Government of Western Australia
North Metropolitan TAFE

Portfolio Assessment Task 2

Qualification national code and title	ICT50220 Dip Advanced Programming
Unit/s national code/s and title/s	ICTPRG535 – Build advanced user interfaces ICTPRG547 – Apply advanced programming skills in another language

Assessment type (☑):

☒ Questioning (Oral/Written)

☐ Practical Demonstration

☐ 3rd Party Report

☒ Portfolio

Assessment Resources:

Python3 interpreter.

Python IDE, like PyCharm, or VSCode (the latter is not supported by the college), with the ability to use Python Virtual Environments.

Access to Office 365 and Microsoft Word.

Git and access to GitHub.

Use of some of these items may not occur in this part of the assessment task.

Assessment Due

This item is due:

- **Week 7, 17:00 (5pm) on the day of the scheduled lecture**

Refer to Blackboard for most accurate dates, which may alter due to unforeseen circumstances. We also endeavour to update these documents at the same time.

It is advantageous for you to attempt to meet this deadline.

Assessment Instructions:

Portfolio Assessment Task 2

Qualification national code and title	ICT50220 Dip Advanced Programming
Unit/s national code/s and title/s	ICTPRG535 – Build advanced user interfaces ICTPRG547 – Apply advanced programming skills in another language

Scenario

You are employed as a junior software developer at a boutique software house called *Softwares-R-Us* in Perth.

You have been assigned to the Games team and will provide some of the code required for various games that the company builds and releases.

You will be updating the existing code from the previous task to store player passwords in a secure manner. You will be using hashing to accomplish this.

Instructions

Your code must adhere to certain style guides, the most important being PEP-8 – Style Guide for Python Code. You should familiarise yourself with PEP-8 before continuing as SRU have adopted PEP-8 as their code style.

Use of a Git repository is standard practice at *Softwares-R-Us* and most of the steps require the use of Git and GitHub.

There are multiple steps in this task, and you must perform each step to a satisfactory level. Please note that you'll need the results of the tasks outlined in this assessment tool for future tasks as well.

You may answer any questions in the provided template (if available) and the use of screenshots is encouraged. However, you must also provide the actual source code as a ZIP-file of the project for any programming tasks. *Make sure to remove the Virtual Environment folder (*`venv` *or* `.venv`*) from the ZIP-file before uploading.*

You must document your code properly. Use docstrings where needed including but not limited to entire classes and methods. Use inline comments to clarify certain parts of code.

If you use any external resources, you should provide references.

Assessment Instrument:

Portfolio Assessment Task 2

Qualification national code and title	ICT50220 Dip Advanced Programming
Unit/s national code/s and title/s	ICTPRG535 – Build advanced user interfaces ICTPRG547 – Apply advanced programming skills in another language

Step 1 – Knowledge Question (40-70 words)

In your own words, describe what hashing is in general.

Step 2 – Knowledge Question (60-100 words)

Research hashing algorithms. Describe advantages and disadvantages for at least three different hashing algorithms. Please provide references for external resources.

Step 3 – Knowledge Question (50-90 words)

Provide a stepwise description (algorithmic) of a) how you can store a password safely using hashing techniques and b) how you can verify that some string is the right password?

Step 4 – Knowledge Question (20-40 words)

What is the purpose of a "salt" when hashing a password? What are the two most important properties of a "salt"?

Step 5 – Add password to Player class

In this step, you will be adding functionality to the `Player` class to store a password in a safe manner. This step has multiple parts to it.

a. Install the external package `argon2-cffi` from pypi using pip or your IDE. *Document how you installed this package.*
b. Read the documentation for this package to understand what this package has to offer.
c. Add a method `add_password` to the `Player` class. It should accept a single argument (a string), which is the plaintext password. Determine which function to use from the argon2 package and implement the function to calculate a *hashed* version of the password. Store this value in a private instance variable. Do *not* create a property for this value. You may have to initialise this instance variable in the initialiser method.
d. Add a method `verify_password` to the `Player` class. It should also accept a single argument (a string), which is the plaintext password which should be checked, and return a `Boolean` indicating whether there is a match. Implement this method to verify that the provided password matches the stored password.
e. Create at least two unit tests to check whether your implementations work correctly.
f. Commit your changes and push to your remote GitHub repository.
g. Answer the question "How does the `argon2-cffi` package handle salt?"

Portfolio Assessment Task 2

Qualification national code and title	ICT50220 Dip Advanced Programming
Unit/s national code/s and title/s	ICTPRG535 – Build advanced user interfaces ICTPRG547 – Apply advanced programming skills in another language

Submitting your work

ZIP your entire project into a single file.

Make sure to *remove the Python Virtual Environment folder (called* `venv` *or* `.venv`*) from the ZIP-file* before uploading it into the Blackboard assessments area.