Testing Policy Document

Optimize Prime

Definition of Testing:

Testing is done to effectively and efficiently provide timely, accurate, and useful information of the current state of the application. This is to ensure the software fulfills all its requirements while remaining in a healthy condition (bug free, etc).

Description of the test process:

Unit tests:

Goals:

- Detect defective code in units
- Reduce risk of unit failure in Production

Unit tests must comply with all coding standards.

Integration:

Goals:

- Detect defects in unit interfaces
- Reduce risk of dataflow and workflow failures in Production

After every major feature commit a test should be written to validate that the feature is working and giving the expected result. Theses tests must be added to automatically run on our continuous integration service (Travis CI).

Quality Level to be achieved:

Tests should result in no outstanding high severity faults prior to version releases. All server side tests must succeed for server commits to be deemed ready for merging into develop. Current running instances of the development server can only be updated if all unit tests pass.

Approach to Test Process Improvement

Tests should be reviewed alongside code reviews after a sprint. Tests should be on the same quality standard as the production code released to end-users.

Current continuous integration results:

Current tests are executed on Travis CI:

Backend: https://travis-ci.org/COS301-OptimizePrime/COS301_DND_Backend

A badge is added to the repository to show the current result status:

Current backend tests include:
Example if the tests being run:

```
===Runing python tests!===
============================================ test session starts =============================================
platform linux -- Python 3.6.5, pytest-3.5.1, py-1.5.3, pluggy-0.6.0
benchmark: 3.1.1 (defaults: timer=time.perf_counter disable_gc=False min_rounds=5 min_time=0.000005 max_time=1
.0 calibration_precision=10 warmup=False warmup_iterations=100000)
rootdir: /home/thomas/Repos/COS301_DND_Backend/test, inifile:
plugins: benchmark-3.1.1
collected 25 items

test_sessions.py .........................                                                          [100%]


------------------------------------------------ benchmark: 1 tests -------------------------------------------
-----
Name (time in s)                  Min     Max    Mean  StdDev  Median     IQR  Outliers     OPS  Rounds  Itera
tions
-----------------------------------------------------------------------------------------------------------------
-----
test_create_rpc_good_login     1.8277  1.9431  1.8834  0.0496  1.9023  0.0825       2;0  0.5309       5
    1
-----------------------------------------------------------------------------------------------------------------
-----

Legend:
  Outliers: 1 Standard Deviation from Mean; 1.5 IQR (InterQuartile Range) from 1st Quartile and 3rd Quartile.
  OPS: Operations Per Second, computed as 1 / Mean
========================================== 25 passed in 112.59 seconds ===========================================
===Runing dart tests!===
00:00 +0: test_create_rpc_good_login
Client received: SUCCESS
00:02 +1: All tests passed!
```

Front-end: https://travis-ci.org/COS301-OptimizePrime/COS301-DnD
When a push or pull request takes place on GitHub, Travis tests integration with Android by building an APK and also runs the tests we have created for the application located in the tests folder.

These tests include:

1) Open App:
    a) Opens the Application to see if it runs.

2) Login Page Test
    a) Opens the Login Page to see if it loads correctly.
3) Menu Drawer Open/Close Test:
    a) Tests if the menu drawer opens and closes correctly.
4) Menu Navigation Test:
    a) Test the menu navigation functionality by tapping on all the icons in the drawer and testing if the app correctly navigates to all pages.
5) Home Page Test:
    a) Tests if the Home Page loads correctly
6) Character Selection Page Test:
    a) Tests if the Character Selection Page loads correctly.
    b) Tests tap on Character functionality that shows more details about the Character.
7) Character Preview Test:
    a) Tests if a preview of the Character pops up when a swipe is made on a Character.
8) Monster Journal Page Test:
    a) Tests if the Monster Journal Page loads correctly.
    b) Tests tap on Monster functionality that shows more details about the Monster.
9) Create Session Test:
    a) Tests functionality when a user taps on the ”Create Session” button.
10) View Races Page:
    a) Tests if the Monster Journal Page loads correctly.
    b) Tests tap on Monster functionality that shows more details about the Monster.

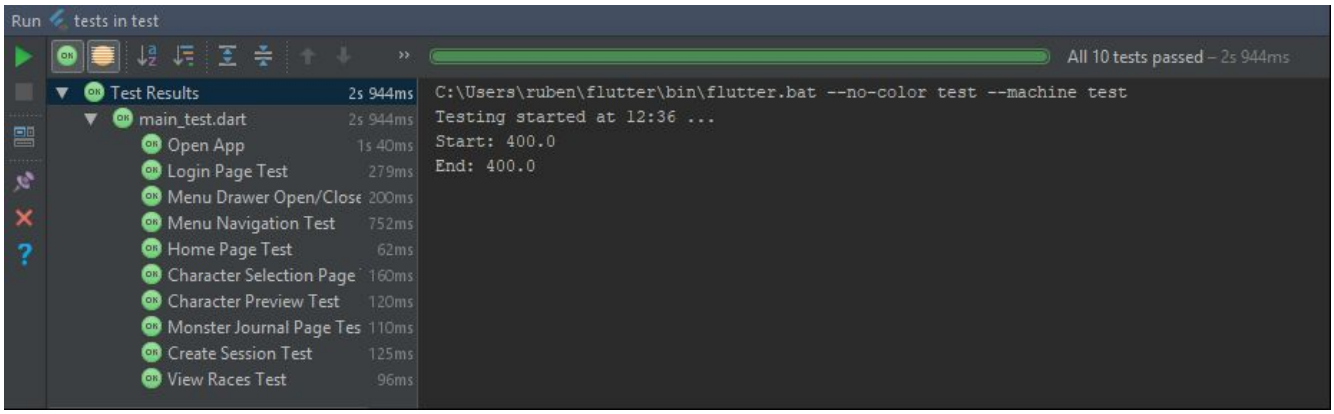
Taken from https://travis-ci.org/COS301-OptimizePrime/COS301-DnD:

```
The command "./flutter/bin/flutter -v build apk" exited with 0.
$ ./flutter/bin/flutter test
00:10 +8: - Character Preview Test
Start: 400.0
End: 400.0
00:10 +10: All tests passed!


The command "./flutter/bin/flutter test" exited with 0.

Done. Your build exited with 0.
```

Taken from Android Studio:


Run
tests in test
All 10 tests passed – 2s 944ms
Test Results 2s 944ms
main_test.dart 2s 944ms
Open App 1s 40ms
Login Page Test 279ms
Menu Drawer Open/Close 200ms
Menu Navigation Test 752ms
Home Page Test 62ms
Character Selection Page 160ms
Character Preview Test 120ms
Monster Journal Page Tes 110ms
Create Session Test 125ms
View Races Test 96ms
C:\Users\ruben\flutter\bin\flutter.bat --no-color test --machine test
Testing started at 12:36 ...
Start: 400.0
End: 400.0