ECM1417
Web Development

Project Specification
Arcade game website

Rui Jin, Zhe Wang, Jia Hu and Matt Collison
January 2022

This document outlines the specification and submission requirements for the ECM1417 Web Development module.

1 Requirements

In this assignment you will deploy a LAMP stack web server to your Azure VM and design a website that enables the game tetris to be played in a browser. The requirements detailed below specify exactly how this website should be built.

This specification is subject to change for 7 days after release to clarify any queries. After 26th February the document will not be changed.

1.1 Web server architecture

The web server will be hosted on the Microsoft Azure platform that was introduced in week 1. Each student has been allocated an individual VM on which you will be expected configure and run a LAMP stack. This means on your Linux VM you will need to install and run Apache httpd, MySQL and PHP.

The Apache httpd web server should be configured to run php and handle HTTP requests.

MySQL will need to be installed and the database will be created using the sql script provided on ELE in the assessment information tile. The given sql script will create the database and tables and this database structure should not be changed. You are also given the sql queries to populate and read from the database tables to enable registration, user login, storing of scores from completed games and reading data to populate the leaderboard. You will then need to write php code using mysqli to integarte the data from these queries within your web application.

The php webpages will need to be developed to enable gameplay and webpage design features as specified below.

1.2 Website design

1.2.1 Webpages

Four webpages should be created for the website. The landing (index) page, registration page, leaderboard page and the game page. These webpages must use the following names:

1. index.php
2. register.php
3. tetris.php
4. leaderboard.php

1.2.2 navbar and main content

All pages will have a shared navbar menu at the top of the page. The navbar menu must include the following content items with their corresponding name attribute:

1. Home - name="home"
2. Play Tetris - name="tetris"
3. Leaderboard - name="leaderboard"

The navbar items must use the Arial font in bold with fontsize 12px. The Home item should be aligned on the left and the Play Tetris and Leaderboard items on the right. The navbar background colour must be 'blue'.

Each webpage must then contain the rest of the content for the webpage in a div with class attribute value of "main". The main div will have a background image, this must be the Tetris image (tetris.png) provided on ELE, set at 95% width of the main container and in a fixed position in the centre, vertically and horizonally.

1.2.3 Landing page

The landing page (index.php) must display the welcome message 'Welcome to Tetris' if the user is logged in followed by a button with the content 'Click here to play' that contains a hyperlink to the tetris.php page. Alternatively, if the user isn't logged in the landging page should provide a login form with input boxes for username and password followed by a paragraph with the content "Don't have a user account? Register now" where the Register now text has a hyperlink to register.php. The username field should have a placeholder of 'username'. The login form or welcome message must be in a div with a grey (hexcode c7c7c7) background and 5px box-shadow and the box must be centred.

1.2.4 Registration webpage

The registration webpage will contain a form in a div with a grey (hexcode c7c7c7) background and 5px box-shadow. The form will contain text input fields for 'First Name', 'Last name' and 'Username'; Two password input fields, with placeholders of 'Password' and 'Confirm password'; and finally a radio field with name="display" and values of 'yes' or 'no' for whether to 'Display Scores on leaderboard'. The action on submitting the form will be a POST request to index.php.

1.2.5 Tetris webpage

The tetris page is where the main tetris game will be played. This webpage will have a div with a grey (hexcode c7c7c7) background and 5px box-shadow where the game tetris can be played. The Tetris game will be played on a board 10 blocks wide and 20 blocks tall on a div with id tetris-bg and the background image image tetris-grid-bg.png as provided on ELE. The game should implement the following features with the implementations details given below:

1. The game should start when the user clicks a button with content 'Start the game'. The button should disappear when the game starts.
2. A new tetris game piece should appear at the top of the board to start the game or when the previous game piece contacts the bottom of the board or another block.
3. The game should end when the game pieces pile up and reach the top of the board.
4. Left and right arrow keys should be used to direct the game pieces when they are falling.
5. A score should be incremented by one every time a new game piece appears.

The tetris game pieces must be rendered as a combination of sqaure block div elements with the class attribute value set to "block". The tetris grid background image should align with block positions. Each type of block should be given an id that provides the correct background colour for that type of game piece.

JavaScript code should be used to create the logic that will run the tetris game. The two main data structures that will be used in the logic:

- A 10x20 2-D array of strings to represents the tetris grid. Each string will start empty and contain the 'id' of the div that will be displayed in that position on the grid.
- An associative array for the shape of game pieces. Keys for the associative array will be the 'id' of the div variables and the values will contain the coordinates of the sqaures that make up each shape for the relevant type of game piece. For example, the sqaure piece will have an 'id' of O and represent four blocks with the coordinates array [[1,1],[1,2],[2,1],[2,2]].

The game pieces associative array will be:

- "L" => [[1,1],[1,2],[1,3],[2,3]]
- "Z" => [[1,1],[2,1],[2,2],[2,3]]
- "S" => [[1,2],[2,1],[2,2],[3,1]]
- "T" => [[1,1],[2,1],[2,2],[3,1]]
- "O" => [[1,1],[1,2],[2,1],[2,2]]
- "I" => [[1,1],[1,2],[1,3],[1,4]]

The logic below should be used to implement the game logic. This algorithm will also guide the corresponding updates to the div elements that will display the game as it is played on the website:

1. A new game piece should be randomly selected and a variable 'currentBlock' be assigned to the game piece.
2. The program should check if the starting coordinates for 'currentBlock' at the top of the grid are empty in the tetris grid array. If any of the coordinates are not empty the game should end and the current points submitted to the server in a post request to leaderboard.php. If the relevant coordinates are empty, these coordinates on the tetris grid array should be updated with the letter of the game piece defined by the key of currentBlock.
3. The "block" div elements that make up the currentBlock should be created on the webpage to display the addition of currentBlock at the top of the tetris grid.
4. If the left or right key are pressed the currentBlock should be moved (within the bounds of the grid and without overlapping existing pieces) and the "block" div elements updated using the transform: translate(x,y) css attribute.
5. After one second or when the down key is pressed the program should check if the coordinates one below currentBlock are occupied (not empty).
   (a) If the coordinates are empty
      - the coordinates of currentBlock on the tetris grid array should be moved down by one on the vertical axis. The div elements position should also be updated.

   (b) If the coordinates are not empty
      - The div blocks represented by currentBlock on the tetris grid array should be set with the div position attribute.
      - The program should check if any rows of the tetris grid are complete. If they are, that row should be removed and the position of all rows above moved down in

the array. The "block" div elements will then need to be repositioned to show the updated positions on the tetris grid.

- Return to step 1 to create a new game piece and reassigned currentBlock.

Bonus marks can be achieved by implementing the spin function when the up button is pressed and a sound track during gameplay.

1.2.6 Leaderboard

The leaderboard page will have a div with a grey (hexcode c7c7c7) background and 5px box-shadow where a table is displayed containing two columns for 'Username' and 'Score' populated with all games stored in the database for users that selected to display their scores. The table should have border spacing at 2px and the background colour for table header cells set as 'blue'.

2 Mark scheme - total 30 marks

A series of tests written in Selenium will be executed on the live websites running on the Azure VMs to verify if the specific html and css attributes are complete. All the criteria in the specification will be strictly translated into tests except the Tetris gameplay logic which will be manually tested by playing the game on the webpage.

2.1 Index page (10 marks)

1. The VM give a response to a HTTP request to the web server - 2 marks
2. A 200 response code to a HTTP GET request to index.php - 2 marks
3. A navbar and content div that fulfill the required css and html attributes for index.php - 2 marks
4. A login form and session handling attributes that fulfill the specified attributes for index.php - 2 marks
5. A 200 response code and user creation to a HTTP POST request with details to create a new user - 2 marks

2.2 Leaderboard and registration page (10 marks)

1. A 200 response code to a HTTP request to leaderboard.php - 2 marks
2. A navbar and content div that fulfill the css and html attributes for leaderboard.php - 2 marks
3. A leaderboard table with relevant and correctly styled content and attributes in leaderboard.php - 2 marks
4. A 200 response code to a HTTP POST request to leaderboard.php that adds a new game score - 2 marks
5. A registration.php page with correct content and styling on the form - 2 marks

2.3 Tetris page (10 marks)

1. A 200 response code to a HTTP GET request to tetris.php - 2 marks
2. A navbar and content with the cirect styling on the tetris background image that fulfill the css and html attributes - 2 marks
3. An attempt at an effective tetris game - 2 marks
4. A complete tetris game - 2 marks
5. Bonus features in the Tetris game, such as turning pieces and a sound track - 2 marks

2.4 Submission

A submission needs to be made via bart but marking will be done primarily on the Azure VMs. The bart submission will be used as evidence of the work after the initial submission date and you should submit a zip file containing all your php, css and js files as well as a README file that provides the link to your Azure labs VM. If nothing is submitted to bart you will receive a zero mark. If the bart submission is late this will be treated as a late submission.

3 Summary

In summary, this project requires students to combine multiple components of a full stack web application to deploy a website that runs the game Tetris.

If you have any questions at any point throughout the project please do not hesitate to get in touch, m.collison@exeter.ac.uk.