Assessment Information/Brief 2025-26

Module title	**Client Server Systems**
CRN	50249
Level	5
Assessment title	Assignment 1 – Server Side Application
Submission/Assessment Date	The submission deadline is **10/11/2025** by no later than 16:00. Any submission received after 16:00 (even if only by a few seconds will be considered as late). For coursework assessments only: students with a Reasonable Adjustment Plan (RAP) or Carer Support Plan should check your plan to see if an extension to this submission date has been agreed.
Module Leader/ Assessment set by	**Lee Griffiths contact via MS Teams here**
Weighting within module	This assessment is worth 50% of your overall module mark.

Assessment task details and instructions

Your task overall for the module is to develop a data-driven online client server system called "**petWatch**" aimed at managing and providing a social network to help pet owners find pets that have gone missing. The system should also allow users who spot a pet that looks missing to add a "sighting" – more details on the specific user journey further down.

You must build the system using **PHP**, an **SQL** data source and **HTML/CSS** using an **Object Oriented** approach to the code and data acquisition and the Model-View-Controller (MVC) design pattern and template provided in workshops.

The work will be split over both trimester 1 and 2. The first trimester and this brief focuses on core architecture and data storage. The second trimester will add JavaScript and mapping so users can see the location of sightings on a map and there will be another brief that covers that part.

Noteworthy details to gain high marks in Trimester 1 include:

- The core PHP architecture **needs** to follow an **Object Oriented, Model-View-Controller** (MVC) Design Pattern approach. This is covered in the lectures and workshops and you practice it by following the workshop material closely. **You are not allowed to use jQuery or a high level PHP framework.** You need to use and adapt the MVC templates that we provide and that you use in the workshops and this is **very important!**

- For the system developed there should be two types of users. 1) a pet owner who can manage listings for their pets in the system using a Create-Read-Update-Delete (CRUD) approach 2) a browsing user who can browse and search pet descriptions as well as leaving a "sighting" status for a particular pet.

- The system must be secure and therefore all users are required to login before they can do anything else. For users, each record needs to include: a unique id, a unique username, a password. All passwords must be appropriately encrypted in the database.

- Browsing users do not need to login to browse or search the data but are required to login to leave a sighting of a pet (to be discussed in lectures).

- A pet record will be unique; a sighting record will need to contain location (at least latitude/longitude which are crucial for trimester 2 work).
  The detailed design of the database and tables for this work will be covered in lectures and a collaborative exercise between lecturers and students.

- For testing and **marking** you must have **specific test** accounts with these exact details:

Pet Owner	**Browsing user**
• User ID (auto generated in the DB) • Username: Lee • Password: See advice below • User type: Owner	• User ID (auto generated in the DB) • Username: Zara • Password: See advice below • User type: User

Passwords need to follow a recognised standard e.g.

Password security starts with creating a strong password. A strong password is:
- At least 12 characters long but 14 or more is better.
- A combination of uppercase letters, lowercase letters, numbers, and symbols.
- Not a word that can be found in a dictionary or the name of a person, character, product, or organization.

(ref: https://support.microsoft.com/en-gb/windows/create-and-use-strong-passwords-c5cebb49-8c53-4f5e-2bc4-fe357ca048eb)

- The website's main home page must be named **index.php and** should start with a welcome page containing appropriate menu links to browse and search the data as well as login. There is no need to create a user registration page and your user accounts should be generated and added to the database manually following the

advice on passwords above – this is to reduce the possibility of a security breach in your system.

- Browsing users should be able to list all **pets** which should indicate their current status (missing or found) and be able to search for a pet based on parameters such as type, description, location and the search should be sophisticated **enough to narrow down results** including ordering of the results. Splitting the list into pages with suitable page navigation will get the highest marks.

- Authenticated browsing users can provide updates on the status of a pet e.g. “Spotted near the bus station” – but that will be handled in Trimester 2 so no need to work on this feature for Trimester 1 other than having appropriate fields in the database tables which will be discussed in class.

- Authenticated (logged in) pet owner users should be able to manage (CRUD) all of their pet entries including missing status update. e.g. “found”

- Listings and Search results should be displayed using HTML with CSS (the use of Bootstrap or Materialize or other CSS frameworks is encouraged). **You are not allowed to use jQuery** because **1) you won’t need to for what is required** and **2) in Trimester 2 you will need to develop raw JavaScript** in your solution and you are marked on your ability to develop and implement well designed JavaScript code.

- The database design will be developed in the lectures in week 3 so that all students use the same design. You will need to revise your L4 Databases SQL work as the solution will contain multiple database tables that are lined together using **foreign keys.** You must use SQLite at the datasource as shown in Workshop 7.

- There will be a need to use the geolocation data (latitude/longitude) for the work in Trimester 2 which involves maps (e.g. Google maps) so think about how you will include this information in a pet sighting record and you can store dummy information for now.

- Your site must be designed and built to handle 100’s of owners and pet records and 100’s of browsing users. You can generate test/mock data with a tool like **www.mockaroo.com** or **ChatGPT** and this will be discussed in lectures.

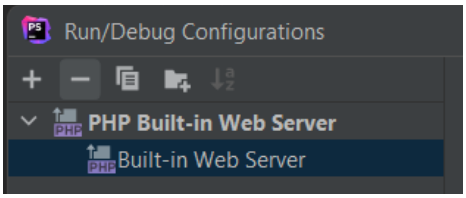
- You must consider **security and performance** at every step and design the user experience for multiple platforms and abilities. For example, security should consider filtering malicious code from any text that users can enter, including SQL injection (details on Blackboard). Performance should minimise page weight including media associated with an item description,

- You must create and administer the database using only SQLite technology as covered in Worksop 7 exercises. Your system must be made to run on a Microsoft Windows installation of phpStorm and XAMPP with the PHP CLI installed at c:\ xampp\php\php.exe consistent with the workshops exercises, demonstrations in lectures and the university lab computer installations. This is **very important!**

Using Generative Artificial Intelligence (GenAI) tools	The module content provides a good starting point for an MVC design pattern using Object Oriented approach to data access so the template projects provided are recommended. Any code in the module slides and workshops can and should be used. The code can be improved on and extended, but any substantial blocks of code copied from someone else or found on the internet, or deemed generated by AI that directly affects the solution will be deemed as **plagiarism**. Allowing others to copy your code will be deemed as **plagiarism**. More information can be found here https://www.salford.ac.uk/askus/topics/admin-essentials/academic-misconduct#definitions
Word count/ duration (if applicable)	Not applicable however see the next section on what you need to submit.

How to submit **Please read this carefully.**

1. Your solution must be created on a Microsoft Windows (10/11) based PC to run on localhost using phpStorm's "Built in Web Server", with the XAMPP based PHP interpreter and SQLite data libraries. When the project is opened on any other PC with the same configuration it must work. You cannot easily develop the work on a MacOS or Linux OS as it will not be compatible with the marking device specification.



   If your solution does not properly load into phpStorm or load correctly from localhost when run you will get 0% because it cannot be marked.

2. A Word or .pdf document **must be submitted** to Blackboard in the Assessments area which contains all your main project code including the .php and .phtml files presented neatly and readable in the following way:

   a. Front page with full name, university ID and title (Client Server systems Assignment 1) **and URL of your system** e.g. **localhost:8000/assignment1**
   In case your work does not run properly we made need to contact you and this is why the submission is not to be anonymous.

   b. Pet owner and browsing user account details including passwords that will be used when marking the work so should be easy to copy and paste into the system login area (see later).

c. a copy of the Assessment Criteria and Marking Scheme grid (see later) with your self-assessment of your performance for each requirement – use the yellow highlighter tool like this in Word to highlight what you have completed.

d. Model files

e. Controller files and associated View files

f. Any CSS files you have created (not Bootstrap files)

g. Clear screenshot of the main database table records.

3. A compressed .**zip** folder (not **rar**) **must be submitted** to Blackboard in the Assessments area containing the complete folders and all files associated with your solution to the assessment task – there is a typical submission limit of 50MB – you should not need to exceed that.

For this assignment we would expect around 6 model files and 4 controller/view pairs. This is just a guide. There is a sample document on Blackboard.

Feedback	Your work for this assignment will be marked from the live system online and the document containing your code. Feedback, and marks will be provided via Blackboard as soon as possible and aiming for week 12 of Trimester 1. No marks will be released until all submissions are marked and an indicator of marking progress will be given on Blackboard and updated regularly. Feedback will be in the form or a breakdown of the marks for each row in the indicative mark scheme (previous page) and according to the achievements (columns) with some formative statements supporting the work done.
Assessment criteria	**See "Marking Scheme" on the next page (page 6)** You should look at the assessment criteria to find out what we are specifically looking at during the assessment.
Assessed intended learning outcomes	On successful completion of this assessment, you will be able to: 1. Understand a range of server-side technologies 2. Design, create, test and demonstrate software implementing a data-driven web application,

programming in industry standard scripting languages and connecting to industry standard database packages;

Marking scheme Percentage Mark	**Technical implementation (20 marks)**	**User Feature (20 marks)**	**Browsing User Display (20 marks)**	**Search feature (20 marks)**	**Create/Edit Records (20 marks)**
Outstanding 90–100%	**An outstanding implementation of OO code using the MVC pattern consistently and effectively. Inheritance, classes, properties, and methods are applied with precision and meaningful names. Demonstrates outstanding performance enhancements. Code is exceptionally well organised, excellently commented, and highly readable. Fully meets all technical requirements using SQLite via phpStorm/XAMPP/PHP CLI. Documentation is of outstanding quality.**	**An outstanding user feature set, including secure login with encrypted passwords, anti-spam, sessions, validation, and malicious code filtering. Driven by outstanding OO design and excellently commented code. Test accounts fully working.**	**An outstanding browsing feature, allowing retrieval and display of pet and sighting details with responsive CSS/Bootstrap, efficient paging, and large datasets. Driven by outstanding OO design and excellently commented code.**	**An outstanding interactive faceted search that reliably narrows output to fewer than 10 results using all record parameters. Implements complex SQL statements. Driven by excellent, logical OO design. Excellently commented code. Fully tested and fully functional.**	**An outstanding implementation of record management, allowing owners to add, view, and edit pet records, with browsing users able to leave sighting reports. Driven by outstanding OO design and excellently commented code.**
Excellent 80–89%	**An excellent implementation of OO code using the MVC pattern. Inheritance and class structures are well designed and appropriately applied. Includes some excellent performance enhancements. Code is well organised, consistently commented, and clearly readable. Meets technical requirements. Documentation is excellent.**	**An excellent user feature set, including secure login with password encryption, session management, and validation. Driven by excellent OO design and well-commented code. Test accounts working.**	**An excellent browsing feature, retrieving and displaying pet/sighting details with images. Includes a responsive layout and basic paging. Well-structured OO design, well commented.**	**An excellent free-text search facility using 3 parameters combined. Good OO implementation with well-commented code. Functional and reliable search with minor issues.**	**An excellent implementation of record management, with owners able to add/view records and browsing users able to leave reports. Good OO design and excellent commenting.**
Very Good 70–79%	**A very good implementation of OO code with strong adherence to MVC. Some inheritance and design are evident, though not always consistent. Code is generally well organised, readable, and well commented. Documentation is very good. Meets technical requirements with only minor issues.**	**A very good user feature set, with functional login, encryption, and session management. Some validation attempted. Driven by very good OO design and adequately commented code. Test accounts mostly working.**	**A very good browsing feature, retrieving item details with some formatting (tables/divs). Layout functional, with very good OO design and adequate comments.**	**A very good free-text search facility working but basic. Some filters implemented, minor issues present. Basic OO implementation. Some code comments included.**	**A very good implementation of record management. Users can add/view records and leave reports, though with some limitations. Adequate OO design and comments.**
Good 60–69%	**A good implementation of OO code with reasonable adherence to MVC. Classes, methods, and database access are generally correct, with some gaps or inconsistencies. Some enhancements are attempted. Code is readable with sufficient comments.**	**A good user feature set, including working login and sessions. Basic OO design and sufficient commenting. Test accounts functional.**	**A good browsing feature, retrieving some details with simple listings. Basic OO structure, sufficient comments.**	**A good free-text search facility implemented with noticeable limitations. Some basic OO**	**A good implementation, with partial record management features working, though with issues. Basic OO design.**

	Documentation is good. Meets most technical requirements.			**design.**	
Fair 50–59%	**A fair implementation of OO code with partial use of MVC. Limited inheritance or inconsistent class use. Code organisation is variable, and comments are present but basic. Documentation is fair. Meets most but not all technical requirements.**	**A fair implementation of user features, with login partly functional or inconsistent. Limited session handling and sparse commenting. Test accounts partly working.**	**A fair attempt at browsing features. Data displayed but with errors or weak layout. Minimal OO use. Comments basic.**	**A fair basic search facility that may have issues with filtering or multiple parameters. Minimal OO design. Some code comments included.**	**A fair attempt at record management, with features partly functional but inconsistent. Sparse comments.**
Adequate 40–49%	**An adequate attempt at OO coding with minimal adherence to MVC. Limited understanding of inheritance or class design. Code is weakly structured, with few comments. Documentation is adequate but incomplete. Meets requirements but with significant runtime issues.**	**An adequate attempt at user features, with incomplete or buggy login. Minimal commenting. Test accounts unreliable.**	**An adequate attempt, retrieving limited data with buggy or inconsistent output. Poor commenting and weak code quality.**	**An adequate basic text search partially implemented; may not return correct results. Sparse code comments. Limited OO design.**	**An adequate attempt, with limited record creation/editing and weak OO design. Poor presentation.**
Needs Improvement 30–39%	**A system needing improvement, showing only a small amount of code with weak OO implementation. Very limited or superficial use of MVC. Organisation and commenting are poor. Documentation is minimal. Does not fully meet technical requirements.**	**A system needing improvement, with only a basic or broken login attempt. Poorly presented, with weak OO design.**	**A system needing improvement, attempting to retrieve data but not fully functional. Weak presentation, some reliance on copied code.**	**A needs improvement text search implemented only in part; often fails or returns incorrect data. Poorly structured code. Significant copied content.**	**A system needing improvement, with attempted record features but largely non-functional. Poor presentation and some reliance on copied code.**
Needs Significant Revision 20–29%	**A system requiring significant revision, with very little working OO code. MVC is used incorrectly or superficially and not reliably running on the required stack. Documentation is missing or highly limited. Heavy reliance on copied code from.**	**A system requiring significant revision, with very limited login functionality and little evidence of secure or structured coding.**	**A system requiring significant revision, with barely functional retrieval and poor documentation.**	**A needs significant revision search/filters partially implemented or mostly non-functional. Little or no OO design. Minimal or no code comments. Extensive copied code.**	**A system requiring significant revision, with very minimal attempt at record features and little structure.**
Needs Substantial Work 0–19%	**A system needing substantial work, with no functioning OO design or MVC implementation. Code is largely non-functional, poorly presented, and dependent on copied content. Not built on the specified platform and failing to run. Documentation is absent.**	**A system needing substantial work, with no functioning login or user features.**	**A system needing substantial work, with no ability to retrieve or display database details.**	**A needs substantial work system with no functional search implemented. No OO design. Little or no code comments. Extensive copied code.**	**A system needing substantial work, with no implementation of record management.**

Employability skills developed / demonstrated

You will develop a range of employability skills sought by employers through each assessment.

Through this assessment will have an opportunity to develop and demonstrate the following employability skills:

(please put a cross in the box for the skill and level demonstrated in the assessment)

Skill	I	U	A	D
Communication	x			
Critical Thinking and Problem Solving			x	
Data Literacy			x	
Digital Literacy			x	
Industry Awareness			x	
Innovation and Creativity			x	
Proactive Leadership	x			
Reflection and Life-Long Learning			x	
Self-management and Organisation			x	
Team Working	x			

I = You will have been introduced to this skill

U = You will have developed an understanding of this skill in the context of your subject

A = You will be able to apply this skill in the context of your subject

D = You will have demonstrated an enhanced understanding and application of this skill in a wider context

Support for this Assessment

You can obtain support for this assessment by attending all timetabled sessions and working through all published material on Blackboard. You can ask for help in timetabled workshop sessions and **MS Teams chat** with the module leader. During busy parts of the teaching period MS Teams replies may take up to 3 working days and you must have excellent attendance in timetabled sessions for questions to be answered via MS Teams. Make sure you have explored all the teaching materials before asking questions. When asking for help be specific and show the problem using screen shots (snipping tool in Windows), code sections and error messages.

Other sources of support

	Understanding your assessment brief/assessment tips for success Develop your academic and digital skills Assessment rules and processes Support services **Issues affecting your assessment** If exceptional circumstances have affected your ability to complete this assessment, you can find more information about the Exceptional Circumstances Procedure (previously Personal Mitigating Circumstances) here. Independent advice is available from the Students' Union Advice Centre. **Academic Integrity and Academic Misconduct** You must learn and demonstrate good academic conduct (academic integrity). Good academic conduct includes the use of clear and correct referencing of source materials. Academic integrity & referencing Referencing Academic misconduct is an action which may give you an unfair advantage in your academic work. Some examples are plagiarism, asking someone else to write your assessment for you, unauthorised use of AI or taking notes into an exam. The University takes all forms of academic misconduct seriously.
In year retrieval scheme	Your assessment is not eligible for in year retrieval.
Reassessment arrangements	If you fail your assessment, and are eligible for reassessment, you will be able to find the date for resubmission on your module site in Blackboard. There is no resubmission if you are on a retake attempt. For students with accepted personal mitigating circumstances for absence/non submission, this will be your replacement assessment attempt. We know that having to undergo a reassessment can be challenging however support is available. Have a look at all the sources of support outlined earlier in this brief. Assignment 2 of this module (in Trimester 2 – January to May) will require you to focus on some aspects of your Assignment 1 system and refining them by making use of more advanced dynamic client server technologies to

improve the user experience (UX) e.g. dynamic searches and real-time interaction, geolocation, imaging as appropriate all using AJAX techniques – details to follow.

Completion of the work in Trimester 2 depends on the completion of the work in Trimester 1 and you will be supported in Trimester 2 if you have not met the requirements for a pass in the Trimester 1, Assignment 1 work.