Logo

Description automatically generated

**Software Quality Assurance (COMP 6710)**

**Prof: Akond Rahman, PhD**

**Project Report**

**Date: April 27th, 2023**

**Done By:**

**Logan Smith - lws0022**

**Satya Sai Teja Charla - szc2039**

**Table of Contents**

|  |  |
| --- | --- |
| Description | Page Numbers |
| Summary | 1 |
| Static Analysis | 2 |
| Fuzzing | 4 |
| Forensics | 6 |
| Conclusion | 8 |
| Reference | 8 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Summary:**

The objective of this project is to integrate software quality assurance activities into an existing Python project. Whatever we learned from our workshops will be integrated in the project by apply the following activities related to software quality assurance:

1. Create a Git Hook that will run and report all security weaknesses in the project in a CSV file whenever a Python file is changed and committed.
2. Create a fuzz.py file that will automatically fuzz 5 Python methods of your choice. Report any bugs you discovered by the fuzz.py file. fuzz.py will be automatically executed from GitHub actions.
3. Integrate forensics by modifying 5 Python methods of your choice.

**Project for Software Quality Assurance (CSC 5710/6710)**

**TeamLS-SQA2023-Auburn**

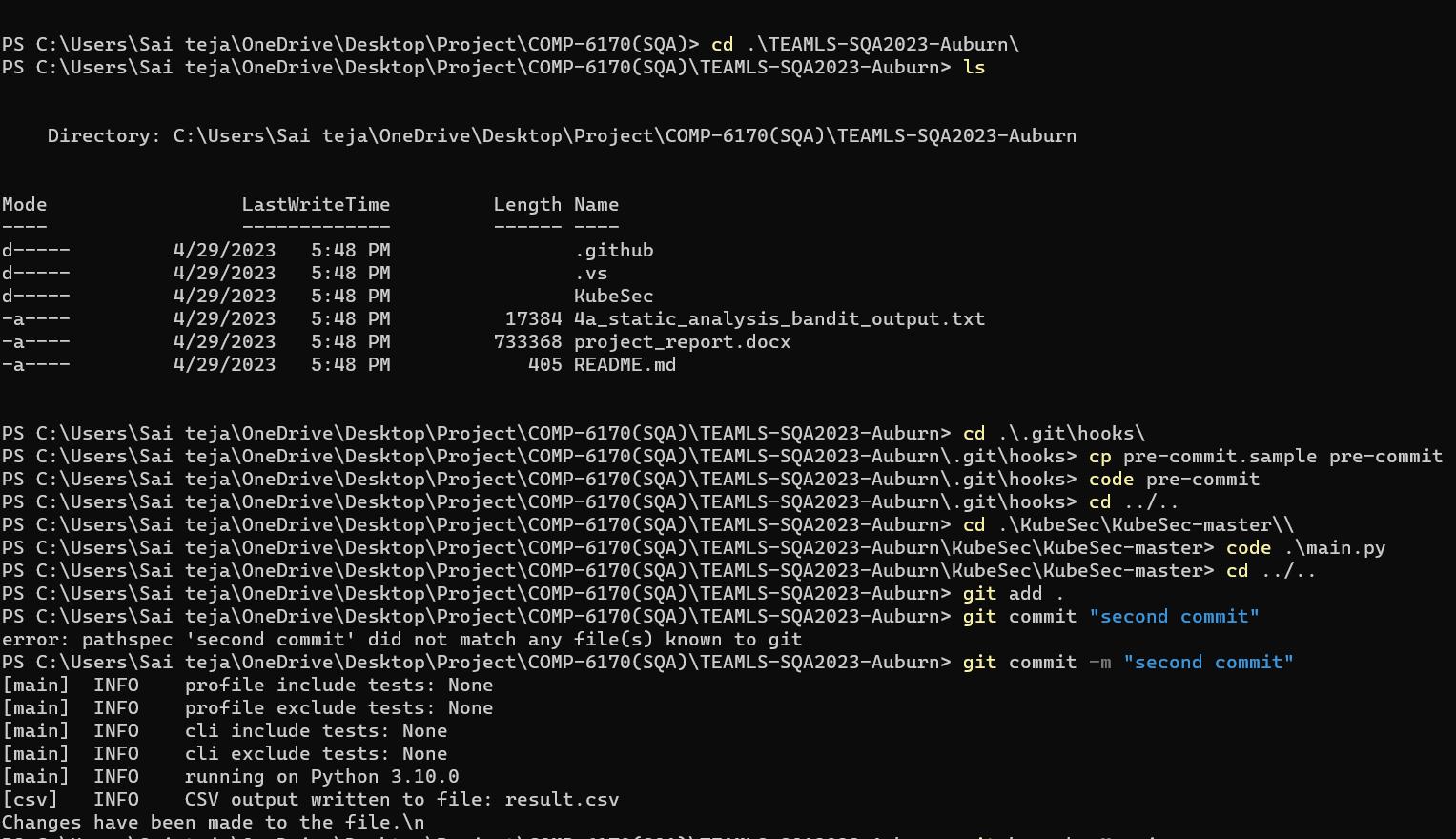
**1. Static Analysis:**

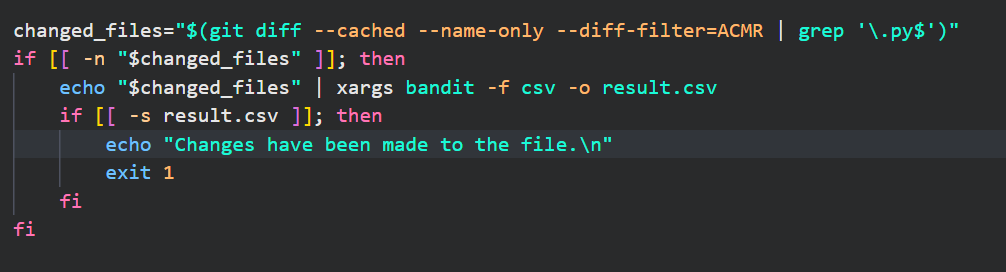
After creating a git-repo, I’ve cloned the repository onto my machine and made some changes to

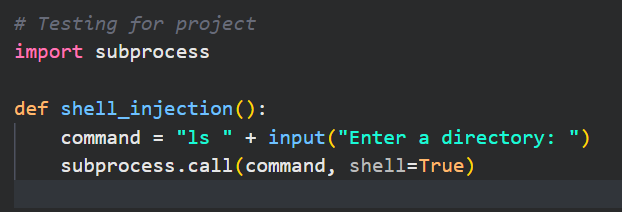
**‘./git/hooks/pre-commit.sample’** file by copying the contents in that file and created a new file named ‘**pre-commit’** after that I’ve modified the main.py file in the same repository such that to see the effects of the modified **pre-commit** hook.

Finally, I ran bandit -r command to see any security weaknesses in the provided file and recorded the output.

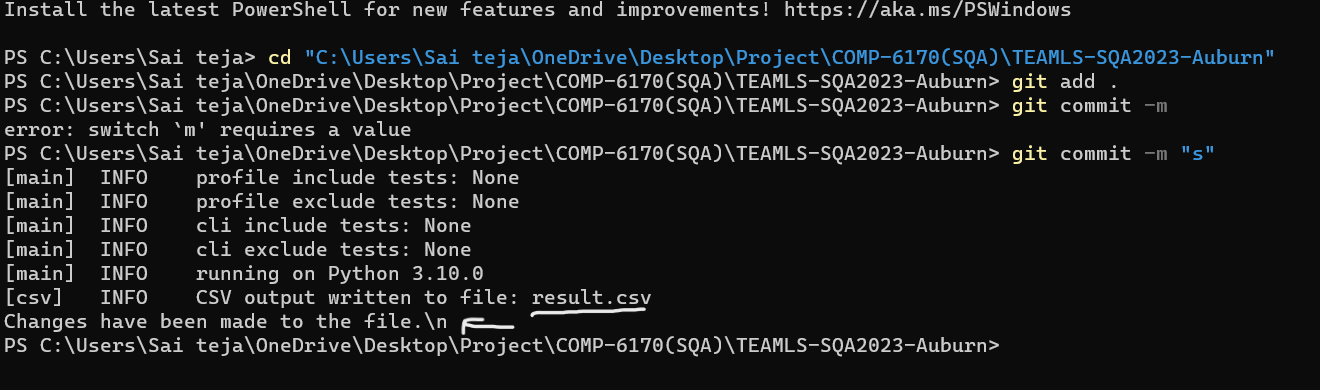
Down below are the screenshot taken during the execution of this segment:





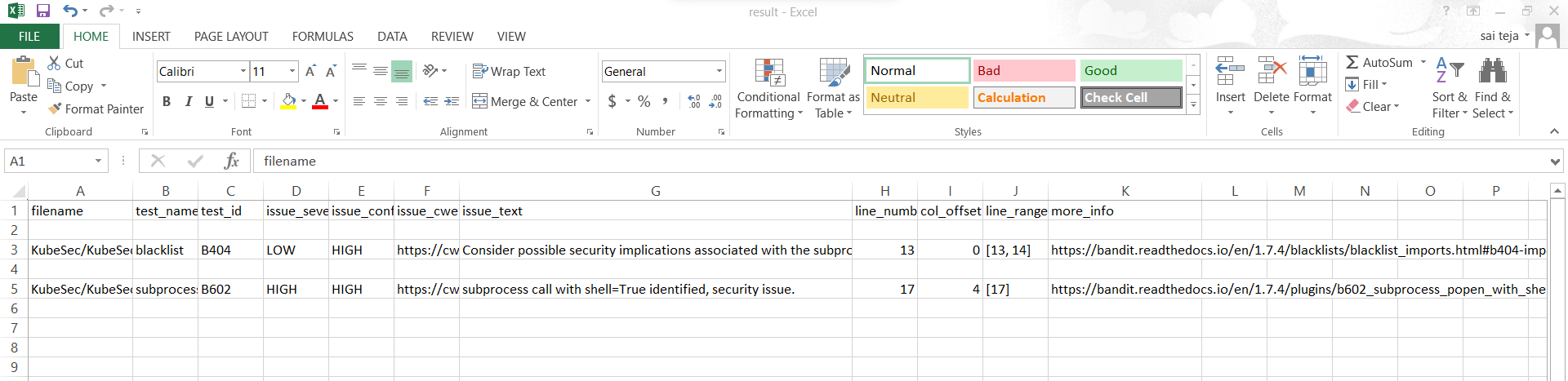


Now, when we use ‘commit’ on the terminal.



You can see that ‘result.csv’ has been created and it throws out the warning if any changes have been made to the files.

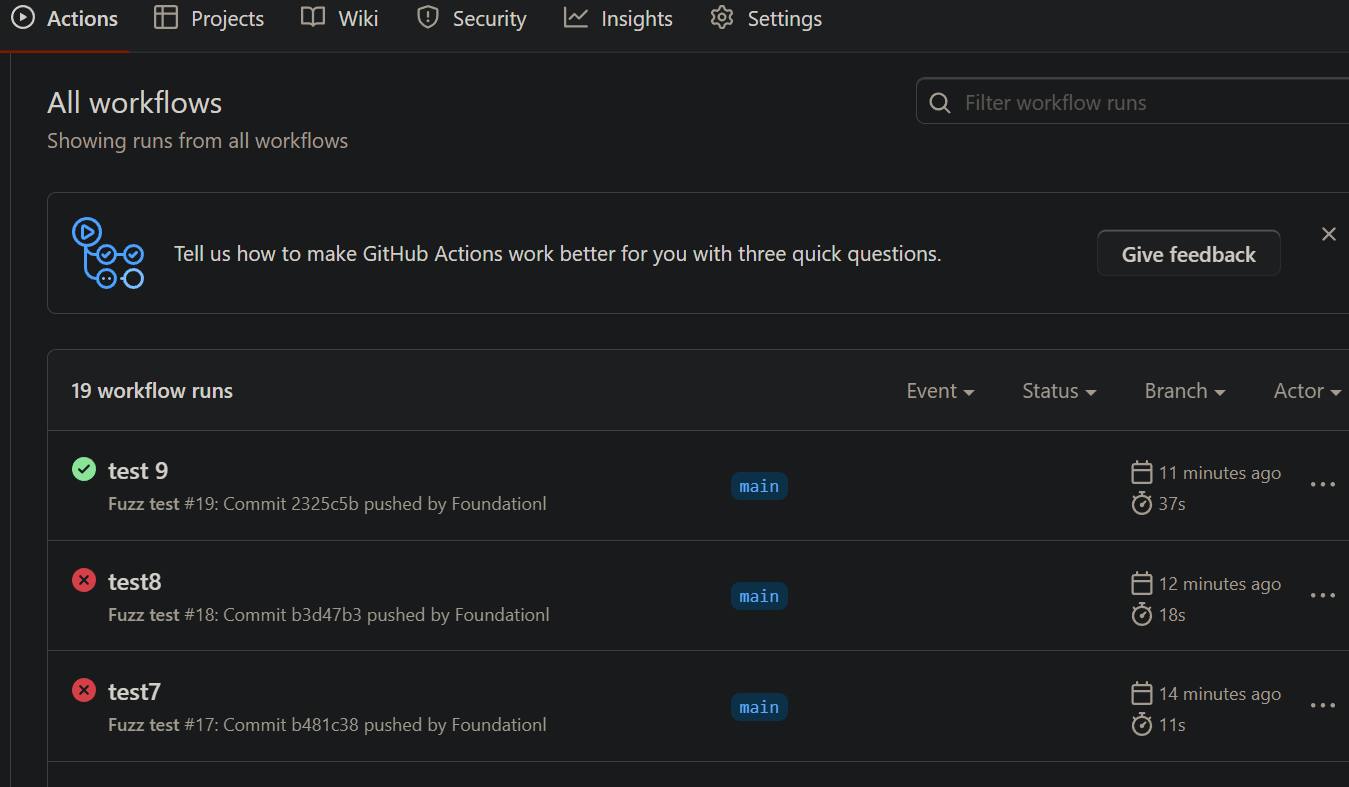
In that result.csv file, It records all the security weaknesses with filename, test\_id, issues ..etc.



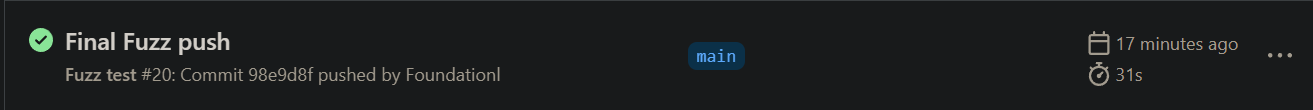
I’ve tried pushing the pre-commit file but it didn’t work, but I’ll be uploading the result.csv file and pre-commit file on canvas if needed.

**1. Fuzzing:**

After creating a git-repo, I’ve cloned the repository onto my machine and made some changes. First, I had to add a **‘.github/workflow/main.yml’** file so that on actions like pushing, the fuzzing will run on the 5 chosen functions and print a report. After several test and review iterations I was able to print a report from workflows.



I now was ready to set up my Fuzzing function. This involved finding 5 methods throughout the zip to test. These chosen methods where {Class Scanner [Function isValidUserName, isValidPasswordName, isValidKey], Class Parser [Function keyMiner, checkIfValidHelm]}. The inputs chosen where a random generated int, a random generated string of fixed size, and NULL. The Fuzz.py will test those 5 methods with these inputs. The fuzzing function would then print successful and unsuccessful tests in a report. The unsuccessful test will also display the error associated with the failure. When pushing to GitHub you can go to actions and find under workflows the latest commit and view report, for reference the final working Fuzz push.



In this workflow you will be able to see the fuzz report which prints the tests after first iteration.

**Fuzz.py output:**

Graphical user interface, text, application, email

Description automatically generated

A picture containing text

Description automatically generated

**3. Software Forensics:**

Created a simple python file which will log whenever a particular python method is used.

Text

Description automatically generated

After importing this file onto Fuzz.py file, to actually record it, first we need to use the following snippet of code.

logObj = logger.giveMeLoggingObject()

logObj.info( <Add the comment> )

After running that particular file, it will automatically generate a log file in the current directory.

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

**Conclusion:**

To summarize, my teammate and I have integrated software quality assurance activities into an existing Python project.

Static Analysis: Created a git hook which will report any security weaknesses detected from the files. If any changes are made to the existing files, all the security weaknesses in it will be reported in a CSV file.

Fuzzing: Created a python file called ‘Fuzz.py’ which will run on the 5 chosen functions and print a report. After several test and review iterations we were able to print a report from workflows.

Forensics: This last bit of project was easier to implement compared to others, but it is an important one.

Whatever we learned from our workshops has been integrated in this project.

**References:**

Canvas files - <https://auburn.instructure.com/courses/1478245/files>

StackOverFlow - <https://stackoverflow.com/>