



<b>FACULTY/COLLEGE</b>	College of Business and Economics		
<b>SCHOOL</b>	School of Consumer Intelligence and Information Systems		
<b>DEPARTMENT</b>	Applied Information Systems		
<b>CAMPUS(ES)</b>	APB		
<b>MODULE NAME</b>	Development Software 2A		
<b>MODULE CODE</b>	DSW02A1		
<b>SEMESTER</b>	First		
<b>ASSESSMENT OPPORTUNITY, MONTH AND YEAR</b>	Assessment: Semester Test 2 May 2025		
<b>ASSESSMENT DATE</b>	07 May 2025 @ 08:00	<b>SUBMISSION DATE</b>	24 May 2025 @ 12h00
<b>ASSESSOR(S)</b>	Mr Ronny Mabokela		
<b>DURATION</b>	4 hours	<b>TOTAL MARKS</b>	100
<b>NUMBER OF PAGES OF QUESTION PAPER (Including cover page)</b>	9		

**INFORMATION/INSTRUCTIONS:**

- ✓ Read the instructions carefully to avoid making mistakes.
- ✓ This is NOT an open-book assessment.
- ✓ There are 4 questions and answer all questions.
- ✓ Submit the necessary files for codes where necessary.
- ✓ Read the questions carefully and answer only what is required.
- ✓ Make sure that your submission is in a zipped file.
- ✓ **DO NOT SHARE THE QUESTION PAPER IN THE WHATSAPP GROUP**
- ✓ **EACH STUDENT MUST ACCESS THE QUESTION PAPER FROM THE BLACKBOARD**

## QUESTION 1

[25 MARKS]

Caster Semenya, a renowned Olympic distance runner, has broken multiple IAAF records in recent years. In 2017, she broke the world record for the 600m event with a time of 1:21.77 minutes in Berlin. The following year, at the Diamond League 18, she broke the 800m record previously held by Pamela Jelimo with a time of 1:54.26. Pamela Jelimo, a Kenyan middle-distance runner, won the gold medal in the 800m event at the 2008 Olympics and was the first Kenyan woman to win an Olympic gold medal. Semenya also set new records in the 1500m and 1000m events, breaking the South African records previously held by Zola Budd and Ilse Wicksell, respectively. She is currently aiming to qualify for the World Athletics Championships in Oregon and the Commonwealth Games in Birmingham in 2022, with the qualification mark for the women's 5000m event set at 15:10.00.

As a system developer, creating a card that lists the time Semenya should reach each kilometre if she maintains her set target pace would be beneficial. This would allow Semenya to monitor her pace during long races and adjust accordingly if she is running too fast or too slow. By doing so, she can continue to break records and achieve her goals.

Start by writing the HTML code that produces the below picture:



- Create a CSS stylesheet that applies a pink background colour to the legend element of the pace runner chart. As a developer, write a JavaScript function that adds the following behaviour to your web page's pace runner chart:
  - The pace runner chart includes fields for entering a distance (in kilometres) and target pace (in minutes per km), as well as a table for displaying the runner's pace chart.
  - Upon initial load of the page, the table should be hidden and only the header row with column names should be displayed.
  - When the runner clicks the calculate button, the table should be populated with data and displayed.
  - Each row in the pace chart table should include a kilometre number and the time at which the runner should reach that kilometre, each in its own table cell.
  - Since performing arithmetic on times is difficult, your code should utilize a provided function that will multiply times for you.
  - Note: Please provide the function name and the provided time multiplication function for me to help you better.

```
// returns a 'mm:ss' or 'hh:mm:ss' time string representation
```

```
// the given time "mytime" multiplied by the given factor "myfunction" function
```

```
Calculate_time(mytime, myfunction) {
```

```
...  
}
```

Create the Calculate\_time function and implement it to perform necessary calculations.

- Include a final row in the pace chart table for the exact distance value entered in the distance field, even if it is a non-integer value such as 25.5km.
- When the calculate button is clicked after a pace chart has already been generated, ensure all previous data is cleared before adding new data to the pace chart table.
- Always retain the header row when generating the pace chart, as it should never be removed or regenerated.
- All input values must be valid and handled appropriately, with reasonable values entered in the distance and pace fields.
- Inject content into the page using the innerHTML property for plain text and ensure the prototype JavaScript library is loaded before your script is loaded.

## QUESTION 2

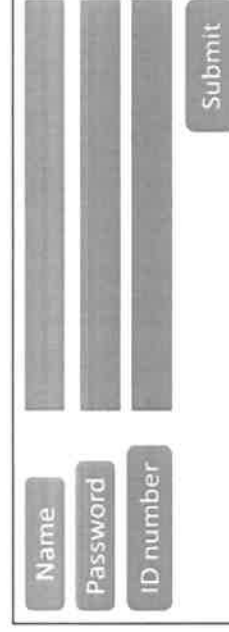
[25 MARKS]

30% of online users have been victims of security breaches caused by weak passwords. 88.6% of respondents use two-factor authentication. Password managers and cyber security software are great solutions for managing passwords and preventing unauthorized data leaks, especially for people with less cybersecurity. In 2020, Gauteng's local government website fall prey to hackers. The department has expressed concern that some personal information may have been accessed and sent outside of the organisation, saying it was in the process of establishing the exact nature of data that had been accessed.

Data protection requires a strong password and security. You now work as a cybersecurity engineer for the Gauteng Department of Justice. You are required to build a simple system that can validate the given ID number used for customer registration.

Write PHP code that can process the following HTML form:

```
<form action=" protectaccess.php" method="post">  
<fieldset> <input type="text" name="name"/>  
Name <br /> <input type="text" name="pw" /> Password <br />  
<input type="text" name="IDnumber" /> ID number <br/> <input type="submit" />  
</fieldset>  
</form>
```



The image shows a web form with three input fields and a submit button. The first field is labeled 'Name', the second 'Password', and the third 'ID number'. To the right of these fields is a button labeled 'Submit'.

Your PHP code should be able to do the following:

- Consider the CSS styling as shown in the above sample picture of the HTML form.
- Examine the name, password, and ID number submitted, and verify that they are valid as indicated in the output results below.
- The registration platform must capture the name, password, and security ID number.
- When the user hit the submit button, the platform should perform the validation of each of the inputs.
- A valid name is not an empty string and cannot be numbered.
- A valid password is any string that is at least 10 characters long with a mixture of numbers and characters.
- A valid ID number contains exactly 12 digits.
- The ID number can contain dashes but not dots between groups of three digits.
- No other characters may be part of an ID number.

For example, the following are some examples of valid and invalid ID numbers:

Valid	Invalid
123456786544	2457.1543.5676
245-154-873-888	foo123456doo
394850980323	123.4567810.
983 434 259 900	1234-5678:8986
111222333444	102:304:506:658

The PHP code that you write should output the following:

- Level 1 heading stating whether the data was Successful or Access Denied Invalid Data.
- Successful must be coloured with green and Access Denied Invalid Data in red.
- Paragraph containing the details separated by commas as shown below.
- The password should be replaced by a string of (\*) characters of equivalent length.
- Remove any dashes or dots out of the ID number while displaying the password with a string.

For example, some outputs of your PHP script for form input:

Form Input	Output
name: Bontle password: Bon123!	Successful.
ID: 245-154-873-888	Bontle, *****, 245154873888
name: John password: mmm@430	Access Denied! Invalid data.
ID: 111/112/222/	John, *****, 111112222
Name: Thabang password: Thab\$\$066	Access Denied! Invalid data.
ID: foo123456doo	Thabang, *****, foo123456doo

Form Input	Output
name: 12345 password: Bon123!	Access Denied! Invalid data.
ID: 123-456-786	12345, *****, 12345678
name: John password: mmmmm	Access Denied! Invalid data.
ID: 1111122222	John, *****, 111112222
Name: Thabang password: \$\$066	Access Denied! Invalid data.
ID: foo123456doo	Thabang, *****, foo123456doo

Use regular expressions for pattern matching, validation, and replacement. HTML content is stored in the file `protectaccess.html` and some resulting content is stored in the file **`accessresults.txt`** both of which you must place into your page.

### QUESTION 3

[25 MARKS]

To keep up with demand, restaurants of all sizes need to provide solutions that let customers browse their menus, select options, and pay for their carryout or delivery orders online. Several online ordering systems for restaurants are available in the market that help you create a customized ordering experience on your restaurant website or mobile app. They also allow restaurants to place QR codes around their restaurant for a contactless digital menu experience that takes diners to a restaurant's online ordering system. To help you in your search for the best solution for your restaurant, we have highlighted the top 11 online ordering system options to consider. One best systems are called the **GEU ordering system**. Customise your **GEU** ordering system by writing a PHP code for two partial web pages for ordering food from **GEU** online store.

Food item	<input type="text" value="milk"/>
Quantity	<input type="text" value="10"/>
<input type="button" value="Order"/>	

- The first page you will write is a form named **geuorder.php** that allows the user to choose what kind of food to buy and how many to buy. In your form, include a drop-down menu of food items available to purchase.
  - Include a text box for the user to select or enter a quantity of the item to purchase and an "Order" button to submit the form. (3 marks)
- The food items to list should be based on what JPG food images are available in the blackboard on the resource directory or folder. For example, if the food items contain banana.jpg and chicken.jpg, then "banana" and "chicken" appear in the drop-down list.
- Now, you can write the last portion of **geu\_order.php** that would appear between, `<body>` and `</body>`, you may write your own CSS if need may be.
  - The second page you need to write is named **geu\_submit-order.php**. The form in **geu\_order.php** submits its data as a POST request to **geu\_submit-order.php**. (3 marks)
  - The output of **geu\_submit-order.php** is an HTML page fragment, a single paragraph indicating information about the order as described below. (3 marks)
- **NB: Do not use print or echo statements.**
  - The GEU store's current inventory is stored in a file named **geu\_system\_inventory.txt** on the server in the current directory. Each line of the file represents one item in the store, its quantity, and its price per unit, separated by tabs. (3 marks)

Create the file and make sure that its contents are valid, and that there are no duplicates. Here is an example inventory:

apple	4	5.00
chicken	2	3.25
cookie	38	1.25
milk	9	4.50
tomato	27	2.50
Banana	20	2.00
Tinfish	8	10.0

In general, your task is to look up the price per unit of the item the user is ordering and use this price to compute the total order cost. For example, if the item costs R5.50 and the user orders 7 of them, the total order is  $7 * 5.50 = \text{R}38.50$ . (2 marks) The page's output in the general successful case is to inform the user that the order was successful, display the total order cost (you do not need to round it), and show the user a series of images representing what was ordered. For example, if the user orders 4 apples, your output should display 4 copies of apple.jpg.

**Order Successful R20 is your total price. Each apple is R5. Here is what you ordered {Display pictures of 4 apples} (3 marks)**

Here is another output from ordering 15 of the item cookies. (The output wraps to the next line in the browser) The store is only able to complete an order if that food item is in the geu system inventory and the store has enough of that item in stock.

**Message: Order Successful R20 is your total price. Each cookie is R1.25. Here is what you ordered {Display pictures of 15 cookies} (3 marks)**

For example, if the inventory matches the above text file (which has 9 “tinfish” in stock), and the user tries to order 10 of “tinfish”, the following error should be displayed: The items in this GEU system inventory may overlap with the images that were listed previously, but there might be some images that do not have representation in the inventory text file and vice versa. If an item is not present in the inventory file, assume that its quantity available in stock is 0 and display the same sort of error message.

**Error Message: Sorry we do not have 10 milk in stock, we only have 9 milk in Stock (3 marks)**

You do not need to modify the `geu_system_inventory.txt` file or update the quantity of the item being ordered. If the food item or quantity parameters are not passed, issue an error. If they are present, you may assume that the food item value passed is a string and the quantity value passed is a positive integer.

NB: Please resize all the pictures provided accordingly and save the test information into a new file called **results.txt**. (2 marks)

## QUESTION 4

[25 MARKS]

Suppose a web service named **flights.php** is located on your web server in the same directory as your code. This service outputs file data describing flights between various cities. In this problem, you will write PHP code to contact the web service (using a GET request), examine its file data, and display a list of possible flight prices and carriers. You are also requested to use JavaScript where possible.

The page contains two textboxes where the user can specify the departure and destination, a check box that they can check if they want to only see non-stop flights (flights with 0 stops) and a “Go!” button. When the button is pressed, you should send a PHP request passing the “start” parameter. You can assume that the user has typed a valid location into each box before pressing the button. The file data returned by the web service consists of a list of destinations, each of which has a list of flights associated with it. The list of flights contains lists which each contain a price, a carrier and the number of stops.

Edinburgh:

start: Seattle flights:

carrier: Delta, price: 812, stops: 2  
carrier: Air France, price: 1020, stops: 0  
carrier: Air France, price: 1190, stops: 3

New York:

start: Seattle flights:

carrier: British Airlines, price: 782, stops: 1  
carrier: Delta, price: 1562, stops: 2  
carrier: United price: 957, stops: 1  
carrier: KLM, price: 687, stops: 3  
carrier: KLM, price: 1458, stops: 1

The relevant existing HTML on the page is the following:

```
<div>
<label>Departure: <input type="text" id="start" /></label>
<label>Destination: <input type="text" id="dest" /></label>
<label>Non-stop? <input type="checkbox" id="stops"></label>
<button id="go">Go!</button>
</div>
<div id="results"></div>
```

When clicking the “Go!” button, clear previous results and read the file data with PHP.

- Add an h1 to the results div containing the text “**Flights from**” and then the start and destination locations.
- Turn each flight’s data into a paragraph in the results div. In each paragraph, write the price of the ticket (with an “\$”) followed by the word “from”, the carrier’s name and then the word “with”, the number of stops and the word “stops”.
- If the flight has a price below \$1000 display its row in bold.
- For the example in the file shown above, the page is shown twice below, once with only non-stop flights and once with all flights. “from”, is the carrier’s name and then the word “with”, is the number of stops and the word “stops”.
- If the flight has a price below \$1000 display its row in bold.

For the example in the file shown here, the page is shown twice below, once with only non-stop flights and once with all flights



Start location:  End location:  Non-stop? ☐ Go!

## Flights from seattle to Edinburgh

\$812 from Delta with 2 stops

\$1020 from Air France with 0 stops

\$1190 from Air France with 3 stops

Start location:  End location:  Non-stop? ☒ Go!

## Flights from seattle to Edinburgh

\$1020 from Air France with 0 stops

- **NB:** All flights prices must be converted to **South African Rands** only when you display the information of the search.

You may assume that the file data is valid in the format described previously, the data typed into the text boxes are valid, and that the “.php” service is reachable. You may not use any JavaScript libraries such as Prototype and JQuery.

