INFORMATION SYSTEM DEVELOPMENT
VOLUME 7 NO.1 (2022) | p27-33
e-ISSN 2528-5114 | p-ISSN 2477-863X
http://dx.doi.org/10.19166/xxxx

PERANCANGAN SISTEM PENGUMPULAN TREN HARGA PRODUK BERBASIS ANDROID DARI TOKOPEDIA

Ricky[1)], Frans Panduwinata[2*)], Dion Krisnadi[3)]

[1,2,3]Informatika, Fakultas Ilmu Komputer, Universitas Pelita Harapan
E-mail: [1]rg0025@student.uph.edu, [2]frans.panduwinata@uph.edu, [2]dion.krisnadi@uph.edu

Abstract – Online marketplace is one of the technology development which is helpful potential buyer to search intended product. Developer always try to make shopping process become easier and informative. However, there is one important feature that is not implemented by developer, namely product price movement trend presentation feature. Therefore, a system is made for present price movement trend information to user. The system consists of three parts, namely database (cloud firestore), Python based backend application and Android based client application. Testing use black box method and questionnaire. Black box testing result shows that all system function already running according to expectation. User can view product price movement trend information, track product and create new product tracking request from client application. Questionnaire result shows score 82.50% (help shopping process), 86.25% (easy to understand given information), 85% (information completeness), 91.25% (application interface), 97.5% (application running smoothness) dan 80% (system feasibility for public).

***Keywords**: Online Marketplace, Android Application*

Abstrak – *Marketplace online merupakan salah satu perkembangan teknologi yang membantu calon pembeli untuk mencari produk yang diinginkan. Pengembang selalu berusaha untuk membuat proses belanja menjadi mudah dan informatif. Namun, terdapat satu fitur penting yang tidak diimplementasi oleh pengembang, yaitu fitur penyajian informasi tren pergerakan harga produk. Maka, sebuah sistem dibuat untuk menyajikan informasi tren pergerakan harga kepada pengguna. Sistem terdiri dari tiga bagian, yaitu basis data (cloud firestore), aplikasi backend berbasis Python dan aplikasi klien berbasis Android. Pengujian menggunakan metode black box dan kuesioner. Hasil pengujian black box menunjukkan bahwa seluruh fungsi sistem sudah berjalan sesuai dengan harapan. Pengguna dapat melihat informasi tren pergerakan harga produk, melacak produk dan membuat permintaan pelacakan produk baru dari aplikasi klien. Hasil kuesioner menunjukkan skor 82.50% (membantu proses belanja), 86.25% (kemudahan memahami informasi yang diberikan), 85% (kelengkapan informasi), 91.25% (tampilan aplikasi), 97.5% (kelancaran aplikasi berjalan) dan 80% (kelayakan sistem untuk masyarakat luas).*

Kata Kunci: *Online Marketplace, Android Application*

PENDAHULUAN

Dewasa ini, ada banyak perkembangan teknologi yang dapat mempermudah kehidupan manusia, salah satunya adalah *e-commerce*. Khususnya di Indonesia, salah satu bentuk *e-commerce* [1][2] yang paling banyak dikenal adalah *marketplace* daring. Pada *marketplace* daring, pengguna bisa dengan mudah membuat toko daring dan langsung mempromosikan barangnya untuk dijual. Juga sebaliknya, pengguna bisa mencari barang yang ingin dibeli hanya dalam hitungan menit. Selain itu, para pembeli mempunyai kebebasan dan kemudahan yang lebih ketika melakukan proses belanja melalui *marketplace* daring, sehingga hal ini meningkatkan kepuasan pembeli.

Para pengembang *marketplace* daring selalu berusaha untuk mempermudah segala proses *online shopping* bagi pengguna, pada kasus ini khususnya untuk pembeli. Dimulai dari proses pencarian produk, penyajian detail produk, proses pembayaran, hingga proses pengiriman barang sudah dirancang oleh para pengembang untuk menciptakan pengalaman *online shopping* yang lancar. Namun, terdapat satu hal penting yang seharusnya bisa ditambahkan pada saat proses pencarian produk, yaitu penyajian informasi tentang tren pergerakan harga produk.

Sayangnya, salah 1 situs *marketplace* daring terbesar di Indonesia yaitu, Tokopedia yang memiliki peringkat Alexa 14 di Indonesia [3], masih belum menyajikan informasi tren pergerakan harga untuk setiap produknya. Padahal, penyajian informasi produk yang baik merupakan hal yang sangat penting pada saat calon pembeli akan menentukan pembelian [4]. Oleh karena itu, sistem pengumpulan data harga produk berbasis Python [5] yang menggunakan basis data NoSQL [6][7] Cloud Firestore [8] dan juga aplikasi Android [9][10] akan dikembangkan pada penelitian ini.

METODOLOGI PENELITIAN

Aplikasi *client* akan dirancang untuk perangkat dengan sistem operasi Android dengan versi minimal 5.0 yaitu

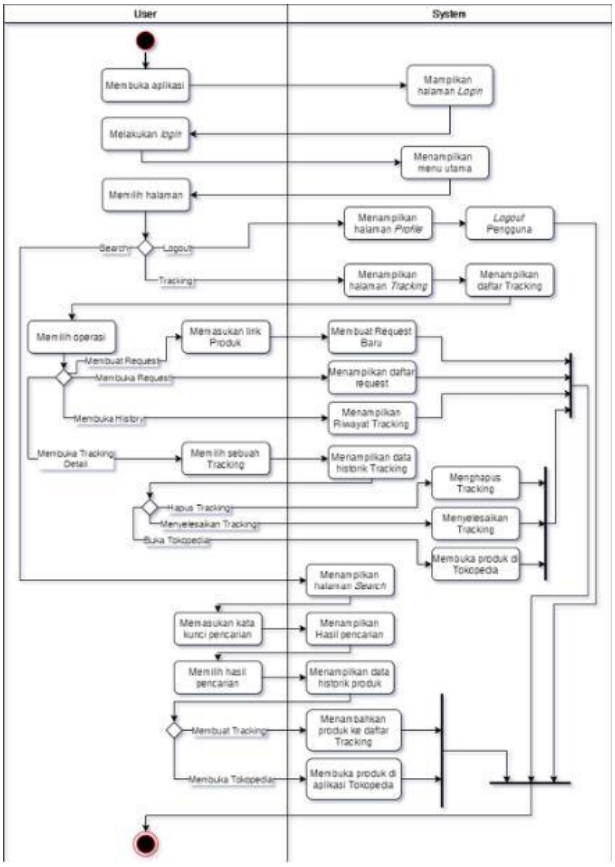
Lollipop. Basis data akan dirancang menggunakan Cloud Firestore dengan SDK versi 22.0.0 untuk Android dan SDK versi1.9.0 untuk Python. Data akan diambil dari Tokopedia. Data merupakan data produk yang terdiri dari nama produk, URL gambar produk, nama toko, harga produk, jumlah terjual, jumlah stok, jumlah dilihat, jumlah review, dan skor review. Jumlah data yang diambil selama penelitian adalah 50 produk selama 6 bulan dengan pengambilan data dilakukan satu kali setiap hari. Pengambilan data akan diimplementasikan dengan bahasa pemrograman Python dengan modul *requests* dan Beautiful Soup 4. Sistem pengambilan data akan ditempatkan pada sebuah Raspberry Pi versi 0w. Pengujian menggunakan metode *black box testing* [11].

HASIL DAN PEMBAHASAN

Bagian ini akan membahas proses umum aplikasi *client*, proses *create* dan *update* pada aplikasi *backend*, implementasi basis data, tampilan antar muka aplikasi *client* dan hasil uji coba aplikasi

Proses Umum Aplikasi *Client*

Pertama pengguna akan diminta untuk *login* menggunakan akun Google mereka. Setelah itu, pengguna akan diarahkan ke halaman utama, di mana akan ditampilkan total selisih harga yang telah didapat dan juga total transaksi yang telah dilakukan. Selanjutnya pengguna dapat memilih tiga produk yang sedang dilacak (Tracking). Pengguna dapat mengakses halaman "Tracking Detail" dengan memilih salah satu Tracking dari daftar, yang kemudian akan menampilkan data historik mengenai Tracking tersebut. Pada halaman ini, pengguna dapat memilih untuk menyelesaikan Tracking, menghapus Tracking atau mengakses halaman produk di Tokopedia. Selain itu, pada halaman "Tracking", pengguna juga dapat membuat "Tracking Request" untuk menambahkan produk baru yang ingin mereka lacak dengan memasukkan link halaman produk dari Tokopedia. Selain cara ini, pengguna juga dapat membuat "Tracking Request" langsung dari aplikasi Tokopedia, dengan cara masuk ke halaman produk dan memilih "Share / Bagikan Link" dan memilih aplikasi klien pada daftar aplikasi. Kemudian, *link* produk yang diambil melalui "Share Intent" akan digunakan oleh aplikasi untuk membuat "Tracking Request". Pengguna juga dapat melihat riwayat "Tracking Request" yang telah dibuat dan riwayat "Tracking" yang telah selesai dilakukan selama ini.



Gambar 1. *Activity Diagram* Proses Umum Aplikasi *Client*

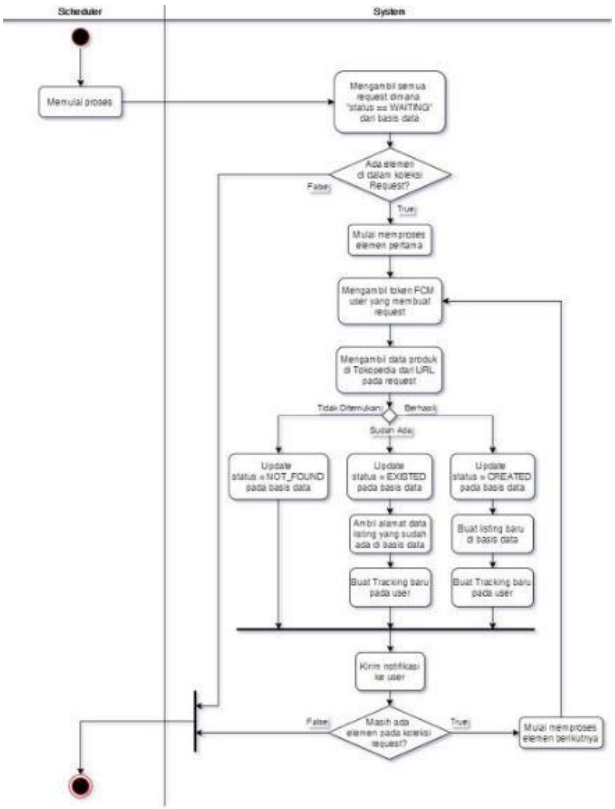
Pada halaman "Search", pengguna dapat melakukan pencarian sebuah produk yang telah tersimpan di dalam basis data. Setelah pengguna memasukkan kata kunci untuk dicari, daftar hasil pencarian produk akan ditampilkan. Dari daftar ini, pengguna dapat memilih salah satu produk untuk masuk ke dalam halaman "Search Detail". Pada halaman "Search Detail", data historik produk akan ditampilkan. Pada halaman ini, pengguna dapat memulai melacak produk atau mengakses halaman produk di Tokopedia. Pada halaman "Profile", pengguna dapat melihat data mengenai total selisih data yang telah didapat dan total jumlah transaksi yang telah dilakukan selama ini. Gambar 1 menunjukkan *activity diagram* untuk proses umum aplikasi *client*.

Proses *Create* pada Aplikasi *Backend*

Proses akan dijalankan secara berkala setiap 5 menit oleh *scheduler*. Lalu aplikasi akan meng-*query* semua *request* yang memiliki kode status "WAITING". Status ini menandakan bahwa *request* tersebut baru saja dibuat oleh pengguna aplikasi klien dan belum diproses oleh aplikasi *backend*. Untuk setiap *request*, aplikasi *backend* akan meng-*query* data produk ke *server* Tokopedia dengan URL yang ada di dalam dokumen *request*. Aplikasi juga akan mengambil data token FCM pembuat *request* untuk digunakan kemudian dalam pengiriman notifikasi.

Diterima <02122021>, Revisi <20122021>, Diterima untuk publikasi <28012022>. Copyright © 2022 Published by Universitas Pelita Harapan PSDKU Medan Jurusan Sistem Informasi, ISSN: 2528-5114

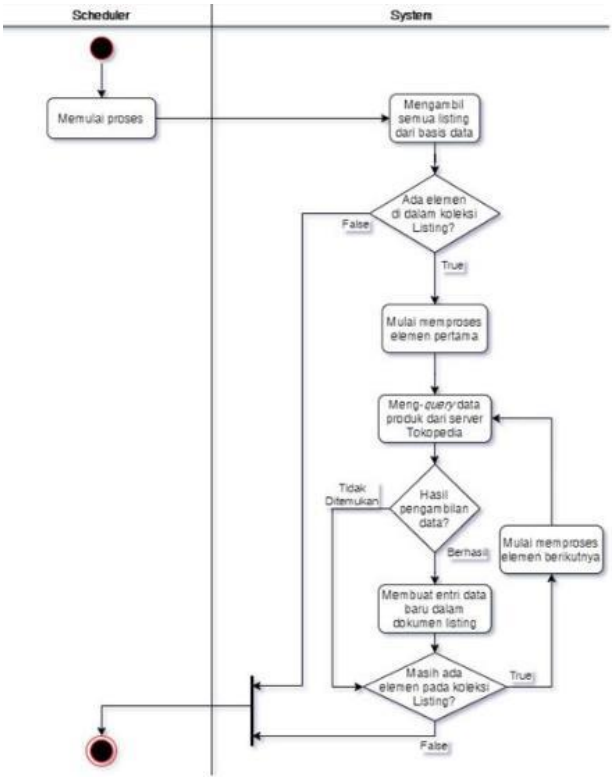
Terdapat tiga kemungkinan status hasil pengambilan data: “Not Found”, “Existed” dan “Created”. “Not Found” menandakan bahwa proses pengambilan data tidak membuahkan hasil. Hal ini dapat disebabkan karena produk yang dicari sudah dihapus atau *link* yang diberikan tidak tepat. Aplikasi *backend* akan meng-*update* status *request*. “Existed” menandakan bahwa produk yang ingin dicari sudah pernah dibuat sebelum pada basis data. Aplikasi *backend* akan meng-*update* status *request* menjadi “Existed”. Kemudian, aplikasi akan menggunakan data dari *listing* yang sudah ada di basis data untuk membuat *tracking* baru untuk pengguna yang membuat *request* ini. “Created” menandakan bahwa proses pengambilan data produk telah berhasil dilakukan tanpa masalah. Aplikasi *backend* akan meng-*update* status *request* menjadi “Created”. Kemudian, aplikasi akan membuat *listing* dari produk yang dicari di dalam basis data. Lalu, aplikasi akan menggunakan data dari *listing* baru tersebut untuk membuat *tracking* baru untuk pengguna yang membuat *request* ini. Gambar 2 menunjukkan *activity diagram* untuk proses *create* aplikasi *backend*.



Gambar 2. *Activity Diagram* Proses *Create* Aplikasi *Backend*

Proses *Update* pada Aplikasi *Backend*

Proses akan dijalankan secara berkala setiap 24 jam oleh *scheduler*. Lalu, aplikasi akan meng-*query* semua *listing* yang ada pada basis data. Untuk setiap *listing* tersebut, aplikasi *backend* akan meng-*query* data produk ke *server* Tokopedia. Apabila hasil pengambilan data berhasil, maka aplikasi akan membuat entri data baru dalam dokumen *listing* di basis data. Apabila gagal, maka aplikasi akan melanjutkan ke *listing* berikutnya. Gambar 3 menunjukkan *activity diagram* untuk proses *update* aplikasi *backend*.



Gambar 3. *Activity Diagram* Proses *Update* Aplikasi *Backend*

Spesifikasi

Implementasi Basis Data

Naskah menggunakan tipe huruf Times New Roman dengan ukuran huruf seperti yang telah dicontohkan pada panduan penulisan ini. Jarak spasi adalah *single* dan isi tulisan atau naskah menggunakan perataan kiri-kanan *(justified).* Harap memperhatikan tulisan asing untuk dilakukan cetak miring *(Italic).*

Tabel 1. Tabel Daftar *Collections* pada Firestore

Nama *Collection*	Deskripsi
Listing	Menyimpan data produk berupa nama produk, link produk, listing ID, data historik terakhir dan lain-lain. Juga menyimpan *collections* data yang menyimpan seluruh data historik produk (harga, jumlah terjual, jumlah dilihat, dan lain-lain)
Scraper	Menyimpan data *request*, yaitu permintaan pengguna untuk memulai melacak produk baru. Data yang disimpan adalah *link* produk, kode status *request*, *timestamp request* dibuat, *timestamp request* diproses, dan

Diterima <02122021>, Revisi <20122021>, Diterima untuk publikasi <28012022>. Copyright © 2022 Published by Universitas Pelita Harapan PSDKU Medan Jurusan Sistem Informasi, ISSN: 2528-5114

	ID pengguna yang membuat *request*.
Users	Menyimpan data pengguna yaitu nama pengguna, total selisih harga yang didapat, jumlah transaksi dan *array* yang berisi *id* produk yang sedang dilacak. Juga menyimpan *collections activeTracking* yang menyimpan data mengenai produk yang sedang dilacak.

Tabel 2 menunjukkan data yang disimpan oleh *collections listing*.

Tabel 2. Tabel *Detail* Data pada *Collection Listing*

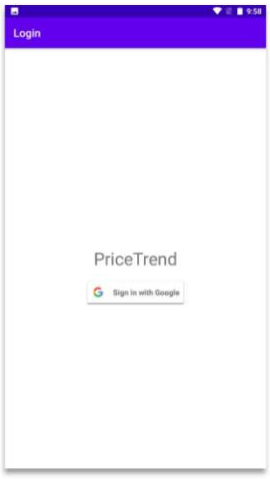
Field Name	Tipe Data	Deskripsi
listingID	*String*	Menyimpan data *ID listing*
listing Name	*String*	Menyimpan data nama dari produk
listing Thumb URL	*String*	Menyimpan data *link* URL dari foto *thumbnail* produk
listingURL	*String*	Menyimpan data *link* URL dari Produk
storeName	*String*	Menyimpan nama dari toko yang menjual produk
tags	*Array of String*	Menyimpan data kata kunci pencarian dari produk
latestData	*Map*	Menyimpan satu baris data historik terakhir dari produk
data	*Collection*	Menyimpan data historik dari produk

Tabel 3 menunjukkan data yang disimpan oleh *collections scraper*.

Tabel 4 menunjukkan data yang disimpan oleh *collections users*.

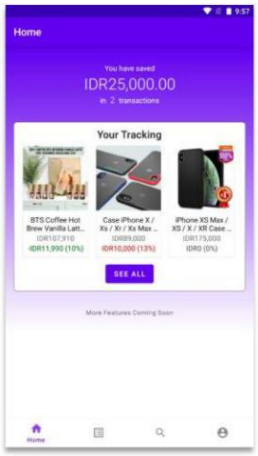
Tampilan Antar Muka Aplikasi *Client*

Aplikasi berbasis Android yang digunakan oleh pengguna diimplementasi dengan menggunakan bahasa pemrograman Kotlin. Pada awal aplikasi, akan ditampilkan halaman *Login* (Gambar 4).



Gambar 4. Tampilan Halaman *Login*

Setelah melakukan *login*, pengguna masuk ke dalam halaman *Home* (Gambar 5).



Gambar 5. Tampilan Halaman *Home*

Dari halaman Home, pengguna dapat mengakses halaman-halaman lainnya melalui *navigation bar*. Pada halaman *Tracking* (Gambar 6), pengguna dapat melihat informasi mengenai *tracking* apa yang sedang mereka miliki.

Gambar 6. Tampilan Halaman *Tracking*

Diterima <02122021>, Revisi <20122021>, Diterima untuk publikasi <28012022>. Copyright © 2022 Published by Universitas Pelita Harapan PSDKU Medan Jurusan Sistem Informasi, ISSN: 2528-5114

Dari halaman *Tracking*, pengguna dapat mengakses halaman *Request* dan *History*. Pada halaman *Request* (Gambar 7) ditampilkan daftar *request* yang telah pengguna buat selama ini.

Gambar 7. Tampilan Halaman *Request*

Pada halaman *History* (Gambar 8) ditampilkan daftar *tracking* yang telah pengguna selesaikan.

Gambar 8. Tampilan Halaman *History*

Ukuran Halaman

Ukuran halaman adalah A4 (210 mm x 297 mm). Khusus di bagian judul dari artikel, *margin* halaman adalah 30 mm atas, 25 mm bawah-kiri, dan 17 mm kanan. Pada halaman selanjutnya, *margin* halaman adalah 25 mm atas-bawah-kiri dan 17 mm kanan. Lebar kolom adalah 80,5 mm and lebar *gutter* (jarak antar kolom) adalah 7 mm.

Layout Naskah

Penulisan *paper* harus mengikuti layout naskah seperti yang ada pada *template* ini. Untuk memudahkan pembuatan *paper*, penulis dapat langsung menggunakan *template* panduan ini.

Tabel 3. Tabel *Detail* Data pada *Collection Scraper*

Field	Tipe	Deskripsi
url	String	Menyimpan data link URL yang diberikan pengguna untuk diproses
status Code	*Number*	Menyimpan data kode status dari pemrosesan *request*
requests	*Time stamp*	Menyimpan data waktu dari pembuatan *request*
respon seTs	*Time stamp*	Menyimpan data waktu dari penyelesaian pemrosesan *request*
listing URL	*String*	Menyimpan data *link* URL dari produk di Tokopedia
listing Doc Addr	*String*	Menyimpan alamat tempat data *listing* yang diproses tersimpan
user	*Map*	Menyimpan data ID pengguna yang membuat *request*

Tabel 4. Tabel *Detail* Data pada *Collection Users*

Field Name	Tipe Data	Deskripsi
display Name	*String*	Menyimpan data nama dari pengguna
total Saving	*Number*	Menyimpan data total selisih harga yang telah didapatkan
trx Count	*Number*	Menyimpan data jumlah transaksi yang telah dilakukan
fcm Token	*String*	Menyimpan data token Firebase Cloud Messaging milik pengguna
activeTracking Metadata	*Array of String*	Menyimpan *listingID* dari setiap *tracking* yang dimiliki pengguna
active Trackings	*Collection*	Menyimpan data *tracking* yang dimiliki pengguna
savings History	*Collection*	Menyimpan data *tracking* yang sudah diselesaikan pengguna

Pada halaman *Search* (Gambar 9), pengguna dapat melakukan pencarian produk yang sudah tersimpan pada basis data.

Diterima <02122021>, Revisi <20122021>, Diterima untuk publikasi <28012022>. Copyright © 2022 Published by Universitas Pelita Harapan PSDKU Medan Jurusan Sistem Informasi, ISSN: 2528-5114

Gambar 9. Tampilan Halaman *Search*

Hasil Uji Coba Aplikasi

Penelitian ini menggunakan dua metode untuk menguji sistem yang telah dibuat, yaitu metode *black box* dan metode kuesioner. Kuesioner dibagikan kepada responden. Pengujian *black box* dilakukan oleh para responden kuesioner sebelum responden mengisi kuesioner. Hasil pengujian *black box* dapat dilihat pada Tabel 5.

Tabel 5. Tabel Hasil Pengujian *Black Box*
<belum bisa paste data table-nya>

Berdasarkan hasil pengujian *black box* dapat dilihat bahwa semua hasil aktual yang didapat sesuai dengan hasil yang diharapkan. Pengujian berikutnya adalah pengujian dengan metode kuesioner. Pengujian ini dilakukan untuk mengetahui tanggapan dan penilaian responden. Pernyataan yang diberikan kepada responden dapat dilihat pada Tabel 6.

Tabel 6. Tabel Pernyataan Kuesioner

No.	Pernyataan
1.	Informasi yang diberikan sistem dapat membantu dalam proses belanja
2.	Fitur-fitur yang ada dapat mudah dipahami
3.	Informasi produk yang disajikan sudah lengkap
4.	Aplikasi memiliki tampilan yang menarik
5.	Aplikasi dapat berjalan dengan lancar pada device anda
6.	Sistem sudah layak digunakan untuk masyarakat luas

Responden memberikan satu dari lima tanggapan untuk setiap pernyataan yang diberikan. Setiap pilihan tanggapan memiliki skala penilaian dari 0 (terendah) hingga 4 (tertinggi). Tabel 7 menampilkan daftar pilihan tanggapan beserta skor.

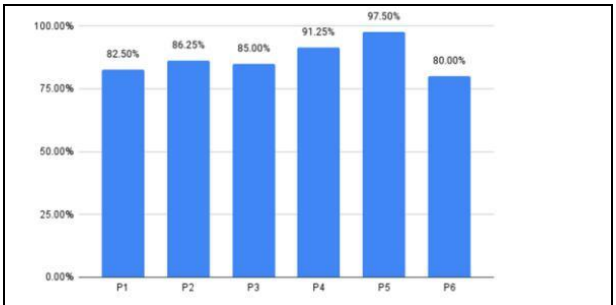
Tabel 7. Tabel Daftar Pilihan Tanggapan Kuesioner

Tanggapan	Skor
Sangat Setuju	1
Setuju	2
Netral	3
Tidak Setuju	4
Sangat Tidak Setuju	5

Dari hasil kuesioner ini dilakukan perhitungan skor akhir dari setiap pernyataan menggunakan rumus yang bisa dilihat pada Persamaan (1). Dengan nilai ini, maka dapat ditarik kesimpulan penilaian dari aplikasi.

$$= \quad 100\% \qquad (1)$$

p merupakan nilai skor akhir dari sebuah pernyataan, total skor merupakan jumlah skor sebuah pernyataan dari seluruh responden, skor maksimal merupakan skor tertinggi yang bisa diberikan kepada sebuah pernyataan yaitu 4 dan n adalah jumlah dari responden. Hasil skor akhir dari setiap pernyataan dapat dilihat pada Gambar 10.



Gambar 10. Hasil Skor Akhir Kuesioner

Pada Gambar 10, dapat dilihat bahwa pernyataan pertama mengenai informasi yang diberikan dapat membantu dalam proses belanja mendapatkan skor akhir 82.50%. Pernyataan kedua mengenai kemudahan fitur untuk dipahami mendapatkan skor 86.25%. Pernyataan ketiga mengenai kelengkapan informasi produk mendapatkan skor akhir 85%. Pernyataan keempat mengenai tampilan aplikasi mendapatkan skor akhir 91.25%. Pernyataan kelima mengenai aplikasi dapat berjalan dengan lancar mendapatkan skor akhir 97.5%. Pernyataan terakhir mengenai kelayakan sistem untuk masyarakat luas mendapatkan skor akhir 80%.

KESIMPULAN

Berdasarkan hasil pengujian dan analisis terhadap sistem agregasi harga produk dari online marketplace, dapat disimpulkan bahwa:

1. Sistem yang sudah dibuat telah menjawab semua rumusan masalah dengan cara membuat sistem yang terdiri dari tiga bagian. Bagian pertama adalah sistem pengumpul data berbasis Python yang mengambil data produk dari server Tokopedia setiap harinya dan menyimpan data tersebut ke basis data. Sistem pengumpul data ini ditempatkan di sebuah Raspberry Pi 0w yang menggunakan cron untuk menjadwalkan pembaruan data produk. Bagian kedua adalah basis data yang diimplementasi dengan Firestore yang dapat menampung data produk dan pengguna.

Diterima <02122021>, Revisi <20122021>, Diterima untuk publikasi <28012022>. Copyright © 2022 Published by Universitas Pelita Harapan PSDKU Medan Jurusan Sistem Informasi, ISSN: 2528-5114

Bagian terakhir adalah aplikasi klienberbasis Android yang telah diimplementasi dengan Kotlin. Aplikasi klien menggunakan Firebase Android SDK yang disediakan oleh Google untuk membaca dan menyimpan data pada basis data.

2. Sistem mendapatkan respon positif dari responden kuesioner. Skor akhir tertinggi sebesar 97.50% tercatat pada aspek kelancaran berjalannya aplikasi. Skor terendah sebesar 80% tercatat pada aspek kelayakan sistem untuk masyarakat luas.
3. Berdasarkan pengujian *black box*, aplikasi *backend* terbukti dapat memproses *request* pelacakan produk baru dan juga memperbarui data historik produk yang sudah ada; aplikasi klien terbukti dapat menampilkan data pengguna dan melakukan berbagai operasi, seperti menyelesaikan *tracking*, menghapus *tracking*, memulai *tracking* dan lainnya.

DAFTAR PUSTAKA

[1] Goetsch, Kelly. ECommerce in thecloud:bringingelasticitytoeCommerce. Sebastopol: O'Reilly Media, 2014.

[2] Tian, Lin, Asoo Vakharia, Yinliang Tan, and Yifan Xu. "Marketplace, Reseller, or Hybrid: Strategic Analysis of an Emerging E-Commerce Model."Production and Operations Management, 2018: 1595-1610.

[3] Alexa. Alexa - Top Sites in Indonesia. 2020. https://www.alexa.com/topsites/countries/ID (diakses 2 Desember 2021)

[4] Chen, Xiayu, Qian Huang, and Robert Davison. "The role of website quality and social capital in building buyers' loyalty." International Journal of Information Management, 2017: 1563-1574.

[5] Downey, Allen B. Think Python. Sebastopol: O'Reilly Media, 2015.

[6] Fowler, Martin, and Pramod J Sadalage. NoSQL distilled. Boston: Addison-Wesley, 2012.

[7] Sullivan, Dan. NoSQL for mere mortals. Upper Saddle River, NJ: Pearson Education, 2015.

[8] Google.Cloud Firestore.2020.https://firebase.google.com/docs/firestore (diakses 2 Desember 2021).

[9] Lou, Tian. A Comparison of Android Native App Architecture - MVC, MVP and MVVM. Master's Thesis, Eindhoven: Eindhoven University of Technology, 2016.

[10] Mohammadpur, Davud, and Ali

[11] Mahjur. "Separation of collection concern." ksii transactions on internet and information systems, 2011: 135-147.

[12] Khan, Ehmer, and Farmeena Khan. "A Comparative Study of White Box, Black Box and Grey Box Testing Techniques." International Journal of Advanced Computer Science and Applications, 2012: 12-15

Diterima <02122021>, Revisi <20122021>, Diterima untuk publikasi <28012022>. Copyright © 2022 Published by Universitas Pelita Harapan PSDKU Medan Jurusan Sistem Informasi, ISSN: 2528-5114