Dokumen Sistem Informasi Perpustakaan New Generation

Created by
Kelompok 04

1. 12S20006 Nadya D. Tambunan
2. 12S20008 Putri Tampubolon
3. 12S20009 Agnes Marpaung
4. 12S20011 Gabriel Panggabean
5. 12S20013 Lydia Tesalonika
6. 12S20015 Niver Titi Sibuea
7. 12S20035 Nemnem Sihombing
8. 12S20036 Winda Sari ButarButar
9. 12S20053 Andri A Hutapea
10. 12S20056 Siska R Manullang

Fakultas Informatika dan Teknik Elektro

S1 Sistem Informasi

2020/2021

1. Deskripsi Singkat Proyek

Sistem Informasi Perpustakaan New Generation merupakan suatu media yang digunakan untuk memenuhi kebutuhan user dalam meminjam, mencari dan mengembalikan buku. Sistem ini dapat menyimpan semua data buku, data user, dan data pengembalian buku. Pada Sistem Informasi Perpustakaan ini seorang pengunjung dapat meminjam buku yang diinginkan, dan mengembalikan buku yang dipinjam.

2. Functional Requirement

2.1 Peminjaman buku

Peminjaman buku merupakan kegiatan yang dilakukan oleh user, dimana user yang akan meminjam buku telah terdaftar sebagai anggota perpustakaan. Adapun tujuan adanya sistem informasi perpustakaan new generation, diharapkan kegiatan peminjaman buku ini dapat memudahkan admin dalam melakukan pengisian data-data user yang akan melakukan peminjaman buku. Data-data yang akan diisi adalahid, ISBN, borrow date, title, author dan return date. Setelah data-data tersebut dimasukkan, sistem akan menyimpan data tersebut sehingga akan dicek dan disesuaikan kembali pada pengembalian buku nantinya.

2.2 Pengembalian buku

Pengembalian buku merupakan kegiatan yang dilakukan oleh user, dimana user akan mengembalikan buku yang telah dipinjam dahulu. Pada sistem informasi perpustakaan new generation ini Sistem dapat mengelola pengembalian buku. User dapat mencatat semua transaksi pengembalian buku. User dapat menampilkan no ISBN buku, Borrow date book, Return date book, dan Status Book. User dapat melihat tanggal buku dipinjam dan tanggal pengembalian buku.

2.3 Mengelola data buku

Mengelola data buku merupakan rules yang dimiliki oleh admin. Pada sistem informasi perpustakaan new generation admin dapat mengedit data buku yang ada pada sistem. Data-data yang dimaksud adalah meliputi : ISBN, judul buku, nama buku, pengarang dan juga jenis buku. Keberadaan data buku ini sendiri dapat mempermudah user atau pengunjung dalam mencari buku, dan melakukan peminjaman buku yang dibutuhkan.

2.4 Autentifikasi

- **login**

  Pada sistem informasi perpustakaan new generation agar admin dan user dapat masuk ke dalam system dibutuhkan login terlebih dahulu. Pada saat login, admin dan user harus mengisi email address dan password yang sesuai dengan milik pengguna. Sebelum melakukan login harus dipastikan akun dari pengguna telah terdaftar dan pengguna dapat masuk kedalam system.

- **logout**

Pada sistem informasi perpustakaan new generation setelah seorang admin atau user melakukan login untuk menggunakan sistem tersebut, maka untuk keluar dari sistem tersebut dilakukan logout pada akun pengguna di dalam sistem.

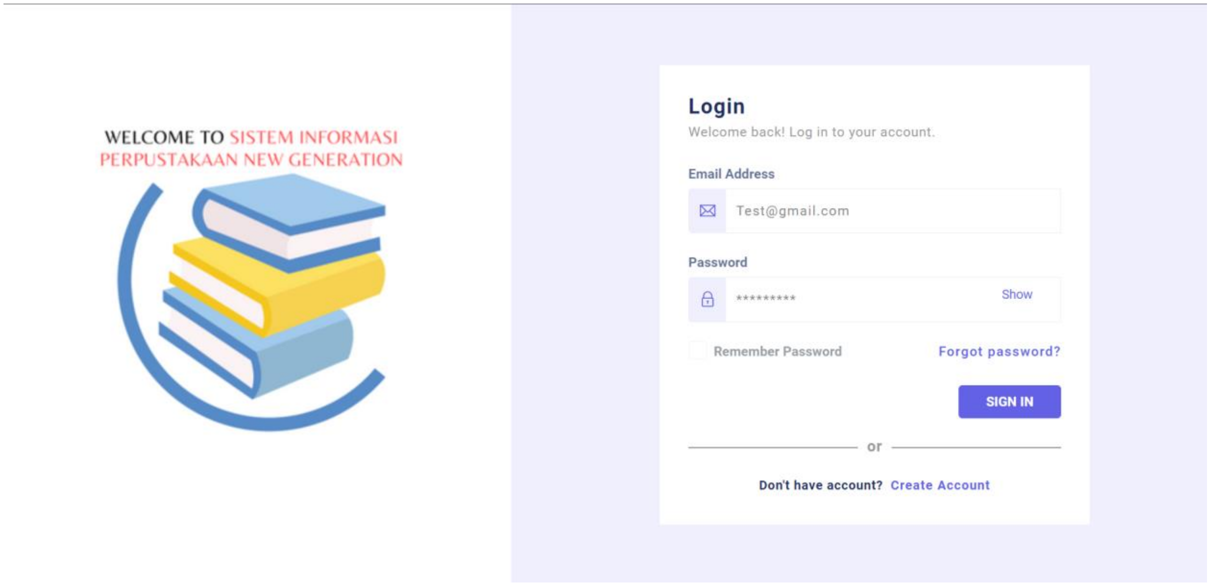
3. Non Functional Requirement

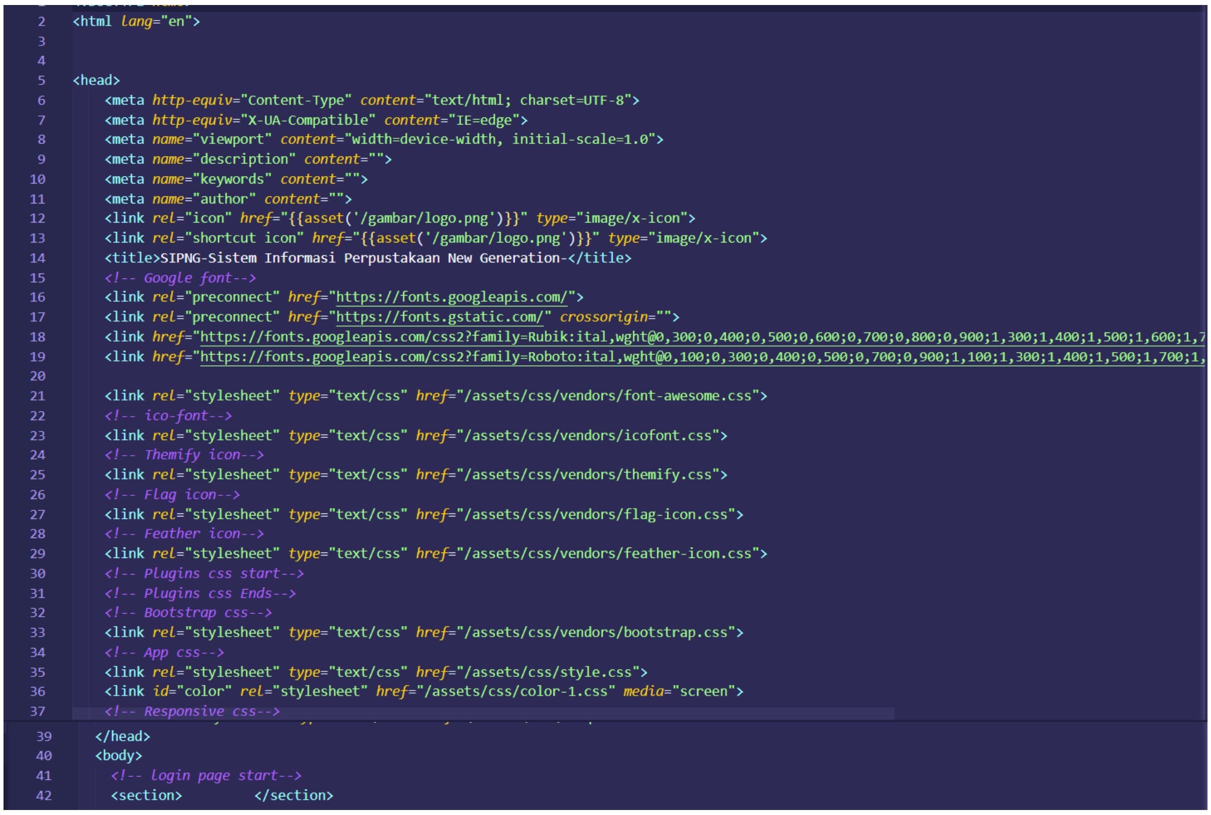
3.1 Pengguna mudah memahami tampilan program
3.2 User Friendly dan User Experience
3.3 Pengguna Mudah Mengoperasikannya
3.4 Terdapat validasi untuk mengurangi kesalahan data

4. Tampilan setiap Functional Requirement dan kodenya (NFR yg no 4)

4.1 Tampilkan ss website dan Code

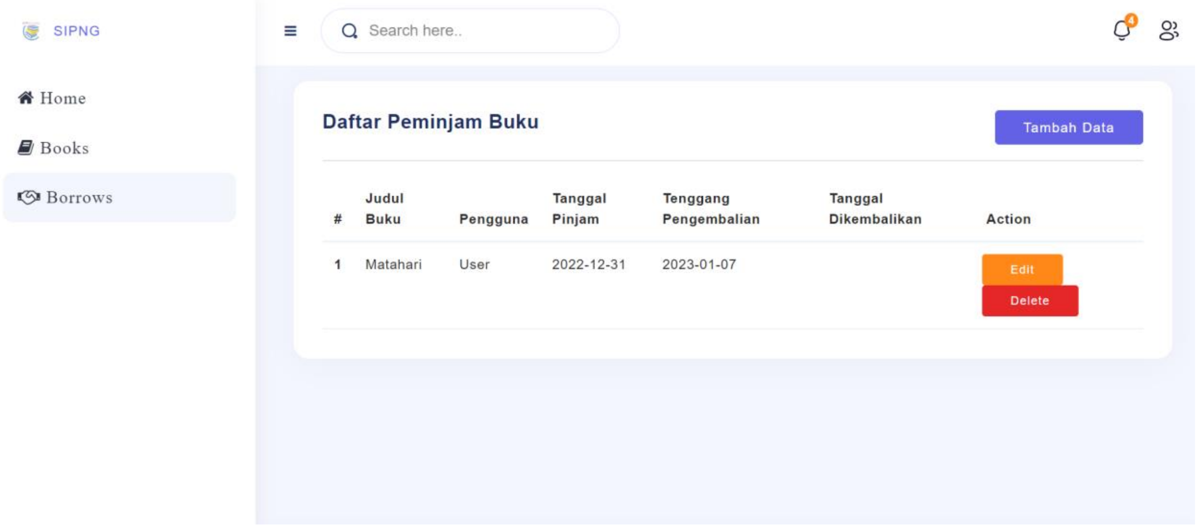
- **Autentifikasi**


WELCOME TO SISTEM INFORMASI PERPUSTAKAAN NEW GENERATION
Login
Welcome back! Log in to your account.
Email Address
Test@gmail.com
Password
****
Show
Remember Password
Forgot password?
SIGN IN
or
Don't have account? Create Account



```html
<html lang="en">


<head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="">
    <meta name="keywords" content="">
    <meta name="author" content="">
    <link rel="icon" href="{{asset('/gambar/logo.png')}}" type="image/x-icon">
    <link rel="shortcut icon" href="{{asset('/gambar/logo.png')}}" type="image/x-icon">
    <title>SIPNG-Sistem Informasi Perpustakaan New Generation-</title>
    <!-- Google font-->
    <link rel="preconnect" href="https://fonts.googleapis.com/">
    <link rel="preconnect" href="https://fonts.gstatic.com/" crossorigin="">
    <link href="https://fonts.googleapis.com/css2?family=Rubik:ital,wght@0,300;0,400;0,500;0,600;0,700;0,800;0,900;1,300;1,400;1,500;1,600;1,7
    <link href="https://fonts.googleapis.com/css2?family=Roboto:ital,wght@0,100;0,300;0,400;0,500;0,700;0,900;1,100;1,300;1,400;1,500;1,700;1,

    <link rel="stylesheet" type="text/css" href="/assets/css/vendors/font-awesome.css">
    <!-- ico-font-->
    <link rel="stylesheet" type="text/css" href="/assets/css/vendors/icofont.css">
    <!-- Themify icon-->
    <link rel="stylesheet" type="text/css" href="/assets/css/vendors/themify.css">
    <!-- Flag icon-->
    <link rel="stylesheet" type="text/css" href="/assets/css/vendors/flag-icon.css">
    <!-- Feather icon-->
    <link rel="stylesheet" type="text/css" href="/assets/css/vendors/feather-icon.css">
    <!-- Plugins css start-->
    <!-- Plugins css Ends-->
    <!-- Bootstrap css-->
    <link rel="stylesheet" type="text/css" href="/assets/css/vendors/bootstrap.css">
    <!-- App css-->
    <link rel="stylesheet" type="text/css" href="/assets/css/style.css">
    <link id="color" rel="stylesheet" href="/assets/css/color-1.css" media="screen">
    <!-- Responsive css-->
</head>
<body>
  <!-- login page start-->
  <section>          </section>
```

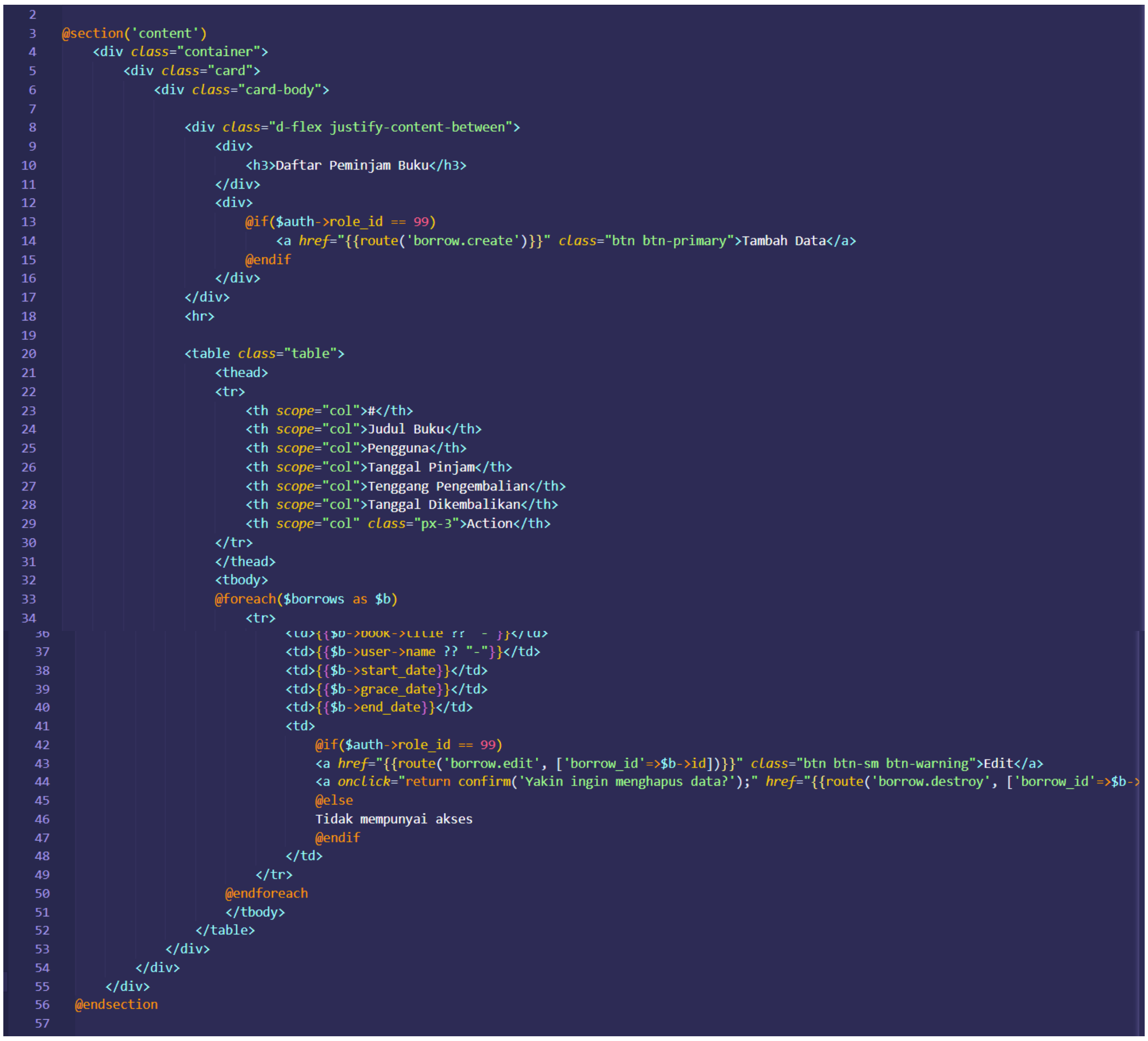
- **Peminjaman**



Daftar Peminjam Buku

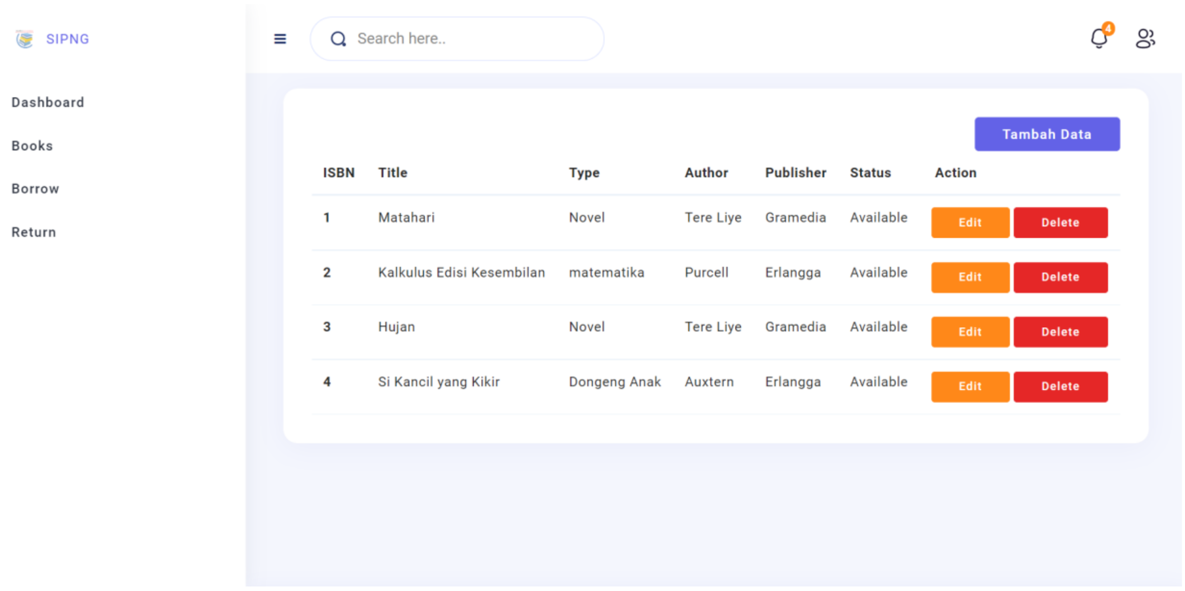
Tambah Data

#	Judul Buku	Pengguna	Tanggal Pinjam	Tenggang Pengembalian	Tanggal Dikembalikan	Action
1	Matahari	User	2022-12-31	2023-01-07		Edit Delete

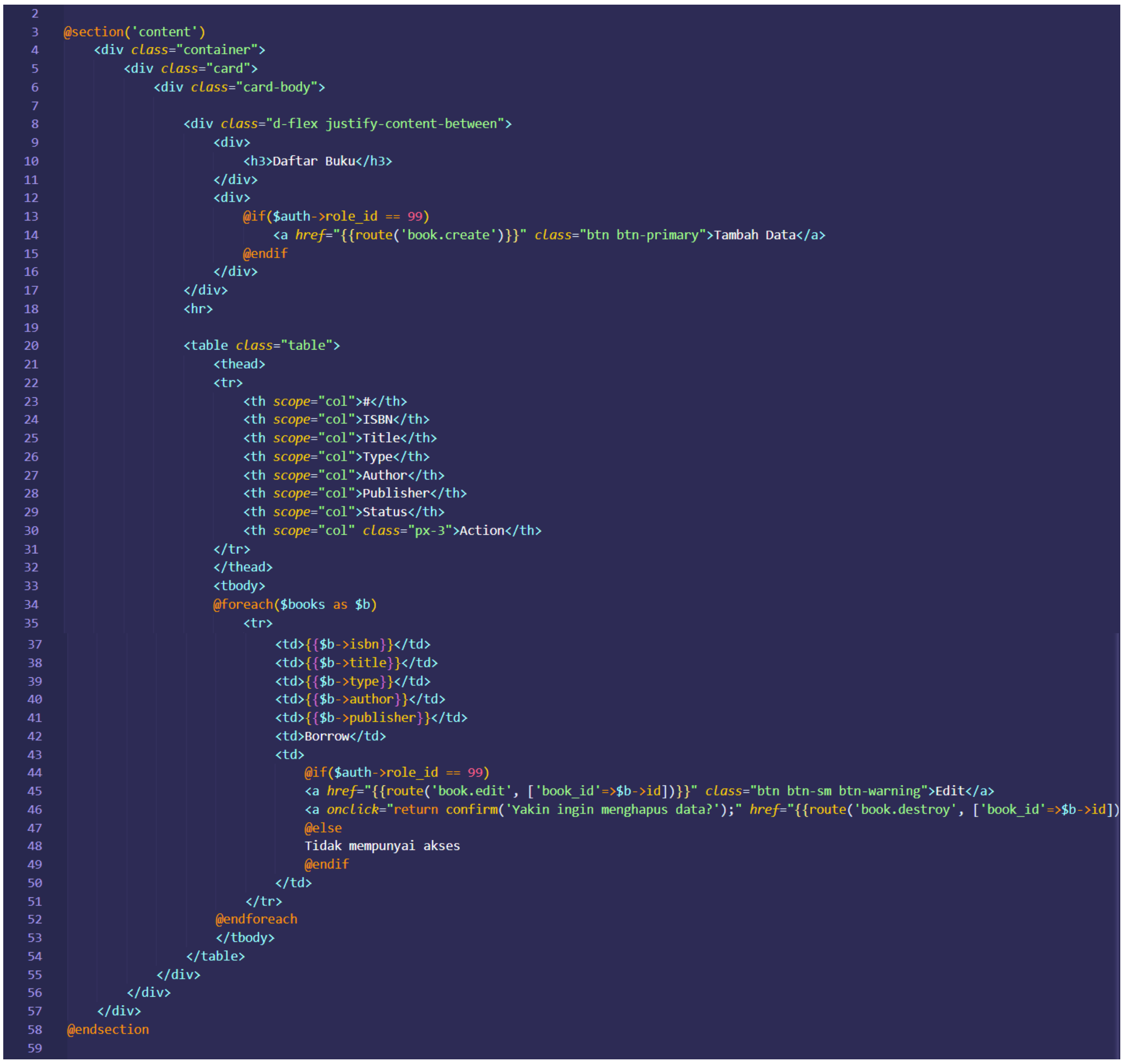


```
@section('content')
    <div class="container">
        <div class="card">
            <div class="card-body">

                <div class="d-flex justify-content-between">
                    <div>
                        <h3>Daftar Peminjam Buku</h3>
                    </div>
                    <div>
                        @if($auth->role_id == 99)
                            <a href="{{route('borrow.create')}}" class="btn btn-primary">Tambah Data</a>
                        @endif
                    </div>
                </div>
                <hr>

                <table class="table">
                    <thead>
                    <tr>
                        <th scope="col">#</th>
                        <th scope="col">Judul Buku</th>
                        <th scope="col">Pengguna</th>
                        <th scope="col">Tanggal Pinjam</th>
                        <th scope="col">Tenggang Pengembalian</th>
                        <th scope="col">Tanggal Dikembalikan</th>
                        <th scope="col" class="px-3">Action</th>
                    </tr>
                    </thead>
                    <tbody>
                    @foreach($borrows as $b)
                        <tr>
                            <td>{{$b->book->title ?? "-"}}</td>
                            <td>{{$b->user->name ?? "-"}}</td>
                            <td>{{$b->start_date}}</td>
                            <td>{{$b->grace_date}}</td>
                            <td>{{$b->end_date}}</td>
                            <td>
                                @if($auth->role_id == 99)
                                <a href="{{route('borrow.edit', ['borrow_id'=>$b->id])}}" class="btn btn-sm btn-warning">Edit</a>
                                <a onclick="return confirm('Yakin ingin menghapus data?');" href="{{route('borrow.destroy', ['borrow_id'=>$b-
                                @else
                                Tidak mempunyai akses
                                @endif
                            </td>
                        </tr>
                    @endforeach
                    </tbody>
                </table>
            </div>
        </div>
    </div>
@endsection
```

- **Data buku**



ISBN	Title	Type	Author	Publisher	Status	Action
1	Matahari	Novel	Tere Liye	Gramedia	Available	Edit Delete
2	Kalkulus Edisi Kesembilan	matematika	Purcell	Erlangga	Available	Edit Delete
3	Hujan	Novel	Tere Liye	Gramedia	Available	Edit Delete
4	Si Kancil yang Kikir	Dongeng Anak	Auxtern	Erlangga	Available	Edit Delete



```
@section('content')
    <div class="container">
        <div class="card">
            <div class="card-body">

                <div class="d-flex justify-content-between">
                    <div>
                        <h3>Daftar Buku</h3>
                    </div>
                    <div>
                        @if($auth->role_id == 99)
                            <a href="{{route('book.create')}}" class="btn btn-primary">Tambah Data</a>
                        @endif
                    </div>
                </div>
                <hr>

                <table class="table">
                    <thead>
                    <tr>
                        <th scope="col">#</th>
                        <th scope="col">ISBN</th>
                        <th scope="col">Title</th>
                        <th scope="col">Type</th>
                        <th scope="col">Author</th>
                        <th scope="col">Publisher</th>
                        <th scope="col">Status</th>
                        <th scope="col" class="px-3">Action</th>
                    </tr>
                    </thead>
                    <tbody>
                    @foreach($books as $b)
                        <tr>
                            <td>{{$b->isbn}}</td>
                            <td>{{$b->title}}</td>
                            <td>{{$b->type}}</td>
                            <td>{{$b->author}}</td>
                            <td>{{$b->publisher}}</td>
                            <td>Borrow</td>
                            <td>
                                @if($auth->role_id == 99)
                                <a href="{{route('book.edit', ['book_id'=>$b->id])}}" class="btn btn-sm btn-warning">Edit</a>
                                <a onclick="return confirm('Yakin ingin menghapus data?');" href="{{route('book.destroy', ['book_id'=>$b->id])
                                @else
                                Tidak mempunyai akses
                                @endif
                            </td>
                        </tr>
                    @endforeach
                    </tbody>
                </table>
            </div>
        </div>
    </div>
@endsection
```

5. Desain API, rules yang diaplikasikan, payload

5.1 Menambahkan data buku

General Information		
Resource URI	:	books/
Business process	:	[FR01] Menambahkan data buku
Tujuan	:	Menambah satu data buku ke dalam *collection*
Responsible Member	:	Putri Rachel Geby Tampubolon (@putritampubolon08)
Applied Rules	:	#1: Forward slash “/” #2: No trailing forward slash “/” #4: Underscore “-” should be avoided #5: Use lowercase …
Developer	:	Putri Rachel Geby Tampubolon
Request		

HTTP Verb	:	POST
Authenticated?	:	Yes, with token (*anonymous access is not allowed*)
User group	:	Admin (*otherwise forbidden*)
Has payload data	:	Yes (JSON)

Data structure	:	{
		"id": 1,
		"isbn": "1111135",
		"title": "Si Kancil yang Kikir",
		"tipe": "Dongeng Anak",
		"author": "Auxtern",
		"publisher": "Erlangga",
		"status": "Available",
		"created_at": null
		},
		"links": {
		"display_book": "/api/book",
		"add_book": "/api/book/add"
		}

Applied Rules	:	#19 : Using POST to create a new resource book …
Possible Response	**:**	**Response 1**
HTTP Status Code	:	201 (Created)
Has payload data	:	Yes (JSON)

Data structure	:	{
		"status": true,
		"message": "Berhasil menambahkan data buku"
		"data": {
		"total": 2,
		"books": [
		{
		"id": 1,
		"isbn": "1111135",
		"title": "Si Kancil yang Kikir"

		"type": "Dongen Anak",
		"author": "Auxtern",
		"publisher": "Erlangga",
		"status": "Available",
		"created_at": null,
		},
		}
		
		"links": {
		
		"display_book": "/api/book",
		
		"add_book": "/api/book/add"
		
		}

Applied Rules	:	#26: Using 201 (Created) for successful insertion. …
Possible Response	**:**	**Response 2**
HTTP Status Code	:	404 (Not Found)
Has payload data	:	Yes (JSON)

Data structure	:	{
		"status": false,
		"message": "Buku dengan ISBN ini telah tersedia"
		}
		
		"links": {
		
		"display_book": "/api/book",
		
		"add_book": "/api/book/add"
		
		}

Applied Rules	:	#37: 404 (NotFound) indicates that the URI is not valid or simply does not exist.
HATEOAS Links	**:**	**Add Books**
URI	:	POST/http://127.0.0.1:8000/api/books
Description	:	Untuk menambahkan data buku baru

5.2 Mengupdate data buku

General Information		
Resource URI	:	books/update-data
Business process	:	[FR01]Mengupdate data buku
Tujuan	:	Mengupdate data buku ke dalam collection
Responsible Member	:	Andri Anjelia Hutapea(@andrianjeliahutapea)
Applied Rules	:	#1: Forward slash “/” #2: No trailing forward slash “/”
Developer	:	Andri Anjelia Hutapea
Request		
HTTP Verb	:	PUT
Authenticated?	:	Yes, with token (*anonymous access is not allowed*)
User group	:	Admin
Has payload data	:	Yes (JSON)

Data structure	:	{
		 "id": 3,
		 "isbn": "1111124",
		 "title": "Hujan",
		 "type": "Novel",
		 "author": "Tere Liyee",
		 "publisher": "Gramedia",
		 "created_at": "2022-12-31 15:19:44",
		 "updated_at": "2022-12-31 15:19:44"
		 },
		 "links": {
		 "display_book": "/api/book",
		 "update_book": "/api/book/update"
		 }

Applied Rules	:	#21: PUT must be used to both create or modify a resource in a store.
Possible Response	**:**	**Response 1**
HTTP Status Code	:	201 (Created)
Has payload data	:	Yes (JSON)

Data structure	:	{
		"message": "Berhasil mengupdate data buku",
		"data": {
		"id": 3,
		"isbn": "1111124",
		"title": "Hujan",
		"type": "Novel",
		"author": "Tere Liyee",
		"publisher": "Gramedia",
		"created_at": "2022-12-31 15:19:44",
		"updated_at": "2022-12-31 15:19:44"
		},
		“links”:{
		"display_book": "/api/book",
		"update_book": "/api/book/update"
		}
		}

Applied Rules	:	#26: 201(Created) indicates that a resource is successfully created..
Possible Response	**:**	**Response 2**
HTTP Status Code	:	404 (Not Found)

Has payload data	:	Yes (JSON)

Data structure	:	{
		 "status": false,
		
		 "error_message": "Update data buku gagal",
		
		 "links": {
		
		 "display_book": "/api/book",
		
		 "update_book": "/api/book/update"
		
		 }
		
		 }

Applied Rules	:	#34: 400(BadRequest) indicates a general failure that is caused by the consumer.
HATEOAS Links	**:**	**Update data books**
URI	:	PUT/http://127.0.0.1:8000/api/books
Description	:	Untuk mengupdate data buku.

5.3 Menghapus data buku

General Information		
Resource URI	:	book/destroy-book
Business process	:	[FR01] Menghapus data buku
Tujuan	:	Menghapus satu data buku ke dalam *collection*
Responsible Member	:	Nadya Dioranta Tambunan (@Nadya06)

Applied Rules	:	#1: Forward slash “/” #2: No trailing forward slash “/” #4: Underscore “-” should be avoided #5: Use lowercase
Developer	:	Nadya Dioranta Tambunan
Request		
HTTP Verb	:	**DELETE**
Authenticated?	:	Yes, with token (*anonymous access is not allowed*)
User group	:	Admin (*otherwise forbidden*)
Has payload data	:	Yes (JSON)

Data structure	:	{
		  "data": {
		
		“isbn": "1111135",
		"title": "Si Kancil yang Kikir",
		"type": "Dongeng Anak",
		"author": "Auxtern",
		"publisher": "Erlangga",
		"status": "Available",
		"created_at": null
		
		},
		"links": {
		
		"display_data": "/api/book
		
		}
		
		}

Applied Rules	:	#23: DELETE should be used to remove a data or to make it unavailable.

Possible Response	**:**	**Response 1**
HTTP Status Code	:	200 (Ok)
Has payload data	:	Yes (JSON)

Data structure	:	{
		"message": "Delete data successfully",
		"links": {
		"display_data": "/api/book",
		"delete_data": "/api/book/delete"
		}
		}

Applied Rules	:	#25: 200(OK) indicates that the request is completely executed without problems.
Possible Response	**:**	**Response 2**
HTTP Status Code	:	400 (Bad request)

Data structure	:	{
		"error_message": "Delete data failed",
		"links": {
		"display_book": "/api/book",
		"develop_book": "/api/book/develop",
		"manage_book": "/api/book/manage"
		}
		}

Applied Rules	:	#37: 404(NotFound) indicates that the URI is not valid or simply does not exist.
HATEOAS Links	**:**	**Delete data book**
URI	:	DELETE/http://127.0.0.1:8000/api/books

Description	:	This link will direct the user to the delete data book.

5.4 Mengupdate peminjaman buku

General Information		
Resource URI	:	borrow/edit{id}
Business process	:	[FR02] Mengupdate data peminjaman buku
Tujuan	:	Mengubah status buku menjadi *not available* di *store*
Responsible Member	:	Siska Manullang (@SiskaManullang)
Applied Rules	:	#1: Forward slash “/” #2: Underscore “-” should be avoided #3: Use lowercase
Developer	:	Siska Manullang
Request		
HTTP Verb	:	PUT
Authenticated?	:	Yes, with token (*anonymous access is not allowed*)
User group	:	Admin (*otherwise forbidden*)
Has payload data	:	Yes (JSON)

Data structure	:	{
		  "borrow_id": “1”
		 },
		"links": {
		        "update status": "/borrow/ edit{id}"
		       }

Applied Rules	:	#21: PUT must be used to both create or modify a resource in a store
Possible Response	**:**	**Response 1**
HTTP Status Code	:	201 (Created)
Has payload data	:	Yes (JSON)

Data structure	:	{
		  "status": true,
		  "message": "status changed successfully!"
		  "data": {
		        "id": 1,
		        "isbn": "1111135",
		        "borrow date": "31-12-2022",

		"return date": "06-01-2023", "status": "Borrowed", "created_at": null }
Applied Rules	:	#26: Using 201 (Created)
Possible Response	**:**	**Response 2**
HTTP Status Code	:	404 (Not Found)
Has payload data	:	Yes (JSON)

Data structure	:	{
		"status": false,
		"message": "not exist because book is being borrowed"
		}

Applied Rules	:	#37: 404 (NotFound) indicates that the URI is not valid or simply does not exist.
HATEOAS Links	**:**	**Add Books**
URI	:	POST/http://127.0.0.1:8000/api/borrow
Description	:	Untuk mengubah status buku

5.5 Menghapus data peminjaman buku

General Information		
Resource URI	:	borrow/delete-data{id}
Business process	:	[FR02] Menghapus data peminjaman buku
Tujuan	:	Menghapus satu data peminjaman buku pada collection fitur
Responsible Member	:	Nemnem Sihombing (@nemnemsihombing35)
Applied Rules	:	#1: Forward slash “/” #2: No trailing forward slash “/” #4: Underscore “-” should be avoided #5: Use lowercase
Developer	:	Nemnem Sihombing
Request		
HTTP Verb	:	**DELETE**
Authenticated?	:	Yes, with token (*anonymous access is not allowed*)
User group	:	Admin (*otherwise forbidden*)
Has payload data	:	Yes (JSON)
Data structure	:	{ { “borrow_id” : “1” },

		"links": {
		"display data": "/api/data",
		"delete_data": "/api/data/delete"
		}
		}

Applied Rules	:	#23: DELETE should be used to remove a data or to make it unavailable
Possible Response	**:**	**Response 1**
HTTP Status Code	:	200(OK)
Has payload data	:	Yes (JSON)

Data structure	:	{
		"message": "Delete data success",
		"links": {
		"display_data": "/api/data",
		"delete_data": "/api/data/delete"
		}
		}

Applied Rules	:	#25: 200(OK) indicates that the request is completely executed without problems
Possible Response	**:**	**Response 2**
HTTP Status Code	:	400(Bad Request)
Has payload data	:	Yes (JSON)

Data structure	:	{
		"error_message": "Delete data failed",
		"links": {
		"display_return": "/api/return",
		"develop_return":"/api/return/develop",
		"manage_return":"/api/return/manage"
		}
		}

Applied Rules	:	#37: 404 (NotFound) indicates that the URI is not valid or simply does not exist.
HATEOAS Links	**:**	Delete Books return data
URI	:	POST/http://127.0.0.1:8000/api/return
Description	:	Untuk menghapus status peminjaman buku

5.6 Mengupdate data pengembalian buku

General Information		
Resource URI	:	borrow/edit{id}
Business process	:	[FR03] Mengupdate data peminjaman buku
Tujuan	:	Mengubah status buku menjadi *available*
Responsible Member	:	Nadya Dioranta Tambunan (@Nadya06)
Applied Rules	:	#1: Forward slash “/” #2: Underscore “-” should be avoided #3: Use lowercase
Developer	:	Nadya Dioranta Tambunan
Request		
HTTP Verb	:	PUT
Authenticated?	:	Yes, with token (*anonymous access is not allowed*)
User group	:	Admin (*otherwise forbidden*)
Has payload data	:	Yes (JSON)

Data structure	:	{
		"id": 1,
		"isbn": "1111111",
		"borrow date": "31-12-2022",
		"return date": "06-01-2023",
		"status": "Not Available" },
		"links": {
		"update status": "/return/ edit{id}"
		}

Applied Rules	:	#21: PUT must be used to both create or modify a resource in a store
Possible Response	**:**	**Response 1**
HTTP Status Code	:	201 (Created)
Has payload data	:	Yes (JSON)

Data structure	:	{
		"status": true,
		"message": "status changed successfully!"
		"data": {
		"id": 1,
		"isbn": "1111135",
		"borrow date": "31-12-2022",

		"return date": "06-01-2023",
		"status": "Returned",
		"created_at": null
		}
		"links": {
		"develop_return":"/api/return/develop",
		"manage_return":"/api/return/manage"
		}
		}

Applied Rules	:	#26: Using 201 (Created)
Possible Response	**:**	**Response 2**
HTTP Status Code	:	404 (Not Found)
Has payload data	:	Yes (JSON)

Data structure	:	{
		"status": false,
		"message": "Return a book failed",
		"links": {
		"develop_return":"/api/return/develop",
		"manage_return":"/api/return/manage"
		}
		}

Applied Rules	:	#37: 404 (NotFound) indicates that the URI is not valid or simply does not exist.
HATEOAS Links	**:**	**Add Books**
URI	:	POST/http://127.0.0.1:8000/api/return
Description	:	Untuk menghapus status pengembalian buku

6. Implementasi API dan potongan code

7. Test Cases

7.1 Menambahkan data buku

Test ID	NRF01-01-API

FR/NFR ID	NFR02 : Add Book
Test Name	Add book data according to the ISBN entered
Objective	To ensure that books can be added to the system
Description	This mechanism is part of adding book data to the book collection in the database
Preconditio n	The user has login
Date	31 Desember 2022
Tester	Nemnem Sihombing
Programme r	Lydia Tesalonika Harianja

Testing Scenario
1. Set the target URL (/users/authenticate). 2. Set the request method to POST. 3. Set a JSON-formatted data with the given configuration and send it.

Evaluation Criteria
1. All data in the database has been successfully added;and 2. The given credentials are matched with the stored data.

Test Cases				
ID	**Input**	**Expected Behavior**	**Actual Behavior**	**Verdict**

NFR01-01-API-01	Fill in the URI id http://localhost:8000/api/books and add book data: isbn: "1111135" title": "Si Kancil yang Kikir" tipe: "Dongeng Anak" author: "Auxtern" publisher": "Erlangga"	Response status code 201 (created) and in JSON format containing message: "Successfully added data"	Displays the response status code 201 (created) and in JSON format containing the message: "Successfully added data"	[x] passed [ ] failed
NFR01-01-API-02	Fill in book data with previously available data such as ISBN data isbn: "1111135" title": "Si Kancil yang Kikir" tipe: "Dongeng Anak" author: "Auxtern" publisher": "Erlangga"	Displays a response status 404 (not found) and in JSON format containing the message: "This book with ISBN is available"	Displays a response status 404 (not found) and in JSON format containing the message: "This book with ISBN is available"	[x]passed [ ] failed
NFR01-01-API-03	Adding book data with an incorrect format such as ISBN is not added	Will display a response status code 400 (Bad REQUEST) with JSON format and contains an incorrect data format message	Will display a response status code 400 (Bad REQUEST) with JSON format and contains an incorrect data format message	[x] passed [ ] failed

Notes
Successfully added data and Testing went smoothly.

7.2 Mengupdate data buku

Test ID	NFR01-02-API
FR/NFR ID	NFR02 – Update Book
Test Name	Update book data in the Book database
Objective	To ensure that previously entered book data can be updated
Description	This test is part of updating the book data that was previously added. This test is used to test and ascertain whether the user is able to update any book data in the system and ensure that the changed data is appropriate.
Precondition	There is book data in the database
Date	31 Desember 2022
Tester	Agnes Marpaung
Programmer	Siska Manullang
Testing Scenario	
1. Set the target URI (/books//Update-book) 2. Set the request method to PUT. 3. Update Book	

Evaluation Criteria

1. Updating all data in the book database has been successfully stored in the website database.

Test Cases

ID	Input	Expected Behavior	Actual Behavior	Verdict
NFR01-02-API-01	Correct Input : { "id": 3, "isbn": "1111124", "title": "Hujan", "type": "Novel", "author": "Tere Liyee", "publisher": "Gramedia", "created_at": "2022-12-31 15:19:44", "updated_at": "2022-12-31 15:19:44" },	Displays response status code 201 (Created) in JSON format containing the message: "Successfully changed data"	Displays the response status code 201 (Created) in JSON format and an output message in JSON containing: "Successfully Changed data"	[x] passed [ ] failed

NFR01-02-API-02	Incorrect Input : { "id": 3, "isbn": "1111124", "title": "Hujan", "type": "Novel", "author": "Tere Liyee", "publisher": "Gramedia", "created_at": "2022-12-31 15:19:44", "updated_at": "2022-12-31 15:19:44" },,	A status code 404 (Not Found) response will appear in JSON format and contains the message: "Book data not available"	Displays the response status code 404 (Not Found) in JSON format and contains the message: "Book data not available"	[x] passed [ ] failed
NFR01-02-API-03	Correct Format : { "isbn": "1111124", "title": "Hujan", "type": "Novel", "author": "Tere Liyee", "publisher": "Gramedia",	A response status code 400 (Bed Request) will be displayed in JSON format and contains the message: "Incorrect data format"	Displays the response status code 400 (Bed Request) in JSON format and contains the message: "Data format is incorrect"	[x] passed [ ] failed

	"created_at": "2022-12-31 15:19:44", "updated_at": "2022-12-31 15:19:44" },			

Notes
Successfully added data and Testing went smoothly.

7.3 Menghapus data buku

Test ID	NFR01-03-API
FR/NFR ID	NFR02 : Destroy Book
Test Name	Destroy Book
Objective	To Ensure that book can be deleted
Description	This test aims to ensure that the system can delete books that are already stored in the system.
Precondition	Book was already in the system
Date	31 Desember 2022

Tester	Winda Sari ButarButar			
Programmer	Niver Titi Sibuea			
Testing Scenario				
1. Set the target URI (/books//destroy-book) 2. Set the request method to Delete. 3. delete book				
Evaluation Criteria				
1. Book successfully deleted 2. Data is updated in the database				
Test Cases				
ID	**Input**	**Expected Behavior**	**Actual Behavior**	**Verdict**
NFR01-03-API-01	Add book data along with the ID of the book to be deleted	A 200-based response with a JSON-formatted payload contains the authenticated user.	returns status code 200 (Ok) in JSON format containing the message " Successfully deleted data"	[x] passed [ ] failed
NFR01-03-API-02	Delete a previously deleted book	A 404-based response with a JSON-formatted payload contains “Book data is not available” message.	A 404-based response with a JSON-formatted payload contains “Book data is not available” message.	[x] passed [ ] failed

NFR01-03-API-03	Delete data with improper formatting	Displaying a 400 Bad Request Response Code Containing the Message "Data Format Is Incorrect"	Displaying a 400 Bad Request Response Code Containing the Message "Data Format Is Incorrect"	[x] passed [ ] failed
Notes				
Successfully added data and Testing went smoothly.				

7.4 Mengupdate peminjaman buku

Test ID	NFR02-01-API
FR/NFR ID	NFR02 – Update Book borrowing
Test Name	update the borrowing status of books in the database
Objective	To ensure that the book borrowing has been successful and the book data has been updated
Description	This test is part of the borrowing of books that have been carried out, This test is useful to ascertain whether the data of the borrowed book has been updated its status in the system to the status of 'borrowed' .
Precondition	There is book data in the database
Date	31 Desember 2022
Tester	Lydia Tesalonika
Programmer	Siska Manullang
Testing Scenario	

1. Set the target URI (/book borrowing//update-book borrowing) 2. Set the request method to PUT. 3. Update Book				
Evaluation Criteria				
1. The status of the borrowing book update has been successfully saved in the database.				
Test Cases				
ID	**Input**	**Expected Behavior**	**Actual Behavior**	**Verdict**
NFR02-01-API-01	Inputting data on the book to be borrowed Correct Input : “borrow_id” : “1” },	Displays response status code 201 (Created) in JSON format containing the message: "Successfully borrowing book"	Displays the response status code 201 (Created) in JSON format and an output message in JSON containing: "Successfully borrowing book"	[X] passed [ ] failed

NFR02-01-API-02	Inputting data on the book to be borrowed Correct Input : “borrow_id” : “1” },	A status code 404 (Not Found) response will appear in JSON format and contains the message: "Books have been borrowed "	Displays the response status code 404 (Not Found) in JSON format and contains the message: "Books have been borrowed "	[X] passed [ ] failed
NFR02-01-API-03	Inputting data on the book to be borrowed Incorrect Input : “id” : “1” },	A status code 400(Bad Request) response will appear in JSON format and contains the message: "Error message/ Incorrect data format"	Displays the response status code 400 (Bad Request) in JSON format and contains the message: "Error message/Incorrect data format"	[X] passed [ ] failed
Notes				
Successfully borrowing book and Testing went smoothly.				

7.5 Menghapus data peminjaman buku

Test ID	NFR02-02-API
FR/NFR ID	NFR03 : destroy book borrow data
Test Name	Destroy book borrow data
Objective	To Ensure that book borrow data can be deleted
Description	This test aims to ensure that the system can delete book borrow data that are already stored in the system.
Precondition	Book borrow data was already in the system
Date	31 Desember 2022
Tester	Niver Titi Sibuea
Programmer	Siska Manullang
Testing Scenario	
1. Set the target URI (borrow/delete-data{id}) 2. Set the request method to Delete. 3. delete book borrow data	
Evaluation Criteria	
1. Book borrow data successfully deleted 2. Data is updated in the database	
Test Cases	

ID	Input	Expected Behavior	Actual Behavior	Verdict
NFR02-02-API-01	Add book borrow data along with the ID of the book to be deleted	A 200-based response with a JSON-formatted payload contains the authenticated user.	Borrows status code 200 (Ok) in JSON format containing the message " Successfully deleted data"	[x] passed [ ] failed
NFR02-02-API-02	Delete a previously deleted data	A 404-based response with a JSON-formatted payload contains “Book data is not available” message.	A 404-based response with a JSON-formatted payload contains “Book data is not available” message.	[x] passed [ ] failed
NFR02-02-API-03	Delete data with improper formatting	Displaying a 400 Bad Request Response Code Containing the Message "Data Format Is Incorrect"	Displaying a 400 Bad Request Response Code Containing the Message "Data Format Is Incorrect"	[x] passed [ ] failed
Notes				
Successfully added data and Testing went smoothly.				

7.6 Mengupdate data pengembalian buku

Test ID	NFR03-01-API
FR/NFR ID	NFR03 – Update Book
Test Name	Updating the book return data in the database
Objective	To ensure that previously entered book data can be updated

Description	This test is used to test and ascertain whether the user is able to update any book data in the system and ensure that the changed data is appropriate.			
Precondition	There is book data in the database			
Date	31 Desember 2022			
Tester	Gabriel Panggabean			
Programmer	Siska Manullang			
Testing Scenario				
1. Set the target URI (/return//return) 2. Set the request method to PUT. 3. Update Book				
Evaluation Criteria				
1. Update the success book return data 2. Data is update in database				
Test Cases				
ID	**Input**	**Expected Behavior**	**Actual Behavior**	**Verdict**

NFR03-01-API-01	Correct Input : { "book_id": 1, "isbn": "1111135", "title": "Si Kancil yang Kikir", "tipe": "Dongeng Anak", "author": "Auxtern", "publisher": "Erlangga", },	Displays response status code 201 (Created) in JSON format containing the message: "Successfully changed data"	Displays the response status code 201 (Created) in JSON format and an output message in JSON containing: "Successfully Changed data"	[x] passed [ ] failed
NFR03-01-API-02	Incorrect Input : { "book_id": 1, "isbn": "1111135", "title": "Si Kancil yang Kikir", "tipe": "Dongeng Anak", "author": "Auxtern", "publisher": "Erlangga", },	A status code 404 (Not Found) response will appear in JSON format and contains the message: "Book data not available"	Displays the response status code 404 (Not Found) in JSON format and contains the message: "Book data not available"	[x] passed [ ] failed

NFR03-01-API-03	Correct Format : { "isbn": "1111135", "title": "Si Kancil yang Kikir", "tipe": "Dongeng Anak", "author": "Auxtern", "publisher": "Erlangga", },	A response status code 400 (Bed Request) will be displayed in JSON format and contains the message: "Incorrect data format"	Displays the response status code 400 (Bed Request) in JSON format and contains the message: "Data format is incorrect"	[x] passed [ ] failed
Notes				
Successfully added data and testing went smoothly				

Refleksi KEL-4:

Dari awal pengerjaan proyek, kelompok kami sudah berusaha dengan semaksimal mungkin dengan harapan hasil yang terbaik.
Kami sudah memberikan pendapat, ide, kreativitas dan logika yang kami miliki pada proyek ini.
Ketika pengerjaan proyek kami memiliki beberapa kesulitan dalam pengerjaan proyek, mungkin adanya miskomunikasi antar anggota tim yang mengakibatkan kami kesulitan dalam menyatukan pendapat kami, dan menemukan solusi atas permasalahan kami.
Ada kalanya kami kesulitan dalam memahami setiap tugas yang diberikan, dan kesulitan dalam pemahaman, pengimplementasian dan pengerjaan proyek.
Kurangnya dalam pendefinisian dan analisis terhadap masalah yang dihadapi, sehingga agak sulit dalam menentukan solusi terhadap masalah yang kami alami dan tidak mengatur waktu mengerjakan proyek dengan baik sehingga terkadang menjadi terburu-buru dalam melakukan submitan.

Saran:
Semoga tahun depan lebih baik lagi dan pembagian kelompoknya lebih baik lagi serta tidak berdasarkan ipk agar yang mahasiswa yang kurang mampu dalam mata kuliah ini dapat diajari oleh temannya yang lebih mampu.

Pembagian Tugas :
Application Developer : Lidya, Gabriel, Siska, Putri
Rest API : Nadya, Andri, Niver
Deployment : Winda, Nemnem, Agnes