Manchester Metropolitan University

Department of Computing and Mathematics

ASSIGNMENT COVER SHEET

Unit title:	6G5Z2107: Web Design and Development
Assignment set by:	Stuart Cunningham
Assignment ID:	2CWK50
Assignment title:	The PHP-focused Assessment
Assignment weighting:	50%
Type: (Group/Individual)	Individual
Hand-in deadline:	13th December 2019 at 21:00
Hand-in format and mechanism:	Submission is online, via Moodle. More information is available in the attached coursework specification.
Support:	Staff office hours are on Moodle along with support videos from each of the relevant labs. More information is available in the attached coursework specification.

Learning outcomes being assessed:

LO1: Deploy client-side JavaScript libraries to add dynamic functionality within a web page;

LO2: Use open-source tools and technologies to develop object-oriented and reusable server-side scripts that obtain, validate, process and store user input from web pages;

LO3: Integrate client-side and server-side coding into coherent web applications;

LO4: Manage communication sessions to authorise users and perpetuate their data across multiple pages.

Note: it is your responsibility to make sure that your work is complete and available for marking by the deadline. Make sure that you have followed the submission instructions carefully, and your work is submitted in the correct format, using the correct hand-in mechanism (e.g. Moodle upload). If submitting via Moodle, you are advised to check your work after upload, to make sure it has uploaded properly. Do not alter your work after the deadline. You should make at least one full backup copy of your work.

Penalties for late hand-in: see Regulations for Undergraduate Programmes of Study (http://www.mmu.ac.uk/academic/casqe/regulations/assessment.php). The timeliness of submissions is strictly monitored and enforced.

All coursework has a late submission window of **5 working days**, but any work submitted within the late window will be capped at 40%, unless you have an agreed extension. Work submitted after the 5-day window will be capped at zero, unless you have an agreed extension.

Please note that individual tutors are unable to grant extensions to coursework. Extensions can only be granted on the basis of a PLP, or approved Exceptional Factors (see below).

Exceptional Factors affecting your performance: see Regulations for Undergraduate Programmes of Study (https://www.mmu.ac.uk/academic/casqe/regulations/assessment/docs/ug-regs.pdf). For advice relating to exceptional factors, please see the following website: https://www2.mmu.ac.uk/student-case-management/guidance-for-students/exceptional-factors/ or visit a Student Hub for more information.

Plagiarism: Plagiarism is the unacknowledged representation of another person's work, or use of their ideas, as one's own. Manchester Metropolitan University takes care to detect plagiarism, employs plagiarism detection software, and imposes severe penalties, as outlined in the Student Handbook (http://www.mmu.ac.uk/academic/casqe/regulations/docs/policies_regulations.pdf and Regulations for Undergraduate Programmes (http://www.mmu.ac.uk/academic/casqe/regulations/assessment.php). Bad referencing or submitting the wrong assignment may still be treated as plagiarism. If in doubt, seek advice from your tutor.

As part of a plagiarism check, you may be asked to attend a meeting with the Unit Leader, or another member of the unit delivery team, where you will be asked to explain your work (e.g. explain the code in a programming assignment). If you are called to one of these meetings, it is very important that you attend.

Assessment Criteria:	Indicated in the attached assignment specification.
Formative Feedback:	A formative submission opportunity is available on the following date: **22nd November 2019** More details about the formative submission opportunity is available in the attached coursework specification.
Summative Feedback Format:	Written feedback in the form of a commented mark grid, plus a general comment on the whole submission. Generalised feedback will be offered to all students as a group.

1. Introduction

This assessment is coursework based, and has a single main component, worth 50% of the overall unit mark. The tasks you are required to complete for this assessment are detailed in this coursework specification.

2. Aim

This unit encourages you to gain practical experience of the world of web development. By the end of the unit, the idea is that you have completed the development of two large-scale web applications, each of which incorporate several technologies – mirroring the sorts of tasks you would be required to carry out in the role of a web developer. These completed web applications can form examples of completed work for future job applications, projects for you to discuss when applying for placement opportunities, may be useful tools for you to make use of in your later work, or form part of a portfolio for any future creative endeavours.

This aim of this coursework is to give you a platform to demonstrate what you have learned about working with and making use of persisting data in dynamic web environments. This will involve sending and receiving data between the client and server, in a web scenario, and being able to store data using client and server-side approaches.

In particular, the following skills will be essential for successful completion of this coursework:

- **Problem solving:** You will need to develop solutions for many problems along the way as you encounter them whilst developing your solution. You can apply any problem-solving techniques you have learnt in your studies so far to these difficulties.
- **Investigation skills:** You will need to examine and understand existing processes and systems to help shape your own designs.
- **Technical skills:** You will need to develop your PHP, MySQL and JavaScript skills in order to complete this assessment, supported by the fundamentals of HTML.
- **Project planning:** The assessment requires you to plan, and consider which elements of the work you will attempt in which order. You may find you cannot complete one bit without another, but finding solutions to this (through planning and careful testing) is part of the challenge.

3. Coursework Overview

To complete this assessment task, you are required to develop a survey website. The precise details of the coursework are explained in Section 4 below, but the website should allow users to register and log in, create, manage and share surveys, and analyse the data obtained in surveys.

Of course, a survey tool can be used for lots of other purposes, such as application or request forms. A special “admin” account allows developers to monitor *who* is using the site and perform administrative functions.

4. The PHP-focused Assessment (2CWK50)

a. Outline

To complete this assessment, you must further the development of a dynamic, server-driven website using PHP, from skeleton code that has been provided for you[1]. Specific details of what you need to add are detailed below. **You must work from the skeleton code you have been provided. All code you submit must be your own, unaided work.** You must not submit code you have acquired from **any other sources** including, but not limited to, online tutorials or repositories.

Your submission will be marked in the following thematic areas:

- User Accounts
- Analysis and Design
- Survey Management
- Survey Results
- Code Quality

Please ensure you have examined the mark scheme closely, to ensure that work you are completing is part of the mark scheme.

b. Additional Guidance

User Accounts

Users should be able to sign-up to the site using a unique username (one not already in use), and a password. Users should then be able to log in to the site using sessions (supported by cookies to store session IDs) and enter their account data, which is stored in a database table. This data should be inputted using a simple form and should *at least* include the following details:

- A first name
- A surname
- An email address
- A date of birth
- A telephone number

The form should validate all user input on the client-side, using the rules below, and these validation steps should be repeated on the server-side to prevent any potential abuse:

- Check a valid first name has been supplied AND that it's not too long for the database table
- Check a valid surname has been supplied AND that it's not too long for the database table
- Check an email address has been supplied AND that it's a valid email address AND that it's not too long for the database table
- Check a date of birth has been supplied AND that it's a valid date
- Check a telephone number has been supplied AND that it's a valid telephone number AND that it's not too large for the database table

The site should also feature an Administrator (admin) account, which can be used to keep control of the site using the username: "admin" and the password: "secret". Admin should be able to access a list of all users and, upon clicking on a username, that user's associated account information should be displayed (note: your create_data.php script should create fake profile data for a sensible number of users to allow you to test this functionality). The admin user should be able to manage user accounts by performing tasks such as: update a user's account details; change a user password; create a user account; delete a user account; and so on. Ideally, the admin should also be able to view and manage all user surveys, which gives

[1] Available on the *Web Design and Development* Moodle area.

them the ability to amend or delete any questions or surveys that they consider to be unsuitable.

As a starting point, you can refer to:

- Persisting Data II – Sessions (see the lab sheet about starting and ending sessions as well as managing user session data)
- Persisting Data III – Databases and Remote Storage (see the lab sheet about interacting with databases and tables using PHP)
- Persisting Data IV - Client and Server Validation (see the lab sheet for examples of validating profile data on both the client-side and the server-side)

Analysis and Design

You are not required to do any programming for this area. Instead, you should document the investigation and appraisal you have done into existing survey websites and explain how these have influenced your own website design and implementation. You can think on this task as being a small piece of competitor analysis, where you want to explore what's already out there, so you can try to improve upon it, avoid the negatives, and integrate the positives.

There are lots of web-based survey tools out there already and it's a great idea to create trial accounts so that you can research these systems. This will help you to shape your own designs and functionality. A few of the sites particularly worth reviewing are:

- Survey Monkey: https://www.surveymonkey.com/
- Google Forms: https://www.google.com/forms/about/
- Lime Survey: https://www.limesurvey.org/
- Survey Planet: https://surveyplanet.com/
- Checkbox: https://www.checkbox.com/

Your analysis of competitor sites should follow an approach that you can decide for yourself. Examining each site and evaluating it against a common set of criteria will make it easier for you to draw comparisons between them. You can define the criteria yourself, but you may wish to consider things like:

- Layout/presentation of surveys
- Ease of use
- User account set-up/login process
- Question types
- Analysis tools

You should include your analysis and design documentation as part of the site itself and a new page for you to extend (called competitors.php) has been included in the skeleton code. You should make use of simple HTML formatting to improve the clarity of your analysis wherever appropriate. For example, using tables, bullet point lists, images, hyperlinking to relevant materials, and so forth.

This element of the coursework can be started straightaway and you are advised to begin it as soon as possible to come up with initial designs and plans for your own web site before you start to develop your PHP code.

Survey Management

In its simplest form, users should be able to activate and share a simple survey that already exists as part of the overall site (a PHP file containing a survey that has already been created by you). This simple survey should contain at least 5 questions and use at least 3 different types of question (for example: numeric entry; text entry; selecting a value from a drop-down menu; selecting a value from a series of radio buttons; selecting values using check boxes; selecting a value using a slider; and so on). There should be a database that will collect the responses of each participant (the person completing the survey). The type and topic of survey you choose to create for this is up to you. It could be about favourite foods, pets, computer games, films, television, music, and so on. You are free to use your imagination. However, as inspiration, you might find the following types of question format useful to get started:

- Customer Satisfaction Questions: https://survicate.com/customer-satisfaction/survey/most-popular-questions/
- Sample Survey Questions Answers and Tips: https://www.constantcontact.com/aka/docs/pdf/survey_sample_qa_tips.pdf

But, a static survey isn't all that suitable to the end users of our site. Ideally, we want them to be able to create their own surveys, about any topic they like. A much more powerful approach, would be to let them create their own surveys. In this way, users can have features, such as:

- Create surveys
- Set the title and instructions for their survey
- Decide how many questions there will be in each survey
- Choose the question types
- Set the wording or information presented as part of the questions and response options
- Have a unique URL for each survey

This is a much more valuable tool and allows custom surveys to be created by each user. Changes to a survey should appear dynamically, without the need for a page refresh. You may also want to consider introducing client and server-side validation, depending upon the question choices for the survey, to make sure that participants are entering acceptable responses.

To take things further, you can introduce more advanced types of survey structure and questions. You should think about your own innovative features to add, but you could think consider features like:

- Mandatory questions that must be answered
- Use of logic in survey progression (e.g. "If the answer to Q6 is YES then go to Q10 otherwise go to Q7")
- Follow-up questions related to previous question responses (e.g. "Earlier, you rated our product as *somewhat poor*, please explain your reasons for this choice")
- Questions that require participants to watch a video or perform a task before answering
- Questions demanding different types of interaction (e.g. selecting a point in two-dimensional space, such as a graph or image)
- Questions that display examples of existing responses and ask the participant to provide their own view or opinion (e.g. "Here is a summary of how others have rated this product, do you agree with this response?"

As a starting point, you can refer to:

- Case Study – Toy Workshop (see the code for examples of how to let users create, edit and delete their data)
- Persisting Data V – APIs and Data Structures (see the lab sheet for more on working with multiple tables)

Survey Results

Once surveys start to receive responses our users will want to take a look at the data they are getting from participants. Basic functionality in processing the results will include being able to view the raw survey data as well as to perform high-level summaries of it, depending upon the question types. The following are examples of basic survey result processing tasks that are required:

- View all survey responses in a table
- Indicate total number of survey responses
- Indicate total number of responses to each survey question
- Summaries of responses to constrained questions (e.g. "42 participants own a dog and 68 participants own a cat")

This type of functionality can then be expanded to make the results more visually appealing and useful to users, by representing it in visually compelling ways. These could then be used in reports or documents, for example. To do this, users should be able perform tasks such as:

- Graphs (bar, line, pie, column and so on) of survey and question responses. This information should be provided to developers in graphical form, and you are strongly advised to investigate the use of JavaScript plugins and libraries like Google Charts.
- Export raw survey responses in a suitable format (CSV, XLS, etc.)

To take things further, you may wish to introduce other ways for users to inspect their responses in a deeper way. You should think about your own innovative features you can add, but you could consider features like:

- Update the results page dynamically, without the user having to reload the page
- Being able to filter results (e.g. "Show me the ratings for my product only from participants who own a dog")
- Export filtered survey responses (e.g. "Export only the raw data from participants between the ages of 21 and 45") in a suitable format (CSV, XLS, etc.)
- Graphs showing the distribution of responses, such as for questions using a Likert scale[2], etc.
- Cross-plotting data from questions (e.g. "Plot participant age against their rating of the product to see if there is a trendline")
- Word clouds for participant text box responses. You should investigate JavaScript libraries that can be used to help
- Suitable statistics for questions and surveys (e.g. "What is the mean, minimum and maximum survey participant age?")

As a starting point, refer to:

- Persisting Data VI – Images and Dashboards (see the lecture slides and lab sheet for more on graphical summaries and interactive filtering)
- Case Study – Toy Workshop (see the code for an example of how display and filter information from user wish lists)

[2] You'll have seen these, where questions have responses like: 1 = Strongly disagree; 2 = Disagree; 3 =Neutral; 4 = Agree; and 5 = Strongly Agree. See here for more info: https://www.britannica.com/topic/Likert-Scale

Code Quality

Code quality is very important and it's vital that you consider this component right from the start. You should be commenting *all* the new code that you write. It's fine if your files originate from the lab exercises or skeleton code, but the comments from those original files don't count towards your marks here. It's the *new* comments you've written to describe the *new* lines of code you've added to the file that are important.

You should ensure that you are using good practices when writing your code, particularly with reference to indention, the use of curly braces, whitespace, and semi-colons, etc. As a starting point, you may consider following a style guide[3] to help you format your code in a readable, professional manner. Ensuring that you are structuring your code *consistently* is an essential component to earn good marks here.

You will be rewarded appropriately for use of sensible features of PHP to solve your problems. Appropriate use of variables, arrays, objects and functions will all earn you marks – and misusing these features may be penalised.

You should include a README.txt file as part of your submission. This should include details on how to set up your submission (e.g., "run create_data.php", "navigate to login.php", a username/password to log in with, etc.) and documentation for how your site works. The documentation doesn't need to be very long, but it does need to clearly explain what your site can do, covering all its individual pages.

A useful tip is to break the documentation up into sections for each of the features in the mark scheme (User Accounts, Design and Analysis, Survey Management, Survey Results) and briefly describe how each feature works. Once you have completed the documentation for each feature you have addressed, you should naturally find you have described all of the pages in your site.

c. Submission

Submission of this coursework will be online, through the University's Virtual Learning Environment (Moodle). You must include all of your code, your README file (see previous section), and a create_data.php file that sets up your database, database tables, and testing data (a starting example for you to extend is provided in the skeleton code).

Please make sure to compress your work into a single file in the ***.zip format.** This is the only accepted format for submission of your work. When compressing these things, it is important that your directory structure is preserved.

[3] For example, the Framework Interoperability Group's PHP style guide: https://github.com/php-fig/fig-standards/blob/master/accepted/PSR-2-coding-style-guide.md

5. Marking Scheme

Your work in this coursework will be graded in several areas, attracting a number of marks for each. These areas are identified in the coloured columns shown in the table below. Guidance on the specific assessment criteria for each area is included below. Your marks awarded for each area will be combined to give an overall mark for 2CWK50, which is worth 50% of the final unit mark.

User Accounts			Analysis and Design			Survey Management			Survey Results			Code Quality		
New Account Features	**Mark**	**/**	**Chosen Site 1**	**Mark**	**/**	**Simple Survey**	**Mark**	**/**	**Simple Results**	**Mark**	**/**	**Commenting**	**Mark**	**/**
Username		1	Layout/presentation of surveys		1	Database table(s) working		2	Show total # respondents		1	Comments added		1
Password		1	Ease of use		1	3 types of question in form		3	Show # question respondents		1	Comments meaningful		1
First name		1	Account set-up/login process		1	5 questions in a survey form		5	Display all responses in a table		4	Useful README included		1
Surname		1	Question types		1			**10**	Summarize question responses		4			**3**
Email address		1	Analysis tools		1	**Custom and Complex Surveys**					**10**	**Coding Style**		
DOB		1			**5**	Create custom survey		1	**Graphs and Data Manipulation**			Indentation		1
Telephone		1	**Chosen Site 2**			Set survey details (title, etc.)		1	Static Graphs and Charts		2	Consistency		1
Login successful		1	Layout/presentation of surveys		1	Setup survey questions		1	Interactive Graphs and Charts		2	File naming		1
Update account details		2	Ease of use		1	Client validation		1	Export responses to CSV or XLS		1	Site visually attractive		1
		10	Account set-up/login process		1	Server validation		1	Other advanced features		5			**4**
Client Validation			Question types		1	Other advanced features		5			**10**	**Use of PHP**		
First name		1	Analysis tools		1			**10**				Advanced PHP features		1
Surname		1			**5**							Own functions		1
Email address		1	**Chosen Site 3**									Suitable data structures		1
DOB		1	Layout/presentation of surveys		1									**3**
Telephone		1	Ease of use		1									
		5	Account set-up/login process		1									
Server Validation			Question types		1									
First name		1	Analysis tools		1									
Surname		1			**5**									
Email address		1	**Overall Analysis**											
DOB		1	Quality of writing		3									
Telephone		1	Depth of critical analysis		3									
		5	Overall conclusions		3									
Admin Account			Images where appropriate		1									
Tools available		1			**10**									
Manage users		2												
Manage surveys		2												
		5												
Section Total		**25**	**Section Total**		**25**	**Section Total**		**20**	**Section Total**		**20**	**Section Total**		**10**

For example, a student earning 12 marks in *User Accounts*, 8 in *Analysis and Design*, 15 marks in *Survey Management*, 20 marks in *Survey Results*, and 5 marks in *Code Quality* will receive a total mark of 60% (12 + 8 + 15 + 20 + 5).

6. Support for this Assessment

Help! I don't know where to begin or what to do!

Don't panic! This assessment has many starting points, and many routes through the piece of work, which can be scary if you do not know where to start. We introduced important concepts for the assessment in the laboratory exercises, so perhaps you could think about which laboratory exercises you are most comfortable with, and start building your assessment work from these.

In the first instance, you should flag any queries or concerns you have with the unit teaching team (in the studio session, or, by email), with the understanding that the sooner you talk to them, the sooner a resolution can be reached.

If in doubt, examine the mark scheme, and try to plan where you think you can earn the marks to get at least a passing grade. If you can pick a starting point (e.g. working on your user accounts), and decide you are aiming for 10 marks in this area, you might find you can exceed your original expectations once you have made a start. To borrow an analogy from the art world: *draw the big brushstrokes first and then start to add colour and detail as you go along.*

You should make good use of the materials provided for you on Moodle. There are numerous resources which you may find useful – particularly those in the *Looking to take it further?* section of each week. Lynda.com videos and the W3C Schools materials are an excellent source of information and a great way to refamiliarize yourself with some of the core concepts of the unit.

It may also be advisable to acquire one or more of the books on the unit reading list, which you can check on Moodle.

Opportunities for Formative Feedback

There is an opportunity for formative feedback on your coursework progress on **22nd November 2019**. You **must** submit your "work in progress" at this date to receive feedback on your progress so far. An online Moodle submission will be available for you to submit your work – how ever far you have progressed – which also offers a way to practice preparing your work for your final submission.

Your Final Feedback

You will receive written feedback on your assignment, in the form of a commented assessment grid identical to that included above, with a short comment on each thematic area, and a general comment covering your entire piece of work.

When, where and how can I get support from the unit team?

Assessment support is available by arrangement, and if you find that you are struggling with the assessment, you should contact your tutor **at the earliest possible opportunity** to arrange support.

You can contact your tutor using the following details:

Dr Stuart Cunningham
John Dalton E114
s.cunningham@mmu.ac.uk
0161 247 1459

Please check Moodle, or the signs on my office door, to confirm my most up-to-date office hours.