COMP 3350 Project Iteration 1

Group 3 (The Job Costing Application)

Planning and process (5/6)

- GIT (1.5/1.5)
    - is accessible (0.5/0.5)
    - Version control is being used properly for example, has more than one committer and commits are reasonable size and frequency. They are not only big commits at the end. (1/1)

Comments: GIT is accessible, two members have committed more than 3 to 4 times of the rest of the team. Plus, some of GIT features are unused like issues.

- Architecture sketch (1.25/1.5)
    - Should be 3-layer (0.5/0.5)
    - Should have some high-level classes in each layer (not include very low level details) (0.5/0.5)
    - Should show the relationships between classes (0.25/0.5)

Comments: Provided a basic three tier architecture with some explanation but the relations are not clear. (-0.25), you need to improve on your architecture diagram as it is poorly presented.

- Updated plan (1.5/2)
    - Plan should be up-to-date (if there is any change to the previous plan for Iteration 1 it should be explicit and justified) (0.5/0.5)
    - Big user stories for iteration 2, if it was not already in plan (0.25/0.5)
    - Development tasks assigned in iteration 1 (what exactly has been done by developers) (0.5/0.5)
    - The time planned for the development tasks and detailed user stories and the actual time it took, in iteration 1 (0.25/0.5)

Comments: Big stories for 2nd iteration is not provided and referring to iteration 0 submission big stories are not labeled for any specific iteration (-0.25). There is no time sheet specifying actual time taken to implement the development tasks. (-0.25)

- Wiki (0.75/1)
    - Should include description of the content of the submission. Can include other things as well. (0.75/1)

Comments: Wiki should have provided either the architecture document or the link to architecture document and log files. (-0.25)

Functionality (3.5/6)

- Works on both emulator and tablet device. (2/2)
- The developed program conforms the updated plan (the stories that are claimed to be implemented, are indeed there) (1/1)
- Database stub and its interface (0/1)
- At least one completely functional GUI, which performs end-to-end processing for at least one big story (0.5/1)
- No easy bug (No crashes or unexpected behavior while trying normal scenarios) (0/1)

Comments: Application started on both emulator and device but it would crash by clicking on any menu (Order #1). (-1) There is no stub implementation of database interface (-1), since there is no feature that would work end to end (-0.5).

Bug: none of the features were working properly on Nexus 7 device plus application would have crashed on most of the features like summary, add order and so on.

Implementation (4/4)

- Appropriate package structure for code and the test base (1/1)
- Good standard coding style (2/2)
    - Informative naming
    - Comments explain "why" and not "What"
    - No to-do
    - Too much code duplication (copy-paste)
- No obvious design smells (1/1)
    - Classes are in the wrong package (e.g., logic is developed in the UI layer)
    - Big classes: Classes are taking too much responsibility (SRP)
    - Very long methods (over 20 lines)
    - Wrong usage of inheritance

Comments: No major issues, ManualCalculator seems to be activity class but it is under business logic classes.

Unit tests (0/4)

Automated JUnit test cases and test suites are available (0/1)

Passes all unit tests for domain objects and business logic (0/1)

Reasonable test coverage of normal and corner cases (0/2)

Comments: There is no unit test or description on how to get them.

Penalties ()

- Log file (up to -2 if missing or incomplete)
- Missing libraries. Unspecified dependencies. (up to -2)

Comments:

Total (12.5/20)