CSC7062 Assessment Report

Name: Daniel McAuley
Student number: 40125497

Website URL: http://dmcauley21.lampt.eeecs.qub.ac.uk/powerliftingcentralv2/
Video Demo: https://web.microsoftstream.com/video/673eb28d-c076-4ab8-b505-472522d37111

Database

Sourcing Data

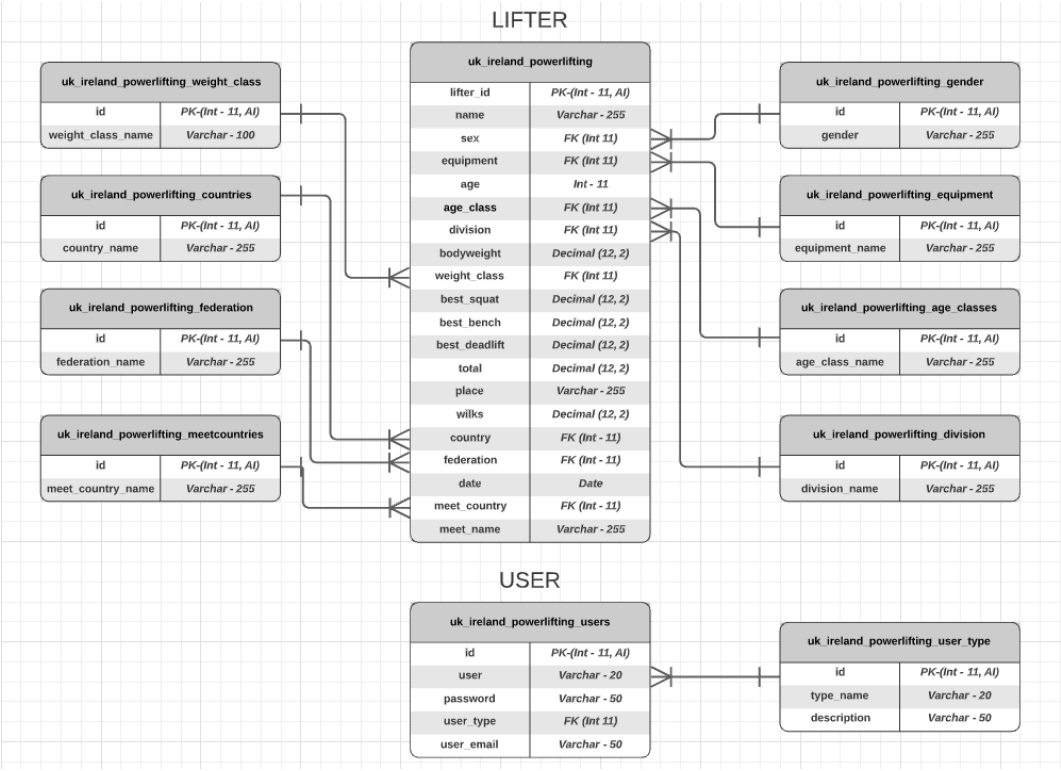
The MySQL database for this website is based on the dataset from Open Powerlifting and can be found at https://www.kaggle.com/open-powerlifting/powerlifting-database?select=openpowerlifting.csv. An explanation of why this dataset was chosen can be found in the in-term report. To reduce the number of rows from over the original 1 million+, the filter functionality within Microsoft Excel[1] was used. Only the columns which contained information intended to be shown on the final website where kept, which left approximately 20.

Cleaning Data

A PHP script was written in Visual Studio Code (VSC)[2] to take the dataset from its original CSV format and import it into the MySQL database. This script provided several uses with the first being the speed taken to import 3000+ rows automatically at the click of a button. Next, as some of the rows contained blank data, it allowed for the "cleaning" or processing of the data before it was imported. Finally, as the database had been designed to reduce reoccurring entries, many of the columns had to be processed to coincide with the foreign keys (FK) within the database, ultimately allowing for normalisation to happen[3].

Database Design

Normalisation with database design cuts down on repeating data entries and allows for easier management of data, including entering, updating, and deleting data. For this reason, data within the MySQL database were normalised out into their own separate tables, using FK constraints to link them. All the relationships in this particular database where one to many. For example, many lifters could be in the same weight class and one weight class could have many lifters. The design of the tables and their relationships is represented in the below entity relationship diagram (ERD).



Importing Data

To import the dataset, a connection within the MySQL database was created within the PHP script and run by simply requesting the URL on a web browser.

API

Design & implementation

An Application Programming Interface (API) acts as a middleman between the client and the database[4] and allows a client (someone who does not have direct access to the MySQL database) to carry out functionality like creating data, reading data, updating data, or deleting data, otherwise known as CRUD functionality[5]. The API, which this website uses, is based on the above database, and was designed and implemented with CRUD functionality in mind. A widely recognised format for clients to send and receive data from an API is in JSON format, confining with the architectural constraints of a REST API (Representational State Transfer)[6, 7, 8]. Links to some of the API endpoints used within this website, along with request bodies and authentication details can be found in the Appendix.

Security

Basic Authentication was used to add a layer of security to the API. This method allows the client to pass a simple username and password through the HTTP request header, which is then checked against the corresponding API credentials[9]. HTTP requests to this API were sent from the website using a combination of cURL (a PHP library)[10, 11], AJAX[12] and XMLHttpRequest[13], allowing for Basic Auth details to be passed in the header.

Comparison (Proposed vs Current Functionality)

Proposed Functionality	Current Functionality
• Display informative information on powerlifting. • Show recent trends in graph format using the proposed dataset. • Have a responsive, modern user interface (UI) design. • Display statistics on lifters, representative of the data that a web-user can search and filter. • Include a contact form that makes it possible to submit questions to the website. • Link a social media account or a news feed, to the home page that would show recent posts and articles.	• Displays information about what powerlifting is and an explanation of the three main exercises that make up the sport; the squat, bench-press and deadlift. • Shows 4 trends from the data contained in the database, in graph format that the web-user can change by clicking on the button they desire. • The UI design and implementation is modern and responsive to smaller screen sizes. Most of the front end was created using Bootstrap. • Relevant statistics about each lifter is displayed in a table format that can be searched and filtered. • Web-users can utilise the contact form to submit questions they may have. • A login process introduces security and authentication, letting admin and registered users to carry out additional CRUD functionality. • Registered admin users of the site can carry out full CRUD functionality, which makes requests to the REST API. • From the “Stats” page, once a lifter’s name is clicked, an expansion of that particular lifter is displayed showing all occurrences of that lifter with a dynamic graph representing their progress over time.

- Successfully implemented functionality from the proposed original.

- Functionality that was not implemented into the final website from the proposed original.

- Extra functionality included that was *not* originally proposed.

Bootstrap[21], JavaScript (including the popular JS library; jQuery[22]) and a custom CSS sheet make up a large part of the websites front end.

Most of the originally proposed functionality was successfully implemented into the website achieving the original goal of building an informative website on powerlifting. A jQuery plug-in named Datables[14] was utilized allowing the easy pagination, search capabilities and column ordering of data that could be integrated and styled with Bootstrap's table classes. The data shown within the tables is a result of displaying the return of a request made to the API, achieved using JavaScript's AJAX method[12].

Graphs are used at 2 places within the website. The first being the Trends page which again, requests data from the API through a HTTP request, processes the data and displays 4 different trends in a bar chart format with one example being average wilks score (a formula used to compare the strength of lifters) for male lifters over different weight classes. Web-users can switch between graphs without having to move to a different page, because of jQuery's ".click(function())" [15].

The second occurrence of graphs on the website, is when an individual lifters information is shown, this time in the form of a line graph presenting a visual representation of a lifter's progress over time. To get the information needed for this graph, another request was made to an API endpoint, with a URL parameter of the lifters name being passed. The JavaScript library used to achieve these graphs is Chart.js[16].

The social media content was not included in the final implementation of the website because of the project specification stating, "The use of any social media, e.g., Twitter, Facebook API's or any external frameworks should be avoided."

As the development of the site progressed, additional knowledge that was gained in the process opened the doors for new functionality and features. This additional functionality concentrated around having a login feature that allowed administrative users of the site to carry out full create, read, update, and delete (CRUD) capabilities[5] on the data entries.

The process of creating and updating lifter entries, involves the admin web-user filling in or editing existing information of a standard HTML form[17]. Once the submit button on the form is clicked, a JavaScript "event listener" triggers a function that packages the form data up into JSON format and sends it to the API endpoint using an XMLHttpRequest[13] object. The sending of data in JSON format, coincides with the recommendation of the software architectural style, REST (Representational state transfer)[6, 7, 8].

For the Read and Delete functionality, requests are sent to the API using PHP's cURL library[10]. With the GET requests (used to allow read functionality), the JSON response format is decoded into an associative array[18] and "echoed" out into a HTML table format where appropriate.

Security and authentication were enforced on the website largely from PHP's $_SESSION supergloabal variable[19], allowing information to be stored in a variable that could be used across multiple pages. When a user logs in, the credentials are authenticated against the stored user and password details in the database. If there is a match a $_SESSION variable is set and subsequently used to check the login status on other pages of the site. This checking serves two uses; to prevent web-users from accessing secure pages directly through the URL (in which case they are redirected to the login page) and to determine which content/ page is shown (e.g., if the user is logged in as an admin, the Stats page will look different compared to a general visitor to the website).

Modularisation and code efficiency was increased by making use of PHP's include() function[20], HTML's <script src=''></script> tag and an appropriate ordering of directory files within Visual Studio Code. The aim with this is to reduce the repetition of code, making it easier to read and update, as there will only be one place to make changes. Examples of this are that all pages point to one CSS page to apply styling and code for reoccurring layouts like navigation bars, header images and footers may only be written once.

Admin Login Credentials

Username: admin

Password: password123

References

1. Microsoft. 2020. Excel Spreadsheets (2020). [Software]. [Accessed on 21/04/2021].
2. Microsoft. 2021. Visual Studio Code (2021). [Software]. [Accessed on 21/04/2021].
3. Juliano Rabelo (August 14, 2020), Techopedia, Foreign Keys. Accessed on 21/04/2021 using https://www.techopedia.com/definition/7272/foreign-key#:~:text=A%20foreign%20key%20is%20a,establishing%20a%20link%20between%20them.
4. Petr Gazarov (December 19, 2019), freeCodeCamp, What is an API? In English, please. Accessed on 21/04/2021 using https://www.freecodecamp.org/news/what-is-an-api-in-english-please-b880a3214a82/
5. CodeCademy, What is CRUD? Access on 21/04/2021 using https://www.codecademy.com/articles/what-is-crud
6. John Au-Yeung & Ryan Donovan (March 2, 2020), The Overflow, Best practices for REST API design. Accessed on 21/04/2021 using https://stackoverflow.blog/2020/03/02/best-practices-for-rest-api-design/
7. Vaibhav Kandwal (April 22, 2020), freeCodeCamp, REST API Tutorial – REST Client, REST Service, and API Calls Explained With Code Examples. Accessed on 21/04/2021 using https://www.freecodecamp.org/news/rest-api-tutorial-rest-client-rest-service-and-api-calls-explained-with-code-examples/
8. Zell Liew (January 17, 2018), Smashing Magazine, Understanding and using REST API's. Accessed on 21/04/2021 using https://www.smashingmagazine.com/2018/01/understanding-using-rest-api/
9. Hacking with PHP, Authentication over HTTP. Accessed 21/04/2021 using http://www.hackingwithphp.com/15/4/3/authentication-over-http
10. PHP.net. Client URL Library (2020). Accessed on 21/04/2021 using https://www.php.net/manual/en/book.curl.php
11. PHP.net. Basic curl example (2020). Accessed on 21/04/2021 using https://www.php.net/manual/en/curl.examples-basic.php.
12. W3schools.com, AJAX Introduction (2021). Accessed on 21/04/2021 using https://www.w3schools.com/xml/ajax_intro.asp
13. W3schools.com, XMLHttpRequest (2021). Accessed 21/04/2021 using https://www.w3schools.com/xml/xml_http.asp
14. SpryMedia Ltd. DataTables (2021) [Software]. Accessed on 21/04/2021 using https://datatables.net/
15. W3Schools.com, jQuery click() method. Accessed 21/04/2021 using https://www.w3schools.com/jquery/event_click.asp
16. Nick Dawnie (2020). Chart.js. [Software] Accessed on 21/04/2021 using https://www.chartjs.org/
17. MDN Web Docs (2021), <form>. Accessed on 21/04/2021 using https://developer.mozilla.org/en-US/docs/Web/HTML/Element/form
18. PHP.net. Json_decode (2020). Accessed on 21/04/2021 using https://www.php.net/manual/en/function.json-decode.php
19. W3schools.com, PHP Sessions. Accessed on 21/04/2021 using https://www.w3schools.com/php/php_sessions.asp
20. PHP.net. Include (2020). Accessed on 21/04/2021 using https://www.php.net/manual/en/function.include.php
21. Bootstrap, v4.6 (2021). [Software]. Accessed 21/04/2021 using https://getbootstrap.com/
22. JQuery, v3.6.0 (2021). [Software]. Accessed 21/04/2021 using https://jquery.com/

Appendix

API Details

The documentation for the API can be found at:

https://documenter.getpostman.com/view/14969191/TzJu8c5g

Alternatively, the API end point details can be found below. Please note that the API has been created so that it only accepts data in the body in JSON format. This applies to any POST and PUT requests. If testing in Postman, sending these key – value pairs in a "form-data" format will *not* work.

The Basic Authentication details for *every* endpoint is:

- Username: admin
- Password: password123

POST - Create a new lifter entry.

http://dmcauley21.lampt.eeecs.qub.ac.uk/powerliftingcentralv2/api/lifter/create.php

Body

```
{
  "name": "Joe Blog",
  "sex": "M",
  "age": "32",
  "bodyweightkg": "79",
  "best_squat": "150",
  "best_bench": "110",
  "best_deadlift": "190",
  "place": "1",
  "country": "Ireland",
```

```
  "federation": "IPF",
  "date": "09/04/2021",
  "meet_country": "China",
  "meet_name": "China Championships"
}
```

GET - View all lifters.

http://dmcauley21.lampt.eeecs.qub.ac.uk/powerliftingcentralv2/api/lifter/read.php

---

GET - View a single lifter.

http://dmcauley21.lampt.eeecs.qub.ac.uk/powerliftingcentralv2/api/lifter/read_single.php?lifter_id=45090

Request Params

lifter_id	45090

GET - Search lifters by name.

http://dmcauley21.lampt.eeecs.qub.ac.uk/powerliftingcentralv2/api/lifter/read_name.php?name=Stephen%20Smith

Request Params

name	Stephen%20Smith

PUT - Update a lifter entry.

http://dmcauley21.lampt.eeecs.qub.ac.uk/powerliftingcentralv2/api/lifter/update.php

Body


```
{
  "lifter_id": "45090",
  "name": "Stephen Smith Updated",
  "gender": "Male",
  "age": "43",
  "bodyweight": "110.40",
  "best_squat": "220.00",
  "best_bench": "140.00",
  "best_deadlift": "195.00",
  "place": "2",
  "country_name": "Ireland",
  "federation": "PA",
  "date": "2015-04-26",
  "meet_country_name": "Australia",
  "meet_name": " NSPC Mini Raw Open II"
}
```

DELETE - Delete a lifter entry.

http://dmcauley21.lampt.eeecs.qub.ac.uk/powerliftingcentralv2/api/lifter/delete.php?lifter_id=45090

Request Params

lifter_id	45090