External Sources

The javascript of our game has solely come from the w3schools game at: https://www.w3schools.com/graphics/game_intro.asp

The first function *choseCharacter* allows the user to choose their own character out of 5 different preset models for the game. This is done by taking the value of the textbox on the home page, then associating that number with a certain character. If the input is invalid, the user cannot start the game. This allows the user to customize the game. We have done so through our understanding of the new component function, as we are able to preload multiple images as one variable into the component function.

The second function *startJetpackGame* creates the game area, the sounds and the components (ex: obstacles) used in the game. Even though the new sound and new component were learned from ww3schools at https://www.w3schools.com/graphics/game_components.asp, the rest of this function is our own work. Additionally, the w3schools iteration only includes one sound and one component, and we have been able to add to this.

This sound component works by creating different sounds by calling the sound constructor function, and by creating different components by calling the component constructor function. We edited the component function from the original to allow an image to show up on the component, and allow the user to change that image through the usage of the variable chosenCharacter and chosenCharacterFlying. We also incorporated many different noises, such as constant game audio, audio that is played when moving, etc, and not just death noises.

The canvas function was also learned at https://www.w3schools.com/graphics/game_canvas.asp

This.function learned from https://www.w3schools.com/js/js_this.asp

In the end, with everything we learned above, we added the following things:

- Different noises + multiple components
- Ability to use a variable in component to set different images
- Changed the obstacles generated to have a different array for both the top obstacles and bottom obstacles, allowing us to generate obstacles at different times and at different speeds
- Added levels to the game
    - With our knowledge learned, we changed the speeds by changing the .x values and .y values of the obstacles to make them appear like they were moving faster to the side and up or down
    - Added skip level
- Added respawn

- Changed movement to key press with .keyCode() and preventDefault()
  - .keyCode() was learned in advance and checked on google, while preventDefault was learned solely from google
- Slightly changed game gravity to better simulate jetpack
- Also added de-accelerate with the press of any other key other than the space button
- Added a high score function through our knowledge of the frameNo
  - Also how we made our levels
    - At different frameNo different events occur
- Changed obstacles to images which are also alternating through the same knowledge used as for the characters changing skins
- Altered all functions from w3schools iteration to make the code more efficient with the above changes
  - Code is entirely written by ourselves but certain functions are very similar to the w3schools functions as they were necessary for the game and were not previously learned
- Below styles

The hovering design for the buttons was inspired by https://www.designskilz.com/cool-css-buttons/
All of the buttons contain this hovering design except for the start button on the character select page. The reason for that is due to the fact that when we tried to implement the hover effect on that button, it would continuously "jump" up and down so the button could not be clicked. We tried for multiple hours to try and fix it with many things like changing the number of degrees the button is translated and tried using different transformation styles for it but it never seemed to work. So, instead, we found a pseudo class code which was inspired by https://developer.mozilla.org/en-US/docs/Web/CSS/:not to exclude the hover from that one button. To do the hover, however, there were multiple CSS transformations used. They include WebKit-perspective which give a 3D element some perspective and WebKit transform which help to transform a 2D or 3D object. There are a few adjustments we did add to the button however to make it my own. For starters, we changed the colour of the font so that it has a colour changing font instead of having a basic white or black font. Next, we changed the amount that the button rotates when the cursor hovers over it. We didn't want the rotation to be so pronounced so we reduced the amount of rotation by 5 degrees. Next, we changed the button colour to a black background to suit the theme of our game and wechanged the shadow colour and border colour so that when the button was rotated, it looked like it was moving towards you. Finally,   the ease time from 2 seconds to 3 seconds as the users to see and admire the button moving just in case 2 seconds too short.