Cairo University  
Faculty of Computers and Information



**CS251**

**Software Engineering I**

ST-Comm

Software Design

Ahmed Hussein Karam

Ahmed Mohamed Ahmed

Mariam Ashraf Fekry

Andrew Emad Nassif

December 2016

Contents

[Team 3](#_Toc468575261)

[Document Purpose and Audience 3](#_Toc468575262)

[System Models 3](#_Toc468575263)

[I. Class diagrams 4](#_Toc468575264)

[II. Sequence diagrams 6](#_Toc468575266)

[Class - Sequence Usage Table 10](#_Toc468575267)

[Ownership Report 20](#_Toc468575268)

[References 20](#_Toc468575270)

[Authors 20](#_Toc468575271)

# Team

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Email** | **Mobile** |
| 20140015 | Ahmed Hussein Karam | ahkcsit@gmail.com | 01023355193 |
| 20140035 | Ahmed Mohamed Ahmed | aabdelmeged70@yahoo.com | 01200875427 |
| 20140263 | Mariam Ashraf Fekry | mariamashraf096@gmail.com |  |
| 20140081 | Andrew Emad Nassif | andrewen2010@yahoo.com | 01271122945 |

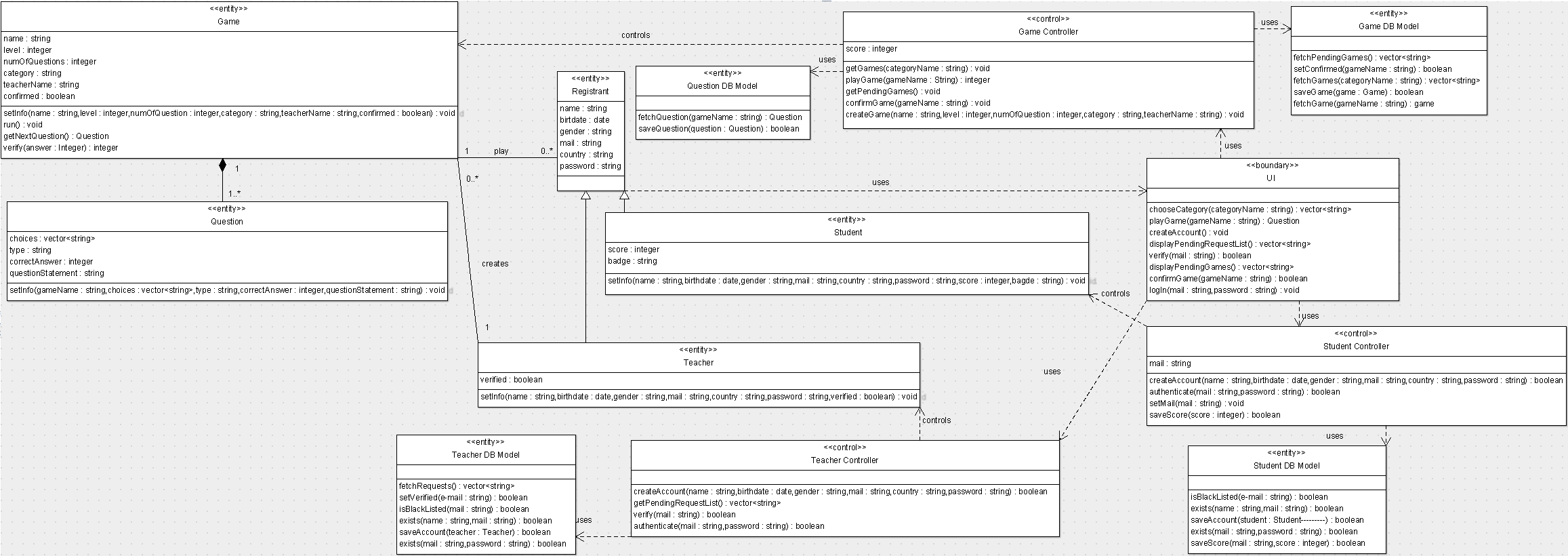
# Document Purpose and Audience

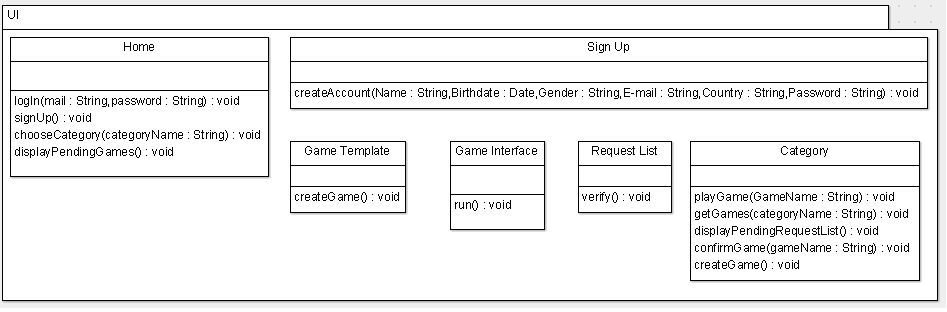
This document targets development team and the customer. It illustrates website static structure through a class diagram. It also illustrates the dynamic behavior and interaction among objects through sequence diagrams. Those sequence diagrams provide the following scenarios:

* Creating a game
* Playing a game
* Confirming a game
* Creating student account
* Creating teacher account
* Verifying teacher account

# System Models

## I. Class diagrams

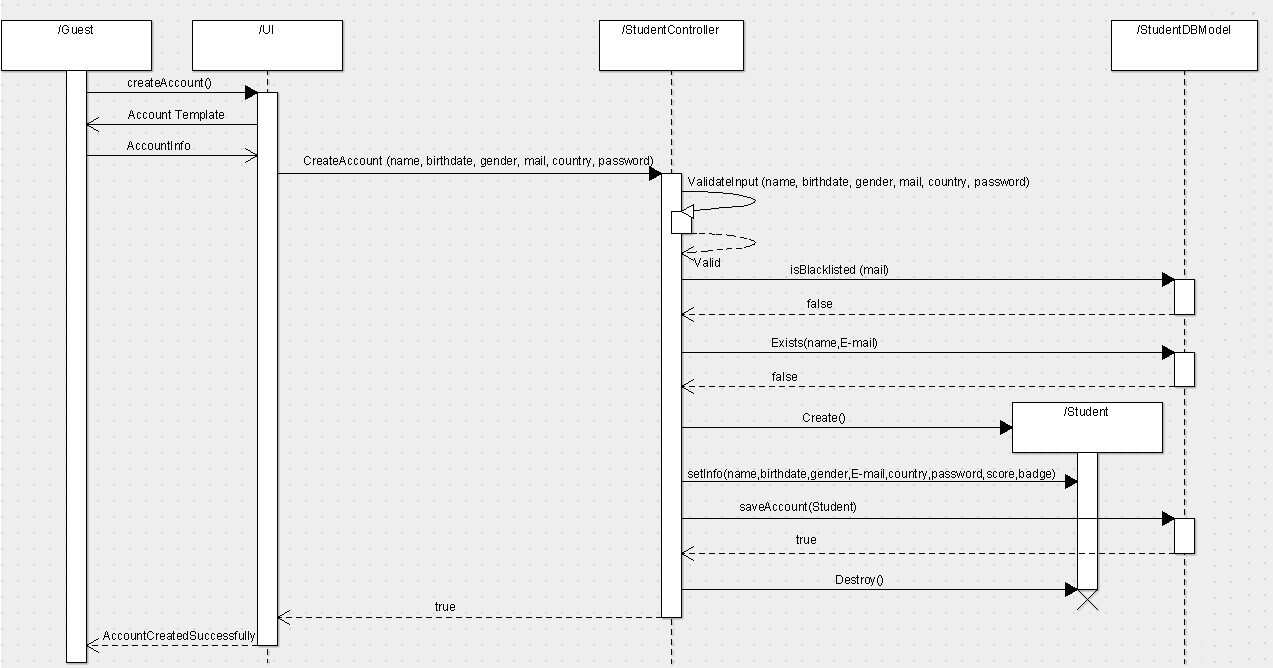




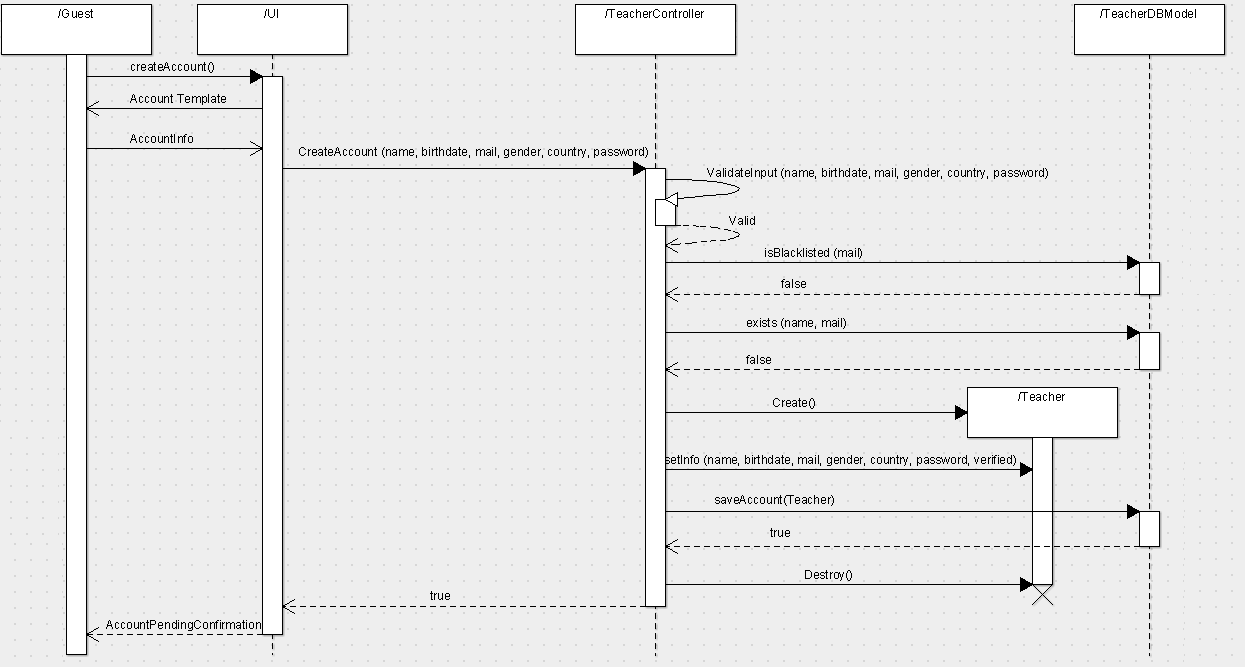
| **Class ID** | **Class Name** | **Description & Responsibility** |
| --- | --- | --- |
| 1 | Registrant | * Keeps the data of the currently logged in user * Acts as parent for Student and Teacher classes * This class contains attributes common for all user types |
| 2 | Teacher | * Keeps the data of the currently logged in teacher * Data kept in this class complements those inherited from Registrant class |
| 3 | Student | * Keeps the data of the currently logged in student * Data kept in this class complements those inherited from Registrant class |
| 4 | UI | A package that provides interfaces that enable registrants choose from the following options:   * Log in to the website * Choose game category * Create an account * Play a game * Verify created teacher accounts (for administrators only) * Confirm a new game (for administrators only) |
| 5 | Game | * Responsible for running game * Provides users with questions stored in the game and checks for answer correctness |
| 6 | Question | * Keeps question type (MCQ, True/ False), question statement, choices and correct answer |
| 7 | Game Controller | * Provides functionalities used to create and play games * Enables administrators to confirm created games |
| 8 | Student Controller | * Controls student authentication to the website * Enables Students to create new accounts * Keeps the mail of the current player |
| 9 | Teacher Controller | * Controls teacher authentication to the website * Enables Teachers to create new accounts * Enables administrators to verify created teacher accounts |
| 10 | Game DBModel | Uses the database to:   * Search for games * Save new games * Manipulate game data |
| 11 | Student DBModel | Uses the database to:   * Search for student blacklisted mails * Search for student accounts * Save student scores and account data |
| 12 | Teacher DBModel | Uses the database to:   * Search for teacher blacklisted mails * Search for teacher accounts * Manipulate teacher data * Save teacher account data * Enable administrators to fetch teacher pending accounts to be verified |
| 13 | Question DBModel | Uses the database to:   * Save a new question * Get an existing question |

## II. Sequence diagrams

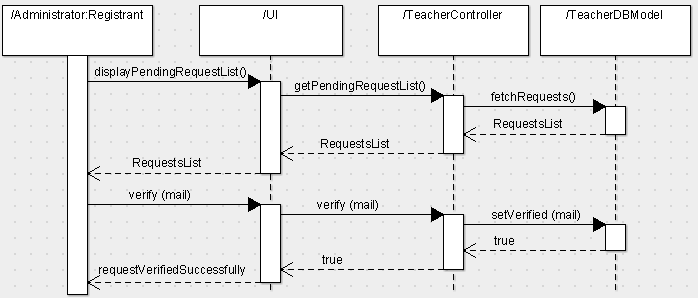
1

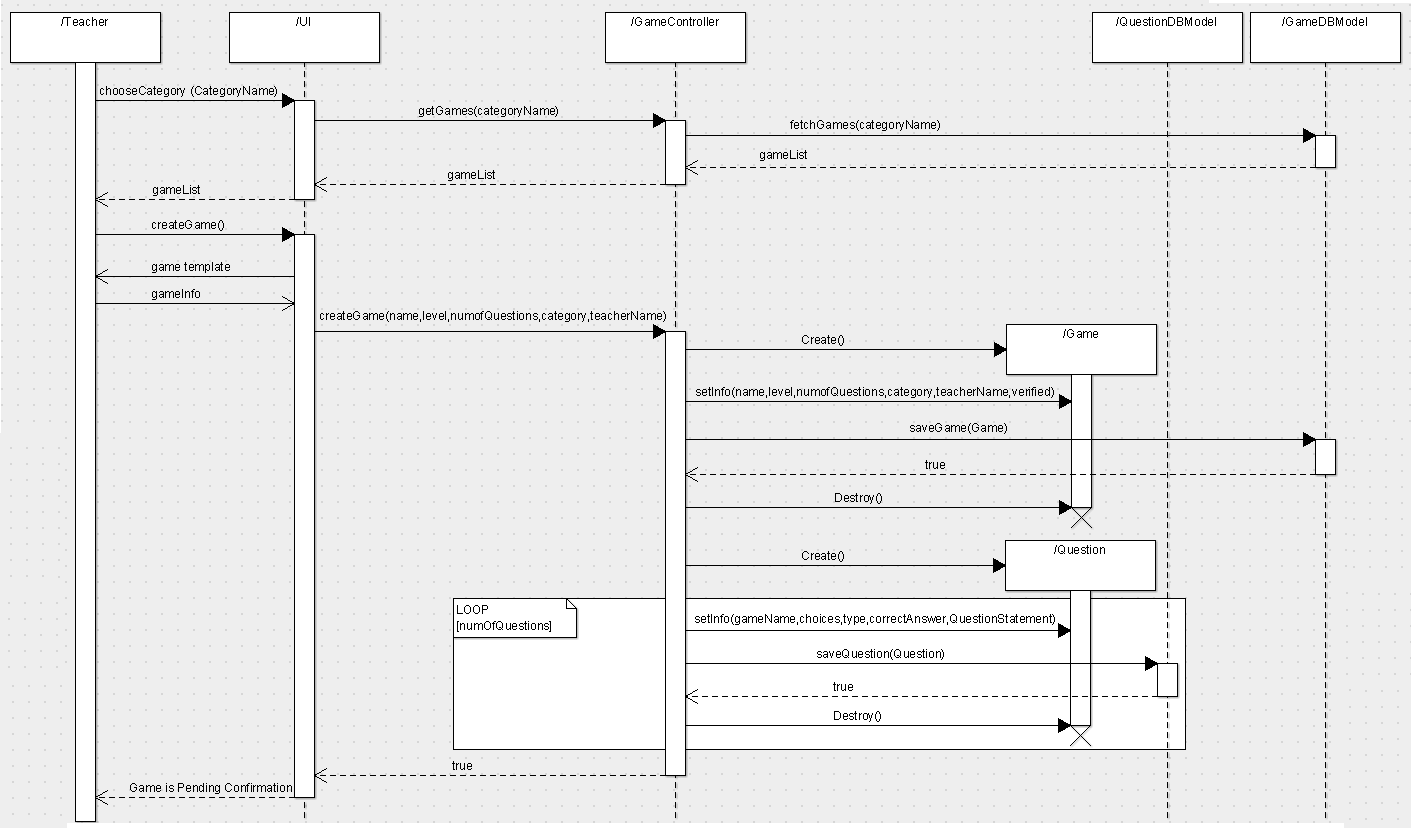


2



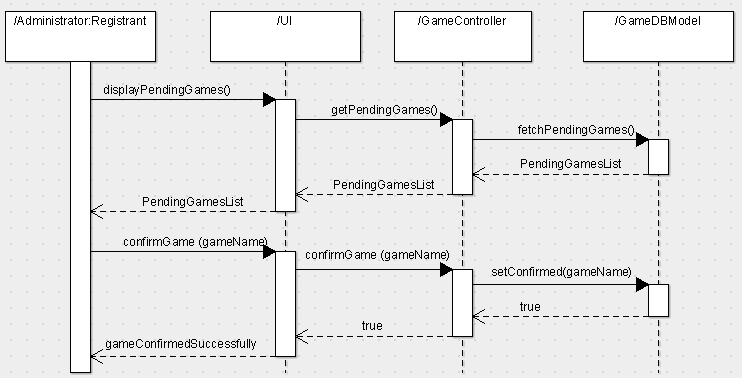
3

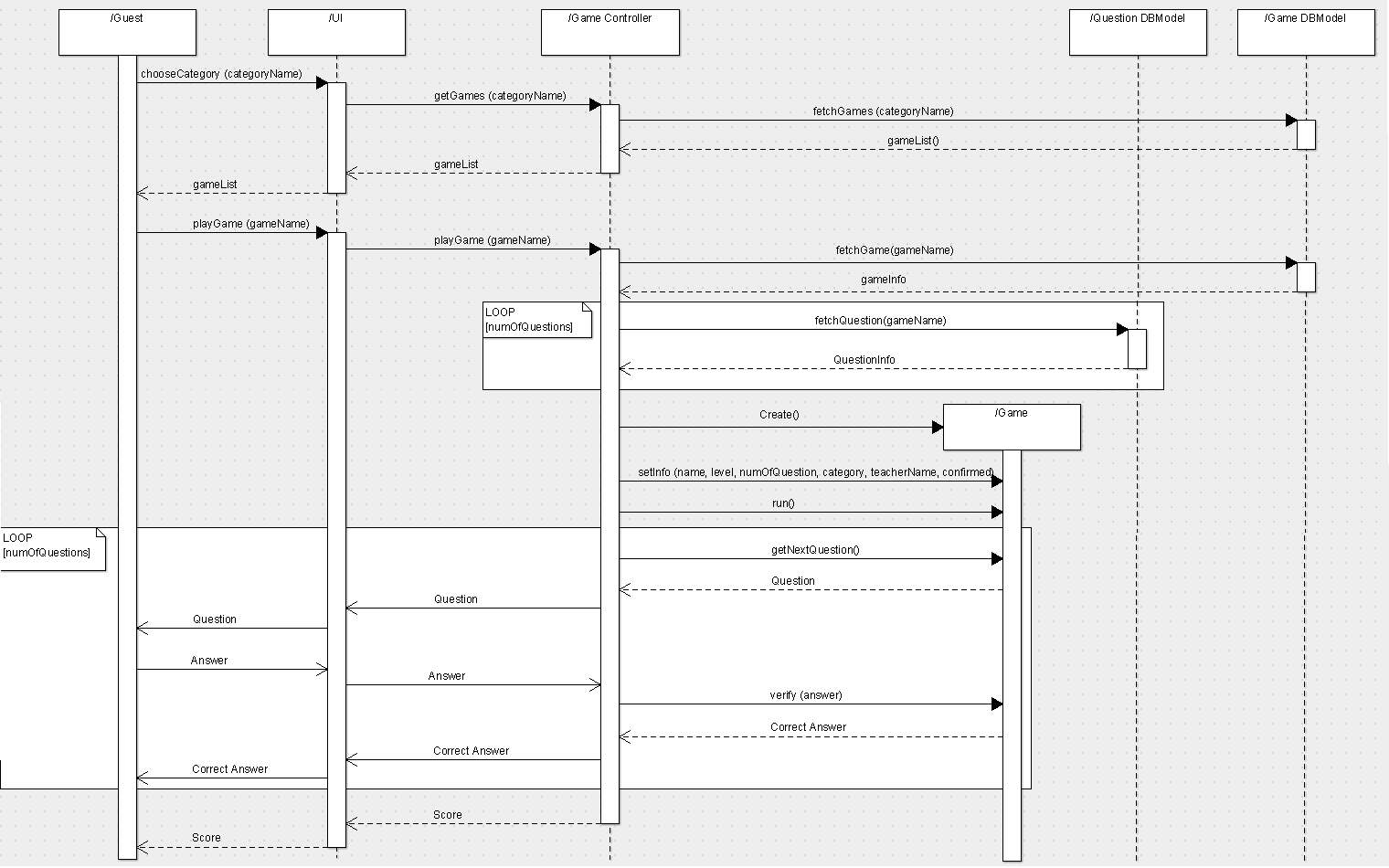




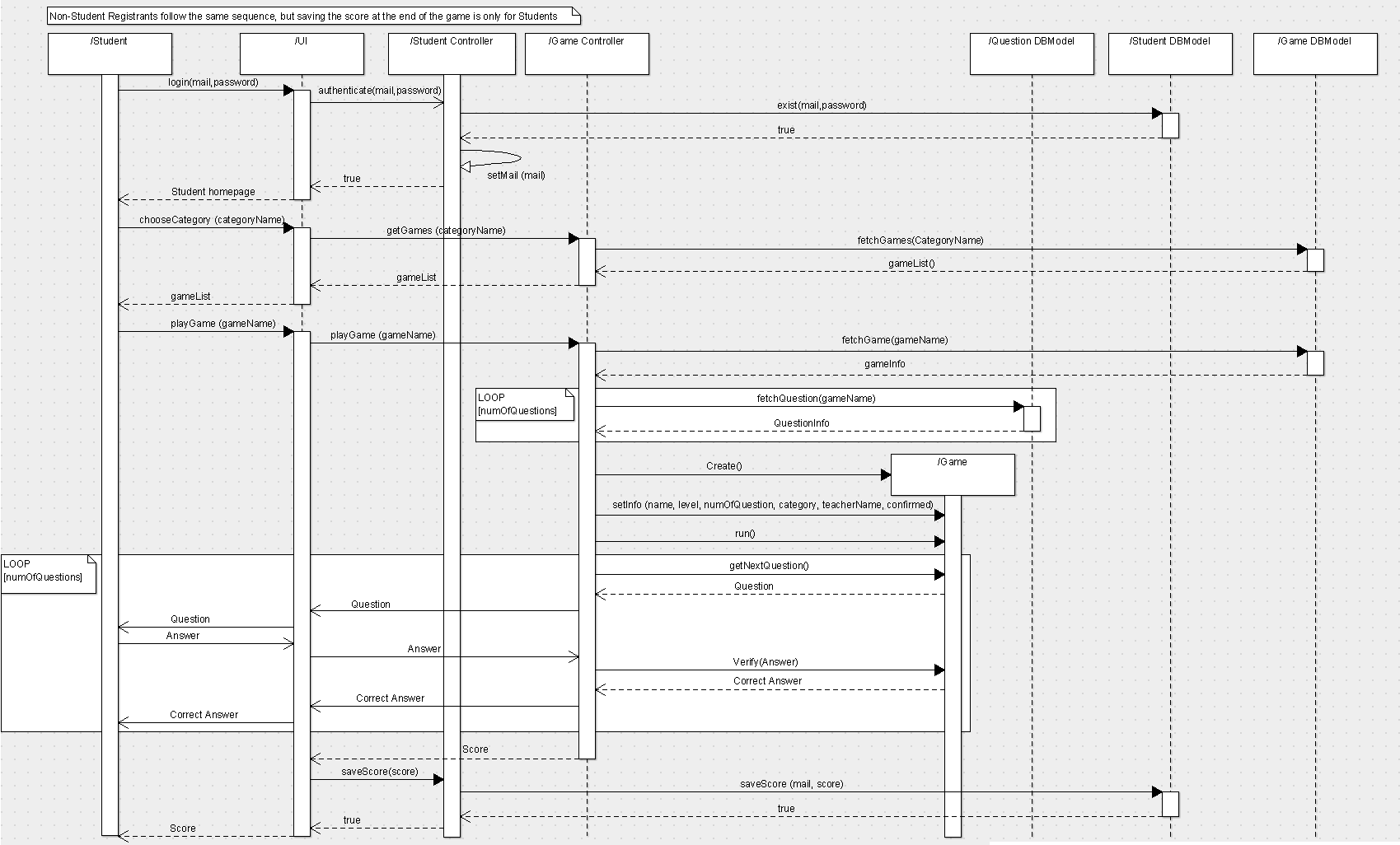
4

5





6



7

### Class - Sequence Usage Table

| **Class Name** | **Sequence Diagrams** | **Overall used methods** |
| --- | --- | --- |
| Student | 1 | **Sequence 1:**  createAccount()  createAccount(name,birthdate,gender,E-mail,country,password)  validateInput(name,birthdate,gender,E-mail,country,password)  isBlacklisted(E-mail)  exists(name,E-mail)  setInfo(name,birthdate,gender,E-mail,country,password,score,badge)  saveAccount(Student) |
| Teacher | 2 | **Sequence 2:**  createAccount()  createAccount(name,birthdate,E-mail,gender,country,password)  validateInput(name,birthdate,E-mail,gender,country,password)  isBlacklisted(E-mail)  exists(name,mail)  setInfo(name,birthdate,E-mail,gender,country,password,verified)  saveAccount(teacher) |
| Game | 4,6,7 | **Sequence 4:**  chooseCategory(categoryName)  getGames(categoryName)  fetchGames(categoryName)  createGame()  createGame(name,level,numofQuestions,category,teacherName)  setInfo(name,level,numofQuestions,category,teacherName,verified)  saveGame(game)  setInfo(gameName,choices,type,correctAnswer,QuestionStatement)  saveQuestion(question)  **Sequence 6:**  chooseCategory(categoryName)  getGames(categoryName)  fetchGames(categoryName)  playGame(gameName)  playGame(gameName)  fetchGame(gameName)  fetchQuestion(gameName)  run()  getNextQuestion()  verify(answer)  **Sequence 7:**  login(mail,password)  authenticate(mail,password)  exist(mail,password)  setMail(email)  chooseCategory(categoryName)  getGames(categoryName)  fetchGames(categoryName)  playGame(gameName)  playGame(gameName)  fetchGame(gameName)  fetchQuestion(gameName)  getNextQuestion()  saveScore(score)  verify(answer)  saveScore(mail,score)  setInfo(name,level,numofQuestions,category,teacherName,verified) |
| Question | 4 | **Sequence 4:**  chooseCategory(categoryName)  getGames(categoryName)  fetchGames(categoryName)  createGame()  createGame(name,level,numofQuestions,category,teacherName)  setInfo(name,level,numofQuestions,category,teacherName,verified)  saveGame(game)  setInfo(gameName,choices,type,correctAnswer,QuestionStatement)  saveQuestion(question) |
| UI | 5,4,1,2,6,7,3 | **Sequence 5:**  displayPendingGames()  getPendingGames()  fetchPendingGames()  confirmGame(gameName)  confirmGame(gameName)  **Sequence 4:**  chooseCategory(categoryName)  getGames(categoryName)  fetchGames(categoryName)  createGame()  createGame(name,level,numofQuestions,category,teacherName)  setInfo(name,level,numofQuestions,category,teacherName,verified)  saveGame(game)  setInfo(gameName,choices,type,correctAnswer,QuestionStatement)  saveQuestion(question)  **Sequence 1:**  createAccount()  createAccount(name,birthdate,gender,E-mail,country,password)  validateInput(name,birthdate,gender,E-mail,country,password)  isBlacklisted(E-mail)  exists(name,E-mail)  setInfo(name,birthdate,gender,E-mail,country,password,score,badge)  saveAccount(student)  **Sequence 2:**  createAccount()  createAccount(name,birthdate,E-mail,gender,country,password)  validateInput(name,birthdate,E-mail,gender,country,password)  isBlacklisted(E-mail)  exists(name,mail)  setInfo(name,birthdate,E-mail,gender,country,password,verified)  saveAccount(teacher)  **Sequence 6:**  chooseCategory(categoryName)  getGames(categoryName)  fetchGames(categoryName)  playGame(gameName)  playGame(gameName)  fetchGame(gameName)  fetchQuestion(gameName)  run()  getNextQuestion()  verify(answer) |
|  |  | **Sequence 7:**  login(mail,password)  authenticate(mail,password)  exist(mail,password)  setMail(email)  chooseCategory(categoryName)  getGames(categoryName)  fetchGames(categoryName)  playGame(gameName)  playGame(gameName)  fetchGame(gameName)  fetchQuestion(gameName)  getNextQuestion()  saveScore(score)  verify(answer)  saveScore(mail,score)  setInfo(name,level,numofQuestions,category,teacherName,verified)  **Sequence 3:**  displayPendingRequestList()  getPendingRequestList()  fetchRequests()  verify(E-mail)  verify(E-mail)  setVerified(E-mail) |
| Game Controller | 5,4,6,7 | **Sequence 5:**  displayPendingGames()  getPendingGames()  fetchPendingGames()  confirmGame(gameName)  confirmGame(gameName)  **Sequence 4:**  chooseCategory(categoryName)  getGames(categoryName)  fetchGames(categoryName)  createGame()  createGame(name,level,numofQuestions,category,teacherName)  setInfo(name,level,numofQuestions,category,teacherName,verified)  saveGame(game)  setInfo(gameName,choices,type,correctAnswer,QuestionStatement)  saveQuestion(question)  **Sequence 6:**  chooseCategory(categoryName)  getGames(categoryName)  fetchGames(categoryName)  playGame(gameName)  playGame(gameName)  fetchGame(gameName)  fetchQuestion(gameName)  run()  getNextQuestion()  verify(answer)  **Sequence 7:**  login(mail,password)  authenticate(mail,password)  exist(mail,password)  setMail(email)  chooseCategory(categoryName)  getGames(categoryName)  fetchGames(categoryName)  playGame(gameName)  playGame(gameName)  fetchGame(gameName)  fetchQuestion(gameName)  getNextQuestion()  saveScore(score)  verify(answer)  saveScore(mail,score)  setInfo(name,level,numofQuestions,category,teacherName,verified) |
| Teacher Controller | 2,3 | **Sequence 2:**  createAccount()  createAccount(name,birthdate,E-mail,gender,country,password)  validateInput(name,birthdate,E-mail,gender,country,password)  isBlacklisted(E-mail)  exists(name,mail)  setInfo(name,birthdate,E-mail,gender,country,password,verified)  saveAccount(teacher)  **Sequence 3:**  displayPendingRequestList()  getPendingRequestList()  fetchRequests()  verify(E-mail)  verify(E-mail)  setVerified(E-mail) |
| Student Controller | 1,7 | **Sequence 1:**  createAccount()  createAccount(name,birthdate,gender,E-mail,country,password)  validateInput(name,birthdate,gender,E-mail,country,password)  isBlacklisted(E-mail)  exists(name,E-mail)  setInfo(name,birthdate,gender,E-mail,country,password,score,badge)  saveAccount(student)  **Sequence 7:**  login(mail,password)  authenticate(mail,password)  exist(mail,password)  setMail(email)  chooseCategory(categoryName)  getGames(categoryName)  fetchGames(categoryName)  playGame(gameName)  playGame(gameName)  fetchGame(gameName)  fetchQuestion(gameName)  getNextQuestion()  saveScore(score)  verify(answer)  saveScore(mail,score)  setInfo(name,level,numofQuestions,category,teacherName,verified) |
| Game DBModel | 5,4,6,7 | **Sequence 5:**  displayPendingGames()  getPendingGames()  fetchPendingGames()  confirmGame(gameName)  confirmGame(gameName)  **Sequence 4:**  chooseCategory(categoryName)  getGames(categoryName)  fetchGames(categoryName)  createGame()  createGame(name,level,numofQuestions,category,teacherName)  setInfo(name,level,numofQuestions,category,teacherName,verified)  saveGame(game)  setInfo(gameName,choices,type,correctAnswer,QuestionStatement)  saveQuestion(question)  **Sequence 6:**  chooseCategory(categoryName)  getGames(categoryName)  fetchGames(categoryName)  playGame(gameName)  playGame(gameName)  fetchGame(gameName)  fetchQuestion(gameName)  run()  getNextQuestion()  verify(answer)  **Sequence 7:**  login(mail,password)  authenticate(mail,password)  exist(mail,password)  setMail(email)  chooseCategory(categoryName)  getGames(categoryName)  fetchGames(categoryName)  playGame(gameName)  playGame(gameName)  fetchGame(gameName)  fetchQuestion(gameName)  getNextQuestion()  saveScore(score)  verify(answer)  saveScore(mail,score)  setInfo(name,level,numofQuestions,category,teacherName,verified) |
| Question DBModel | 4,6,7 | **Sequence 4:**  chooseCategory(categoryName)  getGames(categoryName)  fetchGames(categoryName)  createGame()  createGame(name,level,numofQuestions,category,teacherName)  setInfo(name,level,numofQuestions,category,teacherName,verified)  saveGame(game)  setInfo(gameName,choices,type,correctAnswer,QuestionStatement)  saveQuestion(question)  **Sequence 6:**  chooseCategory(categoryName)  getGames(categoryName)  fetchGames(categoryName)  playGame(gameName)  playGame(gameName)  fetchGame(gameName)  fetchQuestion(gameName)  run()  getNextQuestion()  verify(answer)  **Sequence 7:**  login(mail,password)  authenticate(mail,password)  exist(mail,password)  setMail(email)  chooseCategory(categoryName)  getGames(categoryName)  fetchGames(categoryName)  playGame(gameName)  playGame(gameName)  fetchGame(gameName)  fetchQuestion(gameName)  getNextQuestion()  saveScore(score)  verify(answer)  saveScore(mail,score)  setInfo(name,level,numofQuestions,category,teacherName,verified) |
| Teacher DBModel | 2,3 | **Sequence 2:**  createAccount()  createAccount(name,birthdate,E-mail,gender,country,password)  validateInput(name,birthdate,E-mail,gender,country,password)  isBlacklisted(E-mail)  exists(name,mail)  setInfo(name,birthdate,E-mail,gender,country,password,verified)  saveAccount(teacher)  **Sequence 3:**  displayPendingRequestList()  getPendingRequestList()  fetchRequests()  verify(E-mail)  verify(E-mail)  setVerified(E-mail) |
| Student DBModel | 1,7 | **Sequence 1:**  createAccount()  createAccount(name,birthdate,gender,E-mail,country,password)  validateInput(name,birthdate,gender,E-mail,country,password)  isBlacklisted(E-mail)  exists(name,E-mail)  setInfo(name,birthdate,gender,E-mail,country,password,score,badge)  saveAccount(student)  **Sequence 7:**  login(mail,password)  authenticate(mail,password)  exist(mail,password)  setMail(email)  chooseCategory(categoryName)  getGames(categoryName)  fetchGames(categoryName)  playGame(gameName)  playGame(gameName)  fetchGame(gameName)  fetchQuestion(gameName)  getNextQuestion()  saveScore(score)  verify(answer)  saveScore(mail,score)  setInfo(name,level,numofQuestions,category,teacherName,verified) |

# Ownership Report

|  |  |
| --- | --- |
| **Item** | **Owners** |
| Document Purpose and Audience | All Team Members |
| Class diagrams |
| Sequence diagrams 1, 2 | Mariam Ashraf |
| Sequence diagrams 3, 5 | Andrew Emad |
| Sequence diagram 4 | Ahmed Mohamed |
| Sequence diagram 6 | Ahmed Hussein |
| Class - Sequence Usage Table | Ahmed Mohamed |
| Ownership Report | Ahmed Hussein |

GitHub Repository: <https://github.com/AndrewEmad/ST-Comm.git>

# References

* http://www.mhhe.com/engcs/compsci/pressman/graphics/Pressman5sepa/common/cs1/design.pdf

# Authors

* Mostafa Saad and Mohammad El-Ramly (Edited by Mohamed Samir)