**Università di Pavia**

**Computer Engineering**

***Web and Multimedia Technologies* – Project Report**

|  |  |
| --- | --- |
| **Surname** | Mono |
| **Name** | Bill |
| **Student ID** | 479379 |
| **E-mail** | bill.mono01@universitadipavia.it |
| **Date of written test** | 21/02/2020 |

|  |
| --- |
| **Project title** (website subject) |
| Asteroid (game) |

|  |  |  |
| --- | --- | --- |
| **HTML version** | | |
| 🗹 | HTML5 | 🞎 | HTML4 | 🞎 | XHTML |

|  |
| --- |
| **Client-side technologies used** |
| Javascript | |

|  |
| --- |
| **Server-side technologies used** |
| PHP | |

|  |
| --- |
| **Website content description** |
| Describe the website in terms of:   * *Content of the single pages* and/or of the *main sections* (e.g., “The products page lists all the…”, “The ‘News’ section presents…”) * *Graphic design* (e.g., “The navigation menu is located in the left part of the page, immediately below the header, and…”)   Specify whether, for the definition of the content and/or of the graphic structure of your site, you have “drawn inspiration” from one or more existing websites (indicating their URLs).  This section can be extended to the next page, if necessary―but possibly avoid exceeding 50 lines.  Asteroid is a game set in space. We are in year 2100, terrestrial technologies have evolved, resources in the planets are scarce, therefore we must move towards space in search of new riches. We explore the space with our spaceship in search of these riches. During our journey we will have to be careful to avoid all the asteroids, which will arise in our path.  index.php: It only contains the title of the site, once the title appears at center of the screen, you will be directed to the home page.  On every page of the website except for the index.php page, we have a header and a footer, and among them we have a container present on each page, whose content varies according to the page where we are.  homepage.php: On the home page you will be asked to log in if you have not already done so, otherwise there will be two buttons, that are below a short presentation of the game, one of them, it will send you to the game and the other that will disconnect the user.  login.php: On the login page you will be asked to insert your login credentials. If you have not yet registered on the site, you can go to the signin page by clicking on the bar containing the word ''signin''. Once you have inserted your data, click on the "enter" button. If you do not insert the username or password, or if the password and username do not correspond to any user included in the database, an error message will appear below the fields.  signin.php: On the signin page you will be asked to fill in all the fields to complete the registration, once you have filled in all the fields click the "signin" button, you will be redirected to the login page, where you will have to insert the password and username chosen at the time of registration. If you do not fill in all the fields or the password and the password confirmation do not match, an error message will appear below the fields.  game.php: On the game page there is the username, a score field, and three buttons one below the other,the first button send you to the ranking page, the second button restart the game, and the third button stop the music. In addition to this, in the left of the page there is the game screen created with the tag canvas of HTML5 and Javascript.  Before playing, you will be asked if you want to use voice recognition in the game, if you answer ''yes'' you will not be able to use the keyboard but only voice recognition, the game speed will obviously be lower compared to a game played on the keyboard due to the fact that it is more complicated to move the spaceship only with the voice. If instead you answer ''no'',you will use the keyboard and not the voice recognition. The answer can be given either via the keyboard by clicking on the yes or no buttons, or with the voice saying YES or NO depending on whether you want to use voice recognition or not.  gameover.php:In the ranking page there are two rankings, the first is the general one and shows the top 4 positions with the best score, taking them from the users registered on the site. The second ranking shows the 3 best personal scores. Here too voice recognition is active, so to play another game just say "GAME" or click on the play button again that is below the ranking, while to return to the home page just click the go to home button that is also below the ranking or just simply say "HOME".  classfica.php, loginproc.php, signinproc.php,logout.php,database.php: They interact with the server side.  I don't have drawn inspiration by any site, because I was already prepared on this. |
| **Technical description of the website** |
| Describe the website implementation in terms of:   * *Folders (directories) and files*: list the folders that are part of the project, describing their content (e.g., “The folder img contains all the images of the website, while the folder…”). Clearly specify the name of the Home Page file * *HTML structure of pages*: indicate whether there are “structures” shared by all or many pages of the website, such as headers (e.g., <header> tag), generic containers (<div>, <section>), navigation sections (<nav>), main content (<article>), footer (<footer>), etc. * *CSS*: indicate the main selectors for which you have defined styles (shortly describing them), including possible classes or pseudo-classes (e.g., “The style for the #main selector defines an absolute positioning and…”; “The generic class .ital specifies that…”) * *JavaScript (and other possible client-side technologies used)*: illustrate the purpose of the employed JavaScript code (as well as of other possible client-side codes), shortly describing it (without necessarily providing all technical details, however); e.g., “The JavaScript function slideSh() allows to display a sequence of images which are placed in the folder …”). Indicate the URL(s) of the possible page(s) from which the JavaScript (or other technology) code has been “copied” and adapted to your own needs * *Server-side technologies*: indicate the purpose of the employed server-side code (e.g., PHP, ASP, Node.js, etc.), shortly describing it (without necessarily providing all technical details, however); e.g., “The PHP code at the beginning of the <body> of the page is used to…” * *Development tools employed*: HTML/CSS/JavaScript/etc. editors, possible image editing tools (e.g., Gimp, Photoshop, …), etc.   Specify whether, for the creation of the website, you have “drawn inspiration” from existing websites or templates (indicating their URLs), and whether you have used specific frameworks (such as Bootstrap). Also indicate anything you deem useful to make your implementation choices clear.  This section can be extended to the next page, if necessary―but possibly avoid exceeding 90 lines.  FOLDERS : The css folder contains a css file for each page of the site, the img\_game folder contains the images of the elements of the game (space ships, asteroids, gold coin, red coin, diamond, hourglass), then it contains the background used in the various web pages, and the various backgrounds used during the game. The js folder contains the javascript files, this files contain the various functions used throughout the site and especially in the game. The music folder contains the music of the game, the sound of the gameover, and the sound when you take a coin. The php folder contains almost all files with the php extension, the index, the home page and the game page are left out.  HTML: As I said before, the header and footer tags are shared on all web pages, in the header there is the title of the game which is then repeated on each page. In the footer, on the other hand, there are information about the reserved rights. Another tag shared by almost all pages is the div tag with class attribute equal to content, it is the container of each page.  CSS: In the homepage.css file, the "tasti" class specifies how buttons will be displayed within the site, some properties of this class are letter-spacing 2px, color #fff, background-color black. When the cursor goes over it the background color turns white and the font black. Also in this file, the style of the header and footer is specified. The header has the following characteristics margin 0, top 0px, height 140px, width 100%, background black. In the header there is the site title, which is defined in the title class, some properties of this top class: 0px, width: 100%. The h1 tags that are inside the title class have these characteristics, letter-spacing 9px, margin-left 80px, margin-top 40px, text-decoration-line none, font family Roboto, font-size: 30px, color: white, float left, text-transform: uppercase. Another important class defined in this file is the content class, it defines the background of each page of the site, here I list some properties, margin-left: 0px, height: 100%, background-image: url ('../ img\_game /space4.jpg'),background-size cover. Another important thing is that all web site page are responsive, so for example for screens with a width not exceeding 375px, the h1 tag inside the title class has a font-size equal to 20px and no longer equal to 30px. The other css files are very similar to each other, the most important part is that they change the contents of the container, they are enclosed in the content class described above. For example in the signin.css file the div tag contained in the div tag with the content class, has as id home5, the main properties of this selector are margin-top 50px, margin-left 40px, text-align center, background rgba (255 , 255, 255, .2), background-size: 100% 100%, border: # 2e2e33 solid 0.5px, the position is absolute with respect to the div I mentioned before.  PHP: database.php, in this file we create the connection to the "asteroid" database, if the connection fails you receive an error message through the die() function. With the statement "include ('database.php')", we will have an open session and a connection to our database in each of our web pages. In the home.php file, the statement "if (isset ($ \_ SESSION ['LOGGATO']]))", means if the superglobal variable "LOGGATO" exists then the two game and exit buttons will be displayed, otherwise the button login will be displayed, because you still have to log in. In the login and signin pages, the form data is sent respectively to the loginproc.php and signinproc.php pages with the $ POST method, in these pages the data is processed, for example in the case of signin the data is saved in the database, if the registration procedure is fine, a new row is inserted in the user table of the asteroid database. Another thing to say about the loginproc.php file is that the "LOGGATO" global supervariable is set here to 1, and a new session is created with the data of the user who has just logged in. In the game, I had to find a way to pass the data from the score variable in javascript to a variable in php to insert them in the ranking. I created in the game.php file an input field inside a form to take account of the score. Once the game is over, by clicking on both ranking and play again, the data is cleaned off to the server which enters the score with the username in the ranking table. Once this is done you will be redirected to the requested page.  JAVASCRIPT:The javascript part is the most important part of my site. I can divide the work in javascript language in two parts, the part for the animation of the site, and the part related to the game. In the animation part of the site, I can place the index.js file. The animation () function present in this file is called when the page is loaded (body tag), this calls the update () function every 50 milliseconds. The update () function modifies the blur of the list containing the word asteroid, when the blur value becomes less than or equal to -1, the clearInterval method is called which blocks the loop and we are redirected to the home page. I have tried to implement this game also to people with movement disabilities, so I have implemented speech recognition. To do this, I took inspiration from these two websites <https://davidwalsh.name/speech-recognition> and <https://codingislove.com/speech-with-javascript/>, trying to modify them as much as possible to adapt them to my website. So in the home.js file this mode is implemented, if we pronounce the word "game", a new document is loaded with the assign method , the same thing appens if we pronounce the word "login" or "home". entry\_game (), entry\_login (), and quit () are three functions that also use the assign method to upload a new document, and are called when the user clicks the "login" button for entry\_login () function, the "exit" button for the quit () function, and the "game" button for the entry\_game () function. In the login.js and signin.js files, which functions are called in the login.php and signin.php files, there is no voice recognition, this is because I did not know how to process the form data by not submitting the data, click the "enter" button. In these files, a check is made regarding the insertion in all fields of the form, and in the registration process a check is made regarding the correspondence between the password and the confirmation of the password. If these checks fail, an error message is displayed.  The gameover.js file is very similar to the home.js file, voice recognition is also used here, and the assign method to redirect you to other pages. The play\_again (), go\_to\_home () functions are called when the user clicks the "play again" or "go to home" button. In the game.js file there is all the javascript code related to the game, we start from the speak\_recognition (check) function, in the game.php page as mentioned above you will be asked to use or not use voice recognition. After answering the function, the speak\_recognition (check) function takes the value of the response as a parameter, among the options "yes" or "no". Then it hides the contents of the window containing the request for the use of voice recognition and makes the canvas window visible, finally calls some functions. If the answer is "yes", the full\_object () function is called, which fills an array containing the elements of the game in different quantities. the random\_choice () function is called, which fills an array of 8 elements with the elements of the vector of the full\_object () function taken randomly, this is what I need to show the game elements on the display randomly and with different probability. The enemy\_position () and coin\_position () functions are also called, which initialize the positions of the asteroids and the coins, filling an array containing the positions (x and y) with random numbers, the function random\_coin (), instead it serves to fill a vector with random numbers between 120 and 170, will be useful later when we have to update the positions of the coins. The difficulty () function instead, depending on the time spent in the game, increases the game speed. Finally the draw () function is called which is the most important function that allows us to represent the elements of the game in the canvas window. So these are the functions called by the speak\_recognition (check) function if the parameter is equal to "yes", if the parameter is equal to "no", the same functions are called, except for the sound\_game (control) function. This function, enables or disables the game music based on the value of the parameter passed to the function. The draw () function is the most important function, because it allows us to modify the canvas window. Through the drawImage () method we can draw an image in our canvas. The draw () function calls other functions inside. The enemy\_update\_cordinate () and coin\_update\_cordinate () functions called by the draw () update the coordinates of the game elements. The enemy\_collision (), coin\_collision () functions manage the collision between the elements of the game and our spacecraft. In particular in the coin\_collision (), the collision behavior will depend on the element, if for example the element is a gold coin, the score will increase by 1, while if it is a red coin, the score will increase by 5 and so on . The last function I want to talk about is the move (e) function, this is called each time a button is pressed (onkeydown event). If the button pressed is the left key of the keyboard, our spaceship will move to the left, otherwise if it is the right key it will move to the right. |