UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II

PROGETTAZIONE DI UN CLASS DIAGRAM VOLTO ALL'IMPLEMENTAZIONE DI UN HACKATHON

Autore:

Corso di OBJECT ORIENTATION

Gabriele Letizia N86005267

Fabio Iannicelli N86005383

ANNO ACCADEMICO 2024/2025

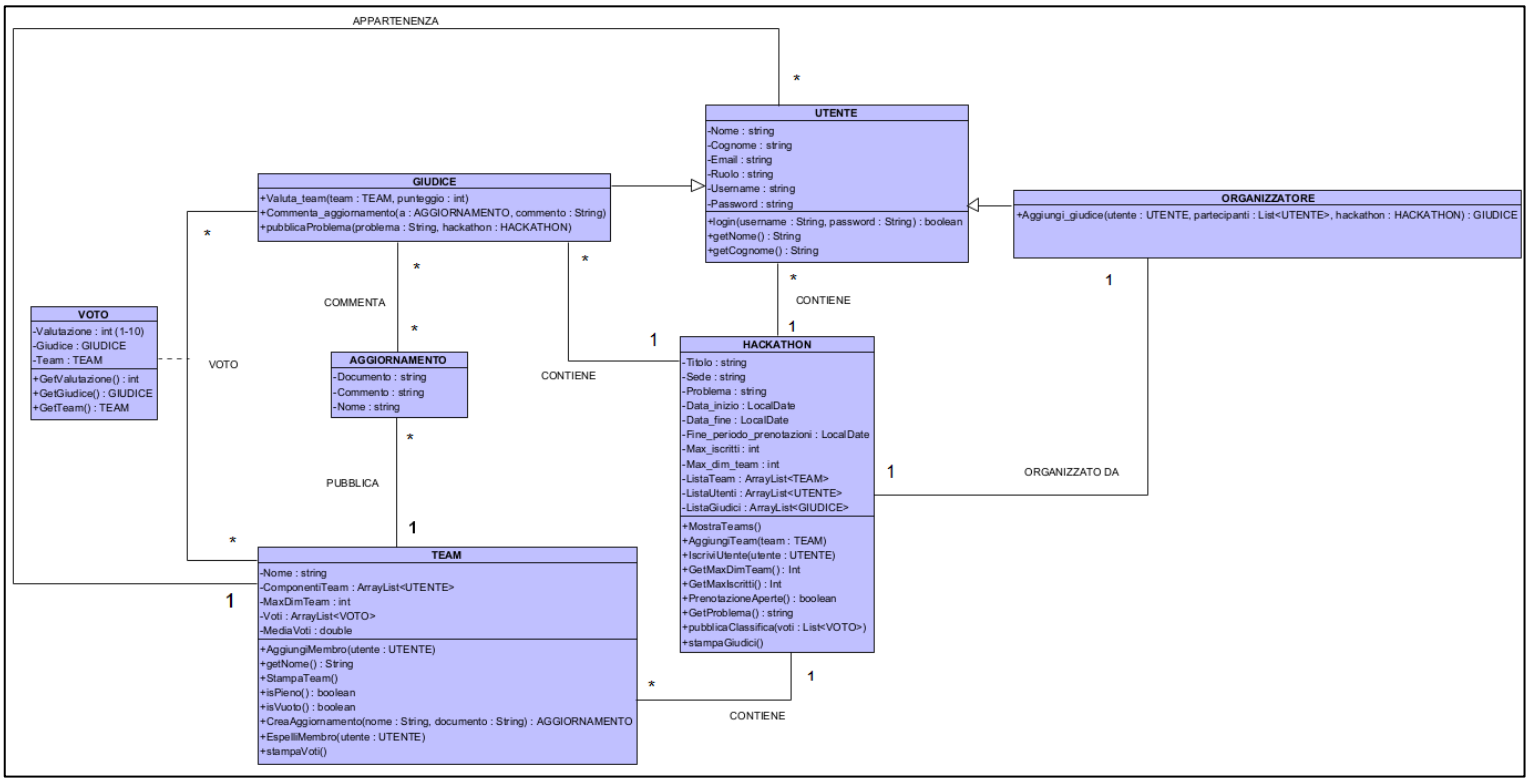
REPOSITORY GITHUB: https://github.com/galetizia/ProgettoObject.git

PROGETTAZIONE DEL CLASS DIAGRAM

1.1 | INTRODUZIONE

Un hackathon, ovvero una "maratona di hacking", è un evento durante il quale, vari team di partecipanti si sfidano per progettare e implementare nuove soluzioni basate su una certa tecnologia o mirate a un certo ambito applicativo.

1.2 | CLASS DIAGRAM



DOCUMENTAZIONE RELATIVA ALLE CLASSI

2.1 | CLASSE HACKATHON

La classe Hackathon rappresenta una singola edizione di un hackathon, con attributi e metodi per gestire le informazioni e le operazioni relative all'evento.

ATTRIBUTI:

- **Titolo**: Nome identificativo dell'hackathon.
- **Sede**: Luogo in cui si svolge l'evento.
- **Data inizio e data fine**: Date di inizio e conclusione dell'hackathon.
- **Fine periodo prenotazioni**: Termine ultimo per le iscrizioni (solitamente due giorni prima della data di inizio).
- **Max iscritti**: Numero massimo di partecipanti ammessi.
- **Max dimensione team**: Numero massimo di membri per team.
- **Lista partecipanti**: Elenco degli utenti iscritti.
- **Lista team**: Elenco dei team registrati.
- **Lista giudici**: Elenco dei giudici assegnati all'evento.

METODI:

- **MostraTeams()** : Stampa a video la lista dei team partecipanti.
- **AggiungiTeam(team : Team)** : Aggiunge un team alla lista dei team registrati.
- **IscriviUtente(utente : Utente)** : Aggiunge un utente alla lista dei partecipanti.
- **Int GetMaxDimTeam()** : Restituisce la dimensione massima consentita per un team.
- **Int GetMaxIscritti()** : Restituisce il numero massimo di partecipanti ammessi.
- **Boolean PrenotazioniAperte()** :
    - Restituisce *TRUE* se la data corrente è antecedente alla scadenza delle iscrizioni.
    - Restituisce *FALSE* se il termine è scaduto.
- **String GetProblema()** : Restituisce la traccia del problema assegnato ai team.
- **PubblicaClassifica(voti : List<Voto>)** : Stampa la classifica dei team con le relative medie dei voti.
- **StampaGiudici()** : Stampa a video l'elenco dei giudici

RELAZIONI CON ALTRE CLASSI:

- **Organizzatore**:
  - Ogni hackathon ha **esattamente un organizzatore**
- **Utente**:
  - Un hackathon può avere **N utenti** iscritti.
- **Giudice**:
  - Un hackathon può avere **N giudici**.
- **Team**:
  - Un hackathon può avere **N team** registrati.

2.2 | CLASSE TEAM

La classe Team rappresenta un gruppo di lavoro partecipante a un hackathon, con attributi e metodi per gestire membri, votazioni e aggiornamenti.

ATTRIBUTI:

- **Nome** : Identificativo unico del team.
- **Dimensione massima** : Numero massimo di membri (definito dall'hackathon di riferimento).
- **Lista componenti** : Utenti appartenenti al team.
- **Media voti** : Punteggio medio assegnato dai giudici.
- **Lista voti** : Dettaglio delle valutazioni ricevute (uno per giudice).

METODI:

- **AggiungiMembro(utente :Utente)** : Aggiunge un utente alla lista dei componenti, se non si è superata la dimensione massima.
- **String GetNome()** : Restituisce il nome del team.
- **StampaTeam()** : Stampa a video il nome del team ed elenco dei membri.
- **Boolean isPieno()** :
    - Restituisce *TRUE* se il team ha raggiunto la dimensione massima.
    - Restituisce *FALSE* altrimenti.
- **Boolean isVuoto()** :
    - Restituisce *TRUE* se il team non ha membri.
    - Restituisce *FALSE* altrimenti.
- **Aggiornamento CreaAggiornamento(nome: String, documento: String)** : Crea e restituisce un nuovo oggetto Aggiornamento associato al team.
- **EspelliMembro(utente: Utente)** : Rimuove un utente dalla lista dei componenti.
- **StampaVoti()** : Stampa a video i voti ricevuti dai giudici, con eventuale media.

RELAZIONI CON ALTRE CLASSI:

- **Hackathon**:
  - Un team partecipa a **un solo hackathon**.
- **Aggiornamento**:
  - Un team può creare **più aggiornamenti**.
- **Utente**:
  - Un team può includere **più utenti**.
- **Giudice**:
  - Un team può essere valutato da **più giudici**.

2.3 | CLASSE AGGIORNAMENTO

La classe Aggiornamento rappresenta una modifica o un avanzamento inviato da un team durante un hackathon, con attributi per tracciare il contenuto e i commenti dei giudici.

ATTRIBUTI:

- **Nome**: Identificativo dell'aggiornamento.
- **Documento**: File allegato dal team.
- **Commenti**: Lista di commenti/testi lasciati dai giudici.

La classe non presenta metodi al di fuori del costruttore.

RELAZIONI CON ALTRE CLASSI:

- **Team**:
    - Un aggiornamento è associato a **un solo team**.
- **Giudice**:
    - Un aggiornamento può ricevere commenti da **N giudici**.

2.4 | CLASSE UTENTE

La classe Utente rappresenta un partecipante generico alla piattaforma, con attributi per l'identificazione e metodi base per l'autenticazione e la gestione del profilo.

ATTRIBUTI:

- **Nome**
- **Cognome**
- **E-mail** (univoca, usata per l'account)
- **Ruolo** (valore di default: "Partecipante")
- **Username** (identificativo per il login)
- **Password**

METODI:

- **Boolean Login(username: String, password: String)** : Verifica le credenziali e restituisce:
    - *TRUE* se l'autenticazione è corretta.
    - *FALSE* se fallisce (username/password errati).
- **String GetNome()** : Restituisce il nome dell'utente.
- **String GetCognome()** : Restituisce il cognome dell'utente.

RELAZIONI CON ALTRE CLASSI:

- **Hackathon:**
    - Un utente può partecipare a **un solo hackathon** alla volta.
- **Team**:
    - Un utente può far parte di **un solo team**.

GERARCHIA DI EREDITARIETÀ:

La classe Utente è **classe padre** per:

- Giudice
- Organizzatore

2.5 | CLASSE GIUDICE

La classe Giudice estende la classe Utente senza aggiungere attributi propri, ma implementa metodi specifici per la valutazione dei team, la gestione degli aggiornamenti e la pubblicazione delle tracce dell'hackathon.

ATTRIBUTI:

- **Eredita tutti gli attributi da** Utente : (nome, cognome, e-mail, ruolo, username, password).

METODI:

- **ValutaTeam(team: Team, punteggio: int)**:
  - Aggiunge un voto alla lista dei voti del Team specificato.
- **String CommentaAggiornamento(a: Aggiornamento, commento: String)**:
  - Aggiunge un commento all'Aggiornamento del team.
- **PubblicaProblema(problema: String, hackathon: Hackathon)**:
  - Imposta il problema da risolvere nell'hackathon.

RELAZIONI CON ALTRE CLASSI:

- **Aggiornamento**:
  - Un giudice può commentare **N aggiornamenti**.
- **Team**:
  - Un giudice può valutare **N team**.
- **Hackathon**:
  - Un giudice partecipa a **un solo hackathon**.

2.6 | CLASSE ORGANIZZATORE

La classe Organizzatore estende Utente senza attributi aggiuntivi, ma aggiunge metodi per gestire gli hackathon e promuovere utenti a giudici.

ATTRIBUTI:

- **Eredita tutti gli attributi da** Utente: (nome, cognome, e-mail, ruolo , username, password), l'unica modifica che effettueremo è modificare il ruolo, dandogli il ruolo di organizzatore.

METODI:

- **Giudice AggiungiGiudice(utente: Utente, partecipanti: List<Utente>, hackathon: Hackathon)**:
    - Promuove un utente a giudice per un hackathon specifico, aggiornando a "giudice" il ruolo di un partecipante e aggiungendolo alla lista dei giudici

RELAZIONI CON ALTRE CLASSI:

- **Hackathon**:
    - Un organizzatore gestisce **un solo hackathon** alla volta.

2.7 | CLASSE VOTO (Classe associativa)

La classe Voto rappresenta la valutazione assegnata da un giudice a un team durante un hackathon, fungendo da **classe associativa** tra Giudice e Team.

ATTRIBUTI:

- **Valutazione**: int (da 1 a 10).
- **Giudice**: Riferimento all'oggetto Giudice che ha assegnato il voto.
- **Team**: Riferimento all'oggetto Team valutato.

METODI:

- **int GetValutazione()** : Restituisce il punteggio assegnato.
- **Giudice GetGiudice()** : Restituisce il giudice autore della valutazione.
- **Team GetTeam()** : Restituisce il team valutato.

RUOLO NEL DIAGRAMMA UML:

La classe Voto è una **classe associativa** che collega Giudice e Team, arricchendo la relazione con attributi aggiuntivi (il punteggio).