1. Team Information

<Team X> (X is the letter your team is assigned in Group information and POD meeting).
<Enter your team information here: full name, student id, contact email>

2. Selected Metrics and Correlation analysis

You are required to collect the following metrics. For the proposal milestone, **you do NOT need to complete the data collection, but you should decide metric 5 and 6.** Particularly, you need to clearly state and describe what are the metric 5 and 6 your team will collect for completing the course project. You will also need to properly cite related work on metric 5 and 6. You can use existing tools to collect the metrics in later milestones. We will have tutorials on some tools (e.g., Jacoco).

Metric 1&2: Two test coverage metrics, i.e., statement coverage and branch coverage. Test coverage metrics are measured from existing developers' test cases (You do NOT need to write new test cases for the open-source systems).
Metric 3: Test suite effectiveness, e.g., mutation score of a test suite [1][3].
Metric 4: One complexity metric (e.g., McCabe, Halstead)

Metric 5: You will propose one or use an existing one metric from literature **that quantifies one aspect of software maintenance efforts**. You cannot reuse metric 1&2. Examples: code change-proneness, metrics from code review activities [2], metrics to measure developers' quality assurance efforts before release (e.g., improved coverage metrics between versions).

Metric 6: You will propose one or use an existing one metric from literature that measures one of software quality attributes. Examples: bug-introducing change rate (i.e., the percentage of commits that introduce bugs), bad fix rate.

For **Metric 5** and **6**, you need to properly define and describe them: algorithm (how to calculate the metric from software artifacts), and/or text descriptions. Also, you should properly cite related work if the definition or a similar definition is defined by current literature.

You will conduct the following correlation analyses (**in later milestones, not NOW**):

- Correlation between each coverage metric (Metric 1&2) and test suite effectiveness (Metric 3). The rationale is that test suites with higher coverage might show better test suite effectiveness.
- Correlation between one complexity metric (Metric 4) and each coverage metric (Metric 1&2). The rationale is that classes with higher complexity are less likely to have high coverage test suites.

- Correlation between each coverage metric (Metric 1&2) and Metric 6. Include the rationale behind this correlation. An example of such rationale can be "Classes with low test coverage contain more bugs".
- Correlation between Metric 5 and Metric 6. Include the rationale behind this correlation**.**

3. Related Work

Describe related work in literature that is related to the metrics 1-6.
Note that this section will be further expanded in the final report. Write down your best effort for now.

4. Selected Open-Source Systems

List the **four** systems that you will analyze in this section. At least two open-source systems should have at least 100K SLOC. Here are some references for Java projects:
- https://www.openhub.net/tags?names=java
- https://projects.apache.org/projects.html?language#Java

Include the names of the open-source systems, version number, and describe how the open-source systems satisfy the requirement of SLOC, and the need of calculating the metrics 1-6. For example, the existence of an issue-tracking system is needed for calculating the number of post-release defects.

Note that You may change the analyzed systems in later milestones as long as the new ones still satisfy the requirements.

5. Resource Planning

Describe the tasks each team member is assigned **for the entire project**.

Note that the resource planning is subject to change as the project continues. Write down your best estimation and planning for now.

References

Add references and cite them in the proposal (e.g., in 2. Selected Metrics and Correlation analyses and 3. Related Work). See some examples below.

[1] Laura Inozemtseva and Reid Holmes. 2014. Coverage is not strongly correlated with test suite effectiveness. In Proceedings of the 36th International Conference on Software Engineering (ICSE 2014). ACM, New York, NY, USA, 435-445. DOI: https://doi.org/10.1145/2568225.2568271

[2] Shane McIntosh, Yasutaka Kamei, Bram Adams, and Ahmed E. Hassan. 2014. The impact of code review coverage and code review participation on software quality: a case study of the qt, VTK, and ITK projects. In Proceedings of the 11th Working Conference on Mining Software Repositories(MSR 2014). ACM, New York, NY, USA, 192-201. DOI: http://dx.doi.org/10.1145/2597073.2597076

[3] René Just, Darioush Jalali, Laura Inozemtseva, Michael D. Ernst, Reid Holmes, and Gordon Fraser. 2014. Are mutants a valid substitute for real faults in software testing?. In Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE 2014). ACM, New York, NY, USA, 654-665. DOI: https://doi.org/10.1145/2635868.2635929