ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)

PERANCANGAN APLIKASI E-COMMERCE MENGGUNAKAN CLOUD COMPUTING DAN ARSITEKTUR SEVERLESS LAMBDA

Liem Margareth Santoso✉, Frederik Samuel Papilaya
Program Studi Sistem Informasi, Universitas Kristen Satya Wacana, Salatiga, Indonesia
Email: 682020119@student.uksw.edu

DOI: https://doi.org/10.46880/jmika.Vol8No2.pp155-163

ABSTRACT

E-commerce applications are a rapidly growing field within the digital industry. The use of cloud computing technology and Serverless Lambda has provided numerous benefits in the development and management of e-commerce applications. However, the need for flexibility, scalability, and operational efficiency is increasingly driving the adoption of serverless architecture as an approach to building adaptive and cost-effective e-commerce applications. This application enables sellers to market products, receive orders, and manage inventory efficiently, while customers can easily browse product catalogs, add items to their shopping cart, and complete payments. Through performance analysis and evaluation, we demonstrate that serverless Lambda architecture can automatically scale according to application traffic needs. Additionally, the serverless approach reduces infrastructure management overhead, allowing developers to focus on developing features and functionalities that directly impact the user experience. The methods used in this research are UML (Unified Modeling Language) and serverless Lambda-based Cloud Computing architecture.

Keyword: Cloud Computing, Arsitektur Serverless, Severless Lambda, Unified Modeling Language.

ABSTRAK

Aplikasi e-commerce merupakan salah satu bidang yang terus berkembang dengan pesat dalam industri digital. Penggunaan teknologi cloud computing dan Serverless Lambda telah memberikan banyak keuntungan dalam pengembangan dan pengelolaan aplikasi e-commerce. Namun, kebutuhan akan fleksibilitas, skalabilitas, dan efisiensi operasional semakin mendorong adopsi arsitektur serverless sebagai pendekatan untuk membangun aplikasi e-commerce yang adaptif dan hemat biaya. Aplikasi ini memungkinkan penjual untuk memasarkan produk, menerima pesanan, dan mengelola inventaris secara efisien, sementara pelanggan dapat dengan mudah menjelajahi katalog produk, menambahkan item ke keranjang belanja, dan menyelesaikan pembayaran. Melalui analisis kinerja dan evaluasi, kami menunjukkan bahwa arsitektur serverless lambda mampu memberikan skalabilitas secara otomatis sesuai dengan kebutuhan lalu lintas aplikasi. Selain itu, pendekatan serverless juga mengurangi beban administrasi infrastruktur, memungkinkan pengembang untuk fokus pada pengembangan fitur dan fungsionalitas yang berdampak langsung pada pengalaman pengguna. Metode yang digunakan pada penelitian ini adalah UML (Unified Modeling Language) dan Arsitektur Cloud Computing berbasis serverless lambda.

Kata Kunci: Cloud Computing, Arsiterktur Severless, Severless Lambda, Unified Modeling Language.

PENDAHULUAN

Perkembangan teknologi informasi tentu berdampak signifikan dalam dunia bisnis (Aprianto, 2021). Khususnya dalam perpindahan perdagangan yang sebelumnya menggunakan sistem tradisional berganti menjadi sistem modern. Bila dulu pembeli harus ke toko untuk membeli atau melihat suatu barang atau produk sekarang dengan adanya kemudahan teknologi kita dapat membeli barang melalui internet. Seiring dengan perkembangan teknologi persaingan perdagangan juga tidak bisa dihindari. Pada persaingan global ini bukan hanya kualitas barang saja yang harus diperhatikan namun pemberian sistem pelayanan jasa juga harus menjadi suatu pertimbangan.

E-Commerce saat ini telah menjadi salah satu elemen kunci dalam transformasi bisnis global. Transaksi perdagangan tidak lagi terbatas pada interaksi fisik antara pembeli dan penjual, melainkan terjadi secara virtual melalui platform *online*. Baik untuk bisnis kecil maupun besar, *e-commerce* menawarkan berbagai manfaat, termasuk akses pasar yang lebih luas, biaya operasional yang lebih rendah (Sugiarti et al., 2020), kemampuan untuk menjangkau

pelanggan di seluruh dunia, dan fleksibilitas dalam pengelolaan inventaris dan proses transaksi.

Teknologi *cloud computing* dan arsitektur *serverless* telah membuka pintu bagi inovasi baru dalam berbagai bidang, termasuk dalam pembangunan aplikasi. Aplikasi berbasis *cloud computing* memberikan fleksibilitas, skalabilitas, dan efisiensi yang tinggi dalam menyediakan layanan kepada pengguna (Alfarizi & Ikasari, 2023). Sementara itu, arsitektur *serverless,* yang memungkinkan pengembang untuk fokus pada kode bisnis mereka tanpa harus mengelola infrastruktur, telah menjadi pendekatan yang populer dalam pengembangan aplikasi modern.

Dengan memanfaatkan layanan *cloud computing,* perusahaan *e-commerce* dapat mengoptimalkan operasional mereka dengan meningkatkan skalabilitas dan elastisitas infrastruktur, yang memungkinkan untuk menangani lonjakan lalu lintas tanpa meningkatkan biaya secara signifikan. Selain itu, fleksibilitas yang ditawarkan oleh *cloud* memungkinkan untuk integrasi yang lebih mudah dengan layanan lain, memungkinkan pemilik bisnis untuk menyediakan pengalaman pengguna yang lebih responsif dan terjangkau.

Dalam penelitian sebelumnya yang berjudul "Perancangan Sistem Informasi *E-Commerce* Berbasis *Website* Menggunakan Metode *Waterfall*" fokus utamanya adalah pada pengembangan aplikasi *e-commerce* berbasis *website* dan *Android.* Penelitian ini bertujuan untuk mempermudah proses transaksi dengan memanfaatkan teknologi elektronik seperti *smartphone* atau komputer dan tidak lagi menggunakan metode jual beli manual (Al Ghani et al., 2022). Kesimpulan dari penelitian tersebut menyoroti kesamaan dalam topik, yaitu perancangan aplikasi *e-commerce.* Penelitian tersebut menjelaskan tujuan pembuatan aplikasi *e-commerce* yang dirancang, dengan fokus pada aplikasi berbasis *website* dan Android. Dengan memanfaatkan *cloud computing* untuk *e-commerce,* Anda dapat meningkatkan efisiensi operasional, mengurangi risiko, dan menyediakan pengalaman pengguna yang lebih baik. Ini memungkinkan bisnis *e-commerce* untuk beradaptasi dengan cepat terhadap perubahan pasar dan kebutuhan pelanggan.

Sebagai perbedaan, penulis penelitian ini mengambil pendekatan yang berbeda dengan mengeksplorasi perancangan aplikasi e-commerce menggunakan teknologi Cloud Computing dan serverless architecture. Pendekatan ini bertujuan untuk memanfaatkan keunggulan teknologi cloud dan arsitektur serverless dalam meningkatkan efisiensi dan skalabilitas aplikasi e-commerce, yang berbeda dari metode konvensional yang telah dibahas sebelumnya.

Melalui penelitian diharapkan dapat memberikan kontribusi dalam pengembangan aplikasi *E-commerce* yang lebih efisien, terjangkau, dan dapat diandalkan. Dengan memanfaatkan kekuatan *cloud computing* dan arsitektur *serverless* Lambda, perusahaan dapat memperluas jangkauan bisnis mereka, meningkatkan efisiensi operasional, dan memberikan pengalaman pengguna yang lebih baik.

TINJAUAN PUSTAKA

E-commerce (electronic commerce)

E-commerce atau perdagangan elektronik, adalah sistem yang memungkinkan individu dan perusahaan untuk melakukan transaksi jual beli barang dan jasa secara *online* melalui internet. Dengan *e-commerce,* pelanggan dapat membeli produk atau layanan secara digital dan melakukan pembayaran elektronik tanpa harus berkunjung ke toko fisik. Hal ini telah mengubah lanskap bisnis secara global, membuka peluang baru bagi pelaku usaha untuk mencapai pasar yang lebih luas dan meningkatkan kenyamanan serta aksesibilitas bagi konsumen. *E-commerce* menghubungkan penjual dan pembeli dari berbagai penjuru dunia, memungkinkan mereka untuk bertransaksi secara digital dari mana saja selama terhubung dengan internet. Teknologi EFT (*Electronic Funds Transfer*) juga memfasilitasi pengiriman dana antara pelaku ekonomi di berbagai belahan dunia dengan cepat (Hanim, 2011).

Cloud Computing

Menurut (Scale, 2009) *cloud computing* dapat didefinisikan sebagai penggunaan dan berbagi aplikasi serta sumber daya dalam suatu lingkungan jaringan untuk melakukan pekerjaan tanpa perlu khawatir tentang kepemilikan dan pengelolaan sumber daya dan aplikasi jaringan tersebut. Dengan *cloud computing,* sumber daya komputer untuk melakukan pekerjaan dan data mereka tidak lagi disimpan di komputer pribadi seseorang, tetapi di-*hosting* di tempat lain untuk dapat diakses dari mana saja dan kapan saja.

Serverless Architecture

Serverless architecture adalah pengembangan perangkat lunak di mana pengembang tidak perlu

memikirkan infrastruktur server fisik secara langsung. Dalam arsitektur *serverless*, penyedia layanan cloud mengelola infrastruktur server untuk menjalankan kode aplikasi, sehingga pengembang hanya perlu fokus pada penulisan kode aplikasi dan tidak perlu memikirkan tentang manajemen server, *provisioning,* atau skalabilitas. Dengan komputasi tanpa server, tugas manajemen infrastruktur seperti penyediaan kapasitas dan patching ditangani oleh penyedia layanan, sehingga pengembang hanya perlu fokus pada penulisan kode yang melayani pelanggan. Dengan demikian, hal ini dapat mengurangi biaya pengeluaran tambahan dan juga dapat menghemat waktu dan energi bagi pengembang (Oktaviani et al., 2021).

AWS Lambda

AWS Lambda adalah layanan komputasi tanpa server yang ditawarkan oleh *Amazon Web Services* (AWS). Dengan AWS Lambda, pengembang dapat mengeksekusi kode tanpa perlu memikirkan manajemen infrastruktur server secara langsung. Layanan ini memungkinkan pengguna untuk menjalankan kode dalam berbagai bahasa pemrograman, seperti *Python,* Node.js, Java, dan lainnya, sebagai respons terhadap peristiwa atau panggilan API. Lambda berperan dalam penelitian ini dengan melakukan kueri ke *database* setiap jangka waktu tertentu, kemudian mengirimkan notifikasi melalui layanan AWS SNS *(Simple Notification Service*) ketika terdeteksi kondisi suhu yang tidak normal (Mulyani & Oktiawati, 2022).

Amazon DynamoDB

Amazon DynamoDB adalah layanan *database* NoSQL yang di-*host* yang menyediakan penyimpanan data yang cepat dan fleksibel. DynamoDB dirancang untuk menyimpan dan mengakses data terstruktur secara terdistribusi. Ini dapat menangani jumlah data besar dengan cepat dan dapat diandalkan. DynamoDB mendukung berbagai model data, termasuk key-value dan *document data models*. Ini juga menyediakan fitur seperti dukungan transaksi, *indexing*, dan *autoscaling* untuk menangani beban kerja yang berfluktuasi

DynamoDB tanpa server menghilangkan kebutuhan untuk mengelola infrastruktur server secara langsung. Dengan pendekatan ini, tidak perlu khawatir tentang penyediaan atau pemeliharaan server, serta proses pemasangan dan pengelolaan perangkat lunak. Sebagai gantinya, DynamoDB tanpa server memungkinkan fokus pada pengembangan aplikasi tanpa terganggu oleh tugas administratif yang berkaitan dengan server. Layanan ini secara otomatis menangani penalaan tabel, menyesuaikan kapasitas untuk mempertahankan kinerja yang optimal, sehingga mudah menyesuaikan diri dengan kebutuhan aplikasi tanpa harus melakukan pengaturan manual (Sitompul et al., 2019).

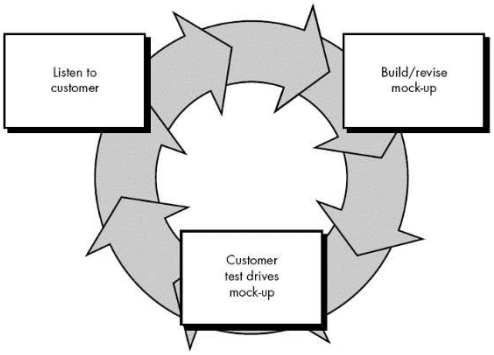
METODE PENELITIAN

Metode Pengumpulan Data

Dalam mengumpulkan informasi yang akan digunakan dalam menulis penelitian ini digunakan metode pengumpulan data yang melalui beberapa tahap, sebagai berikut:

1. Kajian Pustaka
   Informasi diperoleh dengan meninjau berbagai literatur dari berbagai jurnal ilmiah melalui situs-situs internet, serta mempelajari konten dari jurnal tersebut dan situs web yang relevan dengan topik penelitian.
2. Observasi
   Data diperoleh melalui pengamatan langsung di lokasi. Sebagai contoh, mengamati aktivitas yang terjadi di restoran yang berkaitan dengan masalah penelitian.
3. Wawancara
   Data dikumpulkan melalui wawancara tatap muka dengan pihak yang berkepentingan untuk mendapatkan sumber informasi

Metode Pengembangan Perangkat Lunak



Gambar 1. Tahapan Prototyping

Dalam penelitian ini, digunakan metode penelitian prototyping dengan tahapan sebagai berikut:

1. Mendengarkan Pelanggan:
   Langkah pertama dalam tahap prototype adalah mendengarkan pelanggan secara aktif. Dalam tahap ini melibatkan pengumpulan umpan balik,

kebutuhan, dan harapan dari calon pengguna atau pelanggan terhadap prototype yang akan dibangun.

2. Membangun dan memperbaiki prototype:
   Setelah mendapatkan pemahaman mengenai kebutuhan pelanggan, tim pengembangan mulai membangun prototype produk atau layanan. Prototype adalah versi awal dari produk atau layanan yang dirancang untuk mendemonstrasikan fitur-fitur utama dan fungsionalitasnya. Prototipe ini biasanya tidak sepenuhnya sempurna, tetapi fokus pada merepresentasikan ide-ide inti produk. Setelah prototype pertama dibangun, tim pengembangan memperbaiki dan meningkatkannya berdasarkan umpan balik yang diterima dari pelanggan atau pengujian internal. Proses ini melibatkan iterasi, di mana prototipe diperbaiki, diuji lagi, dan kemudian diperbaiki lagi untuk mencapai tingkat kualitas dan fungsionalitas yang diinginkan.
3. Uji Coba oleh Pelanggan:
   Setelah prototipe telah diperbaiki sebanyak mungkin, tahap selanjutnya adalah menguji coba prototipe tersebut oleh pelanggan yang sesungguhnya. Uji coba dilakukan untuk memastikan bahwa prototipe dapat memenuhi kebutuhan dan harapan pelanggan secara efektif. Umpan balik yang diterima dari uji coba pelanggan dapat digunakan untuk memperbaiki prototipe lebih lanjut sebelum produk akhir diluncurkan ke pasar.

Analisis Kebutuhan

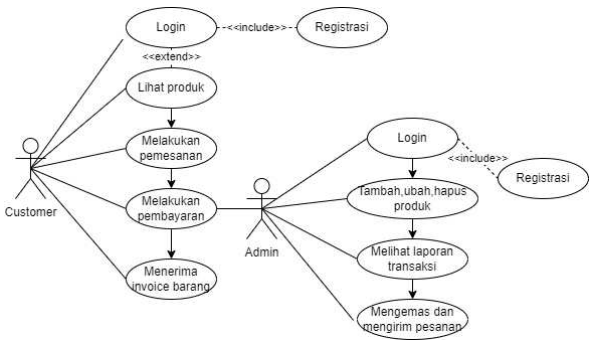
Hak Akses	Deskripsi
Customer	Registrasi data *customer* diperlukan untuk melakukan *login*, melihat produk, melakukan pemesanan dan melakukan pembayaran.
Admin	Registrasi data admin yang diperlukan untuk *login*, mengelola data produk seperti menambah, mengubah, dan menghapus produk, menerima pemesanan dan pembayaran dari *customer.*

Membangun Prototype

Setelah memperhatikan kebutuhan pelanggan melalui pendekatan prototyping, langkah selanjutnya adalah merancang aplikasi berdasarkan analisis dan interaksi di SCA.ID. Desain ini mencakup struktur dan fungsionalitas aplikasi yang diusulkan, yang diperlihatkan melalui diagram UML seperti Use Case Diagram, Activity Diagram, dan Class Diagram, bersama dengan tampilan antarmuka pengguna (UI)

***Use Case* Diagram**

Diagram Use Case menjelaskan bagaimana pengguna akan berinteraksi dengan sistem untuk mencapai tujuan tertentu. Ini membantu dalam pemahaman yang lebih baik tentang kebutuhan pengguna dan skenario penggunaan sistem. Berikut adalah perancangan *Use Case* diagram:



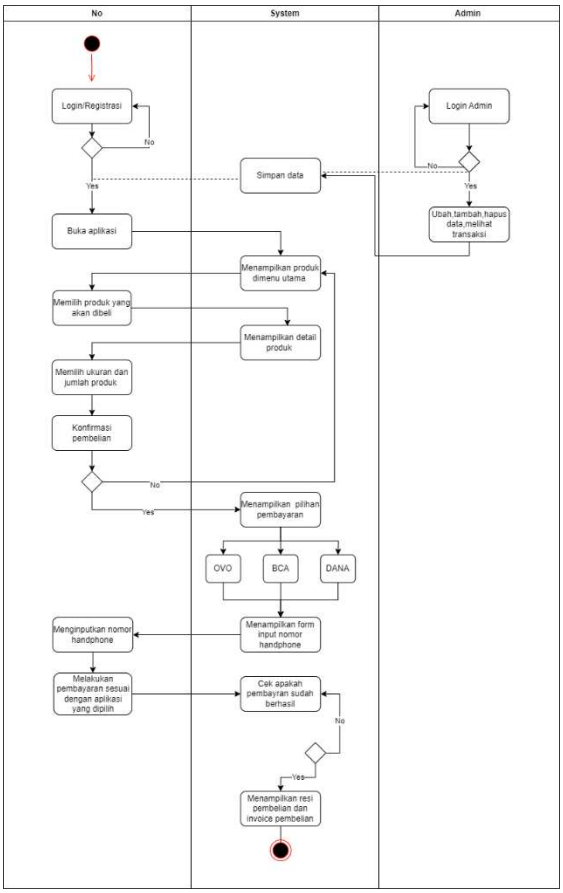
Gambar 2. *Use Case* Diagram

Pada gambar 2. *Use Case* Diagram menunjukkan bahwa terdapat 2 aktor yaitu *customer* dan admin yang memiliki peran tersendiri dalam menjalankan kebutuhan mereka. *Customer* dan admin diharuskan untuk melakukan registrasi terlebih dahulu untuk mendapatkan hak akses yang akan digunakan nantinya.

Customer memiliki hak akses untuk melihat produk ,melakukan pemesanan dan melakukan pembayaran. Dan admin memiliki hak akses untuk mengelola data produk seperti produk seperti menambah, mengubah, dan menghapus produk, menerima pemesanan dan pembayaran dari *customer*

***Activity* Diagram**

Activity diagram adalah representasi visual dari alur kerja atau aktivitas di dalam suatu proses. Kegunaannya untuk menggambarkan urutan langkah-langkah dan aktivitas yang perlu dilakukan di suatu proses atau skenario tertentu. *Activity* diagram membantu dalam memodelkan proses bisnis, alur kerja perangkat lunak, atau interaksi sistem dengan sistem lain atau pengguna. *Activity* diagram membantu dalam memahami logika operasional suatu sistem atau proses secara visual.



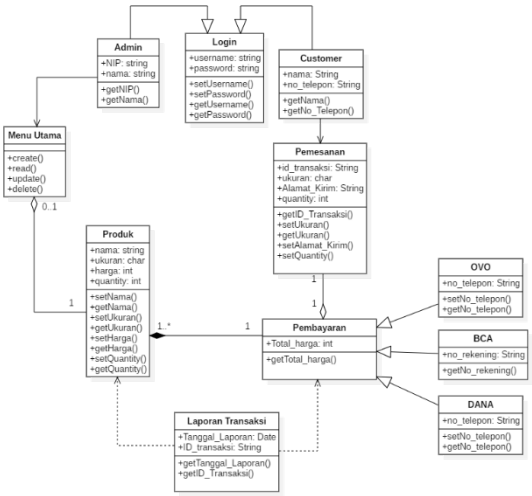
Gambar 3. *Activity* Diagram

Pada gambar 3 Activity Diagram dapat dilihat untuk customer dapat melakukan pemesanan, customer akan diinta untuk melkaukamn registrasi atau login terlebih dahulu lalu sistem akan menyimpan data customer. Saat login berhasil maka customer bisa melihat dan memilih produk yang telah ditampilkan oleh sistem dan customer bisa melakukan pemesanan dan pembayaran dan sistem akan menampilkan nota pembayaran dari pesanan customer.

Dalam proses bisnis ini admin memiliki peran untuk menambah detail dan data produk,mengubah produk menghapus produk dan melihat transaksi yang telah dilakukan oleh customer.

Class Diagram

Class diagram adalah ilustrasi bagaimana bagian-bagian dari suatu sistem perangkat lunak bekerja bersama-sama dan saling berhubungan. Dalam diagram ini menunjukkan jenis informasi apa yang disimpan oleh bagian-bagian tersebut, apa yang dapat mereka lakukan, dan bagaimana mereka berhubungan satu sama lain. *Class* diagram membantu dalam memahami struktur data dan hubungan antar bagian-bagian dari sistem perangkat lunak, termasuk pewarisan, asosiasi, agregasi, dan komposisi.



Gambar 4. *Class* Diagram

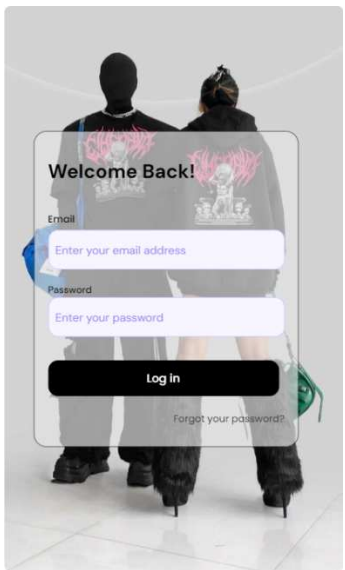
Pada gambar 4 dapat dilihat tampilan atribut serta *field* yang digunakan di *database* aplikasi SCA.ID

HASIL DAN PEMBAHASAN

Implementasi Tampilan *UI (User Interface)* Pada Sistem Aplikasi Android

Implementasi tampilan *User Interface* pada sistem aplikasi android yang akan digunakan oleh *customer* dan admin dalam menjalankan kebutuhannya sesuai dengan hak akses sistem masing-masing

Tampilan Halaman *Login Customer*



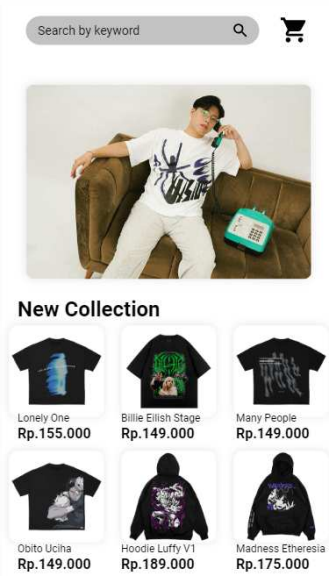
Gambar 5. Halaman *Login Customer*

ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)

Pada gambar 5 merupakan implementasi tampilan login di aplikasi android SCA.ID. *Customer* mengisi email dan *password* yang sah untuk dapat masuk ke dalam aplikasi SCA.ID.

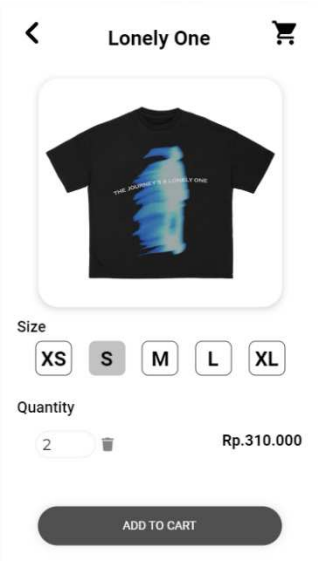
Tampilan Halaman Menu Utama *Customer*

Saat *customer* berhasil login maka halaman menu uatma *customer* ditampilkan seperti pada gambar 6. Pada halaman ini customer dapat mencari produk yang diinginkan dan dapat melihat produk yang ditampilkan.



Gambar 6. Halaman Menu Utama

Tampilan Halaman Detail Produk *Customer*

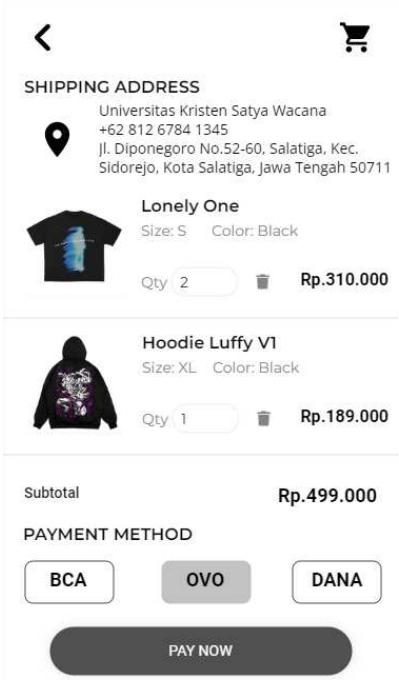


Gambar 7. Halaman Detail Produk

Saat customer menekan sebuah produk pada halaman utama maka akan dialihkan pada halaman detail produk. Pada gambar 7 dapat dilihat halaman detail produk yang berisi informasi prosuk seperti nama produk, ukuran produk, harga produk dan tombol masukan produk ke dalam keranjang.

Tampilan Halaman Detail Pemesanan Customer

Setelah *customer* memilih produk yang akan dibeli maka akan dialihkan halaman detail pemesanan. Pada gambar 8. dapat dilihat halaman detail pemesanan yang berisi alamat pengiriman pesanan, produk yang akan dibeli oleh *customer* dan pilihan metode pembayaran yaitu melalui OVO, DANA atau Bank BCA.

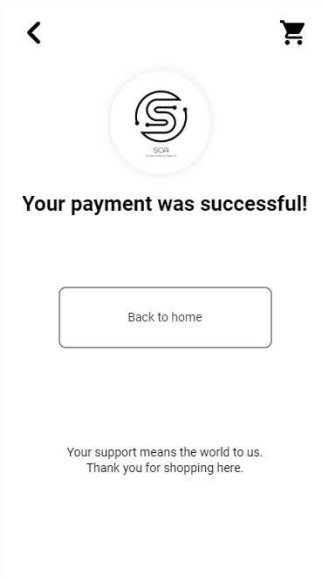


Gambar 8. Halaman Detail Pemesanan

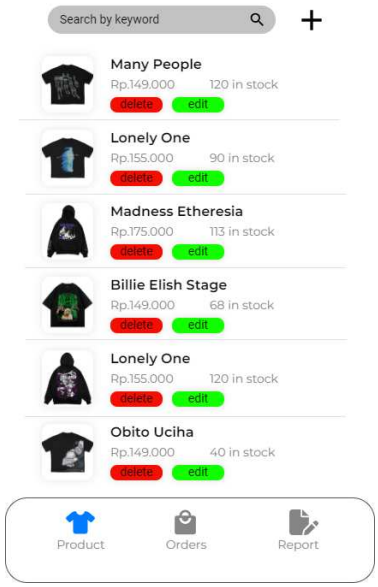
Tampilan Halaman Pemberitahuan Pembayaran Berhasil

Pada gambar 9 merupakan halaman pemebritahuan bahwa pembayaran pemesanan yang dilakuakn oleh *customer* telah berhasil. Lalu aplikasi *customer* akan kembali ke halaman utama.

ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)



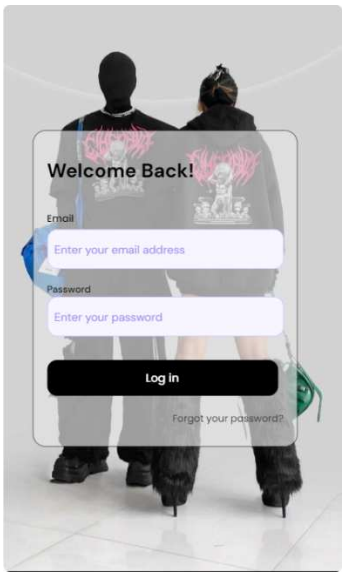
Gambar 9. Halaman Pemberitahuan Pembayaran Berhasil



Gambar 11. Halaman Tambah,Hapus dan Edit Produk Admin

Tampilan Halaman *Login* Admin

Pada gambar 10 merupakan implementasi tampilan *login* di aplikasi android SCA.ID. Admin mengisi email dan *password* yang sah untuk dapat masuk ke dalam aplikasi SCA.ID.

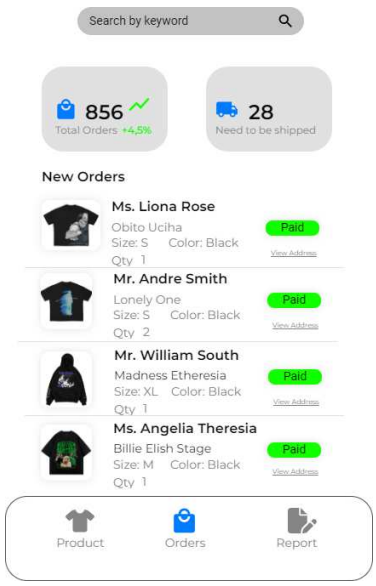


Gambar 10. Halaman *Login* Admin

Setelah Admin berhasil *login* maka pada Admin dapat menambah produk baru,menghapus produk yang diinginkan dan mengedit informasi produk pada halaman tambah ,hapus dan edit produk seperti pada gambar 11.

Tampilan Halaman *Order* Admin

Pada gambar 12. Merupakan halaman *orders* dinama admin dapat melihat pesanan baru beserta detail pesanan seperti tipe baju, ukuran baju, jumlah pesanan, nama *customer,* alamat pengiriman, dan pesanan yang harus dikirimkan ke *customer.*



Gambar 12. Halaman *Orders* Admin

Tampilan Halaman Laporan Transaksi Admin

Pada Gambar 13 merupakan halaman laporan transaksi yang berisi jumlah pengunjung *customers,* total pesanan yang dilakukan *customers*, total berapa

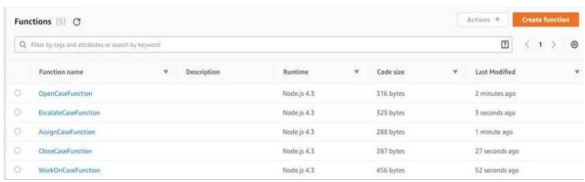
ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)

kali produk dilihat, persentase percakapan dilakukan dengan *customers,* dan diagram pendapatan toko.

Gambar 13. Halaman Laporan Transaksi Admin

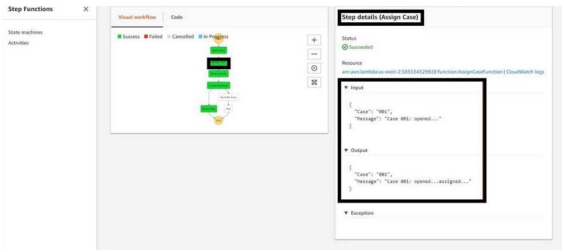
Alur Kerja dengan Menggunakan AWS Lambda

Fungsi pertama, *AssignCaseFunction,* bertanggung jawab atas penetapan kasus serta pembaruan status pesan. Sementara itu, *WorkOnCaseFunction* berperan dalam memberikan pemberitahuan mengenai status penyelesaian kasus dan mengembalikan nilai bersamaan dengan pesan pembaharuan. Fungsi terakhir, *CloseCaseFunction,* digunakan untuk menutup kasus. Ketika alur kerja dieksekusi oleh *step function,* setiap langkah akan diperiksa, dan ID kasus yang dimasukkan akan diteruskan dari langkah satu ke langkah berikutnya. Terdapat juga pembaruan pesan setiap fungsi lambda menyelesaikan tugasnya, seperti yang diilustrasikan dalam Gambar 14.

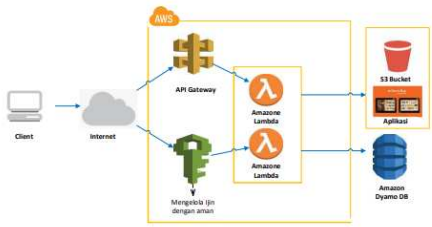


Gambar 14. Fungsi AWS Lambda

Pada gambar 15. menunjukkan langkah eksekusi alur kerja, termasuk *input* dan *output* setiap status. Yang awalnnya sebuah kotak dialog di isi ID untuk *inputcaseID* "001", yang kemudian dieksekusi. Setelah tereksekusi gulir ke bawah di bagian Riwayat eksekusi bagaimana step *function* memanggil fungsi lambda. Riwayat Eksekusi menunjukkan bagian Riwayat kejadian eksekusi. Setiap langkah eksekusi akan terlihat bagaimana *Step Functions* memanggil fungsi Lambda dan meneruskan data di antara fungsi yang sudah dibuat.



Gambar 15. Ekseskusi Alur Kerja



Gambar 16. Arsiterktur Sistem

Sistem aplikasi *E-commerce* direncanakan dengan menggunakan arsitektur *serverless* Lambda di AWS. Klien akan berinteraksi melalui browser untuk menjalankan fungsi-fungsi aplikasi. Antarmuka aplikasi akan di-*hosting* menggunakan Amazon S3 untuk menyajikan sumber daya statis aplikasi web. Fungsi-fungsi dinamis akan diimplementasikan melalui AWS Lambda, yang akan dipanggil melalui Amazon API *Gateway* sebagai RESTful API. Pengaturan akses yang aman akan ditangani oleh layanan AWS *Identity and Access Management* (IAM), memastikan izin yang sesuai untuk interaksi antar layanan. Data yang diproses oleh aplikasi akan disimpan dan diakses menggunakan *Amazon DynamoDB*. Dengan demikian, sistem *E-commerce* dibangun sepenuhnya dengan menggunakan layanan AWS Lambda.

KESIMPULAN

Dalam penelitian ini, beberapa perancangan dan prototipe aplikasi *E-Commerce* telah dihasilkan. Ini mencakup tampilan antarmuka pembelian pengguna dan antarmuka pengelolaan penjualan. Selain desain tampilan, penelitian ini juga mencakup desain sistem menggunakan diagram *Use Case,* diagram *Activity*, dan diagram *Class* untuk memahami proses bisnis aplikasi tersebut. Analisis menunjukkan bahwa sistem

penjualan *online* yang efisien memerlukan metode yang terintegrasi, termasuk proses pemesanan oleh pelanggan, pengelolaan oleh admin, serta pembayaran yang mudah. Dalam hal arsitektur, *Cloud Computing* berbasis serverless lambda diusulkan sebagai solusi yang baik. Hal ini akan mendukung aplikasi *E-Commerce* dengan skalabilitas yang memadai untuk pemasaran global, serta mengurangi biaya operasional.

DAFTAR PUSTAKA

Al Ghani, R., Azani, N. W., Auliani, S. N., Maharani, S., Gustinov, M. D., & Hamzah, M. L. (2022). Perancangan Sistem Informasi E-Commerce Berbasis Website Menggunakan Metode Waterfall. *Prosiding Seminar Nasional Teknologi Informasi Dan Bisnis*, 99–106.

Edwin Kiky Aprianto, N. (2021). Peran Teknologi Informasi dan Komunikasi dalam Bisnis. *International Journal Administration, Business and Organization (IJABO)* |, *2*(1), 1–7. https://ijabo.a3i.or.id

Hanim, L. (2011). Pengaruh perkembangan teknologi informasi terhadap keabsahan perjanjian dalam perdagangan secara elektronik (e-commerce) di era globalisasi. *Jurnal Dinamika Hukum*, *11*, 56–66.

Mulyani, A., & Oktiawati, U. Y. (2022). Implementasi Arsitektur Serverless Internet of Things pada Monitoring Cold Chain. *Journal of Internet and Software Engineering*, *3*(1).

Nafis Alfarizi, D., & Heidiani Ikasari, I. (2023). Tinjauan Literatur Terhadap Pemanfaatan Cloud Computing. *JURIHUM: Jurnal Inovasi Dan Humaniora*, *01*(01). https://jurnalmahasiswa.com/index.php/jurihum

Oktaviani, D., Papilaya, F. S., & Tanaem, F. (2021). *Perancangan Aplikasi E-Menu Restaurant dengan Menggunakan Cloud Computing dan Serverless Architecture Lambda*. *12*.

Sitompul, S. C., Jamaluddin, Simamora, R. J., & Perangin-angin, R. (2019). Aplikasi Pengaduan Masyarakat Berbasis Mobile Web di Kecamatan Tarutung. *METHOMIKA: Jurnal Manajemen Informatika & Komputerisasi Akuntansi, 3*(2), 136–142.

Sugiarti, Y., Sari, Y., & Hadiyat, M. A. (2020). Peranan E-Commerce Untuk Meningkatkan Daya Saing Usaha Mikro Kecil Dan Menengah (Umkm) Sambal Di Jawa Timur. *Kumawula: Jurnal Pengabdian Kepada Masyarakat*, *3*(2).