StudentHub
Piattaforma per la Gestione della Carriera Universitaria

Diego Andruccioli, Rei Mici, Giovanni Morelli

Anno Accademico 2025/2026

Sommario

StudentHub è una piattaforma web progettata per la gestione della carriera universitaria, sviluppata nell'ambito del corso di Ingegneria dei Sistemi Web presso l'Alma Mater Studiorum - Università di Bologna. L'obiettivo principale del progetto è trasformare la classica gestione del libretto universitario in un'esperienza interattiva e coinvolgente, integrando elementi di Gamification e visualizzazione dati avanzata.

Il sistema adotta un'architettura disaccoppiata (Client-Server), garantendo scalabilità e mantenibilità del codice.

Indice

1 Introduzione

StudentHub è una piattaforma web progettata per la gestione della carriera universitaria. L'obiettivo principale del progetto è trasformare la classica gestione del libretto universitario in un'esperienza interattiva e coinvolgente, integrando elementi di Gamification e visualizzazione dati avanzata.

Il sistema adotta un'architettura disaccoppiata (Client-Server), garantendo scalabilità e mantenibilità del codice.

2 Features

In questo capitolo vengono descritte le funzionalità principali della piattaforma, suddivise per tipologia di utente.

2.1 Features Utenti

Gli utenti standard (Studenti) hanno accesso alle seguenti funzionalità:

- **Gestione Carriera**: Possibilità di aggiungere, modificare ed eliminare gli esami sostenuti, specificando voto, crediti (CFU) e data.
- **Dashboard Statistica**: Visualizzazione di grafici interattivi che mostrano l'andamento della media nel tempo, la distribuzione dei voti e la proiezione del voto di laurea.
- **Gamification**: Sistema di progressione basato su Punti Esperienza (XP). Ogni esame superato conferisce XP in base al voto e ai crediti.
- **Badge e Obiettivi**: Sblocco automatico di riconoscimenti (badge) al raggiungimento di determinati traguardi (es. "Primo 30", "Media del 28").
- **Classifica (Leaderboard)**: Possibilità di confrontare il proprio punteggio XP con quello degli altri studenti registrati alla piattaforma.
- **Impostazioni Personalizzate**: Gestione delle preferenze di visualizzazione, inclusa la modifica delle soglie di colore per i voti.

2.2 Features Admin e SuperAdmin

Il sistema prevede ruoli amministrativi con privilegi elevati:

Admin

- Accesso alla lista completa degli utenti registrati.
- **Gestione scalabile**: Sistema di paginazione server-side per gestire migliaia di utenti senza rallentamenti.
- Visualizzazione di statistiche globali della piattaforma (es. numero totale di esami registrati).
- Accesso a una classifica globale non anonimizzata.

SuperAdmin

- Comprende tutte le funzionalità dell'Admin.
- Gestione dei ruoli utente (promozione e retrocessione).
- Eliminazione di account amministrativi.

3 Tecnologie Utilizzate

3.1 Frontend

Il client è sviluppato utilizzando le seguenti tecnologie:

- **Vue.js** (v3.5.24): Framework progressivo per interfacce utente.
- **Pinia** (v3.0.4): State Management ufficiale e modulare per Vue.js.
- **Vite** (v7.2.4): Build tool di nuova generazione per frontend rapidi.
- **Tailwind CSS** (v4.1.18): Framework CSS utility-first per lo styling.
- **TypeScript (v5.9.3)**: Logica applicativa fortemente tipizzata per garantire sicurezza e scalabilità.

3.2 Backend

Il server è sviluppato utilizzando:

- **Node.js**: Runtime JavaScript lato server.
- **Express.js** (v4.19.2): Web framework per Node.js.
- **TypeScript** (v5.4.5): Superset tipizzato di JavaScript per una maggiore robustezza del codice.

3.3 Database

- **MySQL**: Database relazionale per la persistenza dei dati.

3.4 Containerizzazione

- **Docker & Docker Compose**: Per l'orchestrazione dell'ambiente di sviluppo.

4 Pacchetti Installati

Di seguito vengono elencati i principali pacchetti di terze parti utilizzati.

4.1 Frontend

Installazione dipendenze: `npm install`

`axios` **(^1.13.2)**
: Libreria per effettuare richieste HTTP (API Client) verso il backend in modo centralizzato.

`pinia` **(^3.0.4)**
: Store manager ufficiale per Vue.js, utilizzato per la gestione dello stato globale.

`vue-router` **(^4.6.3)**
: Gestore ufficiale del routing per Vue.js, permette la navigazione tra le pagine (SPA).

`chart.js` **e** `vue-chartjs`
: Librerie per la creazione di grafici statistici interattivi e responsivi.

`tailwindcss`
: Motore per la generazione dei fogli di stile utility-first.

4.2 Backend

Installazione dipendenze: `npm install`

`express` **(^4.19.2)**
: Core framework per la creazione del server web e la gestione delle rotte API.

`mysql2` **(^3.9.7)**
: Driver client ottimizzato per la connessione e l'interazione con il database MySQL.

`bcrypt` **(^5.1.1)**
: Libreria per l'hashing sicuro delle password.

`jsonwebtoken` **(^9.0.2)**
: Strumento per la generazione e validazione dei token JWT.

`cors` **(^2.8.5)**
: Middleware per abilitare e configurare il Cross-Origin Resource Sharing.

`dotenv` **(^16.4.5)**

Modulo per caricare le variabili d'ambiente da un file `.env`.

5 Struttura del Progetto

Di seguito è riportata l'organizzazione delle directory del progetto.

```
StudentHub/
|-- backend/                    # Logica Server
|   |-- src/
|   |   |-- config/             # Configurazioni DB e variabili ambiente
|   |   |-- controllers/        # Gestione richieste HTTP
|   |   |-- middleware/         # Controlli intermedi (es. Autenticazione)
|   |   |-- routes/             # Definizione degli endpoint API
|   |   |-- services/           # Logica di business e accesso ai dati
|   |   |-- types/              # Definizioni dei tipi TypeScript
|   |   |-- utils/              # Funzioni di utilità
|   |-- sql/                    # Script di inizializzazione DB
|   |-- server.ts               # Entry point backend
|
|-- frontend/                   # Interfaccia Utente
|   |-- src/
|   |   |-- api/                # Configurazione Client HTTP (Axios)
|   |   |-- assets/             # Risorse statiche
|   |   |-- components/         # Componenti Vue riutilizzabili
|   |   |-- pages/              # Viste principali
|   |   |-- router/             # Configurazione rotte
|   |   |-- stores/             # Gestione stato (Pinia)
|   |   |-- App.vue             # Componente Root
|   |   |-- main.ts             # Entry point frontend
|
|-- docker-compose.yml          # Configurazione Container
+-- README.md                   # Documentazione progetto
```

6 Installazione

È possibile avviare il progetto tramite Docker (consigliato per compatibilità universale) o manualmente clonando la repository.

Repository GitHub: https://github.com/diegoandruccioli/StudentHub.git

6.1 Metodo 1: Docker (Windows, Mac, Linux)

Requisiti: Docker Desktop installato.

1. Aprire il terminale nella cartella principale del progetto.
2. Eseguire il comando:

   ```
   docker compose up --build
   ```

3. Il sistema sarà accessibile ai seguenti indirizzi:
   - Frontend: http://localhost:5173
   - Backend: http://localhost:3000

6.2 Metodo 2: Installazione Manuale

Prerequisiti: Node.js v18+, MySQL Server in esecuzione.

1. **Configurazione Database**: Eseguire gli script in `backend/sql/` (`init.sql` e `seed.sql`).
2. **Setup Backend**:

   ```
   cd backend
   npm install
   cp .env.example .env # Modificare con credenziali DB
   npm run dev
   ```

3. **Setup Frontend**:

   ```
   cd frontend
   npm install
   npm run dev
   ```

7 API Endpoints

Il backend espone le seguenti API RESTful.

AUTH

- `POST /api/auth/register`: Registrazione nuovo utente.
- `POST /api/auth/login`: Autenticazione utente.
- `POST /api/auth/logout`: Disconnessione.

EXAMS

- `GET /api/exams`: Lista esami dell'utente loggato (filtri supportati).
- `POST /api/exams`: Aggiunta di nuovi esami.
- `PUT /api/exams/:id`: Modifica di un esame esistente.
- `DELETE /api/exams/:id`: Rimozione di un esame.

STATS & USERS

- `GET /api/stats`: Statistiche aggregate (media, base laurea, proiezioni).
- `GET /api/users/leaderboard`: Classifica globale utenti.

GAMIFICATION

- `GET /api/gamification/status`: Livello e progressi XP correnti.
- `GET /api/gamification/badges`: Catalogo obiettivi disponibili.
- `GET /api/gamification/my-badges`: Obiettivi sbloccati dall'utente.

SETTINGS

- `GET /api/settings`: Recupero preferenze utente.
- `PUT /api/settings`: Aggiornamento preferenze.

ADMIN

- `GET /api/admin/users`: Lista utenti paginata. Restituisce anche statistiche globali.
- `PUT /api/admin/users/:id/role`: Modifica ruolo utente.
- `DELETE /api/admin/users/:id`: Eliminazione admin.

8 Database

Il sistema utilizza un database relazionale MySQL composto dalle seguenti entità:

- **Utenti**: Memorizza credenziali, dati anagrafici e progressi di gamification (XP).
- **Esami**: Memorizza i singoli esami sostenuti, collegati agli utenti. Include vincoli di integrità per i voti.
- **Livelli**: Tabella di configurazione per le soglie di livello basate sugli XP.
- **Obiettivi**: Tabella di configurazione dei badge ottenibili.
- **Obiettivi_Sbloccati**: Tabella di associazione che traccia i badge ottenuti dagli utenti.
- **Impostazioni_Utente**: Memorizza le preferenze di visualizzazione per ogni utente.

9 Accessibilità

Il progetto pone attenzione all'accessibilità visiva attraverso l'uso di token semantici definiti nel design system. Non vengono utilizzati colori hardcoded, ma variabili CSS (es. `-color-primary`, `-color-accent`) che garantiscono coerenza in tutta l'applicazione. L'utente ha inoltre la possibilità di personalizzare la visualizzazione dei voti tramite le impostazioni, scegliendo tra temi diversi o adattando le soglie cromatiche (es. "Semaforo") alle proprie necessità visive.

10 Documentazione e Mockup

La progettazione dell'interfaccia ha seguito un approccio *Mobile-First*, con un focus sulla chiarezza dei dati accademici.

10.1 Wireframe e Flusso Utente

Abbiamo progettato le seguenti viste principali:

1. **Login/Register**: Design minimale con validazione in tempo reale.
2. **Dashboard Studente**: Visualizzazione a "Card" per i dati critici.
3. **Carriera**: Tabella responsive con filtri dinamici.
4. **Gamification**: Barre di progresso animate e Badge colorati.

10.2 Risorse Grafiche

- **Mockup Iniziale (Concept)**: Visualizza su Canva
- **Mockup Finale (Prodotto)**: Visualizza su Canva