

Library Management System for Group-Based Student Borrowing

Temitope Oyemade

<https://temitopejoshua.github.io/db-personal-app/>

Department of Computer Science

Bowie State University

1. Executive Summary

This document outlines the design, development, and deployment of a Library Management System tailored to manage book borrowing and return processes in a classroom setting. The system addresses the needs of 78 library books, a class of students categorized into groups, and their specific borrowing preferences. It incorporates features such as borrowing status, return tracking, group membership identification, and book ratings.

2. Problem Definition

The challenge is to design a database system for a class with students divided into four groups (A-D), each consisting of three members. Each student has three preferred books from a longlist of 78 books. The system should manage book borrowing, return dates, and due notifications. The database must allow searching by student name to display borrowed books and corresponding due dates. Additional features include average book rating calculations and listing books by rating performance.

3. Requirements

- Functional Requirements:
 - Book Management: Create and manage a database of 78 books with relevant details (title, author, rating, etc).
 - Student Management: Store student information, including names, groups, and book preferences.
 - Borrowing Records: Track borrowing status, due dates, and return information.
 - Queries: Users can search by student name, group membership, and book availability.
 - Reporting: Display books with above and below-average ratings.
- Non-Functional Requirements:
 - Usability: User-friendly interface for queries.
 - Accessibility: Hosted on GitHub Pages.

4. Proposed Solution

We propose a Library Management System with the following features:

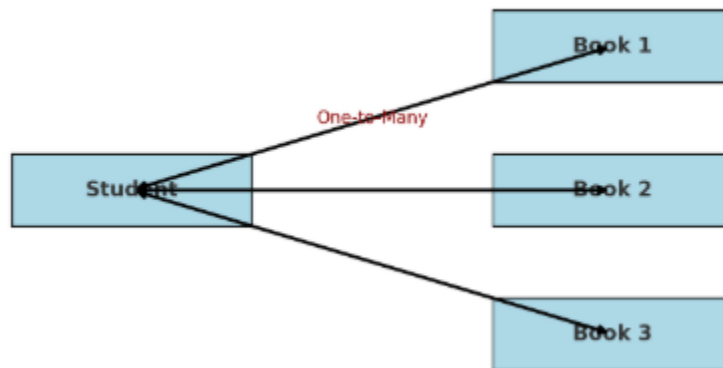
- A relational database to manage books, students, borrowing transactions, and group memberships.
- A user interface (UI) for querying borrowing details and viewing group information.
- A backend system to calculate average ratings and display books above or below the average.
- A method to dynamically update and query the database.

The system will be developed using Java for the backend, MySQL for the database, and a web interface hosted on GitHub Pages.

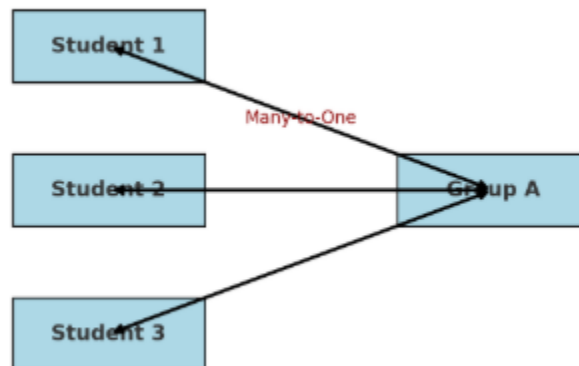
5. Design and Methodologies

5.1. Database Design

- Entities:
 - Books (BookID, Title, Author, Rating, BorrowedStatus)
 - Students (StudentID, FirstName, LastName, Group, PreferredBooks)
 - Borrowing (TransactionID, BookID, StudentID, BorrowDate, ReturnDate)
 - Groups Table: (group_id, group_name)
- Relationships and Entity-Relationship Diagram (ERD):
 - One-to-Many: A student can borrow multiple books.



- Many-to-One: Multiple students (3) belong to a group.



Normalization: Ensuring no data redundancy and efficient querying by dividing the database into normalized tables.

- SQL Queries:
 - Query for borrowed books and return status:

SELECT * FROM books WHERE status = 1;

Local instance 3306 - Warning - not supported

1 + use library;
2 + Checked out books
3 + SELECT * FROM books WHERE status = 1;

100% 0 38.2

Result Grid Filter Rows Q Limit to 1000 rows

id	isbn	title	publisher_id	format	pages	published	year	status
1	9780459798867	Doublet	18	paperback	112	2003-08-02 00:00:00	2003	1
2	9781028071338	Wings	3	paperback	368	2003-07-16 00:00:00	2003	1
3	9781945891181	The Cheaper According to the New World	32	paperback	114	2003-03-02 00:00:00	2003	1
4	9781714502020	Time Stalker	38	hardcover	304	2002-04-27 00:00:00	2002	1
5	9781945406555	The Birthday Party	27	paperback	464	2003-01-04 00:00:00	2003	1
6	9781762776827	Pyra	17	paperback	394	2003-04-06 00:00:00	2003	1
7	9781913091963	Get Born	5	paperback	300	2002-05-22 00:00:00	2002	1
8	9781913091870	Headless	5	paperback	178	2002-03-03 00:00:00	2002	1
9	978190886051	Heaven	16	paperback	147	2002-05-12 00:00:00	2002	1
10	9781911264858	Love in the Big City	38	paperback	217	2001-10-26 00:00:00	2002	1
11	9781911264858	Master Books, Beverly	38	paperback	183	2001-12-02 00:00:00	2002	1
12	9781908138519	The Book of Esther	31	paperback	354	2001-09-19 00:00:00	2002	1
13	9781767930535	More Than I Love My Life	9	paperback	368	2001-08-26 00:00:00	2002	1
14	9781913505189	Phenotypes	1	paperback	292	2002-01-01 00:00:00	2002	1
15	9781913507721	A New Name: Sepulchre Writ	8	paperback	308	2001-09-08 00:00:00	2002	1
16	9781911264811	Temp of Fate	38	paperback	128	2001-09-12 00:00:00	2002	1
17	9781911662553	The Books of Jacob	35	paperback	340	2001-11-15 00:00:00	2002	1
18	9781918277182	Curved Bones	8	paperback	261	2001-06-15 00:00:00	2002	1
19	9781910806838	The War of the Post	18	hardcover	36	2001-07-02 00:00:00	2002	1
20	9781762776738	When We Cease to Understand the W	13	hardcover	166	2000-04-03 00:00:00	2001	1
21	9781925400793	An Inventory of Losses	12	hardcover	266	2000-08-20 00:00:00	2001	1
22	9781762770862	All Night All Blood & Black	17	hardcover	182	2000-11-03 00:00:00	2001	1
23	978192547435	Live in the Future	38	hardcover	341	2000-09-19 00:00:00	2001	1
24	9781913507170	Secret Detail	5	paperback	144	2000-05-08 00:00:00	2001	1
25	9781911395733	Summer Brother	32	paperback	385	2001-04-08 00:00:00	2001	1
26	9781908810601	The Pear Field	14	paperback	163	2000-10-30 00:00:00	2001	1
27	9781908810601	The Perfect Home	35	hardcover	340	2000-10-30 00:00:00	2001	1
28	9781911340044	The Discomfort of Evening	4	paperback	160	2000-03-03 00:00:00	2000	1
29	9781908400661	The Solifament of The Greenhouse	3	paperback	328	2000-07-17 00:00:00	2000	1

books 13

Action Output

Time	Action	Response	Duration / Fetch Time
2020-07-11 10:07:55	SELECT * FROM books WHERE status = 1 LIMIT 0, 1000	43 rows returned	0.0013 sec / 0.000000

SELECT * FROM books WHERE status = 0;

1 + use library;
2 + Checked out books
3 + SELECT * FROM books WHERE status = 0;

100% 0 38.2

Result Grid Filter Rows Q Limit to 1000 rows

id	isbn	title	publisher_id	format	pages	published	year	status
4	9781928144091	Standing Heavy	12	paperback	282	2002-05-20 00:00:00	2002	0
5	9781928174318	Is Mother Dead	28	paperback	380	2002-09-27 00:00:00	2002	0
6	9781928174318	Am Harvill, Live It Life	12	hardcover	418	2002-04-27 00:00:00	2002	0
7	9781908471088	White Ice Vows Dreaming	5	paperback	368	2002-02-20 00:00:00	2002	0
8	9781911484872	A System to Magnificent & to Binding	20	paperback	344	2002-03-09 00:00:00	2002	0
9	9781768255007	North Building	8	paperback	272	2002-05-16 00:00:00	2002	0
10	9781908810601	Uranic Avenue	12	paperback	143	2001-07-07 00:00:00	2002	0
11	9781908810601	After the Sun	11	paperback	118	2001-08-31 00:00:00	2002	0
12	9781911506782	Watchwords	1	paperback	178	2002-04-02 00:00:00	2002	0
13	9781913505030	In Memory of Memory	5	paperback	610	2001-09-07 00:00:00	2002	0
14	9781908810601	The Gorgias of Breathing in Red	6	paperback	187	2001-04-01 00:00:00	2002	0
15	9781908810601	The Empress	11	paperback	138	2002-10-31 00:00:00	2002	0
16	9781918400661	The Adventures of China Iron	2	paperback	290	2018-11-14 00:00:00	2018	0
17	9781911507484	The Light Life	39	hardcover	344	2018-11-14 00:00:00	2018	0
18	9781911506913	The City Name: Sepulchre L&	5	paperback	383	2018-10-10 00:00:00	2018	0
19	9781908132208	Sermon	35	hardcover	380	2018-09-26 00:00:00	2018	0
20	9781911506928	The Hermit	1	paperback	180	2018-10-04 00:00:00	2018	0
21	9781911506928	The Years	15	paperback	234	2018-09-26 00:00:00	2018	0
22	9781908400661	The Stone Skye	24	paperback	418	2017-08-26 00:00:00	2018	0
23	9781908400661	The White Book	18	paperback	181	2018-04-26 00:00:00	2018	0
24	9781908400661	The World Lives On	28	paperback	320	2018-05-01 00:00:00	2018	0
25	9781908400661	On My Love	3	paperback	132	2017-09-14 00:00:00	2018	0

books 14

Action Output

Time	Action	Response	Duration / Fetch Time
2020-07-11 10:08:07	SELECT * FROM books WHERE status = 0 LIMIT 0, 1000	27 rows returned	0.0012 sec / 0.000000

- Query for group affiliations and preferred books.

```
CREATE TABLE `groups` (  
  `id` int NOT NULL AUTO_INCREMENT,  
  `name` varchar(45) DEFAULT NULL,  
  PRIMARY KEY (`id`),  
  UNIQUE KEY `id_UNIQUE` (`id`)  
) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=utf8mb4  
COLLATE=utf8mb4_0900_ai_ci  
SELECT count(sg.student_id) as no_of_students_in_group, g.name FROM student_group sg,  
library.groups g  
WHERE sg.group_id = g.id GROUP BY g.name;
```

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane displays a tree view of the database structure, including tables like 'book_groups', 'book_rating', 'books', 'groups', 'student_group', and 'students'. The 'groups' table is selected. The main editor window shows the following SQL query:

```
1. use library;  
2.  
3. -- Groups  
4. SELECT * FROM student_group;  
5.  
6. SELECT count(sg.student_id) as no_of_students_in_group, g.name FROM student_group sg, library.groups g  
7. WHERE sg.group_id = g.id GROUP BY g.name;  
8.
```

Below the query editor, the 'Result Grid' shows the results of the query. The grid has two columns: 'no_of_students_in_group' and 'name'. The results are as follows:

no_of_students_in_group	name
7	GROUP A
8	GROUP B
8	GROUP C
8	GROUP D

At the bottom, the 'Action Output' pane shows the execution details for the query, including the duration and fetch time for each row.

SELECT book_id,b.isbn, b.title,g.name as group_name FROM book_groups bg, books b,
library.groups g
WHERE bg.book_id = b.id AND bg.group_id = g.id;

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'Schemas' tree with the 'library' database selected. The main editor shows a SQL query: `use library;` followed by a comment `-- Your code should show which books for each group;` and the query `SELECT book_id,b.isbn, b.title,g.name as group_name FROM book_groups bg, books b, library.groups g WHERE bg.book_id = b.id AND bg.group_id = g.id;`. The bottom panel shows the 'Result Grid' with 32 rows and 5 columns: book_id, isbn, title, group_id, and group_name. The results are filtered to show only books belonging to 'GROUP A'.

book_id	isbn	title	group_id	group_name
81	9780261064722	The Faculty of Dreams	GROUP A	
85	9781315895784	The Years	GROUP A	
88	9781910701664	The 7th Function of Lan...	GROUP A	
73	9781825486854	The Silent Skyline	GROUP A	
77	9781899122286	Do! My Love	GROUP A	
15	9781252711238	Whales	GROUP A	
16	9781839164218	In Mother Dead	GROUP A	
110	9781782278627	Pyne	GROUP A	
114	9781813581816	Parasite	GROUP A	
118	9781893364532	China Knows	GROUP A	
23	9781913067791	A New Name: Septology	GROUP A	
26	9781318277182	Cursed Bunny	GROUP A	
31	9781319410019	An Inventory of Lies	GROUP A	
34	9781313591712	Minor Detail	GROUP A	
38	9781904871480	The Pear Field	GROUP A	
42	9781318174454	The Eighth Life	GROUP A	
98	9781529403827	Typ	GROUP A	
93	9781747305453	Mac and the Problem	GROUP A	
54	9781912340166	Celestial Bodies	GROUP A	
58	9780300224513	Love in the New Millenn...	GROUP A	
82	9781788160619	The Pine Islands	GROUP A	
68	9781786016609	Frankenstein in Baghdad	GROUP A	
70	9781910701880	The Dinner Guest	GROUP A	
74	978025572067	The White Book	GROUP A	
78	9781915884432	Flights	GROUP A	
3	9781842861131	The Gospel According to...	GROUP C	
7	9781529427630	Jani Hendrix Live in Liv...	GROUP C	

SELECT book_id,b.isbn, b.title,g.name as group_name FROM book_groups bg, books b, library.groups g
WHERE bg.book_id = b.id AND bg.group_id = g.id AND g.name = 'GROUP A';

Local Instance SQL - Warning - not supported

Administration Schemas book book_groups

SCHEMAS

- filter objects
- db
- db_class
- db_class_hw2
- library
 - Tables
 - book_groups
 - book_rating
 - books
 - groups
 - student_group
 - students
 - Views
 - Stored Procedures
 - Functions
- sys

```

1 use library;
2
3 -- You code should later on add user input so that if user input Group A it should produce books belong to A, or B, or C or D
4
5 SELECT book_id, b.isbn, b.title, g.name as group_name FROM book_groups bg, books b, library_groups g
6 WHERE bg.book_id = b.id AND bg.group_id = g.id AND g.name = 'GROUP A';
7
8
9

```

100% 1/7

Result Grid Filter Rows Export

book_id	isbn	title	group_name
1	070802770007	Booker	GROUP A
2	070817062305	Time Machine	GROUP A
3	07081802770008	White We Were Dreaming	GROUP A
4	07081706230507	Python 3.0 Intro	GROUP A
5	07081812840009	Angry Women, Angry	GROUP A
6	07081813001109	Phenomena	GROUP A
7	07081812000509	The Book of Job	GROUP A
8	07081811000707	Whitehead's	GROUP A
9	07081812007000	In Memory of Memory	GROUP A
10	07081810000009	The Endgame	GROUP A
11	07080713430004	The Descent of Evening	GROUP A
12	07081810000013	The Other Name, Secret	GROUP A
13	07081700070009	Julia Lynn	GROUP A
14	07081811070005	At Day	GROUP A
15	07081810000015	James for the Remnant	GROUP A
16	07080610004700	The Faculty of Dreams	GROUP A
17	07081810000009	The Name	GROUP A
18	07081810000004	The New	GROUP A
19	07081810000004	The New Function of Love	GROUP A
20	07081810000004	The Book of Job	GROUP A
21	07081810000004	One, My Love	GROUP A

Result 00 Read Only

Local Instance SQL - Warning - not supported

Administration Schemas book book_groups

SCHEMAS

- filter objects
- db
- db_class
- db_class_hw2
- library
 - Tables
 - book_groups
 - book_rating
 - books
 - groups
 - student_group
 - students
 - Views
 - Stored Procedures
 - Functions
- sys

```

1 use library;
2
3 -- You code should later on add user input so that if user input Group A it should produce books belong to A, or B, or C or D
4
5 SELECT book_id, b.isbn, b.title, g.name as group_name FROM book_groups bg, books b, library_groups g
6 WHERE bg.book_id = b.id AND bg.group_id = g.id AND g.name = 'GROUP B';
7
8
9

```

100% 71/4

Result Grid Filter Rows Export

book_id	isbn	title	group_name
3	07080800070008	White	GROUP B
4	07080800070008	In Mother David	GROUP B
5	07081802000007	Pige	GROUP B
6	07081810000009	Phenomena	GROUP B
7	07081800000000	China Kines	GROUP B
8	07081810000009	A New Name, Secretly 30-03	GROUP B
9	07081810000009	Curved Runny	GROUP B
10	07081810000009	An Treasury of Losses	GROUP B
11	07081810000009	New Land	GROUP B
12	07081810000009	The Past Past	GROUP B
13	07081810000009	The Right Life	GROUP B
14	07081810000009	Ty	GROUP B
15	07081810000009	Mac and His Problem	GROUP B
16	07081810000009	Conductor Book	GROUP B
17	07081810000009	Love is the New Millennium	GROUP B
18	07081810000009	The First World	GROUP B
19	07081810000009	Phenomena in Booked	GROUP B
20	07081810000009	The Outer World	GROUP B
21	07081810000009	The White Book	GROUP B
22	07081810000009	Agnes	GROUP B

Result 01 Read Only

Local Instance 3306 - Warning - not supported

Administration Schemas book_groups

1 use library;
2
3 -- You code should later on add user input so that if user input Group A it should produce books belong to A, or B, or C or D.
4
5 SELECT book_id, b.isbn, b.title, g.name as group_name FROM book_groups bg, books b, library_groups g
6 WHERE bg.book_id = b.id AND bg.group_id = g.id AND g.name = 'GROUP C';
7
8

100% 18 Filter Rows Export

book_id	isbn	title	group_name
1	978-0438611811	The Grapes According to the New World	GROUP C
7	978-0102942782	Jane Heredia Live it Live	GROUP C
11	978-0153076807	Old Fern	GROUP C
13	978-0109848257	Heaven	GROUP C
18	978-0821581980	The Book of Esther	GROUP C
25	978-0699636502	After the Sun	GROUP C
27	978-0696366306	The Way of Zen Poet	GROUP C
31	978-0783275682	At Night All Blood is Black	GROUP C
36	978-012687139	Summer Brother	GROUP C
38	978-0602079511	The Perfect Nine	GROUP C
43	978-0644444617	The Enlightenment of The Greenyago	GROUP C
44	978-0644444617	Paras on the Top of My Tongue	GROUP C
47	978-032079222	Hell Log	GROUP C
58	978-013983715	Drive Your Paw Over the Bones of the	GROUP C
59	978-0132014582	NEARLY 27	GROUP C
20	978-011908329	The Harlequin	GROUP C
87	978-0445275202	Go, West, Go	GROUP C
71	9780841424127	The Flying Machine	GROUP C
74	978-0786180124	The World Goes On	GROUP C

Result 52

Local Instance 3306 - Warning - not supported

Administration Schemas book_groups

1 use library;
2
3 -- You code should later on add user input so that if user input Group A it should produce books belong to A, or B, or C or D.
4
5 SELECT book_id, b.isbn, b.title, g.name as group_name FROM book_groups bg, books b, library_groups g
6 WHERE bg.book_id = b.id AND bg.group_id = g.id AND g.name = 'GROUP D';
7
8

100% 714 Filter Rows Export

book_id	isbn	title	group_name
1	978-0438611811	Standing Heavy	GROUP D
8	978-0104928285	The Sifted Peas	GROUP D
12	978-0146848472	A System On Mayflower it is landing	GROUP D
16	978-0191284866	Love in the Big City	GROUP D
28	978-0176232835	More Than I Love My Life	GROUP D
29	978-0191284811	Bank of Soul	GROUP D
35	978-0176232918	When We Learn to Understand the World	GROUP D
37	978-0302947436	I Live in the Shark	GROUP D
38	978-0192788749	The Dangers of Smoking in Bed	GROUP D
41	978-0191646841	The Adventures of Chick & Bob	GROUP D
44	978-0176760070	The Morning Police	GROUP D
46	978-0191287197	The Purple Mirror	GROUP D
50	978-0176816238	Sanctuary	GROUP D
56	978-0348578827	Four Brothers	GROUP D
60	978-0191134488	The Book of Moral Stories	GROUP D
64	978-0607060580	The Shape of the Moon	GROUP D
66	978-0176768817	Like a Flying Shadow	GROUP D
72	9780807948804	The Invisible	GROUP D
76	9780807954482	Vampire Sublime 1	GROUP D

Result 53

- Query for book ratings above or below average.

Books rated higher than the median rate, median rate = 3

SELECT b.id, b.isbn, b.title, rating FROM book_rating br, books b WHERE br.book_id = b.id
AND br.rating > 3;

1 use Library;

2

3 -- Books rated higher than the median rate, median rate = 3

4 SELECT book_id, b.isbn, b.title, rating FROM book_rating br, books b WHERE br.book_id = b.id AND br.rating > 3

5 ORDER BY br.rating ASC;

book_id	isbn	title	rating
29	9781620975251	The Perfect Nine	4
3	9781642861161	The Gospel According to the New World	4
6	9781935174318	To Mother, I Said	4
7	9781529427820	Joni Handia Live in Live	4
8	9781544493655	The Birthday Party	4
9	9781854270258	When We Were Dreaming	4
10	9781782278627	Pyre	4
76	9780887055422	Vernon Scholtes 1	4
74	9780525573667	The White Book	4
14	9781911891875	Paradise	4
18	9781999368432	Elena Knows	4
19	9781982104786	The Book of Mother	4
20	9781781332835	More Than I Love My Life	4
23	9781625448654	The Silver Spade	4
24	9781911264811	Tomb of Sand	4
72	9780887056504	The Imposter	4
21	9780817424297	The Winding Mountain	4
30	9781529450793	An Inventory of Losses	4
21	9781782275862	At Night All Blood Is Black	4
30	9780300247435	I Live in the Storm	4
23	9781911891830	In Memory of Memory	4
34	9781911891772	Minor Detail	4
25	9781812867138	Sunrise Brother	4
37	9781999498865	The Synagogue	4
38	9781629425801	The Four Seasons	4
77	9781999722764	Da, My Love	4
41	178057134884	The Discomfort of Evening	4
42	9781918174854	The Dying Life	4
44	9781787300700	The Memory Palace	4

Result 19

Action Output

Time Action Response Duration / Fetch Time

23 20:11:54 SELECT book_id, b.isbn, b.title, rating FROM book_rating br, books b WHERE br.book_id = b.id AND br.rating > 3 30 row(s) returned 0.00061 sec / 0.000...

Books rated lower than the median rate, median rate = 3 SELECT b.id,b.isbn, b.title, rating FROM book_rating br, books b WHERE br.book_id = b.id AND br.rating < 3;

1 use Library;

2

3 -- Books rated lower than the median rate, median rate = 3

4 SELECT book_id, b.isbn, b.title, rating FROM book_rating br, books b WHERE br.book_id = b.id AND br.rating < 3

5 ORDER BY br.rating ASC;

book_id	isbn	title	rating
76	978180956432	Right	2

Result 25

Action Output

Time Action Response Duration / Fetch Time

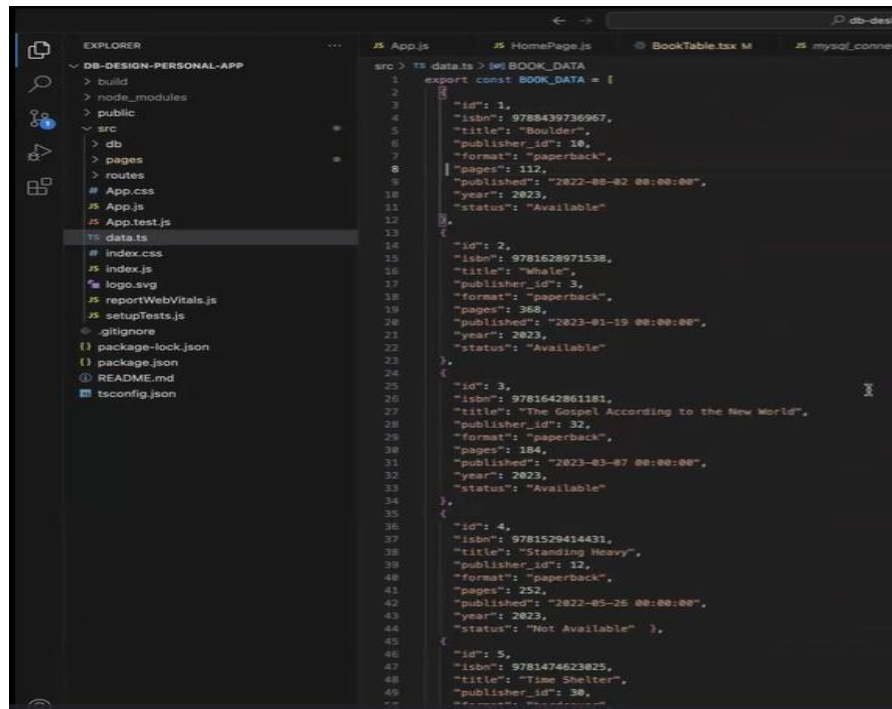
24 20:12:55 SELECT book_id, b.isbn, b.title, rating FROM book_rating br, books b WHERE br.book_id = b.id AND br.rating < 3 1 row(s) returned 0.00060 sec / 0.000...

- Sequelize models: These are models to import the codes from MySQL database
 - Student data Model:


```
db-design-personal-app
src > pages > StudentsTable.tsx > ...
1 import * as React from 'react';
2 import Paper from '@mui/material/Paper';
3 import Table from '@mui/material/Table';
4 import TableBody from '@mui/material/TableBody';
5 import TableCell from '@mui/material/TableCell';
6 import TableContainer from '@mui/material/TableContainer';
7 import TableHead from '@mui/material/TableHead';
8 import TablePagination from '@mui/material/TablePagination';
9 import TableRow from '@mui/material/TableRow';
10 import TextField from '@mui/material/TextField';
11 import {STUDENT_DATA} from '../data';
12
13 interface Column {
14   id: string;
15   label: string;
16   minWidth?: number;
17   align?: 'right';
18   format?: (value: any) => string;
19 }
20
21 const columns: readonly Column[] = [
22   { id: 'id', label: 'ID', minWidth: 50 },
23   { id: 'first_name', label: 'First Name', minWidth: 150 },
24   { id: 'last_name', label: 'Last Name', minWidth: 200 },
25 ];
26
27 export default function StudentsDataTable() {
28   const [rows, setRows] = React.useState<any[]>([]);
29   const [filteredRows, setFilteredRows] = React.useState<any[]>([]);
30   const [search, setSearch] = React.useState('');
31   const [page, setPage] = React.useState(0);
32   const [rowsPerPage, setRowsPerPage] = React.useState(10);
33
34   React.useEffect(() => {
35     const loadCSV = async () => {
36       setRows(STUDENT_DATA);
37       setFilteredRows(STUDENT_DATA);
38     };
39     loadCSV();
40   }, []);
41
42   const handleChangePage = (event: unknown, newPage: number) => {
43     setPage(newPage);
44   };
45
46   const handleChangeRowsPerPage = (event: React.ChangeEvent<HTMLInputElement>) => {
47     setRowsPerPage(event.target.value);
48     setPage(0);
49   };
50 }
```

```
db-design-personal-app
src > pages > StudentsTable.tsx > ...
27 export default function StudentsDataTable() {
58
59   const handleSearch = (event: React.ChangeEvent<HTMLInputElement>) => {
60     const value = event.target.value.toLowerCase();
61     setSearch(value);
62     setFilteredRows(rows.filter(row =>
63       row.first_name.toLowerCase().includes(value) ||
64       row.last_name.toLowerCase().includes(value)
65     ));
66   };
67
68   return (
69     <div>
70       <h2>Student Data</h2>
71       <TextField
72         label="Search by first/last name"
73         variant="outlined"
74         value={search}
75         onChange={handleSearch}
76         sx={{ marginBottom: 2 }}
77       />
78       <Paper sx={{ width: '100%', overflow: 'hidden' }}>
79         <TableContainer sx={{ maxHeight: 440 }}>
80           <Table stickyHeader aria-label="sticky table">
81             <TableHead>
82               <TableRow>
83                 {columns.map((column) => (
84                   <TableCell
85                     key={column.id}
86                     align={column.align}
87                     style={{ minWidth: column.minWidth }}
88                   >
89                     {column.label}
90                   </TableCell>
91                 ))}
92             </TableHead>
93             <TableBody>
94               {filteredRows
95                 .slice(page * rowsPerPage, page * rowsPerPage + rowsPerPage)
96                 .map((row, index) => (
97                   <TableRow hover role="checkbox" tabIndex={-1} key={index}>
98                     {columns.map((column) => (
99                       const value = row[column.id];
100                       return (
101                         <TableCell key={column.id} align={column.align}>
102                           {column.format ? column.format(value) : value}
103                         </TableCell>
104                       );
105                     ))}
106                   </TableRow>
107                 ))}
108             </TableBody>
109           </Table>
110         </TableContainer>
111       </Paper>
112     </div>
113   );
114 }
```

- Book data Model:



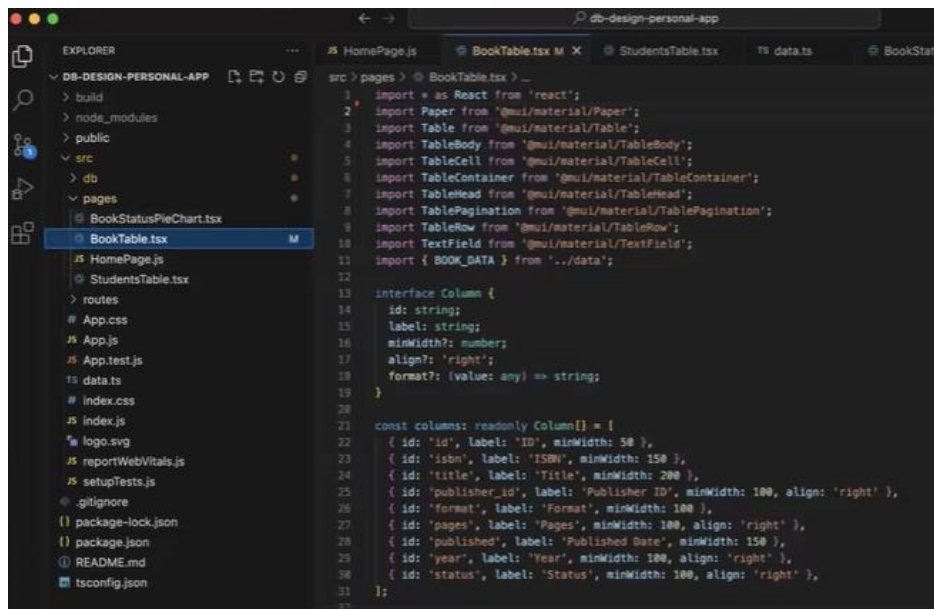
The screenshot shows a VS Code editor with the Explorer sidebar on the left displaying the project structure. The main editor window shows the `App.js` file with the following code:

```

1  export const BOOK_DATA = [
2    {
3      "id": 1,
4      "isbn": 9788439736967,
5      "title": "Boulder",
6      "publisher_id": 10,
7      "format": "paperback",
8      "pages": 112,
9      "published": "2022-08-02 00:00:00",
10     "year": 2023,
11     "status": "Available"
12   },
13   {
14     "id": 2,
15     "isbn": 9781628971538,
16     "title": "Whale",
17     "publisher_id": 3,
18     "format": "paperback",
19     "pages": 368,
20     "published": "2023-01-19 00:00:00",
21     "year": 2023,
22     "status": "Available"
23   },
24   {
25     "id": 3,
26     "isbn": 9781642861181,
27     "title": "The Gospel According to the New World",
28     "publisher_id": 32,
29     "format": "paperback",
30     "pages": 184,
31     "published": "2023-03-07 00:00:00",
32     "year": 2023,
33     "status": "Available"
34   },
35   {
36     "id": 4,
37     "isbn": 9781529414431,
38     "title": "Standing Heavy",
39     "publisher_id": 12,
40     "format": "paperback",
41     "pages": 252,
42     "published": "2022-05-26 00:00:00",
43     "year": 2023,
44     "status": "Not Available"
45   },
46   {
47     "id": 5,
48     "isbn": 9781474623025,
49     "title": "Time Shelter",
50     "publisher_id": 30,
51     "format": "paperback",
52     "pages": 112,
53     "published": "2022-08-02 00:00:00",
54     "year": 2023,
55     "status": "Available"
56   }
57 ]

```

- Book Table Model:



The screenshot shows a VS Code editor with the Explorer sidebar on the left displaying the project structure. The main editor window shows the `BookTable.tsx` file with the following code:

```

1  import * as React from 'react';
2  import Paper from '@mui/material/Paper';
3  import Table from '@mui/material/Table';
4  import TableBody from '@mui/material/TableBody';
5  import TableCell from '@mui/material/TableCell';
6  import TableContainer from '@mui/material/TableContainer';
7  import TableHead from '@mui/material/TableHead';
8  import TablePagination from '@mui/material/TablePagination';
9  import TableRow from '@mui/material/TableRow';
10 import TextField from '@mui/material/TextField';
11 import { BOOK_DATA } from '../data';
12
13 interface Column {
14   id: string;
15   label: string;
16   minWidth?: number;
17   align?: 'right';
18   format?: (value: any) => string;
19 }
20
21 const columns: readonly Column[] = [
22   { id: 'id', label: 'ID', minWidth: 50 },
23   { id: 'isbn', label: 'ISBN', minWidth: 150 },
24   { id: 'title', label: 'Title', minWidth: 200 },
25   { id: 'publisher_id', label: 'Publisher ID', minWidth: 100, align: 'right' },
26   { id: 'format', label: 'Format', minWidth: 100 },
27   { id: 'pages', label: 'Pages', minWidth: 100, align: 'right' },
28   { id: 'published', label: 'Published Date', minWidth: 150 },
29   { id: 'year', label: 'Year', minWidth: 100, align: 'right' },
30   { id: 'status', label: 'Status', minWidth: 100, align: 'right' },
31 ];

```

- Database Connector:

```

src > db > sequelize-project > mysql_connector.js
1  const { Sequelize } = require('sequelize');
2
3
4  const dbConnector = new Sequelize({
5    database: 'library',
6    username: 'capricorn',
7    password: 'welcome123',
8    host: 'localhost',
9    port: 3386,
10   dialect: 'mysql',
11 })
12
13 dbConnector.authenticate().then(() => {
14   console.log('Connected to the database successfully')
15 }).catch(error => console.log('An error occured when trying to connect to the database ' + error))
16
17 module.exports = {
18   dbConnector
19 }

```

- Routers: These were used to connect the front-end our backend database
 - React: The front end was built with React

```

src > index.js
1  import React from 'react';
2  import ReactDOM from 'react-dom/client';
3  import './index.css';
4  import App from './App';
5  import reportWebVitals from './reportWebVitals';
6
7
8  const root = ReactDOM.createRoot(document.getElementById('root'));
9  root.render(
10   <React.StrictMode>
11     <App />
12   </React.StrictMode>
13 );
14
15 // If you want to start measuring performance in your app, pass a function
16 // to log results (for example: reportWebVitals(console.log))
17 // or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
18 reportWebVitals();
19

```

- HashRoute: HashRoute was imported for connectivity with GitHub

```

src > routes > NavigationRoutes.js
1  import React from 'react';
2  import { Router, Routes } from 'react-router-dom';
3  import HomePage from '../pages/HomePage';
4  import StickyHeadTable from '../pages/BookTable';
5  import StudentsDataTable from '../pages/StudentsTable';
6  import BookStatusPieChart from '../pages/BookStatusPieChart';
7  import { HashRouter } from 'react-router-dom';
8
9
10 function RootNavigation() {
11   return (
12     <HashRouter>
13       <Routes>
14         <Route path="/" element={<HomePage/>} />
15         <Route path="/book-data" element={<StickyHeadTable/>} />
16         <Route path="/student-data" element={<StudentsDataTable/>} />
17         <Route path="/book-status-analysis" element={<BookStatusPieChart/>} />
18       </Routes>
19     </HashRouter>
20   );
21 }
22
23 export default RootNavigation;
24

```

5.2. Front-end Demonstration

Click the link <https://temitopejoshua.github.io/db-personal-app/> to access the front-end of the database through GitHub.

temitopejoshua.github.io/db-personal-app/#/login

Login

Email

Password

Login

Enter email and password

temitopejoshua.github.io/db-personal-app/#/login

Login

Email

Password

Login

If the correct email and password are entered, access will be granted.

temitopejoshua.github.io/db-personal-app/#/login

temitopejoshua.github.io says

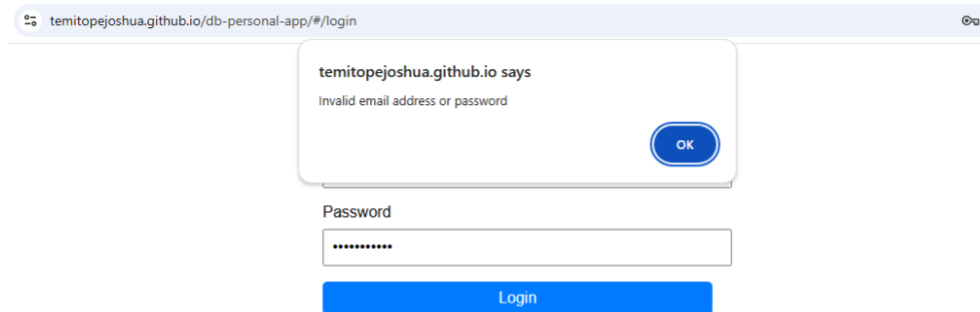
Login successful!

OK

Password

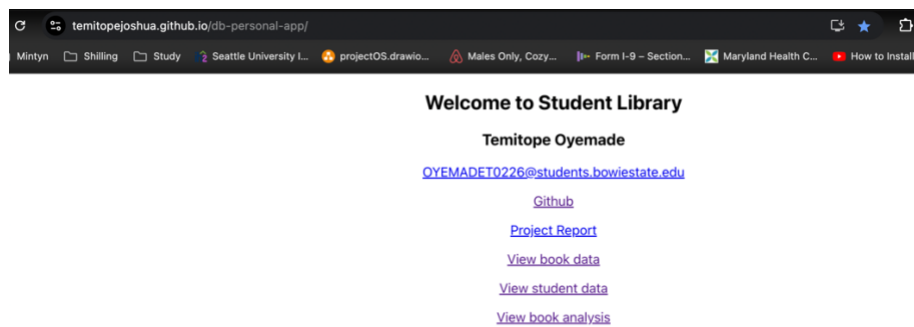
Login

Otherwise, access will not be granted.



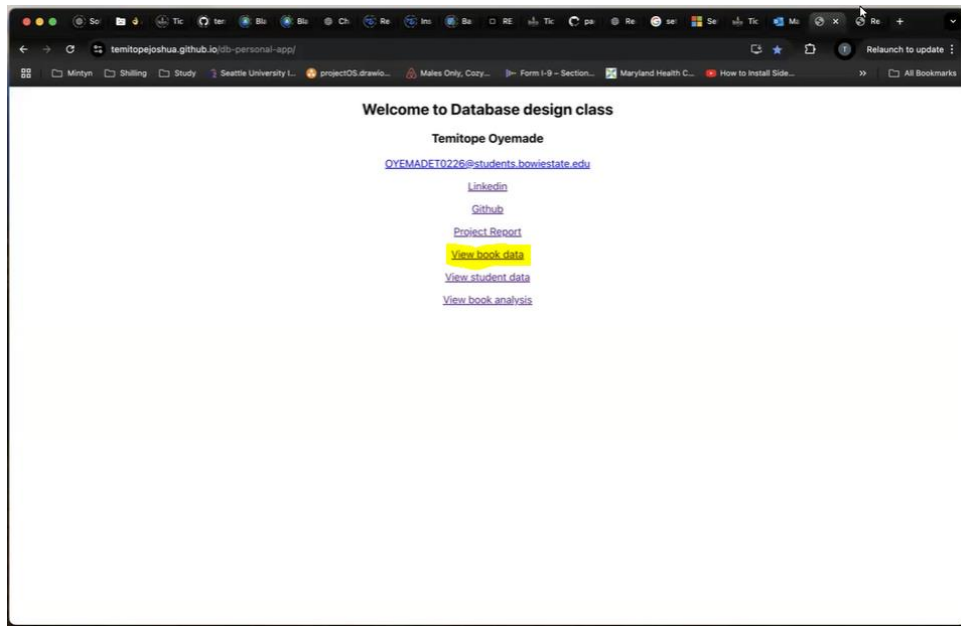
The screenshot shows a web browser window with the address bar displaying "temitopejoshua.github.io/db-personal-app/#/login". A white error message box is centered on the page, stating "temitopejoshua.github.io says Invalid email address or password" with an "OK" button. Below the error box, there is a "Password" label, a password input field containing eight dots, and a blue "Login" button.

Once access is granted take any of the actions below to explore the page.



The screenshot shows the homepage of a web application titled "Welcome to Student Library". The user is identified as "Temitope Oyemade" with the email "OYEMADET0226@students.bowiestate.edu". Below the email, there are five links: "Github", "Project Report", "View book data", "View student data", and "View book analysis". The browser's address bar shows "temitopejoshua.github.io/db-personal-app/" and the tabs include "Mintyn", "Shilling", "Study", "Seattle University L...", "projectOS.drawio...", "Males Only, Cozy...", "Form I-9 - Section...", "Maryland Health C...", and "How to install S".

On the website, click on “View book data” to view the book data



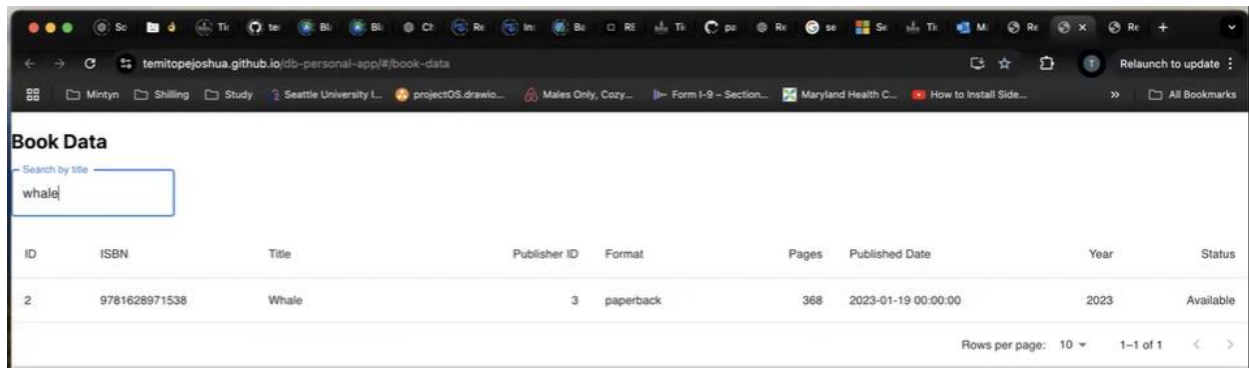
Webpage displaying Book Data

A screenshot of a web browser displaying a table of book data. The table is titled "Book Data" and has a search bar labeled "Search by title". The table has 9 columns: ID, ISBN, Title, Publisher ID, Format, Pages, Published Date, Year, and Status. There are 7 rows of data. The browser's address bar shows the URL "temitopejoshua.github.io/db-personal-app/#/book-data".

ID	ISBN	Title	Publisher ID	Format	Pages	Published Date	Year	Status
1	9788439736967	Boulder	10	paperback	112	2022-08-02 00:00:00	2023	Available
2	9781628971538	Whale	3	paperback	368	2023-01-19 00:00:00	2023	Available
3	9781642861181	The Gospel According to the New World	32	paperback	184	2023-03-07 00:00:00	2023	Available
4	9781529414431	Standing Heavy	12	paperback	252	2022-05-26 00:00:00	2023	Not Available
5	9781474623025	Time Shelter	30	hardcover	304	2022-04-21 00:00:00	2023	Available
6	9781839764318	Is Mother Dead	29	paperback	330	2022-09-27 00:00:00	2023	Not Available
7	9781529427820	Jimi Hendrix Live in Lviv	12	hardcover	416	2023-04-27 00:00:00	2023	Not Available

Rows per page: 10 1-10 of 78

Search by Title/Borrower/Year – for example Whale, or Paradais or Temitope– as you begin to type, it begins to filter out your search

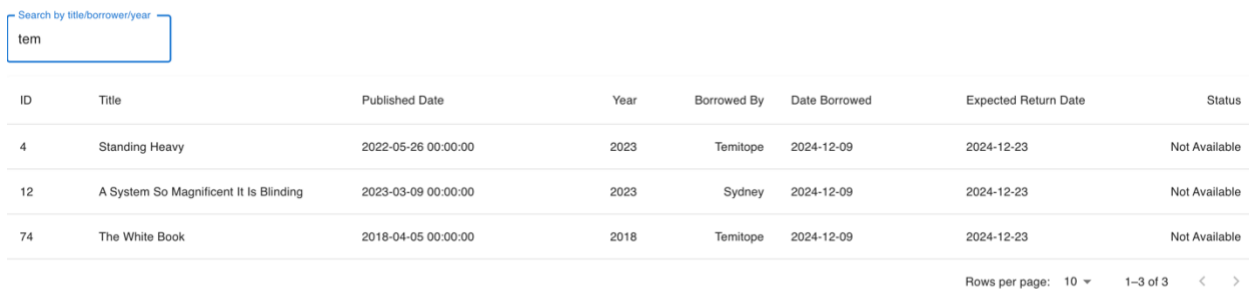


The screenshot shows a web browser window with the URL `temitopejoshua.github.io/db-personal-app/#/book-data`. The page is titled "Book Data" and has a search bar labeled "Search by title" with the text "whale" entered. Below the search bar is a table with the following data:

ID	ISBN	Title	Publisher ID	Format	Pages	Published Date	Year	Status
2	9781628971538	Whale	3	paperback	368	2023-01-19 00:00:00	2023	Available

At the bottom right of the table, it says "Rows per page: 10" and "1-1 of 1".

Book Data

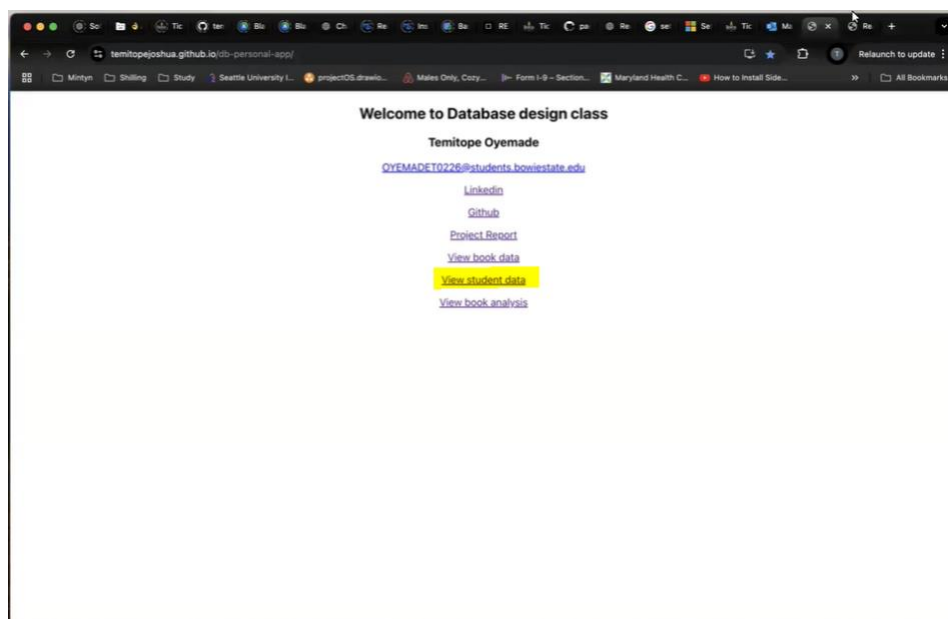


The screenshot shows a web browser window with the URL `temitopejoshua.github.io/db-personal-app/#/book-data`. The page is titled "Book Data" and has a search bar labeled "Search by title/borrower/year" with the text "tem" entered. Below the search bar is a table with the following data:

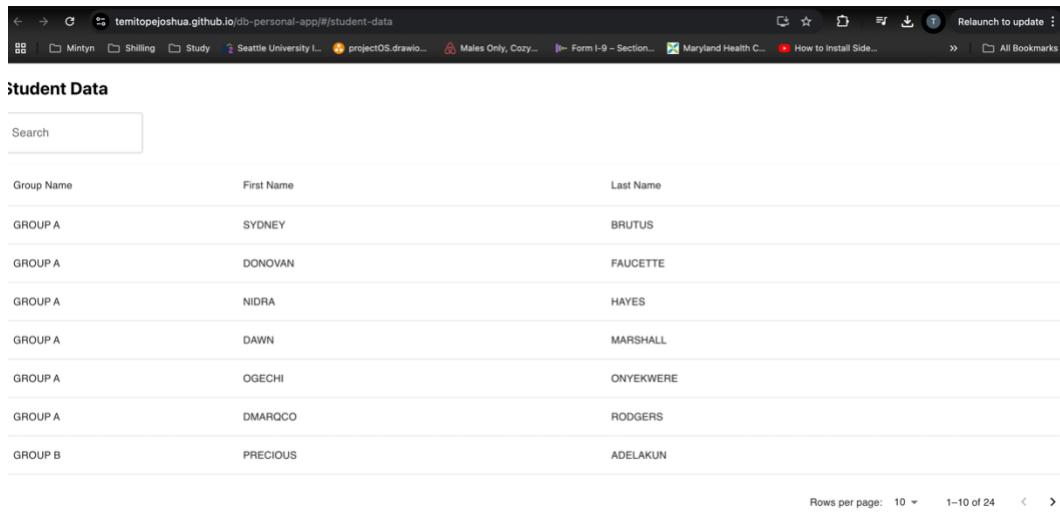
ID	Title	Published Date	Year	Borrowed By	Date Borrowed	Expected Return Date	Status
4	Standing Heavy	2022-05-26 00:00:00	2023	Temitope	2024-12-09	2024-12-23	Not Available
12	A System So Magnificent It Is Blinding	2023-03-09 00:00:00	2023	Sydney	2024-12-09	2024-12-23	Not Available
74	The White Book	2018-04-05 00:00:00	2018	Temitope	2024-12-09	2024-12-23	Not Available

At the bottom right of the table, it says "Rows per page: 10" and "1-3 of 3".

On the website, Click on “View student data” to view student data



Webpage displaying Student Data and the group they belong to, you can view students that belong to a group by simply typing the group name in the search box.

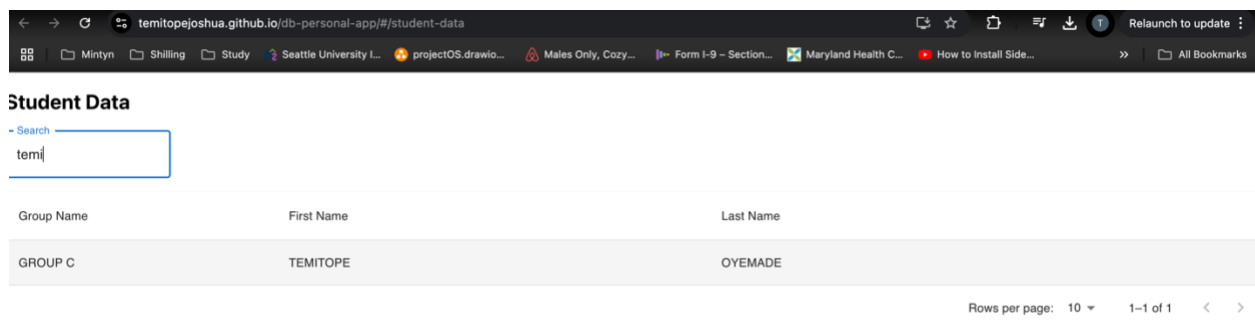


The screenshot shows a web browser window with the URL `temitopejoshua.github.io/db-personal-app/#/student-data`. The page title is "Student Data". There is a search box with the placeholder text "Search". Below the search box is a table with three columns: "Group Name", "First Name", and "Last Name". The table contains the following data:

Group Name	First Name	Last Name
GROUP A	SYDNEY	BRUTUS
GROUP A	DONOVAN	FAUCETTE
GROUP A	NIDRA	HAYES
GROUP A	DAWN	MARSHALL
GROUP A	OGECHI	ONYEKWERE
GROUP A	DMARQCO	RODGERS
GROUP B	PRECIOUS	ADELAKUN

At the bottom right of the table, there is a pagination control showing "Rows per page: 10" and "1-10 of 24".

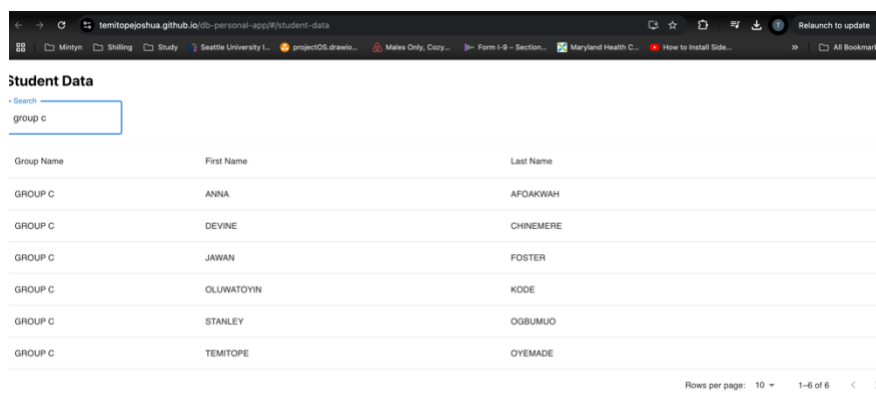
Users can search student data by first, last name or group name. For example as you begin to type “te” for Temitope all the name that have “te” are displayed or “ja” for Jared, all the name with “ja” are displayed.



The screenshot shows the same web browser window as before, but the search box now contains the text "temi". The table below the search box now only displays one row of data:

Group Name	First Name	Last Name
GROUP C	TEMITOPE	OYEMADE

The pagination control at the bottom right now shows "Rows per page: 10" and "1-1 of 1".

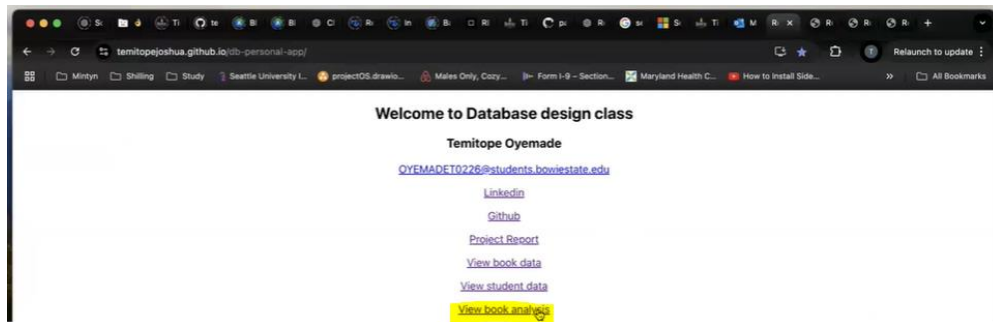


The screenshot shows the same web browser window as before, but the search box now contains the text "group c". The table below the search box now displays six rows of data:

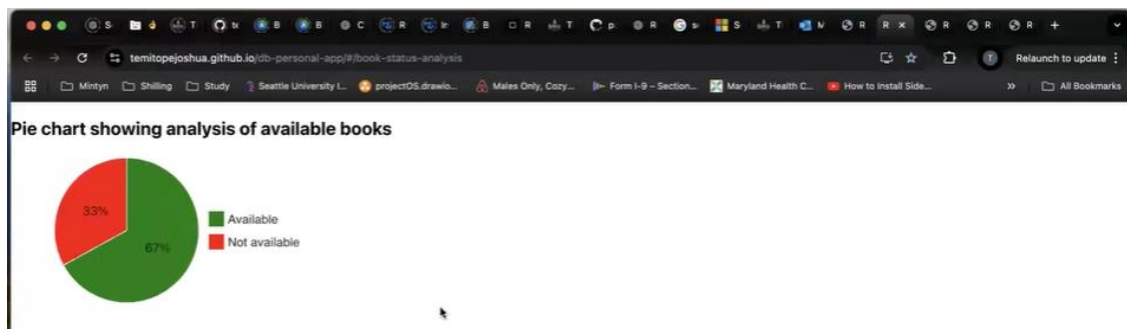
Group Name	First Name	Last Name
GROUP C	ANNA	AFOAKWAH
GROUP C	DEVINE	CHINEMERE
GROUP C	JAWAN	FOSTER
GROUP C	OLUWATOYIN	KODE
GROUP C	STANLEY	OSBUMUO
GROUP C	TEMITOPE	OYEMADE

The pagination control at the bottom right now shows "Rows per page: 10" and "1-6 of 6".

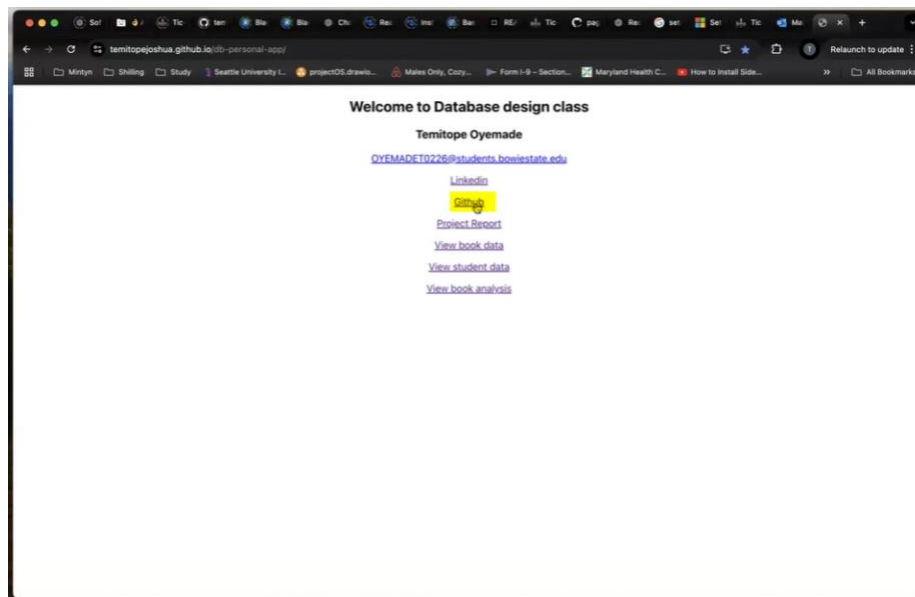
On the front end, Click on “View book analysis” to view the data analysis of book availability.



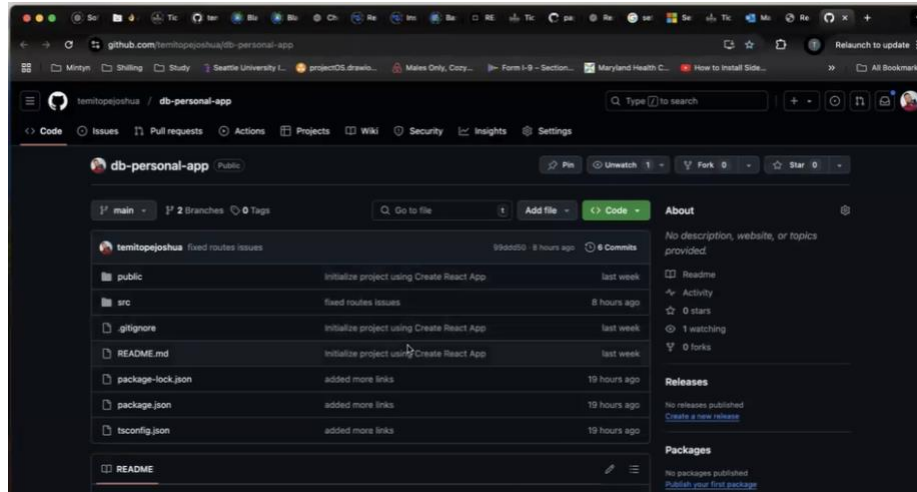
The pie chart shows a quick view of the percentage of books that are available and unavailable.



On the website, Click on “GitHub” to view the GitHub Page.



Website displaying GitHub Page with all the project packages and codes



5.3. Tools and Technologies

- **Programming Language:**
 - **Java:** Chosen for its platform independence, strong database connectivity, and robust features, Java was used to build the core application logic for interactive database management.
- **Backend Framework:**
 - **Node.js:** Utilized for its lightweight, event-driven architecture, Node.js efficiently handled concurrent requests and facilitated dynamic interactions between the client and the database.
- **ORM (Object-Relational Mapping):**
 - **Sequelize:** This ORM simplified database operations with MySQL, providing an abstraction layer over raw SQL queries. This reduced development time, enabled clean code, and improved maintainability.
- **Database:**
 - **MySQL:** Selected for its reliability, scalability, and support for SQL standards, MySQL efficiently managed the relational data needed for the project. Its robust querying capabilities allowed seamless handling of borrowing and returning operations.
- **Web Hosting:**
 - **GitHub Pages:** Used to host project documentation and static resources, ensuring easy public access, version control, and collaborative development.
- **Frontend Framework:**
 - **React with HashRouter:** React enabled the development of a dynamic and responsive user interface, while HashRouter provided seamless navigation with GitHub.

6. System Features

6.1. Borrowing Management

- Specify the name of the student.
- Display borrowed books, borrower details, and return dates.
- Dynamically update the borrowing and return status.

6.2. Rating Analysis

- Show books rated higher or lower than the average rating.
- Allow users to input new ratings to update average dynamically.

6.3. Group Management

- Display students in groups (A, B, C, D).
- Show books borrowed by students in each group.
- Allow querying of a student's group based on their name.

7. Conclusion and Recommendations

The Library Book Borrowing Management System successfully manages book inventory, student assignments, and borrowing activities while enabling meaningful queries and reports. Future improvements could include automated notifications for due dates and overdue penalties.

Appendices

Source Code Repository Link: [<https://github.com/temitopejoshua/db-personal-app>]

Github Page: <https://temitopejoshua.github.io/db-personal-app/>