Implementasi Arsitektur Microservices menggunakan RESTful API untuk Website Online Course Esploor

Tata Redha Al Fath*[1], Imam Husni Al Amin[2]
[1,2]Universitas Stikubank;Jl. Tri Lomba Juang No.1, (024) 8451976
e-mail: *[1]tataredhaalfath13@gmail.com,
[2]imam@edu.unisbank.ac.id

Abstrak

Perkembangan teknologi yang pesat mempengaruhi berbagai aspek kehidupan seperti bidang pendidikan. Banyak platform pembelajaran online yang menyediakan berbagai materi yang dibutuhkan masyarakat. Aplikasi kursus online biasanya terdiri dari banyak layanan yang saling terhubung. Secara tradisional aplikasi dibangun dengan pendekatan monolitik yang menjadikan semua layanan berada pada satu codebase yang sama sehingga sering mengalami banyak masalah ketika sistem berkembang semakin besar. Untuk memudahkan dalam pemeliharaan dan pembaharuan, aplikasi kursus online akan dibuat menggunakan arsitektur microservices. Microservices ditujukan agar setiap service dapat melakukan satu fungsionalitas dengan baik. Antar service satu dengan service lain dapat terhubung menggunakan Application Programming Interface (API) seperti REST. Perancangan sistem dilakukan dengan metode Web Service Implementation Methodology (WSIM). Aplikasi kursus online ini menggunakan dua bahasa yaitu JavaScript dan PHP pada sisi BackEnd nya dan MySQL sebagai Database Management System (DBMS) untuk menyimpan data dan teknologi ReactJs untuk membangun tampilan antarmuka pengguna.

Kata kunci*— Microservices, Online Course, RESTful API*

Abstract

Rapid technological developments affect various aspects of life, such as education. Many online learning platforms provide various materials needed by the community. Online course applications usually consist of many interconnected services. Traditionally, applications are built with a monolithic approach that makes all services in the same codebase, so they often experience many problems as the system grows larger. The online course application will be created using a microservices architecture to facilitate maintenance and update. Microservices are intended so that each service can perform one function well. One service with another service can be connected using the Application Programming Interface (API) such as REST. The system design uses the Web Service Implementation Methodology (WSIM). This online course application uses two languages, JavaScript and PHP on the BackEnd side and MySQL as a Database Management System (DBMS) to store data and ReactJs technology to build a user interface.

Keywords— *Microservices, Online Course, RESTful API*

1. PENDAHULUAN

Dokumen Perkembangan teknologi yang sangat pesat memiliki pengaruh dalam berbagai aspek kehidupan dan memberikan andil besar terhadap berbagai bidang seperti pendidikan. Perkembangan teknologi ini memberikan kemudahan pada proses belajar mengajar sehingga dapat dilakukan dimana saja dan kapan saja [1]. Seiring dengan berkembanganya teknologi, banyak platform pembelajaran online yang menyediakan berbagai materi pembelajaran yang diperlukan oleh masyarakat. Penggunaan platform pembelajaran online ini juga kian meningkat semenjak aktivitas belajar mengajar di tiadakan karena pandemi corona. Kenaikan ini akan terus bertambah selama kebijakan belajar dirumah masih diberlakukan [2].

Aplikasi kursus online biasanya terdiri dari banyak layanan yang saling terhubung satu sama lain untuk mempermudah proses pemeliharaan dan pembaharuan sistem. Perkembangan teknologi dalam membangun sebuah aplikasi semakin berkembang pesat terutama aplikasi berbasis website.

Banyak teknologi yang dapat digunakan dalam membangun sebuah aplikasi sehingga penentuan teknologi yang akan digunakan inilah yang nanti menentukan apakah aplikasi yang dibangun akan lebih unggul dibandingkan aplikasi yang lainnya atau tidak.

Secara tradisional, aplikasi website dibangun menggunakan pendekatan monolitik. Ketika membangun sistem berskala besar dengan pendekatan monolotik, berbagai masalah akan timbul dari waktu ke waktu. *Codebase* dari arsitektur monolitik akan terus membesar seiring bertambahnya fitur. Pertumbuhan *codebase* yang terus membesar membuat pengembang mengalami kesulitan untuk mengetahui letak perubahan yang akan dilakukan untuk membuat fitur baru dikarenakan artitektur monoltik membuat semua layanan berada pada satu *codebase* yang sama [3].

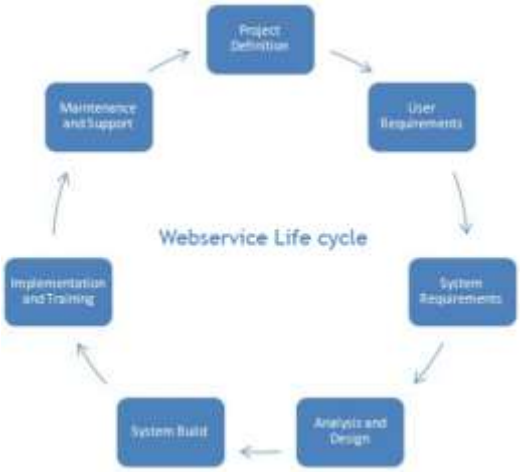
Saat ini *microservices* merupakan artitektur yang banyak dilirik oleh para *developer* dalam mengembangkan sebuah aplikasi karena banyak kelemahan pada arsitektur monolitik yang dapat diatasi oleh artitektur *microservices*. Artitektur *microservices* dipilih sebagai alternatif untuk artitektur monolitik karena lebih sederhana untuk di *scalable* dan lebih fleksibel [4].

Artitektur *microservice* merupakan perangkat lunak besar yang kompleks dan terdiri dari beberapa layanan. Arsitektur *microservices* ditujukan agar setiap services dapat fokus melakukan satu tugas atau fungsionalitas tertentu dengan baik. Satu *service* dengan *service* lain dapat saling berkomunikasi dan menghasilkan *business value* sehingga pengembang aplikasi tidak perlu merubah keseluruhan aplikasi ketika melakukan pengembangan dan hanya perlu menambah service baru ke dalam aplikasi tersebut [5]. Arsitektur *microservices* dapat dikembangkan dengan bahasa pemrograman apapun dan tiap *service* dapat saling terhubung menggunakan *Application Programming Interface* (API) seperti REST.

Penelitian ini bertujuan untuk merancang sebuah website online *course* Esploor dengan memanfaatkan arsitektur *microservices* menggunakan RESTful API untuk memberikan kemudahan layanan pembelajaran melalui platform website online, serta penggunaan artitektur *microservices* bertujuan untuk memudahkan pemeliharaan dan pengembangan sistem.

2. METODE PENELITIAN

Perancangan sistem dilakukan dengan metode Web *Service Implementation Methodology* (WSIM), *Web Service Implementation Methodology* (WSIM) merupakan skema pendekatan sistematis dalam membangun web *service* dengan memanfaatkan metodelogi pengembanga perangkat lunak *agile* dan memperluas metodelogi tersebut dengan menerapkan kegiatan spesifik web *service* dan peran yang sesuai dan produk kerja yang di hasilkan dalah sebuah proses [6]. Dengan metode ini *developer* dapat dengan mudah mengimplementasikan arsitektur *microservices* ke website online *course* Esploor. Tahapan pengembangan meliputi perancangan dan *analisys* sistem hingga *deployment*. Siklus pengembangan web *service* ditunjukan pada Gambar 1.

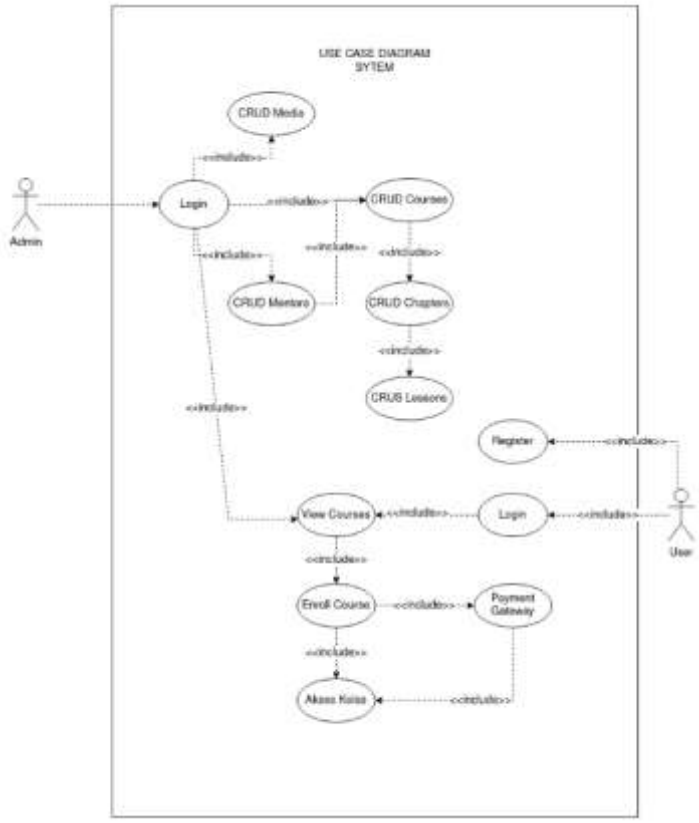


Gambar 1 *Web Service Implementation Lifecucle*

2.1 Perancangan Sistem

Analisa kebutuhan *system* bertujuan untuk menentukan komponen yang diperlukan selama proses pengembangan aplikasi online *course* Esploor. UML (*Unified Modeling Language*) Adalah sebuah bahasa yang menjadi standar dalam industri unutk visualisasi, merancang dan mendokumentasikan sistem. *Use Case*

Diagram adalah salah satu contoh diagram UML yang menggambarkan hubungan interaksi antara system dan actor. *Use Case* dapat mendeskripsikan tipe intraksi antar penggun dengan *system*. Rancangan *use case* diagram sistem yang ada pada aplikasi ini ditunjukan pada Gambar 2.



Gambar 2 ***Use Case Diagram***

2. 2 Perancangan Database

Aplikasi ini memiliki struktur database seperti dibawah ini

Gambar 3 ***Class Diagram***

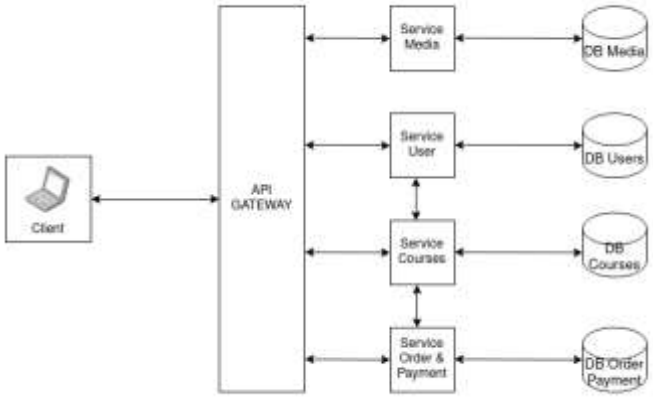
3. HASIL DAN PEMBAHASAN

Aplikasi website online *course* Esploor dibangun dengan mengimplementasikan artitektur *microservices* dan didukung dengan dua bahasa pemrograman pada sisi *BackEnd* nya yaitu JavaScript dan PHP. Dengan menggunakan *microservices*, beberapa modul yang beroperasi dengan

bahasa permrograman yang berbeda dapat terkoneksi dan beroperasi sesuai dengan fungsi kerja masing-masing *service* karena *microservices* tidak dibatasi pada satu penggunaan bahasa pemrograman saja [7]. Digunakan juga MySQL sebagai *Database Management System* (DBMS) untuk menyimpan data. Teknologi lain yang digunakan ialah ReactJs yang digunakan untuk membangun tampilan antarmuka pengguna. Menurut Sanchit Aggarwal, ReactJs merupakan pustaka JavaScript untuk membangun komponen antarmuka pengguna yang dapat digunakan kembali atau *reusable* [8].

3.1 Arsitektur Microservices

Pembuatan desain arsitektur *microservices* website online *course* Esploor berguna untuk memberi gambaran terhadap layanan dan alur komunikasi data antar layanan yang akan dibuat. Gambaran arsitektur yang dibuat sebagai berikut dimana client akan melakukan *request* ke API Gateway terlebih dahulu kemudian dari API Gateway akan meneruskan request yang di terima ke *service* yang di inginkan. Service yang menerima request akan memproses *request* yang masuk dan mengolahnya ke *database*. Setelah proses selesai, *service* akan megirimkan respon berupa data ke API Gateway untuk di teruskan lagi kepada client. Arsitekrut *microservices* ditunjukan pada Gambar 4.



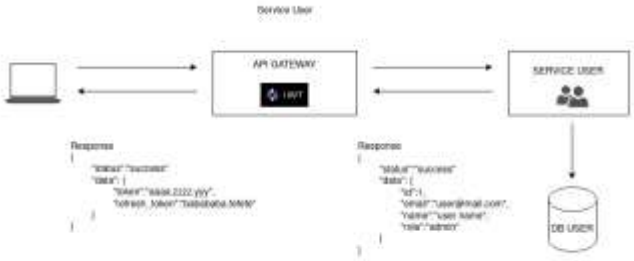
Gambar 3 Rancangan Arsitektur *Microservices*

3.1.1 Flow Service

Pada arsitektur *microservices*, layanan mikro yang digunakan untuk membangun website online *course* Esploor ini menerapkan konsep REST API sebagai cara komunikasi data antar *service* nya. Pada penelitian ini, penulis menggunakan empat *service* yang terpisah, yaitu *service user* yang berguna untuk menyimpan data, kemudian *service* media sebagai media data dan *storage* seperti gambar, yang ketiga *service course* yang berfungsi untuk menyimpan data dari kursus yang tersedia dan yang terakhir ialah *sevice order payment* yang berfungsi untuk mengatur layanan *payment gateway*. Pada dasarnya *Payment Gateway* berfungsi sebagai saluran yang ternkripsi yang secara aman mengirimkan detail transaksi dari pembeli ke bank untuk di setujui [9]. Midtrans merupakan salah satu *Payment Gateway* yang menyediakan kebutuhan bagi para pebisnis online dengan memberikan pelayanan dengan berbagai metode pembayaran, card payment, bank transfer, direct debit, *e-wallet* dan *over the counter* merupakan metode pembayaran yang disediakan oleh Midtrans [10].

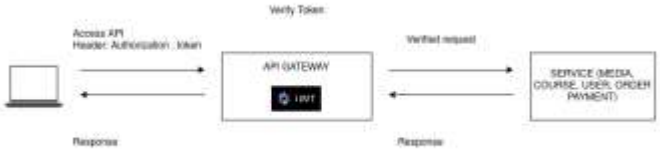
a. Service User

Pada service user akan terdapat flow *authentication* seperti berikut. Client mengakses API Gateway untuk proses authentication, kemudian *service* user menerima *request* dari API Gateway berupa *authentication*. Kemudian *service user* akan melakukan pengecekan ke *database* apakah *username* dan *password* yang dikirimkan oleh *user* valid atau tidak. Setelah data di temukan di *database service user*, *service user* akan mengirimkan data *user* yang telah di temukan ke API Gateway. Kemudian data yang diterima oleh API Gateway akan di proses oleh JWT *(JavaScript Web Token)* untuk dibuat menjadi token terenkripsi dan token tersebut akan di kirimkan ke *client* sehingga client bisa menggunakan token tersebut untuk mengakses API yang di telah proteksi oleh JWT. Alur *service user* ditunjukan pada Gambar 4



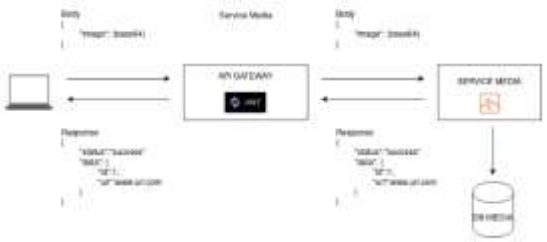
Gambar 4 *Flow Service User*

Kemudian ada *flow verify* token dimana token yang dimiliki *user* akan diverifikasi keabsahannya. Alurnya ialah, *client* akan mengakses API Gateway yang di proteksi oleh JWT dan *client* mengirimkan token pada *header* dengan *key Authorization*, kemudian API Gateway akan melakukan pengecekan. Apabila token invalid maka API Gateway memberikan response token tidak valid kepada *user*, dan ketika token tersebut valid maka API Gateway akan meneruskan permintaan dari *user* ke *service* yang di tuju. Setelah service memproses permintaan dari *user*, *service* akan memberikan respon berupa data dan responnya akan diteruskan dari API Gateway kepada *user*. Alur *verify* token ditunjukan pada Gambar 5.



Gambar 5 *Flow Verify Token*

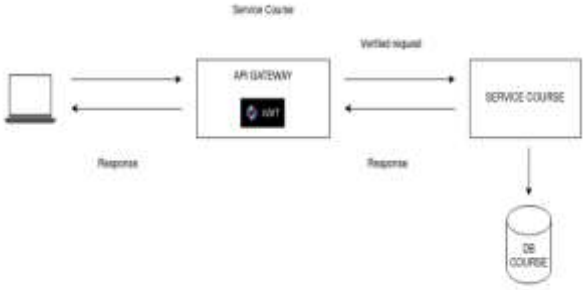
b. *Service Media*
Alur kerja pada *service* media ialah, client mengirim data kepada API Gateway berupa gambar yang telah di lakukan pengkodean dengan algoritma *base64*, Algoritma *base64* merupakan teknik konversi pengkodean *radix-64*, teknik ini merupakan cara untuk merubah input numeric menjadi sebuah karakter [11]. Kemudian dari API Gateway akan melakukan pengecekan token dengan JWT, apabila token valid maka API Gateway meneruskan permintaan dari *client* ke *service* media. Kemudian service media akan memproses data yang diterima dan disimpan di database. Setelah data berhasil disimpan, *service* media akan mengirimkan sebuah respon dengan status *success* dan mengirim data berupa id dan *url* dari gambar yang telah disimpan kepada API Gateway dan API Gateway akan meneruskan respon tersebut kepada *client*. Alur *service* media ditunjukan pada Gambar 6.



Gambar 6 *Flow Service Media*

c. *Service Course*
Alur kerja pada *service course* ialah, *client* melakukan *request* ke API Gateway, kemudian API Gateway melakukan pengecekan token, jika token tidak valid, API Gateway akan memberikan respon token tidak valid kepada *user* dan apabila token sudah valid, API Gateway akan meneruskan *request* ke *service course*. Kemudian *service course* memproses *request* yang masuk dan mengolahnya di *database*. Setelah data di olah,

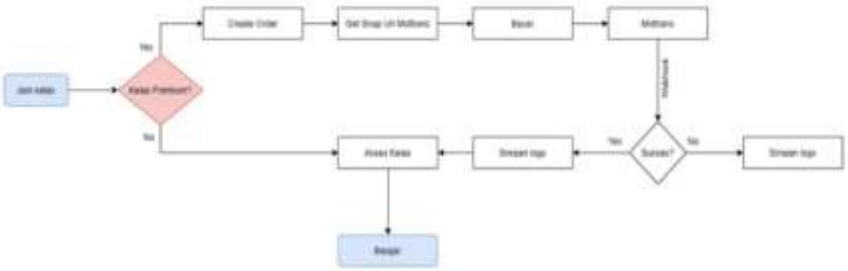
service course akan memberikan respon ke API Gateway dan diteruskan lagi ke *client*. Alur *service course* ditunjukan pada Gambar 7.



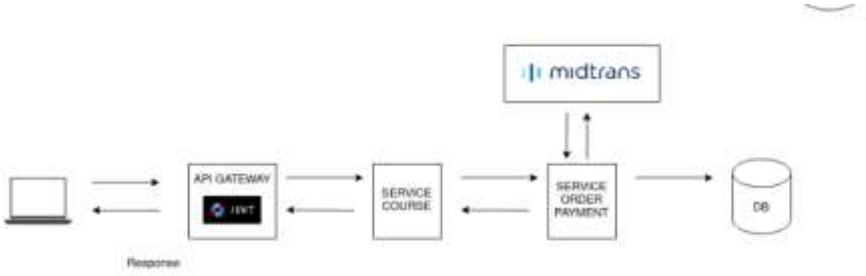
Gambar 7 *Flow Service Course*

d. *Service Order Payment*

Berikut merupakan *flow service order payment.* Pertama *client* yang ingin *join* kelas pembelajaran melakukan *request* ke website online *course* Esploor. Kemudian website akan melakukan pengecekan apakah kelas yang dipilih merupakan kelas *premium* atau tidak. Apabila *user* memilih kelas *free*, *client* akan langsung medapatkan akses kelas tersebut dan ketika kelas yang dipilih merupakan kelas *premium, service order payment* akan melakukan *create order*, kemudian *service* akan mengambil *snap url* dari Midtrans dan *client* akan diarahkan untuk melakukan pembayaran melalui *payment gateway* Midtrans. Apabila *client* telah selesai melakukan pembayaran, *client* akan mendapatkan akses kelas yang diinginkan. Alur *service order payment* ditunjukan pada Gambar 8 dan Gambar 9.



Gambar 8 *Flow Client Order*



Gambar 9 *Flow Service Order Payment*

3. 2 Implementas

Berikut merupakan tampilan antarmuka yang dapat diakses oleh *client*. Halaman antarmuka dibagun menggunakan teknologi ReactJs. ReactJs merupakan *open-source library* JavaScript yang deklaratif, efisien dan fleksibel untuk membangun antarmuka pengguna. ReactJs memungkinkan untuk membuat user interface yang kompleks dengan set kode kecil yang terisolasi yang sering disebut sebagai sebuah "komponen". ReactJs biasanya digunakan untuk menangani pembuatan tampilan pada suatu aplikasi berbasis website [12].

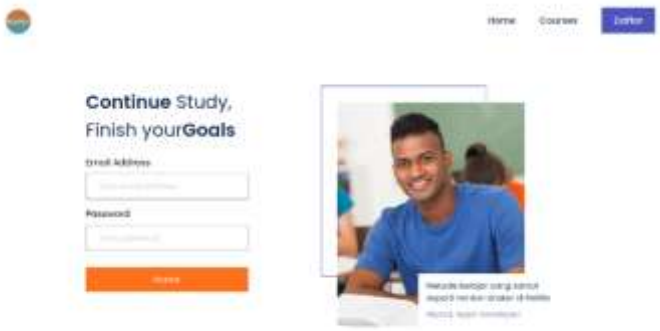
Halaman utama atau home page menampilkan informasi yang pertama kali dilihat oleh *client* ketika mengakses website online couse Esploor. Tampilan halaman utama ditunjukan pada Gambar 10.

Gambar 10 Halaman Utama

Halaman ini berisi list dari kelas yang tersedia di website online *course* Esploor. Halaman list kelas ditunjukan pada Gambar 11.

Gambar 11 Halaman List Kelas

Sebelum bergabung dengan kelas pembelajaran yang di inginkan, *client* diharuskan untuk melakukan proses *authentication* terlebih dahulu. Halaman *login* ditunjukan pada Gambar 12.



Gambar 12 Halaman Login

Apabila *client* belum memiliki *account*, client bias melakukan pendaftaran terlebih dahulu pada halaman *register*. Halaman register ditunkukan pada Gambar 13.

Gambar 13 Halaman Register

Setelah *login / register*, *client* akan diarahkan ke *member area*. Ada beberapa menu yang tersedia di halaman *member area* seperti menu *My Class* yang berfungsi untuk menampilkan daftar kelas yang telah dimilihin oleh *client*. Kemudian ada menu *library* yang mengarahkan *client* ke tampilan daftar list kelas yang tersedia pada website online *course* Esoloor. Selanjutnya ada halaman *Transaction* yang berisi catatan transaksi yang telah dilakukan oleh *client*. Kemudian ada halaman *setting* dimana client bisa mengubah profile yang dimiliki dan yang terakhir ada menu *logout* untuk keluar dari proses *authentication*. Halaman member ditunjukan pada gambar 14.

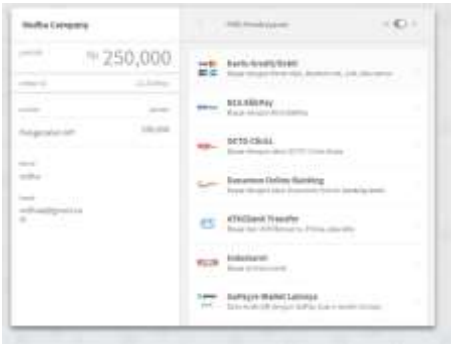
Gambar 14 Halaman Member

Client bisa melihat detail dari kelas yang diinginkan. Kemudian *client* bisa melakukan *enroll* kelas untuk bergabung. Apabila client memilih kelas gratis maka client bisa langsung bergabung kedalam kelas, akan tetapi apabila *client* memilih kelas *premium* maka *client* akan diarahkan ke halaman transaksi terlebih dahulu dan melakukan pembayaran sebelum bisa begabung kedalam kelas. Halaman detail kelas ditunjukan pada Gambar 15.

Gambar 15 Halaman Detail Kelas

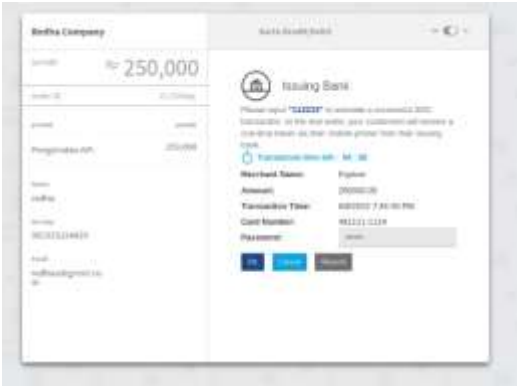
Ketika *client* memilih untuk bergabung dengan kelas *premium*, *client* akan diminta untuk melakukan pembayaran kelas melalui *payment gateway*. Pembayaran bisa dilakukan dengan berbagai metode seperti pembayaran melalui kartu kredit, pembayaran melalui transfer antar bank, pembayaran

melalui *e-wallet*, pembayaran melalui *minimarket* dan masih banyak lagi. Setelah *client* memilih metode pembayaran, *client* akan diberikan tampilan transaksi dan *client* diminta untuk segera melakukan pembayaran. Setelah pembayaran berhasil, *client* akan di arahkan kembali ke *member area* dan *clinet* sudah bisa mengakses kelas yang sebelumnya dipilih. Proses pembayaran ditampilkan pada Gambar 16 hingga 19.

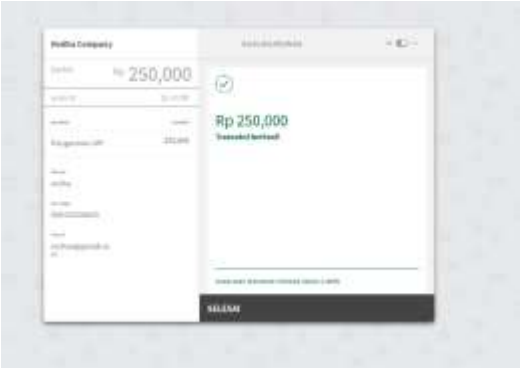


Gambar 16 Halaman Informasi Pembayaran

Gambar 17 Halaman Pembayaran



Gambar 18 Halaman Detail Pembayaran



Gambar 19 Halaman Sukses Pembayaran

Setelah *client* melakukan pembayaran, *client* akan mendapatkan akses kelas yang di inginkan dan bisa mengikuti proses pembelajaran. Halaman akses kelas ditampilkan pada Gambar 20 dan 21.

Gambar 20 Halaman Sukses Bergabung Kelas

Gambar 21 Halaman Akses Kelas

4. KESIMPULAN

Dari hasil penelitian ini, didapatkan kesimpulan bahwa arsitektur *microservices* memberikan kemudahan pada proses pengembangan untuk masa mendatang dikarenakan artitektur *microservices* tidak dibatasi hanya pada satu *codebase* saja melainkan bisa memisahkan program yang besar menjadi beberapa *service* terpisah. Ketika pengembang ingin membuat fitur baru, pengembang hanya perlu membuat *service* baru sesuai dengan fitur yang dibutuhkan. Arsitektur *microservices* juga tidak dibatasi oleh satu bahasa pemrograman saja, pengembang bisa menggunakan lebih dari satu bahasa pemrograman dan melakukan komunikasi antar *services* menggunakan *Application Programming Interfaces* (API). Pada penelitian ini, penulis menggunakan dua bahasa pemrograman dalam mengembangkan *system* online *course* Esploor yaitu bahasa pemrograman PHP dan Javascript.

Dari penelitian ini masih banyak kekurangan yang ada pada *system* yang telah dibuat, pengembangan aplikasi selanjutnya diharapkan dapat dibuat juga dengan berbasis aplikasi mobile baik android maupun ios serta tersedianya fitur *Progressive Web Apps* (PWA) sehingga aplikasi yang dibuat dapat lebih praktis untuk di akses oleh pengguna melalui perangkat selular. Diharapkan pula terdapat fitur untuk forum diskusi sehingga pengguna bisa saling berdiskusi mengenai materi yang ada di aplikasi ini.

DAFTAR PUSTAKA

[1] S. Suryadi, 2019, "Peranan Perkembangan Teknologi Informasi Dan Komunikasi Dalam Kegiatan Pembelajaran Dan Perkembangan Dunia Pendidikan," *J. Inform.*, vol. 3, no. 3, pp. 9–19, doi: 10.36987/informatika.v3i3.219.

[2] D. Purwanto *et al.*, 2021, "Aplikasi Kursus Online Berbasis Web Service Menggunakan Arsitektur Microservices," pp. 978–979.

[3] Yuri Chandra Tri Putra, 2020, Thomas Adi Purnomo Sidi, and Joseph Eric Samodra, "Implementasi Arsitektur Microservice pada Aplikasi Web Pengajaran Agama Islam Home Pesantren," *J. Inform. Atma Jogja*, vol. 1, no. 1, pp. 88–97.

[4] F. Dahri, A. M. El Hanafi, D. Handoko, and N. Wulan, 2022, "Implementation of Microservices Architecture in Learning Management System E-Course Using Web Service Method," *Sinkron*, vol. 7, no. 1, pp. 76–82, doi: 10.33395/sinkron.v7i1.11229.

[5] I. Y. B. Purnama, Heri, 2010, "APLIKASI PENGELOLAAN SKRIPSI DI STMIK AKAKOM YOGYAKARTA MENGGUNAKAN ARSITEKTUR MICROSERVICE DENGAN Node.js," pp. 1–28, [Online]. Available: http://perpus.akakom.ac.id/

[6] E. Lee, P. TAN, Y. CHENG, and X. XU, "Web Service Implementation Methodology," *Organ. …*, no. September, pp. 1–35, 2005, [Online]. Available: https://www.oasis-open.org/committees/download.php/9516/FWSI-IMSC-Document-01.doc

[7] S. Dharma Handayani and U. Uminingsih, 2020, "Pengorganisasian Kerja Sistem Parkir Menggunakan Arsitektur Microservice," *J. Teknol.*, vol. 13, no. 1, pp. 27–35, [Online]. Available: https://ejournal.akprind.ac.id/index.php/jurtek/article/view/2891

[8] S. Aggarwal, 2018, "Modern Web-Development Using ReactJS | Document Object Model | Model–View–Controller," *Int. J. Recent Res. Asp.*, vol. 5, no. 1, pp. 133–137, [Online]. Available: https://www.scribd.com/document/379709841/Modern-Web-Development-Using-ReactJS#download

[9] M. Febrianto, 2020, "Penerapan Payment Gateway Dan Tracking Barang Pada E-Commerce Toko Dazzle Berbasis Website(API)," *Tugas Akhir thesis, Univ. Technol. Yogyakarta.*.

[10] T. R. Saputro and J. Sutopo, 2019, "Penerapan Payment Gateway Sebagai Sistem Verifikasi Pembayaran Pada Website Pemesanan Paket Wisata," *Tugas Akhir thesis, Univ. Technol. Yogyakarta*.

[11] M. Hayaty and M. D. Putra, 2021,"Enkripsi Dan Dekripsi Gambar Dengan Menggunakan Perpaduan Algoritma Base64 Dan Rc4," *Core It*, vol. 5, pp. 1–6, 2018, [Online]. Available: https://ojs.amikom.ac.id/index.php/semnasteknomedia/article/view/1986

[12] Nasution and L. Iswari, "Penerapan React JS Pada Pengembangan FrontEnd Aplikasi Startup Ubaform," *Automata*, vol. 2, no. 2.