LaSalle College
Montréal

Mid-term Evaluation: 35%

Course Identification

Name of program – Code:	COMPUTER SCIENCE TECHNOLOGY – PROGRAMMING (420.BP)
Course title:	**WEB SERVER APPLICATIONS DEVELOPMENT I**
Course number:	420-DW3-AS
Group:	07200
Teacher's name:	Jean-François Parent
Duration:	Extended
Semester:	Fall 2022

Student Identification

Name: _____ Student number: _____

Date: _____ Result: _____

☐ I declare that this is an original work, and that I credited all content sources of which I am not the author (online and printed, images, graphics, films, etc.), in the required quotation and citation style for this work.

Standard of the Evaluated Competencies
Statement of the evaluated competency – Code
Develop transactional Web applications – 00SU
Evaluated elements of the competency 00SU
1. Analyze the application developement project *2. Prepare the computer development environment* *4. Program the Web interface* *5. Program the server-side application logic* *6. Program the client-side application logic* *7. Control the quality of the application* *8. Participate in the deployment of the application on the Web host* *9. Produce the documentation*

Instructions

- Students will submit a NetBeans project and a cheat sheet individually.
- It is the teacher's responsibility to identify language errors. If such errors are found, teacher may deduct up to 5% of the final grade (IPEL – Article 5.7).
- Plagiarism, attempts at plagiarism or complicity in plagiarism during a summative evaluation results in a mark of zero (0). In the case of recidivism, in the same course or in another course, the student will be given a grade of '0' for the course in question. (IPEL – Article 5.16).
- Deadlines are shared on Omnivox in the assignment box and must be respected.

- Please see the detailed rubric in the document for the breakdown of the mark for the individual work.

TOTAL: 100 POINTS

If we feel that your answers may not be yours, the department reserves the right to complete your evaluation with a virtual meeting to verify that you have reached the required competency.

Project 1

Guidelines

Note: This is **NOT** a group project

You have to do this project by implementing all the following working methods:

1. All the files required in this project must be included in a **Git repository**. Everytime you work on the project, and the end of every day of work, you must **commit** your work to the repository (even if the code temporarily contains errors). For every commit you do, write a **commit message** which should include your **name** and **student number**, the **date** of the commit, and a **short description** of what you did during that day. For example, this is a valid commit message:

```
Bill Torvalds (1244556)        2022-01-30        Created NetBeans project and empty folders.
```

Note: You may also add more information in the commit message, like a version number, or important steps in the development of your website.

2. At the very top of **all the text files** you submit (i.e. all your **.php** files, **.css** file(s), your **PHP cheat sheet**, etc. you must write a **revision history** which should include **all the commit messages**, including your **name** and **student number**, the **date** of the commit, and a **short description** of the work done:

```
#Revision history:
#
#DEVELOPER                  DATE          COMMENTS
#Bill Torvalds (1244556)    2022-01-30    Created NetBeans project and empty folders.
#Bill Torvalds (1244556)    2022-02-01    Created buy.php file and completed about
#                                         50% of requirements for this page.
#Bill Torvalds (1244556)    2022-02-03    Created and completed orders.php file.
#Bill Torvalds (1244556)    2022-02-05    Fixed the bug in the taxes calculation.
#Bill Torvalds (1244556)    2022-02-10    Commented debugging code. Entire project
#                                         completed at 100%.
```

3. Use a **lot of comments** in the code of **all your files**. Use only // or # (you can use /* */ but only for debugging).

4. Use **folders** for CSS, Images, JavaScript, PHP common functions, and more.

5. Use **constants** to avoid hard-coding. For example use:

define("CUSTOMER_FIRSTNAME_MAX_LENGTH", **15**); #Max length of firstname
define("FILE_PHP_COMMON", FOLDER_PHP . "**commonFunctions.php**");

Note: MAX is an abbreviation but its meaning is clear.

6. Always use **relative paths**. Never use absolute paths for files and folders. Make sure the constants/variables containing files and folders are using the **correct letter case**. For example if a filename is PHPFunctions.php, don't create a constant with the value "phpfunctions.php" (lowercase) or your website may crash (on your server or on a Linux server).

7. Give **clear names** to your **variables** and **functions**.

8. Use **correct indentation** for the code and the curly brackets.

9. You must always use **functions** to generate the HTML code. For example, <!DOCTYPE html> should only be typed once in all the project, even if three different pages echo it. These common functions should be placed in a PHP file located in the PHP functions folder.

10. Every page of your website must **send all the HTTP headers** required to **prevent page caching**, so when users will reload the page, they will always get the latest version of your files.

11. Every page of your website should be able to display all the canadian french characters. So all the files from your website must handle **UTF-8** properly with the correct **HTML tag** and **UTF-8** HTTP header.

12. You must always send **all the network headers before** you start to echo the <!DOCTYPE html> text and the rest of the HTML code.

13. Your PHP code must **always generate valid HTML**. Use **View Page Source** in your browser to make sure all your pages contain **no malformed html** (which would be displayed in red).

14. You must **protect all the PHP pages** against HTML and JavaScript injection.

15. If an **error/exception** occurs in your code, save the details into a specific **folder** and **log file** (give them names). The log file should contain all these details: the **description** of the error, the error **code** (if available), the **date and time** when the problem happened (format should be *year/month/day hour:minute:second*), the name of the **PHP file** and the **line number** of the error. You should also create a boolean **constant** to **show** the details in the browser (for debugging) or to **hide** the details to the users and just display a generic error (for production).

16. When all your tests are done and conclusive, **comment** all your debugging code and set the constant of guideline 15. to **false** because the browser (client side) must **never show technical** (server side) **information** to the users.

Description of the project

For this project, you have to invent a company **name** of your own and find a **logo** for it. You also have to determine which kind of **product** this company will sell.

Then you have to create a small website to record all the orders made by customers. This website project name should be your **student number**.

All the files should use a common PHP functions .php file for common operations.

At the bottom of every page, there should be the following text:

> **Copyright <your name (your student number)> 2022**

Instead of typing 2022 you have to use the **current year** of the server current date, so on january 1st 2023, the website will display, for example:

> **Copyright Bill Torvalds (1244556) 2023**.

You must not use any style attribute directly in HTML tags. All the pages should make use of at least one **.css file** and all the pages must have a **background color**. The design of your pages must not look too minimalistic.

The website consists of 3 pages, which must display a **different title** in the **tabs**:

- **Home page**

  This is the welcome page. Display the **logo** of the company and compose **two or three sentences** of your own to describe the company and the products it sells. The logo should also be included in **all the pages** of your website.

  This page, and all the others, should have the same **navigation menu** to browse in the 3 pages of your website.

  On this page you should have a section for advertising. This section should **display randomly only one picture of a product** among a list of 5 products that you could sell on your website (software, products, food, etc). All the pictures should have the same size on the screen. Among these 5 ads, one of them (choose which one) is sold twice the price to our clients and this advertising (and this one only) should be displayed **100% bigger**, and it should have a **green border** of **6 px**. If any of these ads are clicked, redirect the user to any **website** (no need to redirect to 5 different websites).

- **Buying page**

This page contains a form with the following fields (your website should display * = required and display the red * near required textboxes):

- **Product code** *

The product code cannot be empty, cannot be longer than 25 characters, and must always contain the letters PRD (uppercase, lowercase, or both). For example: 45-Prd-MOUSE is a valid product code. 44b-Keyboard is not a valid product code.

- **Customer first name** *

The first name cannot be empty, and cannot be longer than 20 characters.

- **Customer last name** *

The last name cannot be empty, and cannot be longer than 20 characters.

- **Customer City** *

The city cannot be empty, and cannot be longer than 30 characters.

- **Comments**

The user may add optional comments to the transaction. Allowed string length goes from 0 to 200 characters.

- **Price** *

The price of the product must be entirely numeric. It must not be a negative value and cannot be higher that 10,000.00$.

- **Quantity** *

The quantity sold of the product must be a numeric value between 1 and 99. No decimals are allowed, so a quantity of 1.3 is not valid.

The form should have **submit** button. If the data from some fields is not valid, you have to write in **red**, **near every corresponding field**, a clear message telling the user how to solve the problem (for example: the last name cannot contain more than 20 characters). You must also retain the typed value.

When all the data is valid, show a **confirmation message** to the user. You also have to multiply the price by the quantity. This will give you the **subtotal**. You then have to apply the local taxes of 16.1% to this subtotal to get the **taxes amount**. Finally, add the subtotal and the taxes amount to get the **grand total**. The result of all these calculations **must always contain only 2 digits**. For example, if you have calculated 12.249765 for the grand total, keep only 12.25 in the variable.

Create an array with the data for all the following values: ProductCode, FirstName, LastName, City, Price, Quantity, Comments, Subtotal, Taxes amount and Grand total.

Convert this array to a **json string** and save it in a **file** to keep the orders. This file must be located in a **folder** which should be located in your project along with the PHP, CSS and other folders. **Important:** don’t overwrite the existing data in the file! After the file is saved properly, show a **confirmation message** to the user and clear all the form to create another sale.

- **Orders page**

This page must contain a **HTML table** showing all the orders made on the website. You must thus open orders file saved earlier (the code must not crash if the file does not exist) and generate a HTML <table> with the appropriate **column headers**.

Then you have to **read** all the lines of the **orders.txt** file one by one, and for each line generate a **<tr>**, and generate a **<td>** for each field found in that line of the file.

All the borders must be visible, so the table may look like this:

Product ID	First name	Last name	City	Comments	Price	Quantity	Subtotal	Taxes	Grand total
Phelmet4b	Ben	Masvidal	Montréal		49.99$	2	99.98$	16.10$	116.08$
Pgloves675	Justin	Legault	Québec	10 % rebate	22.49$	1	22.49$	3.62$	26.11$

Important: the dollar ($) signs must not be saved in the text file, but should only be added in the HTML <table> for display.

In that page, the user may specify a **action** parameter in the url.

So for example if the user specifies **orders.php?action=print**, the **background color** of that page should be **white**. Also, to save more ink/toner, the **opacity** of all the regular images (like the logo) should be of **0.3**.

If the user specifies **orders.php?action=color**, you should change the color of the amounts in the **subtotal** column. If the subtotal is **less than 100.00$**, it should be displayed in **red**. If the amount is **between 100.00$ and 999.99$**, the amount should be displayed in **light orange**. And finally if the amount is of **1000.00$ or more**, the amount should be displayed in **green**. Choose color tones that are easy to read.

This page should also have a **link to download** your **PHP cheat sheet** asked at the beginning of the semester. Your file should be placed in the same **folder** used to save the orders.txt file.

Files to submit

When your website is ready, make sure the .git folder is included in your project folder. Then create a single **.zip file** (compressed folder) containing your entire **NetBeans PHP project**,which should include your **PHP cheat sheet**, and your **.git folder**. Make sure your .zip file is not corrupted by extracting its contents into a different folder. **Upload** the verified **.zip file** on **Omnivox** to submit your project.

CORRECTION GRID FOR REQUIREMENTS

Compentency : Deploy transactional Web applications – 00SU	
Elements of competencies: Analyze the application developement project (00SU.1)	
Performance criteria	**weight**
Accurate analysis of design documents (00SU.1.1)	/5.5
Proper identification of the tasks to be carried out (00SU.1.2)	/5
Elements of competencies: Prepare the computer development environment (00SU.2)	
Performance criteria	**weight**
Proper installation of the Web development platform and the development database management system (00SU.2.1)	/3
Proper installation of software and libraries (00SU.2.2)	/1
Appropriate configuration of the version control system (00SU.2.3)	/1
Proper importing of the source code (00SU.2.4)	/4
Elements of competencies: Program the Web interface (00SU.4)	
Performance criteria	**weight**
Appropriate use of markup language (00SU.4.1)	/4
Suitable creation and use of style sheets (00SU.4.2)	/3
Proper integration of images (00SU.4.3)	/3
Suitable creation of Web forms (00SU.4.4)	/3
Adaptation of the interface based on the display format and resolution (00SU.4.5)	/1
Elements of competencies: Program the server-side application logic (00SU.5)	
Performance criteria	**weight**
Proper programming of interactions between the Web interface and the user (00SU.5.2)	/4
Appropriate use of data exchange services (00SU.5.5)	/5.5
Proper application of internationalization techniques (00SU.5.6)	/1
Precise application of secure programming techniques (00SU.5.7)	/6
Elements of competencies: Program the client-side application logic (00SU.6)	
Performance criteria	**weight**
Proper programming of interactions between the Web interface and the user (00SU.6.3)	/4
Systematic use of Web form data validation techniques (00SU.6.4)	/3
Web forms in compliance with usability requirements (00SU.6.5)	/3
Elements of competencies: Control the quality of the application (00SU.7)	
Performance criteria	**weight**
Precise application of test plans (00SU.7.1)	/3
Thorough reviews of code and security (00SU.7.2)	/3
Relevance of the corrective actions (00SU.7.3)	/3
Compliance with issue tracking and version control procedures (00SU.7.4)	/3
Compliance with design documents (00SU.7.5)	/6
Elements of competencies: Participate in the deployment of the application on the Web host (00SU.8)	
Performance criteria	**weight**
Accurate identification of the domain name (00SU.8.1)	/3
Appropriate configuration of the application on the Web host (00SU.8.2)	/2
Proper application of the procedure for migrating the service onto the Web host (00SU.8.3)	/2
Precise application of security measures (00SU.8.4)	/3
Compliance with search engine indexing requirements (00SU.8.5)	/1
Elements of competencies: Produce the documentation (00SU.9)	
Performance criteria	**weight**
Proper identification of the information to be written up (00SU.9.1)	/8
Clear record of the work carried out (00SU.9.2)	/3

CORRECTION GRID FOR LANGUAGE

Clear Communication	Clear Comm., **most of the time**	Vague Communication	Unclear Communication
- 0	- 0,5	- 1,5	- 2
(Word Choice) Use of precise and rich vocabulary	(Word Choice) Use of precise vocabulary	(Word Choice) Use of imprecise vocabulary	(Word Choice) Use of inappropriate vocabulary
- 0	- 0,5	- 1,5	- 2
(Format/Type of work) Respect of norms	(Format/Type of work) Respect of **most of the** norms	(Format/Type of work) Non-respect of the norms	(Format/Type of work) Inappropriate in relation to the required norms
- 0	- 0,5	- 1,5	- 2
(Linguistic Code) (≤2 mistakes / page)	(Linguistic Code) (3-7 mistakes/page)	(Linguistic Code) (8-10 mistakes/ page)	(Linguistic Code) (>10 mistakes/page)
- 0	- 0,5 - 2.5	- 2.5 - 3.5	- 4