Phase II Assignments

Phase II R2M2D2

Phase II. R2M2D2 Additional Requirements, Model Refinement, Design
You need to add the functionalities described in the "Project - Phase II" document. The requirements might have alternatives; you need to make a choice and add those features to your program. Please go over Phase II requirements.

- During the meetings of December 15-16, you are expected to discuss about the details of the functional and non-functional requirements.
- During the meetings of December 22-23, you will be discussing your new use cases, SSDs, changes in the domain model, and additional designs, and submit them on Sunday (**25th of December**).

What to submit:
R2/M2 Related: your group needs to prepare the following documents due to changes in the requirements and also uncompleted requirements and analysis from Phase I:
Functional requirements

1. Functional Requirements. New use case or subfunction narratives - We expect you to submit **two use case**s (even though requirements might have more, **two use case stories** are sufficient.)

Other (if any changes in the following) :

1. Refined Domain models. See Chapters 9 & 31.
2. Selected System Sequence Diagrams. (**2 SSDs**)
3. Selected "Operation Contracts". See Chapters 11 and 32. (**2 Operation Contracts**)

D2 Related: Using your requirements document from Requirement II (and possibly I) and considering your TA's feedback, your group needs to discuss the application of design patterns. **Submit sequence and class design diagrams** illustrating the use of **two** GoF Patterns.
Use of GIT: In the meetings, you should demonstrate that your group uses git. We will particularly check branches created by different members for new additions/functionalities/bug fixes etc, to the code, merge, merge request, some other member approving the merge, etc.
NOTE: Highlight the classes added during the Design II phase in your diagrams. We will evaluate your design based on **GoF patterns.** We will check if your design follows the **Model-View separation**

Phase II Testing

In this assignment, every team member will do the following for a non-trivial function/method.

- (Every team member) Choose a non-trivial method to test (is complex enough to write minimum of 5 tests)
  - Write specifications for the method (write requires, modifies, and effects clauses in a comment block in the source code)
  - Specifications will be merged into the main development branch by each member separately.
  - Apply Black-Box and GlassBox techniques to come up with a minimum of 5 test cases.
- (For the group) Choose one non-trivial class to test. Write overview, abstract function, representation invariant, repOk method, and write at least 5 tests to check the class (abstract data type))
- Use JUnit to do the tests.
- Use a separate branch to run your tests (every member does this individually). In the meeting, every member must demonstrate that they used git to implement the tests.

Meeting and Submissions
Come with a test plan draft to the meeting to discuss your tests, possibly more than half already done. Submit the test plan and its results, together with which git branch to test. We will check the corresponding git test branch and corresponding classes.

Final Demo

FINAL DEMO

The grading of the final demo is composed of two parts: Demo (60 points) + Code and Design (40 points).

In the demo part, we will focus on the application's functionality. In the code and design part, we will check the implementation of design patterns.

1. Demo (60 Points)

Demonstration of building mode **-** 10 points.

Demonstration of power-ups **-** 10 points

- Extra time (2 points)
- Hint (2 points)
- Protection vest (2 points)
- Plastic bottle (2 points)
- Extra life (2 points)

Demonstration of time-wasting alien - 15 points:

- There are three different cases specified in the project document.
- Working as expected for each case is 5 points.

Demonstration of other aliens **-** 6 points (3 points each)

- Shooter alien (3 points)

- Blind alien (3 points)

Pause/Resume - 1 point

Login/Signup - 1 point

Help screen - 1 point

Save/Load - 15 points

Correct score-board update (time and lives) - 1 point

Code and Design (40 Points)

Correct implementation of controller - 10 pts

Adapter and Strategy - 14 pts

Model view Separation - 16 points (0, 8, 12, 16)

Final Report

Compile a report for your final project by combining all requirements and design documents.
Please make sure you adhere to the following rules.

1. Your submitted document should be in pdf format
2. It should have a consistent, single table of contents that indicates the contents of your requirements and design documents.
3. Nice font (please treat it as an official document and prepare it well.)
4. Your report should follow the following format:

- Title page
- Table of Contents
- Introduction (including the vision)
- Teamwork organization; you should discuss the work organization between team members briefly (for example: which member worked on which feature/Git branch).
- Use Case Diagrams
- Selected use case narratives (5 essential/complex selected use cases).
- System Sequence Diagrams (5 essential/complex selected)
- Operation Contracts (5 essential/complex selected)
- Sequence/Communication Diagrams (3 important/complex selected)from each)
- Class Diagrams (classes related to the sequence/communication diagrams)
- Package diagrams (layers and packages)
- Discussion of design alternatives and design patterns and principles. (including GRASP principles and GoF patterns).
- Supplementary Specifications.
- Glossary.