JITET (Jurnal Informatika dan Teknik Elektro Terapan)
Vol.11 No.3 S1, pISSN:2303-0577 eISSN:2830-7062 http://dx.doi.org/10.23960/jitet.v11i3%20s1.3506

PERANCANGAN *MICROSERVICE* BERBASIS *REST API* PADA *GOOGLE CLOUD PLATFORM* MENGGUNAKAN *NODEJS* DAN *PYTHON*

Royyan Fajrul Falah[1*], Muhamad Komarudin[2], Mahendra Pratama[3]

1,2,3 Teknik Informatika, Universitas Lampung, Jl. Prof. Soemantri Brojonegoro, Bandar Lampung 35145

Riwayat artikel:
Received: 15 Agustus 2023
Accepted: 2 September 2023
Published: 11 September 2023

Keywords:
Penyakit daun padi; Microservice; REST API; NodeJS; Python;

Corespondent Email:
royyan005@gmail.com

Abstrak. Di Indonesia, penyakit penting pada daun padi ialah hawar daun bakteri, penyakit tungro, bercak daun, dan hawar pelepah daun. Penyakit-penyakit tersebut sangat berpengaruh terhadap hasil panen dan kualitas panen dari komoditas padi. Berdasarkan masalah diatas, salah satu cara penganggulangannya adalah dengan membuat sebuah aplikasi yang dapat mendeteksi penyakit daun padi. Aplikasi pendeteksi penyakit daun padi ini bernama RIFSA (Rice Farmer Assistant) berbasis mobile. Untuk mendukung aplikasi tersebut, dibuatlah sistem microservice berbasis REST API menggunakan NodeJS dan Python pada Google Cloud Platform. Microservice berbasis REST API telah berhasil dibuat menggunakan NodeJS dan Python dengan fitur yaitu Authentication, Hasil Panen, Inventaris, Keuangan, dan Penyakit.

Abstract. *In Indonesia, important disease of a rice plant's leaf is bacterial leaf blight, tungro disease, leaf spot, and leaf sheath blight. Those disease affects harvest results and harvest quality of rice plants. By problems mentioned above, one of the solutions is to make a rice leaf disease detection app. This rice leaf disease detection application is called RIFSA (Rice Farmer Assistant), mobile based. To support this application, a REST API-based microservice system was created using NodeJS and Python on Google Cloud Platform. Microservice system was successfully build with features such as authentication, harvest result, inventory, finances and disease*.

1. PENDAHULUAN

Salah satu komoditas tanaman pangan unggulan di Indonesia adalah padi. Hasil produksi dari padi ini masih menjadi bahan makanan pokok bagi masyarakatnya.[1] Indonesia mengimpor 1,6 juta ton padi pada tahun 2011 dan meningkat menjadi 1,9 juta ton padi pada tahun 2012. Hal ini membuktikan bahwa padi merupakan tanaman pangan pokok bagi masyarakat Indonesia.[2]

Penyakit pada padi sangat berpengaruh terhadap hasil panen dan kualitas panen dari komoditas padi. Para petani adalah pihak yang paling dirugikan dengan hadirnya penyakit daun padi tersebut.[3] Oleh karena itu, perlu adanya usaha penanggulangan penyakit supaya hasil panen dari para petani padi tetap berkualitas.

Berdasarkan masalah diatas, salah satu cara penganggulangannya adalah dengan membuat sebuah aplikasi yang dapat mendeteksi penyakit daun padi. Adanya aplikasi ini berfungsi untuk mempermudah para petani padi untuk mengidentifikasi penyakit yang menjangkit tanaman padi milik mereka serta cara penganggulangannya agar cepat diatasi. Aplikasi pendeteksi penyakit daun padi ini bernama RIFSA (*Rice Farmer Assistant*) berbasis *mobile*. Rifsa adalah salah satu *capstone project* pada Bangkit Academy 2022.

Microservice adalah sebuah *framework* arsitektur rekayasa perangkat lunak yang saat ini telah populer digunakan oleh para developer di seluruh dunia.[4] Pemilihan penggunaan gaya arsitektur *microservice* didasarkan pada kebutuhan aplikasi RIFSA dimana terdapat

beberapa *service* berbeda yang membutuhkan bahasa pemrograman yang berbeda pula. Oleh karena itu, digunakan arsitektur *microservice* untuk mendukung kebutuhan dari aplikasi RIFSA supaya dapat berjalan secara efektif.

2. TINJAUAN PUSTAKA

2.1. Bangkit Academy 2022

Program Bangkit Academy 2022 dilaksanakan oleh Direktorat Jenderal Pendidikan Tinggi (Ditjen Dikti) bekerja sama dengan Google, Gojek, Tokopedia dan Traveloka, berdasarkan pemberitahuan resmi Kementerian Pendidikan dan Kebudayaan RI tertanggal 11 Desember 2020. Program Bangkit Academy 2022 mencakup tiga jalur pembelajaran: Machine learning dengan TensorFlow (Machine learning), Pemrograman dengan Android (Mobile Development) dan Cloud Computing dengan Compute Engine (Cloud Computing).

2.2. Microservice

Arsitektur *microservice* adalah sistem pengembangan perangkat lunak yang dibangun dari susunan beberapa *service* kecil yang berkomunikasi melalui HTTP dengan mekanisme sederhana. Arsitektur *microservice* membuat pengembangan sebuah aplikasi menjadi lebih cepat, lebih mudah, dan dapat diperbaharui secara terpisah. [5]

2.3. Google Cloud Platform

Google Cloud Platform adalah layanan *public cloud computing* dari Google yang terdiri dari beragam layanan. Dengan menggunakan layanan yang ada pada *Google Cloud Platform*, kita dapat membuat aplikasi *virtual machine*, jaringan, hingga *machine learning* sesuai keinginan dan kebutuhan kita.[6]

2.4. NodeJS

Node.js adalah sistem perangkat lunak yang didesain untuk pengembangan aplikasi web. Aplikasi ini ditulis dalam bahasa JavaScript, menggunakan basis *event* dan *asynchrounous* I/O. Tidak seperti kebanyakan bahasa JavaScript yang dijalankan pada peramban, Node.js dieksekusi sebagai aplikasi *server*. Aplikasi ini terdiri dari V8 *JavaScript Engine* buatan Google dan beberapa modul bawaan yang terintegrasi seperti modul http, modul *filesystem*, modul *security* dan beberapa modul penting lainnya.[7]

2.5. Python

Python adalah sebuah bahasa pemrograman yang memiliki sifat *interpreter, interactive, object-oriented*, dan dapat beroperasi hampir di semua platform seperti Mac, Windows, dan Linux. *Python* termasuk dalam bahasa pemrograman tingkat tinggi. *Python* termasuk bahasa pemograman yang mudah dipelajari karena sintaks yang jelas, dapat dikombinasikan dengan penggunaan modul-modul siap pakai, dan struktur data tingkat tinggi yang efisien.[8]

2.6. REST API

REST API adalah salah satu jenis *web service. REST API* memungkinkan untuk melakukan *request system* untuk mengakses dan memanipulasi teks yang direpresentasikan dari sebuah *web service. REST API* sering digunakan untuk pengembangan *web service* yang scalable, mudah dipahami, dan interoperabilitas yang baik.[9]

2.7. Machine Learning

Machine learning adalah bagian dari *Artificial Intelligence (AI). Machine learning* dapat dikatakan sebagai sebuah aplikasi atau algoritma matematika yang melakukan pembelajaran melalui data dan menghasilkan prediksi masa depan berdasarkan data tersebut. *Machine learning* dibagi menjadi 3 jenis*: Supervised Learning, Unsupervised Learning, dan Reinforcement Learning.*[10]

2.8. Extreme Programming (XP)

Extreme Programming (XP) adalah metode pengembangan perangkat lunak yang sederhana dan mencakup salah satu metode tangkas yang dipelopori oleh Kent Beck, Ron Jeffries, dan Ward Cunningham. Extreme Programming (XP) merupakan sebuah proses rekayasa perangkat lunak yang cenderung menggunakan pendekatan berorientasi objek dan sasaran dari metode ini adalah tim yang dibentuk dalam skala kecil sampai medium

serta metode ini juga sesuai jika tim dihadapkan dengan requirement yang tidak jelas maupun terjadi perubahan–perubahan requirement yang sangat cepat. [11]

2.9. Pengujian Blackbox

Black box testing merupakan pengujian kualitas perangkat lunak yang berfokus pada fungsionalitas perangkat lunak. Pengujian black box bertujuan untuk menemukan fungsi yang tidak benar, kesalahan antarmuka, kesalahan pada struktur data, kesalahan performansi, kesalahan inisialisasi dan terminasi. [12]

2.10. Performance Test

Performance testing merupakan salah satu jenis pengujian yang digunakan untuk menguji stabilitas dan keandalan sistem. Pengujian akan dilakukan dengan jumlah pengguna yang berubah-ubah dalam suatu waktu dan akan diuji bagaimana respon yang diberikan oleh sistem saat diakses dengan jumlah pengguna yang banyak dan dalam waktu tertentu. Performance testing dilakukan untuk memastikan bahwa sistem tidak akan crash di bawah situasi krisis. Tujuan dari Performance testing adalah untuk memastikan kegagalan sistem dan untuk memantau bagaimana sistem pulih kembali dengan baik, jika sistem dapat memulihkan tanpa kehilangan data dan tanpa menimbulkan kebocoran keamanan berarti telah berhasil melewati Performance testing. [13]

2.11. BlazeMeter

BlazeMeter adalah platform pengujian kinerja perangkat lunak dan aplikasi web yang didesain untuk memudahkan pengujian kinerja aplikasi dengan skala besar dan kompleks. Platform ini dapat melakukan pengujian kinerja aplikasi secara terdistribusi dan dapat mensimulasikan pengguna yang beroperasi dari berbagai lokasi geografis.

3. METODE PENELITIAN

3.1. Waktu dan Tempat Penelitian

Waktu penelitian dilakukan pada bulan Desember 2022 sampai dengan April 2023 yang bertempat di Universitas Lampung. Berikut adalah tabel 1 yang menunjukkan jadwal kegiatan penelitian yang dilakukan.

Tabel 1 Waktu Penelitian

No	Aktivitas	Desember 2022				Januari 2023				Februari 2023				Maret 2023				April 2023			
		Minggu ke -																			
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1	*Studi Literatur*																				
2	*Analisis Kebutuhan*																				
3	*Planning*																				
4	*Design*																				
5	*Coding*																				
6	*Testing*																				
7	*Iterasi Sistem*																				
8	*Pelaporan*																				

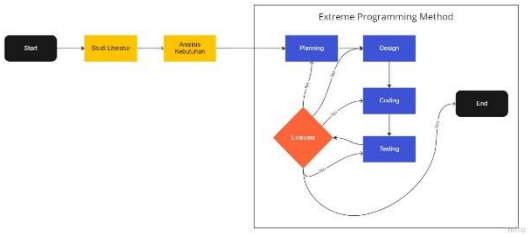
3.2. Alat Penelitian

Tabel 2 Alat Penelitian

No	Nama Alat	Spesifikasi	Deskripsi
1.	Laptop	Asus Nitro 5 AN515-55 Core i5 10300H dengan RAM 16 GB dan sistem operasi windows 10	Perangkat keras yang digunakan dalam proses pembuatan *Microservice*
2.	Aplikasi Visual Studio Code	Version 1.74.2	Perangkat lunak yang digunakan dalam proses pembuatan *Microservice*
3.	Postman	Version 10.70	Perangkat lunak yang digunakan untuk melakukan dokumentasi API dan *testing* API
4	BlazeMeter	-	Platform untuk melakukan *Performance test* berbasis JMeter
5	Virtual Machine (Fitur Compute Engine pada *Google Cloud Platform*)	E2-Small (2 vCPU, 2 GB RAM, 10 GB SSD Disk, Debian OS)	Mesin virtual untuk melakukan proses *deployment*

3.3. Tahapan Penelitian

Penelitian ini menggunakan metodologi SDLC Agile Model XR *(Extreme Programming)* yang meliputi *Planning, Design, Coding, dan Testing*. Kemudian selama proses penelitian dilakukan Pelaporan. Metode ini merupakan salah satu model dari metode *Agile. Extreme Programming* berfokus pada *coding* sebagai aktivitas utama di semua tahap pada siklus pengembangan yang dengan cepat merespons permintaan customer dan membuat software berkualitas yang lebih baik. Tahapan penelitian menggunakan metode XR disajikan dalam gambar berikut :



Gambar 1 Tahapan Penelitian

4. HASIL DAN PEMBAHASAN

4.1. Iterasi 1

Pada metode Extreme Programming (XP), terdapat iterasi yang menandakan 1 alur penuh pengembangan sistem informasi yang terdiri dari planning, design, coding, dan testing.

4.1.1. Planning

Pada proses ini, dilakukan 2 langkah, yaitu mengidentifikasi permasalahan dan analisa kebutuhan untuk sistem *microservice* untuk aplikasi RIFSA. Dari proses tersebut didapat sebagai berikut :

1. Identifikasi Masalah

Permasalahan arsitektur pada pembuatan *microservice* untuk aplikasi RIFSA adalah bagaimana merancang microservice yang dapat melayani pemrosesan model *machine learning* serta melayani proses autentikasi, authorisasi, dan manipulasi data pada database. Kemudian permasalahan selanjutnya adalah bagaimana service tersebut dapat saling berkomunikasi.

2. Analisis Kebutuhan

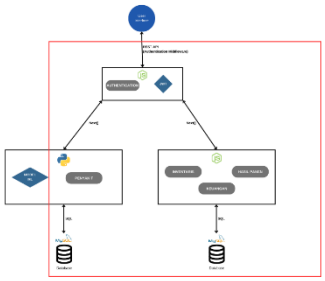
a. *Microservice* dapat memproses data masukan menggunakan model *machine learning.*
b. *Microservice* dapat melakukan proses autentikasi dan authorisasi pada setiap endpoint nya.
c. *Microservice* dapat saling berkomunikasi menggunakan *REST API.*
d. *Microservice* dapat melakukan manipulasi data pada database.

4.1.2. Design

Pada proses ini, dilakukan desain sistem menggunakan *Architecture diagram*, *usecase diagram*, serta desain *database* menggunakan ERD (*Entity Relationship Diagram*).

a. *Architecture Diagram*

Architecture Diagram menggambarkan desain arsitektur untuk aplikasi RIFSA. *Architecture Diagram* dapat dilihat pada gambar di bawah ini.



Gambar 2 Architecture Diagram Microservice RIFSA

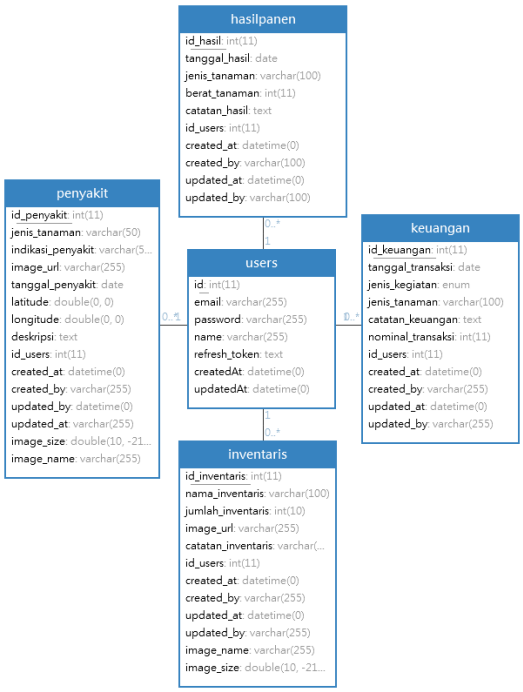
b. *Use case Diagram*

Use case diagram menggambarkan interaksi antar aktor yang terlibat dalam aplikasi RIFSA. *Use case diagram* dapat dilihat pada gambar di bawah ini.

Gambar 3 Use Case Diagram Aplikasi RIFSA

c. ERD (*Entity Relation Diagram*)

Tahap ini digunakan untuk membuat rancangan *database* dalam bentuk diagram seperti gambar di bawah ini.



Gambar 4 ERD (Entity Relation Diagram)

d. Desain Antarmuka

Desain Antarmuka untuk Aplikasi RIFSA dibagi menjadi beberapa bagian seperti halaman *Login*, halaman *Home*, halaman keuangan, halaman deteksi penyakit, halaman inventaris, dan halaman informasi profil *user*. Rancangan desain antarmuka ini dibuat berdasarkan *use case diagram* dan *entity relationship diagram* yang sudah dibuat sebelumnya yang diterapkan pada platform *mobile* android menggunakan bahasa Kotlin. Karena batasan penelitian ini mengacu pada *microservice*, maka tidak ada desain antarmuka yang dibuat oleh penulis.

e. *Data Dictionary*

Tabel 3 Data Dictionary Entity User

Atribut	Tipe Data	Ukuran	Keterangan
Id	Int	11	Id untuk user (*unique*)
Email	Varchar	255	Email pengguna
Password	Varchar	255	Password pengguna
Name	Varchar	255	Nama pengguna
Refresh_token	Text	0	Token untuk mendapatkan token baru
Created_at	Datetime	0	Waktu data pengguna dibuat
Updated_at	Datetime	0	Waktu data pengguna diubah

Tabel 4 Data Dictionary Entity Hasil Panen

Atribut	Tipe Data	Ukuran	Keterangan
Id_hasil	Int	11	Id (*unique*) untuk data hasil panen
Tanggal_hasil	Date	0	Tanggal panen dilakukan
Jenis_tanaman	Varchar	100	Jenis tanaman yang dipanen
Berat_tanaman	Int	11	Berat tanaman yang dipanen
Catatan_hasil	Text	0	Catatan untuk data hasil panen
Id_users	Int	11	Field relasi untuk table user
Created_at	Datetime	0	Waktu data hasil panen dibuat
Created_by	Varchar	100	User yang melakukan pembuatan data hasil panen
Updated_at	Datetime	0	Waktu data hasil panen diubah
Updated_by	Varchar	100	User yang melakukan perubah data hasil panen

Tabel 5 Data Dictionary Entity Inventaris

Atribut	Tipe Data	Ukuran	Keterangan
Id_inventaris	Int	11	Id (*unique*) untuk data inventaris
Nama_inventaris	Varchar	100	Nama barang inventaris

Jumlah_inventaris	Int	10	Jumlah barang inventaris
Image_url	Varchar	255	Url gambar yang disimpan
Catatan_inventaris	Text	0	Catatan untuk data inventaris
Id_users	Int	11	Field relasi untuk table user
Created_at	Datetime	0	Waktu data inventaris dibuat
Created_by	Varchar	100	User yang melakukan pembuatan data inventaris
Updated_at	Datetime	0	Waktu data inventaris diubah
Updated_by	Varchar	100	User yang melakukan perubah data inventaris
Image_name	Varchar	255	Nama gambar yang disimpan
Image_size	Double	10	Besar gambar yang disimpan

Tabel 6 Data Dictionary Entity Keuangan

Atribut	Tipe Data	Ukuran	Keterangan
Id_keuangan	Int	11	Id (*unique)* untuk data keuangan
Tanggal_transaksi	Date	0	Tanggal transaksi dilakukan
Jenis_kegiatan	Enum	0	Deskripsi jenis kegiatan (debit/kredit)
Jenis_tanaman	Varchar	100	Jenis tanaman yang terlibat transaksi
Catatan_keuangan	Text	0	Catatan untuk data keuangan
Nominal_transaksi	Int	64	Nominal transaksi yang dilakukan
User_id	Int	11	Field relasi untuk table user
Created_at	Datetime	0	Waktu data keuangan dibuat
Created_by	Varchar	100	User yang melakukan pembuatan data keuangan
Updated_at	Datetime	0	Waktu data keuangan diubah
Updated_by	Varchar	100	User yang melakukan perubah data keuangan

Tabel 7 Data Dictionary Entity Penyakit

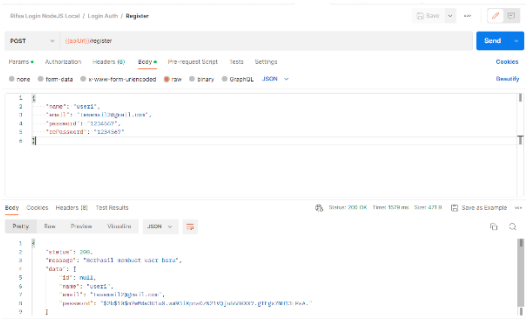
Atribut	Tipe Data	Ukuran	Keterangan
Id_penyakit	Int	11	Id (*unique)* untuk data inventaris
Jenis_tanaman	Varchar	50	Jenis tanaman yang dideteksi
Indikasi_penyakit	Varchar	50	Deskripsi indikasi penyakit tanaman
Image_url	Varchar	255	Url gambar yang disimpan
Tanggal_penyakit	Date	0	Tanggal penyakit dideteksi
Latitude	Double	0	Koordinat penyakit dideteksi
Longitude	Double	0	Koordinat penyakit dideteksi
Deskripsi	Text	0	Deskripsi penyakit
Id_users	Int	11	Field relasi untuk table user
Created_at	Datetime	0	Waktu data penyakit dibuat
Created_by	Varchar	100	User yang melakukan pembuatan data penyakit
Updated_at	Datetime	0	Waktu data penyakit diubah
Updated_by	Varchar	100	User yang melakukan perubah data penyakit
Image_name	Varchar	255	Nama gambar yang disimpan
Image_size	Double	10	Besar gambar yang disimpan

4.1.3. Coding

Pada tahap iterasi pertama, proses *coding* dilakukan untuk menggunakan *NodeJS* dengan *framework* ExpressJS. Dalam hal ini *source code* yang dibangun menggunakan konsep *REST API*. Pada tahap iterasi pertama, proses *coding* dilakukan untuk membangun fitur *register*, *login*, *logout*, CRUD hasil panen, CRUD keuangan, dan CRUD inventaris. *Source code* untuk fitur-fitur tersebut adalah sebagai berikut :

4.1.3.1. Fitur Authorization

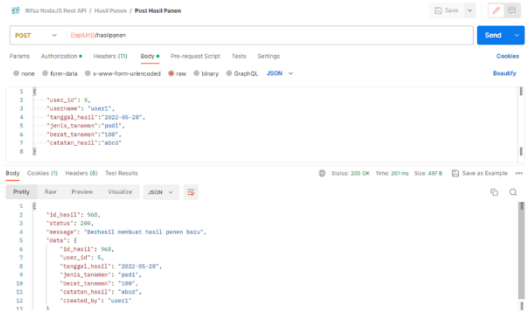
Fitur pertama pada proses *coding* di tahap iterasi pertama adalah fitur *Authorization*. Di dalamnya, terdapat beberapa fitur seperti *register*, *login*, dan *logout*.



Gambar 5 Hasil Fitur Authorization Pada Postman

4.1.3.2.Fitur CRUD Hasil Panen

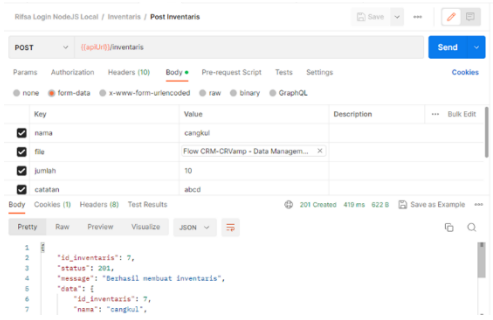
Fitur selanjutnya adalah fitur CRUD (*Create*, *Read*, *Update*, *Delete*) untuk data hasil panen. Fitur ini terbagi menjadi 4 sub fitur. Fitur *create* berfungsi untuk membuat data hasil panen baru, fitur *read* berfungsi untuk melihat/membaca data hasil panen, fitur *update* berfungsi untuk mengubah data hasil panen yang sudah ada, dan fitur *delete* berfungsi untuk menghapus sebuah data hasil panen.



Gambar 6 Hasil Fitur CRUD Hasil Panen Pada Postman

4.1.3.3.Fitur CRUD Inventaris

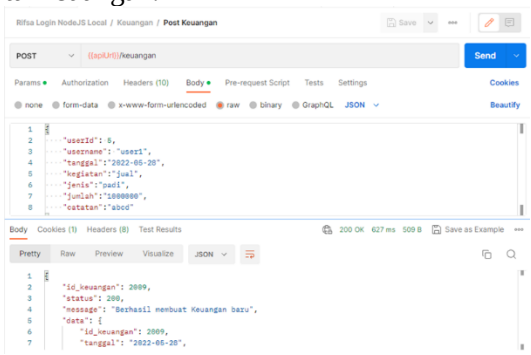
Fitur selanjutnya adalah fitur CRUD (*Create*, *Read*, *Update*, *Delete*) untuk data Inventaris. Fitur ini terbagi menjadi 4 sub fitur. Fitur *create* berfungsi untuk membuat data Inventaris baru, fitur *read* berfungsi untuk melihat/membaca data Inventaris, fitur *update* berfungsi untuk mengubah data Inventaris yang sudah ada, dan fitur *delete* berfungsi untuk menghapus sebuah data Inventaris.



Gambar 7 Hasil Fitur CRUD Inventaris Pada Postman

4.1.3.4. Fitur CRUD Keuangan

Fitur selanjutnya adalah fitur CRUD (*Create*, *Read*, *Update*, *Delete*) untuk data Keuangan. Fitur ini terbagi menjadi 4 sub fitur. Fitur *create* berfungsi untuk membuat data Keuangan baru, fitur *read* berfungsi untuk melihat/membaca data Keuangan, fitur *update* berfungsi untuk mengubah data Keuangan yang sudah ada, dan fitur *delete* berfungsi untuk menghapus sebuah data Keuangan.



Gambar 8 Hasil Fitur CRUD Keuangan Pada Postman

4.1.3.5. Daftar Endpoint

Setelah dilakukan proses *coding*, didapatkan sejumlah endpoint yang dapat digunakan pada *service* pertama menggunakan *NodeJS*. Detail setiap endpoint dapat dilihat pada tabel di bawah ini :

Tabel 8 Daftar Endpoint pada Service NodeJS

No	Method	Route	Fungsi
1	POST	/login	Masuk ke aplikasi dan mendapatkan token
2	POST	/register	Mendaftarkan akun pengguna ke aplikasi
3	DELETE	/logout	Keluar dari aplikasi

4	POST	/hasilpanen	Menambahkan data hasil panen
5	GET	/hasilpanen?user_id=	Mendapatkan semua data hasil panen yang sudah dibuat oleh pengguna
6	GET	/hasilpanen?user_id=&id_hasil=	Mendapatkan data hasil panen berdasarkan id data hasil panen
7	PUT	/hasilpanen?id_hasil=	Memodifikasi data hasil panen
8	DELETE	/hasilpanen?id_hasil=	Menghapus data hasil panen
9	POST	/inventaris	Menambahkan data inventaris
10	GET	/inventaris?user_id=	Mendapatkan semua data inventaris yang sudah dibuat oleh pengguna
11	GET	/inventaris?user_id=&id_inventaris=	Mendapatkan data inventaris berdasarkan id data inventaris
12	PUT	/inventaris?id_inventaris=	Memodifikasi data inventaris
13	DELETE	/inventaris?id_inventaris=	Menghapus data inventaris
14	POST	/keuangan	Menambahkan data keuangan
15	GET	/keuangan?user_id=	Mendapatkan semua data keuangan yang sudah dibuat oleh pengguna
16	GET	/keuangan?user_id=&id_keuangan=	Mendapatkan data keuangan berdasarkan id data keuangan
17	PUT	/keuangan?id_keuangan=	Memodifikasi data keuangan
18	DELETE	/keuangan?id_keuangan=	Menghapus data keuangan

4.1.4. Testing

Proses selanjutnya adalah *testing*. Pada proses ini akan menampilkan *Performance test* dengan parameter *load* dan *response time* dari masing-masing fitur.

4.1.4.1. Fitur Authorization

Gambar 9 Performance Test Fitur Authorization

4.1.4.2. Fitur Hasil Panen

Gambar 10 Performance Test Fitur Hasil Panen

4.1.4.3. Fitur Inventaris

Gambar 11 Performance Test Fitur Inventaris

4.1.4.4. Fitur Keuangan

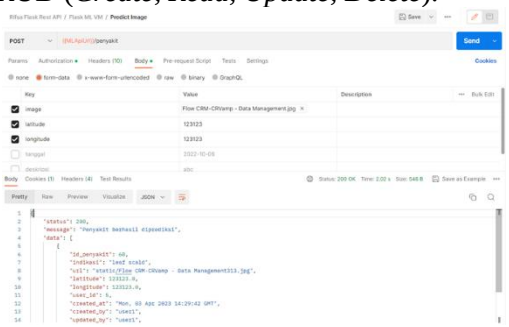
Gambar 12 Performance Test Fitur Keuangan

4.2. Iterasi 2

Iterasi ke 2 ditujukan untuk pembuatan *REST API* menggunakan *Python* dengan *framework* flask. Tahapan proses iterasi 2 hanya terdiri dari *coding* dan *testing*, berbeda dari iterasi 1 yang terdiri dari tahap *planning*, *design*, *coding*, dan *testing*. Hal ini karena tahap *planning* dan *design* sudah dilakukan pada iterasi 1 dan pada iterasi 2 menggunakan kaidah *planning* dan *design* yang sama dengan iterasi 1. Rincian proses *coding* dan *testing* pada iterasi 2 adalah sebagai berikut :

4.2.1. Coding

Proses *coding* pada iterasi 2 dilakukan menggunakan *Python* dengan *framework* Flask. *Source code* dibangun menggunakan konsep *REST API*. Pada tahap iterasi 2, proses *coding* dilakukan hanya untuk 1 fitur utama yaitu penyakit, dimana didalamnya terdapat fitur CRUD (*Create*, *Read*, *Update*, *Delete*).



Gambar 13 Hasil Fitur CRUD Penyakit Pada Postman

4.2.1.1. Daftar Endpoint

Setelah dilakukan proses *coding*, didapatkan sejumlah endpoint yang dapat digunakan pada *service* kedua menggunakan *Python*. Detail setiap endpoint dapat dilihat pada tabel di bawah ini :

Tabel 9 Daftar Endpoint pada Service Python

No	Method	Route	Fungsi
1	POST	/penyakit	Mendeteksi penyakit
2	GET	/penyakit	Mendapatkan semua data penyakit yang sudah dibuat oleh pengguna
3	GET	/penyakit/{id_penyakit}	Mendapatkan data penyakit berdasarkan id data penyakit
4	PUT	/penyakit/{id_penyakit}	Memodifikasi hasil pendeteksian penyakit
5	DELETE	/penyakit/{id_penyakit}	Menghapus data penyakit

4.2.2. Testing

Proses selanjutnya adalah *testing*. Pada proses ini akan menampilkan *Performance test* dengan parameter *load* dan *response time* dari keseluruhan fitur.

Gambar 14 Performance Test Fitur Penyakit

5. KESIMPULAN

Kesimpulan yang didapatkan pada penelitian ini sebagai berikut :

1. *Microservice* berbasis *REST API* telah berhasil dibuat menggunakan *NodeJS* dan *Python* dengan fitur yaitu *Authentication*, Hasil Panen, Inventaris, Keuangan, dan Penyakit yang sudah sesuai dengan kaidah *RESTful Handbook*

2. Berdasarkan hasil *Performance test*, dari keseluruhan hasil tes didapat bahwa server mampu menerima rata-rata 40,07 hits/detik dan rata-rata response time sebesar 60,41 ms

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada seluruh pihak yang telah membantu dalam pengerjaan jurnal baik dalam pengembangan sistem maupun penulisan penelitian ini.

DAFTAR PUSTAKA

[1] Fatmawati M. Lumintang, “Analisis Pendapatan Petani Padi Di Desa Teep Kecamatan Langowan Timur,” *Jurnal Emba*, Vol. 1, No. 3, Pp. 991–998, 2013.

[2] C. V Donggulo, I. M. Lapanjang, And U. Made, “Pertumbuhan Dan Hasil Tanaman Padi (Oryza Sativa L) Pada Berbagai Pola Jajar Legowo Dan Jarak Tanam Growth And Yield Of Rice (Oryza Sativa L.) Under Different Jajar Legowo System And Planting Space,” Palu, Apr. 2017.

[3] B. Nuryanto, “Pengendalian Penyakit Tanaman Padi Berwawasan Lingkungan Melalui Pengelolaan Komponen Epidemik,” *Jurnal Penelitian Dan Pengembangan Pertanian*, Vol. 37, No. 1, P. 1, May 2018, Doi: 10.21082/Jp3.V37n1.2018.P1-8.

[4] H. Purnama And I. Y. B, “Aplikasi Pengelolaan Skripsi Di Stmik Akakom Yogyakarta Menggunakan Arsitektur Microservice Dengan Node.Js Heri Purnama 1) Indra Yatini B 2) A Bstract,” 2016. [Online]. Available: Http://Perpus.Akakom.Ac.Id/

[5] A. Qalam, J. Ilmiah Keagamaan Dan Kemasyarakatan, And Z. Riko Virgiawan,

"Perancangan Arsitektur Backend Microservice Pada Startup Campaign.Com," *Jurnal Ilmiah Keagamaan Dan Kemasyarakatan*, Vol. 16, No. 1, 2022, Doi: 10.35931/Aq.V16i1.

[6] J. A. Falaq, R. Tulloh, And M. Iqbal, "Implementasi Jaringan Hotspot Berbayar Berbasis Voucher Menggunakan Platform Google Cloud Implementation Of A Paid Hotspot Network Based On Vouchers Using The Google Cloud Platform," 2021.

[7] M. Iqbal C. R., "Implementasi Klien Sip Berbasis Web Menggunakan Html5 Dan Node.Js," 2012.

[8] S. Nasional And R. X. F. Ums, "Deteksi Wajah Metode Viola Jones Pada Opencv Menggunakan Pemrograman Python," 2012.

[9] A. Rulloh, "Implementasi Rest Api Pada Aplikasi Panduan Kepaskibraan Berbasis Android," *Teknikom*, Vol. 1, No. 2, Pp. 2598–2958, 2017.

[10] J. Homepage, A. Roihan, P. Abas Sunarya, And A. S. Rafika, "Ijcit (Indonesian Journal On Computer And Information Technology) Pemanfaatan Machine Learning Dalam Berbagai Bidang: Review Paper," 2019.

[11] A. Supriyatna, "Metode Extreme Programming Pada Pembangunan Web Aplikasi Seleksi Peserta Pelatihan Kerja," *Jurnal Teknik Informatika*, Vol. 11, No. 1, Pp. 1–18, May 2018, Doi: 10.15408/Jti.V11i1.6628.

[12] Y. Dwi Wijaya And M. Wardah Astuti, "Pengujian Blackbox Sistem Informasi Penilaian Kinerja Karyawan Pt Inka (Persero) Berbasis Equivalence Partitions Blackbox Testing Of Pt Inka (Persero) Employee Performance Assessment Information System Based On Equivalence Partitions," *Jurnal Digital Teknologi Informasi*, Vol. 4, P. 2021, 2021.

[13] N. Luh, A. S. Ginasari, K. S. Wibawa, N. Kadek, And A. Wirdiani, "Pengujian Stress Testing Api Sistem Pelayanan Dengan Apache Jmeter," 2021.