PERANCANGAN *MICROSERVICE* BERBASIS *REST API* PADA *GOOGLE CLOUD PLATFORM* MENGGUNAKAN *NODEJS* DAN *PYTHON* (STUDI KASUS: APLIKASI PENDETEKSI PENYAKIT DAUN PADI)

(Skripsi)

Oleh

ROYYAN FAJRUL FALAH
NPM 1915061040

FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2023

PERANCANGAN *MICROSERVICE* BERBASIS *REST API* PADA *GOOGLE CLOUD PLATFORM* MENGGUNAKAN *NODEJS* DAN *PYTHON* (STUDI KASUS: APLIKASI PENDETEKSI PENYAKIT DAUN PADI)

Oleh

ROYYAN FAJRUL FALAH

Sebagai Salah Satu Syarat untuk Mencapai Gelar

SARJANA TEKNIK

Pada

Jurusan Teknik Elektro

Fakultas Teknik Universitas Lampung

FAKULTAS TEKNIK

UNIVERSITAS LAMPUNG

BANDAR LAMPUNG

2023

ABSTRAK

PERANCANGAN *MICROSERVICE* BERBASIS *REST API* PADA *GOOGLE CLOUD PLATFORM* MENGGUNAKAN *NODEJS* DAN *PYTHON* (STUDI KASUS: APLIKASI PENDETEKSI PENYAKIT DAUN PADI)

Oleh

ROYYAN FAJRUL FALAH

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.

Kata Kunci : Penyakit daun padi, *Microservice*, *REST API*, *Google Cloud Platform*, *NodeJS*, *Python*

ABSTRACT

BUILDING REST API BASED MICROSERVICE ON GOOGLE CLOUD PLATFORM USING NODEJS AND PYTHON (STUDY CASE : RICE LEAF DISEASE DETECTION APP)

By

ROYYAN FAJRUL FALAH

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.

Keywords : *Rice leaf disease*, *Microservice*, *REST API*, *Google Cloud Platform*, *NodeJS*, *Python*

Judul Skripsi	:	**PERANCANGAN *MICROSERVICE* BERBASIS *REST API* PADA *GOOGLE CLOUD PLATFORM* MENGGUNAKAN *NODEJS* DAN *PYTHON* (STUDI KASUS: APLIKASI PENDETEKSI PENYAKIT DAUN PADI)**
Nama Mahasiswa	:	Royyan Fajrul Falah
Nomor Pokok Mahasiswa	:	1915061040
Program Studi	:	S1 Teknik Informatika
Jurusan	:	Teknik Elektro
Fakultas	:	Teknik

MENYETUJUI

1. Komisi Pembimbing

Pembimbing Utama	Pembimbing Utama
Muhamad Komarudin, S.T., M.T.	**Mahendra Pratama, S.T., M.Eng.**
NIP 19681207 199703 1 006	NIP 19911215 201903 1 013

2. Mengetahui

Ketua Jurusan Teknik Elektro	Ketua Program Studi Teknik Informatika
Herlinawati, S.T., M.T.	**Mona Arif Muda, S.T., M.T.**
NIP 19710314 199903 2 001	NIP 19711112 200003 1 002

MENGESAHKAN

1. Tim Penguji

Ketua : **Muhamad Komarudin, S.T., M.T.** .....

Sekretaris : **Mahendra Pratama, S.T., M.Eng.** .....

Penguji : **Ir. Meizano Ardhi Muhammad, S.T., M.T.** .....

2. Dekan Fakultas Teknik

Dr. Eng. Ir. Helmy Fitriawan, S.T., M.Sc.
NIP 19750928 200112 1 002

Tanggal Lulus Ujian Skripsi : **9 Juni 2023**

SURAT PERNYATAAN

Puji syukur saya panjatkan kehadirat Allah SWT yang atas rahmat dan karunia-Nya, saya dapat menyelesaikan skripsi berjudul "PERANCANGAN *MICROSERVICE* BERBASIS *REST API* PADA *GOOGLE CLOUD PLATFORM* MENGGUNAKAN *NODEJS* DAN *PYTHON* (STUDI KASUS: APLIKASI PENDETEKSI PENYAKIT DAUN PADI)". Sebagai salah satu syarat untuk mendapatkan gelar Sarjana Teknik di Fakultas Teknik Universitas Lampung.

Terima kasih saya ucapkan kepada rekan-rekan, bapak dan ibu dosen yang telah mendukung selama pengerjaan Skripsi ini sehingga saya dapat menyelesaikannya.

Dalam penyusunan Skripsi ini, saya menyadari sepenuhnya bahwa masih banyak kekurangan dalam penyajian penulisan yang perlu diperbaiki. Oleh karena itu, saran dan kritik akan sangat berarti bagi saya. Saya juga berharap, Skripsi ini akan berguna dan bermanfaat sebagai bentuk referensi ataupun penambahan wawasan.

Bandar Lampung, 2023

Royyan Fajrul Falah

RIWAYAT HIDUP

Penulis dilahirkan di Bandar Lampung, pada tanggal 13 Desember 2001. Penulis merupakan anak pertama dari pasangan Bapak Ir. Mohamad Zamil dan Ibu Ir. Uriptin Rahayu.

Penulis menyelesaikan pendidikannya di SD Alam Lampung pada tahun 2013, SMP Alam Lampung pada tahun 2016, dan SMA Negeri 2 Bandar Lampung pada tahun 2019. Pada tahun 2019 penulis terdaftar sebagai mahasiswa Program Studi Teknik Informatika, Jurusan Teknik Elektro, Fakultas Teknik Universitas Lampung melalui jalur SBMPTN. Selama menjadi mahasiswa, penulis melakukan beberapa kegiatan, antara lain:

1. Menjadi anggota biasa Himpunan Mahasiswa Teknik Elektro Universitas Lampung, Departemen Komunikasi dan Informasi, Divisi Media Informasi pada tahun 2020.
2. Menjadi Asisten Laboratorium Teknik Komputer Jurusan Teknik Elektro Universitas Lampung pada tahun 2021 hingga 2023.
3. Mengikuti *bootcamp* yang diadakan oleh GRADIEN mengenai *Website Development* pada tahun 2021
4. Menjadi Kepala Departemen Komunikasi dan Informasi di Himpunan Mahasiswa Teknik Elektro Universitas Lampung pada tahun 2021
5. Mengikuti program Magang Kampus Merdeka sebagai *Backend Engineer Intern* di PT Kalbe E-Health (KlikDokter) pada tahun 2021.
6. Mengikuti program Studi Independen Bersertifikat Kampus Merdeka dari Kementerian Pendidikan dan Budaya di Bangkit Academy 2022 mengambil jalur *Cloud Computing* pada tahun 2022.

7. Melaksanakan Kuliah Kerja Nyata di Desa Banding Agung, Kecamatan Talang Padang, Kabupaten Tanggamus, Provinsi Lampung pada bulan Juli sampai Agustus 2023.

MOTTO

"*Keep striving. Eventually, it will come to an end.*"

(Penulis)

"Allah akan meninggikan orang-orang yang beriman diantaramu dan orang-orang yang diberi ilmu pengetahuan beberapa derajat."

(Q.S. Al-Mujadalah : 11)

"Apapun yang menjadi takdirmu, akan mencari jalannya menemukanmu."

(Ali Bin Abi Thalib)

"*Yesterday is history, tomorrow is mistery, but today is a gift. That is why it's called present.*"

(Master Oogway)

"Accept the things to which fate binds you, and love the people with whom fate brings you together, but do so with all your heart."

(Marcus Aurelius)

PERSEMBAHAN

Bismillaahirrohmaanirrahiim,
Dengan mengharapkan ridho dari Allah SWT,
Kupersembahkan karyaku ini untuk orang-orang yang kusayangi
dengan setulus hati:
Orangtua tercinta,
Keluargaku,
Teman-Temanku,
Dan
Orang-orang yang telah membantu hidupku

Terimakasih untuk segalanya,
Kalian adalah hartaku yang paling berharga.

SANWACANA

Puji syukur kehadirat Allah SWT yang telah melimpahkan rahmat dan hidayat-Nya sehingga penulis dapat menyelesaikan penyusunan skripsi/tugas akhir ini dengan judul "Perancangan *Microservice* Berbasis *Rest Api* Pada *Google Cloud Platform* Menggunakan *Nodejs* Dan *Python* (Studi Kasus: Aplikasi Pendeteksi Penyakit Daun Padi)". Dalam pelaksanaan dan pembuatan skripsi/tugas Akhir ini penulis menerima dukungan baik secara moril maupun materil yang sangat berharga dari berbagai pihak. Oleh karena itu, penulis ingin mengucapkan terima kasih kepada semua pihak yang telah membantu, khususnya kepada:

1. Kedua orangtua tercinta dan seluruh keluarga penulis yang tidak hentinya mendo'akan serta memberikan dorongan semangat dan materi;
2. Bapak Dr. Eng. Helmy Fitriawan, S.T., M.Sc., selaku Dekan Fakultas Teknik Universitas Lampung;
3. Ibu Herlinawati, S.T., M.T. selaku Ketua Jurusan Teknik Elektro Universitas Lampung;
4. Bapak Mona Arif Muda, S.T.,M.T. selaku Ketua Program Studi Teknik Informatika Universitas Lampung dan telah membantu proses kelancaran pengerjaan penelitian;
5. Bapak Muhamad Komarudin, S.T., M.T. selaku Pembimbing Utama dan Pembimbing Akademik yang selalu meluangkan waktunya untuk memberikan bimbingan dan dukungan serta memudahkan penulis dalam menyelesaikan penelitian ini;
6. Bapak Mahendra Pratama, S.T., M.Eng. selaku Pembimbing Pendamping yang selalu memberikan dukungan serta bimbingan agar menjadi lebih baik;
7. Bapak Ir. Meizano Ardhi Muhammad, S.T., M.T. selaku Penguji yang telah memberikan banyak saran dan masukan terhadap penelitian ini;

8. Mbak Rika selaku Admin Program Studi Teknik Informatika yang telah banyak membantu penulis dalam segala urusan administrasi selama perkuliahan;
9. Seluruh dosen dan staf Jurusan Teknik Informatika Unila yang memberi masukan dan mempermudah proses pembuatan skripsi / tugas akhir ini.
10. Sahabat – sahabat grup Laut Nabila, Presil, Irfan, Tania, Ais, Rachel, Naufal, Iqbal, Sanjaya, Rama, Agung dan Surya yang sangat penulis sayangi. Terima kasih sudah menjadi sahabat terbaik selama awal kuliah sampai saat ini.
11. Alfiyah Widyaningsih, partner penulis yang penulis sayangi. Terima kasih telah menemani proses penulisan skripsi ini hingga akhir.
12. Angkatan ETERNITY 2019 yang telah menjadi teman seperjuangan sejak mahasiswa baru. Terimakasih telah mewarnai masa perkuliahan penulis dan menulis banyak cerita bersama.
13. Seluruh teman-teman yang terlibat langsung maupun tidak langsung dalam penyelesaian skripsi yang tidak dapat penulis sebutkan satu persatu.

Penulis berharap agar laporan ini dapat menjadi referensi bagi pengembangan keilmuan di bidang teknik informatika. Oleh karena itu, semoga penelitian ini bermanfaat bagi yang membacanya.

Bandar Lampung, 30 Juli 2023

Penulis,

Royyan Fajrul Falah

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

I. PENDAHULUAN

1.1. Latar Belakang

Indonesia adalah negara dengan jenis pertanian tropika dimana sebagian besar daerahnya adalah daerah tropis dan dipengaruhi langsung oleh garis khatulistiwa yang memotong negara ini menjadi dua. 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]

Di Indonesia, penyakit penting pada daun padi ialah hawar daun bakteri (*Xanthomonas campestris pv. oryzae*), penyakit *tungro* (virus *tungro*), bercak daun *pyricularia* (*Pyricularia grisea*), dan hawar pelepah daun (*Rhizoctonia solani Kuhn*) [3]. Penyakit-penyakit tersebut sangat berpengaruh terhadap hasil panen dan kualitas panen dari komoditas padi. Para petani adalah pihak yang paling dirugikan dengan hadirnya penyakit daun padi tersebut. Oleh karena itu, perlu adanya usaha penanggulangan penyakit supaya hasil panen dari para petani padi tetap berkualitas. Para petani biasanya mencari solusi atas penyakit pada tanaman mereka pada tim dinas petanian setempat. Namun, karena keterbatasan dari tim dinas pertanian, seperti keterbatasan waktu, biaya, dan keilmuan maka penanggulangan penyakit pada daun padi perlu alternatif penanganan lain. [4]

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.

Bangkit Academy 2022 diadakan oleh Direktorat Jenderal Pendidikan Tinggi (Ditjen Dikti) yang bekerjasama dengan Google, GoTo, dan Traveloka berdasarkan surat edaran resmi Kemdikbud RI pada 11 Desember 2020. Terdapat 3 *learning path* pada Bangkit Academy 2022 yaitu *Machine learning, Android Developer,* dan *Cloud Computing.* Pada akhir program Bangkit Academy 2022 diadakan *capstone project*, dimana siswa terdiri dari 6 orang dengan 3 *learning path* berbeda. Pada *capstone project* aplikasi RIFSA, anggota terdiri dari 2 orang dari masing-masing *learning path* dengan pembagian sebagai berikut: pembuatan model *machine learning* dilakukan oleh Ade Fiqri Tjikoa (M2269J2320) dan Dede Ikhsan Dwi Saputra (M2296F2520), pembuatan UI dan aplikasi berbasis *android* dilakukan oleh Dio Farrel P.R (A2296F2519) dan I Wayan Alston A. (A2296F2521), kemudian pembangunan *backend* dan *deployment* dilakukan oleh M. Rafie Azmi (C2004F0405) dan Royyan Fajrul Falah (C2248F2199).

Microservice adalah sebuah *framework* arsitektur rekayasa perangkat lunak yang saat ini telah populer digunakan oleh para developer di seluruh dunia. Setiap layanan yang berjalan didalam *microservice* berjalan pada prosesnya masing-masing. Setiap layanan tersebut berkomunikasi melalui API (*application programming interface)* [5]. 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.

1.2. Rumusan Masalah

Berdasarkan latar belakang, rumusan masalah untuk penelitian ini adalah bagaimana membuat sebuah *microservice* untuk aplikasi RIFSA menggunakan *NodeJS* dan *Python* supaya menambah efektivitas aplikasi RIFSA dengan menggunakan metode *Extreme Programming*.

1.3. Batasan Masalah

Batasan masalah dalam penelitian ini adalah :

1. *Output* dari penelitian ini adalah *response* dari *microservice* dalam format JSON. Penelitian ini tidak menyediakan output berupa *user interface.*
2. Fitur yang akan dibuat menjadi *microservice* adalah fitur *authentication* hasil panen, inventaris, keuangan, dan penyakit dalam bentuk CRUD (*Create*, *Read*, *Update*, *Delete*)
3. *Resource* untuk *testing* yang digunakan pada BlazeMeter adalah *free tier* sehingga membatasi hasil *testing*
4. *Pada* fitur deteksi penyakit, 1 request terbatas hanya dapat menerima 1 gambar pada satu waktu

1.4. Tujuan Penelitian

Tujuan penelitian ini adalah membuat arsitektur *microservice* untuk fitur yang ada pada aplikasi RIFSA dalam bentuk *REST API* menggunakan *NodeJS* dan *Python*.

1.5. Manfaat Penelitian

Penelitian ini diharapkan dapat membermudah proses *development* dari aplikasi RIFSA dan mempermudah *developer* dalam melakukan *scale up* aplikasi serta menambah efektivitas dari aplikasi RIFSA. Aplikasi RIFSA sendiri berguna untuk mempermudah petani padi dalam mengelola dan menjaga kualitas tanaman padi nya dengan menyediakan fitur untuk mengelola hasil panen, keuangan, inventaris, serta mendeteksi penyakit daun padi serta cara penanggulangannya

1.6. Sistematika Penulisan Laporan

Sistematika penulisan yang digunakan pada skripsi ini adalah sebagai berikut:

BAB I : PENDAHULUAN

Bab ini berisi latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian dan sistematika penulisan laporan.

BAB II : TINJAUAN PUSTAKA

Pada bab ini membahas mengenai dasar teori yang digunakan sebagai sumber dalam memahami permasalah dalam melakukan penelitian mengenai PERANCANGAN *MICROSERVICE* BERBASIS *REST API* PADA *GOOGLE CLOUD PLATFORM* MENGGUNAKAN *NODEJS* DAN *PYTHON* (STUDI KASUS: APLIKASI PENDETEKSI PENYAKIT DAUN PADI).

BAB III : METODOLOGI PENELITIAN

Pada bab ini membahas mengenai metodologi penelitian yang digunakan dalam PERANCANGAN *MICROSERVICE* BERBASIS *REST API* PADA *GOOGLE CLOUD PLATFORM* MENGGUNAKAN *NODEJS* DAN *PYTHON* (STUDI KASUS: APLIKASI PENDETEKSI PENYAKIT DAUN PADI).

BAB IV : HASIL DAN PEMBAHASAN

Pada bab ini membahas mengenai hasil serta pembahasan yang diperoleh dalam penelitian PERANCANGAN *MICROSERVICE* BERBASIS *REST API* PADA *GOOGLE CLOUD PLATFORM* MENGGUNAKAN *NODEJS* DAN *PYTHON* (STUDI KASUS: APLIKASI PENDETEKSI PENYAKIT DAUN PADI).

BAB V : SIMPULAN DAN SARAN

Pada bab ini berisi tentang kesimpulan dari hasil penelitian yang telah dilakukan serta saran-saran sebagai masukan untuk penelitian lebih lanjut di masa mendatang.

DAFTAR PUSTAKA

LAMPIRAN

II. 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*). Di akhir program, setiap peserta akan bekerja dengan peserta dari jalur pembelajaran yang berbeda untuk membuat proyek akhir dengan tim yang terdiri dari 6 orang. *Capstone Project* Bangkit dibagi menjadi 7 tema dan agenda pembangunan, seperti :

1. *Economic & Social Resilience*
2. *Character-Building & Community Empowerment*
3. *Environtmental Conservation, Disaster Resilience and Climate Change*
4. *Healthcare*
5. *Mobilty & Smart City*
6. *Tourism, Creative, and Digital Economy*
7. *Education, Training, Upskilling & Research*
8. *Women Empowerment and Child Protection*

Pada pengerjaan proyek akhir Bangkit atau *Capstone project* Bangkit, setiap tim *Capstone* terdiri dari 6 orang dengan 3 *learning path* berbeda (*Cloud computing*, *Machine learning*, *Mobile development*) yang berisikan masing-masing 2 orang

dari setiap *learning path.* ID untuk tim *capstone project* yang didapat adalah C22-PS081 dengan anggota sebagai berikut :

1. Tjikoa, Ade Fiqri (M2269J2320) - *Machine learning* - Universitas Mulawarman
2. Dede Ikhsan Dwi Saputra (M2296F2520) - *Machine learning* - Universitas Pembangunan Nasional Veteran Jawa Timur
3. I Wayan Alston Argodi (A2296F2521) - *Mobile Development* - Universitas Pembangunan Nasional Veteran Jawa Timur
4. Dio Farrel Putra Rachmawan (A2296F2519) - *Mobile Development* - Universitas Pembangunan Nasional Veteran Jawa Timur
5. Muhammad Rafie Azmi (C2004F0405) - *Cloud Computing* - Institut Teknologi Sepuluh Nopember
6. Royyan Fajrul Falah (C2248F2199) - *Cloud Computing* - Universitas Lampung

Tim C22-PS081 memilih tema *Economic & Social Resilience (including agricultural & food security and sustainability, infrastructure, & regional development).* Sebelum menentukan topik dan nama aplikasi, tim berdiskusi mengenai permasalahan yang akan dibahas. Penentuan tema dipilih berdasarkan riset dan diskusi oleh seluruh anggota tim. Berdasarkan kesepakatan dan hasil diskusi, maka ditentukan nama *Capstone Project* untuk tim C22-PS081 adalah RIFSA (*Rice Farmer Assistant*). Tujuan dibuatnya aplikasi RIFSA adalah untuk membantu petani padi untuk mengelola pertanian padi mereka. Dengan adanya fitur pengelolaan hasil panen, inventaris, keuangan, serta fitur deteksi penyakit daun padi diharapkan dapat mempermudah petani padi di indonesia untuk mengelola lahan padi mereka serta dapat mendeteksi penyakit yang menjangkit tanaman padi mereka dan mengetahui cara penanggulangan nya secara cepat.

Proses produksi *Capstone Project* untuk aplikasi RIFSA dilaksanakan selama 4 minggu. Proses komunikasi tim RIFSA dilakukan melalui grup Whatsapp, Discord, Google Meet, dll. Pertemuan Google Meet dilakukan setiap hari pada jam tertentu untuk memantau perkembangan dari setiap anggota tim. Untuk memantau dan

membagi tugas lebih jelas antar tim, maka dibuatlah *Project Scope, Timeline*, dan *Deliverables.* Berikut ini adalah *Project Scope, Timeline*, dan *Deliverables* untuk tim RIFSA :

Tabel 2.1 *Project Scope, Timeline*, dan *Deliverables* untuk tim RIFSA Week 1

***Week* 1**		
Machine learning	***Mobile Development***	***Cloud Computing***
• *Collecting images of rice leaves on the internet (day 1-2)* • *Validating the images (day 3-4)* • *Preprocessing images (day 3-4)* • *Finding the optimal transfer learning model for our model (day 5-6)* • *Create the model with transfer learning using Python Tensorflow (day 7-12)*	• *Making User Interface design from wireframe until HI-FI design (day 1 - 4)* • *Check if the user interface is clear (day 1 - 4)* • *Translate final design to XML code (day 4 - 8)*	• *Designing the service to be used (Day 1 - Day 2)* • *Choosing google cloud storage and data services (Day 3)* • *Building cloud firestore infrastructure (Day 4 - Day 7)* • *Building and setting cloud storage to GCP (Day 4 - Day 7)*

Tabel 2.2 *Project Scope, Timeline*, dan *Deliverables* untuk tim RIFSA Week 2

Week 2		
Machine learning	***Mobile Development***	***Cloud Computing***
• *Create the model with transfer learning using Python Tensorflow (day 7-12)* • *Evaluate and test the model to achieve the desired accuracy with development dataset (day 10-12)* • *Deploying to android application (day 12-14)*	• *Translate final design to XML code (day 4 - 8)* • *Implementing functional code and back end services (day 10 -14)* • *Testing functional code and Record feature(day 10 - 14)*	• *Creating RESTful API with NodeJS for data retrieving (Day 8 - Day 14)* • *Deploying Application Programming Interface on postman (Day 14 - Day 17)*

Tabel 2.3 *Project Scope, Timeline*, dan *Deliverables* untuk tim RIFSA Week 3

Week 3		
Machine learning	***Mobile Development***	***Cloud Computing***
• *Test and evaluate the ML model that has already been deployed on android (day 15-19)*	• *Implementing Plant diseases detection and recording diseases that have been detected (day 15 - 19)* • *Testing the machine learning feature (day 15 -19)*	• *Deploying Application Programming Interface on postman (Day 14 - Day 17)* • *Optimizing devices (Day 18 - Day 21)*

Tabel 2.4 *Project Scope, Timeline*, dan *Deliverables* untuk tim RIFSA Week 4

Week 4		
Machine learning	***Mobile Development***	***Cloud Computing***
• *Fix the model if the accuracy and speed that has been deployed on android does not reach the expected target (day 19-30)*	• *Testing the application and make sure all feature working as we expected (day 19 - 30)*	• *Maintenance front-end & back-end issues (Day 22 - Day 28)*

Proses pembuatan RIFSA dimulai pada tahap identifikasi masalah dan analisis kebutuhan pengguna, dengan rincian sebagai berikut :

1. Identifikasi Masalah

Permasalahan utama yang melatarbelakangi pembuatan aplikasi RIFSA adalah karena adanya penyakit pada daun padi yang menyebabkan menurunnya hasil panen petani padi serta kurangnya sistem yang dapat membantu manajemen hasil panen, inventaris, dan keuangan dalam sektor pertanian padi.

2. Analisis Kebutuhan
    a. Petani dapat mendaftar menggunakan *email*
    b. Petani dapat melakukan *login* menggunakan *email* dan *password*
    c. Petani dapat menyimpan, melihat, mengubah, dan menghapus pendataan hasil panen
    d. Petani dapat menyimpan, melihat, mengubah, dan menghapus pendataan keuangan
    e. Petani dapat menyimpan, melihat, mengubah, dan menghapus pendataan inventaris
    f. Petani dapat mendeteksi penyakit pada daun padi dan menyimpan datanya pada database

g. Petani dapat melihat, mengubah, dan menghapus data penyakit

2.1 *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. Beberapa kelebihan *Microservice* adalah pembuatan kode aplikasi yang lebih sedikit dan mandiri, perangkat lunak yang mudah pemeliharaannya, mudah diperluas, dan memudahkan pengembang menggunakan setiap bahasa pemrograman dan *framework*. [6]

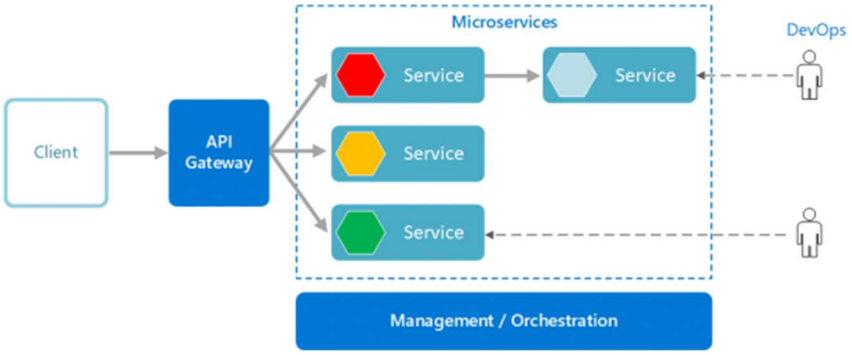
Beberapa manfaat menggunakan *microservice* adalah :

a. *Technology Heterogenity*
   Melalui sistem yang terdiri dari beberapa layanan, pengembang dapat menentukan penggunaan teknologi dari setiap layanan. Hal ini memungkinkan pengembang untuk memilih teknologi yang tepat untuk setiap layanan, dari pada menggunakan teknologi berperforma rendah yang berfungsi untuk semua orang. *Microservice* juga memungkinkan pengebang untuk mengadopsi teknologi baru dengan cepat dengan resiko yang minimal.
b. *Resilient*
   Penggunaan arsitektur *microservice* menjadikan aplikasi lebih tangguh, mencegah terjadinya kesalahan. Kerusakan dapat mudah terdeteksi dan hanya dialokasikan ke layanan tertentu. Dibandingkan dengan aplikasi monolitik biasanya kesalahan yang terjadi mengarah ke seluruh aplikasi untuk berhenti bekerja.
c. *Scaling*
   Ketika beban *server* melebihi kapasitas, kinerja *server* membutuhkan proses scalling agar kinerja *server* tetap terjaga. Dalam aplikasi monolitik scalling hanya men *deploy* beberapa service saja. Hal ini menjadikan proses *scaling*

lebih efisien karena bisa berjalan pada perangkat yang lebih kecil. Dengan karakteristik ini, kelebihannya adalah :

a) Lebih sedikit kode dan bersifat mandiri jadi dapat dilakukan pengujian aplikasi dengan sendiri.
b) Mudah dalam perawatan perangkat lunak.
c) Dapat melakukan proses pengiriman *software* yang bersifat independen.
d) Mudah skalabilitas.
e) Pengembang dapat membangun aplikasi dengan berbagai bahasa pemrograman dan *framework*.

Arsitektur *Microservice* dijelaskan dalam gambar berikut.



Gambar 2.1 Arsitektur *Microservice*

Sumber : (Microsoft, 2023)

2.2 *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.[7] *Platform* ini menyediakan berbagai macam layanan hosting mulai dari komputasi, penyimpanan dan pengembangan aplikasi yang berjalan di perangkat keras *Google*. Layanan *Google Cloud Platform* dapat diakses dengan menggunakan *software* pengembang, administrator *cloud*, dan TI profesional lainnya yang menggunakan internet publik

atau melalui koneksi jaringan khusus. Beberapa produk *cloud computing* di Google Cloud meliputi:

- *Google Compute Engine*, merupakan layanan (IaaS) yang memberikan dukungan *instance* VM untuk *hosting*.
- *Google App Engine*, merupakan *platform* yang berfungsi sebagai layanan (PaaS) yang memberikan developer perangkat lunak dan akses ke *hosting* Google yang dapat diskalakan.
- *Google Cloud Storage*, merupakan *platform* penyimpanan *cloud* yang dirancang untuk menyimpan kumpulan data besar dan tidak terstruktur. Google juga menawarkan opsi penyimpanan *database*, termasuk penyimpanan non-relasional *Cloud Datastore for NoSQL*, penyimpanan cloud SQL for MySQL, dan *database Cloud Bigtable* asli Google.
- *Google Kubernetes Engine* (GKE), merupakan sistem manajemen dan orkestrasi untuk kontainer Docker yang berjalan dalam layanan *cloud* publik Google. Google Kubernetes Engine didasarkan pada Kubernetes, sistem manajemen kontainer *open source* Google.
- *Google Cloud's operation suite*, merupakan seperangkat alat terintegrasi untuk memantau, mencatat, dan melaporkan layanan terkelola yang mendorong aplikasi dan sistem di *Google Cloud*.
- *Serverless Computing*, menyediakan alat dan layanan *Cloud Functions* dan *Cloud Run*.
- Layanan *Database*, mencakup *Cloud Bigtable CloudSpanner*, dan *CloudSQL*.

2.3 *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.[8]

Framework NodeJS yang digunakan pada penelitian ini adalah ExpressJS. Express adalah *framework* aplikasi web yang minimalis dan fleksibel dari *NodeJS* yang menyediakan serangkaian fitur tangguh untuk aplikasi web dan *mobile*. ExpressJS menyediakan 4 mekanisme :

1. Penulisan penanganan *(handler)* untuk melakukan permintaan dengan kata kerja HTTP yang berbeda di jalur URL atau rute *(routes)* yang berbeda
2. Terintegrasi dengan mesin rendering *'view'* atau terintegrasi dengan bagian *frontend* sehingga dapat menghasilkan *response* dengan memasukan data ke dalam tampilan atau *user interfaces template*.
3. Mengatur pengaturan web aplikasi secara umum seperti port yang digunakan untuk menghubungkan server dengan lokasi dalam tampilan atau *user interfaces template* yang digunakan untuk merender *response*
4. Menambahkan pemrosesan permintaan tambahan "*middleware*" pada setiap titik dalam rute *(route)* permintaan (*request)*

ExpressJS adalah *unopiniated framework* yaitu *framework* yang tidak memiliki ketentuan dan hanya memiliki sedikit batasan tentang cara terbaik untuk menggunakan komponen yang ada di dalamnya. Hal ini membuat ExpressJS dikenal sebagai *framework* yang lebih fleksibel. Oleh karena itu, penelitian ini menggunakan *framework* ExpressJS. [9]

2.4 *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.[10] Beberapa sifat *Python* seperti *syntax* yang mudah diingat dan mudah dimengerti, dapat digunakan untuk berbagai hal seperti pengembangan web, pengolahan data, analisis data, dan banyak hal lain, serta bersifat publik dan tersedia secara gratis sehingga banyak fitur kustomisasi yang dapat dikembangkan dan digunakan oleh developer di seluruh dunia.

Framework Python yang digunakan pada penelitian ini adalah Flask. Flask adalah *micro-framework* yang di desain untuk menciptakan aplikasi web dengan cepat. Flask hanya mengimplementasikan fungsionalitas utama sehingga membuat *developer* fleksibilitas untuk menambahkan fitur sesuai kebutuhan. Flask juga mendukung fungsionalitas untuk *debugger, full request object,* sistem *routing* untuk *endpoint,* kontrol *cache,* tanggal, *cookeis,* dll. Oleh karena itu, penelitian ini menggunakan *framework* Flask. [11]

2.5 *REST API*

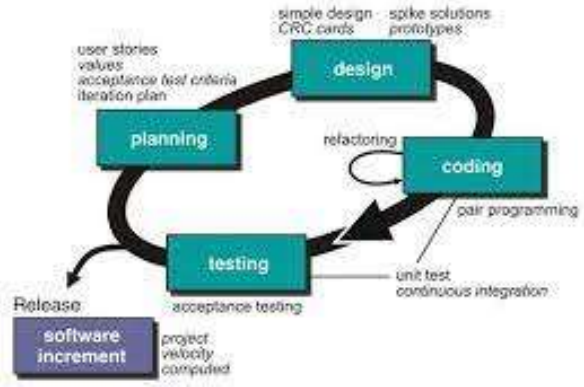
REST API adalah salah satu jenis *web service. REST API* memungkinkan untuk melakukan *request system* untuk mengakses dan memanipulasi teks yang direpresentasikan dari sebuah *web service.* Dalam penggunaannya, *REST API* terbukti lebih cepat dalam mentransfer data daripada metode lain yang serupa, misalnya SOAP. *REST API* sering digunakan untuk pengembangan *web service* yang scalable, mudah dipahami, dan interoperabilitas yang baik. [12]

2.6 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*.[13]

2.7 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. XP adalah salah satu metode tangkas yang paling banyak digunakan dan menjadi pendekatan yang sangat terkenal. Tujuan XP adalah tim yang terbentuk antara kursus berukuran kecil hingga menengah, tidak perlu menggunakan tim besar. Hal ini dimaksudkan untuk mengatasi persyaratan yang tidak jelas dan perubahan persyaratan dengan sangat cepat. 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.[14]



Gambar 2. 2 Skema XP *practices*

Sumber : (A Supriyatna, 2018)

Tahapan pada XP adalah sebagai berikut:

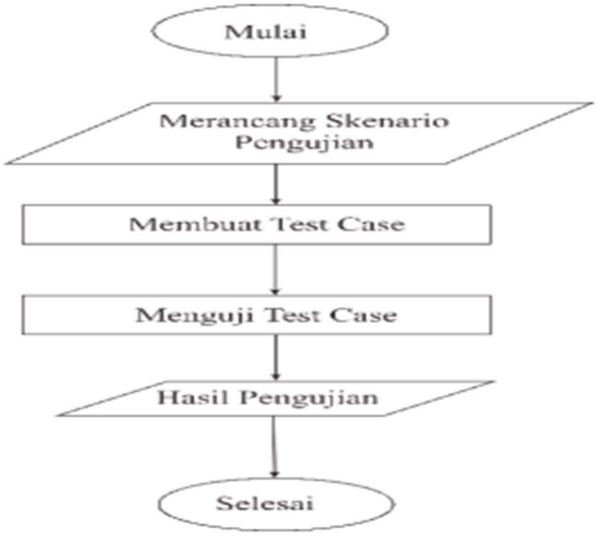
1. Perencanaan (*Planning*): Tahap ini dimulai dengan mengumpulkan kebutuhan aktifitas sistem agar pengguna dapat memahami proses bisnis sistem dan mendapatkan gambaran yang jelas tentang fitur utama, fungsionalitas, dan keluaran yang diinginkan.
2. Perancangan (*Design*): Tahap perancangan melibatkan pembuatan pemodelan sistem dan basis data yang menggambarkan hubungan antar data. Pemodelan sistem menggunakan Unified Modelling Language (UML), yang terdiri dari beberapa diagram seperti ERD (*Entity Relation Diagram*), *Use case Diagram, Architecture Diagram,* dan Desain Antarmuka.
3. Pengkodean (*Coding*): Tahap ini melibatkan implementasi dari perancangan model sistem yang telah dibuat menjadi kode program yang menghasilkan *microservice*.
4. Pengujian (*Testing*): Tahap ini merupakan tahap pengujian terhadap *microservice* yang sudah dibangun, fokus pada fitur dan fungsionalitas keseluruhan sistem, dan ditinjau oleh pengguna sistem. Metode pengujian yang digunakan adalah *Black-Box Testing* dengan melakukan pengujian terhadap masukkan dan keluaran yang dihasilkan sistem dan dan melakukan *Performance test* .
5. Peningkatan Perangkat Lunak (*Software Increment*): Tahap ini merupakan tahap pengembangan sistem yang sudah dibuat secara bertahap, setelah sistem diterapkan dalam organisasi dengan menambahkan layanan atau konten yang mengakibatkan bertambahnya kemampuan fungsionalitas dari sistem. [15]

2.8 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. Pengujian Black Box bertumpu pada memastikan tiap proses sudah berfungsi sesuai dengan kebutuhan yang diharapkan. Penguji dapat mengartikan himpunan kondisi masukan dan menjalankan pengujian pada pengkhususan fungsi dari sistem. Sehingga

pengujian merupakan suatu cara pelaksanaan program yang bertujuan menemukan kesalahan atau *error* kemudian memperbaikinya sehingga sistem dapat dikatakan layak untuk digunakan. [16]

Dalam black-box testing, pengujian dilakukan dengan memberikan masukan atau input tertentu ke perangkat lunak yang diuji dan mengevaluasi keluaran atau output yang dihasilkan. Metode ini biasanya dilakukan melalui serangkaian skenario pengujian yang dirancang untuk menguji berbagai kemungkinan masukan dan kondisi. Keuntungan utama dari black-box testing adalah bahwa tidak diperlukan pengetahuan teknis tentang internal struktur kode program atau algoritma perangkat lunak yang diuji. Oleh karena itu, pengujian dapat dilakukan oleh orang yang tidak terbiasa dengan bahasa pemrograman tertentu. Selain itu, metode ini dapat membantu mengidentifikasi kesalahan dan cacat yang mungkin tidak terdeteksi selama pengembangan perangkat lunak.



Gambar 2. 3 Diagram alir pengujian *Blackbox*

Sumber : (S, Uminingsih,Muhamad Nur Ichsanudin, Muhammad Yusuf, 2022)

Teknik pengujian pada Black Box testing ada banyak macamnya yaitu:

a. Teknik Equivalence Partitioning yaitu dengan cara melakukan partition atau pembagian menjadi beberapa partisi dari input data.
b. Teknik Boundary Value Analysis yaitu dengan cara mencari adakah error dari luar atau sisi dalam software, minimum maupun maximum nilai dari error yang di temukan.

c. Teknik Fuzzing yaitu merupakan teknik untuk mencari Bug/gangguan dari software dengan menggunakan injeksi data yang terbilang cacat .
d. Teknik Cause-Effect Graph ialah suatu Teknik testing dimana menggunakan graphic sebagai acuannya. Dimana dalam grafik ini menggambarkan relasi diantara efek dan penyebabnya.
e. Teknik Orthogonal Array Testing adalah jenis Teknik yang digunakan jika input domain yang relative terbilang kecil ukurannya, tetapi cukup berat untuk digunakan dalam skala besar.
f. Teknik All Pair Testing yaitu semua pasangan dari test case di desain sedemikian rupa agar dapat di eksekusi semua kemungkinan kombinasi diskrit dari seluruh pasangan berdasar input parameternya, Tujuan testing ini adalah memiliki pasangan test case yang mencakup semua pasangan tersebut.
g. Teknik state Transition yaitu teknik yang berguna untuk melakukan pengetesan terhadap kondisi dari mesin dan navigasi dalam bentuk grafik. [17]

2.9 Performance test

Sistem yang baik tentunya harus memiliki kualitas API yang mumpuni, untuk memastikan kualitas dari API pada sistem dibutuhkan pengujian seperti Performance testing. Performance testing merupakan salah satu jenis pengujian yang digunakan untuk menguji stabilitas dan keandalan sistem. Performance testing adalah cara untuk melihat bagaimana dampak dari suatu situasi yang berpotensi menimbulkan error pada tingkat ketahanan 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. Pengujian ini terutama menentukan ketahanan sistem dan penanganan kesalahan dalam kondisi beban yang sangat berat.

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. Performance testing dapat dilakukan melalui load testing tools, seperti salah satunya adalah BlazeMeter. [18]

2.10 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.

BlazeMeter menyediakan berbagai macam fitur dan alat pengujian untuk membantu pengujian kinerja aplikasi dengan cepat dan mudah. Platform ini dapat menskalakan pengujian kinerja dari beberapa ratus hingga jutaan pengguna secara bersamaan, dan menyediakan metrik kinerja yang akurat dan detil untuk membantu identifikasi masalah dan kelemahan dalam aplikasi.

Beberapa fitur dan alat yang disediakan oleh BlazeMeter antara lain:

a. Simulasi pengguna dalam jumlah besar dan terdistribusi dari berbagai lokasi geografis.
b. Perekaman skenario pengguna dengan mudah melalui browser extension atau melalui API.
c. Pengujian kinerja aplikasi dengan skala besar melalui cloud computing.
d. Kemampuan untuk mengukur dan memonitor kinerja aplikasi secara real-time.
e. Integrasi dengan berbagai platform pengembangan perangkat lunak dan alat pengujian lainnya.

BlazeMeter dapat digunakan oleh berbagai macam organisasi, mulai dari startup hingga perusahaan besar, dan dapat digunakan untuk menguji berbagai jenis aplikasi, seperti aplikasi web, aplikasi *mobile*, dan aplikasi berbasis cloud.

2.11 Penelitian Terkait

Penelitian terkait berisi artikel ilmiah ataupun jurnal yang berhubungan dengan penelitian ini dan dijadikan referensi. Beberapa hal yang dapat dijadikan referensi seperti kesamaan studi kasus ataupun penggunaan metode dalam pengembangan sistem. Beberapa artikel atau jurnal ilmiah yang penulis jadikan referensi untuk penelitian ini adalah :

1. Penelitian dari Dedy Puji Purwanto yang berjudul "*Rancang Bangun Back-End "Siap": Sistem Informasi Aspirasi Dan Pengaduan Masyarakat Berbasis Web Dengan Menggunakan Metode Microservice Springboot*"[19] menerangkan mengenai *Black Box testing*. *Blackbox testing* adalah pengujian yang dilakukan untuk memeriksa apakah fungsional aplikasi berjalan dengan baik dan benar tanpa mengetahui proses yang terjadi di dalam aplikasi. Pengujian *blackbox* dilakukan dengan membuat suatu *test case* yang berupa *test input* dan *output* yang diharapkan dari fungsional aplikasi. Pengujian bisa dilakukan pada aplikasi yang tidak menggunakan algoritma atau tingkat granulitas yang rendah.
2. Penelitian dari Rama Rahmanda yang berjudul "*Rancang Bangun Aplikasi Berbasis Microservice Untuk Klasifikasi Sentimen. Studi Kasus: Pt. Yesboss Group Indonesia (Kata.Ai)*"[20] menjelaskan bahwa menggunakan arsitektur *Microservice* sebuah perangkat lunak dapat mengurangi beban *server* karena arsitektur ini membagi fungsionalitas menjadi sejumlah *service* dan memberikan akses secara cepat. Dalam kasus penelitian yang dibuat oleh penulis supaya *verstand service* yang dimiliki oleh kata.ai dapat dikembangkan tanpa mengganggu *service* lainnya.
3. Penelitian dari Y.Yohakim Marwanta yang berjudul "*Implementasi Arsitektur Microservice Pada Pembuatan Surat Unit Kegiatan Mahasiswa Informatika Dan Komputer Menggunakan Node.Js*"[21] menerangkan bahwa REST adalah suatu arsitektur metode komunikasi yang menggunakan protokol HTTP untuk pertukaran data. Penelitian ini menghasilkan sebuah sistem aplikasi Pembuatan Surat Unit Kegiatan Mahasiswa Informatika dan Komputer dengan arsitektur *microservice* yang berkomunikasi menggunakan metode REST dan menggunakan bahasa *NodeJS* sebagai sistem *backend*-nya.

4. Penelitian dari Heri Purnama yang berjudul "*Aplikasi Pengelolaan Skripsi Di Stmik Akakom Yogyakarta Menggunakan Arsitektur Microservice Dengan Node.Js*"[5] menjelaskan bahwa *microservice* adalah sistem yang dapat digunakan untuk membangun layanan seara terpisah. Penelitian ini menghasilkan aplikasi pengelolaan skripsi menggunakan bahasa *NodeJS* dan *database* MongoDB di atas sistem operasi Linux : Ubuntu 14.04 LTS serta mengimplementasikan *Private Cloud* menggunakan platform *Docker*.
5. Penelitian dari Zaky Riko Virgiawan yang berjudul "*Perancangan Arsitektur Backend Microservice Pada Startup Campaign.Com*"[6] menjelaskan bahwa dalam memulai pengembangan perangkat lunak, salah satu poin terpenting adalah menentukan arsitektur teknologi sesuai dengan analisa kebutuhan produk. Penelitian ini menghasilkan desain arsitektur *microservice* untuk sistem *backend* pada *startup* campaign.com dengan penggunaan algoritma *Docker* yang membuat sistem menjadi efisien terhadap penulisan kode.

Tabel 2.5 Penelitian Terkait

Judul	**Peneliti**	**Tahun**	**Hasil Penelitian**
Rancang Bangun Back-End "Siap": Sistem Informasi Aspirasi Dan Pengaduan Masyarakat Berbasis Web Dengan Menggunakan Metode *Microservice* Springboot	Dedy Puji Purwanto	2017	Aplikasi SIAP: Sistem Informasi layanan Aspirasi dan Pengaduan Masyarakat berbasis web yang bisa dengan mudah **mengimplementasikan layanan *E-government*** Aspirasi dan Pengaduan Masyarakat.

Tabel 2.5 Penelitian Terkait Lanjutan

Judul	**Peneliti**	**Tahun**	**Hasil Penelitian**
Rancang Bangun Aplikasi Berbasis *Microservice* Untuk Klasifikasi Sentimen. Studi Kasus: Pt. Yesboss Group Indonesia (Kata.Ai)	Rama Rahmanda	2018	Aplikasi web service yang terintegrasi dengan layanan Verstand disertai model klasifikasi sentimen untuk dapat mengetahui opini pengguna lebih jauh, sehingga sistem akan bisa **memberikan jawaban yang tepat untuk kebutuhan pengguna.**
Implementasi Arsitektur *Microservice* Pada Pembuatan Surat Unit Kegiatan Mahasiswa Informatika Dan Komputer Menggunakan Node.Js	Y.Yohakim Marwanta	2019	Aplikasi Pembuatan Surat Unit Kegiatan Mahasiswa Informatika dan Komputer dengan menggunakan Node.js yang dapat digunakan oleh sekretaris UKM untuk **mengatasi keterlambatan dalam pembuatan surat atau perbaikan surat pada saat ia mengikuti perkuliahan.**
Aplikasi Pengelolaan Skripsi Di Stmik Akakom Yogyakarta Menggunakan Arsitektur *Microservice* Dengan Node.Js	Heri Purnama	2019	Aplikasi pengelolaan Skripsi untuk **menghindari plagiarisme** menggunakan arsitektur *microservice*, nodejs, mongodb, dan docker.

Tabel 2.5 Penelitian Terkait Lanjutan

Judul	**Peneliti**	**Tahun**	**Hasil Penelitian**
Perancangan Arsitektur *Backend Microservice* Pada Startup Campaign.Com	Zaky Riko Virgiawan	2022	Mengetahui perbedaan antara arsitektur monolitik dan *microservice* dengan kekurangan dan kelebihannya. Yang mana sebenarnya dapat mengguanakan kedua arsitektur tersebut, namun **lebih efisien jika menggunakan arsitektur *microservice*.**

III. METODOLOGI 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 3.1 yang menunjukkan jadwal kegiatan penelitian yang dilakukan.

Tabel 3.1 Jadwal 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

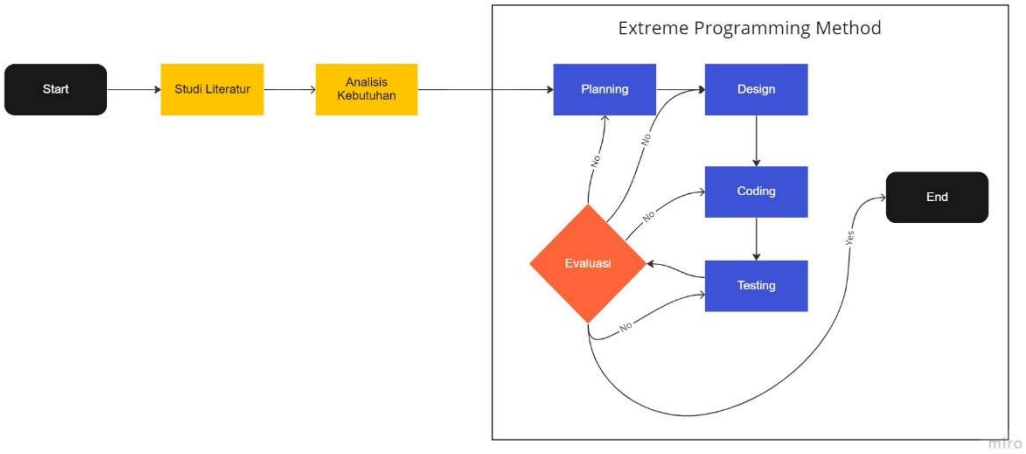
Alat yang digunakan dalam penelitian ini adalah sebagai berikut :

Tabel 3.2 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 3.1 Tahapan penelitian

3.3.1. Planning

Tahap pertama dari model XR adalah *planning.* Pada tahap ini informasi yang dibutuhkan untuk pembuatan aplikasi RIFSA dikumpulkan. Kebutuhan fungsional dari aplikasi RIFSA ditentukan pada tahap ini. Secara rinci, tahapan *Planning* dijelaskan pada bab metodologi penelitian.

3.3.2.Design

Langkah selanjutnya adalah mendesain sistem aplikasi RIFSA. Tahap ini dimaksud untuk memudahkan memahami dan membuat aplikasi RIFSA. Pada tahap ini desain sistem yang akan dibuat mencakup *flow diagram*, *usecase diagram*, dan *entity relationship diagram*. Sedangkan untuk desain antarmuka tidak termasuk berdasarkan batasan masalah yang telah dibuat.

3.3.3. Coding

Tahap selanjutnya dari model *Extreme Programming* adalah melakukan pengembangan *microservice* menggunakan *NodeJS* dengan *framework* ExpressJS dan *Python* dengan *framework* Flask. Pembuatan *microservice* untuk aplikasi RIFSA didukung dengan *software* seperti Visual Studio Code dan Postman.

3.3.4. Testing

Proses *testing* atau pengujian *microservice* untuk aplikasi RIFSA dilakukan dengan dua metode yaitu *functional testing* dan *Performance test*. Pengujian dilakukan untuk mengetahui apakah *microservice* untuk aplikasi RIFSA yang telah dibuat sesuai dengan kebutuhan dari aplikasi RIFSA tersebut.

Functional testing dilakukan dengan menguji setiap *endpoint* menggunakan *software* Postman. Pada *software* postman akan dimasukkan *method* HTTP yang sesuai serta *body request* berdasarkan *test scenario* yang dibuat. Kemudian dari *test scenario* tersebut akan dilakukan pengujian dan melihat apakah hasil sistem sesuai dengan yang diharapkan. Format *test scenario* berisi jenis skenario, prosedur pengujian, masukan, dan ekspektasi luaran [22]. Tabel contoh *test scenario* adalah sebagai berikut :

Tabel 3. 3 Tabel Contoh *Test Scenario*

No	Skenario	Prosedur Pengujian	Masukan	Ekspektasi Luaran	Hasil
1	*Normal/Alternatif*	*Berisi tata cara untuk melakukan pengujian beserta deskripsinya*	*Berisi masukan yang akan digunakan*	*Berisi deskripsi ekspektasi luaran dari hasil test*	*Sukses/Gagal*
2	*Normal/Alternatif*	*Berisi tata cara untuk melakukan pengujian beserta deskripsinya*	*Berisi masukan yang akan digunakan*	*Berisi deskripsi ekspektasi luaran dari hasil test*	*Sukses/Gagal*

Performance test dilakukan pada platform *BlazeMeter*. Pengetesan akan dilakukan dengan menguji *endpoint* berdasarkan pengelompokan fitur. Tes ini akan terbagi menjadi fitur *authentication,* fitur hasil panen, fitur inventaris, fitur keuangan, dan fitur penyakit. Parameter pengetesan akan terbagi menjadi 2, yaitu *load* dan *response time*. Pengetesan dilakukan 1 (satu) kali selama rentang waktu 1 menit. Kemudian akan didapat grafik hasil pengetesan.

V. KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan hasil yang telah disampaikan, maka didapat kesimpulan 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

5.2. Saran

Saran untuk pengembangan selanjutnya adalah sebagai berikut :

1. Penelitian selanjutnya agar dilakukan dengan bahasa dan *framework* yang berbeda agar dapat menjadi perbandingan.
2. Penelitian selanjutnya agar dapat meningkatkan parameter *testing* tidak hanya terfokus pada *load* dan *response time* saja.
3. Penelitian selanjutnya agar dapat dilakukan dengan arsitektur lain selain *REST API*, misalnya GraphQL, SOAP, dll.

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] R. Dwi Prastyo and D. A. Puryono, "Sistem Informasi Pendeteksi Hama Penyakit Tanaman Padi Menggunakan Metode Fuzzy Tsukamoto Berbasis Android," CDROM, 2018.

[5] 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/

[6] 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.

[7] 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.

[8] M. Iqbal C. R., "Implementasi Klien SIP Berbasis Web Menggunakan HTML5 dan Node.js," 2012.

[9] Nasution, "IMPLEMENTASI MONGO DB, EXPRESS JS, REACT JS DAN NODE JS (MERN) PADA PENGEMBANGAN APLIKASI FORMULIR, KUIS, DAN SURVEI ONLINE," Yogyakarta, 2021.

[10] S. Nasional and R. X. F. Ums, "DETEKSI WAJAH METODE VIOLA JONES PADA OPENCV MENGGUNAKAN PEMROGRAMAN *PYTHON*," 2012.

[11] K. Salo, "Comparative study on *Python* web frameworks: Flask and Django," Helsinki, May 2020.

[12] A. Rulloh, "Implementasi *REST API* pada Aplikasi Panduan Kepaskibraan Berbasis Android," *Teknikom*, vol. 1, no. 2, pp. 2598–2958, 2017.

[13] 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.

[14] 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.

[15] E. B. Pratama and Ade Hendini, "IMPLEMENTASI *EXTREME PROGRAMMING* PADA PERANCANGAN SIMRS (SISTEM

INFORMASI MANAJEMEN RUMAH SAKIT)," *JURNAL KHATULISTIWA INFORMATIKA*, 2022.

[16] 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.

[17] M. Nur Ichsanudin, M. Yusuf, S. Jurusan Rekayasa Sistem Komputer, J. Teknik Industri, I. AKPRIND Yogyakarta, and R. Artikel, "PENGUJIAN FUNGSIONAL PERANGKAT LUNAK SISTEM INFORMASI PERPUSTAKAAN DENGAN METODE BLACK BOX TESTING BAGI PEMULA INFO ARTIKEL ABSTRAK," vol. 1, no. 2, pp. 1–8, 2022, doi: 10.55123.

[18] N. Luh, A. S. Ginasari, K. S. Wibawa, N. Kadek, and A. Wirdiani, "Pengujian Stress Testing API Sistem Pelayanan dengan Apache JMeter," 2021.

[19] H. Suryotrisongko, S. Kom, and M. Eng, "FINAL PROJECT-KS141501 APPLICATION BACKEND DEVELOPMENT OF 'SIAP' : 'SISTEM INFORMASI ASPIRASI DAN PENGADUAN MASYARAKAT' APPLICATION BASED ON WEB USING *MICROSERVICE* SPRINGBOOT METHOD," 2017.

[20] I. Aris Tjahyanto, "INDONESIA (KATA.AI) DEVELOPMENT OF *MICROSERVICE* BASED APPLICATION FOR SENTIMENT CLASSIFICATION. CASE STUDY: PT. YESBOSS GROUP INDONESIA (KATA.AI)," 2018.

[21] Yy. Marwanta, "IMPLEMENTASI ARSITEKTUR *MICROSERVICE* PADA PEMBUATAN SURAT UNIT KEGIATAN MAHASISWA INFORMATIKA DAN KOMPUTER MENGGUNAKAN NODE.JS," 2019.

[22] W. A. Fillah, “RANCANG BANGUN APLIKASI E-INKUBATOR: PLATFORM PENYEDIA LAYANAN INKUBASI DAN INVESTASI BAGI UMKM BERBASIS *MICROSERVICE*,” 2019.

LAMPIRAN