



Assessment Brief

Module Code

SEN6000

Module Title

Advanced Programming

Academic Year

25/26

Semester

1

Module Leader email

Bal-shargabi@cardiffmet.ac.uk

Contents

ASSESSMENT DETAILS.....	2
SUBMISSION DETAILS	5
ASSESSMENT CRITERIA	5
FURTHER INFORMATION	9
Who can answer questions about my assessment?	9
Referencing and independent learning	9
Technical submission problems	9
Late submission and mitigating circumstances	9
Academic Misconduct.....	9
How is my work graded?.....	10

Assessment Details

Assessment title	Abr.	Weighting
WRIT1 – written Reassessment	WRIT1	75%
Pass marks are 40% for undergraduate work and 50% for postgraduate work unless stated otherwise.		

Task/Assessment brief:

Project Tasks: Based on your project proposal, do the following

1. Designing the Homepage: (5%)

Create the project (web, mobile, or desktop) application and add a background image. Make sure to keep the image in the same directory as the project to avoid inconsistencies. Next, your code must set the title of the library project window with an appropriate library name. Then, when you run the code, the window should appear. Explain the process step by step by providing screenshots and code snippets.

2. Database Connection (10%)

- Establish a connection to the database using the correct credentials for MySQL or any other database. Ensure the application can interact with the database as needed. Provide detailed explanations and screenshots or code snippets. Also explain how you did you handle security in your connection

3. Adding the Buttons/links(15%)

- Add CRUD (Create, Read, Update, Delete) buttons to the application's interface: Then, write the code(provide code snippets or screenshots) and explanations, you should also provide a sample code snippet or screenshots. It is also required to connect each button to the database.
 - Delete Record: Remove a record from the database.
 - Update Record: Modify an existing record in the database.
 - Search Record: Retrieve specific data from the database.
 - Add Record: Insert a new record into the database.

4. Software design, architecture and implementation: (60%)

4.1 Software Design (20%)

- UML Diagrams:** Create UML diagrams (Use Case and Class diagrams) based on for the for your project.
- Design Patterns:** Define and plan the implementation of core design patterns (e.g., Singleton, Factory) and why.
- Adherence to Design Principles:** Ensure the design follows SOLID principles or the DRY principle and provide an example with a sample code snippet.
- Software Architecture style:** Follow the MVC architecture, and provide evidence in the documentation through diagrams and code examples (code snippets)

4.2 Software Implementation (40%)

- Module Implementation:** Implement all modules of your project as proposed for your case study.

- **Advanced Data Structures:** Use advanced data structures (e.g., trees, graphs) where applicable for modules, at least two modules and provide a sample code snippet and explain the selection of the data structure.
 - **Core Design Patterns Implementation:** Apply and implement core design patterns (e.g., Singleton, Factory) in any of the modules of your project along with a sample code snippet and explanations.
 - **Concurrency Features:** Develop features that handle concurrent operations (e.g., managing multiple user requests simultaneously) along with a sample code snippet.
 - **Exception Handling:** Integrate robust exception handling throughout the application.
 - **Unit Testing:** Write unit tests for a specific module (e.g., Delete Record button). Include detailed input, output, expected outcomes, and pass/fail results, with screenshots of tests and outcomes.
 - **Integration Testing:** Perform integration testing across the developed modules and provide sample of test cases and documented results.
5. **Prepare comprehensive documentation covering the design, architecture, implementation, and testing of the project. Ensure the documentation meets professional and ethical standards (for example, citations, references, figures, tables, and appendix). (10%)**
 6. **Reflections:** Reflect on your journey in completing this assessment. Your reflection should evaluate the main challenges you faced and the strategies you used to meet the requirements of the brief. If relevant, refer to the feedback from your previous submission. (10%)

Word count (or equivalent):

3000 +/-10% words

This is a reflection of the effort required for the assessment. Word counts will normally include any text, tables, calculations, figures, subtitles and citations. Reference lists and contents of appendices are excluded from the word count. Contents of appendices are not usually considered when determining your final assessment grade.

Academic or technical terms explained:

- **Design Patterns** – Proven, reusable solutions to common software design problems (e.g., Singleton, Factory).
- **Design Principles** – Guidelines for building maintainable and scalable code (e.g., SOLID, DRY).
- **MVC Architecture (Model–View–Controller)** – A common software design structure that separates data (Model), user interface (View), and control logic (Controller).
- **Modules** – Distinct parts of a system, each responsible for a specific functionality.
- **Advanced Data Structures** – Specialised ways of organising and storing data (e.g., trees, graphs) to support efficient operations.
- **Concurrency** – Ability of a program to execute multiple tasks at the same time (e.g., handling multiple user requests simultaneously).
- **Exception Handling** – A Mechanism for managing and responding to errors in a controlled way during program execution.
- **Unit Testing** – Testing small, individual parts (units) of code to verify they work correctly.
- **Integration Testing** – Testing how different modules or components of a system work together.

Artificial Intelligence Models – Guidance for this assessment:

Artificial Intelligence (AI) models can be a powerful tool to support your learning. The University has provided some resources to support you in its appropriate usage:

- [Library Services AI Hub](#)
- [Student Guide to AI and Assessment](#)
- [Code of Conduct for Students on the use of AI](#)
- [Cite Them Right resource on citing materials relating to AI \(if permitted\)](#)

As per the academic regulations ([Academic Handbook Ah1_08](#)), in all cases you must submit work that is your own, acknowledging any part of it that has been informed by another source – including that which is AI generated. Upon submission of work, you will be asked to confirm the following statement:

I confirm that this assignment is my own work, except where I have acknowledged the use of works from other sources, including the use of any artificial intelligence (AI) tools, in accordance with what is allowable as described in the assessment brief.

Please note the following:

- AI should not be used as a substitute for your own knowledge, and you should never include any material that you do not understand and could not explain if asked.
- Not being able to explain your work when asked is likely to be a key factor when considering cases of academic misconduct related to AI.

The following information provides specific guidance for this assessment about what level of AI use is appropriate for this assessment. Remember that in all cases you must submit work that is your own, acknowledging any part of it that has been provided by another source.

NO USE OF GENERATIVE AI EXPECTED • Your assignment should be produced using information sourced by you from your learning materials and academic sources and cited appropriately. • AI tools for checking spelling, grammar and referencing may be used.	
AI ACKNOWLEDGED • You can use AI tools to learn about your topic, as part of your study, or in preparing initial guidance on assignments (e.g. headline structure, suggestions for inclusion of topics). • Any materials that you have sourced from AI should be rewritten or reconfigured and integrated into your own work and referenced appropriately. It is recommended that this is confirmed by a relevant academic source. • Any support gained from AI should be acknowledged in a statement at the end of the assignment, making clear what the support was, and how you used it and developed it for your own work. Example statements are available in the Student Code of Conduct.	
AI EMBEDDED • Use of AI is an integral and expected part of the assessment. • The explicit inclusion of AI within the assessment means that instructions on the expected use will be part of the assessment brief. • Your assessment brief will describe how you should acknowledge the way in which you used AI tools.	

Submission Details

Submission Deadline:	This will be provided on the Moodle submission point.	Estimated Feedback Return Date	This will normally be 20 working days after initial submission.
Submission Time:	By 4.00pm on the deadline day.		
Moodle/Turnitin:	Any assessments submitted after the deadline will not be marked and will be recorded as a non-attempt unless you have had an extension request agreed or have approved mitigating circumstances. See the School Moodle pages for more information on extensions and mitigating circumstances.		
File Format:	The assessment must be submitted as a pdf document (save the document as a pdf in your software) and submit through the Turnitin submission point in Moodle. Your assessment should be titled with your: student ID number, module code and assessment ID, e.g. st12345678 BHL5007 WRIT1		
Late Submission Window Eligibility	Where submissions are eligible for the late-submission window this will be communicated in the relevant assessment submission point within Moodle.		
Feedback	Feedback for the assessment will be provided electronically via Moodle. Feedback will be provided with comments on your strengths and the areas which you can improve. View the guidance on how to access your feedback. All marks are provisional and are subject to quality assurance processes and confirmation at the programme Examination Board.		

Assessment Criteria

Learning outcomes assessed
• Evaluate the theory and concepts of contemporary/industry standard programming and design in the software development life cycle. • Review industry standards of professional and ethical software development, software carpentry and codemanship. • Construct advanced programming concepts through consistent problem-solving and program implementation. • Validate the correctness of complex programs with advanced type systems.
Other skills/attributes developed
This includes elements of the Cardiff Met EDGE (Ethical, Digital, Global and Entrepreneurial skills) and other attributes developed in students through the completion of the module and assessment. These will also be highlighted in the module guidance, which should be read by all students completing the module. Assessments

are not just a way of auditing student knowledge. They are a process which provides additional learning and development through the preparation for and completion of the assessment.

Ethical	Usability, and user experience are two important factors when designing and implementing a software application that is interacted by the users. The information, messages generated by the system, and information collected by the client program (e.g., personal data) should be handled, saved and processed with proper care adhering to international and regional laws and regulations.
Digital	Software engineering and programming skills accumulated during the course and through the assignment will enhance student's hands on digital skills, use of IDEs, build tools, inter-component/process communication skills, and industry standard programming and design best practices.
Global	Software Engineering principles design patterns discussed in this class universally accepted. Skills gained through this are location and language agnostic and are applicable across the globe.
Entrepreneurial	Problem solving skills, knowledge acquired on client-server architecture, programming and system designing skills are pivotal for anyone who expects to aspire to be an entrepreneur in a technical domain.

Marking/Assessment Criteria

Criteria	Learning Outcome(s)	Weighting
Designing the Homepage	LO1	5%
Database Connection	LO1, LO2	10%
Adding CRUD Buttons/Links	LO1, LO4	10%
Software Design (UML, Patterns, Principles, Architecture)	LO2, LO3	20%
Software Implementation (Modules, Data Structures, Patterns, Concurrency, Exception Handling, Testing)	LO1, LO2, LO3, LO4	35%
Documentation	LO3	10%
Reflection		10%
Total		100%

Marking/Assessment Criteria

Criteria	1-29% (Clear Fail)	30-39% (Narrow Fail)	40%-49% (3rd)	50%-59% (2:2)	60%-69% (2:1)	70%-79% (1st)	80%-100% (1st Class +)
Designing the Homepage (5%)	Very poor or no homepage; lacks core elements (e.g., no background image, title not set)	Poor homepage design; minimal features; inconsistent layout; title or image missing, no evidence of code snippet and screenshot	Average design; includes basic features but lacks polish; some inconsistencies of evidence of code snippet and screenshot	Fairly clear design; all required features present; design is mostly consistent and functional, limited code code snippet and screenshot	Clear, functional design; well-structured layout; consistent elements, clear code snippet and screenshot	Excellent, intuitive design; very well-structured and user-friendly; visually appealing, descriptive code snippet and screenshot	Outstanding, user-friendly design; highly engaging, clear, and well-polished; professional quality, with code snippet and screenshot
Database Connection (10%)	Very poor or no database connection; unable to function; insecure and improperly configured, no explanation and code screenshot	Poor connection; frequent errors, unreliable; basic, non-parameterised connection prone to security risks, poor or no explanation and code screenshot	Average connection; some issues, slow or unreliable at times; minimal security, limited use of parameterisation, limited explanation and code screenshot	Fairly stable connection; minor issues but generally functional; some security features and basic parameterisation implemented with explanations and screenshot	Clear connection; stable, functional, and responsive; secure connection using proper parameterization to prevent SQL injection, with clear explanation and screenshot	Excellent connection; very stable, reliable, and responsive; well-secured with parameterized queries, ensuring data integrity and security and Excellent explanation and screenshot	Outstanding connection; efficient, fast, stable, and error-free; highly secure, fully parameterized queries, with robust authentication and encryption measures and Excellent explanation and screenshot
Adding CRUD Buttons/Links (10%)	Very poor or no CRUD functionality; buttons/links not connected; no explanation provided and evidence like a screenshot of code	Poor CRUD implementation; some buttons/links don't work; incomplete or unclear explanation and evidence, like a screenshot of code	Average CRUD implementation; basic functionality, but some features may be limited or inconsistently explained and a screenshot	Good CRUD implementation; all buttons/links work; clear explanation with some details and Screenshots	Very good CRUD implementation; detailed explanation, code snippet, well-documented, consistent functionality	Excellent CRUD implementation; well-documented with clear, code snippet detailed explanations; all features fully functional	Outstanding CRUD functionality; highly detailed and precise documentation; code snippet, exceptional implementation, seamless integration
Software Design (UML Diagrams, Design Patterns, MVC Architecture) (20%)	Very poor or no design; lacks clarity, coherence, and adherence to design principles; no or very poor UML diagrams; no effective use of design patterns; no MVC structure	Poor design; weak coherence and adherence to principles; UML diagrams are incomplete or incorrect; design patterns poorly applied; MVC structure is unclear	Average design; some principles applied but inconsistently; basic UML diagrams; generic use of design patterns; partial MVC structure evidence	Good design; fairly clear adherence to principles; clear UML diagrams; appropriate use of design patterns; reasonable MVC structure evidence	Very good design; strong application of design principles; accurate UML diagrams; consistent use of patterns; well-structured MVC evidence	Excellent design; thorough use of principles; detailed UML diagrams; well-documented design patterns; highly effective MVC implementation with evidence	Outstanding design; seamless integration of principles; highly detailed, precise UML diagrams; innovative, effective use of design patterns; clear, efficient, and well-documented MVC architecture.

Criteria	1-29% (Clear Fail)	30-39% (Narrow Fail)	40%-49% (3rd)	50%-59% (2:2)	60%-69% (2:1)	70%-79% (1st)	80%-100% (1st Class +)
Software Implementation (Modules, Data Structures, Concurrency, Exception Handling, Testing) (35%)	Very poor or no implementation; missing core modules; no use of advanced data structures; lack of concurrency; no error handling; little to no testing provided	Poor implementation; incomplete modules; minimal or incorrect use of data structures; weak concurrency; minimal exception handling; poor testing, with few or unclear examples of testing	Average implementation; basic modules present but may be incomplete; generic use of data structures; basic concurrency; inconsistent error handling; limited testing with some examples of automated unit and integration testing provided	Good implementation; most modules functional; reasonable use of data structures; clear concurrency; sufficient error handling; good testing, including examples of how automated unit and integration testing were conducted	Very good implementation; complete modules; appropriate and efficient use of data structures; effective concurrency; robust exception handling; thorough testing with clear examples of automated unit and integration testing methods	Excellent implementation; well-structured and complete modules; advanced and effective use of data structures; efficient concurrency; strong error handling; comprehensive testing; detailed examples of automated unit and integration testing with clear explanations on how tests were designed and executed	Outstanding implementation; seamless, highly efficient modules; innovative use of data structures; flawless concurrency; exceptional error handling; detailed, well-documented examples of unit and integration testing, showing a clear testing strategy and methodology
Documentation (10%)	Very poor or no documentation; lacks clarity and structure; missing key sections, no references, or citation of tables and figures	Poor documentation; incomplete, unclear, or difficult to follow; some sections are missing, no references, or citation of tables and figures	Average documentation: covers basic aspects but lacks depth; some inconsistencies. Inconsistent use of references, or citation of tables and figures	Good documentation; clear and detailed; most required sections covered, consistent use of references, or citation of tables and figures	Very good documentation; comprehensive, clear, and logically structured; all sections well-covered, good use of references, or citation of tables and figures	Excellent documentation; highly detailed, professional, clear, and consistent; well-organised, Excellent use of references, or citation of tables and figures	Outstanding documentation; exceptional clarity, detail, and coverage; professional standard, well-structured, and easy to follow, use of references, Outstanding or citation of tables and figures and appendices
Reflection (10%)	No reflection provided; fails to address feedback or show improvements	Minimal reflection; feedback mentioned but not adequately addressed or implemented	Basic reflection; some feedback considered, but changes are minimal or unclear	Good reflection; feedback acknowledged and some changes implemented, with brief explanations	Very good reflection; feedback clearly acknowledged; changes implemented with explanations on how feedback improved the report	Excellent reflection; feedback fully integrated into the project; clear, detailed explanations of how feedback was addressed and improved the project	

Further Information

Who can answer questions about my assessment?

Questions about the assessment should be directed to the staff member who has set the task/assessment brief. This will usually be the Module Leader. They will be happy to answer any queries you have.

Staff members can often provide feedback on an assignment plan but cannot review any drafts of your work prior to submission. The only exception to this rule is for Dissertation Supervisors to provide feedback on a draft of your dissertation.

Referencing and independent learning

Please ensure you reference a range of credible sources, with due attention to the academic literature in the area. The time spent on research and reading from good quality sources will be reflected in the quality of your submitted work.

Remember that what you get out of university depends on what you put in. Your teaching sessions typically represent between 10% and 30% of the time you are expected to study for your degree. A 20-credit module represents 200 hours of study time. The rest of your time should be taken up by self-directed study.

Unless stated otherwise you must use the **HARVARD** referencing system. Further guidance on referencing can be found in the Study Smart area on Moodle and [using Cite Them Right](#) (use your university login details to access the site). Correct referencing is an easy way to improve your marks and essential in achieving higher grades on most assessments.

Technical submission problems

It is strongly advised that you submit your work at least 24 hours before the deadline to allow time to resolve any last minute problems you might have. If you are having issues with IT or Turnitin you should contact the IT Helpdesk on (+44) 2920 417000. You may require evidence of the Helpdesk call if you are trying to demonstrate that a fault with Moodle or Turnitin was the cause of a late submission.

Late submission and mitigating circumstances

If you are experiencing personal difficulties which are impacting your ability to engage with assessment, there are a range of support options available, including via your Personal Academic Tutor, the [Mitigating Circumstances](#) procedure and the [Support to Study procedure](#).

The Mitigating Circumstances policy and procedure sets out when and how students can:

- Submit eligible assessments late but have it recorded as 'on time', up to four times per level of study.
- Self-declare significant exceptional circumstances in order to defer the assessment to the next submission opportunity.

More information about Late Submission and Mitigating Circumstances is available on [MetCentral](#).

Students with a disability or chronic illness should contact the University's Student Wellbeing Team to discuss any appropriate reasonable adjustments which may be made to their learning and assessments.

Academic Misconduct

Cardiff Met takes issues of academic misconduct **extremely seriously**. The University has procedures and penalties for dealing with academic misconduct. These are explained in full in the University's Academic Misconduct regulations and procedures under [Volume 1, Section 8](#) of the Academic Handbook. The Module Leader reserves the right to interview students regarding any aspect of their work submitted for assessment.

Types of Academic Misconduct, include:

Plagiarism, which can be defined as using without acknowledgement another person's words or ideas and submitting them for assessment as though it were one's own work, for instance by copying, translating from one language to another or unacknowledged paraphrasing. Further examples include:

- Use of any quotation(s) from the published or unpublished work of other persons, whether published in textbooks, articles, the Web, or in any other format, where quotations have

not been clearly identified as such by being placed in quotation marks and acknowledged.

- Use of another person's words or ideas that have been slightly changed or paraphrased to make it look different from the original.
- Summarising another person's ideas, judgments, diagrams, figures, or computer programmes without reference to that person in the text and the source in a bibliography/reference list.
- Use of assessment writing services, essay banks and/or any other similar agencies (NB. Students are commonly being blackmailed after using essay mills).
- Use of unacknowledged material downloaded from the Internet.
- Re-use of one's own material except as authorised by your degree programme.

Collusion, which can be defined as when work that has been undertaken with others is submitted and passed off as solely the work of one person. Modules will clearly identify where joint preparation and joint submission are permitted, in all other cases they are not.

Fabrication of data, making false claims to have carried out experiments, observations, interviews or other forms of data collection and analysis, or acting dishonestly in any other way.

How is my work graded?

Assessment grading is subject to thorough quality control processes. You can view a summary of these processes on the Assessment Explained Infographic.

Grading of work at each level of Cardiff Met degree courses is benchmarked against a set of general requirements set out in [Volume 1, Section 4](#) of our Academic Handbook. A simplified version of these Grade Band Descriptors (GBDs) with short videos explaining some of the academic terminology used can be accessed for [Foundation](#), [1st year](#), [2nd year](#) and [3rd year](#) undergraduate and [MSc programmes](#).

We would strongly recommend looking at the [Study Smart](#) area of Moodle to find out more about assessments and key academic skills which can have a significant impact on your grades. Always check your work thoroughly before submission.

Cardiff Met

MetCaerdydd