Performa Algoritma *Bubble Sort* Dan *Quick Sort* Pada *Framework Flutter* Dan *Dart* SDK (Studi Kasus Aplikasi E-Commerce)

Dendi Rizka Poetra*[1], Nur Hayati[2]
[1,2]Universitas Nasional; Jl. Sawo Manila No.61, RT.14/RW.7, Pejaten Bar., Kec. Ps. Minggu, Kota Jakarta Selatan, Daerah Khusus Ibukota Jakarta 12520, Telp. (021) 7806700
[3]Fakultas Teknologi Komunikasi dan Informatika, FTKI UNAS, Jakarta
e-mail: ***[1]dendirizkapoetra2018@student.unas.ac.id**, [2]nurhayati@civitas.unas.ac.id

Abstrak

Pengurutan data merupakan salah satu teknik dalam struktur data, dan struktur data tentu memiliki berbagai macam algoritmanya. Pada aplikasi e-commerce tentu sangat membutuhkan algoritma pengurutan data yang baik dan efisien dari segi performa kecepatan dan juga akurasi keakuratan pengurutan data. Pada penelitian ini dilakukan pengujian performa algoritma pengurutan data yaitu Algoritma Bubble Sort dan Quick Sort yang mana media pendukung pengujiannya adalah aplikasi e-commerce yang dibangun dengan framework Flutter dan Bahasa pemrograman Dart. Dataset yang digunakan diperoleh dari Tokopedia dengan keyword kacamata secara random. Hasil pengujian dari kedua algoritma yang telah dibagi menjadi 5 iterasi menunjukan bahwa terdapat perbedaan waktu eksekusi disemua iterasi, dimana algoritma bubble sort membutuhkan 0.315543 detik untuk 2000 data, sedangkan algoritma quick sort hanya membutuhkan waktu 0.011126 detik untuk 2000 data. Kedua algoritma tersebut memperoleh 100% akurasi disemua iterasinya dalam proses pengurutan data.

Kata kunci— Performa, Bubble Sort, Quick Sort, E-Commerce, Flutter, Dart.

Abstract

Sorting data is one of the techniques in data structures, and data structures certainly have various algorithms. In e-commerce applications, really need a good and efficient data sorting algorithm in terms of speed performance and also the accuracy of data sorting accuracy. In this study, the performance of the data sorting algorithm was tested, namely the Bubble Sort and Quick Sort algorithms where the supporting media for the test was an e-commerce application built with the Flutter framework and the Dart programming language The dataset used was obtained from Tokopedia with the keyword glasses randomly. The test results of the two algorithms that have been divided into 5 iterations show that there are differences in execution time in all iterations, where the bubble sort algorithm requires 0.315543 seconds for 2000 data, while the quick sort algorithm only takes 0.011126 seconds for 2000 data. Both algorithms obtain 100% accuracy in all iterations in the data sorting process.

Keywords— *Performance, Bubble Sort, Quick Sort, E-Commerce, Flutter, Dart*

1. PENDAHULUAN

Pada penelitian pertama dilakukan oleh (Arifin, R. W., dan Setiyadi, D. 2020). Menguji performa dari algoritma *bubble sort* dan *quick sort*, proses pengujiannya diimplementasikan pada Bahasa pemgrograman C++. Hasil dari penelitian tersebut membuktikan bahwa algoritma *quick sort* secara performa lebih baik dibanding algoritma *bubble sort* karena dari hasil analisa peneliti tersebut algoritma *quick sort* dalam penggunaan memori lebih sedikit dibandingkan algoritma *bubble sort* [1].

Penelitian kedua dilakukan oleh (Rahayuningsih, P. 2016). Menguji performa dari beberapa algoritma *sorting* yang mana terdapat algoritma *bubble sort* dan juga *quick sort*, penelitian ini diimplementasikan pada aplikasi desktop yang dibangun menggunakan Bahasa pemrograman Visual Basic 6.0. Hasil dari penelitian ini menjelaskan secara rinci bahwa algoritma *bubble sort* membutuhkan waktu 6.84 detik untuk mengurutkan 250 data, sedangkan algoritma *quick sort* hanya membutuhkan waktu 0.11 detik untuk mengurutkan 250 data [2].

Penelitian ketiga dilakukan oleh (Eko Saputro, F., Nidaul Khasanah, F. 2018). Menguji performa dari algoritma *bubble sort* yang diimplementasikan pada Bahasa *C++*. Hasil dari penelitian ini menjelaskan bahwa algoritma *bubble sort* membutuhkan waktu 0.094 detik untuk mengurutkan 100 data [3].

Penelitian keempat selanjutnya dilakukan oleh(Pratama, M. A., Desiani, A., dan Irmeilyana. 2017). Menguji performa dari beberapa algoritma *sorting* yang mana salah satunya adalah algoritma *quick sort*, proses pengujian diimplementasikan pada bahasa pemrograman *C++*. Hasil penelitian tersebut membuktikan bahwa algoritma *quick sort* membutuhkan waktu 0.08007 detik untuk mengurutkan 100 data [4].

Dalam dunia pemrograman tentu algoritma merupakan bagian terpenting yang tidak dapat dipisahkan. Dalam proses pembelajaran pemrograman hal yang paling dasar adalah memahami konsep dari algoritma. Secara garis besar algoritma merupakan susunan logis yang diurutkan secara sistematis yang digunakan untuk memecahkan sebuah permasalahan [3], [5]. Selain algoritma, struktur data juga merupakan bagian terpenting dalam dunia pemrograman. Salah satu metode struktur data adalah pengurutan atau *sorting* data. Pada saat ini terdapat berbagai macam metode pengurutan data yang memiliki kelebihan dan kekurangannya masing-masing [6]–[8].

Perkembangan teknologi dibidang bisnis semakin pesat, khususnya sistem penjualan *E-Commerce* yang sudah sangat banyak pada saat ini. *E-Commerce* dapat diartikan sebagai transaksi jual/beli melalui media elektronik yang terhubung oleh internet. lalu *e-commecers* juga menjadi alternatif dari sistem penjualan konvensional [9].

Metode pengurutan data merupakan salah satu algortima dengan metode dasar yang dapat diimplementasikan pada aplikasi *e-commerce*, seperti contoh untuk mengurutkan harga produk dilist produk dari yang termurah hingga termahal atau sebaliknya. Aplikasi *e-commerce* tentu memiliki data yang besar dengan jumlah yang tidak sedikit apabila pemilihan algoritma tidak tepat maka akan timbul masalah pada sisi *UX(User Experience)* akibat alur pengurutan yang tidak ringkas akan menyebabkan waktu tunggu saat memuat/load data yang tidak sebentar dikarenakan algoritma dari metode tersebut membutuhkan waktu untuk mengeksekusi data.

Saat ini ada berbagai macam metode pengurutan data yang dapat dipelajari dan diimplementasikan, untuk membatasi luas dari pembahasan ini penulis hanya akan membahas 2 metode pengurutan data, yaitu algoritma *bubble sort* dan algoritma *quick sort*. Pembahasan penelitian ini akan difokuskan untuk melakukan pengujian performa pada kedua metode pengurutan data pada aplikasi *e-commerce* dengan *framework flutter* dan Bahasa pemrograman *dart*.

Berdasarkan apa yang telah penulis uraikan diatas tentu kedua algoritma tersebut memiliki kekurangan dan kelebihannya masing-masing. Pada penelitian ini akan menggunakan aplikasi *e-commerce* sebagai media pendukungnya yang dibuat menggunakan *framework flutter* dan Bahasa pemrograman *dart*. Pengujian akan dibagi menjadi 5 iterasi dengan panjang data yang berbeda-beda. Adapun tujuan dari penelitian ini adalah untuk mengetahui performa dari kedua algoritma tersebut yang mana nantinya akan diketahui secara jelas performa dari segi kecepatan dan juga akurasi keakuratan dalam pengurutan data sesuai dengan panjang data dari masing masing iterasi.

2. METODE PENELITIAN

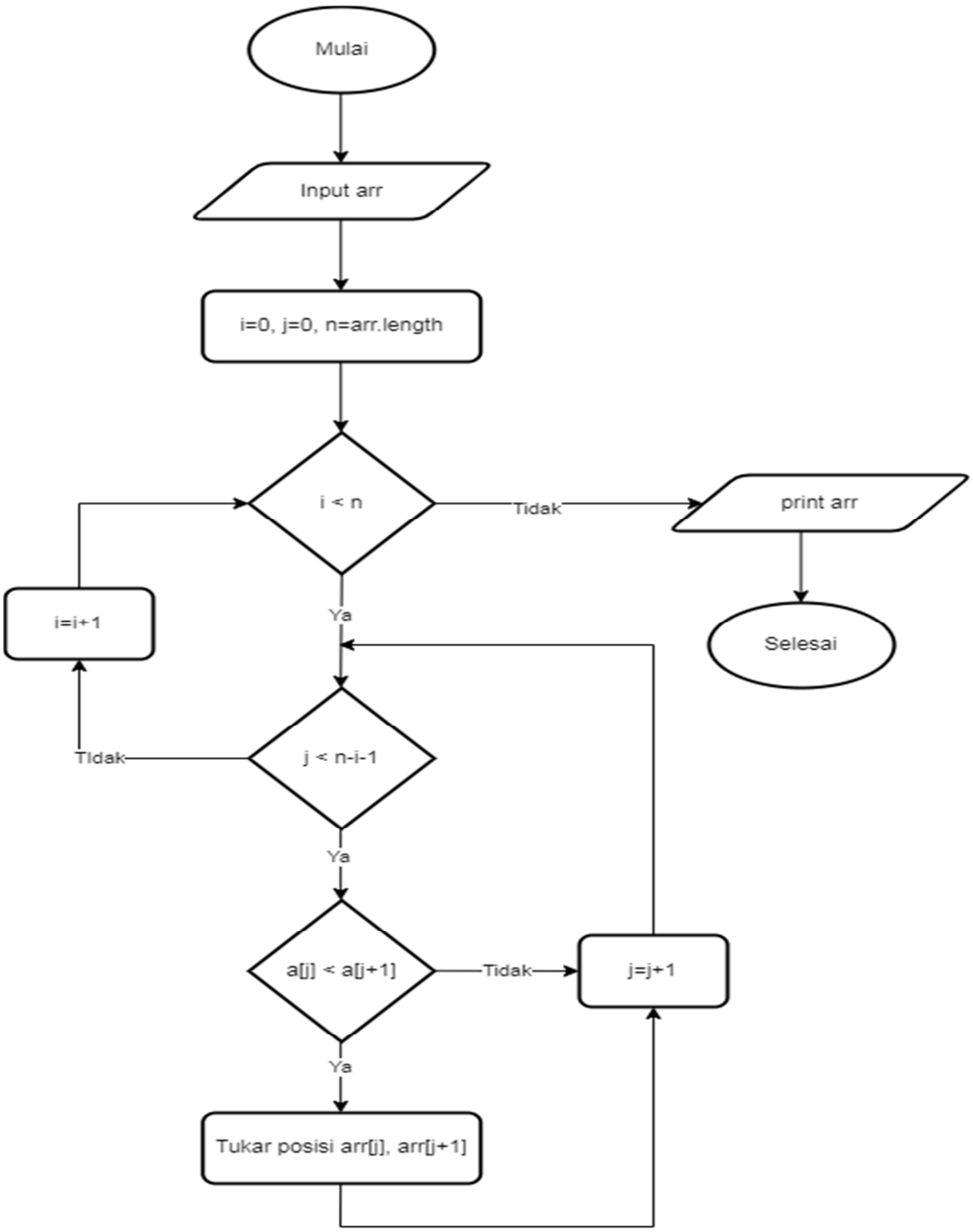
Terdapat 1 sumber data yang akan digunakan pada penelitian ini, sumber data yang digunakan dikumpulkan melalui *scaraping* data website resmi tokopedia menggunakan *web scaraper* dengan *keyword* pencarian kacamata. Jadi, data dikumpulkan dengan *keyword* kacamata secara random yang berhasil dikumpulkan total 2000 data yang kemudian digunakan untuk dataset pada penelitian ini.

Parameter yang akan digunakan adalah harga produk sebagai input datanya kedalam algoritma. Jadi, produk akan diurutkan sesuai dengan harga produknya masing-masing. Implementasinya akan disama ratakan pada saat pengujiannya dimedia pendukung penelitian yaitu aplikasi *e-commerce*. Kedua algoritma akan mengurutkan data harga produk secara *ascending* atau dari yang terkecil ke terbesar.

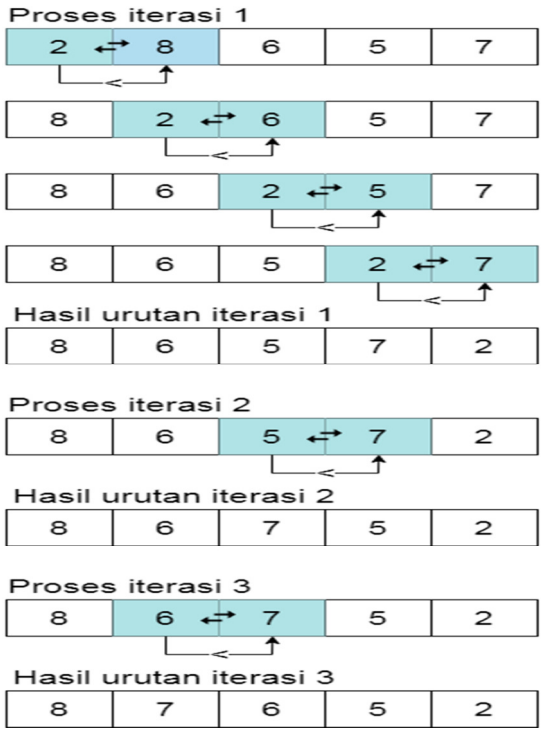
2.1 Algoritma Bubble Sort

Bubble Sort merupakan algoritma pengurutan data yang mana metodenya adalah dengan cara membandingkan elemen sekarang dengan elemen berikutnya, apabila elemen sekarang lebih besar/kecil dari elemen berikutnya maka akan dilakukan penukaran posisi(*index*), jika tidak maka tidak perlu menukar posisi(index) datanya. Untuk contoh kasus, n merupakan jumlah data didalam *array*, jika n = 10 maka akan dilakukan (n - 1) = 9 iterasi penukaran posisi(*index*) data(Mulai dari 0 hingga n – 2) [2], [10]–[12].

Pada algoritma ini metode yang dilakukan ialah membandingakan setiap elemen, lalu akan dilakukan penukaran apabila terdapat elemen yang urutannya belum sesuai. Metode ini akan terus dilakukan oleh algoritma ini hingga tidak ada lagi data yang belum sesuai urutannya. Berikut dibawah ini gambar 1 merupakan *flowchart* dari algoritma *Bubble Sort* dan gambar 2 merupakan simulasi dari algoritma *bubble sort*.



Gambar 1. Flowchart Algoritma Bubble Sort

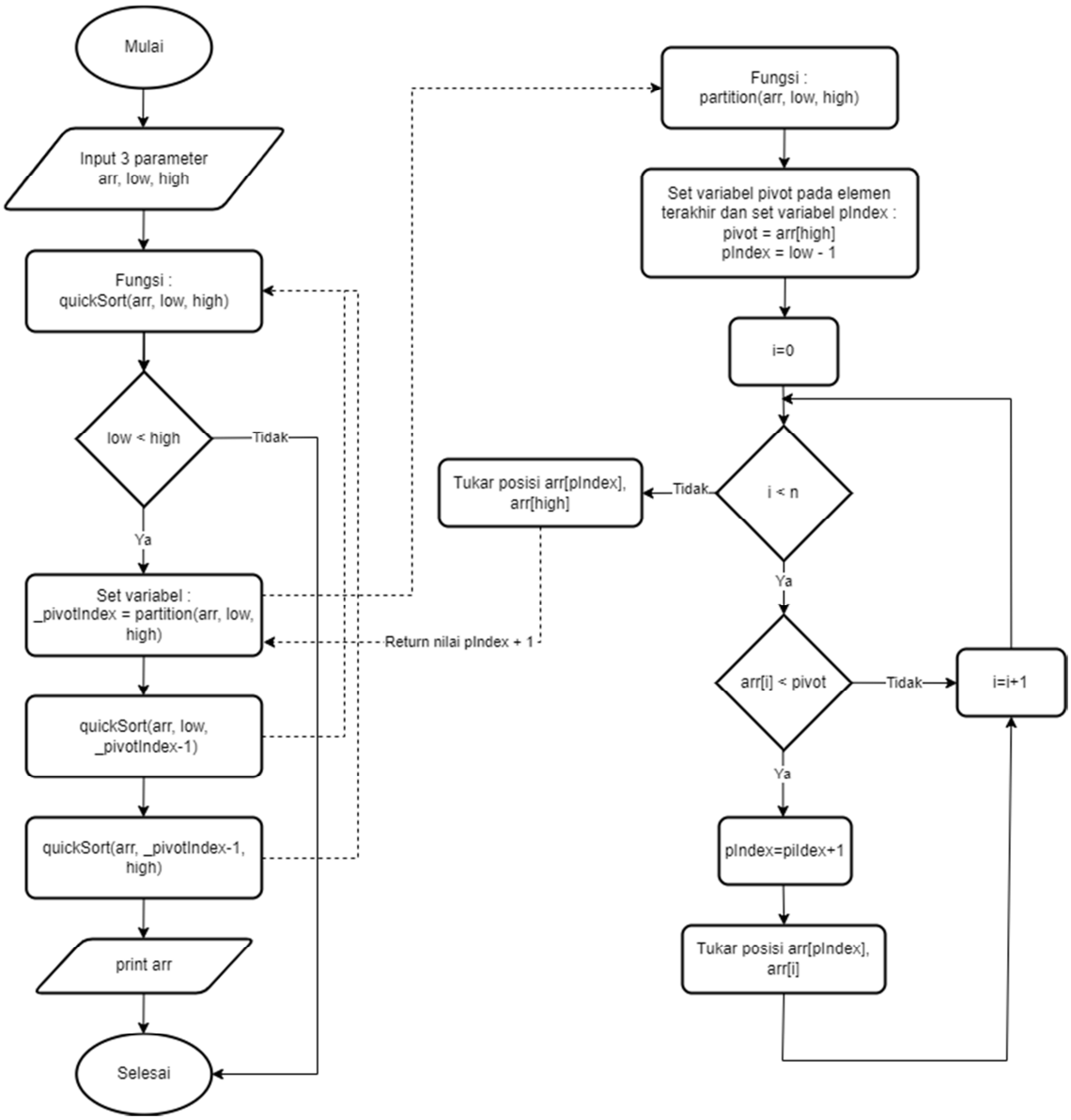


Gambar 2. Simulasi Algoritma Bubble Sort

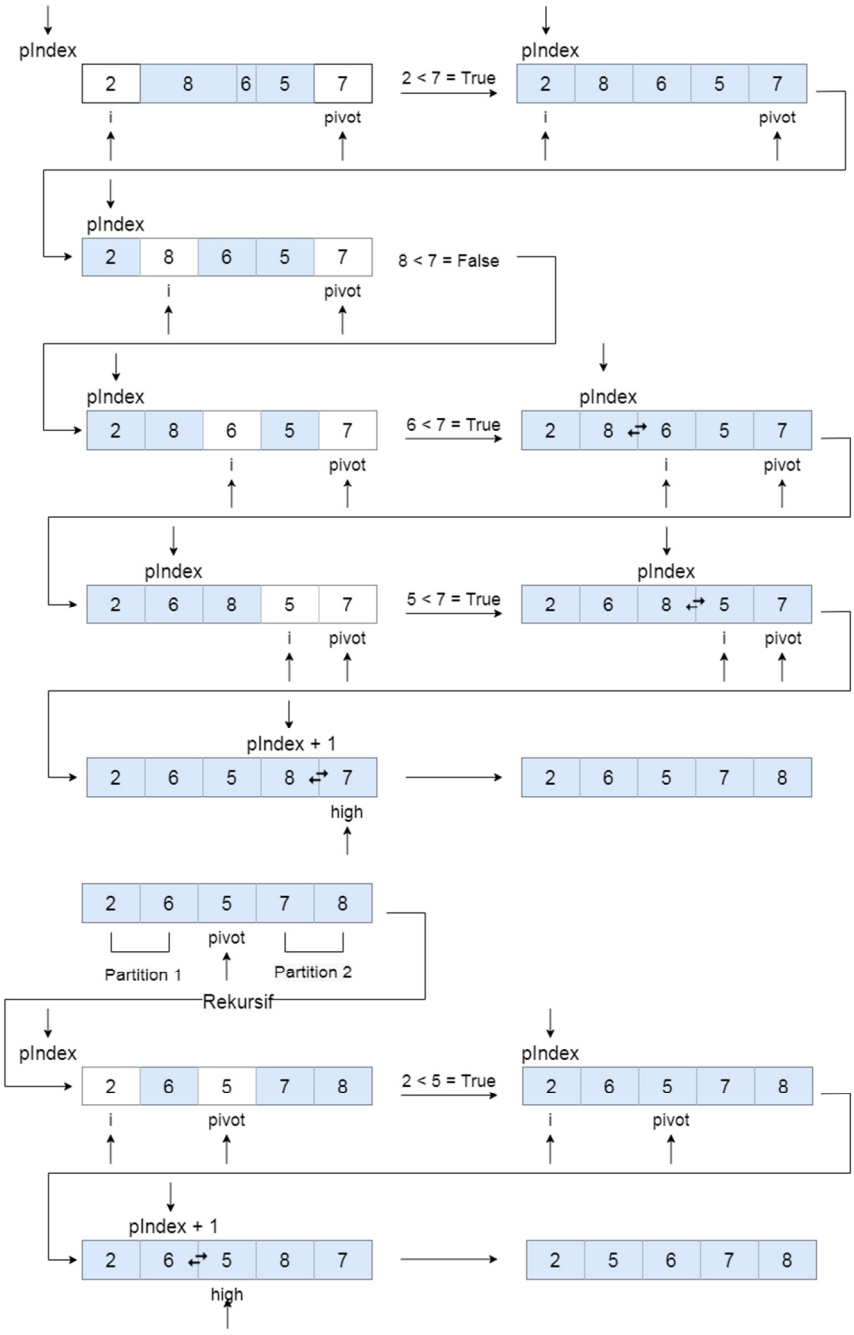
2.2 Algoritma Quick Sort

Quick Sort merupakan algoritma pengurutan data yang sangat bergantung pada elemen pivot dengan cara kerja metodenya dengan mereduksi tahap demi tahap sehingga menjadi 2 bagian yang lebih kecil. Cara kerja tersebut biasa disebut *Divide and Conquer*. Elemen angka atau n yang besar akan akan dipartisi menjadi dua *sub*-elemen yang mana salah satunya berisi elemen yang lebih kecil dari pivot. Kemudian quick sort akan memanggil dirinya sendiri secara rekursif untuk mengurutkan elemen angka [2], [10]–[12].

Algoritma ini merupakan algoritma yang sangat cepat jika dibandingkan dengan algoritma pengurutan data yang lain, dikarenakan algoritma ini melakukan pengurutan data dengan membagi masalah menjadi *sub* masalah dan sub masalah dibagi kembali menjadi sub-sub masalah yang mana menghasilkan pengurutan data yang lebih cepat. Pada gambar 2 merupakan *flowchart* dari algoritma *Quick Sort* dan gambar 3 merupakan simulasinya.



Gambar 3. Flowchart Algoritma Quick Sort



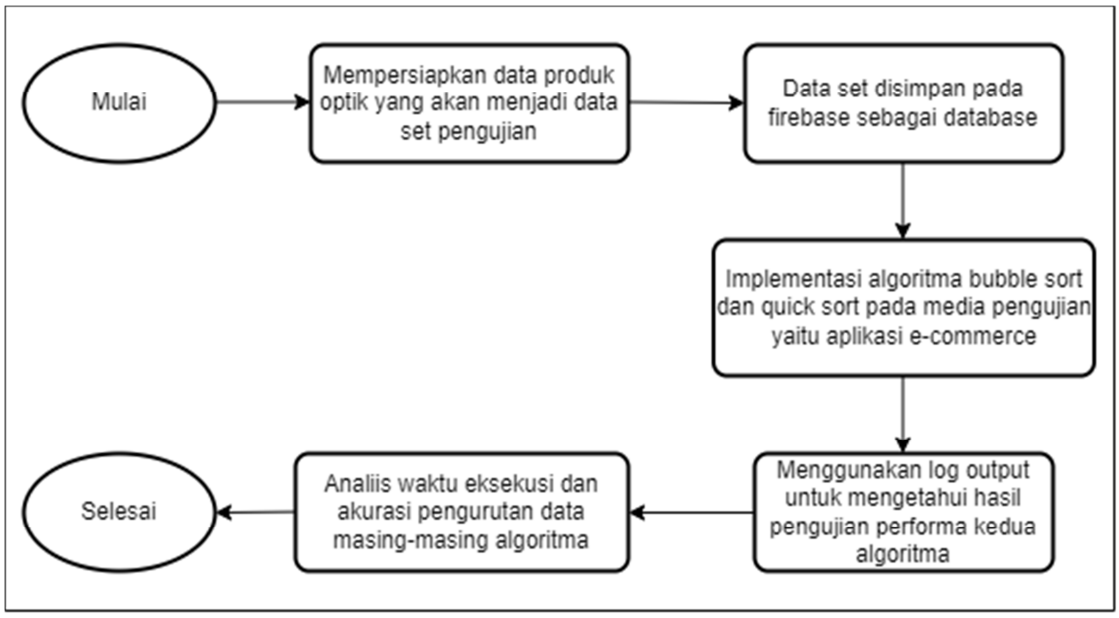
Gambar 4. Simulasi Algoritma Quick Sort

Untuk menghitung *error rate* yang akan menjadi hasil akhir akurasi dari setiap algoritma yang diuji dengan menggunakan rumus:

$$Error\ Rate\ \% = \frac{jumlah\ pengurutan\ data\ \ salah}{total\ jumlah\ pengurutan\ data} * 100 \quad (1)$$

$$Akurasi\ \% = 100\% - Error\ Rate\ \% \quad (2)$$

Adapun tahapan penelitian yang akan dilakukan adalah melakukan *scraping* data pada tokopedia dengan *keyword* kacamata yang menghasilkan data sebanyak 2000 data. Membuat program *backend* pendukung penelitian menggunakan *google colab* dengan Bahasa *python* untuk melakukan import 2000 data ke *firebase*. *Firebase* digunakan sebagai *database* aplikasi *e-commerce*. Setelah semua kebutuhan penelitian telah selesai disiapkan, maka selanjutnya akan dilakukan pengujian performa kedua algoritma sebanyak 5 iterasi dengan panjang data yang berbeda-beda disetiap iterasinya pada media pendukung yaitu aplikasi *e-commerce* yang dibangun menggunakan *framework flutter* dan Bahasa *dart*. Setelah itu mengambil *sample* pengujian aplikasi *e-commerce* lalu melampirkan *log output* waktu eksekusi kedua algoritma dan menghitung akurasi ketepatan pengurutan data. Langkah terakhir membuat tabel hasil pengujian dan membuat grafik histogram dari data hasil pengujian.



Gambar 5. Tahapan Penelitian

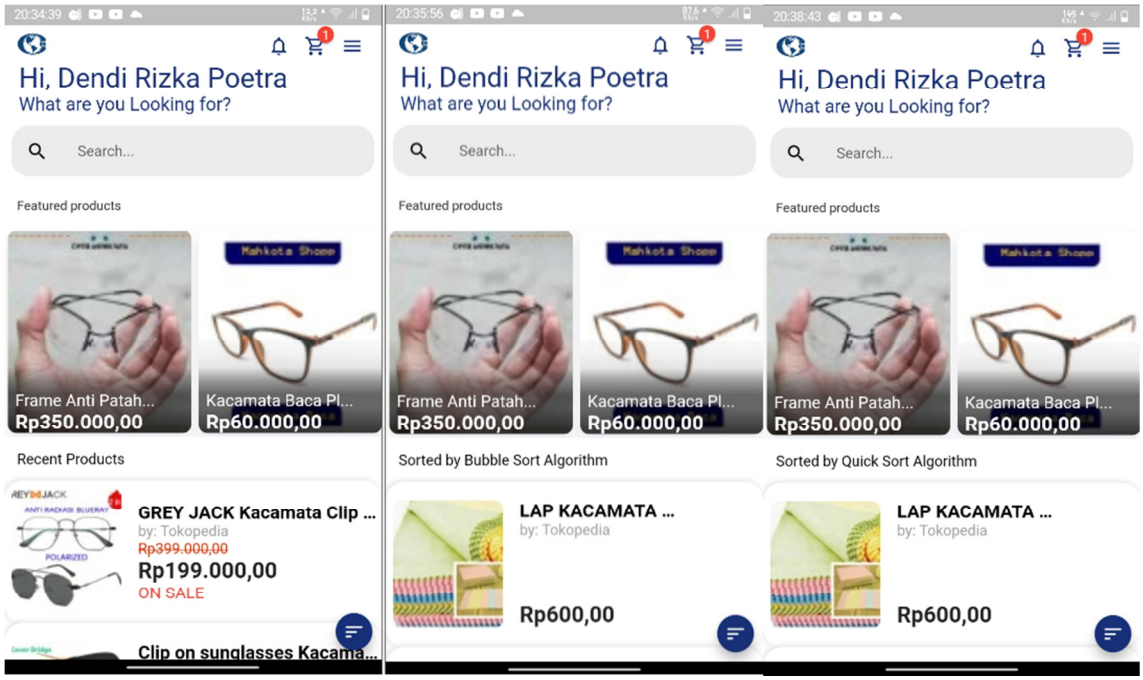
3. HASIL DAN PEMBAHASAN

Algoritma *bubble sort* dan *quick sort* telah diimplementasikan pada media pendukung penelitian yaitu aplikasi e-commerce yang dibangun dengan *framework flutter* dan Bahasa pemrograman *dart*.

Terdapat struktur utama aplikasi, yaitu *class service* yang langsung berhubungan dengan *firebase* sebagai *database* aplikasi, *class model* yang berguna untuk memanipulasi data dari *firebase* yang berbentuk *json* atau *key* : *value*. *class provider* yang digunakan untuk mengelola *state* dan juga sebagai penghubung antara *class model* dan *view* yang mana *class provider* memiliki akses penuh terhadap aliran data mana saja yang harus ditampilkan di*view* dan juga data yang tidak harus ditampilkan di*view*. Kedua algoritma diletakkan pada *class provider* karena seperti yang sudah diuraikan *class provider* merupakan pusat aliran data yang mana akan jauh lebih memudahkan dalam hal implementasi maupun eksekusi pengujiannya.

Terdapat 2 jenis *log output* utama yang akan digunakan sebagai acuan dalam pengambilan data pengujian yaitu log waktu eksekusi algortima yang telah disematkan pada blok/*method code* dari masing-masing algoritma menggunakan fungsi *stopwatch* yang mana *package* dari fungsi tersebut telah tersedia pada *dart sdk*, dan 2 buah variabel bertipe data *list* akan menampung setiap harga produk yang telah diurutkan untuk melakukan *cross-check* ketepatan dalam hal pengurutan data akan menggunakan formula excell yang kemudian akan dihitung menggunakan rumus *error rate* untuk mencari nilai akurasinya yang telah diuraikan pada metode penelitian.

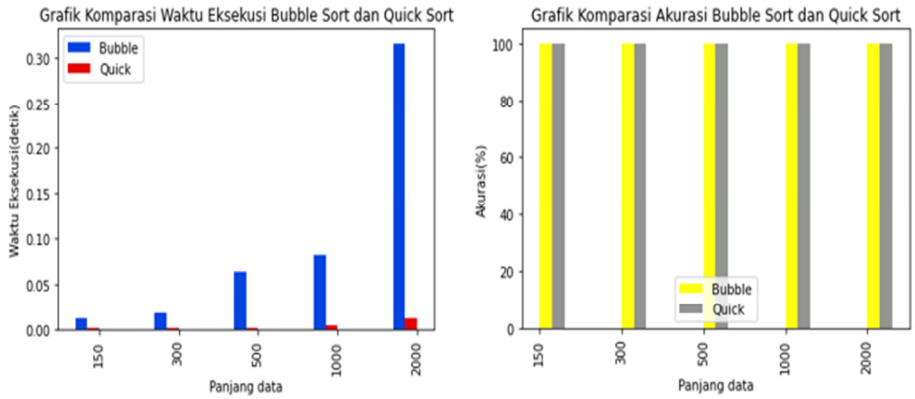
Berikut terlampir pada gambar 6 merupakan *UI* dari halaman utama aplikasi yang mana terdapat *recent product* atau produk yang terbaru dan belum dilakukan pengurutan data, *sorted by bubble sort algorithm* merupakan produk yang telah diurutkan menggunakan algoritma *bubbe sort* dan *sorted by quick sort algorithm* merupakan produk yang telah diurutkan menggunakan algoritma *quick sort*.



Gambar 6. UI Halaman Utama

Tabel 1. Komparasi Performa Algoritma Bubble Sort dan Quick Sort

Algoritma *Sorting*	**Waktu Eksekusi(Per-detik) & Akurasi(%)**									
	150 data		**300 data**		**500 data**		**1000 data**		**2000 data**	
Bubble Sort	0.01114	100	0.01804	100	0.06305	100	0.08254	100	0.315543	100
Quick Sort	0.00034	100	0.00063	100	0.00163	100	0.00351	100	0.011126	100



Gambar 7. Grafik Komparasi Performa Algoritma Bubble Sort dan Quick Sort

Berdasarkan Tabel 1 dan Gambar 7 yaitu, hasil dari pengujian performa algoritma *bubble sort dan quick sort* telah menunjukan bahwa waktu eksekusi algoritma *bubble sort* paling cepat adalah 0.01114 detik dengan 150 data, sedangkan yang terlama adalah 0.315543 detik dengan 2000 data. Selanjutnya untuk waktu eksekusi algoritma *quick sort* paling cepat adalah 0.00034 detik dengan 150 data, sedangkan yang terlama adalah 0.011126 detik dengan 2000 data. Lalu untuk akurasi kedua algoritma menunjukan 100% keakuratan dalam pengurutan data disemua iterasi yang telah diuji. Dan keseluruhan iterasi yang diuji menunjukan dengan sangat jelas bahwa algoritma *quick sort* jauh lebih cepat dibandingakan dengan algoritma *bubble sort* dari segi performa waktu eksekusi.

4. KESIMPULAN

Berdasarkan hasil pengujian yang telah diperoleh pada kasus data pengurutan harga produk pada aplikasi *e-commerce* yang dibangun pada *framework flutter* dan Bahasa pemrograman *dart* dengan algoritma *bubble sort* dan algoritma *quick sort* menunjukan performa yang sangat berbeda pada segi waktu eksekusi. Dari kedua algoritma tersebut menghasilkan selisih waktu yang sangat terlihat dan signifikan perbedaannya pada semua iterasi dengan data yang berbeda-beda disetiap iterasinya. Tetapi performa dari segi akurasi dalam pengurutan data disemua iterasi menunjukan hasil yang identik atau sama persis yaitu, 100% yang mana itu berarti tidak ada data yang salah urutan pada kedua algoritma tersebut. Dari segi performa waktu eksekusi algoritma *quick sort* jauh lebih cepat dibandingkan dengan algoritma *bubble sort*, tetapi dari segi performa akurasi keduanya menghasilkan akurasi yang sama baiknya.

5. SARAN

Pada penelitian ini data yang digunakan untuk pengujian tergolong kecil karena keterbatasan media pengujian atau aplikasi yang hanya mampu melampirkan data ke UI maksimal 2000 data. Jika dipaksakan menggunakan data lebih dari 2000 data, aplikasi akan force close atau menutup sendiri secara tiba-tiba, untuk itu pengembangan penelitian selanjutnya disarankan menggunakan data yang lebih besar dan memperbaiki media penelitian atau aplikasi agar dapat melampirkan data lebih dari 2000 data agar menghasilkan hasil yang lebih optimal dari penelitian ini.

DAFTAR PUSTAKA

[1] Arifin, R. W., dan Setiyadi, D. 2020. *Algoritma Metode Pengurutan Bubble Sort dan Quick Sort Dalam Bahasa Pemrograman C++. Information System For Educators and Professionals*. *No.4, Vol.*2, 178–187.

[2] Rahayuningsih, P. 2016. *Analisis Perbandingan Kompleksitas Algoritma Pengurutan Nilai (Sorting). Jurnal Evolusi*. *No.*4, *Vol.*1, 64–75.

[3] Eko Saputro, F., Nidaul Khasanah, F. 2018. *Teknik Selection Sort dan Bubble Sort Menggunakan Borland C++. Jurnal Mahasiswa Bina Insani*. *No.2, Vol.*2, 136–145.

[4] Pratama, M. A., Desiani, A., dan Irmeilyana. 2017. *Analisis Kebutuhan Waktu Algoritma Insertion Sort, Merge Sort, dan Quick Sort dengan Kompleksitas* Waktu. In *Computer Science and ICT*. *No.*1, *Vol.*1, 31-34.

[5] Retnoningsih, E. 2018. *Algoritma Pengurutan Data (Sorting) Dengan Metode Insertion Sort dan Selection Sort. P2M STMIK Bina Insani Information Management for Educators and Professionals*. *No. 1, Vol.*2, 95–106.

[6] Rizkyatul Basir, R. 2020. *Analisis Kompleksitas Ruang dan Waktu Terhadap Laju Pertumbuhan Algoritma Heap Sort, Insertion Sort dan Merge dengan Pemrograman Java. Satuan Tulisan Riset dan Inovasi Teknologi*. *No.*2, *Vol.*5, 109-118.

[7] Rumapea, Y. Y. P. 2017. *Analisis Perbandingan Metode Algoritma Quick Sort dan Merge Sort Dalam Pengurutan Data Terhadap Jumlah Langkah dan Waktu. Jurnal Methodika. No.2, Vol.3,* 5-9.

[8] Anggreani, D., Wibawa, A. P., Purnawansyah, P., dan Herman, H. 2020. *Perbandingan Efisiensi Algoritma Sorting Dalam Penggunaan Bandwidth. ILKOM Jurnal Ilmiah*. *No.*2, *Vol.*12, 96–103.

[9] Putra, T. A., dan Santoso, Y. 2019. *Implementasi E-Commerce Sebagai Media Penjualan Online pada Toko Tatashops. Jurnal IDEALIS. No.*2, *Vol.*6, 408-413.

[10] Verma, M., dan Chowdhary, K. R. 2018. *Analysis of Energy Consumption of Sorting Algorithms on Smartphones. Proceedings of 3rd International Conference on Internet of Things and Connected Technologies (ICIoTCT). No.*3, *Vol.*89, 472-475.

[11] Chauhan, Y., dan Duggal, A. 2020. *Different Sorting Algorithms Comparison Based Upon The Time Complexity. International Journal of Research and Analytical Reviews*. *No.*3, *Vol.*7, 114-121.

[12] Kumar, S., & Singla, P. 2019. *Sorting Using A Combination of Bubble Sort*, *Selection Sort & amp; Counting Sort. International Journal of Mathematical Sciences and Computing*. *No.*2, *Vol.*5, 30–43.