# RAIT Library Management System

This is a full-stack library management application built with a microservices architecture. It includes a Next.js frontend, three separate backend services for authentication, library functions, and payments, and uses MongoDB and PostgreSQL for data storage.

## Project Structure

```
rait-library-project/
├── auth-service/        # (Node.js) Manages user registration and login
├── capstone/            # (Next.js) The main frontend application
├── library_service/     # (Java/Spring Boot) Manages books and borrowing
├── payment-service/     # (Node.js) Manages fine payments with Razorpay
└── README.md            # This file
```

## Prerequisites

Before you begin, ensure you have the following installed on your system:

- **Node.js** (v18 or later)
- **Java** (JDK 17 or later)
- **Maven**
- **MongoDB** (must be running locally)
- **PostgreSQL** (must be running locally)

## Step 1: Database Setup

You need two databases running on your local machine.

### MongoDB (for auth-service)

- Ensure your local MongoDB server is running. The service will automatically create the authdb database and its collections on first run.

### PostgreSQL (for library_service)

1. Start your local PostgreSQL server.
2. Connect to it using a client like psql or a GUI tool.
3. Create a new database named librarydb:
   CREATE DATABASE librarydb;

## Step 2: Backend Services Setup

You will need to open **3 separate terminals** for the backend services.

### 1. Auth Service (Port 3001)

- **Terminal 1:**
  cd auth-service
  npm install
  # Create a .env file in this directory with your JWT_SECRET
  # Example .env content:
  # JWT_SECRET=your_super_secret_key_here
  node index.js

  *You should see "Auth service running on port 3001" and "MongoDB connected successfully".*

### 2. Payment Service (Port 3002)

- **Terminal 2:**
  cd payment-service
  npm install
  # Create a .env file in this directory with your Razorpay keys
  # Example .env content:
  # RAZORPAY_KEY_ID=your_key_id
  # RAZORPAY_KEY_SECRET=your_key_secret
  node index.js

  *You should see "Payment service running on port 3002".*

### 3. Library Service (Port 8080)

- **Terminal 3:**
  - **Configuration:** Open library_service/src/main/resources/application.properties. Make sure the spring.datasource.password matches your local PostgreSQL password.
  - **Run the service:**
    cd library_service
    mvn spring-boot:run

*Wait for the Spring Boot application to start. You will see a lot of logs, ending with a message that the application has started.*

## Step 3: Frontend Setup

You will need **1 more terminal** for the frontend application.

- **Terminal 4:**
  cd capstone

```
npm install
# Create a .env.local file in this directory with your Razorpay Key ID
# Example .env.local content:
# NEXT_PUBLIC_RAZORPAY_KEY_ID=your_key_id
npm run dev
```

*You should see that the server has started on http://localhost:3000.*

## Step 4: Access the Application

Once all services and the frontend are running, you can open your web browser and navigate to:

**http://localhost:3000**

You should see the landing page, and the entire application should be fully functional.