COMP 3350 Project Iteration 1

Group 5 (Animal Inventory Management 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 and all members have reasonable commits of reasonable size.

- Architecture sketch (1.5/1.5)
    - 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 (0.5/0.5)

Comments: Well-presented 3-layer architecture diagram is provided.

- Updated plan (1.25/2)
    - 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 (0/0.5)
    - 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: There is no big stories specified for iteration 2 in previously submitted documents (-0.5), also there is a file for iteration 0 which is in .rtf extension! (You should provide your file in PDF). The time spent on tasks should be specified in front of each detailed tasks not just a sum of hours per developer. (-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: There should be links from your wiki to your logs and architecture diagram. (-0.25) Your wiki should either include architecture sketch or log files, if not it should have a link to the file.

Functionality (3/6)

- Works on both emulator and tablet device. (1/2)
- 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 (0/1)
- No easy bug (No crashes or unexpected behavior while trying normal scenarios) (1/1)

Comments: The application is not running in android studio's emulator or on Nexus 7 device. (-1), checked the user stories by going through codebase. There is no specific stub implementation of database interface. (-1) Not able to verify one completely functional GUI (-1)

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 issue, there is proper packaging structure in place.

Unit tests (3/4)

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

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

Reasonable test coverage of normal and corner cases (1/2)

Comments: There are limited test methods in DatabaseManagerTest.java class. All your methods should be covered by the unit tests(-1)

Penalties ()

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

Comments: No issue.

Total (15/20)