every 11th enemy.
There are moving parts in the game area (other than the player and enemies, so e.g. some floors fall apart)	3	The collectibles all drop, this is to make it more dynamic and to make it so you can't just snipe enemies and makes you require to pay attention if you want to collect more points and be more mindful of positioning.
Gamer needs to use both keyboard and mouse to meaningfully control the player character	3	You use W to jump, A and D to go left and right respectively. Spacebar to turn on powerup. Use the mouse to aim, and hold down left click on mouse to shoot.

Game uses physics engine, so that there are falling parts / enemies / players	2	You need to “jump” to stay up. All the collectibles drop with physics to make the game more dynamic.
There are enemies that can hurt the player	3	All enemies hurt as they ram into the player and get destroyed. The player has limited health. There’s an invulnerability between every hit to make the player be able to recover and not get a game over from
There is music and sound effects when player shoots/jumps or anything like that	3	There are both music and sound effects when you destroy enemies, when you shoot, when you destroy asteroids, and when you get destroyed yourself.
	32	

Suggested Feature	**Max points**	**How this feature was fulfilled**
Comprehensive menu with instructions to everything	2	There is a main menu which has responsive controls, it also has instructions on how to play at the bottom of the screen.
Scoreboard which saves locally	3	The highscores/scoreboard gets saved in a json within local storage that Phaser 3 uses. (in the browser) This storage can fit 5-8 MB if I

		remember correctly, so it will not run out.
Player can aim with their cursor and the player model responds to aiming by looking at the mouse pointer direction	3	The player aim is calculated based on mouse pointer, I also implemented a projectile that points in the direction it is shot towards (irrespective of the current player's position, so they go straight).
Player can use powerups	3	The player can collect a powerup dropped by an asteroid, which when pressed on the spacebar turns the player invincible for 10 seconds, this is marked by turning the player model green with a green tint. The player can only have 1 powerup at a time.