Progetto di Ingegneria del Software:

Design

D'Amicis Salvatore

Marone Christian

Romeo Matteo

Urzino Davide

Indice

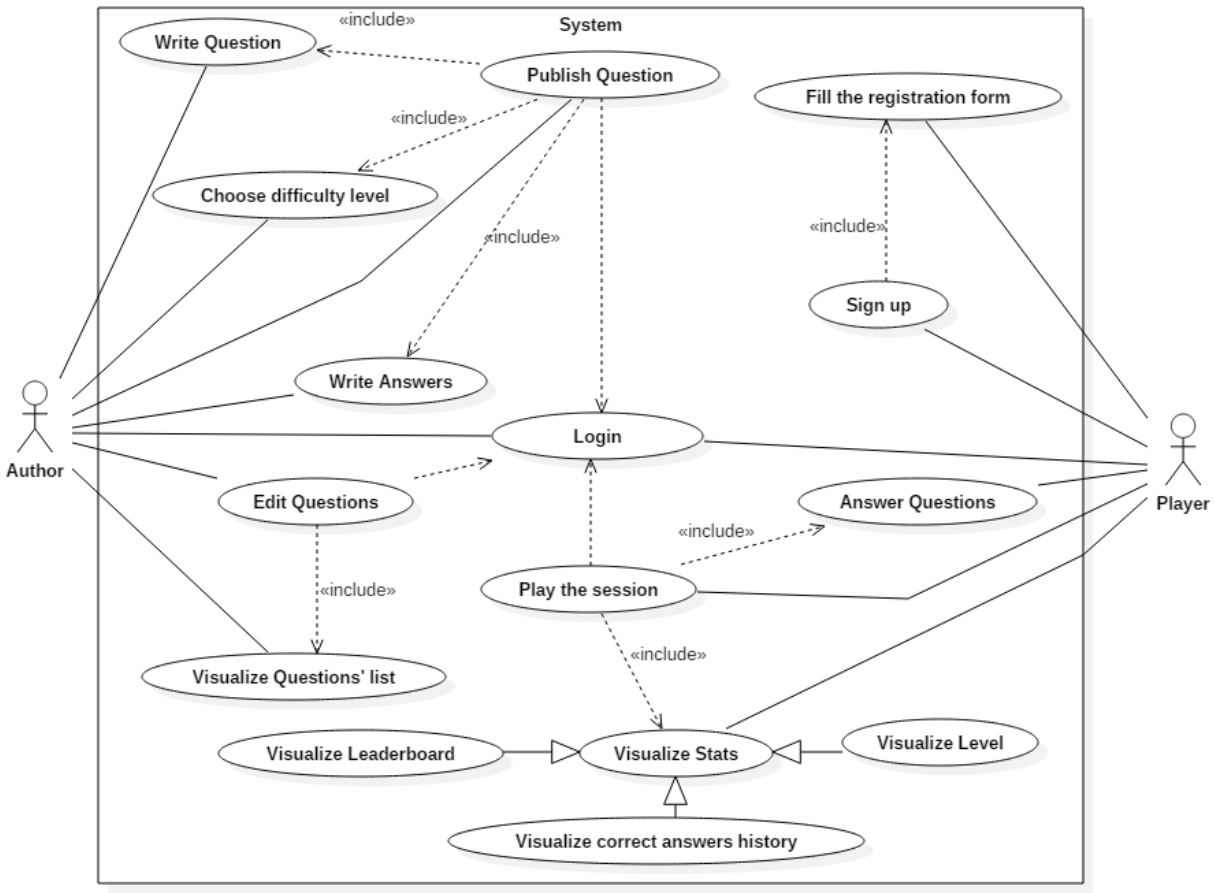
Introduzione

Fasi di lavoro

La stesura dell'UML di Quooz ha il ruolo di modellizzare in che modo sarà implementato il software. Per questo motivo, usufruendo dei diagrammi che questo "linguaggio" mette a disposizione è stato possibile delineare il comportamento del programma. Dopo aver immaginato come ogni singola funzionalità dovesse essere implementata, questa è stata trasferita in uno o più diagrammi al fine di rappresentarla. Perciò è stato ritenuto opportuno fare: uno Use Case Diagram, un Class Diagram, due Activity Diagrams, due Statechart Diagrams, due Sequence Diagrams, due Collaboration Diagrams, un Object Diagram, un Component Diagram ed un Deployement Diagram.

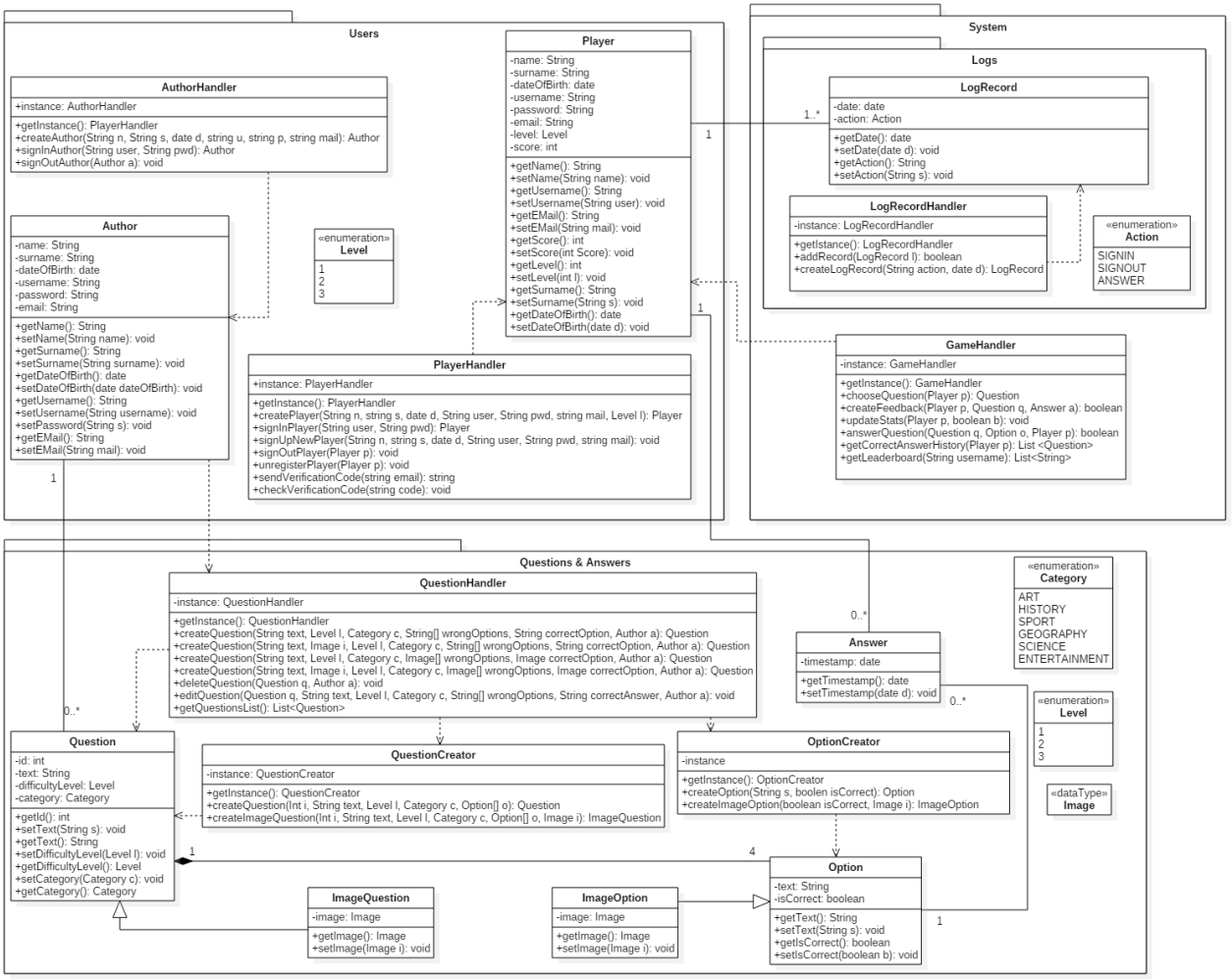
UML

UseCase Diagram



Decisi i due attori abbiamo individuato tutte le attività ad essi collegate. L'**Autore** ha il compito di aggiungere e modificare le domande dopo aver effettuato il login, il **giocatore**, invece, deve effettuare la registrazione inserendo tutte le credenziali. Dopo aver effettuato il login, può iniziare una sessione di gioco rispondendo alle domande poste dal sistema. Al giocatore è inoltre permesso di visualizzare tutte le sue **statistiche**: livello, domande con risposta corretta e la classifica con i punteggi di tutti i giocatori.

Class Diagram



Partendo dagli elementi individuati nei requisiti sono state realizzate le seguenti classi fondamentali: **Question, Answer, Option, Author, Player, LogRecord**; correlate delle seguenti classi per gestirle: **QuestionHandler, QuestionCreator, OptionCreator, PlayerHandler, GameHandler, LogRecordHandler, AuthorHandler.**

Question

La classe **Question** ha una serie di attributi che descrivono e categorizzano una domanda. Essa è supportata dalla classe **QuestionCreator** che contiene una serie di metodi volti a istanziare oggetti di tipo Question. Inoltre **QuestionHandler** ha il compito di permettere ad un **Author** di creare, modificare domande ed ottenere la **QuestionList**. Ogni istanza di Question è composta da 4 oggetti di tipo Option di cui uno solo con attributo isCorrect uguale a true. La sottoclasse **ImageQuestion** sarà implementata allo scopo di generare domande con immagine. L'associazione tra Question ed Author serve ad evidenziare il fatto che una domanda è stata scritta da un determinato autore.

Answer

Questa classe rappresenta la risposta data da un Player ad una domanda posta dal sistema. L'unico attributo per questa classe è **TimeStamp** che ha la funzione di catturare l'istante in cui una risposta è data dall'utente. Questo attributo ha lo scopo di supportare tutte quelle funzionalità che richiedono di sapere quando è stata data una risposta.

Option

Option è la classe che identifica una delle 4 possibili risposte per una domanda. **text** ed **isCorrect** sono rispettivamente il testo ed un flag che esprime la veridicità dell'opzione. Un opzione può essere anche di tipo immagine, per questo sarà implementata la sottoclasse **ImageOption**. **OptionCreator** invece sarà addetta a creare istanze appartenenti ad Option. **QuestionHandler**, di cui abbiamo già parlato in precedenza, ha anche l'utilità di correlare una domanda alle sue opzioni.

Author

Author implementa la figura dell'autore. La sua gestione è presa in carico da **AuthorHandler**, nelle fasi di sign-in/out. La classe Author per mezzo di **QuestionHandler** creerà e modificherà tutte le domande che verranno fornite nel gioco. Anche la possibilità di visualizzare la **QuestionList** sarà implementata attraverso la medesima classe.

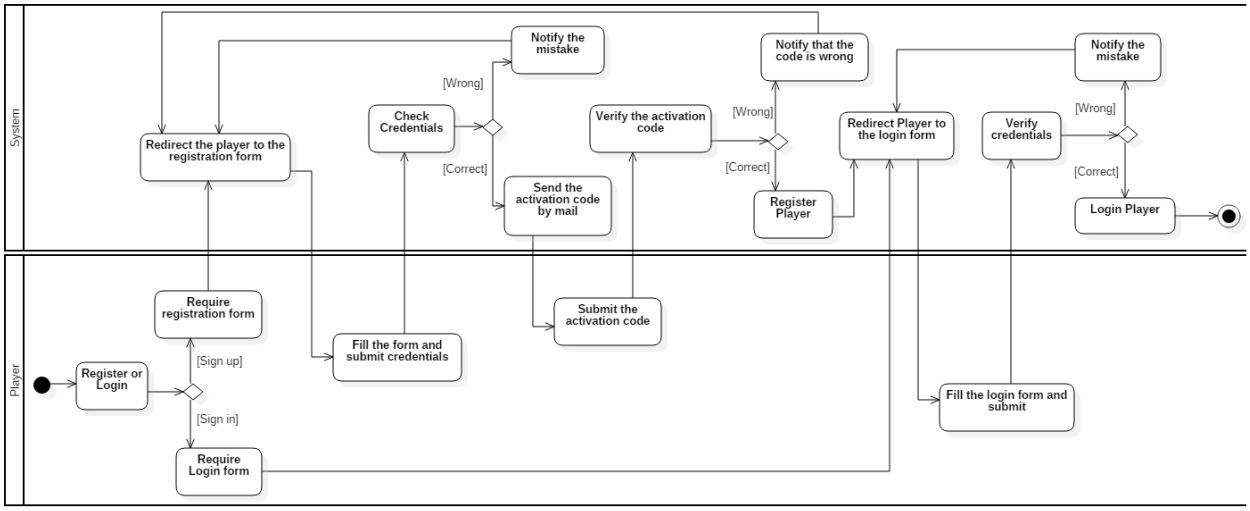
Player

La seguente classe interpreta il giocatore all'interno del gioco. **PlayerHandler** registrerà e permetterà l'accesso ai giocatori verificandone le credenziali. **GameHandler** invece sceglie la domanda per il giocatore, ne prende la risposta e restituisce un feedback. Inoltre aggiorna per lui le statistiche e quando interpellato gliele mostra. Tutte le sue azioni sono registrate in un giornale di log di cui si preoccuperà la classe **LogRecordHandler.**

LogRecord

LogRecord identifica le azioni fatte dai giocatori nel gioco. Ogni istanza di questa classe rappresenta un evento di sign-in/out o risposta a domanda fatta da un Player. **LogRecordHandler** ha una serie di metodi che quando chiamati da **GameHandler** e **QuestionHandler** danno origine ad oggetti di tipo **LogRecord** che verranno utilizzati per il mantenimento del file di log.

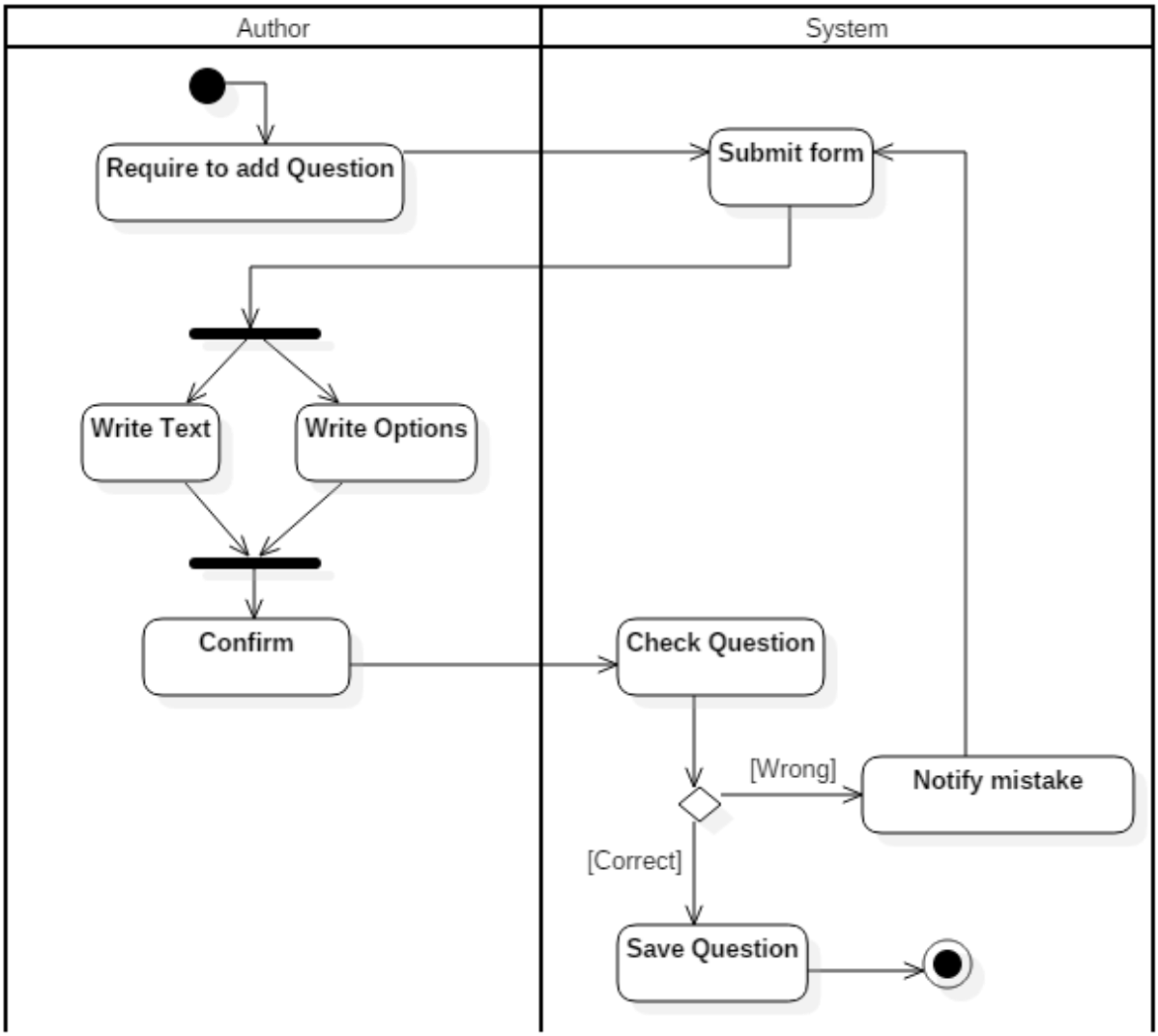
Activity Diagram - Login



Questo Activity Diagram mostra tutte le attività che l'utente e il sistema devono completare per effettuare il login. Per prima cosa l'utente ha la possibilità di effettuare il **log-in** oppure, nel caso in cui non sia già registrato, di fare **sign-up**.

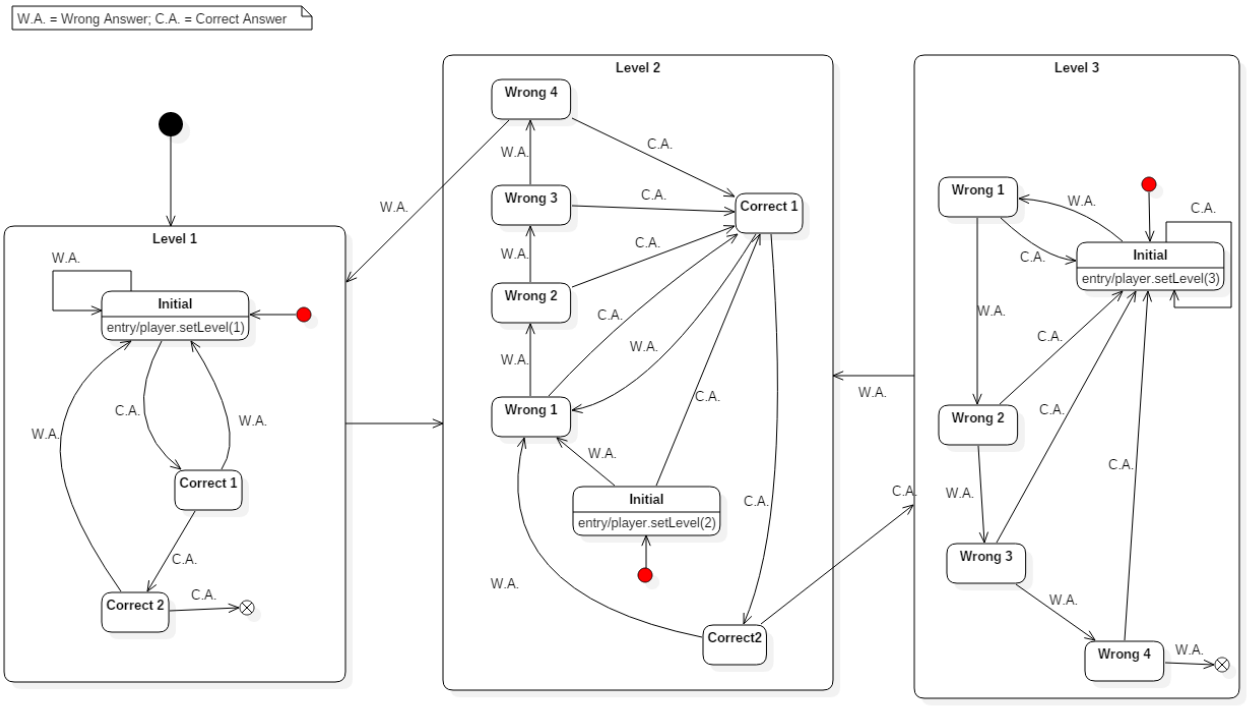
Procedendo con la fase di registrazione, dopo aver inserito tutte le credenziali, una volta controllate dal sistema, se corrette viene inviata una **E-mail** contenente un codice di attivazione. Quest'ultimo dovrà essere inserito dal nuovo utente per completare la registrazione. A questo punto il sistema reindirizza il giocatore al modulo di login dove potrà inserire **username** con **password**. Dopo una verifica da parte del sistema, il giocatore potrà iniziare una sessione di gioco.

Activity Diagram – Add Question



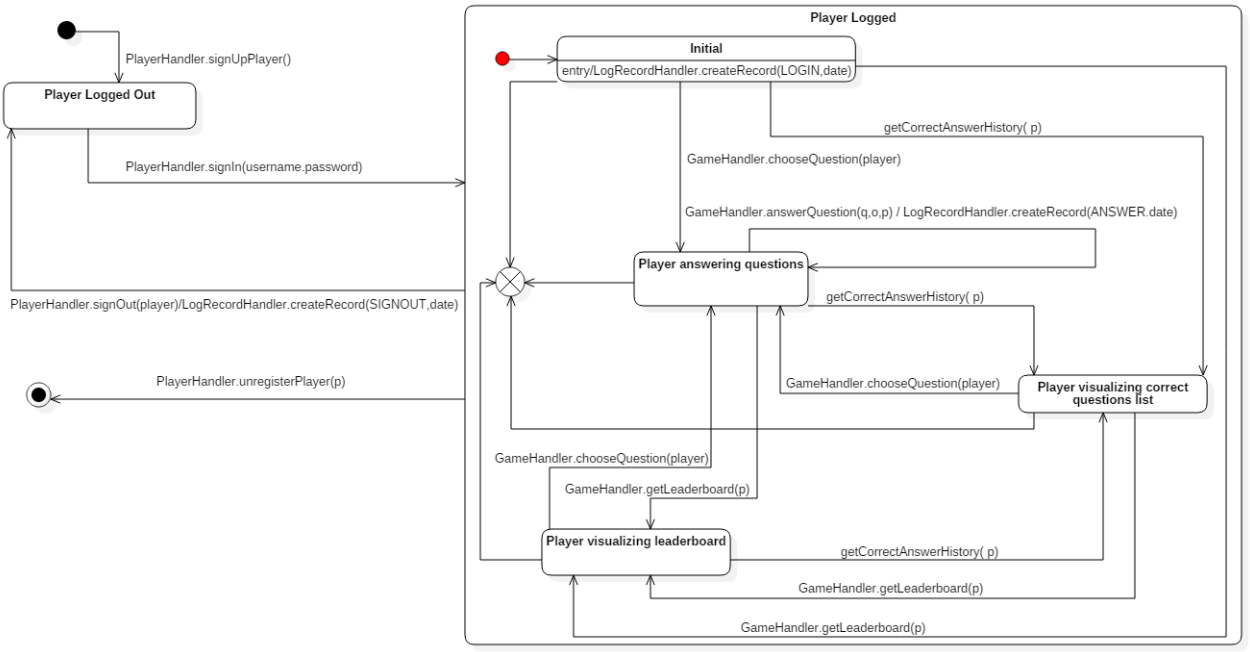
Il digramma mostra tutte le fasi per l'**aggiunta di una domanda** da parte dell'autore, e come quest'ultimo interagisce con il sistema al fine di compiere tutte le sue attività. L'autore richiederà la possibilità di inserire la domanda con relative opzioni (di cui solo una vera). Dopo aver controllato che tutto sia stato inserito correttamente, il sistema procederà a salvare la domanda altrimenti genererà una notifica di errore che consentirà all'autore di rimediare.

Statechart Diagram – Level



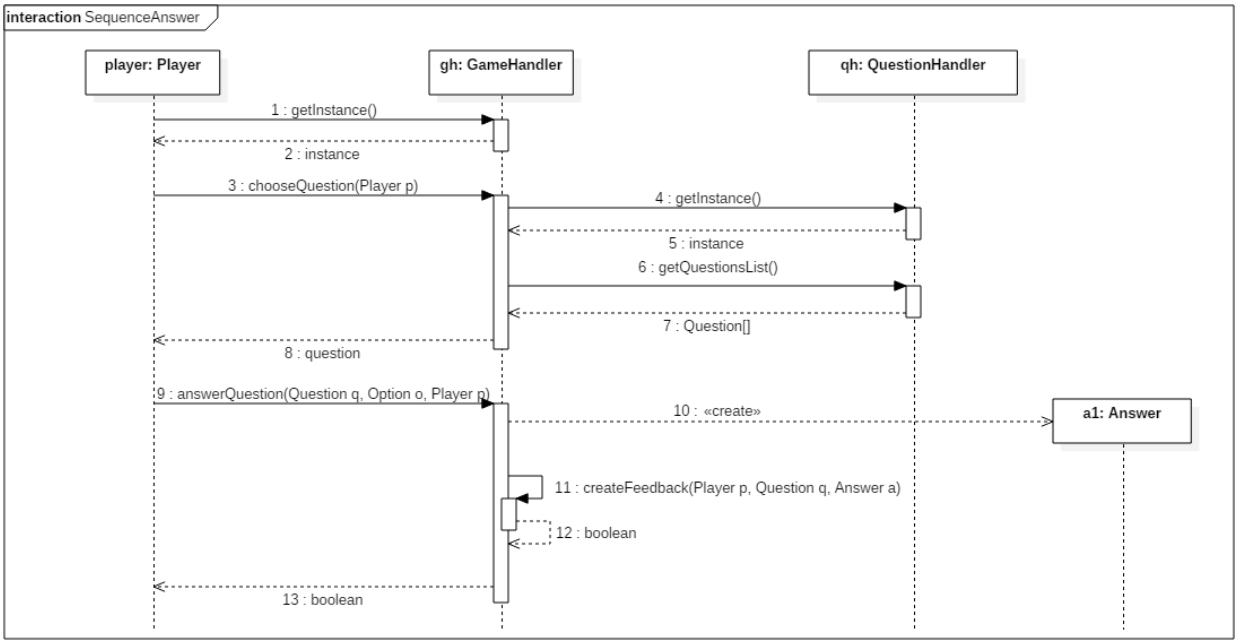
Il seguente Statechart Diagram evidenzia l'evoluzione del livello del giocatore e di conseguenza di quello delle domande proposte in funzione dell'esito delle risposte. I **livelli** sono tre e si comincia dal livello 1. Da ogni livello si può progredire solo se vengono fornite **tre risposte corrette** consecutivamente, altrimenti, nel caso in cui vengano date **cinque risposte sbagliate** in sequenza, il livello verrà decrementato (tranne nel caso in cui si è già al primo livello che non è preceduto da altri). Fintanto che non si verificherà una sequenza di risposte giuste o sbagliate che permetterà di cambiare livello, quest'ultimo rimarrà invariato.

Statechart Diagram – Player Status



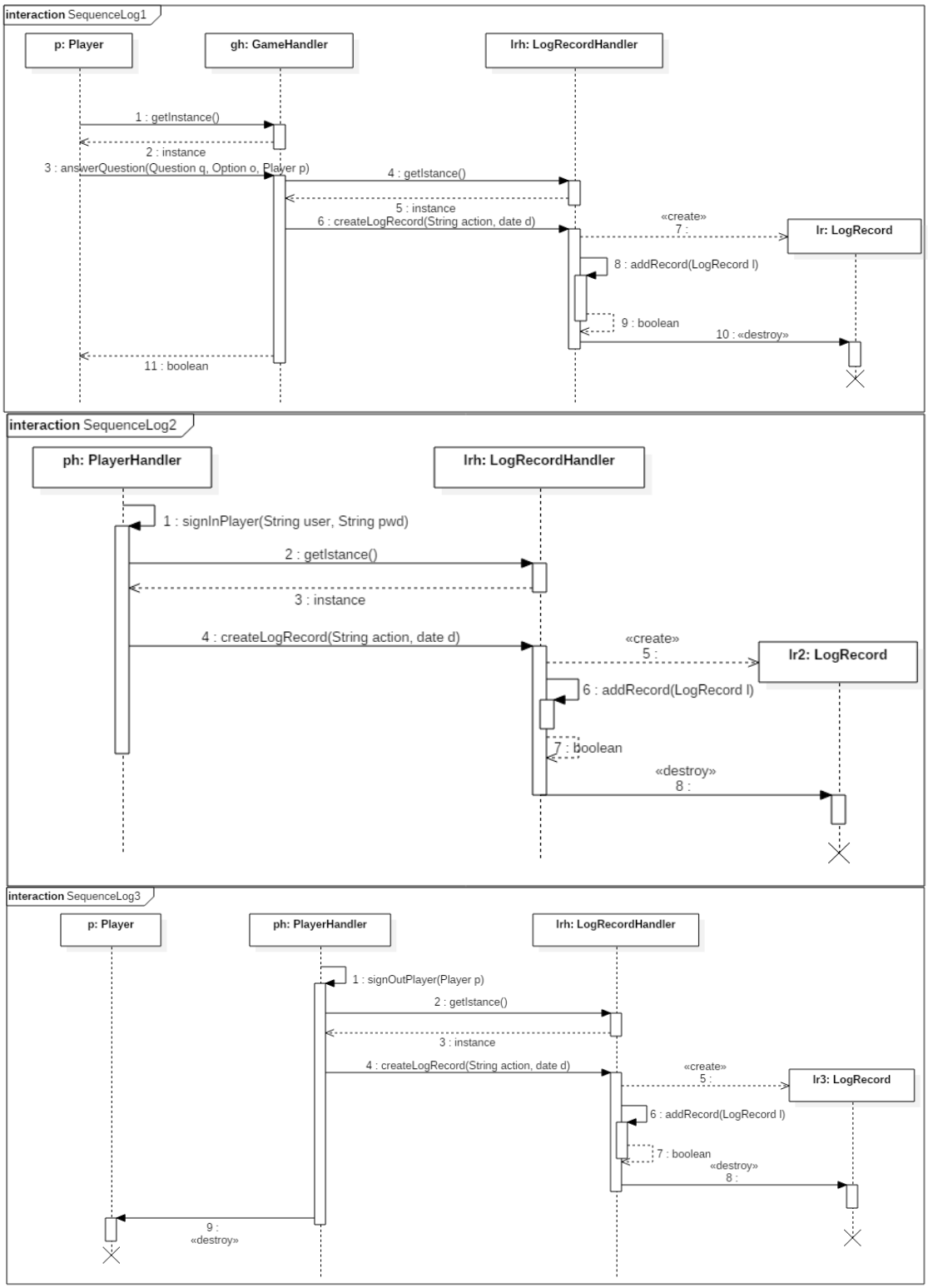
Questo diagramma analizza lo stato del giocatore, dalla sua registrazione al gioco, fino alla cancellazione. Il Player può trovarsi in due macro-stati, ovvero **“Player Logged”** e **“Player Logged Out”**; può passare dall’uno all’altro tramite **Sign-in** ed **Sign-out**. Quando il giocatore avrà effettuato l’accesso potrà transitare in sotto-stati a seconda delle azioni che svolge durante la sua sessione di gioco (rispondere alle domande, visualizzare le proprie statistiche o la classifica).

Sequence Diagram – Answer Question



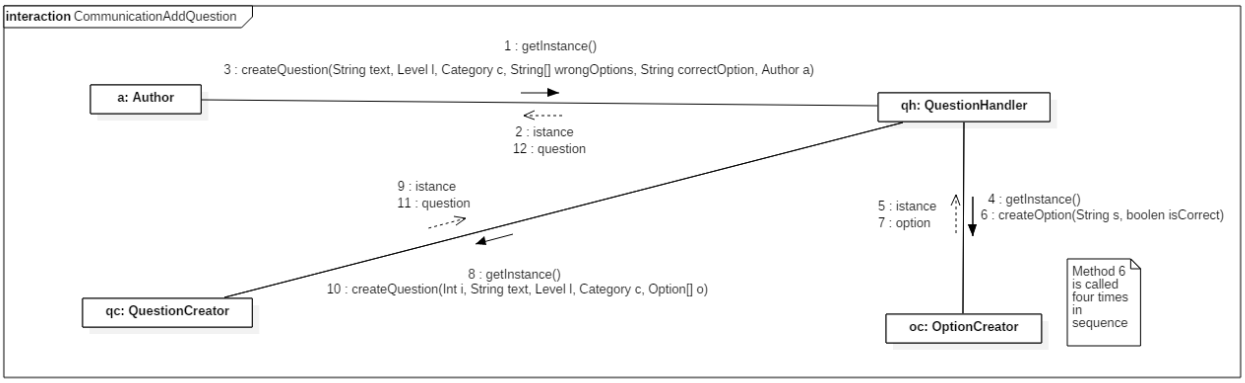
Lo scenario scelto per il primo Sequence Diagram è la selezione della risposta ad una domanda. Per far ciò il **Player** richiede un istanza del **GameHandler**, solo dopo che gli sarà restituita potrà ricevere una domanda. Di conseguenza sarà necessario richiamare un'istanza del **QuestionHandler** attraverso il quale sarà prelevata la domanda scelta dalla **QuestionList** e girata al Player. A questo punto sarà il Player a fornire la risposta al GameHandler il quale istanzierà un oggetto di **Answer** e richiamerà il metodo che gli permette di generare il **Feedback** da tornare all'utente.

Sequence Diagram – LogFile Actions



II secondo Sequence Diagram riguarda la gestione del file log. Quando il giocatore vuole rispondere ad una domanda che gli è stata posta chiede un'istanza del **GameHandler**. Quando quest'ultimo riceve la risposta a sua volta richiede un'istanza del **LogRecordHandler** per poter generare un nuovo **LogRecord**. Allo stesso modo quando il **PlayerHandler** fa accedere il giocatore chiede un'istanza al LogRecordHandler che genera il relativo LogRecord. La stessa identica procedura si verifica anche in caso di **Sign-out**.

Comunication Diagram – Add Question

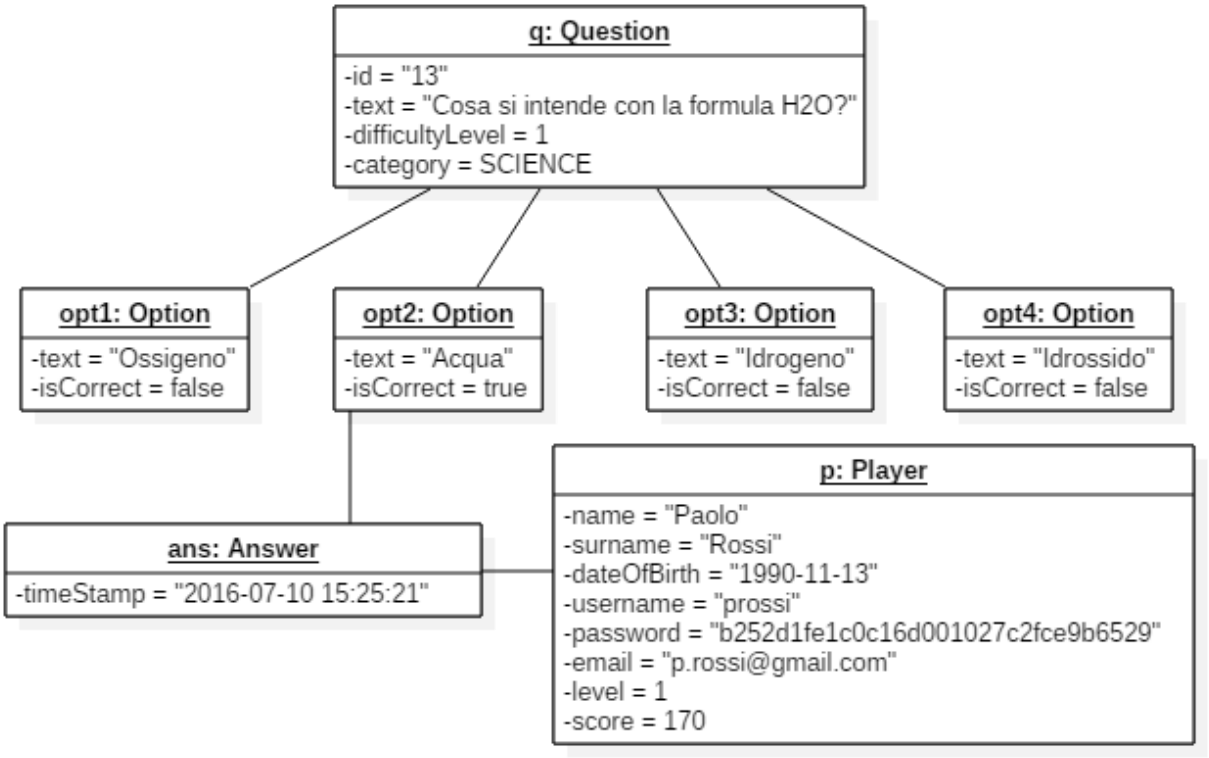


Il collaboration diagram riportato rappresenta la cooperazione che avviene tra le istanze del sistema al fine di aggiungere una domanda da parte dell'Author:

L'**Author**, dopo la richiesta dell'istanza del QuestionHandler, gli delega il compito di aggiungere una nuova domanda. A sua volta, il **QuestionHandler**, che deve gestire la creazione delle opzioni relative alla domanda, richiesta l'istanza dell'**OptionCreator**, gli delega il compito di aggiungere le opzioni: questo procedimento di aggiunta delle opzioni viene effettuato quattro volte così da creare il giusto numero di istanze.

Infine, il QuestionHandler richiede l'istanza del QuestionCreator che si occupa di ultimare la creazione della domanda finale, comprensiva di opzioni. A ritroso, tramite i valori di ritorno, l'istanza della **domanda creata** torna all'Author.

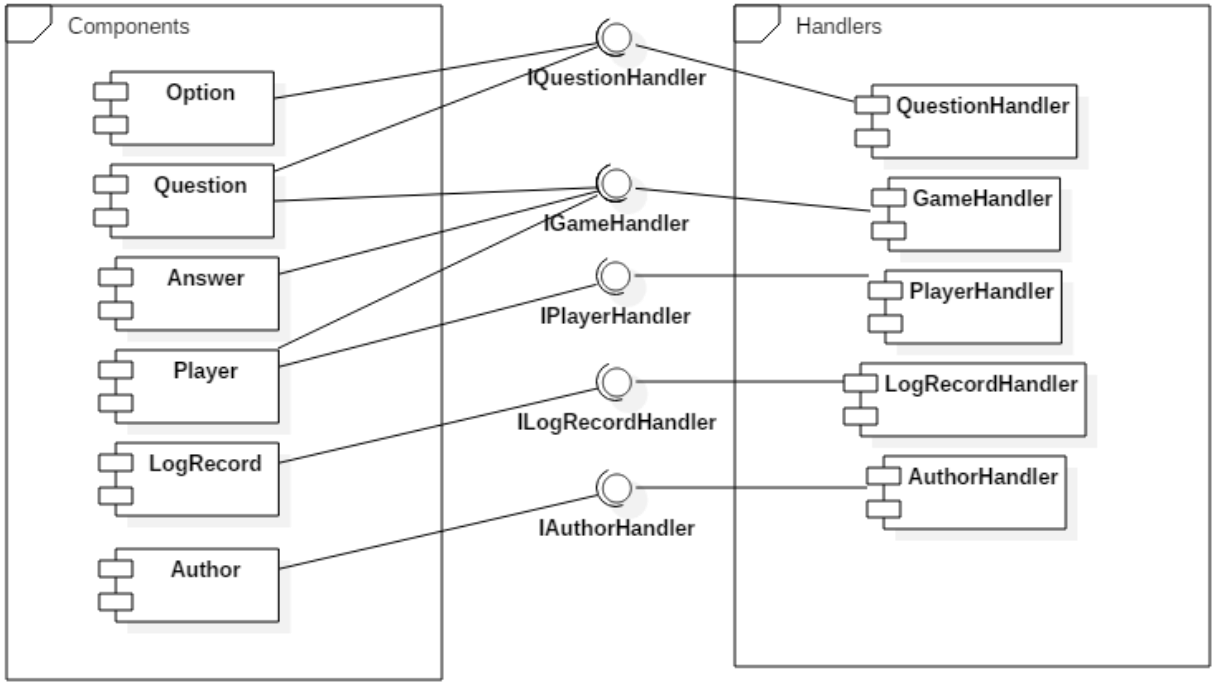
Object Diagram – Question & Answer



L'Object Diagram riportato offre una rappresentazione statica del sistema nel momento in cui un **Player** risponde ad una domanda. La struttura che collega domande e opzione sia ben definita: Ad ogni domanda corrispondono sempre **4 opzioni differenti** di cui solo una è corretta (isCorrect==true).

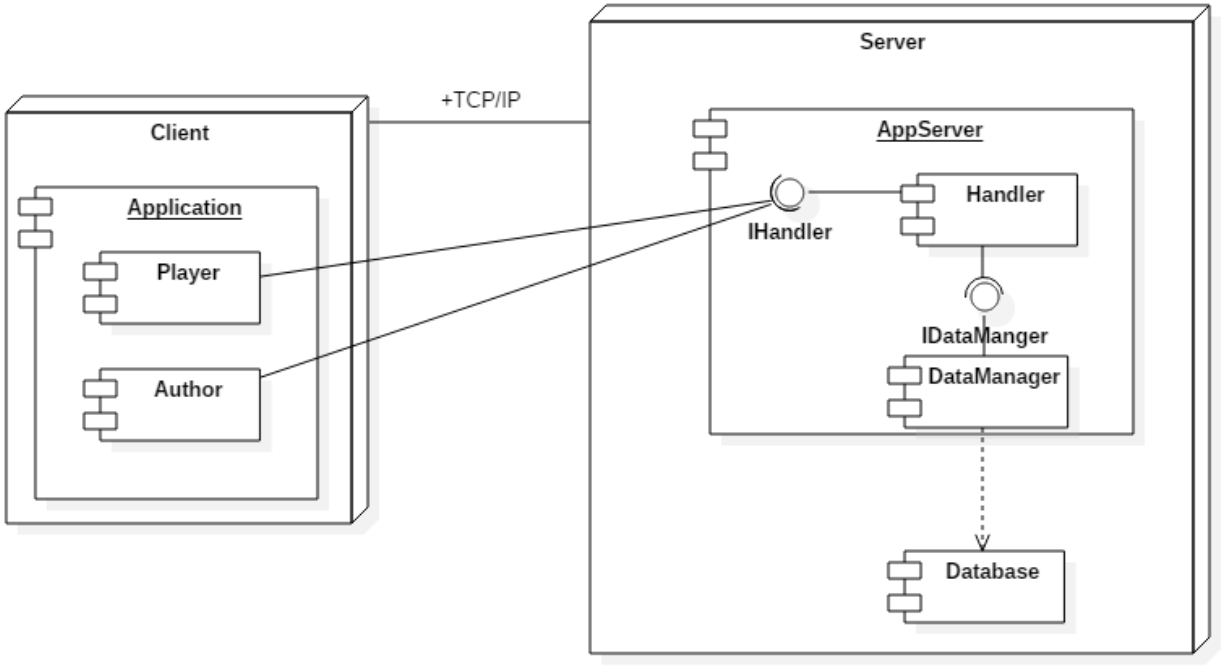
La risposta data è associata invece al Player e ad una sola risposta, che sia essa corretta o sbagliata.

Component Diagram



Il Component Diagram mostra la **struttura interna** del **sistema software** attraverso i suoi principali componenti mettendo in risalto come questi si interfaccino tra loro al fine di realizzare le funzionalità del sistema.

Deployment Diagram



Il Deployment Diagram mostra l'architettura del software. Evidenzia l'applicazione sul **Client** che permetterà ai **giocatori** di rispondere alle domande e di scalare la classifica e agli **autori** di creare e editare domande. Sul **Server** sono invece presenti tutti i gestori del sistema di gioco che comunicano con il database per il prelievo e la gestione dei dati.