

BOOK RENTAL

**LAPORAN PROJECT TENGAH SEMESTER
WEB TECHNOLOGY 3 – SEMESTER VII
TAHUN AKADEMIK 2022/2023**

Dosen Pengampu:
Pak Fikri Fadillah



KWIK KIAN GIE
SCHOOL OF BUSINESS

Disusun Oleh:
Johanes Yefta (58190266)

DESKRIPSI WEBSITE

Aplikasi web ini dirancang untuk menyediakan jasa peminjaman buku secara daring yang dapat diakses semua orang. *Book Rental* bertujuan untuk membantu meningkatkan kemampuan literasi khususnya bagi Indonesia.

Tujuan dari website ini adalah untuk menjangkau para masyarakat dari berbagai pelosok Indonesia agar tetap terhubung dengan masyarakat perkotaan, sehingga kita sebagai masyarakat kota juga tahu bahwasannya ada orang-orang yang mengharapkan bantuan kita.

Website *Book Rental* dibagi menjadi beberapa *pages*, antara lain: *Login*, *Home*, *Wishlist*, dan *Add Book*. Halaman utama dari website ini adalah halaman *login*, dimana sistem akan meminta kredensial valid pengguna. Setelah berhasil melakukan verifikasi identitas, maka website akan diteruskan ke halaman *Home*. Halaman ini menyediakan fitur yang dapat mempermudah *user* dalam mencari buku, antara lain:

1. *Search Feature*

Fitur pencarian disini dapat dipecah menjadi beberapa sub-bagian, antara lain:

- a. *Search by Book Title*
- b. *Search by Book Author*
- c. *Search by ISBN Number*
- d. *Sort by all available book*

2. *Wishlist Feature*

User dapat menambahkan buku yang ingin dibaca ke wishlist agar dapat mempermudah pencarian ketika buku tersebut ingin dibaca.

3. *CRUD Feature*

Fitur yang dapat dimanfaatkan bagi admin untuk dapat melakukan operasi berupa penambahan buku, penyuntingan buku, penghapusan buku, dan menampilkan buku dari *server* ke *client* atau sebaliknya. Penulis juga memberikan hak kepada pengguna untuk dapat melakukan penambahan buku ke server. Dengan adanya fitur ini, diharapkan aplikasi ini bisa menyediakan repositori yang lengkap bagi para penggunanya.

4. *Login Feature*

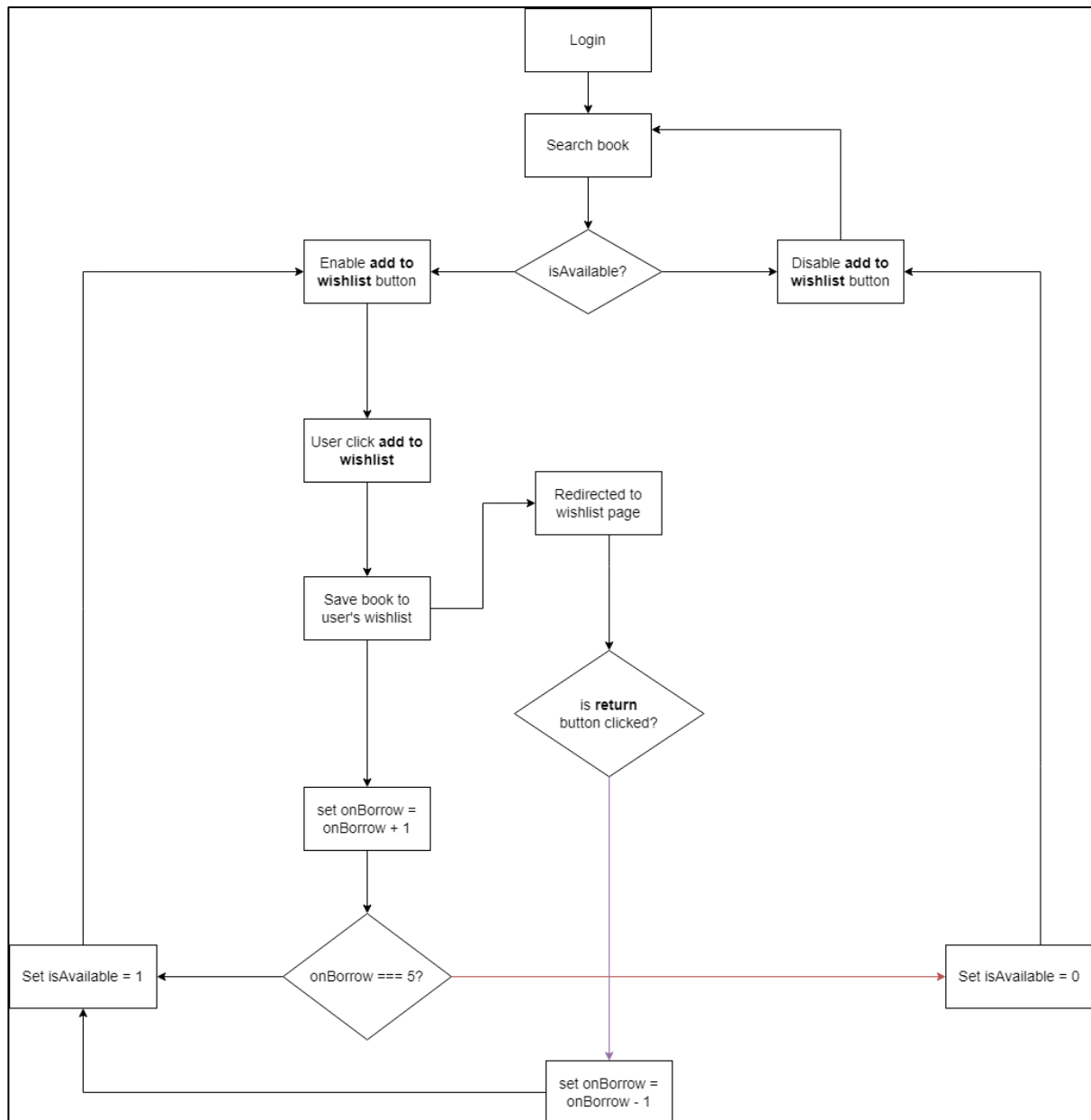
Fitur autentikasi pengguna agar dapat mengakses aplikasi web ini. Fitur ini menggunakan JWT sebagai token yang nantinya digunakan untuk memvalidasi apakah user ini benar terdaftar atau tidak.

Halaman *wishlist* akan ditampilkan bila pengguna menambahkan buku ke *wishlist*. Halaman ini akan menampilkan buku-buku apa yang telah ditambahkan oleh pengguna ke *wishlist*. Selain itu, di halaman ini *user* bisa menambahkan buku lain dan mengembalikan buku.

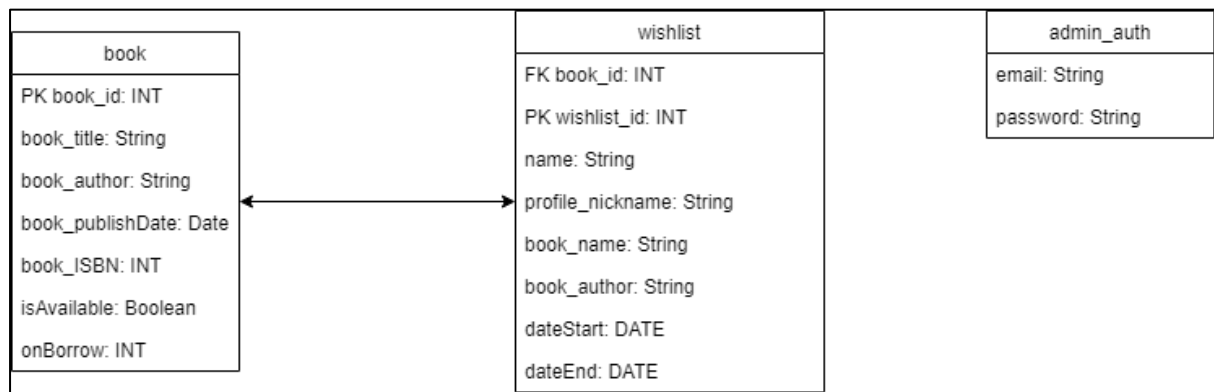
.

PROJECT DIAGRAM FLOW

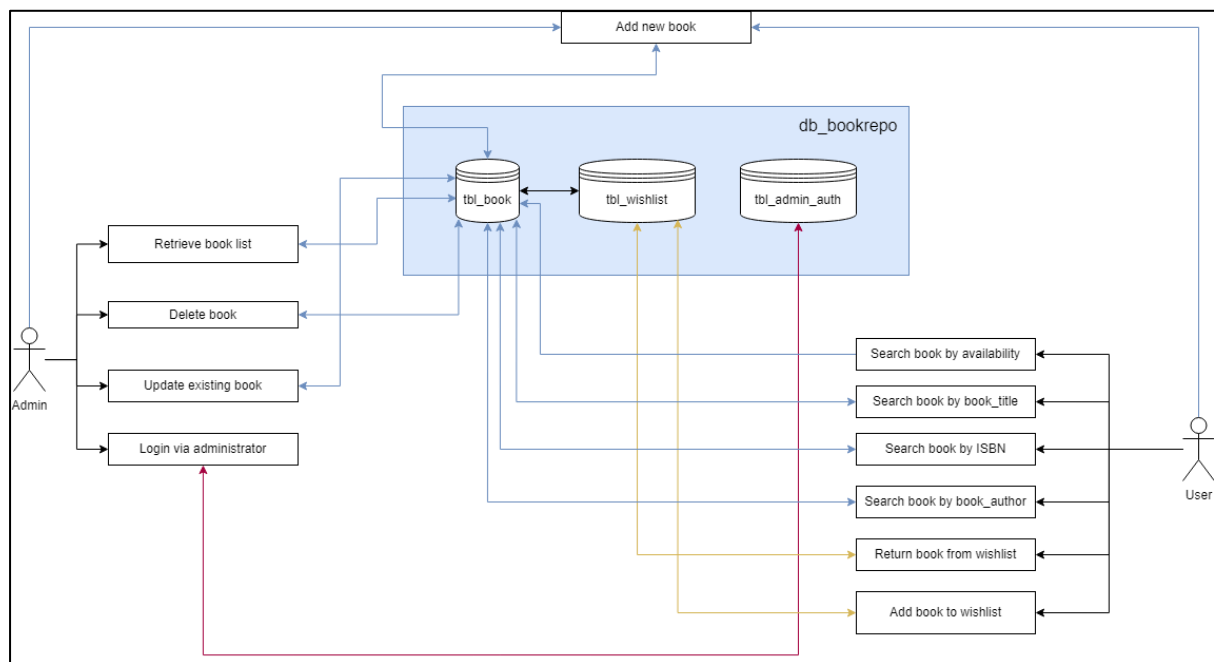
Activity Diagram:



Database Relationship Diagram:



Use Case Diagram:



LINK AKSES PROYEK

Berikut saya lampirkan *link github* untuk *source code* beserta aset yang saya gunakan dalam proyek kali ini: <https://github.com/yefanmaa/react-book-rental>

(Panduan dan instruksi untuk menjalankan proyek ini dapat dilihat pada file README.md).

SOURCE CODE

Berikut adalah kode untuk bagian **backend** (index.js):

```
const express = require('express');
const bodyParser = require('body-parser');
const cors = require('cors');
const app = express();
const port = 3301;
const mysql = require('mysql');

// to connect form into action
app.use(express.urlencoded({
  extended: true
}));

app.use(cors());

app.use(express.json());
// app.use(bodyParser.urlencoded( { extended: true }));

// initialize server listening to ther port
app.listen(3301, () => {
  console.log(`Running on port ${port}`);
});

// display something in browser landing page
app.get("/", (req, res) => {
  res.send('Welcome back!');
});

// create database connection
const conn = mysql.createPool({
  connectionLimit: 10,
  host: '127.0.0.1',
  user: 'root',
  password: '',
  database: 'bookrepo'
});

// checking error and get connection
conn.getConnection((err, connection) => {
  console.log('Database connected succesfully ');
})

// display all books data
app.get('/books', (req, res) => {
  let sql = "SELECT * FROM book";
  let query = conn.query(sql, (err, results) => {
```

```

        if (err) {
            console.log(err);
        } else {
            res.send(results);
        }
    });
});

// display all books data based on their id
app.get('/books/:id', (req, res) => {
    let sql = "SELECT * FROM book WHERE book_id="+req.params.id;
    let query = conn.query(sql, (err, results) => {
        if (err) {
            console.log(err);
        } else {
            res.send(results);
        }
    });
});

// get data based on book title that user search
app.get('/book/search-title', function (req, res, next) {
    let sql = "SELECT * FROM book WHERE book_title LIKE " + `'%` +
req.query.keyword + `'%`;
    let query = conn.query(sql, (err, results) => {
        if (err) {
            console.log(err);
        } else {
            res.send(results);
        }
    });
});

// get data based on book author that user search
app.get('/book/search-author', function (req, res, next) {
    let sql = "SELECT * FROM book WHERE book_author LIKE " + `'%` +
req.query.keyword + `'%`;
    let query = conn.query(sql, (err, results) => {
        if (err) {
            console.log(err);
        } else {
            res.send(results);
        }
    });
});

// get data based on book isbn that user search
app.get('/book/search-isbn', function (req, res, next) {

```

```

    let sql = "SELECT * FROM book WHERE book_ISBN LIKE " + `%` +
req.query.keyword + `%`;
    let query = conn.query(sql, (err, results) => {
      if (err) {
        console.log(err);
      } else {
        res.send(results);
      }
    });
  });
});

// get data based on availability status
app.get('/book/status', function (req, res) {
  let sql = "SELECT * FROM book WHERE isAvailable = '1'";
  let query = conn.query(sql, (err, results) => {
    if (err) {
      console.log(err);
    } else {
      res.send(results);
    }
  })
});

// get data based on user wishlist
app.get('/my-wishlist/', function (req, res, next) {
  const name = req.query.name;
  let sql = "SELECT * FROM wishlist WHERE name= ?";
  let query = conn.query(sql, name, (err, results) => {
    if (err) {
      console.log(err);
    } else {
      res.send(results);
    }
  });
});

// add new data
app.post('/addBooks', (req, res) => {
  const title = req.body.book_title;
  const author = req.body.book_author;
  const publishdate = req.body.book_publishDate;
  const isbn = req.body.book_ISBN;
  const isavailable = req.body.isAvailable;
  const onborrow = req.body.onBorrow;

  let data = {book_title: req.body.book_title, book_author:
req.body.book_author, book_publishDate: req.body.book_publishDate, book_ISBN:

```



```

req.body.book_ISBN, isAvailable: req.body.isAvailable, onBorrow:
req.body.onBorrow});
    let sql = "INSERT INTO book (book_title, book_author, book_publishDate,
book_ISBN, isAvailable, onBorrow) VALUES (?, ?, ?, ?, ?, ?)";
    let query = conn.query(sql, [title, author, publishdate, isbn,
isavailable, onborrow], (err, results) => {
        if (err) {
            console.log(err);
        } else {
            res.send(results);
        }
    });
});

// add new data into wishlist table
app.post('/addToWishlist', (req, res) => {
    const name = req.body.name;
    const bookName = req.body.book_name;
    const bookAuthor = req.body.book_author;
    const datestart = req.body.dateStart;
    const dateend = req.body.dateEnd;

    let sql = "INSERT INTO wishlist (name, book_name, book_author, dateStart,
dateEnd) VALUES (?, ?, ?, ?, ?)";
    let query = conn.query(sql, [name, bookName, bookAuthor, datestart,
dateend], (err, results) => {
        if (err) {
            console.log(err);
        } else {
            res.send(results);
        }
    });
});

// edit book data based on their id
app.put('/editBooks/:id', (req, res) => {

    const title = req.body.book_title;
    const author = req.body.book_author;
    const publishdate = req.body.book_publishDate;
    const isbn = req.body.book_ISBN;
    const isavailable = req.body.isAvailable;
    const onborrow = req.body.onBorrow;

    let sql = "UPDATE book SET book_title=?, book_author=?,
book_publishDate=?, book_ISBN=?, isAvailable=?, onBorrow=? WHERE
book_id="+req.params.id;

```

```

    let query = conn.query(sql, [title, author, publishdate, isbn,
isavailable, onborrow], (err, results) => {
      if (err) {
        console.log(err);
      } else {
        res.send(results);
      }
    });
  });
});

// delete book data based on their title
app.delete('/deleteBooks/:book_title', (req, res) => {
  const name = req.params.book_title;
  let sql = "DELETE FROM book WHERE book_title = ?";
  let query = conn.query(sql, name, (err, results) => {
    if (err) {
      console.log(err);
    } else {
      res.send(results);
    }
  });
});

// delete book data after removed from wishlist
app.delete('/delete-wishlist-book/:wishlist_id', (req, res) => {
  const id = req.params.wishlist_id;
  let sql = "DELETE FROM wishlist WHERE wishlist_id = ?";
  let query = conn.query(sql, id, (err, results) => {
    if (err) {
      console.log(err);
    } else {
      res.send(results);
    }
  })
});

// get book data after removal
app.get('/wishlist/book-data/:book_id', (req, res) => {
  const id = req.params.book_id;
  let sql = "SELECT * FROM book WHERE book_id = ?";
  let query = conn.query(sql, id, (err, results) => {
    if (err) {
      console.log(err);
    } else {
      res.send(results);
    }
  })
});

```

Kode pada halaman login (**login.js**):

```
import React, { useState } from "react";
import axios from "axios";
import { setAuthToken } from "../helpers/setAuthToken"
import Alert from 'react-bootstrap/Alert';
import Form from 'react-bootstrap/Form';
import Button from "react-bootstrap/Button";

function Login() {

  const [emailInput, setEmailInput] = useState("");
  const [passInput, setPassInput] = useState("");
  const [showAlert, setShowAlert] = useState(false);

  const handleSubmit = (email, password) => {

    //reqres registered sample user
    const loginPayload = {
      email: 'eve.holt@reqres.in',
      password: 'cityslicka'
    }

    axios.post("https://reqres.in/api/login", loginPayload)
      .then(response => {
        //get token from response
        const token = response.data.token;

        //set JWT token to local
        localStorage.setItem("token", token);

        //set token to axios common header
        setAuthToken(token);

        //redirect user to home page
        if (emailInput !== loginPayload.email || passInput !==
loginPayload.password) {
          setShowAlert(true);

          setTimeout(() => {
            setShowAlert(false);
          }, 3000);
        } else {
          window.location.href = '/home'
        }
      })
      .catch(err => console.log(err));
  }
}
```

```

};

return (
  <div>
    <div className="global-wrapper-login">

      <div className="container-login">
        <h2 style={{marginTop: '5px', marginBottom: '2px',
textAlign:'center', fontSize:'30px'}}>Welcome back!</h2>
        <p style={{marginBottom: '15px', textAlign:'center',
fontSize:'14px', opacity:0.4}}>Please provide your identity.</p>
        <Form
          onSubmit={(event) => {
            event.preventDefault()
            const [email, password] = event.target.children;
            handleSubmit(email, password);
          }}
        >
          <Form.Group className="mb-3">
            <Form.Label>Email</Form.Label>
            <Form.Control type="email" style={{opacity:
0.5}} placeholder="Enter your email address" id="email" name="email"
onChange={(e) => setEmailInput(e.target.value)} />
          </Form.Group>

          <Form.Group className="mb-3">
            <Form.Label>Password</Form.Label>
            <Form.Control type="password" style={{opacity: 0.5}}
placeholder="Enter your password" id="password" name="password" onChange={(e)
=> setPassInput(e.target.value)} />
          </Form.Group>

          <Button className="btn-login" type="submit"
variant="primary">Login</Button>
        </Form>

        <Alert show={showAlert} key="danger" variant="danger">Invalid
username or password!</Alert>
      </div>
    </div>
  </div>

);
}
export default Login

```

Kode untuk halaman Home (**home.js**):

```
import React, { useState } from "react";
import { useNavigate } from 'react-router-dom';
import Dropdown from 'react-bootstrap/Dropdown';
import DropdownButton from 'react-bootstrap/DropdownButton';

// initialize variable called query to pass data to 'search-book' component
var query = '';

function Home () {

  // initialize navigation variable
  const navigate = useNavigate();

  // useState to get keyword value
  const [keyword, setKeyword] = useState('');
  query = keyword;

  // function to handle user when click submit button
  const handleSubmit = event => {
    event.preventDefault();
    setKeyword(keyword);
    navigate(`/book/search-title?keyword=${keyword}`);
  }

  const submitStatusFilter = () => {
    navigate(`/book/status=available`);
  }

  const submitAuthorFilter = (event) => {
    event.preventDefault();
    setKeyword(keyword);
    navigate(`/book/search-author?keyword=${keyword}`);
  }

  const submitISBNFilter = (event) => {
    event.preventDefault();
    setKeyword(keyword);
    navigate(`/book/search-isbn?keyword=${keyword}`);
  }

  return (
    <div>
      <section id="section1">
        <div className="container">
          <div className="icon-container">
```

```

        <i className="bi bi-person-circle"></i>
      </div>

      <p className="h1" style={{paddingBottom: '25px', color:
'white'}}><span className="highlight-1">Find your book</span> and start
reading <br></br> with our complete e-library.</p>

      <div className="search">
        <form onSubmit={handleSubmit} method="get"
className="form-container">
          <input type="text" className="form-control"
name="keyword" placeholder="What book you want to read?" onChange={event =>
setKeyword(event.target.value)} value={keyword}/>
          <Dropdown>
            <DropdownButton id="dropdown-item-button"
title="Search By">
              <Dropdown.Item as="button">Book
Title</Dropdown.Item>
              <Dropdown.Item as="button"
onClick={submitAuthorFilter}>Book Author</Dropdown.Item>
              <Dropdown.Item as="button"
onClick={submitISBNFilter}>ISBN</Dropdown.Item>
            </DropdownButton>
          </Dropdown>

          {/* <button type="submit" className="btn btn-
primary">Search</button> */}
        </form>
      </div>

      <div className="filterisAvailable">
        <button onClick={submitStatusFilter} className="btn
btn-outline-light">See Available Book</button>
      </div>

      <div className="upload">
        <a href="/add/book">♥ Upload a book ♥</a>
      </div>
    </div>
  </section>
</div>

)
}

export default Home;
export {query};

```

Kode untuk halaman wishlist (**wishlist.js**):

```
import React, {useState, useEffect} from "react";
import { useNavigate } from "react-router-dom";
import '../wishlist.css'
import Axios from "axios";
import { Plus } from "react-bootstrap-icons";

const Wishlist = (props) => {

  const navigate = useNavigate();

  // handling dynamic title of this page
  useEffect(() => {
    document.title = props.title;
  }, [props])

  // useState to handling list of book that requested
  const [bookWishList, setBookWishList] = useState([]);

  // requesting books based on NAME
  useEffect(() => {
    Axios.get('http://localhost:3301/my-wishlist', { params: { name:
'alimc23' }})
      .then(response => {
        setBookWishList(response.data);
      })
  }, [])

  return (
    <div className="body">
      <div className="global-container">
        <div className="header-component">
          <h3 style={{marginLeft: '60px', marginBottom: '30px'}}>My
Wishlist <span style={{fontWeight: '100', fontSize:
'20px'}}>` (@alimc23)`</span></h3>
          <button onClick={() => navigate('/home')} type="button"
className="btn btn-dark" style={{marginRight: '60px', backgroundColor:
'transparent', border: 'none'}}><Plus size={"30px"} style={{marginBottom:
'5px'}}/>Add Book</button>
        </div>
        <div className="card-wrapper">
          {bookWishList.map((val, i) => {

            // remove GMT from date info
```



```

    );
}

export default Wishlist;

```

Kode untuk halaman *upload book* (**add-book.js**):

```

import React, { useState, useEffect } from "react";
import Axios from "axios";
import Form from 'react-bootstrap/Form';
import Button from "react-bootstrap/Button";
import { useNavigate } from "react-router-dom";
import "../global-style.css"

const AddBook = (props) => {

  // handling dynamic title of this page
  useEffect(() => {
    document.title = props.title;
  }, [props])

  const [isAvailableVal, setIsAvailableVal] = useState('Available');

  // initialize navigation variable
  const navigate = useNavigate();

  // useState to set new value of specific field
  const [addBook, setAddBook] = useState('');
  const [addAuthor, setAddAuthor] = useState('');
  const [addPublishDate, setPublishDate] = useState('');
  const [addISBN, setAddISBN] = useState('');
  const [addIsAvailable, setAddIsAvaialble] = useState('');
  const [addOnBorrow, setAddOnBorrow] = useState('');

  // handling cancel button
  const HandlingCancel = () => {
    navigate('/home');
  }

  // handling form when user submit new data
  const uploadBook = () => {

    // sending form data to backend ("db_key", "formValue")
    const bookData = {"book_title": addBook, "book_author": addAuthor,
"book_publishDate": addPublishDate, "book_ISBN": addISBN, "isAvailable":
addIsAvailable, "onBorrow": addOnBorrow};

```



```

        { /* form group of book author */ }
        <Form.Group className="mb-3">
            <Form.Label>Author</Form.Label>
            <Form.Control type="text" placeholder="Who is
author of this book?" onChange={ (e) => e.target.value === '' ?
setAddAuthor('Null') : setAddAuthor(e.target.value)} />
        </Form.Group>

        { /* form group of publish date */ }
        <Form.Group className="mb-3">
            <Form.Label>Publish Date</Form.Label>
            <Form.Control type="date" placeholder="When this
book been published?" onChange={ (e) => e.target.value === '' ?
setPublishDate('2022-11-24') : setPublishDate(e.target.value)} />
        </Form.Group>

        { /* form group of book ISBN */ }
        <Form.Group className="mb-3">
            <Form.Label>ISBN</Form.Label>
            <Form.Control type="number" placeholder="Tell me
the ISBN number of that book" onChange={ (e) => e.target.value === '' ?
setAddISBN('0000000') : setAddISBN(e.target.value)} />
        </Form.Group>

        { /* form group of book availability status */ }
        <Form.Group className="mb-3">
            <Form.Label>Availability</Form.Label>
            <Form.Control type="text" placeholder="Is this
book available?" disabled defaultValue={isAvailableVal} value={isAvailableVal}
onChange={ () => isAvailableVal} />
        </Form.Group>

        { /* form group of how many people rent that book */ }
        <Form.Group className="mb-3">
            <Form.Label>Total Borrow</Form.Label>
            <Form.Control type="number" min={0} max={5}
placeholder="How many people rent this book currently?"
defaultValue={0} onChange={convertIsAvailableStatus} />
        </Form.Group>

        <div className="button-control-container">
            <Button className="btn-add" type="submit"
variant="success"><b>Upload</b></Button>
            <Button className="btn-add" variant="outline-
danger" onClick={HandlingCancel}><b>Back</b></Button>
        </div>
    </Form>

```

```

        </div>
      </div>
    </div>
  );
}

export default AddBook;

```

Kode untuk menampilkan seluruh buku (**get-all-book.js**):

```

import React from "react";
import '../global-style.css';
import Axios from 'axios';
import { useState, useEffect } from 'react';
import { BookmarkPlusFill } from 'react-bootstrap-icons';
import { PencilFill } from "react-bootstrap-icons";
import { TrashFill } from "react-bootstrap-icons";
import ModalDelete from "../components/modals/modal-delete";
import previewImage from '../assets/images/no-image-placeholder.png';
import ModalEdit from "../components/modals/modal-edit";

const GetAllBook = (props) => {

  // handling dynamic title of this page
  useEffect(() => {
    document.title = props.title;
  }, [props])

  // useState to handling list of book that requested
  const [bookList, setBookList] = useState([]);

  // requesting ALL books
  useEffect(() => {
    Axios.get('http://localhost:3301/books').then((response) => {
      setBookList(response.data);
    })
  }, [])

  // useState to close or open deletion modal
  const [showModalDelete, setShowModalDelete] = useState(false);

  // useState to close or open editing modal
  const [showModalEdit, setShowModalEdit] = useState(false);

  // useState for targeting which data wants to be deleted
  const [target, setTarget] = useState();

```

```

// useState for every data field to be edit
const [getID, setGetID] = useState();
const [title, setTitle] = useState();
const [author, setAuthor] = useState();
const [publishDate, setPublishDate] = useState();
const [ISBN, setISBN] = useState();
const [isAvailable, setIsAvailable] = useState();
const [onBorrow, setOnBorrow] = useState();

var convertPublishDate = '';

return (
  <div className="container-sm">
    <h2 className="title">Showing All Books:</h2>
    {bookList.map((val) => {

      // function to show deletion modal
      const handleShowModalDelete = () => {
        setShowModalDelete(true);
        setTarget(val.book_title);
      }

      // function to show editing modal
      const handleShowModalEdit = () => {
        setShowModalEdit(true);
        setTitle(val.book_title);
        setAuthor(val.book_author);
        convertPublishDate = new
Date(val.book_publishDate).toISOString().split("T")[0];
        setPublishDate(convertPublishDate);
        setISBN(val.book_ISBN);
        setIsAvailable(val.isAvailable);
        setOnBorrow(val.onBorrow);
        setGetID(val.book_id);
      }

      // function to execute deletion
      const submitDeletion = () => {
        console.log(target);
        Axios.delete(`http://localhost:3301/deleteBooks/${target}`
).then(response => {
          setBookList(response.data);
        })
        window.location.reload(false);
      }

      return (
        <div className="card-container">

```



```
export default GetAllBook;
```

Kode untuk melakukan pencarian buku (**search-book.js**):

```
import React from "react";
import '../global-style.css';
import Axios from 'axios';
import { useState, useEffect } from 'react';
import { BookmarkPlusFill } from 'react-bootstrap-icons';
import { PencilFill } from "react-bootstrap-icons";
import { TrashFill } from "react-bootstrap-icons";
import { query } from "../home";
import { useLocation } from "react-router-dom";
import ModalDelete from "../components/modals/modal-delete";
import previewImage from '../assets/images/no-image-placeholder.png';
import ModalEdit from "../components/modals/modal-edit";

const SearchBook = (props) => {

  // handling dynamic title of this page
  useEffect(() => {
    document.title = props.title;
  }, [props])

  // useState to handling list of book that requested
  const [bookList, setBookList] = useState([]);

  // requesting books based on KEYWORD
  useEffect(() => {
    Axios.get('http://localhost:3301/book/search-title?', { params: {
keyword: query }})
      .then(response => {
        setBookList(response.data);
      })
  }, [])

  // useState to close or open deletion modal
  const [showModalDelete, setShowModalDelete] = useState(false);

  // useState for targeting which data wants to be deleted
  const [target, setTarget] = useState();

  // useState to close or open editing modal
```

```

const [showModalEdit, setShowModalEdit] = useState(false);

// useState for every data field to be edit
const [getID, setGetID] = useState();
const [title, setTitle] = useState();
const [author, setAuthor] = useState();
const [publishDate, setPublishDate] = useState();
const [ISBN, setISBN] = useState();
const [isAvailable, setIsAvailable] = useState();
const [onBorrow, setOnBorrow] = useState();

// to convert date
var convertPublishDate = '';

// tracking url changes
const location = useLocation();

// get specific pathname from url
const redirect = location.pathname + location.search;

// search book through the specific pathname
useEffect(() => {
  Axios.get(`http://localhost:3301` + redirect)
    .then(response => {
      setBookList(response.data);
    })
}, [redirect]);

return (
  <div className="container-sm">

    <h2 className="title">Showing All Books:</h2>
    {bookList.map((val, i) => {

      // function to show deletion modal
      const handleShowModalDelete = () => {
        setShowModalDelete(true);
        setTarget(val.book_title);
      }

      // function to show editing modal
      const handleShowModalEdit = () => {
        setShowModalEdit(true);
        setTitle(val.book_title);
        setAuthor(val.book_author);
        convertPublishDate = new
Date(val.book_publishDate).toISOString().split("T")[0];
        setPublishDate(convertPublishDate);

```



```

        setISBN(val.book_ISBN);
        setIsAvailable(val.isAvailable);
        setOnBorrow(val.onBorrow);
        setGetID(val.book_id);
    }

    // function to execute deletion
    const submitDeletion = () => {
        console.log(target);
        Axios.delete(`http://localhost:3301/deleteBooks/${target}`).then(
            response => {
                setBookList(response.data);
            })
        window.location.reload(false);
    }

    return (
        <div className="card-container" key={i}>
            <div className="img-part">
                <img src={previewImage} alt="this is a preview for
book cover"></img>
            </div>

            <div className="desc-part">
                <div className="button-group">
                    <button type="button" className="btn btn-info"
onClick={handleShowModalEdit}><PencilFill style={{width: '17px', height:
'17px', color: 'white'}} /></button>
                    <button type="button" className="btn btn-danger"
onClick={handleShowModalDelete}><TrashFill style={{width: '17px', height:
'17px'}} /></button>

                    {showModalEdit && <ModalEdit
showModalEdit={showModalEdit} setShowModalEdit={setShowModalEdit}
getId={getID} title={title} author={author} publishDate={publishDate}
isbn={ISBN} isAvailable={isAvailable} onBorrow={onBorrow} />}
                    {showModalDelete && <ModalDelete
showModalDelete={showModalDelete} setShowModalDelete={setShowModalDelete}
submitDeletion={submitDeletion} />}
                </div>

                <h4>{val.book_title}</h4>
                <p><i>{val.book_author}</i></p>

                <div>
                    <p className="book-information">Book ISBN:
{val.book_ISBN}</p>

```

```

                <p className="book-information">Published at:
{val.book_publishDate}</p>
            </div>

            <button type="button" className="btn btn-dark"
disabled={val.isAvailable === 0 ? true : false}><BookmarkPlusFill
style={{width: '17px', height: '17px', marginTop: '-2px', marginRight: '5px'}}
/> Add to wishlist</button>

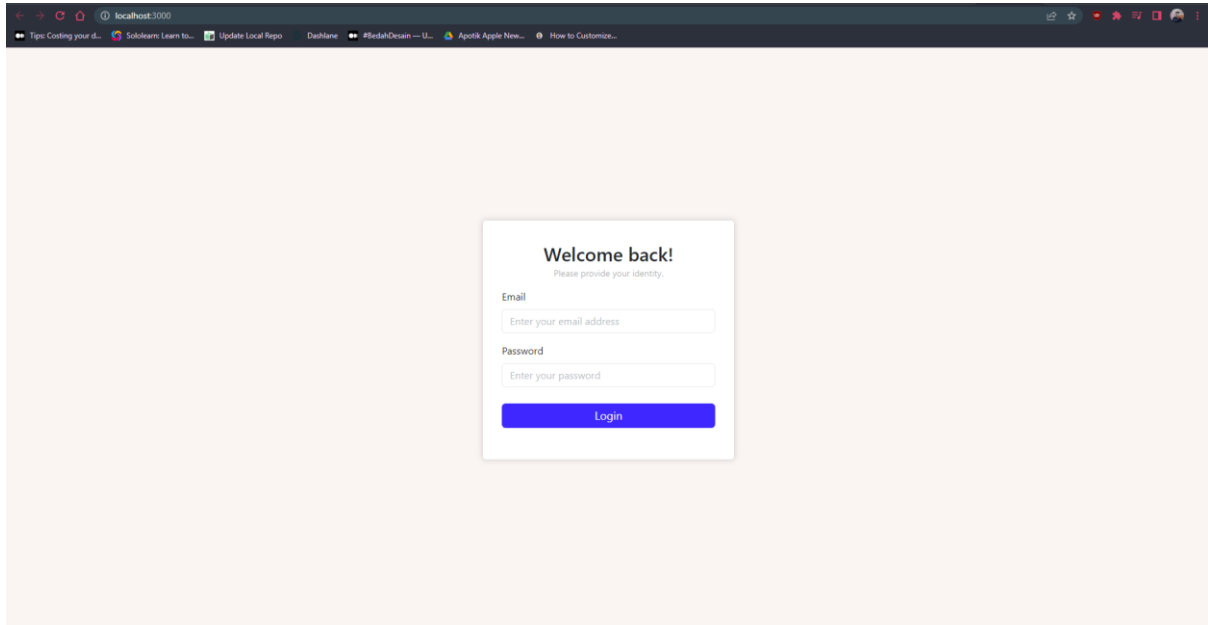
            <p>{val.isAvailable === 0 ? <span
className="info">`\Book is reached maximum rent! (Temporarily not
available)`</span> : <span></span>}</p>
            </div>
        </div>
    )
    }
}
</div>
)
}

export default SearchBook;

```

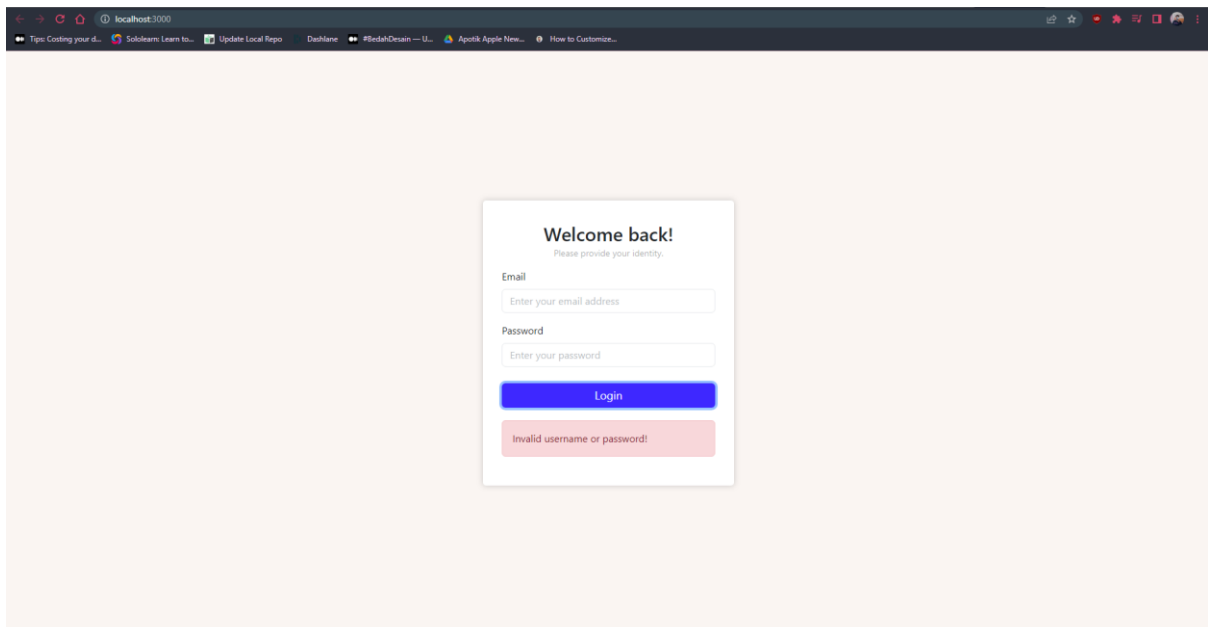
SCREENSHOT / DOCUMENTATION

Berikut adalah tampilan *login page* dari website *Book Rental*. Pengguna perlu menginput *email* dan *password* yang sesuai di *database* untuk dapat mengakses website ini. Jika terdapat ketidakcocokan, maka akan muncul *alerts*.



The screenshot shows a web browser window with the address bar displaying 'localhost:3000'. The page has a light beige background. In the center, there is a white login card with a subtle shadow. The card contains the text 'Welcome back!' followed by 'Please provide your identity.' Below this, there are two input fields: 'Email' with the placeholder 'Enter your email address' and 'Password' with the placeholder 'Enter your password'. At the bottom of the card is a blue 'Login' button.

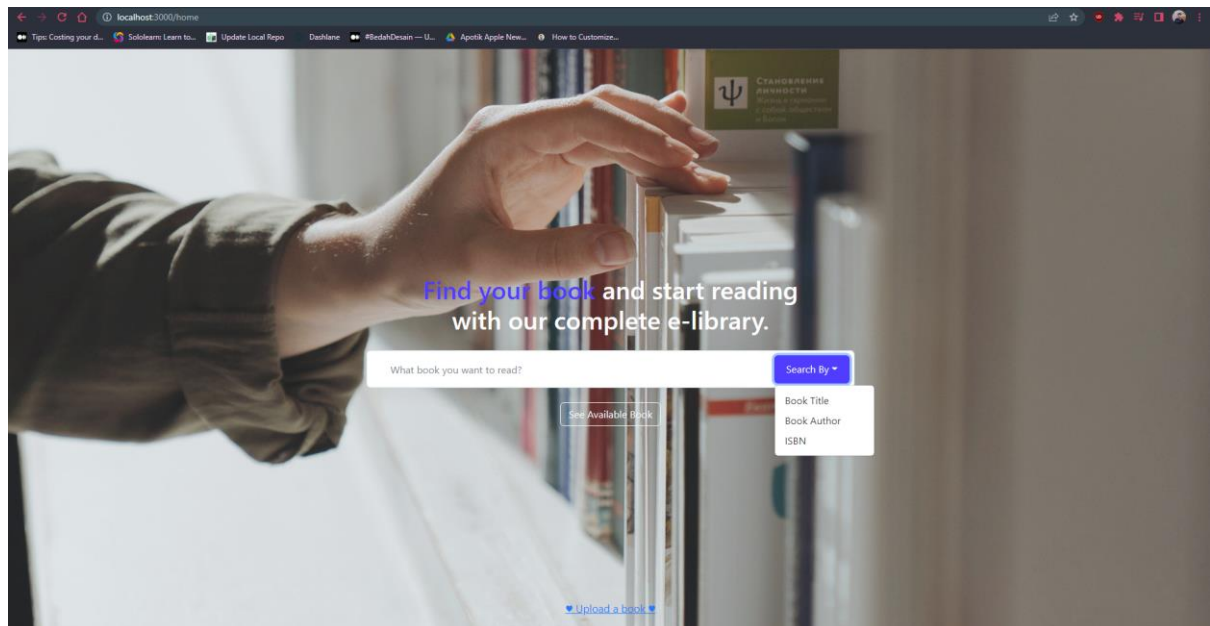
Tampilan awal halaman login



This screenshot shows the same login page as the previous one, but with an error message. Below the blue 'Login' button, a red error box appears with the text 'Invalid username or password!'.

Login page jika username atau password tidak valid

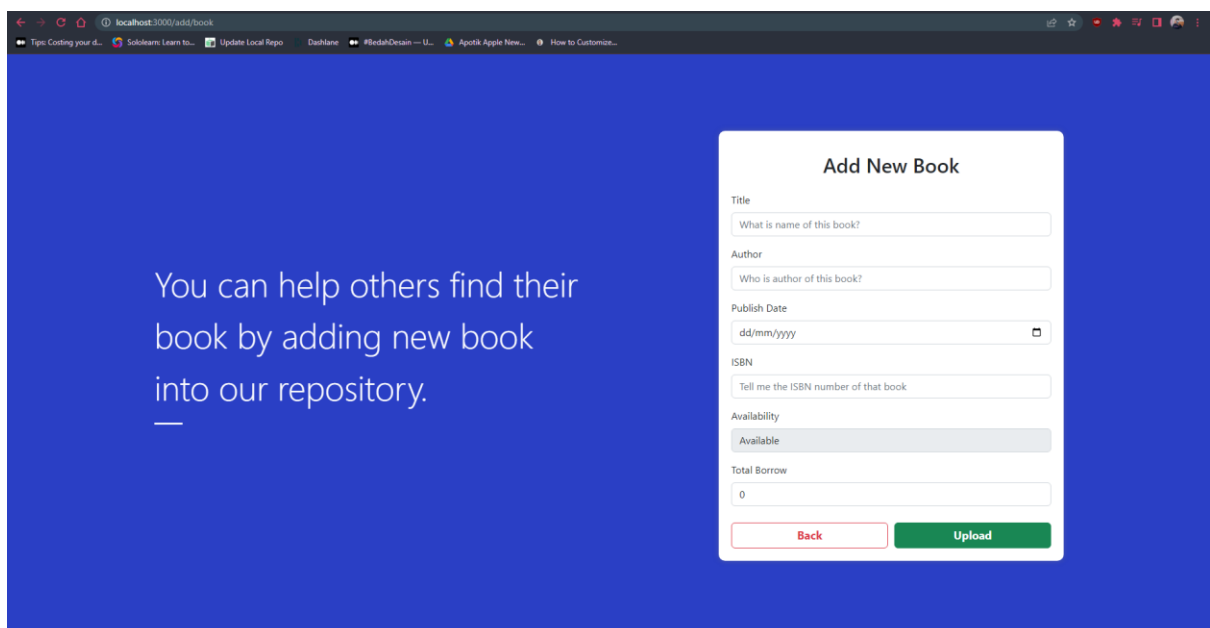
Apabila pengguna berhasil melakukan login, maka pengguna akan diarahkan ke halaman utama. Berikut adalah tampilan halaman utama:



Tampilan dari halaman utama

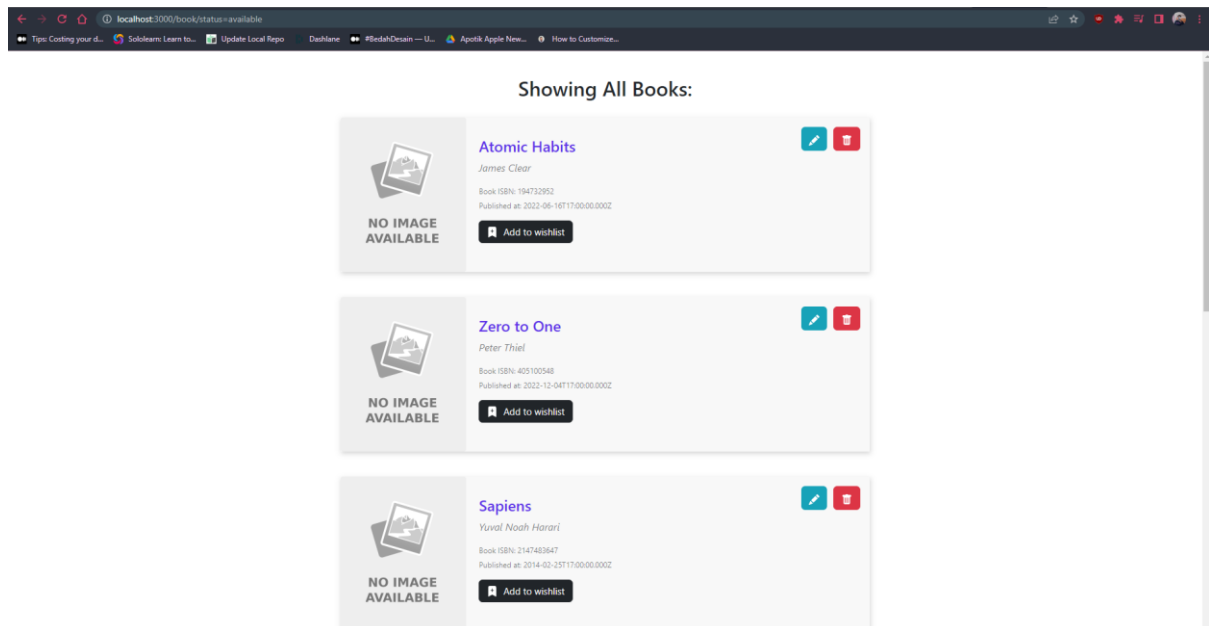
Terlihat bahwa disini terdapat beberapa *button* yang dapat diakses pengguna. Button di bagian bawah berfungsi untuk mengunggah buku ke server. Tombol yang bertuliskan **See Available Book** merupakan tombol untuk melakukan filterisasi buku berdasarkan ketersediaan.

Selanjutnya, pengguna dapat melakukan pencarian buku dan menekan tombol dropdown untuk melakukan filterisasi berdasarkan nama buku, penulis buku, ataupun nomor ISBN. Berikut adalah halaman yang akan tertuju apabila pengguna menekan tombol **upload a book**:



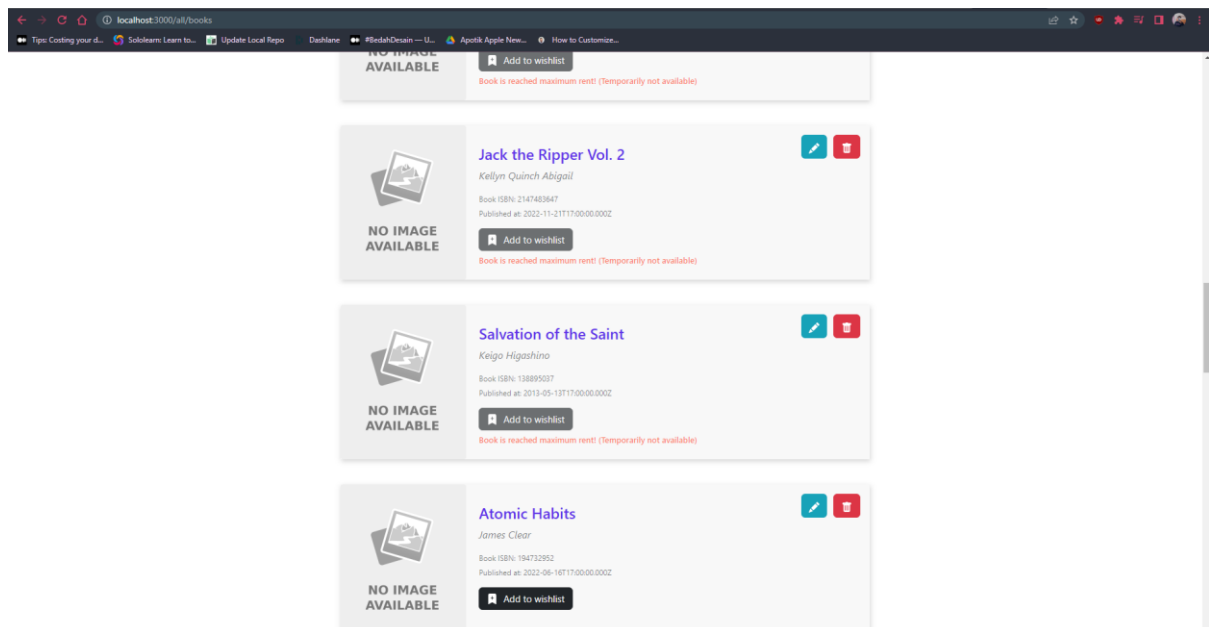
Tampilan dari halaman upload a book

Namun, jika pengguna menekan tombol **See Available Book**, maka inilah yang akan ditampilkan kepada pengguna:



Tampilan jika pengguna menekan tombol See Available Book

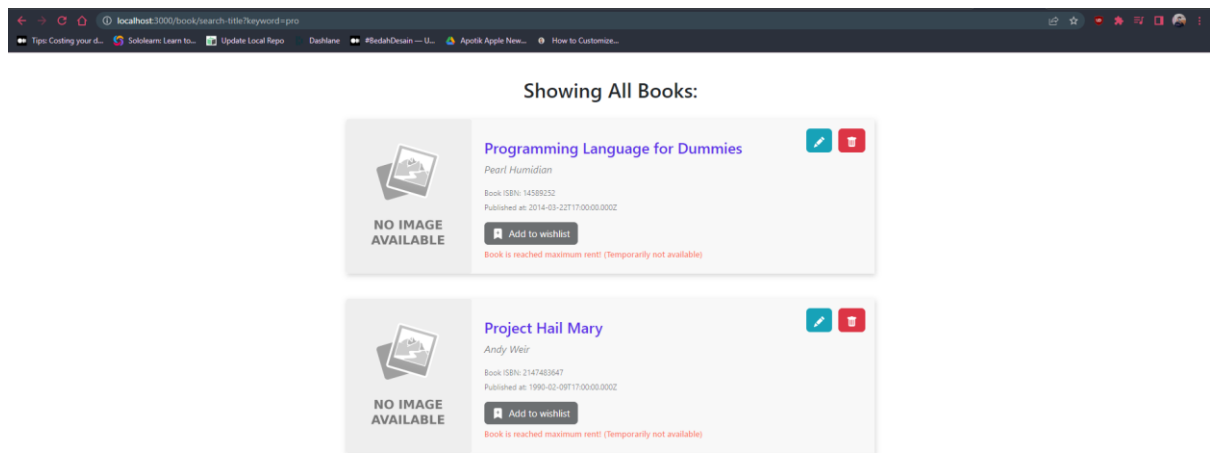
Disini, kita dapat melihat semua buku yang tersedia dan dapat dimasukkan ke wishlist. Pengguna juga dapat melihat seluruh buku yang ada pada server. Berikut adalah tampilan nya:



Tampilan dari seluruh buku yang ada di server

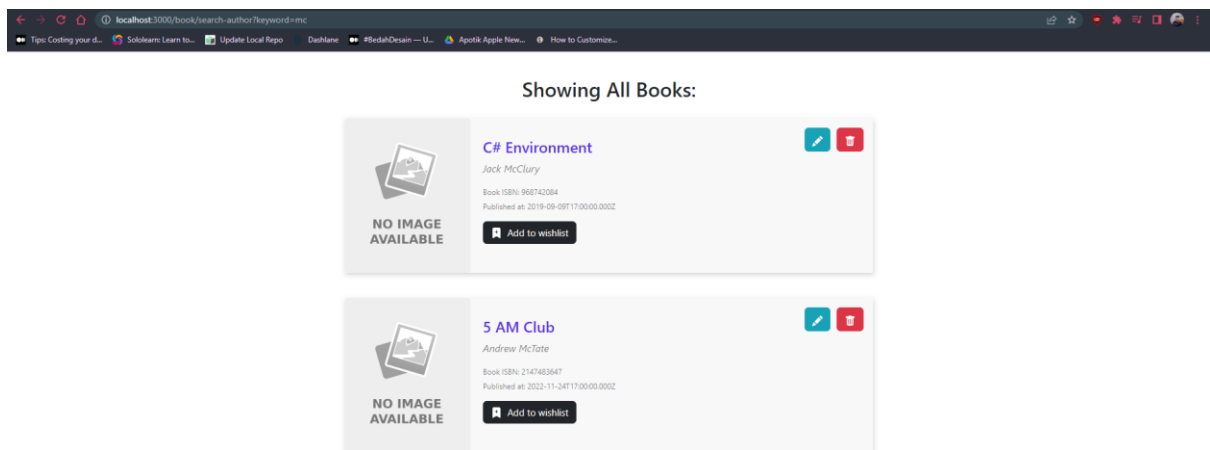
Buku yang telah mencapai maksimum peminjaman akan mendapat alert dan pengguna tidak dapat menekan tombol **Add to Wishlist**. Sedangkan buku yang belum mencapai maksimum peminjaman tidak terdapat alert dan pengguna dapat menekan tombol **Add to Wishlist**.

Jika pengguna mengetik “**pro**” pada halaman utama lalu melakukan filterisasi data berdasarkan **nama buku**, maka sistem akan menampilkan seluruh buku yang mengandung huruf “pro” di judulnya.



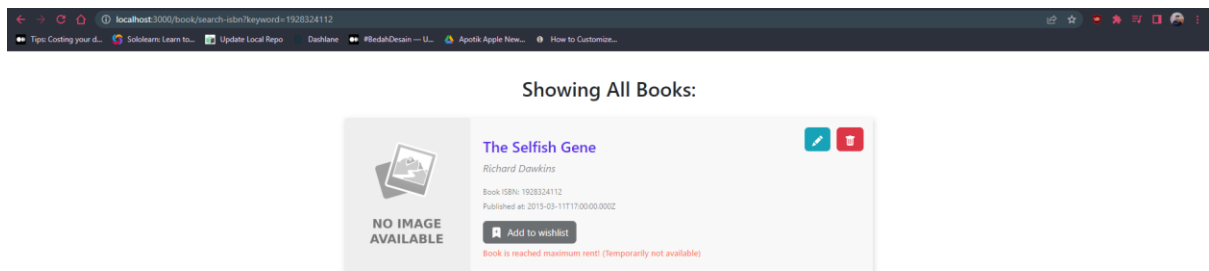
Tampilan jika pengguna mengetik keyword “pro” pada search input search by book title

Jika pengguna mengetik “**mc**” pada halaman utama lalu melakukan filterisasi data berdasarkan **penulis buku**, maka sistem akan menampilkan seluruh buku yang mengandung huruf “mc” pada nama penulis bukunya.



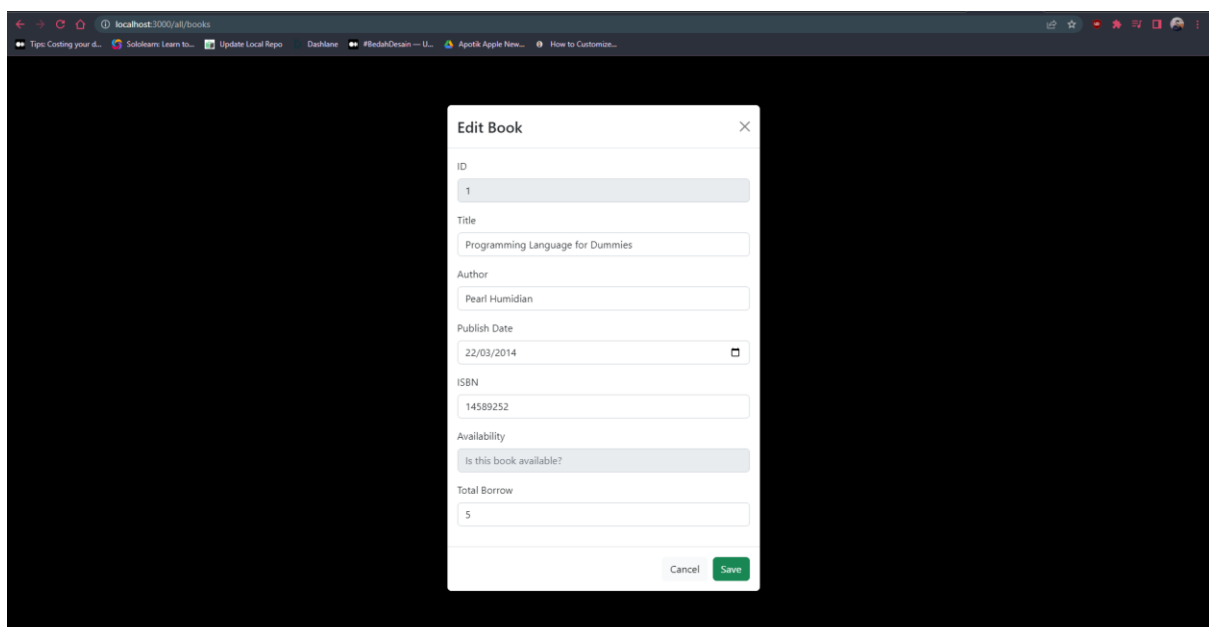
Tampilan jika pengguna mengetik keyword “mc” pada search input dan melakukan search by book author

Jika pengguna mengetikkan nomor ISBN pada halaman utama lalu melakukan filterisasi data berdasarkan **nomor ISBN**, maka sistem akan menampilkan seluruh buku yang mengandung nomor ISBN pada buku tersebut.



Tampilan jika pengguna mengetikkan nomor ISBN pada search input dan melakukan search by ISBN

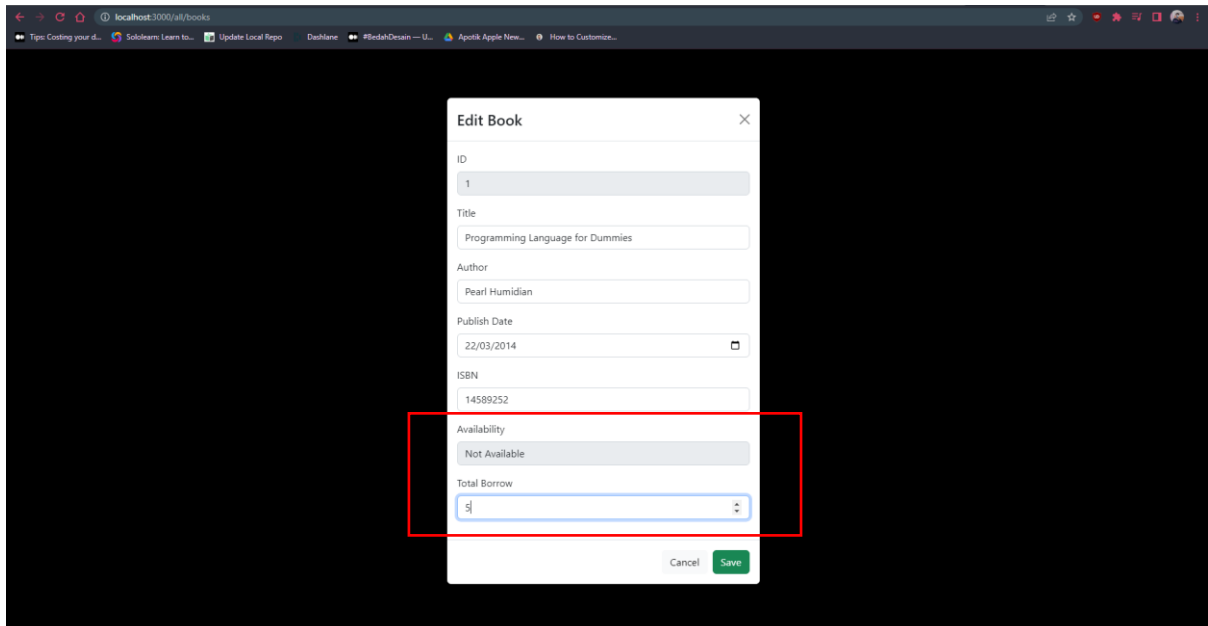
Selanjutnya, gambar dibawah ini merupakan tampilan apabila *admin* ingin melakukan penyuntingan buku:



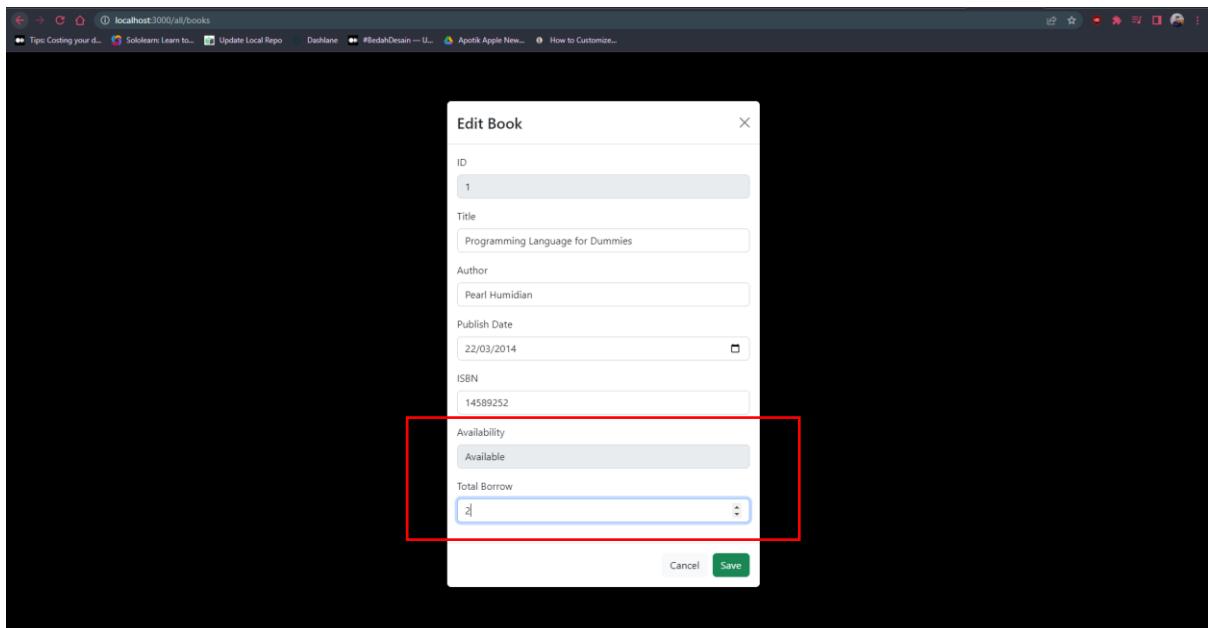
Tampilan modal apabila admin ingin melakukan penyuntingan buku

Disini, input **availability** akan berubah secara dinamis tergantung dari input **total borrow**. Apabila total borrownya menyentuh 5, maka availability akan berubah menjadi **not available**.

Hal ini mengindikasikan bahwa buku tidak tersedia sementara waktu sampai ada salah satu pengguna yang mengembalikan buku ini dari wishlist mereka.

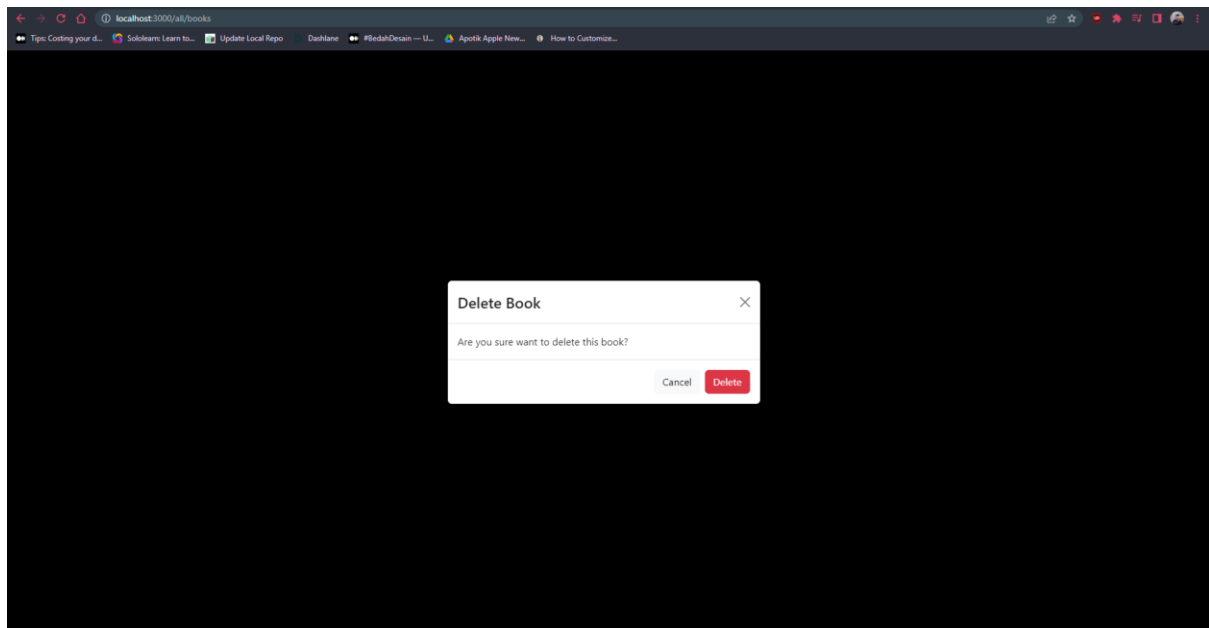
A screenshot of a web browser showing a modal titled "Edit Book". The modal contains several input fields: ID (1), Title (Programming Language for Dummies), Author (Pearl Humidian), Publish Date (22/03/2014), and ISBN (14589252). Below these, the "Availability" field is set to "Not Available" and the "Total Borrow" field is a dropdown menu set to "5". A red rectangle highlights the "Availability" and "Total Borrow" fields. At the bottom right of the modal are "Cancel" and "Save" buttons. The browser's address bar shows "localhost:3000/all/books".

Tampilan modal apabila admin memberikan angka 5 pada input total borrow

A screenshot of the same "Edit Book" modal. In this state, the "Availability" field is set to "Available" and the "Total Borrow" dropdown menu is set to "2". A red rectangle highlights the "Availability" and "Total Borrow" fields. The "Cancel" and "Save" buttons remain at the bottom right. The browser's address bar shows "localhost:3000/all/books".

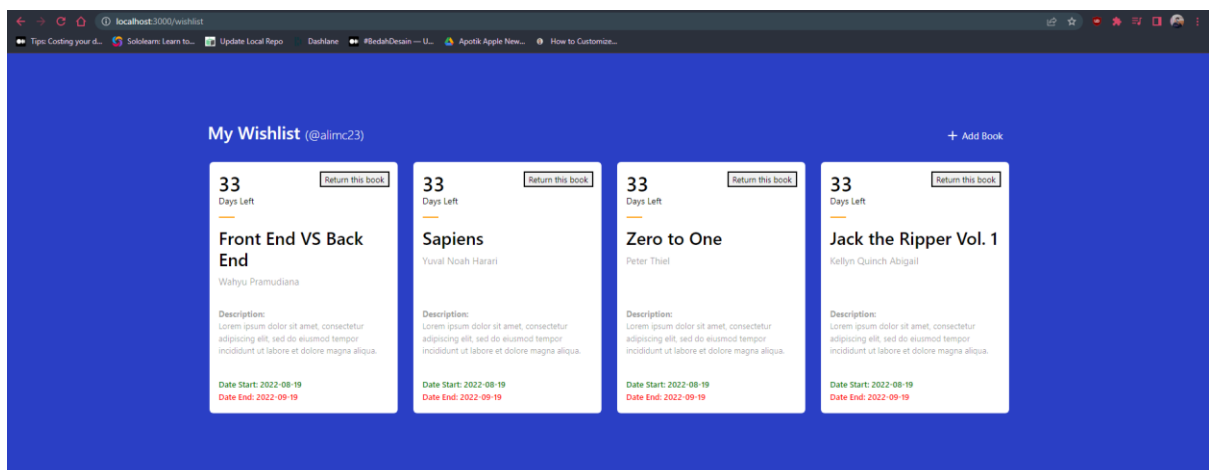
Tampilan modal apabila admin memberikan di bawah 5 pada input total borrow

Gambar dibawah ini merupakan tampilan apabila *admin* ingin melakukan penghapusan data:



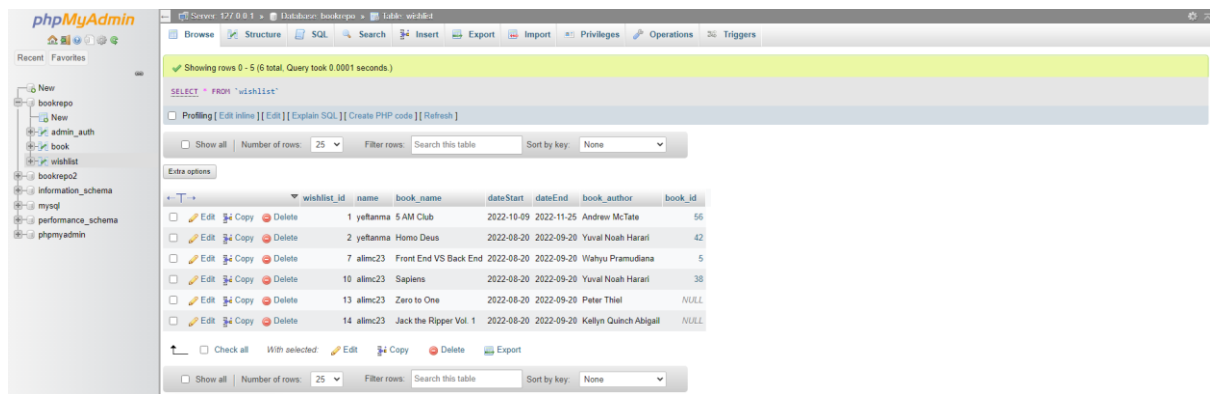
Tampilan modal apabila admin menghapus buku

Kemudian, ini adalah halaman *wishlist* yang akan ditampilkan apabila pengguna memasukkan buku yang diinginkan ke dalam *wishlist* mereka:



Tampilan halaman wishlit apabila user menambahkan buku ke wishlist

Keempat buku ini diambil dari tabel wishlist pada database:



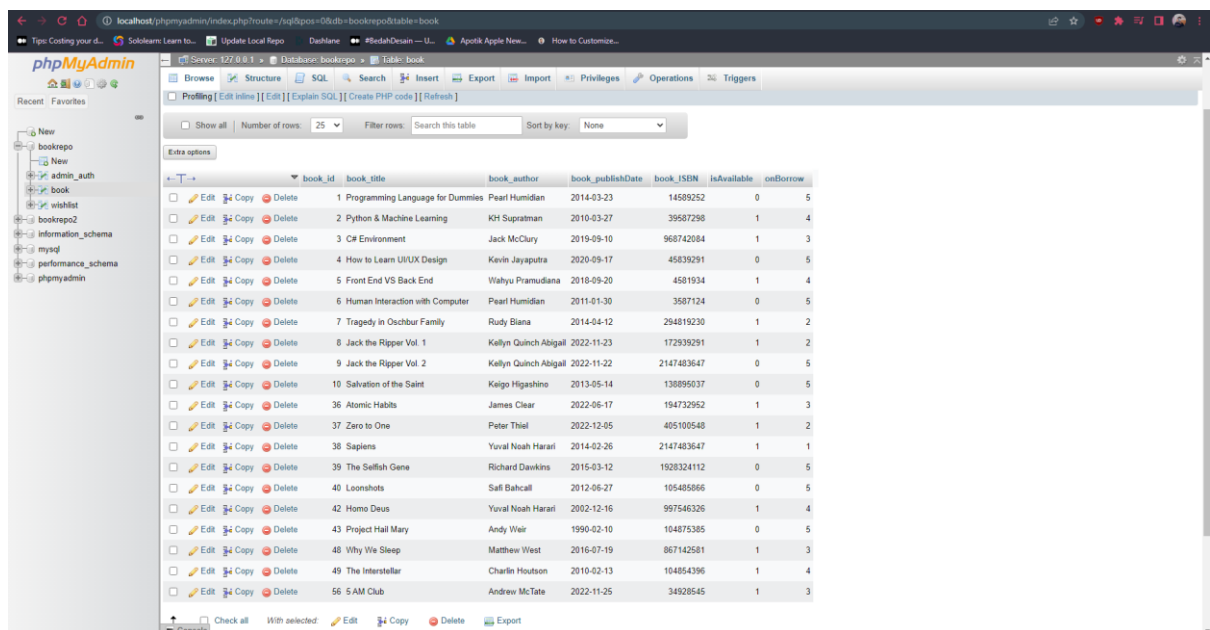
Showing rows 0 - 5 (5 total. Query took 0.0001 seconds)

```
SELECT * FROM `wishlist`
```

Number of rows: 25

wishlist_id	name	book_name	dateStart	dateEnd	book_author	book_id
1	yefanna	5 AM Club	2022-10-09	2022-11-25	Andrew McTate	56
2	yefanna	Homo Deus	2022-08-20	2022-09-20	Yuval Noah Harari	42
7	alimc23	Front End VS Back End	2022-08-20	2022-09-20	Wahyu Pramudiana	5
10	alimc23	Sapiens	2022-08-20	2022-09-20	Yuval Noah Harari	38
13	alimc23	Zero to One	2022-08-20	2022-09-20	Peter Thiel	NULL
14	alimc23	Jack the Ripper Vol. 1	2022-08-20	2022-09-20	Kellyn Quinch Abigail	NULL

Tampilan tabel wishlist pada database



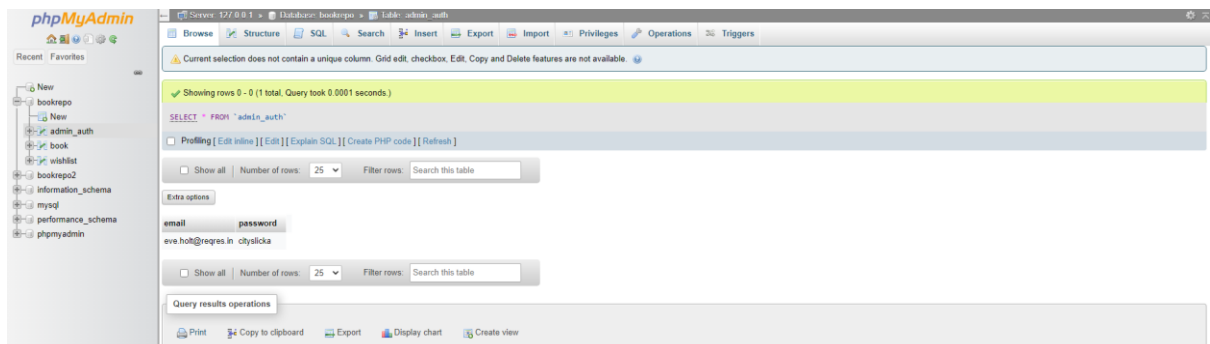
Showing rows 0 - 55 (56 total. Query took 0.0001 seconds)

```
SELECT * FROM `book`
```

Number of rows: 25

book_id	book_title	book_author	book_publishDate	book_ISBN	isAvailable	onBorrow
1	Programming Language for Dummies	Pearl Humidian	2014-03-23	14589252	0	5
2	Python & Machine Learning	KH Supratman	2010-03-27	39587296	1	4
3	C# Environment	Jack McClary	2019-09-10	968742084	1	3
4	How to Learn UNIX Design	Kevin Jayaputra	2020-09-17	45839291	0	5
5	Front End VS Back End	Wahyu Pramudiana	2018-09-20	4581934	1	4
6	Human Interaction with Computer	Pearl Humidian	2011-01-30	3587124	0	5
7	Tragedy in Oshbur Family	Rudy Biana	2014-04-12	254819230	1	2
8	Jack the Ripper Vol. 1	Kellyn Quinch Abigail	2022-11-23	172939291	1	2
9	Jack the Ripper Vol. 2	Kellyn Quinch Abigail	2022-11-22	2147483647	0	5
10	Salvation of the Saint	Keigo Higashino	2013-05-14	130895037	0	5
36	Atomic Habits	James Clear	2022-06-17	194732952	1	3
37	Zero to One	Peter Thiel	2022-12-05	405100548	1	2
38	Sapiens	Yuval Noah Harari	2014-02-26	2147483647	1	1
39	The Selfish Gene	Richard Dawkins	2015-03-12	1928324112	0	5
40	Loonshots	Safi Bahcall	2012-06-27	105485666	0	5
42	Homo Deus	Yuval Noah Harari	2002-12-16	997546326	1	4
43	Project Hall Mary	Andy Weir	1990-02-10	104875385	0	5
48	Why We Sleep	Matthew West	2016-07-19	867142581	1	3
49	The Interstellar	Charlin Houtson	2010-02-13	104854396	1	4
56	5 AM Club	Andrew McTate	2022-11-25	34928545	1	3

Tampilan tabel book pada database



Showing rows 0 - 0 (1 total. Query took 0.0001 seconds)

```
SELECT * FROM `admin_auth`
```

Number of rows: 25

email	password
eve.holt@regres.in	chylsicka

Query results operations

Print Copy to clipboard Export Display chart Create view

Tampilan tabel admin_auth pada database

Daftar Referensi

1. Semua *image* pada website diambil dari unsplash.com
2. CSS framework yang saya gunakan adalah bootstrap dan react-bootstrap.
3. Ikon yang tertera pada website diambil dari bootstrap-icons.

“Sekian dari laporan ini saya buat. Terima Kasih”