Ioana-Gabriela Chelaru
248-1

QA Report

Project used:

My batchelors thesis application, which is a small cross-platform application written in python, that can observe a subject's facial features from a continuous video-stream and detect its early signs of drowsiness. The targeted users are mainly drivers, the end goal being eliminating the risk of falling asleep while driving, therefore reducing the number of car crashes.

Code review tools used:

Since my project is written in python, my choice was **Pylint**. The results I obtained can be summarized into: *"Pylint found 24 errors, 3 warnings, 18 conventions, 4 refactors and 2 infos"*

The major disadvantage I noticed is that all the errors reported back are *"unable to import module X"* or *"module X does not contain Y"* which are erroneous, since my application doesn't have any issues with any of those modules.

On the other hand, the warnings, conventions and refactors seem to be really useful, showing me that there are things that I didn't even considered or knew so far.

Figures 1 and 2 represent some of the most practical suggestions I found, and as it can be seen, they are related to using too many local variables or abuse of branches, unnecessary keyword and unused variables.
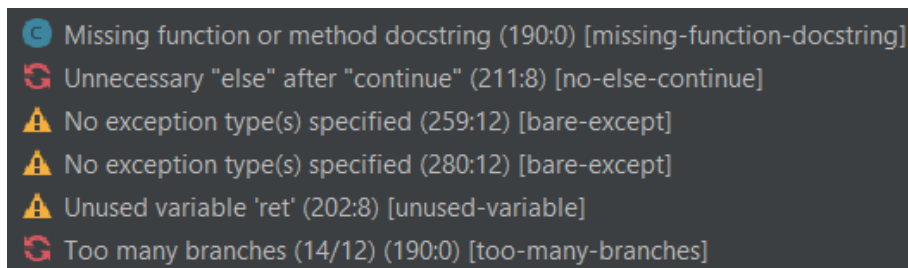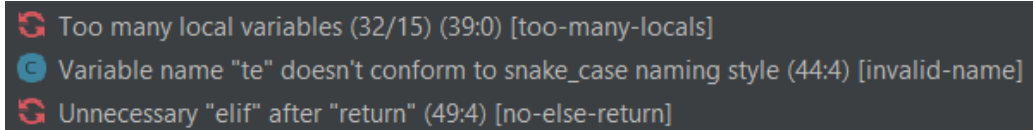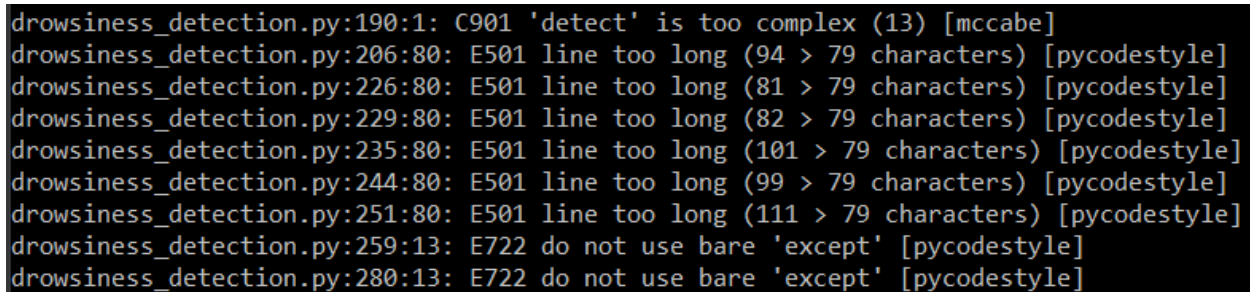


```
C  Missing function or method docstring (190:0) [missing-function-docstring]
R  Unnecessary "else" after "continue" (211:8) [no-else-continue]
A  No exception type(s) specified (259:12) [bare-except]
A  No exception type(s) specified (280:12) [bare-except]
A  Unused variable 'ret' (202:8) [unused-variable]
R  Too many branches (14/12) (190:0) [too-many-branches]
```

Figure 1 - Pylint

```
🔄 Too many local variables (32/15) (39:0) [too-many-locals]
ⓒ Variable name "te" doesn't conform to snake_case naming style (44:4) [invalid-name]
🔄 Unnecessary "elif" after "return" (49:4) [no-else-return]
```

Figure 2 - Pylint

As a curiosity of mine, I decided to give **Pylama** a try aswell. Figure 3 describes what most of the output looks like. Unlike Pylint, this tool is a wrapper for various others (*ex: pycodestyle, pydocstyle, PyFlakes, Mccabe, etc.*) and one thing that I noticed is that Pylint gave back a far more thorough report than this one. The output generated by Pylama consists mostly of lines being too long and the detection of a too complex function.

```
drowsiness_detection.py:190:1: C901 'detect' is too complex (13) [mccabe]
drowsiness_detection.py:206:80: E501 line too long (94 > 79 characters) [pycodestyle]
drowsiness_detection.py:226:80: E501 line too long (81 > 79 characters) [pycodestyle]
drowsiness_detection.py:229:80: E501 line too long (82 > 79 characters) [pycodestyle]
drowsiness_detection.py:235:80: E501 line too long (101 > 79 characters) [pycodestyle]
drowsiness_detection.py:244:80: E501 line too long (99 > 79 characters) [pycodestyle]
drowsiness_detection.py:251:80: E501 line too long (111 > 79 characters) [pycodestyle]
drowsiness_detection.py:259:13: E722 do not use bare 'except' [pycodestyle]
drowsiness_detection.py:280:13: E722 do not use bare 'except' [pycodestyle]
```

Figure 3 - Pylama