# Tutorial letter 103/0/2022

## Advanced Internet Programming

## ICT3612

### Year
### Module

### School of Computing

This tutorial letter contains information about assignment 3

BAR CODE

# 1. INTRODUCTION

Dear Student

This tutorial letter contains the requirements of Assignment 3. **The due date is 15 August 2022**. *(This is your last assignment for this year and the due date was scheduled as late as possible, to assist students in completing this important assignment. No late submissions can be accommodated and ONLY submissions via myUnisa can be accommodated.)* Do your best to submit 2-3 days before this date, to prevent any possible technical difficulties.

Please read this tutorial letter carefully and ensure to submit your own work. Make sure that you understand all the rules and requirements. Also, please read the document *Policy for Copyright Infringement and Plagiarism* with specific reference to point 4.3 and 5 from the Learner Support & Regions / Student Policies & Rules / Policy for Copyright Infringement and Plagiarism from Unisa's website.

Furthermore, we have the right to request an oral examination if we deem it necessary to complete the assessment process.

*4.3 Dishonest practices may also amount to criminal offences, such as fraud, theft and criminal copyright liability. Such dishonest practices include the following:*

*4.3.1 copying information from another person (e.g. another student's assignment or portfolio) and submitting identical work where such work is not the result of teamwork and indicated as such by all participants;*

*4.3.2 buying an essay from a ghost-writing service and pretending that it is one's own work;*

*4.3.3 asking someone else to do an assignment on one's behalf.*

*5. CONTRAVENTION OF THIS POLICY*

*A student or an employee who is guilty of the infringement of copyright or unethical practice will be subject to the applicable disciplinary code.*

Best wishes in completing your assignment 3.

# 2. COMPLETING ASSIGNMENT 3

a. As you did in ICT2613, first complete all the assignment tasks on your computer.

b. For each task page, create a text file with the same name. For example, for `task1.php` you will have an extra file named `task1.txt`. This text file will contain the PHP code for that task (leave out any HTML code that is not part of the PHP code). For each task you will thus have a `.php` and a `.txt` file. Because it is a text file, the server will not parse (interpret) the PHP code in the file, allowing the marker

to view your code for that task.

c. Create a file named menu.inc. Inside this file, write HTML code that will produce a **horizontal menu** with links to all the different task pages.

| Task 1 | Task 2 | Task 3 | Task 4 | etc.

d. Create a page named `index.php`. Just after the `<body>` tag, include the following line of PHP code:

```
<?php include 'menu.inc'; ?>
```

e. Include this line of PHP code in every `.php` task page. If you want to change a link for whatever reason, then you only need to change it once in `menu.inc`, as opposed to every task page.

## 3. MAKING YOUR ASSIGNMENT 3 AVAILABLE FOR MARKING

a. Open an account on a free web hosting server **of your choice**. Make sure this web hosting service supports PHP and MySQL. Examples of such free hosting servers are https://www.000webhost.com/ and https://infinityfree.net/.

https://www.000webhost.com/ worked well in 2019, 2020 and 2021. When you register a site on the server of your choice, you must use a site name that contains your student number as well as any other word/s of your choice. Here is an example: 8785634-wanghor. 000webhost.com. If your site contains only your student number, it is relatively easy for other students to guess and gain access to your code. **For this reason, if your site name contains only your student number, it will not be marked.**

b. Transfer your .php and .txt files to your web site. For some guidance, we will upload a document that outlines how to upload on https://www.000webhost.com/ in Additional Resources folder during the year. Also make sure that you have read through the account details as displayed/sent to you after registration. Remember to upload the index.php page as well, overwriting the one that may already be in there. Alternatively, if there is an index.html, index.htm, default.htm page in the directory, then delete it.

You will also be required to recreate your database on this free web hosting server using phpMyAdmin.

c. You will submit the URL of your web site to us for marking (see the ASSIGNMENT 3 SUBMISSION PROCEDURE section). The URL must take the user to the homepage of your website and this homepage must display the horizontal menu you have created as stated in Section 2, step c. Your web site containing your full assignment 3 must be available for marking via the horizontal menu.

d. At the bottom of each page, include the relevant text file in a centered `iframe` with an exact width of 1200px and a height of 400px. Here is how you do it:

```
<iframe src="task1.txt" height="400" scrolling="yes" width="1200px">
<p>Your browser does not support iframes.</p></iframe>
```

e. If required, and in the text file, use a series of forward slashes across the page to separate sections of

code, e.g.:

```
////////////////////////////// Task1 (a) ////////////////////////
                        Your code for task 1 (a)
////////////////////////////// Task1 (b) ////////////////////////
                        Your code for task 1 (b)
                                  etc.
```

**IMPORTANT:**

a. If your code is not available in an `iframe` on a task page then that task will not be marked. The code you submit to *my*Unisa is only used when we suspect plagiarism.

b. If the menu link to a task page or the task page itself is not available, then a task cannot be marked. It cannot be left to the marker to work out how to access a task page.

c. If the text file cannot be loaded in the `iframe` then the task page will start loading and then go blank. One reason why this happens is because file names hosted on a server are case-sensitive. `Task1.txt` is not the same as `task1.txt`. The same holds true for `.php` file names.

d. We mark the result (output) of your code. No output then no mark. We use the code in the `iframe` to verify the output was generated as per the task requirement. In other words, we do not mark code only.

## 4. ASSIGNMENT 3 SUBMISSION PROCEDURE

### This section outlines the EXACT steps to follow when submitting assignment 3 for marking via *my*Unisa. It is very important to read and clearly understand these steps.

**STEP 1:**

Create a folder named **Assignment3** on your computer.

**STEP 2:**

- To access your site for marking, we will require a working URL. This URL will load the default launch page (`index.php` - which contains the horizontal menu).

- Inside the folder created in step 1, create a new text file and save it as `URL.txt`. **Copy (do not type)** the URL of your web site into this file. Test the URL and make sure we can access your site. **If we cannot access your site because of a wrong URL or because you forgot to include the URL/text file, then your assignment 3 cannot be marked.** This is no different to failing to answer a question in a sit-down examination.

- Copy all your assignment 3 code (`php` files) and all the `iframe` text files into the folder you have created above. Do not include any other "clutter" e.g. images you may have used. There should **only** be `.php` and `.txt` files in this folder. Your zip file should not be more than 100KB in size.

- Create another text file, and name it `connection.txt`. Inside this file, provide your **live** site's database name, username and password (note it is NOT the database connection details as used on your own computer, but the connection details used on the site that you are currently submitting for marking ).

- Create another text file named `login_details.txt`. In this file provide all the necessary login details required to access the file manager you have used to upload your files. If you used a FTP client, provide the required details.

**STEP 3:**

Using WinZip, zip this folder. WinZip will automatically give the zip file the name of your folder (e.g. Assignment3.zip).

**STEP 4:**

Upload this zip file to *my*Unisa under Assignment 3. Ensure you upload before the due date in case you experience internet connection problems etc.

## 5. WEB SITE AVAILABILITY & OTHER RULES AND NOTES

a. If your menu is not available and we are required to use a browser back-button to access a previous page, you will be penalized 10% of the final mark awarded. **If a task page is not found because of an error you made in the menu link to that page, that task will not be marked.** Note there is a difference between a page not loading because the server is slow and a page not found. We revisit sites/pages that load slowly.

b. Many free web hosting servers delete sites that are not accessed regularly. Nevertheless, you should visit your site at least once a week until you have received your mark. If your site is not available, restore it as quickly as possible using the same server site name and notify us via email (thomaa@unisa.ac.za) that your site was down and that it has been restored. **Your original URL from the submitted zip file will be used to access the restored site i.e. we will not accept a different URL.**

c. You are not allowed to use any PHP frameworks or software that generates (complete) code on your behalf. You must code the entire application by yourself.

d. You are expected to submit your own work. If we suspect plagiarism, we will award 0 marks for copied work.

e. Web hosting sites can have down times and if so, please try later. For this reason, please do not wait for the due date of the assignment to upload files on the web server. Try to do it in advance.

f. Marks for assignment 3 are awarded solely based on the website (navigation, output, code in iframes, etc) you have created. Note that the code that you have submitted on myUnisa is used only for checking plagiarism and record keeping.

## 6. SUPPORT PROVIDED

a. Tasks in this assignment can be done mainly using the prescribed chapters of the textbook.
b. Try and make use of Google to find solutions to errors. As a PHP-developer you WILL spend hours on Internet forums seeking solutions to problems.
c. Make use of PHP documentation available online (see chapter 2 on how to access it)
d. Participate in *my*Unisa forum
e. E-mail the lecturer for help

## 7. ASSIGNMENT 3

**1.** Make use of comments in your code
**2.** All the tasks and subtasks (excluding code in the iframe) will produce output of some sort to the screen. We first consider the output produced, and then look at the code to see how the output was produced.
**3.** When a task has subtasks (marked using (a) and (b)), label the subtasks clearly in the output wherever possible.

**Task 1: Chapter 13: page name=task1.php** **24 marks**

This task consists of two different subtasks.

**(a)** **[12 marks]**

Code a function (`boolToText()`) that takes two arguments; the first argument takes a boolean value (0 or 1) and the second argument takes the format in which the given boolean value must be displayed. The default value for the second argument is 1.

When the value of the second argument is `1`, the function displays `"False"` for `0` and `"True"` for `1`. When the value of the second argument is `2`, the function displays `"No"` for `0` and `"Yes"` for `1`. When the value of the second argument is `3`, the function displays `"Negative"` for `0` and `"Positive"` for `1`. For any other format, the function displays `0` for `0` and `1` for `1`.

Invoke the function four times as follows:

```
boolToText(1);
boolToText(0, 2);
boolToText(1, 3);
boolToText(0, 5);
```

**(b)** **[12 marks]**

Write code for a function that accepts any number of arguments, and displays the total number of arguments and the total number of numeral arguments passed to it in each invocation. The function must make use of a variable-length parameter list.

For example, if the function is invoked with the arguments `"Thando", 23, "Busi", 40`, it will display the following:

```
Total number of arguments: 4, total number of numerals in these arguments: 2
```

On the other hand, if the function is invoked with the argument `"Mutsa"`, it will display the following:

```
Total number of arguments: 1, total number of numerals in these arguments: 0
```

Write code to invoke the function twice with arguments:
(1) `"Thando", 23, "Busi", 40`
(2) `"Mutsa"`

**Task 2: Chapter 14: page name=task2.php** **15 marks**

Write code for a class named `Square` with two properties; one to store the name of the shape (in this case `"Square"`) and one to represent the length of one side of a square. The class must have

- a constructor that accepts one parameter to initialise the length of the square. The constructor also initialises the name of the shape to `"Square"`
- getter methods to return values of properties
- one setter method to change the value of the length of one side of a square
- a method `getArea()` that calculates and returns the area of a square.
  Area of a square is $length\ of\ one\ side \times length\ of\ one\ side$
- a method `getPerimeter()` that calculates and returns the perimeter of a square
  Perimeter of a square is $4 \times length\ of\ one\ side$

Create an object of the `Square` class. Demonstrate how each method in the class is invoked using the created `Square` object and display output for each method invocation, where appropriate.

**Task 3: Chapter 14: page name=task3.php** **25 marks**

ICT3612 has three assignments. Assignments 1, 2 and 3 contributes 10%, 10% and 80% respectively toward the year mark. For example, if a student obtained 70%, 80% and 50% for assignments 1, 2 and 3 respectively, their year mark will be 55 (70×.1+80 ×.1+50 ×.8).

- Write code for a class named `AssignmentRecord` that satisfies the following:

- It has four private properties for storing a student number and three assignment marks.
- It has three class constants to store the weights of assignment contributions to the year mark. Initialise these constants to values .1,.1 and .8.
- It has a constructor with four parameters, which are used to initialise the properties of the class.
- It has a method that calculates and returns the year mark. Use the properties and constants of the class to calculate the year mark.
- It has a method `__toString()` that returns values of the properties of the class in comma separated values format.
For example, this method should return 123456,70,80,50 for an assignment record of a student (123456) with marks 70%, 80% and 50% for assignments 1, 2 and 3 respectively.

- Write code for a class named `FullRecord`, a subclass of `AssignmentRecord`, that satisfies the

following:

- It has one private property to store the exam mark of a student.
- It has a constructor with five parameters that are used to initialise the properties of this class and `AssignmentRecord`.
- It has a method `__toString()` that returns values of the properties of this class and `AssignmentRecord` in comma separated format of the This method must use the value returned by `__toString()` in `AssignmentRecord`.

For example, this method should return 123456,70,80,50,55 for a student (123456) with marks 70%, 80%, 50%, 55% for assignments 1, 2 and 3, and exam respectively.

- Write code to create a `FullRecord` object. Invoke the method to calculate the year mark and display the year mark for this object. Invoke the `__toString()` and display this string for this object.

**Task 4: Chapter 15 : page name=task4.php** **15 marks**

This task consists of two different subtasks.

**(a)** **[10 marks]**

Write a class named `Validate` that can be used to validate user selected usernames and passwords for a system.

The class has two static properties and they both store regular expressions to validate usernames and passwords. The username must only consist of four lowercase letters. The password must be a 6 to 8-digit number.

The class has two static methods. The first method accepts a username and validates it against the regular expression for username and returns true or false. The second method accepts a password and validates it against the regular expression for password and returns true or false.

Invoke the methods in `Validate` twice each, with usernames and passwords that will return true and false in different calls of the relevant methods.

**(b)** **[5 marks]**

For the given pattern `/^[01]?\d\/[0-3]\d\/\d{4}$/` present one string that will be accepted by the pattern and another string that will not be accepted by the pattern.

Code the pattern and the input strings in `preg_match()`. Display the values returned by `preg_match()` for both valid and invalid input strings.

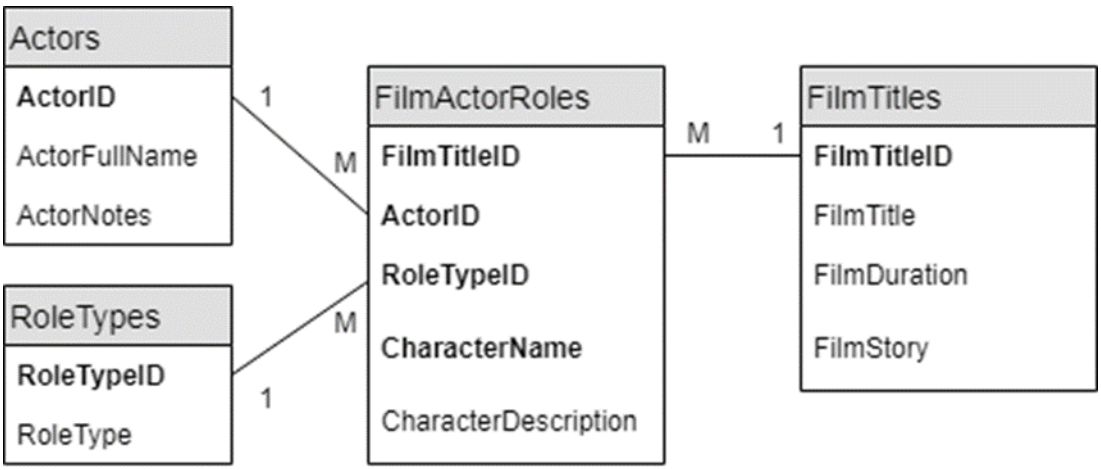
**Task 5: Chapter 16: page name=task5.php** **10 marks**

Refer to the figure in the Section titled 'How to apply the third normal form' in Chapter 16.

Explain briefly what each table represent (i.e. the entity it represents). List all the primary and foreign keys for each table.

**Task 6: Chapter 16, 17, 18, 19: page name=task6.php** **25 marks**

Create a database with four tables. The basic structure of the tables and relationships between the tables are illustrated below:




Populate the tables with realistic data, with at least 5 rows of data in each table. Note that the SQL data type for each column is not given in the diagram. So you have to decide on this aspect.

Write PHP code to connect to the database and display the contents of all four database tables in four different HTML tables on the web page.

Below the HTML tables, there must be a form with five radio buttons where the user can choose to view the results of one of the five different SELECT queries at a time. Details of these five SELECT queries are given below:

1) The first SELECT query must make use of the ORDER BY clause
2) The second SELECT query must make use of the LIKE operator
3) The third SELECT query must make use of an inner join that joins two tables
4) The fourth SELECT query must make use of a WHERE clause with the OR logical operator
5) The fifth SELECT query must make use of the aggregate function MAX.

You can decide on the other details of these SELECT queries.

**Task 7: Chapter 21: page name=task7.php** **12 marks**

List five secure website URLs, the respective certification authorities for the site's digital secure certificate and the date on which the digital secure certificate expires.

**Task 8: Chapter 23: page name=task8.php** **14 marks**

This task consists of two different subtasks. Both these subtasks involve modification of `task3.php`.

(a) Add a function named `writeToFile()` that accepts an array of `FullRecord` objects. The function iterates through the array and write the string returned by the `__toString()` function for each object into a file named `fullrecords.txt`. Each line of data `fullrecords.txt` represent the string representation of a different object.

Invoke the function with an array of at least five `FullRecord` objects.

Create a hyperlink pointing to the file `fullrecords.txt` and include it on the task page. When the user clicks on the link, the browser can then download it. (7)

(b) Write code to read the file `fullrecords.txt` in order to create an array of `FullRecord` objects. An object of `FullRecord` is created from the information given in each line of the file, and each created object is added to an array. Iterate through the array and invoke `__toString()` on each object. Display the string returned by `__toString()` for each object. (7)

**Task 9: Chapter 13 to 23: page name=task9.php** **30 marks**

In this task you will provide a solution or part of a solution to a problem for a community of your choice. The solution you are expected to design and code is a web-based database application. The purpose of these tasks is for you to apply your skills to address a real community-based problem.

The term community should be taken in a broader sense and the choice of community is entirely yours. Examples of your communities are student, residential, work and religious communities. The problem you are trying to address could be based on your personal experience in the community or gathered from the members of a community.

Given below is an example of a problem and a possible web-based database solution to address this problem. This is a sample case – do not use this for your assignment, please identify a unique problem relevant to your community of choice.

**Problem:** Companies/municipalities are expected to consult residents of an area to build/include certain types of infrastructure (examples include cellphone and webcam towers). These consultations happen during scheduled face-to-face meetings with the residents, which are poorly attended. Residents want to have their opinion on the proposed infrastructure noted but feel overlooked because they could not attend face-to-face consultation meetings.

**Solution:** In addition to the face-to-face consultations, an online database application that allows residents to indicate their support or opposition to the proposed infrastructure development can address this issue to a certain extent.

**(a)** Explain the community-based problem you are trying to address in this question. Also explain your proposed solution to address the problem. Describe the problem and the solution using at least 5 sentences (see above description of a sample problem and a possible solution). Display this explanation on the task page. **(5)**

**(b)** The solution must have at least two database tables, user-friendly forms/options to add, modify and delete data from the database table and to view the data in the database table in appropriate formats. The code in the solution must be structured using the Model View Controller (MVC) Pattern. When using MVC pattern your solution will have numerous PHP files. For the purpose of displaying the code in iframe, you can include code in all the files into one txt file for this task. **(25)**