Università degli Studi di Padova
Corso di Laurea in Informatica
A. A. 2025/26

Relazione

Progetto di Tecnologie Web

Studenti	Sofia De Blasi - matricola - email Nenad Radulovic - matricola - email Giacomo Speggiorin - matricola - email Nicola Simionato - matricola - email
Indirizzo del sito	indirizzo
Utenti	**Username:** user, **Password:** user **Username:** admin, **Password:** admin

Indice

1 Introduzione

1.1 Com'è nata l'idea

L'idea di questo progetto nasce da una domanda che ci siamo posti più volte nella nostra quotidianità: *"Cosa potremmo fare questo weekend?"*. Nelle città universitarie vengono spesso organizzati eventi e attività dedicate agli studenti, è però seccante doverne andare ogni volta alla ricerca. Bisogna sapere innanzitutto cosa si sta cercando (il che non è una cosa scontata), e serve poi andarsi a sfogliare di volta in volta le pagine social degli organizzatori, che sono sempre diversi.

Abbiamo quindi pensato a quanto sarebbe comodo e utile se fossero gli organizzatori stessi ad andare incontro agli studenti, e a quanto si semplificherebbe la ricerca se esistesse un'unica bacheca dove tutti gli organizzatori possano appendere i loro volantini.

Nel mentre buttavamo giù le idee il discorso si è ampliato, e ci siamo resi conto che la fuori ci sono un sacco di servizi di cui gli studenti hanno bisogno ma non hanno un punto di riferimento sul dove andarli a cercare.

1.2 L'obbiettivo di StudentSpace

StudentSpace propone una *bacheca online* che raccolga in un unico luogo eventi, attività e servizi di interesse per gli studenti, andando oltre il contesto strettamente accademico.

Attraverso la piattaforma sarà possibile trovare eventi sportivi, attività serali, concerti, e altre occasioni di socialità. Potranno inoltre dare la loro disponibilità ad altri studenti per ripetizioni, oppure consultare annunci di Affitti relativi alla propria città.

L'obbiettivo principale è garantire a studenti e studentesse un accesso rapido e completo a ciò che potrebbe interessarli. Allo stesso tempo, si vuole agevolare gli organizzatori, che si rivolgono a questo target, fornendo loro un canale unico e mirato per raggiungere un pubblico ampio e specifico.

2 Analisi del lavoro svolto

2.1 Processo decisionale

Durante tutto lo sviluppo del progetto ci si è trovati a dover prendere diverse decisioni in merito a design, funzionalità, contenuto ecc. Nella maggior parte dei casi questi punti suono stati discussi in presenza, tra tutti i membri del gruppo. Ovviamente, questo metodo comporta delle chiare limitazioni, si è quindi deciso di adottare anche dei metodi di comunicazione (sia sincroni che asincroni) alternativi.

Per quanto riguarda le comunicazioni asincrone, è stata usata ampiamente la piattaforma di messaggistica **??**. Si è deciso inoltre di adottare **??** come strumento di comunicazione asincrona, in supporto alle riunioni in presenza.

2.2 Modalità di lavoro

Per adattarsi meglio alle esigenze del gruppo si è deciso che i membri del gruppo avrebbero svolto ciascuno le proprie attività in maniera prevalentemente asincrona. Questa decisione è stata presa per riuscire a dare più libertà ai singoli membri nell'organizzare il proprio lavoro. La scelta è stata inoltre supportata dalla volontà del gruppo di dare maggiore fluidità allo sviluppo del progetto. Il lavoro è stato infatti suddiviso in modo che ogni membro andasse a lavorare su punti diversi rispetto agli altri, cosicché nessuno dovesse preoccuparsi che il proprio lavoro andasse in conflitto con quello di qualcun altro. A sostegno di ciò si è deciso di implementare un sistema di versionamento del codice (con annesso hosting online) tramite **Git** e **Github**.

2.3 Divisione del lavoro

Visto lo scopo di natura accademica del progetto si è ritenuto opportuno che ciascun membro del gruppo fosse coinvolto (quantomeno in parte) in ogni ambito del progetto. Questo ha permesso a ogni membro di avere un idea generale sul funzionamento del programma, e di mettere mano direttamente a ogni fase del suo sviluppo.

3 Progettazione

3.1 Tipologie di Utenti

Il sistema prevede due tipologie di utente:

- **Utente non registrato:** utente che esplora le categorie e legge annunci
  - **Azioni e permessi:** visualizzazione completa delle varie pagine, navigazione tra categorie e annunci, ricerca e filtraggio, apertura dettaglio annuncio, avvio della procedura di registrazione/login
  - **Limitazioni:** accesso negato alle funzioni di pubblicazione e gestione degli annunci → inviti chiari a registrarsi quando tenta azioni riservate.
- **Utente registrato:** utente che oltre alle funzionalotà che erdità dall'utente non registrato può pubblicare annunci e gestirli.
  - **Azioni e permessi:** creazione, modifica ed eliminazione dei propri annunci; gestione profilo.
  - **Limitazioni:** non può modificare annunci altrui; vincoli su campi obbligatori e formati (validazione client/server).

3.2 Analisi dei contenuti

Il sito web si propone come una bacheca universitaria online, in cui gli utenti possono pubblicare e consultare annunci suddivisi in quattro categorie principali: **Affitti**, **Esperimenti**, **Eventi** e **Ripetizioni**. L'analisi dei contenuti si concentra sui tipi di annunci, sui campi comuni a tutte le categorie e sui campi specifici per ciascuna categoria.

3.2.1 Tipi di annunci

- **Affitti**: annunci relativi a stanze, appartamenti o posti letto disponibili per studenti.
- **Esperimenti**: annunci riguardanti attività di laboratorio, progetti di ricerca o collaborazioni scientifiche.
- **Eventi**: annunci di conferenze, seminari, feste universitarie, attività culturali o sportive.
- **Ripetizioni**: annunci di studenti che offrono supporto didattico su materie specifiche.

3.2.2 Campi generici

Ogni annuncio, indipendentemente dalla categoria, contiene i seguenti campi:

- **Categoria**: selezionata dall'utente tra le quattro disponibili.
- **Titolo**: breve descrizione sintetica dell'annuncio.
- **Data di pubblicazione**: generata automaticamente dal sistema.
- **Città**: riferimento all'utente registrato che ha pubblicato l'annuncio.
- **Descrizione**: testo libero con i dettagli dell'annuncio.
- **Autore**: riferimento all'utente registrato che ha pubblicato l'annuncio.

3.2.3 Campi specifici per categoria

Oltre ai campi generici, ciascuna categoria prevede attributi specifici (3 per categoria):

- **Affitti**:
  - Prezzo mensile
  - Indirizzo
  - Numero coinquilini
- **Esperimenti**:
  - Laboratorio di riferimento
  - Durata prevista
  - Compenso
- **Eventi**:
  - Data Eventi
  - Luogo
  - Costo entrata
- **Ripetizioni**:
  - Materia
  - Livello (scuola superiore, università, ecc.)
  - Prezzo orario

3.3 Descrizione delle funzionalità

Il sistema offre una serie di funzionalità fondamentali per soddisfare i requisiti del progetto e le esigenze degli utenti.

3.3.1 Registrazione e login

- Registrazione di nuovi utenti: è permessa, per gli utenti che volessero ampliare la loro esperienza di utilizzo, la creazione di un account. La registrazione richiederà all'utente di fornire le seguenti informazioni:
  - Nome
  - Cognome
  - Email
  - Citta: intesa come città universitaria di riferimento.
  - Password
- Login: processo di identificazione degli utenti registrati che sblocca tutte le funzionalità.
- Qualora l'utente inserisca dei dati non validi verrà bloccato l'invio della form e verranno mostrati dei messaggi di errore (si veda la parte relativa alla validazione).

3.3.2 Pubblicazione annunci

- È possibile, per gli utenti registrati, pubblicare annunci compilando un'apposito form. Sono necessari per la pubblicazione:
  - Campi generici: titolo, descrizione e categoria.
  - Campi specifici mostrati dinamicamente in base alla categoria scelta.
- In seguito alla pubblicazione sarà possibile risalire all'autore dell'annuncio.
- Qualora l'utente inserisca dei dati non validi verrà bloccato l'invio della form e verranno mostrati dei messaggi di errore (si veda la parte relativa alla validazione).

3.3.3 Modifica, eliminazione e salvataggio degli annunci

- Un utente registrato può decidere di salvare degli annunci aggiungendoli alla propria lista dei preferiti.
- L'autore di un annuncio può decidere di modificare o cancellare i propri annunci.
- Un annuncio non apparirà più all'interno della lista dei preferiti degli utenti dopo essere stato eliminato dal sistema.

3.3.4 Filtraggio e ricerca annunci

- È possibile applicare dei filtri per migliorare la ricerca degli annunci. Sono previsti:
  - Filtri generici (categoria, data di pubblicazione).

  - Filtri specifici per categoria

- È inoltre possibile effettuare una ricerca testuale sul contenuto del titolo e della descrizione.

- Qualora vengano inseriti nei campi liberi dei valori compromettenti il sistema li ignorerà ma non verrà visualizzato alcun errore.

3.4 Requisiti tecnici

3.4.1 Accessibilità e responsività

- Layout realizzato con CSS3, Flexbox e Grid.

- Separazione netta tra contenuto (HTML5), presentazione (CSS) e comportamento (JS/PHP).

- Compatibilità cross-browser e adattamento a diverse dimensioni di schermo.

- Uso di etichette semantiche, contrasto cromatico adeguato e font leggibili.

- Validazione lato client (formato email, lunghezza password) e lato server.

3.5 Archietettura del sistema

3.6 Struttura del sito

3.7 Mockup

3.8 Responsive Web Design e Mobile First

4 Sviluppo

4.1 Database

4.1.1 Analisi dei requisiti del database e progettazione logica

La base di dati deve supportare la gestione degli utenti e degli annunci pubblicati sulla bacheca universitaria. In particolare, deve garantire la memorizzazione dei campi generici comuni a tutte le categorie di annunci e dei campi specifici per ciascuna categoria. Inoltre, deve consentire operazioni di inserimento.

Entità principali

- **Utente**: rappresenta gli utenti registrati alla piattaforma.
  `IdUtente(PK)`, `Nome`, `Cognome`, `Email (UNIQUE)`, `Password`, `IdCitta (FK)*`.

- **Città**: rappresenta la sede universitaria o città di riferimento di un utente o di un annuncio.
  `IdCitta(PK)`, `NomeCitta (UNIQUE)`.

- **Annuncio**: rappresenta un contenuto pubblicato sulla bacheca.
  `IdAnnuncio(PK)`, `Titolo`, `Descrizione`, `DataPubblicazione (DEFAULT CURRENT_TIMESTAMP)`, `IdUtente (FK)`, `NomeCategoria`, `IdCitta (FK)`.

- **Specializzazioni di Annuncio**: tabelle figlie che estendono Annuncio con 3 attributi specifici.

  - **Affitti**: `IdAnnuncio (PK, FK)`, `PrezzoMensile`, `Indirizzo`, `NumeroInquilini`.

  - **Esperimenti**: `IdAnnuncio (PK, FK)`, `Laboratorio`, `DurataPrevista`, `Compenso`.

  - **Event**: `IdAnnuncio (PK, FK)`, `DataEventi`, `Luogo`, `CostoEntrata`.

  - **Ripetizioni**: `IdAnnuncio (PK, FK)`, `Materia`, `Livello`, `PrezzoOrario`.

- **ImmaginiAnnuncio**: rappresenta le immagini associate ad un annuncio.
  `IdImmagine(PK)`, `IdAnnuncio (FK)`, `Percorso`, `AltText*`, `Decorativa`, `Ordine`.

Relazioni tra entità

- **Pubblicazione:** collega `Utente` e `Annuncio`.
  `Utente (1,1)` $\longrightarrow$ `Annuncio (0,N)`
  Ogni annuncio è pubblicato da un solo utente, mentre un utente può pubblicare più annunci.

- **Collocazione:** collega `Città` e `Annuncio`.
  `Città (1,1)` $\longrightarrow$ `Annuncio (0,N)`
  Ogni annuncio è associato ad una sola città, mentre una città può avere più annunci.

- **SedeUniversitaria:** collega `Città` e `Utente`.
  `Città (1,1)` $\longrightarrow$ `Utente (0,N)`
  Ogni utente è associato ad una sola città, mentre una città può avere più utenti.

- **Associazione immagini:** collega `Annuncio` e `ImmaginiAnnuncio`.
  `Annuncio (1,1)` $\longrightarrow$ `ImmaginiAnnuncio (0,N)`
  Ogni annuncio può avere da zero a più immagini, mentre ogni immagine è riferita ad un solo annuncio.

4.1.2 Schema ER

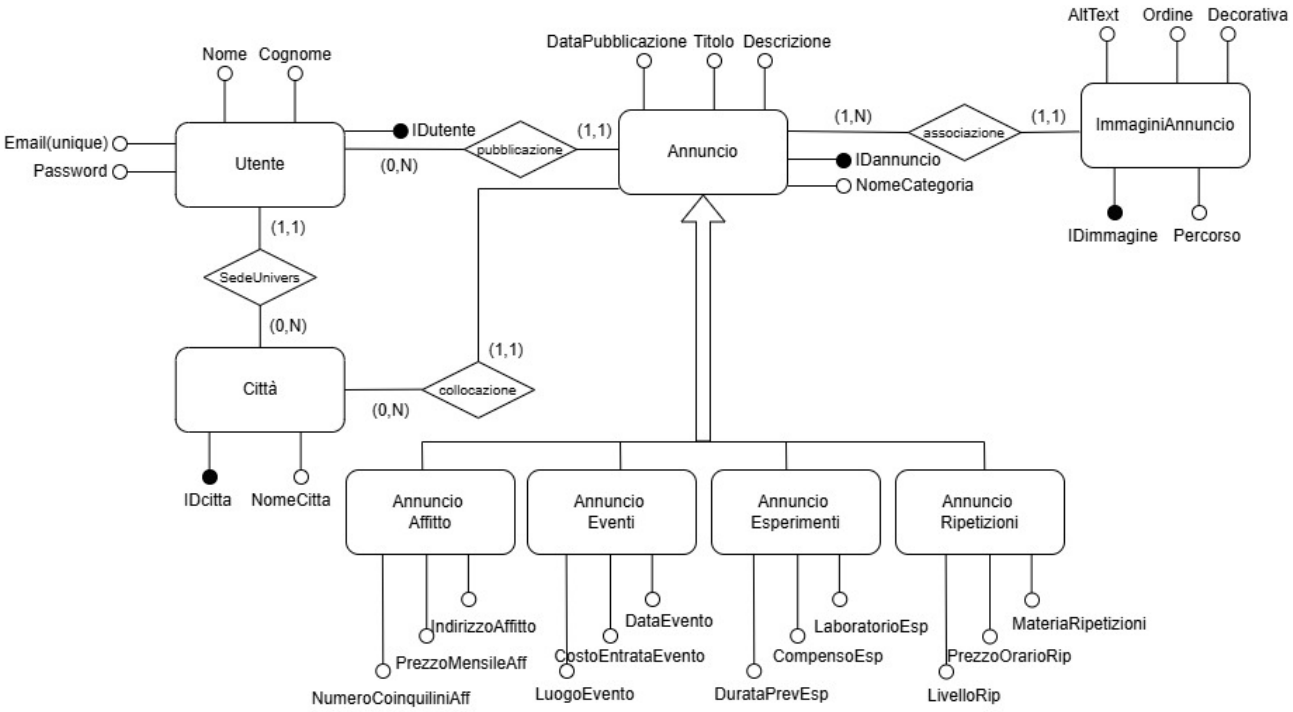

Figura 1: Schema ER

Generalizzazione IS-A Nel modello concettuale iniziale, l'entità `Annuncio` fungeva da padre nella generalizzazione per le quattro categorie specifiche figlie (`Affitti`, `Esperimenti`, `Eventi`, `Ripetizioni`). In fase di progettazione logica si è ristrutturato lo schema ER con una **sostituzione della generalizzazione con associazioni**.
Questa scelta consente di: evitare la ripetizione dei campi generici, mantenere la normalizzazione, facilitare l'estendibilità del modello, separare logicamente i dati specifici per categoria.
Tuttavia aggiunge un vincolo, ossia: ogni occorrenza del padre `Annuncio` non può partecipare contemporaneamente a due relazioni `IS-[categoria]`.

4.1.3 Schema ER ristrutturato

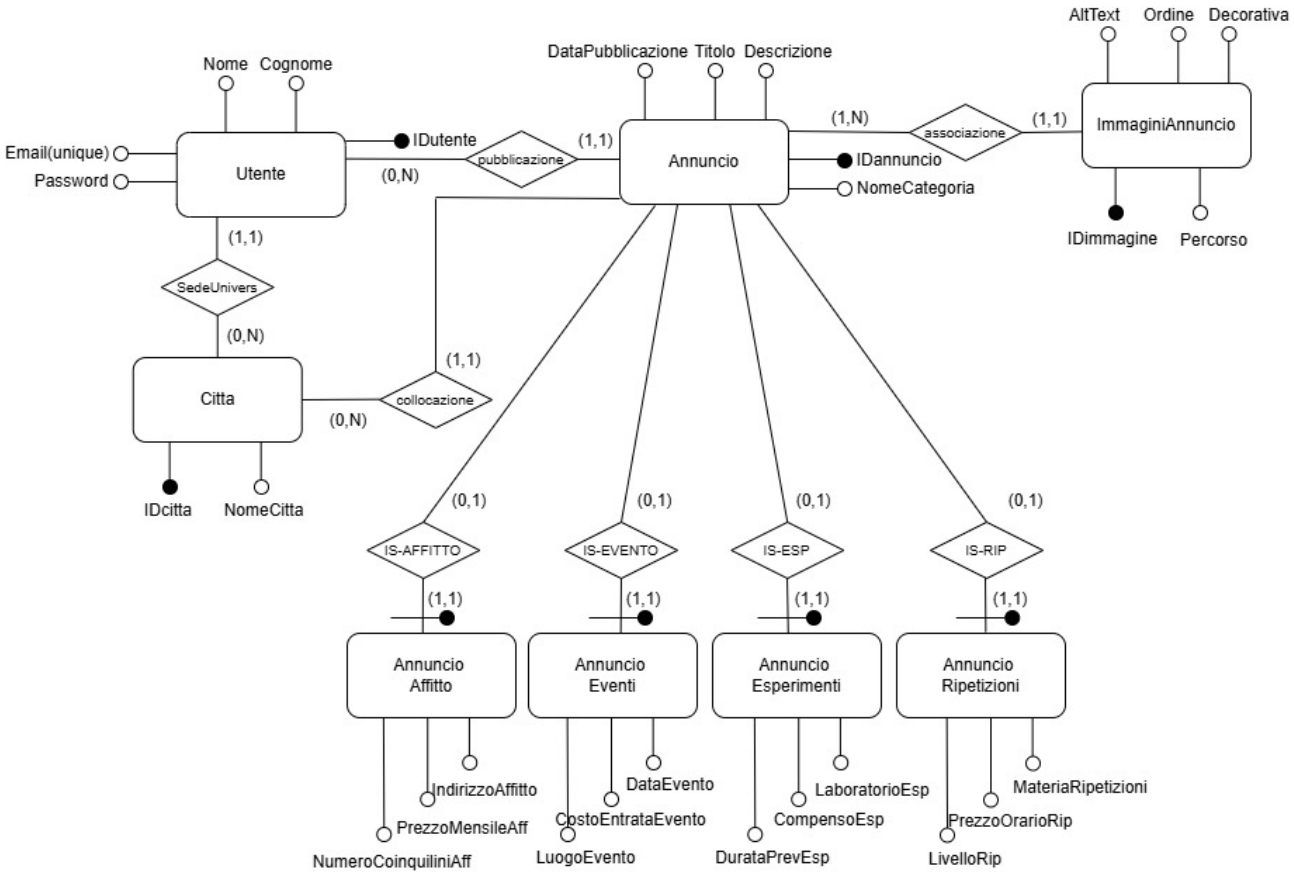

Figura 2: Schema ER ristrutturato

Relaazione IS-[categoria]: collega `Annuncio` alle specializzazioni (`Affitti`, `Esperimenti`, `Eventi`, `Ripetizioni`).
`Annuncio (1,1)` $\longrightarrow$ `Specializzazione (0,1)`
Ogni annuncio può avere al massimo una specializzazione, e ogni specializzazione è riferita ad un solo annuncio.

4.1.4 Scelta delle chiavi primarie

Nella progettazione del database si è deciso di utilizzare come chiavi primarie codici (ID numerici autoincrementali).

Questa scelta è motivata da diversi fattori:

- **Stabilità**: gli attributi naturali (es. `Email`,
- **Efficienza**: gli ID numerici sono più leggeri da gestire negli indici e nelle join rispetto a stringhe lunghe come email o nomi.
- **Semplicità**: l'uso di codici numerici rende più chiaro e uniforme lo schema, evitando di dover scegliere chiavi composite o troppo complesse.

- **Integrità**: gli attributi naturali vengono comunque vincolati con `UNIQUE` (es. `Email`,
- **Estendibilità**: se in futuro si aggiungessero nuove categorie o sedi, l'uso di ID permetterebbe di gestire facilmente l'espansione senza modificare le relazioni esistenti.

4.1.5 Schema logico (relazionale)

Di seguito è riportato lo schema logico, in cui l'asterisco dopo il nome di un attributo indica che tale attributo può assumere valori nulli.

- **Utente**(IdUtente, Nome, Cognome, Email [UNIQUE], Password, SedeUniversitaria*
  - Utente.SedeUniversitaria → Citta.IdCitta
- **Citta**(IdCitta, NomeCitta [UNIQUE])
- **Annuncio**(IdAnnuncio, Titolo, Descrizione, DataPubblicazione, Autore, Categoria, Collocazione)
  - Annuncio.Autore → Utente.IdUtente
  - Annuncio.Collocazione→ Citta.IdCitta
- **Affitti**(IdAnnuncio, PrezzoMensileAff, IndirizzoAffitti, NumeroCoinquilini)
  - Affitti.IdAnnuncio→ Annuncio.IdAnnuncio
- **Esperimenti**(IdAnnuncio, LaboratorioEsp, DurataPrevEsp, CompensoEsp)
  - Esperimenti.IdAnnuncio→ Annuncio.IdAnnuncio
- **Eventi**(IdAnnuncio, DataEventi, LuogoEventi, CostoEntrataEventi)
  - Eventi.IdAnnuncio→ Annuncio.IdAnnuncio
- **Ripetizioni**(IdAnnuncio, MateriaRipetizioni, LivelloRip, PrezzoOrarioRip)
  - Ripetizioni.IdAnnuncio→ Annuncio.IdAnnuncio
- **ImmaginiAnnuncio**(IdImmagine, IdAnnuncio (FK), Percorso, AltText*, Decorativa, Ordine)
  - ImmaginiAnnuncio.IdAnnuncio → Annuncio.IdAnnuncio

4.1.6 Normalizzazione

Lo schema progettato rispetta la terza forma normale (3NF):

- **1NF**: tutti gli attributi sono atomici e non esistono campi multivalore o ripetuti.
- **2NF**: ogni tabella ha una chiave primaria semplice (IdUtente,
- **3NF**: non esistono dipendenze transitive tra attributi non chiave.

4.1.7 Vincoli di integrità

Lo schema logico rispetta i principali vincoli di integrità:

- **Unicità**: Email utente,
- **Dominio**: campi obbligatori definiti come NOT NULL.
- **Referenziali**: ON DELETE CASCADE sulle specializzazioni; SET NULL in Annuncio su Utente, Categoria, Citta nel caso vengano eliminati.
- **Generalizzazione IS-A**: ogni annuncio può avere al massimo una specializzazione coerente con la categoria.

4.2 Front-end (HTML, CSS, JS)

4.2.1 HTML

Nel documento è stato inserito il tag:

```
<meta name="viewport" content="width=device-width"/>
```

Tale configurazione permette ai fogli di stile di riferirsi alla dimensione effettiva del dispositivo e non alla sua risoluzione fisica. In questo modo si evita che, su smartphone ad alta densità di pixel, venga caricata la visualizzazione desktop, garantendo un comportamento responsivo coerente con il paradigma *mobile first*.

Nella pagina *esplora* è stato scelto di utilizzare l'elemento semantico `<aside>` per contenere i filtri di ricerca, invece di un elemento `<section>`. Tale scelta deriva dal fatto che i filtri rappresentano contenuti accessori rispetto al flusso principale della pagina, costituito dall'elenco degli annunci. L'uso di `<aside>` migliora la chiarezza semantica e supporta una migliore interpretazione da parte delle tecnologie assistive.

Per quanto riguarda le singole card, gestite tramite `cardTemplate.html`, è stato deciso di utilizzare elementi `<h3>` per i titoli degli annunci, anziché una struttura basata su `<dt>` e `<dd>`, come nel seguente esempio:

```
<dt>Titolo:</dt>
<dd>[TitoloAnnuncio]</dd>
```

L'uso degli heading consente di riflettere correttamente la gerarchia logica dei contenuti: `<h2>` identifica la sezione principale (ad esempio l'elenco degli annunci), mentre `<h3>` rappresenta ciascun singolo annuncio. Questa scelta migliora la struttura semantica complessiva, favorisce un migliore posizionamento nei motori di ricerca e permette agli screen reader di interpretare più efficacemente l'organizzazione informativa della pagina.

4.2.2 CSS e decisioni di design (immagini, palette colori)

4.3 Back-end (PHP, Database)

4.4 Validazione e Sicurezza

- **Lato client**: controlli immediati con JavaScript (campi obbligatori, formati corretti).
- **Lato server**: verifica dati ricevuti prima dell'inserimento nel database.
- Messaggi di errore chiari e accessibili.
- non sono concessi tag negli input

5 Test

5.1 Test funzionali

Descrizione dei test su login, registrazione, pubblicazione e filtraggio annunci.

5.2 Test di compatibilità

Test su diversi browser (Chrome, Firefox, Edge, Safari) e dispositivi (desktop, tablet, smartphone).

5.3 Test di accessibilità

Color contrast, navigazione da tastiera, screen reader.

6 Accessibilità

6.1 Introduzione

6.2 WAI-ARIA e screen-reader

6.3 Progressive Enhancement

6.4 Supporto ai dispositivi "legacy"

6.5 Ulteriori funzionalità di accessibilità

7 SEO e prestazioni

7.1 Analisi delle possibili ricerche su Google

8 Strumenti utilizzati

- **Canva** https://www.canva.com/it_it/
  Piattaforma di progettazione grafica per creare contenuti visuali per presentazioni, siti web e prodotti simili.

- **Discord** https://discord.com/
  Piattaforma di messaggistica istantanea e distribuzione digitale, utilizzata per la comunicazione del gruppo. Supporta chiamate vocali, videochiamate, messaggi di testo e condivisione file.

- **Git** https://git-scm.com/
  Applicazione software per il controllo di versione distribuito, usata per il versionamento del codice e della documentazione.

- **Github** https://github.com/
  Servizio di hosting per progetti software che implementa Git, utilizzato per la collaborazione e la condivisione del codice.

- **LaTeX** https://www.latex-project.org/get/
  Linguaggio di marcatura del testo, usato per la preparazione della relazione accademica.

- **Silktide** https://silktide.com/
  Estensione che facilita i test sull'accessibilità.

- **W3C Validator** http://validator.w3.org/
  Strumento per validare il markup HTML/XHTML.

- **WhatsApp** https://www.whatsapp.com/
  Piattaforma di messaggistica istantanea usata per la comunicazione rapida tra i membri del gruppo.

- **WebAIM** https://webaim.org/resources/contrastchecker/
  Strumento online per verificare il contrasto dei colori e migliorare l'accessibilità.

9 Guida all'installazione del progetto