**CPEN 321: Software Engineering**
**Fall 2019**

# M5 (W12): Testing

Each group must submit a report by 3pm on the day before the weekly meeting with the TA.
A report must contain two parts, in one pdf file. Only one report per group is submitted.

**PART 1**:
A 1-page status update that includes

1) A high-level description of the progress since the previous milestone (M4).
2) The plans until the next milestone (M6).
3) Major decisions and changes in the scope of the project (since M0).
4) The contributions of the individual team members to the work done so far.

**PART 2**:
For this milestone, you will perform **a systematic testing** of your project. In particular, you will perform:

1. Automated unit and integration testing for **your** back-end;
2. Automated UI testing for the three main use cases of **your** project – with external API, with push notifications, and with non-trivial logic, as defined in M1;
3. Automated or manual testing for two of **your** non-functional requirements from M1;
4. Manual customer acceptance testing for the app of **your partner team**.

You will need to set up and run the following tools:

- Travis CI to run all your tests every time a code is committed to your repository.
- Jest (https://jestjs.io) as a testing and code coverage framework for the **back-end**.
- Testing frameworks for the **front-end**:
  - Android native: Espresso (https://developer.android.com/training/testing/espresso).
  - iOS native: XCUITest (https://developer.apple.com/library/archive/documentation/DeveloperTools/Conceptual/testing_with_xcode/chapters/09-ui_testing.html).
  - React Native: Detox (https://github.com/wix/Detox).

Then, prepare and submit the following artefacts.

1) For the automated back-end testing [35 points]:
   - A list of unit tests for each of your back-end node.js module. Make sure you mock all modules besides the one under test. Also, make sure you check success and failure conditions of each method, invalid method parameters, etc. [10 points].
   - A list of all integration tests. Each integration test should correspond to a use case of your system from M1 (all use cases, not only the three major ones). An integration test needs to check a sequence of events realizing the corresponding use case by invoking component APIs (think sequence diagrams). [10 points].

- A statement- and method-level coverage report for each unit and integration test, for all unit tests together, and for all integration tests together [10 points].
- A table summarizing the total number of tests per category (unit and integration) and an automated execution log for these tests, including their pass / fail status. [5 points].
    - Some tests may still fail at this point, but they will need to pass by M6.

2) For the automated front-end UI testing [30 points]:
    - A list of tests you created, with one paragraph describing how you encoded success / failure criteria for each UI test (test oracle) and 2-3 example screenshots [20 points].
    - A log of the automated execution of these tests on the device, including their pass / fail status [10 points].
        - Some tests may still fail at this point, but they will need to pass by M6.
        - Detox tests might not run on iOS physical devices. If you plan to run your app on an iOS device, you need to specify that in your report and then run the tests on the emulator.

3) For testing of non-functional requirements [20 points]:
    - Pick two non-functional requirements identified in M1. For each requirement, write a one-paragraph verification plan. [10 points].
        - Strive to pick requirements that you can verify automatically; when not possible, manual verification is acceptable.
    - Show the log with your verification results [10 points].

4) For manual customer acceptance testing for the app of **your partner team** [15 points]:
    - Meet with your customer team and test their app. Ask them to test yours.
    - Report one major fault you found in **your partner's team app.** The report should contain the buggy execution scenario (sequence of events, with screenshots) and the description of the fault. [15 points]**.**
        - Yes, you will find some major fault – there is no app without faults 2 weeks before the final deadline.
        - If you report that you cannot find any major fault and then your TA does, you will lose marks. If the TA does not find any major issues either, you will get the full mark.
    - Report one major fault that your peer-team found **in your app.** Provide a short description, no screenshots are needed**.**
        - If your peer team did not find any major faults in your code, ask your teammates working on the backend to test the app. Report the identified faults.
        - If you report that neither your peer team nor your own team member can find major faults in your app and then your TA does, you both will lose marks. If the TA does not find any faults in your app, you will get the full mark.

Bring your phone and computer to the meeting and be ready to run the test and/or show their code.