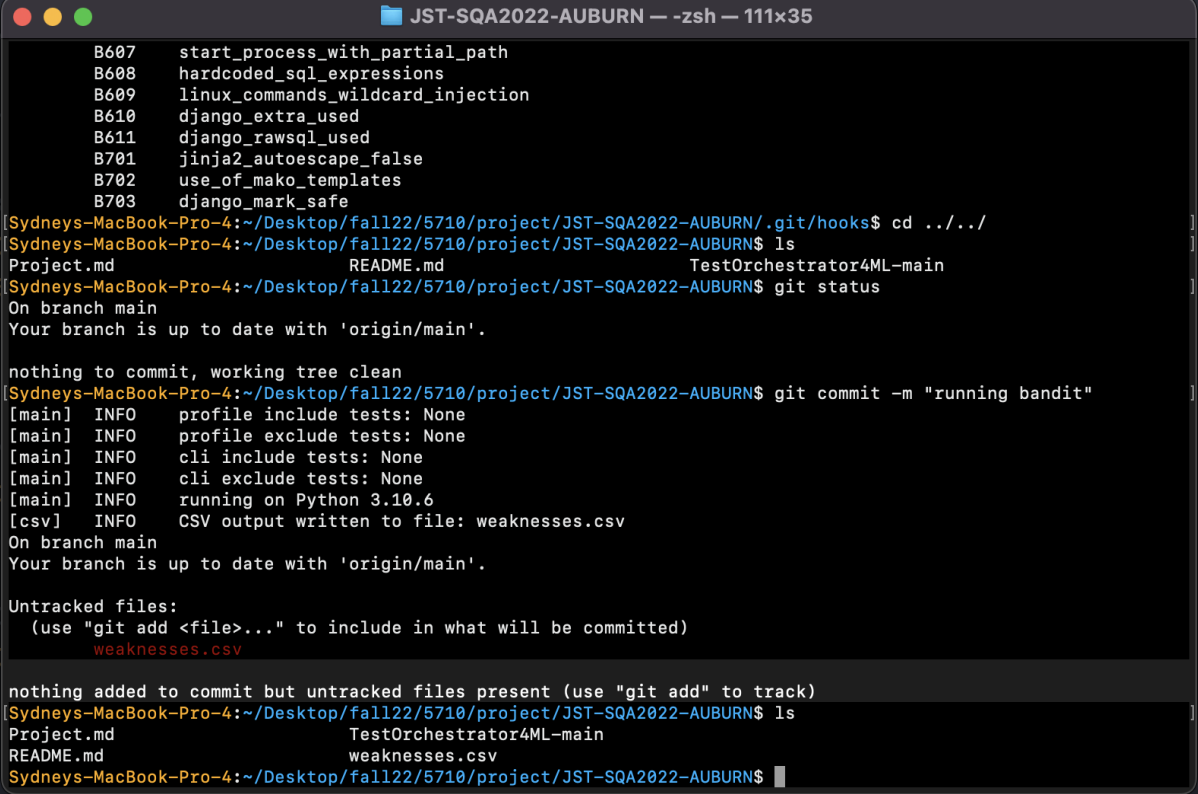


# SQA Project

Team Name: JST (Joshua Byers, Thomas Murphy, Sydney Carter)

## a. Git Hooks - Sydney

I created a pre-commit file that is executed automatically when a file is committed. It runs bandit to find security weaknesses in all python files. I used flags to save these weaknesses to a csv file named weaknesses.csv. From this project I learned how to use git hooks to automate certain tasks to run when committing or merging files, etc. This is a very useful tool especially when working with large projects since it will speed up development processes and allow me to automate specific tasks I need performed on many files at a time. The following is a screenshot showing the terminal output of pre-commit running:



```
JST-SQA2022-AUBURN - zsh - 111x35
B607 start_process_with_partial_path
B608 hardcoded_sql_expressions
B609 linux_commands_wildcard_injection
B610 django_extra_used
B611 django_rawsql_used
B701 jinja2_autoescape_false
B702 use_of_mako_templates
B703 django_mark_safe
[Sydney-MacBook-Pro-4:~/Desktop/fall22/5710/project/JST-SQA2022-AUBURN/.git/hooks$ cd ../../]
[Sydney-MacBook-Pro-4:~/Desktop/fall22/5710/project/JST-SQA2022-AUBURN$ ls
Project.md README.md TestOrchestrator4ML-main
[Sydney-MacBook-Pro-4:~/Desktop/fall22/5710/project/JST-SQA2022-AUBURN$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
[Sydney-MacBook-Pro-4:~/Desktop/fall22/5710/project/JST-SQA2022-AUBURN$ git commit -m "running bandit"
[main] INFO profile include tests: None
[main] INFO profile exclude tests: None
[main] INFO cli include tests: None
[main] INFO cli exclude tests: None
[main] INFO running on Python 3.10.6
[csv] INFO CSV output written to file: weaknesses.csv
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    weaknesses.csv

nothing added to commit but untracked files present (use "git add" to track)
[Sydney-MacBook-Pro-4:~/Desktop/fall22/5710/project/JST-SQA2022-AUBURN$ ls
Project.md TestOrchestrator4ML-main
README.md weaknesses.csv
[Sydney-MacBook-Pro-4:~/Desktop/fall22/5710/project/JST-SQA2022-AUBURN$
```

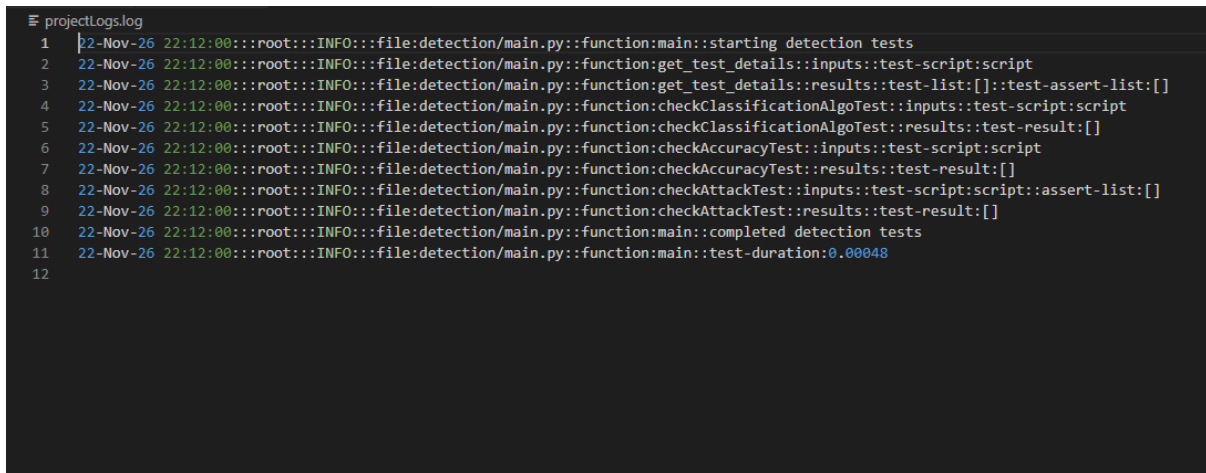
## b. Fuzzing - Thomas

I created a fuzz.py file, inside of which I created an array of several different values of different data types. After this, I imported 5 methods from the files given in the project. For each method, I ran a for loop with a try catch statement to catch any potential errors being thrown. In this I learned the different ways that fuzzing can be used to debug and find potential errors or vulnerabilities in code. It is important to test with many different inputs over several different types in order to ensure the quality of a method being tested.

```
(base) thomasmurphy@MacBook-Pro-136 JST-SQA2022-AUBURN % python3 fuzz.py
unsupported operand type(s) for -: 'str' and 'str'
unsupported operand type(s) for -: 'int' and 'str'
unsupported operand type(s) for -: 'dict' and 'str'
unsupported operand type(s) for -: 'int' and 'str'
unsupported operand type(s) for -: 'int' and 'str'
unsupported operand type(s) for -: 'str' and 'str'
object of type 'int' has no len()
object of type 'int' has no len()
object of type 'int' has no len()
object of type 'int' has no len()
object of type 'int' has no len()
object of type 'int' has no len()
Expected 2D array, got scalar array instead:
array=input.
Reshape your data either using array.reshape(-1, 1) if your data has a single feature or array.reshape(1, -1) if it contains a single sample.
Expected 2D array, got scalar array instead:
array=input.
Reshape your data either using array.reshape(-1, 1) if your data has a single feature or array.reshape(1, -1) if it contains a single sample.
Expected 2D array, got scalar array instead:
array=input.
Reshape your data either using array.reshape(-1, 1) if your data has a single feature or array.reshape(1, -1) if it contains a single sample.
Expected 2D array, got scalar array instead:
array=input.
Reshape your data either using array.reshape(-1, 1) if your data has a single feature or array.reshape(1, -1) if it contains a single sample.
Expected 2D array, got scalar array instead:
array=input.
Reshape your data either using array.reshape(-1, 1) if your data has a single feature or array.reshape(1, -1) if it contains a single sample.
Expected 2D array, got scalar array instead:
array=two.
Reshape your data either using array.reshape(-1, 1) if your data has a single feature or array.reshape(1, -1) if it contains a single sample.
Expected 2D array, got scalar array instead:
array=two.
Reshape your data either using array.reshape(-1, 1) if your data has a single feature or array.reshape(1, -1) if it contains a single sample.
Expected 2D array, got scalar array instead:
array=two.
Reshape your data either using array.reshape(-1, 1) if your data has a single feature or array.reshape(1, -1) if it contains a single sample.
Expected 2D array, got scalar array instead:
array=two.
Reshape your data either using array.reshape(-1, 1) if your data has a single feature or array.reshape(1, -1) if it contains a single sample.
Expected 2D array, got scalar array instead:
array=two.
Reshape your data either using array.reshape(-1, 1) if your data has a single feature or array.reshape(1, -1) if it contains a single sample.
Expected 2D array, got scalar array instead:
array=cars.
Reshape your data either using array.reshape(-1, 1) if your data has a single feature or array.reshape(1, -1) if it contains a single sample.
Expected 2D array, got scalar array instead:
array=77.
Reshape your data either using array.reshape(-1, 1) if your data has a single feature or array.reshape(1, -1) if it contains a single sample.
Expected 2D array, got scalar array instead:
array={'dictionary': 'definition'}.
Reshape your data either using array.reshape(-1, 1) if your data has a single feature or array.reshape(1, -1) if it contains a single sample.
Expected 2D array, got scalar array instead:
array=99.
Reshape your data either using array.reshape(-1, 1) if your data has a single feature or array.reshape(1, -1) if it contains a single sample.
Expected 2D array, got scalar array instead:
```

### c. Forensics

I created a logging.py file based off of the one I made for workshop 9. Because I was modifying 5 functions from the “detection” directory, I also put the logging.py file in the “detection” directory. The methods I chose to modify were: main, get\_test\_details, checkClassificationAlgoTest, checkAccuracyTest, and checkAttackTest. I noticed that some of these functions were printing data to the console, so I replaced some of them with logging statements. I also logged the input and results for each of the functions. Because some of the functions were not called by the main function, I also added calls to each of the functions I modified at the end of main. The logging file is located at the root directory of the repository. I learned that using a logging library is a better way of logging than by printing to the console. The below screenshot shows the log file after running the main.py file.



```
1 22-Nov-26 22:12:00::root::INFO::file:detection/main.py::function:main::starting detection tests
2 22-Nov-26 22:12:00::root::INFO::file:detection/main.py::function:get_test_details::inputs::test-script:script
3 22-Nov-26 22:12:00::root::INFO::file:detection/main.py::function:get_test_details::results::test-list:[]::test-assert-list:[]
4 22-Nov-26 22:12:00::root::INFO::file:detection/main.py::function:checkClassificationAlgoTest::inputs::test-script:script
5 22-Nov-26 22:12:00::root::INFO::file:detection/main.py::function:checkClassificationAlgoTest::results::test-result:[]
6 22-Nov-26 22:12:00::root::INFO::file:detection/main.py::function:checkAccuracyTest::inputs::test-script:script
7 22-Nov-26 22:12:00::root::INFO::file:detection/main.py::function:checkAccuracyTest::results::test-result:[]
8 22-Nov-26 22:12:00::root::INFO::file:detection/main.py::function:checkAttackTest::inputs::test-script:script::assert-list:[]
9 22-Nov-26 22:12:00::root::INFO::file:detection/main.py::function:checkAttackTest::results::test-result:[]
10 22-Nov-26 22:12:00::root::INFO::file:detection/main.py::function:main::completed detection tests
11 22-Nov-26 22:12:00::root::INFO::file:detection/main.py::function:main::test-duration:0.00048
12
```