Coursework Brief and Front Sheet PGT

This front sheet for assessments is designed to contain the brief, the submission instructions, and the actual student submission for any WMG assessment. As a result, the sheet is completed by several people over time and is therefore split up into sections explaining who completes what information and when. Yellow highlighted text indicates examples or further explanation of what is requested, and the highlight and instructions should be removed as you populate 'your' section. This sheet is only to be used for components of assessment worth more than 3 CATS (e.g. for a 15 credit module, weighted more than 20%; or for a 10 credit module, weighted more than 30%).

To be completed (highlighted parts only) by the Programme Administration after approval and prior to issuing of the assessment; to be consulted by the Student(s) so that you know how and when to submit:

Date set	November 2025
Submission date (excluding extensions)	Monday 15th December 2025 by 12:00pm (UK Time)
Submission guidance	To be submitted electronically via Tabula
Marks return date (excluding extensions)	21st January 2026

To be completed by the Module Leader/Tutor prior to approval and issuing of the assessment; to be briefed to the Student(s) so that they understand the Coursework Brief, its context within the module, and any specific criteria and advice from the tutor:

Module Title & Code	Programming for Artificial Intelligence WM9QF-15
Component Type	Individual Assignment
Module Leader	Amir Kayhani
Module Tutor	Amir Kayhani
Assessment Title	Assignment
Weighting (%)	70%

Coursework Brief

Task 1: Data Insights Dashboard for Public Health Reports

You have been asked to develop a Python-based **data insights tool** for a team of researchers analysing **public health data** (e.g., vaccination rates, disease outbreaks, or mental health reports).

The goal is to support **data access, filtering, cleaning, summarisation, and presentation** (not predictive modelling). This simulates what a software developer might build to support an AI or data science team.

Design and implement a Python program that:

Core Functionalities*:

1. **Data Access & Loading**
    - Read data from **at least one source:**
        - Public dataset (e.g. CSV, JSON)
        - Public API (e.g. WHO, UK government data, etc.)
        - A database (e.g. relational database)
    - Load the data into a local or cloud database.
2. **Data Cleaning & Structuring**
    - Handle missing or inconsistent data
    - Convert types (dates, numbers)
    - Create data structures (e.g., dictionaries, lists of records)
3. **Filtering and Summary Views**
    - Allow users to filter data by criteria (e.g., country, date range, age group)
    - Generate summaries such as:
        - Mean, min, max, or counts
        - Trends over time
        - Grouped results (e.g., by country or region)
4. **Presentation Layer**
    - Command-line interface (CLI), menu, or simple UI
    - Generate visual outputs (e.g., charts with matplotlib, tables with pandas)
5. **Extension Features (according to the scenario's requirements)**
    - CRUD functionalities on the DB to create, read, update and delete the data.
    - Export filtered data or summaries as CSV
    - Log all the user activities into a log file

***Core functionalities** are indicative requirements of the software solution, and you may identify other requirements as part of your requirement analysis.

Reflective Report Guidance for Task 1

Your report should critically document your software development. Including **(<u>but not limited to</u>)**:

1. **Project overview**: Aim, scope, key functionality
2. **Software engineering and design**:
    - How your code is organised
    - Requirement analysis
    - Data structures used and why
3. **External tools**:
    - Which libraries/APIs did you use, and what alternative libraries or APIs could you use
4. **Testing and debugging**:
    - Strategy used
5. **Use of AI-generated code** (e.g. GitHub Copilot, ChatGPT):
    - Where and how did you use these tools
    - Whether you modified or validated outputs
    - Ethical considerations and originality
6. **Reflection**:
    - Critically evaluate the implemented project and discuss the alternative methods that could have been used. Consider the future expansion of the project, optimisation and requirements of using HPC methods (e.g. GPU/CUDA computing)

You must follow the programming and software development/engineering best practices and provide all the software development artefacts and documentations (e.g. UML and ER Diagrams, etc.) in your reflective report.

Task 2: Data Structure and Algorithm

Implement a Python code that includes the following components. You must use an **appropriate data structure** to represent and manipulate the data efficiently, based on the scenario provided.

1. **Data Structure Design**
    - Choose and implement an appropriate structure (e.g., graph, tree, or queue-based structure) to model the data.
    - Ensure it efficiently supports updates or queries as required.
2. **Algorithm Implementation**
    - Implement one or more relevant algorithms from the module content to implement the scenario's requirements.
    - Algorithms must operate on the data structure you created.
3. **Application-Specific Extension**

    - Other requirements, such as visualisation of the implemented data structure (if applicable), as specified in the scenario.

Reflective Report (Analysis and Evaluation) Guidance for Task 2:

Submit a short technical report (750–1000 words) covering the following:

1. **Justification of Design**
    - Justify your choice of data structures and algorithms.
    - Analyse computational complexity.
    - Discuss alternative approaches and why your choice was suitable.
2. **Evaluation and Scalability**
    - Critically evaluate how your solution would perform with larger or dynamic datasets.
    - Discuss its suitability for real-world use cases.

For Both Tasks 1 and 2:

You are expected to adopt a **test-driven development (TDD)** approach during the project. This means you should:

- **Write test functions first**, before implementing the full functionality
- Ensure that each new function or feature is supported by one or more **automated test cases**

Your final code submission must include:

- A separate test file or module in your Git repository (e.g. test_main.py)
- A link to your Git repository for each task on the first page of the report, for each task **<u>and</u>** the screenshots of your code, along with the outputs in your report. You must make sure your Git repository is accessible for all the module tutors but not for everyone. (you must create private repositories but add the module tutors as collaborators)
- Tests that demonstrate correctness and handle edge cases (e.g., empty input, missing data, invalid types)
- The quality of the code and programming are important and including comments to explain the code is essential.
- Your submission must include a Readme file to explain how to run the codes (commands, etc.)

⚠ Writing tests after the full implementation may result in reduced marks for code structure and software engineering practice. You must use Git for version control. Your repository history will be reviewed for appropriate frequency, quality and commits.

⚠ Accurate referencing and in-text citation are essential.

⚠ **Using AI is <u>only</u> permitted for writing or fixing programming code.** You can copy and adapt the codes that your tutors shared with you during the module (in the Git repo) without referencing, but you need to provide references and in-text citations if you copy or adapt any code from any other sources.

Assessment Length/ Word count	The maximum word count is 2,800 words for both tasks combined. The word count figure includes quotations, tables, figures, footnotes, endnotes, in-text citations, titles, abstracts & summaries. It does not include the questions as set, or other material from the rubric or supplementary information such as tables of contents, tables of figures, tables of tables, the reference list, or bibliographies. Material submitted in an annex or appendix is also outside the word count. There is +10% allowed on the word count without penalty. If you exceed the word count by more than 10% to 30%, a penalty of 10% on the original mark awarded will be applied from the original total mark but capped at the pass mark. If you exceed the word count by more than 30%, the final mark will be capped at the pass mark.		
ARTIFICIAL INTELLIGENCE (AI)			
AI Scale		**NO AI**	You must not use AI at any point during the assessment. You must demonstrate your core skills and knowledge.
		AI PLANNING	You may use AI for planning, idea development, and research. Your final submission should show how you have developed and refined these ideas.
	✓	**AI COLLABORATION**	You may use AI to assist with specific tasks such as drafting text, refining and evaluating your work. You must critically evaluate and modify any AI-generated content you use.
		FULL AI	You may use AI extensively throughout your work either as you wish, or as specifically directed in your assessment. Focus on directing AI to achieve your goals while demonstrating your critical thinking.
		AI EXPLORATION	You should use AI creatively to solve the task.
AI Policy	Unauthorised and undisclosed use of Artificial Intelligence may result in a finding of Academic Misconduct and an academic sanction		

	under Regulation 11. Guidance for WMG student on the use of Artificial Intelligence can be found here.
Module Learning Outcomes (numbered)	1. Synthesize the theories and concepts of software engineering and object-oriented programming, along with fundamental algorithms and data structures in Python. 2. Develop effective and independent software solutions in the Python programming language at an intermediate level. 3. Develop a practical understanding of SQL language and databases and implement computer programs to store and retrieve data from a database for a practical application. 4. Collaboratively develop and present software applications/solutions using Python and databases 5. Critically evaluate software development lifecycle practices and design conceptual and practical solutions
Learning Outcomes assessed in this assessment (numbered)	1, 2 and 3
Learning Outcomes	Please remember that you must meet all Learning Outcomes to achieve a pass mark.
Late Submission Policy	If work is submitted late, penalties will be applied at the rate of **5 marks per University working day** after the due date, up to a **maximum of 10 working days** late. After this period, the mark for the work will be reduced to 0 (which is the maximum penalty). "Late" means **after the submission deadline time as well as the date** – work submitted after the given time even on the same day is counted as 1 day late.
Resit Policy	If you fail this module and/or component, the University allows students to remedy failure (within certain limits). Decisions to authorise resits are made by Exam Boards. These will be issued at specific times of the year, depending on your programme of study. More information can be found from your programme office if you are concerned. If this is **already a resit** attempt, this means you will not be eligible for an additional attempt. The University allows as standard a maximum of two attempts on any assessment (i.e. only one resit). Students can only have a third attempt under exceptional circumstances via a Mitigating Circumstances Panel decision.
Retention of Drafts & Records	Please ensure that you retain any drafts of your work, associated notes, records of research should they be required by the marker or moderator. You do not need to submit these, but they should be retained until after the end of your registration.

Where to get help:

1. Talk to your module tutor if you do not understand the question or are unsure as to exactly what is required.
2. There are also numerous online courses provided by the University library to help in academic referencing, writing, avoiding plagiarism and a number of other useful resources. https://warwick.ac.uk/services/library/students/your-library-online/
3. If you have a problem with your wellbeing, it is important that you contact your personal tutor or wellbeing support services https://warwick.ac.uk/services/wss