STRATEGI *TEST-DRIVEN DEVELOPMENT* DALAM ARSITEKTUR *MICROSERVICES* UNTUK OPTIMALISASI PENGEMBANGAN APLIKASI *PAYROLL*

SKRIPSI

AI DINA AGUSTIN
20210040065

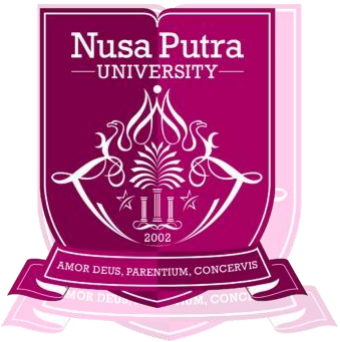
PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK, KOMPUTER DAN DESAIN
UNIVERSITAS NUSA PUTRA
SUKABUMI
2025

STRATEGI *TEST-DRIVEN DEVELOPMENT* DALAM ARSITEKTUR *MICROSERVICES* UNTUK OPTIMALISASI PENGEMBANGAN APLIKASI *PAYROLL*

SKRIPSI

Diajukan Untuk Memenuhi Salah Satu Syarat Dalam Menempuh
Gelar Sarjana Komputer

AI DINA AGUSTIN
20210040065



PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK, KOMPUTER DAN DESAIN
UNIVERSITAS NUSA PUTRA
SUKABUMI
2025

PERNYATAAN PENULIS

JUDUL	:	STRATEGI *TEST-DRIVEN DEVELOPMENT* DALAM ARSITEKTUR *MICROSERVICES* UNTUK OPTIMALISASI PENGEMBANGAN APLIKASI *PAYROLL*
NAMA	:	AI DINA AGUSTIN
NIM	:	20210040065

"Saya menyatakan dan bertanggungjawab dengan sebenarnya bahwa Skripsi ini adalah hasil karya sendiri kecuali cuplikan dan ringkasan yang masing-masing telah saya jelaskan sumbernya. Jika pada waktu selanjutnya ada pihak lain yang mengklaim bahwa Skripsi ini sebagai karyanya, yang disertai dengan bukti-bukti yang cukup, maka saya bersedia untuk dibatalkan gejar Sarjana Komputer saya beserta segala hak dan kewajiban yang melekat pada gelar tersebut".

Nusa Putra
UNIVERSITY
2002
AMOR DEUS, PARENTIUM, CONCERVIS

Sukabumi, 15 Juli 2025

AI DINA AGUSTIN

Penulis

PENGESAHAN SKRIPSI

JUDUL	:	STRATEGI *TEST-DRIVEN DEVELOPMENT* DALAM ARSITEKTUR *MICROSERVICES* UNTUK OPTIMALISASI PENGEMBANGAN APLIKASI *PAYROLL*
NAMA	:	AI DINA AGUSTIN
NIM	:	20210040065

Skripsi ini telah diujikan dan dipertahankan di depan Dewan Penguji pada Sidang Skripsi tanggal 15 Juli 2025 Menurut pandangan kami, Skripsi ini memadai dari segi kualitas untuk tujuan penganugerahan gelar Sarjana Komputer (S.Kom)

Sukabumi, 15 Juli 2025

Pembimbing I	Pembimbing II
Alun Sujjada, S.Kom., M.T	**Anggun Fergina, M.Kom**
NIDN. 0718108001	NIDN. 0407029301
Ketua Penguji	Ketua Program Studi

Muhammad Ikhsan Thohir, M.Kom	**Ir Somantri, S.T., M.Kom**
NIDN. 0415049302	NIDN. 0419128801

PLH. Dekan Fakultas Teknik, Komputer dan Desain

Ir. Paikun, S.T., M.T., IPM., ASEAN Eng
NIDN. 0402037401

ABSTRACT

The development of the Payroll application at PT Sineas Kreatif Indonesia *faces challenges in testing inter-service communication due to its microservices architecture. This study evaluates the effectiveness of the Test-Driven Development (*TDD*) approach in optimizing system development.* TDD *is implemented following the Red-Green-Refactor cycle using three unit testing frameworks: Jest, Mocha, and Chai. The research employs a mixed methods approach, combining qualitative observation with quantitative metrics such as code coverage and defect rate.*

The findings indicate that TDD *improves system reliability through early error detection, enhances code quality with code coverage exceeding 96%, and reduces the defect rate to 4.33 per 1000* LOC. *The adopted microservices architecture also demonstrates fault tolerance across services. A comparison with the non-*TDD *approach shows significant improvements in performance and system quality.*

Keywords: *Test-Driven Development, Microservices, Payroll, Unit Testing, Code Coverage, Defect Rate*

Nusa Putra UNIVERSITY 2002 AMOR DEUS, PARENTIUM, CONCERVIS

ABSTRAK

Pengembangan aplikasi *Payroll* di PT Sineas Kreatif Indonesia menghadapi tantangan dalam pengujian komunikasi antar-layanan karena menggunakan arsitektur microservices. Penelitian ini mengevaluasi efektivitas pendekatan *Test-Driven Development* (TDD) dalam mengoptimalkan pengembangan sistem tersebut. TDD diterapkan dengan mengikuti siklus *Red-Green-Refactor* menggunakan tiga framework unit testing, yaitu Jest, Mocha, dan Chai. Metode penelitian yang digunakan adalah *mixed methods*, yang menggabungkan pendekatan kualitatif melalui observasi dan pendekatan kuantitatif melalui pengukuran metrik seperti *code coverage* dan *defect rate*.

Hasil penelitian menunjukkan bahwa TDD mampu meningkatkan keandalan sistem dengan deteksi kesalahan lebih awal, meningkatkan kualitas kode dengan *code coverage* mencapai >96%, dan menurunkan defect rate hingga 4,33 per 1000 LOC. Arsitektur *microservices* yang digunakan juga menunjukkan toleransi terhadap kegagalan antar-layanan. Perbandingan dengan pendekatan non-TDD memperlihatkan peningkatan signifikan dalam performa dan kualitas sistem.

Kata kunci: *Test-Driven Development*, *Microservices*, *Payroll, Unit Testing, Code Coverage, Defect Rate*

Nusa Putra UNIVERSITY
AMOR DEUS, PARENTIUM, CONCERVIS

KATA PENGANTAR

Assalamualaikum Warahmatullahi Wabarakatuh

Segala puji dan syukur penulis panjatkan ke hadirat Allah SWT atas limpahan rahmat dan hidayah-Nya akhirnya penulis dapat menyelesaikan skripsi yang berjudul "Strategi *Test-Driven Development* dalam Arsitektur *Microservices* untuk Optimalisasi Pengembangan Aplikasi *Payroll*" sebagai salah satu syarat untuk memperoleh gelar Sarjana pada Program Studi Teknik Informatika.

Tujuan penulisan skripsi ini adalah untuk mengevaluasi penerapan pendekatan *Test-Driven Development* (TDD) dalam arsitektur *microservices* pada pengembangan aplikasi *Payroll*. Penelitian ini diharapkan dapat memberikan kontribusi dalam pengembangan perangkat lunak yang lebih efisien, handal, dan terstruktur.

Melalui kesempatan ini, penulis ingin menyampaikan rasa terima kasih yang sebesar-besarnya kepada semua pihak yang telah memberikan bantuan dan dukungan, sehingga penulisan skripsi ini dapat terselesaikan. Ucapan terima kasih secara khusus penulis tujukan kepada:

1. Panutanku, Bapak Badri Sanusi, terimakasih telah menjadi sosok luar biasa yang mampu mendidik dan membesarkan penulis dengan penuh kasih sayang, kerja keras, serta semangat yang tak pernah padam. Dukungan dan motivasi beliau menjadi fondasi utama yang mengantarkan penulis hingga berhasil menyelesaikan studi dan meraih gelar sarjana. Pintu Surgaku, ibu Mamah terima kasih yang tak terhingga penulis sampaikan atas segala do'a, cinta, pengorbanan, dan dukungan yang tanpa lelah diberikan selama ini. Ibu selalu menjadi sumber kekuatan di saat penulis merasa lelah dan putus asa. Setiap do'a yang ibu panjatkan dan setiap peluh yang ibu curahkan menjadi penerang di setiap langkah perjalanan ini. Maaf atas segala kesulitan dan waktu panjang yang harus ibu lalui bersama penulis. Terima kasih atas kesabaran, nasihat penuh makna, serta ketulusan yang tak pernah pudar, meskipun terkadang pemikiran kita tidak selalu sejalan. Ibu adalah tempat pulang terbaik dan sumber kekuatan terbesar dalam hidup penulis.
2. Rektor Universitas Nusa Putra, Bapak Dr. H. Kurniawan, ST., M.Si., MM.

3. Wakil Rektor I Bidang Akademik Universitas Nusa Putra, Bapak Anggy Pradiftha Junfitharana, S.Pd., MT.
4. Ketua Program Studi Teknik Informatika Universitas Nusa Putra, Bapak Ir. Somantri, ST., M.Kom., atas bimbingan, motivasi, dan kesempatan yang telah diberikan selama masa studi.
5. Dosen Pembimbing I, Bapak Alun Sujjada, S.Kom., MT., yang dengan penuh kesabaran dan dedikasi telah membimbing penulis dari awal hingga akhir proses penyusunan skripsi ini.
6. Dosen Pembimbing II, Ibu Anggun Fergina, M.Kom., yang telah memberikan banyak masukan, saran konstruktif, dan arahan yang sangat membantu dalam penyempurnaan skripsi ini.
7. Ucapan terima kasih juga penulis sampaikan kepada rekan-rekan seperjuangan satu angkatan yang telah menjadi teman belajar, diskusi, serta berbagi suka dan duka selama menempuh pendidikan. Semoga segala perjuangan dan impian kita dapat terwujud dengan baik di masa depan.
8. *Last but not least*, terima kasih untuk diri sendiri, yang telah mampu bertahan, berjuang, dan tidak menyerah di tengah berbagai tekanan dan tantangan. Terima kasih telah terus melangkah meski sering merasa lelah dan putus asa. Ini adalah pencapaian yang layak untuk dibanggakan. Terima kasih karena telah menjadi versi terbaik diri sendiri sepanjang perjalanan ini. *I wanna thank me for just being me at all times.*

Wassalamualaikum Warahmatullahi Wabarakatuh.

Sukabumi, 5 Juli 2025

Penulis

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI SKRIPSI UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik UNIVERSITAS NUSA PUTRA, saya yang bertanda tangan di bawah ini:

Nama : Ai Dina Agustin

NIM 20210040065

Program Studi : Teknik Informatika

Jenis karya : Skripsi

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Nusa Putra **Hak Bebas Royalti Non Eksklusif (*Non-exclusive Royalty- Free Right*)** atas karya ilmiah saya yang berjudul :

"STRATEGI *TEST-DRIVEN DEVELOPMENT* DALAM ARSITEKTUR *MICROSERVICES* UNTUK OPTIMALISASI PENGEMBANGAN APLIKASI *PAYROLL"*

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Universitas Nusa Putra berhak menyimpan, mengalih media/format- kan, mengelola dalam bentuk pangkalan data (database), merawat, dan memublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : SUKABUMI

Pada tanggal : 15 Juli 2025

Yang menyatakan

(Ai Dina Agustin)

DAFTAR ISI

Nusa Putra UNIVERSITY 2002 AMOR DEUS, PARENTIUM, CONCERVIS

Nusa Putra UNIVERSITY 2002 SUB DEUS, PARENTIUM, CONCILIO

Nusa Putra UNIVERSITY 2008 AMOR DEUS, PARENTIUM, CONCERVIS

DAFTAR TABEL

Nusa Putra
UNIVERSITY
2002
AMOR DEUS, PARENTIUM, CONCERVIS

DAFTAR GAMBAR

Nusa Putra UNIVERSITY 2002 AMOR DEUS, PARENTIUM, CONCERVIS

DAFTAR LAMPIRAN

Nusa Putra
UNIVERSITY
2002
AMOR DEUS, PARENTIUM, CONCERVIS

BAB I
PENDAHULUAN

1.1 Latar Belakang

PT Sineas Kreatif Indonesia merupakan perusahaan berskala besar yang bergerak pada bidang Teknologi yang memiliki jumlah karyawan yang banyak. Dengan jumlah karyawan yang terus bertambah, perusahaaan ini memerlukan sistem pengelolaan sumber daya yang efisian, khususnya dalam aspek penggajian. Pengelolaan penggajian yang efektif sangat penting untuk memastikan bahwa setiap karyawan menerima haknya secara tepat waktu dan tanpa kesalahan. Oleh karena itu, diperlukan sistem *Payroll* yang andal dan dapat mengakomodasi kebutuhan perusahaan secara optimal.

Dalam beberapa tahun terakhir, pengembangan perangkat lunak telah banyak beralih ke arsitektur *microservices*. Arsitektur ini menawarkan fleksibilitas, skalabilitas, dan efisiensi yang lebih baik dibandingkan dengan pendekatan *monolithic*[1]. Oleh karena itu, PT Sineas Kreatif Indonesia memilih untuk menggunakan arsitektur *microservice* sebagai arsitektur dalam pengembangan sistem *Payroll.* Dalam arsitektur *microservices*, sistem dipecah menjadi layanan-layanan kecil yang dapat dikelola secara independen. Hal ini memungkinkan pengembangan dan penyebaran fitur baru dilakukan dengan lebih cepat dan efisien[2]. Namun, keberhasilan arsitektur ini bergantung pada kemampuan setiap layanan untuk berjalan dengan baik, berkomunikasi dengan baik, dan bebas dari kesalahan logika yang dapat mengganggu keseluruhan sistem[3]. Meskipun memiliki banyak keunggulan, implementasi arsitektur *microservices* juga memiliki kendala tersendiri. Beberapa masalah yang sering muncul adalah kegagalan komunikasi antar-layanan, kesulitan dalam mengelola dependensi, serta kendala dalam pengujian integrasi. Saat ini, banyak *developer* masih mengandalkan pengujian manual untuk memastikan sistem berjalan dengan baik. Namun, pendekatan ini sering kali memakan waktu lama dan kurang efektif dalam mendeteksi kesalahan tersembunyi, sehingga keandalan sistem menjadi terganggu dan pengalaman pengguna pun dapat menurun[4].

Untuk mengatasi hal tersebut, salah satu solusi yang ditawarkan adalah menggunakan pendekatan *Test-Driven Development* (TDD). TDD merupakan pendekatan pengembangan perangkat lunak yang mengutamakan penulisan pengujian sebelum kode utama dikembangkan. Alasan utama mengapa pengujian ditulis terlebih dahulu adalah untuk memastikan bahwa setiap fungsi dikembangkan sesuai dengan kebutuhan dan berjalan sebagaimana mestinya sejak awal. Dengan menuliskan pengujian terlebih dahulu, *developer* dipaksa untuk berpikir dari sudut pandang hasil akhir dan bagaimana sistem seharusnya berperilaku, sehingga dapat mendeteksi kesalahan lebih awal dan mencegah *bug* tersembunyi. Pendekatan ini juga membuat proses pengembangan menjadi lebih terstruktur dan terarah[5]. Selain itu, TDD juga memberikan manfaat lain seperti peningkatan dokumentasi kode melalui *unit testing*. TDD mengikuti siklus pendek yang disebut *Red-Green-Refactor*. Pada tahap pertama (*red*), *developer* menulis pengujian pertama kali yang ditujukan untuk gagal karena kode implementasi belum ada. Selanjutnya, *developer* membuat kode untuk memastikan pengujian tersebut berhasil (*green*). Setelah itu, desain kode dioptimalkan tanpa mengubah perilaku fungsionalitasnya (*refactor*) untuk menjaga agar sistem tetap mudah diperbaiki dan dikembangkan. Siklus ini memberikan umpan balik langsung yang penting dalam konteks *microservices* yang kompleks[6].

Dalam konteks arsitektur *microservices*, TDD menjadi pendekatan penting untuk mengurangi kompleksitas sistem dengan memastikan bahwa setiap layanan dapat berfungsi dengan baik baik secara individual maupun sebagai bagian dari keseluruhan sistem[5]. Dalam penerapan TDD pada arsitektur *microservices*, pemilihan *framework unit testing* yang tepat menjadi aspek krusial. *Framework* yang sesuai harus mampu menangani pengujian komunikasi antar-layanan serta integrasi sistem secara efisien. Beberapa kriteria utama dalam menentukan kesesuaian *framework* mencakup efisiensi dalam mendeteksi dan menangani kesalahan, cakupan pengujian terhadap berbagai skenario komunikasi antar-layanan, kemudahan implementasi dalam lingkungan pengembangan, serta kompatibilitas dengan arsitektur *microservices* yang digunakan.

Beberapa *framework unit testing* yang umum digunakan dalam pengembangan perangkat lunak berbasis JavaScript adalah Jest, Mocha, dan Chai. Jest adalah

framework testing lengkap yang dikembangkan oleh Facebook, dikenal dengan kemudahan konfigurasi, kecepatan tinggi, dan fitur bawaan seperti *mocking* dan *code coverage*. Mocha adalah test runner yang fleksibel dan sering digunakan bersama Chai, yaitu assertion library yang memungkinkan penulisan pengujian yang lebih ekspresif dan terbaca. Ketiga *framework* ini mendukung pendekatan *Test-Driven Development* dan sangat sesuai digunakan dalam lingkungan pengembangan *microservices* karena dapat membantu dalam menguji fungsi, integrasi, serta komunikasi antar-layanan secara efisien.

Penerapan metode *Test-Driven Development* (TDD) telah dibuktikan efektif dalam beberapa penelitian sebelumnya. Misalnya, Jonathan dan Pakereng (2021) menerapkan TDD untuk mengembangkan aplikasi pemantauan penyebaran COVID-19. Mereka menemukan bahwa TDD mempermudah *developer* dalam mendeteksi kesalahan saat implementasi kode. Selain itu, kode yang dihasilkan lebih rapi karena melewati proses *refactoring*[7].

Goldan et al. (2022) juga menerapkan TDD dalam pengembangan sistem informasi manajemen aset. Hasilnya menunjukkan bahwa sistem yang dikembangkan lebih fleksibel terhadap perubahan, dengan *defect* dan kesalahan yang lebih mudah ditemukan sejak awal[8]. Hal ini memungkinkan penghematan waktu dan biaya dalam jangka panjang. Temuan ini sejalan dengan penelitian Staegemann et al. (2022) yang mengungkapkan bahwa TDD efektif dalam menurunkan kepadatan cacat (*defect density*), yang berpengaruh pada efisiensi proses pengembangan sistem[9].

Penelitian ini bertujuan untuk mengevaluasi penerapan TDD dalam arsitektur *microservices*, khususnya dalam mengatasi masalah pengujian komunikasi antar-layanan. Penelitian ini juga akan mengidentifikasi bagaimana berbagai *framework unit testing* seperti Jest, Mocha, dan Chai dapat mendukung implementasi TDD secara efektif[10].

Berdasarkan latar belakang tersebut, penelitian ini akan mengevaluasi penerapan TDD dalam arsitektur *microservices*, dengan fokus pada pengujian komunikasi antar-layanan dan integrasi. Implementasi TDD dalam penelitian ini akan diterapkan pada pengembangan aplikasi *Payroll* di PT. Sineas Kreatif Indonesia. Judul penelitian ini adalah: **"Strategi *Test-Driven Development* dalam**

Arsitektur *Microservices* untuk Optimalisasi Pengembangan Aplikasi *Payroll*".

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan, rumusan masalah dalam penelitian ini adalah:

1. Bagaimana penerapan *Test-Driven Development* (TDD) dapat mendukung pengujian layanan dalam arsitektur *microservices* untuk optimalisasi pengembangan aplikasi *Payroll* di PT Sineas Kreatif Indonesia?
2. Bagaimana efektivitas *framework unit testing* seperti Jest, Mocha dan Chai dalam mendukung implementasi TDD pada pengembangan aplikasi *Payroll* berbasis *microservices*?

1.3 Batasan Masalah

Agar penelitian ini lebih terarah dan fokus, beberapa batasan masalah yang diterapkan adalah sebagai berikut:

1. Penelitian ini hanya berfokus pada penerapan *Test-Driven Development* (TDD) dalam konteks arsitektur *microservices* yang digunakan pada pengembangan aplikasi *Payroll* di PT Sineas Kreatif Indonesia. Penelitian tidak mencakup pendekatan *monolithic* atau pengujian pada arsitektur lain.
2. Evaluasi pengujian dilakukan pada *framework* pengujian unit *testing* tertentu, yaitu Jest, Mocha, dan Chai, yang dipilih berdasarkan popularitas, dukungan terhadap TDD, dan relevansinya dalam konteks pengujian layanan *backend* berbasis *microservices*.
3. Analisis pengujian difokuskan pada komunikasi antar-layanan (*inter-service communication*) dan integrasi antar-layanan dalam arsitektur *microservices*.
4. Studi kasus dilakukan pada pengembangan aplikasi *Payroll* dengan model *microservices* sederhana untuk menjaga konsistensi, fokus evaluasi, dan keterbatasan waktu penelitian.

5. Implementasi hanya dilakukan pada layanan *backend*, tanpa melibatkan komponen *frontend* atau integrasi dengan layanan eksternal di luar sistem aplikasi *Payroll*.

1.4 Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah sebagai berikut:

1. Menganalisis penerapan *Test-Driven Development* (TDD) dalam mendukung pengujian layanan berbasis arsitektur *microservices* pada pengembangan aplikasi *Payroll*.
2. Mengevaluasi efektivitas *framework* pengujian unit *testing*, yaitu Jest, Mocha, dan Chai, dalam implementasi TDD untuk *microservices* pada aplikasi *Payroll*.

1.5 Manfaat Penelitian

Manfaat dari penelitian ini, peneliti berharap memberikan manfaat kepada pihak yang terkait, sebagai berikut:

a. Bagi Perguruan Tinggi
   1) Menambah referensi penelitian khususnya di bidang pengembangan perangkat lunak modern, dengan fokus pada penerapan *Test-Driven Development (*TDD) dalam arsitektur *microservices*.
   2) Mendorong mahasiswa dan peneliti lain untuk mengembangkan penelitian serupa yang relevan dengan perkembangan teknologi industri.

b. Bagi Penulis
   1) Memperluas wawasan dan keterampilan dalam penerapan TDD serta pengujian perangkat lunak berbasis *microservices*.
   2) Meningkatkan kemampuan analisis, pemecahan masalah, dan pengambilan keputusan dalam konteks pengujian perangkat lunak.
   3) Memberikan pengalaman berharga dalam menyusun penelitian berbasis pengembangan perangkat lunak.

c. Bagi *Developer* Perangkat Lunak

1) Memberikan panduan praktis untuk mengimplementasikan TDD secara efektif pada arsitektur *microservices*.
2) Membantu *developer* memilih *framework* pengujian *unit testing* yang paling sesuai dengan kebutuhan pengujian.
3) Mendukung peningkatan kualitas perangkat lunak melalui pendekatan pengujian yang lebih terstruktur dan efisien.

d. Bagi Perusahaan

1) Mendukung pengembangan aplikasi *Payroll* dengan pendekatan TDD yang dapat meningkatkan kualitas, keandalan, dan efisiensi pengujian sistem berbasis *microservices*.
2) Memberikan rekomendasi *framework* pengujian *unit testing* yang sesuai dengan kebutuhan perusahaan untuk meningkatkan efisiensi dalam pengujian perangkat lunak.
3) Menyediakan solusi untuk mengatasi masalah dalam pengujian komunikasi antar-layanan dan integrasi, sehingga membantu perusahaan menghasilkan aplikasi yang lebih stabil dan siap untuk digunakan di lingkungan produksi.

1.6 Sistematika Penulisan

Penelitian ini disusun dengan sistematika sebagai berikut:

BAB I PENDAHULUAN

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

BAB II TINJAUAN PUSTAKA

Bab ini membahas teori-teori yang mendasari penelitian, meliputi *konsep Test-Driven Development (TDD)*, arsitektur *microservices*, tantangan pengujian dalam *microservices*, dan *framework* pengujian *unit testing* seperti Jest, Mocha, dan Chai. Selain itu, bab ini juga mengulas penelitian terdahulu yang relevan untuk memberikan landasan dan menunjukkan posisi penelitian ini dalam konteks studi yang lebih luas.

BAB III METODE PENELITIAN

Bab ini menjelaskan metode penelitian yang digunakan, meliputi pendekatan penelitian, objek penelitian, parameter evaluasi *framework* pengujian, serta alat dan bahan yang digunakan. Bab ini juga memaparkan langkah-langkah implementasi penelitian, termasuk proses pengujian dan penerapan TDD.

BAB IV IMPLEMENTASI DAN PENGUJIAN SISTEM

Bab ini menyajikan hasil penelitian yang diperoleh dari implementasi TDD pada pengembangan aplikasi *Payroll* berbasis *microservice*s. Pembahasan dilakukan dengan menganalisis efektivitas *framework* pengujian yang digunakan serta evaluasi komunikasi antar-layanan dan integrasi sistem.

BAB V PENUTUP

Bab ini berisi kesimpulan dari hasil penelitian yang telah dilakukan serta saran yang dapat diberikan untuk penelitian lebih lanjut atau pengembangan sistem di masa mendatang.

DAFTAR PUSTAKA

Berisi referensi yang digunakan dalam penelitian, seperti buku, jurnal, artikel ilmiah, dan sumber lainnya yang relevan.

LAMPIRAN

Berisi dokumen pendukung penelitian, seperti hasil pengujian, kode program, diagram sistem, serta data lain yang relevan untuk mendukung pemahaman terhadap penelitian ini.

RIWAYAT HIDUP

Bagian ini berisi informasi pribadi penulis dan riwayat pendidikan, serta pengalaman yang relevan (organisasi, atau pelatihan).

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan hasil penelitian dan pengujian yang telah dilakukan terhadap penerapan *Test-Driven Development* (TDD) dalam arsitektur *microservices* pada pengembangan aplikasi *Payroll* di PT Sineas Kreatif Indonesia, maka diperoleh beberapa kesimpulan sebagai berikut:

1. Penerapan TDD terbukti efektif dalam mendukung pengujian layanan berbasis arsitektur *microservice*s. Dengan mengikuti siklus *Red-Green-Refactor*, pengujian dilakukan sebelum implementasi kode, sehingga potensi kesalahan dapat dideteksi lebih awal. Proses ini meningkatkan keandalan antar-layanan serta mengurangi kompleksitas debugging pada sistem *Payroll* yang bersifat modular.
2. *Framework* unit *test*ing seperti Jest, Mocha, dan Chai berperan penting dalam mendukung implementasi TDD. Hasil observasi menunjukkan bahwa Jest unggul dalam kecepatan eksekusi dan integrasi dengan ekosistem Node.js, sementara Mocha dan Chai lebih fleksibel untuk pengujian yang kompleks dan assertion detail. Kombinasi ketiganya memberikan hasil pengujian yang optimal untuk sistem berbasis *microservice*s.
3. Pendekatan TDD terbukti efektif dalam meningkatkan kualitas sistem. Hal ini dibuktikan melalui hasil pengujian pada metrik code coverage yang tinggi (rata-rata di atas 96%) serta defect rate yang rendah (4.33/1000 LOC) dengan klasifikasi kualitas "Good". Artinya, TDD tidak hanya meningkatkan cakupan pengujian secara signifikan, tetapi juga mampu menurunkan jumlah cacat pada kode secara konsisten, yang menunjukkan keefektifannya dalam menghasilkan kode yang andal dan *maintainable*.
4. Arsitektur *microservices* yang diterapkan berhasil menunjukkan isolasi dan toleransi kegagalan antar layanan. Pengujian menunjukkan bahwa ketika salah satu layanan dimatikan (*auth*, *employee*, atau *payroll*), layanan lainnya tetap dapat berjalan secara independen, dan sistem tetap memberikan

respons error yang informatif. Ini menunjukkan keberhasilan penerapan prinsip fault tolerance dalam arsitektur *microservice*s.

5. Perbandingan pendekatan TDD vs non-TDD menunjukkan hasil yang sangat kontras. Dengan TDD, terjadi peningkatan *code coverage* hingga 17–32% dan penurunan *defect rate* sebesar 18.66 poin, dari "*Very Poor*" menjadi "Good". Ini membuktikan bahwa pendekatan TDD tidak hanya memperbaiki kualitas sistem, tetapi juga efisien dalam jangka panjang.
6. Pertimbangan penggunaan arsitektur microservices perlu disesuaikan dengan kebutuhan dan skala organisasi. Meskipun microservices menawarkan keunggulan seperti skalabilitas, modularitas, dan fault tolerance, pendekatan ini juga memerlukan sumber daya yang lebih besar, seperti infrastruktur multi-server, pengelolaan antar layanan yang kompleks, serta biaya operasional yang lebih tinggi. Oleh karena itu, untuk perusahaan kecil hingga menengah, arsitektur monolitik atau modular yang sederhana dapat menjadi pilihan yang lebih efisien dan tepat guna.
7. Microservices bukanlah kebutuhan mutlak dalam sistem payroll, terutama untuk perusahaan dengan skala penggajian yang belum kompleks. Arsitektur ini lebih cocok diterapkan pada perusahaan berskala besar yang memiliki kebutuhan integrasi sistem lintas departemen, volume transaksi tinggi, dan tim pengembangan yang terdistribusi. Untuk perusahaan kecil, penggunaan microservices dapat membebani dari segi teknis maupun finansial.

5.2 Saran

Berdasarkan kesimpulan di atas, berikut adalah beberapa saran yang dapat disampaikan untuk pengembangan selanjutnya:

1. Perluasan implementasi TDD pada seluruh siklus pengembangan aplikasi. Saat ini penerapan TDD masih terbatas pada layanan *backend*. Untuk meningkatkan kualitas sistem secara menyeluruh, disarankan agar prinsip TDD juga diterapkan pada layanan *frontend*, serta pengujian integrasi antar sistem eksternal.

2. Integrasi dengan pipeline CI/CD. *Framework* pengujian seperti Jest dan Mocha sebaiknya diintegrasikan dengan Continuous Integration untuk memastikan bahwa setiap perubahan kode tetap melalui proses *unit testing* secara konsisten dan cepat.
3. Pemilihan *framework* pengujian yang sesuai dengan kompleksitas proyek. Meskipun Jest menunjukkan kinerja tinggi, *developer* tetap perlu mempertimbangkan kompleksitas skenario pengujian dan mempertimbangkan kombinasi *framework* (seperti Chai untuk assertion yang lebih kaya) demi hasil yang optimal.
4. Evaluasi berkelanjutan terhadap *defect rate*. Meskipun sudah berada dalam kategori "*Good*", *defect rate* perlu terus dievaluasi secara berkala untuk memastikan stabilitas sistem seiring pertumbuhan kompleksitas layanan *microservices* yang dikembangkan.
5. Pelatihan dan pembiasaan TDD untuk tim *developer*. Untuk menjamin keberhasilan implementasi TDD jangka panjang, perlu ada pelatihan dan pembiasaan siklus *Red-Green-Refactor* dalam setiap proses pengembangan, sehingga tercipta budaya *clean code* dan pengujian menyeluruh sejak tahap awal.
6. Evaluasi kelayakan penggunaan arsitektur microservices pada fase awal pengembangan. Jika perusahaan belum memiliki kebutuhan skalabilitas tinggi atau belum siap dari segi infrastruktur, maka pendekatan arsitektur monolitik masih sangat relevan dan efisien. Penerapan microservices sebaiknya dilakukan secara bertahap dan disesuaikan dengan kesiapan teknis dan finansial perusahaan.

Nusa Putra

DAFTAR PUSTAKA

.

[1] J. Fritzsch, J. Bogner, A. Zimmermann, and S. Wagner, "From monolith to *microservice*s: A classification of refactoring approaches," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11350 LNCS, pp. 128–141, 2019, doi: 10.1007/978-3-030-06019-0_10.

[2] Riyanto, Irman Hermadi, and Yani Nurhadryani, "Analisis Uji Performa Aplikasi Dari Hasil Implementasi *Refactor*ing Arsitektur Monolitik Ke Mikroservis dengan Decomposition dan Strangler Pattern," *J. Sist. Cerdas*, vol. 6, no. 3, pp. 189–203, 2023, doi: 10.37396/jsc.v6i3.352.

[3] M. Waseem, P. Liang, M. Shahin, A. Di Salle, and G. Márquez, "Design, monitoring, and *test*ing of *microservice*s systems: The practitioners' perspective," *J. Syst. Softw.*, vol. 182, 2021, doi: 10.1016/j.jss.2021.111061.

[4] S. Vimaleshwaran and S. Lanka, "Evaluating the Impact of Test-Driven Development (TDD) on Software," *Int. J. Sci. Eng. Appl.*, vol. 12, no. 05, pp. 87–92, 2023, doi: 10.7753/ijsea1205.1026.

[5] K. Beck, *Test-Driven Development : By Example*. Boston: *Add*ison Wesley, 2002.

[6] J. Langr, *Modern C++Programming with Test-Driven Development*. Dallas, Texas: The Pragmatic Programmers, LLC, 2013.

[7] F. Jonathan and M. A. I. Pakereng, "Test-Driven Development pada Pengembangan Aplikasi Android untuk Memantau COVID-19," *IJCIT (Indonesian J. Comput. Inf. Technol.*, vol. 6, no. 1, pp. 20–24, 2021, doi: 10.31294/ijcit.v6i1.9502.

[8] R. Goldan, Y. Elzatar, A. H. Brata, and A. P. Kharisma, "Pengembangan Sistem Informasi Inventarisasi Aset menggunakan Metode Test Driven Development (Studi Kasus: Universitas Mulia)," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 6, no. 1, pp. 19–29, 2022, [Online]. Available: http://j-ptiik.ub.ac.id

[9] D. Staegemann *et al.*, "A Literature Review on the Challenges of Applying Test-Driven Development in Software Engineering," *Complex Syst.*

Informatics Model. Q., vol. 2022, no. 31, pp. 18–28, 2022, doi: 10.7250/csimq.2022-31.02.

[10] M. A. A. Moseh, N. Ameen Al-Khulaidi, A. H. Gumaei, A. Alsabry, and A. A. A. Musleh, "Classification and Evaluation *Framework* of Automated *test*ing tools for agile software: Technical Review," *4th Int. Conf. Emerg. Smart Technol. Appl. eSmarTA 2024*, no. July, 2024, doi: 10.1109/eSmarTA62850.2024.10638902.

[11] N. S. Elgheriani, N. Ali, and S. Ahmed, "Micro*service*s VS. Monolithic Architechture [The Differential Structure Between Two Architechture] Ministry of Technical and Vocation Education, Libya," 2022, [Online]. Available: http://dx.doi.org/10.47832/2717-8234.12.47

[12] W. Ren and S. Barrett, "Test-driven development, engagement in activity, and maintainability: An empirical study," *IET Softw.*, vol. 17, no. 4, pp. 509–525, 2023, doi: 10.1049/sfw2.12135.

[13] D. Agha, R. Sohail, A. F. Meghji, R. Qaboolio, and S. Bhatti, "Test Driven Development and Its Impact on Program Design and Software Quality: A Systematic Literature Review," *VAWKUM Trans. Comput. Sci.*, vol. 11, no. 1, pp. 268–280, 2023, doi: 10.21015/vtcs.v11i1.1494.

[14] S. M. Tampubolon and T. Raharjo, "Unveiling the Benefits and Challenges of Test-Driven Development in Agile: A Systematic Literature Review," *Indones. J. Comput. Sci.*, vol. 13, no. 2, pp. 2189–2204, 2024, doi: 10.33022/ijcs.v13i2.3857.

[15] M. M. Moe, "Unit Test using Test-Driven Development Approach to Sup*port* Reusability," *Int. J. Trend Sci. Res. Dev.*, vol. Volume-3, no. Issue-3, pp. 194–196, 2019, doi: 10.31142/ijtsrd21731.

[16] M. Richards, *Software Architecture Pattern: Second Edition*, 2nd ed. Gravenstein Highway North: O'Reilly Media, 2022.

[17] S. Newman, *Building Microservices: Designing Fine-Grained Systems*. Gravenstein Highway North: O'Reilly Media, 2015.

[18] C. Richardson, *Microservices Patterns: With examples in Java*. Shelter Island: Manning Publications Co., 2019.

[19] P. Sistem, I. Akademik, and T. Informatika, "Implementasi unit *test*ing dan

end-to-end *test*ing pada sistem informasi akademik teknik informatika," vol. 9, no. 4, pp. 2208–2219, 2024.

[20] A. Setiawan, K. I. Satoto, and R. R. Isnanto, "Implementasi Automated Unit Dan Integration Testing Pada Pengujian Perangkat Lunak (Studi Kasus Aplikasi Penjualan Buku Online)," *Transient*, vol. 2, no. 3, pp. 708–713, 2013, [Online]. Available: https://ejournal3.undip.ac.id/index.php/transient/article/view/3907

[21] S. Kumar and S. Bansal, "Comparative Study of Test Driven Development with Traditional Techniques," *Int. J. Soft Comput. Eng.*, vol. 3, no. 1, pp. 352–360, 2013.

[22] M. Gartner, *ATDD by Example: A Practical Guide to Acceptance Test-Driven Development*. Westfrod: Pearson Education, Inc, 2012.

[23] Jest, "Getting Started." Accessed: Jan. 24, 2025. [Online]. Available: https://jestjs.io/docs/getting-started

[24] A. Fu'adi and A. Prianggono, "Analisa dan Perancangan Sistem Informasi Akademik Akademi Komunitas Negeri Pacitan Menggunakan Diagram UML dan EER," *J. Ilm. Teknol. Inf. Asia*, vol. 16, no. 1, p. 45, 2022, doi: 10.32815/jitika.v16i1.650.

[25] K. 'Afiifah, Z. F. Azzahra, and A. D. Anggoro, "Analisis Teknik Entity-Relationship Diagram dalam Perancangan Database Sebuah Literature Review," *Intech*, vol. 3, no. 2, pp. 18–22, 2022, doi: 10.54895/intech.v3i2.1682.

[26] U. Rusmawan, *Teknik Penulisan Tugas Akhir dan Skripsi Pemrograman*. 2024. doi: 987-602-04-9680-1.

[27] R. Andarsyah, C. Yuda Pratama, and H. D. Kishendrian, "Implementasi Code Coverage Pada Chatbot Telegram Sebagai Media Alternatif Sistem Informasi," *J. Tek. Inform.*, vol. 14, no. 2, p. 9568, 2022.

[28] W. Creswell and J. D. Cresswell, *RESEARCH DESAIN QUALITATIVE, QUANTITATIVE, AND MIXED METHODS APPROACHES*. 2022. [Online]. Available: https://books.google.co.id/books?id=Rkh4EAAAQBAJ&printsec=frontcover&hl=id#v=onepage&q&f=false

[29] Ardiansyah, R. Alam, and Z. Arifin, “Pengaruh Kepuasan, Kepercayaan dan Harga terhadap Loyalitas Penumpang Maskapai Garuda Indonesia Di Bandar Udara Internasional Sultan Hasanuddin Makassar,” *SEIKO J. Manag. Bus.*, vol. 7, no. 1, pp. 619–632, 2024.

[30] Deni Murdiani and Muhamad Sobirin, “Perbandingan Metodologi Waterfall Dan Rad (Rapid Application Development) Dalam Pengembangan Sistem Informasi,” *J. Inform. Teknol. dan Sains*, vol. 4, no. 4, pp. 302–306, 2022, doi: 10.51401/jinteks.v4i4.2008.

[31] L. Setiyani, “Desain Sistem : Use Case Diagram Pendahuluan,” *Pros. Semin. Nas. Inov. Adopsi Teknol. 2021*, no. September, pp. 246–260, 2021, [Online]. Available: https://journal.uii.ac.id/AUTOMATA/article/view/19517

Nusa Putra UNIVERSITY 2002 AMOR DEUS, PARENTIUM, CONCERVIS