**Cairo University**
**Faculty of Computers and Information**

# CS251

# Software Engineering I

## IntelliFun

# Software Requirements Specifications

**Bayan Mokhtar**
**Ghada Othman**
**Menna Fateen**
**Salma Hassan**

**November 2016**

# CS251: Phase 1- BGMS-16
# Project: IntelliFun

# Software Requirements Specifications

## Contents

# CS251: Phase 1- BGMS-16
# Project: IntelliFun

# Software Requirements Specifications

## Team

| ID | Name | Email | Mobile |
|---|---|---|---|
| 20140350 | Mennatullah Seif Fateen | menna.fateen@stud.fci-cu.edu.eg | 01150449981 |
| 20140346 | Bayan Mokhtar Ali | bayanmokhtar@gmail.com | 01068233553 |
| 20140200 | Ghada Othman Sadek | ghada.othmaan@gmail.com | 01008663431 |
| 20140326 | Salma Hassan Othman | salma.hassan@stud.fci-cu.edu.eg | 01018373247 |

## Document Purpose and Audience

The main purpose of this document is to provide the reader with a comprehensive and thorough description of the project's functionality and features. The said features were documented after analyzing the client's needs. All functional and nonfunctional requirements that were formed as a result of the requirements elicitation process are included. The procedure in which the system and user would interact under all circumstances is likewise documented. Therefore, this document should act as the main reference for both the client and the developers during the software delivery lifecycle. Accordingly, the client, the developers and also the stakeholders of the project are the primary audience of this document.

## Introduction

### Software Purpose

The purpose of this software is simple. It is to be developed mainly to create a simultaneously enjoyable and educational environment for young students to learn and communicate while having fun. The students that are to play the games on the site can also compete with each other for a higher overall score. This should make the software engaging and appealing to young students. In addition, teachers can help the content of the website to thrive and include even more fun and challenging games by creating their own games in various subjects.

### Software Scope

This software system will be a game-based educational website for school students. It is designed to strengthen students' education and knowledge through an enjoyable and entertaining manner by providing games in different subjects that reinforce their understanding. Students will be able to choose a specific subject and play any game included in it while being able to view their overall score at any time. Moreover, they will be able to comment/rate any game. Teachers will be able to create new games using provided templates and under any subject category. They will also be able to edit or remove any of their own games.

# CS251: Phase 1- BGMS-16
# Project: IntelliFun

# Software Requirements Specifications

## Definitions, Acronyms, and Abbreviations

| Term | Definition |
| --- | --- |
| Student | Person who uses the website to play the games and learn in a fun educational environment. |
| Teacher | Person who uses the site to create games for students to use and help students. |
| User | Student or Teacher. |
| CAPTCHA | Stands for Completely Automated Public Turing Test to tell computers and humans apart.<br>A program that checks for spam by generating tests to tell humans and bots apart. |

## Requirements

### Functional Requirements

1. The system should allow the user to create an account and login. In the case of signing up, the system should display a registration form with the following fields: first name, last name, email, password and a submit button. When the submit button is pressed, an automatic email is sent to the user and the account is created. In case of signing in, entering wrong information whether by incorrect password or invalid email format, the system should display the message "invalid email or password. Please try again" until correct credentials are entered.

2. The system should allow the user to choose either a game type or a topic, if their choice is a game type, the system should list all the games under the chosen type, if their choice is a topic, the games listed should all be under the chosen topic. The system should allow the user to choose either a game type or a topic, if their choice is a game type, the system should list all the games under the chosen type, if their choice is a topic, the games listed should all be under the chosen topic. The user should then be able to play any game from the listed ones.

3. Update student score should be triggered when the game is finished. When the game is finished, a request is sent to the server that saves the game end-date and the player's end-score to the database.

4. The system should have ready-made game-templates for games to be created. The teacher should choose a game-template and continue with entering the game's instructions and pressing submit. When the submit button is pressed, a request is sent to the server that saves the game to the database.

5. The system should only allow users registered as teachers to create a new game using given game-templates.

# Software Requirements Specifications

6. The system allows a student to rate or comment to any game based on a 5-point system, 1 being the lowest score, and 5 being the highest. After the student plays said game, a window appears asking if they want to rate or leave a comment.
7. The system allows a teacher to edit a game only created by them. The system asks for their password as a verification before going through with the modification.
8. The system allows a teacher to remove a game only created by them. Prior to a teacher attempting to remove a game, the system should verify they're the creator of this game. The system should produce a dialog box that says "Are you sure you want to delete "game-name"?" as a verification before going through with the deletion. If the user confirmed, the game will be unlinked to their account and they will no longer be able to retrieve it. The backend will keep the game for retrieving data.
9. The system allows a user to view profiles. Whether it's their own profiles, or their colleagues.
10. The system keeps track of every game ratings, and automatically removes games at the bottom of said ratings-list with a latency of no greater than 12 months.
11. The system should allow a user to search for any game. Upon entering the search query and hitting search, the system should list all possible matches for the desired game.
12. The system should allow the user to choose a subject and view the games categorized under the chosen subject.

## Non Functional Requirements

**Usability**
- Simple to create an account; user needs just 3 steps to create the account.
- Easy to create a game; teacher can create a game using the provided templates easily.
- Simple to play a game; user should be able to learn how to play the game through the ongoing guidelines during play time.
- The system should always keep users informed about what is going on through appropriate feedback; all delays in the system longer than 0.5 seconds will produce a dialog box that says "Please wait".
- System interface should be easy to browse and use.

**Robustness**
- The system should be able to check the validation of the user inputs.

**Response Time**
- Authentication email should be sent within 1 minute maximally after the user creates the account.
- The response time during a game should not exceed 0.5 second.
- The game should be loaded and opened in less than 10 seconds.

## Software Requirements Specifications

- o The game should be uploaded to the system within 1 minute on average.

**Availability**
- o The percentage of system failure should not exceed 5%, the system should be available most of the time.

**Scalability**
- o System should be able to contain 100,000 games.
- o System be able to support up to 2000 game players simultaneously.

**Maintainability**
- o The system should be adaptive in order to update the game templates or types and fix its own problems.

**Safety**
- o The system should automatically backup all student scores and games created to ensure that all the data is safe from lost.
- o By using secured protocols, the system ensures that the information is securely transmitted to the server.

**Security**
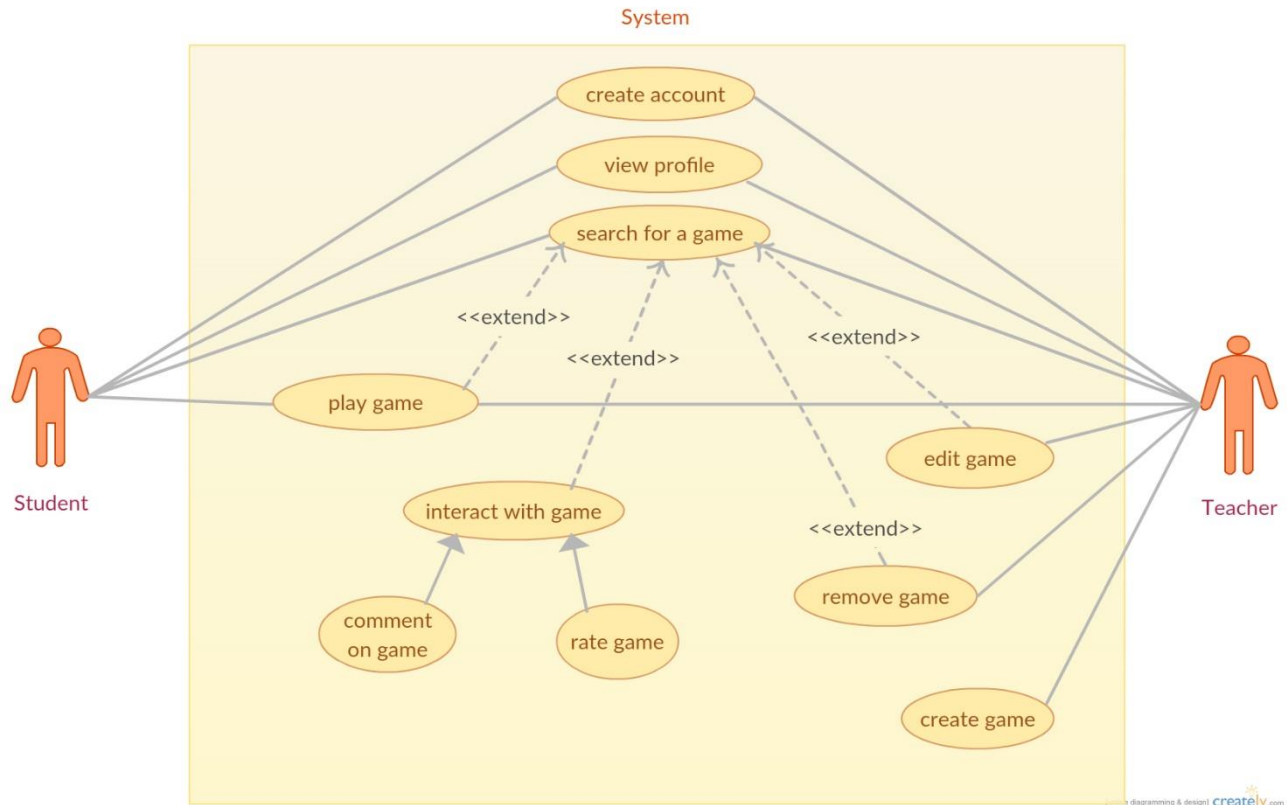- o The system should be able to protect all data from any unauthorized access or attacks.

# Software Requirements Specifications

## System Models

### Use Case Model

# Software Requirements Specifications

## Use Case Tables

| Use Case ID: | UC-1 | |
|---|---|---|
| Use Case Name: | Create Account | |
| Actors: | Student<br>Teacher | |
| Pre-conditions: | The user should have an email. | |
| Post-conditions: | User has created an account successfully. | |
| Flow of events: | **User Action** | **System Action** |
| | 1. The user opens the website and presses the sign up button. | |
| | | 2. The system opens the sign up page. |
| | 3. The user enters their first and last name. | |
| | | 4. System validates name by checking if there are any invalid characters or numbers. (e.g. name can't contain symbols like @, $, or #) |
| | 5. The user chooses a username and enters it in its textbox. | |
| | | 6. The system checks the availability of the username.<br>6.1. If the username is available a green mark is shown.<br>6.2. If the username is not available the user is asked to choose another one. |
| | 7. The user writes a password it in its textbox. | |
| | | 8. The system checks the validation of the password regarding the explained restrictions. |

# Software Requirements Specifications

| | | |
|---|---|---|
| | | 8.1. If the password is valid a green mark is shown.<br>8.2. If the password is not valid the user is asked to write another one. |
| | 9. The user chooses whether they are a teacher or a student. | |
| | | 10. The system saves the user's choice. |
| | 11. The user enters email in its textbox. | |
| | | 12. The system checks the validation of the email.<br>12.1. If the saved user choice is a teacher, the email should have a .edu format.<br>12.2. If the email is valid a green mark is shown<br>12.3. If the email is not valid (no @ is written) the user is asked to write another one. |
| | 13. The user presses the 'sign up' button. | |
| | | 14. The account is created and the system sends an authentication email to the specified email written by the user.<br>14.1. If the user clicks the authentication link within 2 days the account will be activated.<br>14.2. If the user doesn't click the authentication link within two days the link will be expired. Another email will |

# Software Requirements Specifications

| | | be sent instead if he tried to activate his account. |
|---|---|---|
| **Exceptions:** | **User Action** | **System Action** |
| | Authentication link is never clicked | |
| | | System deactivates the account linked to the email used. |

| Use Case ID: | UC-2 | |
|---|---|---|
| **Use Case Name:** | Play Game | |
| **Actors:** | Student<br>Teacher | |
| **Pre-conditions:** | The user should have an account first. | |
| **Post-conditions:** | The game is played and score is updated. | |
| **Flow of events:** | **User Action** | **System Action** |
| | 1. The user opens the website and logs in. | |
| | | 2. System opens the dashboard with the list of subjects and the grid of most played games. The system also allows the user to search for any game using tags. |
| | 3. The user chooses a specific subject category. | |
| | | 4. A list of the games classified under chosen category is shown. |
| | 5. The user chooses a game to play. | |
| | | 6. The system opens the game and loads it. |
| | 7. The user plays the game according to the specified instructions. | |

## Software Requirements Specifications

| | User Action | System Action |
|---|---|---|
| | 7.1. If the user presses exit button during the game before finishing the round. | |
| | | 8. The system calculates the score of the player as long as he is playing. <br> 8.1. The system exits the game without updating the user's score for this round. |
| | 9. The user finishes the whole round. | |
| | | 10. The system shows the score of the round to the user and asks the user to rate the game. |
| | 11. The user chooses to rate the game. | |
| | | 12. The system saves this rate for the user and recalculate the average rate of the game and saves it. |
| | 13. The user comments on the game. | |
| | | 14. The system saves the comment and notifies the creator of the game. |
| Exceptions: | **User Action** | **System Action** |
| | 1. The user opens the website and logs in. | |
| | | 2. The system validate the login as invalid and asks user to enter their username and password again. |
| Includes: | Rate game, comment on game, search for game | |

# Software Requirements Specifications

| Use Case ID: | UC-3 | |
|---|---|---|
| Use Case Name: | Create Game | |
| Actors: | Teacher | |
| Pre-conditions: | The user should have a teacher account first. | |
| Post-conditions: | The game is created successfully. | |
| Flow of events: | **User Action** | **System Action** |
| | 1. The teacher opens the website and logs in. | |
| | | 2. The system opens dashboard. |
| | 3. The teacher presses 'create a game' button. | |
| | | 4. The system opens the page of creating games. |
| | 5. The teacher chooses a game template. | |
| | | 6. The system opens the chosen template. |
| | 7. The teacher enters game title, subject, and specific topic tags. | |
| | | 8. The system saves the information and proceeds to entering game data page. |
| | 9. The teacher enters the data of the game according to the chosen game category and clicks 'create game' button. | |
| | | 10. The system saves the data, creates the game, and adds it to teacher's profile. |
| Exceptions: | **User Action** | **System Action** |
| | 1. The teacher opens the website and logs in. | |
| | | 2. The system validate the login as invalid and asks teacher to enter |

# Software Requirements Specifications

| | | their username and password again. |
|---|---|---|

| Use Case ID: | UC-4 | |
|---|---|---|
| Use Case Name: | Rate Game | |
| Actors: | Student - Teacher | |
| Pre-conditions: | The user should have an account | |
| Post-conditions: | The average rating of the game is updated | |
| Flow of events: | **User Action** | **System Action** |
| | 1. The user opens the website and logs in. | |
| | | 2. The dashboard is opened the list of subjects appears and a grid of most played games. The system allows the user to search for a game. |
| | 3. The user chooses a subject | |
| | | 4. A list of the games classified under chosen category is shown. |
| | 5. The user chooses a game to open | |
| | | 6. The system opens the game and loads it and the rating button appears on top of the game. |
| | 7. The user rates the game | |
| | | 8. The rating the user entered is saved and average rate is recalculated, appeared and saved. |
| Exceptions: | **User Action** | **System Action** |
| | 1. The user opens the website and logs in. | |

## Software Requirements Specifications

| | | 2. The system validates the login as invalid and asks teacher to enter their username and password again. |
|---|---|---|
| Includes: | Search for game | |

| Use Case ID: | UC-5 | |
|---|---|---|
| Use Case Name: | Comment on game | |
| Actors: | Student Teacher | |
| Pre-conditions: | The user should have an account | |
| Post-conditions: | The comment is added to the game's discussion | |
| Flow of events: | User Action | System Action |
| | 1. The user opens the website and logs in. | |
| | | 2. The dashboard is opened, the list of subjects appears and so does a grid of most played games. The system allows the user to search for a game. |
| | 3. The user chooses a subject | |
| | | 4. A list of the games classified under chosen category is shown. |
| | 5. The user chooses a game to open. | |
| | | 6. The system opens the game and loads it and the comment box appears below the game. |
| | 7. The user enters his comment in the textbox. | |
| | | 8. The system gives the user a CAPTCHA box to protect discussion box from spam |

## Software Requirements Specifications

| | | |
|---|---|---|
| | 9. The user solves the spam test | |
| | | 10. The system adds the user's comment to the discussion. |
| Exceptions: | **User Action** | **System Action** |
| | 1. The user opens the website and logs in. | |
| | | 2. The system validates the login as invalid and asks teacher to enter their username and password again. The system allows the user to search for a game. |
| | 3. The user fails at passing the CAPTCHA text and gives up | |
| | | 4. The comment is not added to the discussion |
| Includes: | Search for game | |

| Use Case ID: | UC-6 | |
|---|---|---|
| Use Case Name: | Edit game | |
| Actors: | Teacher | |
| Pre-conditions: | The teacher has an account and has previously created a game. | |
| Post-conditions: | The new changes or edits are saved. | |
| Flow of events: | **User Action** | **System Action** |
| | 1. The teacher opens the website and logs in to their account | |
| | | 2. The dashboard is opened, the list of subjects appears as well as a grid of most played games. The system allows the user to search for a game. |
| | 3. The teacher chooses a subject | |

# Software Requirements Specifications

| | | |
|---|---|---|
| | | 4. A list of the games classified under chosen category is shown. |
| | 5. The teacher chooses the game to open | |
| | | 6. If the game is not created by the teacher, the system does not show any edit options.<br>7. If the game is created by the teacher, the edit options are shown. |
| | 8. The teacher edits the game and enters new changes. | |
| | | 9. The new changes are saved by the system and the teacher is notified of the successful edit. |
| **Exceptions:** | **User Action** | **System Action** |
| | 1. The teacher opens the website and logs in. | |
| | | 2. The system validates the login as invalid and asks teacher to enter their username and password again. |
| **Includes:** | Search for game | |

| Use Case ID: | UC-7 |
|---|---|
| Use Case Name: | Remove game |
| Actors: | Teacher |
| Pre-conditions: | The teacher has an account and has previously created a game. |
| Post-conditions: | The game previously created is removed from the site. |

# Software Requirements Specifications

| Flow of events: | User Action | System Action |
|---|---|---|
| | 1. The teacher opens the website and logs in to their account | |
| | | 2. The dashboard is opened, the list of subjects appears as well as a grid of most played games. The system allows the user to search for a game. |
| | 3. The teacher chooses a subject | |
| | | 4. A list of the games classified under chosen category is shown. |
| | 5. The teacher chooses the game to open | |
| | | 6. If the game is not created by the teacher, the system does not show any edit options. 7. If the game is created by the teacher, the edit options are shown. |
| | 8. The teacher chooses the delete/remove button. | |
| | | 9. The system prompts the teacher and asks them if they are sure they want to remove the game. |
| | 10. The teacher checks the 'yes' button. | |
| | | 11. The system unlinks the game from their account and they will no longer be able to retrieve it |
| Exceptions: | User Action | System Action |
| | 1. The teacher opens the website and logs in. | |
| | | 2. The system validates the login as invalid and asks teacher to |

# Software Requirements Specifications

|  |  | enter their username and password again. |
|---|---|---|
|  | 3. The teacher does not confirm the remove prompt. |  |
|  |  | 4. The system does not remove the game. |
| Includes: | Search for game |  |

| Use Case ID: | UC-9 | |
|---|---|---|
| Use Case Name: | Search for game | |
| Actors: | Teacher <br> Student | |
| Pre-conditions: | The user has an account | |
| Post-conditions: | The game searched for is found | |
| Flow of events: | **User Action** | **System Action** |
|  | 1. The user opens the website and logs in to their account |  |
|  |  | 2. The dashboard is opened, and the search box appears. |
|  | 3. The user enters the name or search tags for a game in the search textbox. |  |
|  |  | 4. The system searches for the game using the tags entered by the user and shows the results of the search. |
| Exceptions: | **User Action** | **System Action** |
|  | 1. The user opens the website and logs in. |  |
|  |  | 2. The system validates the login as invalid and asks user to enter their username and password again. |

# CS251: Phase 1- BGMS-16
# Project: IntelliFun

## Software Requirements Specifications

### Ownership Report

| Item | Owners |
|------|--------|
| Requirements | *Bayan Mokhtar – Ghada Othman* |
| Use case tables | *The whole team* |
| Document purpose, software purpose and definitions | *Menna Fateen* |
| Use case model & software scope | *Salma Hassan* |

o **Github repository link: https://github.com/mennafateen/IntelliFun**