Hackathon (Traccia 2)

Un hackathon, ovvero una "maratona di hacking", è un evento durante il quale team di partecipanti si sfidano per progettare e implementare nuove soluzioni basate su una certa tecnologia o mirate a un certo ambito applicativo. Ogni hackathon ha un titolo identificativo, si svolge in una certa sede e in un certo intervallo di tempo (solitamente 2 giorni) e ha un organizzatore specifico (registrato alla piattaforma). L'organizzatore seleziona un gruppo di giudici (selezionati tra gli utenti della piattaforma, invitandoli). Infine, l'organizzatore apre le registrazioni, che si chiuderanno 2 giorni prima dell'evento. Ogni evento avrà un numero massimo di iscritti e una dimensione massima del team. Durante il periodo di registrazione, gli utenti possono registrarsi per l'Hackathon di loro scelta (eventualmente registrandosi sulla piattaforma se non lo hanno già fatto). Una volta iscritti, gli utenti possono formare team. I team diventano definitivi quando si chiudono le iscrizioni. All'inizio dell'hackathon, i giudici pubblicano una descrizione del problema da affrontare. Durante l'hackathon, i team lavorano separatamente per risolvere il problema e devono caricare periodicamente gli aggiornamenti sui "progressi" sulla piattaforma come documento, che può essere esaminato e commentato dai giudici. Alla fine dell'hackathon, ogni giudice assegna un voto (da 0 a 10) a ciascun team e la piattaforma, dopo aver acquisito tutti i voti, pubblica le classifiche dei team. Requisito per gruppi da tre persone. Ogni partecipante può contattare un altro utente iscritto e chiedere di unirsi al team di cui fa parte, fornendo un messaggio motivazionale; l'utente può accettare o rifiutare la richiesta. Un iscritto può essere contattato solo se il team di cui è membro non ha raggiunto la dimensione massima.

Descrizione Struttura

1. Classe Utente

Ruolo: Classe base per rappresentare un utente generico del sistema.

Attributi:

- id, username, email, nome, cognome: Identifica l’utente.
- isRegistrato: Indica se l'utente ha completato la registrazione.

Metodi:

- Costruttore: Inizializza i dati dell'utente.
- Getter/Setter: Consentono un accesso controllato agli attributi.

- registra(): Imposta lo stato dell'utente come registrato.

Motivazione: La classe fornisce una struttura comune che si estende alle funzionalità specifiche di sottoclassi come Partecipante, Giudice e Organizzatore, evitando possibili duplicazioni di codice.

2. Classe Hackathon

Ruolo: Rappresenta l'evento Hackathon e gestisce le informazioni principali.

Attributi:

- titolo, sede, dataInizio, dataFine, maxIscritti, maxPersoneInUnTeam, inizioIscrizioni, descrizioneProblema, iscrizioniAperte: Descrivono l'evento e tutte le sue caratteristiche
- giudici, partecipanti, teams: Liste per gestire giudici, partecipanti e teams.

Metodi:

- apriIscrizioni(), chiudiIscrizioni(): Aprono e chiudono le iscrizioni.
- aggiungiGiudice(), aggiungiPartecipante(), aggiungiTeam(): Consentono di aggiungere utenti e team all'hackathon.
- pubblicaProblema(Problema problema): Pubblica il problema.
- getClassifica(): Ordina i team in base al punteggio totale ricevuto dai giudici.

3. Classe Team

Ruolo: Rappresenta un gruppo di partecipanti.

Attributi:

- id, hackathon, nome, fondatore: Identificano il team e il fondatore originario.
- membri, voti, progressi: Liste che gestiscono i membri del team, i voti ricevuti e i progressi effettuati.

Metodi:

- aggiungiMembro(): Aggiunge un nuovo partecipante al team.
- aggiungiProgresso(): Registra un progresso del team.
- puoAccettare(): Controlla se il team ha spazio per nuovi membri.

Motivazione: Fornisce una struttura chiara per rappresentare un team e le sue interazioni. Il metodo puoAccettare() facilita la verifica della capacità del team, migliorando la coerenza.

4. Classe Partecipante

Ruolo: Estende Utente per rappresentare i partecipanti all'hackathon.

Metodi:

- accettaInvito(Invito invito), rifiutaInvito(Invito invito): Consentono di rispondere agli inviti.
- inviaRichiestaConMessaggio(Partecipante destinatario, Team team, String messaggio): Invita un altro partecipante a unirsi al team.

5. Classe Giudice

Ruolo: Estende Utente per rappresentare i giudici dell'evento.

Metodi:

- selezionato(Hackathon hackathon): Notifica al giudice la selezione.
- commentaProgresso(Progresso p, String testo): Aggiunge un commento costruttivo al progresso di un team.

Motivazione: Valuta e monitora il progresso dei team.

6. Classe Invito

Ruolo: Gestisce l'invito di un partecipante a un team.

Attributi:

- mittente, destinatario, team, messaggio: Descrivono i dettagli dell'invito.
- stato: Indica lo stato dell'invito (PENDING, ACCEPTED, DECLINED).

Metodi:

- accetta(), rifiuta(): Consentono al destinatario di accettare o rifiutare l'invito.

7. Classe Problema

Ruolo: Contiene la descrizione del problema da risolvere durante l'hackathon.

Attributi:

- descrizione, hackathon: Forniscono il contesto del problema.

8. Classe Progresso

Ruolo: Rappresenta il progresso dei team durante l'evento.

Attributi:

- team, data, documento, commentiGiudici: Tracciano i progressi e il feedback dei giudici.

9. Classe Commento

Ruolo: Rappresenta il feedback di un giudice su un progresso.

Attributi:

- autore, testo, data: Descrivono il commento e il suo contesto.

10. Classe Organizzatore

Ruolo: Estende Utente per rappresentare l'organizzatore di un hackathon.

Metodi:

- id, username, email, nome, cognome: Identifica l'organizzatore.
- Hackathon creaHackathon(int id, String titolo, String sede, LocalDate dataInizio, LocalDate dataFine, int maxIscritti, int maxPersoneInUnTeam, LocalDate inizioIscrizioni): Crea un nuovo evento Hackathon con i parametri specificati e ne restituisce l'oggetto.

Motivazione:
Rappresenta l'utente che ha il compito di creare e avviare un hackathon. È responsabile della creazione dell'evento, comprese le sue caratteristiche principali e i vincoli di partecipazione.

11. Classe Voto

Ruolo: Rappresenta il punteggio assegnato da un giudice a un team al termine dell'hackathon.

Attributi:

- giudice: Il giudice che ha assegnato il voto.
- team: Il team a cui è stato assegnato il voto.
- valore: Il punteggio assegnato (compreso tra 0 e 10).

Metodi:

- giudice, team, valore: Indica il valore del voto assegnato, da quale giudice e nei confronti di quale team.
- getGiudice(): Restituisce il giudice che ha assegnato il voto.

- setGiudice(Giudice giudice): Imposta il giudice.
- getTeam(): Restituisce il team valutato.
- setTeam(Team team): Imposta il team a cui si riferisce il voto.
- getValore(): Restituisce il valore del voto.
- setValore(int valore): Imposta il valore del voto.

Motivazione: I voti sono utilizzati dalla piattaforma per calcolare il punteggio totale di ogni team e generare la classifica finale dell'hackathon.

Nome o indirizzo della repository github

Nome: TheHeroesSoul/Hackathon
Link: https://github.com/TheHeroesSoul/Hackathon