Cairo University  
Faculty of Computers and Information



**CS251**

**Software Engineering I**

Game to Gain

Software Requirements Specifications

Nada Ashraf

Michael Wageuh

Mina Nabil

Monica Millad

November 2016

Contents

[Team 3](#_Toc402452669)

[Document Purpose and Audience](#_Toc402452670) 4

[Introduction](#_Toc402452671) 4

[Software Purpose](#_Toc402452672) 4

[Software Scope 4](#_Toc402452673)

[Definitions, acronyms, and abbreviations](#_Toc402452674) 5

[Requirements](#_Toc402452675) 5

[Functional Requirements](#_Toc402452676) 5

[Non Functional Requirements](#_Toc402452677) 7

[System Models](#_Toc402452678) 8

[Use Case Model](#_Toc402452679) 8

[Use Case Tables](#_Toc402452680) 9

[Ownership Report](#_Toc402452681) 21

# 

# Team

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Email** | **Mobile** |
| 20140289 | Nada Ashraf Ahmed | nadanodyy@ymail.com | 01100591922 |
| 20140213 | Michael Wageuh Ramzy | michaelwageuh12@hotmail.com | 01200634332 |
| 20140287 | Mina Nabil William | mnabil405@gmail.com | 01142642703 |
| 20140281 | Monica Millad Aziz | Monicamillad18@gmail.com | 01271223662 |

### Git repository link:

### https://github.com/NadaAshrafAhmed/SWE\_1-project

# Requirements Analysis Document

# 1. Document purpose and audience

## 1.1. Purpose

This document includes the requirements analysis. It illustrate the functional and the nonfunctional requirements of the software, as it will be the reference between the client and the developers when any conflict or problem appears.

## 1.2. Audience

The audience of this documents are the client, the users, and the developers who are working on this project.

# 2. Introduction

## 2.1. Purpose of the system

The purpose of the Game to Gain project Is to develop a website called Game to Gain. The website will include educational games, three types of games; match picture game, multiple choice game, run code game. Instead of the normal boring ways of teaching specific topics this website will give the students an interesting way to learn math, science, coding ….

The website will offer its services for free so that any student be able to use it.

## 2.2. Scope of the system

## The website mainly developed to allow students to play games that will help them learn new things. It also provides them with some extra features like; they can add games in a favorite list, they can rate a game, write comment on it and they can show their score for each game and their total score.

## The system also include other user, it’s the teacher. The teacher have the ability to create a game, update or remove a game that he/she had created before, and he/she can try any game on the website.

## 2.3. Definitions, acronyms, and abbreviations

Game to Gain

The name of the educational website that will include the games.

# 3. Requirements

## 3.1. Functional Requirements

The Game to Gain website supports two types of users: the student and the teacher. The functions that are able for the student are starting the web application, creating account, logging in, choosing category, choosing the game, Make his/her favorite list, starting the game, rating game, logging out, and the functions that are able for the teacher are starting the web application, creating account, logging in, creating a game, upload a game, updating game, removing a game, choosing category, choosing game, starting the game, logging out. And the functional requirements are:

### 3.1.1. Creating account

The user must create account on the website to be able to use the website. But he/she should determine how he will deal with the website, as a student or as a teacher. The user will enter his/her basic information like; name, password, E-mail, age, gender.

### 3.1.2. Logging in

To start the service on the website the user should log in with his/her username and password and the system will verify them and send the response back.

### 3.1.3. Choosing the category

### Before the user will choose the game he/she should first choose the category of the game. It could be math, science, history, coding or anything that the website will include.

### 3.1.4. Choosing the game

After the student had chosen the category several games will appear in a tab and he/she will choose one of them to start playing, also the teacher could choose anyone of them to try it.

### 3.1.5. Save a game in his/her Favorite List

### Each student will have a favorite list it will be blank by default but he/she could add any game he/she want in it at any time so he/she could find it again easily.

### 3.1.6. Playing the game

### The student starts any game by choosing the game and then pressing the play button that will appear in the game page.

### 

### 3.1.7. Exiting the game

The student could stop the game by pressing the exit button in the game menu bar and then he/she will go back his/her home page of the account.

### 3.1.8. Rating game

### There is an additional feature in all the games, that the student could rate the game as interesting, normal or boring according to his/her opinion.

### 3.1.9. Creating a game

### This feature available only for the teachers. The website have three standards matching picture game, multiple choice game, and run code game, any teacher could create his/her own game according to any of the three standards. Multiple teachers can create the game together but they should have each other approvals on the website first.

### 3.1.10. Updating a game

### The website gives the ability to the teacher to update any game that he/she created before.

### 3.1.11. Removing a game

### The teacher can remove any game he/she had created before. Also the system have the right to remove any game if it was found boring within a specific criteria and according to the students rating of course.

### 3.1.12. Show recommendation list

### There will be a recommendation list the system will generate in each category, the student can show it and choose any game in it to start playing.

### 3.1.13. Update score

### The AI Computer get the new score of the User in this game. The System add this score to the previous game scores list and then sort this list and keep the first 5 scores only and then update the User total score according to the scores in the new list

### 3.1.14. Show score

### After the User finishing the game the System returns the player's score and the best 5 scores he got previously. The User could show the total score of the games; as The System display the total score of the User and the score of each game that has been played before.

### 3.1.15. Insert comment

### The User have the ability to insert a comment on any game. The System in return notify the Creator of the game that there is a User has commented on his/her game so he/she could reply on the comment.

### 3.1.16. Create score board

### The System will create a score board showing the best 100 Players on the website.

### 3.1.17. Show score board

### The User have the ability to show this score board by choosing its tab in the toolbar.

### 3.1.18. Logging Out

### After the user finishes his/her work on the website he/she will log out from the account.

## 

## 3.3. Nonfunctional Requirements

### 3.3.1. Usability

* The names of the tabs are clear so it’s easy to know what does each tab leading to.
* There won’t be a lot of steps while signing up, it would be just one form to be filled so it will be the minimum number of clicks.
* The user could make the favorite list in a very easy way, he/she should only write the name of the game and its category and press add.
* It’s very easy for the user to reach the game he/she wants, as the website is divided according to the categories.

### 3.3.2. Reliability

* Regardless to the number of users the website will handle it, and the crash error will not exceed 10% of the runtime.

### 3.3.3. Performance

* The loading time of the website after logging in must be smaller than 30 seconds. And the response time after the user chooses the game till the start of the game will be between 10 and 15 seconds.

### 

### 3.3.4. Supportability

* While updating or deleting a game the users that are playing it at the same time will still be able to continue the game but the icon of the game will be hided from other users.
* The system should be stable while uploading a new game.

### 3.3.5. Implementation

* The implementation of the back end phase will be done with php language, and the front end will be done with html and css.
* We could use templates.

### 3.3.6. Interface

* The user interface will be easy as we are dividing the website into tabs with clear names and also the games are divided into categories which will make the user reach his/her desirable game easier.
* Also we chose the colors of the background, tabs, and diagrams according to statistics that determine those kinds of website how they should look like (UX).

### 

### 3.3.7. Packaging

The user don’t need to download anything as the only needed thing having account on the website.

### 3.3.8. Legal

* All the games are for free, except for some advanced like those games that will teach them topics in really short time.

## 

## 4. System models

### 4.1. Use case model

The following diagram clarify the use cases that we will illustrate in the next section.

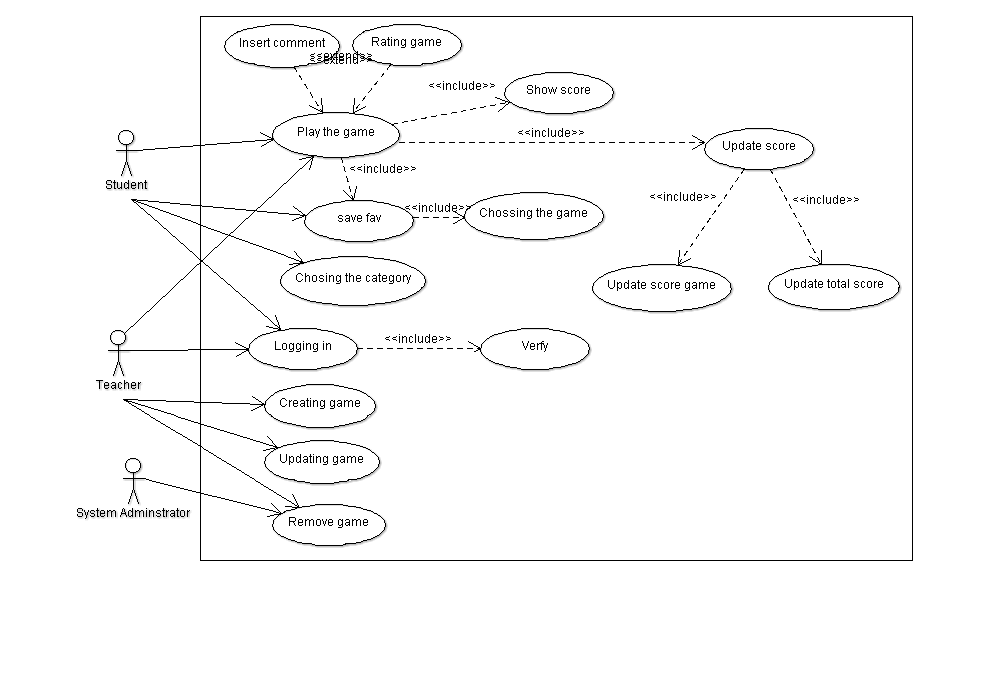


Figure 1: Overview of Game to Gain use cases (UML use case diagram).

**4.1.1. Actors**

**Student:** The student is the user who is able to play a game, rate it, show his/her score.

**Teacher:** The teacher is the user who is able to create, update, delete a game and he/she also can try a game.

**System Administrator:** It’s the one who controls the whole system and it has the ability to remove any game if it was badly rated.

**4.2. Use case tables**

**4.2.1. Logging in**

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 1 | |
| Use Case Name: | Logging in | |
| Actors: | User(Student,Teacher) | |
| Pre-conditions: | The website has been opened | |
| Post-conditions: | Display the website content | |
| Flow of events: | **User Action** | **System Action** |
| 1. User open the Website |  |
|  | 2. System display Welcome page |
| 3. User click on sign in tab |  |
|  | 3. System display page which allow to user to enter Email and Password |
| 4. User Enter Email and Password |  |
|  |  | 5. System Verify user data by using Verify Use case |
|  |  | 6. System display content of website |
|  | 7. User can access the website |  |
| Exceptions: | **User Action** | **System Action** |
| 1. User Enter Email and Password |  |
|  | 2. Invalid User data |
|  |  | 3. System shows an error message and allow the user to enter again |
|  | 4. If User enter Invalid data 3 times |  |
|  |  | 5. System will block this user |
| Includes: | Verify | |

**4.2.2. Choose a Category**

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 2 | |
| Use Case Name: | Choose a Category | |
| Actors: | User(Student,Teacher) | |
| Pre-conditions: | The User is logged in | |
| Post-conditions: | List of the game of the chosen category | |
| Flow of events: | **User Action** | **System Action** |
|  | 1. After the System open the home page which contain a list of the game categories |
| 2. User chooses one category from the list to display its games |  |
|  | 3. System returns the games included within that category |

**4.2.3. Choosing a game**

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 3 | |
| Use Case Name: | Choosing a game | |
| Actors: | User(Student,Teacher) | |
| Pre-conditions: | User has chosen a category | |
| Post-conditions: | The game will open | |
| Flow of events: | **User Action** | **System Action** |
| 1. After the User choose the category |  |
|  | 2. System display a list of games which are belong to this category |
| 3. User choose a game from the list by clicking on the game icon |  |
|  | 4. System loading a game and open it |
| Exceptions: | **User Action** | **System Action** |
| 1. If User click on two games |  |
|  | 2. System load the second one |

**4.2.4. Play a Game**

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 4 | |
| Use Case Name: | Play a game | |
| Actors: | User(Student,Teacher) | |
| Pre-conditions: | Choosing a game | |
| Post-conditions: | Close the game | |
| Flow of events: | **User Action** | **System Action** |
| 1. The user log in |  |
|  | 2. The system open the home page that contain a list of the available categories |
| 3. The user chose the category he/she want to play in |  |
|  | 4. The system open a list of all the games in the chosen category |
| 5. The user select the game he want to play |  |
|  |  | 6. The system open the game that the user selected |
|  | 7. The user start playing the game |  |
|  |  | 8. When the user finish playing the system display a message asking him if he wants to play the same game again? |
| Includes: | Choosing category, Show score, Update score, Choosing a game , save favorite , | |
| Exceptions: | **User Action** | **System Action** |
| 1. If it’s the first time the User plays the game there will be an option to show the demo of the game |  |
|  |  | 2. If the User chooses it the System starts to show it |

**4.2.5. Show Score**

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 5 | |
| Use Case Name: | Show Score | |
| Actors: | User(Student,Teacher) | |
| Pre-conditions: | The game is not running | |
| Post-conditions: | The game score displayed | |
| Flow of events: | **User Action** | **System Action** |
| 1. The User chooses the category |  |
|  | 2. The system returns the games within that category |
| 3. The User chooses the game |  |
|  | 4. The System opens that game |
| 5. The User plays the game |  |
|  |  | 6. After finishing the game the System returns the player's score and the best 5 scores he got previously |
| Exceptions: | **User Action** | **System Action** |
| 1. The User could show the total score of the games |  |
|  | 2. The System display the total score of the User and the score of each game that has been played before |

**4.2.6. Save to Favorite**

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 6 | |
| Use Case Name: | Save to Favorite | |
| Actors: | User(Student,Teacher) | |
| Pre-conditions: | The game is not running | |
| Post-conditions: | The game has been added to the Favorite list | |
| Flow of events: | **User Action** | **System Action** |
| 1. User select Category Tab |  |
|  | 2. System display list of Categories |
|  | 3. User choose the category |  |
|  |  | 4. System display list of games |
|  | 5. User select add to favorite list Which is a small tap beside the Game icon |  |
| Includes: | Choosing a category – Choosing a game | |

**4.2.7. Update Score**

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 7 | |
| Use Case Name: | Update Score | |
| Actors: | AI Computer | |
| Pre-conditions: | A user finish playing a game | |
| Post-conditions: | The game score & the total user score has been updated | |
| Flow of events: | **User Action** | **System Action** |
| 1. The AI Computer get the new score of the User in this game |  |
|  | 2. The System add this score to the previous game scores list and then sort this list and keep the first 5 scores only and then update the User total score according to the scores in the new list |
| Includes: | Update game score, update total user score | |

**4.2.8. Rate a Game**

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 8 | |
| Use Case Name: | Rate a Game | |
| Actors: | User(Student,Teacher) | |
| Pre-conditions: | The game is not running | |
| Post-conditions: | The game rate change | |
| Flow of events: | **User Action** | **System Action** |
| 1. If the User wants to rate a game he/she clicks a button named "rate that game" |  |
|  | 2. The System displays 3 chooses good, normal or bad |
| 3. The User chooses the rate of the game |  |
|  | 4. The System saves the rating of the User |
|  | 5. The System computes the new game rating and displays it to the game |

**4.2.9. Insert Comment**

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 9 | |
| Use Case Name: | Insert Comment | |
| Actors: | Student | |
| Pre-conditions: | The game is not running | |
| Post-conditions: | A new comment added to the game | |
| Flow of events: | **User Action** | **System Action** |
| 1. If the User wants to add a comment to the game, he/she goes to the comments area and starts to write his comment |  |
|  | 2. The System records the written text continuously |
| 3. After finishing writing a comment the User presses a submit button to save his comment |  |
|  | 4. The System saves the comment |
|  | 5. The System displays the comment with the User name at the comments area |
|  |  | 6. If the User who made this comment isn’t the game teacher The System send a notification to the game teacher that there is someone made a comment to his/her game |

**4.2.10. Creating Game**

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 10 | |
| Use Case Name: | Creating Game | |
| Actors: | Teacher | |
| Pre-conditions: | Creating an account as a teacher | |
| Post-conditions: | New game added to the system | |
| Flow of events: | **User Action** | **System Action** |
| 1. The Teacher chose that he want to create a new game |  |
|  | 2. The System list all the categories for the teacher to choose which of them he want to create his game into it |
| 3. The Teacher chose the category of the game he/she want to create the game into |  |
|  | 4. The System open a form containing some questions about the game (e.g. game name, description, number of levels in this game, easy normal or hard game, type of the game “Match pictures”- “Multiple Choices game”- “Run code game”  ,…) |
| 5. The Teacher answer the questions in the form |  |
|  |  | 6. The System ask the Teacher if he/she wants to create this game alone or he/she wants another teacher to create it with him/her |
|  | 7. The Teacher chooses if he/she wants to create this game alone or with other teachers |  |
|  |  | 7.1 If the Teacher chose that he want to create the game with other teachers the System ask the Teacher to enter the other teachers username |
|  | 7.2. The Teacher enters the other teachers usernames that he/she wants to create the game with |  |
|  |  | 7.3 The System sends an invitation to those teachers |
|  |  | 8. The System open the design page according to the chosen game type |
|  | 9. The Teacher(s) create the game design |  |
|  |  | 10. The System attaches demo for the Users who will play this game for the first time to easily understand how to play |
|  |  | 11. The System add the game to the chosen category and to the list of the games created by this teacher/s |

**4.2.11. Updating Game**

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 11 | |
| Use Case Name: | Updating Game | |
| Actors: | Teacher | |
| Pre-conditions: | Creating an account as a teacher & the game is already created and exist | |
| Post-conditions: | The game has been updated | |
| Flow of events: | **User Action** | **System Action** |
| 1. The Teacher chose that he/she wants to update a game |  |
|  | 2. The System lists all the games that this Teacher has created before |
| 3. The Teacher chose the game he/she wants to update |  |
|  | 4. The System creates a copy from the design and the implementation of the chosen game and let the teacher update on it |
| 5. The Teacher makes the changes he/she wants to do in order to update the game |  |
|  |  | 6. The System check if there are Students playing this game now if yes the System wait until they finish playing and during this if a new Student want to play the game he/she will play in the updated version, then the system will replace the old version by the new version |
| Includes: | Remove the game | |

**4.2.12. Remove Game**

|  |  |  |
| --- | --- | --- |
| Use Case ID: | 12 | |
| Use Case Name: | Remove Game, system administrator | |
| Actors: | Teacher, System Administrator | |
| Pre-conditions: | Creating an account as a teacher & the game is already created and exist | |
| Post-conditions: | The game has been removed from the list | |
| Flow of events: | **User Action** | **System Action** |
| 1. If the teacher chose that he/she wants to update a game |  |
|  | 2. The System lists all the games that this Teacher has created before |
| 3. The Teacher chose the game he/she wants to remove |  |
|  | 4. The System will hide the game from the user interface to prevent any new student want to play this game and then check if there are Students playing this game the System wait until they finish playing , then the System will remove the game |
| Exceptions: | **User Action** | **System Action** |
| 1. The System Administrator can remove a game according to specific criteria |  |
|  | 2. The System will hide the game from the user interface to prevent any new Student want to play this game and then check if there are Students playing this game the System wait until they finish playing , then it will remove the game |

# 5. Ownership Report

|  |  |
| --- | --- |
| **Item** | **Owners** |
| Document Purpose and Audience | Monica Millad |
| Introduction | Monica Millad |
| Requirements | All the team members |
| Use Case Model | All the team members |
| Use Case Tables | Michael Wageuh  Mina Nabil  Nada Ashraf |

# 