Quiz Game

# Indice

Sommario

[1 Indice 1](#_Toc1125031468)

[2 Introduzione 2](#_Toc977842122)

[2.1 Informazioni sul progetto 2](#_Toc1655492137)

[2.2 Abstract 2](#_Toc858466892)

[2.3 Scopo 2](#_Toc1034900748)

[3 Analisi 2](#_Toc390150817)

[3.1 Analisi del dominio 3](#_Toc1120349189)

[3.2 Analisi e specifica dei requisiti 3](#_Toc455700814)

[3.3 Use case 8](#_Toc409237354)

[3.4 Pianificazione 9](#_Toc1515077906)

[3.5 Analisi dei mezzi 11](#_Toc40226625)

[3.5.1 Software 12](#_Toc542584867)

[3.5.2 Hardware 12](#_Toc1216800169)

[4 Progettazione 12](#_Toc717977942)

[4.1 Design dell’architettura del sistema 12](#_Toc118373879)

[4.2 Design delle interfacce 12](#_Toc527905496)

[4.3 Design procedurale 16](#_Toc1756709009)

[4.3.1 Diagramma di flusso 17](#_Toc1029029518)

[4.3.2 Diagramma delle classi 17](#_Toc307730649)

[5 Implementazione 17](#_Toc1212835575)

[5.1 Classe Main 18](#_Toc702234177)

[5.2 Classe UsernamePanel 18](#_Toc1208547349)

[5.2.1 Funzione saveUsername 18](#_Toc832287322)

[5.3 Classe SettingsPanel 18](#_Toc118270254)

[5.3.1 Funzione getCategory 18](#_Toc80297990)

[5.3.2 Funzione saveSettings 19](#_Toc1361421776)

[5.4 Classe QuizPanel 19](#_Toc172443261)

[5.4.1 Funzione loadQuestions 19](#_Toc530045656)

[5.4.2 Funzione convertDifficulty 20](#_Toc1997401074)

[5.4.3 Funzione displayQuestions 20](#_Toc1157985152)

[5.4.4 Funzione pointAnswer 22](#_Toc1886878089)

[5.5 Classe ResultPanel 22](#_Toc933881940)

[5.5.1 Funzione getPlayerName 22](#_Toc117199894)

[5.6 Classe LeaderboardManager 22](#_Toc1830102365)

[5.6.1 Funzione updateLeaderboard 22](#_Toc924865249)

[5.6.2 Funzione saveLeaderboard 22](#_Toc1071937800)

[5.7 Classe DetailsPanel 22](#_Toc612793652)

[5.7.1 Funzione formatAnswers 22](#_Toc2039968917)

[6 Test 22](#_Toc2040185140)

[6.1 Protocollo di test 22](#_Toc1992473422)

[6.2 Risultati test 22](#_Toc1551519171)

[6.3 Mancanze/limitazioni conosciute 22](#_Toc660437913)

[7 Consuntivo 22](#_Toc131497566)

[8 Conclusioni 22](#_Toc1559022503)

[8.1 Sviluppi futuri 22](#_Toc1293727555)

[8.2 Considerazioni personali 22](#_Toc159999896)

[9 Bibliografia 22](#_Toc2077263270)

[9.1 Bibliografia per articoli di riviste: 22](#_Toc1763427002)

[9.2 Bibliografia per libri 22](#_Toc276030698)

[9.3 Sitografia 22](#_Toc1087454221)

[10 Glossario 22](#_Toc1543599093)

[11 Indice delle figure 22](#_Toc659044782)

[12 Allegati 22](#_Toc516753696)

# Introduzione

## Informazioni sul progetto

Allievo: Leonardo Sciara

Docente Responsabile: Geo Petrini

SAMT Sezione Informatica 3BB Modulo 306

Inizio: 04.09.2024

Termine: 18.12.2024

## Abstract

È una breve e accurata rappresentazione dei contenuti di un documento, senza notazioni critiche o valutazioni. Lo scopo di un abstract efficace dovrebbe essere quello di far conoscere all’utente il contenuto di base di un documento e metterlo nella condizione di decidere se risponde ai suoi interessi e se è opportuno il ricorso al documento originale.

Può contenere alcuni o tutti gli elementi seguenti:

* **Background/Situazione iniziale**
* **Descrizione del problema e motivazione**: Che problema ho cercato di risolvere? Questa sezione dovrebbe includere l'importanza del vostro lavoro, la difficoltà dell'area e l'effetto che potrebbe avere se portato a termine con successo.
* **Approccio/Metodi**: Come ho ottenuto dei progressi? Come ho risolto il problema (tecniche…)? Quale è stata l’entità del mio lavoro? Che fattori importanti controllo, ignoro o misuro?
* **Risultati**: Quale è la risposta? Quali sono i risultati? Quanto è più veloce, più sicuro, più economico o in qualche altro aspetto migliore di altri prodotti/soluzioni?

## Scopo

Il progetto Quiz Game ha come obiettivo lo sviluppo di un sistema di quiz utilizzando Java, con la possibilità di personalizzare il quiz, rispondere alle domande e osservare la classifica dei migliori giocatori.Lo scopo didattico di questo progetto è quello di creare il miglior programma possibile tramite le conoscenze apprese in questi tre anni di scuola professionale nei vari moduli di programmazione.

# Analisi

## Analisi del dominio

Il programma di questo progetto sarà simile a quelli che trovi online, ti permette di

* Salvarti col tuo username
* Scegliere categorie di domande, livelli di difficoltà e quantità di domande.
* Rispondere a delle domande con una o più opzioni di risposta.
* Visualizzare i punteggi al termine del quiz.

Il sistema sarà gestito attraverso JAVA

## Analisi e specifica dei requisiti

Inserimento nome giocatore:

* Al caricamento nella pagina Username, l'utente deve inserire il proprio nome tramite un input text che viene salvato in un file txt
* Se l'utente non inserisce niente, il tasto “Impostazioni” non compare

Inserimento impostazioni:

* File:
* Dopo aver premuto “Impostazioni”, l’utente va nel file Impostazioni
* Il file contiene 4 tasti; "Categoria", “Difficoltà”, “Quantità Domande” e “Inizio Gioco”
* Se l'utente non ha selezionato una categoria, una difficoltà o la quantità di domande, “Inizia gioco” non appare
* Categoria:
* L’utente sceglie una tra 6 categorie “Sport”, “Storia”, “Cinema”, “Scienza”, “Musica” e “Matematica” tramite dei RadioButton
* Difficoltà:
* L'utente sceglie una tra 3 difficoltà "Facile", "Medio" e "Difficile" tramite uno slider
* Quantità domande:
* L’utente sceglie un numero tra “1” e “30” tramite uno slider

Inizio gioco:

* Quando l'utente preme su "Inizia il gioco", seleziona le quantità di domande e risposte dal file di tipo txt e poi inizia il quiz
* Le domande cambiano in base alla categoria e alla difficoltà scelta
* Ogni domanda ha più possibili risposte, delle quali una o più sono corrette.
* Domande e risposte possono contenere anche delle immagini

Inserimento risposte:

* Le riposte saranno sottoforma di checkbox, se con più risposte, e sottoforma di radiobutton, se con una risposta singola
* Per rispondere alle domande c’è un determinato tempo a disposizione, dopo questo tempo viene calcolato come risposta sbagliata
* A dipendenza della difficoltà, il punteggio di ogni risposta corretta è differente
* Se in una domanda ci sono più risposte corrette e l’utente non inserisce il numero giusto di risposte, o le inserisce, ma sbagliate, il punteggio viene calcolato in base al numero di risposte corrette date e al numero di risposte corrette possibili
* Se l’utente non ha selezionato al meno una risposta, il tasto “Visualizza Risposta”, che serve per verificare la correttezza della risposta, non appare
* Dopo aver premuto “Visualizza Risposta”, il testo del tasto cambierà in “Procedi”, che premendolo, passerà alla domanda successiva

Visualizzazione risultati:

* Al termine del quiz, l’utente può visualizzare la classifica dei 10 migliori giocatori e il suo punteggio totale totalizzato
* L’utente può scegliere il tasto “Visualizza Dettagli", che porta nella pagina Details dove ti dice i punteggi fatti per ogni domanda e le risposte corrette a quella domanda
* L’utente può scegliere il tasto "Nuova Partita", che riporta al file per selezionare categorie, difficoltà e quantità di domande diverse
* L’utente può scegliere il tasto "Nuovo Giocatore", che riporta allo Username file per inserire un nuovo giocatore
* L’utente può scegliere il tasto "Chiudi Gioco", che chiude il gioco

Visualizzazione dettagli:

* L’utente potrà visualizzare i dettagli di ogni domanda, potendo vedere i punti totalizzati in quella domanda
* l’utente può premere il tasto “Indietro”, che porta l’utente alla pagina dei risultati

|  |  |
| --- | --- |
| **ID: REQ-01** | |
| **Nome** | Inserimento nome giocatore |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | L’utente deve inserire il proprio nome |
| **Sotto requisiti** | |
| **001** | L’utente deve inserire il nome desiderato nell’input text |
| **002** | Dopo aver inserito il nome, si dovrà premere “Impostazioni” per passare alla pagina delle impostazioni |

|  |  |
| --- | --- |
| **ID: REQ-02** | |
| **Nome** | Inserimento Impostazioni |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | L’utente deve configurare la categoria, la difficoltà e la quantità delle domande |
| **Sotto requisiti** | |
| **001** | L’utente sceglie una categoria tra “Sport”, “Storia”, “Cinema”, “Scienza”, “Musica” e “Matematica” tramite un RadioButton |
| **002** | L’utente seleziona una difficoltà tra “Facile”, “Medio” e “Difficile” tramite uno slider |
| **003** | L’utente seleziona una quantità di domande tra “1” e “30” tramite uno slider |
| **004** | Dopo aver selezionato i campi, si dovrà premere “Inizia Gioco” per iniziare il gioco |

|  |  |
| --- | --- |
| **ID: REQ-03** | |
| **Nome** | Inizio del gioco |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** | L’utente deve rispondere alle domande che li vengono proposte |
| **Sotto requisiti** | |
| **001** | Vengono caricate domande in modo casuale in base alla categoria e difficoltà scelte |
| **002** | Ogni domanda ha più risposte possibili, una o più corrette. Di queste risposte, quelle sbagliate vengono inserite casualmente |
| **003** | Ogni domanda e risposta possono includere immagini |

|  |  |
| --- | --- |
| **ID: REQ-04** | |
| **Nome** | Inserimento risposte |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** | L’utente deve selezionare le risposte tramite checkbox o radiobutton |
| **Sotto requisiti** | |
| **001** | Le risposte sono in formato checkbox per più risposte e radiobutton per una risposta |
| **002** | L’utente ha 30 secondi per rispondere a ogni domanda e superando questo tempo, la risposta viene considerata sbagliata |
| **003** | Ogni risposta corretta assegna un punteggio al giocatore, in base alla difficoltà scelta però, le risposte corrette hanno un valore diverso tra loro: “Difficile” 20 punti, “Medio” 10 punti e “Facile” 5 punti |
| **004** | Se in una domanda ci sono più risposte corrette e l’utente non inserisce il numero giusto di risposte, o le inserisce, ma sbagliate, il punteggio viene calcolato in base al numero di risposte corrette date e al numero di risposte corrette possibili |
| **004** | Dopo aver inserito al meno una risposta, l’utente deve premere “Visualizza Risposta” per verificare la correttezza della risposta |
| **005** | Dopo aver visualizzato la risposta, l’utente deve premere “Procedi” per passare alla domanda successiva |

|  |  |
| --- | --- |
| **ID: REQ-05** | |
| **Nome** | Visualizzazione risultati |
| **Priorità** | 3 |
| **Versione** | 1.0 |
| **Note** | Al termine del quiz, l’utente può vedere la classifica e il proprio punteggio |
| **Sotto requisiti** | |
| **001** | L’utente può visualizzare la classifica dei 10 migliori giocatori e del proprio punteggio |
| **002** | L’utente può premere “Visualizza dettagli”, che offre la possibilità di visualizzare i dettagli delle varie risposte alle domande |
| **003** | L’utente può premere “Nuova Partita”, che permette all’utente di poter continuare a giocare con lo stesso giocatore, ma con la possibilità di cambiare le impostazioni |
| **004** | L’utente può premere “Nuovo Giocatore”, che permette all’utente di creare un nuovo giocatore con cui giocare |
| **004** | L’utente può premere “Chiudi Gioco”, che chiude il gioco |
| **ID: REQ-06** | |
| **Nome** | Visualizzazione dettagli |
| **Priorità** | 4 |
| **Versione** | 1.0 |
| **Note** | L’utente può visualizzare i dettagli di ogni domanda |
| **Sotto requisiti** | |
| **001** | Vengono caricati i dettagli di ogni domanda, con i loro rispettivi punteggi |
| **002** | Dopo aver visualizzato i dettagli, l’utente può premere “Indietro” per tornare alla pagina dei risultati |

**Spiegazione elementi tabella dei requisiti:**

**Nome**: breve descrizione del requisito

**Priorità**: indica l’ordine in cui viene eseguito quel requisito secondo dei numeri

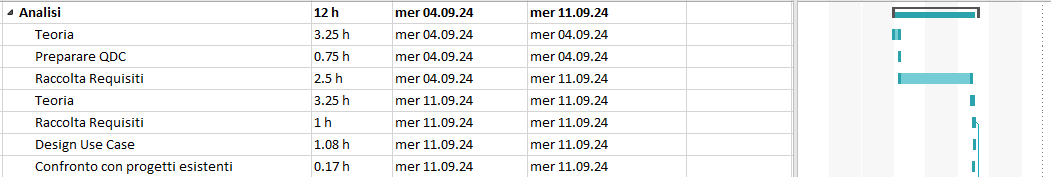
**Versione**: indica la versione del requisito. Ogni modifica apportata avrà una versione aggiornata.

**Note**: descrizione un po’ più dettagliata

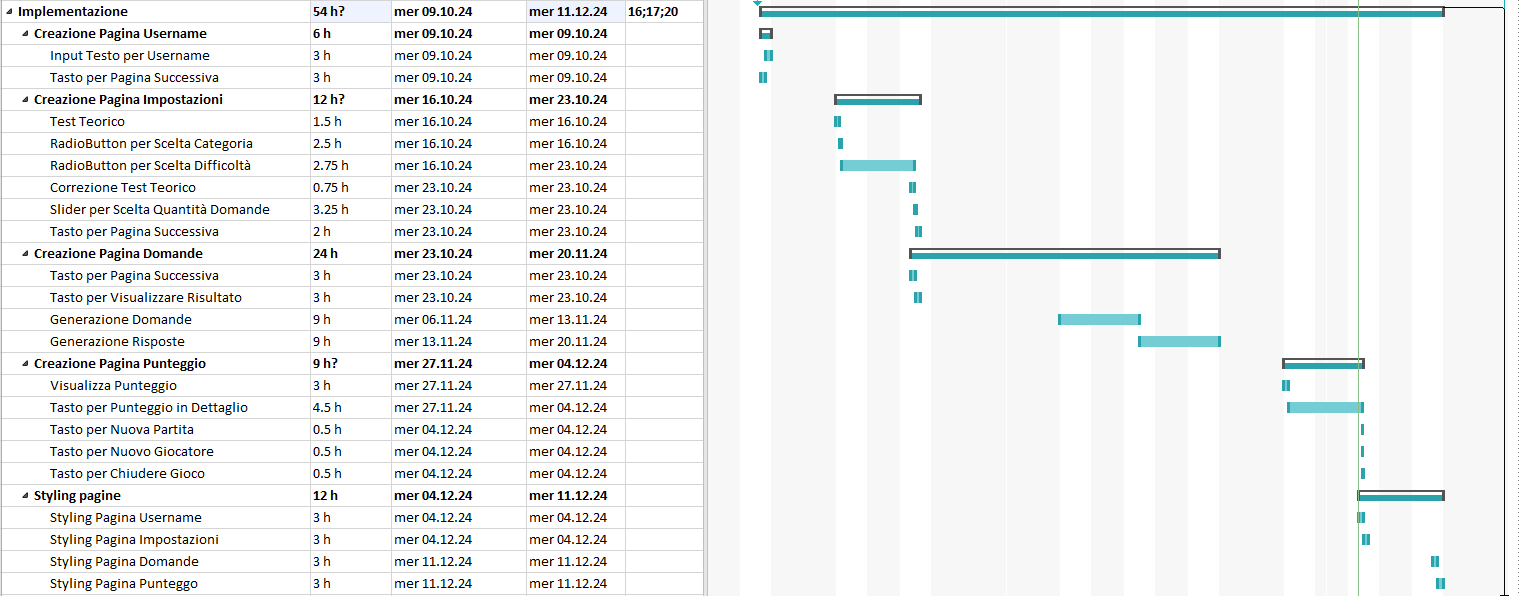
**Sotto requisiti**: elementi che compongono il requisito.

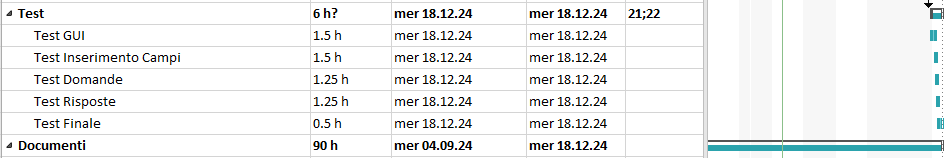
## Use case

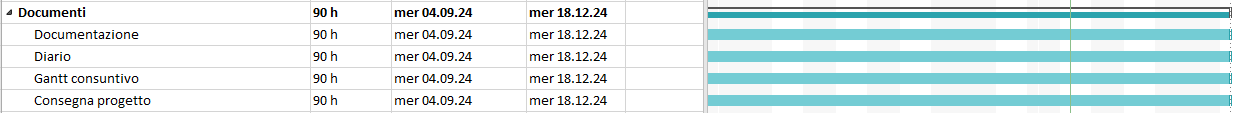
## Pianificazione

Nell’immagine che segue abbiamo l’analisi, la base per poi iniziare il progetto:

Qui sotto invece, troviamo la progettazione del progetto (il predecessore 6 significa che quelle attività dipendono dall’analisi dei requisiti):

A seguire abbiamo l’implementazione del progetto, cioè la sua creazione (I predecessori 16,17,20 significano che l’implementazione dipende dal Diagramma di flusso, dal Design UI e dal Diagramma delle classi):

Poi ci sono i Test, per verificare la correttezza del codice:

Infine ci sono i Documenti, che verranno compilati durante il progetto:

## Analisi dei mezzi

### Software

Il software include:

* JDK (Java Development Kit): per sviluppare il codice.
* NetBeans 20: come Ambiente di Sviluppo Integrato (IDE) per scrivere e gestire il codice.

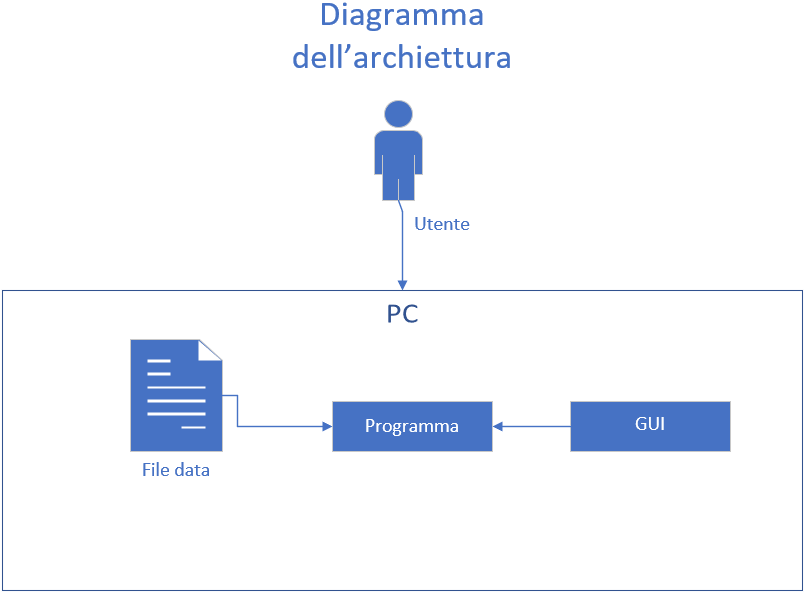
### Hardware

L’hardware include:

* PC di scuola per la realizzazione:
  + CPU : 13th Gen Intel(R) Core(TM) i7-13700
  + GPU : Intel(R) UHD Graphics 770
  + RAM : 32GB
  + SSD : 512GB
* PC di casa sempre per la realizzazione, per poter continuare in caso di ore saltate a lezione

# Progettazione

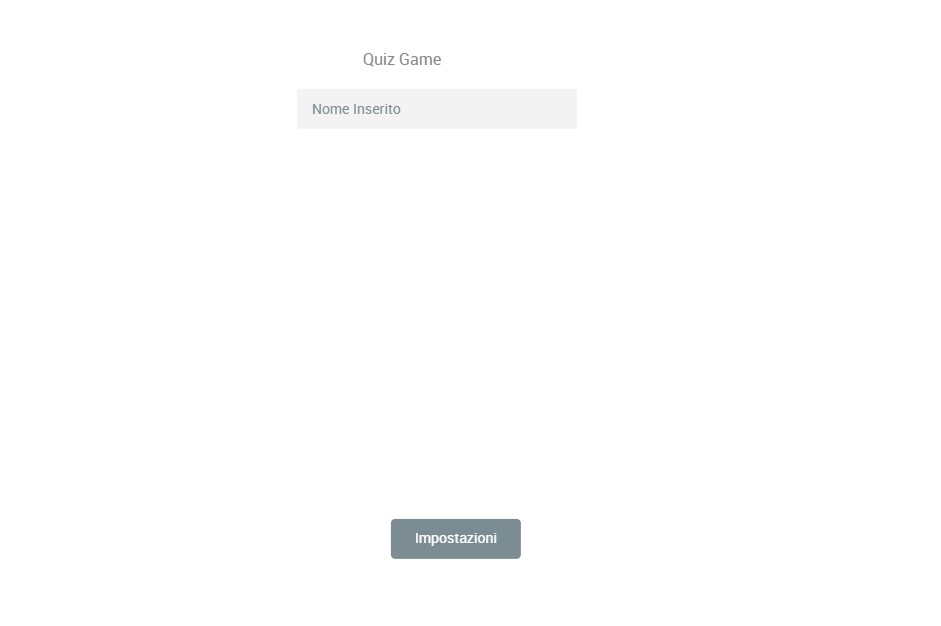
## Design dell’architettura del sistema



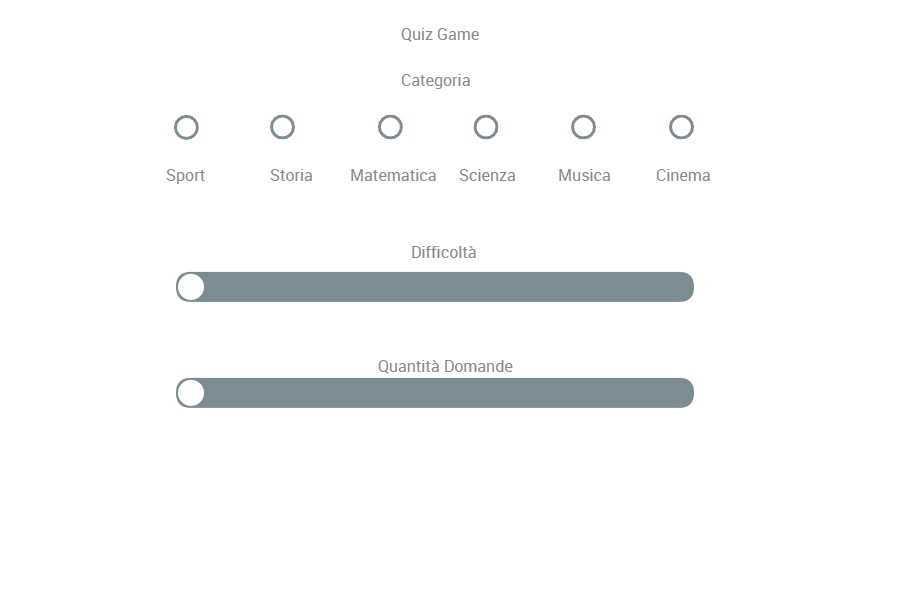
* La struttura del sistema lo schema di rete.
* I componenti che lo compongono.
* I flussi di informazione in ingresso ed in uscita e le relative elaborazioni.

## Design delle interfacce



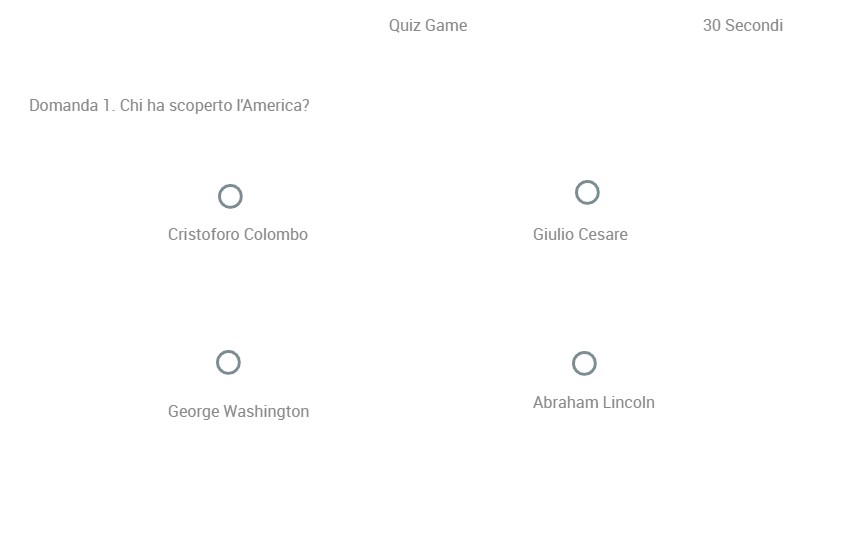


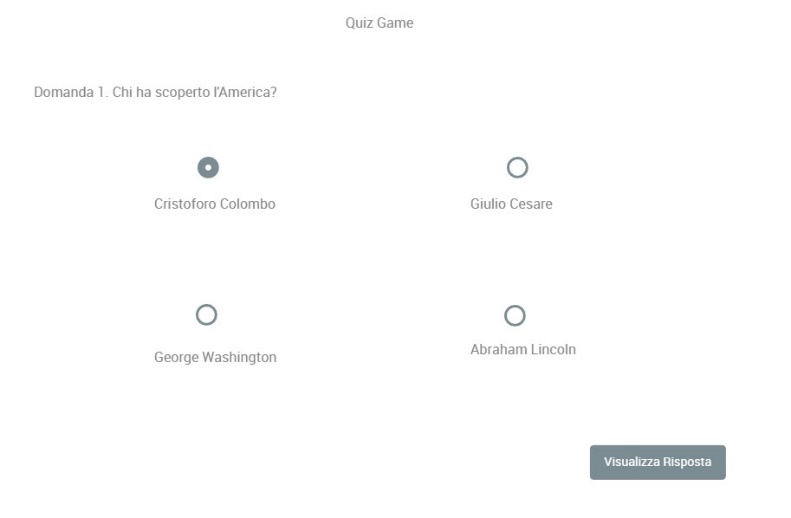
Queste due Interfacce rappresentano la pagina **“Username”**, dove l’utente, una volta inserito il nome, potrà schiacciare il tasto **“Impostazioni”.**

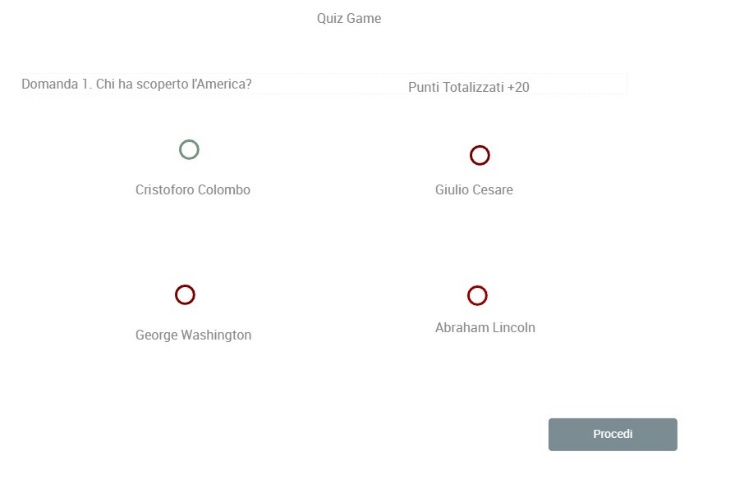




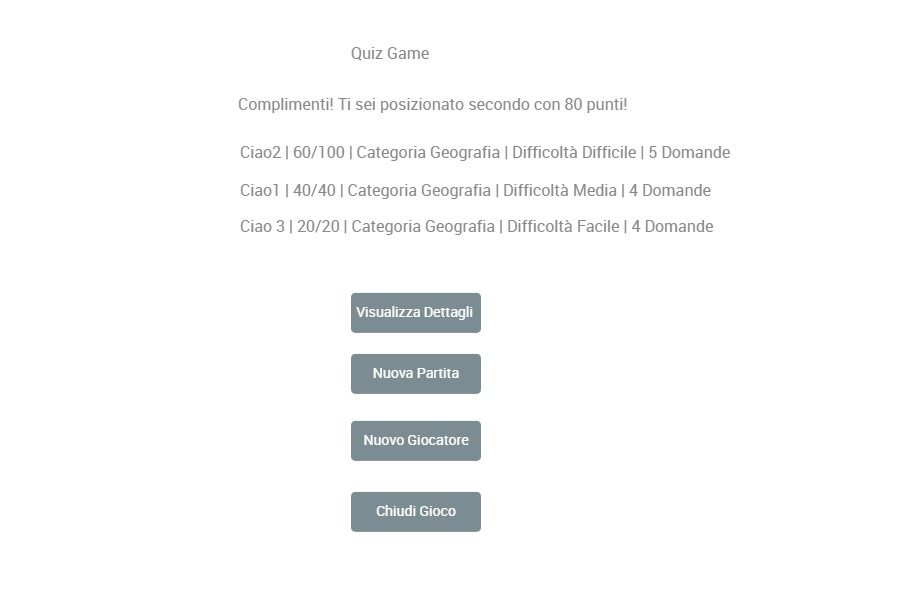
Le due interfacce successive rappresentano la pagina **“Impostazioni”**, l’utente dovrà inserire dei valori in tutti e 3 i campi per poter premere il tasto e passare finalmente al gioco.



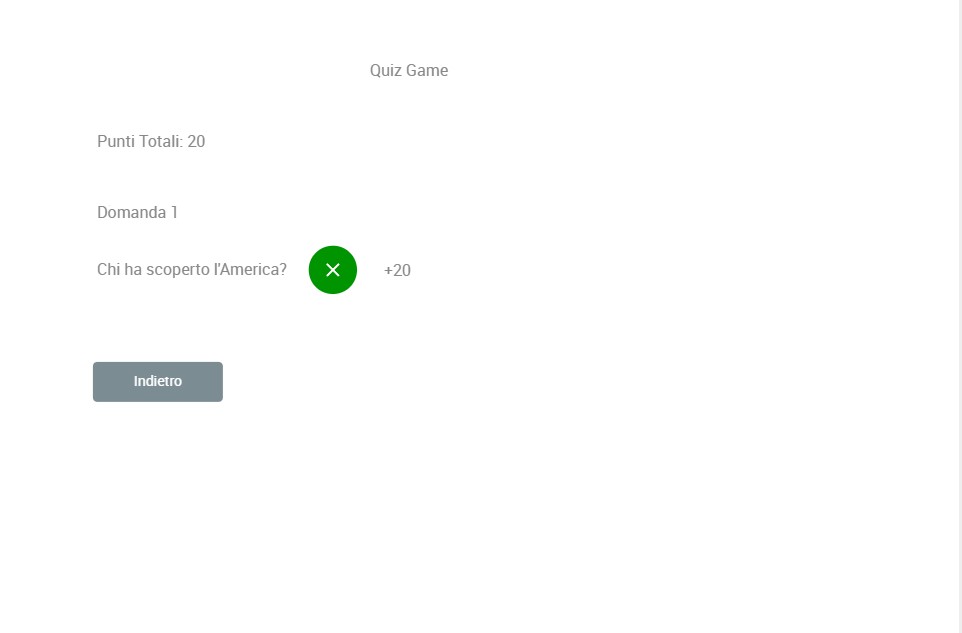




Queste tre interfacce rappresentano la pagina **“Domande”**, dove in base alla difficoltà e alla categoria scelta in precedenza ti escono domande diverse, e in base alla quantità di domande scelte, ti escono quel numero di domande. L’utente deve scegliere al meno una risposta per poter visualizzare la risposta corretta e poi procedere alla prossima domanda oppure a visualizzare il punteggio totalizzato.



Questa interfaccia rappresenta la pagina **“Risultati”**, l’utente può visualizzare i punteggi totalizzati in quella partita e in che posizione si è posizionato nella classifica, l’utente può comunque decidere di visualizzare i dettagli della partita, di iniziare una nuova partita con lo stesso utente, o di giocarne un’altra ma con un altro giocatore, o di chiudere l’applicazione.



Quest’ultima interfaccia rappresenta la pagina **“Dettagli”**, l’utente può visualizzare i dettagli di ogni singola domanda della partita appena giocata.

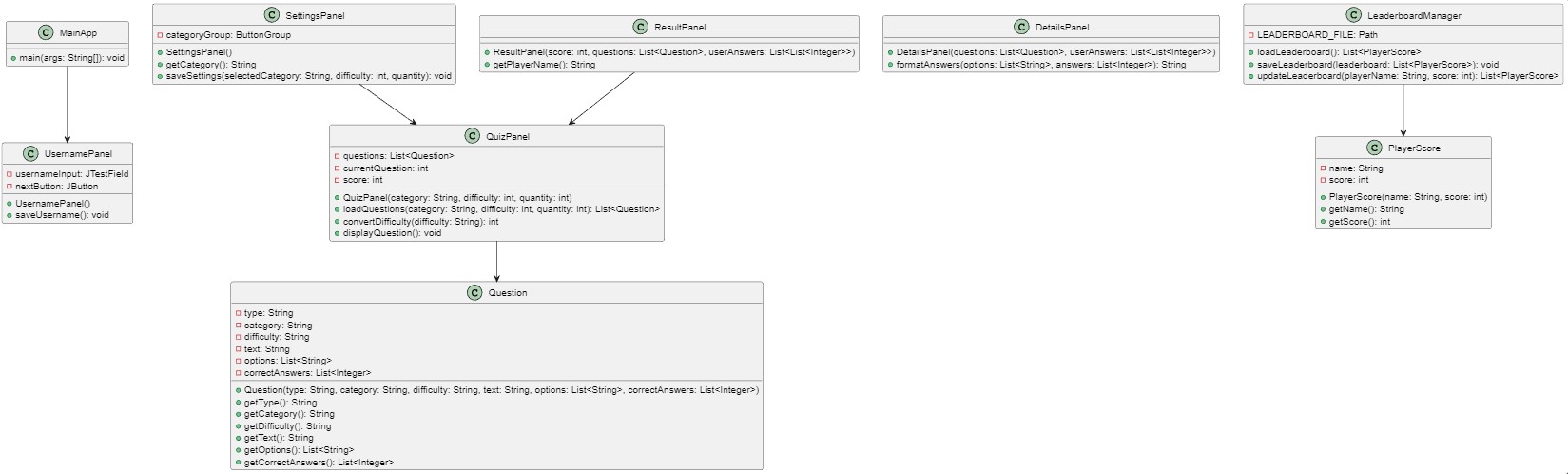
## Design procedurale

Descrive i concetti dettagliati dello sviluppo:

* Diagrammi di flusso
* Diagramma delle classi

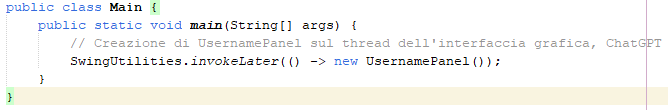
### Diagramma di flusso

### Diagramma delle classi



# Implementazione

## Classe Main

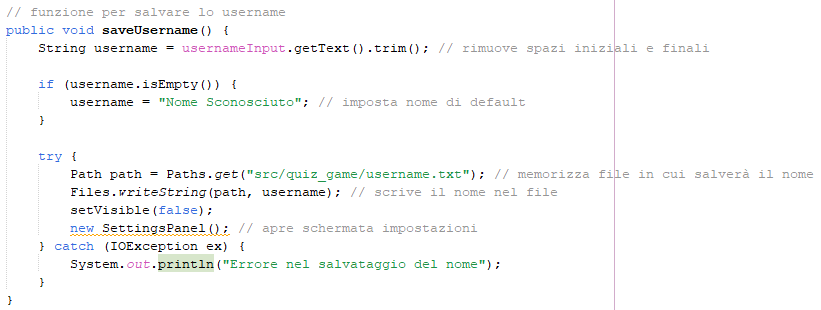
La classe Main è la classe principale del programma, dove parte tutto, infatti, serve per poter avviare l’applicazione. Tramite SwingUtilities.inokeLater richiama la schermata UsernamePanel sull’interfaccia grafica. **È stato richiesto l’aiuto di ChatGPT**

## Classe UsernamePanel

La classe UsernamePanel mostrerà la pagina iniziale dove l’utente sarà obbligato a inserire un nome per poter proseguire con il programma

### Funzione saveUsername

La funzione saveUsername() server per poter memorizzare il nome in un file. Per poter inserire un nome di default ho dovuto usare la funzione trim() che rimuove gli spazi iniziali e finali, in modo da poter controllare se il nome inserito fosse vuoto

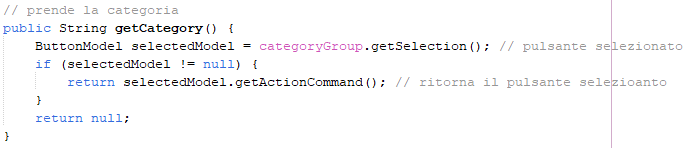


## Classe SettingsPanel

La Classe SettingsPanel illustrerà le impostazioni che dovrà configurare l’utente

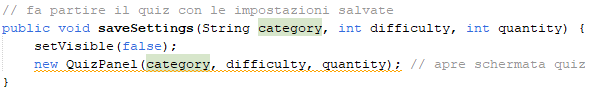
### Funzione getCategory

La funzione getCategory() serve per poter estrarre il valore della categoria selezionata dall’utente. Si fa un getSelection() su categoryGroup, per poter ricavare il valore selezionato da quel gruppo di bottoni e poi un return per ritornare il valore selezionato. (Il controllo per verificare che sia stata selezionata una categoria lo facciamo nel costruttore)



### Funzione saveSettings

La funzione SaveSettings() serve per salvare le impostazioni nella schermata del quiz.

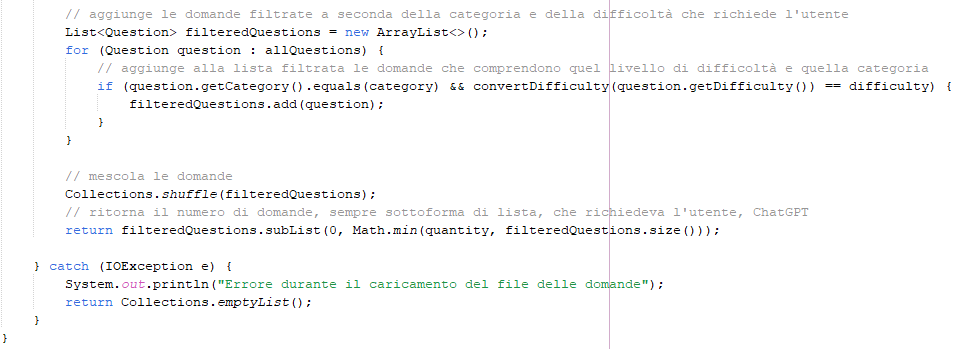
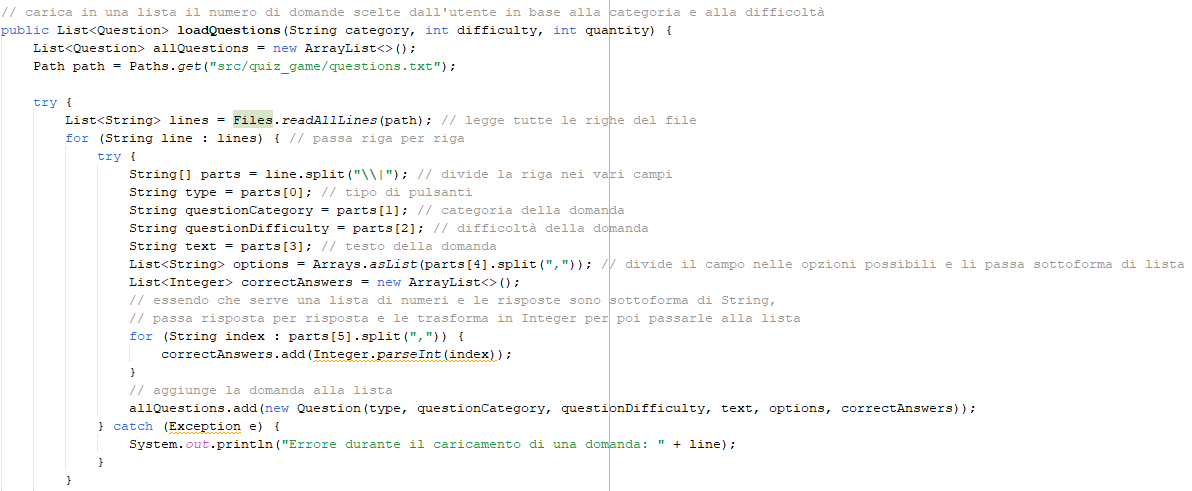


## Classe QuizPanel

La classe QuizPanel mostrerà tutte le domande richieste dall’utente, che verranno filtrate tramite difficoltà e categoria sempre scelta dall’utente.

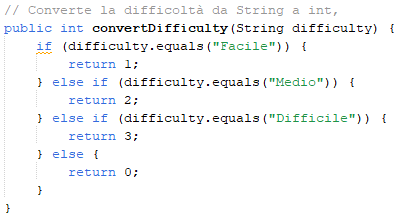
### Funzione loadQuestions

La funzione loadQuestions() serve per poter caricare in una lista il numero di domande, la categoria e la difficoltà che dovranno essere mostrate nel quiz. Memorizza tutte le righe del file delle domande in unica lista, questa lista viene passata e poi letta parte per parte, queste parti vengono memorizzate in variabili o liste, dopo averle memorizzate, inserisce i valori in una lista sottoforma di unica domanda, fino ad arrivare alla fine delle righe. Questa lista, contenente tutte le domande, viene filtrata in base alla categoria e alla difficoltà scelta dall’utente, per poi essere mescolata ed essere troncata con il numero di domande richieste dall’utente.



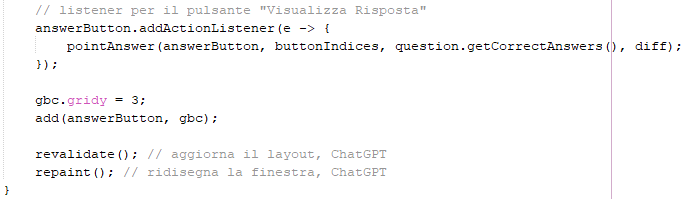
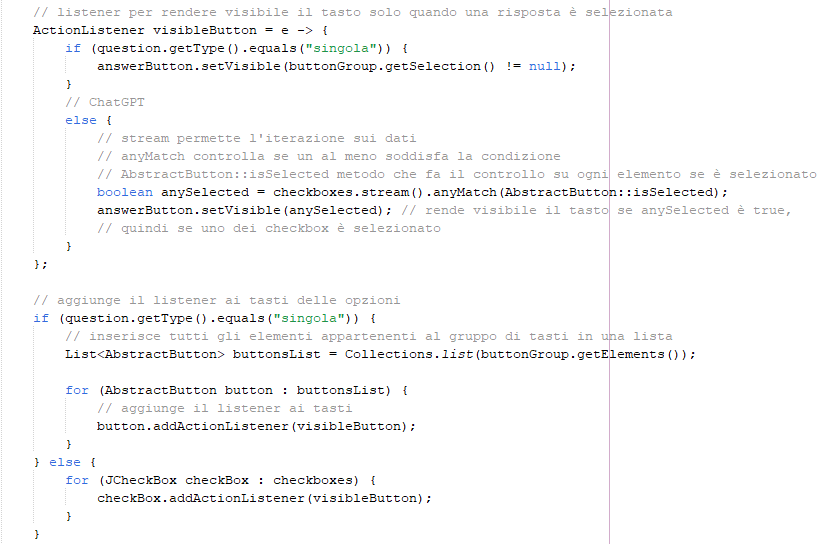
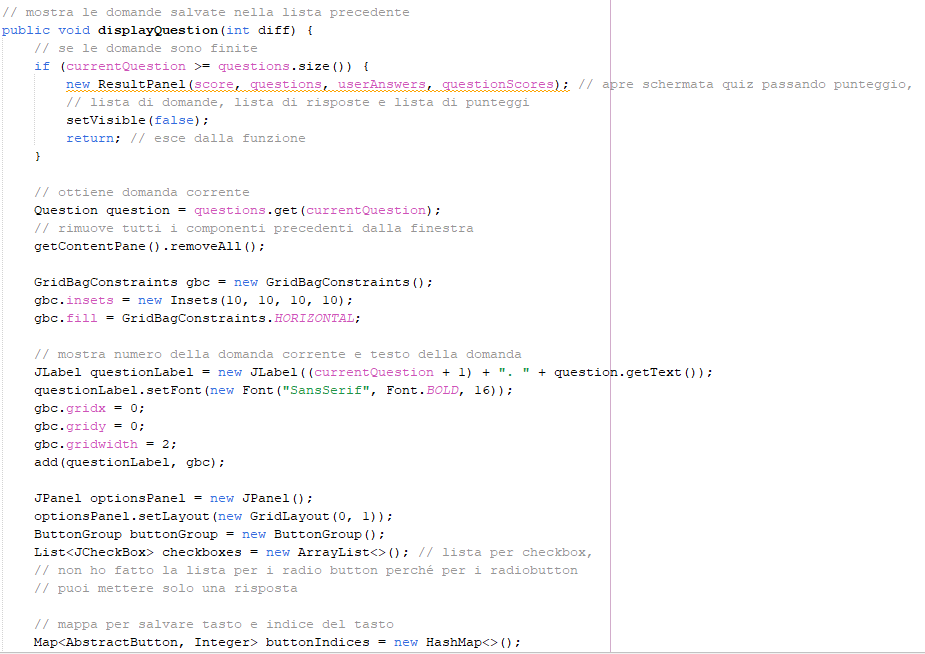
### Funzione convertDifficulty

La funzione convertDifficulty() serve per poter convertire la difficoltà da String e ritornarla in int.



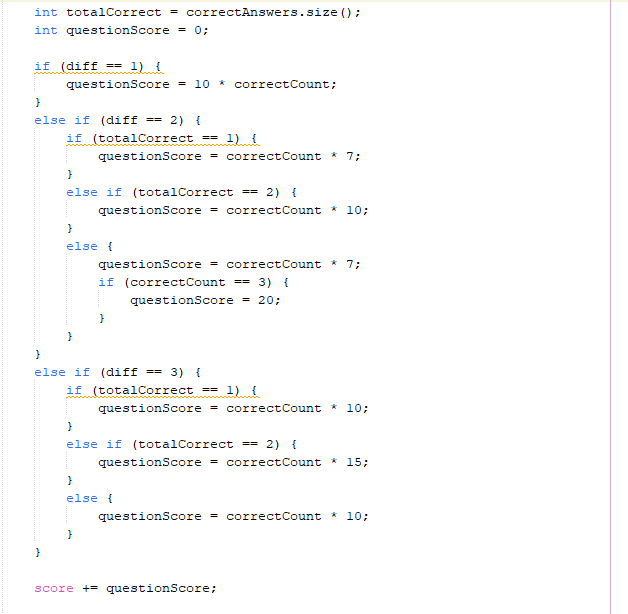
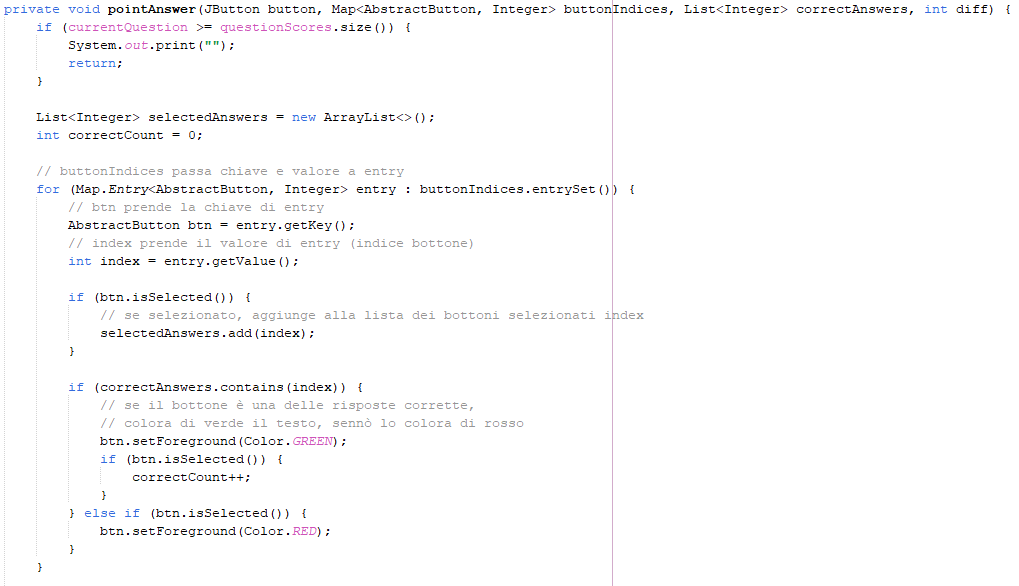
### Funzione displayQuestions

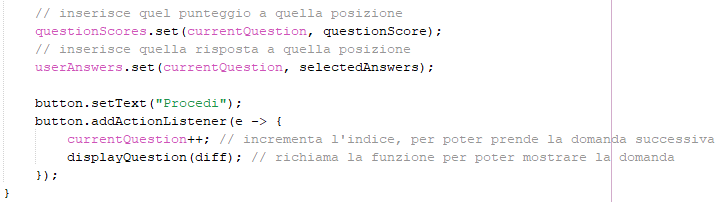
La funzione displayQuestions() serve per poter mostrare a schermo tutte le domande che richiede l’utente. Viene caricato il testo della domanda e i 4 radiobutton o checkbox. Viene caricato un ActionListener a ogni radiobutton o checkbox, così ogni volta che si selezionerà un radiobutton o checkbox verrà attivato l’ActionListener che renderà visibile il tasto per visualizzare la risposta. Dopo aver premuto sul tasto per visualizzare la risposta, l’ActionListener di quel tasto richiamerà un’altra funzione che servirà per aggiungere i punti e visualizzare la risposta. Una volta passato alla prossima domanda, tramite revalidate() e repaint() aggiornerà il layout e ridisegnerà la finestra, **trovato tramite ChatGPT.**



### Funzione pointAnswer

La funzione pointAnswer() serve per tenere conto del punteggio e visualizzare la risposta di ogni domanda. Viene passato la chiave e il valore pulsante per pulsante, se la chiave è selezionata aggiungerà alla lista il pulsante selezionato, se la lista di risposte corrette contiene il valore, il pulsante verrà segnato in verde, altrimenti in rosso. Il punteggio invece, varia a seconda della difficoltà e dalle possibili risposte corrette e quante realmente ne ha selezionate l’utente. Dopo aver premuto sul tasto per procedere alla domanda successiva, revalidate() aggiornerà il layout e repaint() ridisegnerà la finestra, **trovato tramite ChatGPT.**



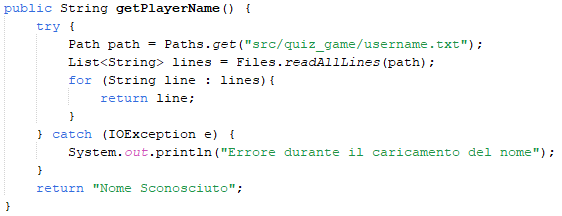


## Classe ResultPanel

La classe ResultPanel mostrerà la classifica dei primi 10 classificati, se presenti, al quiz, con un tasto per visualizzare dettagli, un tasto per iniziare una nuova partita, un tasto per creare un nuovo giocatore e un tasto per chiudere il programma.

### Funzione getPlayerName

La funzione getPlayerName() server per poter ricavare il nome dal file.

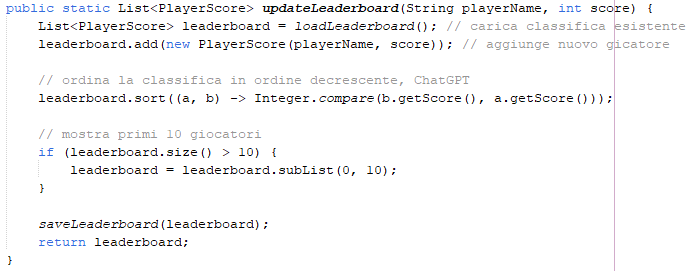


## Classe LeaderboardManager

La classe LeaderboardManager carica la leaderboard già esistente al suo richiamo, questa classe serve per gestire la classifica all’interno della classe ResultPanel.

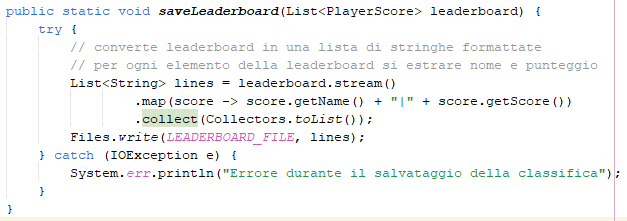
### Funzione updateLeaderboard

La funzione updateLeaderboard() serve per aggiornare la classifica. Viene aggiunto il giocatore che ha appena finito il quiz e riordinata la classifica in modo da avere la classifica in ordine decrescente.



### Funzione saveLeaderboard

La funzione saveLeaderboard() serve per salvare la classfica dopo aver aggiunto un giocatore. La lista leaderboard, che contiene tutti i giocatori coi loro punteggi, viene convertita in una lista, per poi estrarre i nomi e i punteggi dei giocatori e scriverli nel file della classifica.

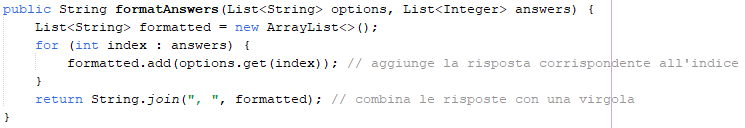


## Classe DetailsPanel

La Classe DetailsPanel mostra in dettaglio ogni domanda che ha fatto il giocatore, con risposta data dal giocatore, risposta corretta e punteggio totalizzato.

### Funzione formatAnswers

La funzione formatAnswers() serve per poter formattare la risposta data dal giocatore e la risposta corretta, in modo da poter essere letta nella schermata dei dettagli. Viene aggiunta alla lista formattata, la risposta che si trova in quella posizione, passando tutte le risposte date dall’utente o tutte le risposte corrette, combinando infine le risposte con una virgola.



# Test

## Protocollo di test

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-001  REQ-01 | **Nome** | Inserimento nome giocatore |
| **Descrizione** | Apparizione tasto “Impostazioni” e salvataggio in file | | |
| **Prerequisiti** |  | | |
| **Procedura** | * Inserire un nome tramite input text * Verificare che il campo venga salvato nel file di tipo txt * Verificare che una volta che il nome venga inserito, il tasto “Impostazioni” appaia | | |
| **Risultati attesi** | Se viene inserito un nome, il tasto “Impostazioni” appare e una volta premuto, il nome viene salvato nel file di tipo txt | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-002  REQ-02 | **Nome** | Inserimento Impostazioni |
| **Descrizione** | Apparizione tasto “Inzia Gioco” | | |
| **Prerequisiti** |  | | |
| **Procedura** | * Selezionare la categoria, la difficoltà e la quantità delle domande * Verificare che il tasto “Inizia Gioco” appaia, dopo aver inserito tutte le impostazioni | | |
| **Risultati attesi** | Il tasto “Inizia Gioco” deve apparire solo quando inserisco tutte le impostazioni | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-003  REQ-03 | **Nome** | Inizio del gioco |
| **Descrizione** | L’utente deve rispondere alle domande che li vengono proposte | | |
| **Prerequisiti** |  | | |
| **Procedura** | * Verificare che le domande siano caricate correttamente in base alla categoria e alla difficoltà scelta * Verificare che le risposte siano correlate in base alla domanda e che le risposte sbagliate vengono prese in modo casuale * Verificare la presenza di immagini in alcune domande e risposte | | |
| **Risultati attesi** | Le domande vengono caricate correttamente, le risposte sono prese dalla domanda giusta e le risposte sbagliate sono prese in modo casuale, e la presenza delle immagini | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-004  REQ-04 | **Nome** | Inserimento risposte |
| **Descrizione** | L’utente deve selezionare le risposte tramite checkbox o radiobutton | | |
| **Prerequisiti** |  | | |
| **Procedura** | * Verificare che le domande con più risposte siano dei checkbox e quelle con una risposta siano un radiobutton * Verificare che una volta finito il tempo per rispondere a una domanda, la risposta viene considerata automaticamente sbagliata * Verificare la giusta distribuzione di punti in base alla difficoltà scelta * Verificare che, se non si ha risposto a tutte le risposte corrette di quella domanda, il punteggio calcolato in base al numero di risposte corrette date e al numero di risposte corrette possibili * Verificare che all’inserimento di una sola risposta, appaia il tasto “Visualizza Risposta” * Verificare che, dopo aver premuto “Visualizza Risposta”, il tasto cambi in “Procedi” | | |
| **Risultati attesi** | Checkbox appare quando ci sono più risposte possibili e radiobutton quando c’è una sola risposta. La risposta venga considerata sbagliata una volta finito il tempo. I punti vengono distribuiti in modo corretto. I punti vengono calcolati in base al numero di risposte corrette date e al numero di risposte corrette possibili. Il tasto “Visualizza Risposta” appare quando inserisco al meno una risposta. Il tasto “Procedi” cambia quando si preme su “Visualizza Risposta” | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-005  REQ-05 | **Nome** | Visualizzazione Risultati |
| **Descrizione** | Al termine del quiz, l’utente può vedere la classifica e il proprio punteggio | | |
| **Prerequisiti** |  | | |
| **Procedura** | * Visualizzare la classifica dei 10 migliori * Verificare che il tasto “Visualizza Dettagli” ti faccia vedere i dettagli delle varie domande * Verificare che il tasto “Nuova Partita” ti faccia cambiare le impostazioni del gioco, ma con le stesse domande * Verificare che il tasto “Nuovo Giocatore” ti faccia inserire un nuovo giocatore * Verificare che il tasto “Chiudi Gioco” ti faccia chiudere il gioco | | |
| **Risultati attesi** | Il tasto “Visualizza Dettagli” fa vedere i dettagli delle domande. Il tasto “Nuova Partita” fa inserire delle nuove impostazioni, ma con lo stesso giocatore. Il tasto “Nuovo Giocatore” fa creare un nuovo giocatore. Il tasto “Chiudi Gioco” chiude il gioco | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-006  REQ-06 | **Nome** | Visualizzazione Dettagli |
| **Descrizione** | L’utente può visualizzare i dettagli di ogni domanda | | |
| **Prerequisiti** |  | | |
| **Procedura** | * Visualizzare dettagli delle domande * Verificare che il tasto “Indietro”, torni alla pagina dei risultati | | |
| **Risultati attesi** | Il tasto “Indietro” torna alla pagina dei risultati | | |

## Risultati test

|  |  |  |
| --- | --- | --- |
| **Test Case** | **Risultato ottenuto** | **Stato** |
| TC-001 | Testo o immagine a conferma del test | Passato |
| TC-002 | Testo o immagine a conferma del test | Passato |
| TC-003 |  | Passato |
| TC-004 |  | Passato |
| TC-005 |  | Passato |
|  |  |  |

## Mancanze/limitazioni conosciute

Descrizione con motivazione di eventuali elementi mancanti o non completamente implementati, al di fuori dei test case. Non devono essere riportati gli errori e i problemi riscontrati e poi risolti durante il progetto.

# Consuntivo

Consuntivo del tempo di lavoro effettivo e considerazioni riguardo le differenze rispetto alla pianificazione (cap. 1.7) (ad esempio Gantt consuntivo).

Stile orizzontale



CONSUNTIVO

Figura 5 Esempio di diagramma di Gantt consuntivo

# Conclusioni

Quali sono le implicazioni della mia soluzione? Che impatto avrà? Cambierà il mondo? È un successo importante? È solo un’aggiunta marginale o è semplicemente servita per scoprire che questo percorso è stato una perdita di tempo? I risultati ottenuti sono generali, facilmente generalizzabili o sono specifici di un caso particolare? ecc.

## Sviluppi futuri

Migliorie o estensioni che possono essere sviluppate sul prodotto.

## Considerazioni personali

Cosa ho imparato in questo progetto? ecc.

# Bibliografia

## Bibliografia per articoli di riviste:

1. Cognome e nome (o iniziali) dell’autore o degli autori, o nome dell’organizzazione,
2. Titolo dell’articolo (tra virgolette),
3. Titolo della rivista (in italico),
4. Anno e numero
5. Pagina iniziale dell’articolo.

## Bibliografia per libri

1. Cognome e nome (o iniziali) dell’autore o degli autori, o nome dell’organizzazione,
2. Titolo del libro (in italico),
3. ev. Numero di edizione,
4. Nome dell’editore,
5. Anno di pubblicazione,
6. ISBN.

## Sitografia

1. URL del sito (se troppo lungo solo dominio, evt completo nel diario),
2. Eventuale titolo della pagina (in italico),
3. Data di consultazione (GG-MM-AAAA).

**Esempio:**

* http://standards.ieee.org/guides/style/section7.html, *IEEE Standards Style Manual*, 07-06-2008.

# Glossario

|  |  |
| --- | --- |
| **Termine** | **Significato** |
| ASCII | American Standard Code for Information Interchange, codice per la codifica di caratteri. |
|  |  |
|  |  |
|  |  |
|  |  |
| ZFS | Zettabyte File System, filesystem opernsource a 128 bit con funzionalità avanzate. |

# Indice delle figure

[Figura 1 Esempio di diagramma di Gantt 7](#_Toc124775984)

[Figura 2 Esempio di diagramma di Gantt 8](#_Toc124775985)

[Figura 3 Diagramma ER Chen 10](#_Toc124775986)

[Figura 4 Diagramma ER Barker 11](#_Toc124775987)

[Figura 5 Esempio di diagramma di Gantt consuntivo 16](#_Toc124775988)

# Allegati

Elenco degli allegati, esempio:

* Diari di lavoro
* Codici sorgente/documentazione macchine virtuali
* Istruzioni di installazione del prodotto (con credenziali di accesso) e/o di eventuali prodotti terzi
* Documentazione di prodotti di terzi
* Eventuali guide utente / Manuali di utilizzo
* Mandato e/o QdC
* Prodotto
* …