**University of Westminster**
Department of Computer Science

# 5BUIS005C, "BIS Design and Development" (2019/20)

| | |
|---|---|
| Module leader | Abarnah Kirupananda |
| Unit | Individual Coursework |
| Weighting: | 50% |
| Qualifying mark | 30% |
| Learning Outcomes Covered in this Assignment: | LO3: identify the typical functional components of Web applications;<br>LO4: create a prototype to show how specifications of a given set of requirements of a medium-scale business can be Implemented using appropriate programming language and tools;<br>LO7: apply a requirements-centred approach to derive modelling solutions applicable in component based implementation environments and implement excerpts from their architecture design models;<br>LO8: define a set of test cases to demonstrate how an implementation can be verified against the system specification; |
| Handed Out: | **Thursday 13 February 2019** |
| Due Date | **Monday 06 April 2020, 13:00pm** |
| Expected deliverables | Electronic file<br>Upload PHP files – University's server<br>Presentation 8-9-10 April 2020 (exact days/times will be confirmed) |
| Method of Submission: | Electronic submission via Blackboard.<br>You must submit a file that will contain the following:<br>• The address on University's server where all your files are uploaded<br>• One paragraph for each task that will provide the name of the file that is the answer to the task and any other information required by the task.<br>• An electronic copy of the produced software application. ALL OO PHP files, MySQL files or any other content used in the application must be included.<br>You should upload only ONE file. You are responsible to provide the correct address for your site at University's server. |
| Type of Feedback and Due Date: | • Verbal feedback during the tutorials<br>• Verbal feedback during presentation<br>• Written feedback and marks 15 working days (3 weeks) after the deadline.<br>**All marks will remain provisional until formally agreed by an Assessment Board.** |

| | |
|---|---|
| BCS Criteria meeting in this assignment | 2.1.3 Problem solving strategies<br>2.1.4 Analyse if/how a system meets current and future requirements<br>2.1.5 Deploy theory in design, implementation and evaluation of systems<br>2.2.1 Specify, design or construct computer-based systems<br>2.2.4 Deploy tools effectively<br>2.3.2 Development of general transferable skills<br>3.1.2 Methods, techniques and tools for information modeling, management and security |

**Assessment regulations**

Refer to section 4 of the "How you study" guide for undergraduate students for a clarification of how you are assessed, penalties and late submissions, what constitutes plagiarism etc.

**Penalty for Late Submission**

If you submit your coursework late but within 24 hours or one working day of the specified deadline, **10 marks** will be deducted from the final mark, as a penalty for late submission, except for work which obtains a mark in the range 40 – 49%, in which case the mark will be capped at the pass mark (40%). If you submit your coursework more than 24 hours or more than one working day after the specified deadline you will be given a mark of zero for the work in question unless a claim of Mitigating Circumstances has been submitted and accepted as valid.

It is recognised that on occasion, illness or a personal crisis can mean that you fail to submit a piece of work on time. In such cases you must inform the Campus Office in writing on a mitigating circumstances form, giving the reason for your late or non-submission. You must provide relevant documentary evidence with the form. This information will be reported to the relevant Assessment Board that will decide whether the mark of zero shall stand. For more detailed information regarding University Assessment Regulations, please refer to the following website: **http://www.westminster.ac.uk/study/current-students/resources/academic-regulations**

## Introduction

In this coursework you are expected to use your knowledge on object oriented PHP in order to construct functional components of Web applications. Furthermore, you will be creating a prototype to show how specifications of given cases can be implemented using object oriented PHP and MySQL. You are asked to test your work on given scenarios in order to demonstrate how your implementation works.

For the needs of this coursework, you must complete part A and part B and give a presentation on the work submitted, during week 12.

## PART A **Total: 45 marks**

You start a new electronic site through which people can auction items they do not want any more. Items auctioned through your site are: pieces of furniture, books and paintings. Users who are interested, can see the current bid and place a new bid. For each item, the following details should be known by the system:

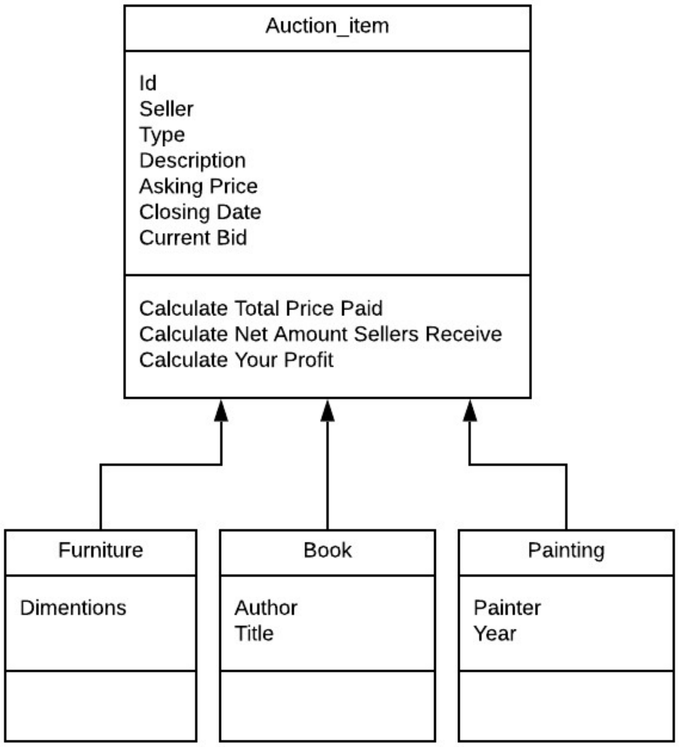
- Unique id number
- The name of the person who sells it (seller),
- The type of the item (furniture, book or painting),
- A brief description,
- The price the seller asks (starting price),
- The last day people can make bids (closing date)
- Current bid

Furthermore, for different types of items, sellers must provide different information. More specifically:

- Furniture: sellers must provide information about the dimensions;
- Book: sellers must provide information about the author and the title of the book;
- Painting: sellers must provide the name of the painter and the year it was painted.

Finally, for each item sold through your site, you charge the buyer a premium of 10% on the selling price (buyers premium) and the seller a flat fee of £10 (sellers premium). As a result, for each item sold in your site you must calculate the total price that the buyer will pay, the net amount that the seller will receive as well as your profit.

The following diagram describes the above scenario:



The following tables, provide information about the items that are auctioned through your site:

| Id | Seller | Type | Description | Asking Price | Closing Date | Current Bid |
|---|---|---|---|---|---|---|
| 1 | Peter | Furniture | Coffee table | 85 | 31/03/2020 | 60 |
| 2 | Ann | Book | Play drama | 15 | 25/03/2020 | 7 |
| 3 | Helen | Painting | Oil on Canvas | 3000 | 30/04/2020 | 850 |
| 4 | John | Painting | Pop Art | 400 | 30/04/2020 | 420 |
| 5 | Wendy | Painting | Spanish School | 15000 | 31/05/2020 | 10000 |

Further information for each item, according to its type, also provided by the sellers:

Item 1: **Dimension**: 85 cm X 35 cm X 52.
Item 2: **Author**: William Shakespeare, **Title**: Macbeth
Item 3: **Painter**: Hannah Fairfield, **Year**: 1839
Item 4: **Painter**: Callypso, **Year**: 2015
Item 5: **Painter**: Goya, **Year**: 1800

Using OO PHP, perform the following tasks:

**1) Create classes** **[15 marks]**

Create all the necessary classes and subclasses, according to the above diagram and description (hint: make sure you define all the properties and the methods (functions)). Create the necessary methods (functions) to calculate the total price that the buyer will pay, the amount that the seller will receive and your profit, if an item is sold in the asking price.

**2) Create instances. [10 marks]**

Create all the necessary instances according to the above description. You get full marks if you use 'construct method'; you get maximum of 5 marks, if you do not use 'construct method'.

**3) Case 1 – apply all functions [3 marks]**

For ***each*** object, calculate and print on the screen: the amount that buyers will pay, the amount that sellers will receive and your own profit, if the object is sold in the asking price. Make sure that you produce meaningful messages, that will include all the necessary information such as: <id>, <asking price>, <buyers premium>, <sellers premium> and the results of the calculations.

**4) Case 2 – Reserved price [6 marks]**

Further to the above description, *for paintings only*, sellers have to provide one more piece of information: reserved price. The reserved price is the minimum price that a seller will accept. If the current bid is under the reserved price, then the painting will remain unsold. This piece of information is confidential. Add the new property (reserved price) in 'Painting' sub-class and use the following data:

| **Id** | **Reserved price** |
|---|---|
| 3 | 1800 |
| 4 | 85 |
| 5 | 9500 |

Create a function for the 'Painting' sub-class that will define if an item will be sold or will remain unsold, in relation to the current bid. Hint: this is a private property, as a result, more functions must be defined in order to set and read the values of the property.

**5) Case 3 – Bids [6 marks]**

According to the above description and diagram a property of the Auction_item class is the 'Current Bid'. On top of that, you want to know all the bids that were made for each item (bids history). Add a new property (bids history) that will hold the following information: the date that a bid was made and the suggested price. Apparently, the new property must be an associative array. Add the new property in the class Auction_item and create a function that will add each new bid in the bid history [Hint: you could use 'array_push']. Create another function that will present the bid history on the screen. It should produce a meaningful message such as: 'For item <id>, the following bids are made: Date <XXX>, Bid: <YYY>'.
Call both functions, as required, with the following data:

| Id | Bids history |
|---|---|
| 1 | 25 January 2020 →20<br>31 January 2020 →28<br>01 February 2020 →35<br>10 February 2020 →45<br>12 February 2020 →60 |
| 3 | 20 January 2020 →550<br>27 January 2020 →600<br>15 February 2020 →850 |

**6) Comments.** **[5 marks]**

You must provide meaningful comments that explain your code.

**PART B** **[45 marks]**

Part B is based on the 'Online Academy' scenario we used during semester 1. For the needs of this coursework, we will refer only to one actor: the tutor. In order to offer tutorials through 'Online Academy' site, tutors must provide the following information: name, email, phone number, qualifications, subjects to teach and rates per hour. Using object oriented PHP and MySQL, you should create a web site that will allow users/administrators to browse for the tutors who offer tutorials for 'Online Academy' and take some actions. In details, you should perform the following tasks:

**1) Preparation and organization of the code / site** **[10 marks]**

- Create your database with all the necessary tables and populate them with the following data *[3 marks]:*

| Name | Email | Phone Number | Qualifications | Subjects to teach | Rate per hour |
|---|---|---|---|---|---|
| Peter Brown | peter@onlineacademy.com | 000111 | BSc Computer Science | Programming<br>Databases | 40 |
| John White | john@onlineacademy.com | 111222 | MSc Biology | Physics<br>Biology | 70 |
| David Red | david@onlineacademy.com | 222333 | BA Art | Design | 50 |
| Sue Black | sue@onlineacademy.com | 333444 | Diploma Biology | Nutrition<br>First Aid | 35 |
| Jane Red | jane@onlineacademy.com | 444555 | PhD in Politics | History<br>Sociology | 65 |
| Abigail Blue | abigail@onlineacademy.com | 555666 | BSc in Multimedia | Design<br>Programming | 45 |

- Create the basic site (you could use the same template we use for the lectures/tutorials and make the necessary changes). You do **not** get extra marks for design elements (e.g. pictures, background etc.). When you call the site, a welcome message appears on the screen and the option 'our tutors', in the navigation bar. Make sure you update the navigation bar as required. *[2 marks]*
- Organize your project and use the appropriate structure and folders. Make sure you keep the project tidy. *[3 marks]*
- Connect your database with your site. *[2 marks]*

**2) Present all tutors on the screen – specific information** **[4 marks]**

When the user clicks the option 'our tutors' in the navigation bar, s/he gets on screen the names of all tutors and the following details for each one: email and subjects to teach.

**3) Get full details for a specific tutor** **[4 marks]**

When the user clicks the name of a tutor a new page is generated, where all the details about this specific tutor (name, email, phone number, qualifications, subjects to teach and rate per hour) are presented.

**4) Add, delete, edit** **[12 marks]**

Furthermore, users must be able to perform the following actions: **add** a new tutor; **delete** an existing tutor; **edit** the details of an existing subject (e.g. modify the subjects).

**5) Optimize code** **[10 marks]**

There are more than one ways to create your code and make it reusable. You get 10 marks if you use the class $database.

**6) Comments.** **[5 marks]**

You must provide meaningful comments that explain your code.

**PART C** **[10 marks]**

During week 12, after the submission of your coursework, you must present your work to your tutor. During the presentation you must demonstrate your site and go through the task you are asked to perform. Exact day/time must be confirmed with your tutor.

**Further instructions**

- All files must be uploaded to Blackboard.
- You must submit a file that will contain the following:
    - One paragraph for each task that will provide the name of the file that is the answer to that task and any other information required by the task.
    - An electronic copy of the produced software application. ALL OO PHP files, MySQL files or any other content used in the application must be included.
- You must upload only **ONE file.**
- Your final mark depends on the quality of your answer. Readability of code and use of descriptive, non-confusing names improve the quality.
- Furthermore, the final mark depends on the **understanding** of the code you demonstrate during the viva and from the comments inside your code. Reduced understanding will affect your final mark.
- DO NOT FORGET: **Plagiarism** is considered cheating, as you have taken the words or ideas of other people and passed them off as your own. The University takes cases of plagiarism very seriously. If you are caught plagiarising you will face disciplinary procedures which could ultimately result in your expulsion. Please refer to the Student Handbook for a clarification of what constitutes plagiarism.

## Marking Scheme

| Criteria | Mark per component | Mark provided | Comments |
|---|---|---|---|
| **PART A** | **45 marks** | | |
| 1) Classes<br>• Classes: 4 marks (1 mark each)<br>• Properties: 2 marks (12 properties – 1 mark every 6 properties)<br>• Functions →<br>○ Calculate what buyers pay (3)<br>○ Calculate what sellers get (3)<br>○ Calculate your profit (3) | **15** | | |
| 2) Instances<br>Full marks (10) will be given if '**construct**' is used; otherwise maximum marks: 5. | **10** | | |
| 3) Case 1<br>Use each one of the functions you created in 1) with all objects. (3 functions, 1 mark each if used with all objects and meaningful messages are generated) | **3** | | |
| 4) Case 2 – reserved price<br>• Add new property and any functions required to use it (2 marks)<br>• Create the 'compare' function (2 marks)<br>• Call functions as required in order to produce a meaningful output (2 marks) | **6** | | |
| 5) Case 3<br>• Add new property as required (1 mark)<br>• Add necessary functions (3 marks)<br>• Call functions as required according to the description (2 marks) | **6** | | |
| 6) Comments *** | **5** | | |
| **PART B** | **45 marks** | | |
| 1) Preparation and organization of the code / site<br>• Database and tables: 3 marks<br>• Basic design, navigation, first screen-welcome message: 2 marks<br>• Organize your project / folders: 3 marks<br>• Connect DB with your site: 2 marks | **10** | | |

| | | | |
|---|---|---|---|
| 2) Present all tutors on the screen – specific information | **4** | | |
| 3) Get full details for a specific tutor | **4** | | |
| 4) Add, delete, edit<br>• Add: 4 marks<br>• Delete: 3 marks<br>• Edit: 5 marks | **12** | | |
| 5) Optimize code; use of $database class | **10** | | |
| 6) Comments *** | **5** | | |
| **PART C** | **10 marks** | | |
| Presentation during week 12 | | | |
| **TOTAL** | **100** | | |

***** About comments**: You get full marks for comments if 80-100% of your code is commented; you get 4 marks if 60%-80% of your code is commented; you get 3 marks if 40%-60% of your code is commented; you get 2 marks if 20%-40% of your code is commented; you get 1 mark if 5%-20% of your code is commented; you get no marks if less than 5% of you code is commented. *Only correct comments get marks.*