Leeds Trinity University | School of Computing and Creative Industries

ASSESSMENT 1 BRIEF

MODULE CODE	COM7033
MODULE TITLE	Secure Software Development
MODULE LEADER	Xin Lu
ASSESSMENT TITLE	Software Artefact
WEIGHTING	70%

ASSESSMENT LEARNING OUTCOMES

Upon successful completion of this assessment, you will be able to:

1. Demonstrate an understanding of secure programming concepts and techniques.
2. Apply programming skills to manipulate and analyse data using popular libraries and frameworks.
3. Demonstrate an understanding of the importance of developing software in an ethical, secure, and professional manner.
4. Develop technical software solutions for complex problems.

INSTRUCTIONS

According to the World Health Organization (WHO), stroke is the second leading cause of death globally, posing a significant challenge to public health systems. In this assessment, you are required to design and develop a secure web-based application for a local hospital to manage a Stroke Prediction Dataset. The purpose of the system is to support doctors and healthcare professionals in recording, managing, and analysing patient data, covering demographics, medical history, and lifestyle factors, to predict the likelihood of a stroke and assist in preventive healthcare.

Dataset

You will use the publicly available **Stroke Prediction Dataset** from Kaggle:
https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset/data

Alternatively, the dataset can be **downloaded directly from the Moodle → Assessment folder** provided for this module.

Dataset Attributes are listed below:

Attribute	Description
id	Unique identifier
gender	“Male”, “Female”, or “Other”
age	Age of the patient
hypertension	0 = No hypertension, 1 = Has hypertension
ever_married	“No” or “Yes”
work_type	“Children”, “Govt_job”, “Never_worked”, “Private”, or “Self-employed”
Residence_type	“Rural” or “Urban”
avg_glucose_level	Average glucose level in the blood
bmi	Body Mass Index
smoking_status	“Formerly smoked”, “Never smoked”, “Smokes”, or “Unknown”
stroke	1 = Had a stroke, 0 = No stroke

Note: "Unknown" in *smoking_status* indicates unavailable information for that patient.

Assessment Tasks
You must build a secure Python Flask web application that allows users to store, manage, and retrieve patient information securely from the dataset. Your application should demonstrate both technical proficiency and secure software development practices.

Core Requirements

1. **Web Application Development**
   - Develop a fully functioning Flask web server with an intuitive and user-friendly interface.
   - Implement CRUD (Create, Read, Update, Delete) functionalities for managing patient data.
2. **Secure Data Management**
   - Support one or more database systems (e.g., **SQLite** and **MongoDB**) for secure data storage and retrieval.
   - Separate databases may be used to improve data management and security (e.g., SQLite for user authentication data, MongoDB for patient records).
3. **Implementation of Secure Programming Practices**
   - Apply **data encryption** (e.g., password hashing or encryption for registration data) to protect sensitive information.
   - Use **input validation and sanitisation** techniques to prevent vulnerabilities such as SQL injection, XSS, or CSRF.
   - Adopt secure session handling and proper error logging mechanisms.
4. **Professional and Ethical Development**
   - Demonstrate awareness of ethical considerations in handling sensitive healthcare data.
   - Follow **secure coding standards** and **professional programming conventions** throughout your implementation.
5. **Testing and Version Control**
   - Develop **unit tests** to ensure functional correctness and system reliability.
   - Utilise **GitHub** for version control, maintaining clear documentation of commits and development progress to evidence professional software engineering practice.

SUBMISSION DETAILS

RELEASE DATE	21 October 2025
SUBMISSION DATE	28 November 2025
DELIVERABLES	Upload your project code to the **private GitHub repository** created in the module's GitHub Classroom. You will find your personal repository link on Moodle. https://github.com/CS-LTU/com7033-assignment-XXXX Replace XXXX with your GitHub username. Once created, the repository is ready for your submission. Ensure that: ○ All source code, documentation, and related files are pushed to the repository before the submission deadline. ○ Your commit history clearly reflects development progress and version control practices.
SUBMISSION DETAILS	Submit your assignment by uploading it to Moodle before midday on the submission date. This deadline will be automatically and strictly enforced. If your submission is late, your grade may be affected. If you have any issues submitting your work, you must email the assessment team and copy in the module leader before the assessment due time. Do not leave your submission until the last minute to avoid any penalties due to problems with the submission portal. Assessment Team: assessment@leedstrinity.ac.uk Module Leader: x.lu@leedstrinity.ac.uk

	We may ask for a demonstration of your work following the submission. If needed, this will be communicated to you individually via email. Please check your emails regularly.

Your feedback / feed forward and mark for this assessment will be provided within 15 working days.

MARKING CRITERIA

Marks are awarded based on the following criteria. Within each part, aim to complete the work for each section before moving on to the next. The following banded marking scheme is used:

Exceptional Distinction	100/95/92 88/85/82		*Pass* *Bare Fail*	58/55/52 48/45/42
Distinction	78/75/72		*Fail*	38/35/32
Merit	68/65/62			

If you have completed all the preparatory exercises and attended your classes, the estimated additional time required to PASS this assessment is approximately 70 hours.

To obtain a PASS mark (50%), you must have:	○ Developed a basic Flask web application with a functional, simple user interface. ○ Used a single database (either SQLite or MongoDB) to store user data. ○ Implemented at least one basic security feature, such as simple input validation or basic password encryption. ○ Used GitHub for version control with at least one commit demonstrating project setup or initial development.
To obtain a MERIT mark (60%), you must have (in addition to the above):	○ Developed a fully functional web application with an enhanced and user-friendly interface. ○ Implemented multiple databases e.g. SQLite for user authentication data and MongoDB for patient records. ○ Demonstrated the ability to add, update, and delete records securely in both databases. ○ Implemented two distinct security features, such as input validation and password hashing. ○ Maintained at least four meaningful GitHub commits, each with clear, descriptive messages. ○ Partially commented the code and implemented at least one unit test to verify functionality.
To obtain a DISCTINCTION mark (70%), you must have (in addition to the above):	○ Developed a fully functional, professionally designed web application with a customised and polished interface. ○ Utilised multiple interconnected databases, enabling secure data management and efficient data retrieval. ○ Implemented more than two secure programming techniques, showing a high level of understanding of security best practices (e.g., encryption, input sanitisation, secure session handling, CSRF protection). ○ Maintained at least eight GitHub commits with detailed, meaningful messages illustrating ongoing development progress. ○ Provided comprehensive code comments and implemented multiple unit tests across different application features. ○ Produced a clear and well-structured README or user guide describing system functionality and installation instructions.

To obtain an EXCEPTIONAL DISTINCTION mark (80%), you must have (in addition to the above):	○ Produced highly efficient, modular, and scalable code, following professional software engineering and secure coding standards. ○ Integrated third-party APIs, frameworks, or libraries to extend functionality or enhance system security (e.g., authentication, data encryption, or external data services). ○ Delivered comprehensive documentation, including detailed usage instructions, API references, and design rationale. ○ Applied comprehensive testing coverage, including unit, integration, and end-to-end tests, demonstrating software reliability and robustness. ○ Maintained an active and well-documented GitHub repository with a clear branching strategy, frequent commits, and pull requests that reflect collaborative and iterative development practices.

USE OF GENERATIVE ARTIFICIAL INTELLIGENCE IN THIS MODULE

You may use generative AI such as Microsoft Copilot to assist you in the process of undertaking the assessment in the following ways: brainstorming, research, planning, feedback, editing.
All use of generative AI must be explicitly acknowledged, and any artificially generated content (e.g. images) explicitly labelled, with the source of the AI tool referenced using current APA referencing conventions You can find further guidance on the library website on their AI webpage).
In submitting your assignment, you agree to disclose the extent to which you have used generative AI in preparing this work and include evidence of your AI use in your appendices (e.g. dated screen shots of your use of this tool or copy and paste your AI chat into Word).
Failure to disclose your generative AI use may result in a zero for your assignment and a referral for academic misconduct (see the Student Academic Misconduct Policy under Essential Info in the MyLTU app).

Include one of the following statements on your assignments.
Either:

This assignment used generative AI in the following ways for the purposes of completing the assignment (choose 1 to 5 of the following): brainstorming, research, planning, feedback, editing.

Or:

This assignment did not use generative AI for the purposes of completing the assignment.

ACADEMIC MISCONDUCT

Academic Misconduct includes all forms of academic dishonesty, whether intentional or accidental, that compromise the integrity of the University's assessment processes. It is essential that you review our Student Academic Misconduct Policy to understand the guidelines and the serious consequences that may arise if they are not followed.

HELP AND SUPPORT

- Please use the module handbook and the Computer Science Community Teams site as a source of information. Do try and find the answer out yourself before reaching out for help.
- Support will be provided via Microsoft Teams and email during office working hours. You can also ask questions during your timetabled sessions. You may request a one-to-one meeting with a tutor during their published office hours.
- The Student Support team are available seven days a week to support you in all aspects of student life. This could be for support relating to your course, your accommodation or for more general advice such as relationships or your wellbeing. Log in to the LTU app to access support services.

- The full set of university guidelines on assessments, deadlines, and extensions is available on the LTU app, please familiarise yourself with the documentation.