

**PENERAPAN MICROSERVICE DENGAN HIERARCHICAL
CLUSTERING UNTUK DEKOMPOSISI DARI MONOLIT
PADA ENTERPRISE RESOURCE PLANNING**

TUGAS AKHIR

**Albertus Septian Angkuw
1119002**



**PROGRAM STUDI INFORMATIKA
INSTITUT TEKNOLOGI HARAPAN BANGSA
BANDUNG
2022**

**PENERAPAN MICROSERVICE DENGAN HIERARCHICAL
CLUSTERING UNTUK DEKOMPOSISI DARI MONOLIT
PADA ENTERPRISE RESOURCE PLANNING**

TUGAS AKHIR

**Diajukan sebagai salah satu syarat untuk memperoleh
gelar sarjana dalam bidang Informatika**

**Albertus Septian Angkuw
1119002**



**INSTITUT
TEKNOLOGI
HARAPAN
BANGSA**

Veritas vos liberabit

**PROGRAM STUDI INFORMATIKA
INSTITUT TEKNOLOGI HARAPAN BANGSA
BANDUNG
2022**

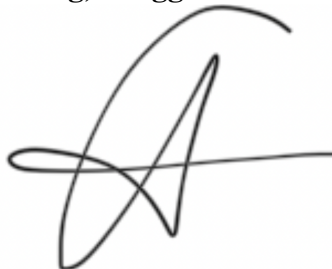
HALAMAN PERNYATAAN ORISINALITAS

**Saya menyatakan bahwa Tugas Akhir yang saya susun ini
adalah hasil karya saya sendiri.**

**Semua sumber yang dikutip maupun dirujuk
telah saya nyatakan dengan benar.**

**Saya bersedia menerima sanksi pencabutan gelar akademik
apabila di kemudian hari Tugas Akhir ini terbukti plagiat.**

Bandung, Tanggal Bulan Tahun

A handwritten signature in black ink, featuring a large, stylized capital 'A' with a horizontal stroke extending to the right and a loop on the left.

**Albertus Septian Angkuw
1119002**

HALAMAN PENGESAHAN TUGAS AKHIR

Tugas Akhir dengan judul:

**PENERAPAN MICROSERVICE DENGAN HIERARCHICAL CLUSTERING
UNTUK DEKOMPOSISI DARI MONOLIT PADA ENTERPRISE RESOURCE
PLANNING**

yang disusun oleh:

**Albertus Septian Angkuw
1119002**

telah berhasil dipertahankan di hadapan Dewan Penguji Sidang Tugas Akhir yang
dilaksanakan pada:

Hari / tanggal : Hari, Tanggal Bulan Tahun

Waktu : Jam (24-HOUR FORMAT, contoh 16.00 WIB) WIB

Menyetujui

Pembimbing Utama:

Pembimbing Pendamping:

Hans Christian Kurniawan, S.T., M.T

NIK

**...
NIK**

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Institut Teknologi Harapan Bangsa, saya yang bertandatangan di bawah ini:

Nama : Nama Pengarang

NIM : NIM

Program Studi : Informatika

demikian pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Institut Teknologi Harapan Bangsa **Hak Bebas Royalti Noneksklusif** (*Non-exclusive Royalty Free Rights*) atas karya ilmiah saya yang berjudul:

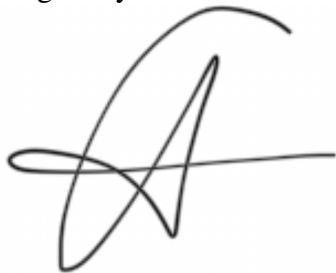
JUDUL TUGAS AKHIR

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Institut Teknologi Harapan Bangsa berhak menyimpan, mengalihmediakan, mengelola dalam pangkalan data, dan memublikasikan karya ilmiah saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Bandung, Tanggal Bulan Tahun

Yang menyatakan



Nama Pengarang

ABSTRAK

Nama : Nama Pengarang
Program Studi : Informatika
Judul : Judul Tugas Akhir dalam Bahasa Indonesia

Lorem ipsum dolor sit amet, quidam dicunt blandit duo in. Cu sed dictas vidisse admodum, at qualisque scripserit est, est case salutandi ea. No quot ornatus probatus nec, movet quodsi forensibus pri ad. His esse wisi vocent et, ex est mazim libris quaeque. Habeo brute vel id, inani volumus adolescens et mei, solet mediocrem te sit. At sonet dolore atomorum sit, tistique sapientem contentiones no vix, dolore iriure ex vix. Vim commune appetere dissentiet ne, aperiri patrioque similique sed eu, nam facilisis neglegentur ex. Qui ut tistique voluptua. Ei utroque electram gubergren per. Laudem nonumes an vis, cum veniam eligendi liberavisse eu. Etiam graecis id mel. An quo rebum iracundia definitionem. At quo congue graeco explicari. Cu eos wisi legimus patrioque. Cum iisque offendit ei. Ei eruditi lobortis pericula sea, te graeco salutatus sed, ne integre insolens mei. Mea tale aliquam minimum te. Eu mel putant virtute, essent inermis nominavi mea no. Laoreet indoctum sea te. Te scripta fabulas duo, pro doming recusabo voluptaria at. Cu sed numquam inciderint, ei minim altera disputando cum, te nec graeco maiorum convenire. Cu mel putent rationibus dissentiet. Per vidisse scaevola oportere ei, qui solet molestie eu. Hinc diceret nominati per at, nec dico denique laboramus et. Legere regione his at, aequae decore in mei. Lorem ipsum dolor sit amet, quidam dicunt blandit duo in. Cu sed dictas vidisse admodum, at qualisque scripserit est, est case salutandi ea. No quot ornatus probatus nec, movet quodsi forensibus pri ad. His esse wisi vocent et.

Kata kunci: Sonet, dolore, atomorum, tistique, sapientem.

ABSTRACT

Name : Nama Pengarang
Department : Informatics
Title : Judul Tugas Akhir dalam Bahasa Inggris

Lorem ipsum dolor sit amet, quidam dicunt blandit duo in. Cu sed dictas vidisse admodum, at qualisque scripserit est, est case salutandi ea. No quot ornatus probatus nec, movet quodsi forensibus pri ad. His esse wisi vocent et, ex est mazim libris quaeque. Habeo brute vel id, inani volumus adolescens et mei, solet mediocrem te sit. At sonet dolore atomorum sit, tistique sapientem contentiones no vix, dolore iriure ex vix. Vim commune appetere dissentiet ne, aperiri patrioque similique sed eu, nam facilisis neglegentur ex. Qui ut tistique voluptua. Ei utroque electram gubergren per. Laudem nonumes an vis, cum veniam eligendi liberavisse eu. Etiam graecis id mel. An quo rebum iracundia definitionem. At quo congue graeco explicari. Cu eos wisi legimus patrioque. Cum iisque offendit ei. Ei eruditi lobortis pericula sea, te graeco salutatus sed, ne integre insolens mei. Mea tale aliquam minimum te. Eu mel putant virtute, essent inermis nominavi mea no. Laoreet indoctum sea te. Te scripta fabulas duo, pro doming recusabo voluptaria at. Cu sed numquam inciderint, ei minim altera disputando cum, te nec graeco maiorum convenire. Cu mel putent rationibus dissentiet. Per vidisse scaevola oportere ei, qui solet molestie eu. Hinc diceret nominati per at, nec dico denique laboramus et. Legere regione his at, aequae decore in mei. Lorem ipsum dolor sit amet, quidam dicunt blandit duo in. Cu sed dictas vidisse admodum, at qualisque scripserit est, est case salutandi ea. No quot ornatus probatus nec, movet quodsi forensibus pri ad. His esse wisi vocent et.

Keywords: Sonet, dolore, atomorum, tistique, sapientem.

KATA PENGANTAR

Lorem ipsum dolor sit amet, quidam dicunt blandit duo in. Cu sed dictas vidisse admodum, at qualisque scripserit est, est case salutandi ea. No quot ornatus probatus nec, movet quodsi forensibus pri ad. His esse wisi vocent et, ex est mazim libris quaeque. Habeo brute vel id, inani volumus adolescens et mei, solet mediocrem te sit. At sonet dolore atomorum sit, tistique sapientem contentiones no vix, dolore iriure ex vix. Vim commune appetere dissentiet ne, aperiri patrioque similique sed eu, nam facilisis neglegentur ex. Qui ut tistique voluptua. Ei utroque electram gubergren per. Laudem nonumes an vis, cum veniam eligendi liberavisse eu. Etiam graecis id mel. An quo rebum iracundia definitionem. At quo congue graeco explicari. Cu eos wisi legimus patrioque. Cum iisque offendit ei. Ei eruditi lobortis pericula sea, te graeco salutatus sed, ne integre insolens mei. Mea tale aliquam minimum te. Eu mel putant virtute, essent inermis nominavi mea no. Laoreet indoctum sea te. Te scripta fabulas duo, pro doming recusabo voluptaria at. Cu sed numquam inciderint, ei minim altera disputando cum, te nec graeco maiorum convenire. Cu mel putent rationibus dissentiet. Per vidisse scaevola oportere ei, qui solet molestie eu. Hinc diceret nominati per at, nec dico denique laboramus et. Legere regione his at, aequae decore in mei.

Bandung, Tanggal Bulan Tahun

Hormat penulis,

A stylized, handwritten signature in black ink, consisting of a large, sweeping loop followed by a horizontal line and a small upward stroke.

Nama Pengarang

DAFTAR ISI

ABSTRAK	iv
ABSTRACT	v
KATA PENGANTAR	vi
DAFTAR ISI	vii
DAFTAR TABEL	ix
DAFTAR GAMBAR	x
DAFTAR ALGORITMA	x
DAFTAR LAMPIRAN	xi
BAB 1 PENDAHULUAN	1-1
1.1 Latar Belakang	1-1
1.2 Rumusan Masalah	1-3
1.3 Tujuan Penelitian	1-3
1.4 Batasan Masalah	1-3
1.5 Kontribusi Penelitian	1-3
1.6 Metodologi Penelitian	1-4
1.7 Sistematika Pembahasan	1-4
BAB 2 LANDASAN TEORI	2-1
2.1 Tinjauan Pustaka	2-1
2.1.1 Monolit	2-1
2.1.2 Microservice	2-2
2.1.3 Enterprise Resource Planning	2-4
2.1.4 Analisis Kode	2-5
2.1.5 Clustering	2-5
2.1.6 Dekomposisi	2-6
2.1.7 Teknologi dan Library	2-6
2.1.7.1 Docker	2-6
2.1.7.2 gRPC	2-6
2.2 Tinjauan Studi	2-6

2.3	Tinjauan Objek	2-11
BAB 3	ANALISIS DAN PERANCANGAN SISTEM	3-1
3.1	Analisis Masalah	3-1
3.2	Kerangka Pemikiran	3-1
3.3	Urutan Proses Global	3-2
3.3.1	Proses Clustering	3-3
3.3.2	Dekomposisi Monolitik ke Microservice	3-3
3.3.3	Deployment	3-3
3.3.4	Evaluasi	3-3
BAB 4	IMPLEMENTASI DAN PENGUJIAN	4-1
4.1	Lingkungan Implementasi	4-1
4.1.1	Spesifikasi Perangkat Keras	4-1
4.1.2	Spesifikasi Perangkat Lunak	4-2
4.2	Implementasi Perangkat Lunak	4-2
4.2.1	Implementasi <i>Class</i>	4-3
4.2.1.1	<i>Class</i> Nama_Class_1	4-3
4.2.1.2	<i>Class</i> Nama_Class_2	4-3
4.2.2	Implementasi Numquam	4-3
4.3	Implementasi Nama_Implementasi	4-3
4.4	Implementasi Aplikasi	4-3
4.5	Pengujian	4-4
4.5.1	Pengujian Nama_Pengujian_1	4-4
4.5.2	Pengujian Nama_Pengujian_2	4-6
BAB 5	KESIMPULAN DAN SARAN	5-1
5.1	Kesimpulan	5-1
5.2	Saran	5-1
BAB A	LAMPIRAN A	A-1
	ASJDBAKJSDBKA	A-1
BAB B	DATASET HASIL KUISIONER 2	B-3

DAFTAR TABEL

2.1	Tinjauan Studi	2-6
2.2	Komposisi dari Module pada aplikasi Odoo	2-13
4.1	atribut pada <i>class</i> nama_class_1	4-4
4.2	Daftar <i>method</i> pada <i>class helper</i>	4-5
A-1	<i>Lorem ipsum</i>	A-1
A-1	<i>Lorem ipsum</i>	A-2
B-1	<i>Lorem ipsum</i>	B-3
B-1	<i>Lorem ipsum</i>	B-4

DAFTAR GAMBAR

2.1	Ilustrasi Arsitektur Odoo	2-12
2.2	Diagram Database Odoo	2-14
3.1	Kerangka Pemikiran	3-1
3.2	Diagram Flowchart Proses Global	3-2
3.3	Diagram Deployment	3-3

DAFTAR ALGORITMA

LAMPIRAN A	A-1
LAMPIRAN B	B-3

DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Aplikasi Enterprise Resource Planning(ERP) merupakan perangkat lunak yang digunakan pada perusahaan untuk menjalankan bisnisnya dimana perusahaan dapat mengotomatisasi dan mengintegrasikan sebagian besar proses bisnisnya dengan ini perusahaan bisa menghasilkan serta mengakses informasi secara langsung [1]. Selain itu diharapkan juga aplikasi memiliki skalabilitas terhadap operasi bisnis, yang diperlukan dalam jangka panjang, misalkan ketika perusahaan tumbuh dari waktu ke waktu, serta dalam waktu singkat, misalnya pada saat volume transaksi tinggi seperti belanja Natal [1].

Dalam membangun aplikasi ERP umumnya dibangun dengan beberapa arsitektur seperti *two-tier*, *three-tier/n-tier*, dan *service oriented architecture(SOA)* dimana arsitektur ini disebarkan secara monolit. Saat ini arsitektur yang umum digunakan yaitu SOA karena dapat membantu perancangan ERP menjadi lebih terukur, andal dan fleksibel dengan memecah fungsionalitas menjadi bagian kecil yang dinamakan service [1].

SOA memiliki keuntungan dalam melakukan pemeriksaan status aplikasi, melakukan perutean untuk service backend, meskipun demikian ditemukan bahwa SOA bisa menjadi rumit dan menyebabkan terjadinya bottleneck. Arsitektur Microservice(MSA) dapat menangani kekurangan ini dengan terisolasi, independen dan mudah didistribusikan sehingga memudahkan skalabilitas. Keuntungan terbesarnya yaitu aplikasi bisa dibangun dengan berbagai pilihan teknologi dan memungkinkannya untuk digunakan secara independen satu sama lain. Ini sangat menyederhanakan siklus pengembangan, pengujian, pembuatan, dan penerapan aplikasi karena perubahan terbatas pada satu service daripada seluruh aplikasi [3].

Hal ini dibuktikan juga dengan penelitian sebelumnya yang melakukan uji performa berupa uji beban dari setiap arsitektur. Dimana MSA memiliki throughput yang lebih tinggi pada 1500 pengguna dengan nilai rata-rata 1,1 dibandingkan dengan arsitektur SOA sebesar 0,7 dan monolith sebesar 0,6. Selain itu pada response time MSA lebih cepat 5 detik yaitu sebesar 33 detik dibandingkan dengan monolith sebesar 38 detik dan SOA sebesar 43 detik.

Pada pengukuran jumlah kode response 200(Berhasil), MSA memiliki

jumlah response tertinggi di kode berhasil dengan memiliki jumlah response terkecil di kode 302(Pengalihan), 304(Cache) ,408(Waktu Habis), 500(Kesalahan Internal Server) dan tidak memiliki jumlah response di kode 404(Tidak ditemukan). Dimana pada aspek pemeliharaan aplikasi, MSA lebih unggul daripada SOA dan monolit unggul daripada SOA [4].

Naman manfaat ini hanya dapat dimanfaatkan jika service didekomposisi dengan cara yang paling optimal dengan mempertimbangkan gambaran besar dari seluruh cakupan aplikasi. Jika tidak, desain ini mungkin terbukti kontraproduktif dan menyebabkan latensi, kompleksitas, dan inefisiensi Hal ini diperlukan untuk memisahkan aplikasi menjadi bagian-bagian yang sesuai secara fungsional dan memperoleh service kohesif tinggi dan service yang digabungkan secara longgar diharapkan sebagai hasil dari dekomposisi [3].

Dalam melakukan dekomposisi bisa dilakukan dengan konsep Domain Driven Design(DDD), Functional, Dataflow, dan Dependency Capturing dengan Clustering. Pada hasil evaluasi DDD menunjukkan aplikasi berhasil didekomposisi ke microservice. Dengan pendekatan Functional hasil evaluasi menunjukkan bahwa identifikasi microservice dapat dilakukan lebih cepat. Di pendekatan dataflow ini menunjukkan dekomposisi bisa ditentukan dari pertimbangan coupling dan kohesi. Identifikasi microservice dengan menganalisis ketergantungan proses bisnis dari control, dengan data dan control, data, dan semantic models. Kemudian untuk metode Clustering untuk mengidentifikasi microservice, metode clustering yang digunakan yaitu Hierarchical Clustering. Hasil dari validasi pendekatan ini menunjukkan bahwa pendekatan ini mencapai hasil yang lebih baik daripada pendekatan yang ada dalam hal identifikasi microservice [5].

Pada penelitian ini akan melakukan dekomposisi aplikasi ERP yang disebarkan secara monolit menjadi arsitektur microservice dengan pendekatan menganalisis graph yang dihasilkan dari source code kemudian dilakukan pengelompokan melalui Hierarchical Clustering. Hasil dari pengelompokan akan diimplementasikan dan dilakukan uji beban sehingga mengetahui nilai latensi, jumlah throughput, dan penggunaan sumber daya komputasi. Dengan ini diharapkan bisa menyelesaikan permasalahan yang terjadi di aplikasi ERP seperti kustomisasi dan skalabilitas.

1.2 Rumusan Masalah

Berikut adalah rumusan masalah yang dibuat berdasarkan latar belakang diatas.

1. Bagaimana nilai kohesi dan kopel yang dihasilkan dari dekomposisi melalui pendekatan Hierarchical Clustering?
2. Bagaimana performa aplikasi ERP antara arsitektur monolit dan arsitektur microservice dalam kondisi beban yang tinggi?
3. Berapa besar penggunaan sumber daya aplikasi ERP yang digunakan pada arsitektur monolit dibandingkan arsitektur microservice?

1.3 Tujuan Penelitian

Berdasarkan rumusan masalah di atas, maka tujuan tujuan penelitian ini adalah.

1. Menerapkan dekomposisi aplikasi ERP monolit ke microservice dengan pendekatan Hierarchical Clustering.
2. Mencari kelompok service yang memiliki nilai kopel rendah dan nilai kohesi tinggi.
3. Membandingkan performa dan sumber daya aplikasi ERP dengan bentuk pengelompokan yang berbeda di arsitektur microservice.

1.4 Batasan Masalah

Agar penelitian ini menjadi lebih terarah, maka penulis membatasi masalah yang akan dibahas sebagai berikut.

1. Penyebaran aplikasi dilakukan dengan framework docker.
2. Aplikasi yang digunakan adalah aplikasi yang sudah dibangun sebelumnya dan disebarkan dengan arsitektur monolit.
3. Perubahan arsitektur tidak menambah atau mengurangi fungsionalitas dari aplikasi.
4. Hanya module utama pada ERP yang akan dilakukan dekomposisi

1.5 Kontribusi Penelitian

Kontribusi yang diberikan pada penelitian ini adalah sebagai berikut.

1. Memberikan langkah dalam melakukan dekomposisi aplikasi monolit ke microservice dengan Hierarchical Clustering.
2. Mengetahui pengaruh dari performa aplikasi yang sudah dilakukan dekomposisi dengan uji beban pada aplikasi.

3. Membuat aplikasi microservice yang memiliki nilai kohesi tinggi dan nilai kopel rendah.

1.6 Metodologi Penelitian

Tahapan-tahapan yang akan dilakukan dalam pelaksanaan penelitian ini adalah sebagai berikut.

1. Penelitian Pustaka

Penelitian ini dimulai dengan studi kepustakaan yaitu mengumpulkan referensi baik dari buku, paper, jurnal, atau artikel daring mengenai arsitektur microservice, permasalahan pada aplikasi ERP dan dekomposisi monolit ke microservice.

2. Analisis Dilakukan analisis permasalahan yang ada, batasan-batasan yang ditentukan, dan kebutuhan-kebutuhan yang diperlukan untuk menyelesaikan permasalahan yang ditemukan.

3. Perancangan

Pada tahap ini dilakukan perancangan untuk melakukan dekomposisi dari aplikasi arsitektur monolit ke arsitektur microservice dengan dengan pendekatan Hierarchical Clustering.

4. Implementasi

Pada tahap ini mengimplementasikan hasil perancangan dekomposisi ke aplikasi microservice pada aplikasi yang dibuat dengan arsitektur monolit.

5. Pengujian

Pada tahap ini dilakukan pengujian pada aplikasi yang sudah di dekomposisi. Pengujian melalui uji beban akan dilakukan dengan perbandingan antara aplikasi monolit dan aplikasi microservice.

1.7 Sistematika Pembahasan

BAB 1: PENDAHULUAN

Pendahuluan yang berisi latar belakang, rumusan masalah, tujuan penelitian, batasan masalah, kontribusi penelitian, serta metode penelitian.

BAB 2: LANDASAN TEORI

Landasan Teori yang berisi penjelasan dasar teori yang mendukung penelitian ini, seperti arsitektur monolit, arsitektur microservice, hierarchical clustering, dan dekomposisi.

BAB 3: ANALISIS DAN PERANCANGAN

Analisis dan Perancangan yang berisi tahapan penerapan dekomposisi aplikasi

monolit ke microservice dengan hierarchical clustering dan penyebaran aplikasi melalui kontainer.

BAB 4: IMPLEMENTASI DAN PENGUJIAN

Implementasi dan Pengujian yang berisi pembangunan aplikasi dan pengujian dengan mensimulasikan dan mengevaluasi aplikasi yang telah didekomposisi.

BAB 5: KESIMPULAN DAN SARAN

Penutup yang berisi kesimpulan dari penelitian dan saran untuk penelitian lebih lanjut di masa mendatang.

BAB 2 LANDASAN TEORI

Pada bab ini menjelaskan beberapa teori dan jurnal yang berhubungan dengan permasalahan penelitian dan yang digunakan pada proses penelitian.

2.1 Tinjauan Pustaka

Pembahasan mengenai teori-teori tersebut akan dijelaskan sebagai berikut.

2.1.1 Monolit

Monolit yaitu suatu cara untuk melakukan penyebaran. Ketika semua fungsi dalam sistem harus disebar secara bersama-sama, maka itu merupakan monolit. Tantangan yang muncul pada monolit terjadi ketika semakin banyak orang yang bekerja pada aplikasi yang sama. Akibatnya setiap pengembang memiliki kepentingan masing-masing dalam mengelola kode yang sama dan membuat pengambilan keputusan sulit serta tidak fleksibel. Keuntungan yang muncul dengan monolit yaitu penyebaran yang lebih sederhana dan memudahkan pemantauan, pemecahan masalah, dan aktivitas pengujian sistem. monolit dapat menyederhanakan pada penggunaan kembali kode yang sudah dibuat sebelumnya [6].

Terdapat 3 jenis monolit [6]:

1. Single Process Monolith

Dimana sebuah kode disebar dengan satu proses. Dimana setiap kode bisa berada di banyak instances serta tempat penyimpanan dan mendapatkan data disimpan pada suatu database yang sama. Variasi lainnya yaitu modular monolith dimana setiap kode bisa bekerja secara independen tetapi perlu dijadikan satu kesatuan ketika ingin dilakukan penyebaran.

2. Distributed Monolith

Monolit terdistribusi adalah sistem yang terdiri dari beberapa layanan, tetapi untuk apa pun alasannya seluruh sistem harus disebar bersama-sama. Sebuah monolith terdistribusi mungkin akan memenuhi definisi dari sebuah arsitektur service-oriented (SOA).

Monolit terdistribusi biasanya muncul di kondisi dimana tidak cukup fokus pada konsep information hiding dan kohesi dari fungsi bisnis. Akibatnya terbentuklah arsitektur yang memiliki kopel yang tinggi, dimana bisa perubahan menyebabkan kerusakan pada bagian sistem lain.

3. Sistem Black-Box Pihak Ketiga Aplikasi pihak ketiga merupakan sebuah monolit, misalkan sistem penggajian, sistem CRM, dan sistem SDM. Faktor umum yang terjadi yaitu aplikasi ini dibuat dan dikelola oleh orang lain dimana pengembang belum tentu memiliki kemampuan untuk mengubah kode seperti Software-as-a-Service(SaaS).

Keuntungan dari Monolit:

1. Sederhana dalam melakukan pengembangan karena Integrated Development Environment (IDE) dan peralatan pengembang berfokus pada membuat satu aplikasi
2. Mudah untuk melakukan perubahan secara radikal di aplikasi. Perubahan ini bisa dari kode hingga skema database serta proses deployment.
3. Pengujian dilakukan pada satu aplikasi, pengembang dapat membuat pengujian dari awal hingga akhir dengan lebih mudah dan terintegrasi
4. Deployment dilakukan pada satu aplikasi, pengembang hanya menyalin aplikasi dari komputer ke komputer yang lain. Dengan ini aplikasi relatif mudah dilakukan konfigurasi dan mudah diperbanyak jumlah aplikasi.

Tantangan dari monolith

1. Sulit dikembangkan secara berkelanjutan
2. Proses deployment sulit ketika aplikasi sudah di deploy di produksi.
3. Tidak mudah untuk melakukan skalabilitas
4. Terkunci pada teknologi jadul

2.1.2 Microservice

Microservice adalah service yang bisa di deploy secara independen yang dimodelkan berdasarkan bisnis domain. Service ini berkomunikasi satu sama lain melalui jaringan komputer dan bisa dibangun dengan berbagai macam teknologi. Microservice adalah salah tipe dari service oriented architecture (SOA) meskipun ada perbedaan dalam membuat batasan antara service dan deployment secara independent.[6] Service adalah komponen perangkat lunak yang memiliki kegunaannya secara khusus dimana komponen ini bisa berdiri sendiri dan secara independen dilakukan proses deployment. Service memiliki API(Application Programming Interface) yang memberikan akses kepada client untuk melakukan

operasi. Terdapat 2 tipe operasi yaitu perintah dan kueri.[8] API terdiri dari perintah, kueri dan event. Perintah dapat berupa buatPesanan() yang melakukan aksi dan memperbarui data. Kueri dapat berupa cariPesananBerdasarkanID() yang digunakan untuk mengambil data. Service juga dapat membuat suatu event seperti PesananSudahDibuat dimana event ini akan dikonsumsi oleh client-nya.[8] Service API akan mengenkapsulasi internal implementasinya, sehingga pengembang aplikasi tidak bisa menuliskan kode yang melewati API. Akibatnya arsitektur microservice dapat mewajibkan modularitas di aplikasi. Setiap service di arsitektur microservice memiliki masing-masing arsitektur dan dimungkinkan teknologi yang berbeda. Tetapi service memiliki arsitektur hexagonal. Dimana API akan diimplementasi melalui adaptor yang berinteraksi dengan logika bisnis[7]

Ciri Khusus Microservice

1. Kecil dan berfokus pada satu hal dengan baik
2. Otonomi / Bisa berdiri sendiri
3. Proses Deployment secara mandiri
4. Dimodelkan di Sekitar Domain Bisnis
5. Setiap service memiliki data yang dikelola masing-masing

Keuntungan dari Microservice

1. Memudahkan pengembangan aplikasi kompleks dan flexibel Service memiliki ukuran yang kecil sehingga mudah dikelola
2. Service bisa dilakukan skala secara independen
3. Mudah melakukan percobaan dan penggunaan teknologi baru
4. Memiliki isolasi kegagalan lebih baik

Tantangan dari Microservice

1. Menemukan service yang tepat itu sulit
2. Memiliki kompleksitas karena merupakan suatu distributed sistem
3. Membutuhkan koordinasi ketika menambahkan fitur yang menjangkau beberapa service
4. Menentukan kapan untuk melakukan adopsi

Pola Microservice Gambar *.* Pola dalam menyelesaikan masalah di arsitektur Microservice[8]

1. Application patterns Permasalahan yang harus diselesaikan oleh pengembang aplikasi - Proses kueri - Dekomposisi aplikasi menjadi service - Menjaga data konsistensi dan implementasi proses transaksi - Mengotomatiskan proses pengujian service
2. Application infrastructure Permasalah infrastruktur yang memiliki pengaruh pada proses pengembangan aplikasi - Pola komunikasi - Pola mengatasi permasalahan antar service
3. Infrastructure patterns Permasalahan infrastruktur yang muncul diluar dari pengembangan aplikasi Pola Komunikasi Pola Service Deployment Pola observability untuk mengetahui bagaimana aplikasi bekerja

2.1.3 Enterprise Resource Planning

Enterprise Resource Planning(ERP) adalah suatu sistem perangkat lunak yang memungkinkan perusahaan untuk mengotomatiskan dan mengintegrasikan proses bisnisnya dengan komputerisasi. Dengan ini setiap informasi yang diperlukan di proses bisnis dapat dibagikan dan digunakan disemua bagian perusahaan dengan alur terstruktur. Sistem ERP dapat mengeliminasi duplikasi data dan memberikan integrasi data. Sistem ERP memiliki database dimana semua transaksi bisnis dapat direkam, diproses, dipantau dan dilaporkan. Tujuannya agar bisa proses bisnis bisa dilakukan dengan lebih cepat , murah, dan transparan.[1] Sistem ERP dapat memberikan dukungan untuk proses bisnis perusahaan melalui modul yang terpisah. Setiap modul adalah aplikasi perangkat lunak yang dibangun khusus untuk setiap operasi bisnis. Umumnya modul yang ditemukan pada ERP yaitu Modul Produksi, Modul Manajemen Rantai Pasokan, Modul Keuangan, Modul Penjualan & Pemasaran, Modul Sumber Daya Manusia, dan modul pelengkap lainnya seperti e-commerce.[1]

Arsitektur ERP:

1. The tiered architecture
2. Web-based architecture
3. Service oriented architecture
4. Cloud architecture

2.1.4 Analisis Kode

Analisis Kode analisis adalah suatu proses mengekstraksi informasi mengenai suatu program dari kode atau artifak. Proses ini bisa dilakukan secara manual yaitu dengan melihat kode program atau bahasa mesin namun kompleksitas program yang tinggi membuat proses secara manual sangat mahal dan tidak efektif. Sehingga diperlukan peralatan yang dapat membantu proses analisis kode. Peralatan ini dapat memberikan informasi kepada pengembang mengenai program yang dianalisis.[9]

Anatomi Analisis Kode

1. Ekstraksi Data
2. Representasi Informasi
3. Explorasi Pengetahuan

Anatomi Analisis Kode

1. Statik vs Dinamis
2. Sound vs Unsound
3. Flow sensitive vs Flow insensitive
4. Context sensitive vs Context insensitive

Tantangan Kode Analisis

1. Perbedaan bahasa kode program
2. Multi-Language
3. Analisis secara langsung

2.1.5 Clustering

Clustering yaitu suatu proses untuk melakukan pengelompok atau klasifikasi objek. Objek bisa ditentukan dari pengukuran atau berdasarkan hubungan antar objek lainnya. Tujuan dari clustering yaitu untuk menemukan struktur data yang valid. Cluster terdiri dari sejumlah object serupa yang dikumpulkan / dikelompokkan bersama.[10]

Metode Clustering yang umumnya digunakan:

1. Hierarchical Clustering ...

2. Partitional Clustering ...

2.1.6 Dekomposisi

Pemilihan bagian yang ingin didekomposisi untuk menjadi Service:[6]

1. Berdasarkan proses bisnis
2. Berdasarkan sub-domain (DDD)

Pola untuk Proses Dekomposisi:

1. Pola Strangle
2. Pola UI Composition
3. Branch By Abstraction
4. Parallel Run
5. Decorating Collaborator
6. Change Data Capture

Tantangan dan Hambatan Dekomposisi:

1. Latensi Jaringan
2. Menjaga konsistensi data antar service
3. Adanya God Class yang mencegah dekomposisi

2.1.7 Teknologi dan Library

2.1.7.1 Docker

...

2.1.7.2 gRPC

...

2.2 Tinjauan Studi

Pada Tabel 2.1 diberikan penjelasan mengenai studi terkait dalam penelitian:

Tabel 2.1 Tinjauan Studi

No	Peneliti	Judul	Rumusan Masalah	Hasil
----	----------	-------	-----------------	-------

1	Munezero Immaculée Josélyne, Doreen Tuheirwe-Mukasa, Benjamin Kanagwa, dan Joseph Balikuddembe	Partitioning Micro-services: A Domain Engineering Approach [7]	<ul style="list-style-type: none"> • Mengeksplorasi bagaimana cara melakukan pemisahan microservice • Hubungan antara microservice dan Domain Driven Design • Ilustrasi dan evaluasi aplikasi microservice yang dipisah dengan Domain Driven Design 	Dengan Domain Driven Design berhasil dilakukan pemecahan pada aplikasi cuaca. Hasilnya berupa rancangan microservice, namun belum dilakukan implementasi menjadi microservice
2	Shmuel Tyszberowicz, Robert Heinrich, Bo Liu, dan Zhiming Liu	Identifying Micro-services Using Functional Decomposition [8]	<ul style="list-style-type: none"> • Mengetahui dampak analisis dari pemeliharaan sistem • Mengevaluasi dari hasil yang didekomposisi yang berupa desain/rancangan 	Menggunakan contoh Common Component Modeling Example(CoCoME) yang akan diidentifikasi. Evaluasi menentukan pendekatan ini lebih cepat dan tidak membutuhkan banyak usaha dalam mengidentifikasi microservice

3	Khaled Sellami, Mohamed Aymen Saied, dan Ali Ouni	A Hierarchical DBSCAN Method for Extracting Microservices from Monolithic Applications [9]	<ul style="list-style-type: none"> • Penggunaan Density-Based Spatial Clustering of Applications with Noise (DBSCAN) dalam melakukan pengelompokan • Pembuatan microservice yang dibuat secara hirarki yang dikombinasi dengan struktur dan semantik analisis • Penggunaan proses otomatis dalam proses dekomposisi 	Dengan pendekatan ini dekomposisi yang dihasilkan memiliki kohesi yang lebih baik dan lebih sedikit interaksi antar service. Namun perlu dilakukan implementasi dan perbandingan antara teknik dekomposisi lainnya
---	---	--	--	--

4	Shanshan Li, He Zhang, Zijia Jia, Zheng Li, Cheng Zhang, Jiaqi Li, Qiuya Gao, Jidong Ge, dan Zhihao Shan	A dataflow-driven approach to identifying microservices from monolithic applications [10]	<ul style="list-style-type: none"> • Menggabungkan pembuatan Data Flow Driven (DFD) dengan masing-masing detail yang berbeda • Membandingkan 2 skenario bisnis yang dievaluasi. • Menggunakan metrik coupling dan cohesion untuk mengevaluasi hasil dari dekomposisi. 	Untuk mengevaluasi dilakukan dengan melihat nilai coupling dan cohesionnya. Ditemukan bahwa dengan ini identifikasi lebih mudah dioperasikan, mudah dipahami, namun proses ini mungkin membutuhkan waktu yang lama dan bisa tidak efisien
---	--	---	--	---

5	Chaitanya K. Rudrabhatla	Impacts of Decomposition Techniques on Performance and Latency of Microservices [3]	<ul style="list-style-type: none"> • Teknik dekomposisi yang umumnya digunakan • Manfaat serta kekurangan dari setiap dekomposisi • Mencari ampak teknik dekomposisi pada latensi dan kinerja dari sistem yang hasilnya dibandingkan. 	Dari hasil evaluasi diketahui dekomposisi dengan domain driven memiliki performa lebih baik daripada pendekatan melalui entitas. Namun dengan pendekatan hybrid/campuran performa antara domain driven memiliki kesamaan sehingga diperlukan transaksi yang kompleks untuk melihat perbedaan
---	--------------------------	---	--	--

Pada penelitian Munezero Immaculée Josélyne, kendala yang muncul ketika membuat microservice yaitu bagaimana mengetahui batasan antara komponen service. Dengan menggunakan Domain Driven Design bisa memberikan beberapa cara untuk mengetahui batasan pada domain yang besar dan kompleks. Metode Domain Driven Design direkomendasikan dalam proses perancangan microservice.

Kemudian penelitian yang menggunakan pendekatan secara functional dilakukan dari mengidentifikasi microservice dengan spesifikasi use-case dari software requirements. Kemudian melakukan dekomposisi melalui tool visualisasi dimana visualisasi dihasilkan dari relasi antar objek, operasi dan variabel yang berasal dari source code. Selain itu ada pendekatan dengan menggunakan pengelompokan khusus menggunakan hierarchical clustering yang dikombinasi

dengan struktur dan analisis semantic. Dengan ini bisa mendeteksi outlier. Hasil dari pengelompokan akan dikomparasi dan dilakukan evaluasi.

Pendekatan dataflow-driven juga dapat membuat identifikasi secara sistematis dan mudah dipahami dalam melakukan dekomposisi dibandingkan dengan cara manual lainnya. Proses yang dilakukan yaitu dimulai dari menggunakan Use Case Specification dan Logic Bisnis dan kemudian diubah menjadi Data Flow diagram. Hasil dari diagram ini akan memberikan kandidat microservice yang bisa dibuat.

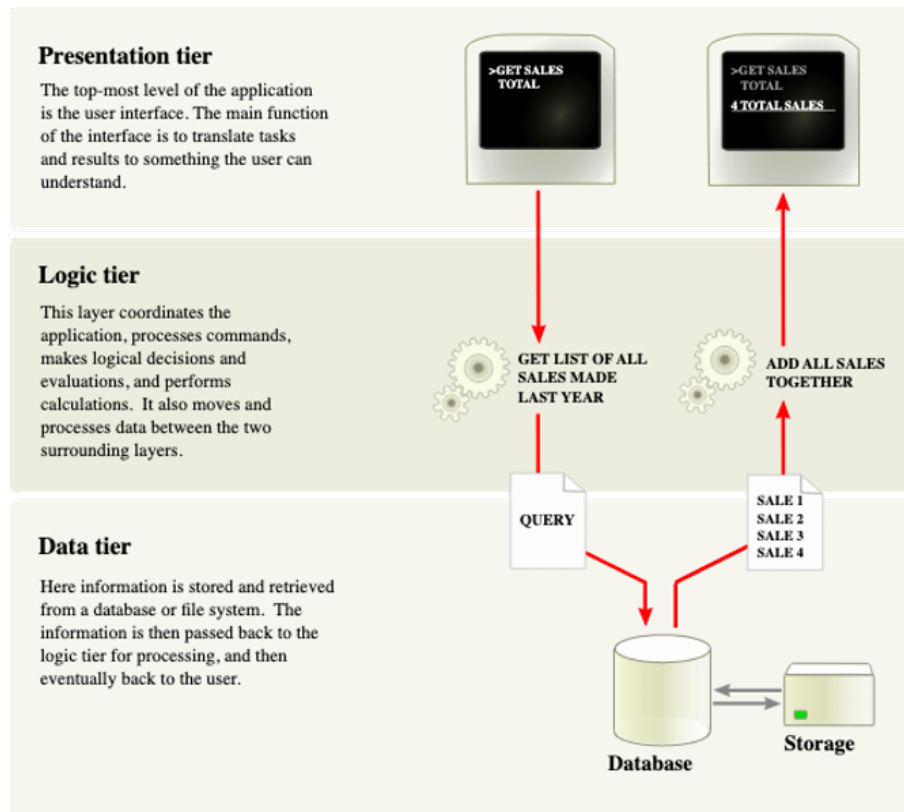
Penelitian lainnya berfokus pada dampak dari pendekatan masing-masing dekomposisi karena bila dekomposisi tidak dilakukan dengan baik maka bisa terjadi permasalahan pada latensi, kompleksitas dan ketidakefisienan. Penelitian ini menggunakan skenario e-commerce pada aplikasi microservice yang didekomposisi dengan pendekatan yang berbeda.

2.3 Tinjauan Objek

Pada bagian ini akan dijelaskan mengenai objek dan aplikasi terkait yang akan digunakan dalam tugas akhir ini. Object yang digunakan adalah sebuah aplikasi Enterprise Resource Planning yang di deploy secara monolit, yaitu Odoo.

Odoo merupakan aplikasi bisnis open source yang dapat mencakup semua kebutuhan perusahaan seperti CRM(Customer Relationship Management), eCommerce, akuntansi, inventaris, POS(Point of Sales), manajemen proyek dan lainnya. Aplikasi ini flexibel karena bisa dikembangkan lebih lanjut bila diperlukan dan bisa diubah karena memiliki lisensi source code yang terbuka.

Arsitektur yang digunakan pada Odoo yaitu three-tier arsitektur dimana tampilan, aturan bisnis dan tempat penyimpanan data memiliki lapisan terpisah. Dengan tujuan memudahkan dan mempercepat pengembang untuk melakukan modifikasi aplikasi tanpa harus mengganggu lapisan lainnya.



Gambar 2.1 Ilustrasi Arsitektur Odoo

Pada tingkatan paling atas yaitu tampilan(presentation tier), tampilan ini yang akan berinteraksi langsung dengan pengguna yang menggunakan aplikasi. Tampilan ini dibangun dengan teknologi web yaitu HTML5, Javascript, dan CSS. Tingkatan dibawahnya yaitu aturan bisnis(logic tier) yang berisi instruksi yang memproses data dan memberikan tanggapan dari interaksi kepada pengguna. Aturan pada Odoo hanya ditulis dalam bahasa pemrograman Python. Sedangkan pada tingkat paling bawah adalah tempat penyimpanan menggunakan DBMS(Database Management System), Odoo hanya bisa mendukung database PostgreSQL.

Odoo memiliki struktur kode yang dibentuk sebagai module untuk setiap fiturnya. Sehingga dari sisi server dan client memiliki hubungan yang disatukan menjadi satu paket tersendiri. Dimana module adalah koleksi dari fungsi dan data untuk menyelesaikan satu tujuan. Modul pada Odoo bisa ditambahkan, diganti, diubah untuk menyesuaikan kebutuhan bisnis. Dimana pada pengguna module dilambangkan dengan nama Apps, tetapi tidak semua module adalah Apps. Modules juga bisa direfrensikan sebagai addons.

Tabel 2.2 Komposisi dari Module pada aplikasi Odoo

Elemen	Keterangan	Contoh
Business Objects	Object yang akan digunakan di module dimana setiap attribute secara otomatis dipetakan ke kolom database dengan ORM	File python yang memiliki class
Objects Views	Menangani bagaimana data ditampilkan di pengguna. Seperti visualisasi form, list, kanban dan lainnya	Berupa file XML dengan struktur yang sudah ditentukan Odoo
Data Files	Mengelola bagaimana model data seperti laporan, konfigurasi data, data contoh dan lainnya	Berupa file XML atau CSV
Web Controllers	Menangani permintaan dari browser/client	File python yang memiliki class namun merupakan turunan dari class <code>odoo.http.Controller</code>
Static Web Data	File yang digunakan hanya ditampilkan kepada client di website	File gambar, File CSS, dan File JavaScript

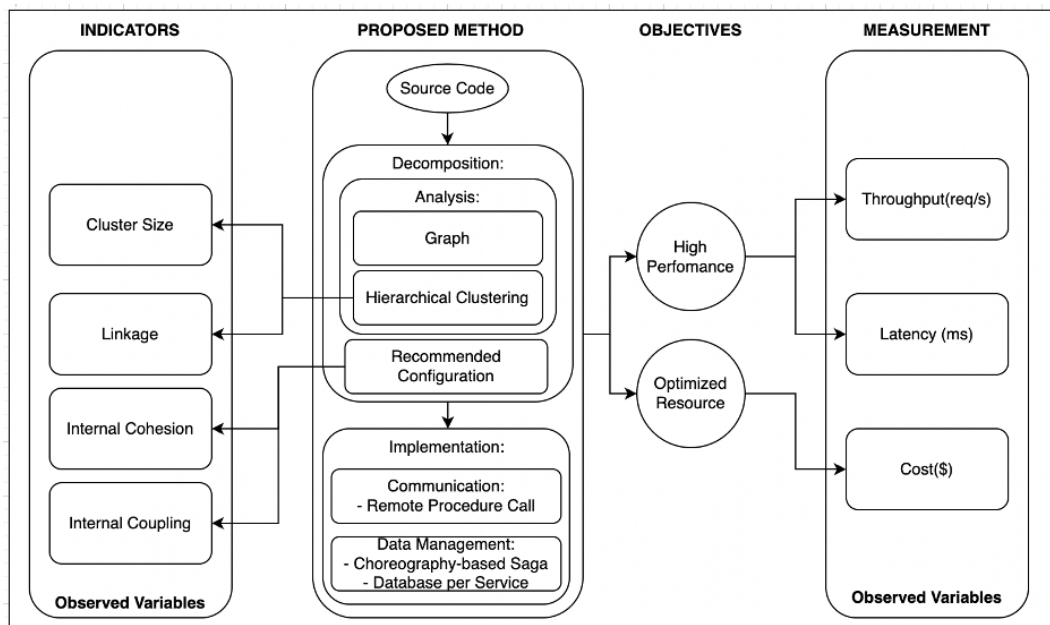
Struktur database yang dibentuk pada konfigurasi umum di Odoo memiliki jumlah tabel ±566 tabel, tabel ini merupakan keseluruhan dari aplikasi dimana dapat diidentifikasi 31 tabel utama yang digunakan pada aplikasi. Berikut adalah diagram dari database yang dibuat dengan alat DBeaver Visualize, dengan attribute hanya sebuah key dari tabel.

BAB 3 ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Masalah

Arsitektur yang digunakan Odoo(ERP) masih berupa monolith sehingga memiliki kelemahan seperti skalabilitas, sulit dikembangkan berkelanjutan, dan terkunci pada teknologi tertentu. Arsitektur Microservice dapat menyelesaikan masalah tersebut Namun diperlukan proses dekomposisi dari mono-to-micro. Proses dekomposisi sendiri tidak mudah dan melelahkan karena masih membutuhkan proses manual. Proses manual ini bisa di"mudahkan" dengan melakukan clustering khususnya dengan metode hierarchichal Sebelum melakukan proses clustering diperlukan analisis kode seperti Call Graph untuk mendapatkan graph dari source code aplikasi Odoo (ERP) Hasil terbaik dari clustering diimplementasikan menjadi service, service dan dilakukan pemecahan dengan pattern yang sudah ditemukan Dilakukan evaluasi dari aplikasi yang dibangun untuk menentukan apakah microservice dan clustering bisa melakukan dekomposisi dapat menyelesaikan permasalahan pada kasus ERP

3.2 Kerangka Pemikiran



Gambar 3.1 Kerangka Pemikiran

Penelitian akan dimulai dengan menggunakan kode sumber aplikasi yang dibuat dengan monolit. Kode sumber ini dilakukan proses dekomposisi yaitu dengan analisis seperti mencari objek beserta atributnya, untuk mencari keterhubungan lebih lanjut tentang objek maka dilakukan pencarian pada

fungsi-fungsi sehingga terbentuklah grafik yang menunjukkan bagaimana keterhubungan objek di aplikasi.

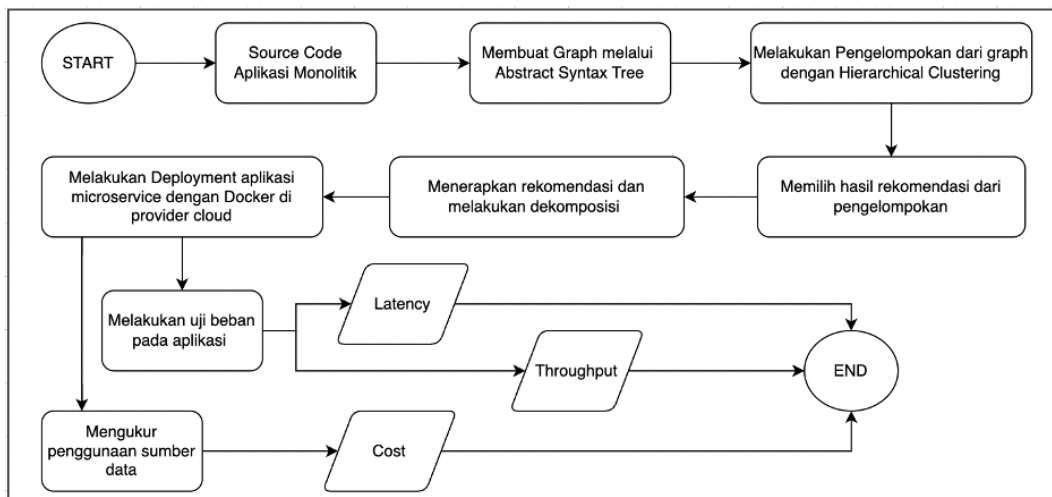
Dari grafik yang sudah dibuat akan dilakukan pengelompokan dengan pendekatan Hierarchical Clustering. Dimana perlu ditentukan jumlah cluster minimum yang harus ditemukan dan pemilihan algoritma Linkage. Metode linkage yaitu menentukan jarak atau kemiripan antara semua objek. Untuk menentukan jarak ini bisa dengan rata-rata, maximum, minimum dan mengecilkan variance.

Pengelompokan dari Hierarchical Clustering akan dipilih dengan mencari nilai cohesion terendah dan nilai coupling tertinggi. Dimana Internal Coupling mengevaluasi tingkat ketergantungan langsung dan tidak langsung antar objek. Semakin banyak dua objek menggunakan metode masing-masing semakin mereka menjadi satu kesatuan. Sedangkan Internal Cohesion akan mengevaluasi kekuatan interaksi antar objek. Biasanya, dua objek atau lebih menjadi interaktif jika metodenya bekerja pada atribut yang sama.

Ketika analisis dekomposisi sudah selesai dilakukan maka akan dilakukan implementasi berdasarkan pengelompokannya masing-masing yang akan menjadi service. Untuk metode komunikasinya antara service yaitu dengan Remote Procedure Call(RPC) dan untuk mengelola data, setiap service memiliki databasenya masing-masing dan untuk menjaga konsistensi data antar service maka digunakan pendekatan SAGA dengan cara choreography.

Untuk mengetahui bagaimana performa dari microservice dibandingkan dengan monolit yaitu dengan test beban. Pengukuran performa dilihat dari throughput, jumlah response, dan latency.

3.3 Urutan Proses Global



Gambar 3.2 Diagram Flowchart Proses Global

3.3.1 Proses Clustering

3.X.X Flowchart Proses Clustering

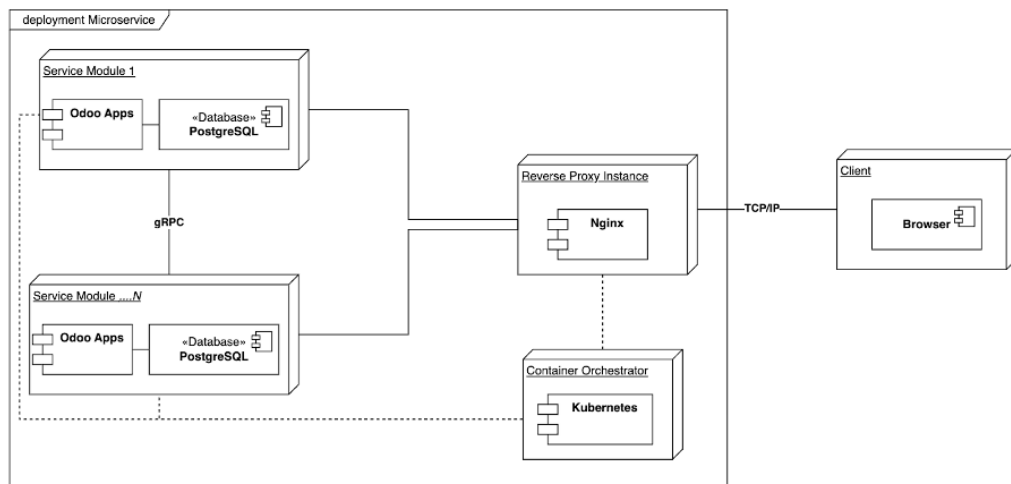
1. Pengambilan Source Code Mengunduh dari branch main/master terbaru dan semua test workflow pass
2. Pembuatan Call Graph Menggunakan library call graph (pycg atau pycallgraph)
3. Ilustrasi Graph Ilustrasi menggunakan tools graphviz
4. Extraksi Call Graph Proses ekstraksi json
5. Pengelompokan dari graph Menggunakan python dan library hirarki clustering Memilih hasil rekomendasi

3.3.2 Dekomposisi Monolitik ke Microservice

Komponen Diagram monolith VS komponen diagram yang diolah dari call graph

Sequence Diagram antar service dan module

3.3.3 Deployment



Gambar 3.3 Diagram Deployment

3.3.4 Evaluasi

Latency & Throughput: JMeter

Cost: AWS Cost Management (Cost Explorer)

BAB 4 IMPLEMENTASI DAN PENGUJIAN

4.1 Lingkungan Implementasi

Mea tale aliquam minimum te. Eu mel putant virtute, essent inermis nominavi mea no. Laoreet indoctum sea te. Te scripta fabulas duo, pro doming recusabo voluptaria at. Cu sed numquam inciderint, ei minim altera disputando cum, te nec graeco maiorum convenire.

Cu mel putent rationibus dissentiet. Per vidisse scaevola oportere ei, qui solet molestie eu. Hinc diceret nominati per at, nec dico denique laboramus et. Legere regione his at, aequae decore in mei. Aliquam tincidunt a nulla ac posuere. Maecenas sapien mi, feugiat sit amet tellus at, dictum varius ante. Cras rutrum facilisis felis at hendrerit. Nullam eleifend sed lorem a iaculis. Donec ut odio at nisl molestie euismod quis et purus. Curabitur eu ex turpis. Etiam maximus metus non iaculis placerat. Sed in risus sodales, posuere elit in, eleifend tellus. Mauris at consectetur arcu. Integer fringilla eros mi, vel volutpat enim commodo ac.

4.1.1 Spesifikasi Perangkat Keras

Suspendisse ac porta diam, ut viverra ante. Aliquam mattis tincidunt diam in molestie. Sed auctor fermentum turpis, sed varius ante. Nulla rutrum, enim et efficitur dignissim, urna diam consequat purus, sit amet elementum nibh mauris ut tellus. Quisque interdum leo ligula, a volutpat mauris viverra ut. Fusce ac felis finibus, convallis ligula a, aliquam nunc. Quisque faucibus ligula et ornare finibus. Morbi maximus dolor vitae dolor tristique, eu sagittis metus auctor. Pellentesque quam lacus, ornare ut est ut, egestas auctor leo. Duis eros neque, mollis quis elit id, cursus egestas neque. Pellentesque ac sapien vitae nulla varius rhoncus. Orci varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Etiam pharetra nisl et massa facilisis aliquet. Nulla sit amet quam enim. Nunc dictum pellentesque orci, at sollicitudin erat condimentum eu. Nullam mi dolor, vestibulum at lacinia quis, feugiat faucibus felis.

Suspendisse ac porta diam, ut viverra ante. Aliquam mattis tincidunt diam in molestie. Sed auctor fermentum turpis, sed varius ante. Nulla rutrum, enim et efficitur dignissim, urna diam consequat purus, sit amet elementum nibh mauris ut tellus. Quisque interdum leo ligula, a volutpat mauris viverra ut. Fusce ac felis finibus, convallis ligula a, aliquam nunc. Quisque faucibus ligula et ornare finibus. Morbi maximus dolor vitae dolor tristique, eu sagittis metus auctor. Pellentesque

quam lacus, ornare ut est ut, egestas auctor leo. Duis eros neque, mollis quis elit id, cursus egestas neque. Pellentesque ac sapien vitae nulla varius rhoncus. Orci varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Etiam pharetra nisl et massa facilisis aliquet. Nulla sit amet quam enim. Nunc dictum pellentesque orci, at sollicitudin erat condimentum eu. Nullam mi dolor, vestibulum at lacinia quis, feugiat faucibus felis.

4.1.2 Spesifikasi Perangkat Lunak

Suspendisse ac porta diam, ut viverra ante. Aliquam mattis tincidunt diam in molestie. Sed auctor fermentum turpis, sed varius ante. Nulla rutrum, enim et efficitur dignissim, urna diam consequat purus, sit amet elementum nibh mauris ut tellus. Quisque interdum leo ligula, a volutpat mauris viverra ut. Fusce ac felis finibus, convallis ligula a, aliquam nunc. Quisque faucibus ligula et ornare finibus. Morbi maximus dolor vitae dolor tristique, eu sagittis metus auctor. Pellentesque quam lacus, ornare ut est ut, egestas auctor leo. Duis eros neque, mollis quis elit id, cursus egestas neque. Pellentesque ac sapien vitae nulla varius rhoncus. Orci varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Etiam pharetra nisl et massa facilisis aliquet. Nulla sit amet quam enim. Nunc dictum pellentesque orci, at sollicitudin erat condimentum eu. Nullam mi dolor, vestibulum at lacinia quis, feugiat faucibus felis.

4.2 Implementasi Perangkat Lunak

Mea tale aliquam minimum te. Eu mel putant virtute, essent inermis nominavi mea no. Laoreet inductum sea te. Te scripta fabulas duo, pro doming recusabo voluptaria at. Cu sed numquam inciderint, ei minim altera disputando cum, te nec graeco maiorum convenire.

Cu mel putent rationibus dissentiet. Per vidisse scaevola oportere ei, qui solet molestie eu. Hinc diceret nominati per at, nec dico denique laboramus et. Legere regione his at, aequae decore in mei Mea tale aliquam minimum te. Eu mel putant virtute, essent inermis nominavi mea no. Laoreet inductum sea te. Te scripta fabulas duo, pro doming recusabo voluptaria at. Cu sed numquam inciderint, ei minim altera disputando cum, te nec graeco maiorum convenire. Cu mel putent rationibus dissentiet. Per vidisse scaevola oportere ei, qui solet molestie eu. Hinc diceret nominati per at, nec dico denique laboramus et. Legere regione his at, aequae decore in mei.

4.2.1 Implementasi *Class*

Cu sed numquam inciderint, ei minim altera disputando cum, te nec graeco maiorum convenire. Cu mel putent rationibus dissentiet. Per vidisse scaevola oportere ei, qui solet molestie eu. Hinc diceret nominati per at, nec dico denique laboramus et. Legere regione his at, aequae decore in mei.

4.2.1.1 *Class Nama_Class_1*

Cu sed numquam inciderint, ei minim altera disputando cum, te nec graeco maiorum convenire. Cu mel putent rationibus dissentiet. Per vidisse scaevola oportere ei, qui solet molestie eu. Hinc diceret nominati per at, nec dico denique laboramus et. Legere regione his at, aequae decore in mei.

4.2.1.2 *Class Nama_Class_2*

Cu sed numquam inciderint, ei minim altera disputando cum, te nec graeco maiorum convenire. Cu mel putent rationibus dissentiet. Per vidisse scaevola oportere ei, qui solet molestie eu. Hinc diceret nominati per at, nec dico denique laboramus et. Legere regione his at, aequae decore in mei.

4.2.2 Implementasi *Numquam*

Cu sed numquam inciderint, ei minim altera disputando cum, te nec graeco maiorum convenire. Cu mel putent rationibus dissentiet. Per vidisse scaevola oportere ei, qui solet molestie eu. Hinc diceret nominati per at, nec dico denique laboramus et. Legere regione his at, aequae decore in mei.

4.3 Implementasi *Nama_Implementasi*

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec ac felis dignissim, iaculis odio ut, euismod quam. Donec vestibulum pellentesque sem, eu aliquet purus lacinia ac. Nam porttitor auctor justo et lobortis. Orci varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Sed et gravida neque. Praesent commodo aliquam vestibulum. Vivamus blandit mattis mi ut euismod. Proin vitae vestibulum orci, eget elementum tellus. Suspendisse potenti.

4.4 Implementasi *Aplikasi*

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec ac felis dignissim, iaculis odio ut, euismod quam. Donec vestibulum pellentesque sem, eu aliquet purus lacinia ac. Nam porttitor auctor justo et lobortis. Orci varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Sed et gravida neque. Praesent commodo aliquam vestibulum. Vivamus blandit mattis mi ut

euismod. Proin vitae vestibulum orci, eget elementum tellus. Suspendisse potenti.

Integer non diam a sem venenatis iaculis. Suspendisse quam leo, ultrices sed mollis sit amet, sagittis sit amet nulla. Nam placerat enim in tellus convallis gravida nec quis ipsum. Sed a dapibus erat. Maecenas suscipit maximus turpis vel tempor. In cursus aliquet tellus id viverra. Aenean venenatis augue magna, at ullamcorper erat tincidunt nec. Etiam nec dolor efficitur, iaculis nulla in, semper mi. Ut consectetur aliquet ex, a tincidunt nisi vulputate non. Proin mauris sapien, ultricies sit amet arcu bibendum, molestie suscipit mi. Mauris laoreet facilisis augue, et interdum purus vehicula sit amet. Fusce porta condimentum cursus.

4.5 Pengujian

Quisque dictum auctor tempor. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Vestibulum ultricies justo elit, sed tincidunt tellus congue quis. Suspendisse potenti. In iaculis volutpat odio sed placerat. Nullam est purus, egestas sit amet sagittis sit amet, eleifend in nisl. Nullam vitae auctor dolor. Nulla non laoreet dolor. Quisque nibh enim, bibendum sit amet tristique sit amet, efficitur nec tellus. Nullam congue ex felis, quis aliquam purus vulputate in. Aliquam in euismod neque. Sed quis odio non ex molestie posuere. Aenean efficitur id ex ut faucibus. Suspendisse imperdiet mattis ipsum, viverra efficitur ligula. Nulla varius lacus massa, ut egestas turpis consequat in. Sed et finibus orci, id tincidunt velit.

4.5.1 Pengujian Nama Pengujian_1

Cu sed numquam inciderint, ei minim altera disputando cum, te nec graeco maiorum convenire. Cu mel putent rationibus dissentiet. Per vidisse scaevola oportere ei, qui solet molestie eu. Hinc diceret nominati per at, nec dico denique laboramus et. Legere regione his at, aequae decore in mei.

Tabel 4.1 atribut pada *class* nama_class_1

atribut:					
Float	C	Float	tol	Float	gamma
Float	a	Float	r	Integer	pos_true
Integer	pos_pred	Integer	net_true	Integer	net_pred
Integer	neg_true	Integer	neg_pred	Float	accuracy_score

Float	precision_score	Float	recall_score	Float	f_score
-------	-----------------	-------	--------------	-------	---------

Tabel 4.2 Daftar *method* pada *class helper*

No.	Method	Masukan	Luaran	Keterangan
1.	<code>__init__</code>	-	-	Konstruktor yang menginisialisasi objek Training dimana proses inisialisasi parameter CNN juga dilakukan.
2.	<code>auto_training</code>	-	float[] float[]	Menjalankan alur proses <i>training</i> secara keseluruhan dimulai dari pengambilan citra <i>host</i> dan <i>watermark</i> dari direktori <i>local</i> , penyisipan <i>watermark</i> , ekstraksi <i>embedding map</i> , hingga pemrosesan <i>embedding map</i> dengan CNN. Fungsi mengembalikan nilai <i>loss</i> dari akurasi.
3.	<code>normalize_watermark</code>	image : float[][]	float[][]	Memroses citra watermark agar dapat digunakan untuk <i>training</i> .
4.	<code>apply_transformations</code>	image : float[][]	float[][]	Menjalankan seluruh transformasi digital pada citra dan menyimpannya sebagai <i>array</i> .
		image : float[][] iswatermark : boolean		

5.	get _embedding_maps	images : float[][] float[][] key : string	float[][]	Mengambil <i>embedding map</i> dari setiap citra yang telah disisipi watermark.
6.	divide _training_images	images : float[][] ground_truth : float[][]	-	Membagi <i>embedding map</i> dan citra <i>ground truth</i> ke dalam <i>batch</i> sesuai <i>batch size</i> yang telah ditentukan.
7.	cross_entropy _per_batch	images : float[][] ground_truth : float[][]	float[][]	menghitung nilai <i>loss</i> setiap citra dalam satu <i>batch</i> terhadap citra <i>ground truth</i> .
8.	run	-	float[] float[]	Menjalankan proses <i>training CNN</i> . Fungsi mengembalikan hasil <i>training</i> dan <i>loss</i> terakhir.
9.	store_params	-	-	Menyimpan seluruh parameter CNN ke dalam direktori <i>local</i> .
10.	normalize _watermark	images : float[][]	float[][]	Menyamakan ukuran dan tipe data watermark.

4.5.2 Pengujian Nama Pengujian 2

Cu sed numquam inciderint, ei minim altera disputando cum, te nec graeco maiorum convenire. Cu mel putent rationibus dissentiet. Per vidisse scaevola oportere ei, qui solet molestie eu. Hinc diceret nominati per at, nec dico denique laboramus et. Legere regione his at, aequae decore in mei.

BAB 5 KESIMPULAN DAN SARAN

5.1 Kesimpulan

Mea tale aliquam minimum te. Eu mel putant virtute, essent inermis nominavi mea no. Laoreet indoctum sea te. Te scripta fabulas duo, pro doming recusabo voluptaria at. Cu sed numquam inciderint, ei minim altera disputando cum, te nec graeco maiorum convenire. Cu mel putent rationibus dissentiet. Per vidisse scaevola oportere ei, qui solet molestie eu. Hinc diceret nominati per at, nec dico denique laboramus et. Legere regione his at, aequae decore in mei Mea tale aliquam minimum te. Eu mel putant virtute, essent inermis nominavi mea no. Laoreet indoctum sea te. Te scripta fabulas duo, pro doming recusabo voluptaria at. Cu sed numquam inciderint, ei minim altera disputando cum, te nec graeco maiorum convenire. Cu mel putent rationibus dissentiet. Per vidisse scaevola oportere ei, qui solet molestie eu. Hinc diceret nominati per at, nec dico denique laboramus et. Legere regione his at, aequae decore in mei. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec ac felis dignissim, iaculis odio ut, euismod quam. Donec vestibulum pellentesque sem, eu aliquet purus lacinia ac. Nam porttitor auctor justo et lobortis. Orci varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Sed et gravida neque. Praesent commodo aliquam vestibulum. Vivamus blandit mattis mi ut euismod. Proin vitae vestibulum orci, eget elementum tellus. Suspendisse potenti.

Integer non diam a sem venenatis iaculis. Suspendisse quam leo, ultrices sed mollis sit amet, sagittis sit amet nulla. Nam placerat enim in tellus convallis gravida nec quis ipsum. Sed a dapibus erat. Maecenas suscipit maximus turpis vel tempor. In cursus aliquet tellus id viverra. Aenean venenatis augue magna, at ullamcorper erat tincidunt nec. Etiam nec dolor efficitur, iaculis nulla in, semper mi. Ut consectetur aliquet ex, a tincidunt nisi vulputate non. Proin mauris sapien, ultricies sit amet arcu bibendum, molestie suscipit mi. Mauris laoreet facilisis augue, et interdum purus vehicula sit amet. Fusce porta condimentum cursus.

5.2 Saran

Mea tale aliquam minimum te. Eu mel putant virtute, essent inermis nominavi mea no. Laoreet indoctum sea te. Te scripta fabulas duo, pro doming recusabo voluptaria at. Cu sed numquam inciderint, ei minim altera disputando cum, te nec graeco maiorum convenire. Cu mel putent rationibus dissentiet. Per

vidisse scaevola oportere ei, qui solet molestie eu. Hinc diceret nominati per at, nec dico denique laboramus et. Legere regione his at, aequae decore in mei Mea tale aliquam minimum te. Eu mel putant virtute, essent inermis nominavi mea no. Laoreet indoctum sea te. Te scripta fabulas duo, pro doming recusabo voluptaria at. Cu sed numquam inciderint, ei minim altera disputando cum, te nec graeco maiorum convenire. Cu mel putent rationibus dissentiet. Per vidisse scaevola oportere ei, qui solet molestie eu. Hinc diceret nominati per at, nec dico denique laboramus et. Legere regione his at, aequae decore in mei. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec ac felis dignissim, iaculis odio ut, euismod quam. Donec vestibulum pellentesque sem, eu aliquet purus lacinia ac. Nam porttitor auctor justo et lobortis. Orci varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Sed et gravida neque. Praesent commodo aliquam vestibulum. Vivamus blandit mattis mi ut euismod. Proin vitae vestibulum orci, eget elementum tellus. Suspendisse potenti.

DAFTAR REFERENSI

- [1] Amini, Mohammad and Abukari, Arnold. (2020). "ERP Systems Architecture For The Modern Age: A Review of The State of The Art Technologies." *Journal of Applied Intelligent Systems and Information Sciences*. Volume 1(2), pp.70-90. Available: <https://doi.org/10.22034/jaisis.2020.232506.1009> . [Accessed: 27-Oct-2022].
- [2] Bender, B.; Bertheau, C. and Gronau, N. (2021). "Future ERP Systems: A Research Agenda." In *Proceedings of the 23rd International Conference on Enterprise Information Systems*. 2, pp.776-783. Available: <http://dx.doi.org/10.5220/0010477307760783>. [Accessed: 27-Oct-2022]
- [3] Chaitanya K. Rudrabhatla. (2020). "Impacts of Decomposition Techniques on Performance and Latency of Microservices." *International Journal of Advanced Computer Science and Applications(IJACSA)*. 11(8). Available: <http://dx.doi.org/10.14569/IJACSA.2020.0110803>. [Accessed: 27-Oct-2022]
- [4] Slamaa, A.A., El-Ghareeb, H.A. , Saleh, A.A. (2021). "A Roadmap for Migration System-Architecture Decision by Neutrosophic-ANP and Benchmark for Enterprise Resource Planning Systems." *IEEE Access* . 9, pp.48583-48604. Available: <https://doi.org/10.1109/ACCESS.2021.3068837>. [Accessed: 27-Oct-2022]
- [5] Söylemez, M.; Tekinerdogan, B.; Kolukısa Tarhan. (2022). "A. Challenges and Solution Directions of Microservice Architectures: A Systematic Literature Review Planning Systems." *Applied Science* . 12(11), pp.48583-48604. Available: <https://doi.org/10.3390/app12115507>. [Accessed: 27-Oct-2022]
- [6] Sam Newman, *Monolith to Microservices*, Sebastopol, CA: O'Reilly Media, Inc., 2020, pp. 12-15 [[Link GoogleDrive](#)]
- [7] Munezero Immaculée Josélyne, Doreen Tuheirwe-Mukasa, Benjamin Kanagwa, and Joseph Balikuddembe. (2018, May). "Partitioning microservices: a domain engineering approach." In *Proceedings of the 2018 International Conference on Software Engineering in Africa (SEiA '18)*. May 2018, pp-43-49. Available: <https://doi.org/10.1145/3195528.3195535>. [Accessed: 27-Oct-2022]

- [8] Tyszberowicz, S., Heinrich, R., Liu, B., Liu, Z. (2018). "Identifying Microservices Using Functional Decomposition." Dependable Software Engineering. Theories, Tools, and Applications. SETTA 2018. vol 10998. Springer, Cham. Available: https://doi.org/10.1007/978-3-319-99933-3_4. [Accessed: 27-Oct-2022]
- [9] Khaled Sellami, Mohamed Aymen Saied, and Ali Ouni. (2022). "A Hierarchical DBSCAN Method for Extracting Microservices from Monolithic Applications" In The International Conference on Evaluation and Assessment in Software Engineering 2022 (EASE 2022). ACM, New York, NY, USA, 11. Available: <https://doi.org/10.1145/3530019.3530040>. [Accessed: 27-Oct-2022]
- [10] Shanshan Li, He Zhang, Zijia Jia, Zheng Li, Cheng Zhang, Jiaqi Li, Qiuya Gao, Jidong Ge, Zhihao Shan. (2019). "A dataflow-driven approach to identifying microservices from monolithic applications." Journal of Systems and Software. Volume 157. Available: <https://doi.org/10.1016/j.jss.2019.07.008>. [Accessed: 27-Oct-2022]

LAMPIRAN A LAMPIRAN A

ASJDBAKJSDBKA

Tabel A-1 *Lorem ipsum*

No	<i>Dolor sit amet</i>	At sonet	Vim commune	At quo congue	Cum iisque
1	Ei utroque electram	0	0	10	Laudem
2	Ei utroque electram	3	0	7	Laudem
3	Ei utroque electram	2	0	8	Laudem
4	Ei utroque electram	0	3	7	Laudem
5	Ei utroque electram	10	0	0	Laudem
6	Ei utroque electram	0	0	10	Laudem
7	Ei utroque electram	3	0	7	Laudem
8	Ei utroque electram	2	0	8	Laudem
9	Ei utroque electram	0	3	7	Laudem
10	Ei utroque electram	10	0	0	Laudem
11	Ei utroque electram	0	0	10	Laudem
12	Ei utroque electram	3	0	7	Laudem
13	Ei utroque electram	2	0	8	Laudem
14	Ei utroque electram	0	3	7	Laudem
15	Ei utroque electram	10	0	0	Laudem
16	Ei utroque electram	0	0	10	Laudem
17	Ei utroque electram	3	0	7	Laudem
18	Ei utroque electram	2	0	8	Laudem
19	Ei utroque electram	0	3	7	Laudem
20	Ei utroque electram	10	0	0	Laudem
21	Ei utroque electram	0	0	10	Laudem
22	Ei utroque electram	3	0	7	Laudem
23	Ei utroque electram	2	0	8	Laudem
24	Ei utroque electram	0	3	7	Laudem
25	Ei utroque electram	10	0	0	Laudem

Tabel A-1 *Lorem ipsum*

No	<i>Dolor sit amet</i>	At sonet	Vim commune	At quo congue	Cum iisque
26	Ei utroque electram	0	0	10	Laudem
27	Ei utroque electram	3	0	7	Laudem
28	Ei utroque electram	2	0	8	Laudem
29	Ei utroque electram	0	3	7	Laudem
30	Ei utroque electram	10	0	0	Laudem
31	Ei utroque electram	0	0	10	Laudem
32	Ei utroque electram	3	0	7	Laudem
33	Ei utroque electram	2	0	8	Laudem
34	Ei utroque electram	0	3	7	Laudem
35	Ei utroque electram	10	0	0	Laudem
36	Ei utroque electram	0	0	10	Laudem
37	Ei utroque electram	3	0	7	Laudem
38	Ei utroque electram	2	0	8	Laudem
39	Ei utroque electram	0	3	7	Laudem
40	Ei utroque electram	10	0	0	Laudem

LAMPIRAN B DATASET HASIL KUISIONER 2

Tabel B-1 *Lorem ipsum*

No	<i>Dolor sit amet</i>	At sonet	Vim commune	At quo congue	Cum iisque
1	Ei utroque electram	0	0	10	Laudem
2	Ei utroque electram	3	0	7	Laudem
3	Ei utroque electram	2	0	8	Laudem
4	Ei utroque electram	0	3	7	Laudem
5	Ei utroque electram	10	0	0	Laudem
6	Ei utroque electram	0	0	10	Laudem
7	Ei utroque electram	3	0	7	Laudem
8	Ei utroque electram	2	0	8	Laudem
9	Ei utroque electram	0	3	7	Laudem
10	Ei utroque electram	10	0	0	Laudem
11	Ei utroque electram	0	0	10	Laudem
12	Ei utroque electram	3	0	7	Laudem
13	Ei utroque electram	2	0	8	Laudem
14	Ei utroque electram	0	3	7	Laudem
15	Ei utroque electram	10	0	0	Laudem
16	Ei utroque electram	0	0	10	Laudem
17	Ei utroque electram	3	0	7	Laudem
18	Ei utroque electram	2	0	8	Laudem
19	Ei utroque electram	0	3	7	Laudem
20	Ei utroque electram	10	0	0	Laudem
21	Ei utroque electram	0	0	10	Laudem
22	Ei utroque electram	3	0	7	Laudem
23	Ei utroque electram	2	0	8	Laudem
24	Ei utroque electram	0	3	7	Laudem
25	Ei utroque electram	10	0	0	Laudem
26	Ei utroque electram	0	0	10	Laudem

Tabel B-1 *Lorem ipsum*

No	<i>Dolor sit amet</i>	At sonet	Vim commune	At quo congue	Cum iisque
27	Ei utroque electram	3	0	7	Laudem
28	Ei utroque electram	2	0	8	Laudem
29	Ei utroque electram	0	3	7	Laudem
30	Ei utroque electram	10	0	0	Laudem
31	Ei utroque electram	0	0	10	Laudem
32	Ei utroque electram	3	0	7	Laudem
33	Ei utroque electram	2	0	8	Laudem
34	Ei utroque electram	0	3	7	Laudem
35	Ei utroque electram	10	0	0	Laudem
36	Ei utroque electram	0	0	10	Laudem
37	Ei utroque electram	3	0	7	Laudem
38	Ei utroque electram	2	0	8	Laudem
39	Ei utroque electram	0	3	7	Laudem
40	Ei utroque electram	10	0	0	Laudem