P-ISSN: 2548-1932
e-ISSN: 2549-7758

IMPLEMENTASI AUTENTIKASI DARI SISI *BACKEND* PADA ARSITEKTUR *MICROSERVICES* MENGGUNAKAN *EXPRESS JS*

Regita Lisgiani[1], Sigit Nurmajid[2]

[1,2]Program Studi Teknik Informatika, Universitas Sangga Buana
e-mail korespondensi: [1]regitalisgianidrajat@gmail.com, [2]sigitnurmajid32@gmail.com

ABSTRACT

Currently, the SuperApps application is being widely used by the community, one example is Gojek. Superapps is a one-stop solution that contains customer needs in an application. Many people choose applications that already have all the features that can meet their daily needs. Superapps have high complexity, starting from security, data, request handling and other system requests. To handle large amounts of data and processes, a Microservice Architecture is required. Microservices allow each feature in the application to do its own development. With Microservices, all code is broken down into independent services that run as separate processes. In the microservice architecture, authentication is very important because it contains user data and security. The authentication system must be made as good as possible so that it can be decentralized in various services. In this journal, the author will analyze, design, and create patterns and implement good code to design good authentication in microservice architectures.

Keywords: SuperApp; security; microservice; authentification

ABSTRAK

Saat ini, aplikasi SuperApps banyak digunakan di masyarakat. Contohnya adalah Gojek. Superapps adalah solusi satu atap yang mencakup kebutuhan pelanggan dalam satu aplikasi. Banyak orang memilih aplikasi yang sudah memiliki semua fitur yang dapat memenuhi kebutuhan sehari-hari mereka. Aplikasi superapps memiliki sistem yang sangat kompleks, mulai dari keamanan, data, pemrosesan permintaan, dan permintaan sistem lainnya. Pemrosesan data dan proses dalam jumlah besar memerlukan arsitektur Microservice. Microservice memungkinkan pengembang melakukan mengembangkan sendiri untuk setiap fitur aplikasi Anda. Microservice membagi semua kode menjadi layanan independen yang berjalan sebagai proses terpisah. Di Dalam arsitektur microservice autentifikasi menjadi salah satu yang sangat penting karena didalamnya mengandung user data yang merupakan bagian penting dari aplikasi.. Sistem autentifikasi harus dibuat sebaik mungkin agar dapat ter desentralisasi di berbagai servicenya dan dapat menjadi jembatan authentifikasi dan security sistem. Dalam jurnal ini, penulis akan menganalisis, merancang, dan membuat pattern dan pengimplementasian kode yang baik untuk merancang autentifikasi yang baik dalam arsitektur microservice.

Kata Kunci: SuperApps; security; microservice; autentikasi

PENDAHULUAN

SuperApps merupakan platform aplikasi yang menyediakan banyak layanan yang dibutuhkan masyarakat berupa fitur fitur aplikasi yang terintegrasi dalam satu aplikasi [1]. Saat ini, customer atau pengguna lebih memilih aplikasi dengan banyak layanan yang dapat memenuhi kebutuhan mereka. Dikarenakan kebutuhan fitur yang banyak sistem superApps sangat kompleks, mulai dari keamanan, data, pemrosesan permintaan, dan permintaan sistem lainnya [2]. Arsitektur yang baik diperlukan untuk menangani sejumlah besar data dan proses. Arsitektur pengembangan aplikasi memiliki dua arsitektur yang sangat populer: monolitik dan Microservice [3]. Microservice adalah pendekatan SDLC yang memecah sistem yang

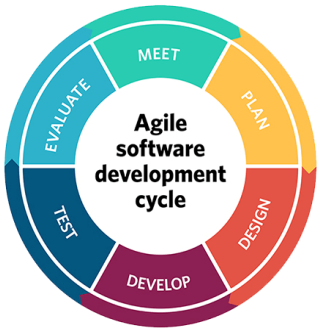
P-ISSN: 2548-1932
e-ISSN: 2549-7758

besar dan kompleks menjadi modul atau service fungsional yang lebih kecil. Microservice membagi semua kode menjadi service independen yang berjalan sebagai proses terpisah. Dalam Microservice, setiap service memiliki skema, kode, dan infrastruktur yang berbeda, tergantung pada kebutuhannya. Karena banyaknya data dan service yang ada dalam arsitektur ini, Microservice sangat rentan terhadap serangan security khususnya pada sistem otentikasi. Oleh karena itu, membangun sistem Microservice memerlukan desain arsitektur otentikasi yang sesuai.

Dalam arsitektur layanan mikro, aplikasi client biasanya perlu menggunakan fungsionalitas dan data dari berbagai service. client harus menangani beberapa pengambilan data ke service service lain [10]. Client memiliki beban yang berat karena menjadi pusat untuk pengambilan dan pengolahan data dari berbagai service. Usability system tentunya terbatas karena semua sistem terdistribusi pusat atau bergantung pada client. Tingkat keamanan pada service menjadi rentan karena banyaknya pertukaran data antar service.

METODE

Tujuan dari metode penelitian pada sistem microservice ini adalah untuk menganalisis dan membuat sistem authentifikasi dan pengambilan data yang baik untuk arsitektur microservice dari segi keamanan dan distribusi data. Metode penelitian yang diterapkan untuk membuat sistem yang akan dibuat adalah menggunakan pendekatan SDLC (Software Development Lifecycle). Software Development Lifecycle atau SDLC adalah proses yang digunakan untuk merancang, mengembangkan, dan menguji perangkat lunak berkualitas tinggi. Tujuan SDLC adalah untuk menyediakan diagram alur terstruktur yang memungkinkan organisasi membuat perangkat lunak berkualitas tinggi yang dapat diselesaikan lebih cepat dan dengan biaya lebih rendah sambil memenuhi dan melampaui harapan pelanggan. Model SDLC yang dipilih dalam metode survei ini adalah model agile [4]. Agile adalah kumpulan teknik pengembangan perangkat lunak yang dilakukan secara bertahap dan berulang (berulang) [5]. Model ini merupakan pengembangan berulang dan



Gambar 1: Siklus pengembangan perangkat lunak

berkelanjutan yang dapat diterapkan pada microservice development yang dapat melakukan perubahan sesuai kebutuhan.

1. Meet

   Pada tahapan ini pengembang akan bertemu untuk menentukan sistem yang akan dibangun beserta requirement yang dibutuhkan dalam sistem. Pada tahapan ini pengembang akan menganalisa

konsep dan sistem apa yang digunakan. Hasil dari fase ini biasanya menghasilkan beberapa dokumentasi yang dibutuhkan selama proses kerja, seperti *flow* aplikasi dan *design* atau UI aplikasi (*Mockup*). Analisis terperinci ini akan membantu untuk memutuskan apakah suatu proyek layak atau tidak sebelum memulai pekerjaan.

2. Plan

   Pada tahapan ini pengembang akan mulai merancang timeline dan tahapan kerja yang akan dilakukan. Pada langkah ini tim pengembang merancang apa saja yang dibutuhkan dalam suatu perangkat lunak yang hendak dibuat. Di dalam tahapan ini terjadi pemilihan arsitektur software di dalam microservice yang akan diterapkan pada sistem.

3. Design

   Pada tahap ini seorang pengembang memulai membuat design untuk sistem yang akan dibuat.

4. Develop

   Setelah dokumentasi design keluar maka, pengembang harus mengimplementasikan hasil desain yang sudah dibuat ke dalam coding.

5. Testing

   Apabila semua tahapan sebelum testing sudah selesai, ini waktu nya untuk pengembang melakukan testing terhadap code yang sudah dibuat. Dan tentunya hasil testing ini diharapkan sesuai dengan syarat - syarat dari desain sistem yang sudah ada.

6. Evaluate

   Akhirnya pada tahap ini customer dan pengembang akan bertemu untuk mengevaluasi dari sistem yang sudah dibuat oleh pengembang

HASIL DAN PEMBAHASAN

Dari hasil penelitian dengan menggunakan model development framework agile diperlukan banyak analisa dan percobaan. Ditemukan bahwa sistem autentifikasi membutuhkan software architecture yang memungkinkan untuk menutup sistem service service dibelakangnya. Software Architecture melingkupi hubungan antara elemen yang ada dalam sistem dan elemen yang berada diluar system (yang dilihat user). Terdapat pattern populer yang ada pada microservices yaitu pattern direct to client dan API Gateway.

Pada pattern direct client to microservices, client akan mengakses langsung ke service service yang mereka butuhkan. Direct client microservice sangat baik untuk microservice dengan skala kecil, khususnya jika client merupakan server side based application [7].

Pada arsitektur *Microservices*, client apps biasanya memerlukan mengkonsumsi functionality dari lebih satu service [9]. Client harus men-handle untuk melakukan multiple call ke microservice. Usability system tentunya terbatas karena semua sistem terdistribusi pusat atau bergantung pada client. Perubahan apapun pada system sangat mempengaruhi client, termasuk error, clustering, server shutdown atau overwork.

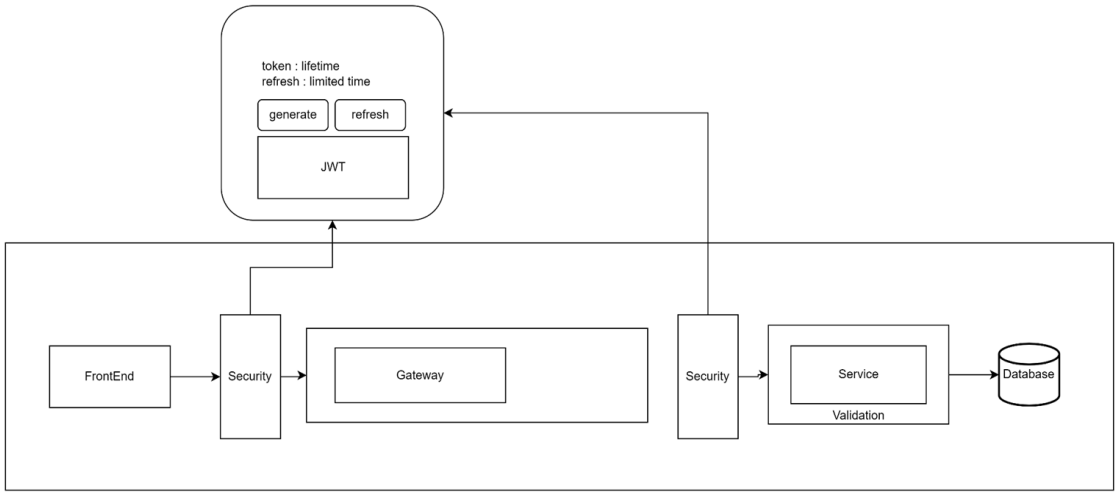
P-ISSN: 2548-1932
e-ISSN: 2549-7758

API Gateway adalah suatu service yang dibuat khusus dan dijadikan sebagai entry point dari dunia luar untuk masuk ke dalam sistem atau architecture kita. Gateway akan berada di antara client dan service service dalam architecture dan berfungsi sebagai reverse [proxy untuk melakukan request ke service-service yang dibutuhkan.

Tools

Tools software atau pemrograman yang digunakan dalam pembuatan autentifikasi system adalah express js dengan bantuan JWT sebagai enkripsi autentikasi berbentuk token [6]. Express.js adalah framework web app untuk Node.js yang ditulis dengan bahasa pemrograman JavaScript. Dimana framework ini digunakan untuk membangun aplikasi dari sisi back end secara efektif dan optimal [8].

Analisa Tahapan Integrasi Sistem

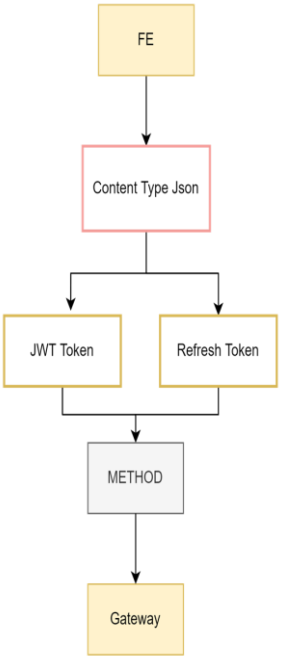


Gambar 2: Flow arsitektur

Flow yang akan dipakai didalam sistem yang akan dibuat adalah seperti gambaran di atas dengan tahapan tahapan yang secara sederhana adalah sebagai berikut

1. Client akan melakukan request data terhadap Gateway
2. Client akan mengirimkan authorization key berupa JWT untuk security dan validasi
3. Gateway akan melakukan validasi security dan input atau request yang masuk dengan cara men decode JWT yang dikirim client
4. Jika Autentikasi berhasil gateway akan meneruskan request ke service user

P-ISSN: 2548-1932
e-ISSN: 2549-7758



Gambar 3: Flow Chart

5. Service akan memvalidasi request dan memproses data lalu mengembalikannya ke gateway.
6. Proses selanjutnya adalah gateway akan melakukan modeling dari user data yang dikirim oleh service user Metode authentikasi yang digunakan adalah dengan menggunakan metode JWT dan refresh token.

JWT akan berlaku sebagai token autentikasi berbentuk random code yang memiliki waktu expired dan berisikan data user yang di enkripsi. Refresh token akan berlaku sebagai token refresh untuk melakukan request ulang terhadap jwt autentifikasi yang expired sehingga client tidak perlu melakukan logout dan login ulang untuk mendapatkan key autentifikasi JWT.

DAFTAR PUSTAKA

[1] *Kupas Tuntas Super App, Aplikasi Serbabisa yang Buat Hidupmu Efisien*. (2020, December 10). Glints Blog. https://glints.com/id/lowongan/super-app/#.Yj2nH01BxPY

[2] *Kenali super app dan beberapa contohnya*. (n.d.). EKRUT. Retrieved May 27, 2022, from https://www.ekrut.com/media/super-app-adalah

[3] Alchuluq, L. M., & Nurzaman, F. "Analisis pada Arsitektur Microservice untuk Layanan Bisnis Toko Online." TEKINFO, 2021, 22.2:61-68.

[4] *Ciptakan Tim yang Tanggap dengan Segala Perubahan Menggunakan Agile Project Management!*. (2021, March 7). Pemimpin.ID. https://pemimpin.id/ciptakan-tim-yang-tanggap-dengan-segala-perubahan-menggunakan-agile-project-management/

[5] *Agile planning: A step-by-step guide and template*. (2018, April 22). Monday.Com Blog. https://monday.com/blog/project-management/agile-planning/

[6] auth0.com. (n.d.). *JWT.IO*. Auth0. Retrieved May 27, 2022, from https://jwt.io/

[7] *What are microservices?* (n.d.). Chris Richardson. Retrieved May 27, 2022, from https://microservices.io/

[8] *Express 5.x*. (n.d.). API Reference. Retrieved May 27, 2022, from https://expressjs.com/en/5x/api.html

[9] Julio, E., & Pakereng, M. A. I. (2021). Implementasi API payment gateway Menggunakan Arsitektur Microservice. *Jurnal Informatika*, *8*(2), 123–130. https://doi.org/10.31294/ji.v8i2.10590

[10] Elsen, R. (2022). Perancangan Arsitektur Microservice untuk Portal Berita Daring. *Jurnal Algoritma*, *18*(2), 352–357. https://doi.org/10.33364/algoritma/v.18-2.875