The Project Demonstration

Version 1.4

EECS 447 Project

The Project Demonstration

EECS 447

Version 1.4

Revision History

Date	Version	Description	Authors
04/05/25	1.0	Document creation, role assignment, and initial division of document sections.	Fatima Avila, Siddh Bharucha, Bhavik Goplani, Vy Luu, Suhaan Syed, Alexis Vielma
04/09/25	1.2	Brainstorming, Supabase Creation	Fatima Avila, Siddh Bharucha, Bhavik Goplani, Vy Luu, Suhaan Syed, Alexis Vielma
04/18/25	1.3	Document finalization and database creation and sql coding.	Fatima Avila, Siddh Bharucha, Bhavik Goplani, Vy Luu, Suhaan Syed, Alexis Vielma
04/26/2025	1.4	Initial draft and database output verification	Fatima Avila, Siddh Bharucha, Bhavik Goplani, Vy Luu, Suhaan Syed, Alexis Vielma

Database Requirements Specifications

1. Demo Queries

```
-- 1. Membership & Account Status

-- List all members with their membership type and account status
SELECT m.member_id, m.name, mt.type_name, m.account_status
FROM members m
JOIN membership_types mt ON m.membership_type_id = mt.type_id;

-- Find all members who are currently 'Overdue'
SELECT member_id, name, contact_info
FROM members
WHERE account_status = 'Overdue';


-- 2. Item Inventory (Books, Magazines, Digital Media)

-- Show all books in the collection with their availability
SELECT li.title, b.author, b.genre, li.availability_status
FROM books b
JOIN library_items li ON b.book_id = li.item_id;

-- List all magazines published in 2024
SELECT li.title, m.issue_number, m.publication_date
FROM magazines m
```

```
JOIN library_items li ON m.magazine_id = li.item_id

WHERE EXTRACT(YEAR FROM m.publication_date) = 2024;



-- List all unavailable digital media

SELECT li.title, dm.creator, dm.format

FROM digital_media dm

JOIN library_items li ON dm.media_id = li.item_id

WHERE li.availability_status <> 'Available';



-- List of all available books (not currently borrowed) within a specific genre.

SELECT

 b.book_id,

 li.title,

 b.author,

 b.genre

FROM

 books b

JOIN library_items li ON b.book_id = li.item_id

WHERE

 li.availability_status = 'Available'

 AND b.genre = 'Mystery';




-- 3. Borrowing Transactions & Late Returns
```

```sql
-- Find all currently borrowed items (i.e., no return date yet)

SELECT bt.borrow_id, m.name AS member, li.title, bt.due_date

FROM borrowing_transactions bt

JOIN members m ON bt.member_id = m.member_id

JOIN library_items li ON bt.item_id = li.item_id

WHERE bt.return_date IS NULL;



-- Members with overdue items and fine incurred

SELECT m.name, li.title, bt.due_date, bt.fine_incurred

FROM borrowing_transactions bt

JOIN members m ON bt.member_id = m.member_id

JOIN library_items li ON bt.item_id = li.item_id

WHERE bt.return_date IS NULL AND bt.due_date < CURRENT_DATE;



-- Members who have borrowed the most books in a particular genre (e.g., "Mystery") in
the last year.

SELECT

 m.member_id,

 m.name,

 COUNT(*) AS borrow_count

FROM

 borrowing_transactions bt

JOIN members m ON bt.member_id = m.member_id

JOIN books b ON bt.item_id = b.book_id

WHERE

 b.genre = 'Mystery'
```

```
 AND bt.borrow_date >= CURRENT_DATE - INTERVAL '1 year'
GROUP BY
 m.member_id,
 m.name
ORDER BY
 borrow_count DESC
LIMIT 5;  -- Top N borrowers


-- Average Borrowing Time (for a genre)
SELECT
 b.genre,
 AVG(bt.return_date - bt.borrow_date) AS avg_borrow_days
FROM
 borrowing_transactions bt
JOIN books b ON bt.item_id = b.book_id
WHERE
 b.genre = 'Mystery'
 AND bt.return_date IS NOT NULL
GROUP BY
 b.genre;




-- 4. Reservations
```

```sql
-- Show all current reservations (not expired)

SELECT m.name AS member, li.title, r.reservation_date, r.expiry_date

FROM reservations r

JOIN members m ON r.member_id = m.member_id

JOIN library_items li ON r.item_id = li.item_id

WHERE r.expiry_date >= CURRENT_DATE;




-- 5. Payments & Fines



-- Total fines paid by each member

SELECT m.name, SUM(p.amount_paid) AS total_paid

FROM payments p

JOIN members m ON p.member_id = m.member_id

GROUP BY m.name;



-- Members who paid more than $10 in fines

SELECT m.name, SUM(p.amount_paid) AS total_paid

FROM payments p

JOIN members m ON p.member_id = m.member_id

GROUP BY m.name

HAVING SUM(p.amount_paid) > 10;



-- Total fines owed by each member, considering overdue books and a daily fine rate
(e.g., $0.25 per day)
```

```
SELECT m.Member_ID, m.Name, SUM((COALESCE(bt.Return_Date, CURRENT_DATE) - bt.Due_Date)
* mt.Fine_Rate) AS Total_Fine

FROM Members m

JOIN Borrowing_Transactions bt ON m.Member_ID = bt.Member_ID

JOIN Membership_Types mt ON m.Membership_Type_ID = mt.Type_ID

WHERE bt.Return_Date IS NULL OR bt.Return_Date > bt.Due_Date

GROUP BY m.Member_ID, m.Name;




-- 6. Notifications Log


-- Show the last 10 notifications sent

SELECT m.name, n.notification_type, n.notification_date

FROM notifications n

JOIN members m ON n.member_id = m.member_id

ORDER BY n.notification_date DESC

LIMIT 10;




-- 7. Analytics / Trends


-- Most borrowed books (Top 5)

SELECT li.title, COUNT(*) AS borrow_count

FROM borrowing_transactions bt

JOIN library_items li ON bt.item_id = li.item_id

JOIN books b ON li.item_id = b.book_id
```

```
GROUP BY li.title

ORDER BY borrow_count DESC

LIMIT 5;



-- Average loan duration by item type

SELECT li.item_type, AVG(bt.return_date - bt.borrow_date) AS avg_days

FROM borrowing_transactions bt

JOIN library_items li ON bt.item_id = li.item_id

WHERE bt.return_date IS NOT NULL

GROUP BY li.item_type;



-- Generate a report of all books due within the next week, sorted by due date.

SELECT

 bt.borrow_id,

 m.name AS member_name,

 li.title,

 bt.due_date

FROM

 borrowing_transactions bt

JOIN members m ON bt.member_id = m.member_id

JOIN books b ON bt.item_id = b.book_id

JOIN library_items li ON bt.item_id = li.item_id

WHERE

 bt.due_date BETWEEN CURRENT_DATE AND CURRENT_DATE + INTERVAL '7 days'

ORDER BY
```

```
 bt.due_date ASC;


-- Most Popular Author in Last Year
SELECT
 b.author,
 COUNT(*) AS borrow_count
FROM
 borrowing_transactions bt
JOIN books b ON bt.item_id = b.book_id
WHERE
 bt.borrow_date >= date_trunc('year', CURRENT_DATE)
GROUP BY
 b.author
ORDER BY
 borrow_count DESC
LIMIT 1;
```

2. Collection Analysis Report

```
------ Member Engagement Report ------

-- 1. Distribution of books by genre
SELECT genre, COUNT(*) AS book_count
FROM books
```

```
GROUP BY genre

ORDER BY book_count DESC;



-- 2. Trends in publication year (proxy for acquisition trend)

SELECT publication_year, COUNT(*) AS books_published

FROM books

WHERE publication_year >= EXTRACT(YEAR FROM CURRENT_DATE) - 15

GROUP BY publication_year

ORDER BY publication_year DESC;



-- 3. Average age of books (in years)

SELECT AVG(EXTRACT(YEAR FROM CURRENT_DATE) - publication_year) AS avg_book_age_years

FROM books;



-- 4. Books with zero borrows (low circulation)

SELECT b.book_id, li.title, b.author, b.genre, b.publication_year

FROM books b

JOIN library_items li ON li.item_id = b.book_id

WHERE NOT EXISTS (

   SELECT 1

   FROM borrowing_transactions bt

   WHERE bt.item_id = li.item_id
```

```
);



-- 5. Borrow count per genre to identify under-represented genres

SELECT b.genre, COUNT(bt.borrow_id) AS total_borrows

FROM books b

JOIN library_items li ON li.item_id = b.book_id

LEFT JOIN borrowing_transactions bt ON bt.item_id = li.item_id

GROUP BY b.genre

ORDER BY total_borrows ASC;



-- 6. Borrow count per author to identify under-represented authors

SELECT b.author, COUNT(bt.borrow_id) AS total_borrows

FROM books b

JOIN library_items li ON li.item_id = b.book_id

LEFT JOIN borrowing_transactions bt ON bt.item_id = li.item_id

GROUP BY b.author

ORDER BY total_borrows ASC;
```

7. Github Repository

- Link: https://github.com/aelxxs/tech-titans