

IMPLEMENTASI ARSITEKTUR *MICROSERVICES* PADA RANCANG BANGUN APLIKASI *MARKETPLACE* BERBASIS WEB

Alessandro Sinambela¹, Ernawati², Funny Farady Coastera³

^{1,2,3}Program Studi Informatika, Fakultas Teknik, Universitas Bengkulu
Jl. WR. Supratman Kandang Limun Bengkulu 38371A INDONESIA
(telp: 0736-341022; fax: 0736-341022)

¹sandroales405@gmail.com,

²ernawati@unib.ac.id,

³ffaradyc@unib.ac.id

Abstrak Kemajuan teknologi informasi saat ini sangat membantu kegiatan manusia dalam melakukan bisnis *online* melalui *marketplace*. Untuk memenuhi kebutuhan konsumen dibutuhkan peranan bidang teknologi dalam proses penjualan, pembelian, hingga efisiensi dan efektifitas. Pasar Tradisional Modern (PTM) merupakan pasar tradisional modern yang berada di kota Bengkulu yang menggunakan kaidah pasar tradisional yaitu tempat bertemunya penjual dan pembeli, hal tersebut menimbulkan masalah yaitu membutuhkan banyak waktu untuk berbelanja kebutuhan sehari-hari. Aplikasi yang akan dibangun untuk mempermudah ibu rumah tangga dalam berbelanja bahan-bahan dapur. Aplikasi ini nantinya akan dibuat berbasis *marketplace* dan menggunakan arsitektur *microservice*. *Microservice* merupakan arsitektur yang membagi sistem menjadi servis-servis kecil. Penggunaan arsitektur tersebut dapat mengatasi masalah pada arsitektur *monolith* yaitu penggunaan *resources* yang besar pada *server*, sulit melakukan pengembangan pada sistem. Untuk proses pengolahan data pada tiap-tiap servis menggunakan *application programming interface* (*Api*) dan menggunakan *javascript web token* (*Jwt*) sebagai otorisasi untuk menggunakan *end point* yang disediakan. Untuk mendukung arsitektur tersebut dapat menggunakan *docker* dalam proses manajemen servis. Penelitian ini terdiri dari lima servis yaitu *api-gateway*, *produk-servis*, *keranjang-servis*, *order-servis*, dan *pengguna-servis* yang berjalan didalam *docker container*. Berdasarkan pengujian yang telah dilakukan, penelitian ini memiliki tingkat keberhasilan 100% pada pengujian *end point* dalam mengolah data pada servis-servis yang ada didalam *docker container*, dan pengujian sistem secara fungsional telah dilakukan menggunakan metode pengujian *black box* dan menghasilkan nilai 100% dengan bermacam skenario.

Kata Kunci: *microservices, api, jwt, marketplace, docker.*

Abstract: Advances in information technology are currently very helpful for human activities in doing online business through marketplaces. To meet consumer needs, technology is needed in the sales, purchasing, to efficiency, and effectiveness processes. Modern Traditional Market (PTM) is a modern traditional market located in the city of Bengkulu which uses traditional market principles, namely a meeting place for sellers and buyers, this creates a problem, which requires a lot of time for daily needs. Applications that will be built to make it easier for housewives in kitchen ingredients. This application will later be made based on the market and using microservices. Microservices is an architecture that divides the system into small services. The use of this architecture can solve problems in monolith architectures, namely the use of large resources on the server, it is difficult to develop on the system. For processing data on each service using a programming interface application (API) and using the javascript web token (Jwt) as authorization to use the provided endpoints. To support this architecture, you can use Docker in the service management process. This research consists of five services, namely fire-gateway, product-service, basket-service, service-order, and user-service which are run in a docker container. Based on what has been done, this study has a 100% increase rate at the endpoint of testing in processing data on services in the docker container, and functional system testing has been carried out using the black box testing method and produces 100% values with various scenarios.

Keywords: microservices, api, jwt, marketplace, docker.

I. PENDAHULUAN

Pada saat ini dengan adanya teknologi informasi merupakan bagian penting untuk membantu pekerjaan manusia. Misalnya saja dalam bidang bisnis, teknologi dan informasi menjadi hal yang paling penting untuk mendukung pengambilan keputusan. Bisnis *online* yang berupa *marketplace* semakin banyak di Indonesia, hal tersebut disebabkan oleh perkembangan teknologi dan adanya perubahan perilaku konsumen [1].

Untuk memenuhi kebutuhan konsumen tersebut dibutuhkan sebuah peranan dari bidang teknologi. Peran teknologi dalam memenuhi kebutuhan manusia yang semakin kompleks sangat dibutuhkan, dimulai proses penjualan, pembelian, hingga efisiensi dan efektifitas. Selain itu, teknologi juga berperan untuk meningkatkan persaingan dalam melakukan bisnis.

Sistem Informasi yang merupakan salah satu bidang ilmu dalam teknologi dan informasi ini digunakan untuk menampilkan informasi yang berguna untuk pengelola, pengambilan keputusan, dan menjalankan aktivitas-aktivitas yang dibutuhkan untuk memenuhi suatu kebutuhan, sedangkan sistem informasi bisnis merupakan kumpulan dari berbagai informasi yang saling berkaitan dengan satu dengan yang lainnya yang diperuntukkan untuk kebutuhan bisnis.

Seiring dengan perkembangan teknologi dan informasi, sistem informasi dan sistem bisnis sering diterapkan dalam kegiatan bisnis *marketplace*. Saat ini hampir semua bisnis sudah masuk ke dunia *marketplace*, dimulai dari produk tradisional hingga produk dari teknologi terbaru. Tidak bisa dipungkiri lagi, *marketplace* saat ini sudah merambah hampir ke seluruh area bisnis. Sehingga organisasi IT kini mulai berlomba untuk menyediakan aplikasi *marketplace* berbasis

layanan bagi masyarakat untuk memudahkan dalam memenuhi kebutuhannya [2].

Pasar Tradisional Modern (PTM) merupakan salah satu pasar tradisional modern yang berada di kota Bengkulu yang menggunakan kaidah pasar tradisional yaitu merupakan tempat bertemunya penjual dan pembeli, dari hal tersebut muncul masalah yaitu membutuhkan banyak waktu khususnya untuk berbelanja kebutuhan sehari-hari sehingga dibangunlah aplikasi yang diharapkan dapat mempermudah ibu rumah tangga khususnya dalam berbelanja kebutuhan sehari-hari. Aplikasi yang akan dibangun untuk mempermudah ibu rumah tangga dalam berbelanja bahan-bahan dapur. Aplikasi ini nantinya akan dibuat berbasis *marketplace* dan menggunakan arsitektur *microservice*.

Microservice merupakan layanan kecil yang terhubung antara satu dengan lainnya untuk membentuk aplikasi kompleks, servis-servis ini terdiri dari blok-blok kecil, terpisah, dan berfokus pada tugas-tugas ringan untuk mendukung kebutuhan sistem yang akan dibangun. Kelebihan arsitektur *microservice* yaitu memudahkan *developer* dalam proses *development* sistem, memungkinkan setiap servis bisa dikembangkan secara *independent*, tidak ada hambatan dalam menggunakan teknologi baru. Arsitektur ini digunakan untuk mengganti penggunaan arsitektur *monolitik* yang merupakan sebuah arsitektur yang semua komponennya menjadi satu kesatuan. Kekurangan arsitektur *monolitik* yaitu menggunakan banyak *resources*, sulit melakukan *maintenance*, dan jika terjadi *error* pada satu fungsi maka akan mempengaruhi seluruh aplikasi, hal ini dikarenakan karena menggunakan *single database* [3].

Oleh karena itu, penggunaan *microservices* akan menjadi alternatif yang baik untuk mengatasi penggunaan *resource* yang banyak yang ada pada *web service marketplace* yang masih menggunakan arsitektur monolitik. Studi kasus yang digunakan pada pasar tradisional modern lantai satu karena menjual produk-produk *fresh food*. Sehingga pada penelitian ini, penulis akan menggunakan arsitektur *microservice* pada *marketplace* dengan judul “Implementasi Arsitektur *Microservices* Pada Rancang Bangun Aplikasi *Marketplace* Berbasis Web”.

II. LANDASAN TEORI

A. Arsitektur *Microservice*

Microservice berarti membagi aplikasi menjadi layanan yang lebih kecil dan saling terhubung tidak seperti aplikasi monolitik. Setiap *microservice* merupakan aplikasi kecil yang memiliki arsitektur heksagonal sendiri yang terdiri dari logika beserta berbagai adaptasinya.

Arsitektur *microservice* merupakan alternatif arsitektur yang lebih terukur dan lebih fleksibel. Pada arsitektur *microservice*, sistem informasi dirancang untuk terdistribusi dan menyediakan layanan secara lebih fokus dan spesifik. Permasalahan besar akan dipecah menjadi beberapa solusi kecil yang disusun dalam satu *service*, dimana setiap *service* memiliki tanggung jawabnya sendiri. Dengan pendekatan ini, suatu sistem informasi akan terdiri dari beberapa *service* yang dapat dikelola dan didistribusikan secara *independent*, hal ini akan lebih memudahkan sistem untuk beradaptasi terhadap perubahan kebutuhan [4].

B. *Rest API*

REST merupakan gaya arsitektur dalam mendesain sebuah web service di mana desain

REST memiliki resource yang dapat diakses melalui sebuah alamat HTTP URL yang unique. REST juga memungkinkan klien dapat melakukan request melalui protokol HTTP dengan mudah menggunakan URI. Masing-masing alamat URL mengacu kepada kumpulan program yang akan dieksekusi dan akan mengembalikan pesan kepada pengirim perintah. REST mengirimkan perintah yang akan dikerjakan oleh server menggunakan metode-metode HTTP *request method* yang disebut *verb* [5].

Terdapat delapan HTTP *request method*, yaitu GET, POST, PUT, DELETE, OPTIONS, HEAD, TRACE, dan CONNECT. Dalam penggunaan API REST hanya menggunakan empat dari metode-metode tersebut, yaitu: GET, POST, PUT, dan DELETE [6].

- 200 OK

Perintah yang dikirim ke *server* benar dan berhasil dijalankan.

- 400 Bad Request

Perintah yang dikirim ke *server* berisi isian yang salah.

- 401 Unauthorized

Pengirim perintah mengirimkan kode kunci yang salah.

- 403 Forbidden

Pengirim perintah tidak memiliki hak akses ke dalam resource yang dituju.

- 404 Not Found

Resource yang dituju tidak ditemukan dalam *server*.

- 429 Too Many Requests

Pengirim perintah mengakses mencapai/melebihi dari limit yang telah ditentukan dari batas waktu tertentu.

- 500 Internal Server Error

Server atau potongan program dalam *resource* mengalami kesalahan.

C. API

Application programming interface (API) merupakan suatu dokumentasi yang terdiri dari *interface*, fungsi, kelas, struktur dan sebagainya untuk membangun sebuah perangkat lunak. Dengan adanya *API* ini, maka memudahkan *programmer* untuk “membongkar” suatu *software*, kemudian dapat dikembangkan atau diintegrasikan dengan perangkat lunak yang lain. *API* dapat dikatakan sebagai penghubung suatu aplikasi dengan aplikasi lainnya yang memungkinkan *programmer* menggunakan sistem *function*. Proses ini dikelola melalui sistem operasi. Keunggulan dari *API* ini adalah memungkinkan suatu aplikasi dengan aplikasi lainnya dapat saling berhubungan dan berinteraksi [7].

D. JSON

JSON (*JavaScript Object Nation*) adalah format pertukaran data ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh komputer. Format ini dibuat berdasarkan bagian dari bahasa pemrograman JavaScript, Standar ECMA-262 Edisi ke-3- Desember 1999. *JSON* merupakan format teks yang tidak bergantung pada bahasa pemrograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Python. Oleh karena sifat-sifat tersebut, menjadikan *JSON* ideal sebagai bahasa pertukaran data [7].

E. Marketplace

Marketplace didefinisikan sebagai sebuah sistem informasi antar-organisasi yang

memungkinkan pembeli dan pemasok berpartisipasi untuk bertukar informasi tentang harga dan produk yang ditawarkan. Perusahaan yang mengoperasikan sistem disebut sebagai perantara dimana pelaku pasar adalah pembeli dan penjual, pihak ketiga yang independen, atau konsorsium perusahaan-perusahaan. Secara umum online shop dan online marketplace tidaklah berbeda. Mereka adalah produk dari *e-commerce*. Tidak ada teori khusus yang membahas perbedaan dari keduanya karena secara umum mereka didefinisikan sama yaitu sebagai bagian dari *e-commerce* tempat terjadinya transaksi antara pembeli dan penjual secara online namun secara khusus perbedaannya pada skala orang yang terlibat dalam transaksi dan pihak yang terlibat dalam transaksi. *Online marketplace* adalah sebuah *online shop* dengan model bisnis *marketplace* concentrator dimana pemilik online shop adalah hanya sebagai fasilitator yang memusatkan berbagai macam informasi mengenai produk dan jasa dari berbagai penjual sehingga pembeli bisa membandingkan harga [8].

F. Docker

Docker adalah sebuah platform terbuka untuk siapapun yang bertujuan menggunakan sebuah platform untuk membangun, mendistribusikan dan menjalankan aplikasi dimanapun seperti laptop, data center, virtualmachine ataupun cloud. Docker merupakan *open source software* di bawah Lisensi Apache Versi 2.0 yang bisa dipergunakan secara gratis. Saat ini Docker hanya bisa berjalan pada Linux, tetapi bisa menggunakan *virtual machine* pada operating system windows, atau menggunakan Boo2docker.

Docker menggunakan arsitektur client-server. Docker client menghubungi Docker daemon, yang

melakukan pekerjaan berat, menjalankan, dan mendistribusikan Docker container anda. Kedua Docker client dan daemon dapat berjalan pada sistem yang sama. Docker client dan daemon berkomunikasi via sockets atau lewat API yang disediakan Docker [9].

G. Node Js

Node.js adalah sistem perangkat lunak yang didesain untuk pengembangan aplikasi web. Aplikasi ini ditulis dalam bahasa JavaScript, menggunakan basis event dan asynchronous I/O. Tidak seperti kebanyakan bahasa JavaScript yang dijalankan pada peramban, Node.js dieksekusi sebagai aplikasi server. Aplikasi ini terdiri dari V8 JavaScript Engine buatan Google dan beberapa modul bawaan yang terintegrasi. Modul-modul yang digunakan dalam implementasi klien SIP ini antara lain Sip.js sebagai implementasi protokol SIP pada Node.js, Websocket-Node yang merupakan implementasi Websocket pada Node.js dan Express yang merupakan kerangka kerja HTTP pada Node.js [10].

III. METODE PENELITIAN

A. Jenis Penelitian

Pada penelitian ini akan dibangun aplikasi *marketplace* menggunakan arsitektur *Microservices* sebagai solusi untuk mengurangi penggunaan *resources* yang berlebihan dibagian *server*. Jenis penelitian yang dilakukan adalah penelitian terapan atau *Applied Research*. Penelitian terapan berfungsi untuk mencari solusi tentang masalah-masalah tertentu. Tujuan utama penelitian terapan adalah pemecahan masalah praktis yang timbul ataupun menghasilkan suatu produk yang memiliki fungsi praktis lainnya.

B. Metode Pengumpulan Data

1. Studi Pustaka

Metode ini dilakukan dengan cara mengumpulkan data dari berbagai literatur, seperti buku, jurnal, buku elektronik, dan media internet yang berkaitan dengan penelitian yang akan dibuat untuk membantu proses pengerjaan tugas akhir.

2. Studi Lapangan

Metode ini dilakukan dengan cara melakukan pengamatan langsung pada objek yang diteliti untuk memperoleh data. Adapun teknik pengumpulan data yang dilakukan adalah sebagai berikut :

- Observasi

Melakukan pengamatan langsung ke Pasar Tradisional Modern (PTM) guna mendapatkan data yang asli untuk menunjang proses pengerjaan tugas akhir. Data tersebut meliputi data penjual, dan data produk yang dijual.

C. Metode Pengembangan Sistem

Metode Pengembangan yang digunakan pada sistem yang akan dibuat adalah metode *Prototype* yang merupakan versi awal dari sistem perangkat lunak yang digunakan untuk mengimplementasikan konsep-konsep percobaan perancangan, dan menemukan banyak masalah dan solusi yang memungkinkan. Adapun tahapan dalam metode pengembangan sistem ini adalah :

1. Tahap *Requirement Analysis and Definition*

Pada tahap ini dilakukan analisis terhadap masalah yang terjadi pada objek penelitian. Analisis permasalahan dilakukan dengan studi pustaka, dan studi lapangan. Selain melakukan analisis pemasalahan, analisis kebutuhan ini nantinya dijadikan alat bantu yang digunakan dalam proses pembuatan *prototype* hingga menjadi aplikasi final.

2. Tahap *User Interface Prototyping*

Setelah analisis kebutuhan sistem telah dilakukan, pada tahap ini dilakukan identifikasi kembali mengenai kebutuhan sistem. Apabila kebutuhan sistem telah teridentifikasi dengan baik, dapat dilakukan proses selanjutnya yaitu pembuatan *user interface prototype* yang merupakan tampilan aplikasi yang akan dibangun.

3. Tahap *Architecture & Component Design and Prototyping*

Setelah *prototype user interface* telah dibuat, proses selanjutnya adalah membuat *design, prototype* arsitektur, dan komponen aplikasi yang akan dibangun.

4. Tahap *Implementation and System Testing*

Jika seluruh proses sudah dilakukan maka dilakukan proses pengujian atau testing aplikasi untuk menguji atau mengetahui kualitas dari aplikasi yang telah dibangun, hasil pengujian tersebut akan di evaluasi dan disesuaikan dengan kebutuhan yang ditetapkan sebelumnya.

D. Metode Pengujian Sistem

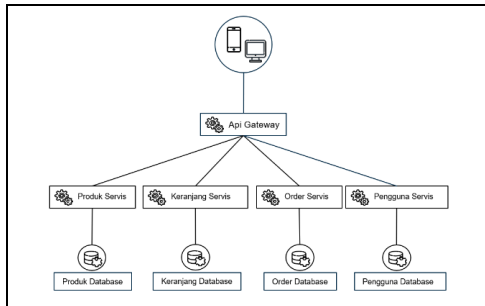
Pengujian yang dilakukan pada aplikasi yang akan dibuat menggunakan *black-box testing*. Pengujian menggunakan *black-box testing* dilakukan dengan cara mengamati hasil eksekusi *end point* yang ada pada *api* melalui data uji dan memeriksa fungsional dari aplikasi yang telah dibuat.

IV. ANALISIS DAN PERANCANGAN

A. Analisis

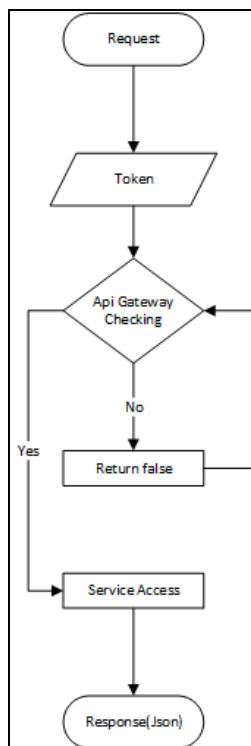
Analisis sistem merupakan penguraian dari suatu sistem yang utuh kedalam bagian-bagian komponennya dengan maksud untuk mengidentifikasikan dan mengevaluasi permasalahan, kesempatan, hambatan yang terjadi.

Apabila digambarkan maka implementasi *microservices* akan memiliki desain infrastruktur seperti Gambar dibawah ini:



Gambar 1 Desain Arsitektur *Microservice*

Untuk alur kerja pada saat melakukan *request* pada suatu servis dapat dilihat dari diagram alir dibawah ini :



Gambar 2 Diagram Alir Kerja *Microservice*

Dalam proses pembuatan *marketplace* yang menggunakan arsitektur *monolith* sudah diketahui sebelumnya bahwa sistem akan banyak menggunakan banyak *resources*, susah dalam *maintenance* sistem untuk mengatasi itu maka digunakanlah arsitektur baru yaitu arsitektur *microservice*, pada penggunaannya arsitektur

microservice ini terdiri dari banyak servis-servis. Servis tersebut saling terhubung, *Api-Gateway* berfungsi untuk menghubungkan servis dan sebagai pengaman agar servis yang ada tidak dapat diakses oleh *public*. Sebelum dapat menggunakan *servis* produk, keranjang, *order* dan pengguna, pengguna harus *login* terlebih dahulu, apabila proses autentikasi berhasil, maka *Api-Gateway* akan memberikan kode akses berupa *token* yang berfungsi untuk autorisasi sebelum dapat mengakses servis, seperti pada Gambar 2.

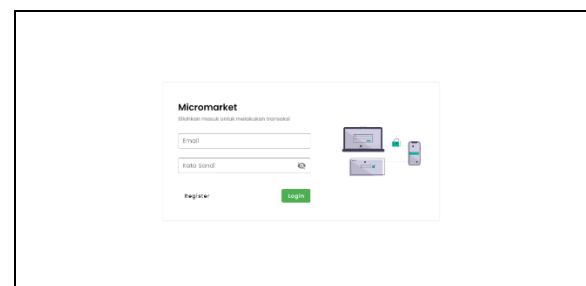
I. PEMBAHASAN

Pada bab ini akan dijelaskan mengenai hasil dan pembahasan dari aplikasi yang telah dibuat, dan penelitian ini telah dihasilkan aplikasi *marketplace* yang menggunakan arsitektur *microservices* berdasarkan analisis yang telah dijelaskan sebelumnya. Penjelasan pada bab ini antara lain terdiri dari implementasi antar muka, implementasi arsitektur dan pengujian *Black Box*.

A. Implementasi Antarmuka

- Halaman *Login*

Halaman *login* merupakan halaman untuk melakukan autentifikasi pengguna sebelum dapat menggunakan aplikasi *marketplace*. Tampilan halaman login dapat dilihat pada Gambar 3.



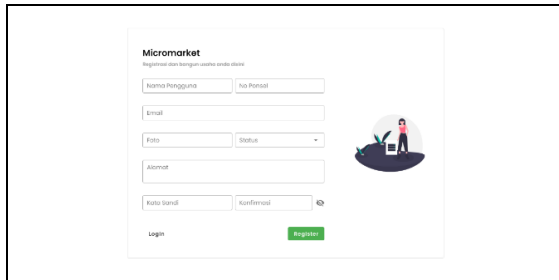
Gambar 3 Halaman *Login*

Pada Gambar 3 untuk melakukan *login* pengguna memasukkan *email* dan kata sandi yang telah dibuat pada registrasi pengguna, apabila

proses autentifikasi berhasil maka pengguna dapat menggunakan sistem.

- Halaman Registrasi

Halaman Registrasi merupakan halaman yang digunakan untuk pendaftaran pengguna aplikasi. Tampilan halaman registrasi dapat dilihat pada Gambar 4.

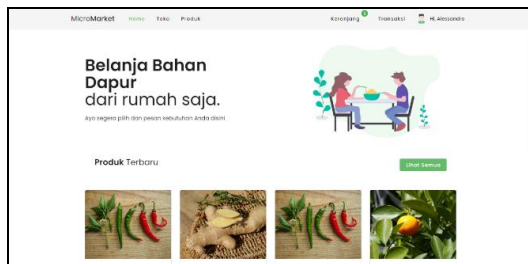


Gambar 4 Halaman Registrasi

Pada Gambar 4 terdapat beberapa *form* yang harus diisi oleh pengguna, *form* tersebut harus terisi seluruhnya agar dapat melakukan registrasi pengguna, jika berhasil maka sistem akan menampilkan halaman login agar pengguna dapat masuk dan menggunakan sistem.

- Halaman Utama

Halaman Utama merupakan halaman yang menampilkan beberapa produk yang baru yang dimasukkan oleh penjual. Tampilan halaman utama dapat dilihat pada Gambar 5.

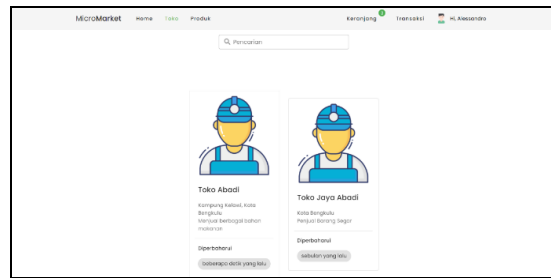


Gambar 5 Halaman Utama

Pada Gambar 5 terdapat beberapa produk yang dijual, dan pada *navigasi bar* ditampilkan *link* menuju halaman toko, produk, keranjang, dan transaksi. *Link* keranjang dan transaksi akan ditampilkan jika pengguna telah masuk kedalam sistem.

- Halaman Toko

Halaman Toko merupakan halaman yang menampilkan pengguna yang berstatus sebagai penjual. Tampilan halaman toko dapat dilihat pada Gambar 5.4.

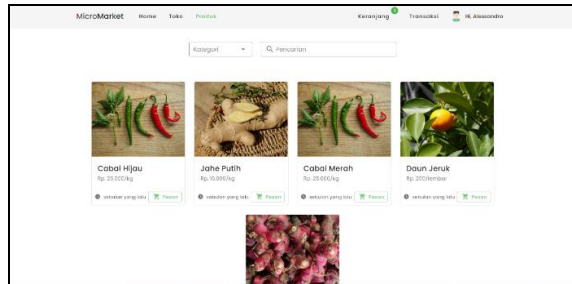


Gambar 6 Halaman Toko

Pada Gambar 6 ditampilkan pengguna yang berstatus sebagai penjual, pengguna juga dapat mencari toko berdasarkan nama toko.

- Halaman Produk

Halaman Produk merupakan halaman yang menampilkan seluruh produk yang dijual oleh penjual. Tampilan halaman produk dapat dilihat pada Gambar 7.

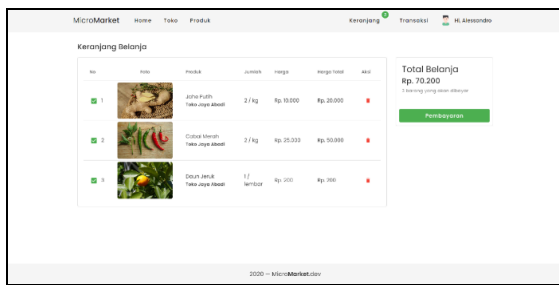


Gambar 7 Halaman Produk

Pada Gambar 7 ditampilkan seluruh produk dan pengguna dapat melakukan fungsi pencarian, dan pengguna dapat melakukan fungsi *filter* berdasarkan kategori produk yang ada.

- Halaman Keranjang

Halaman Keranjang merupakan halaman yang menampilkan produk yang ingin dibeli oleh pengguna. Tampilan halaman keranjang dapat dilihat pada Gambar 8.

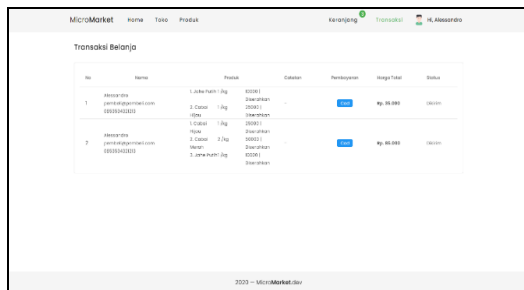


Gambar 8 Halaman Keranjang

Pada Gambar 8 terdapat produk yang telah dimasukkan pengguna kedalam keranjang, produk ini merupakan produk yang ingin dibeli oleh pengguna, pengguna dapat memilih produk yang ingin dibeli, ditampilkan juga jumlah harga total, dan jumlah produk yang ingin dibeli oleh pengguna.

- Halaman Transaksi

Halaman Transaksi merupakan halaman yang menampilkan seluruh transaksi yang dilakukan pengguna. Tampilan halaman transaksi dapat dilihat pada Gambar 9.



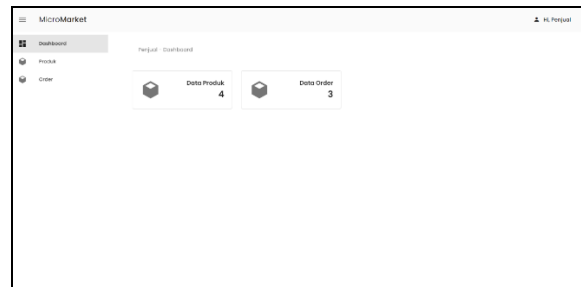
Gambar 9 Halaman Transaksi

Pada Gambar 9 terdapat seluruh transaksi yang telah dilakukan pengguna, ditampilkan nama penerima, *detail* produk yang dibeli, catatan pembeli, metode pembayaran, harga total produk, dan status transaksi yang dilakukan pembeli, pada halaman ini pengguna hanya dapat melihat data transaksi yang telah dibuat.

- Halaman Dashboard Penjual

Halaman *Dashboard* Penjual merupakan halaman yang menampilkan monitoring data manajemen yang ada pada sistem apabila masuk

sebagai penjual. Tampilan halaman *dashboard* penjual dapat dilihat pada Gambar 10.

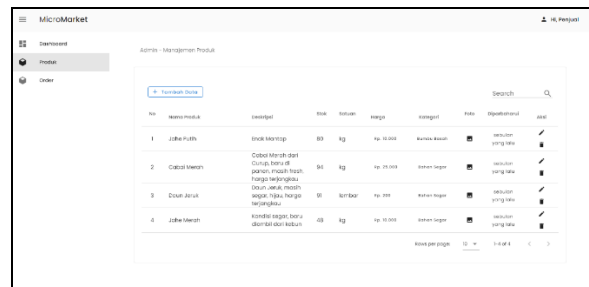


Gambar 10 Halaman Dashboard Penjual

Pada Gambar 10 ditampilkan jumlah data yang ada pada manajemen produk, manajemen produk merupakan data produk yang dimasukkan oleh penjual, dan tampilan juga jumlah data pada manajemen *order* yang merupakan data order pembelian pembeli terhadap produk dijual penjual.

- Halaman Produk

Halaman Produk merupakan halaman yang menampilkan produk yang dijual oleh penjual. Tampilan halaman produk dapat dilihat pada Gambar 11.



Gambar 11 Halaman Produk

Pada Gambar 11 ditampilkan data produk yang dimasukkan oleh penjual, data itu terdiri dari nama produk, deskripsi, stok, satuan, harga, kategori, dan foto. *Button* tambah data berfungsi untuk menambahkan data produk penjual, *button* pensil berfungsi untuk mengubah data, dan *button* sampah berfungsi untuk menghapus data produk.

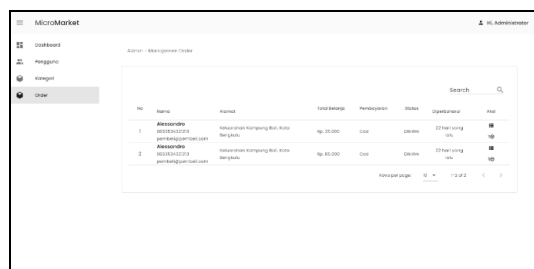
- Halaman Order Penjual

Halaman Order Penjual merupakan halaman yang menampilkan data *order detail* yang

kategori, dan *button* kotak sampah berfungsi untuk menghapus data kategori.

- Halaman Order Admin

Halaman *Order Admin* merupakan halaman yang menampilkan data *order* yang dilakukan pembeli, produk yang sudah dikumpulkan penjual akan diproses oleh administrator untuk segera dilakukan pengiriman ke alamat tujuan. Tampilan halaman *order* penjual dapat dilihat pada Gambar 16.



Gambar 16 Halaman Order Admin

Pada Gambar 16 ditampilkan data order yang meliputi nama pembeli, alamat pembeli, jumlah total belanja dan status *order*. *Button list* pada kolom aksi berfungsi untuk melihat detail produk yang dibeli oleh pembeli, dan *button box* berfungsi untuk mengubah status *order*.

B. Pengujian Endpoint Api

Berikut merupakan pengujian terhadap *endpoint* yang ada pada tiap-tiap servis yang telah dibuat.

Tabel 1 Pengujian Endpoint Api

No	URL	Method	Response	Hasil
Pengguna Servis				
1	/pengguna/login	Post	HTTP Response 200	[√]Berhasil []Tidak Berhasil
2	/pengguna/decode	Post	HTTP Response 200	[√]Berhasil []Tidak Berhasil
3	/pengguna	Get	HTTP Response 200	[√]Berhasil []Tidak Berhasil

No	URL	Method	Response	Hasil
Pengguna Servis				
4	/pengguna	Post	HTTP Response 201	[√]Berhasil []Tidak Berhasil
5	/pengguna/:id	Get	HTTP Response 200	[√]Berhasil []Tidak Berhasil
6	/pengguna/:id	Put	HTTP Response 200	[√]Berhasil []Tidak Berhasil
7	/pengguna/:id	Delete	HTTP Response 200	[√]Berhasil []Tidak Berhasil
8	/pengguna/change_password	Post	HTTP Response 200	[√]Berhasil []Tidak Berhasil
Produk Servis				
9	/produk/kategori	Get	HTTP Response 200	[√]Berhasil []Tidak Berhasil
10	/produk/kategori	Post	HTTP Response 201	[√]Berhasil []Tidak Berhasil
11	/produk/kategori/:id	Get	HTTP Response 200	[√]Berhasil []Tidak Berhasil
12	/produk/kategori/:id	Put	HTTP Response 200	[√]Berhasil []Tidak Berhasil
13	/produk/kategori/:id	Delete	HTTP Response 200	[√]Berhasil []Tidak Berhasil
14	/produk	Get	HTTP Response 200	[√]Berhasil []Tidak Berhasil
15	/produk/keranjang	Get	HTTP Response 200	[√]Berhasil []Tidak Berhasil
16	/produk	Post	HTTP Response 201	[√]Berhasil []Tidak Berhasil
17	/produk/:id	Get	HTTP Response 200	[√]Berhasil []Tidak Berhasil

No	URL	Method	Response	Hasil
Pengguna Servis				
18	/produk/:id	Put	HTTP Response 200	[√]Berhasil []Tidak Berhasil
19	/produk/:id	Delete	HTTP Response 200	[√]Berhasil []Tidak Berhasil
Keranjang Servis				
20	/keranjang	Get	HTTP Response 200	[√]Berhasil []Tidak Berhasil
21	/keranjang	Post	HTTP Response 201	[√]Berhasil []Tidak Berhasil
22	/keranjang/:id	Put	HTTP Response 200	[√]Berhasil []Tidak Berhasil
23	/keranjang/:id	Delete	HTTP Response 200	[√]Berhasil []Tidak Berhasil
Order Servis				
24	/order	Get	HTTP Response 200	[√]Berhasil []Tidak Berhasil
25	/order/detail	Get	HTTP Response 200	[√]Berhasil []Tidak Berhasil
26	/order	Post	HTTP Response 201	[√]Berhasil []Tidak Berhasil
27	/order/:id	Put	HTTP Response 200	[√]Berhasil []Tidak Berhasil
28	/order/detail/:id	Put	HTTP Response 200	[√]Berhasil []Tidak Berhasil
29	/order/:id	Delete	HTTP Response 200	[√]Berhasil []Tidak Berhasil

V. KESIMPULAN DAN SARAN

A. Kesimpulan

Berdasarkan hasil penelitian, pengujian, implementasi dan pembahasan yang telah dilakukan, maka didapatkan kesimpulan sebagai berikut :

1. Penelitian ini menghasilkan aplikasi *marketplace* menggunakan arsitektur *microservices* berbasis *web*, arsitektur *microservices* terdiri lima servis meliputi *api-gateway*, produk servis, keranjang servis, *order* servis dan pengguna servis.
2. Aplikasi *marketplace* menggunakan arsitektur *microservices* telah berhasil dibangun dengan hasil persentase pengujian *black box* 100% dari 25 aktivitas berhasil dan hasil persentase pengujian *endpoint api* 100% dari 29 aktivitas berhasil.
3. Aplikasi *marketplace* telah dilakukan uji kelayakan menggunakan kuisioner kepada pedagang Pasar Tradisional Bengkulu (PTM) didapat nilai variabel tampilan 4,54 dengan kategori sangat baik, variabel kemudahan pengguna 4,63 dengan kategori sangat baik, dan variabel kinerja sistem 4,22 dengan kategori baik.

B. Saran

Berdasarkan hasil penelitian, pengujian, implementasi dan pembahasan yang telah dilakukan, maka penulis menyarankan sebagai berikut :

1. Peneliti selanjutnya dapat menggunakan teknologi pembayaran berupa *payment-gateway* untuk mempermudah proses pembayaran.
2. Peneliti selanjutnya dapat menggunakan *Docker* dan *Kubernetes* untuk manajemen

container pada *Docker* agar dapat memaksimalkan kinerja *container*.

3. Peneliti selanjutnya dapat melakukan *benchmark* terhadap *rest api* apabila menggunakan bahasa pemrograman yang berbeda dalam mengatasi request dengan data yang banyak.

VI. REFERENSI

- [1] C. H. "Pemanfaatan Teknologi Informasi dan Komunikasi dalam Pembelajaran di SMA Muhammadiyah Tarakan," *Jurnal Kebijakan dan Pengembangan Pendidikan*, pp. 184-192, 2014.
- [2] K. Konsep dan Aplikasi Sistem Pendukung Keputusan, Yogyakarta: Andi, 2007.
- [3] M. Abdillah, I. and R. Hidayati, "Penerapan Metode Analytic Network Process (ANP) Berbasis Android Sebagai Sistem Pendukung Keputusan Dalam Pemilihan Tempat Kos," *Jurnal Coding, Rekayasa Sistem Komputer Untan*, pp. 12-22, 2018.
- [4] T. Handayani, "Penerapan Analytic Network Process(ANP) Pada Sistem Pendukung Keputusan," *Jurnal Transformatika*, pp. 73-78, 2017.
- [5] R. "ANALISA PERBANDINGAN METODE ANP DAN SAW DALAM MENENTUKAN MAHASISWA TERBAIK," *Jurnal Mantik Penusa*, pp. 43-50, 2019.
- [6] A. Ohri and P. K. Singh, "Development of Decision Support System for Municipal Solid Waste Management in India," *International Journal of Environment Banaras Hindu Univesity 1*, 2010.
- [7] K. B. Surarso and W. A. Syafei, "Implementasi Metode Analytic Network Proses Untuk Penentuan Prioritas Penanganan Jalan Berdasarkan Tingkat Pelayanan Jalan," *Jurnal Sistem Informasi Bisnis*, pp. 105-113, 2016.
- [8] R. and M. Shalahuddin, *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek*, Bandung: Informatika Bandung, 2018.