ATTACK DEFENSE

by PentesterAcademy

Name	Bandit: Scanning Python Code for Issues
URL	https://www.attackdefense.com/challengedetails?cid=2053
Type	DevSecOps: Static Code Analysis

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

Challenge Description

Bandit is an open source tool that finds common security issues in Python code. On execution, the tool processes each file to build AST (Abstract Syntax Tree) nodes and then plugins are executed against the AST nodes. The final output is shown to the user as the report.

A Kali CLI machine (kali-cli) is provided to the user with Bandit installed on it. The source code for the web application is provided in the home directory of the root user.

Objective: Scan the code using bandit utility and find the security issues!

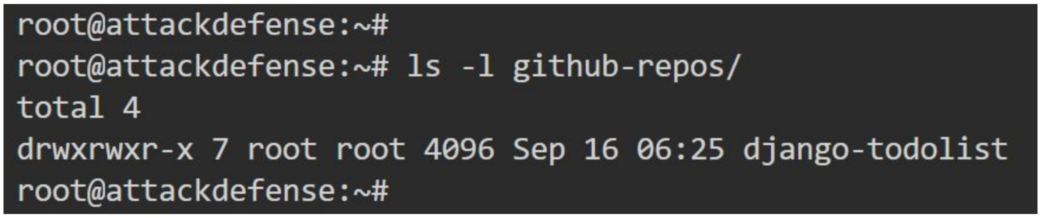
Instructions:

- The source code of web application is provided at /root/github-repos

Solution

Step 1: Check the provided web applications.

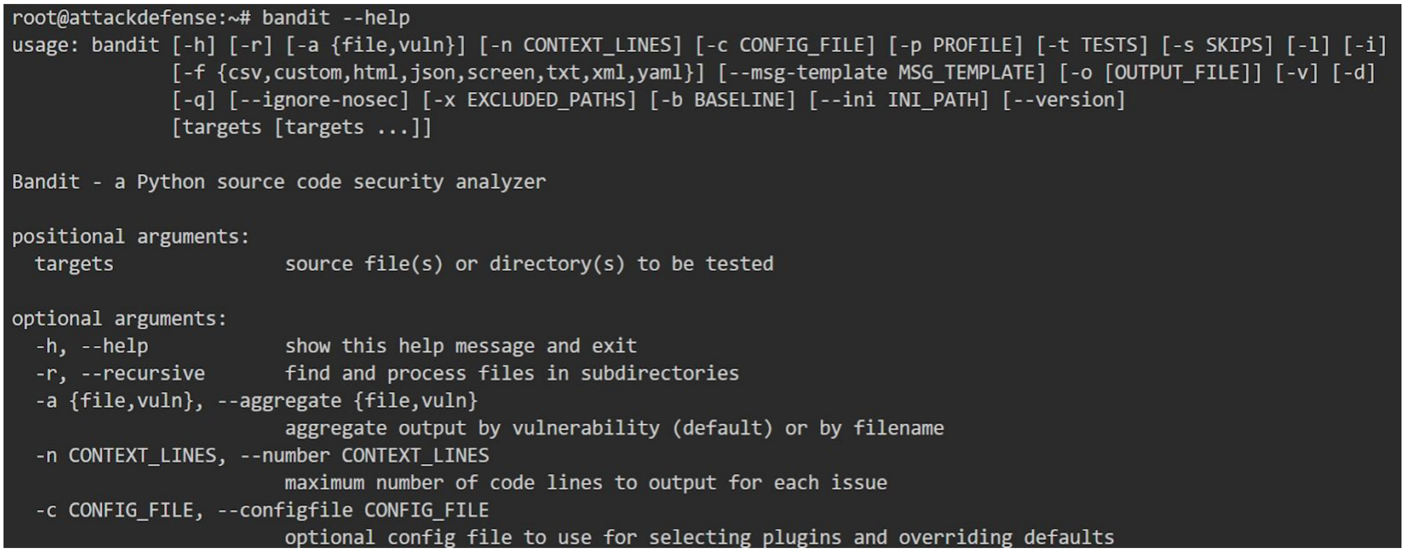
Command: ls -l github-repos



```
root@attackdefense:~#
root@attackdefense:~# ls -l github-repos/
total 4
drwxrwxr-x 7 root root 4096 Sep 16 06:25 django-todolist
root@attackdefense:~#
```

Step 2: Check the available options of the bandit tool.

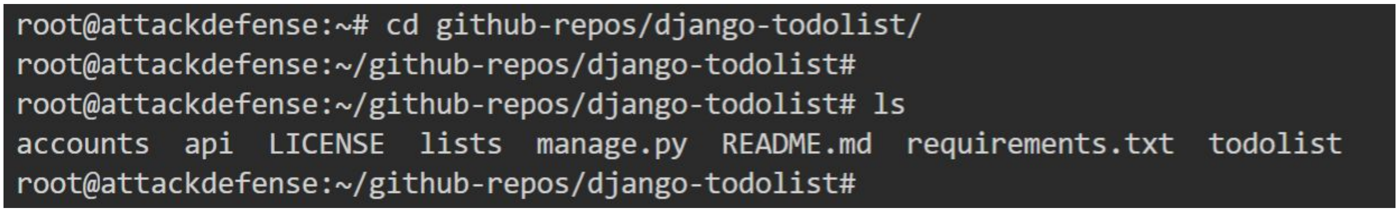
Command: bandit --help



```
root@attackdefense:~# bandit --help
usage: bandit [-h] [-r] [-a {file,vuln}] [-n CONTEXT_LINES] [-c CONFIG_FILE] [-p PROFILE] [-t TESTS] [-s SKIPS] [-l] [-i]
              [-f {csv,custom,html,json,screen,txt,xml,yaml}] [--msg-template MSG_TEMPLATE] [-o [OUTPUT_FILE]] [-v] [-d]
              [-q] [--ignore-nosec] [-x EXCLUDED_PATHS] [-b BASELINE] [--ini INI_PATH] [--version]
              [targets [targets ...]]

Bandit - a Python source code security analyzer

positional arguments:
  targets               source file(s) or directory(s) to be tested

optional arguments:
  -h, --help            show this help message and exit
  -r, --recursive       find and process files in subdirectories
  -a {file,vuln}, --aggregate {file,vuln}
                        aggregate output by vulnerability (default) or by filename
  -n CONTEXT_LINES, --number CONTEXT_LINES
                        maximum number of code lines to output for each issue
  -c CONFIG_FILE, --configfile CONFIG_FILE
                        optional config file to use for selecting plugins and overriding defaults
```

Step 3: Change to the cloned directory and check its contents.

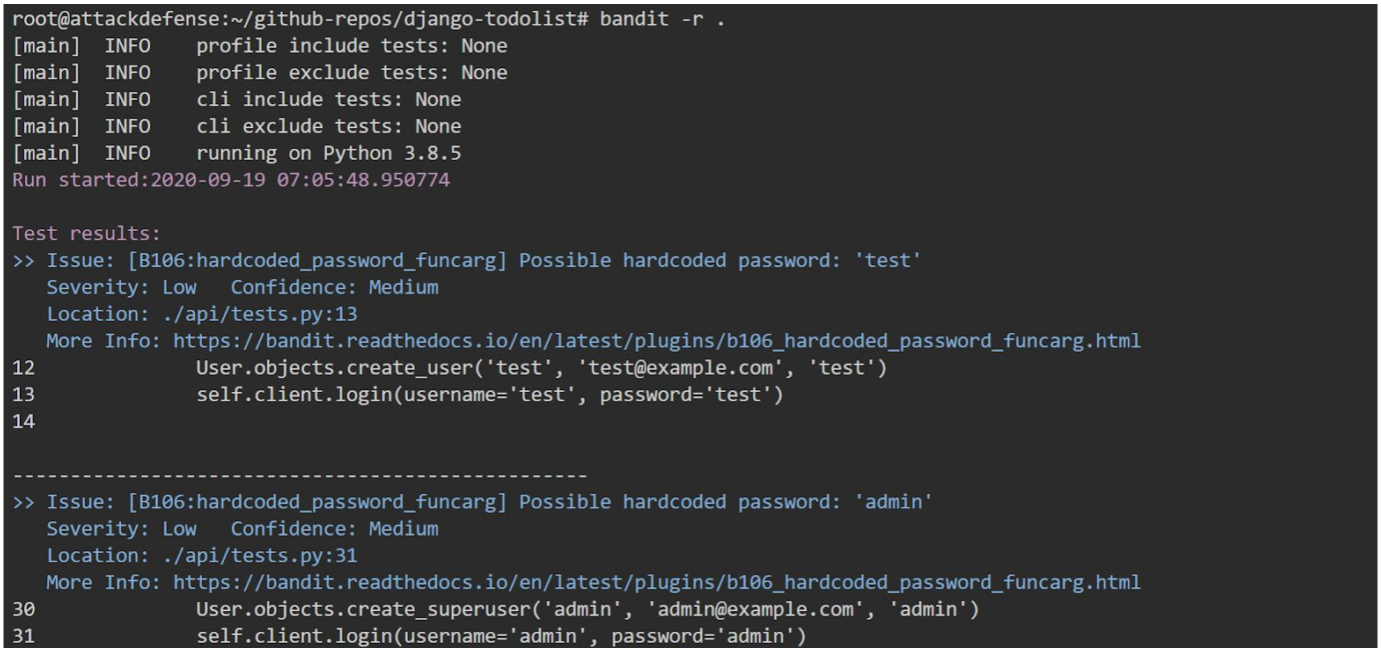
Commands:
cd github-repos/django-todolist
ls



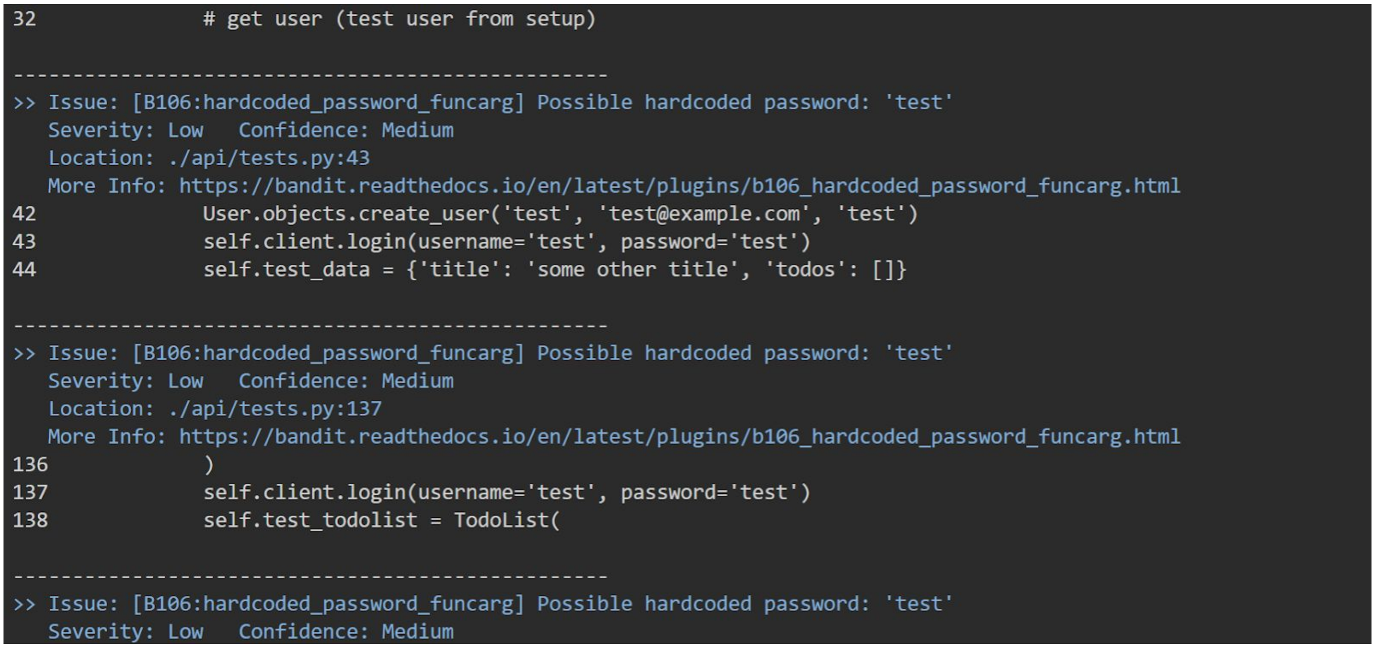
```
root@attackdefense:~# cd github-repos/django-todolist/
root@attackdefense:~/github-repos/django-todolist#
root@attackdefense:~/github-repos/django-todolist# ls
accounts  api  LICENSE  lists  manage.py  README.md  requirements.txt  todolist
root@attackdefense:~/github-repos/django-todolist#
```

Step 4: Run the bandit command to test the security weaknesses in the code repository.

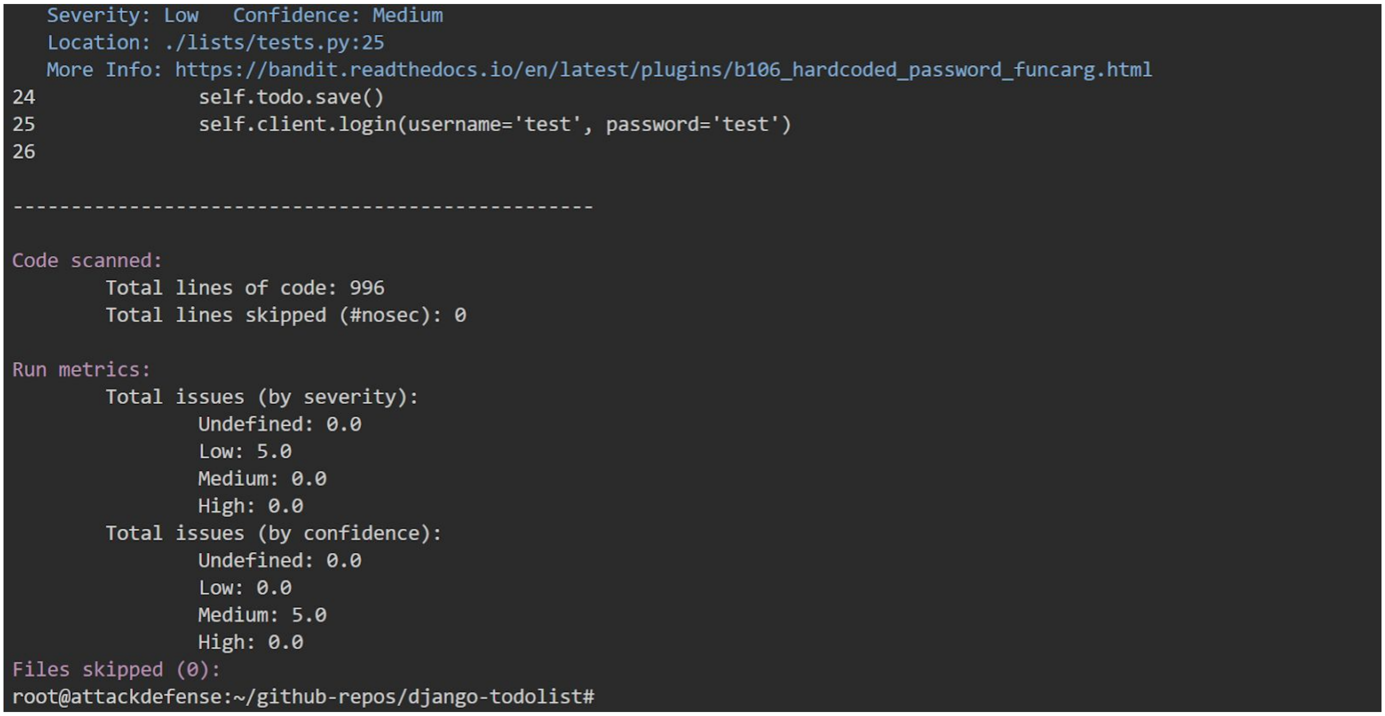
Command: bandit -r .



```
root@attackdefense:~/github-repos/django-todolist# bandit -r .
[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 3.8.5
Run started:2020-09-19 07:05:48.950774

Test results:
>> Issue: [B106:hardcoded_password_funcarg] Possible hardcoded password: 'test'
   Severity: Low   Confidence: Medium
   Location: ./api/tests.py:13
   More Info: https://bandit.readthedocs.io/en/latest/plugins/b106_hardcoded_password_funcarg.html
12              User.objects.create_user('test', 'test@example.com', 'test')
13              self.client.login(username='test', password='test')
14

--------------------------------------------------
>> Issue: [B106:hardcoded_password_funcarg] Possible hardcoded password: 'admin'
   Severity: Low   Confidence: Medium
   Location: ./api/tests.py:31
   More Info: https://bandit.readthedocs.io/en/latest/plugins/b106_hardcoded_password_funcarg.html
30              User.objects.create_superuser('admin', 'admin@example.com', 'admin')
31              self.client.login(username='admin', password='admin')
```



```
32              # get user (test user from setup)

--------------------------------------------------
>> Issue: [B106:hardcoded_password_funcarg] Possible hardcoded password: 'test'
   Severity: Low   Confidence: Medium
   Location: ./api/tests.py:43
   More Info: https://bandit.readthedocs.io/en/latest/plugins/b106_hardcoded_password_funcarg.html
42              User.objects.create_user('test', 'test@example.com', 'test')
43              self.client.login(username='test', password='test')
44              self.test_data = {'title': 'some other title', 'todos': []}

--------------------------------------------------
>> Issue: [B106:hardcoded_password_funcarg] Possible hardcoded password: 'test'
   Severity: Low   Confidence: Medium
   Location: ./api/tests.py:137
   More Info: https://bandit.readthedocs.io/en/latest/plugins/b106_hardcoded_password_funcarg.html
136             )
137             self.client.login(username='test', password='test')
138             self.test_todolist = TodoList(

--------------------------------------------------
>> Issue: [B106:hardcoded_password_funcarg] Possible hardcoded password: 'test'
   Severity: Low   Confidence: Medium
```



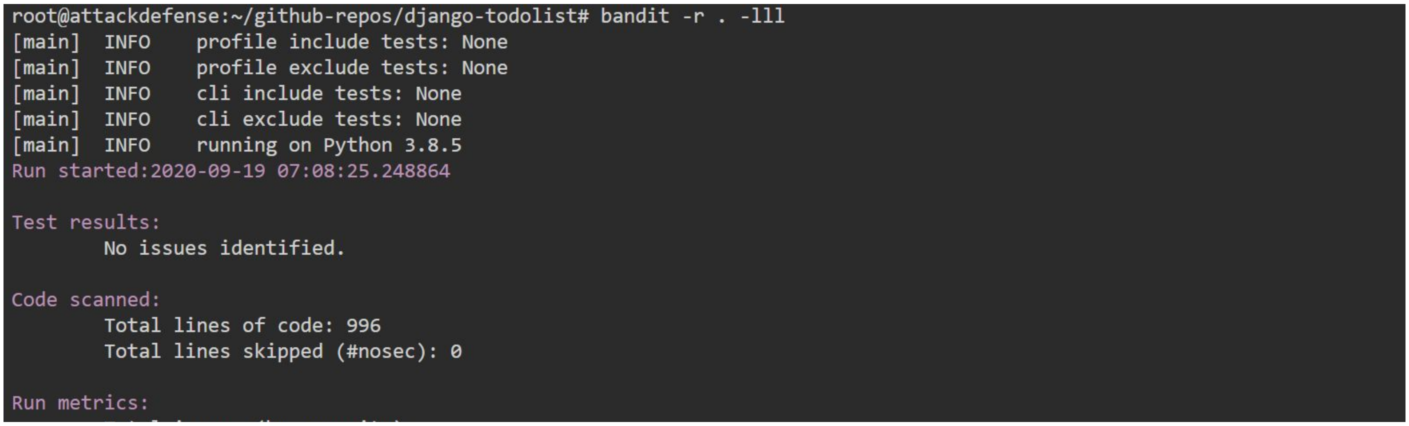
```
   Severity: Low   Confidence: Medium
   Location: ./lists/tests.py:25
   More Info: https://bandit.readthedocs.io/en/latest/plugins/b106_hardcoded_password_funcarg.html
24              self.todo.save()
25              self.client.login(username='test', password='test')
26

--------------------------------------------------

Code scanned:
        Total lines of code: 996
        Total lines skipped (#nosec): 0

Run metrics:
        Total issues (by severity):
                Undefined: 0.0
                Low: 5.0
                Medium: 0.0
                High: 0.0
        Total issues (by confidence):
                Undefined: 0.0
                Low: 0.0
                Medium: 5.0
                High: 0.0
Files skipped (0):
root@attackdefense:~/github-repos/django-todolist#
```

Issues Detected

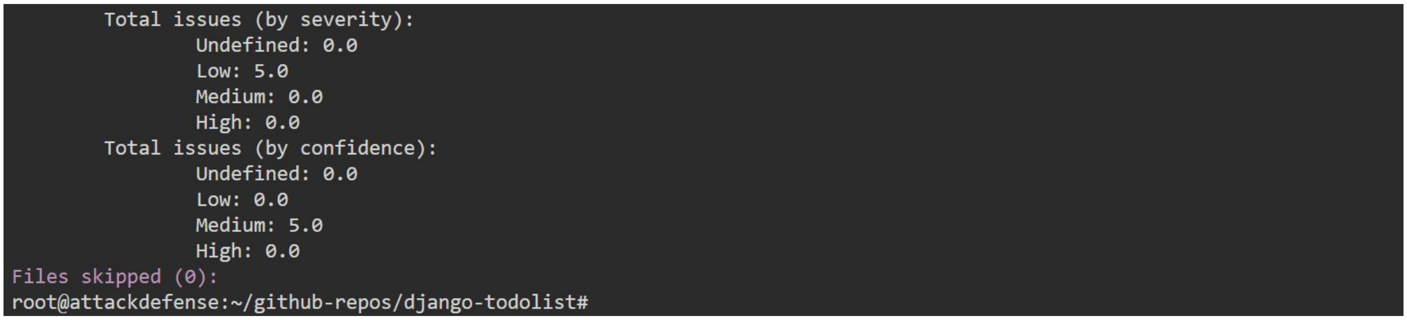
- Hardcoded credentials are found in multiple files.

Step 5: Filter the issues reported on the basis of severity levels. For example, -l argument for LOW, -ll for MEDIUM, -lll for HIGH severities.

Command: bandit -r . -lll



```
root@attackdefense:~/github-repos/django-todolist# bandit -r . -lll
[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 3.8.5
Run started:2020-09-19 07:08:25.248864

Test results:
        No issues identified.

Code scanned:
        Total lines of code: 996
        Total lines skipped (#nosec): 0

Run metrics:
```

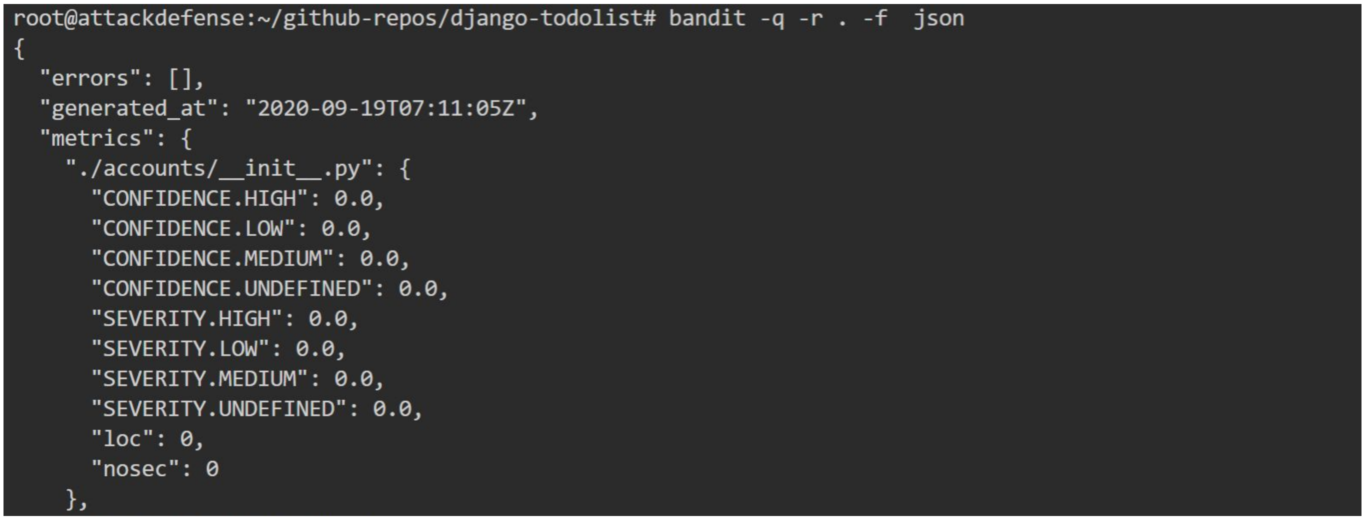


```
        Total issues (by severity):
                Undefined: 0.0
                Low: 5.0
                Medium: 0.0
                High: 0.0
        Total issues (by confidence):
                Undefined: 0.0
                Low: 0.0
                Medium: 5.0
                High: 0.0
Files skipped (0):
root@attackdefense:~/github-repos/django-todolist#
```

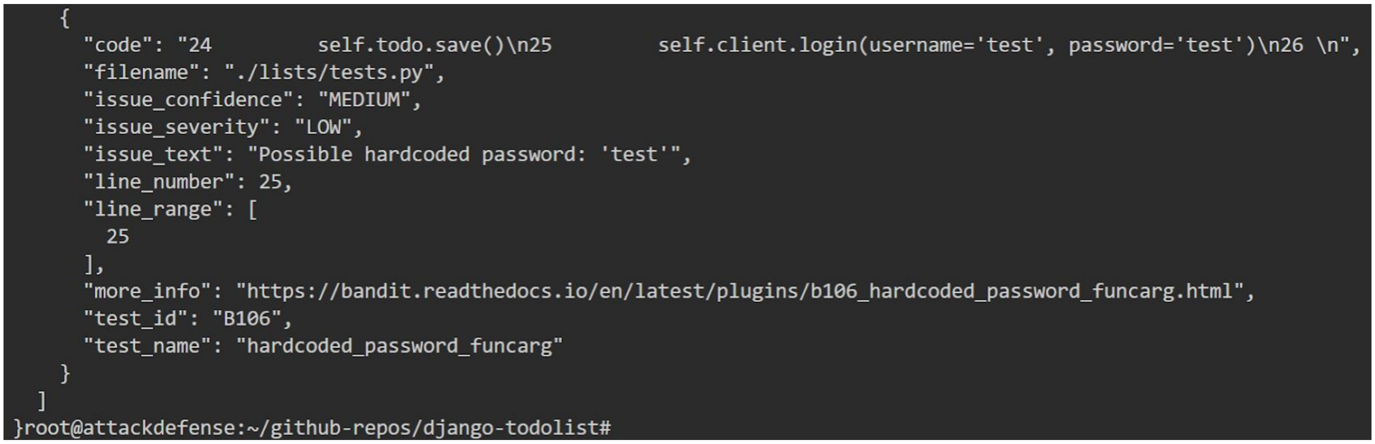
No issues have a high severity.

Step 6: Use the following bandit feature to store all the issues reported in a JSON format.

Command: bandit -q -r . -f json



```
root@attackdefense:~/github-repos/django-todolist# bandit -q -r . -f  json
{
  "errors": [],
  "generated_at": "2020-09-19T07:11:05Z",
  "metrics": {
    "./accounts/__init__.py": {
      "CONFIDENCE.HIGH": 0.0,
      "CONFIDENCE.LOW": 0.0,
      "CONFIDENCE.MEDIUM": 0.0,
      "CONFIDENCE.UNDEFINED": 0.0,
      "SEVERITY.HIGH": 0.0,
      "SEVERITY.LOW": 0.0,
      "SEVERITY.MEDIUM": 0.0,
      "SEVERITY.UNDEFINED": 0.0,
      "loc": 0,
      "nosec": 0
    },
```



```
    {
      "code": "24         self.todo.save()\n25         self.client.login(username='test', password='test')\n26 \n",
      "filename": "./lists/tests.py",
      "issue_confidence": "MEDIUM",
      "issue_severity": "LOW",
      "issue_text": "Possible hardcoded password: 'test'",
      "line_number": 25,
      "line_range": [
        25
      ],
      "more_info": "https://bandit.readthedocs.io/en/latest/plugins/b106_hardcoded_password_funcarg.html",
      "test_id": "B106",
      "test_name": "hardcoded_password_funcarg"
    }
  ]
}root@attackdefense:~/github-repos/django-todolist#
```

Learnings

Perform Static Analysis on the source code using bandit utility.