BFM1 — BFM1 TASK 1: OBJECT-ORIENTED APPLICATION DEVELOPMENT

SOFTWARE I – C# — C968
PRFA — BFM1

TASK OVERVIEW SUBMISSIONS EVALUATION REPORT

COMPETENCIES

4041.3.1 : Classes and Interfaces
The graduate designs software solutions with appropriate classes, objects, methods, and interfaces to achieve specific goals.

4041.3.2 : Object-Oriented Principles
The graduate implements object-oriented design principles (e.g., inheritance, encapsulation, and abstraction) in developing applications for ensuring the application's scalability.

4041.3.3 : Application Development
The graduate produces applications using high-level programming language constructs to meet business requirements.

4041.3.4 : Exception Handling
The graduate incorporates simple exception handling in application development for improving user experience and application stability.

4041.3.5 : User Interface Development
The graduate develops user interfaces to meet project requirements.

INTRODUCTION

Throughout your career in software design and development, you will be asked to create applications with various features and functionality based on business requirements. When a new system is developed, typically the process begins with a business analyst gathering and writing these business requirements, with the assistance of subject matter experts from the business. Then a system analyst works with several application team members and others to formulate a solution based on the requirements. As a developer, you would then create a design document from the solution and finally develop the system based on your design document.

For this assessment, you will create a C# application using the solution statements provided in the requirements section.

The skills you showcase in your completed application will be useful in responding to technical interview questions for future employment. This application may also be added to your portfolio to show to future employers.

Your submission should include a zip file with all the necessary code files to compile, support, and run your application.

Note: The preferred integrated development environment (IDE) for this assignment is Visual Studio. Refer to your course of study for instructions on how to install and use this application. If you choose to use another IDE, you must export your project into Visual Studio format for submission.

Your submission should include a zip file with all the necessary code files to compile, support, and run your application. The zip file submission must also keep the project file and folder structure intact for the Visual Studio IDE.

SCENARIO

You are working for a small manufacturing organization that has outgrown its current inventory system. They have been using a spreadsheet program to manually enter inventory additions, deletions, and other data from a paper-based system but would now like you to develop a more sophisticated inventory program.

They have provided you with a mock-up of the user interface to use in the design and development of the system (see the attached "GUI Mock-Up") and a class diagram to assist you in your work (see the attached "UML Class Diagram"). The organization also has specific business requirements that must be included as part of the application. A system analyst from your company created the solution statements outlined in the requirements section based on the manufacturing organization's business requirements. You will use these solution statements to develop your application.

REQUIREMENTS

Your submission must be your original work. No more than a combined total of 30% of the submission and no more than a 10% match to any one individual source can be directly quoted or closely paraphrased from sources, even if cited correctly. An originality report is provided when you submit your task that can be used as a guide.

You must use the rubric to direct the creation of your submission because it provides detailed criteria that will be used to evaluate your work. Each requirement below may be evaluated by more than one rubric aspect. The rubric aspect titles may contain hyperlinks to relevant portions of the course.

I. User Interface

Create a C# application with a graphical user interface (GUI) based on the attached "GUI Mock-Up." Write code to display each of the following forms in the GUI:

A. A main form, showing the following controls:
- buttons for "Add," "Modify," "Delete," "Search" for parts and products, and "Exit"
- lists for parts and products
- text boxes for searching for parts and products
- title labels for parts, products, and the application title

B. An add part form, showing the following controls:
- radio buttons for “In-House” and “Outsourced” parts
- buttons for “Save” and “Cancel”
- text boxes for ID, name, inventory level, price, max and min values, and company name or machine ID
- labels for ID, name, inventory level, price/cost, max and min values, the application title, and company name or machine ID

C. A modify part form, with fields that populate with data from an existing Part, showing the following controls:
- radio buttons for “In-House” and “Outsourced” parts
- buttons for “Save” and “Cancel”
- text boxes for ID, name, inventory level, price, max and min values, and company name or machine ID
- labels for ID, name, inventory level, price, max and min values, the application title, and company name or machine ID

D . An add product form, showing the following controls:
- buttons for “Save,” “Cancel,” “Add” part, and “Delete” part
- text boxes for ID, name, inventory level, price, and max and min values
- labels for ID, name, inventory level, price, max and min values, and the application
- a grid view for all parts
- a grid view for parts associated with the product
- a “Search” button and a text field with an associated list for displaying the results of the search

E. A modify product form, with fields that populate with data from an existing product, showing the following controls:
- buttons for “Save,” “Cancel,” “Add” part, and “Delete” part
- text boxes for ID, name, inventory level, price, and max and min values
- labels for ID, name, inventory level, price, max and min values, and the application “all candidate parts”
- a grid view for parts associated with the product
- a “Search” button and a text box with associated list for displaying the results of the search

II. Application

Now that you’ve created the GUI, write code to create the class structure provided in the attached “UML (unified modeling language) Class Diagram.” Enable each of the following capabilities in the application:

F. Using the attached “UML Class Diagram,” create appropriate classes and instance variables with the following criteria:
- five classes with the all associated properties
- variables are accessible/modifiable through properties

G. Add the following functionalities to the main form, using the methods provided in the attached “UML Class Diagram”:
- redirect the user to the “Add Part,” “Modify Part,” “Add Product,” or “Modify Product” forms
- delete a selected part or product from the grid view
- search for a part or product and display matching results

- exit the main form

H. Add the following functionalities to the part forms, using the methods provided in the attached "UML Class Diagram":
   1. "Add Part" form
      - select "In-House" or "Outsourced"
      - enter name, inventory level, price, max and min values, and company name or machine ID
      - save the data and then redirect to the main form
      - cancel or exit out of this form and go back to the main form
   2. "Modify Part" form
      - select "In-House" or "Outsourced"
      - modify or change data values
      - save modifications to the data and then redirect to the main form
      - cancel or exit out of this form and go back to the main form

I. Add the following functionalities to the product forms, using the methods provided in the attached "UML Class Diagram":
   1. "Add Product" form
      - enter name, inventory level, price, and max and min values
      - save the data and then redirect to the main form
      - associate one or more parts with a product
      - remove or disassociate a part from a product
      - cancel or exit out of this form and go back to the main form
   2. "Modify Product" form
      - modify or change data values
      - save modifications to the data and then redirect to the main form
      - associate one or more parts with a product
      - remove or disassociate a part from a product
      - cancel or exit out of this form and go back to the main form

J. Write code to address the following conditions with exception handling code:
   - Detect non-numeric values in textboxes that expect numeric values
   - Min should be less than Max; and Inv should be between those two values
   - Prevent the user from deleting a product that has a Part associated with it
   - Confirm "Delete" actions

File Restrictions

File name may contain only letters, numbers, spaces, and these symbols: ! - _ . * ' ( )
File size limit: 200 MB
File types allowed: doc, docx, rtf, xls, xlsx, ppt, pptx, odt, pdf, txt, qt, mov, mpg, avi, mp3, wav, mp4, wma, flv, asf, mpeg, wmv, m4v, svg, tif, tiff, jpeg, jpg, gif, png, zip, rar, tar, 7z

RUBRIC

A:MAIN FORM

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
No code is provided for a main form.	The main form does not include all required controls or does not present a professional appearance. The code contains errors or is incomplete.	The main form includes all required controls and presents a professional appearance. The code is complete and functions properly.

B:"ADD PART" FORM

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
No code is provided.	The "Add Part" form does not include all required controls or does not present a professional appearance. The code contains errors or is incomplete.	The "Add Part" form includes all required controls and presents a professional appearance. The code is complete and functions properly.

C:"MODIFY PART" FORM

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
No code is provided.	The "Modify Part" form does not include all required controls or does not present a professional appearance. The code contains errors or is incomplete.	The "Modify Part" form includes all required controls, and presents a professional appearance. The code is complete and functions properly.

D:"ADD PRODUCT" FORM

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
No code is provided.	The "Add Product" form does not include all required controls or does not present a professional appearance. The code contains errors or is incomplete.	The "Add Product" form includes all required controls and presents a professional appearance. The code is complete and functions properly.

E:"MODIFY PRODUCT" FORM

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
No code is provided.		The "Modify Product" form in-

	The "Modify Product" form does not include all required controls or does not present a professional appearance. The code contains errors or is incomplete.	cludes all required controls and presents a professional appearance. The code is complete and functions properly.

F:CLASS STRUCTURE

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
No code is provided.	The application does not include all required classes or includes incorrectly associated properties. The instance variables are accessible/modifiable outside getter and setter methods. The code is incomplete or does not function properly.	The application includes all required classes and correctly associated properties. The instance variables are accessible/modifiable through those properties. The code is complete and functions properly.

G:MAIN FORM FUNCTIONS

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
No code is provided.	The main form does not include all required functionalities or does not reflect the methods in the attached class diagram. The code contains errors or is incomplete.	The main form includes all required functionalities and reflects the methods in the attached class diagram. The code is complete and functions properly.

H1:"ADD PART" FORM FUNCTIONS

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
No code is provided.	The "Add Part" form does not include all required functionalities or does not reflect the methods in the attached class diagram. The code contains errors or is incomplete.	The "Add Part" form includes all required functionalities and reflects the methods in the attached class diagram. The code is complete and functions properly.

H2:"MODIFY PART" FORM FUNCTIONS

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
No code is provided.	The “Modify Part” form does not include all required functionalities or does not reflect the methods in the attached class diagram. The code contains errors or is incomplete.	The “Modify Part” form includes all required functionalities and reflects the methods in the attached class diagram. The code is complete and functions properly.

I1:“ADD PRODUCT” FORM FUNCTIONS

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
No code is provided.	The “Add Product” form does not include all required functionalities or does not reflect the methods in the attached class diagram. The code contains errors or is incomplete.	The “Add Product” form includes all required functionalities and reflects the methods in the attached class diagram. The code is complete and functions properly.

I2:“MODIFY PRODUCT” FORM FUNCTIONS

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
No code is provided.	The “Modify Product” form does not include all required functionalities or does not reflect the methods in the attached class diagram. The code contains errors or is incomplete.	The “Modify Product” form includes all required functionalities and reflects the methods in the attached class diagram. The code is complete and functions properly.

J:EXCEPTION HANDLING CODE

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
No code is provided.	The application does not include the 4 required exception handling code, or the language used in the custom error messages is not appropriate or distracts from the application. The code contains errors or is in-	The application includes the 4 required exception handling code and uses appropriate language in the custom error messages. The code is complete and functions properly.

complete.

PROFESSIONAL COMMUNICATION:

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
Content is unstructured, is disjointed, or contains pervasive errors in mechanics, usage, or grammar. Vocabulary or tone is unprofessional or distracts from the topic.	Content is poorly organized, is difficult to follow, or contains errors in mechanics, usage, or grammar that cause confusion. Terminology is misused or ineffective.	Content reflects attention to detail, is organized, and focuses on the main ideas as prescribed in the task or chosen by the candidate. Terminology is pertinent, is used correctly, and effectively conveys the intended meaning. Mechanics, usage, and grammar promote accurate interpretation and understanding.

SUPPORTING DOCUMENTS

GUI Mock Up.docx

UML Class Diagram.pdf