COMP4050 - Software Engineering Practices

Developer Manual

School of Computing

Macquarie University, Balaclava Rd,

Macquarie Park NSW 2109

Project Description

This is the project, wherein we are supposed to make a testing tool for processing that can be used in education as well as for tutors and lecturers. Due to this reason, our project is intended to tackle the problem by developing a CLI or GUI program that enables users, mostly tutors, to submit assignment submissions. The program will deliver the individual tests that each student's tutor had prepared for. These tests can provide brief results that include student cumulative grades, error warnings, and a list of all tests that passed or failed while the code was being executed.

Please ensure you have read both the README and the User Manual before processing.

Project Requirements

The below requirements are based on the MoSCow Prioritization.

Must have

- Processing code will be exported to Java.
- Static analysis will be run on this Java code.
- JUNIT tests provided by the lecturer will be run if present against the Java code.
- That the program does not error on runtime will be checked via running the code and checking the exit code of the application.
- Repo quality control will be designed and implemented.
- A report will be generated, for the individual projects passed into the program, in a format yet to be decided.

Should have

- Test object creation and call functions off the object.
- Test key presses call the correct object.
- Test events
- Method isolation

Could Have

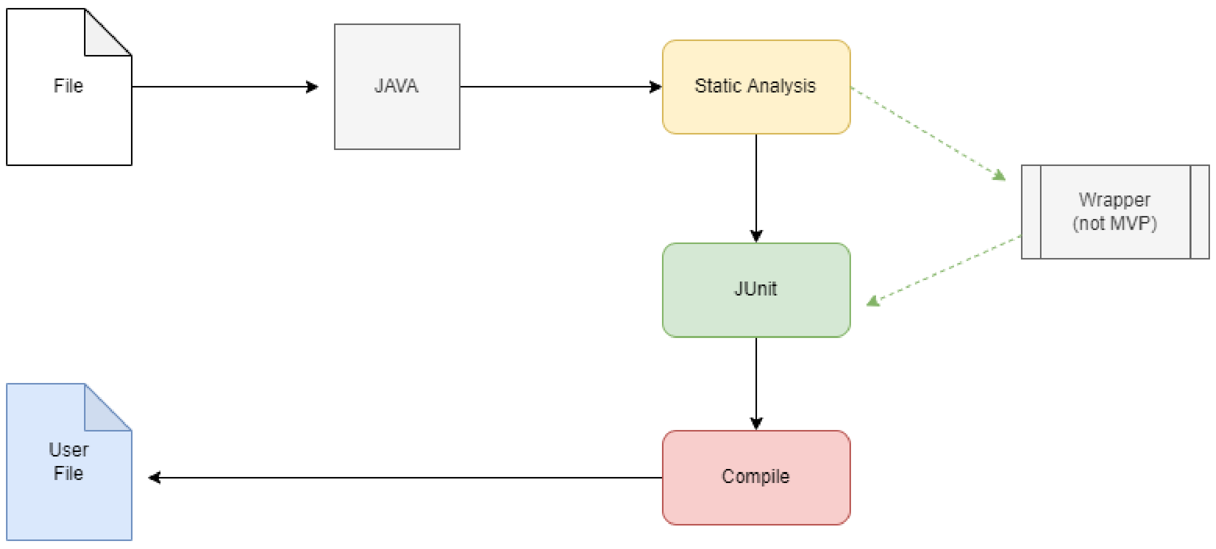
- Potential UI implemented being a website or an application.
- Drag and drop functionality if UI is implemented

Won't Have

- Visual testing

Project Architecture

- For this project, Java will be utilized primarily for coding.
- The final product will be a CLI program that exports the processing code to Java and then executes unit tests created by the instructor as well as internal tests as needed. The application may have a user interface (UI) after MVP, or if considered unnecessary, new functionality will be introduced or existing ones may be improved in the CLI application.
- To assess the quality of the code, static analysis libraries and applications will be employed.
- A report will be returned either via the console or via a file that contains information about the quality of the tested code.



Project Testing

To test this project, first, it needs to set up a few testing environments. First and foremost, the user needs to set up Maven. Maven is a build management tool that will allow us to build our project consistently and standardize between operating systems. It will also handle package management and various other problem. The main advantage of implementing Maven into this project is that it will help to develop a standard building system as well as provide accurate project details and motivate to have better development practices. Furthermore, We can utilize Maven's great testing infrastructure to execute our JUNIT tests. The main advantage of Maven on JUNIT testing is that it helps to set up test cases in their environment instead of customizing the build for test preparation. Maven also helps in finding running tests using their test case naming conventions as well as keeping test source code in a separate, but parallel source tree.

Setting up the environment:

In order to set up the environment first clone this project's repository:

```
https://github.com/5seaton4/Comp4050_Team2_PizzaCrew.git
```

Setting up the Maven environment:

To set up the Maven environment in the terminal run the following command :

```
./mvnw -v
```

After running this command in the terminal, you should get something like the below output:

```
Maven home: C:\Users\Priyanshi Patel\.m2\wrapper\dists\apache-maven-3.8.6-bin\67568434\apache-maven-3.8.6
Java version: 1.8.0_281, vendor: Oracle Corporation, runtime: C:\Program Files\Java\jdk1.8.0_281\jre
Default locale: en_SG, platform encoding: Cp1252
OS name: "windows 10", version: "10.0", arch: "amd64", family: "windows"
```

A Maven project can be imported by your preferred IDE, and you can then compile from there. You can use the following commands to compile from the command line.

```
./mvnw compile
```

Then in order to get the *hawaiian.jar* packages, you need to run the following command:

```
./mvnw package
```

To execute the tests in the repository, you need to run the following command:

```
./mvnw test
```

After the tests have been running successfully, it shows that the Maven environment has been successfully added to the repository and it is working properly.

Running the application

See the User manual on instructions on how to run the program.