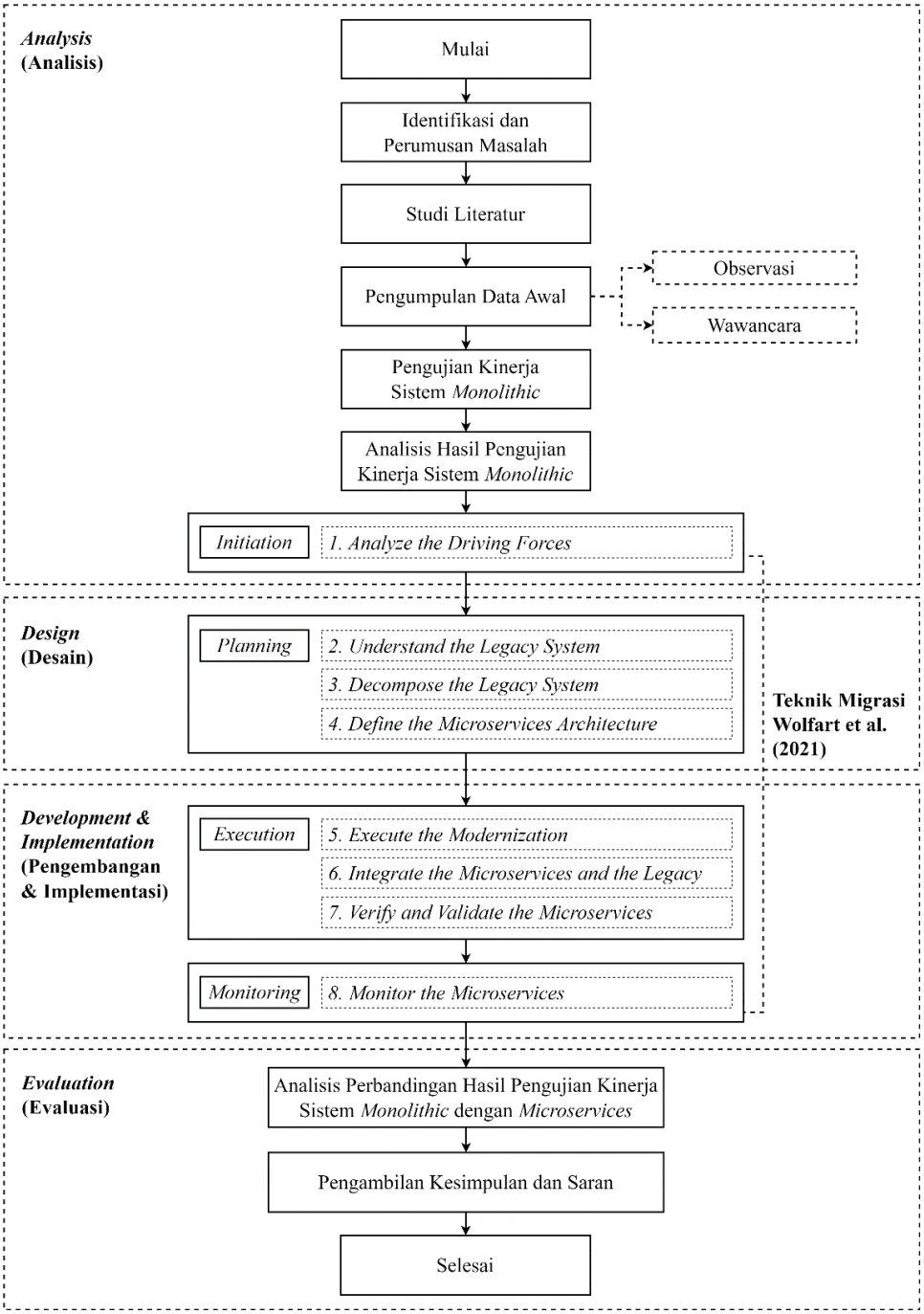
BAB III
METODOLOGI PENELITIAN

3. 1 Desain Penelitian

Desain penelitian yang merupakan representasi langkah-langkah yang dilakukan untuk melakukan penelitian ini ditunjukkan dalam Gambar 3. 1 berikut.



Gambar 3. 1 Desain Penelitian

Pada penelitian ini, digunakan metode penelitian *ADDIE* sebagai dasar tahap penelitian. *ADDIE* merupakan suatu metode pengembangan produk yang didasarkan pada konsep pengembangan sistematis, yang menjadi alat efektif dalam desain instruksional dan pengembangan perangkat lunak sistem informasi. Metode ini terdiri dari 5 fase penelitian, yaitu *Analysis* (Analisis), *Design* (Desain), *Development* (Pengembangan), *Implementation* (Implementasi), dan *Evaluation* (Evaluasi) (Reiser & Dempsey, 2018). Berdasarkan Gambar 3. 1, terdapat 15 tahap yang dilakukan dalam penelitian ini. Berikut uraian dari ke-15 tahap tersebut.

1. Identifikasi dan Perumusan Masalah

Pada tahap ini, dilakukan peninjauan lebih dalam mengenai masalah atau fenomena yang diteliti. Pada tahap ini dilakukan pengamatan terhadap permasalahan yang ada. Hal ini diperlukan agar peneliti dapat menentukan fokus penelitian serta membantu dalam merumuskan masalah dan tujuan penelitian pada tahap selanjutnya.

Pada tahap ini juga dilakukan perumusan masalah yang merupakan langkah awal dalam penelitian dan bertujuan untuk menentukan masalah atau permasalahan yang dijawab dalam penelitian. Pada tahap ini dilakukan beberapa langkah, seperti identifikasi topik penelitian, tinjauan pustaka, dan perumusan pertanyaan penelitian untuk memfokuskan penelitian dan menentukan teknik dan metode yang tepat untuk menjawab pertanyaan penelitian.

2. Studi Literatur

Tahap studi literatur merupakan tahap yang bertujuan untuk memperoleh pemahaman yang mendalam mengenai topik penelitian yang dilakukan. Pada tahap ini dilakukan pencarian dan kajian terhadap sumber-sumber literatur mengenai Tracer Study, arsitektur *monolithic,* arsitektur *microservices, gRPC API*, *API Gateway*, metode dekomposisi *Domain-Driven Design*, teknik dekomposisi, dan pengujian kinerja aplikasi. Kajian dan sumber literatur yang digunakan seperti buku, jurnal, prosiding, artikel, situs internet resmi dan sumber informasi lainnya. Tahap ini bertujuan untuk memperoleh gambaran umum tentang topik penelitian, menentukan kerangka konseptual, memperoleh informasi mengenai teori topik yang diangkat, pencarian referensi metode yang relevan, serta mengidentifikasi masalah yang ada pada topik penelitian.

3. Pengumpulan Data Awal

Tahap ini bertujuan untuk melakukan pengumpulan data awal menggunakan instrumen (alat) pengumpulan data yang telah dibuat berdasarkan jenis data yang dikumpulkan dan metode pengumpulan data yang dipilih. Berikut beberapa jenis pengumpulan data yang digunakan dalam penelitian ini.

1) **Observasi**: Peneliti melakukan observasi terhadap aplikasi Tracer Study UPI untuk mengumpulkan data terkait aplikasi. Observasi yang dilakukan mencakup bagaimana aplikasi bekerja, proses bisnis dalam aplikasi, basis data dan jenis data yang digunakan, fitur dalam aplikasi, dan pengujian kinerja aplikasi. Pengumpulan data dengan cara observasi dilakukan dengan bantuan instrumen observasi pada *Lampiran 2. Pedoman Observasi* dan *Lampiran 3. Lembar Hasil Observasi.*
2) **Wawancara**: Peneliti mewawancarai staf administrasi dan pengembang aplikasi Tracer Study UPI untuk mengetahui informasi yang mendukung penelitian, motivasi pengembangan, kebutuhan sistem, kendala yang dialami, serta harapan terhadap aplikasi baik selama penggunaan maupun saat pengembangan Tracer Study.

Tahap pengumpulan data ini menghasilkan data-data dan informasi yang dibutuhkan pada tahap-tahap penelitian berikutnya. Informasi dan data yang diperoleh berupa informasi umum aplikasi di Universitas Pendidikan Indonesia, informasi terkait proses bisnis aplikasi Tracer Study UPI, motivasi penerapan arsitektur *microservices,* spesifikasi aplikasi dan *server,* teknologi yang digunakan dalam pengembangan, dan lain-lain. Data dan informasi tersebut sangat penting untuk memetakan kondisi awal sebelum melakukan perubahan pada sistem.

4. Pengujian Kinerja Sistem *Monolithic*

Pada tahap ini dilakukan pengujian kinerja *(performance testing)* terhadap aplikasi *monolithic* Tracer Study UPI untuk mengukur kinerja aplikasi sebelum penerapan arsitektur *microservices.* Ini dimaksudkan untuk mengukur kinerja aplikasi yang sudah ada agar dapat dianalisis untuk membantu proses perancangan solusi. Pengujian dilakukan dengan menggunakan alat JMeter dan beberapa skenario pengujian. Metrik yang dinilai berupa *response time* dan *throughput*

aplikasi. Hasil pengujian ini digunakan pada tahap penelitian berikutnya untuk dianalisis.

5. Analisis Hasil Pengujian Kinerja Sistem *Monolithic*

Pada tahap ini dilakukan evaluasi terhadap hasil pengujian kinerja aplikasi Tracer Study saat ini saat masih menggunakan arsitektur *monolithic* dan belum menerapkan arsitektur *microservices.* Hasil evaluasi ini digunakan sebagai acuan dalam perancangan arsitektur *microservices* agar pengembangan dapat berfokus pada bagian kinerja yang perlu ditingkatkan. Data-data tersebut sangat penting dalam merancang *microservices* yang tepat, sehingga hasil yang diinginkan dapat dicapai dengan lebih efisien. Selain itu, hasil analisis ini juga dapat membantu dalam menentukan prioritas implementasi dan alokasi sumber daya yang diperlukan dalam proses perancangan dan pengembangan *microservices.*

6. *Initiation: Analyze the Driving Forces*

Pada tahap ini dianalisis hasil identifikasi *driving forces* atau motivasi dan tujuan apa saja yang ingin dicapai dengan dilakukannya dekomposisi atau migrasi sistem *monolithic* ke *microservices.* Dari daftar tersebut ditentukan prioritas motivasi/tujuan/manfaat migrasi dengan pola *trade-off.* Kegiatan memprioritaskan *driving forces* tersebut dilakukan pada tahap pengumpulan data. Pada tahap ini, hasilnya dijabarkan dan dianalisis untuk membantu pembuatan keputusan apakah proses migrasi perlu dilanjutkan.

7. *Planning: Understand the Legacy System*

Pada tahap ini dilakukan analisis *legacy system* atau sistem *monolithic* terdahulu (sistem yang sudah ada) untuk mengetahui proses bisnis, kebutuhan sistem, teknologi yang digunakan, fitur-fitur yang ada, serta interaksi dan keterkaitan antarfitur. Tahap ini melakukan analisis pada hasil pengumpulan data menggunakan metode observasi dan wawancara.

Hasil observasi dan wawancara yang dianalisis menghasilkan gambaran sistem yang divisualisasikan dalam bentuk diagram, seperti diagram *UML (Unified Modelling Language),* diagram *ERD (Entity Relationship Diagram),* dan diagram arsitektur sistem. Visualisasi sistem tersebut digunakan dalam tahap penelitian berikutnya. Tahap ini sangat penting untuk memastikan bahwa semua aspek sistem telah dipahami dengan baik sebelum melangkah ke tahap desain dan implementasi

solusi yang diusulkan. Dari tahap ini juga dihasilkan daftar fitur pada *legacy system,* spesifikasi sistem, dan teknologi pengembangan sistem.

8. *Planning: Decompose the Legacy System*

Pada tahap ini dilakukan desain dekomposisi sistem *monolithic* menjadi *microservices* dengan menggunakan pendekatan *Domain-Driven Design (DDD).* Desain *DDD* pada tahap ini dibuat berdasarkan analisis diagram *UML* dan *ERD* pada tahap sebelumnya. Tahap ini menghasilkan daftar fitur dan layanan yang menjadi kandidat *microservices.*

9. *Planning: Define the Microservices Architecture*

Pada tahap ini dilakukan perancangan sistem *microservices,* arsitektur sistem, *database,* spesifikasi *API*, dan penentuan protokol komunikasi antar-*service.* Pada tahap ini juga dilakukan penentuan teknologi yang digunakan dalam pengembangan *microservices,* seperti bahasa pemrograman, *database,* serta spesifikasi *server.* Perancangan dan penentuan ini dilakukan berdasarkan kebutuhan sistem, *driving forces,* dan studi literatur pada tahap-tahap penelitian sebelumnya.

10. *Execution: Execute the Modernization*

Tahap ini mengimplementasikan hasil perencanaan dan desain *microservices* pada tahap sebelumnya. Pada tahap ini dilakukan rancang bangun sistem *microservices* dengan mengikuti rancangan *Domain-Driven Design*, yaitu dalam setiap *service* melingkupi *domain* bisnis tunggal. Aktivitas yang dilakukan pada tahap ini mencakup pengkodean *(coding),* dokumentasi kode, pengujian integrasi *(integration test),* dan kontainerisasi aplikasi.

11. *Execution: Integrate the Microservices and the Legacy*

Pada tahap ini sistem *microservices* yang telah dikembangkan kemudian diintegrasikan dengan sistem *monolithic.* Karena pengembangan *microservices* mengikuti pola *Strangler Fig*, maka pengembangan *service* dilakukan secara bertahap satu per satu. Oleh karena itu, sistem *monolithic* lama masih diperlukan agar aplikasi dapat berjalan secara utuh, sampai seluruh *service* dikembangkan dan sepenuhnya berganti menjadi *microservices.* Integrasi dilakukan dengan mengembangkan *API Gateway* dan *Integration Glue Code.*

Pada tahap ini juga dilakukan penerapan atau *deployment* sistem *microservices* ke *server* sesuai dengan desain arsitektur yang telah dibuat pada tahap sebelumnya.

Deployment dilakukan dengan menggunakan prinsip *containerization* dari Docker, yang oleh beberapa penelitian dinilai banyak memberikan manfaat dan cocok diterapkan pada arsitektur *microservices.*

12. *Execution: Verify and Validate the Microservices*

Pada tahap ini, sistem *microservices* yang sudah berjalan diverifikasi dan divalidasi menggunakan beberapa pengujian *black box*, seperti *integration/component testing* dan *performance testing.* Pengujian yang dilakukan bertujuan untuk memastikan sistem *microservices* yang dirancang berjalan dengan baik sesuai dengan spesifikasi, serta mengukur pengaruh dari penerapan arsitektur *microservices* pada sistem Tracer Study UPI. Selain itu, pengujian kinerja atau *performance testing* pada tahap ini juga dilakukan untuk menilai perubahan kinerja yang terjadi setelah proses migrasi.

Jenis pengujian kinerja yang dilakukan adalah *general performance testing, load testing, soak/stability testing,* dan *stress testing.* Pengujian ini dilakukan dengan menggunakan bantuan aplikasi sebagai alat pengujian yaitu Apache JMeter. Metrik atau indikator yang dinilai dan dianalisis pada pengujian ini yaitu *response time* dan *throughput.* Pengujian kinerja yang dilakukan pada tahap ini memiliki beberapa tahapan, yaitu: 1) mengidentifikasi lingkungan pengujian; 2) mengidentifikasi kriteria pengujian kinerja; 3) merencanakan dan merancang pengujian; 4) mengonfigurasi lingkungan pengujian; 5) mengimplementasikan rancangan pengujian; 6) menjalankan pengujian; dan 7) menganalisis dan melaporkan hasil serta mengulangi pengujian.

13. *Monitoring: Monitor the Microservices*

Tahap ini menjadi tahap terakhir pada bagian pengembangan *microservices,* yaitu melakukan pemantauan pada sistem *microservices* yang telah dikembangkan. Tahap ini bertujuan untuk memeriksa kesehatan sistem, ketersediaan, kemacetan atau kegagalan, kinerja, dan penggunaan sumber daya. Pemantauan dilakukan dengan mengumpulkan metrik-metrik tersebut secara berkala dengan menggunakan bantuan alat pengujian kinerja.

14. Analisis Perbandingan Hasil Pengujian Kinerja Sistem *Monolithic* dengan *Microservices*

Pada tahap ini dilakukan analisis perbandingan terhadap hasil pengujian kinerja sistem *monolithic* pada tahap pengumpulan data awal dengan hasil pengujian kinerja sistem *microservices* pada tahap pengembangan sistem. Perbandingan hasil pengujian kinerja dilakukan untuk melihat perbedaan yang terjadi sebelum dan sesudah proses migrasi/dekomposisi dilaksanakan. Dengan membandingkan hasil pengujian dari kedua sistem, dapat diidentifikasi peningkatan kinerja yang diharapkan dari penerapan arsitektur *microservices.*

Hasil analisis perbandingan kedua arsitektur sistem ini menjadi dasar untuk evaluasi akhir penelitian, memberikan pemahaman yang mendalam mengenai efektivitas dari proses dekomposisi/migrasi arsitektur sistem, serta memberikan rekomendasi untuk pengembangan lebih lanjut pada sistem Tracer Study UPI. Tahap ini juga memberikan gambaran mengenai kelebihan, kekurangan, dan tantangan selama proses migrasi dari arsitektur *monolithic* ke *microservices.*

15. Pengambilan Kesimpulan dan Saran

Tahap ini bertujuan untuk merangkum hasil-hasil yang diperoleh selama proses penelitian serta memberikan rekomendasi dan saran terhadap pengembangan Tracer Study UPI untuk perbaikan dan penelitian selanjutnya. Pada tahap ini, hasil analisis dan temuan penelitian dideskripsikan secara menyeluruh untuk menjawab rumusan masalah yang telah didefinisikan pada tahap awal penelitian. Kesimpulan diambil berdasarkan hasil-hasil dari tahapan penelitian sebelumnya. Berdasarkan kesimpulan yang diperoleh, dirumuskan saran dan rekomendasi untuk pengembangan sistem dan penelitian lebih lanjut.

3. 2 Pendekatan Penelitian

Pendekatan penelitian yang digunakan dalam penelitian ini adalah pendekatan kualitatif dan kuantitatif. Pendekatan kualitatif digunakan untuk mengumpulkan data dari sisi kebutuhan pengguna, kebutuhan pengembang, dan sisi kualitas aplikasi. Pendekatan kuantitatif digunakan untuk mengukur kinerja aplikasi setelah dilakukan pengembangan menggunakan arsitektur *microservices* dan metode dekomposisi *Domain-Driven Design*.

3. 3 Jenis Penelitian

Penelitian ini menggunakan jenis penelitian pengembangan (*development research*) dengan pendekatan desain (*design approach*). Penelitian pengembangan bertujuan untuk menghasilkan suatu produk, yaitu aplikasi Tracer Study dengan mengimplementasikan arsitektur *microservices* menggunakan *gRPC API* dan metode dekomposisi *Domain-Driven Design* pada aplikasi Tracer Study Universitas Pendidikan Indonesia.

3. 4 Lokasi dan Waktu Penelitian

Lokasi penelitian dilakukan di Universitas Pendidikan Indonesia. Waktu penelitian dilakukan selama 8 bulan, yaitu pada bulan Desember 2023, Januari, Februari, Maret, April, Mei, Juni, dan Juli 2024.

3. 5 Objek Penelitian

Objek penelitian yang dijadikan studi kasus pada penelitian ini adalah aplikasi Tracer Study Universitas Pendidikan Indonesia. Aplikasi Tracer Study UPI merupakan aplikasi yang digunakan sebagai instrumen pengumpulan data dan *feedback* dari alumni UPI. Data yang dikumpulkan berupa data pekerjaan terbaru, penghasilan, kesesuaian bidang pekerjaan dengan pendidikan, dan lain-lain. *Feedback* yang diperoleh dari alumni dibutuhkan untuk perbaikan serta pengembangan kualitas dan sistem pendidikan tinggi di Universitas Pendidikan Indonesia. *Feedback* ini juga digunakan untuk memetakan dunia usaha dan industri agar jarak diantara kompetensi yang diperoleh saat kuliah dengan tuntutan dunia kerja dapat diminimalkan.

3. 6 Variabel Penelitian

Variabel penelitian dalam penelitian ini terdiri dari variabel bebas (*independent variable*) yaitu implementasi arsitektur *microservices* menggunakan *gRPC API* dan metode dekomposisi *Domain-Driven Design*, dan variabel terikat

(*dependent variable*) yaitu kinerja aplikasi Tracer Study Universitas Pendidikan Indonesia.

3. 7 Jenis dan Sumber Data

Jenis data yang dikumpulkan dalam penelitian ini adalah data primer dan data sekunder. Data primer diperoleh dari hasil observasi sistem dan wawancara dengan pengembang aplikasi dan staf administrasi aplikasi Tracer Study Universitas Pendidikan Indonesia. Data sekunder diperoleh dari dokumen-dokumen yang terkait dengan aplikasi Tracer Study dan teknologi arsitektur *microservices* dan *gRPC API*.

3. 8 Teknik Pengumpulan Data

Pengumpulan data primer dan sekunder pada penelitian ini dilakukan melalui beberapa teknik berikut.

1. Studi Literatur

Studi literatur dilakukan untuk memperoleh informasi mengenai arsitektur *microservices*, *monolithic, gRPC API, API Gateway*, metode dekomposisi *Domain-Driven Design,* teknik dekomposisi, pengujian kinerja, dan aplikasi Tracer Study. Dalam penelitian ini, dilakukan pencarian data dan informasi dari berbagai sumber, termasuk buku referensi, hasil penelitian sejenis yang melakukan migrasi sistem *monolithic* ke *microservices*, jurnal-jurnal hasil penelitian, prosiding, serta situs internet yang relevan. Hal ini dilakukan untuk dapat mengumpulkan teori, pengalaman, dan data yang diperlukan untuk menjawab permasalahan yang diteliti secara efektif dan efisien. Data ini digunakan sebagai dasar dalam pengembangan aplikasi Tracer Study menggunakan arsitektur *microservices* dan *gRPC API* serta metode dekomposisi *Domain-Driven Design.*

2. Observasi

Observasi dilakukan untuk mengamati dan memahami kondisi aplikasi Tracer Study Universitas Pendidikan Indonesia. Observasi yang dilakukan berupa pengamatan pada proses bisnis, spesifikasi sistem dan *server, database,* fitur-fitur dan layanan aplikasi, serta kinerja aplikasi sebelum dan sesudah penerapan

arsitektur *microservices.* Observasi juga dilakukan pada kinerja aplikasi Tracer Study UPI sebelum penerapan arsitektur *microservices.*

3. Wawancara

Wawancara dengan staf pengembang aplikasi *monolithic* Tracer Study UPI untuk mengetahui informasi dan data-data yang mendukung penelitian, motivasi pengembangan arsitektur *microservices,* kebutuhan sistem, serta kendala dan harapan selama pengembangan dan pemeliharaan aplikasi Tracer Study UPI. Pengumpulan data dengan cara wawancara dilakukan dengan mengajukan pertanyaan wawancara yang telah disusun dalam Tabel 3. 1.

Tabel 3. 1 Daftar Pertanyaan Wawancara Pengumpulan Data Awal

Kode	Pertanyaan	Sumber
P1	Berapa jumlah pengguna aplikasi Tracer Study UPI?	Jumlah pengguna dapat mempengaruhi beban dan kinerja sistem (Laudon & Laudon, 2012).
P2	Kapan waktu akses dengan traffic tertinggi pada aplikasi Tracer Study UPI?	Mengetahui pola akses pengguna dapat membantu dalam perencanaan kapasitas dan optimasi kinerja (Hoffer et al., 2019).
P3	Berapa jumlah total aplikasi yang dimiliki UPI?	Aplikasi yang jumlahnya banyak dalam satu organisasi dapat menimbulkan masalah dalam hal integrasi dan pemborosan sumber daya (Sommerville, 2011). Kesulitan dalam integrasi antaraplikasi dapat menghambat bisnis, aliran informasi dan menyebabkan silo data (inkonsistensi, fragmentasi, duplikasi data) (Hoffer et al., 2019). Dengan *microservices*, setiap layanan dapat berkomunikasi dan mengakses data melalui *API* standar, sehingga integrasi antara layanan menjadi lebih mudah dan konsisten, serta data dapat dikelola secara terpusat (Namiot & Sneps-Sneppe, 2014; Richardson, 2018). Selain itu, dapat juga dibangun berbagai *services* umum yang dapat digunakan kembali dan diintegrasikan oleh aplikasi lain, sehingga mengurangi redundansi, serta
P4	Berapa jumlah aplikasi yang dibangun oleh tim DSTI UPI?	
P5	Berapa jumlah aplikasi yang dibangun mandiri oleh Fakultas atau Program Studi?	
P6	Berapa jumlah aplikasi yang dibangun oleh vendor atau pihak luar?	
P7	Apakah terdapat aplikasi dengan karakteristik yang sama sehingga menyebabkan redundansi aplikasi (misalnya Fakultas A membangun	

Kode	Pertanyaan	Sumber
	aplikasi yang sama dengan Fakultas B)?	meningkatkan efisiensi pengembangan dan penggunaan sumber daya (Newman, 2021).
P8	Apakah terdapat kendala dalam skalabilitas dan performa aplikasi Tracer Study UPI? Misalnya saat *traffic* sedang sangat tinggi	Penskalaan sistem *monolithic* tidak dapat terpusat pada bagian yang mengalami *load* tinggi, sedangkan arsitektur *microservices* memungkinkan penskalaan dilakukan hanya pada layanan tertentu secara mandiri (Newman, 2021; Richardson, 2018).
P9	Apakah keseluruhan aplikasi Tracer Study UPI perlu di-*deploy* ulang jika terdapat perubahan kecil pada kode program?	Penerapan atau *deployment* aplikasi *monolithic* memakan waktu lebih banyak karena seluruh aplikasi harus di-*deploy* ulang bahkan untuk perubahan kecil sekalipun (Richardson, 2018). Sedangkan *microservices* memungkinkan setiap layanan di-*deploy* masing-masing, sehingga hanya layanan yang mengalami perubahan yang perlu di-*deploy* ulang, ukuran layanan lebih kecil dan mempercepat proses penerapan (Newman, 2020).
P10	Apakah kegagalan sistem pada salah satu layanan dapat mempengaruhi keseluruhan aplikasi?	Komponen dalam sistem *monolithic* bersifat *tightly-coupled* atau berhubungan erat, sehingga perubahan atau kegagalan pada satu bagian dapat mempengaruhi bagian lain dan berpotensi menggagalkan seluruh sistem. Sedangkan *microservices* bersifat *loosely-coupled* dan saling terisolasi antar-*service,* sehingga kegagalan sistem hanya dapat terjadi pada *service* terkait tanpa mempengaruhi *service* lain (Newman, 2021).
P11	Apakah terdapat kendala terkait *incompatible dependencies* akibat dari ketergantungan erat terhadap teknologi atau *dependency* yang digunakan? Misalnya sistem membutuhkan teknologi versi terbaru, namun jika teknologi	Sistem *monolithic* berpotensi mengalami *dependency hell* atau ketergantungan erat dengan *library, package,* atau teknologi yang digunakan, sehingga ketika dilakukan *upgrade* versi dapat mempengaruhi semua layanan dan menyebabkan kegagalan karena tidak kompatibel. Sebaliknya, sistem *microservices* mengisolasi ketergantungan dengan *dependency* dalam masing-masing *service* individu,

Kode	Pertanyaan	Sumber
	tersebut di-*upgrade* maka akan menciptakan kegagalan pada sistem lain.	mengurangi risiko konflik versi dan memudahkan *upgrade* versi (Newman, 2021).
P12	Apakah terdapat kendala terkait penggunaan sumber daya yang besar akibat dari teknologi yang digunakan?	Penggunaan teknologi seperti bahasa pemrograman yang lebih modern memungkinkan penggunaan sumber daya yang lebih efisien, mengurangi *overhead,* dan meningkatkan keamanan sistem. Teknologi yang lebih modern juga mendapatkan pembaruan yang lebih baik dan berkelanjutan dibandingkan dengan teknologi yang sudah usang, sehingga dapat menjaga kompatibilitas dan selalu mendapatkan peningkatan fitur (Anderson, 2020; Cox et al., 2022).
P13	Apakah terdapat kendala terkait kinerja aplikasi akibat dari teknologi yang digunakan?	
P14	Apa jenis arsitektur sistem yang digunakan di UPI?	Pemahaman tentang arsitektur aplikasi membantu dalam proses analisis, evaluasi dan perbaikan sistem (Pressman, 2010).
P15	Apa jenis arsitektur sistem yang diterapkan pada aplikasi Tracer Study UPI?	
P16	Mohon berikan skor 1-5 kepada setiap *driving forces* atau motivasi penerapan *microservices* berikut untuk membuat keputusan migrasi dan menentukan prioritas dalam pengembangan. a) *Optimized scalability* b) *Independent and automated deploy* c) *Easier maintenance and evolution* d) *Independence of team* e) *Loosely coupled service* f) *Cohesive service* g) *Technology flexibility*	Penggunaan pola *trade-off* untuk mengambil keputusan dapat dilakukan dengan membuat daftar manfaat penerapan *microservices,* kemudian memberikan skor/bobot kepada setiap aspek yang ingin diraih dari migrasi sistem *monolithic* ke *microservices* (Newman, 2020; Wolfart et al., 2021). Terdapat beberapa aspek motivasi atau *driving forces* dalam Wolfart et al. (2021) yang dapat dijadikan dasar tujuan dilakukannya migrasi sistem *monolithic* ke *microservices.*

Kode	Pertanyaan	Sumber
	h) *Infrastructure facilities* i) *Agility enabler* j) *Easier reuse* k) *Reduce time to market*	

3. 9 Teknik Analisis Data

Teknik analisis data yang digunakan dalam penelitian ini adalah sebagai berikut.

1. Analisis Kualitatif

Analisis kualitatif dilakukan dengan menganalisis data yang diperoleh dari hasil studi literatur, observasi, dan wawancara dengan pengembang aplikasi. Analisis ini dibutuhkan untuk mengetahui kebutuhan pengembang, serta kendala yang dihadapi saat mengembangkan dan menggunakan aplikasi Tracer Study. Analisis ini juga dibutuhkan untuk melakukan dekomposisi aplikasi yang sudah ada dan perancangan aplikasi yang mengimplementasikan arsitektur *microservices.*

2. Analisis Kuantitatif

Analisis kuantitatif dilakukan dengan mengukur kinerja aplikasi sebelum dan setelah dilakukan pengembangan menggunakan arsitektur *microservices* menggunakan metode dekomposisi *Domain-Driven Design*. Analisis ini dilakukan dengan mengukur waktu respon *(response time)* aplikasi, jumlah pengguna yang dapat dilayani secara bersamaan *(throughput)*, dan persentase kegagalan dalam menangani *request (error rate).*

3. 10 Alat dan Bahan Penelitian

Pada penelitian ini digunakan alat dan bahan yang mendukung berjalannya proses penelitian agar penelitian berjalan efektif dan mendapatkan hasil yang optimal. Alat penelitian merupakan perangkat yang digunakan selama proses penelitian, sedangkan bahan penelitian merupakan sesuatu yang dianalisis atau data-data yang diteliti. Dalam penelitian ini, digunakan alat dan bahan sebagai berikut.

3. 10. 1. Alat Penelitian

Alat penelitian yang digunakan dalam penelitian ini berupa perangkat keras *(hardware),* peladen *virtual (Virtual Private Server/VPS),* dan perangkat lunak *(software)* sebagai berikut.

1. Perangkat Keras

a. *Processor* : Intel Core i5-8250U
b. *Memory* : *12GB RAM*
c. *Storage:* : *512GB SSD* dan *1TB HDD*

2. Peladen *Virtual*

a. *CPU* : 2 *Core*
b. *Memory* : *8GB RAM*
c. *Storage* : *40GB SSD*
d. *Operating System* : Linux Ubuntu 20.04 LTS 64 *bit*

3. Perangkat Lunak

a. *Operating System* : Windows 11 Home Single Language 64 bit
b. *Code Editor* : Visual Studio Code
c. *Browser* : Google Chrome
d. *Diagram Maker* : Draw.io
e. *Software Testing* : Postman dan Apache JMeter
f. *Container Manager* : Docker

3. 10. 2. Bahan Penelitian

Bahan penelitian yang digunakan dalam penelitian ini meliputi:

1. **Data hasil observasi** sistem Tracer Study UPI berupa proses bisnis sistem, alur kerja sistem, pengolahan data pada sistem, model basis data, fitur-fitur dan layanan, spesifikasi sistem, serta arsitektur sistem.
2. **Data hasil wawancara** dengan pengembang mengenai data-data dan informasi lapangan, motivasi penerapan *microservices,* kebutuhan sistem, kendala dan harapan terhadap sistem baik pada saat proses pengembangan maupun penggunaan. Data-data ini dijadikan dasar selama pengembangan *microservices* pada sistem Tracer Study UPI.

3. **Data hasil pengujian kinerja sistem *monolithic*** Tracer Study UPI yang dikumpulkan menggunakan Apache JMeter pada saat tahap pengumpulan data awal. Data ini digunakan sebagai acuan pengembangan *microservices* dengan berfokus pada peningkatan kinerja sistem.
4. **Data hasil pengujian kinerja sistem *microservices*** Tracer Study UPI yang dikumpulkan menggunakan Apache JMeter pada saat tahap eksekusi pengembangan *microservices.* Data ini digunakan sebagai salah satu ukuran untuk menilai keberhasilan migrasi sistem dari *monolithic* ke *microservices* dengan membandingkan kedua hasil pengujian.