# 

|  |
| --- |
| **COURSEWORK ASSESSMENT SPECIFICATION** |

|  |  |
| --- | --- |
| **Module Title:** | Principles of Software Engineering |
| **Module Number:** | LD7092 |
| **Module Tutor Name(s):** | Ning Tse, Zainab Mutlaq Ibrahim |
| **Academic Year:** | 2022-23 |
| **% Weighting  (to overall module):** | 100% |
| **Coursework Title:** | Practical Solution Prototype & Critical Discussion |
| **Average Study Time Required by Student:** | 60-70 hours |

**Dates and Mechanisms for Assessment Submission and Feedback**

|  |
| --- |
| **Date of Handout to Students:**  During first week of teaching. |
| **Mechanism for Handout to Students:**  Via Blackboard; briefing in face-to-face session**.** |
| **Date and Time of Submission by Student:**  Submitted on **29th Aug 2023** (no later than 16:00) |
| **Mechanism for Submission of Work by Student:**  The assignment must be submitted via the Turnitin submission point on Blackboard site. |
| **Date by which Work, Feedback and Marks will be returned to Students:**  Within 30 working days after the submission date. |
| **Mechanism for return of assignment work, feedback and marks to students:**  Formal feedback will be made available via Blackboard following completion of all reviews and internal moderation of results. |

## Learning Outcomes tested in this assessment

This assignment will assess the following learning outcomes:

1. Demonstrate critical knowledge and understanding of appropriate software engineering theories, technologies and methodologies.
2. Ability to apply software engineering knowledge to a complex business challenge or project.
3. Critically reflect on professional, legal, social, security and ethical issues related to the design, development and use of software in an organisational context.
4. Ability to critically evaluate your Software Engineering skills.
5. Demonstrate problem solving skills in complex situations in the context of the design, development and use of software solution.

## General Information

This assignment constitutes 100% towards the final mark for this module. Any queries relating to this assignment should be directed to module leader, ning.tse@northumbria.ac.uk.

## Type of the submission required

This is an INDIVIDUAL piece of work divided into two tasks as follows

|  |  |
| --- | --- |
| 1. Practical Solution Prototype (50%) | Design and development of a program prototype solution using OOP language such as Java on the given case study. This activity assesses module learning outcomes 1, 2 & 5. (2000 words) |
| 2. Critical Discussion (50%) | Critical discussion of some issues relating to Software engineering. This activity assesses module learning outcomes 3 & 4. (2000 words) |

## Case Study: Good Tasty Restaurant Management System (GTRMS)

Good Tasty Restaurant (GTR) is a traditional Italian restaurant based in London with a team of high-quality chefs who prepare delicious Italian cuisines for the patrons. In recent time, GTR owner decided to develop a new information system to manage the restaurant and serve their patrons in efficient way.

You are required to design and develop a user-friendly, desktop application to assist GTR staff including chefs, waiters and managers in managing the restaurant inventory (tables, chairs, dishes, glasses, etc.), food and beverages ingredient, and serving Italian cuisines for patrons using an advanced Restaurant Management System.

The new system has to be designed in an appropriate way to enable the restaurant staff and patrons to access the system and perform different activities such as: the chefs to manage the F&B ingredient, make sure they are available, and prepare the food orders for patrons; the waiters to take the orders from patrons and serve them; the manager to record all orders, prepare the bill and manage users including chefs and waiters records.

The GTR system should able to manage F&B ingredient records, automatic send reminders to chefs for purchasing ingredients from the suppliers, for manager to analysis the sales records and human resources and much more.

The system should allow registered users to use it only after login. Key features of the GTR system include:

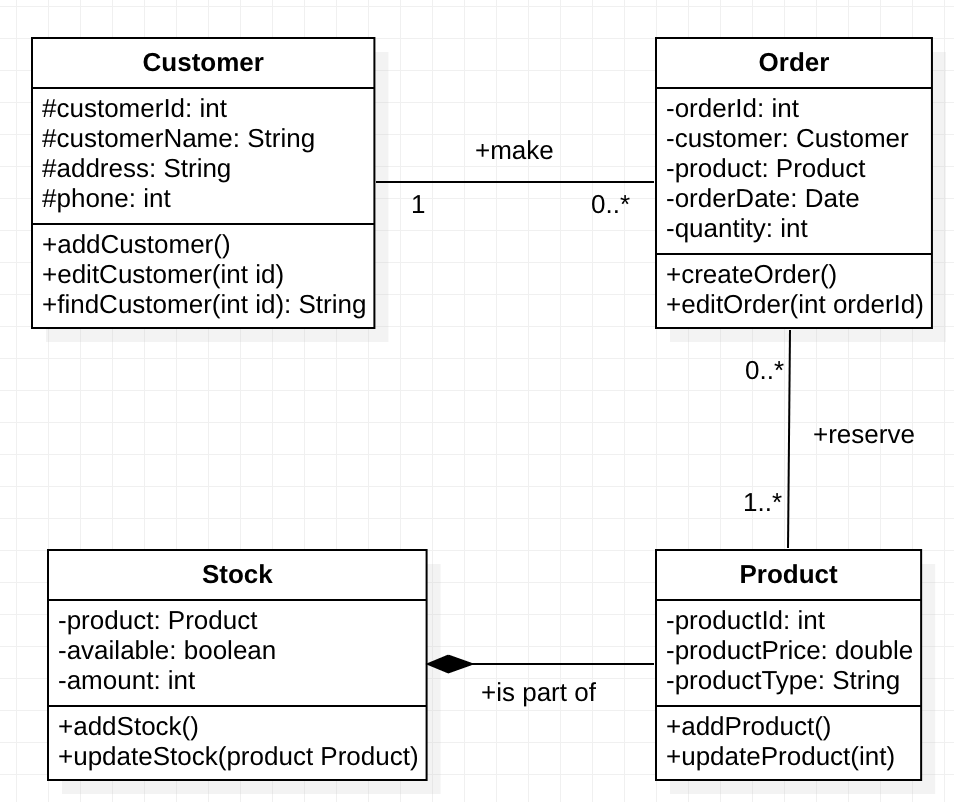
* User friendly GUI based system for staff and patrons to use in the restaurant
* Menu generation, food order preparation, bill & receipt printing
* Send reminder messages to chefs to order ingredients
* Inventory management
* Staff record management
* Sales records generation
* Any other relevant services

**Task 1:** **Design and Development** (50%, Learning Outcome 1 & 2 & 5)

Using the provided case study, develop a solution from a system analyst’s perspective that includes the following:

* Produce a **use case diagram** to model the functional requirements from the scenario narrative.
  + Showing actors and the use cases they perform.
  + Use inheritance where necessary of both use cases and actors.
  + Use of <<includes>> and <<extends>> where necessary.
  + Use house style model to document one primary use case.
* Produce a **class diagram** to model the system structure you propose for your potential solution. Decouple as much as possible with Boundary, Control and Entity classes. This should include:
  + Detailed classes with all key variables/attributes and datatypes with access modifiers (private, public or protected) clearly shown in adherence to encapsulation principal of OOP.
  + Procedures or functions with (parameters and their datatypes shown) as well as return type for the class functions. (see diagram example below)
  + Use of class relationships (e.g. dependency association, aggregation, composition and inheritance).
  + An example of a basic class diagram for product ordering system is shown in Figure 1.
* Produce a **sequence diagram** to model functional requirements from the scenario narrative for one key primary use case.
  + Showing actors and interacting objects.
  + Messages (Methods) calls between objects with parameters and return types were necessary.
  + Use of alternatives and loops where necessary.
  + An example of a detailed sequence diagram is shown in Figure 2.
* Develop the **proposed solution** using OOP language (Java with NetBeans IDE)
  + Produce an appropriate test plan with test cases.
  + Document your solution with relevant screenshots and snippets of codes.

NOTE: The solution is only going to be a prototype of the software architect level, but a skeletal code should be produced. You do not need to write advanced level functionality into the methods or permanent data storage, but you must build and design the Forms (JFrames and JPanels) and link them to each other with relevant GUI components such as buttons, text fields, labels etc. The program should be executable (i.e. no compile time errors that prevent it from being executed).



*Figure 1, a basic class diagram for product ordering system*

Diagram

Description automatically generated

*Figure 2, an example of detailed sequence diagram*

**Task 2: Reflection and Critical Evaluation** (50%, Learning Outcome 3 & 4)

You are required to write a report to address the following topics:

* Using software engineering literature, critically evaluate the Software Development Life Cycle (SDLC) methodology you have chosen for your solution development with relevant examples from your work (clarify how you applied it in the various SDLC phases).
* Critically reflect on professional, legal, social, security and ethical issues related to the design and development of your solution.

This is meant to be a critical discussion and not just a description. Your discussion should not be limited to your practical system (the prototype), but a discussion of the broader issues involved in the design of your solution such as the functional requirements, use case documentation, the main classes involved in the class diagram and sequence diagram with justification of any important design decisions.

## Format and Submission Requirements

* Submission will be via Turnitin on Blackboard; please see the front cover for the submission date. You can submit your report for originality checking via the ‘Draft Submission Point’. However, you must ensure the final version is correctly submitted to the ‘Final Submission Point’ before the deadline.
* Combine task 1 & 2 into one single MS Word document. Your discussion for the practical solution commentary should contain 2000 words and for the reflection and critical evaluation 2000 words.
* Add a cover page with your name, student ID, tutor’s name and word counts for each task.
* Include table of contents, page numbers, captions for all figures and tables. Layout should make reasonable use of margins, numbered headings, 1.2~1.3 line spacing, Times New Roman font style and 12pt font size.
* Referencing should be in the Harvard or APA style. If you require guidance on citation, please use the booklet ‘Cite Them Right’ available from the City Campus library. The online Northumbria Library is also a good source of references at this academic level as the Library subscribes to the leading information sources in Computing. Since Computer Science is highly dynamic, recent references to conference and journal papers from the last five years are preferred.
* Name your report with tutor’s initial, programme code, your student ID & first name: <tutor initial>\_<programme code>\_<student ID><First name>.docx. Eg, NT\_CT\_ w21012345John.docx for a Computing & Technology group 2 student attending Ning’s class.
* Project directory should also be submitted in one single zip file with same filename format, ie <tutor initial>\_<programme code>\_<student ID><First name>.zip. Simply submitting individual .java files will receive a mark of Zero for the program section. Use meaningful naming conventions for variables, methods and classes.

## Assessment Criteria/Marking Scheme:

|  |  |
| --- | --- |
| **Description** | **Marks** |
| **Task 1: Practical Solution (50%) (suggested word limit for this task is 2,000 words)** | |
| Use case modelling   * List of functional requirements * *Logical design showing* actors and associations including inheritance (generalisation/specialisation) of actors and use cases as appropriate. * Comprehensive use case documentation using house style. | 10 |
| Class diagram:   * *Logical* design *showing classes and members (*functions and procedures with parameters and return types) * Efficient design using object-oriented features such as encapsulation, abstraction, inheritance, polymorphism and composition/aggregation where applicable. * Use of design patterns e.g. MVC model. * Documentation of main classes and associations. | 10 |
| Sequence diagram:   * *A logical design showing a*ll objects involved in the use case scenario. * Messages/Method calls that reflects class diagram structure and communications. * Parameters and return types shown. * Control structure using alternatives and loops. | 10 |
| Program & Test Plan:   * A working prototype which reflects the class diagram designs. Use of coding standards, comments. * Use of OOP principles (e.g. abstraction, inheritance, encapsulation, polymorphism where applicable). * Test plan with appropriate test cases. * Documentation of the prototype is essential. | 20 |
| **Task 2: Reflection & Critical Discussion (50%) (suggested word limit for this task is 2,000 words)** | |
| Critical evaluation of SDLC and justification of chosen methodology in context of your solution with relevant examples for each phase of SDLC. | 20 |
| Critically reflect on professional, ethical and security issues related to the design, development and use of the software in the given project context. | 20 |
| Citations, references and report structure | 10 |
| **Total** | **100** |

**Formative Feedback:** There will be an opportunity for formative feedback during the semester. You are advised to start working on this assignment as early as possible so that you can seek clarification from the module tutor regarding any questions you might have during the semester. Note that tutors will not predict your grade, and you should not take the lack of comment on any aspect of your work as indicating that it is correct. You should make every effort to take advantage of formative feedback as tutors will not comment on draft work at other times. Remember that you will get more useful feedback from us by asking specific questions than just presenting us with your documentation and asking, ‘Is this right?’

**Academic Integrity Statement**: You must adhere to the university regulations on academic conduct. Formal inquiry proceedings will be instigated if there is any suspicion of plagiarism or any other form of misconduct in your work. Refer to the University’s Assessment Regulations for Northumbria Awards if you are unclear as to the meaning of these terms. The latest copy is available on the University website [here](https://www.northumbria.ac.uk/about-us/university-services/academic-registry/quality-and-teaching-excellence/assessment/guidance-for-students/).

## Penalties for Exceeding Word Limits: The actual word count is to be declared on the front of the assessment submission.

|  |  |
| --- | --- |
| **Under the word limit** | No Penalty: In not making use of the full word count, students may have self-penalised their work. If students have been able to achieve the requirements of the assessment using fewer words than allocated, they will not be penalised. |
| **Up to 10% over word limit** | No Penalty: Situation flagged by tutor in feedback but over-run is tolerated and no deduction is made from the final mark. |
| **More than 10% over the word limit** | The marker will stop reading when they judge that the word count exceeds the recommended word count by more than 10% i.e. for a 4000 word essay, the marker will read only the first 4400 words and will indicate on the text where they stop reading.  The content following this point will not be read and a mark will be awarded only for the content up to this point. |

**Late Submission Policy**: For coursework submitted up to 1 calendar day (24 hours) after the published hand-in deadline without approval, 10% of the total marks available for the assessment (i.e.100%) shall be deducted from the assessment mark. Penalties will be applied as defined in the University Policy on the late submission work. The latest copy is available on the University website [here](https://www.northumbria.ac.uk/about-us/university-services/academic-registry/quality-and-teaching-excellence/assessment/guidance-for-students/).

**For clarity:** 1 calendar day counts from 16:01 of 28th to 16:00 of 29th, December 2022. A late piece of work that would have scored 65%, 55% or 45% had it been handed in on time will be awarded 55%, 45% or 35% respectively as 10% of the total available marks will have been deducted.

**Failure to Submit:** The University requires all students to submit assessed coursework by the deadline stated in the assessment brief.  Where coursework is submitted without approval after the published hand-in deadline, penalties will be applied as defined in the University Policy on the Late Submission of Work.

## Module Specific Assessment Criteria

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Criteria** | **Scales** | | | | | | |
| **0-39% Standard Not Met** | **40-49% Standard Not Met** | **50-59% Meets Standard 1** | **60-69% Meets Standard 2** | **70-79% Standard 3** | **80-89% Exceeds Standard 2** | **90-100% Exceeds Standard 3** |
| **Use case model**  **[10%]** | Student is not able to demonstrate logical design. Actors or use cases are not available. | Insufficient;  Student is not able to demonstrate logical design.  Actors, use cases and associations are less than the expected minimum. | Student is able to demonstrate logical design.  Key actors, use cases and associations are modelled in a way that meets the fundamental requirements.  No house style documentation is presented. | Student is able to demonstrate logical design.  Actors, use cases and associations includes all primary actors and functions.  In addition to the use of inheritance and good design of use case documentation using house style. | Logical design meets standard 2 plus very good use of inheritance, includes and extends. | Logical design meets standard 3 plus excellent use of inheritance, includes and extends.  Clearly exceeds taught material. | Student is able to demonstrate outstanding logical design.  All actors, use cases and associations (primary use cases, some secondary use cases).  In addition to excellent use of inheritance, includes and extends with no errors or mistakes.  Clearly exceeds taught material. |
| **Outcome** | **[0-3]** | **[4]** | **[5]** | **[6]** | **[7]** | **[8]** | **[9-10]** |
| **Class Diagram**  **[10%]** | Student is not able to demonstrate logical design. Essential classes are not available. | Insufficient;  Student is not able to demonstrate logical design.  Essential classes are available however, no (or very bad) properties, operations and associations. | Student is able to demonstrate logical class diagram modelling.  Main classes, properties and associations are modelled in a way that meets main requirements. | Student meets standard 1 plus good use of multiplicities, inheritance, and encapsulation. | Student meets standard 2 plus very good use of multiplicities, inheritance, encapsulation, composition and aggregation.  Clear description on the classes. | Student meets standard 3 plus excellent use of multiplicities, inheritance, encapsulation, composition and aggregation.  Clearly exceeds taught material. | Student meets standard 3 plus innovative modelling demonstrated, with excellent use of multiplicities, inheritance, encapsulation, composition and aggregation with no errors or mistakes.  Clearly exceeds taught material. |
| **Outcome** | **[0-3]** | **[4]** | **[5]** | **[6]** | **[7]** | **[8]** | **[9-10]** |
| **Sequence Diagram**  **[10%]** | Key lifelines / objects missing, no or very bad messages communication | Insufficient Key lifelines / objects available, bad messages communication | Key lifeline / objects available, with adequate messages communication  Sufficient sequence diagrams to demonstrate different types of messages communication | Major lifeline / objects available, with satisfactory sequence of messages communication | Major lifeline / objects available, with very good sequence of messages communication plus good control structure | All lifeline / objects available, with excellent and detailed sequence of messages communication plus excellent control structure.  Clearly exceeds taught material. | All lifeline / objects available, with outstanding and detailed sequence of messages communication plus excellent control structure with no errors or mistakes.  Clearly exceeds taught material |
| **Outcome** | **[0-3]** | **[4]** | **[5]** | **[6]** | **[7]** | **[8]** | **[9-10]** |
| **Prototype Development**  **[20%]** | A design of a naïve program with basic views. | Basic program with Frame and panel and Basic documentation. | Adequate working program with multiple panels views and documentation. | A satisfactory program with multiple views and classes.  Use of some OOP implementation and documentation plus testing. | A good working program with multiple views and multiple classes using OOP implementation and documentation that reflects CD design.  Use of coding standards, comments and testing table. | A very good working program that goes beyond the good stage with proper implementation and documentation that reflects an appropriate CD design. | An excellent working program that goes beyond very good stage with proper. OOP coding, documentation and comprehensive testing table.  In addition to the use of advanced data structure techniques. |
| **Outcome** | **[0-5]** | **[6-9]** | **[10-11]** | **[12-14]** | **[15-16]** | **[17-18]** | **[19-20]** |
| **Critical evaluation of SDLC and justification of chosen methodology in context of your solution**  **[20%]** | Basic description of SDLC, without critical evaluation of SDLC Methodologies | Insufficient;  Basic description of SDLC, and Methodologies.  However, no critical evaluation or justification of chosen methodology in context of the given solution | Adequate description of SDLC, and Methodologies with satisfactory evaluation and justification of chosen methodology in context of the given solution.  Adequate citation and references to support ideas and arguments. | Satisfactory evaluation of SDLC, and Methodologies with critical reflection and justification of chosen methodology in context of the given solution.  Satisfactory citation and references to support ideas and arguments. | Critical evaluation of SDLC, and Methodologies with critical reflection and justification of chosen methodology in context of the given solution with proper and relevant examples.  Good citation and references to support ideas and arguments. | Critical evaluation of SDLC, and Methodologies with critical reflection and justification of chosen methodology based on a comparisons by other methodologies including advantages and disadvantages in context of the given solution with proper and relevant examples.  Very good citation and references to support ideas and arguments. Clearly exceeds taught material. | Outstanding reflection and critical evaluation of SDLC, and Methodologies with critical reflection and justification of chosen methodology based on a comparison with other methodologies including advantages and disadvantages in context of the given solution with proper and relevant examples.  Excellent citation and references to support ideas and arguments. Clearly exceeds taught material. |
| **Outcome** | **[0-5]** | **[6-9]** | **[10-11]** | **[12-14]** | **[15-16]** | **[17-18]** | **[19-20]** |
| **Professional, ethical and security issues [20%]** | Only basic description of professional, ethical and security issues. | Insufficient;  Adequate description of professional, ethical and security issues without relating it to the design, development and use of software in an organisational context | Satisfactory description of professional, ethical and security issues with relevant examples related to the system design, development and use in the given context.  Adequate citation and references to support ideas and arguments. | Good evaluation of professional, ethical and security issues with good examples related to the system design, development and use in the given context.  Satisfactory citation and references to support ideas and arguments. | Critical evaluation and reflection on professional, ethical and security issues with very good examples related to the system design, development and use in the given context.  Good citation and references to support ideas and arguments. | Critical evaluation and reflection on professional, ethical and security issues with excellent examples related to the system design, development and use in the given context.  Very good citation and references to support ideas and arguments.  Clearly exceeds taught material. | Critical evaluation and reflection on professional, ethical and security issues with excellent discussion and examples related to the system design, development and use in the given context.  Excellent citation and references to support ideas and arguments.  Clearly exceeds taught material. |
| **Outcome** | **[0-5]** | **[6-9]** | **[10-11]** | **[12-14]** | **[15-16]** | **[17-18]** | **[19-20]** |
| **Citations and references and report structure**  **[10%]** | No or very insufficient citations and references, with bad report structure | Insufficient citations and references, with inadequate report structure | Adequate citations, references and report structure | Satisfactory citations, references and report structure | Very Good citations, references and report structure | Excellent citations, references and report structure | Outstanding citations, references and report structure. |
| **Outcome** | **[0-3]** | **[4]** | **[5]** | **[6]** | **[7]** | **[8]** | **[9-10]** |
| **Overall Mark** | **[ / 100 ]** | | | | | | |
| **Further Comments:** | | | | | | | |