Book.js=let books = [];

const addBook = ({ name, year, author, summary, publisher, pageCount, readPage, reading }) => {

  const newBook = {

    id: Date.now().toString(),

    name,

    year,

    author,

    summary,

    publisher,

    pageCount,

    readPage,

    reading,

    finished: pageCount === readPage,

  };

  books.push(newBook);

  return newBook;

};

const getAllBooks = () => books;

const getBookById = (id) => books.find((book) => book.id === id);

const updateBook = (id, { name, year, author, summary, publisher, pageCount, readPage, reading }) => {

  const index = books.findIndex((book) => book.id === id);

  if (index === -1) return null;

  books[index] = { id, name, year, author, summary, publisher, pageCount, readPage, reading, finished: pageCount === readPage };

  return books[index];

};

const deleteBook = (id) => {

  const index = books.findIndex((book) => book.id === id);

  if (index === -1) return false;

  books.splice(index, 1);

  return true;

};

export default { addBook, getAllBooks, getBookById, updateBook, deleteBook };

handler.js=import Book from '../models/book.js';

const addBookHandler = (req, h) => {

  const { name, year, author, summary, publisher, pageCount, readPage, reading } = req.payload;

  if (!name) {

    return h.response({

      status: 'fail',

      message: 'Gagal menambahkan buku. Mohon isi nama buku',

    }).code(400);

  }

  if (readPage > pageCount) {

    return h.response({

      status: 'fail',

      message: 'Gagal menambahkan buku. readPage tidak boleh lebih besar dari pageCount',

    }).code(400);

  }

  const newBook = Book.addBook({ name, year, author, summary, publisher, pageCount, readPage, reading });

  return h.response({

    status: 'success',

    message: 'Buku berhasil ditambahkan',

    data: {

      bookId: newBook.id,

    },

  }).code(201);

};

const getAllBooksHandler = (req, h) => {

  const books = Book.getAllBooks();

  const filteredBooks = books.map((book) => ({

    id: book.id,

    name: book.name,

    publisher: book.publisher,

  }));

  return h.response({

    status: 'success',

    data: {

      books: filteredBooks,

    },

  }).code(200);

};

const getBookByIdHandler = (req, h) => {

  const { bookId } = req.params;

  const book = Book.getBookById(bookId);

  if (!book) {

    return h.response({

      status: 'fail',

      message: 'Buku tidak ditemukan',

    }).code(404);

  }

  return h.response({

    status: 'success',

    data: {

      book,

    },

  }).code(200);

};

const updateBookHandler = (req, h) => {

  const { bookId } = req.params;

  const { name, year, author, summary, publisher, pageCount, readPage, reading } = req.payload;

  if (!name) {

    return h.response({

      status: 'fail',

      message: 'Gagal memperbarui buku. Mohon isi nama buku',

    }).code(400);

  }

  if (readPage > pageCount) {

    return h.response({

      status: 'fail',

      message: 'Gagal memperbarui buku. readPage tidak boleh lebih besar dari pageCount',

    }).code(400);

  }

  const updatedBook = Book.updateBook(bookId, { name, year, author, summary, publisher, pageCount, readPage, reading });

  if (!updatedBook) {

    return h.response({

      status: 'fail',

      message: 'Gagal memperbarui buku. Id tidak ditemukan',

    }).code(404);

  }

  return h.response({

    status: 'success',

    message: 'Buku berhasil diperbarui',

    data: {

      book: updatedBook,

    },

  }).code(200);

};

const deleteBookHandler = (req, h) => {

  const { bookId } = req.params;

  const success = Book.deleteBook(bookId);

  if (!success) {

    return h.response({

      status: 'fail',

      message: 'Buku tidak ditemukan',

    }).code(404);

  }

  return h.response({

    status: 'success',

    message: 'Buku berhasil dihapus',

  }).code(200);

};

export { addBookHandler, getAllBooksHandler, getBookByIdHandler, updateBookHandler, deleteBookHandler };