

# 書籍管理 Web アプリ

木村友亮

## 1. 設計のコンセプト

今回作成したシステムのコンセプトは、私が書籍管理をする上で欲しい機能を実装することにした。特に、書籍の表紙で一覧表示、Markdown 形式での読書記録である。書籍のタイトルだけでは思い出せない場合に表紙での表示はわかりやすい。読書記録に Markdown 形式を採用した理由は私の好みでもあるが、情報の整理と構造化がしやすいため記録用に適していると考えた。紙・電子書籍の両方を購入する人に向けて購入した書籍を一括管理できるシステムを目指す。

## 2. 目標

課題に取り組む上で、自分で決めたスケジュール通りに実装を進めることを目標とした。また、ソースコードを機能別に分けることで保守性を上げることを目標とした。保守性向上のために views フォルダ、routes フォルダ、controllers フォルダのように機能別にソースファイルを分割することにした。

## 3. 機能概要

本システムは、Node.js と Express と prisma(sqlite3)を使用し書籍管理 Web アプリケーションを実装している。利用者は 1 人として設計している。本システムは、ログイン機能、GoogleBooksAPI を使用した書籍検索機能、書籍についての Markdown 形式の読書記録の登録・変更・削除機能、書籍一覧表示機能を持つ。

## 4. 画面の状態遷移

本システムは、1 人用とし管理者はいないため、利用者のエン트리ポイントは src/index.js で npm start で実行する。

書籍管理 Web アプリシステムの画面遷移図を以下に示す。

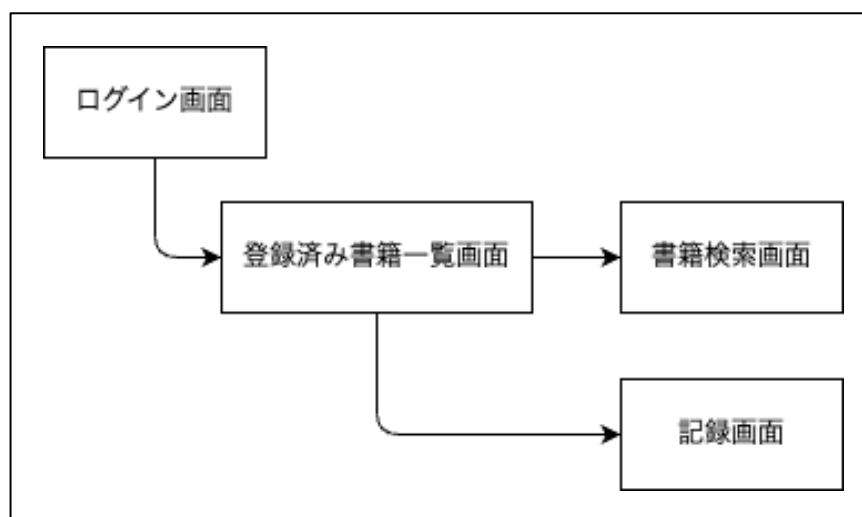


図 1 書籍管理 Web アプリシステム画面遷移図

## 5. テーブル一覧

書籍管理 Web アプリシステムで仕様するデータベースのテーブル一覧を以下に示す。

表 1 書籍管理 Web アプリシステムテーブル一覧

テーブル名	列名	データ型	主キー	必須入力	用途
USERS_TABLE	User_ID	INTEGER	Yes	Yes	ID(連番)
USERS_TABLE	User_Name	TEXT		Yes	ユーザー名
USERS_TABLE	User_Password	TEXT		Yes	パスワード
BOOK_READING_RECORDS_TABLE	Record_ID	INTEGER	Yes	Yes	ID(連番)
BOOK_READING_RECORDS_TABLE	Record_User_ID	INTEGER		Yes	外部キー
BOOK_READING_RECORDS_TABLE	Record_Title	TEXT		Yes	タイトル
BOOK_READING_RECORDS_TABLE	Record_Book_Image_Url	TEXT		Yes	表紙
BOOK_READING_RECORDS_TABLE	Record_Content	TEXT		No	記録内容
BOOK_READING_RECORDS_TABLE	Record_CreateDate	DATETIME		Yes	登録日
BOOK_READING_RECORDS_TABLE	Record_UpdateDate	DATETIME		Yes	更新日

## 6. ER 図

書籍管理 Web アプリ用データベースの ER 図を以下に示す。

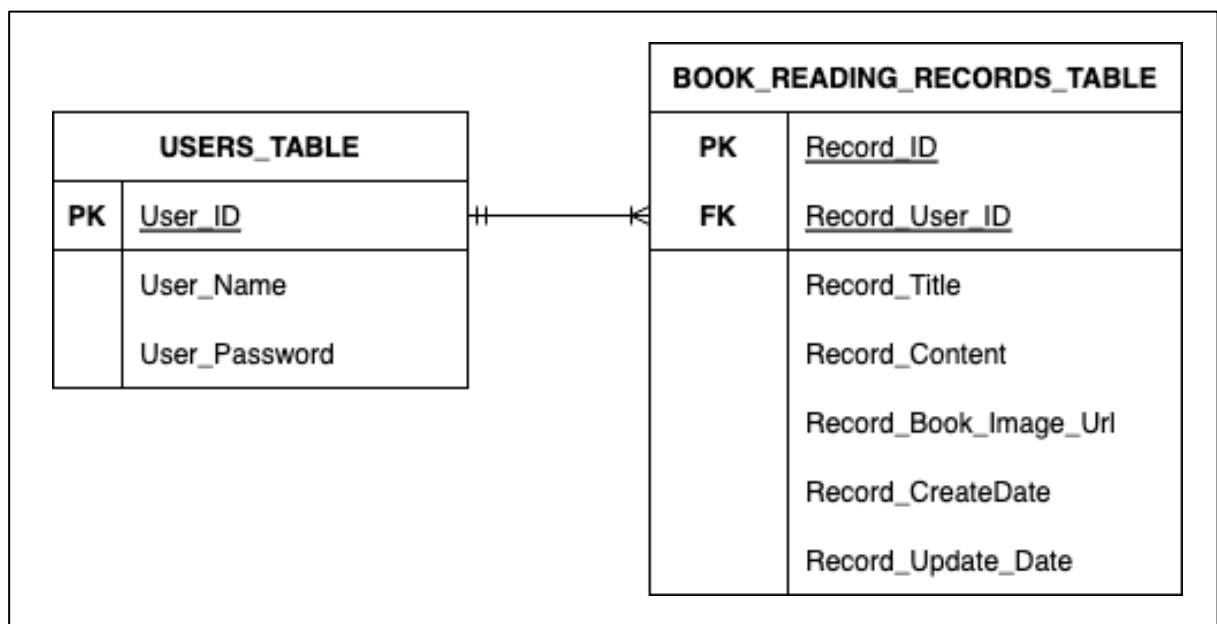


図 2 書籍管理 Web アプリ用データベースの ER 図

## 7. 画面設計

### 7.1 ログイン画面

ログイン画面は、本システム利用者用のエントリーポイントである。パスワードは「\*\*\*\*\*」のように隠される。ログインボタンを押下することで USERS\_TABLE から入力されたユーザー情報を検索し該当する場合登録済み書籍一覧画面へ遷移する。

### Login Page

ユーザー名

パスワード

ログイン

図 3 ログイン画面

### 7.2 登録済み書籍一覧画面

登録済み書籍一覧画面は、登録されているスケジュールをカード形式で表示する。登録した書籍の表紙とタイトル、編集ボタン、削除ボタンを1セットとしている。編集ボタンを押下することで記録画面へ遷移する。削除ボタンを押下することで登録済み書籍を1件分削除する。表紙を表示することでデザイン性を上げた。

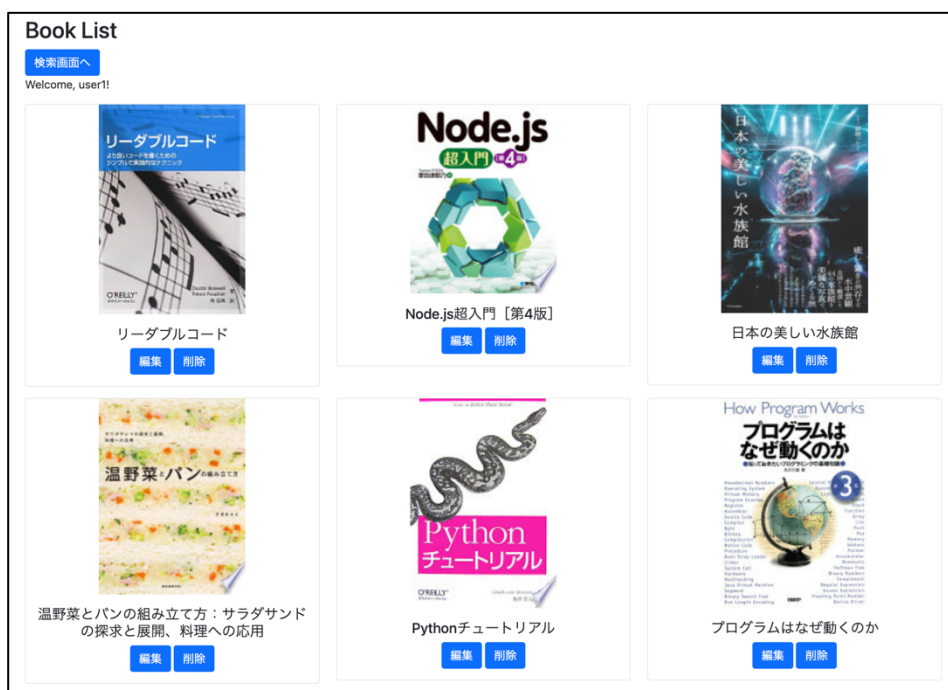


図 4 登録済み書籍一覧画面(テストデータ 6 件登録済み)

## 7.3 書籍検索画面

書籍検索画面は、タイトルを入力して検索する。最大で 40 個の書籍データを表示する。取得情報は、GoogleBooksAPI の登録データに依存するため、表紙データがない場合がある。登録済み書籍一覧画面へボタンを押下することで登録済み書籍一覧画面へ遷移する。

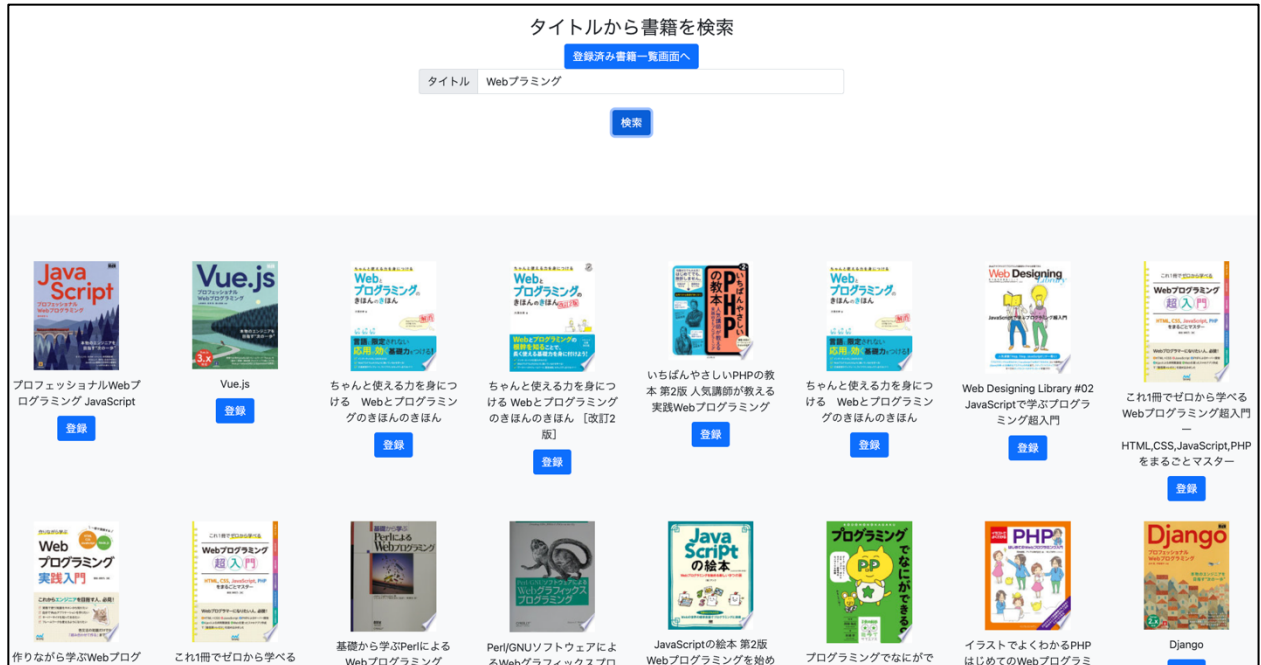


図 5 書籍検索画面(「Web プログラミング」で検索)

## 7.4 記録画面

記録画面は、登録済みの書籍の記録内容の変更を行う。更新ボタンを押すことで入力した記録が保存され、表示されている Markdown の内容も更新される。戻るボタンを押下することで登録済み書籍一覧画面へ遷移する。

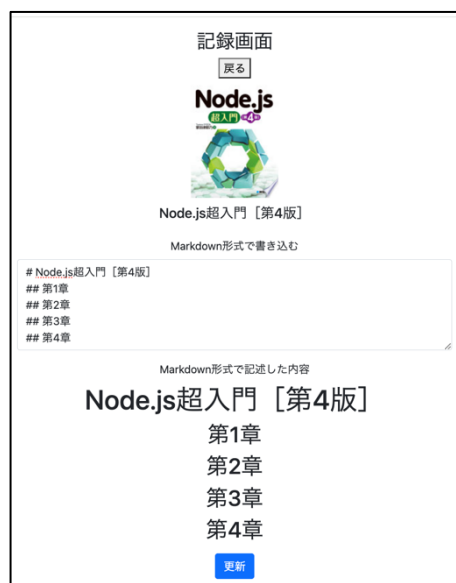


図 6 記録画面

## 8. 実装

### 8.1 実装環境

実装環境を以下に示す。

- MacBook Air M1(2020) •チップ:Apple M1 2020 •メモリ:16GB
- OS:macOS Sonoma バージョン 14.2.1 •Visual Studio Code 1.85.2 (Universal)

### 8.2 環境設定

環境設定を以下に示す。

- Node.js 21.5.0 •express 4.17.1 •prisma 5.8.1
- chrome Version 121.0.6167.85 (Official Build) (arm64)

## 9. 工夫点

プログラムの保守性を上げるためにソースコードを機能別にフォルダで分けた。views フォルダ、routes フォルダ、controllers フォルダのように機能別にソースファイルを分割したことで不具合箇所の特定までの時間を短縮することができた。また、機能追加を容易に追加できるようにした。書籍データを取得できる API には、国立国会図書館サーチ API や図書館 API などがあったが GoogleBooksAPI は json データで取得が可能かつ API キーなしで URL を発行することで書籍情報が取得できる。また、URL 形式の書籍表紙データ取得できるため、実装の容易さと表紙データが決め手となり本システムに採用した。書籍を一括管理する上でタイトルのみのリストデータでは見栄えが悪く感じたため表紙を表示することにした。

## 10. 反省点

反省点として、機能の不足が挙げられる。今回作成したシステムは、私個人が読書記録をつけることを想定して実装している。そのため、不特定多数の利用に対応できるブラウザ環境で動作するメリットが薄くなっている。また、保存する書籍データが少ないと感じた。ISBN コードや著者、出版社など書籍の情報は利用者によって必要な情報が変わる。本システムは、利用者のニーズを想定した設計が足りないと言える。今後は、機能追加などを行い実用に耐えうるシステムとして改良していきたい。

## 11. 参考文献

- 掌田津耶乃,2023 年 7 月 15 日,「Node.js 超入門第 4 版」,株式会社秀和システム,p.306～448
- 「Prisma/ドキュメント/SQLite」(<https://www.prisma.io/docs/orm/overview/databases/sqlite>)(閲覧日:2024 年 1 月 20 日)
- 「JavaScript で Google Books APIs 使って本を検索して結果一覧を表示するツールを作ってみた」(<https://zenn.dev/rabee/articles/javascript-tool-google-books-api>)(閲覧日:2024 年 1 月 25 日)
- 「【JavaScript】Google Books API を使って書籍検索アプリを作成」(<https://qiita.com/hukuryo/items/b6797549a48ad5cdfe74>)(閲覧日:2024 年 1 月 25 日)