UNIVERSITÀ DEGLI STUDI DI NAPOLI

PARTHENOPE

DIPARTIMENTO DI INGEGNERIA

CORSO DI LAUREA IN INGEGNERIA E SCIENZE INFORMATICHE PER LA CYBERSECURITY

ANNO ACCADEMICO 2024/2025

ALUNNO

SIMONE D'ANNA, 0334000158

Tema Progetto

Traccia d’Esame – Progetto di Sistema Informativo per la Gestione di un’Associazione (AssoHub)

Il progetto *AssoHUB* nasce dall’esigenza reale di digitalizzare e semplificare la gestione interna di un’associazione, sostituendo le procedure manuali e la frammentazione dei dati con un sistema informativo moderno, sicuro e accessibile via web.
L’obiettivo è fornire uno strumento completo che consenta al direttivo di coordinare in modo organizzato iscritti, eventi e movimenti economici, garantendo al contempo una fruizione chiara e intuitiva per i soci e una maggiore trasparenza nelle attività associative.

Struttura del sistema

Al centro dell’applicazione si trova un **database SQL relazionale**, progettato per ospitare tutte le informazioni fondamentali al funzionamento dell’associazione. La base dati è strutturata in modo da garantire integrità, coerenza e tracciabilità, consentendo query efficienti e una gestione scalabile dei dati.

Ogni utente registrato dispone di un **account personale con credenziali univoche**, ed è inquadrato in una delle due categorie principali:

- **Associati**: rappresentano la base sociale dell’associazione. Il sistema gestisce le loro informazioni anagrafiche, i contatti, lo stato della tessera associativa e i versamenti delle quote, includendo importo, data, anno di riferimento e stato del pagamento. Gli associati hanno accesso alla homepage pubblica con gli eventi futuri e possono iscriversi alle attività organizzate, con la possibilità di consultare lo storico delle proprie partecipazioni.
- **Amministratori**: costituiscono il direttivo e dispongono di funzionalità gestionali avanzate. Possono pianificare eventi, registrare e monitorare i movimenti economici, gestire l’elenco degli iscritti e supervisionare l’andamento generale dell’associazione. Ogni movimento economico (entrata o uscita) può essere collegato a un evento specifico o registrato come operazione indipendente, favorendo una contabilità interna chiara e completa.

Funzionalità principali

Il sistema garantisce una serie di servizi centrali:

- **Gestione centralizzata degli utenti** con autenticazione e differenziazione dei privilegi.
- **Amministrazione delle quote associative** con monitoraggio puntuale dei pagamenti, scadenze e rinnovi annuali.
- **Organizzazione e pianificazione degli eventi**, con gestione degli iscritti, tracciamento delle presenze e reportistica delle attività.
- **Contabilità integrata**, che consente la registrazione e la consultazione di entrate e uscite, distinguendo sponsorizzazioni, donazioni, affitti, spese logistiche e altre voci.
- **Dashboard riservata al direttivo**, che fornisce una visione di sintesi dell’andamento economico e organizzativo dell’associazione.

Valore aggiunto del progetto

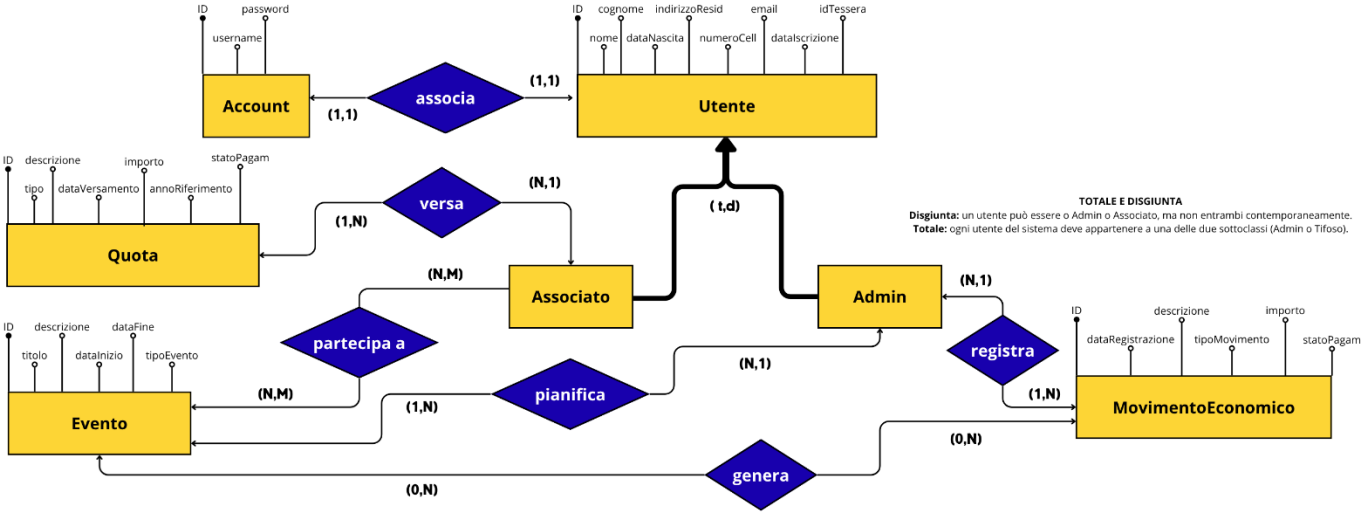
Il progetto si distingue per la sua **forte aderenza a un'esigenza reale**: supportare concretamente la vita quotidiana di un'associazione, migliorando **trasparenza, tracciabilità e organizzazione**. Rispetto a una gestione tradizionale cartacea o basata su file non centralizzati, *AssoHUB*:

- riduce gli errori umani e la perdita di informazioni,
- aumenta la **trasparenza** verso i soci,
- facilita il lavoro del direttivo,
- offre una piattaforma digitale accessibile anche da remoto.

Inoltre, l'interfaccia web è progettata con un approccio **user-friendly**, per permettere anche a utenti non esperti di accedere facilmente alle informazioni e partecipare attivamente alla vita associativa.

Obiettivi del progetto

1. **Analisi e progettazione concettuale**
   A partire dalla descrizione del contesto associativo, è stato analizzato il dominio della gestione di un'associazione e costruito un modello entità-relazione completo, includendo iscritti, quote associative, eventi, movimenti economici e ruoli utente. Sono state gestite le relazioni di appartenenza e partecipazione, motivando le scelte effettuate per garantire coerenza, integrità e tracciabilità dei dati.



2. **Progettazione logica**
   Il modello concettuale è stato tradotto in un modello logico relazionale, con la definizione delle tabelle, degli attributi e delle chiavi primarie/esterne.
   Particolare attenzione è stata posta nella normalizzazione, così da evitare ridondanze e assicurare una corretta gestione delle relazioni tra soci, pagamenti, eventi e movimenti contabili.

   Iscritto(id_iscritto **PK**, nome, cognome, email **UNIQUE**, telefono, ruolo)

   QuotaAssociativa(id_quota **PK**, anno, importo, data_pagamento, stato, id_iscritto **FK → Iscritto**)

   Vincolo unico: (anno, id_iscritto)

   Evento(id_evento **PK**, titolo, descrizione, data, luogo)

   Partecipazione(id_iscritto **FK → Iscritto**, id_evento **FK → Evento**, presenza)

   Vincolo unico: (id_iscritto, id_evento)

   MovimentoEconomico(id_movimento **PK**, tipo, importo, data, descrizione, id_evento **FK →**

   Evento NULL)

   Utente(id_utente **PK**, username **UNIQUE**, password_hash, ruolo, id_iscritto **FK → Iscritto**)

3. **Progettazione fisica**
   Lo schema logico è stato implementato in SQL, ottimizzando la struttura delle tabelle per supportare le funzionalità del sistema. Sono stati definiti indici per velocizzare le ricerche, vincoli per garantire l'integrità referenziale e procedure per il controllo degli accessi in base al ruolo (Associato o Amministratore).

4. **Implementazione applicativa**
   Realizzare un'applicazione web **Django** che gestisca i dati modellati (iscritti, eventi, quote, movimenti economici) e offra **almeno quattro funzionalità** chiave per il direttivo e per gli associati, rispettando i vincoli:
   - **Backend:** Django
   - **View/UI: Django Template System**
   - **Stile: Bootstrap CSS**
   - **JavaScript:** solo dove strettamente indispensabile (es. conferme modal, datepicker, validazioni minime)

Funzionalità implementate (selezione minima garantita: 4)

1. **Registrazione e autenticazione utenti**
   - Registrazione associati (signup), login/logout, recupero password.
   - Ruoli: **Associato** (base) e **Amministratore** (direttivo).
   - Pagina profilo con dati anagrafici e stato tessera.
   - **Access control** con login_required e PermissionRequiredMixin.
2. **Gestione eventi e partecipazioni**
   - CRUD eventi (titolo, descrizione, luogo, data/ora, capienza).
   - Iscrizione/cancellazione associato; tracciamento presenze.
   - **Elenco pubblico** degli eventi futuri (homepage) + **dettagli evento**.
   - (Opz.) esportazione CSV degli iscritti all'evento.
3. **Gestione quote associative (pagamenti)**
   - Registrazione versamenti (importo, data, anno, stato).
   - Storico quote per associato; filtro per anno/stato (pagato, pending, scaduto).
   - Riepilogo di cassa delle quote incassate per periodo.
4. **Movimenti economici interni (entrate/uscite)**
   - CRUD movimenti (categoria, importo, data, descrizione), con **opzionale** collegamento a un evento.
   - **Dashboard direttivo**: totale entrate/uscite, saldo, classifica per categoria (tabelle/indicatori).
   - Filtri per intervallo date e categorie (donazioni, sponsorizzazioni, affitti, spese).
5. **Area Associato (personale)** *(opzionale se già coperte le 4 precedenti)*
   - Panoramica: prossimi eventi, mie iscrizioni, stato quote, dati profilo.
   - (Opz.) preferenze notifiche / newsletter.

Le 4 funzionalità minime consigliate: **(1) Autenticazione**, **(2) Eventi**, **(3) Quote**, **(4) Movimenti economici**. L'**Area Associato** può essere il quinto modulo opzionale.

Consegna

Il progetto deve essere caricato su un repository **GitHub privato**. Il link al repository va condiviso via e-mail con il docente **almeno 7 giorni prima della data dell'esame**.

Il repository deve contenere:

- Documentazione con il modello informativo e le scelte progettuali.
- Codice sorgente completo e funzionante.
- Dati di esempio sufficienti per mostrare il funzionamento del sistema (**dump del database**).
- Istruzioni per installazione e avvio del progetto.

Bonus (facoltativo)

È possibile includere una **simulazione di attacco** al sistema, ad esempio:

- SQL injection, • attacco a dizionario,
- attacco brute-force.

Lo scopo è mostrare come vulnerabilità comuni possono essere sfruttate e come è possibile prevenirle con misure appropriate.