Development Individual Project: Coding Output and Evidence of Testing

# Introduction

The use of the School Management System is becoming more important especially after the covid-19 pandemic (Falana et al., 2021). Since school management systems handle sensitive academic records, it has to be designed in such a way as to protect the data stored with them (RSI Security, 2021).

This document contains a simple school management system written in Python, demonstrating its security and testing procedures. The screenshots of the software from Figure 1 to 13 shows evidence of what happens at the various points of running the software.

* **Evidence of Application.py:** The output of selecting – Run the application securely and logging in as an account with administrator and non-administrator privileges.

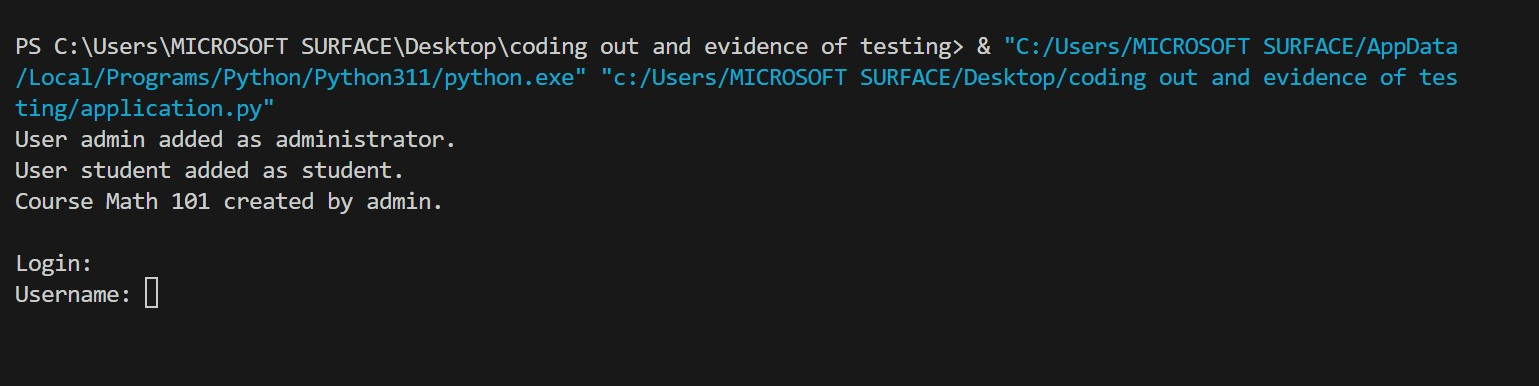


Figure 1: evidence of running the application for the first time

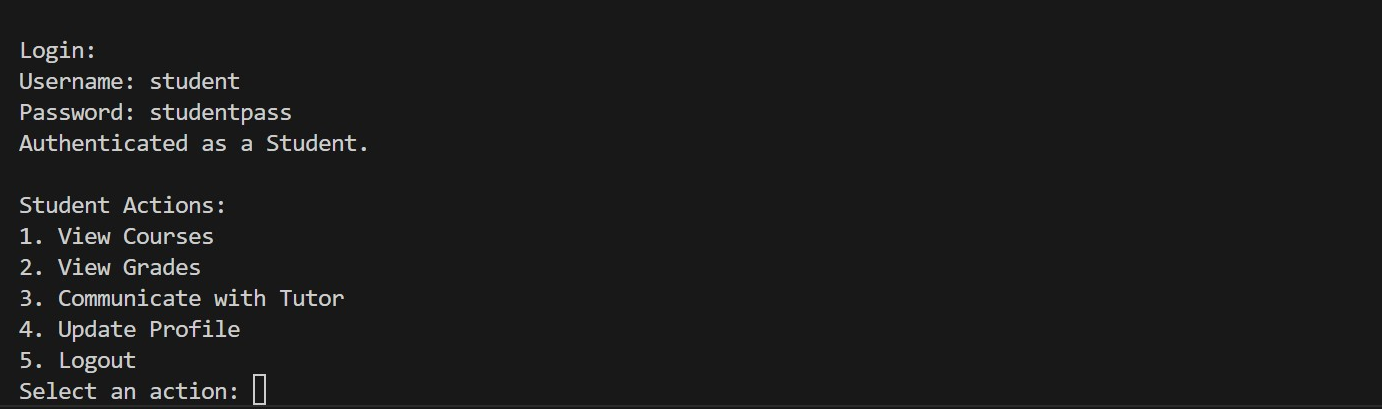


Figure 2: evidence of student login

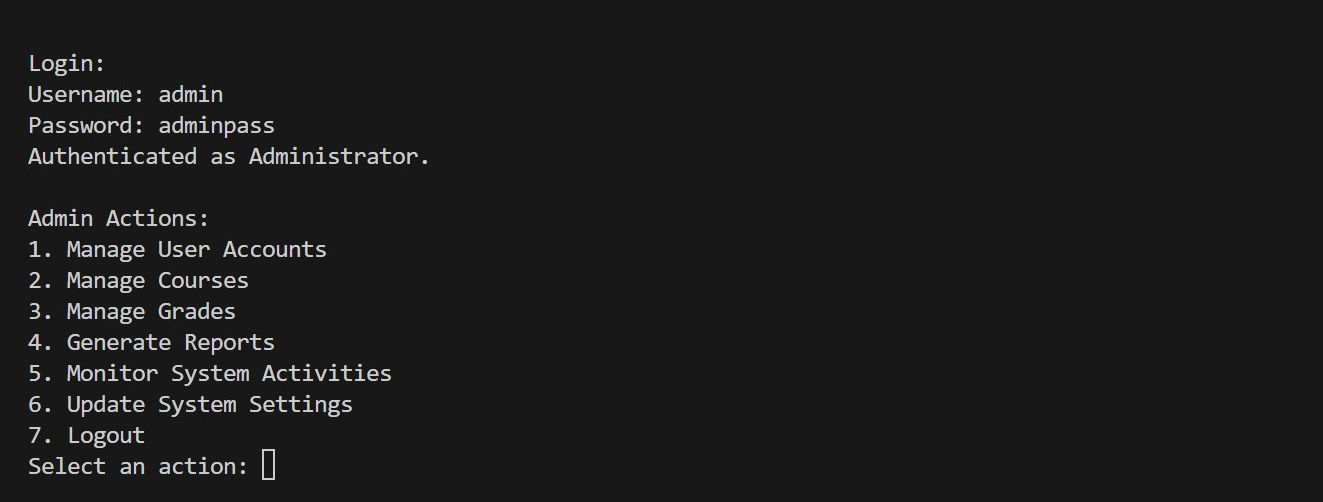


Figure 3: evidence of admin login

* **Api.py:** A user will be presented with options to run the software securely, insecurely or exit the application.



Fig 4: Evidence of prompt when API.py is run

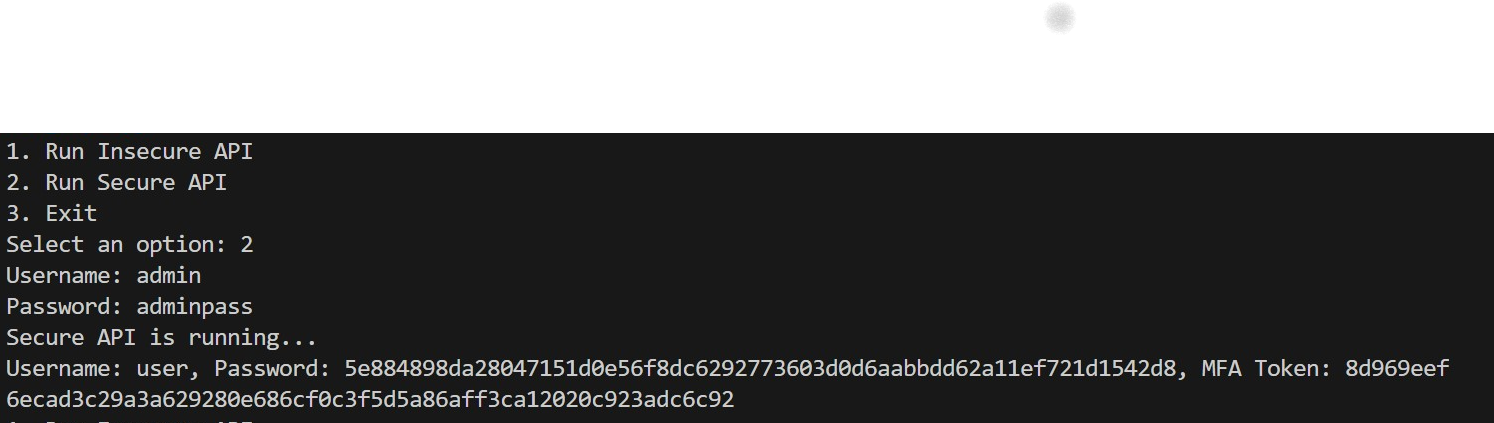


Fig 5: Evidence of running securely with administrator account



Fig 6: Evidence of running secure API with non-administrator account

* **Insecure\_api.py:**

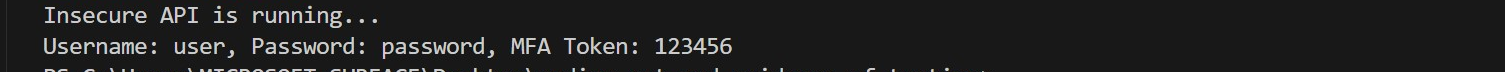


Fig 7: Evidence of running insecure\_api.py

* **Secure\_api.**py: When the Secure API file is started a user is presented with the server’s terminal output.
* Evidence of hashed password and MFA Tokens.

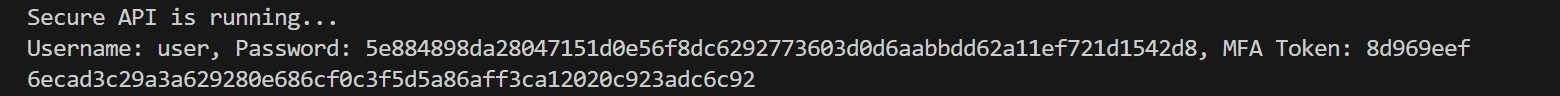


Fig 8: Evidence of running secure\_api.py with hashed password and MFA tokens shown

**Hacker Requirements:**

* A hacker should have the ability to carry out a brute force attack on the local network system
* A hacker should have the ability to carry out a Denial-of-Service attack on the local network system
* A hacker should be able carry out an API injection attack.

You must now provide evidence of brute\_force attack by calling a brute\_force.py file. Please note, correct credentials (username and password) are entered. Similarly, provide evidence of DoS with different scenarios like attacking computer prior to the attack, loading URL and IP address of the server before the attack,

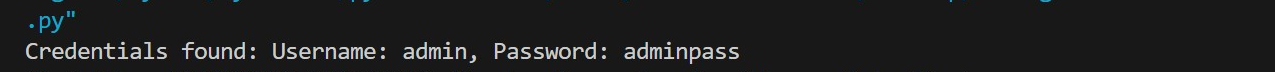


Fig 9: Evidence or running brute\_force.py

**API Injection:** Provide evidence of uploading a file without any validation regarding its size or type, which allows malicious files to be uploaded.

**Tests:** Flake8 – checks code against the best practices of the PEP8 coding style, Pyflakes, Pycodestyle, Pydocstyle, Pylama (Pylint, Radon), Unittest.

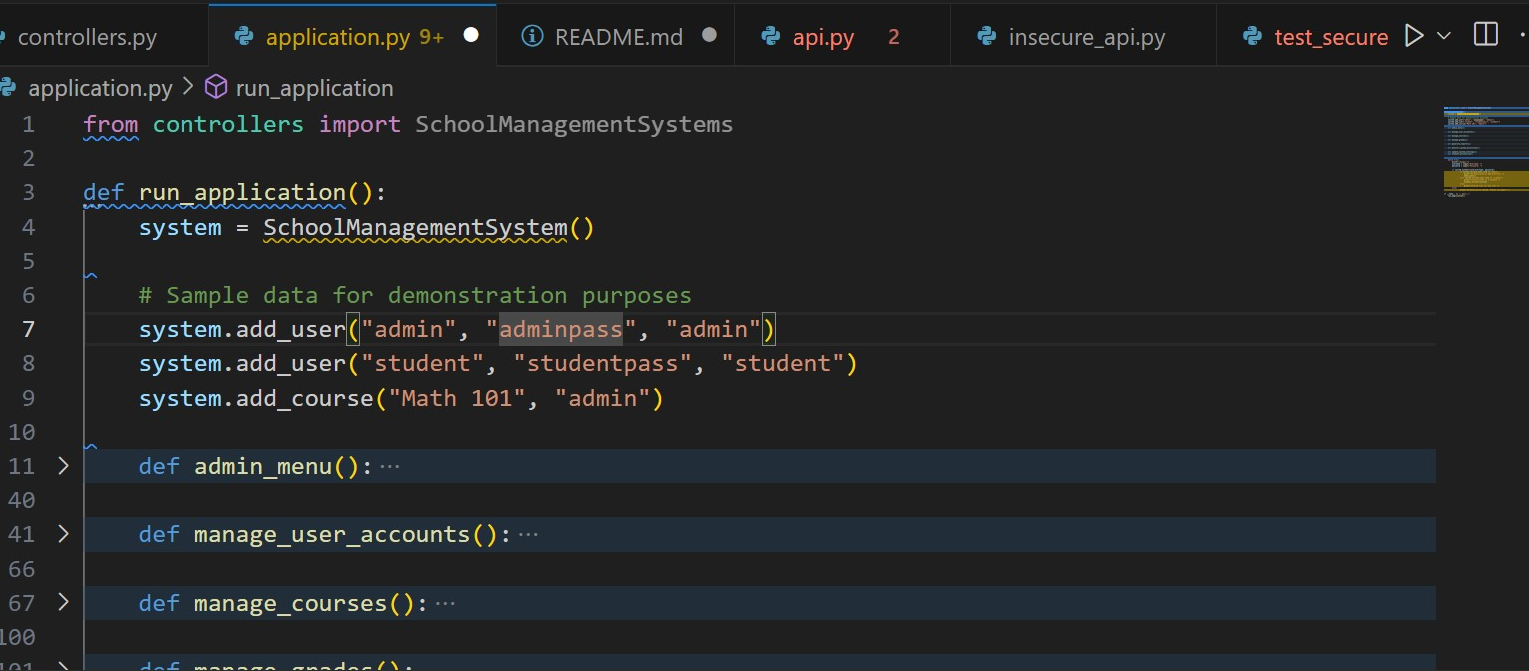


Fig 10: Evidence of error detected by pylint and pylance on line 4

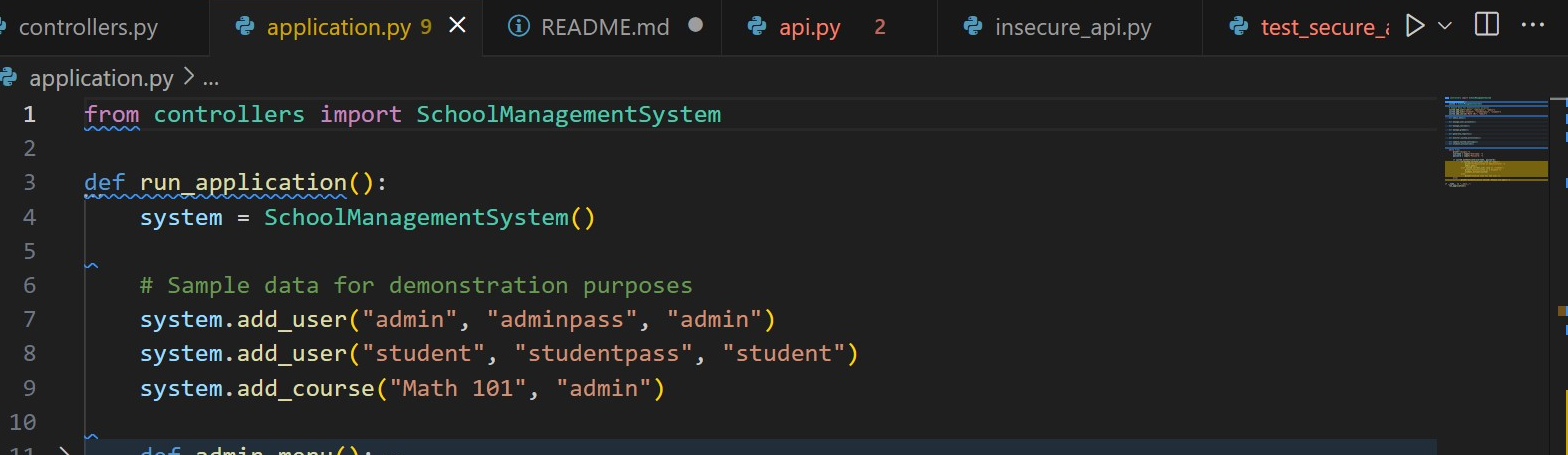


Fig 11: Error on line 4 disappears after removing the last ‘s’ on line 1

**Flake8:** Commented out a line in the api.py file incorrectly, which will be reported by Flake8. Provide evidence. Then correct the error and run the file, Flake8 will not found any error. Provide evidence.

Like Flake8, produce evidences for rest of the tests (mentioned above under the heading **Tests).**

**Tests**

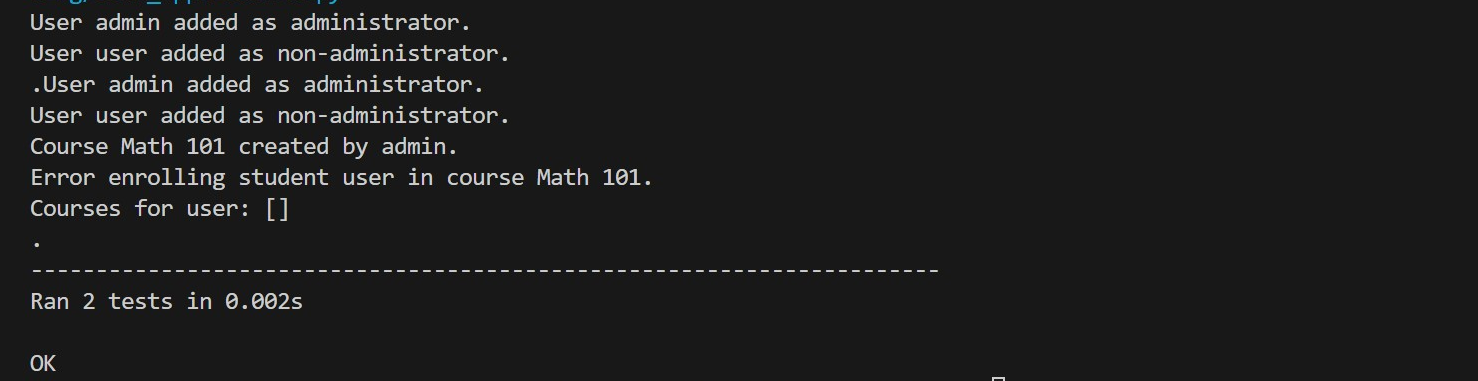
****

Fig 12: Evidence or running tests in test\_application.py

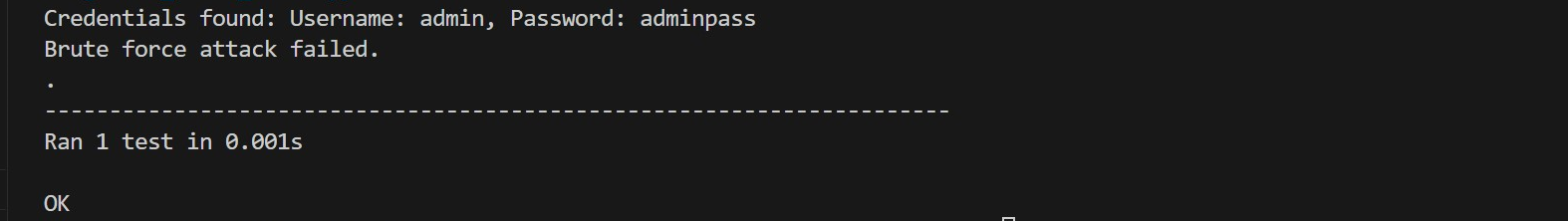


Fig 13: Evidence of running tests in test\_bruteforce.py

**Conclusion**

A good practice in building a useful school management system is in the implementation of the security features. This has been demonstrated in the various pieces of evidence in this document, by the use of static file analysis tools like pylint and the use of secure algorithms.

**Reference**

* Falana, O.J., Ebo, I.O. & Odom, I.S. (2021) Se-LMS: Secured e-learning management systems for smart school. International Journal of Software Engineering and Computer Systems, 7(1): 36-46.
* RSI Security, 2019. Common cybersecurity threats in education. [online] Available from: <https://blog.rsisecurity.com/common-cyber-security-threats-in-education/> [Accessed 10 August 2024].