Hello World
Jurnal Ilmu Komputer
https://jurnal.ilmubersama.com/index.php/hello_world

Sistem Informasi

Penerapan *API* Berbasis *REST* Guna Mengefisiensi Pendistribusian Data dan Aksebilitas Data di BPS Provinsi Jawa Barat

Karenina Casandra Rahmat [*], *Sen Yung*

Program Studi Sistem Informasi, STMIK AMIK Bandung, Kota Bandung, Indonesia

INFORMASI ARTIKEL

Diterima Redaksi: 10 September 2024
Revisi Akhir: 25 Maret 2025
Diterbitkan *Online*: 25 Maret 2025

KATA KUNCI

Distribusi Data
Representational State Transfer (*REST*)
Framework CodeIgniter

KORESPONDENSI [(*)]

Phone: +62 831-0070-1027
E-mail: kareninacasandrar76@gmail.com

A B S T R A K

Penelitian ini bertujuan untuk mengembangkan aplikasi manajemen distribusi data di Badan Pusat Statistik (BPS) Provinsi Jawa Barat yang terintegrasi langsung dengan basis data internal, mengatasi keterbatasan metode *file sharing* berbasis *cloud storage*. Keterbatasan yang dihadapi meliputi ketergantungan pada layanan pihak ketiga, kapasitas penyimpanan terbatas, dan data yang tidak dapat diperbarui secara otomatis. Aplikasi ini dibangun dengan menggunakan arsitektur *Representational State Transfer* (*REST*) dan *framework CodeIgniter*, serta menerapkan mekanisme *0* untuk memastikan keamanan akses data. Aplikasi yang diusulkan memungkinkan distribusi data secara real-time, memperbarui data secara otomatis, dan mempercepat proses distribusi. Pengujian menggunakan metode *User Acceptance Testing* (*UAT*) menunjukkan bahwa aplikasi usulan memberikan peningkatan signifikan dalam kemudahan penggunaan dan efisiensi dengan skor 90%, dibandingkan dengan *cloud storage* yang memperoleh skor 83,3% dan 73,3%. Tingkat kepuasan pengguna juga lebih tinggi, yaitu 83,3% dibandingkan dengan 76,6% pada *cloud storage*. Secara keseluruhan, aplikasi ini berhasil meningkatkan efisiensi, keamanan, dan kemudahan distribusi data di BPS Provinsi Jawa Barat serta memiliki potensi untuk dikembangkan lebih lanjut guna memenuhi kebutuhan distribusi data yang terus meningkat.

PENDAHULUAN

Badan Pusat Statistik (BPS) Provinsi Jawa Barat saat ini menggunakan metode *file sharing* melalui layanan *cloud storage* seperti *OneDrive* untuk pendistribusian data. Meski pun metode ini memiliki beberapa manfaat, namun terdapat beberapa keterbatasan yang perlu diperhatikan. Salah satunya adalah ketergantungan pada penyedia layanan pihak ketiga, yang menimbulkan masalah terkait privasi, keamanan, serta skalabilitas penyimpanan [1] meski pun menawarkan banyak keuntungan, disamping itu penggunaan *cloud storage* memiliki ketergantungan pada pihak ketiga seperti biaya tambahan untuk menambah kapasitas penyimpanan. Hal ini sangat relevan dalam konteks BPS, di mana ketergantungan pada *cloud storage* dapat menimbulkan tantangan jika layanan terganggu atau kapasitas penyimpanan tidak lagi memadai.

Selain tidak terintegrasi dengan sumber data internal BPS. Data yang tersimpan di *cloud storage* tidak secara otomatis diperbarui jika ada perubahan pada basis data. Hal ini membuat data yang didistribusikan bisa menjadi tidak akurat, sehingga dapat menimbulkan risiko dalam penyediaan data yang valid dan terkini. Proses distribusi data juga membutuhkan waktu terutama untuk *file* berukuran besar karena data harus diunggah terlebih dahulu.

Untuk mengatasi permasalahan ini, penelitian ini mengusulkan solusi berbasis web yang menggunakan *Application Programming Interface* (*API*) dengan arsitektur R*epresentational State Transfer* (*REST*). Penggunaan *REST* memungkinkan interaksi antara sistem terdistribusi dengan memanfaatkan prinsip-prinsip seperti komunikasi stateless, respons yang disimpan dalam bentuk *cache*, dan penggunaan *URL* sebagai pengenal sumber daya [2]. Dengan arsitektur

https://doi.org/10.56211/helloworld.v3i4.636

ini, data dapat diakses secara real-time, aplikasi juga terintegrasi langsung dengan sumber data, dan data dapat selalu diperbarui secara otomatis tanpa perlu sinkronisasi manual.

Dengan menggunakan *API* berbasis *REST*, diharapkan solusi ini akan mempercepat proses distribusi, mengurangi ketergantungan pada penyedia layanan *cloud storage*, meningkatkan keamanan, dan memberikan fleksibilitas bagi TIM IPDS dalam pengelolaan data. Hal ini juga memungkinkan aplikasi dikembangkan lebih lanjut sesuai dengan kebutuhan masa depan BPS Provinsi Jawa Barat.

TINJAUAN PUSTAKA

Application Programming Interface (API)

API merupakan sebuah antarmuka yang digunakan untuk dapat mengakses aplikasi atau layanan dari sebuah program dengan memanggil fungsi melalui *Hyper Text Transfer Protokol* (*HTTP*) dan mendapatkan respon berupa *JavaScript Object Notation* (*JSON*) atau *Extensible Markup Language* (*XML*) sehingga memungkinkan pengembang untuk menggunakan fungsi yang sudah tersedia dari aplikasi lain [3]. *API* adalah perantara atau jembatan yang memungkinkan suatu aplikasi atau perangkat lunak untuk dapat berkomunikasi dan berinteraksi dengan perangkat lainnya yang berbeda. Dengan berperan sebagai penghubung API juga dapat memberikan akses fungsional sehingga pengguna dapat mengakses fitur atau layanan yang tersedia pada suatu aplikasi atau sistem lain. *API* sendiri terbagi menjadi beberapa jenis yang dibedakan berdasarkan fungsi atau protokol yang digunakan serta aspek lainnya. Melansir dari halaman resmi *International Business Machine* jenis-jenis *API* utama antara lain adalah *API Private, API Public, API Partner* dan *API Composite* [4].

Protokol API

Untuk mengatur *API* yang akan dikembangkan maka pengembang harus mengikuti aturan dan teknik-teknik yang sudah ditetapkan dan aturan yang telah ditetapkan disebut sebagai protokol. Halaman resmi *IBM* juga menjelaskan bahwa protokol atau aturan yang ditetapkan ini berperan untuk menentukan perintah, tipe data hingga format serta hal lainnya. Protokol itu antara lain adalah (1) *Representational State Transfer (REST)* menggunakan metode *Hyper Text Transfer Protocol (HTTP)* untuk dapat terhubung dan berkomunikasi dengan *server REST* juga dapat menggunakan *(XML)* atau *JavaScript Object Notation (JSON)* untuk mengirimkan data. (2) *Simple Object Access Protocol (SOAP)* protokol yang lebih kompleks dari *REST* ini dapat melakukan pertukaran data dengan format *XML* bahkan dilingkungan atau bahasa yang berbeda. (3) *Remote Procedure Call (RPC)* protokol sederhana yang memungkinkan pemanggilan prosedur pada *server* dari jarak jauh dengan menggunakan berbagai bahasa pemrograman.

Representational State Transfer (REST)

REST adalah arsitektur *client-server* artinya klien dapat mengirimkan permintaan ke *server* kemudian *server* akan memberikan respon menggunakan protokol komunikasi data HTTP [5]. Pendefinisian lainnya mengartikan *REST* sebagai prinsip arsitektur yang dapat melakukan transmisi data melalui antarmuka terstandarisasi seperti *HTTP*. *Server* akan menyediakan *resource* yakni sumber daya/data yang akan diidentifikasi oleh *Universal Resource Identifiers* (*URI*) kemudian direpresentasikan dalam bentuk *JSON* atau *XML* sehingga dapat diakses oleh klien [3]. Arsitektur *REST* menggunakan metode *HTTP* seperti *GET, POST, PUT dan DELETE* untuk dapat melakukan operasi *CRUD* pada data dan menganggap data atau objek yang diakses melalui *API* sebagai *resource* kemudian merepresentasikan dalam bentuk *JSON* atau *XML.*

Protokol Hyper Text Transfer Protocol (HTTP)

Sebagai alat untuk dapat melakukan pertukaran data *REST* menggunakan protokol *HTTP* [udaya]. *HTTP* merupakan sebuah protokol standar yang digunakan untuk berkomunikasi dan mengirimkan data di internet atau *World Wide Web (WWW)* serta memiliki standar kode respon yang digunakan ketika klien mengirimkan permintaan. Kode respon ini terbagi menjadi lima kelompok utama seperti *Informational Response* (kode 1XX), *Success Response* (kode 2XX), *Redirection Message* (kode 3XX), *Client Error Message* (kode 4XX) dan *server Error Response* (5XX) [6].

Framework CodeIgniter (CI)

Framework adalah sebuah struktur konseptual dasar yang dapat digunakan untuk memecahkan sebuah permasalahan atau suatu isu yang kompleks [7], *framework* juga bersifat multi-fungsi sehingga dapat dimanfaatkan untuk membantu mempercepat proses perencanaan, pembuatan, pengujian dan pemeliharaan [8]. *CodeIgniter* sendiri merupakan sebuah *framework* yang menyediakan solusi untuk menangani masalah kompleks dan menawarkan beragam keunggulan seperti konfigurasi yang minim, dukungan pengguna dengan dokumentasi yang jelas serta keamanan bawaan yang kuat [9]. *CodeIgniter* memiliki ukuran yang kecil sehingga tidak membutuhkan *resource* yang besar terutama dalam hal pengeksekusian dan penyimpanan dan menerapkan konsep MVC dengan begitu pengembang dapat memisahkan layer application-logic dengan presentation dan menjadikan ukuran file lebih kecil untuk memudahkan pemeliharaan [yukum]. Berdasarkan penjelasan tersebut *CodeIgniter* adalah framework *open-source* yang ringan dengan ukuran kecil sehingga menjadikannya populer dan handal karena mudah dipelajari dan digunakan didukung dengan dokumentasi yang jelas, minimnya konfigurasi dilengkapi keamanan bawaan. *CodeIgniter* juga menerapkan konsep *Model View Controller* sehingga memungkinkan pengguna untuk memisahkan data, tampilan dan logika program sehingga menjadikan kode program lebih terstruktur.

Konsep *MVC* sendiri memungkinkan kode *PHP*, kueri *MySQL*, *JavaScript* dan *Cascading Style Sheets* (*CSS*) dapat dipisahkan karena konsep ini terdiri dari *Model*, *View* dan *Controller* dimana *model* berhubungan dengan basis data untuk melakukan manipulasi data. *View* sendiri merupakan kode program berupa *template* yang menampilkan data pada peramban sedangkan *controller* merupakan kode program yang digunakan untuk mengontrol aliran model dan *view* [10].

Hypertext Pre-Processor (PHP)

PHP memungkinkan pengembang untuk dapat membuat web yang bersifat dinamis maksudnya web tersebut mampu berinteraksi dengan pengguna contohnya ketika pengguna mengunjungi *web e-commerce* dan interaksi yang dimaksud yakni keranjang belanja, formulir alamat pengiriman dan masih banyak lagi. *PHP* cukup populer karena mudah dipelajari dan dipahami terutama bagi pemula, selain itu dapat digunakan secara gratis dan dijalankan diberbagai sistem operasi seperti *Windows*, *Linux* mau pun *Mac OS*. *PHP* juga merupakan bahasa pemrograman yang berbasis *website* dan memiliki sifat *server-side* yakni *PHP* akan memproses seluruh baris kode program didalam *web server* [11].

MySQL

MySQL adalah sebuah *Database Management System (DBMS)* yang *multi-thread* dan *multi-user* yang dapat menyimpan data dalam tabel terpisah dan menempatkan seluruh data dalam satu ruang besar [12]. *MySQL* digemari karena dapat diakses secara gratis dan juga menawarkan keunggulan seperti (1) Mampu terhubung dengan jaringan internet dan dapat dijangkau dari jauh. (2) Menyediakan kapasitas yang besar hingga *Gigabyte* sekali pun. (3) Ringan dan tidak membebani kinerja *server* dari computer [13].

MySQL adalah turunan dari konsep *Structured Query Language* sehingga untuk dapat menjalankan *database relational* maka dibutuhkan kemampuan untuk mengenal dan memahami penggunaan *SQL*. *SQL* sendiri adalah perintah yang digunakan untuk mengolah, menampilkan dan memanipulasi data yang tersimpan dalam *RDBMS* [14]. SQL terdiri dari tiga jenis perintah utama yaitu *Data Definition Language* (*DDL*) yakni perintah yang digunakan untuk membuat atau mengubah struktur *database*. Lalu *Data Manipulation Language* (*DML*) adalah perintah yang digunakan untuk memanipulasi data seperti menambah, menghapus dan mengubah data didalam tabel sedangkan *Data Control Language* (*DCL*) adalah perintah yang digunakan untuk mengatur hak akses pengguna dalam menggunakan basis data [14].

Cloud Storage

Cloud storage sebagai layanan penyimpanan *file* yang berbasiskan jaringan internet. *File* yang disimpan tersebut dapat diakses dan dikelola dari berbagai tempat selama pengguna terhubung dengan *cloud storage* melalui jaringan internet [15]. Layanan *cloud storage* sendiri terbagi menjadi beberapa jenis seperti (1) *Personal cloud storage*, jenis *mobile cloud storage* yang dapat digunakan untuk menyimpan data milik pribadi dan dapat diakses dari mana saja. (2) *Private cloud storage* jenis yang biasa digunakan untuk penggunaan organisasi karena menyediakan tingkat keamanan dan kendali yang lebih baik. (3) *Public cloud storage*, jenis yang memungkinkan pengguna individu atau organisasi menyimpan, mengubah dan mengelola data secara bersama-sama karena data akan disimpan pada suatu *server* atau repositori. (4) *Hybrid cloud*

storage, jenis penyimpanan gabungan antara *private* dan *public cloud storage* sehingga memungkinkan data penting tertentu disimpan secara *private* dan data lainnya disimpan secara *public* [15].

Database

Database adalah kumpulan data yang saling terkait dan disimpan bersama dengan redundansi terkontrol untuk dapat melayani satu atau lebih aplikasi secara optimal [16]. *Database* atau basis data terdiri dari beberapa komponen seperti (1) Item Data yakni bagian atau potongan informasi yang berbeda namun dapat dijelaskan pada bagian sebelumnya. (2) Relasi yakni mewakili korespondensi antara berbagai elemen data. (3) Pembatasan yakni predikat penentu status basis data yang sesungguhnya. (4) Skema yakni gambaran bagaimana suatu sistem atau proses dapat bekerja dengan tujuan memberikan panduan visual bagaimana komponen yang ada saling berelasi dan berinteraksi [16].

Unified Modelling Language (UML)

UML adalah bahasa yang digunakan untuk memodelkan suatu sistem perangkat lunak. Dengan *UML* pengguna dapat memvisualisasikan sistem yang kompleks ke dalam bentuk diagram yang lebih mudah dibaca dan dipahami. *UML* juga dibagi menjadi beberapa kategori seperti *Structured Diagram* yang fokus menggambarkan tentang apa yang ada didalam sistem dan *Behavioral Diagram* fokus menggambarkan apa yang dilakukan oleh sistem.

Entity Relationship Diagram (ERD)

ERD digunakan untuk merancang basis data dan menggambarkan relasi antar setiap objek beserta dengan atribut yang dimiliki [17]. *ERD* dapat membantu memvisualisasikan bagaimana satu data berhubungan dengan data lainnya selain itu dengan ERD juga dapat diketahui bagaimana suatu data yang diwakili oleh entitas memiliki karakteristik yang digambarkan dengan atribut serta bagaimana hubungannya dengan data (entitas) lainnya yang ditunjukkan melalui relationship.

Black Box Testing

Pengujian *Black Box* adalah metode uji fungsionalitas sistem sehingga dapat menunjukkan kesalahan pada aplikasi [18], metode pengujian yang cukup populer karena fokus pada fungsionalitas dan yang utama adalah pengujian ini tidak memerlukan pemahaman yang mendalam struktur internal atau memperhatikan dan mengerti kode pemrograman dari perangkat lunak yang diuji. Pengujian black box memiliki banyak metode seperti *All Pair Testing, Boundary Value Analysis, Cause-effect Graph, Equivalence Partitioning, Fuzzing, Orthogonal Array Testing, Orthogonal Array Testing* [18].

User Acceptance Testing (UAT)

UAT adalah proses verifikasi yang dilakukan untuk memastikan sistem bekerja dengan baik dan sudah sesuai bagi pengguna. Pengujian umumnya dilakukan oleh pengguna akhir lalu hasil idenfikasi akan dievaluasi kemudian diperbaiki [19] hasil dari pengujian *UAT* ini akan memberikan informasi dibutuhkan untuk dapat mengevaluasi kualitas layanan [20].

METODOLOGI

Penelitian ini dilakukan dengan pendekatan kualitatif karena penelitian ini berangkat dari pengalaman dan perspektif orang terhadap suatu fenomena yang terjadi dilapangan dan penelitian ini bertujuan untuk mempelajari secara mendalam mengenai bagaimana suatu fenomena dapat terjadi.

Pengumpulan data ini dilakukan untuk memperoleh data yang relevan dan objektif sebagai dasar untuk mencapai tujuan penelitian. Pada penelitian ini pengumpulan data dilakukan dengan metode:

1. Observasi
   Kegiatan pengamatan dilakukan secara langsung selama Penulis melaksanakan Program *Enrichment* II di BPS Provinsi Jawa Barat TIM IPDS.

 https://doi.org/10.56211/helloworld.v3i4.636

2. Wawancara
   Wawancara yang dilakukan melibatkan interaksi secara langsung antara Penulis sebagai pewawancara dengan anggota TIM IPDS yang dengan jabatan Pranata Komputer Ahli Pertama sebagai responden. Wawancara ini dilakukan untuk mengetahui kendala atau keterbatasan yang dialami langsung oleh TIM IPDS khususnya dalam proses pendistribusian data secara internal.

Metode pengembangan sistem yang digunakan pada penelitian ini adalah Scrum. Dengan metode ini maka pembangunan aplikasi dapat dilakukan melalui tiga tahapan utama yaitu (1) *Product Backlog* yakni mengidentifikasi kebutuhan lalu memasukkan kebutuhan tersebut menjadi *user story* untuk kemudian dilakukan prioritisasi. (2) *Sprint Planning* yakni menentukan tugas yang akan dikerjakan dengan memilih fitur sesuai tingkat prioritas. (3) *Sprint* yakni memulai implementasi fitur sesuai tugas hingga selesai dan melakukan pengujian untuk memastikan semua fitur sudah berfungsi dengan baik hingga menunjukkan hasil yang telah selesai dikerjakan dan mengumpulkan *feedback.*

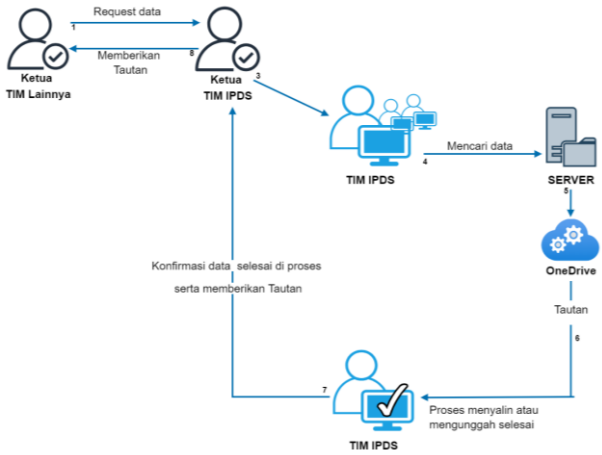
HASIL DAN PEMBAHASAN

Analisis Sistem Berjalan

Kelemahan atau kekurangan dari suatu sistem dapat merujuk pada kegagalan, dalam mencapai tujuan. Dan berikut beberapa kekurangan dari sistem pendistribusian data yang berjalan saat ini melalui penggunaan layanan *cloud storage:*

1. Proses distribusi data membutuhkan waktu untuk mengunggah data terlebih dahulu ke *cloud storage.*
2. Data di *cloud storage* tidak otomatis mengalami pembaruan karena tidak terintegrasi dengan sumber data.
3. Adanya pihak ketiga yakni penyedia *cloud storage.*
4. Keterbatasan kapasitas penyimpanan dan membutuhkan biaya tambahan untuk menambah kapasitas penyimpanan.
5. Proses pendistribusian data dibagikan melalui tautan yang berisiko disalahgunakan oleh pihak yang tidak berwenang jika pemilik tautan tidak membatasi akses dari tautan tersebut.
6. Proses pendistribusian data dapat terhambat jika layanan *cloud storage* mengalami pemeliharaan atau pun gangguan.

Untuk dapat menggunakan data yang dibutuhkan maka ketua tim internal lainnya terlebih dahulu mengajukan kebutuhan data kepada Ketua TIM IPDS. Kemudian Ketua TIM IPDS akan memberikan rincian data yang dibutuhkan kepada anggota TIM IPDS untuk dipersiapkan dan dibagikan.



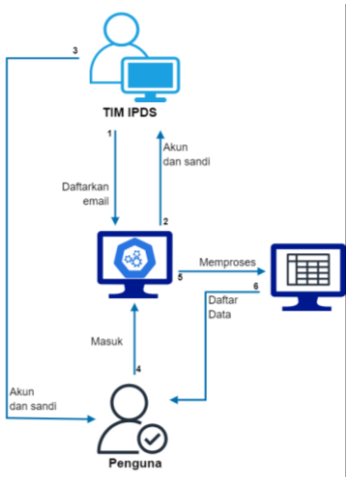
Gambar 1. Alur Proses Pendistribusian Data Yang Berjalan

Anggota TIM IPDS akan mempersiapkan data untuk membagikannya melalui *OneDrive* setelah berhasil mengunggah data maka tautan akan dibagikan kepada Ketua TIM IPDS dan terakhir Ketua TIM IPDS memberikannya kepada ketua tim internal yang mengajukan.

Analisis Sistem Usulan

Penelitian ini mengusulkan pengimplementasian aplikasi pendistribusian data berbasis web dengan arsitektur *REST*. Aplikasi yang diusulkan ini dirancang untuk dapat mengatasi beberapa kelemahan pada penggunaan *cloud storage* seperti keterbatasan kapasitas penyimpanan dan proses unggah data yang memakan waktu. Dengan aplikasi berbasis web yang diusulkan tenaga dan waktu yang dibutuhkan untuk mengunggah data dapat diminimalkan, selain itu pertumbuhan ukuran data yang terus meningkat dapat ditangani dengan baik melalui skalabilitas sistem yang fleksibel dan optimal. Aplikasi ini memungkinkan kendali akses penuh terhadap seluruh data yang dimiliki atau pun kendali akses pengguna terhadap aplikasi dengan menerapkan mekanisme *role-based access control* sehingga risiko adanya akses yang tidak sah dapat diminimalkan.

TIM IPDS akan bertanggung jawab sebagai administrator dan membuatkan akun bagi pengguna agar dapat mengakses aplikasi. Setelah memiliki akun maka pengguna dapat mengakses aplikasi untuk melihat kumpulan data yang tersedia dan mengunduhnya jika dibutuhkan.

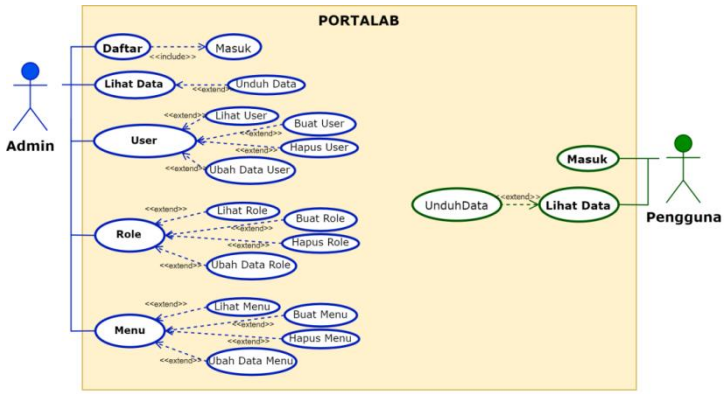


Gambar 2. Alur Sistem Usulan

Implementasi arsitektur *REST* juga memberikan fleksibilitas dalam integrasi dan pengelolaan data. Pengguna dapat dengan mudah mengakses data secara efisien termasuk mengunduh data dalam berbagai format, seperti *.xlsx* dan *.sql*.

Analisis Kebutuhan Sistem

Kebutuhan fungsional dipisahkan berdasarkan aktor yang akan berinteraksi dengan fitur-fitur seperti yang digambarkan pada *use case diagram* berikut.



Gambar 3. Fungsionalitas Aplikasi Usulan

Detail kebutuhan fungsional pada diagram tersebut dapat dilihat pada tabel dibawah ini yang dibuat secara terpisah untuk kedua aktor.

Tabel 1. Definisi *Use Case* - Admin

UCA-No.	Nama Use Case	Deskripsi
UCA-01.	Masuk	Aktor masuk dengan *email* dan kata sandi yang telah dibuat sebelumnya.
UCA-02.	Lihat Data	Aktor dapat melihat daftar data yang tersedia dan melihat detail data.
UCA-03.	Unduh Data	Aktor dapat mengunduh data yang dibutuhkan.
UCA-04.	Buat Akun	Aktor mendaftarkan email dan data Ketua TIM Internal sebagai pengguna.
UCA-05.	Hapus	Aktor dapat menghapus akun pengguna.
UCA-06.	Ubah Akun	Aktor dapat mengubah data akun pengguna.
UCA-07.	Buat *Role*	Aktor dapat membuat *role* untuk mengatur hak akses pengguna.
UCA-08	Hapus *Role*	Aktor dapat menghapus *role* yang telah dibuat.
UCA-09	Ubah *Role*	Aktor dapat mengubah data *role* yang telah dibuat
UCA-10	Buat Menu	Aktor dapat menambahkan menu baru yang telah dibuat melalui pengkodean.
UCA-11	Tambah Menu	Aktor dapat mengubah data atau pengaturan pada menu yang telah dibuat.
UCA-12	Hapus Menu	Aktor dapat menghapus menu yang telah dibuat.

Tabel 2. Definisi *Use Case* - Pengguna

UCP-No.	Nama Use Case	Deskripsi
UCP-01.	Masuk	Aktor dapat masuk dan mengakses sistem dengan akun yang diberikan admin.
UCP-02.	Lihat Data	Aktor dapat melihat daftar data yang tersedia beserta detail data.
UCP-03.	Unduh Data	Aktor dapat mengunduh data yang dibutuhkan.

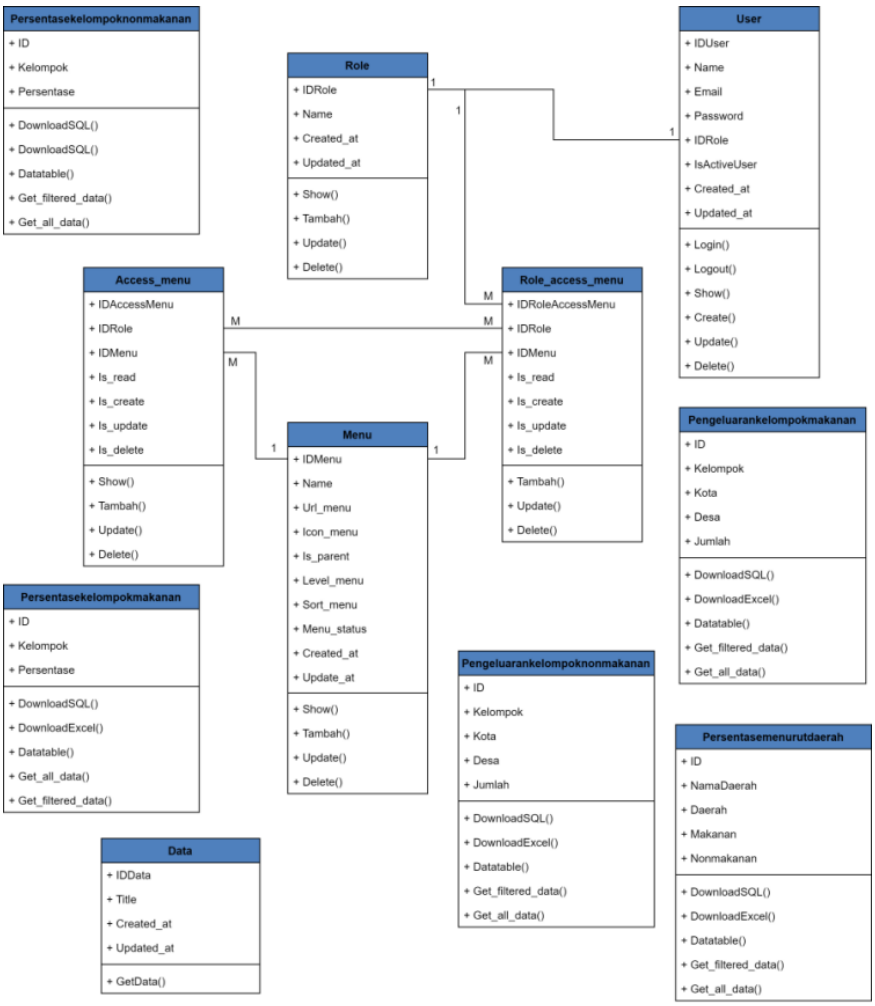
Kebutuhan non-fungsional tidak berhubungan secara langsung dengan fitur dari suatu sistem aplikasi dan lebih merujuk pada karakteristik seperti kualitas, kinerja, keamanan dan sebagainya. Berikut kebutuhan non-fungsionalitas pada aplikasi yang dibangun.

Tabel 3. Kebutuhan non-fungsional.

No	Atribut	Kebutuhan
KFN-01	Kinerja	Sistem memiliki kinerja yang cepat dan efisien.
KFN-02	Ketersediaan	Sistem dapat diakses setiap waktu dan dimana saja.
KFN-03	Kegunaan	Sistem mudah dipahami dan digunakan oleh pengguna.
KFN-04	Antar Muka	Tampilan simpel namun menarik, *user-friendly* dan data yang *readable.*
KFN-05	Keamanan	Sistem harus melindungi data dan informasi dari pengguna akses yang tidak sah.

Rancangan *Class Diagram*

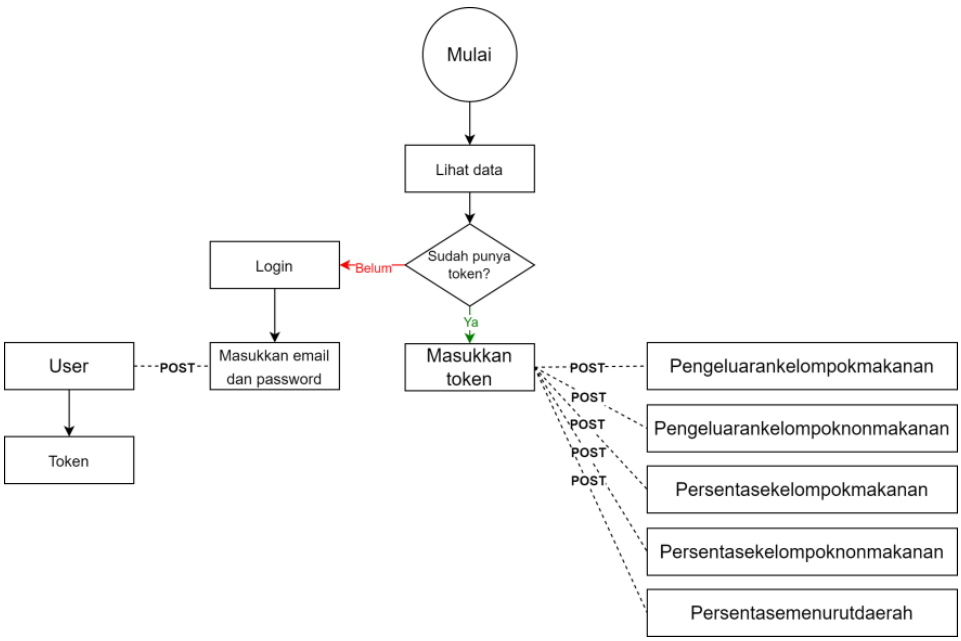
Berikut gambaran atribut dan metode yang dimiliki oleh setiap kelas yang ada serta relasi antar kelas. Diagram ini menunjukkan relasi yang erat antara tabel-tabel utama seperti *User*, *Role*, Menu, dan Data.



Gambar 4. *Class Diagram*

Rancangan Entity Relationship Diagram

ERD ini fokus menggambarkan pada proses pengguna ketika ingin mengakses *endpoint* tertentu. Untuk dapat mengakses *endpoint* tersebut pengguna harus memverifikasi identitasnya dengan akun yang telah didaftarkan oleh admin. Jika sudah diverifikasi maka pengguna akan mendapatkan token sebagai kunci akses untuk dapat mengakses *endpoint* yang dituju.



Gambar 5. *ERD API* Pada Aplikasi

 https://doi.org/10.56211/helloworld.v3i4.636

IMPLEMENTASI

Perangkat Keras

Perangkat keras yang digunakan untuk membangun aplikasi pada penelitian ini menggunakan laptop *ASUS* model X455LD dengan rincian sebagai berikut

Tabel 4. Spesifikasi Perangkat Keras

No	Perangkat Keras	Spesifikasi
1.	*Processor*	Intel(R) Core(TM) i%-4210U CPU @ 1.70GHz (4 CPUs), ~2.4GHz
2.	*Display*	14" (16:9), LED-backlit HD(1366x768)
3.	*Memory*	12GB DDR3L-SDRAM
4.	*Storage*	256GB-500GB SSD-HDD
5.	*Graphics*	NVIDIA® GeForce® GT 820M
6.	*Input device*	*Keyboard, Mouse*

Perangkat Lunak

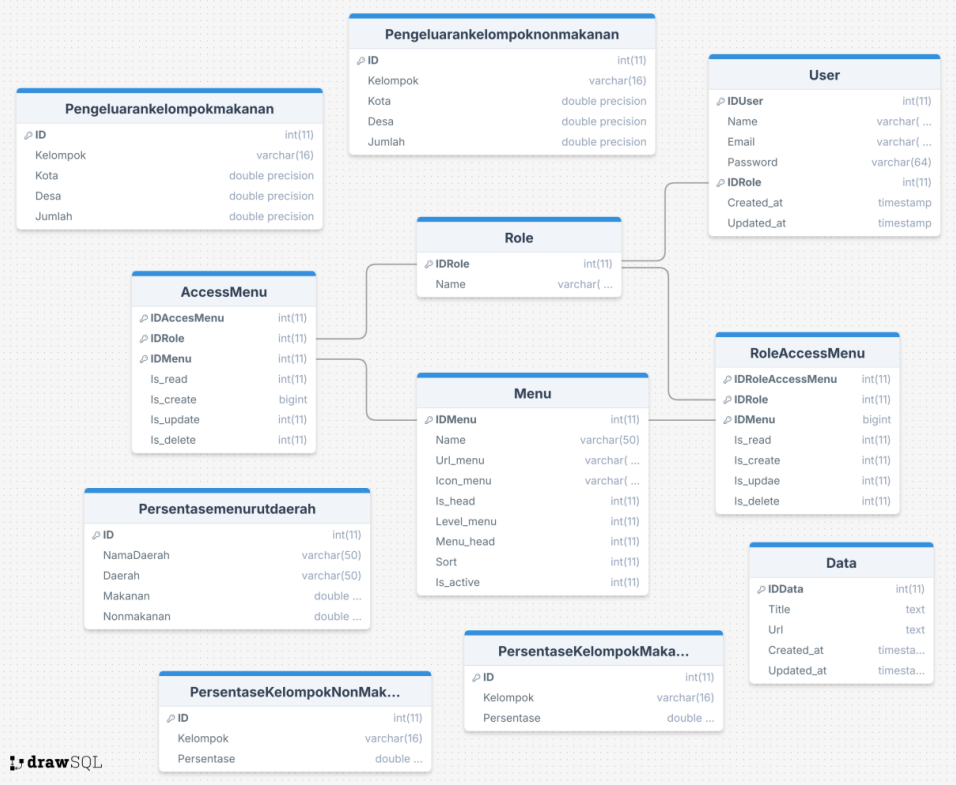
Perangkat lunak serta platform pendukung yang digunakan untuk membangun aplikasi pada penelitian ini sebagai berikut:

Tabel 5. Spesifikasi Perangkat Lunak

No	Perangkat Lunak	Spesifikasi
1.	Sistem Operasi	*Windows 10*
2.	Kode Editor	*Visual Studio Code*
3.	Peramban	*Chrome*
4.	*Local Server*	*XAMPP*
5.	Basis Data	*MySQL*
6.	*Framework*	*CodeIgniter*
7.	Bahasa pemrograman	*PHP*
8.	*Tools Tambahan*	*Postman*
9.		*Composer*
10.	*Platform* Pemodelan Sistem	*Draw.io*

Basis Data

Model basis data ini mendefinisikan tabel yang didalamnya mencakup informasi lainnya seperti tipe data, panjang data yang dapat dilihat secara keseluruhan pada struktur basis data dibawah ini.



Gambar 6. Struktur Basis Datas

REST

Tabel 6. *Request* Dan R*esponse POST* - Login

Variabel	Keterangan
Method	*POST*
Endpoint	http://localhost/portal/api/auth/login
Headers	-
Body	*Key:Email, Password* *Value:* sring, string
Description	Pada bagian *Body* data dikirimkan berupa *form-data* dalam bentuk *Key-Value Pairs*. Data yang dikirimkan adalah *email* dan *password* yang telah terdaftar dan ada didalam basis data dengan tujuan mendapatkan *token* otentikasi.
Status Code	200 OK
Content	{ "*userdata*": { "*admin*": *true*, "*name*": "Adi Widyatama", "*email*": "admin@gmail.com", "*id_user*": "2", "*id_role*": "1", "nama_*role*": "Administrator" }, "*token*":"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJhZG1pbiI6dHJ1ZSwibmFtZSI6IkFkaSBXaWR5YXRhbWEiLCJlbWFpbCI6ImFkbWluQGdtYWlsLmNvbSIsImlkX3VzZXIiOiIyIiwiaWRfcm9sZSI6IjEiLCJuYW1hX3JvbGUiOiJBZG1pbmlzdHJhdG9yIiwiQVBJX1RJTUUiOjE3MjQwNzg5MjJ9.kX4Jl6txSSd0eAQRStuZP6LqmawKXQd4TtCMok0sFIk", "*status*": *true*, "*response*": "*Login Success*!", "*code*": 200

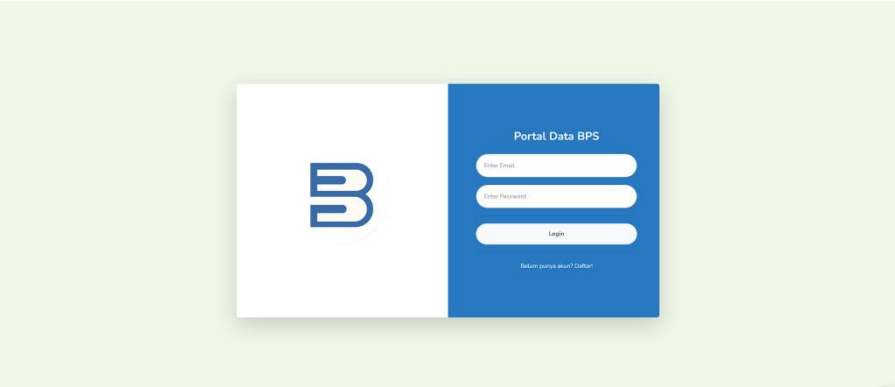
Variabel	Keterangan
	}

Tabel 7. Request Dan Response POST - Lihat Data

Variabel	Keterangan
Method	*POST*
Endpoint	http://localhost/portal/api/pengeluarankelompokmakanan
Headers	*Key: Authorization* *Value:* eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJhZG1pbiI6dHJ1ZSwibmFtZSI6IkFkaSBXaWR5YXRhbWEiLCJlbWFpbCI6ImFkbWluQGdtYWlsLmNvbSIsImlkX3VzZXIiOiIyIiwiaWRfcm9sZSI6IjEiLCJuYW1hX3JvbGUiOiJBZG1pbmlzdHJhdG9yIiwiQVBJX1RJTUUiOjE3MjQwNzg5MjJ9.kX4Jl6txSSd0eAQRStuZP6LqmawKXQd4TtCMok0sFIk
Body	-
Description	Menyertakan *Authorization* dan token yang didapatkan dari *endpoint* sebelumnya yakni */auth/login* pada bagian *Headers* dalam bentuk *Key-Value Pairs.*
Status Code	200 OK
Content	{ "data": [{ "id": "0", "kelompok": "Padi-padian", "kota": "62.877", "desa": "77.664", "jumlah": "66.331" }, { "id": "1", "kelompok": "Umbi-umbian", "kota": "5.666", "desa": "4.449", "jumlah": "5.381" }, { "id": "2", "kelompok": "Ikan", "kota": "39.863", "desa": "30.445", "jumlah": "37.663" },], "*length*": 3, "*start*": 0, "*recordsFiltered*": 14, "*recordsTotal*": 14, "*code*": 200, "*status*": *true* }

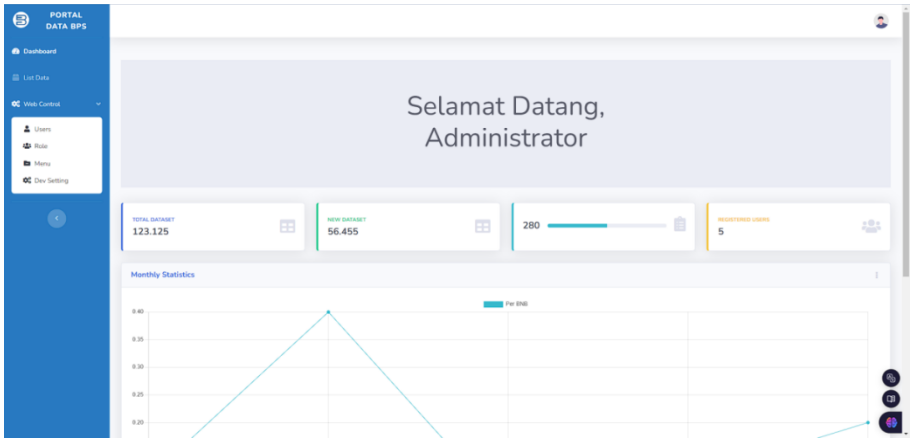
Antar Muka

Menampilkan laman *login* berisi *field email* dan *password* serta tombol *login*. Tampilan login ini menampilkan laman yang sama bagi admin atau pengguna.



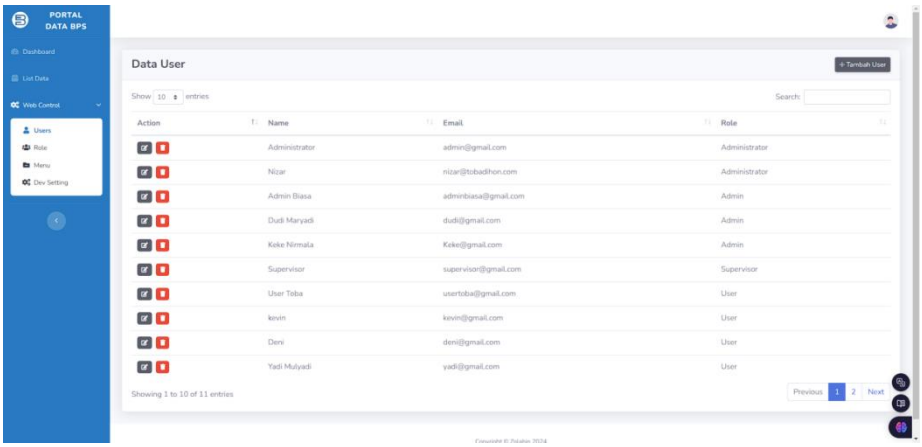
Gambar 7. Implementasi Antar Muka - Masuk

Menampilkan laman beranda atau laman utama bagi admin atau pengguna. Dari laman ini aktor dapat mengakses fitur lainnya yang tersedia di bilah navigasi.



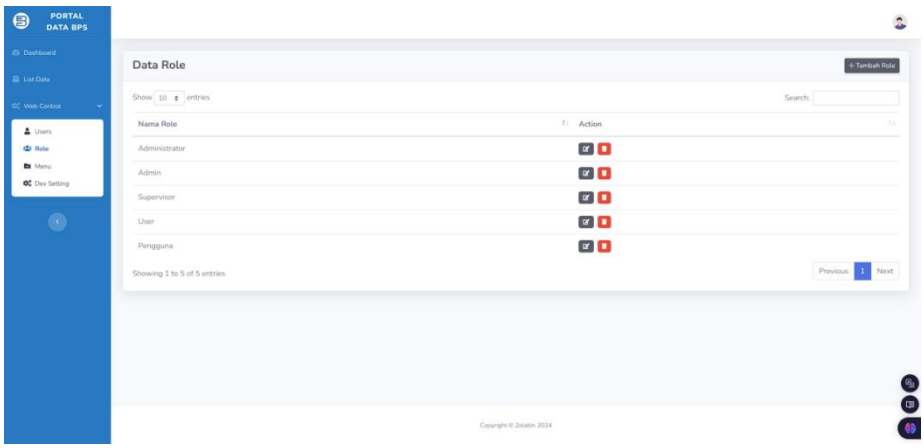
Gambar 8. Implementasi Antar Muka - Beranda Admin

Menampilkan laman data akun pada admin yang berisi tabel yang memuat informasi detail akun dan tombol aksi ubah yang akan mengarahkan ke laman ubah data dan hapus yang akan menampilkan pesan konfirmasi untuk menghapus data.



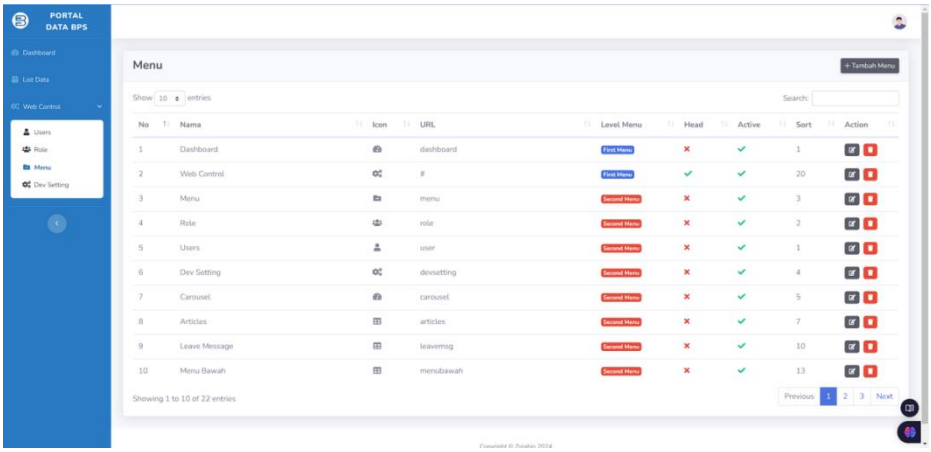
Gambar 9. Implementasi Antar Muka - Data Akun

Menampilkan laman data *role* pada admin yang berisi tabel yang memuat informasi detail *role* dan tombol aksi ubah yang akan mengarahkan ke laman ubah data dan hapus yang akan menampilkan pesan konfirmasi untuk menghapus data.



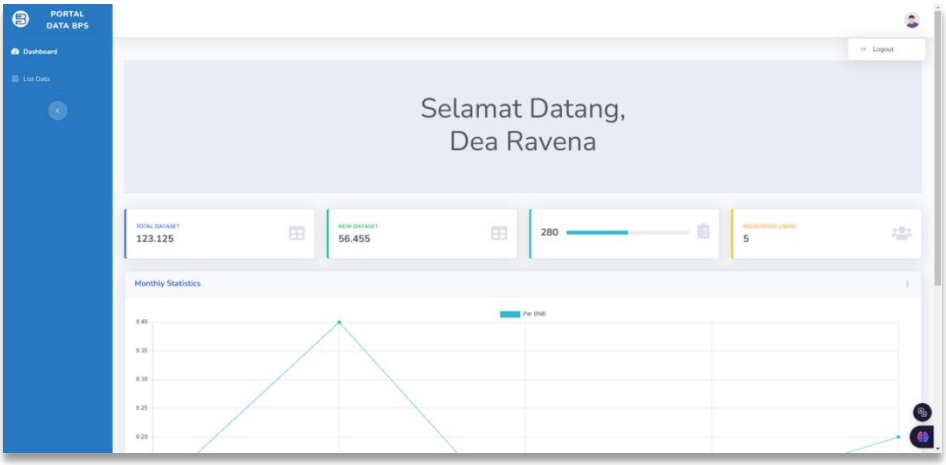
Gambar 10. Implementasi Antar Muka - Data *Role*

Menampilkan laman data menu pada admin yang berisi tabel yang memuat informasi detail menu dan tombol aksi ubah yang akan mengarahkan ke laman ubah data dan hapus yang akan menampilkan pesan konfirmasi untuk menghapus data.



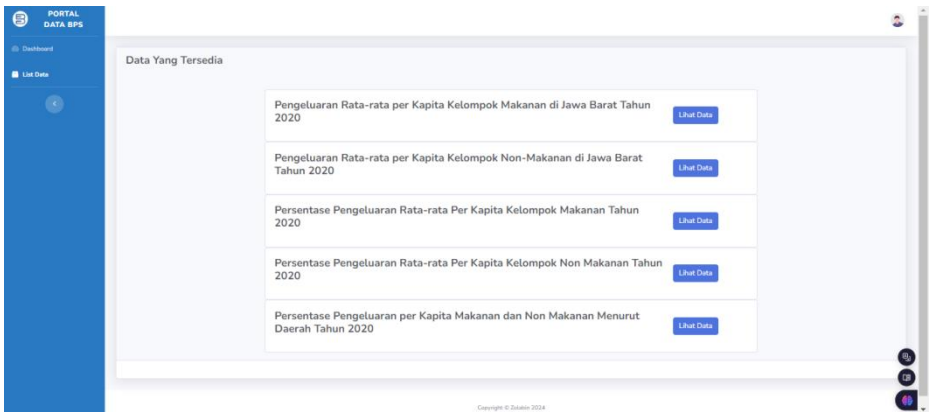
Gambar 11. Implementasi Antar Muka - Data Menu

Menampilkan laman beranda atau laman utama bagi pengguna. Dari laman ini aktor dapat mengakses fitur lainnya yang tersedia di bilah navigasi.



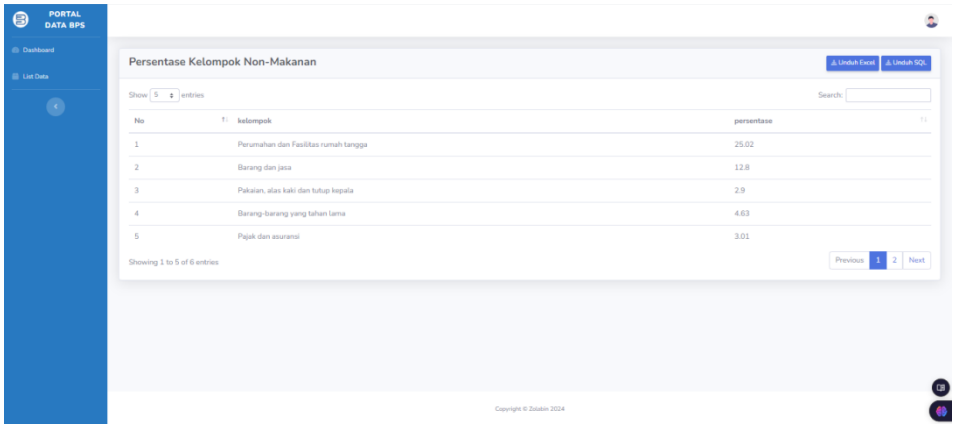
Gambar 12. Implementasi antar muka - Pengguna - Beranda

Menampilkan laman daftar data yang tersedia yang dapat diakses oleh aktor. Laman ini menampilkan judul data dan tombol aksi lihat data untuk mengarahkan aktor ke laman detail data.



Gambar 13. Implementasi Antar Muka - Lihat Data

Menampilkan laman detail data dari data yang sebelumnya dipilih oleh aktor. Laman ini menampilkan tabel, kolom pencarian, tobol aksi unduh dengan format *excel* atau pun *sql*.



Gambar 14. Implementasi Antar Muka - Detail Data

PENGUJIAN

Black Box Metode *State Transition*

Berikut ini skenario dari pengujian *state transition* yang dimulai (dari) suatu laman atau suatu status ke status lainnya (tujuan) yang dipicu oleh suatu (aksi) yang dilakukan oleh aktor.

Tabel 8. Skenario Pengujian State Transition - Admin

No	Dari	Aksi	Tujuan
STA-01	*Form* Masuk	Klik Masuk	Memasuki laman beranda
STA-02	Beranda	Klik List Data	Menampilkan laman List Data
STA-03	List Data	Klik Lihat Data	Menampilkan laman Detail Data
STA-04	Detail Data	Klik Download SQL/Excel	Menampilkan *file browser*
STA-05	Beranda	Klik *Web Control*	Menampilkan detail menu *Web Control*
STA-06	Beranda	Klik *Users*	Menampilkan laman *Data User*
STA-07	Data *User*	Klik Tambah *User*	Menampilkan laman Tambah *User*
STA-08	Data *User*	Klik Hapus	Menampilkan dialog konfirmasi hapus data
STA-09	Data *User*	Klik Ubah Data	Menampilkan laman Ubah Data *User*
STA-10	Beranda	Klik *Role*	Menampilkan laman Data *Role*
STA-11	Data *Role*	Klik Tambah *Role*	Menampilkan laman Tambah *Role*
STA-12	Data *Role*	Klik Hapus	Menampilkan dialog konfirmasi hapus data
STA-13	Data *Role*	Klik Ubah Data	Menampilkan laman Ubah Data *Role*
STA-14	Beranda	Klik Menu	Menampilkan laman Data Menu

 https://doi.org/10.56211/helloworld.v3i4.636

No	Dari	Aksi	Tujuan
STA-15	Data Menu	Klik Tambah Menu	Menampilkan laman Tambah Menu
STA-16	Data Menu	Klik Hapus	Menampilkan dialog konfirmasi hapus data
STA-17	Data Menu	Klik Ubah Data	Menampilkan laman Ubah Data Menu
STA-18	Beranda	Klik *Logout*	Menampilkan dialog konfirmasi keluar dari aplikasi

Tabel 9. Skenario Pengujian State Transition - Pengguna

No	Dari	Aksi	Tujuan
STP-01	*Form* Masuk	Klik Masuk	Memasuki laman beranda
STP-02	Beranda	Klik List Data	Menampilkan laman List Data
STP-03	List Data	Klik Lihat Data	Menampilkan laman Detail Data
STP-04	Detail Data	Klik Download *SQL/Excel*	Menampilkan file browser
STP-05	Beranda	Klik *Logout*	Menampilkan dialog konfirmasi keluar dari aplikasi

Berikut hasil dari pengujian *state transition* berdasarkan skenario diatas.

Tabel 10. Hasil Pengujian *State Transition* - Admin

No	Skenario	Output	Kesimpulan
HSTA-01	Admin mengisi *field* pada laman masuk	Aplikasi menampilkan dialog konfirmasi berhasil memasuki laman Beranda	*Valid*
HSTA-02	Admin mengklik List Data pada laman Beranda	Menampilkan laman List Data berisi daftar data	*Valid*
HSTA-03	Admin mengklik Lihat Data pada salah satu list data	Menampilkan laman Detail Data	*Valid*
HSTA-04	Admin mengklik Download *SQL/Excel* pada laman Detail Data	Menampilkan *file browser* dan lokasi *default* penyimpanan file unduhan	*Valid*
HSTA-05	Mengklik Web Control pada laman Beranda	Menampilkan detail pilihan menu *Web Control*	*Valid*
HSTA-06	Admin mengklik *Users* pada detail pilihan menu *Web Control*	Menampilkan laman Data *User*	*Valid*
HSTA-07	Admin mengklik Tambah *User* pada laman Data *User*	Menampilkan laman Tambah *User*	*Valid*
HSTA-08	Admin mengklik Hapus pada detail Data *User* dilaman Data *User*	Menampilkan dialog konfirmasi menghapus Data *User* yang dipilih	*Valid*
HSTA-09	Admin mengklik Ubah Data pada detail Data *User* dilaman Data *User*	Menampilkan laman Ubah Data *User*	*Valid*
HSTA-10	Admin mengklik *Role* pada detail pilihan menu *Web Control*	Menampilkan laman Data *Role*	*Valid*
HSTA-11	Admin mengklik Tambah *Role* pada laman Data *Role*	Menampilkan laman Tambah *Role*	*Valid*
HSTA-12	Admin mengklik Hapus pada detail Data *Role* dilaman Data *Role*	Menampilkan dialog konfirmasi menghapus Data *Role* yang dipilih	*Valid*
HSTA-13	Admin mengklik Ubah Data detail Data *Role* dilaman Data *Role*	Menampilkan laman Ubah Data *Role*	*Valid*
HSTA-14	Admin mengklik Menu pada detail pilihan menu *Web Control*	Menampilkan laman Data Menu	*Valid*
HSTA-15	Admin mengklik Tambah Menu pada detail Data Menu pada laman Data Menu	Menampilkan laman Tambah Data Menu	*Valid*
HSTA-16	Admin mengklik Hapus pada detail Data Menu pada laman Data Menu	Menampilkan dialog konfirmasi menghapus Data Menu yang dipilih	*Valid*
HSTA-17	Admin mengklik Ubah Data pada detail Data Menu pada laman Data Menu	Menampilkan laman Ubah Data Menu	*Valid*

No	Skenario	Output	Kesimpulan
HSTA-18	Admin mengklik *Logout*	Menampilkan dialog konfirmasi keluar dari aplikasi	*Valid*

Tabel 11. Hasil Pengujian *State Transition* - Pengguna

No	Skenario	Output	Kesimpulan
HTP-01	Pengguna mengisi field pada laman masuk	Aplikasi menampilkan dialog konfirmasi berhasil memasuki laman Beranda	*Valid*
HTP-02	Pengguna mengklik List Data pada laman Beranda	Menampilkan laman List Data berisi daftar data	*Valid*
HTP-03	Pengguna mengklik Lihat Data pada salah satu list data	Menampilkan laman Detail Data	*Valid*
HTP-04	Pengguna mengklik Download *SQL/Excel* pada laman Detail Data	Menampilkan *file browser* dan lokasi *default* penyimpanan file unduhan	*Valid*
HTP-05	Pengguna mengklik *Logout*	Menampilkan dialog konfirmasi keluar dari aplikasi	*Valid*

Black Box Metode *Equivalance Partioning*

Skenario pengujian ini dilakukan dengan mengisi *field* yang ada pada fitur-fitur tertentu utamanya seperti *CRUD*, dengan memasukkan input yang benar, input yang salah dan mengosongkan field tersebut untuk melihat apakah hasil keluarannya sesuai dengan yang diharapkan.

Tabel 12. Skenario pengujian *Equivalance Partioning*

No	Fitur	Skenario	Hasil yang diharapkan	Kesimpulan
SP-01	Masuk	Memasukkan *email* dan sandi yang *valid*	Berhasil memasuki sistem dan tampil beranda	*Valid*
SP-02	Masuk	Memasukkan *email* dan sandi yang *invalid*	Gagal memasuki sistem dan tampil pesan kesalahan	*Invalid*
SP-03	Masuk	Mengosongkan seluruh atau salah satu *field*	Gagal memasuki sistem dan tampil pesan kesalahan	*Invalid*
SP-04	Tambah *User*	Mengisi seluruh *field* dengan data yang sesuai	Data *User* tersimpan dan tampil dialog pesan *Success*	*Valid*
SP-05	Tambah *User*	Mengisi seluruh *field* dengan data yang tidak sesuai	Menampilkan *error message* dibawah *field* dan data gagal disimpan	*Valid*
SP-06	Tambah *User*	Mengosongkan seluruh atau salah satu *field*	Menampilkan *error message* dibawah *field* dan data gagal disimpan	*Valid*
SP-07	Hapus *User*	Mengklik *Delete* pada dialog konfirmasi	Menampilkan dialog Success dan data berhasil dihapus	*Valid*
SP-08	Hapus *User*	Mengklik *Cancel* pada dialog konfirmasi	Menutup dialog konfirmasi hapus dan data tidak dihapus	*Invalid*
SP-09	Ubah Data *User*	Mengubah *field* dengan data yang sesuai	Menampilkan dialog *Success* dan data berhasil diubah dan disimpan	*Valid*
SP-10	Ubah Data *User*	Mengubah *field* dengan data yang tidak sesuai	Menampilkan *error message* dibawah *field* dan data gagal diperbarui	*Invalid*
SP-11	Ubah Data *User*	Mengosongkan seluruh atau salah satu *field*	Menampilkan *error message* dibawah *field* dan data gagal diperbarui	*Invalid*
SP-12	Tambah *Role*	Mengisi seluruh *field* dengan data yang sesuai	Data *Role* tersimpan dan tampil dialog pesan *Success*	*Valid*
SP-13	Tambah *Role*	Mengisi seluruh *field* dengan data yang tidak sesuai	Menampilkan *error message* dibawah *field* dan data gagal disimpan	*Invalid*
SP-14	Tambah *Role*	Mengosongkan seluruh atau salah satu *field*	Menampilkan *error message* dibawah *field* dan data gagal disimpan	*Invalid*

 https://doi.org/10.56211/helloworld.v3i4.636

No	Fitur	Skenario	Hasil yang diharapkan	Kesimpulan
SP-15	Hapus *Role*	Mengklik *Delete* pada dialog konfirmasi	Menampilkan dialog Success dan data berhasil dihapus	*Valid*
SP-16	Hapus *Role*	Mengklik *Cancel* pada dialog konfirmasi	Menutup dialog konfirmasi hapus dan data tidak dihapus	*Invalid*
SP-17	Ubah Data *Role*	Mengubah *field* dengan data yang sesuai	Menampilkan dialog Success dan data berhasil diubah dan disimpan	*Valid*
SP-18	Ubah Data *Role*	Mengubah *field* dengan data yang tidak sesuai	Menampilkan *error message* dibawah *field* dan data gagal diperbarui	*Invalid*
SP-19	Ubah Data *Role*	Mengosongkan seluruh atau salah satu *field*	Menampilkan *error message* dibawah *field* dan data gagal diperbarui	*Invalid*
SP-20	Tambah Menu	Mengisi seluruh *field* dengan data yang sesuai	Data Menu tersimpan dan tampil dialog pesan *Success*	*Valid*
SP-21	Tambah Menu	Mengisi seluruh *field* dengan data yang tidak sesuai	Menampilkan *error message* dibawah *field* dan data gagal disimpan	*Invalid*
SP-22	Tambah Menu	Mengosongkan seluruh atau salah satu *field*	Menampilkan *error message* dibawah *field* dan data gagal disimpan	*Invalid*
SP-23	Hapus Menu	Mengklik *Delete* pada dialog konfirmasi	Menampilkan dialog Success dan data berhasil dihapus	*Valid*
SP-24	Hapus Menu	Mengklik *Cancel* pada dialog konfirmasi	Menutup dialog konfirmasi hapus dan data tidak dihapus	*Invalid*
SP-25	Ubah Data Menu	Mengubah *field* dengan data yang sesuai	Menampilkan dialog *Success* dan data berhasil diubah dan disimpan	*Valid*
SP-26	Ubah Data Menu	Mengubah *field* dengan data yang tidak sesuai	Menampilkan *error message* dibawah *field* dan data gagal diperbarui	*Invalid*
SP-27	Ubah Data Menu	Mengosongkan seluruh atau salah satu *field*	Menampilkan *error message* dibawah *field* dan data gagal diperbarui	*Invalid*
SP-28	*Logout*	Mengklik Keluar	Berhasil keluar dari aplikasi dan menampilkan laman Masuk	*Valid*
SP-29	*Logout*	Mengklik *Cancel*	Gagal keluar dari aplikasi	*Invalid*

Input dibagi menjadi beberapa kelompok seperti (1) Input Benar, menjawab dengan format data yang benar dan sesuai ketentuan. (2) Input Salah, memasukkan format data yang tidak valid atau data yang tidak sesuai dengan ketentuan yang ditetapkan. (3) Input Kosong, tidak memberikan input sama sekali.Maka akan didapatkan hasil kesimpulan berupa *valid* atau *invalid*. *Invalid* disini artinya ketika memberikan input yang salah atau kosong dan sistem merespon dengan benar (menganggap input tersebut tidak *valid*).

User Acceptance Testing

Hasil pengujian yang dilakukan ini menunjukkan apakah aplikasi yang dibangun sudah memenuhi persyaratan fungsional atau belum. Data ini didapatkan dari hasil kuesioner dengan 10 butir pertanyaan sebagai berikut.

Tabel 13. Daftar pertanyaan kuesioner *UAT*

No	Pertanyaan
1.	Apakah penggunaan layanan *cloud storage* pada proses pendistribusian data sangat memudahkan?
2.	Apakah penggunaan aplikasi ini pada proses pendistribusian data dapat memudahkan?
3.	Apakah layanan *cloud storage* dapat mendistribusikan data dengan efisien dan cepat?
4.	Apakah aplikasi ini dapat mendistribusikan data dengan efisien dan cepat?
5.	Apakah kemampuan layanan *cloud storage* dapat memenuhi kebutuhan pendistribusian data di masa depan?
6.	Apakah kemampuan aplikasi ini dapat memenuhi kebutuhan pendistribusian data dimasa depan?
7.	Apakah proses pendistribusian data dengan layanan *cloud storage* cukup andal dan konsisten?
8.	Apakah proses pendistribusian data dengan aplikasi ini cukup andal dan konsisten?
9.	Apakah Anda puas dengan layanan *cloud storage* secara keseluruhan?
10.	Apakah Anda puas dengan aplikasi ini secara keseluruhan?

Dan berikut adalah hasil dari kuesioner yang diperoleh dari 6 responden. Jawaban sudah dijumlahkan dengan mengalikan jumlah responden yang memilih masing-masing jawaban dengan bobot berikut 5 (Sangat Setuju), 4 (Setuju), 3 (Netral), 2 (Tidak Setuju) dan 1 (Sangat Tidak Setuju). Setelah didapatkan total skor dari tabel diatas maka selanjutnya adalah menghitungnya sebagai berikut

1. Jumlah skor ideal = bobot tertinggi (5) x jumlah responden (6) = 30
2. Persentase = (total skor / jumlah skor ideal) x 100

Tabel 14. Hasil Pengujian *UAT* Terhadap *Cloud Storage*

Aspek Pengujian	Frekuensi Jawaban					Total Skor	Persentase
	5	4	3	2	1		
1. Kemudahan	3	2	-	1	-	15+8+2 = 25	(25/30) x 100 = 83,3%
2. Efisiensi	1	3	1	1	-	5+12+3+2 = 22	(22/30) x 100 = 73,3%
3. Kebutuhan dimasa mendatang	3	2	-	1	-	15+8+2 = 25	(25/30) x 100 = 83,3%
4. Keandalan dan konsistensi	2	3	1	-	-	10+12+3 = 25	(25/30) x 100 = 83,3%
5. Kepuasan	1	4	-	1	-	5+16+2 = 23	(23/30) x 100 = 76,6%

Selanjutnya hasil dari kuesioner penilaian terhadap aplikasi usulan dengan jumlah responden yang sama dan jawaban yang sudah dijumlahkan dengan mengalikan jumlah responden yang memilih masing-masing jawaban.

Tabel 15. Hasil Pengujian *UAT* Terhadap Aplikasi Usulan

Aspek Pengujian	Frekuensi Jawaban					Total Skor	Persentase
	5	4	3	2	1		
1. Kemudahan	3	3	-	-	-	15+12 = 27	(27/30) x 100 = 90%
2. Efisiensi	3	4	-	-	-	15+12 = 27	(27/30) x 100 = 90%
3. Kebutuhan dimasa mendatang	1	5	-	-	-	5+20 = 25	(25/30) x 100 = 83,3%
4. Keandalan dan konsistensi	1	5	-	-	-	5+20 = 25	(25/30) x 100 = 83,3%
5. Kepuasan	1	5	-	-	-	5+20 = 25	(25/30) x 100 = 83,3%

Aplikasi usulan memiliki persentase lebih tinggi yakni 90% dibandingkan persentase *cloud storage* sebesar 83,3% pada aspek kemudahan hal ini menunjukkan bahwa aplikasi usulan lebih mudah digunakan. Begitu juga dengan efisiensinya aplikasi usulan memiliki nilai yang sama yakni 90% dan lebih tinggi dari efisiensi cloud storage yang hanya sebesar 73,3% hal ini menunjukkan bahwa aplikasi usulan cukup efisien, sedangkan pada potensi kemampuan aplikasi berkembang sesuai kebutuhan di masa mendatang keduanya memiliki persentase yang sama yakni sebesar 83,3% hal ini menunjukkan bahwa keduanya dapat memenuhi kebutuhan yang diharapkan pada masa yang mendatang.

Sama halnya dengan aspek keandalan dan konsistensi yang juga memiliki persentase sebesar 83,3% untuk kedua sistem, hal ini menunjukkan bahwa keduanya dapat diandalkan dan terakhir adalah persentase kepuasan dimana aplikasi usulan memiliki persentase 83,3% sedangkan *cloud storage* sebesar 76,6% sehingga hal ini menunjukkan bahwa pengguna cukup puas dengan aplikasi usulan.

Secara keseluruhan, aplikasi usulan menunjukkan performa yang lebih baik dalam aspek kemudahan, efisiensi dan kepuasan, sementara *cloud storage* dan aplikasi usulan memiliki kesetaraan aspek keandalan dan konsistensi serta kemampuan berkembang sesuai kebutuhan dimasa mendatang.

KESIMPULAN DAN SARAN

Penelitian ini telah berhasil membangun sebuah aplikasi manajemen distribusi data yang mengadopsi arsitektur *REST* dan memanfaatkan *framework CodeIgniter*, aplikasi ini mampu mengintegrasikan data dari sumber data untuk menyediakan

 https://doi.org/10.56211/helloworld.v3i4.636

data yang akurat dan terkini. Penerapan mekanisme *role-based access control* memastikan keamanan data dan mencegah akses yang tidak sah.

Dari hasil pengujian dengan metode *User Acceptance Testing* (*UAT*) terhadap aplikasi usulan pada penelitian ini juga menunjukkan peningkatan signifikan dalam hal kemudahan dan efisiensi. Berdasarkan hasil pengujian aplikasi usulan memperoleh skor 90% untuk aspek kemudahan dan efisiensi, dibandingkan dengan *cloud storage* yang masing-masing hanya mencapai 83,3% dan 73,3%. Tingkat kepuasan pengguna pada aplikasi usulan juga lebih tinggi yaitu 83,3% dan persentase *cloud storage* sebesar 76,6% sedangkan dari segi keandalan dan potensi pengembangan keduanya memiliki persentase yang sama yaitu 83,3%.

Secara keseluruhan, aplikasi usulan ini menawarkan peningkatan dalam hal efisiensi, keamanan, dan kemudahan dalam proses pendistribusian data di BPS Provinsi Jawa Barat, serta memiliki potensi lebih untuk dikembangkan lebih lanjut guna memenuhi kebutuhan distribusi data yang terus meningkat.

Berdasarkan kesimpulan di atas, beberapa saran yang dapat diberikan untuk pengembangan dan peningkatan aplikasi manajemen distribusi data ini adalah sebagai berikut:
Pengembangan Fitur Tambahan: Aplikasi dapat diperkuat dengan menambahkan fitur-fitur tambahan yang dapat meningkatkan kegunaannya, peningkatan fungsi pencarian dan tabel yang lebih dinamis atau pengembangan *dashboard* yang lebih interaktif untuk memudahkan pengguna dalam mengakses informasi penting. Integrasi dengan Sistem Lain: Aplikasi dapat diintegrasikan dengan platform analitik atau visualisasi data eksternal seperti *Tableau* atau *Power BI* untuk memberikan fungsionalitas yang lebih komprehensif dalam pengolahan data. Peningkatan Keamanan: Penggunaan enkripsi data atau autentikasi multi=faktor dan *audit log* yang lebih mendetail dapat menjadi langkah tambahan untuk meningkatkan keamanan pada aplikasi.

DAFTAR PUSTAKA

[1] G. Ramachandra, M. Iftikhar, and F. A. Khan, "A Comprehensive Survey on Security in Cloud Computing," *Procedia Comput. Sci.*, vol. 110, no. 2012, pp. 465–472, 2017, doi: 10.1016/j.procs.2017.06.124.

[2] R. T. Fielding and R. N. Taylor, "Principled Design of the Modern Web Architecture," *ACM Trans. Internet Technol.*, vol. 2, no. 2, pp. 115–150, 2002, doi: 10.1145/514183.514185.

[3] B. H. Hasanuddin, Hari Asgar, "Rest Api," *J. Inform. Teknol. dan Sains*, vol. 4, no. 1, pp. 412–416, 2022, doi: 10.3139/9783446473157.024.

[4] M. Goodwin, "What Is An API (Application Programming Interface)?," Internet: https://www.ibm.com/id-id/topics/api, [May 26, 2024].

[5] D. Saputra, "Analisis Perbandingan Performa Web Service Rest Menggunakan Framework Laravel, Django Dan Ruby On Rails Untuk Akses Data Dengan," *J. Bangkit Indones.*, vol. 7, no. 2, p. 17, 2018, doi: 10.52771/bangkitindonesia.v7i2.90.

[6] I. A. K. P. Paramitha, D. M. Wiharta, I. M. Arsa, and Suyadnya, "PERANCANGAN DAN IMPLEMENTASI RESTFUL API PADA SISTEM INFORMASI MANAJEMEN DOSEN UNIVERSITAS UDAYANA," *J. SPEKTRUM*, vol. 9, no. 3, 2022.

[7] M. A. H. S. Muh Rais, "Inventory Information System Of Goods Using CodeIgniter Freamework," *Patria Artha Technol. J.*, vol. 3, no. 1, 2019.

[8] A. Haniefardy, M. B. A. Fadhillah, and S. Rochimah, "Tinjauan Literatur Sistematis: Pengaruh Penggunaan Framework Khusus dalam Proses Pengembangan Web dan Pembuatan Web," *Matrix J. Manaj. Teknol. dan Inform.*, vol. 9, no. 2, pp. 68–73, 2019, doi: 10.31940/matrix.v9i2.1161.

[9] CodeIgniter, "Welcome to CodeIgniter4," https://codeigniter.com/user_guide/intro/index.htm, [Aug. 23, 2024]. Internet:

[10] M. Destiningrum and Q. J. Adrian, "Sistem Informasi Penjadwalan Dokter Berbassis Web Dengan Menggunakan Framework Codeigniter (Studi Kasus: Rumah Sakit Yukum Medical Centre)," *J. Teknoinfo*, vol. 11, no. 2, p. 30, 2017, doi: 10.33365/jti.v11i2.24.

[11] R. Mugi, N. Musriatun, and S. A. Rian, "Perancangan Sistem Informasi Dengan PHP Dan MYSQL Untuk Pendaftaran Sekolah Di Masa Pandemi," *Comput. Sci.*, vol. 2, no. 1, pp. 50–58, 2022, [Online]. Available: http://jurnal.bsi.ac.id/index.php/co-science

[12] O. S. Parulian, *3 Days With MySQL for your Application: MySQL untuk Pemula*. Onesinus Saut Parulian, 2018.

[13] A. B. Putra and S. Nita, "Perancangan dan Pembangunan Sistem Informasi E-Learning Berbasis Web (Studi Kasus Pada Madrasah Aliyah Kare Madiun)," *Semin. Nas. Teknol. Inf. dan Komun. 2019*, vol. 1, no. 1, pp. 81–85, 2019.

[14] D. Setiyadi, "Stuctured Query Language (SQL) untuk Purchase Order (PO) menggunakan SQL Server," *Bina*

Insa. ICT J., vol. 6, no. 1, pp. 75–88, 2019.
[15] S. Al-Maktabah, J. Perpustakaan, A. dan Dokumentasi, L. Tantowi, and L. Wijayanti, "Draf Artikel pada Dokumen Digital," *Shaut Al-Maktabah J. Perpustakaan, Arsip dan Dokumentasi*, vol. 15, no. 1, pp. 118–131, 2023.
[16] M. Riyan Dirgantara, S. Syahputri, and A. Hasibuan, "Pengenalan Database Management System (DBMS)," *J. Ilm. Multidisipline*, vol. 1, no. 6, pp. 300–301, 2023, [Online]. Available: https://doi.org/10.5281/zenodo.8123019
[17] T. H. Ilhaam Syarifudding Akbar, "Pengembangan Entit Relationship Diagram Database Toko Online Ira Surabaya," *J. Ilm. Comput. Insight*, vol. 3, no. 2, 2021.
[18] Uminingsih, M. Nur Ichsanudin, M. Yusuf, and S. Suraya, "Pengujian Fungsional Perangkat Lunak Sistem Informasi Perpustakaan Dengan Metode Black Box Testing Bagi Pemula," *STORAGE J. Ilm. Tek. dan Ilmu Komput.*, vol. 1, no. 2, pp. 1–8, 2022, doi: 10.55123/storage.v1i2.270.
[19] E. C. Foster and S. Godbole, *Database systems: A pragmatic approach*. 2016. doi: 10.1007/978-1-4842-1191-5.
[20] H. Sujaini *et al.*, "Evaluasi Kinerja Internet Kampus Universitas Tanjungpura dengan Analisis Quality of Service dan User Acceptance Test," *J. Edukasi dan Penelit. Inform.*, vol. 9, no. 1, p. 89, 2023, doi: 10.26418/jp.v9i1.63541.