School of Computing, Edinburgh Napier University

1. Module number	*SET08108*
2. Module title	*Software Development 2*
3. Module leader	*Neil Urquhart*
4. Tutor with responsibility for this Assessment Student's first point of contact	*Neil Urquhart* E: n.urquhart@napier.ac.uk T: 0131 455 2655
5. Assessment	*Practical*
6. Weighting	*80%*
7. Size and/or time limits for assessment	*OO design and development as specified. You should not spend more than 50 hours on this. A short (~5 minute) demonstration will be required prior to hand-in.*
8. Deadline of submission Your attention is drawn to the penalties for late submission	***By the 5th August 2016 you should have:*** • *Demonstrated your work in class* • *Created a working repository on BitBucket* • *Uploaded a PDF of your code listings and class diagram* • *Uploaded a .ZIP to Moodle containing your project*
9. Arrangements for submission	*Upload your files to Moodle.* *Contact Neil Urquhart to arrange a time to demonstrate your work. Please contact well before the August 5th deadline.*

10. Assessment Regulations All assessments are subject to the University Regulations**.**	*No exemptions*
11. The requirements for the assessment	*See attached*
12. Special instructions	The teaching team will make arrangements for demonstrations and publicise this to students during the lectures, via WebCT and via Email. Students must remain in contact with the teaching team.
13. Return of work and feedback	Instant, personalised oral feedback will be available at the demonstration.
14. Assessment criteria	See attached. Please note that checks for plagiarism will be made on electronic submissions. Students may be required to attend a further demonstration if there exists doubts as to the authorship of work.

Coursework resit

Please rework your submission for coursework 2, please ensure that you take into account the feedback given to you after your initial submission.

Coursework 2

Neil & Ben’s Pizzeria sells quality pizzas and pasta dishes, customers can eat in the restaurant or have their food delivered.

The Pizzeria has a number of dishes on its menu, each dish has the attributes shown below. Each customer is either a sit-in customer or a delivery customer (who has their meal delivered to them by a driver).

You must design a system (based on the supplied screen shots) that will manage the menu (allowing dishes to be added and deleted) and create bills for customers.

Menu items have the following attributes

Attribute	**Type**	**Validation**
Description	String	Not blank
Vegetarian	Boolean	True/false
Price	Int	In range 0 to 100000 (pence)

Sit-in Orders have the following attributes:

Attribute	**Type**	**Validation**
Table	int	1 – 10
Items	Dishes	

Delivery orders have the following attributes

Attribute	**Type**	**Validation**
Customer Name	String	Required
Delivery Address	String	Required
Items	Dishes	

Each sit-in order should be allocated to a member of staff (a server, for whom we record their name and a staff id). Each delivery order is allocated to a driver (for whom we record their name, staff id and car registration).

The main order window should allow a server to create a bill, they should do the following:

1. Select server from a drop down list
2. Select dishes ordered – could use a button for each dish or a list box
3. If the customer is sit-in enter a table number
4. If the customer is a delivery select a driver and enter a name and address
5. Press a Create Bill button
6. A bill will then be generated, which will show all of the dishes ordered and the total amount to pay. If the customer is a delivery a 15% delivery charge should be added. The bill should be displayed in a text box (or similar on the screen.)

A second window for use by the manager should allow the following operations (some of these may require further windows)

1. Add/Remove/Amend menu items
2. Add/Remove/Amend servers
3. Add/Remove/Amend drivers
4. List all sit in orders placed (server, table, amount paid)
5. List all delivery orders placed (server, driver, customer name, amount paid)
6. List all menu items and the quantities of each ordered
7. List all the orders taken by a specified server

Tasks

1. Create a class diagram showing your design (use the diagramming tool from the Architecture menu in Visual Studio, automatically generated diagrams **are not acceptable**). You do not need to include GUI classes (forms) at this stage.
2. Implement the classes identified in your diagram using C#
3. Add a GUI (using WPF) to allow the following
    a. Add a new order (sit-in)
    b. Add a new order (delivery)
    c. List all menu items
    d. Add/Remove/Amend menu items
    e. Add/Remove/Amend servers
    f. Add/Remove/Amend drivers
    g. List all sit in orders placed (server, table, amount paid)
    h. List all delivery orders placed (server, driver, customer name, amount paid)
    i. List all menu items and the quantities of each ordered
    j. List all the orders taken by a specified server

4. Create Unit tests for your Server and Menu classes. They should test the correctness of the methods and properties associated with these classes
5. Update your class diagram to show the classes added to implement the GUI

Optional Tasks

You may wish to pick one of the following optional tasks (which may be carried out in addition to the basic or advanced tasks). You will need to research how these tasks will be carried out.

- Store the data held by the system in a file so that it may be saved and reloaded between sessions
- Use a database to store the data held by the system

General Points

Class Diagrams

Class diagrams should show:

- Private properties
- Public properties (with get/set as appropriate)
- Public and private methods
- Relations between classes (e.g. 1:1, 1:M or inheritance)

Coding standards & Use of Bitbucket.

1. Your Bitbucket username must be your matriculation number
2. The repository used for this assessment must be called "assessment2"
3. You must grant Neil & Ben read access to your repository, which must be maintained until the module results have been published.

IF YOU DO NOT FOLLOW THE ABOVE STEPS YOU MAY NOT RECEIVE ANY MARKS FOR THE SOURCE CONTROL SECTION

4. Code should be committed and pushed on a regular basis with comments added to the commit.

5. All classes should have comments at the top to note
   i. Author name
   ii. Description of class purpose
   iii. Date last modified
6. Methods should all have a comment to describe their purpose
7. Properties should all have a comment to describe their purpose
8. All code should be indented as appropriate. EG

   …

```
for (int counter = 1; counter <= 1000; counter++)
{
        if (counter == 10)
                break;
        Console.WriteLine(counter);
}
```

   …

9. Use descriptive variable and method names (.e.g. no use of 'x' or 'y'!)
10. Code is written in a succinct fashion (i.e. no unnecessary code.)

Demonstration

When demonstrating you must have a copy of the demonstration sheet printed out, this will be filled in during the demonstration. You must also have printed copies of the class diagrams.

For the purposes of demonstration, enter the data given in the sample data in Appendix 1.

Submission

Please submit a printed copy of your class diagrams and all code that you have written to the School Office no later than the deadline shown on the cover sheet. You must include the code for your test harnesses and screenshots of the successful tests. Your design and your code will be subject to code/design review.

Please make sure that your BitBucket repository contains a valid Visual studio 2013 solution that may be downloaded and compiled. Please make sure that all of your code/projects/documentation is included. Finally please upload .ZIP archive of your repository into Moodle, these files may be accessed by the external examiners or module moderators at a future date.

Demonstration Sheet

Name___________________________ **Matric Number**_________________

Demonstrator ____________________ **Date/Time** ______________________

1. **Development tasks**

Item (note that where there is a choice the demonstrator will decide which item the student has to demonstrate)	**Mark 0,1,2** 0 – no attempt 1 – partial attempt 2 – fully working
Add a new sit-in order **or** Add a new delivery order	
Create a bill for a sit-in order **or** Create a bill for a take-out order	
Add a new menu item	
Add a new server	
List all orders allocated to a specified server	
List all sit-in orders **or** list all delivery orders	
List quantities sold for each menu item	

2. **User Interface**

Item	**Mark 0,2,4** 0 – difficult to use 2 – usable 4 – exemplary
Clarity and functionality of UI	

3. **Testing**

Item	**Mark 0,1,2** 0 – no attempt 1 – partial attempt 2 – fully working
Execute automated tests for Server	
Execute automated tests for Menu	

4. **Optional tasks**

Item	**Mark 0,2,4** 0 – no attempt 2 – partial attempt 4 – fully working
Use of database OR Serialisation (Delete as appropriate)	

Notes:

TOTAL ______ /24

Design/Code Review Sheet

Name__________________________ Matric Number_________________

All sections are marked out of 2 (0= no attempt,1=reasonable attempt,2= fully working).

Please note that in situations where substantial sections of the code are missing or incorrect as evidenced by the demonstration then marks in this section may be reduced or capped at the discretion of the module leader.

Item	**Mark (0,1,2)**
Class diagram: Identification of classes	
Class diagram: Use of inheritance and relations	
Class diagram: GUI classes separated from business classes	
Appropriate data types used for properties	
Properties declared as private and accessors used	
Code is adequately commented	
Code matches class diagram	

Source Control	**Mark (0,2,6)**
BitBucket repository contains source, with evidence of regular commits	0 – Not used 2 – Code in repository 6 – Evidence of regular commits

Total____/24

<u>Total marks</u>

From demo _______ /24

From review _______/24

Total _______ /48

Appendix 1 : Sample Data

Menu Items

Description	**Price**	**Vegetarian**
Tomato Pizza	£4.50	Y
Peperoni Pizza	£5.00	N
Spaghetti Bolognaise	£6.00	N
Tomato Pasta	£3.50	Y

Servers

Name	**ID**
Ben	1
Louise	2
Neil	3

Drivers

Name	**ID**	**Car Reg**
Jim	4	VSC 86
David	5	BFS 1L
Sian	6	CSG 773S