Cairo University Faculty of Computers and Information



**CS352 – Software Engineering II**

**Phase 1 Template**

**2017**

**Project Team**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Email** | **Mobile** |
| 20140291 | Nour Ahmed Abbas | nourahmed@stud.fci-cu.edu.eg | 01211196825 |
| 20140245 | Mahmoud Ahmed Mahmoud | [ouka.ahmed2@gmail.com](mailto:ouka.ahmed2@gmail.com) | 01025517515 |
| 20140225 | Mohamed Ragab Mohamed | mohamed225.mr@gmail.com | 01096023385 |
| 20140211 | Mazen Mohamed Sayed | mazen.mohamed.967@gmail.com | 01009066848 |
| 20140226 | Mohamed Rashad | mu\_ha\_mm\_ad@bk.ru |  |

**Staff:**

**Dr Amr Kamel** [a.kamel@fci-cu.edu.eg](mailto:a.kamel@fci-cu.edu.eg)

**Dr Khadiga Mohamed kelbedweihy@fci-cu.edu.eg**

**Eng Ragia Mohamed r.mohamed@fci-cu.edu.eg**

Contents

[Instructions [To be removed] 3](#_Toc476413280)

[Review Check List 3](#_Toc476413281)

[Testing 5](#_Toc476413282)

[Git repository link 6](#_Toc476413283)

**Design and Code Checklist**

**Design Principles**

1. Does the design follow SOLID principles?

-Open, close Principles: Teacher and student have the same data base, they should have one interface because in case of adding new type of account they will repeat the functions again.

- Class game doesn’t follow (S, O, D) principles in “gameType” value.

- Using the database as static class limits the future extensions in the database, and leads to unnecessary code repetition every time you need to extend the database…all of this violates(S,O,D) principles.

1. Does the design follow OOP rules?

- Encapsulation concept is broken due to the use of static values are values across the hole code

1. Is the design simple and easy to modify?

-no , 50%,use of static values across complicates the project

**Coding Standards**

1. Is the code understandable and readable?

Yes, 50%, the code is readable, but it is not very understandable because the repetitive usage of "static" keyword with classes.

1. Does the code follow Java Coding Style?

Yes, 90%, the static final values don’t follow the style.

1. Is indentation used properly?

Yes, 90% , indentation is used properly at most of the code.

1. Do variable have good names?

Yes, (100 %) All variable names are descriptive and understood.

**Comments**

1. Is the code commented enough?

No, (40%) Comments only exist in ‘some’ functions describing only particular cases in these functions.

There are no comments before all the functions describing their “Functionality, input & output”.

1. Is every class and method commented?

No, because we have a lot of code not commented only 50% of code commented.

1. Do comments follow Javadoc style?

No, 100% there is not even one function that has comment on it.

1. Is Javadoc generated for all the code?

No, Javadoc is generated in some parts not all the code, 50%.

1. Are there useless / wrong comments?

Yes, (5%) There are no wrong comments, but there exist little number of unused comments used during testing the output of some functions.

There is a whole unused class commented and left in the code.

**Code Structure**

1. Does the code follow the design precisely?

Yes, 80%, in play game sequence the field never call the database class, instead it return local array of games.

1. Are there very long classes or methods?

No, (0%) All classes don’t exceed 200 lines at max and most of the functions don’t exceed 50 lines at max.

1. Is there repeated code? (put in a function)

No, no repeated code in same class %0.

**Error Handling**

1. Does the code handle errors and exceptions?

Yes, 60% , Exceptions are handled, but most of wrong user input errors are not handled

1. Is defensive programming used to avoid errors?

Defensive programming is not used completely to avoid error because it doesn't handle some corner cases such as when adding new game - T/F questions the answer doesn't validate if it is garbage or not , and some setters function doesn't check if the data entered is valid or not, 80%.

**Logic**

1. Do loops have correct conditions and bounds?

All loops have correct conditions and bounds, 100%.

1. Do loops always terminate?

Yes, all loops will terminate all the time, 100%.

**Overall**

1. **Are the design and code of good quality?**

**-75%**

# Testing

**Example**

* 1. **UserTesting class**

|  |  |  |  |
| --- | --- | --- | --- |
| **Number** | **Testing function** | **Description** | **Result** |
| **1.** | **getQuestion (int idx)** | **The function is tested using addQuestion(string, string, string) function, getQuestion(int) function fails to retrieve a question if the passed index is out of boundary.** | **Failed** |
| **2.** | **search (string name)** | **This code add more than one game with the same name,so the search return wrong index for the game** | **Failed** |
| **3.** | **addStudent(Strudent)** | **This code add more than one student with the same username,and password, and email.** | **failed** |
| **4** | **Verify(string username,string password** | **This code add more than one student with the same username,and password, and email.so when one of them login is appear that just the old account is login** | **failed** |

# Git repository link

https://github.com/supernour09/testing.git