FACULTY/COLLEGE		College of Business and Economics	
SCHOOL		School of Consumer Intelligence and Information Systems	
DEPARTMENT		Applied Information Systems	
CAMPUS(ES)		APB	
MODULE NAME		Development Software 2A	
MODULE CODE		DSW02A1	
SEMESTER		First	
ASSESSMENT OPPORTUNITY, MONTH AND YEAR		Graded Lab 5 May 2022	
ASSESSMENT DATE	04 May 2022 @ 15:00	**SUBMISSION DATE**	05 May 2022 @ 23:59
ASSESSOR(S)	Mr. Ronny Mabokela		
DURATION	1 day	**TOTAL MARKS**	40
NUMBER OF PAGES OF QUESTION PAPER (Including cover page)			3

INFORMATION/INSTRUCTIONS:

- ✓ Read instruction carefully to avoid making mistakes.
- ✓ This is an open-book assessment.
- ✓ There are 2 questions and answer all questions.
- ✓ Submit the necessary files (xxx.html, xxx.css) for the codes.
- ✓ Read the questions carefully and answer only what is required.
- ✓ Make sure that your submission is in a zipped.

QUESTION 1 (PHP) [20 MARKS]

1.1. A palindrome is a word or phrase that is identical forward or backward, such as the word “racecar.” A standard palindrome is similar to a perfect palindrome, except those spaces and punctuation are ignored in a standard palindrome.

For example, “Madam, I’m Adam” is a standard palindrome because the characters are identical forward or backward, provided you remove the spaces and punctuation marks.

Write a script that checks words or phrases stored in two separate string variables to determine if they are a perfect palindrome. You can modify the program to check for standard palindromes.

- Save the perfect palindrome script as ***PerfectPalindrome.php*** and the standard palindrome script as ***StandardPalindrome.php.***

1.2. Write a PHP program that checks the elements of a string array named **$Passwords**. Use regular expressions to test whether each element is a strong password. For this exercise, a strong password must have at least one number, one lowercase letter, one uppercase letter, no spaces, and at least one character that is not a letter or number. (Hint: Use the [**^0-9A-Za-z**] character class.)

The string should also be between 8 and 16 characters long. The **$Passwords** array should contain at least 10 elements, and at least six of the elements should fail. Ensure that one entry fails each of the five regular expression tests and that at least one fails because of the length.

Display whether each password in the **$Passwords** array was strong enough and display the test or tests that a password failed if it is not strong enough. Save the script as ***PasswordStrength.php***.

QUESTION 2 (PHP) [40 MARKS]

Write a complete PHP web service **Ama2000.php** that processes the data containing the names of the babies which are born early 2000’s. The PHP service reports the best (lowest) popularity ranking that name has ever held in the South African Home Affairs database.

Create a service which accepts a GET request parameter called name with the following cases:

- Its output is a single line of plain text containing the best ranking number, or -1 if no name parameter is passed or if the name is not found in the file.
- The input file to read, **ama2000.txt**, is in the below format, with each line containing a baby's first name followed by some number of popularity rankings.

The sample data below has 11 rankings per line, but for full credit your code should work regardless of how many rankings (1 or more) are on each line assigned to a name.

Bontle 31 25 24 26 30 46 93 163 209 289 382
Bonang 66 79 84 94 93 78 70 108 127 142 177

Bohlale 631 752 712 664 720 933 636 752 680 856 918
Martha 44 55 16 66 898 977 98 567 38 89 90 128 678 890
Martin 69 58 34 68 45 76 28 78 98 90 78 87 84 83 85 97
Mart 55 88 09 87 54 56 78 65 78 65 76 34 76 89
.....

For example, if your service were requested as **Ama2000.php?name=Bontle**, its output would be **24**, since that is Martha's best popularity ranking.

- If the request is **Ama2000.php?name=Bonolo**, its output would be -1 since that name is not in the file.
- If the request is **Ama2000.php?name= 66**, its output would be -1 since that name is not in the file.
- If the request is **Ama2000.php?name= B**, its output would be -1 since that name is not in the file.
- Assume John is the file has ranking of 88 16 34 16 590 67 265 987 657 87 545, If the request is **Ama2000.php?name= John**, its output would be 16 since the ranking 16 appears twice in the file.

Ensure that your code matches case-insensitively; for example, the request **Ama2000.php?name=BoHLalE** should match the name Bohlale in the data file.