**Introduction**

This report informs essential information about the SENG201 project and how the process is executed/implemented by Bach Vu and Linh Luu.

The report covers six parts. The first 3 parts are: the structure of the application, unit test coverage and the pros and cons of its maintenance, reflections about what went well and not well during development phases. The fourth and fifth sections are about the declaration of partners’ contribution, included a brief describe of key involvement in project by individual and feedbacks on the assignment. Lastly, there is an attachment of the UML class diagram.

**The structure of your application and any design choices you had to make. We are particularly interested in communication between classes and how inheritance was used. You might want to reference your UML class diagram.**

Our project is called “SpaceExplorer” which is a strategy and adventurous game. It is a cross-platform game which is supported by Java SDK10.0 on PC (not on mobile).

About its structure, we built our apps primarily based Object-oriented, very few static fields. A super-abstract class named Entity to display objects on screen, has subclass include Outpost, BlackHole and Planet classes.

The independent Crew and Galaxy classes are not inherited by any others. However, Crew-class implement the entire status of each crew members in Spaceship (extends Outpost), and Galaxy-class groups space objects (instance of Entity) together. In details, every crew has health, hunger and morale, and their actions associate with Spaceship stat and inventory via using supplements.

The stock interface is implemented through classes Stock\_Medicine, Stock\_Food and Ship Module. This is because each of them has a boost value and amount, and can be “used” in someway.

We have some static classes, use to enhance UI of the game such as Messbox, incomingMessage (for tutorial) and resize images.

**• Where collections have been used and in what form.**

* We mainly used ArrayList to store crew action in crews; crew, stock and ship modules in ship; entity (blackhole, planet, outpost) in galaxy.
* We sometimes use array for some configuration, such as imageList of crew and entity.

**An explanation of unit test coverage, and why you managed to get a high/low percentage coverage.**

The bonus statistics of the spaceship with each special type of crews. For example, if one of four crews is a captain, then moral of each crew increased by 25. The statistics of each crew via health, hunger and morale and their changes after using supplements. Crew action are properly catch if any error should happen.

Whether the use of stock is valid and how attributes of this object associate with other objects. (amount, boostValue to health/morale/hunger). The correctness of the outpost trade section.

Test the movement of the ship. Galaxy structure (explain in demo/readme)

=> warrant/guarantee the balance of the game, keep it sustainable and detect unintended effect side.

**Our thoughts and feedback on the project.**

We assume this project successfully not only improved our knowledge in JAVA, but also broaden our insight/mind about how a real game is made. Teamwork and communication skills are necessary for a successful project.

If everyone is seriously aware of their duties and perform them passionately, and creatively with a harmoniously teamwork, the whole progress would high potentially smooth. Although facing difficulties is inevitable during the project, be patient and discriminately figure out the problems which might be the optimal solution. That was also what we suffered from/experienced in the project and how we went through those problems.

In overall, we are both satisfied with the outcome of this project. This is an ideal example/assignment for students to introduce JAVA and a strict chain of requirements that software engineers have to manage in teamwork projects.

**A brief retrospective of what went well, what did not go well, and what improvements you could make for your next project.**

At every beginning of the project, we got confused at each other ideas to construct and develop the GUI game because of different point of views. We skipped the sketching the outline part and went straight ahead to coding which brought us various struggles. We took first days to figure out, then decided to roll back to the first step and the progress finally started working very well on track.

Got a little trouble with the length of classes. Especially, the GUI Game Environment contained a huge amount of details, therefore, it was hard to keep track what was going on and also the readability was downgraded significantly.

In the following projects, we would consider more seriously from a better sight of how software engineer must have and strictly comply with software requirements. During the project, there were no conflict happened.

**The eﬀort spent (in hours) in the project per student. • A statement of agreed % contribution from both partners.**

We have put so much effort on this assignment since very first weeks it released before the term break. Roughly 5-8h per weeks include planning and discussion.

After a short discussion, we agreed that Bach Vu contributed 60%, and Linh Luu did 40% the project. Bach was mainly responsible for Implements classes in SpaceVessel package, ship movement (Keylistener), Implement trading/currency associate with outpost, Integration testing, fixed the tough bugs, Load/Save game and write Javadoc. Linh focus on Comandline app and testing Junit (early stage), Work with graphics, mainly in Backend package, UI testing, storyline/tutorial compose, and Interface design, class diagram.

Both involved in development of GUI package, which integrate all objects to the main game and display in multiple frame (GameEnvironement serve as the MDF frame).

Individual parts are alr shown in JAVA doc.