Progetto d’esame

Tecnologie e applicazioni web, a.a 2017/2018

Si realizzi un’applicazione web, comprensiva di server con API stile REST e front-end di tipo SPA, che permetta ad una community di utenti di giocare al gioco della battaglia navale. Il sistema deve permettere la gestione degli utenti (registrazione, login ed eventuale cancellazione), la ricerca di giocatori disponibili ad iniziare una partita, la logica di gioco e una classifica generale degli utenti in base al numero di partite vinte, perse e giocate.

La logica di gioco

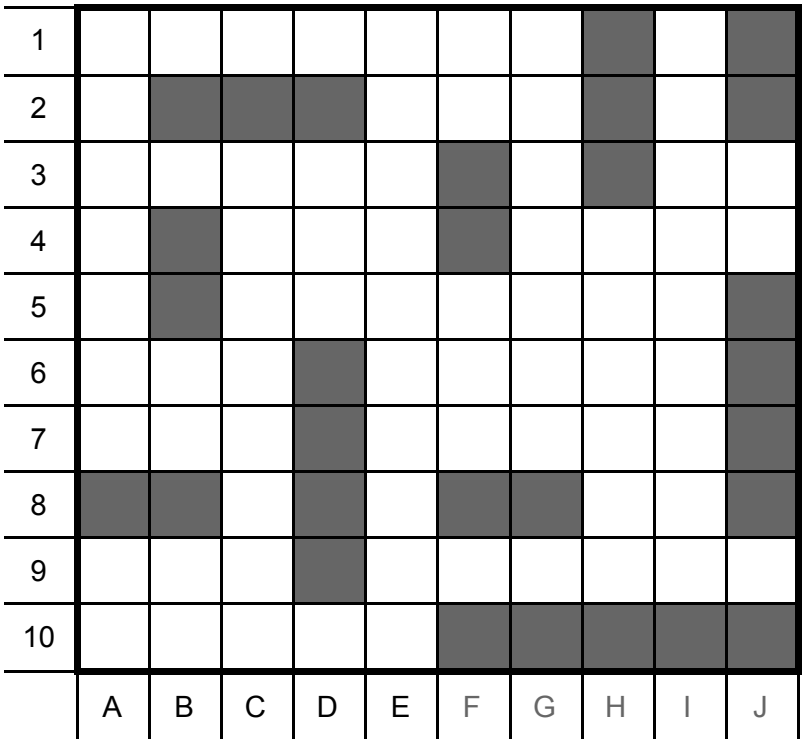
Una partita del gioco della battaglia navale si svolge a turni tra due utenti. Ciascun utente ha a disposizione due tabelle di dimensione fissa di `10x10` caselle. Una tabella si riferisce al proprio campo di gioco, l’altra si riferisce al campo dell’avversario.
La partita si svolge in due fasi. Nella prima fase ciascun utente posiziona nel proprio campo di gioco delle navi che occupano una sequenza di caselle di dimensione fissa. Nella seconda fase ciascun utente, a turno, seleziona una casella avversaria su cui “sparare un colpo di cannone”. Se il colpo va a segno, cioè la casella selezionata è occupata da una nave avversaria, la casella è marcata come “colpita”. Se il colpo è invece sparato in una casella che non contiene alcuna nave avversaria, questa viene marcata come “mancata”. Quando tutte le caselle occupate da una determinata nave avversaria sono colpite, il gioco indica che la nave corrispondente è stata affondata.
Il giocatore che per primo colpisce e affonda tutte le navi avversarie vince la partita.

Fase di posizionamento

Il gioco prevede (senza possibilità di modifica) le seguenti navi:

Tipo	Dimensione (righe x colonne)	Quantità
Cacciatorpediniere	1x2 o 2x1	4
Sottomarino	1x3 o 3x1	2
Corazzata	1x4 o 4x1	2
Portaerei	1x5 o 5x1	1

Ciascuna nave può essere posizionata solo in orizzontale oppure in verticale. Può essere adiacente ai bordi del campo di gioco ma non ad una qualsiasi altra nave.
Un esempio di possibile posizionamento delle navi nel campo da gioco è il seguente:



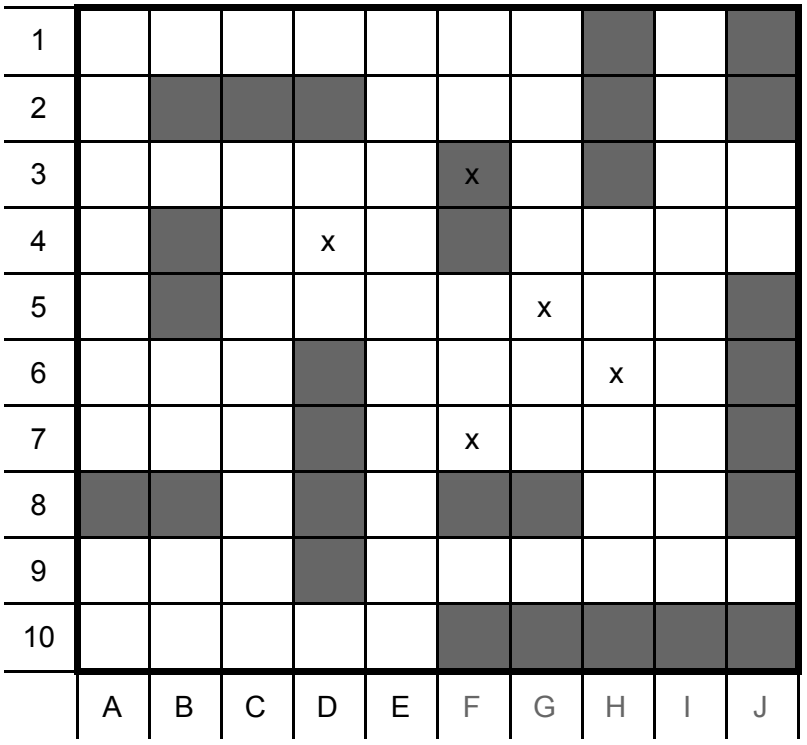
La fase di posizionamento avviene in modo indipendente tra i due giocatori senza che ciascuno di essi possa vedere o intervenire nel posizionamento avversario.
Quando entrambi i giocatori terminano il posizionamento di tutte le navi che hanno a disposizione si passa alla fase di gioco.

Fase di gioco

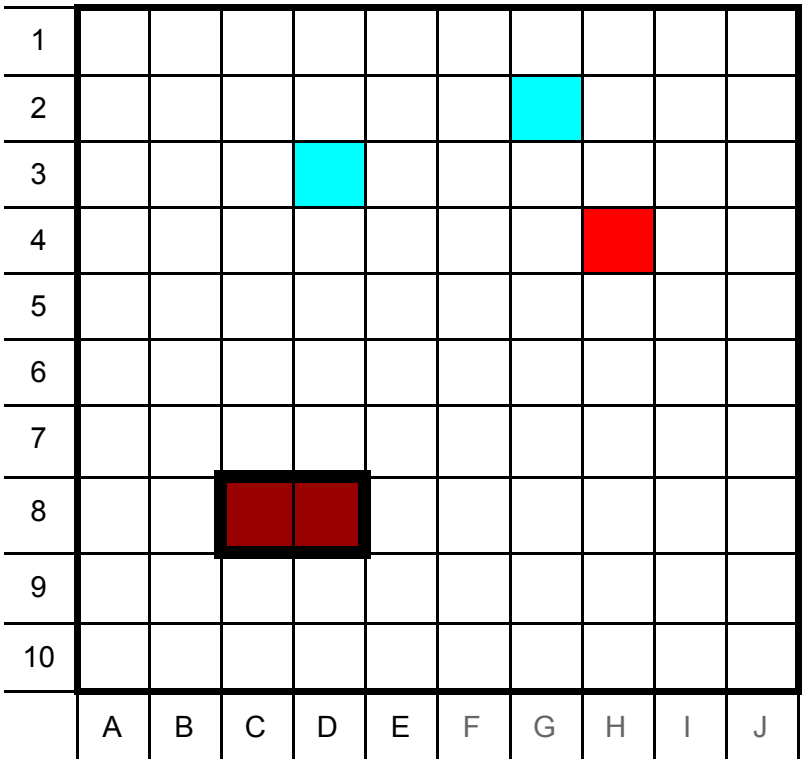
Il giocatore che inizia la partita è estratto a sorte. Durante il proprio turno, il giocatore deve sparare un colpo di cannone selezionando una casella del campo avversario. Il colpo può essere sparato soltanto se non sono stati sparati altri colpi precedentemente nella stessa casella. Il sistema verifica se il colpo è andato a segno, colpendo una nave avversaria, oppure no. Lo stato della casella selezionata (mancata o colpita) è indicato nella tabella che si riferisce al campo avversario e il turno passa all'altro giocatore.
Il giocatore avversario effettua le stesse operazioni, seleziona una casella e riceve notifica di eventuale navi colpite e/o affondate. Nel proprio campo viene visualizzata la casella su cui l'avversario ha sparato il proprio colpo.
Un esempio della possibile situazione di gioco di un giocatore dopo 5 turni è indicato nelle due tabelle seguenti. La grafica indica con una x la casella su cui l'avversario ha sparato un colpo, mentre in rosso le caselle colpite nel campo avversario, in blu quelle mancate e in

rosso scuro le navi affondate. La grafica è puramente indicativa e non va necessariamente replicata nell'implementazione da realizzare

Proprio campo:



Campo avversario:



Architettura

Il sistema deve essere composto da:

- Un web service con API in stile REST, implementato in Javascript o Typescript su ambiente Node.js. Il servizio deve inoltre utilizzare:
    - Il DBMS MongoDB per gestire la persistenza dei dati
    - Il middleware Express.js per la gestione del routing

- Una web application client in stile SPA realizzata con il framework Angular
- Un applicazione mobile ibrida realizzata con Apache Cordova
- Un client desktop realizzato con Electron

Le applicazioni mobile e desktop sono sostanzialmente identiche alla web application Angular browser-based ma possono aggiungere funzionalità aggiuntive, a discrezione dello studente (ad esempio l'applicazione mobile può presentare elementi di interfaccia tipici del contesto mobile). L'applicazione mobile può utilizzare componenti e funzionalità fornite dal framework Ionic

Funzionalità

Il sistema deve implementare le seguenti funzionalità:

1. Gestione degli utenti
    a. Registrazione di nuovi utenti con almeno due diversi ruoli: giocatori e amministratori
    b. Possibilità di cancellazione di utenti esistenti da parte degli amministratori
    c. Login degli utenti con le proprie credenziali di accesso. Tutte le funzionalità del sistema (a parte la registrazione di nuovi utenti) sono accessibili previo login obbligatorio
2. Interazione tra utenti
    a. Invio di messaggi privati tra due utenti qualsiasi
    b. Visualizzazione della lista di messaggi ricevuti da altri utenti
    c. Visualizzazione delle statistiche relative ad un utente (numero di partite vinte, perse, giocate)
    d. Classifica generale dei 10 migliori giocatori (in base al numero di partite vinte)
3. Gestione partite
    a. Creazione di una nuova partita e attesa giocatori (la partita appena creata è nello stato di "attesa")
    b. Ricerca delle partite nello stato di attesa
    c. Partecipazione ad una partita in attesa. Quando un giocatore si unisce ad una partita in attesa la partita inizia e viene rimossa dalla lista delle partite in attesa

4. Logica di gioco
    a. Una partita in corso segue la logica di gioco descritta in precedenza. Un giocatore può partecipare ad una sola partita alla volta. Quando la partita termina, entrambi gli utenti possono tornare a creare una nuova partita o partecipare ad una in attesa

I gruppi sono liberi di implementare funzionalità aggiuntive oltre quelle precedentemente elencate. L'implementazione o meno di funzionalità aggiuntive non preclude né garantisce il conseguimento della lode ma contribuisce a definire un giudizio positivo.

Consegna

La consegna avviene esclusivamente mediante piattaforma moodle alla pagina:

https://moodle.unive.it/mod/assign/view.php?id=64284

Ciascun gruppo deve consegnare un solo file, in formato zip, chiamato `<nomegruppo>.zip`.

Il file zip al suo interno deve contenere:

- La relazione di ciascuno studente, in formato pdf col nome `<nome>_<cognome>_<matricola>.pdf`
- Un file README.txt che spiega chiaramente come eseguire l'intera applicazione. Ad esempio contiene la lista di comandi per installare le dipendenze (npm install o altro), eseguire il processo del DBMS, il server e i relativi client
- Tutti i sorgenti necessari alla sua esecuzione

NOTA: Non consegnare le librerie esterne installate attraverso il gestore pacchetti. In altre parole, tutte le directory `node_modules` devono essere vuote.

Relazione

Congiuntamente al progetto, ciascuno studente deve consegnare una relazione.
La relazione va compilata in forma strettamente individuale e porrà le basi per la discussione orale del progetto.

La relazione deve contenere:

- Una descrizione dell'architettura del sistema, quali sono le componenti e in che modo queste concorrono a soddisfare le features richieste;
- Una descrizione del modello dei dati utilizzato. Quali sono le collezioni e qual'è la struttura dei documenti di ciascuna collezione che vengono memorizzati nel database;
- Una descrizione delle API fornite dalla componente server. La descrizione deve contenere in modo chiaro la lista degli endpoint, degli eventuali parametri e il formato dei dati che vengono scambiati nelle richieste HTTP;
- Una descrizione di come è stata realizzata l'autenticazione degli utenti, con relativo workflow;
- Una descrizione del client web realizzato con il framework Angular. In particolare, la lista dei vari Component, dei Servizi e delle Routes (se implementate);
- Alcuni esempi, corredati possibilmente da screenshots, del workflow tipico dell'applicazione.

Altre informazioni

Per domande o chiarimenti è sempre possibile contattare il professore via mail all'indirizzo filippo.bergamasco@unive.it

Per domande sulla modalità di esame si faccia inoltre riferimento alla seguente pagina:

https://moodle.unive.it/mod/page/view.php?id=58582

Filippo Bergamasco,
Tecnologie e Applicazioni Web, a.a. 2017/2018