Heat maps

BFM2 — BFM2 TASK 1: OBJECT-ORIENTED APPLICATION DEVELOPMENT

SOFTWARE I – C# — C968

PRFA — BFM2

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, the process typically begins with a business analyst gathering and writing the business requirements with the assistance of subject matter experts from the business. A system analyst then 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.

Note: The preferred integrated development environment (IDE) for this assignment is Visual Studio. Refer to the course 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 a 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. The similarity report that 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.

*Tasks may **not** be submitted as cloud links, such as links to Google Docs, Google Slides, OneDrive, etc., unless specified in the task requirements. All other submissions must be file types that are uploaded and submitted as attachments (e.g., .docx, .pdf, .ppt).*

Create a C# application based on the attached "GUI Mock-Up" and the attached "UML Class Diagram" by doing the following:

I. User Interface

A. Create 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. Create 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 or cost, max and min values, the form title, and company name or machine ID

C. Create 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 form title, and company name or machine ID

D. Create 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 form title
- 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. Create 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 "all candidate parts" grid
- a grid view for parts associated with the product
- a "Search" button and a text box with an associated list for displaying the results of the search

II. Classes

F. Create and implement the appropriate classes, methods, and properties as specified in the attached "UML Class Diagram."

G. Add the following behaviors to the main form using the classes, methods, and properties implemented in part F:
- 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 behaviors to the part forms using the methods provided in the attached "UML Class Diagram":
1. Add the following behaviors to the "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. Add the following behaviors to the "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 behaviors to the product forms using the methods provided in the attached "UML Class Diagram":

1. Add the following behaviors to the "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. Add the following behaviors to the "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. Validate the user input to prevent the following **four** conditions:
- non-numeric values in textboxes that expect numeric values
- min from being greater than max; inventory from being outside the min and max bounds
- the user deleting a part that is associated with a product
- deleting a part or product without confirming it first

K. Demonstrate professional communication in the content and presentation of your submission.

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, csv, 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
A main form is not provided.	The main form does not include *all* required controls, or the form contains errors.	The main form includes *all* required controls, and the form functions properly.

B:"ADD PART" FORM

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
An "Add Part" form is not provided.	The "Add Part" form does not include *all* required controls, or the form contains errors.	The "Add Part" form includes *all* required controls and functions properly.

C:"MODIFY PART" FORM

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
A "Modify Part" form is not provided.	The "Modify Part" form does not include *all* required controls or does not include fields that populate with data from an existing part. Or the form contains errors.	The "Modify Part" form includes *all* required controls, has fields that populate with data from an existing part, and functions properly.

D:"ADD PRODUCT" FORM

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
An "Add Product" form is not provided.	The "Add Product" form does not include *all* required controls. Or the form contains errors.	The "Add Product" form includes *all* required controls and functions properly.

E:"MODIFY PRODUCT" FORM

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
A "Modify Product" form is not provided.	The "Modify Product" form does not include *all* required controls or does not include fields that populate with data from an existing product. Or the form contains errors.	The "Modify Product" form includes *all* required controls, includes fields that populate with data from an existing product, and functions properly.

F:CLASS STRUCTURE

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
No classes, methods, or properties are provided.		The application includes and implements *all* required classes,

	The application does not include *all* required classes, methods, or properties. Or the classes, methods, or properties are not implemented.	methods, and properties.

G:MAIN FORM BEHAVIORS

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
No behaviors are provided for the main form.	The main form does not include *all* required behaviors or does not use the classes, methods, and properties implemented in part F. Or the form contains errors.	The main form includes *all* required behaviors using the classes, methods, or properties implemented in part F. The form functions properly.

H1:"ADD PART" FORM BEHAVIORS

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
No behaviors are provided for the "Add Part" form.	The "Add Part" form does not include *all* required behaviors or does not reflect the methods in the attached class diagram. Or the form contains errors.	The "Add Part" form includes *all* required behaviors and reflects the methods in the attached class diagram. The form functions properly.

H2:"MODIFY PART" FORM BEHAVIORS

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
No behaviors are provided for the "Modify Part" form.	The "Modify Part" form does not include *all* required behaviors or does not reflect the methods in the attached class diagram. Or the form contains errors.	The "Modify Part" form includes *all* required behaviors and reflects the methods in the attached class diagram. The form functions properly.

I1:"ADD PRODUCT" FORM BEHAVIORS

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
No behaviors are provided for the "Add Product" form.	The "Add Product" form does not include *all* required behaviors or does not reflect the methods in the attached class	The "Add Product" form includes *all* required behaviors and reflects the methods in the attached class diagram. The form functions properly.

	diagram. Or the form contains errors.	

I2:"MODIFY PRODUCT" FORM BEHAVIORS

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
No behaviors are provided for the "Modify Product" form.	The "Modify Product" form does not include *all* required behaviors or does not reflect the methods in the attached class diagram. Or the form contains errors.	The "Modify Product" form includes *all* required behaviors and reflects the methods in the attached class diagram. The form functions properly.

J:VALIDATE USER INPUT

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
No validation is provided for user input.	The application does not provide validation of user input to prevent *all* 4 of the given conditions. The language used in the custom error messages is not appropriate for the corresponding conditions.	The application validates user input to prevent *all* 4 of the given conditions. The language used in the custom error messages is appropriate for the corresponding conditions.

K: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