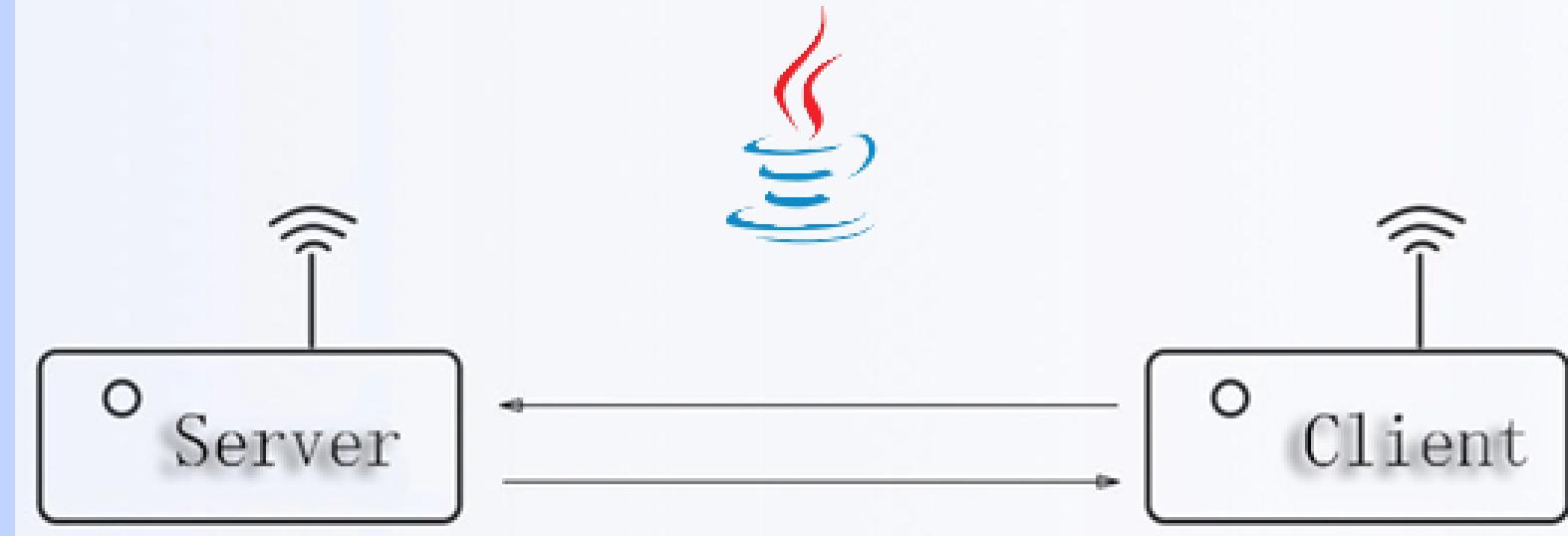


PROGETTO SOCKET JAVA

GARA DI PROGRAMMAZIONE

Socket



COSA FA IL NOSTRO PROGETTO

Il progetto implementa un sistema di Gara di quiz di programmazione multi-utente basato su architettura Client-Server utilizzando Sockets in Java.

I partecipanti, dopo aver effettuato l'accesso con un nome, sono sottoposti a una serie di domande di programmazione pre-caricate, ciascuna con un punteggio assegnato.

Una volta che i partecipanti terminano di rispondere a tutte le domande, il sistema calcola il loro punteggio totale e aggiorna in tempo reale una classifica globale visualizzata su tutti i client connessi.

REQUISITI FUNZIONALI

ID	DESCRIZIONE
RF1	Avvio del sistema: Il progetto deve avviarsi e funzionare correttamente.
RF2	Avvio del server: Il server deve avviarsi correttamente e rimanere in ascolto per le connessioni dei client sulla porta designata (1234).
RF3	Avvio del client: Il client deve avviarsi correttamente e mostrare l'interfaccia grafica (GUI) iniziale (PaginaIniziale).
RF4	Inserimento nome: All'avvio, il client deve richiedere all'utente l'inserimento del proprio nome e inviarlo al server subito dopo la connessione.
RF5	Lettura domande: Il server deve leggere correttamente da un file di testo (questions.txt) le domande, le risposte corrette e i relativi punteggi.
RF6	Invio domande casuali: Il server deve mescolare le domande e inviarle al client una per una, in ordine casuale.
RF7	Visualizzazione domande: Il client (PaginaDomande) deve visualizzare correttamente le domande ricevute dal server, aggiornando il contatore della domanda corrente.
RF8	Invio risposte: Il client deve inviare correttamente le risposte dell'utente al server tramite il pulsante "Invia".
RF9	Calcolo punteggio: Il server deve assegnare un punteggio totale in base alla corrispondenza tra le risposte inviate dal client e le risposte corrette .
RF10	Classifica: Il server deve salvare il punteggio, aggiornare la classifica globale e inviare un broadcast di aggiornamento a tutti i client attivi quando un partecipante termina il quiz.
RF11	Visualizzazione classifica: Il client (PaginaClassifica) deve visualizzare correttamente la classifica finale in ordine decrescente di punteggio.

REQUISITI NON FUNZIONALI

ID	DESCRIZIONE
RNF1	Prestazioni: La comunicazione tra client e server deve avvenire senza ritardi percepibili e il cambio schermata deve avvenire in meno di 1 secondo.
RNF2	Robustezza: Il sistema deve continuare a funzionare correttamente anche in caso di input imprevisti, vuoti o incompleti.
RNF3	Stabilità: Il sistema deve mantenere una connessione stabile tra client e server per tutta la durata del quiz.

FORMATO DEL PROTOCOLLO

Il protocollo si basa sullo scambio di stringhe testuali su Sockets TCP in Java.

I messaggi sono interpretati attraverso prefissi o la posizione sequenziale nel flusso di comunicazione.

MESSAGGI STRUTTURATI CON TAG E DELIMITATORE | DA SERVER A CLIENT

MITTENTE → DESTINATARIO	TAG/TIPO DI MESSAGGIO	SINTASSI ESATTA INVIATA	DATI CONTENUTI
S → C	DOMANDA	`"DOMANDA `	" + d.testo` (testo della domanda)
S → C	PUNTEGGIO	`"PUNTEGGIO `	" + punteggioTotale` (punteggio realizzato da quell'utente)
S ↔ C	CLASSIFICA	`"CLASSIFICA `	" + classifica

MESSAGGI SEMPLICI DA CLIENT A SERVER

MITTENTE → DESTINATARIO	TIPO DI MESSAGGIO	SINTASSI ESATTA INVIATA	DATI CONTENUTI
C → S	Nome Utente	nome (Stringa letta da in.readLine())	Stringa contenente il nome inserito dal giocatore.
C → S	Risposta Utente	risposta (Stringa letta da in.readLine())	Stringa contenente la risposta testuale.
C → S	Comando Richiesta Classifica	"DAMMI_CLASSIFICA"	Comando per forzare il Server a inviare la classifica subito.
C → S	Comando Chiusura	"CHIUDI_CONNESSIONE"	Comando per segnalare la disconnessione controllata al Server.

SEQUENZA DI ESECUZIONE DEL PROTOCOLLO

FASE 1: INIZIALIZZAZIONE E IDENTIFICAZIONE

1. Connessione Client: Il Client stabilisce la connessione TCP sulla porta designata (1234).
 2. Invio Nome ($C \rightarrow S$): Il Client invia immediatamente il Nome Utente (nome) al Server.
- Il primo messaggio che riceve il server.

FASE 2: SVOLGIMENTO DEL QUIZ

1. Invio Domanda ($S \rightarrow C$): Il Server invia la prima domanda casuale utilizzando il tag DOMANDA (DOMANDA|testo...).
2. Invio Risposta ($C \rightarrow S$): Il Client invia la Risposta Utente (risposta) al Server dopo che l'utente ha premuto "Invia".
3. Ciclo: I passi sopra vengono ripetuti per tutte le domande (totaleDomande = 10).

FASE 3: CONCLUSIONE E CLASSIFICA

1. Calcolo e Notifica Punteggio ($S \rightarrow C$): Terminato il ciclo, il Server calcola il punteggio, aggiorna la classifica globale e invia il punteggio finale all'utente con il tag PUNTEGGIO (PUNTEGGIO|punteggio).
2. Richiesta Classifica ($C \rightarrow S$): Dopo il quiz, la PaginaClassifica si apre e invia il comando "DAMMI_CLASSIFICA" al Server per assicurarsi di ricevere la classifica più recente.
3. Broadcast Classifica ($S \leftrightarrow C$): Il Server chiama broadcastClassifica(), inviando il messaggio CLASSIFICA (CLASSIFICA|...) a tutti i client attivi.
4. Ascolto Classifica ($S \rightarrow C$): Il Client rimane in ascolto nel thread dedicato e riceve gli aggiornamenti CLASSIFICA
5. Terminazione ($C \rightarrow S$): Alla chiusura della finestra, il Client invia il comando "CHIUDI_CONNESSIONE", permettendo al Server di chiudere il gestore di questo client e la socket in modo controllato.

CASI D'USO

ID	CASO D'USO	ATTORI	RESPONSABILITÀ NEL SISTEMA
UC1	Avvia Sessione di Gioco	Partecipante	L'utente avvia la trasmissione
UC2	Somministra Quiz Casuale	Server	Il server pone le domande in ordine casuale.
UC3	Registra e Calcola Risultato	Server	Il server calcola il risultato e aggiorna la classifica
UC4	Richiede Classifica Corrente	Partecipante	Il client richiede la classifica aggiornata
UC5	Trasmette Aggiornamento Classifica	Server	Il server manda la classifica in aggiornata in funzione di UC3 e UC4
UC6	Visualizza Classifica Aggiornata	Partecipante	L'utente ottiene(UC5) e visualizza la classifica aggiornata.
UC7	Chiusura	Partecipante	L'utente chiude la pagina e chiude quindi anche la connessione

UML

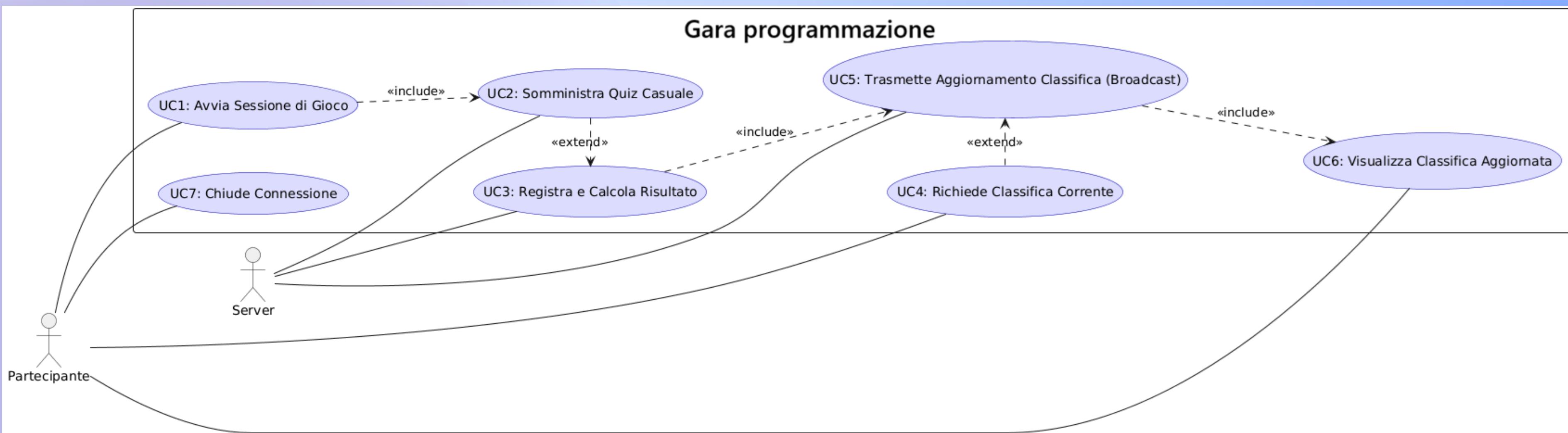
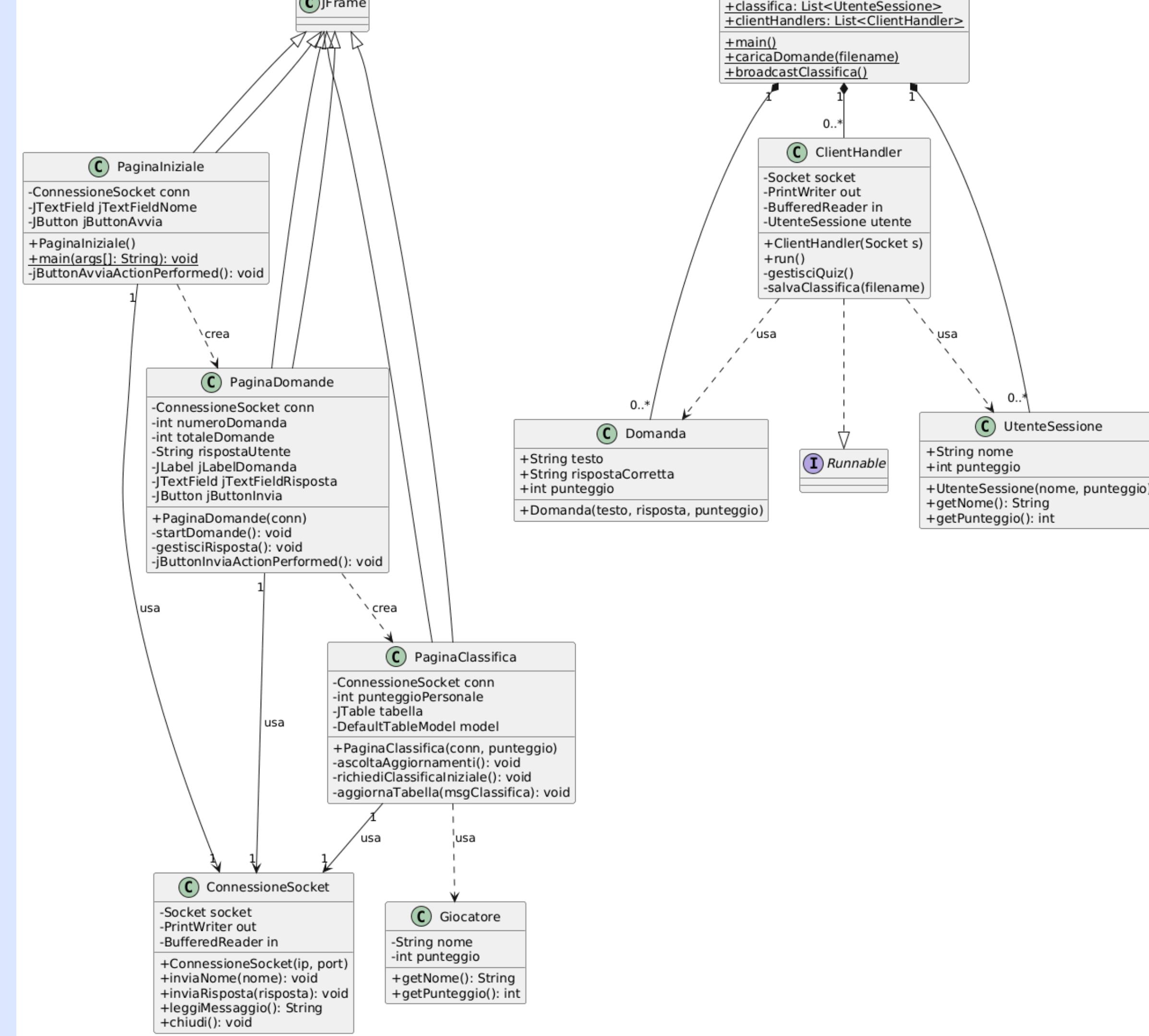


DIAGRAMMA DELLE CLASSI



WIREFRAME

All' avvio dopo aver inserito il nome , premendo il bottone Avvia, si apre la pagina delle domande.

Su ogni domanda , bisogna inserire la risposta propria , poi si preme invia e vengono mostrate in sequenza le varie domande.

Arrivati all' ultima domanda , premendo invia , apre la pagina della classifica

Inserire qui il proprio nome
Poi premere il bottone Avvia

Inserisci il nome

Avvia gara

Avvia

Domanda i di n
programma

risposta aperta

invia

Hai realizzato ... punti

Classifica

Nome	Punteggio

Qui si vede il nostro punteggio,
che è anche nella tabella

Inserire qui la propria risposta
Dopo di che premere il bottone Invia

MOCKUP : LE NOSTRE GUI

The image displays two separate windows side-by-side, each featuring a light gray header bar with standard window controls (minimize, maximize, close).

Left Window (Inserisci il nome):

- Title:** INSERISCI IL NOME
- Content:** A large white input field.
- Buttons:** A horizontal line separates the input field from a blue button labeled "AVVIA".

Right Window (Domanda 1 di 12):

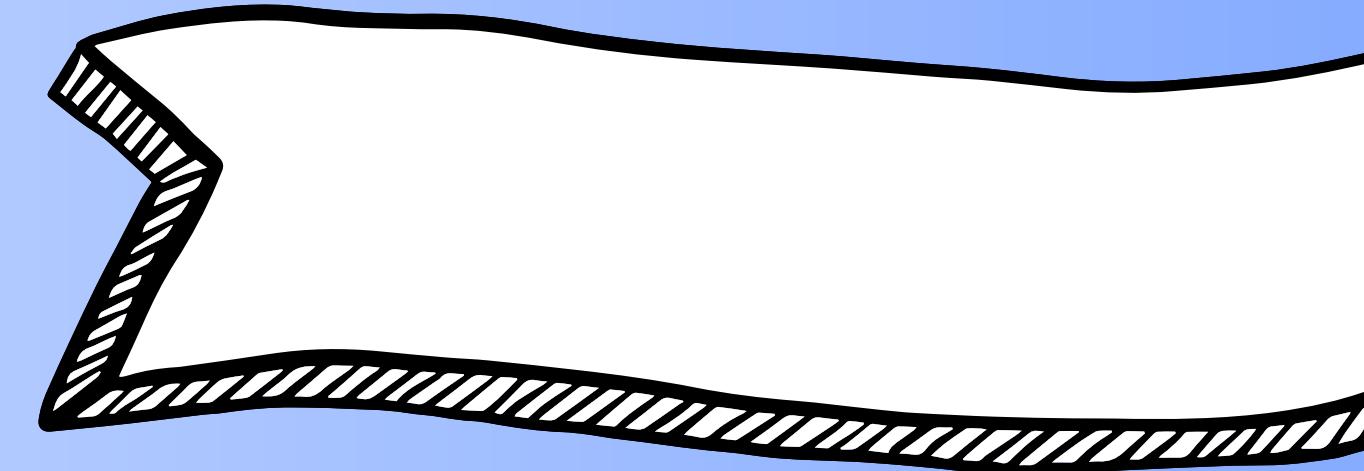
- Title:** Domanda 1 di 12
- Content:** A question: "Come si selezionano tutti i record da utenti in SQL?" followed by a large white input field.
- Buttons:** A blue button labeled "Inserisci la tua risposta" next to the input field, and a smaller blue button labeled "Invia" below it.

The image shows a third window titled "Classifica" with a light gray header bar.

Content:

- A message: "Hai realizzato 29 punti"
- A table with two columns: "Nome" and "Punteggio".
- The table contains one row with the data: "Abed" and "29".

CONCLUSIONI



Il progetto ha consentito di applicare e approfondire le competenze in programmazione Java, concentrandosi sulla comunicazione di rete e la programmazione concorrente tramite Sockets e Thread. La parte più complessa è stata sicuramente come mostrare la classifica