**PROPOSTA PROGETTO BASI DI DATI**

**Password: sapofede2003 codice 2576**

**COME USARE GITHUB?**

1. \*\*Creazione di un repository:\*\* //FATTA

- Il primo passo è creare un repository su GitHub. Puoi farlo cliccando sul pulsante "New" nella pagina principale del tuo account GitHub.

- Assegna un nome significativo al repository e scegli se renderlo pubblico (accessibile a tutti) o privato (accessibile solo a te e agli utenti che inviti).

2. \*\*Aggiungere il collaboratore:\*\* //FATTA

- Una volta creato il repository, puoi aggiungere il tuo compagno come collaboratore. Vai nelle impostazioni del repository, seleziona "Collaborators" e invita il tuo compagno utilizzando il suo username GitHub o il suo indirizzo email associato al suo account GitHub.

3. \*\*Clonare il repository:\*\*

- Ora tu e il tuo compagno dovreste clonare il repository sul vostro computer locale. Utilizza il comando `git clone` seguito dall'URL del repository che trovi nella pagina del repository su GitHub.

4. \*\*Lavorare sui file:\*\*

- Ogni volta che modifichi i file, è importante aggiungere, committare e pushare le modifiche al repository remoto. Utilizza i comandi `git add`, `git commit` e `git push` per fare ciò.

5. \*\*Gestione dei rami (branches):\*\*

- È una pratica comune lavorare su rami separati per sviluppare nuove funzionalità o risolvere problemi. Puoi creare un nuovo ramo usando `git checkout -b nome-del-ramo` e spostarti tra i rami con `git checkout nome-del-ramo`.

- Quando hai completato le modifiche sul tuo ramo, puoi fare un pull request (PR) per unire le modifiche al ramo principale del repository.

6. \*\*Gestione dei conflitti:\*\*

- Durante il processo di fusione (merge) dei rami o dei pull request, potrebbero verificarsi conflitti se entrambi i collaboratori hanno modificato lo stesso file nello stesso punto. GitHub fornisce strumenti per risolvere questi conflitti direttamente sulla piattaforma.

7. \*\*Revisione del codice:\*\*

- GitHub offre strumenti per la revisione del codice tramite i pull request. Dopo aver creato un pull request, il tuo compagno (e altri collaboratori) può esaminare le tue modifiche, lasciare commenti e approvare il pull request prima che venga effettuata la fusione.

8. \*\*Aggiornamento del repository locale:\*\*

- Periodicamente, assicurati di aggiornare il tuo repository locale con le modifiche fatte da altri collaboratori sul repository remoto. Puoi farlo utilizzando il comando `git pull`.

9. \*\*Comunicazione:\*\*

- Assicuratevi di comunicare regolarmente tramite chat, email o altri mezzi per coordinare il lavoro e risolvere eventuali problemi o dubbi.

Seguendo questi passaggi, potrai collaborare efficacemente con il tuo compagno utilizzando GitHub per gestire il vostro progetto condiviso.

L’obiettivo del progetto è realizzare un sistema di supporto ad un ente che organizza campionati sportivi in cui militano più squadre di diversi sport(calcio,pallacanestro,pallavolo).

Ogni squadra può iscriversi al campionato inserendo le proprie informazioni: ogni **squadra** è posseduta da un **presidente**, il quale si affida ad un **direttore sportivo**. Questo, coi fondi fornitogli dal patrimonio del presidente e dai suoi **sponsor,** si occupa di acquistare tramite il mercato sportivo un **allenatore**, più **giocatori** e qualche elemento dello **staff** (richiestogli dall'allenatore stesso).

Ogni squadra potrà affrontare le altre, iscritte allo stesso campionato, in varie **partite** (squadra casa vs squadra ospite) le quali verranno arbitrate da un **arbitro** tra quelli del campionato.

Queste partite si svolgeranno in uno **stadio**, il quale richiede un affitto per poter essere usato, e produrranno un **tabellino** con le statistiche riguardanti la partita stessa.

Il campionato verrà sponsorizzato da varie **aziende** e potrà essere trasmesso su diversi **canali**, i quali porteranno fondi per l’organizzazione assieme alla vendita dei **biglietti** per vedere la partita.

Appena ci si iscrive al campionato, alla squadra viene assegnata una quota N di punti. Ogni sconfitta apporta alla squadra -M punti e ogni vittoria +M punti. Una partita non può concludersi con un pareggio. Superata negativamente una quota di L punti si viene retrocessi e quindi esclusi dal campionato. All' arrivo di Q (Q>N) punti si vince il campionato e si ottiene un premio in denaro. Le variabili N,M,L,Q sono relative ai vari sport (calcio,pallacanestro,pallavolo) e quindi non hanno lo stesso valore per sport diversi.

Le funzionalità offerte dalla piattaforma agli utenti sono le seguenti:

1.Iscrizione al campionato della propria squadra.

2.Proporre scambi di giocatori tra i propri giocatori e quelli delle altre squadre:

* + Registrazione degli scambi e dei trasferimenti tra le squadre.

3.Registrare gli infortuni dei giocatori durante le partite o gli allenamenti, con relativa durata prevista di assenza e stato di recupero.

4.Gestione degli sponsor delle squadre, registrando i contratti di sponsorizzazione e le relative condizioni.

Inoltre l’utente potrà consultare informazioni aggregate come:

1.Visualizzare i giocatori/squadre con le statistiche migliori del campionato.

2.Visualizzazione della classifica del campionato in tempo reale.

3.Visualizzazione dei giocatori infortunati.

4.Visualizzazione del proprio registro di transazioni finanziarie: per acquisti di giocatori, allenatori, staff e altre spese.

5. Visualizzazione del calendario del campionato in cui è iscritta la propria squadra.

Infine l’amministratore del sito potrà:

1.Inserire o togliere giocatori dalle squadre in caso di trasferimento.

2.Togliere dal campionato le squadre con meno di L punti in caso di retrocessione.

3.Bloccare/sbloccare un calciatore espulso in una partita.

4.Gestire le spese.

5.Visualizzare un resoconto dei ricavi dai vari campionati (il calcolo avverrà automaticamente).

6.Organizzazione di partite tra le squadre:

* + Registrazione dei dettagli delle partite (eventi, goal, ammonizioni, espulsioni).
  + Calcolo automatico dei punti di ogni squadra per ogni partita giocata.