

# UNIVERSITÀ DEGLI STUDI DI SALERNO

DIEM - Dipartimento di Ingegneria dell'Informazione ed Elettrica e Matematica Applicata



Corso di Laurea Triennale in Ingegneria Informatica

## Progetto - Mobile Programming **Bibliotech**

Gruppo 17:  
**Emanuele Tocci - 0612707488**  
**Alessio Leo - 0612707279**  
**Rossella Pale - 0612707284**  
**Claudia Montefusco - 0612707404**

ANNO ACCADEMICO 2024/2025

# Indice

<b>1</b>	<b>Introduzione</b>	<b>2</b>
1.1	Panoramica . . . . .	2
1.2	Obiettivo del Progetto . . . . .	2
1.3	Design dell'interfaccia . . . . .	2
<b>2</b>	<b>Schermate principali</b>	<b>4</b>
2.1	MainView . . . . .	4
2.1.1	Homepage . . . . .	4
2.1.2	Libreria . . . . .	4
2.1.3	Statistiche . . . . .	5
2.2	Aggiunta/Modifica libro . . . . .	6
2.3	Dettaglio libro . . . . .	6
2.4	Ricerca tramite API . . . . .	7
<b>3</b>	<b>Architettura dell'applicazione</b>	<b>8</b>
3.1	Organizzazione del codice . . . . .	8
3.2	Librerie e dipendenze esterne . . . . .	8
<b>4</b>	<b>Persistenza dei dati</b>	<b>8</b>
4.1	Implementazione . . . . .	8
4.2	Strategia di sincronizzazione . . . . .	9
4.3	Schema del database . . . . .	9
<b>5</b>	<b>Libri di esempio per il test</b>	<b>9</b>
<b>6</b>	<b>Documentazione tecnica</b>	<b>10</b>

## 1 Introduzione

### 1.1 Panoramica

**Bibliotech** è un'applicazione mobile multiplataforma sviluppata per facilitare la gestione della libreria personale degli utenti. L'applicazione consente di **catalogare** i libri, sia quelli già letti che quelli da leggere, e di arricchire le informazioni relative a ciascun libro con note personali e valutazioni. Bibliotech offre inoltre funzionalità per l'organizzazione dei libri tramite categorie, la ricerca rapida e la visualizzazione di statistiche di lettura.

### 1.2 Obiettivo del Progetto

L'obiettivo principale di Bibliotech è offrire agli utenti uno strumento semplice ed efficace per gestire la propria libreria personale di libri. L'applicazione permette di **catalogare**, **organizzare** e tenere traccia dei volumi posseduti, facilitando la consultazione, la ricerca e la gestione delle proprie letture in modo pratico e intuitivo.

Gli utenti possono:

1. Aggiungere libri manualmente o tramite ricerca
2. Aggiungere note personali e valutazioni per ogni libro
3. Ricercare libri
4. Esplorare statistiche di lettura per monitorare i propri progressi
5. Organizzare la libreria per generi e stato

### 1.3 Design dell'interfaccia

Alla fase di definizione delle funzionalità è seguita la fase di **design** con la progettazione dei wireframe e dei **mockup**, utilizzati come bozza iniziale per la struttura dell'applicazione.

# 1 INTRODUZIONE



(a) Mockup Homepage

The page displays a search bar with the placeholder "Hinted search book" and a search icon. It shows a grid of books with categories: Fantasy, Thriller, Science, and an "Add" button. Two specific books are highlighted: "Flutter Engineering" by Majid Hajian and "Essentials of Software Engineering". To the right is a form for adding a new book, including fields for Title, Author, Language, ISBN, Category, Publication Date, and Personal Notes, with an "Aggiungi" button at the bottom.

(b) Mockup Libreria

This is a detailed view of the book addition form. It includes fields for "Titolo del libro", "Autori", "Lingua", "ISBN", "Categoria", "Data Pubblicazione", and "Note personali". An "Aggiungi" button is located at the bottom right.

(c) Mockup Aggiunta

The page shows a book cover for "Programmare per il Web Mobile" by Maximiliano Firtman. It has a rating of 5 (8) stars. Below the cover are tabs for "Info" and "Note". Under "Info" are sections for AUTORI (Maximiliano Firtman), LINGUA (Italiano), PAGINE (600), TRAMA ( lorem ipsum ), ISBN (48388829202), STATO (Da leggere), CATEGORIA (Horror), and DATA PUBBLICAZIONE (2025-05-14). There is also a pencil icon for editing.

(d) Mockup Dettagli Libro



(e) Mockup Statistiche

## 2 Schermate principali

### 2.1 MainView

MainView rappresenta lo scheletro dell'applicazione, mettendo a disposizione:

- Scaffold condiviso
- Barra di navigazione (BottomNavigationBar) condivisa
- Action Button condivisa per l'aggiunta rapida di un libro

Da tale schermata è possibile navigare verso 3 "sotto-schermate": Homepage (2.1.1), Libreria (2.1.2) e Statistiche (2.1.3).

#### 2.1.1 Homepage



La schermata **Home** si compone di 4 sezioni principali:

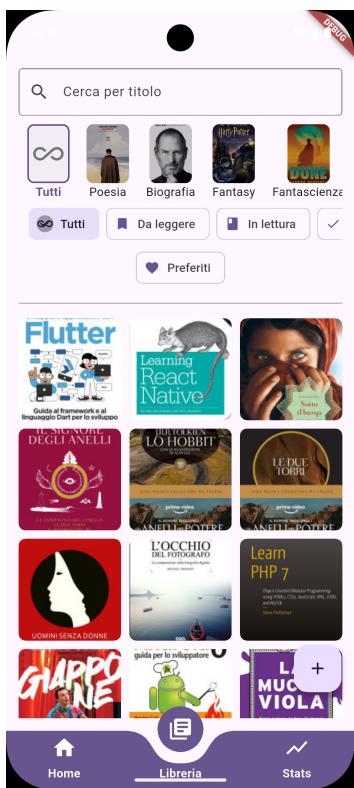
- Sezione di **benvvenuto** che invita l'utente ad aggiungere un nuovo libro alla libreria;
- **Citazioni** letterarie randomizzate in base alla data odierna;
- Carosello contenente i libri che il sistema **consiglia** (randomicamente) all'utente;
- Carosello contenente gli **ultimi libri** che ha aggiunto l'utente;

Figura 1: Schermata Home

#### 2.1.2 Libreria

La schermata **Libreria** consente di organizzare la propria collezione mediante una griglia che mostra visivamente le copertine dei libri presenti. Ogni copertina è cliccabile e consente di aprire il dettaglio completo (2.3) del libro selezionato. I libri possono essere filtrati secondo vari criteri e ricercati per titolo.

## 2 SCHERMATE PRINCIPALI



Si compone su grandi linee di 2 sezioni:

- Filtraggio e ricerca
- Lista dei libri presenti in libreria

La vista consente:

- Ricercare un libro per titolo
- Filtrare per genere
- Filtrare per stato
- Filtrare per preferiti

Figura 2: Schermata Libreria

### 2.1.3 Statistiche

La schermata "Statistiche" fornisce una panoramica visiva e sintetica dello stato della libreria dell'utente.

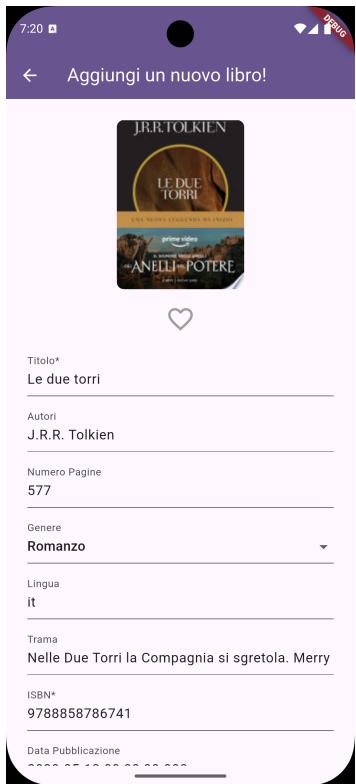


La schermata si compone essenzialmente di 5 sezioni:

- Conteggio dei libri in base allo stato
- Storico recensioni (voto)
- Stima su lettura e note
- Diagramma dei generi più letti
- Diagramma di tutti i generi presenti in libreria

Figura 3: Schermata Statistiche

## 2.2 Aggiunta/Modifica libro



La schermata consente l'**aggiunta manuale** di un nuovo libro o la **modifica** di un libro esistente.

Si compone di un insieme di `TextField` e `DropdownButtonFormField` che consentono l'inserimento manuale di tutti i parametri necessari al salvataggio del libro nella libreria. Il controllo sui campi inseriti dall'utente è delegato al relativo controller (`AggiuntaModificaController`), il quale si occupa di validare principalmente il **titolo** e l'**isbn** inserito... l'inserimento è consentito solamente se tali campi obbligatori sono **validi**.

Figura 4: Schermata Aggiunta/Modifica

## 2.3 Dettaglio libro

La schermata di dettaglio consente la visualizzazione dei dettagli un **generico libro**, che sia presente o meno all'interno della libreria. Consente l'**inserimento rapido** di un libro esistente (ottenuto tramite API) o la **rimozione** qualora il libro fosse

già presente in libreria. L'utente che effettua la ricerca nel catalogo Google Books può quindi decidere se inserire direttamente il libro con i parametri forniti dall'API, oppure **personalizzare** lo stesso, andandone a modificare manualmente i parametri.

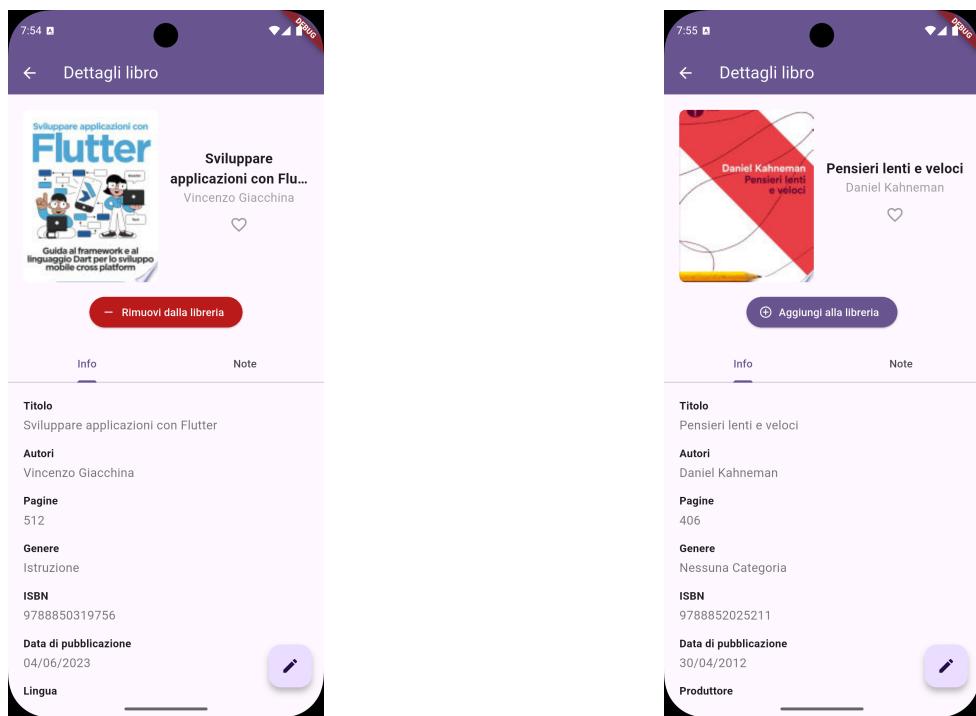
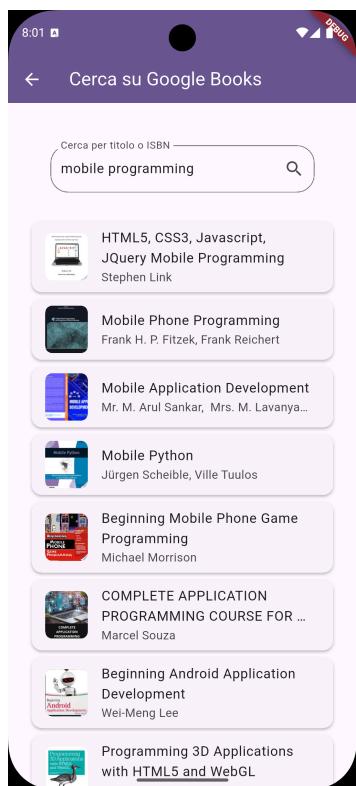


Figura 5: Schermata Dettaglio

Il pulsante di **modifica** posto in basso a destra consente infatti l’apertura della schermata di **Aggiunta/Modifica** (2.2), i cui campi vengono pre-caricati dal libro che si sta modificando.

### 2.4 Ricerca tramite API



La schermata consente di **interfacciarsi** con l’**API**: l’utente cerca un libro tramite titolo o ISBN e il sistema mostra i risultati ottenuti dal catalogo *Google*. Gli elementi della lista dei risultati ottenuti sono cliccabili e riportano alla schermata di **Dettaglio** (2.3).

Figura 6: Schermata Ricerca tramite API

## 3 Architettura dell'applicazione

L'app è stata realizzata in Flutter e, per garantire chiarezza, modularità e facilità di manutenzione, è stata progettata seguendo alcuni principi architetturali:

- **Separazione delle responsabilità:** L'organizzazione del codice si ispira (per quanto possibile) al pattern *Model-View-Controller MVC*;
- **Gestione centralizzata dello stato:** la classe Libreria funge da modello dati centralizzato mediante l'implementazione del pattern **Singleton**, il quale assicura un'unica istanza condivisa in tutta l'applicazione. L'integrazione con il package **Provider** consente inoltre di propagare automaticamente le modifiche dello stato ai widget interessati, mantenendo la UI sempre sincronizzata con i dati.

### 3.1 Organizzazione del codice

Il codice è stato organizzato in varie directory specializzate, le quali seguono il seguente schema organizzativo:

```
doc ..... Documentazione progettuale
lib/
  components ..... Custom Widgets da usare nelle varie schermate
  models ..... Modelli a oggetti
  screens ..... Schermate dell'applicazione
  services ..... Logica di business
    apis/
    controllers/
    db/
    utilities/
    themes/
```

All'interno di ogni cartella è stato comunque inserito un file `README.md` che ne spiega lo scopo.

### 3.2 Librerie e dipendenze esterne

- |                                 |                      |                   |
|---------------------------------|----------------------|-------------------|
| • Curved Navigation Bar         | • Sqflite            | • Intl            |
| • Curved Labeled Navigation Bar | • Http               | • Dartdoc         |
| • Image Picker                  | • Provider           | • Flutter Rating  |
| • Path Provider                 | • Sqflite Common Ffi | • Fl-chart        |
|                                 | • Isbn               | • Google Book API |

## 4 Persistenza dei dati

### 4.1 Implementazione

La **persistenza dei dati** è realizzata tramite un database **SQLLite**, integrato direttamente nell'applicazione. Questa soluzione garantisce la conservazione affidabile delle informazioni

tra le sessioni, offrendo al contempo ottime prestazioni e scalabilità anche in presenza di grandi quantità di dati.

L'implementazione è stata effettuata utilizzando il plugin `sqflite` per dispositivi mobili assicurando così un'integrazione nativa con l'ambiente `Flutter`.

Per la gestione della persistenza è stata realizzata una classe dedicata (`DatabaseHelper`) che si occupa di tutte le operazioni di accesso e modifica al database. La classe in questione è stata implementata come **Singleton** al fine di evitare l'accidentale creazione di più connessioni simultanee alla base di dati.

### 4.2 Strategia di sincronizzazione

Per assicurare la coerenza tra lo stato dell'applicazione e i dati memorizzati, è stato adottato un approccio "*Database-First*": le operazioni **CRUD** vengono eseguite prima sul database... solo in caso di esito positivo, la struttura interna viene aggiornata.

Nonostante ciò, il db, è stato utilizzato esclusivamente per garantire persistenza... il resto delle operazioni di filtraggio e ricerca sono state implementate direttamente sulla struttura interna.

### 4.3 Schema del database

libri
isbn TEXT PRIMARY KEY
titolo TEXT NOT NULL
autori TEXT
numeroPagine INTEGER
genere TEXT
lingua TEXT
trama TEXT
dataPubblicazione TEXT
voto REAL
copertina TEXT
note TEXT
stato TEXT
publisher TEXT

Figura 7: Schema della tabella `libri` del database SQLite

## 5 Libri di esempio per il test

Dal momento che l'aggiunta manuale di un libro è consentita solamente se il codice *ISBN* associato è valido, si propongono un'insieme di libri che possono essere utilizzati in fase di testing manuale:

1. *Il mondo deve sapere* , 9788876382406
2. *1984* , 9788834741580
3. *Dracula* , 9788818028157
4. *Ghiblioteca* , 9791259570833
5. *Sviluppare applicazioni con Flutter* ,  
9788850319756
6. *Wonder. Tutte le storie* ,  
9788809905832
7. *Steve Jobs (Italian Edition)* ,  
9788852021060
8. *Le cronache di Narnia* , 9788804669234

## 6 Documentazione tecnica

La **documentazione tecnica** del codice sorgente è stata generata mediante Dartdoc ed è disponibile presso il seguente link: <https://emanueleletocci.github.io/Bibliotech/>.