COURSEWORK SPECIFICATION
COMM108 – Data Systems
Module Leader: **Dr Tianjin Huang**
Academic Year: 2025/26

Title: **Coursework**

Submission deadline: **20 November 2025**

This assessment contributes **30%** of the total module mark and assesses the following **intended learning outcomes**:

- Understand how to construct a database in which to store data for a given application.
- Understand how to design applications that uses an API to access and modify data stored in a database.
- Document and defend the design of a database for a given application.
- Contrast between different types of database tools.
- Design an appropriate data storage scheme for a project within a chosen problem domain.

Plagiarism

This is an individual assessment.

Plagiarism is interpreted by the university as the act of presenting the work of others as one's own work, without acknowledgement. It is considered academically fraudulent and an offence against university discipline. Your attention is drawn to the university's regulations on plagiarism.

Generative AI

This assessment has been categorised as **AI-Minimal** where you may use AI tools for checking spelling and grammar mistakes only. You can find further information in the university's policies around using AI in assessed work.

Instructions

In the UK, agriculture is a significant contributor to the economy, but it also plays a role in the country's energy consumption and environmental impact. As part of a broader commitment to sustainability and reducing carbon emissions, farmers are increasingly adopting innovative practices aimed at enhancing energy efficiency and minimizing environmental harm. This coursework allows students to explore these initiatives through the lens of a MySQL database project.

You are a data analyst working for a consultancy that supports agricultural businesses in their sustainability efforts. Your task is to develop a MySQL database that captures various sustainability initiatives undertaken by farms across the UK, focusing specifically on energy usage and its impact on crop production and environmental health.

You have been provided with some data that describes:

- **Farm ID and Location:** Identifiers and geographical details of the farms.
- **Crop Details:** Information on various crops grown, including crop IDs and names.
- **Soil Health Metrics:** Data on pH levels, nitrogen, phosphorus, and potassium levels.
- **Resource Usage:** Types and quantities of resources applied to the crops, including energy usage.
- **Sustainability Initiatives:** Descriptions of initiatives being implemented, their expected impacts, and environmental scores.

You must undertake the following tasks:

1. Create a normalised design for a relational database. You must indicate the associations between entities and the fields each entity contains. Alongside the **entity relationship diagram** you must provide a rationale for the design.
2. Implement SQL that creates a MySQL database implementing the tables in the ERD and insert the test data provided in the sample file.
3. Design a RESTful API to access the database you've implemented. You must describe the HTTP access method each API route will use, and outline the parameters each will take. Note: You do not need to provide code for this part of the assessment.
4. Provide an alternative design for the database that uses a document-based database model. You must provide a balanced reflection of the pros and cons behind the relational and document-based approaches you could take for this specific application.

You must submit one .zip file containing:
- A **four page report** describing your work.
- A .sql file that creates and populates the database.

Marking criteria

Your work will be assessed according to the following criteria:

Database Design (25%)			
Fail (0-49): The database design is incomplete, lacking necessary tables or relationships. No understanding of normalization is demonstrated.	**Pass (50-59):** Basic database structure is established with some relevant tables and fields. Some normalization attempts are evident, but relationships may not be fully accurate.	**Merit (60-69):** A well-structured database design is presented with relevant tables and relationships. Normalization is applied adequately, demonstrating an understanding of the data structure.	**Distinction (70+):** Exceptional database design with clear, logical relationships. All relevant tables are included, and normalization is thoroughly executed, showing deep understanding.
Database implementation (25%)			
Fail (0-49): The database is poorly implemented or does not function. Significant errors prevent the use of the database.	**Pass (50-59):** Basic implementation is successful, but there are some functional issues or data integrity problems. Basic SQL queries can be run.	**Merit (60-69):** Good implementation with minor issues. SQL queries are functional, and data is entered correctly. Some optimization may be lacking.	**Distinction (70+):** Excellent implementation with no issues. All SQL queries work efficiently, and the database performs well under load. Data integrity is maintained.
RESTful API Design (20%)			
Fail (0-49): No RESTful API design is presented, or the design is poorly conceived with unclear endpoints and HTTP methods.	**Pass (50-59):** A basic RESTful API design is created, but endpoints may be poorly defined, and there is minimal consideration of appropriate parameters and HTTP methods.	**Distinction (70+):** An exceptional RESTful API design is provided with clear, logically organized endpoints. Each endpoint includes well-defined parameters and appropriate HTTP methods. The design demonstrates a comprehensive understanding of API architecture and user needs.	**Distinction (70+):** An exceptional RESTful API design is provided with clear, logically organized endpoints. Each endpoint includes well-defined parameters and appropriate HTTP methods. The design demonstrates a comprehensive understanding of API architecture and user needs.
Document-based design alternative (30%)			
Fail (0-49): No alternative design is presented, or the comparison between relational and document-based databases is superficial and lacks understanding. Little to no insights into the strengths and weaknesses of each approach are provided.	**Pass (50-59):** A basic alternative design using a document-based database is provided, but the comparison with the relational database is minimal. Some strengths and weaknesses are mentioned, but insights are limited.	**Merit (60-69):** A thoughtful alternative design using a document-based database is presented. The comparison with the relational database is clear, discussing several strengths and weaknesses, including aspects such as data structure flexibility and scalability.	**Distinction (70+):** A comprehensive and well-structured alternative design using a document-based database is provided. The comparison with the relational database is insightful, thoroughly analyzing the pros and cons of each approach, including considerations of scalability, performance, data integrity, and use cases, demonstrating deep understanding and critical thinking.