const express = require("express");

const app = express();

app.use(express.json());

const { open } = require("sqlite");

const sqlite3 = require("sqlite3");

const path = require("path");

const dbPath = path.join(\_\_dirname, "goodreads.db");

let db = null;

const initializeDBAndServer = async () => {

  try {

    db = await open({

      filename: dbPath,

      driver: sqlite3.Database,

    });

    app.listen(3000, () => {

      console.log("Server Running at localhost:3000");

    });

  } catch (e) {

    console.log(`DB Error: ${e.errorMessage}`);

  }

};

initializeDBAndServer();

//GET Books API

app.get("/books/", async (req, res) => {

  const getBooksQuery = `SELECT \* FROM book ORDER BY book\_id ASC;`;

  const booksArray = await db.all(getBooksQuery);

  res.send(booksArray);

});

//GET BOOK API

app.get("/books/:id", async (req, res) => {

  const { id } = req.params;

  const getBookQuery = `SELECT \* FROM book WHERE book\_id=${id};`;

  const book = await db.get(getBookQuery);

  res.send(book);

});

app.post("/books/", async (request, response) => {

  const bookDetails = request.body;

  const {

    title,

    authorId,

    rating,

    ratingCount,

    reviewCount,

    description,

    pages,

    dateOfPublication,

    editionLanguage,

    price,

    onlineStores,

  } = bookDetails;

  const createBookQuery = `INSERT INTO book(title,author\_id,rating,rating\_count,review\_count,

    description,pages,date\_of\_publication,edition\_language,price,online\_stores)

    VALUES(

        '${title}',

        ${authorId},

        ${rating},

        $${ratingCount},

        ${reviewCount},

        '${description}',

        ${pages},

       '${dateOfPublication}',

       '${editionLanguage}',

        ${price},

        '${onlineStores}'

    );`;

  const responseData = await db.run(createBookQuery);

  const bookId = await responseData.lastID;

  response.send({ bookId });

});

//API UPDATE book

app.put("/books/:bookId/", async (request, response) => {

  const { bookId } = request.params;

  const bookDetails = request.body;

  const {

    title,

    authorId,

    rating,

    ratingCount,

    reviewCount,

    description,

    pages,

    dateOfPublication,

    editionLanguage,

    price,

    onlineStores,

  } = bookDetails;

  const updateBookQuery = `UPDATE book SET title = '${title}', author\_id = ${authorId},

rating=${rating},rating\_count = ${ratingCount},review\_count = ${reviewCount},

description = '${description}',pages = ${pages},date\_of\_publication ='${dateOfPublication}',

edition\_language = '${editionLanguage}',price = ${price}, online\_stores = '${onlineStores}'

WHERE book\_id = ${bookId};`;

  await db.run(updateBookQuery);

  response.send("Book Updated Successfully");

});

// DELETE BOOK API

app.delete("/books/:bookId/", async (request, response) => {

  const { bookId } = request.params;

  const deleteBookQuery = `DELETE FROM book WHERE book\_id = ${bookId};`;

  await db.run(deleteBookQuery);

  response.send("Book Deleted Successfully");

});

// AUTHORS API

app.get("/authors/:authorId/books/", async (request, response) => {

  const { authorId } = request.params;

  const getAuthorBooksQuery = `SELECT \* FROM book WHERE author\_id = ${authorId};`;

  const booksArray = await db.all(getAuthorBooksQuery);

  response.send(booksArray);

});

Module.exports = app;