GRIFFITH UNIVERSITY

Week 10 (3 marks)

Part A: Lab Tasks (Don't submit this part)

Tutor will demonstrate this part.

Objective:

- Run Bandit on Python scripts.
- Understand the meaning of four common Bandit findings (B101, B105, B110, B112).
- Explain the security risk behind each finding
- Apply secure coding practices to remediate issues

Introduction

What is SAST? - Static Application Security Testing

What is Bandit? - Python-focused SAST tool

Common Security Issues:

1. **Assert Misuse (B101)**
    - **Using assert for security or input validation.**
    - **Risk: assert can be disabled in production, leading to skipped checks.**
2. **Hardcoded Secrets (B105)**
    - **Passwords written directly in code.**
    - **Risk: Exposed credentials compromise system security.**
3. **Silent Exception Handling (B110, B112)**
    - **try: ... except: pass or except: continue.**
    - **Risk: Errors are ignored, bugs and attacks go undetected.**

Activity 1: Activate 2810ICT and Install bandit

open a terminal window (or Anaconda prompt window), and activate 2810ICT environment

```
> conda activate 2810ICT
> conda install bandit -c conda-forge
```

Activity 2: Hands-on Activity

Students work in pairs on the provided **insecure_examples.py** file.

Student tasks:

1. Run Bandit, capture the output.
   ```
   > bandit insecure_example.py
   ```
2. Identify the four issues flagged.
3. Fix the code to remove warnings (rewrite).
4. Re-run Bandit to confirm "no issues".

Part B: Individual Assignment (To be submitted)

This assignment is designated as an independent individual task. Each student is expected to complete and submit their own work without assistance from others. If anything is unclear, you are encouraged to seek clarification from your tutor/lecturer. The objective is to assess your personal understanding of the course content and your ability to apply relevant project management concepts and techniques independently.

Files required for submission:

- Submit **a docx or pdf document** which covers only Part B
- Submit the source code, **secure_app_module.py**

Context

You will apply **Static Application Security Testing** to a small Python module that intentionally contains some issues. You will run bandit, identify the issues, remediate the code, and document your decisions.

Task 1: Run bandit and identify all issues

Run bandit and capture the output of the identified issues as a **screenshot**.

Task 2: Refactor to remove all issues

Create **secure_app_module.py** that preserves the intended behaviour but eliminates the issues. Re-run bandit on this secure implementation and capture a **screenshot** of the results for submission.

Please refer to the submission page for the Marking Rubric.