IMPLEMENTASI LAYANAN *CLOUD COMPUTING* CI/CD PADA APLIKASI PENDETEKSI KEPITING SOKA

LAPORAN KERJA PRAKTIK

Disusun Oleh:

Gabriel Fransiscus Tumewu

20013001

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS KATOLIK DE LA SALLE MANADO
2023

IMPLEMENTASI LAYANAN *CLOUD COMPUTING* CI/CD PADA APLIKASI PENDETEKSI KEPITING SOKA

LAPORAN KERJA PRAKTIK

Ditulis untuk Memenuhi Persyaratan Mata Kuliah Kerja Praktik

(INF2217401

Disusun Oleh:

Gabriel Fransiscus Tumewu

20013001

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS KATOLIK DE LA SALLE MANADO

2023

LEMBAR PENGESAHAN

LAPORAN KERJA PRAKTIK

Judul:

IMPLEMENTASI LAYANAN *CLOUD COMPUTING* CI/CD PADA APLIKASI PENDETEKSI KEPITING SOKA

Telah disetujui dan disahkan pada tanggal: 31 Januari 2024

Oleh:

Bangkit Academy

Dita Anggraini Ekawati
Cohort Manager

LEMBAR PERNYATAAN

Yang bertanda tangan dibawah ini:

Nama	: Gabriel Fransiscus Tumewu
NIM	: 20013001
Tempat/tanggal lahir	: Manado/02 April 2002
Fakultas/Program Studi	: Teknik/Teknik Informatika

Menyatakan bahwa laporan Kerja Praktik dan atau Aplikasi/Program berjudul "**Implementasi Layanan *Cloud Computing* CI/CD pada Aplikasi Pendeteksi Kepiting Soka"** yang penulis buat adalah benar hasil karya penulis dan bukan karya tulis orang lain, baik sebagian atau seluruhnya kecuali dalam bentuk kutipan yang telah disebutkan sumbernya.

Demikian pernyataan ini penulis buat dengan sebenar-benarnya dan apabila pernyataan ini tidak benar, maka penulis bersedia menerima sanksi akademis sesuai dengan yang diterapkan oleh Fakultas Teknik, berupa pembatalan Kerja Praktik dan hasilnya.

Manado, 15 Desember 2023

Yang Menyatakan,

Gabriel Fransiskus Tumewu

Menyetujui,

Dosen Pembimbing I	Dosen Pembimbing II
Steven Pandelaki, S.T., M.Sc.	**Indah Kairupan, S.T., M.Sc.**

Mengetahui,

Ketua Program Studi	Dekan Fakultas Teknik
Vivie Deyby Kumenap, S.T., M.C.S.	**Ronald A. Rachmadi, S.T., M.T.**

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS KATOLIK DE LA SALLE
MANADO

FORM KP - 003

FORMULIR DATA UMUM PERUSAHAAN

NAMA MAHASISWA	: Gabriel Fransiskus Tumewu
NIM	: 20013001
NAMA PERUSAHAAN	: Yayasan Dicoding Indonesia
ALAMAT PERUSAHAAN	: Dicoding Space, Jalan Batik Kumeli No 50, Kecamatan: Cibeunying Kaler, Kelurahan: Sukaluyu, RT: 10, RW: 07, Kota Bandung, Jawa Barat, 40123
DIDIRIKAN TAHUN	: 2018
IJIN USAHA	: NIB (9120304611285)
BIDANG BISNIS	: Jasa Pendidikan Komputer (Teknologi Informasi)
JUMLAH KARYAWAN	: 95
PEMILIK	: Narenda Wicaksono & Kevin Kurniawan
DEWAN DIREKTUR	: Dimas Catur Wibowo
	: Nur Rohman
	: Ahmad Imaduddin
WAKIL PERUSAHAAN	
Tanggal	: 16 Januari 2024
Nama	: Kevin Kurniawan
Jabatan	: Pendiri Yayasan Dicoding Indonesia
(Tanda tangan dan cap perusahaan)	: Yayasan dicoding Indonesia

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS KATOLIK DE LA SALLE
MANADO

FORM KP - 004

FORMULIR PENILAIAN KEMAJUAN KERJA PRAKTEK

A. UMUM

Nama Mahasiswa	: Gabriel Fransiskus Tumewu
NIM Mahasiswa	: 20013001
Program Studi	: Teknik Informatika
Dosen Pembimbing Akademik	: Indah Y. Kairupan, S.T., M.Sc.
Topik/Rencana Bidang	: Implementasi Layanan *Cloud* CI/CD Pada Aplikasi Pendeteksi Kepiting Soka
Pembimbing 1	: Steven Pandelaki, S.T., M.Sc.
Terhitung Mulai	: 14 Agustus 2023
Target Selesai	: 31 Januari 2024

B. KEGIATAN PELAKSANAAN KERJA PRAKTEK

No.	Tanggal	Jenis Kegiatan	Paraf Pembimbing
1.	18 September 2023	Konsultasi Progres Bangkit	
2.	19 Oktober 2023	Konsultasi Progres Bangkit	
3.	25 Oktober 2023	Pembahasan Topik Penelitian	
4.	26 Oktober 2023	Pembahasan Topik Penelitian	
5.	27 Oktober 2023	Konsultasi BAB I	
6.	31 Oktober 2023	Konsultasi BAB I	

7.	2 November 2023	Revisi BAB I	
8.	6 November 2023	Revisi BAB I	
9.	8 November 2023	Konsultasi BAB II	
10.	10 November 2023	Revisi BAB II	
11.	15 November 2023	Konsultasi BAB III	
12.	22 November 2023	Revisi BAB III	
13.	29 November 2024	Konsultasi BAB IV	
14.	11 Desember 2023	Konsultasi Aplikasi	
15.	2 Januari 2024	Konsultasi Progres Bangkit	
16.	22 Januari 2024	Konsultasi Keseluruhan Laporan Akhir	

Manado, 31 Januari 2024

Dosen Pembimbing KP

(Steven Pandelaki, S.T., M.Sc.)

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS KATOLIK DE LA SALLE
MANADO

FORM KP - 005

FORMULIR PENILAIAN PELAKSANAAN KERJA PRAKTEK

Mohon diisi dan dicek seperlunya,

NAMA MAHASISWA	: Gabriel Fransiskus Tumewu
NIM	: 20013001
NAMA PERUSAHAAN	: Yayasan Dicoding Indonesia
ALAMAT PERUSAHAAN	: Dicoding Space, Jalan Batik Kumeli No 50, Kecamatan: Cibeunying Kaler, Kelurahan: Sukaluyu, RT: 10, RW: 07, Kota Bandung, Jawa Barat, 40123
TGL KERJA PRAKTEK	: 14 Agustus 2023 s.d 31 Desember 2023
TOPIK YANG DIBAHAS	: Cloud Computing Learning Path

Nilai Sikap	=	50	60	70	80	(90)	100
Kerajinan	=	50	60	70	80	90	(100)
Prestasi	=	50	60	70	(80)	90	100

KOMENTAR/SARAN

Mahasiswa telah mengikuti kegiatan dengan baik.

NILAI RATA-RATA	: 90
TANGGAL	: 16 Januari 2024
NAMA PENILAI	: Deti Anggraini Ekawati
JABATAN	: Cohort Manager

(Tanda tangan dan cap perusahaan) : 

KATA PENGANTAR

Puji Syukur kepada Tuhan yang Maha Esa atas penyertaan-Nya sehingga penulis dapat menyelesaikan Laporan Kerja Praktik di Bangkit *Academy* dalam rangka Program Magang dan Studi Independen Bersetifikat (MSIB) dengan judul Implementasi Layanan *Cloud* CI/CD pada Aplikasi Pendeteksi Kepiting Soka dengan baik. Pembuatan laporan akhir ini bertujuan untuk memenuhi persyaratan dokumentasi selama program Kampus Merdeka.

Selama mengikuti kegiatan sampai penulisan laporan akhir ini, penulis menyadari bahwa tanpa bantuan dari orang-orang disekitar, maka penulis tidak akan sampai ketahap ini. Untuk itu penulis mengucapkan terima kasih kepada:

1. Prof. Dr. Johanis Ohoitimur, selaku Rektor Universitas Katolik De La Salle Manado.
2. Bapak Ronald Albert Rachmadi, S.T., M.T., selaku Dekan Fakultas Teknik.
3. Ibu Vivie Deyby Kumenap, S.T., M.S.C., selaku Ketua Program Studi Teknik Informatika.
4. Bapak Michael George Sumampow, S.T., M.T., selaku Koordinator Instansi Universitas Katolik De La Salle Manado dalam program MSIB.
5. Bapak Steven Pandelaki, S.T., M.Sc., selaku Dosen Pembimbing I penulis selama pembuatan laporan ini.
6. Ibu Indah Yessi Kairupan, S.T., M.Sc., selaku Dosen Pembimbing akademik dan Supervisor selama kegiatan Studi Independen, juga sebagai dosen pembimbing II selama penulis membuat laporan ini.
7. Tim Bangkit *Academy* yang telah memberikan pengalaman baru kepada penulis selama program MSIB ini.
8. Aditya Fataha Dwijaya, S.T., selaku mentor selama program Studi Independen.
9. Darren Ngoh dan Rahmat Fajri, sebagai *Advisor* yang memberikan arahan dalam pembuatan proyek Capstone.
10. Tim Project Capstone Crabify yang telah berjuang bersama untuk menyelesaikan proyek sampai akhir.
11. Ayah, Ibu, Adik, dan seluruh anggota keluarga besar yang telah mendukung dari awal proses pendaftaran hingga selesai program Studi Independe.

12. Gabrielle N. A. Frederica yang telah menjadi rekan dalam belajar dan pengerjaan proyek semasa Studi Independen.
13. Teman-teman Angkatan 2020 program Studi Teknik Informatika yang telah mendukung penulis selama pembuatan laporan ini.

Oleh sebab itu, penulis ingin menyampaikan terima kasih yang sebesar-besarnya kepada seluruh pihak terkait. Penulis sangat mengharapkan kritik dan saran sebagai masukkan yang berguna untuk perkembangan penulis. Semoga laporan ini dapat berguna bagi pembaca.

Manado, Januari 2024

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1. Latar Belakang Masalah

Aplikasi pendeteksi kepiting soka digunakan untuk mengklasifikasikan kepiting menjadi dua jenis, yaitu kepiting soka dan kepiting biasa. Dalam pembuatan aplikasi ini digunakan layanan *Cloud* berupa *firebase*, yang bertujuan sebagai media penghubung model *machine learning* yang digunakan dengan aplikasi Android yang dibuat.

Penggunaan layanan *firebase* yang terpisah memiliki keuntungan dalam hal implementasi yang cepat. Dalam pengimplementasian layanan *firebase* ini, ditujukan secara khusus pada bagian *user authentication* dan *database*. Penggunaan layanan yang terpisah tentu saja memiliki kelemahan pada bagian lain, seperti kustomisasi yang terbatas yang membatasi perubahan fitur-fitur tertentu dan kurang memiliki kontrol penuh atas infrastruktur dan konfigurasi tingkat rendah dibandingkan dengan menggunakan solusi *cloud* yang lebih kustomisasi. [1]

Continuous Integration dan *Continuous Deployment* (atau *Continuous Delivery*) merupakan praktik pengembangan perangkat lunak yang bertujuan untuk meningkatkan efisiensi dan kualitas pengembangan perangkat lunak dengan cara otomatisasi dan pengujian berkelanjutan. Untuk mengatasi kelemahan tersebut, metode *Continuous Integration* (CI) dan *Continuous Deployment* (CD) dapat diterapkan. CI/CD memungkinkan otomatisasi dan integrasi berkelanjutan dalam siklus pengembangan perangkat lunak. [2] Dengan menerapkan CI/CD, perubahan kode dapat diuji secara otomatis dan diterapkan ke lingkungan produksi dengan cepat dan aman. Ini membantu dalam meningkatkan kualitas perangkat lunak, mendeteksi dini bug, dan mempercepat proses pengiriman perangkat lunak.

Dengan mengintegrasikan CI/CD ke dalam proyek, tim pengembang dapat merespons perubahan dengan lebih cepat, sambil mempertahankan keuntungan implementasi cepat yang diberikan oleh layanan *Firebase*. Hal ini menciptakan siklus pengembangan yang efisien dan responsif, menjembatani kelemahan terkait

kustomisasi dan kontrol infrastruktur yang mungkin dialami dengan penggunaan layanan *Firebase* terpisah.

1.2. Rumusan Masalah

Bagaimana mengimplementasikan *Continuous Integration* dan *Continuous Deployment* CI/CD pada Aplikasi Pendeteksi Kepiting yang dapat membantu layanan *Cloud*?

1.3. Tujuan Kerja Praktik

Mengimplementasikan *Continuous Integration* dan *Continuous Deployment* CI/CD pada Aplikasi Pendeteksi Kepiting yang dapat membantu layanan *Cloud.*

1.4. Manfaat Kerja Praktik

1.4.1 Bagi Pengembang Aplikasi

a. Meningkatkan efektifitas layanan *Cloud*.
b. Meningkatkan kualitas perangkat lunak dengan layanan *Cloud*.

1.4.2 Bagi Penulis

a. Memperoleh wawasan mengenai layanan *Cloud*.
b. Memperoleh pengetahuan dalam pengimplementasian *Continuous Integration* dan *Continuous Deployment* CI/CD pada aplikasi.

1.5. Batasan Masalah

Dalam proses penelitian ini, terdapat beberapa batasan untuk memfokuskan penelitian ini sesuai dengan tujuan diatas.

1. Penelitian hanya akan membahas layanan *Cloud* dengan menggunakan metode CI/CD pada aplikasi pendeteksi kepiting soka.
2. Penelitian hanya akan menggunakan layanan *Cloud App engine*.
3. Penelitian ini hanya akan membahas tentang *cloud* dan tidak akan membahas bagian pengolahan data *machine learning*.

1.6. Sistematika Penulisan

Berikut adalah uraian mengenai struktur penulisan dalam laporan kerja praktik ini, dengan tujuan untuk meningkatkan keteraturan laporan melalui pembagian setiap modul ke dalam beberapa bab berikut:

1. BAB I PENDAHULUAN

   Pada bab ini akan membahas mengenai latar belakang masalah yang diangkat, tujuan kerja praktik, manfaat yang diperoleh dari hasil kerja praktik, batasan masalah, serta sistematika penulisan laporan.

2. BAB II DATA UMUM PERUSAHAAN

   Pada bab ini akan membahas mengenai sejarah singkat dari Bangkit *Academy*, lingkup kerja Bangkit *Academy*, visi, misi dan lingkup pekerjaan yang dilakukan selama berada di Bangkit *Academy*.

3. BAB III LANDASAN TEORI

   Pada bab ini akan membahas mengenai teori-teori yang menjadi landasan dalam pembuatan penelitian ini, teori topik yang diangkat, algoritma yang akan digunakan dan juga metodologi penelitian.

4. BAB IV PEMBAHASAN

   Pada bab ini akan membahas mengenai proses pengumpulan dan pengolahan data, kemudian analisi dan pemecahan masalah berdasarkan metodologi penelitian.

5. BAB V KESIMPULAN DAN SARAN

   Pada bab ini akan membahas mengenai kesimpulan yang diperoleh dalam penelitian ini dan saran untuk pengembangan penelitian ini atau penelitian dengan topik serupa

   .

BAB II
DATA UMUM PERUSAHAAN

2.1. Sejarah Singkat Perusahaan

Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi (Kemendikbudristek) merupakan institusi pemerintah di Indonesia yang memiliki tanggung jawab utama terhadap pengembangan sektor pendidikan, kebudayaan, riset, dan teknologi. Peran kementerian ini sangat signifikan dalam membentuk masa depan bangsa Indonesia melalui inisiatif pembangunan sumber daya manusia yang berkualitas, pengembangan kebudayaan yang kaya dan beragam, serta memajukan riset dan teknologi untuk mencapai kemajuan dan daya saing nasional. [3]

Sejarah Kemendikbudristek dimulai sejak tahun 1945, saat Indonesia meraih kemerdekaannya. Pada periode awal, pengelolaan bidang pendidikan, kebudayaan, riset, dan teknologi dilakukan oleh beberapa lembaga terpisah. Pendidikan sendiri dikelola oleh lembaga yang disebut sebagai "Kementerian Pengajaran". [3]

Seiring perkembangan waktu, perubahan struktur dan fokus tugas lembaga-lembaga tersebut berlangsung, hingga akhirnya membentuk Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi yang memiliki mandat menyeluruh terhadap sektor-sektor tersebut. Transformasi ini mencerminkan komitmen pemerintah Indonesia dalam memajukan pendidikan, melestarikan kebudayaan, mendorong riset, dan mengembangkan teknologi untuk mencapai tujuan pembangunan nasional. [3]

Peran Nadiem Anwar Makarim sebagai Menteri Pendidikan dan Kebudayaan pada tahun 2019 menandai upaya lanjutan untuk membawa perubahan positif dalam sektor pendidikan dan kebudayaan Indonesia. Posisinya dalam Kabinet Indonesia Maju memberikan arah dan kebijakan untuk mencapai tujuan pembangunan nasional di bidang pendidikan dan kebudayaan. Kemudian pada tahun 2020 didirikan sebuah Perusahaan pengajaran secara *online* yang disebut Kampus Merdeka. Dalam Kampus Merdeka ini dijuga salah satu program mengajaran *e-learning,* yakni *Bangkit Academy*.

Bangkit *Academy* merupakan suatu program pembelajaran *online* yang dirancang untuk mempersiapkan karir mahasiswa dengan ketrampilan dalam bidang teknologi dan Perusahaan *startup*. Bangkit didirikan pertama kali pada tahun 2020 dengan dukungan berbagai perusahaan seperti Google, GoTo, dan Traveloka.

2.2. Lingkup Pekerjaan Perusahaan

Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi (Kemendikbudristek) memiliki peran yang sangat penting dalam pembangunan sektor pendidikan, kebudayaan, ilmu pengetahuan, dan teknologi di dalam wilayah negara. Sebagai penanggung jawab utama, Kemendikbudristek bertanggung jawab atas pengembangan kurikulum, penyediaan sarana dan prasarana pendidikan, serta memastikan bahwa pendidikan dan penelitian di Indonesia berada pada jalur yang berkelanjutan dan berdaya saing global.

Salah satu program unggulan yang diinisiasi pemerintah adalah Bangkit *Academy*. Bangkit *Academy* adalah sebuah inisiatif dalam bidang akademik yang khusus berfokus pada pengembangan pembelajaran *online* (*e-learning*). Tujuan utama dari Bangkit *Academy* adalah untuk meningkatkan kualitas sumber daya manusia di Indonesia, terutama dalam bidang teknologi dan digitalisasi. Sejak didirikan pada tahun 2020, Bangkit *Academy* telah menjadi salah satu wadah penting bagi individu yang ingin mengembangkan diri mereka dalam teknologi modern.

Seiring berjalannya waktu, Bangkit *Academy* terus berkembang dan menyediakan beragam program pembelajaran yang relevan dengan tuntutan pasar dan kebutuhan industri. Saat ini, Bangkit *Academy* telah meluncurkan tiga jenis program pembelajaran utama, yaitu Android, *Machine Learning*, dan *Cloud Computing*. Setiap program ini disusun dengan sangat cermat berdasarkan standar global terkini dan dipadukan dengan pengajar dan mentor berkualitas tinggi. Bangkit *Academy* bukan hanya sekadar menyediakan materi pembelajaran, tetapi juga menyelenggarakan berbagai kegiatan pendukung seperti webinar, lokakarya, dan proyek kolaboratif. Hal ini bertujuan untuk memberikan pengalaman belajar yang holistik dan praktis bagi peserta. Selain itu, Bangkit *Academy* juga menjalin

kemitraan dengan berbagai instansi, perusahaan, dan lembaga pendidikan baik di dalam maupun luar negeri, untuk memperluas jaringan dan meningkatkan aksesibilitas terhadap peluang belajar dan karier di bidang teknologi.

Dengan komitmen kuat untuk memberikan pendidikan yang berkualitas tinggi dan relevan dengan perkembangan zaman, Bangkit *Academy* diharapkan dapat berperan aktif dalam mencetak generasi muda yang kompeten dan siap bersaing dalam era digital ini. Dengan dukungan penuh dari Kemendikbudristek serta berbagai pihak terkait lainnya, Bangkit *Academy* berupaya menjadi garda terdepan dalam menggerakkan revolusi pendidikan dan teknologi di Indonesia.

2.2.1. Visi

Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi mendukung Visi dan Misi Presiden untuk mewujudkan Indonesia Maju yang berdaulat, mandiri, dan berkepribadian melalui terciptanya Pelajar Pancasila yang bernalar kritis, kreatif, mandiri, beriman, bertakwa kepada Tuhan Yang Maha Esa, dan berakhlak mulia, bergotong royong, dan berkebinekaan global. [3]

2.2.2. Misi

1. Mewujudkan pendidikan yang relevan dan berkualitas tinggi, merata dan berkelanjutan, didukung oleh infrastruktur dan teknologi.
2. Mewujudkan pelestarian dan pemajuan kebudayaan serta pengembangan bahasa dan sastra.
3. Mengoptimalkan peran serta seluruh pemangku kepentingan untuk mendukung transformasi dan reformasi pengelolaan pendidikan dan kebudayaan. [3]

2.2.3. Logo Kemendikbud

Gambar 2.1 Logo Kementrian Pendidikan dan Kebudayaan

Gambar 2.1 memiliki beberapa artian dari setiap detail yang digunakan untuk membentuk logo tersebut, yakni sebagai berikut:

1. Bidang Segi Lima (Biru Muda) menggambarkan alam kehidupan Pancasila.
2. Semboyan Tut Wuri Handayani digunakan oleh Ki Hajar Dewantara dalam melaksanakan sistem pendidikannya. Pencantuman semboyan ini berarti melengkapi penghargaan dan penghormatan kita terhadap almarhum Ki Hajar Dewantara yang hari lahirnya telah dijadikan Hari Pendidikan Nasional.
3. Belencong Menyala Bermotif Garuda Belencong (menyala) merupakan lampu yang khusus dipergunakan pada pertunjukan wayang kulit. Cahaya belencong membuat pertunjukan menjadi hidup.
4. Burung Garuda (yang menjadi motif belencong) memberikan gambaran sifat dinamis, gagah perkasa, mampu dan berani mandiri mengarungi angkasa luas. Ekor dan sayap garuda digambarkan masing-masing lima, yang berarti: ‘satu kata dengan perbuatan Pancasilais’.
5. Buku merupakan sumber bagi segala ilmu yang dapat bermanfaat bagi kehidupan manusia.
6. Warna: Warna putih pada ekor dan sayap garuda dan buku berarti suci, bersih tanpa pamrih. Warna kuning emas pada nyala api berarti keagungan dan keluhuran pengabdian. Warna biru muda pada bidang segi lima berarti pengabdian yang tak kunjung putus dengan memiliki pandangan hidup yang mendalam (pandangan hidup Pancasila)

2.2.4. Tugas dan Fungsi Kementerian Pendidikan dan Kebudayaan

Tugas dan fungsi Kementerian Pendidikan dan Kebudayaan melibatkan peran kunci dalam pengembangan sektor pendidikan dan kebudayaan di Indonesia. Tanggung jawab utama kementerian ini mencakup merumuskan kebijakan, merancang program, dan mengelola sumber daya dengan tujuan meningkatkan kualitas pendidikan nasional. Berikut ini merupakan rincian tugas dan fungsi Kementerian Pendidikan dan kebudayaan.

2.2.4.1. Tugas

Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi (Kemendikbudristek) memiliki tanggung jawab yang luas dalam menyelenggarakan urusan pemerintahan di bidang pendidikan anak usia dini, pendidikan dasar, pendidikan menengah, dan pendidikan masyarakat, serta pengelolaan kebudayaan. Melalui kebijakan, program, dan pengelolaan sumber daya, Kemendikbudristek berupaya membantu Presiden dalam menyelenggarakan pemerintahan negara dengan fokus pada peningkatan kualitas pendidikan dan pelestarian kebudayaan nasional. Mulai dari pengaturan standar kurikulum, pelatihan tenaga pendidik, hingga pemantauan mutu pendidikan, Kemendikbudristek memastikan bahwa setiap tahap pendidikan, dari anak usia dini hingga menengah, terselenggara dengan baik. Selain itu, melalui program pendidikan masyarakat dan kampanye kesadaran akan pentingnya pendidikan, kementerian ini juga berupaya memastikan akses pendidikan yang merata bagi seluruh lapisan masyarakat. Di samping itu, Kemendikbudristek juga terlibat dalam pengelolaan kebudayaan nasional, termasuk dalam pelestarian warisan budaya, promosi seni dan budaya Indonesia, serta pengembangan kegiatan budaya yang memperkuat identitas bangsa. Dengan demikian, peran Kemendikbudristek sangat penting dalam mendukung visi pemerintah dalam memajukan sektor pendidikan dan kebudayaan untuk kesejahteraan bangsa dan negara [3].

2.2.4.2. Fungsi

a. Perumusan dan penetapan kebijakan di bidang pendidikan anak usia dini, pendidikan dasar, pendidikan menengah, dan pendidikan masyarakat, serta pengelolaan kebudayaan;

b. Pelaksanaan fasilitasi penyelenggaraan pendidikan anak usia dini, pendidikan dasar, pendidikan menengah, dan pendidikan masyarakat, serta pengelolaan kebudayaan;
c. Pelaksanaan kebijakan di bidang peningkatan mutu dan kesejahteraan guru dan pendidik lainnya, serta tenaga kependidikan;
d. Koordinasi pelaksanaan tugas, pembinaan, dan pemberian dukungan administrasi kepada seluruh unsur organisasi di lingkungan Kementerian Pendidikan dan Kebudayaan;
e. Pengelolaan barang milik/kekayaan negara yang menjadi tanggung jawab Kementerian Pendidikan dan Kebudayaan;
f. Pengawasan atas pelaksanaan tugas di lingkungan Kementerian Pendidikan dan Kebudayaan;
g. Pelaksanaan bimbingan teknis dan supervisi atas pelaksanaan urusan Kementerian Pendidikan dan Kebudayaan di daerah;
h. Pelaksanaan pengembangan, pembinaan, dan pelindungan bahasa dan sastra;
i. Pelaksanaan penelitian dan pengembangan di bidang pendidikan anak usia dini, pendidikan dasar, pendidikan menengah, dan pendidikan masyarakat, serta kebudayaan; dan
j. Pelaksanaan dukungan substantif kepada seluruh unsur organisasi di lingkungan Kementerian Pendidikan dan Kebudayaan. [3]

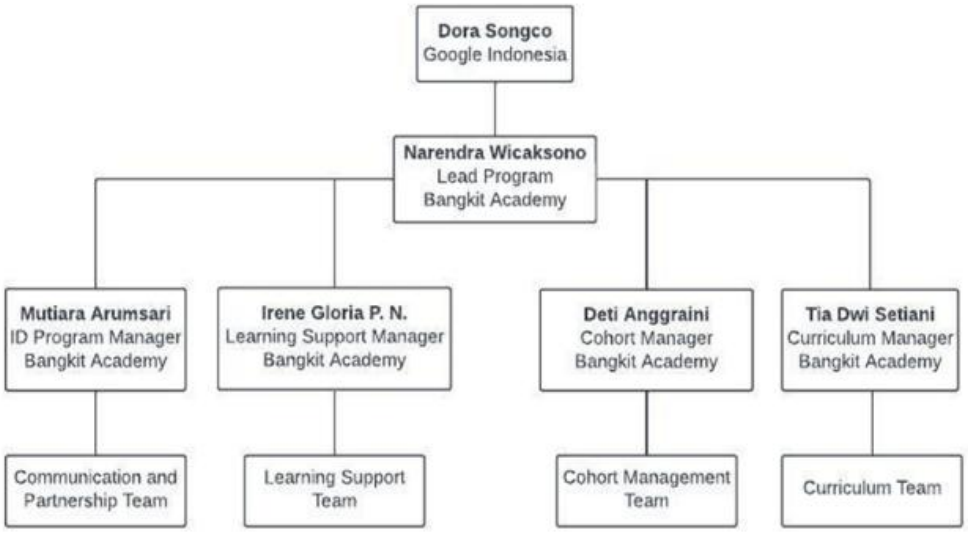
2.2.5. Logo Bangkit *Academy*

Berikut ini merupakan logo dari Bangkit *Academy* yang menyertkan juga logo beberapa Perusahaan yang menandakan bukti kerja sama mereka dengan Bangkit *Academy*.

Gambar 2.2 Logo Bangkit *Academy*

2.2.6. Struktur Organisasi Bangkit *Academy*

Berikut ini merupakan struktur organisasi pada Bangkit *Academy* dengan pembagian tanggung jawab kerja bagi setiap anggota.



Gambar 2.3 Struktur Organisasi Bangkit *Academy*

Struktur organisasi Bangkit *Academy* dirancang dengan cermat untuk memastikan efektivitas dan efisiensi dalam penyelenggaraan program-program pendidikan dan pengembangan sumber daya manusia di bidang teknologi. Dalam struktur ini, setiap anggota memiliki tanggung jawab yang jelas sesuai dengan bidang keahliannya masing-masing. Kepemimpinan yang kuat dipegang oleh seorang Direktur Utama yang bertanggung jawab atas arah strategis dan pengelolaan keseluruhan operasional. Di bawahnya, terdapat divisi-divisi yang terfokus pada aspek-aspek kunci seperti pengembangan kurikulum, rekruitmen tenaga pengajar, pengelolaan *platform e-learning*, serta hubungan masyarakat dan pemasaran. Masing-masing divisi dipimpin oleh seorang manajer yang memiliki pengalaman dan keahlian di bidangnya. Selain itu, ada juga tim teknis yang bertanggung jawab atas pemeliharaan dan pengembangan teknologi informasi yang mendukung jalannya program-program belajar mengajar secara *online*. Tak kalah pentingnya, terdapat pula tim penjaminan mutu yang bertugas memastikan bahwa setiap aspek dari program pendidikan yang diselenggarakan sesuai dengan standar yang ditetapkan. Dengan struktur organisasi yang terencana dengan baik dan pembagian tanggung jawab yang jelas, Bangkit *Academy* siap untuk terus berkembang dan memberikan kontribusi yang signifikan dalam pengembangan sumber daya manusia di Indonesia dalam bidang teknologi dan digitalisasi.

2.3. Lingkup Pekerjaan yang dilakukan

Selama menjalani kerja praktik di Bangkit *Academy* dengan fokus pada pembelajaran dalam bidang *Cloud Computing*, terlibat dalam serangkaian kegiatan yang bertujuan untuk memperdalam pemahaman dan keterampilan dalam lingkup ini. Salah satu kegiatan utama adalah menghadiri konsultasi mingguan bersama mentor, di mana peserta berdiskusi tentang kemajuan proyek, tantangan yang dihadapi, dan strategi yang dapat diterapkan untuk mengatasi masalah yang muncul. Melalui konsultasi ini, mendapatkan bimbingan langsung dari mentor yang berpengalaman dalam mengatasi berbagai aspek teknis maupun non-teknis dalam pengembangan proyek.

Selain itu, juga mengikuti berbagai kelas yang berkaitan dengan *Cloud Computing*, baik yang diselenggarakan oleh Bangkit *Academy* maupun melalui *platform* pembelajaran *online* seperti *Dicoding*, *Coursera*, dan *Google Cloud Skill Boost*. Materi pembelajaran dari berbagai sumber ini mencakup konsep dasar *Cloud Computing*, penggunaan *platform Cloud* secara efektif, praktik terbaik dalam pengelolaan infrastruktur *Cloud*, serta pengembangan aplikasi yang dioptimalkan untuk *Cloud*. Juga mengikuti kelas pengembangan *softskill* dan Bahasa Inggris yang dirancang untuk meningkatkan keterampilan interpersonal, kepemimpinan, dan kemampuan komunikasi, yang sangat penting dalam lingkungan profesional.

Selama proses pembelajaran, terlibat dalam pembuatan rancangan program dan penyusunan laporan *Capstone Project*. Dalam pembuatan rancangan program, mempertimbangkan berbagai faktor, termasuk kebutuhan pengguna, teknologi yang akan digunakan, arsitektur sistem, dan rencana implementasi. Setelah itu, terlibat dalam mengembangkan prototipe aplikasi dan melakukan pengujian untuk memastikan bahwa solusi yang diusulkan dapat berfungsi dengan baik. Laporan *Capstone Project* juga disusun dengan cermat, mencakup deskripsi proyek, tinjauan literatur, metodologi, hasil, analisis, dan rekomendasi untuk pengembangan selanjutnya.

Dalam penerapan materi pembelajaran dalam *Capstone Project*, fokus pada pengembangan *backend* aplikasi dan proses *deployment* data ke *Cloud*. Melalui serangkaian kegiatan ini, berhasil menggabungkan teori dan praktik dalam

pengembangan solusi *Cloud Computing* yang relevan dan efektif. Juga mengembangkan keterampilan kritis seperti pemecahan masalah, kolaborasi tim, dan komunikasi yang membantu menjadi profesional yang lebih komprehensif dan siap untuk tantangan di masa depan.

BAB III
LANDASAN TEORI

3.1. Teori Pendukung

Pada bagian ini akan membahas teori-teori yang menjadi landasan dalam pembuatan laporan penilitian Implementasi Layanan *Cloud Computing CI/CD* pada Aplikasi Pendeteksi Kepiting Soka yang diperoleh dari jurnal, artikel, dan penelitian lain yang memiliki bahasan serupa.

3.1.1. Kepiting Soka

Kepiting Soka, atau dikenal juga sebagai kepiting cangkang lunak atau *soft-shell crab*, merujuk pada kepiting bakau yang telah mengalami proses *molting*. *Molting* adalah suatu tahap di mana kepiting melepaskan cangkangnya yang keras dan tumbuh cangkang yang baru yang masih lunak [3].

Gambar 3.1 Kepiting Soka

Proses *molting* memberikan kepiting sifat cangkang yang lebih lembut dan memungkinkan untuk dimakan secara utuh, tanpa perlu melepaskan bagian keras yang biasanya sulit diakses atau dikonsumsi. Ini menjadikan kepiting soka sangat diminati dalam berbagai masakan karena keterjangkauannya dan rasa dagingnya yang lembut. Kepiting Soka memiliki daya tarik tersendiri di dunia kuliner, terutama karena proses *molting* ini yang membedakannya dari kepiting bakau biasa. Dalam hidangan-hidangan kuliner, kepiting soka sering menjadi pilihan yang disukai karena kepraktisannya dan kemudahan dalam mengonsumsinya.

Dengan daging yang lembut dan cangkang yang lebih mudah diolah, kepiting soka sering menjadi bahan utama dalam berbagai hidangan laut yang nikmat dan menggugah selera. Keunikan ini membuat kepiting soka menjadi salah satu favorit di dunia kuliner, memberikan pengalaman kuliner yang istimewa bagi para pecinta makanan laut.

3.1.2. Kepiting Bakau

Kepiting Bakau, yang dikenal dengan nama ilmiah *Scylla seratta*, merupakan hewan yang dapat ditemukan berkeliaran di kawasan hutan bakau atau *mangrove*. Hidupnya yang terutama berlokasi di wilayah-wilayah ini memungkinkan kepiting bakau untuk memanfaatkan ekosistem *mangrove* sebagai tempat tinggal dan sumber daya makanan. Salah satu ciri khas utama kepiting bakau adalah tubuhnya yang dilapisi oleh cangkang keras, yang dikenal sebagai karapas. Karapas ini memberikan perlindungan kepada kepiting dari berbagai predator yang dapat mengancam keselamatannya [4].

Gambar 3.2 Kepiting Bakau

Tidak hanya itu, kepiting bakau juga memiliki dua capit yang berfungsi ganda sebagai alat untuk memburu mangsa dan sebagai alat pertahanan diri. Capit tersebut memungkinkan kepiting bakau untuk dengan mudah menangkap mangsa yang berada di sekitarnya, dan sekaligus berfungsi sebagai alat pertahanan diri apabila kepiting tersebut merasa terancam. Kehadiran capit ini memberikan

keunggulan kepada kepiting bakau dalam menjalani kehidupannya di lingkungan *mangrove* yang dinamis dan penuh tantangan [4].

Kepiting bakau seringkali ditemui di daerah perairan payau, yang mencakup wilayah-wilayah dengan perpaduan antara air tawar dan air laut. Adaptasi kepiting bakau terhadap perairan payau membuatnya dapat menghuni berbagai jenis lingkungan, mulai dari muara sungai hingga area pantai. Keberadaan kepiting bakau juga memiliki dampak positif terhadap ekosistem *mangrove*, karena mereka berperan sebagai pengatur populasi dan berkontribusi pada keselarasan ekosistem yang kompleks ini [4].

3.1.3. *Cloud Computing*

Cloud computing adalah model pengkomputeran yang memungkinkan akses dan penggunaan sumber daya komputasi (seperti server, penyimpanan, *database*, jaringan, perangkat lunak, analisis data, kecerdasan buatan, dan kekuatan komputasi) melalui internet, tanpa perlu memiliki infrastruktur fisik atau perangkat keras secara lokal.

Cloud Computing dapat dikategori menjadi tiga model, yakni: *Infrastructure-as-a-Service* (IaaS) merupakan salah satu model layanan dalam *cloud computing* yang menyediakan akses virtual ke sumber daya infrastruktur IT melalui internet. Kemudian, *Platform-as-a-Service* (PaaS) merupakan salah satu model layanan dalam *cloud computing* yang menyediakan *platform* pengembangan lengkap untuk membangun, menguji, dan menyebarkan aplikasi perangkat lunak tanpa perlu khawatir tentang kompleksitas infrastruktur di bawahnya. Dan yang terakhir adalah *Soft-as-a-Service* yang merupakan model layanan dalam *cloud computing* yang menyediakan akses ke perangkat lunak melalui internet. Dalam model ini, perangkat lunak tidak diinstal secara lokal pada perangkat pengguna, melainkan di-host dan dioperasikan oleh penyedia layanan *cloud*. [5]

3.1.4. *Continuous Integration*

Continuous Integration (CI) merupakan praktik penting dalam pengembangan perangkat lunak yang bertujuan untuk meningkatkan kualitas dan efisiensi pengembangan melalui otomatisasi dan integrasi kode secara terus-menerus.

Prinsip utama dari CI adalah menggabungkan perubahan kode dari berbagai pengembang ke dalam repositori bersama secara berkala, yang kemudian diuji secara otomatis. Dengan menerapkan CI, setiap kali ada perubahan kode, sistem secara otomatis membangun (*build*), menguji, dan mengintegrasikan perubahan tersebut dengan kode yang sudah ada. Hal ini memastikan bahwa setiap perubahan kode dapat diuji dan diintegrasikan dengan kode yang sudah ada tanpa menghasilkan konflik atau masalah yang dapat mengganggu kelancaran pengembangan perangkat lunak [6].

Beberapa konsep inti dalam *Continuous Integration* meliputi otomatisasi proses pembangunan (*build*), pengujian otomatis, dan penggabungan kode (*merge*) secara teratur. Proses pembangunan (*build*) mengacu pada proses mengkompilasi kode menjadi program yang dapat dijalankan, dan dalam CI, proses ini sering kali diotomatisasi sehingga setiap kali ada perubahan kode, program dapat dibangun secara konsisten dan efisien. Pengujian otomatis juga menjadi komponen penting dalam CI, di mana serangkaian tes otomatis dapat dijalankan secara otomatis untuk memastikan bahwa perubahan kode tidak mengakibatkan regresi atau perubahan yang tidak diinginkan dalam fungsionalitas aplikasi. Selain itu, penggabungan kode (*merge*) secara teratur memastikan bahwa perubahan kode dari berbagai pengembang dapat digabungkan ke dalam repositori bersama dengan lancar dan tanpa konflik [6].

Dengan menerapkan CI, tim pengembangan dapat meningkatkan kualitas perangkat lunak, mengurangi risiko kesalahan, dan mempercepat siklus pengembangan secara keseluruhan. CI juga memungkinkan tim untuk lebih responsif terhadap perubahan dan meningkatkan kolaborasi antar anggota tim. Oleh karena itu, CI telah menjadi praktik standar dalam pengembangan perangkat lunak modern dan terus berkembang seiring dengan perkembangan teknologi dan praktik pengembangan yang baru[6].

3.1.5. *Continuous Deployment*

Continuous Deployment (CD) mewakili suatu pendekatan yang inovatif dalam praktik pengembangan perangkat lunak. Menjadi kelanjutan dari konsep *Continuous Integration* (CI), CD membawa proses ini ke tingkat berikutnya

dengan mengotomatiskan seluruh alur kerja, mulai dari pengujian hingga penyebaran aplikasi ke lingkungan produksi. Dengan penerapan CD, setiap kali ada perubahan kode yang diintegrasikan dan berhasil melewati pengujian dalam lingkungan pengembangan, perangkat lunak tersebut secara otomatis dan tanpa intervensi manusia langsung diterapkan ke lingkungan produksi [6].

Keunggulan utama dari *Continuous Deployment* adalah kemampuannya untuk menghasilkan pengembangan perangkat lunak yang lebih efisien dan responsif. Proses otomatisasi ini meminimalkan risiko kesalahan yang mungkin timbul akibat campur tangan manusia, sehingga menghasilkan penyebaran yang konsisten dan dapat diandalkan. Selain itu, CD juga memungkinkan tim pengembang untuk mendeteksi dan menanggapi perubahan atau masalah segera setelah terjadi, mempercepat siklus pengembangan dan meningkatkan kemampuan untuk memberikan pembaruan atau perbaikan dengan cepat kepada pengguna [6].

Dengan menerapkan *Continuous Deployment*, tim pengembang dapat mencapai tingkat efisiensi yang lebih tinggi, meningkatkan kualitas perangkat lunak, dan secara keseluruhan mengoptimalkan alur kerja pengembangan. Pendekatan ini menjadi kunci dalam dunia pengembangan perangkat lunak modern, di mana kecepatan, ketepatan, dan keandalan menjadi elemen-elemen kritis dalam memenuhi tuntutan pasar dan kebutuhan pengguna.

3.1.6. Google Cloud Platform

Google Cloud Platform (GCP) merupakan penyedia layanan *cloud computing* terkemuka yang dioperasikan oleh Google. *Platform* ini menyajikan beragam layanan termasuk komputasi, penyimpanan, basis data, kecerdasan buatan, analitika, dan banyak lagi. Google Cloud Platform memberikan kemampuan kepada pengguna untuk mengembangkan, mengelola, dan melakukan *deployment* aplikasi mereka menggunakan infrastruktur *cloud* yang dikelola oleh Google [7].

Sebagai salah satu pemimpin dalam industri *cloud computing*, Google Cloud Platform menawarkan infrastruktur global yang memungkinkan pengguna menyimpan, mengelola, dan mengakses sumber daya komputasi dan data secara efisien. Layanan-layanan yang disediakan oleh GCP mencakup berbagai

kebutuhan pengembangan dan pengelolaan aplikasi, mulai dari infrastruktur dasar hingga solusi tingkat lanjut seperti kecerdasan buatan dan analitika data [7].

Pengguna GCP dapat memanfaatkan keunggulan dari jaringan global Google yang luas, yang mencakup pusat data di berbagai lokasi di seluruh dunia. Hal ini membantu meningkatkan kinerja, keandalan, dan skalabilitas aplikasi yang dijalankan di atas *platform* ini. Selain itu, GCP terus melakukan inovasi dengan menyajikan berbagai layanan terbaru untuk memenuhi kebutuhan berkembang dalam dunia teknologi informasi. Dengan menyediakan berbagai fasilitas dan layanan, Google Cloud Platform menjadi pilihan populer bagi perusahaan dan pengembang untuk membangun dan mengelola aplikasi mereka di lingkungan *cloud*. GCP berperan penting dalam mendukung transformasi digital dan inovasi di berbagai industri, memanfaatkan potensi teknologi *cloud* untuk memenuhi tuntutan bisnis yang semakin kompleks dan dinamis [7].

3.1.7. GIT

GIT adalah sistem kontrol versi distribusi yang diperkenalkan oleh Linus Torvalds pada tahun 2005 untuk memonitor perubahan dalam kode sumber selama tahap pengembangan perangkat lunak. Sejak diperkenalkannya, GIT telah menjadi salah satu sistem kontrol versi yang paling diminati dan umum digunakan di seluruh dunia. Keunggulan utama GIT terletak pada kemampuannya untuk melacak perubahan kode secara efisien dan mengelola kode sumber dengan baik. Salah satu fitur utama yang membuat GIT begitu populer adalah kemampuannya dalam manajemen *branch*, yang memungkinkan pengembangan kode yang bersifat paralel dengan mudah. Dengan adanya fitur *branch* ini, tim pengembang dapat bekerja pada fitur atau perbaikan *bug* secara terpisah tanpa mengganggu kode yang sedang dikembangkan oleh anggota tim lainnya [8].

Selain itu, GIT juga menawarkan lingkungan pengembangan yang terdistribusi, yang memungkinkan pengembang untuk bekerja secara mandiri di repositori lokal mereka dan kemudian menggabungkan perubahan dengan repositori utama dengan lancar. Hal ini memfasilitasi kolaborasi tim yang efisien, terutama ketika anggota tim berada di lokasi yang berbeda atau bekerja pada fitur yang berbeda secara bersamaan [8].

Kemampuan GIT dalam melacak setiap perubahan kode, mengelola branch, dan mendukung kolaborasi tim membuatnya menjadi pilihan yang ideal untuk proyek-proyek pengembangan perangkat lunak yang kompleks dan berkelanjutan. Dengan adanya GIT, tim pengembang dapat bekerja dengan lebih efisien, mengurangi risiko konflik dalam pengembangan kode, dan memastikan kelancaran proses pengembangan perangkat lunak secara keseluruhan [8].

3.1.8. GitHub

GitHub adalah sebuah *platform* pengelolaan kode sumber yang telah menjadi sangat populer dan digunakan secara luas di kalangan komunitas pengembang perangkat lunak. Dengan menggunakan GitHub, pengembang perangkat lunak dapat dengan mudah berkolaborasi dalam pengembangan proyek, melakukan manajemen proyek, dan mengendalikan versi menggunakan sistem kontrol versi Git. Layanan berbasis web yang disediakan oleh GitHub memungkinkan tim pengembang untuk bekerja bersama secara efisien, menyimpan kode sumber proyek, dan memantau perubahan yang terjadi dalam kode sumber tersebut [8].

Salah satu fitur utama dari GitHub adalah kemampuannya untuk memfasilitasi kolaborasi dalam pengembangan perangkat lunak. Tim pengembang dapat bekerja secara bersama-sama dalam satu repositori proyek, melakukan perubahan, memberikan umpan balik, dan menggabungkan perubahan kode dengan mudah. Selain itu, GitHub juga menyediakan fitur manajemen proyek yang memungkinkan pengguna untuk mengatur tugas, mengelompokkan masalah, dan melacak perkembangan proyek secara keseluruhan [8].

Dengan menggunakan GitHub, pengembang perangkat lunak juga dapat memanfaatkan fitur pengendalian versi Git untuk melacak setiap perubahan yang terjadi dalam kode sumber proyek. Ini memungkinkan mereka untuk memahami evolusi proyek, melacak kontribusi dari berbagai pengembang, dan mengelola versi perangkat lunak dengan lebih efektif [8]. Secara keseluruhan, GitHub telah menjadi platform yang sangat penting dalam ekosistem pengembangan perangkat lunak saat ini. Dengan menyediakan berbagai fitur yang mendukung kolaborasi, manajemen proyek, dan pengendalian versi, GitHub membantu mempermudah

proses pengembangan perangkat lunak dan memfasilitasi pertukaran ide dan kontribusi antara pengembang perangkat lunak di seluruh dunia.

3.1.9. Firebase

Firebase, sebagai *platform* pengembangan aplikasi *mobile* dan *web* yang disajikan oleh Google, menawarkan sejumlah layanan dan alat yang sangat bermanfaat bagi para pengembang. Tujuannya adalah mempermudah proses pembangunan, pengelolaan, dan penyempurnaan aplikasi dengan mudah dan efisien. *Firebase* memberikan akses ke berbagai fitur yang mencakup, antara lain, *realtime database*, otentikasi pengguna, penyimpanan *file*, dan layanan *backend* tanpa server [1].

Salah satu fitur utama yang ditawarkan oleh *Firebase* adalah *realtime database*, yang memungkinkan pengembang membuat aplikasi yang responsif dengan data yang selalu terbaru secara langsung. Selain itu, otentikasi pengguna memudahkan implementasi sistem keamanan dengan menyediakan opsi otentikasi yang dapat disesuaikan untuk mengamankan akses ke aplikasi. *Firebase* juga menyediakan penyimpanan *file*, memungkinkan pengembang menyimpan dan mengelola berkas-berkas seperti gambar, *video*, atau dokumen secara mudah [1].

Selain itu, *Firebase* mencakup layanan *backend* tanpa server, yang memberikan infrastruktur yang dikelola secara otomatis oleh Google. Ini membantu menghilangkan kebutuhan untuk merancang dan mengelola server sendiri, mempercepat waktu pengembangan dan mengurangi kompleksitas pengelolaan infrastruktur. Dengan *Firebase*, pengembang dapat fokus pada pengembangan fitur-fitur kreatif dari aplikasi mereka tanpa harus terlalu terbebani oleh tugas-tugas teknis yang bersifat *backend* [1]. Dengan demikian, *Firebase* menjadi pilihan populer di kalangan pengembang karena menyediakan solusi terpadu dan lengkap untuk kebutuhan pengembangan aplikasi *mobile* dan *web*, memberikan dukungan yang efektif dari tahap awal pembangunan hingga pengelolaan dan penyempurnaan aplikasi yang sudah diluncurkan.

3.1.10. Python

Python merupakan sebuah bahasa pemrograman tingkat tinggi yang diperkenalkan pada awal 1990-an oleh Guido van Rossum, menekankan pada keterbacaan kode dan produktivitas pengembang. Dengan sintaksis yang sederhana dan mudah dipahami, Python merupakan pilihan yang cocok untuk pemula sekaligus mampu mengatasi pengembangan perangkat lunak yang kompleks.

Python mendukung paradigma pemrograman berorientasi objek, bersifat interpretatif namun dapat dikompilasi, dinamis, dan memiliki tipe data yang kuat. Keberadaan pustaka standar Python yang kaya dan ekosistem yang beragam menjadikannya pilihan favorit untuk berbagai keperluan, seperti pengembangan *web*, analisis data, dan kecerdasan buatan. Terkenal dengan filosofi desainnya yang dikenal sebagai "*The Zen of Python*," Python menonjolkan nilai-nilai keterbacaan dan keindahan dalam penulisan kode [9]. Dalam penelitian ini juga digunakan beberapa *library* Python yang membantu dalam proses pengerjaan, yakni:

3.1.10.1. Flask

Flask adalah sebuah kerangka kerja pengembangan *web* yang memiliki karakteristik ringan dan bersifat mikro, yang dirancang khusus untuk bahasa pemrograman Python. Diciptakan oleh Armin Ronacher, tujuan utama pembuatan Flask adalah untuk menyediakan kerangka kerja yang sederhana, mudah diterapkan, dan dapat dimengerti dengan cepat oleh para pengembang. Fokus utama dari Flask adalah memberikan kemudahan bagi pengembang dalam membangun aplikasi *web* tanpa membebani mereka dengan struktur atau fitur bawaan yang rumit [10]. Dengan sifat ringan dan mikro-nya, Flask memberikan kebebasan kepada pengembang untuk memilih dan menggabungkan komponen-komponen sesuai dengan kebutuhan proyek mereka. Hal ini memungkinkan pengembang untuk membangun aplikasi *web* dengan cara yang lebih modular dan sesuai dengan preferensi masing-masing, tanpa harus terikat pada aturan-aturan yang kaku.

Flask juga dikenal dengan dokumentasi yang baik dan ramah bagi pengembang, memudahkan pengguna untuk memahami dan menguasai kerangka kerja ini. Dengan pendekatan yang bersifat minimalis, Flask memungkinkan

pengembang untuk fokus pada logika bisnis aplikasi tanpa harus terjebak dalam kompleksitas struktur yang tidak diperlukan. Ini membuatnya sangat cocok untuk proyek-proyek kecil hingga menengah, *prototyping*, dan pengembangan aplikasi *web* yang membutuhkan tingkat kebebasan dan fleksibilitas yang tinggi. Secara keseluruhan, Flask menjadi pilihan yang populer di kalangan pengembang Python karena kesederhanaannya, kemudahan penerapan, dan fleksibilitasnya yang memungkinkan pengembangan aplikasi *web* yang efisien dan sesuai dengan kebutuhan [10].

3.1.10.2. Tensorflow

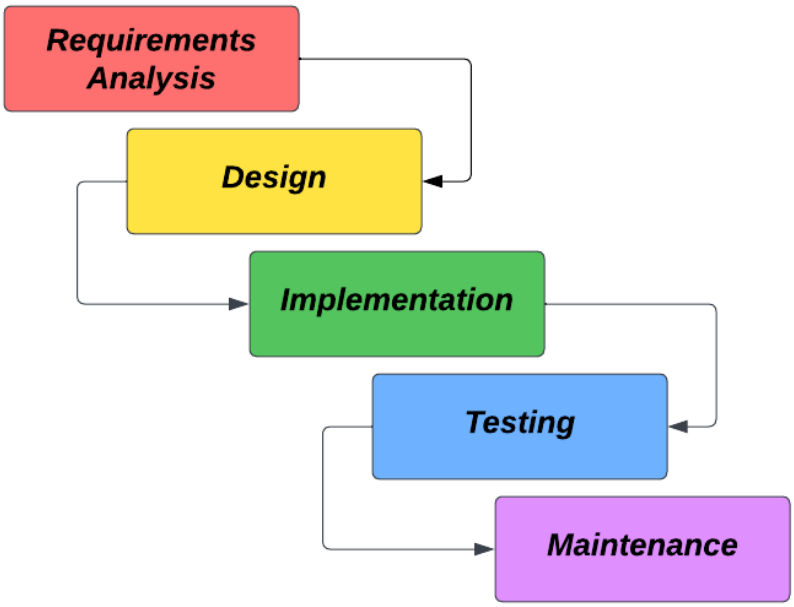
TensorFlow adalah sebuah pustaka sumber terbuka yang dikembangkan oleh tim Google Brain dengan tujuan mendukung berbagai aplikasi dalam pembelajaran mesin (*machine learning*) dan kecerdasan buatan (*artificial intelligence*). Pustaka ini menyajikan infrastruktur yang sangat efisien untuk perancangan dan pelatihan model kecerdasan buatan, terutama dalam implementasi jaringan saraf (*neural networks*) [11]. Dengan TensorFlow, pengembang dapat dengan mudah membuat dan mengimplementasikan model kecerdasan buatan yang kompleks melalui serangkaian API dan alat yang disediakan. Pustaka ini mendukung berbagai jenis model *machine learning*, mulai dari model sederhana hingga model *deep learning* yang lebih kompleks. TensorFlow menyediakan fleksibilitas yang tinggi dalam merancang arsitektur model, serta menyederhanakan proses pelatihan dan inferensi.

Salah satu keunggulan utama TensorFlow adalah kemampuannya untuk mengelola dan memanfaatkan perangkat keras akselerasi seperti GPU dan TPU, yang dapat meningkatkan kinerja pelatihan model secara signifikan. Dengan dukungan yang kuat dari komunitas pengembang dan berbagai tutorial serta sumber daya belajar, TensorFlow telah menjadi salah satu pilihan utama bagi para peneliti dan praktisi dalam dunia *machine learning* dan kecerdasan buatan. TensorFlow juga memperkenalkan mode eksekusi graf (*graph execution mode*), di mana pengguna mendefinisikan dan mengoptimalkan model melalui graf komputasi. Ini membantu dalam meningkatkan efisiensi dan mendukung penelitian dalam skala yang besar [11]. Oleh karena itu, TensorFlow telah menjadi

pilar penting dalam pengembangan aplikasi *machine learning* dan kecerdasan buatan, mempercepat kemajuan di bidang ini dan mendorong inovasi dalam berbagai industri.

3.2. Metodologi Pengembangan Sistem

Metodologi *waterfall*, atau yang dikenal sebagai model pengembangan *waterfall*, merupakan suatu pendekatan yang bersifat linier dan sekuensial dalam pengembangan perangkat lunak. Pendekatan ini merancang proyek ke dalam serangkaian tahapan terurut, dimulai dari tahap perencanaan hingga tahap pemeliharaan [12].



Gambar 3.3 Metodologi *Waterfall*

Kelebihan utamanya terletak pada struktur yang terdefinisi dengan jelas, yang mempermudah pemahaman dan pelaksanaan proyek secara sistematis. Metodologi ini dianggap sesuai untuk proyek-proyek dengan persyaratan yang stabil dan cenderung tidak mengalami banyak perubahan [12]. Meskipun model *waterfall* telah menjadi salah satu pendekatan yang paling awal dan paling banyak digunakan dalam pengembangan perangkat lunak, namun model ini juga memiliki keterbatasan, terutama dalam hal fleksibilitas dan responsivitas terhadap

perubahan kebutuhan pengguna. Namun, model *waterfall* tetap digunakan dalam beberapa konteks di mana persyaratan proyek sangat stabil dan perubahan minim.

3.2.1. Tahapan Metodologi *Waterfall*

Pada bagian ini akan menjelaskan tentang 5 tahapan yang ada dalam metodologi *Waterfall*.

1. *Requirements Analysis*

Pada tahapan ini akan dilakukan pengumpulan data untuk memahami kebutuhan target pengguna dan sistem. Analisis dapat dilakukan melalui metode seperti diskusi, wawancara, dan survei lapangan. Hasil analisis diolah untuk mendapatkan informasi dan persyaratan yang diperlukan sesuai dengan rumusan masalah. Tahap ini mencakup perencanaan aplikasi, fungsi, fitur, dan keluaran yang diinginkan. [12]

2. *Design*

Pada tahapan ini akan dilakukan perancangan desain tampilan, arsitektur sistem, dan alur aplikasi yang dibuat sesuai dengan hasil akhir dari tahapan sebelumnya. Hasil dari tahapan inilah yang akan diterapkan pada tahap berikutnya. [12]

3. *Implementation*

Pada tahapan ini akan dilakukan pengimplementasian dari desain yang telah dirancang pada tahap sebelumnya. Pada tahapan ini akan dilakukan pengkodean program sesuai dengan rancangan. [12]

4. *Testing*

Pada tahapan ini akan dilakukan pengujian dari hasil tahapan sebelumnya. Tahapan ini bertujuan untuk menguji dan memastikan aplikasi dapat berjalan sesuai dengan kebutuhan pengguna dan telah memenuhi tujuan pembuatan. [12]

5. *Maintenance*

Pada tahapan ini akan dilakukan perawatan secara rutin untuk memastikan bahwa aplikasi tetap berjalan sesuai dengan fungsionalitas. Namun dalam penelitian ini tidak akan dilanjutkan sampai ke tahapan ini. [12]

3.2.2. Kakas Pemodelan

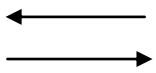
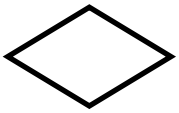
Berikut ini akan membahas mengenai kakas pemodelan yang digunakan dalam penelitian ini.

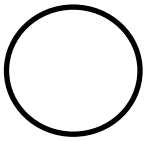
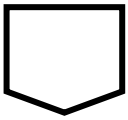
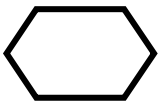
3.2.2.1. *Flowchart*

Flowchart adalah representasi visual yang menggambarkan urutan langkah-langkah atau proses dalam suatu sistem atau program komputer. Dengan menggunakan simbol dan panah, *flowchart* membantu mengilustrasikan aliran logika dari satu langkah atau keputusan ke langkah atau keputusan berikutnya. *Flowchart* sangat umum digunakan dalam pengembangan perangkat lunak, rekayasa proses bisnis, dan desain sistem [13].

Simbol-simbol standar yang digunakan dalam *flowchart* membentuk notasi yang telah distandarisasi untuk memastikan konsistensi pemahaman di seluruh industri. *Flowchart* dapat mencakup berbagai elemen seperti operasi matematika, kondisi atau keputusan, proses, *input/output*, dan banyak lagi. Setiap simbol memiliki makna khusus yang membantu menjelaskan langkah-langkah atau kondisi tertentu dalam alur logika [13].

Tabel 3.1 *Flowchart*

Simbol	Nama Simbol	Keterangan
	*Flow*	Merupakan symbol yang digunakan untuk menghubungkan simbol dengan simbol lainnya.
	*Terminator*	Merupakan simbol yang digunakan untuk mengawali dan mengakhiri alur *flowchart*.
	*Decision*	Merupakan simbol yang digunakan ketika akan dilakukan pemilihan.
	*Input/Output*	Merupakan simbol yang digunakan ketika akan dilakukan proses *input* dan *output*.
	*Process*	Merupakan simbol yang digunakan ketika akan dilakukan sebuah proses.

	*On-Page Reference*	Merupakan simbol yang akan digunakan untuk menghubungkan proses dalam satu halaman yang sama.
	*Off-Page Reference*	Merupakan simbol untuk menghubungkan proses dalam halaman yang berbeda.
	*Document*	Merupakan simbol untuk menyatakan *input* berasal dari dokumen dalam bentuk fisik atau *output* yang perlu dicetak.
	*Display*	Merupakan simbol yang menyatakan peralatan *output* yang digunakan.
	*Predefine Process*	Merupakan simbol yang digunakan untuk melaksanakan suatu bagian atau prosedur.
	*Preparation*	Merupakan simbol yang digunakan untuk menyatakan penyediaan tempat penyimpanan suatu pengolahan untuk memberikan nilai awal.

Keunggulan utama dari penggunaan *flowchart* adalah kemampuannya menyederhanakan pemahaman alur logika suatu proses atau sistem. Ini membantu pengembang, analis, dan pemangku kepentingan lainnya untuk memiliki pandangan visual yang jelas tentang bagaimana suatu sistem atau program berfungsi. Selain itu, *flowchart* berfungsi sebagai alat komunikasi yang efektif di antara anggota tim pengembangan dan pemangku kepentingan, memastikan bahwa semua pihak memiliki pemahaman yang konsisten tentang langkah-langkah atau proses yang diilustrasikan. Dengan menggunakan *flowchart*, tim pengembangan dapat memahami, merencanakan, dan mengkomunikasikan alur logika suatu program atau sistem secara efisien, membantu dalam pengambilan keputusan yang lebih baik dan memastikan keseluruhan proses berjalan dengan lancar dan efektif [13].

3.3. Prosedur Pengumpulan Data

Prosedur pengumpulan data merupakan langkah-langkah atau metode yang digunakan untuk mengumpulkan informasi atau data yang diperlukan dalam suatu penelitian. Prosedur ini penting karena pengumpulan data yang baik dan efektif adalah kunci untuk mendapatkan hasil penelitian yang valid dan dapat diandalkan.

3.3.1. Data Primer

Data primer merujuk pada informasi atau data yang diperoleh secara langsung dari sumber pertama oleh peneliti untuk keperluan penelitian atau analisis tertentu. Pengumpulan data primer dilakukan dengan tujuan khusus untuk memecahkan masalah atau menjawab pertanyaan penelitian yang diajukan. Dalam konteks penelitian, data primer sering kali dianggap sebagai sumber informasi yang orisinal dan langsung terkait dengan objek penelitian.

Sumber data primer adalah pihak atau entitas yang memberikan informasi secara langsung kepada peneliti atau tim penelitian. Hal ini dapat melibatkan interaksi langsung dengan responden, observasi langsung, atau pengumpulan data langsung dari sumber asli tanpa melibatkan pihak ketiga. Misalnya, jika penelitian berkaitan dengan perilaku konsumen, data primer dapat diperoleh melalui survei langsung, wawancara, atau observasi perilaku langsung di lapangan.

Kelebihan penggunaan data primer melibatkan kontrol yang lebih besar atas jenis informasi yang dikumpulkan dan cara pengumpulan data. Peneliti dapat merancang instrumen pengumpulan data dengan cermat sesuai dengan tujuan penelitian dan memastikan bahwa data yang diperoleh relevan dan akurat. Namun, pengumpulan data primer juga dapat menjadi lebih mahal dan memerlukan lebih banyak waktu daripada menggunakan data sekunder, yang merupakan data yang sudah ada dan dikumpulkan oleh pihak lain sebelumnya.

Dengan demikian, penggunaan data primer merupakan pendekatan yang penting dalam penelitian karena memungkinkan peneliti untuk mengumpulkan informasi yang sesuai dengan tujuan penelitian mereka dan memiliki kontrol lebih besar terhadap proses pengumpulan data.

3.3.1.1. Observasi

Observasi merupakan metode penelitian yang melibatkan pengamatan langsung terhadap perilaku, kejadian, atau fenomena tanpa campur tangan atau manipulasi situasi. Dalam konteks penelitian atau ilmiah, observasi digunakan dengan tujuan mendapatkan data, informasi, atau pemahaman lebih mendalam mengenai suatu objek atau situasi yang menjadi fokus pengamatan.

Keunggulan utama dari metode observasi terletak pada kemampuannya untuk mengumpulkan data secara *real-time*. Dengan mengamati peristiwa atau perilaku langsung, peneliti dapat meraih informasi yang mungkin sulit diakses atau tidak dapat diperoleh melalui metode penelitian lainnya. Observasi juga memungkinkan peneliti untuk memahami konteks dan dinamika suatu kejadian secara menyeluruh, tanpa adanya intervensi yang dapat memengaruhi hasil.

Namun, seperti halnya metode penelitian lainnya, observasi juga memiliki batasan. Salah satu batasan utama adalah potensi pengaruh peneliti terhadap situasi atau subjek yang diamati. Kesadaran subjek terhadap adanya observasi dapat memengaruhi perilaku mereka, yang dapat menghasilkan data yang tidak sepenuhnya representatif. Selain itu, observasi mungkin sulit untuk menggambarkan proses mental atau motivasi yang mendasari perilaku yang diamati. Aspek-aspek internal seperti pemikiran, perasaan, atau motivasi seringkali sulit untuk diobservasi secara langsung.

Meskipun memiliki kelebihan dan batasan, observasi tetap menjadi metode yang berharga dalam penelitian, terutama ketika tujuan penelitian adalah memahami dan merekam perilaku secara mendalam. Penggunaan teknologi, seperti kamera pengamatan atau rekaman *video*, dapat membantu mengurangi beberapa batasan yang terkait dengan observasi langsung dan meningkatkan objektivitas dalam mengumpulkan data.

3.3.2. Data Sekunder

Data sekunder merujuk pada informasi atau data yang telah dikumpulkan oleh entitas lain untuk tujuan yang tidak terkait dengan penelitian atau analisis yang sedang dijalankan oleh peneliti tertentu. Data ini telah ada sebelumnya dan diperoleh dari sumber-sumber eksternal seperti literatur, publikasi, basis data, atau

hasil penelitian sebelumnya. Peneliti menggunakan data sekunder sebagai fondasi informasi yang telah terdapat untuk melengkapi atau mendukung penelitian mereka yang sedang berlangsung.

Data ini dimanfaatkan kembali dengan tujuan menyokong temuan baru atau mendukung hipotesis yang diuji dalam konteks penelitian saat ini. Penggunaan data sekunder dianggap efisien dalam hal waktu dan biaya, namun peneliti perlu melakukan analisis dan interpretasi data tersebut sesuai dengan kerangka penelitian yang mereka terapkan. Kesesuaian dan keakuratan data sekunder harus diperhatikan untuk memastikan relevansinya terhadap pertanyaan penelitian yang diajukan.

BAB IV
PEMBAHASAN

4.1. *Requirements Analysis*

Pada bagian *Requirements Analysis* dalam motodologi *Waterfall* merupakan langkah awal yang sangat penting. Tujuannya adalah mendefinisikan dan memahami kebutuhan bisnis serta pengguna sebelum proses pengembangan perangkat lunak dimulai. Tahap ini melibatkan identifikasi *stakeholder*, analisis kebutuhan, dan pembuatan dokumen persyaratan yang melibatkan validasi pihak terkait. Verifikasi persyaratan dilakukan untuk memastikan kesesuaian dengan kebutuhan sebenarnya. Dengan dasar yang kokoh ini, pengembang dapat melanjutkan ke tahap berikutnya dengan keyakinan bahwa proyek sesuai dengan kebutuhan yang terdefinisi dengan baik.

4.1.1. Pengumpulan Data

Dalam perjalanan pembuatannya, aplikasi pendeteksi kepiting soka ini dibangun oleh tiga kelompok, yakni kelompok Machine Learning, C*loud Computing*, dan *Mobile Development*, di mana setiap kelompok memiliki tugas dan tanggung jawabnya masing-masing. Dalam konteks ini, penulis bertanggung jawab di kelompok *cloud computing*, fokus pada pembuatan *backend service*. Oleh karena itu, data yang akan dikumpulkan bersifat sekunder dan terkait dengan implementasi layanan *cloud computing* CI/CD pada aplikasi pendeteksi kepiting soka.

Penerapan *Continuous Integration/Continuous Deployment* (CI/CD) pada aplikasi pendeteksi kepiting soka membawa berbagai manfaat dalam mempercepat dan meningkatkan kualitas pengembangan perangkat lunak. CI/CD merupakan pendekatan otomatisasi yang melibatkan serangkaian langkah, termasuk pengujian dan penyebaran kode, untuk memastikan integrasi yang mulus dan penyebaran yang efisien ke lingkungan produksi.

Dalam konteks CI/CD, setiap kali terjadi perubahan kode pada aplikasi, proses integrasi otomatis (*Continuous Integration*) memastikan bahwa kode tersebut diuji secara menyeluruh untuk mendeteksi potensi masalah. Pengujiannya melibatkan berbagai jenis pengujian, seperti pengujian unit, integrasi, dan fungsional, untuk

memastikan bahwa perubahan tidak menyebabkan kerusakan pada aplikasi. Proses ini dapat dilakukan secara otomatis setiap kali ada pembaruan, meminimalkan risiko kesalahan dan memastikan kualitas perangkat lunak.

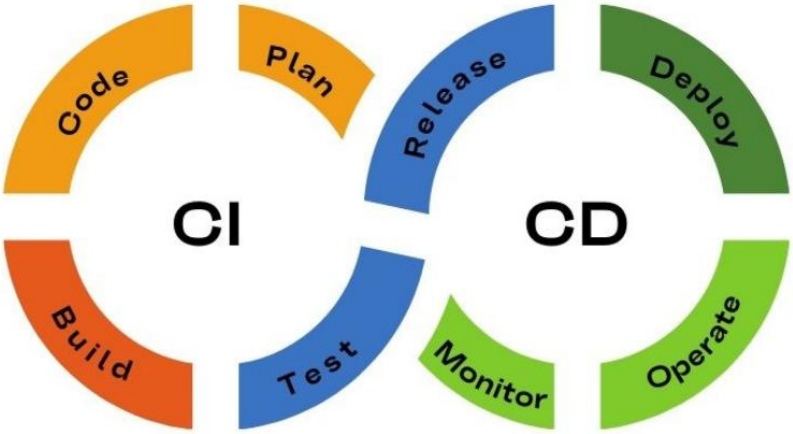
Setelah proses integrasi berhasil, langkah berikutnya adalah *Continuous Deployment*. Ini adalah fase di mana perubahan kode yang telah diuji secara otomatis diterapkan secara langsung ke lingkungan produksi. CI/CD memastikan bahwa penyebaran ini dilakukan dengan cepat dan dapat diulang secara konsisten, sehingga meminimalkan waktu rilis dan meningkatkan responsivitas terhadap perubahan pasar atau kebutuhan pengguna.

Keuntungan CI/CD dalam konteks aplikasi pendeteksi kepiting soka meliputi peningkatan kecepatan pengembangan, deteksi dini terhadap masalah, dan memungkinkan pengembang untuk lebih fokus pada peningkatan fitur daripada menghabiskan waktu ada proses manual yang dapat memakan waktu. Dengan adopsi CI/CD, tim pengembangan dapat merespons perubahan dan menyebarluaskan perbaikan ke lingkungan produksi dengan lebih efisien, mendukung siklus hidup pengembangan perangkat lunak yang adaptif dan responsif.

4.1.2. Analisis dan Pemecahan Masalah

Pada bagian proses analisis ini dilakukan untuk memberikan saran dan solusi bagi para pengembang perangkat lunak pada bagian pengolahan layanan *Cloud* terutama dalam penggunaan metode CI/CD. Permasalahan yang ditemukan dalam penelitian ini adalah kurangnya keefektifan dan efisien dalam penggunaan layanan *Cloud* untuk pengembangan suatau perangkat lunak.

Seperti contoh dalam pengembangan aplikasi pendeteksi kepiting soka berbasis Android yang menggunakan *firebase* sebagai penyedia *Cloud* untuk melakukan *authentification* dan sebagai *database*, menggunakan layanan terpisah dalam pengimplementasiannya sehingga membatasi perubahan fitur-fitur tertentu dan kurang memiliki kontrol penuh atas infrastruktur dan konfigurasi tingkat rendah dibandingkan dengan menggunakan solusi *cloud* yang lebih kustomisasi.



Gambar 4.1 Metode CI/CD

Dengan menerapkan CI/CD, perubahan kode dapat diuji secara otomatis dan diterapkan ke lingkungan produksi dengan cepat dan aman. Ini membantu dalam meningkatkan kualitas perangkat lunak, mendeteksi dini bug, dan mempercepat proses pengiriman perangkat lunak.

4.1.3. Spesifikasi Persyaratan Perangkat Lunak

Pada bagian ini akan membahas persyaratan yang diperlukan dalam mengimplementasikan layanan *cloud computing* CI/CD pada aplikasi pendeteksi kepiting soka.

4.1.3.1. Persyaratan Fungsional

a. Sistem dapat melakukan proses pengujian, pembangunan, dan pengiriman perangkat lunak setiap kali ada perubahan kode secara otomatis.
b. Sistem dapat mengimplementasikan dan mengintegrasikan algoritma pendeteksi kepiting soka.
c. Sistem memiliki lapisan keamanan yang kokoh dan melindungi data dengan baik.

4.1.3.2. Persyaratan Non Fungsional

Berikut ini merupakan daftar spesifikasi persyaratan perangkat lunak secara non-fungsional.

1. Keamanan

Sistem harus memiliki tindakan keamanan yang kuat untuk melindungi integritas dan kerahasiaan kode sumber.

2. Skalabilitas

   Sistem harus dapat diskalakan untuk menangani pertumbuhan volume dan kompleksitas proyek.

3. Kinerja

   Sistem harus memberikan kinerja yang optimal dalam pelaksanaan CI/CD dan proses pendeteksian kepiting soka.

4.1.4. Ruang Lingkup Proyek

Pada bagian ini akan membahas hal-hal yang merupakan ruang lingkup dari penelitian ini:

1. Penelitian ini merupakan gagasan produk untuk *Capstone Project* selama mengikuti Kerja Praktik di Bangkit *Academy*.
2. Penelitian ini merupakan pengembangan pada bagian layanan *Cloud*.
3. Penelitian ini menggunakan layanan *Cloud* GitHub dan Google Cloud Platform (*App engine*).

4.2. *Design*

Pada bagian ini akan menjelaskan tentang proses perancangan sistem layanan *Cloud* yang akan dibangun. Perancangan sistem akan digambarkan dengan menggunakan *Flowchart*.

4.2.1. Pemodelan Sistem

Pemodelan sistem merupakan proses yang melibatkan definisi, perancangan, dan penggambaran suatu sistem dengan memanfaatkan representasi visual atau abstraksi. Tujuan utama dari pemodelan sistem adalah menyajikan gambaran yang sistematis dan terstruktur mengenai operasi, interaksi, serta kontribusi komponen-komponennya dalam menyelesaikan suatu masalah atau memenuhi kebutuhan tertentu. Melalui pendekatan ini, pemodelan sistem menjadi alat penting untuk pemahaman, analisis, dan perencanaan yang efektif terkait dengan perilaku dan dinamika suatu sistem.

4.2.1.1. *Flowchart*

Berikut ini merupakan gambaran alur sistem yang akan digunakan untuk mengimplementasikan layanan *Cloud* pada aplikasi pendeteksi kepiting soka menggunakan *Flowchart*.

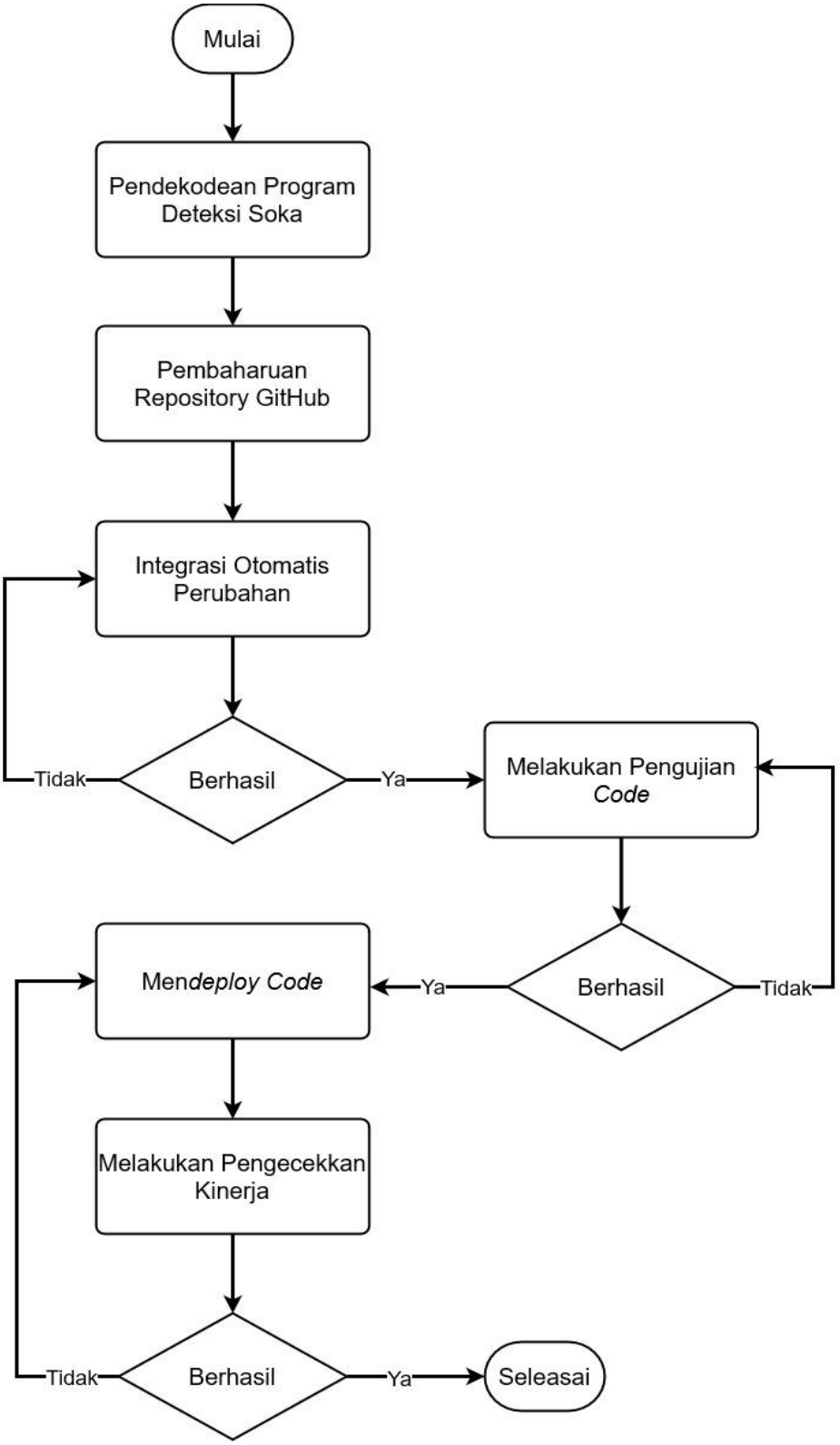
Pada Gambar 4.4 menunjukkan alur sistem metode CI/CD yang akan diimplementasikan pada aplikasi pendeteksi kepiting soka. Alur akan dimulai ketika pengembang melakukan proses pengkodean program kemudan melakukan *deploy code* di *Repository* GitHub yang telah dibuat, setelah itu akan dilakukan integrasi secara otomatis dan jika berhasil akan dilanjutkan pada proses selanjutnya, jika tidak berhasil akan mengulangi proses kembali.

Proses selanjutnya setelah tahap pengujian kode adalah pengecekan error dalam program. Sistem akan menjalankan serangkaian tes untuk memastikan bahwa tidak ada kesalahan atau bug yang terlewat. Jika tes tersebut berhasil, maka proses akan dilanjutkan ke tahap selanjutnya. Namun, jika terdeteksi kesalahan dalam program, sistem akan kembali ke tahap pengujian untuk memperbaiki dan menguji ulang kode tersebut.

Setelah kode dianggap telah lulus pengujian dan bebas dari kesalahan, langkah berikutnya adalah melakukan deploy kode pada layanan cloud development. Proses ini melibatkan penyalinan dan penerapan kode ke lingkungan cloud yang ditentukan untuk dijalankan dan diuji lebih lanjut.

Setelah kode berhasil dideploy, langkah selanjutnya adalah melakukan pengecekan kinerja sistem. Sistem akan mengevaluasi performa aplikasi yang telah di-deploy, termasuk responsivitas, kecepatan, dan keandalan. Jika pengecekan kinerja menunjukkan hasil yang memuaskan, maka proses akan dianggap selesai. Namun, jika terdapat masalah dalam kinerja sistem, sistem akan melakukan evaluasi lebih lanjut dan mungkin memerlukan langkah-langkah tambahan untuk memperbaikinya.

Jika terjadi kegagalan dalam pengecekan kinerja, sistem akan kembali ke tahap deploy kode untuk mengidentifikasi dan memperbaiki masalah yang terjadi. Setelah masalah tersebut diperbaiki, proses akan kembali ke tahap pengecekan kinerja untuk memastikan bahwa sistem berjalan dengan baik sebelum dianggap selesai.



Gambar 4.2 *Flowchart* Sistem CI/CD

Dengan demikian, proses pengujian, deploy, dan pengecekan kinerja merupakan serangkaian langkah yang penting dalam pengembangan perangkat

lunak untuk memastikan bahwa aplikasi yang dihasilkan memiliki kualitas yang baik dan dapat beroperasi dengan baik dalam lingkungan produksi.

4.3. *Implementation*

Pada bagian ini akan membahas tentang proses pengimplementasian layanan *cloud* CI/CD pada aplikasi pendeteksi kepiting dengan menggunakan GitHub dan Google Cloud Platform sebagai media implementasi CI/CD.

4.3.1. Lingkungan Implementasi

Pada bagian ini akan menjelaskan tentang lingkungan pengimplementasian bagian *hardware* dan *software* yang digunakan.

4.3.1.1. *Hardware*

Pada bagian ini akan menunjukkan *hardware* yang digunakan dalam proses pengimplementasian *cloud* CI/CD pada aplikasi pendeteksi kepiting soka.

Tabel 4.1 Lingkungan Implementasi *Hardware*

Nama Perangkat	Spesifikasi *Hardware*
LENOVO ThinkPad	- *Proccessor*: Intel(R) Core(TM) i5-8250U @1.60GHz (8 CPUs), ~1.8GHz. - *Memory*: 8192MB RAM. - *Operating System*: Windows 10 Pro 64-bit.

4.3.1.2. *Software*

Pada bagian ini akan menunjukkan *software* yang digunakan dalam proses pengimplementasian layanan *cloud* CI/CD pada aplikasi pendeteksi kepiting soka.

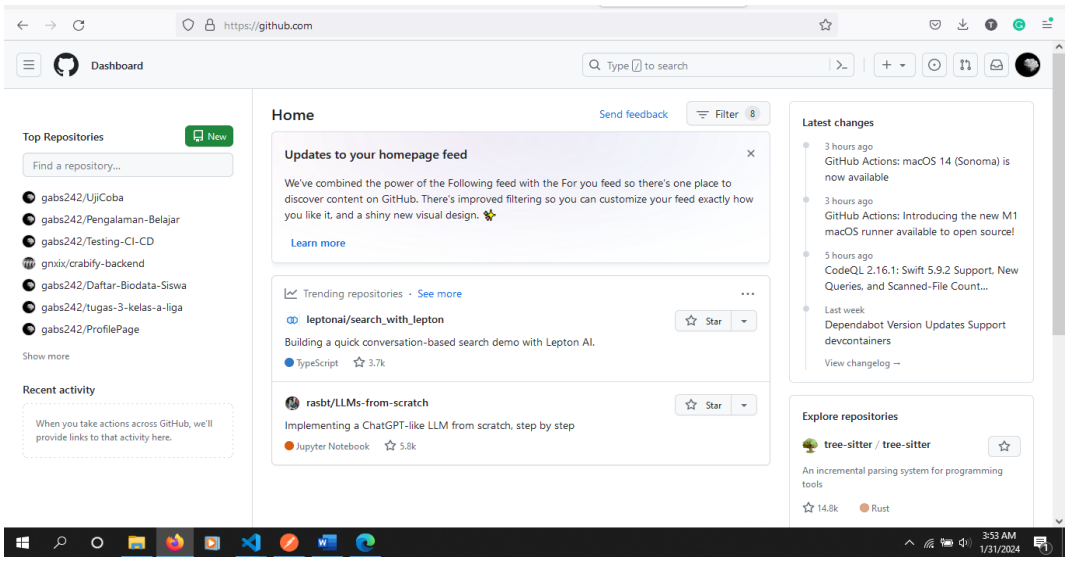
Tabel 4.2 Lingkungan Implementasi *Software*

Nama Perangkat	Kegunaan *Software*
Draw.io	Merancang diagram alur data aplikasi yang dibuat.
Visual Studio Code	Mebuat kode *backend*.
GitHub	Menjadi media implementasi metode *Continuous Integration* (CI).
Google Cloud Platform	Menjadi media implementasi metode *Continuous Deployment* (CD).

4.3.2. Implementasi *Continuous Integration* (CI)

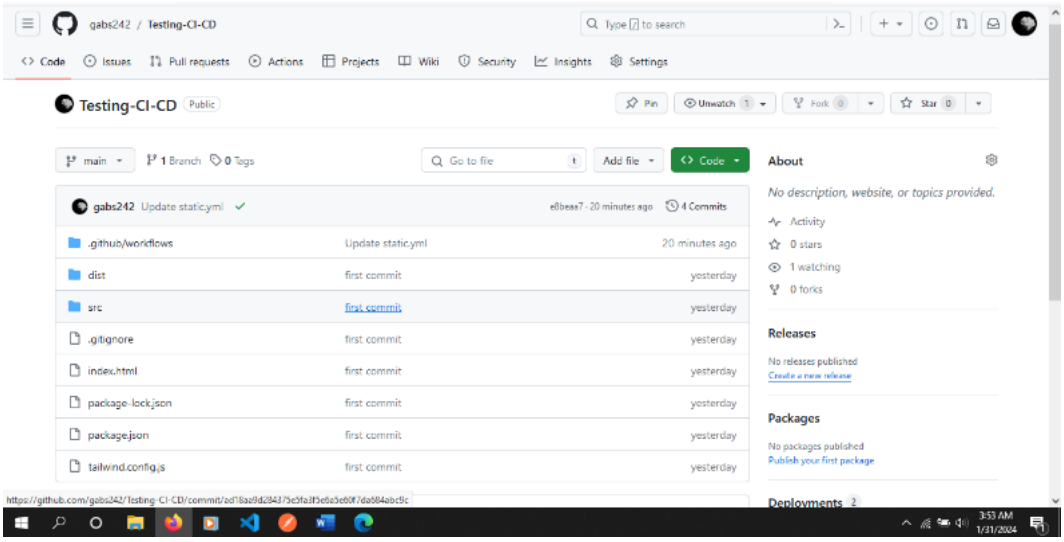
Bagian ini akan membahas secara rinci proses pengimplementasian layanan *Cloud Continuous Integration* (CI) dengan memanfaatkan *platform* GitHub sebagai media pengembangan. Proses ini melibatkan serangkaian langkah-langkah yang dirancang untuk mengintegrasikan CI ke dalam lingkungan pengembangan yang berbasis *cloud*, dengan GitHub sebagai pusat kontrol kode sumber.

Pada Gambar 4.5 menunjukkan tampilan halaman utama pada *website* GitHub yang menampilkan beberapa fitur, khususnya fitur *repository* yang menjadi media penyimpanan *code* program yang telah kita buat sebelumnya atau yang akan kita buat.



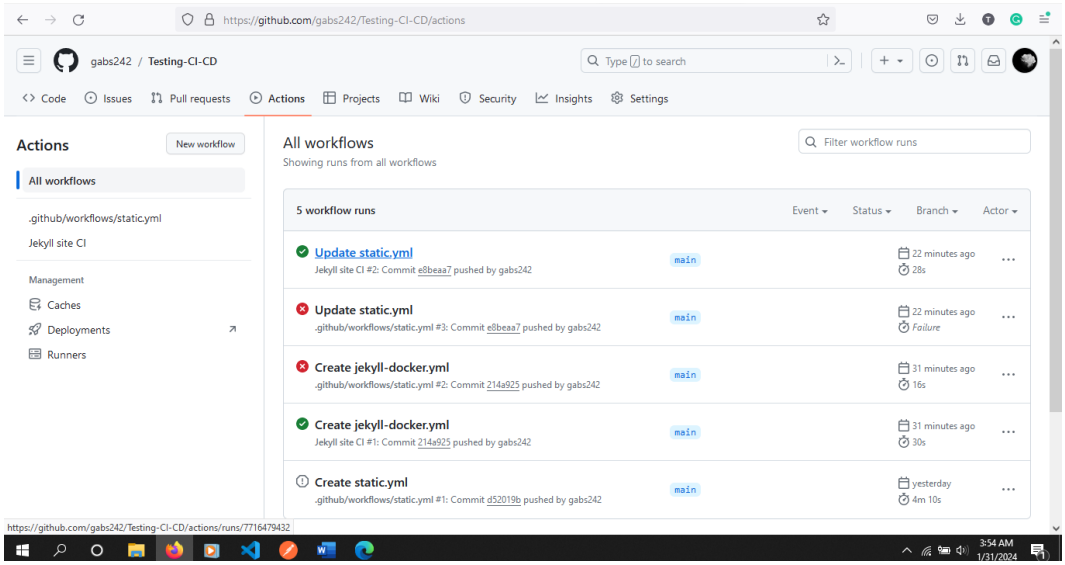
Gambar 4.3 *Dashboard* GitHub

Pada Gambar 4.6 menunjukkan tampilan dari salah satu *repository* yang terlibat dalam proses pengembangan layanan *cloud Continuous Integration* (CI). *Repository* ini berfungsi sebagai tempat penyimpanan kode sumber untuk proyek yang sedang dalam tahap pengembangan, dan antarmuka pengguna menampilkan beberapa *file* kode yang telah dibuat sebelumnya. Dalam konteks CI, *repository* ini mungkin berisikan berbagai *file*, termasuk kode aplikasi, skrip konfigurasi, atau *file* pendukung lainnya yang diperlukan untuk proses pembangunan dan eksekusi proyek. Pembaruan atau perubahan pada kode yang dilakukan oleh pengembang dapat terlihat dalam daftar *file* pada antarmuka *repository*.



Gambar 4.4 *Repository Testing CI*

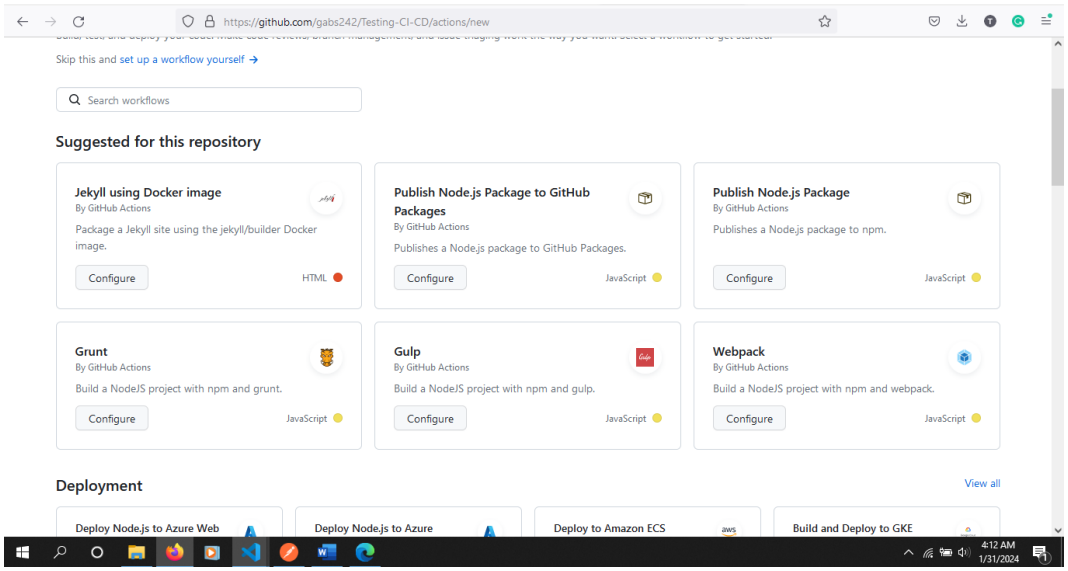
Pada Gambar 4.7, terdapat tampilan bagian *Actions* yang berfungsi sebagai pusat kendali untuk berbagai layanan dalam pengembangan perangkat lunak. Fokus utamanya adalah *Continuous Integration* (CI). Bagian ini memberikan akses cepat ke layanan-layanan penting yang mendukung otomatisasi pengujian, pembangunan proyek, dan pemantauan integrasi kode. Keberadaan ini memberikan keunggulan dalam meningkatkan efisiensi pengembangan, meningkatkan kualitas perangkat lunak, dan mendeteksi potensi masalah secara dini.



Gambar 4.5 Halaman Fitur *Actions*

4.3.2.1. Rekomendasi

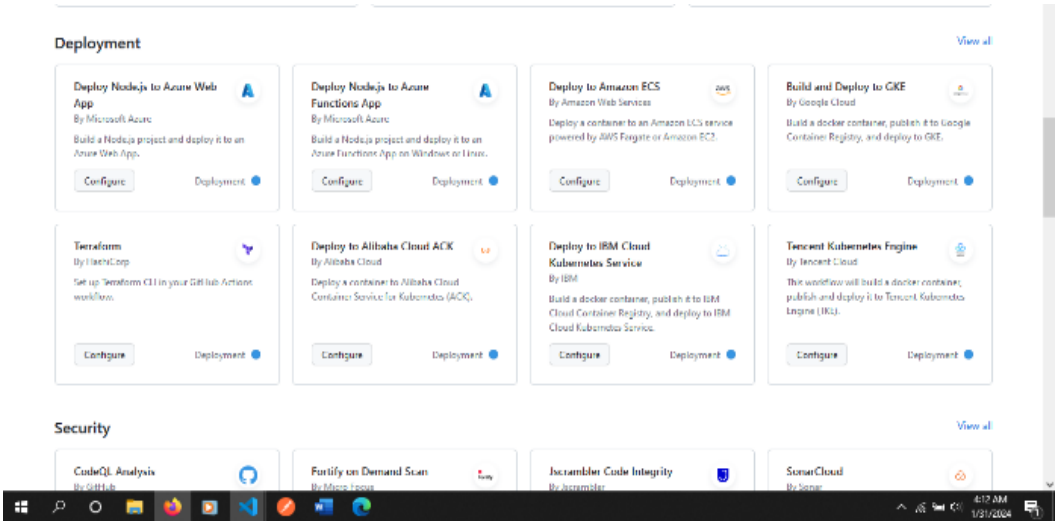
Pada Gambar 4.8 merupakan rekomendasi layanan sesuai jenis bahasa pemrograman atau sistem di *repository*. Pendekatan ini mempermudah pengembang dalam memilih layanan yang sesuai untuk pengembangan perangkat lunak, meningkatkan efisiensi, dan mendukung kolaborasi tim dengan panduan infrastruktur teknologi yang direkomendasikan.



Gambar 4.6 Bagian Rekomendasi

4.3.2.2. Layanan *Deployment*

Pada Gambar 4.9 memperlihatkan beberapa layanan terkait dengan implementasi perangkat lunak, seperti proses *deployment* Node.js ke Azure App dan penerapan di IBM *Cloud Kubernetes Service*.

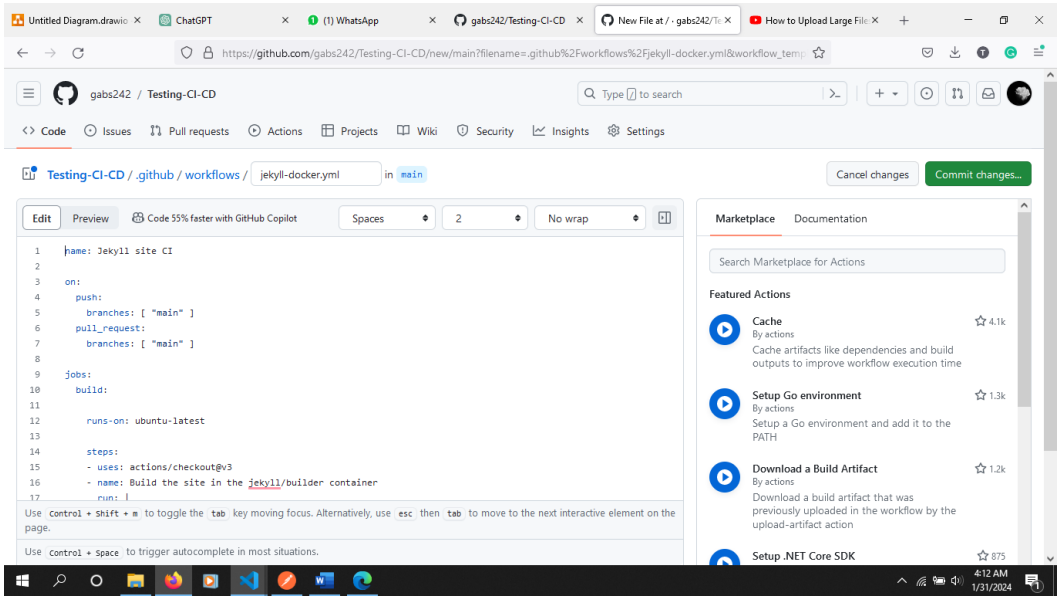


Gambar 4.7 Layanan *Deployment*

Layanan ini memberikan kemudahan bagi pengembang perangkat lunak dalam mengelola, mengonfigurasi, dan meluncurkan aplikasi pada lingkungan yang ditentukan. Pemilihan layanan ini juga dapat disesuaikan dengan kebutuhan spesifik proyek, memastikan pengembangan dan implementasi perangkat lunak berjalan secara efektif.

4.3.2.3. Pembuatan Layanan *Continuous Integration*

Pada Gambar 4.11, terlihat proses pembuatan file.yaml yang merupakan format serialisasi data yang bersifat manusiawi dan mudah dibaca oleh manusia. Meskipun YAML sering digunakan untuk tujuan konfigurasi, sebaiknya diingat bahwa itu bukanlah bahasa markup seperti XML atau HTML. Sebaliknya, YAML lebih mirip dengan bahasa pemrograman yang digunakan untuk menggambarkan data dalam bentuk teks.



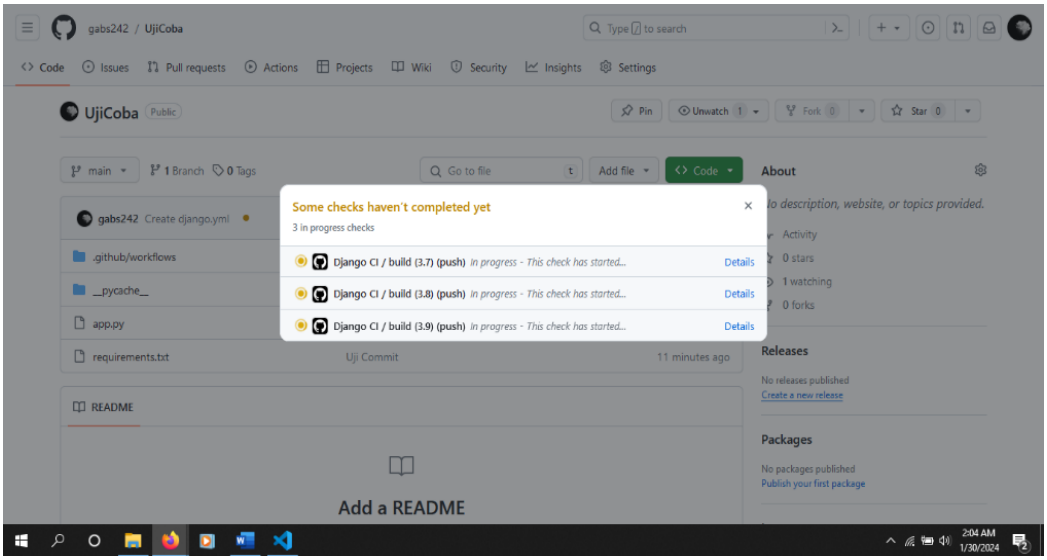
Gambar 4.8 Pembuatan file YAML

Kelebihan utama YAML terletak pada kemampuannya menyediakan representasi data yang bersifat manusiawi, sehingga memudahkan pembacaan dan pemahaman oleh manusia. Strukturnya yang bersifat hierarkis dan formatnya yang ringkas membuatnya menjadi pilihan yang populer untuk menyimpan konfigurasi dalam berbagai aplikasi. Selain itu, YAML juga mendukung jenis-jenis data yang

beragam, termasuk *list*, *dictionary*, dan tipe data lainnya, membuatnya fleksibel untuk digunakan dalam berbagai konteks.

Meskipun seringkali digunakan dalam konteks konfigurasi, YAML sebenarnya bukanlah bahasa markup. Sebaliknya, ia lebih mendekati bahasa pemrograman karena memungkinkan penggunaan variabel, referensi, dan bahkan beberapa ekspresi yang menyerupai kode pemrograman. Ini memberikan tingkat ekspresivitas yang tinggi, yang dapat berguna ketika menggambarkan struktur data yang kompleks atau konfigurasi yang rumit.

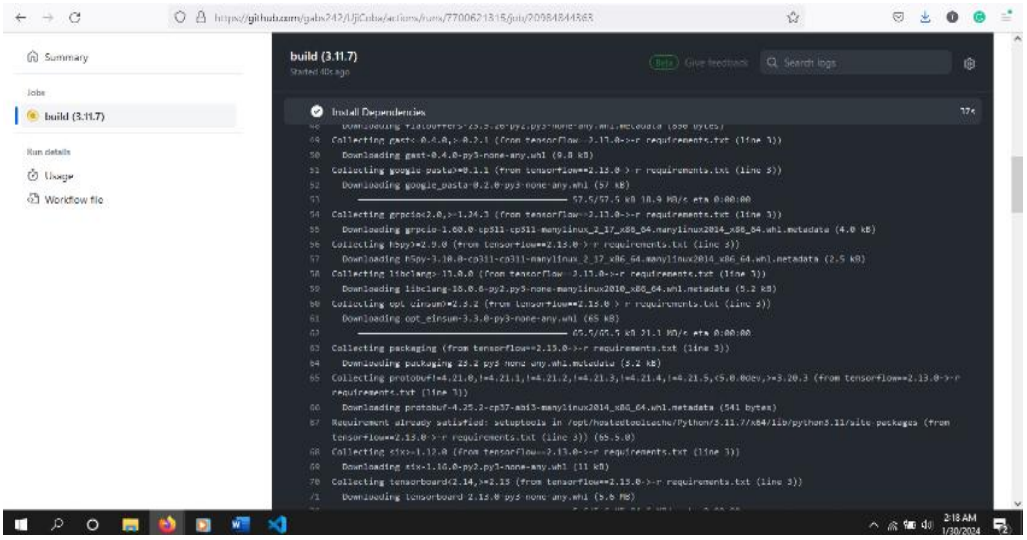
Setelah membuat file.yaml, *Continuous Integration* (CI) di GitHub melibatkan proses pengecekan otomatis setiap kali terjadi perubahan kode dalam repositori. Praktik ini bertujuan untuk memastikan bahwa setiap integrasi kode yang dilakukan oleh pengembang tidak merusak fungsionalitas eksisting atau menimbulkan konflik dengan bagian lain dari kode.



Gambar 4.9 Proses Pengecekan Otomatis

Dengan menggunakan uji otomatis, setiap modifikasi kode yang diusulkan atau diintegrasikan akan melewati serangkaian pengujian untuk memastikan integritas dan kualitasnya. CI memiliki peran penting dalam mendeteksi masalah lebih awal dalam siklus pengembangan perangkat lunak. Dengan melakukan pengecekan secara otomatis, setiap perubahan dapat segera dievaluasi, dan masalah yang mungkin muncul dapat diidentifikasi dan diperbaiki sebelum integrasi penuh ke

dalam proyek. Hal ini membantu mengurangi potensi dampak negatif terhadap kualitas perangkat lunak secara keseluruhan.

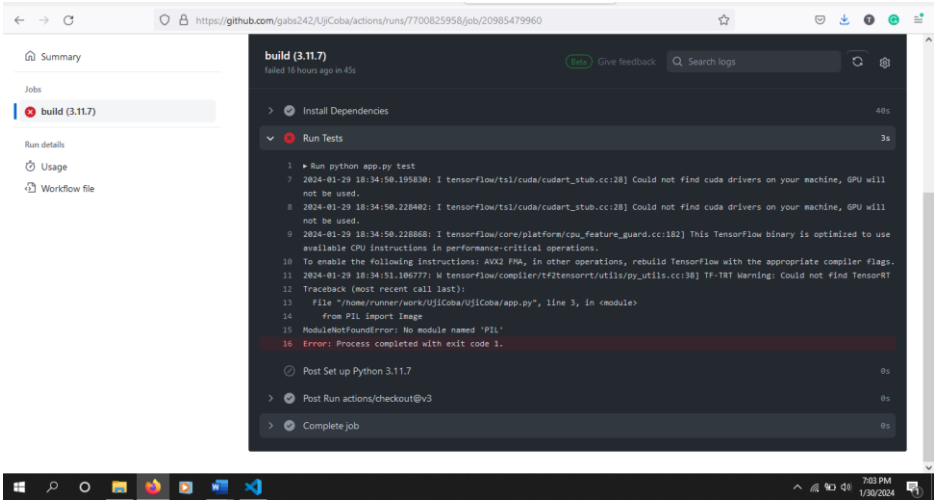


Gambar 4.10 Proses Pengecekan

Selain itu, CI juga berkontribusi pada peningkatan kolaborasi tim. Dengan adanya pengecekan otomatis, setiap anggota tim dapat lebih percaya diri dalam berkontribusi, karena mereka tahu bahwa perubahan yang mereka lakukan akan melalui proses evaluasi yang ketat. Hal ini menciptakan lingkungan yang lebih aman untuk berkolaborasi, mengurangi risiko ketidakcocokan atau konflik kode antar anggota tim.

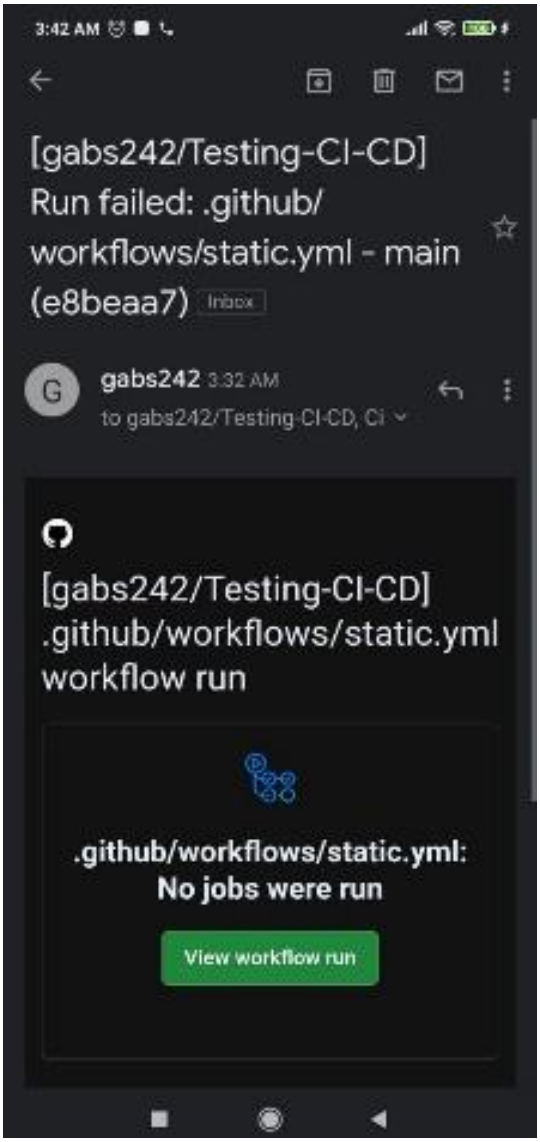
4.3.2.4. Pengecekan *Error*

Berikut ini merupakan tampilan ketika dalam proses pengecekan terdapat *code* yang *error*. Pada *code* dibawah ini memiliki *error* pada bagian modul yang tidak teridentifikasi.



Gambar 4.11 Pengecekan *Error*

Tujuan dari mekanisme ini adalah memungkinkan para pengembang untuk segera mengetahui masalah yang muncul dan meresponsnya dengan cepat, mempercepat proses identifikasi dan perbaikan *bug* dalam siklus pengembangan perangkat lunak. Dengan demikian, notifikasi melalui email menjadi salah satu fitur kritis yang mendukung transparansi dan efisiensi dalam pengelolaan kode sumber dalam lingkungan kolaboratif seperti GitHub.

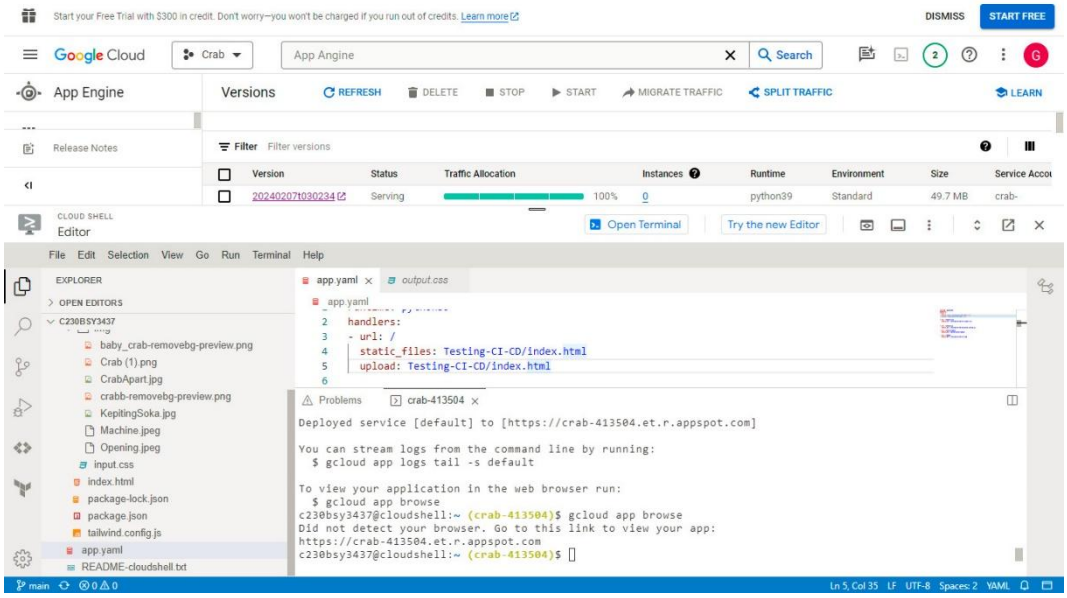


Gambar 4.12 *E-mail* Pesan *Error*

4.3.3. Implementasi *Continuous Deployment* (CD)

Pada tahap ini akan diimplementasikan *Continuous Deployment* (CD) pada aplikasi pendeteksi kepiting soka. Gambar 4.13 menunjukkan hasil implementasi *Continuous Deployment* (CD) pada *App Engine Google Cloud Platform*. *Continuous Deployment* adalah sebuah praktik dalam pengembangan perangkat lunak di mana perubahan kode secara otomatis dideploy ke lingkungan produksi setelah melewati serangkaian tes dan validasi. Dalam konteks *Google Cloud Platform*, penggunaan *App Engine* sebagai *platform* untuk menjalankan aplikasi memungkinkan integrasi yang mulus dengan alur kerja *Continuous Deployment*. Dengan implementasi *Continuous Deployment* pada *App Engine*, setiap kali

terjadi perubahan pada kode aplikasi, proses otomatis akan mulai berjalan. Langkah-langkah ini biasanya melibatkan kompilasi kode, menjalankan unit *test*, dan memastikan bahwa aplikasi berfungsi sebagaimana mestinya dalam lingkungan pengembangan. Setelah melewati semua tahapan uji dan validasi, aplikasi secara otomatis di *deploy* ke lingkungan produksi tanpa intervensi manual.



Gambar 4. 13 Hasil Implementasi *Deployment App Engine*

Manfaat utama dari *Continuous Deployment* adalah mempercepat siklus pengembangan dan pengiriman perangkat lunak. Hal ini juga memungkinkan pengujian kontinu di lingkungan yang serupa dengan produksi, sehingga meminimalkan potensi masalah saat aplikasi dijalankan secara langsung oleh pengguna akhir.

Dalam konteks *Google Cloud Platform, App Engine* menawarkan lingkungan yang dapat diandalkan dan skalabel untuk menjalankan aplikasi. Dengan memanfaatkan fitur-fitur *App Engine* seperti *autoscaling, load balancing*, dan manajemen sumber daya, implementasi *Continuous Deployment* dapat berjalan dengan lancar tanpa khawatir terhadap keandalan dan performa aplikasi. Dengan demikian, gambar 4.13 memberikan gambaran tentang bagaimana *Continuous Deployment* diimplementasikan secara efektif menggunakan *App Engine* di

Google Cloud Platform, memperkuat kemampuan pengembang untuk mengirimkan perubahan aplikasi secara cepat dan aman ke lingkungan produksi.

4.3.4. Implementasi Modul Program

Berikut ini merupakan implementasi modul program yang telah diimplementasikan pada aplikasi ini.

Tabel 4.3 menunjukkan modul klasifikasi YAML. Modul ini bertanggung jawab untuk mengatur konfigurasi klasifikasi menggunakan format YAML. Modul ini berisi pengaturan untuk parameter klasifikasi, seperti penggunaan algoritma klasifikasi, pengaturan preprocessing data, dan konfigurasi lainnya yang diperlukan untuk proses klasifikasi.

Tabel 4.3 Modul Klasifikasi YAML

Nama Modul	Kode Program

file.yaml	name: Jekyll site CI
	
	on:
	  push:
	  branches: ["main"]
	  pull_request:
	  branches: ["main"]
	
	jobs:
	  build:
	
	  runs-on: ubuntu-latest
	
	  steps:
	  - uses: actions/checkout@v3
	  - name: Build the site in the jekyll/builder container
	  run: |
	  docker run \
	  -v ${{ github.workspace }}:/srv/jekyll -v ${{

Nama Modul	Kode Program
	github.workspace }}/_site:/srv/jekyll/_site \ jekyll/builder:latest /bin/bash -c "chmod -R 777 /srv/jekyll && jekyll build --future"
Djago.yaml	name: Django CI on: push: branches: ["main"] pull_request: branches: ["main"] jobs: build: runs-on: ubuntu-latest strategy: max-parallel: 4 matrix: python-version: [3.11.7] steps: - uses: actions/checkout@v3 - name: Set up Python ${{ matrix.python-version }} uses: actions/setup-python@v3 with: python-version: ${{ matrix.python-version }} - name: Install Dependencies run: | python -m pip install --upgrade pip pip install -r requirements.txt - name: Run Tests

Nama Modul	Kode Program
	run: | python app.py test

Tabel 4.4 menunjukkan modul program klasifikasi. Modul ini merupakan inti dari aplikasi, yang berisi implementasi algoritma klasifikasi untuk mendeteksi kepiting soka. Di dalam modul ini, terdapat kode yang menjalankan proses klasifikasi berdasarkan konfigurasi yang telah ditentukan melalui modul klasifikasi YAML. Ini melibatkan pembacaan data, *preprocessing*, pelatihan model, dan evaluasi performa model.

Tabel 4.4 Modul Program Klasifikasi

Nama Modul	Kode Program
Import Library Python	from flask import Flask, request, jsonify from keras.models import load_model from PIL import Image import numpy as np from keras.preprocessing import image from keras.preprocessing.image import ImageDataGenerator from flask_cors import CORS
Flask *App Initialization*	app = Flask(__name__) CORS(app)
Load CNN *model*	model_cnn = load_model("model\model.h5")
Fungsi allowed_file	def allowed_file(filename): return '.' in filename and filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS
Konfigurasi *Data Augmentation*	datagen = ImageDataGenerator(rotation_range=40, shear_range=0.2, width_shift_range=0.2, height_shift_range=0.2, horizontal_flip=True,

Nama Modul	Kode Program
	fill_mode='nearest')

Fungsi *POST*	@app.route('/klasifikasi', methods=['POST'])
	def predict():
	try:
	# Get the uploaded image file
	image_file = request.files['image']
	
	if image_file and allowed_file(image_file.filename):
	
	# Prepare image for prediction
	img = Image.open(image_file)
	img = img.resize((300, 300))
	x = image.img_to_array(img)
	x = x / 255.0
	images = np.expand_dims(x, axis=0)
	
	datagen = ImageDataGenerator(
	rotation_range=40,
	shear_range=0.2,
	width_shift_range=0.2,
	height_shift_range=0.2,
	horizontal_flip=True,
	fill_mode='nearest'
	)

Generate augmented images	augmented_images = datagen.flow(images)
Perform predictions on the augmented images	prediction_array_cnn = model_cnn.predict(augmented_images) average_prediction = np.mean(prediction_array_cnn, axis=0)

Nama Modul	Kode Program
	class_names = ['Kepiting Biasa', 'Kepiting Soka']
Check confidence level	confidence_cnn = np.max(average_prediction) if confidence_cnn < confidence_threshold: return jsonify({"error": "Kepiting tidak terdeteksi dengan keyakinan yang cukup."}), 400

Format the response JSON	predictions = {
	"prediction_cnn": class_names[np.argmax(average_prediction)],
	"confidence_cnn": '{:2.0f}%'.format(100 * np.max(average_prediction)),
	}
	
	return jsonify(predictions)
	else:
	return jsonify({"error": "Invalid file format."}), 400
	except Exception as e:
	return jsonify({"error": str(e)}), 500

	@app.route('/', methods=['GET']) def hello(): return "Crabify"
Fungsi untuk *Endpoint Root*	@app.route('/', methods=['GET']) def hello(): return "Crabify"
API	if __name__ == '_main_': app.run(host='0.0.0.0', debug=True)

sTabel 4.5 menunjukkan modul *interface*. Modul ini bertanggung jawab untuk mengelola antarmuka pengguna aplikasi. Ini termasuk halaman web atau aplikasi mobile yang digunakan oleh pengguna untuk berinteraksi dengan aplikasi pendeteksi kepiting soka.

Tabel 4.5 Modul *Interface*

Nama Modul	Kode Program
Antar muka Aplikasi	<!DOCTYPE html> <html lang="en"> <head> <meta charset="UTF-8"> <meta name="viewport" content="width=device-width, initial-scale=1.0"> <title>Crabify</title> <link href="./dist/output.css" rel="stylesheet"> </head> <body class="font-serif"> <header class="bg-cyan-800"> <img src="./src/img/Crab (1).png" alt="" class="w-12 h-12"> <ul class="text-white flex space-x-2 pt-1"> <li class="text-xs">Home <li class="text-xs">About Crabify <li class="text-xs">Help <li class="text-xs">Contact </header> <main> <h1 class="text-white font-bold text-2xl">

Nama Modul	Kode Program
	Hi Sobat Craby!!! </h1> Pada Website ini kita akan membahas tentang "Kepiting Soka" Ayo Disimak!!! <img src="./src/img/Opening.jpeg" alt="" class=""> <h3 class="text-cyan-800 font-bold text-lg">Apakah Kamu Tahu Proses Molting?</h3> Sobat Craby! Molting adalah proses pelepasan Cangkang kepiting dikarenakan ukuran kepiting sudah lebih besar dari cangkangnya. <img src="./src/img/baby_crab-removebg-preview.png" alt="" > <h3 class="text-cyan-800 font-bold text-lg">Apakah Cangkang Tersebut Tidak Akan Muncul Kembali?</h3> Ohh tentu saja cangkang tersebut akan muncul kembali sobat Craby. Pertumbuhan cangkang kepiting akan kambali dalam kurun waktu tertentu tergantung dengan nutrisi kepiting, kualitas air, dan faktor genetik kepiting. <img src="./src/img/crabb-removebg-preview.png" alt=""> Jadi, untuk kepiting yang telah molting akan disebut dengan kepiting

Nama Modul	Kode Program
	soka atau kepiting cangkang lunak (sfot shell crab). <h3 class="text-cyan-800 font-bold text-lg mt-2">Tempat Pembudidayaan.</h3> Pembudidayaan Kepiting Soka dilakukan dengan menggunakan alat yang bernama Vertical Crab House yang berbentuk kotak yang disusun seperti apartemen, karena itu disebut juga dengan Apartemen Kepiting. <img src="./src/img/CrabApart.jpg" alt=""> <h3 class="text-cyan-800 font-bold text-lg">Proses Klasifikasi.</h3> Sobat Craby!!! Ingin mencoba Klasifikasi Kepiting? Ayo Kita Coba!!! <button class="bg-cyan-800 hover:bg-blue-700 text-white font-bold py-2 px-4 rounded"> Click Here! </button> </main> <footer class="bg-cyan-800"> Belajar Dasar Pemrograman Web © 2023, Gabriel

Nama Modul	Kode Program
	</footer> </body> </html>

Tabel 4.6 menunjukkan modul *deployment*. modul ini mencakup proses deployment aplikasi ke *platform cloud*, seperti *Google App Engine*. Ini termasuk konfigurasi lingkungan *deployment*, skrip otomatisasi *deployment*, dan pengaturan yang diperlukan untuk menjalankan aplikasi secara *online*. Modul ini bertanggung jawab untuk memastikan bahwa aplikasi dapat diakses dan digunakan oleh pengguna secara *online* setelah proses *deployment* selesai.

Tabel 4.6 Modul *Deployment*

Nama Modul	Kode Program
Deployment on Terminal	c230bsy3437@cloudshell:~ (crab-413504)$ gcloud app deploy Services to deploy: descriptor: [/home/c230bsy3437/app.yaml] source: [/home/c230bsy3437] target project: [crab-413504] target service: [default] target version: [20240207t030234] target url: [https://crab-413504.et.r.appspot.com] target service account: [crab-413504@appspot.gserviceaccount.com] Do you want to continue (Y/n)? y Beginning deployment of service [default]... Uploading 2 files to Google Cloud Storage 50% 100%

Nama Modul	Kode Program

	100%
	File upload done.
	Updating service [default]...done.
	Setting traffic split for service [default]...done.
	Deployed service [default] to [https://crab-413504.et.r.appspot.com]
	
	You can stream logs from the command line by running:
	$ gcloud app logs tail -s default
	
	To view your application in the web browser run:
	$ gcloud app browse
	c230bsy3437@cloudshell:~ (crab-413504)$ gcloud app browse
	Did not detect your browser. Go to this link to view your app:
	https://crab-413504.et.r.appspot.com

4.4. *Testing*

Pada tahap ini, akan merinci secara menyeluruh proses pengujian aplikasi yang telah dikembangkan untuk menilai kualitas layanan *cloud computing Continuous Integration/Continuous Deployment* (CI/CD) yang telah diimplementasikan di dalam aplikasi pendeteksi kepiting soka. Pengujian aplikasi menjadi tahap yang sangat penting dalam siklus pengembangan perangkat lunak, terutama ketika mengintegrasikan layanan *cloud computing* seperti CI/CD.

Dengan melakukan pengujian aplikasi yang komprehensif dan menyeluruh, kita dapat memastikan bahwa kualitas layanan cloud computing CI/CD yang diimplementasikan dalam aplikasi pendeteksi kepiting soka mencapai standar yang diharapkan dan memberikan pengalaman yang baik bagi pengguna akhir.

4.4.1. Tujuan Pengujian

Tujuan dari pengujian aplikasi adalah untuk memastikan bahwa aplikasi tersebut berjalan dan terealisasikan dengan baik sesuai dengan spesifikasi yang telah ditetapkan selama pengembangan. Pengujian ini juga berfungsi sebagai tolak ukur kelayakan aplikasi sebelum diberikan kepada pengguna akhir (*end-user*).

Selain itu, pengujian aplikasi membantu dalam mendeteksi masalah dan kesalahan yang mungkin muncul selama penggunaan aplikasi, sehingga dapat dilakukan perbaikan yang diperlukan untuk meningkatkan kualitas aplikasi. Dengan demikian, pengujian aplikasi menjadi langkah penting dalam siklus pengembangan perangkat lunak yang bertujuan untuk memastikan bahwa aplikasi yang dihasilkan memenuhi standar kualitas dan kebutuhan pengguna dengan baik sebelum diperkenalkan kepada pengguna akhir.

4.4.2. Kriteria Pengujian

Dalam melakukan pengujian implementasi layanan *cloud computing* CI/CD pada aplikasi pendeteksi kepiting soka, terdapat beberapa kriteria pengujian yang diperlukan, yaitu:

1. CI/CD yang diimplementasikan dapat melakukan proses *integration* dan *deployment*.
2. CI/CD dapat mengimplementasikan aplikasi yang di *deploy* pada *App Engine* dengan baik.

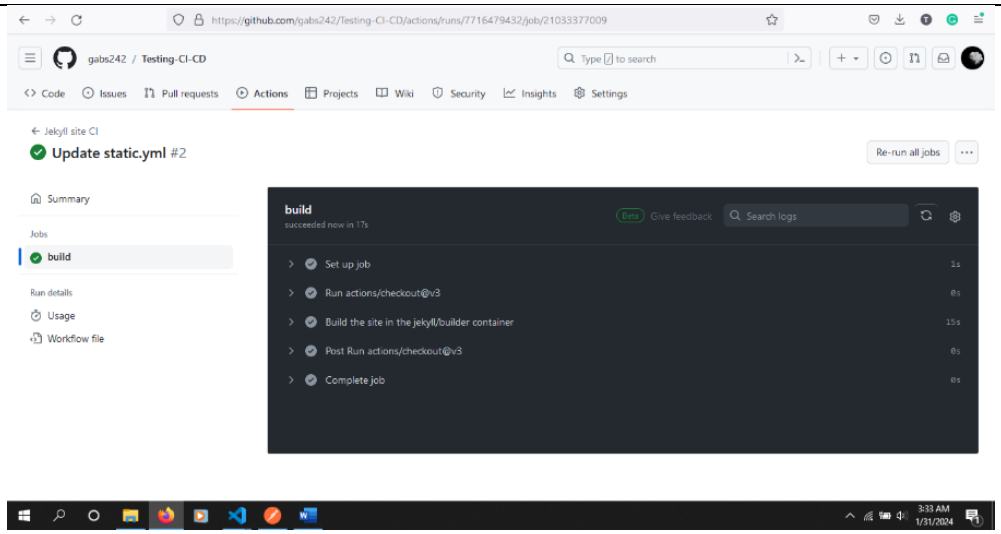
4.4.3. Kasus Pengujian

Pada bagian ini akan membahas beberapa kasus pengujian untuk aplikasi yang telah dibuat ini.

1. Apakah proses *Integration* dapat berjalan dengan baik?
2. Apakah aplikasi yang dibuat dapat di *deploy* menggunakan *App Engine* untuk mengaksesnya secara *online*?
3. Apakah aplikasi yang di *deploy* menggunakan *App Engine* dapat dijalankan dengan baik melalui *link hosting*?

4.4.4. Pelaksanaan Pengujian

Kasus Pengujian	Hasil Pengujian
Apakah proses *Integration* dapat berjalan dengan baik?	Proses *Integration* dapat berjalan dengan baik.

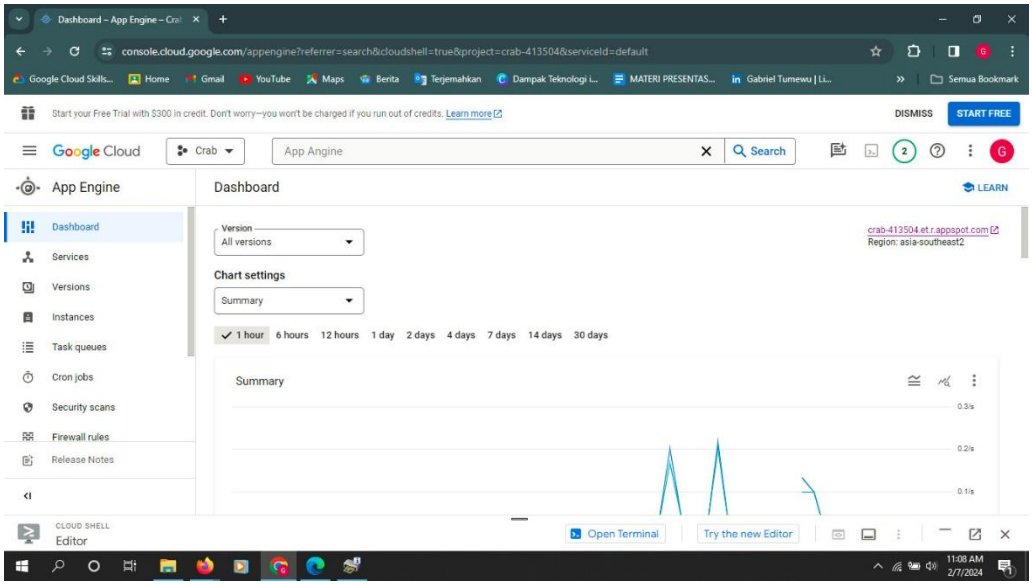


Gambar 4.14 Hasil Pengujian Integrasi

Gambar 4.14 menyoroti keberhasilan hasil pengujian integrasi yang telah dilakukan sebagai bagian integral dari tahap *Continuous Integration* (CI). *Continuous Integration* merupakan praktik pengembangan perangkat lunak yang mempromosikan integrasi kode secara berkelanjutan ke dalam repositori bersama. Dalam konteks ini, Gambar 4.14 mencerminkan pencapaian positif dalam proses ini. Keberhasilan ini menunjukkan bahwa setiap kali kode baru ditambahkan ke repositori, tes otomatis dijalankan secara otomatis untuk memverifikasi bahwa perubahan tersebut tidak mengganggu fungsionalitas keseluruhan sistem. Dengan demikian, Gambar 4.14 merupakan gambaran visual dari pentingnya *Continuous Integration* dalam siklus pengembangan perangkat lunak modern yang efisien dan andal.

Apakah aplikasi yang dibuat dapat di *deploy* menggunakan *App Engine* untuk mengaksesnya secara *online*?	Aplikasi yang dibuat dapat di *deploy* menggunakan *App Engine* untuk mengaksesnya secara *online*

Pada Gambar 4.15, terlihat keberhasilan aplikasi yang telah di *deploy* menggunakan *App Engine* di *Region asia-southeast*-2 (Jakarta). Salah satu indikator keberhasilan tersebut adalah adanya tautan atau link yang diberikan sebagai akses ke aplikasi tersebut. Di pojok kanan atas gambar (yang ditandai dengan warna ungu), terlihat tautan https://crab-413504.et.r.appspot.com. Tautan tersebut merupakan URL yang dapat digunakan untuk mengakses aplikasi yang telah di *deploy*. Dengan adanya tautan ini, pengguna atau tim pengembang dapat dengan mudah mengunjungi aplikasi yang berjalan di lingkungan *App Engine* di *Region asia-southeast*-2 (Jakarta) untuk melakukan pengujian, evaluasi, atau penggunaan secara langsung.



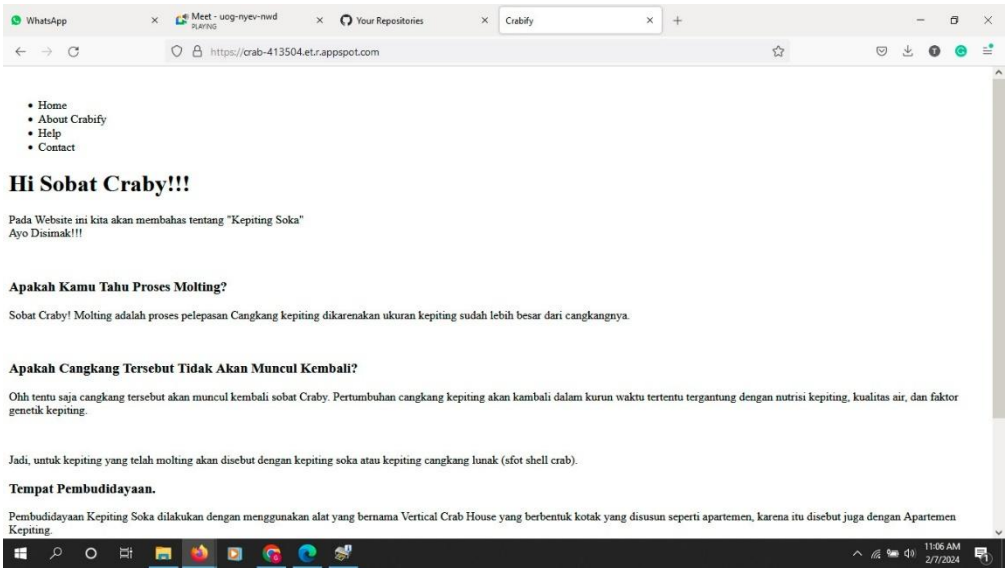
Gambar 4.15 Hasil Pengujian *Deployment* Aplikasi

Pentingnya tautan ini adalah sebagai cara untuk mengukur keberhasilan proses *deployment* aplikasi. Ketika aplikasi berhasil di *deploy*, hal yang selanjutnya penting adalah memastikan bahwa aplikasi tersebut dapat diakses dan berfungsi dengan baik oleh pengguna yang dituju. Oleh karena itu, adanya tautan yang langsung mengarah ke aplikasi yang telah di *deploy* menjadi indikator keberhasilan yang sangat penting. Dengan tautan tersebut, pengguna atau pihak terkait dapat mengakses aplikasi dengan mudah tanpa perlu mengetahui detail teknis atau melakukan langkah-langkah tambahan. Ini mempercepat proses pengujian dan penggunaan aplikasi yang baru dideploy, serta memudahkan evaluasi terhadap perubahan yang telah dilakukan pada aplikasi tersebut.

Apakah aplikasi yang di *deploy* menggunakan *App Engine* dapat dijalankan dengan baik melalui tautan *hosting*?	Aplikasi yang di *deploy* menggunakan *App Engine* dapat dijalankan dengan baik melalui tautan *hosting*.

Pada Gambar 4.16, ditampilkan keberhasilan *deployment* menggunakan *App Engine* yang telah dibuat. Tautan yang diberikan memungkinkan pengguna untuk terhubung ke aplikasi yang telah dibuat, sehingga mereka dapat menampilkan halaman aplikasi pendeteksi kepiting soka dengan mudah.

Tautan ini adalah jembatan langsung antara pengguna dan aplikasi yang telah di-*deploy*. Dengan mengklik tautan yang disediakan, pengguna akan diarahkan secara otomatis ke halaman utama atau halaman yang ditentukan dalam aplikasi tersebut.



Gambar 4.16 Hasil Pengujian Tautan

Keberhasilan *deployment* ditunjukkan dengan ketersediaan tautan ini. Ini menandakan bahwa proses *deployment* telah berhasil dan aplikasi siap untuk digunakan. Dengan demikian, pengguna atau tim pengembang dapat langsung mengakses aplikasi tanpa perlu melakukan langkah-langkah tambahan. Dengan adanya tautan yang langsung terhubung ke aplikasi, proses pengujian, evaluasi, dan penggunaan aplikasi menjadi lebih mudah dan cepat. Hal ini juga membantu memastikan bahwa aplikasi dapat diakses oleh pengguna target dengan lancar, sehingga meningkatkan pengalaman pengguna secara keseluruhan.

4.4.5. Analisis Hasil Pengujian

Dari hasil pengujian yang dilakukan pada Gambar 4.14 dan Gambar 4.15, dapat disimpulkan bahwa aplikasi yang dibuat telah berhasil di-*deploy* menggunakan *App Engine* dan dapat diakses secara *online* melalui tautan *hosting* yang disediakan.

Pada Gambar 4.14 menunjukkan bahwa penerapan *Continuous Integration* (CI) berhasil dilakukan. Dalam konteks ini, Gambar 4.14 mencerminkan pencapaian positif dalam proses ini. Keberhasilan ini menunjukkan bahwa setiap kali kode baru ditambahkan ke repositori, tes otomatis dijalankan secara otomatis untuk memverifikasi bahwa perubahan tersebut tidak mengganggu fungsionalitas keseluruhan sistem.

Pada Gambar 4.15, terlihat bahwa aplikasi berhasil di-*deploy* di *Region asia-southeast*-2 (Jakarta) menggunakan *App Engine*. Salah satu indikator keberhasilan adalah adanya tautan https://crab-413504.et.r.appspot.com yang diberikan sebagai akses ke aplikasi. Ini menunjukkan bahwa proses *deployment* telah berhasil dan aplikasi dapat diakses melalui tautan tersebut. Dengan demikian, aplikasi dapat diuji, dievaluasi, dan digunakan secara langsung oleh pengguna atau tim pengembang.

Pada Gambar 4.16, hasil pengujian menunjukkan bahwa tautan *hosting* tersebut dapat diakses dengan baik dan mengarahkan pengguna ke halaman aplikasi yang sesuai. Ini menegaskan bahwa *deployment* aplikasi menggunakan *App Engine* berjalan dengan baik dan aplikasi dapat dijalankan secara efektif melalui tautan *hosting* yang disediakan.

Secara keseluruhan, hasil pengujian menunjukkan bahwa proses *deployment* aplikasi menggunakan *App Engine* telah berhasil dan aplikasi dapat diakses secara *online* melalui tautan *hosting*. Keberhasilan ini mengindikasikan bahwa aplikasi siap untuk digunakan dan dapat memberikan nilai tambah kepada pengguna atau pemangku kepentingan yang dituju. Dengan demikian, proses pengembangan aplikasi dapat melanjutkan ke tahap selanjutnya dengan keyakinan bahwa aplikasi telah berhasil di-*deploy* dan dapat diakses dengan baik oleh pengguna.

BAB V
KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan pembahasan sebelumnya, dapat disimpulkan beberapa hal sebagai berikut:

1. Implementasi Layanan *Cloud Computing* CI/CD pada aplikasi pendeteksi Kepiting Soka dapat direalisasikan dengan baik. Proses *deployment* aplikasi menggunakan *App Engine* pada *Google Cloud Platform* telah berhasil, dan aplikasi dapat diakses secara *online* melalui tautan yang disediakan. Hal ini menunjukkan bahwa integrasi CI/CD berjalan lancar, memungkinkan pengembang untuk secara efisien mengirimkan perubahan ke lingkungan produksi tanpa adanya masalah signifikan.
2. Selama proses pengerjaan, metode penelitian waterfall telah digunakan dengan baik. Metode *waterfall* merupakan salah satu pendekatan dalam pengembangan perangkat lunak yang membagi proses pengembangan menjadi serangkaian tahapan linear, dimulai dari perencanaan, analisis, desain, implementasi, hingga pengujian. Dalam konteks pembahasan sebelumnya, terlihat bahwa tahapan-tahapan tersebut telah diterapkan secara sistematis, mulai dari perencanaan implementasi layanan *cloud computing* CI/CD hingga hasil pengujian aplikasi yang berhasil di-*deploy*. Hal ini menunjukkan bahwa penerapan metode *waterfall* telah membantu memastikan kualitas dan konsistensi dalam pengembangan aplikasi pendeteksi Kepiting Soka.

Kesimpulan di atas menggarisbawahi bahwa implementasi layanan *cloud computing* CI/CD dan penggunaan metode penelitian *waterfall* telah berkontribusi secara positif terhadap kesuksesan pengembangan aplikasi pendeteksi Kepiting Soka. Dengan demikian, dapat disimpulkan bahwa pendekatan yang digunakan telah memungkinkan pencapaian tujuan proyek dengan baik.

5.2. Saran

Berikut ini akan memberikan beberapa saran yang dapat menjadi acuan bagi penelitian selanjutnya yang memiliki metode dan topik penelitian serupa.

1. Memanfaatkan teknologi *Machine Learning* dan *Internet of Things* untuk meningkatkan akurasi deteksi kepiting soka.
2. Melakukan pengujian dan validasi lapangan yang lebih luas untuk memastikan kinerja aplikasi di berbagai kondisi lingkungan.
3. Fokus pada pengembangan antarmuka pengguna yang lebih intuitif dan ramah pengguna untuk meningkatkan user experience (UX).
4. Menyelidiki potensi penggunaan aplikasi untuk pemantauan dan pengelolaan populasi kepiting soka secara lebih luas.

Dengan menerapkan saran-saran ini, penelitian selanjutnya dapat lebih mendalam dan memberikan kontribusi yang lebih besar dalam pengembangan aplikasi pendeteksi kepiting soka atau topik terkait.

DAFTAR PUSTAKA

1] G. R. Payara and . R. Tanone, "Penerapan Firebase Realtime Database Pada Prototype Aplikasi Pemesanan Makanan Berbasis Android," *Jurnal Teknik Informatika dan Sistem Informasi*, vol. 4, 2019.

2] E. Fauzi, A. Zakiah, Y. Syukriyah and S. Yuliani, "INTEGRATED LEARNING MODEL: A BLEND OF PROJECT-BASED APPROACH AND SDLC CONCEPTS FOR SOFTWARE ENGINEERING COURSES, EVALUATED THROUGH EUCS," *Journal of Information Technology and Computer Science (INTECOMS)*, vol. 6, 2023.

3] K. R. d. T. Kementerian Pendidikan, "Kementerian," [Online]. Available: https://www.kemdikbud.go.id/main/. [Accessed Rabu Januari 2024].

4] H. Q. A'yun, "TRANSFORMASI BENTUK KEPITING BAKAU," INSTITUT SENI INDONESIA YOGYAKARTA, YOGYAKARTA, 2019.

5] S. R. Megumi, "Kepiting Bakau, Krustasea Unggulan Penunggu Hutan Mangrove," greeners.co, 26 Agustus 2019. [Online]. Available: https://www.greeners.co/flora-fauna/kepiting-bakau-penunggu-hutan-mangrove/. [Accessed 14 Januari 2024].

6] N. A. Ali, "Cloud Compuitng: Revolusi Komputer Masa Kini," *DimensiKOOP*, 2019.

7] T. Martensson, "Continuous integration and delivery applied to large-scale software-intensive embedded systems," pp. 1-233, 2019.

8] R. Mok and H. Zou, "Measuring the network performance of Google Cloud Platform," 2020.

9] B. B. Santoso and P. Saian, "Implementasi Flask Framework pada Development Modul Reporting Aplikasi Sistem Informasi Helpdesk," *Jurnal JTIK (Jurnal Teknologi Informasi dan Komunikasi)*, pp. 1-10, 2022.

10] I. Wahyudi and A. Syazili, "Dashboard Monitoring Website Dosen Studi Kasus Universitas Bina Darma," *Jurnal Pengembangan Sistem Informasi dan Informatika*, vol. 2, pp. 1-10, 2021.

11] I. K. S. Buana, "Implementasi Aplikasi Speech to Text untuk Memudahkan Wartawan Mencatat Wawancara dengan Python," *JURNAL SISTEM DAN INFORMATIKA (JSI)*, vol. 2, pp. 1-8, 2020.

12] "APLIKASI ARTIFICIAL INTELLIGENCE UNTUK MENDETEKSI OBJEK BERBASIS WEB MENGGUNAKAN LIBRARY TENSORFLOW JS, REACT JS DAN COCO DATASET," *Jurnal Sistem Informasi (JSiI)*, vol. Vol.9, 2022.

13] J. A. Ramadhan, D. T. Haniva and A. Suharso, "Systematic Literature Review Penggunaan Metodologi Pengembangan Sistem Informasi Waterfall, Agile, dan Hybrid," *Journal Information Engineering and Educational Technology (JIEET)*, vol. Vol.7, 2023.

14] Syamsiah, "PERANCANGAN FLOWCHART DAN PSEUDOCODE PEMBELAJARAN MENGENAL ANGKA DENGAN ANIMASI UNTUK ANAK PAUD RAMBUTAN," *STRING (Satuan Tulisan Riset dan Inovasi Teknologi)*, vol. Vol.4, 2019.

15] A. Jolpano, E. Handayani and G. Saptiani, "Pertumbuhan dan percepatan molting kepiting bakau (Scylla serrata) yang diberi di tambak Desa Salo Palai Kecamatan Muara Badak Kabupaten Kutai Kartanegara," *Jurnal Ilmu Perikanan Tropis Nusantara*, vol. Vol. 2, pp. 1-10, 2023.

16] M. Averyt, J. Chen, M. Khamis, J. Lee and N. Mitta, "CS 4624 Final Report:Crisis Events Webpages Archiving," Virginia Tech, Blacksburg VA 24061, Blacksburg, 2023.

17] A. N. Ziogas, T. Ben-Nun, T. Schneider and T. Hoefler, "NPBench: A Benchmarking Suite for High-Performance NumPy," Department of Computer Science, Zurich, Switzerland, 2021.

18] L. Novitasari and B. Hartono, "Kaji Penerapan Sistem Pengolahan Citra untuk Pendeteksian Penggunaan Masker," *Seminar Nasional Industri dan Teknologi (SNIT)*, pp. 1-11, 2021.

19] A. A. Achyar, A. M. Olow, M. R. Perdana, A. Sundawijaya and A. D. Goenawan, "Identifikasi Ras Wajah dengan Menggunakan Metode Deep Learning Model Keras," *Jurnal Teknik Mesin, Industri, Elektro Dan Informatika (JTMEI)*, vol. Vol.1, pp. 29-37, 2022.