Progetto biblioteca

[Primo Levi Vignola]

[Titolo del corso]

Sommario

[1 Introduzione 1](#_Toc10187730)

[2 U.I (User Interface) 2](#_Toc10187731)

[3 Progettazione Database 2](#_Toc10187732)

[3.1 Inserimento dati e Mockaroo 2](#_Toc10187733)

[3.2 Modello E.R 3](#_Toc10187734)

[4 OAuth2 3](#_Toc10187735)

[4.1 token 3](#_Toc10187736)

[4.2 Postman 4](#_Toc10187737)

[5 Collegamento servizi di terze parti (ISBN) 4](#_Toc10187738)

[5.1 ISBN 4](#_Toc10187739)

[6 WebApi 4](#_Toc10187740)

[6.1 PDO 4](#_Toc10187741)

[6.2 REST SERVICES 5](#_Toc10187742)

[6.2.1 CRUD 5](#_Toc10187743)

# Introduzione

Per la gestione della biblioteca della nostra scuola ci è stato proposto di realizzare un sito web, con il quale è possibile visualizzare i libri presenti all’interno della biblioteca e tenere traccia di quelli dati in prestito.

Il progetto nasce come modifica di un sito preesistente. Durante lo svolgimento del progetto ci siamo dovuti riunire in gruppo più volte, parlare in modo da stabilire un metodo di lavoro e organizzare gruppi per lo svolgimento di determinarti compiti.

Revisionando il progetto precedente abbiamo constatato che non c’era molto da riutilizzare perché già dall’inizio del progetto avevamo le idee chiare su quello che volevamo realizzare.

Durante il primo incontro di gruppo si è deciso come si voleva impostare a grandi linee il sito web e successivamente abbiamo diviso il progetto in tanti piccoli progetti in modo tale da semplificare e velocizzare il lavoro. Le varie divisioni del progetto sono:

* User Interface
* Progettazione Database
* Autentificazione OAuth2
* Collegamento a servizi terzi (ISBN)
* Linguaggio SQL

Scelti i vari gruppi ci siamo messi al lavoro, dove ognuno di noi ha contribuito nel proprio piccolo allo svolgimento e alla realizzazione finale del progetto. Durante lo svolgimento del progetto è capitato molte volte che membri di gruppi diversi si aiutassero a vicenda in modo da raggiungere lo scopo finale comune. Questo concetto di aiuto reciproco nella programmazione, o in generale nella realizzazione di un progetto di media-alta complessità, è una delle forme più efficaci e potenti per risolvere un problema. Il lavoro di gruppo è, nella maggior parte delle volte, il modo migliore di lavorare.

# U.I (User Interface)

# Progettazione Database

Dopo una lunga ed accurata revisione del progetto precedentemente svolto, abbiamo deciso di riprogettare il Database a seconda delle esigenze che ci eravamo prefissati, consentendoci di gestire i libri presenti nella biblioteca e di quelli dati in prestito in una maniera più efficace e intuitiva.

Nel nuovo Database sono presenti 5 tabelle collegate tra loro tramite una tabella di giunzione, comunemente chiamata Junction Tables.

La tabella **Autori** contiene le informazioni dei vari autori dei libri presenti in biblioteca tramite la quale è possibile visualizzare gli autori in base al “Nome” “Cognome” “DataNascita” “DataMorte”.

Nella tabella **Libri** oltre alle informazioni che caratterizzano ogni libro, ad esempio il “Titolo” “AnnoPubblicazione” “ISBN” sono presenti anche la “CollocazioneArmadio” “CollocazioneScaffale” “CollocazioneLuogo” tramite le quali è possibile individuare dove è situato ogni singolo libro.

Sono presenti anche i campi per poter visualizzare la data in cui un utente della biblioteca ha preso in prestito un libro e quando lo ha restituito.

Il campo “idGenere” indica il genere del libro.

La tabella **Utenti** contiene le informazioni personali degli Utenti che sono registrati alla biblioteca.

È stato inserito un campo per valutare se un determinato account di un utente è stato disabilitato o meno ed inoltre è presente l’opzione di un utente di essere un utente normale(guest) o con privilegi di amministratore. Un utente amministratore rispetto a tutte le funzioni che può eseguire un utente normale può anche modificare, eliminare e aggiungere dei libri

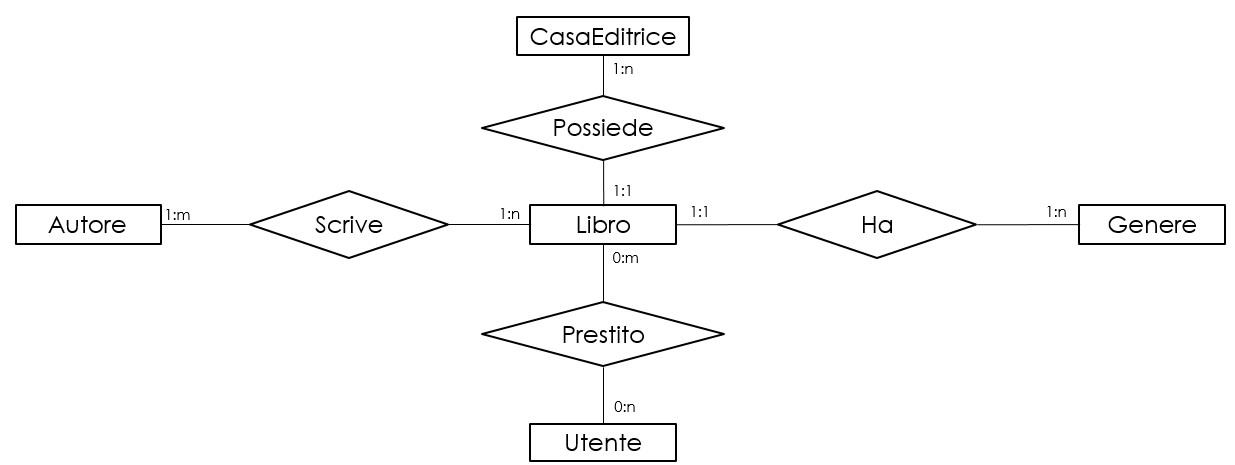
La tabella **Generi** contiene tutti i possibili generi a cui appartengono i libri.

La tabella **Casa Editrici** contiene il nome ed il luogo della sede di ogni casa editrice che ha editato un libro.

## Inserimento dati e Mockaroo

Tramite l’utilizzo di Mockaroo, un generatore di dati casuali con un’interfaccia facile da usare che consente di risolvere i problemi di generazione dei dati in pochi click. Abbiamo inserito e generato i dati richiesti dal database con le loro rispettive caratteristiche. Questo programma ci ha velocizzato molto nella realizzazione del database in maniera da testare il nostro sito web. Visitando il sito web di Mockaroo vengono visualizzate colonne con i nomi dei campi dati, i loro tipi di dati e le opzioni vuote che è possibile modificare. La colonna denominata "Field Name" accetta un semplice inserimento di testo, "Type" accetta il tipo di dati e "Option" accetta l'input numerico. È possibile fare click su un tipo per aprire una finestra che mostra i tipi di dati disponibili dai quali è possibile selezionare quello adatto alle proprie esigenze. Si possono aggiungere altri campi facendo clic sul pulsante "Add Another Field" e rimuoverli facendo click su una piccola croce (x). Nella parte inferiore si trova una casella che specifica il numero di righe che si vorranno generare (è possibile modificarlo con qualsiasi valore inferiore a 1000), il formato in cui verranno generati i dati (anche questo può essere modificato) e un pulsante “Download” per scaricare i dati e anche scegliere di visualizzare in anteprima quali dati verranno scaricati facendo clic sul pulsante "Preview". Esistono anche opzioni per salvare le impostazioni sul profilo attraverso il pulsante "Clone This Schema".

## Modello E.R



# OAuth2

Seguendo la linea guida proposta da [GitHub](https://bshaffer.github.io/oauth2-server-php-docs/cookbook/), abbiamo integrato la funzione di login degli utenti tramite la tecnica definita Oauth2 in linguaggio PHP. Questa tecnica consiste nell’utilizzare come metodo di autenticazione un servizio terzo (chiamato nella nostra architettura Oauth2 Server) che fornisce un token di accesso.   
Il client-side si interfaccia con il server-side tramite l’invio di questo token, rendendo più agevoli, efficienti e sicure le operazioni di login dell’utente. Le nostre Web API verificano il token e rendono possibile l’accesso all’utente solo se il token è valido. Inoltre il server esamina il token e restituisce il livello dell’utente, il quale identifica l'utente, i suoi privilegi e il gruppo di utenti al quale appartiene. Abbiamo poi integrato la nostra applicazione con il resto delle Web API e abbiamo aggiornato il database inserendo le strutture relative agli utenti. Dopo aver implementato la nostra architettura (dall’Oauth2 Server al database), abbiamo testato la nostra applicazione tramite richieste POST su protocollo HTTPS dall’utility Postman. Il token viene restituito in formato JSON con i campi che identificano sia il token sia il livello di privilegi associato all’utente.   
Solo l’utente amministratore ha la possibilità di aggiungere utenti al database L’autenticazione avviene con una richiesta POST al server verso la pagina Token.php che fa uso di librerie preimpostate le quali rendono l’assegnazione di un token facile e automatizzata. Il client riceve il token e lo salva nella cache locale, così da non dover richiedere nuovamente un token per ogni refresh della pagina. Se le informazioni di login sono errate, il server Ouath2 non permette l’accesso, anche se la gestione degli errori deve essere migliorata in futuro. In questo momento, manca la parte di lettura degli utenti.  
In previsione futura, sarà possibile anche controllare l’expires time (tempo di scadenza) del token per rendere più attendibile l’associazione tra utente e token.

## token

Il concetto generale alla base di un sistema di autenticazione basato su token è semplice. Consenti agli utenti di immettere il nome utente e la password per ottenere un token che consenta loro di recuperare una risorsa specifica, senza utilizzare nome utente e password. Una volta ottenuto il token, l'utente può offrire il token, che offre l'accesso a una risorsa specifica per un periodo di tempo, al sito remoto.

## Postman

Postman è un ambiente di sviluppo API che ci aiuta a creare, testare, documentare e monitorare le loro API. Tramite l’interfaccia grafica è possibile selezionare facilmente il tipo di chiamata da effettuate (POST, GET, PUST, DELETE, ecc.), impostare l’url su effettuare la chiamata e inviarla. Postman mette a disposizione una serie di tab per poter impostare i dati per l’autorizzazione della chiamata (ad es. OAuth2), gli headers e il body della chiamata API.

# Collegamento servizi di terze parti (ISBN)

Una funzione aggiuntiva al sito web che abbiamo voluto integrare è il collegamento a un servizio di terze parti che con l’inserimento di un ISBN ci permette di avere informazioni su libri anche non presenti nella biblioteca.

Il servizio di terzi parti a cui ci colleghiamo è Google Books. Con l’inserimento dell’ISBN nel sito, se è esistente, ci verranno restituiti i vari dati del libro.

Il nostro obbiettivo era di integrare più servizi di terze parti in modo da avere più informazioni possibili su un determinato libro, però questo non è stato possibile perché Google Books è l’unico servizio di terze parti gratuito. Avremmo potuto integrare altri servizi di terze parti però sarebbero stati a pagamento, ma essendo un progetto scolastico abbiamo preferito optare per sevizi gratuiti.

Il servizio di terze parti funziona con un collegamento fra il sito web della biblioteca e il sito web di Google, il quale restituisce i dati del libro in formato JSON. Il JSON viene modificato e viene creato un nuovo JSON che conterrà solo le informazioni utili a noi.

Per lo scambio di informazioni utilizziamo il JSON per facilitare la transazione e la leggibilità dei dati.

## ISBN

Il codice **ISBN** (**International Standard Book Number**) è una sequenza numerica di 13 cifre usata internazionalmente per la classificazione dei libri. Sebbene non obbligatorio, il suo uso però è ormai diventato essenziale per l'immissione del prodotto librario nei canali della grande distribuzione.

Ogni codice ISBN identifica in modo univoco ogni specifica edizione di un libro (non però le semplici ristampe, che mantengono lo stesso codice dell'edizione cui si riferiscono) e, una volta assegnato, non può più essere riutilizzato.

# WebApi

## PDO

Per poter connettere la parte Server sviluppata in PHP al database MySQL abbiamo utilizzato PDO [PHP Data Object]. Questo sistema permette di lavorare in un modo abbastanza astratto verso i dati del DB. Infatti grazie a PDO è possibile scrivere tutto il server Web e poi cambiare in base alle proprie necessità la connessione al DB cambiando solo un parametro della connessione.

## REST SERVICES

Per lo sviluppo delle WebAPI è stata utilizzata l’architettura REST [Representational State Transfer]. Questa si basa sul protocollo di comunicazione HTTP e questi non presentato uno stato quindi non esiste in concetto di sessione. Per questo la comunicazione tra la UI e le WebAPI avviene tramite i verbi HTTP:

* GET: recupero di informazione
* POST: modifica di una risorsa
* PUT: inserimento di una risorsa
* DELETE: eliminazione di una risorsa

### CRUD

I CRUD [Create Read Update Delete] consistono nel creare dei file per ogni entità del progetto chiamati “controller” coi quali si fa routing della richiesta HTTP della UI e poi si associa a un verbo HTTP un metodo PHP nel quale verrà poi composta una query con un particolare statment in base alla richiesta fatta.

|  |  |  |
| --- | --- | --- |
| HTTP | PHP | MySQL |
| GET | Read() | Select |
| POST | Update() | Update |
| PUT | Create() | Insert |
| DELETE | Delete() | Delete |