Jurnal Inovasi Teknik dan Edukasi Teknologi, 3(1), 2023, 12–23
ISSN: 2797-7196 (online)
DOI:10.17977/um068v3i12023p12-23

# Pemanfaatan Microservice dengan GraphQL Federation Concept untuk Pengembangan Sistem Informasi Akademik (xSIA)

**[1]Poetri Lestari Lokapitasari Belluano*, [2]Benny Leonard Enrico P*, [1]Purnawansyah, [1]Amaliah Faradibah, [1]Rahmadani**

[1]Universitas Muslim Indonesia, l. Urip Sumoharjo No.km.5, Kota Makassar, Indonesia, 90231
[2]Universitas Pancasakti Makassar Jl. Andi Mangerangi No.73, Kota Makassar, Indonesia 90121
poetrilestari@umi.ac.id, blep@unpacti.ac.id*

Paper received: 03-01-2023; revised: 15-01-2023; accepted: 30-01-2023

**Abstract**

Academic Information System (xSIA) is an application built to manage academic value transaction modules that make it easy for users to manage grades in online academic administration activities. The need for reconstruction of the xSIA microservice architecture from the previously built domain driven design model using the json (javascript object notation) data format, REST (Representational State Transfer and an architectural style for distributed hypermedia systems) communication protocol, authorization and authentication processes occur in each microservice , there is a pooling of data that is charged to the client which has caused the client to make many requests to the various available microservices, as well as making documentation if there are additional microservices. The xSIA system reconstruction was developed by changing the xSIA microservice architecture so that the concept of responsibility authorization and authentication can be carried out according to service needs. The approach to reconstructing the microservice architecture in the xSIA application uses a new concept with the single gateway microservice model and is built using the GraphQL Federation to facilitate data communication between the backend and frontend of the application and can be implemented in various programming languages to minimize downtime when modification process occurs. The results of this study are the xSIA application on the study plan transaction module (krs) using the GraphQL Federation Concept with the single gateway microservice model so that authorization and authentication responsibilities can be carried out according to service requirements with a realtime average of 373.15 milliseconds.

**Keywords:** xSIA; *microservice; GraphQL*

**Abstrak**

Sistem Informasi Akademik (xSIA) adalah aplikasi yang dibangun untuk mengelola modul transaksi nilai akademik yang memberikan kemudahan kepada pengguna mengelola nilai dalam kegiatan administrasi akademik secara *online*. Kebutuhan rekonstruksi arsitektur *microservice* xSIA dari model *domain driven design* yang dibangun sebelumnya menggunakan format data *json (javascript object notation)*, protokol komunikasi *REST (Representational State Transfer and an architectural style for distributed hypermedia systems)*, terjadi proses otorisasi dan otentikasi yang ada di setiap *microservice*, terdapat penyatuan data yang dibebankan kepada *client* telah menyebabkan *client* harus melakukan banyak *request* ke berbagai *microservice* yang tersedia, serta pembuatan dokumentasi jika ada penambahan *microservice.* Rekonstruksi sistem xSIA dikembangkan dengan mengubah arsitektur *microservice* xSIA sehingga konsep *responsibility* autorisasi dan autentifikasi dapat dilakukan sesuai dengan kebutuhan *service*. Pendekatan dalam melakukan rekontruksi *arsitektur microservice* pada aplikasi xSIA menggunakan konsep baru dengan model *single gateway microservice* (layanan satu gerbang) dan dibangun menggunakan *GraphQL Federation* untuk mempermudah komunikasi data antara *backend* dan *frontend* dari aplikasi, serta dapat diimplementasikan di berbagai Bahasa pemrograman sehingga meminimaliasir terjadinya *downtime* saat proses modifikasi terjadi. Hasil penelitian ini berupa aplikasi xSIA pada modul transaksi rencana studi (KRS) menggunakan *GraphQL Federation Concept* dengan model *single gateway microservice* sehingga *responsibility* autorisasi dan autentifikasi dapat dilakukan sesuai dengan kebutuhan *service* dengan rerata *realtime* 373.15 *millisecond*.

**Kata Kunci**: xSIA; *microservice; GraphQL*

## 1. Pendahuluan

Sebuah sistem dianggap berakhir jika tidak ada inovasi yang dilakukan terhadap sistem tersebut, xSIA yang dikembangkan untuk kebutuhan sistem informasi akademik juga harus melakukan inovasi, baik dalam penambahan fitur, pembaharuan teknologi maupun pembaharuan metodologi. Dalam pelaksanaan inovasi dalam ketiga kategori tersebut, sangat sering terjadi kegagalan, salah satu contoh konkrit adalah perubahan dari arsitektur monolithic menjadi arsitektur *microservice*.

Pada penelitian sebelumnya arsitektur *microservice* xSIA dikembangkan dengan model *domain driven design* yang dibangun menggunakan format data JSON (*javascript object notation*), protokol komunikasi REST (*REpresentational State Transfer and an architectural style for distributed hypermedia systems*), proses otorisasi dan otentikasi yang ada di setiap *microservice*, proses penyatuan data yang dibebankan kepada client yang menyebabkan *client* harus melakukan banyak request ke berbagai *microservice* yang tersedia, dan pembuatan dokumentasi jika ada penambahan *microservice*(Lokapitasari Belluano et al., 2019).

Sehingga terjadi beberapa faktor kegagalan diantaranya pemilihan protokol komunikasi antara *microservice*, konsep autorisasi dan autentifikasi, penyatuan data dari berbagai microservice dan dokumentasi yang harus dibuat untuk kebutuhan pengembang lain untuk menggunakan layanan *microservice* tersebut. Adapun alasan dianggap gagal dalam perubahan arsitektur tersebut disebabkan terlalu memakan banyak waktu dan sumberdaya sehingga perlu dilakukan bentuk pendekatan lain dalam perubahan arsitektur *monolithic* menjadi arsitektur *microservice* (Karpen, 2012; Richardson, n.d., 2019).

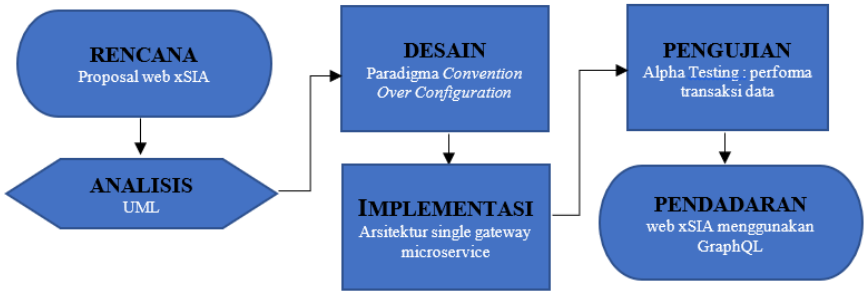
Pendekatan dalam melakukan rekontruksi arsitektur *microservice* pada aplikasi xSIA perlu menggunakan konsep baru dengan model single gateway microservice (layanan satu gerbang) dimana *clients* memanggil *Application Programming Interface Gateway* (API *gateway*), yang meneruskan panggilan ke layanan yang sesuai di bagian belakang(Belluano, 2018; Wasson & Schonning, 2019), dan dibangun menggunakan *GraphQL Federation* untuk mempermudah komunikasi data antara backend dan *frontend* dari aplikasi dimana GraphQL merupakan bahasa *query* baru yang diusulkan oleh *Facebook* untuk mengimplementasikan API berbasis Web dan GraphQL dapat mengurangi ukuran dokumen JSON yang dikembalikan oleh REST API(Brito et al., 2019; Purnawansyah & Faradibah, 2014), dapat diimplementasikan di berbagai bahasa yang tidak bergantung pada bahasa pemrograman di sisi *server* dan database apapun, serta meminimaliasir terjadinya *downtime* saat proses modifikasi dan pengembangan rekonstruksi xSIA.

## 2. Metode

Berikut ini merupakan tahapan dan langkah-langkah dalam penelitian yang dilakukan.

### 2.1. Tahapan Penelitian

Tahapan penelitian menggunakan model Prototyping yang akan dijalankan antara lain:

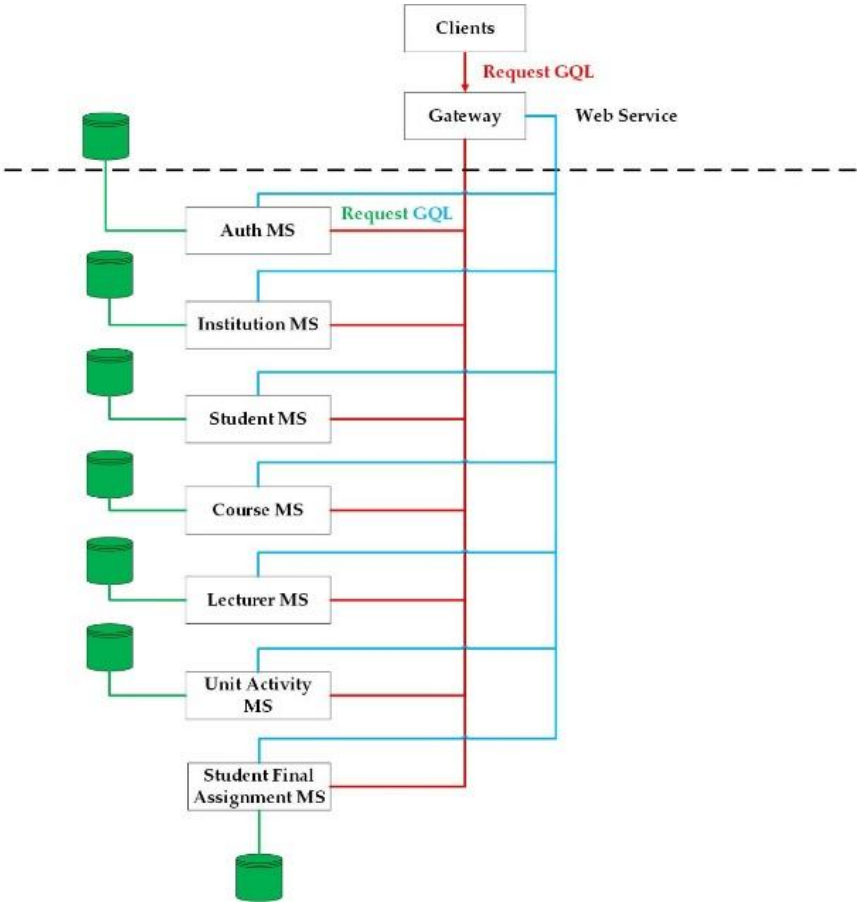



**Gambar 1. Tahapan Penelitian**

- Rencana, tahapan ini dilakukan identifikasi dimana peneliti berusaha menguraikan permasalahan yang ditemui tentang pengembangan sistem informasi berbasis web untuk diimplementasikan pada xSIA(Arifin, 2011; Satoto & Kodrat Iman, 2008)
- Analisis merupakan tahapan pembuatan analisa aliran kerja xSIA yang sedang berjalan sesuai dengan hasil analisis kebutuhan menggunakan Unified Modeling Language (UML) tools
- Desain adalah merancang aliran kerja manajemen, desain pemrograman, dan pengembangan sistem yang menerapkan paradigma Convention Over Configuration
- Implementasi adalah tahap pengembangan untuk membangun atau memperharui sistem aplikasi xSIA sesuai dengan arsitektur single gateway microservice
- Pengujian dilakukan setelah aplikasi dibangun dan dikembangkan menggunakan konsep GraphQL untuk mengetahui performa xSIA sehingga diharapkan sistem dapat berjalan sesuai alur transaksi layanan sistem.

## 2.2. Rancangan Penelitian

Rancangan penelitian yang dipilih adalah rancangan penelitian eksperimen prototipe untuk implementasi konsep GraphQL:




**Gambar 2. Alur xSIA menggunakan arsitektur *single gateway* graphQL**

## 2.3. Teknik Pengumpulan Data

Rancangan penelitian yang dipilih adalah rancangan penelitian eksperimen prototipe untuk implementasi konsep GraphQL:

Teknik pengumpulan data yang akan dilaksanakan, meliputi:

### 2.3.1. Penelitian lapangan (*field research*) yaitu berupa kegiatan yang dilakukan secara langsung ke lokasi penelitian untuk memperoleh data-data konkrit mengenai masalah yang akan dibahas melalui dua cara, diantaranya:

- Pengamatan (observasi), dengan cara mengamati langsung kegiatan pengolahan data nilai akademik siswa di SDN No.133 yang dilaksanakan oleh para entitas yakni Guru Mata Pelajaran.
- Wawancara (interview), Pengumpulan data ataupun informasi yang dilakukan dengan cara berkomunikasi langsung dengan pihak-pihak yang dianggap berkompeten dan mampu memberikan informasi (narasumber) yang lebih terperinci terhadap permasalahn yang sedang diteliti sehubungan dengan pengolahan Sistem Informasi Akademik.

**2.3.2. Penelitian kepustakaan (*library research*) adalah metode data dengan mencari data, mempelajari banyak data dari berbagai sumber buku, modul, artikel baik perpustakaan maupun internet yang berhubungan dengan pokok pembahasan terkait arsitektur layanan sistem satu gerbang (*API gateway microservice*) dan graphQL(Hartig & Hidders, 2019).**

## 3. Hasil dan Pembahasan

Berikut ini merupakan hasil berupa rancangan penelitian eksperimen prototipe dalam implementasi konsep GraphQL:

### 3.1. Rancangan Procedure

Hasil penelitian diimplementasikan dengan spesifikasi *hardware* dan *software*, diantaranya:

- Server hardware berupa Processor 1.8 GHz, Memory 2 GB
- Server Software berupa Operating System ubuntu server 20.04, RDBMS PostgreSQL 14, dan HTTP/2
- Typescript 4.9
- Framework NestJS

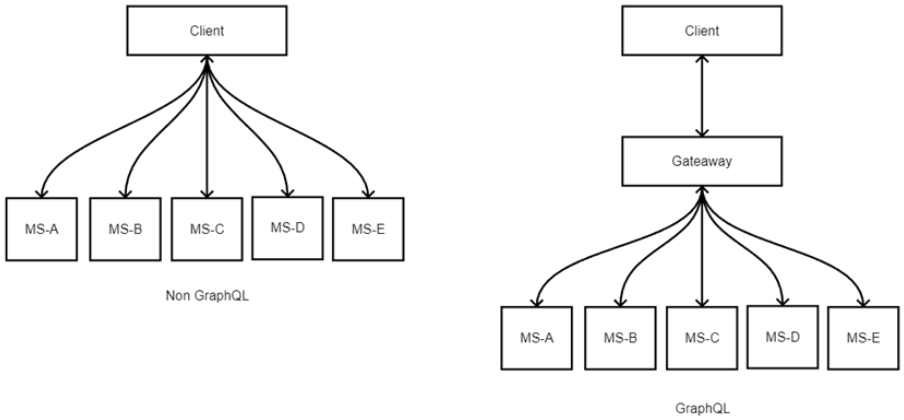
*Microservice* ini dibangun dari konsep pembangunan applikasi monolitik dari penelitian sebelumnnya, dimana modul modul yang ada ditransformasikan menjadi kumpulan *microservice* yang dibangun sekarang. Microservice ini dibangun dengan menggunakan Bahasa pemograman typescript dan menggunakan *framework* NestJS untuk kebutuhan NodeJS applikasi *server*. *Framework* NestJS mempunyai konsep modular untuk setiap s*ub module* yang ditangani oleh *microservice* yang dibangun. *Microservice* yang dibangun terdiri dari sepuluh *microservice* yaitu:

- Microservice Gateaway adalah layanan berfungsi sebagai pintu masuk permintaan terhadap layanan layanan tersedia.
- Microservice Person adalah layanan yang berfungsi menyimpan informasi pribadi seseorang.
- Microservice Institusi adalah layanan yang berfungsi menyimpan informasi sebuah institusi beserta unit kerja yang ada diinstitusi tersebut.
- Microservice Generic adalah layanan yang berfungsi menyimpan data – data referensi umum yang digunakan oleh microservice lain, contohnya adalah data tahun akademik.
- Microservice Contact adalah layanan yang menyimpan alamat dari sebuah institusi maupun individual, selain itu layanan ini juga menyimpan data desa, kecamatan, kabupaten dan propinsi dari sebuah negara.
- Microservice Academic Lecturer adalah layanan yang berfungsi menyimpan informasi dosen baik pangakat akademik, bentuk kontrak serta pangkat golongan.
- Microservice Academic Course adalah layanan yang berfungsi menyimpan matakuliah dari sebuah program studi untuk digunakan pada microservice lain.

- Microservice Unit Campaign adalah layanan yang berfungsi menyimpan aktifitas kegiatan dari sebuah program studi, dimana aktifitas kegiatan tersebut adalah pengajaran, index nilai matakuliah dan pembukaan kelas perkuliahan.
- Microservice Academic Student adalah layanan yang menyimpan menyimpan data mahasiswa dan data aktifitas mahasiswa pada setiap semester tahun akademik.

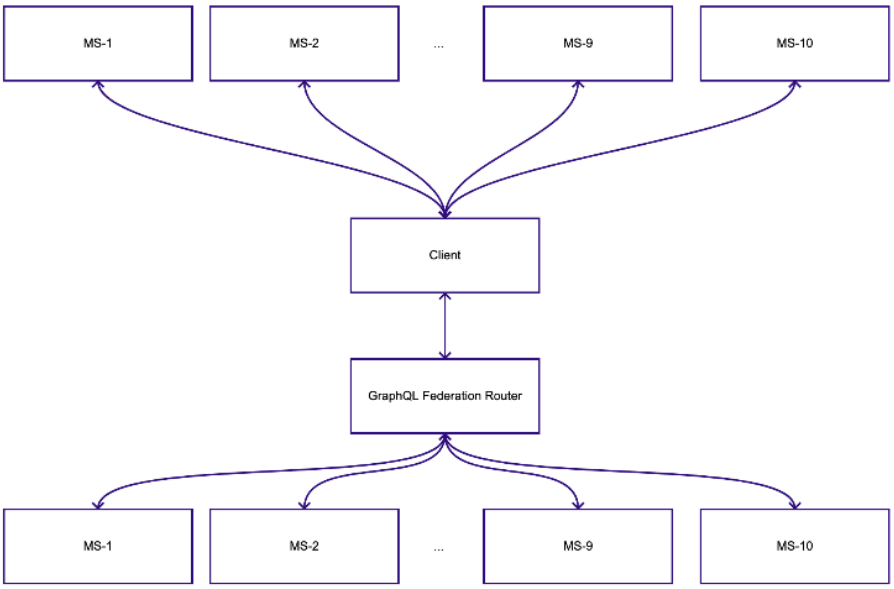
*Microservice* tersebut diatas dirancang untuk saling berkomunikasi dengan menggunakan protocol *graphql* untuk melayani permintaan – permintaan yang masuk melalui *microservice gateaway* dan mengembalikan data yang diinginkan oleh yang meminta data.

Pada Gambar 3. memperlihatkan perbedaan antara layanan *microservice* yang menggunakan protokol GraphQL dan yang tidak menggunakan protokol GraphQL adalah *microservice* yang menggunakan protokol GraphQL terdapat satu layanan yang dinamakan *gateaway* yang menjadi pintu masuk dari permintaan yang muncul dan penggabungan data terjadi pada sisi *server* dibandingkan dengan yang tidak menggunakan GraphQL penggabungan data terjadi di sisi client.




**Gambar 3. Perbedaan Konsep Arsitektur Non-GraphQL dan GraphQL**

Pada Gambar 3. menunjukkan perbedaan konsep dengan membuat rekontruksi arsitektur microservice pada aplikasi xSIA menggunakan konsep model *single gateway microservice* (layanan satu gerbang) yang dibangun menggunakan GraphQL *Federation* untuk mempermudah komunikasi data antara *backend* dan frontend aplikasi.




**Gambar 4. Arsitektur xSIA dengan model satu gerbang menggunakan konsep GraphQL**

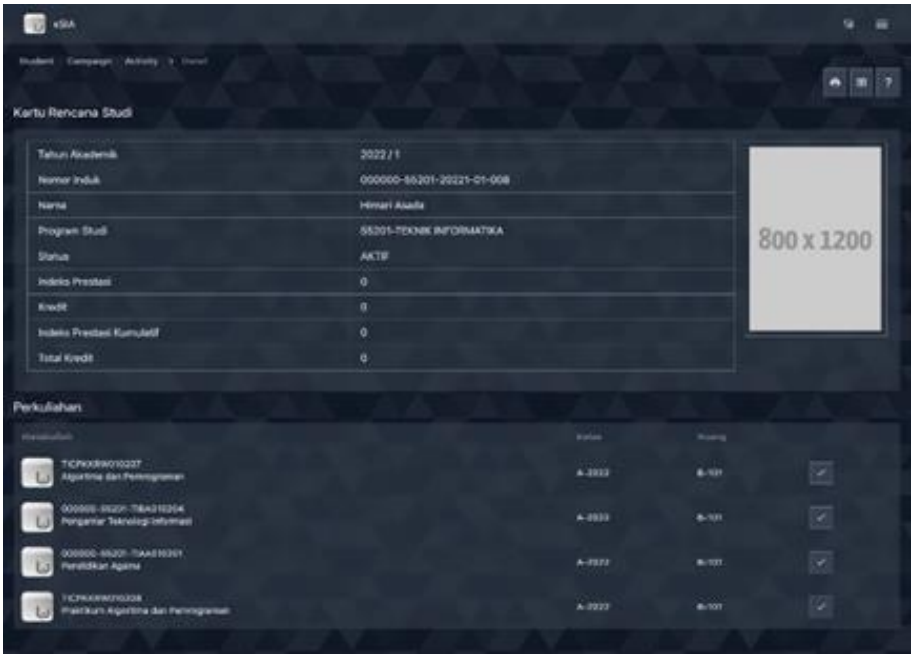
Pada Gambar 4 dengan model satu gerbang (single gateway microservice) dimanfaatkan dengan tujuan responsibility autorisasi dan autentifikasi dapat dilakukan sesuai dengan kebutuhan service yang dikembangkan oleh xSIA, dimana melalui GraphQL API semua data hanya dalam satu permintaan (single request), sehingga menjadikan transaksi data pada aplikasi dapat berjalan dengan cepat bahkan dengan kondisi jaringan internet yang lambat.

GraphQL API diatur berdasarkan *types* dan fields. Types untuk memastikan aplikasi xSIA hanya meminta data yang diperlukan untuk menghindari kesalahan yang dapat terjadi serta dapat menghindari terjadinya manual *parsing code*.

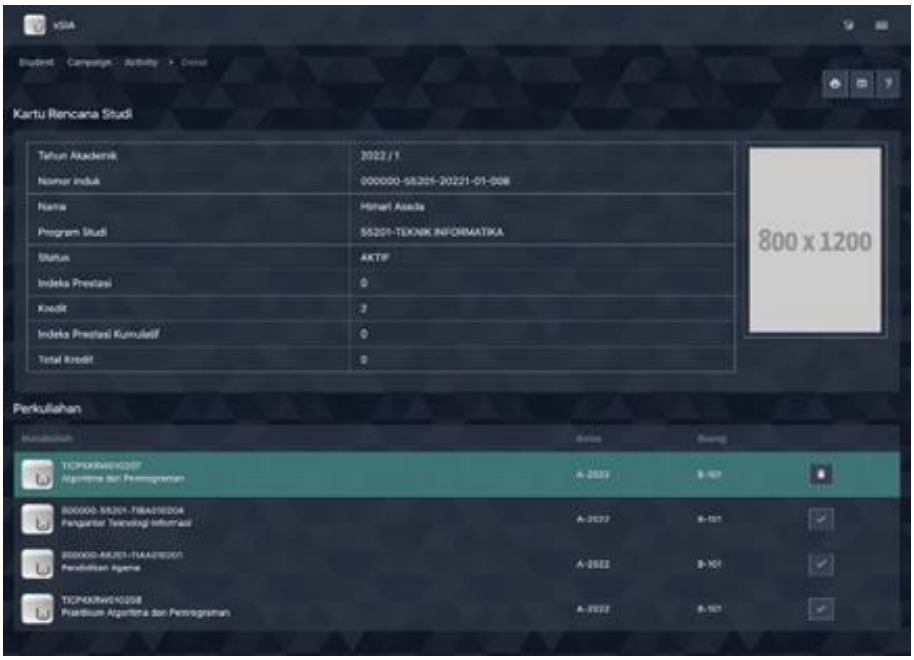
## 3.2. Implementasi antrmuka Pengguna

Aplikasi antar muka dibangun dengan menggunakan bahasa pemograman *Typescript* dengan menggunakan *framework* VueJS, *framework* VueJS adalah *framework* antar muka yang bersifat reaktif terhadap "*Document Object Model*" pada peramban dengan memanfaatkan fitur "*virtual document object model*" untuk menghasilkan reaktifitas yang membuat aplikasi menjadi interaktif. Gambar 5 menunjukkan antar muka yang dibangun ditujukan pada kelompok pengguna dalam lingkup mahasiswa dengan menyediakan fitur – fitur yakni:

- Profil Mahasiswa, dimana pada fitur ini menampilkan data mahasiswa pada aplikasi, seperti data pribadi dan data perkuliahan mahasiswa
- Modul Aktifitas Mahasiswa dimana pada fitur ini terjadi interaksi mahasiswa dengan sistem pada saat merencanakan kegiatan aktifitas perkuliahan.



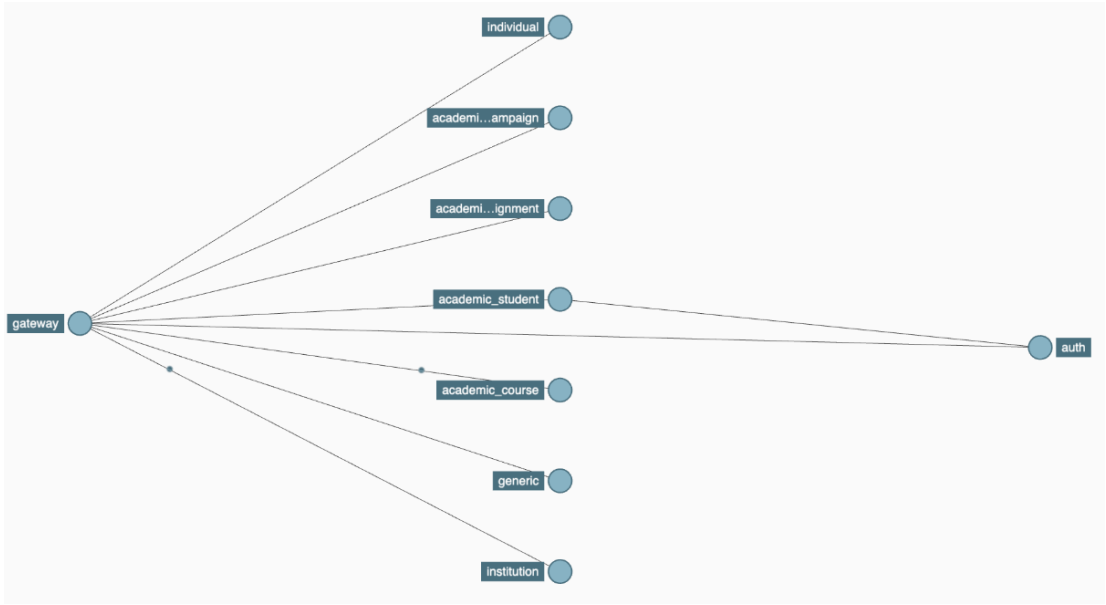

Sebelum memilih matakuliah




Setelah memilih matakuliah

**Gambar 5. Aktivitas mengisi rencana studi mahasiswa**

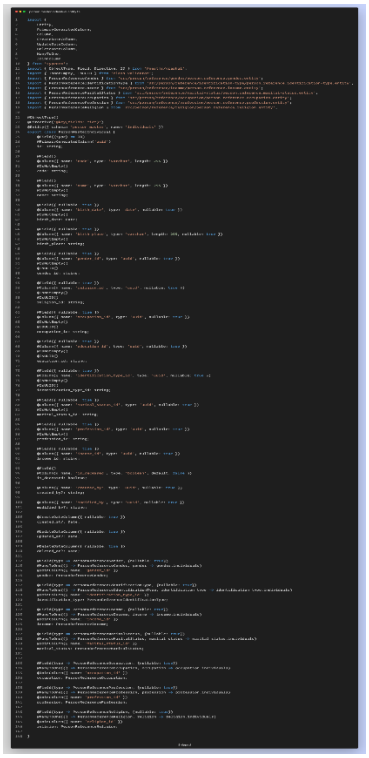
Gambar 6. *Microservice* xSIA dengan konsep GraphQL memiliki *gateway* sebagai gerbang untuk terhubung dengan berbagai *service* yakni individual, *academic_campaign, academic_student, academic_*asignment*, academic_cource, institution, Generic* yang kesemuanya dimanfaatkan sebagai layanan konfirmasi permintaan transaksi *client*.

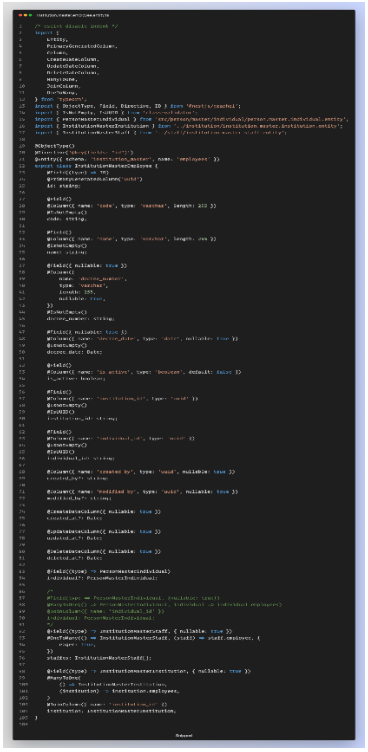



**Gambar 6. Arsitektur service xSIA pada konsep graphQL**

Ketika client memiliki *request* untuk transaksi yang melibatkan data *academic_student*, maka permintaan terhadap sebuah aksi, *academic_student* meneruskan token yang terdapat pada *request* untuk di *authorisasi* oleh auth_*service*, jika *authorisasi* berhasil maka *academic student* melaksanakan permintaan dari *request*, jika tidak maka *academic_student* mengembalikan pesan kepada *client* bahwa permintaan tidak dilaksanakan karena tidak memiliki otorisasi.

Gambar 7(a),(b),(c) adalah simulasi untuk implementasi *code* yang menunjukkan model transaksi yang menghubungkan antara microservice pada xSIA.





Gambar 7(a) Ms_model_individu Gambar 7(b) Ms_institution_model_employee

person.master.individual.entity.ts

```
1  import { Directive, ObjectType, Field, ID } from '@nestjs/graphql';
2  import { InstitutionMasterEmployee } from 'src/institution/master/employee/institution.master.employee.entity';
3  import { OneToMany } from 'typeorm';
4
5  @ObjectType()
6  @Directive('@extends')
7  @Directive('@key(fields: "id")')
8  export class PersonMasterIndividual {
9      @Field((type) => ID)
10     @Directive('@external')
11     id: string;
12
13     @Field((type) => [InstitutionMasterEmployee], {nullable: true})
14     employees?: InstitutionMasterEmployee[];
15 }
```

Snipped

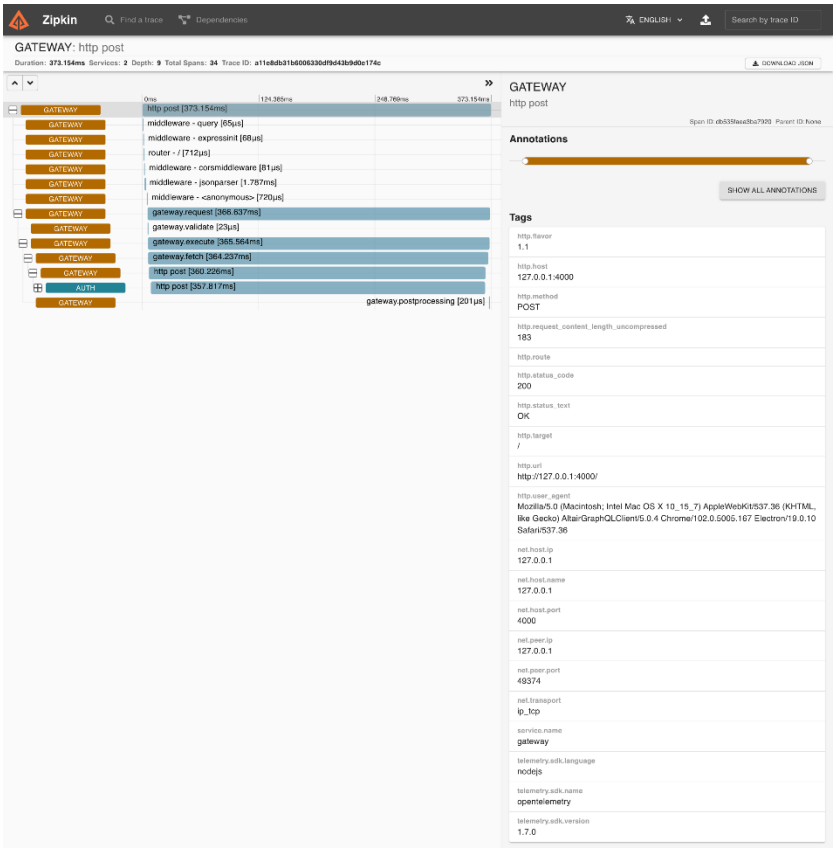
**Gambar 7(c) Master_individual_on_institution**

Gambar 7(a) memperlihatkan hubungan antara model individual dengan model lain secara internal dalam *microservice* individual. Ketika model individual membutuhkan data referensi untuk dipadukan dengan data individual (*lines code* 112 – 146)

Hubungan dengan *microservice* lain pada gambar 7(b) yakni sebuah data employee yang terhubung dengan model lain secara internal (lines code 85 – 98), sedangkan untuk kebutuhan data personal dari seorang *employee* maka dibutuhkan data dari *microservice* individual. Hubungan antara dua *microservice* diatur seperti pada gambar 7(c) yang berada pada microservice institution, dimana hubungan antara dua entitas dari *microservice* yang berbeda (lines 10) bahwa data tersebut diperoleh dari sumber eksternal yaitu *microservice* individual. *Konsep GraphQL Federation* dapat mempermudah pengembang dalam menggunakan layanan sistem xSIA, hal ini ditunjukkan Gambar 8 dimana seluruh *service* tetap berjalan selama aplikasi xSIA berada pada proses pengembangan.

| id | name | namespace | version | mode | pid | uptime | ↺ | status | cpu | mem | user | watching |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 10 | ms_00_gateaway | default | 1.0.0 | fork | 8965 | 3s | 3 | online | 20.3% | 59.5mb | bendo01 | disabled |
| 9 | ms_01_auth | default | 0.0.1 | fork | 8958 | 5s | 1 | online | 32% | 74.1mb | bendo01 | disabled |
| 8 | ms_02_individual | default | 0.0.1 | fork | 8951 | 6s | 1 | online | 35% | 81.0mb | bendo01 | disabled |
| 7 | ms_03_institution | default | 0.0.1 | fork | 8943 | 9s | 1 | online | 36% | 107.0mb | bendo01 | disabled |
| 6 | ms_04_generic | default | 0.0.1 | fork | 8938 | 9s | 1 | online | 35% | 105.6mb | bendo01 | disabled |
| 5 | ms_05_contact | default | 0.0.1 | fork | 8924 | 12s | 1 | online | 32% | 132.3mb | bendo01 | disabled |
| 4 | ms_06_academic_lecturer | default | 0.0.1 | fork | 8923 | 12s | 1 | online | 30% | 127.8mb | bendo01 | disabled |
| 3 | ms_07_academic_course | default | 0.0.1 | fork | 8909 | 16s | 1 | online | 29% | 145.1mb | bendo01 | disabled |
| 2 | ms_08_academic_unit_campaign | default | 0.0.1 | fork | 8908 | 16s | 1 | online | 27% | 144.5mb | bendo01 | disabled |
| 1 | ms_09_academic_student | default | 0.0.1 | fork | 8896 | 19s | 1 | online | 0% | 116.4mb | bendo01 | disabled |
| 0 | ms_10_academic_student_final_assignment | default | 0.0.1 | fork | 8895 | 19s | 1 | online | 16% | 116.7mb | bendo01 | disabled |

**Gambar 8. Service transaction**




**Gambar 9. Pengukuran GraphQL**

Pengukuran responsibility menggunakan standar *OpenTelemetry* untuk memantau kemampuan kinerja xSIA sebagai aplikasi baik di sisi service maupun kinerja sebagai aplikasi *standalone* khususnya mengukur kinerja waktu layanan sistem terdistribusi. Gambar 9 pada akses login setiap permintaan transaksi diawali dengan mengaktifkan *gateway* untuk terhubung ke *microservice* auth sehingga ssitem dapat mengontrol aktivitas transaksi data yang disesuaikan dengan kebutuhan permintaan oleh *client*.

Sistem xSIA memproses aktivitas tiap *request* pada halaman login oleh client secara realtime dengan kecepatan 373.15 *millisecond*. Seluruh aktivitas yang berjalan berhasil terekam dengan beberapa tahapan transaksi yakni :

- Middleware-query: 65 microseconds
- Middleware-expressinit untuk microservice sebesar 68 microsecond
- Router: 712 microseconds
- Middleware-Corsmiddleware untuk keamanan browser: 81 microseconds
- Middleware-jsonparser untuk merubah request menjadi data yang dikirimkan ke service auth: 1,787 microseconds
- Gateway.request sebagai aktivitas permintaan data: 366,367 millisecond
- Gateway.validation yang melaksanakan validasi permintaan data: 23 microsecond
- Gateway.execute yang mengeksekusi data yang tervalidasi: 365.564 millisecond.

- Gateway.fetch yang melakukan persiapan pengiriman data ke client: 365.237 millisecond
- http post yakni kesiapan system mengirim data ke microservice auth: 360.226 millisecond
- http post yakni berhasil mengirim data ke microservice auth: 360.226 millisecond
- gateway postprocessing mengirim data request sesuai permintaan client: 201 microseconds

Tiap prosedur fungsi yang dijalankan memiliki waktu yang berbeda dengan keberadaan *gateway* dan *auth* yang rerata terukur dengan kecepatan *millisecods*, sehingga *responsibility* pada autorisasi dan autentifikasi dilakukan sesuai kebutuhan *service* menggunakan *single gateway microservice*.

## 4. Simpulan

Pendekatan dalam melakukan rekontruksi arsitektur *microservice* pada aplikasi xSIA menggunakan konsep baru dengan model *single gateway microservice* (layanan satu gerbang) dan dibangun menggunakan GraphQL *Federation* untuk mempermudah komunikasi data antara *backend* dan *frontend* dari aplikasi, serta dapat diimplementasikan di berbagai Bahasa pemrograman sehingga meminimaliasir terjadinya *downtime* saat proses modifikasi terjadi. Aplikasi xSIA pada modul transaksi rencana studi (KRS) dapat dilakukan sesuai dengan kebutuhan *service* dengan rerata *realtime* 373.15 *millisecond*, sehingga penerapan konsep GraphQL *Federation* mempermudah pengembang dalam menggunakan layanan sistem xSIA yang dibangun.

## Daftar Rujukan

Arifin, M. (2011). *Aplikasi Web Dengan Simulasi Kredit Menggunakan Codeigniter Framework Pada Toko Langgeng Elektronik*. https://media.neliti.com/media/publications/207585-none.pdf

Belluano, P. L. L. (2018). Pengembangan Single Page Application Pada Sistem Informasi Akademik. *ILKOM Jurnal Ilmiah*, *10*(1), 38–43.

Brito, G., Mombach, T., & Valente, M. T. (2019). Migrating to GraphQL: A Practical Assessment. *2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, 1534–5351. https://doi.org/10.1109/SANER.2019.8667986

Hartig, O., & Hidders, J. (2019). Defining Schemas for Property Graphs by using the GraphQL Schema Definition Language. *Proceedings of the 2nd Joint International Workshop on Graph Data Management Experiences & Systems (GRADES) and Network Data Analytics (NDA)*, *6*. https://doi.org/https://doi.-org/10.1145/3327964.3328495

Karpen. (2012). Antarmuka Sebagai Media Komunikasi Dengan Sistem. *Jurnal Sains Dan Teknologi Informasi*, *Vol 1*.

Lokapitasari Belluano, P. L., Herman, H., & Panggabean, B. L. E. (2019). Pengembangan Antarmuka Aplikasi Menggunakan Prinsip General Data Protection Regulation. *ILKOM Jurnal Ilmiah*, *11*(1), 59–66. https://doi.org/10.33096/ilkom.v11i1.400.59-66

Purnawansyah, P., & Faradibah, A. (2014). *Implementasi Web Service pada Aplikasi Sistem Informasi Akademik dengan Platform Mobile*. *April*.

Richardson, C. (n.d.). *About Microservice.io*. https://microservices.io/

Richardson, C. (2019). *Microservices Pattern*. Manning Publications.

Satoto, & Kodrat Iman. (2008). Analisis Keamanan Sistem Informasi Akademik Berbasis Web Di Fakultas Teknik Universitas Diponegoro. *Artikel Ilmiah Terpublikasi.Universitas Diponegoro*.

Wasson, M., & Schonning, N. (2019). *Build microservices on Azure*.