PENERAPAN ALGORITMA *LINEAR SEARCH* DI APLIKASI *SECONDHAND*

IMPLEMENTATION OF LINEAR SEARCH ALGORITHM IN SECONDHAND APPLICATION

Nely Dwi Agustin[1)], Adi Fajaryanto Cobantoro[2)], Mohammad Bhanu Setyawan[3)]

Khoiru Nurfitri[4)]

[1), 2), 3), 4)] Program Studi Teknik Informatika, Fakultas Teknik, Universitas Muhammadiyah Ponorogo
Jl. Budi Utomo, No.10, Kabupaten Ponorogo, Jawa Timur, 63471
Email : nelydwia@gmail.com[1)], adifajaryanto@umpo.ac.id[2)], m.banu@umpo.ac.id[3)]
nurfitrikhoiru9@gmail.com[4)]

Abstrak

Inflasi di Indonesia mengalami peningkatan dari tahun ke tahun. Adanya inflasi tinggi berdampak pada meningkatnya ongkos produksi barang jadi yang berdampak pada harga barang meningkat. Namun tidak di imbangi dengan penjualan yang baik terhadap barang baru. Hal ini memunculkan sebuah rencana untuk pembuatan e-commerce barang bekas. Penelitian ini bertujuan untuk mengimplementasikan fitur pencarian yang dapat mempermudah pencarian barang bekas berdasarkan keyword. Tahapan implementasi melibatkan Algoritma Linear/Sequential Search dalam bahasa program JavaScript dan PostgreSQL sebagai penyimpanan data yang digunakan. Praktiknya yakni ketika keyword sesuai dengan data yang ada dalam database, hasil pencarian akan ditampilkan. Jika tidak ada kesesuaian data, maka pencarian tidak akan menemukan produk yang relevan. Hasil dari penelitian ini adalah tersedianya aplikasi SecondHand, dengan fitur pencarian dengan Algoritma Linear/Sequential Search yang dapat membantu mempermudah interaksi seller dan buyer. Hasil pengujian white box dan postman/grey box menunjukkan bahwa fitur dan fungsi pencarian berjalan dengan baik, menghasilkan output yang valid, dan waktu eksekusi yang singkat sekitar 613,5 ms atau 0,6135 detik berdasarkan hasil dari lima kali pengujian.

Kata kunci: *Algoritma Linear/ Sequential Search, Barang Bekas, SecondHand.*

Abstract

Inflation in Indonesia has been increasing year by year. The high inflation rate has an impact on the increasing production costs of finished goods, resulting in higher prices which this is not balanced by good sales of new products. This has led to a plan to create a secondhand e-commerce platform. This research aims to implement a search feature that facilitates the search for used goods based on keywords. The implementation process involves the use of the Linear/Sequential Search algorithm in the JavaScript programming language and PostgreSQL as the data storage used. In practice, when the keyword matches the data in the database, the search results will be displayed. If there is no data match, the search will not find any relevant products. The result of this research is the availability of the SecondHand application, with a search feature using the Linear/Sequential Search algorithm, which helps facilitate the interaction between sellers and buyers. The results of white box and postman/grey box testing show that the search feature and its functions work well, producing valid outputs, and have a short execution time of around 613.5 ms or 0.6135 seconds based on the results of five tests.

Keywords : *Linear/Sequential Search Algorithm, Used Goods, SecondHand.*

1. PENDAHULUAN

Dampak Covid-19 terhadap perekonomian Indonesia tercermin dalam peningkatan tingkat inflasi dari tahun 2020 hingga 2022[1]. Pada 2020, inflasi mencapai 1,68%, meningkat menjadi 1,87% pada 2021, dan lonjakan signifikan terjadi pada 2022, mencapai 5,42%[2]–[4]. Hal ini menunjukkan tekanan ekonomi yang signifikan selama periode tersebut[5]. Adanya inflasi yang tinggi berdampak pada meningkatnya ongkos produksi barang jadi[6]. Sehingga menyebabkan harga barang meningkat. Hal tersebut dikarenakan produksi barang jadi meningkat namun tidak diimbangi dengan penjualannya. Ditambah dengan maraknya fenomena *Thrift Shopping* yang berdampak dalam peningkatan konsumsi barang bekas[7]. Ini merupakan tingkah laku kesadaran masyarakat terhadap lingkungan dan tentang kebutuhan untuk menghemat uang di era pandemi *covid-19*[8]–[9].

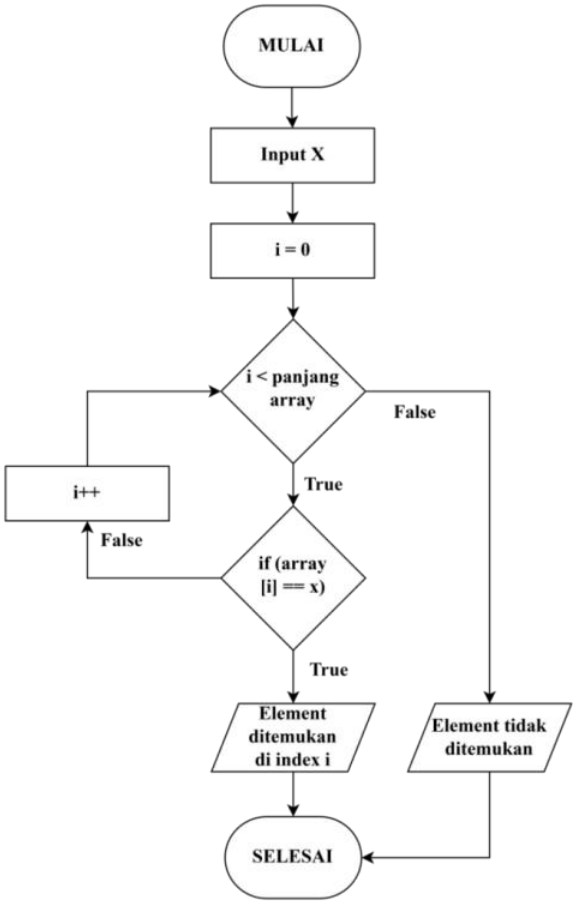
Thrift Shopping saat ini telah menggunakan teknologi seperti aplikasi dan situs web untuk menemukan dan membeli barang bekas, serta menggunakan sistem pembayaran *online*[10]. Konsep *User Experience* aplikasi dan situs web saat ini dianggap sebagai elemen kunci dalam implementasi sistem digital yang efektif[11]. Hal tersebut memudahkan proses belanja bagi pembeli (*buyer*) dan meningkatkan aksesibilitas barang bekas[7]. Namun, salah satu masalah dasar dalam pencarian aplikasi *e-commerce* adalah kemungkinan adanya duplikasi dalam hasil pencarian. *Users* dapat melihat produk yang sama muncul beberapa kali dalam hasil pencarian, yang dapat mengganggu dan memboroskan waktu. Oleh karena itu, peneliti memiliki ide untuk membuat aplikasi *e-commerce* yang diberi nama Secondhand. SecondHand merupakan salah satu *platform* yang dapat melakukan pencarian dengan *Linear/Sequential Search* yang mana merupakan solusi dari permasalahan duplikasi dalam pencarian *item products*. Dengan menggunakan *platform e-commerce* serta media sosial, pembeli dapat bertransaksi dengan mudah tanpa harus datang ke toko fisik[7].

Di dalam aplikasi *e-commerce* SecondHand memiliki fitur pencarian. Dimana untuk pencarian itu ada beberapa algoritma yang paling populer digunakan dalam pembangunan aplikasi yakni algoritma *Linear/Sequential Search*, *Binary Search*, dan *Interpolasi Search*[12]. Secara umum *Linear/Sequential Search* memiliki keunggulan karena tidak memerlukan syarat daftar nilai/data yang harus disusun secara berurutan, sehingga dapat diterapkan pada berbagai jenis data[13]. Dalam konteks pencarian efisien, *Binary Search* unggul dibandingkan *Linear/Sequential Search* karena memerlukan daftar terurut dengan membagi daftar menjadi dua setiap iterasi, sehingga lebih efisien pada data besar[14]. Disisi lain, *Interpolasi Search* tidak membagi daftar menjadi beberapa iterasi seperti *Binary Search*. Sebaliknya, *Interpolasi Search* menggunakan estimasi linier berdasarkan nilai kunci pencarian untuk memprediksi posisi elemen yang dicari dalam daftar terurut[15]. Perkiraan ini membantu mengarahkan pencarian lebih dekat ke elemen yang dicari dengan mempertimbangkan distribusi data linier.

Namun, dalam konteks pembangunan aplikasi, terdapat situasi di mana *Linear/Sequential Search* dapat menjadi pilihan yang lebih baik[16]. Jika daftar nilai/data dari suatu deret tidak terurut, *Binary Search* maupun *Interpolasi Search* tidak dapat digunakan secara langsung. Selain itu, *Linear/Sequential Search* juga lebih fleksibel dalam pembaruan dan penambahan item produk. Sehingga algoritma *Linear/Sequential Search* dapat membantu mengidentifikasi dan menghapus duplikasi dalam hasil pencarian dengan membandingkan setiap *item* secara berurutan. Beberapa contoh aplikasi lain yang menggunakan Algoritma *Linear/Sequential Search*, di antaranya Aplikasi Pembelajaran atau Pendidikan, Aplikasi Berita, dan Aplikasi Pengelola Data[16]–[18]. Berdasarkan statement di atas maka akan dicoba implementasi Algoritma *Linear/Sequential Search* dalam aplikasi SecondHand yang diharapkan bisa mempercepat proses pencarian.

2. DASAR TEORI

Linear search adalah algoritma yang digunakan untuk mencari elemen yang dicari dalam sebuah *array* atau data dengan cara mengecek satu per satu elemen yang ada dalam *array*. Algoritma ini akan berhenti ketika elemen yang dicari ditemukan atau sampai akhir array tercapai. Proses dari *linear search* adalah sebagai berikut:



Gambar 1. *Flowchart Algoritma Linear/ Sequential Search*

Skenario *Algoritma Linear/ Sequential Search* pada gambar 1 :

1) Input keyword *i*,
2) Apakah *i* < panjang *array?* Jika *false* maka elemen tidak ditemukan,
3) Jika *true* maka masuk ke pilihan *if (array[i] == x)*,
4) jika *true* maka elemen langsung ditemukan,
5) Jika *false* maka terjadi perulangan *i++ (increment)* dimana *i + 1* kemudian di kembalikan ke proses sebelumnya hingga elemen berhasil ditemukan.

Contoh, jika kita ingin mencari nilai x dalam sebuah *array* A, dimana x = 7, *array* A = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

1) Mulai dari elemen pertama dari *array* A, yaitu 1.
2) Cek jika 1 sama dengan x (7), jika tidak sama maka lanjutkan ke elemen selanjutnya
3) Cek elemen selanjutnya, yaitu 2.
4) Lanjutkan proses cek elemen sampai elemen ke-7, yaitu 7
5) Nilai x (7) sama dengan elemen ke-7, maka nilai x ditemukan di posisi 6.

Menghitung kompleksitas waktu pada algoritma *Linear/ Sequential* Search :

1) T min (n) : Kompleksitas waktu *minimum* untuk kasus terbaik (*best case*). Kasus terbaik terjadi apabila a1 = X.
   T min (n) = 1
2) T max (n) : Kompleksitas waktu *maximum* untuk kasus terburuk (*worst case*). Kasus terburuk terjadi apabila an = X atau X tidak ditemukan.
   T max (n) = n

3) T avg (n) : Kompleksitas waktu untuk kasus rata-rata (*average case*). Kasus rata-rata terjadi jika X ditemukan pada posisi ke-sekian, maka operasi perbandingan (ak = a) akan dieksekusi sebanyak sekian kali.

$$\text{T avg (n)} = \frac{(1+2+3+\cdots+\text{n})}{\text{n}} = \frac{\frac{1}{2}\text{n}(1+\text{n})}{\text{n}} = \frac{(\text{n}+1)}{2} \qquad \ldots(1)$$

Analisis kompleksitas waktu *Linear/ Sequential Search*. Dalam kasus terbaiknya (*best case*) :

a. Elemen yang dicari ada diposisi pertama.
b. Pencarian berakhir *success* hanya dengan satu perbandingan.
c. Algoritma *Linear/ Sequential Search* mengambil operasi *O(1)*.

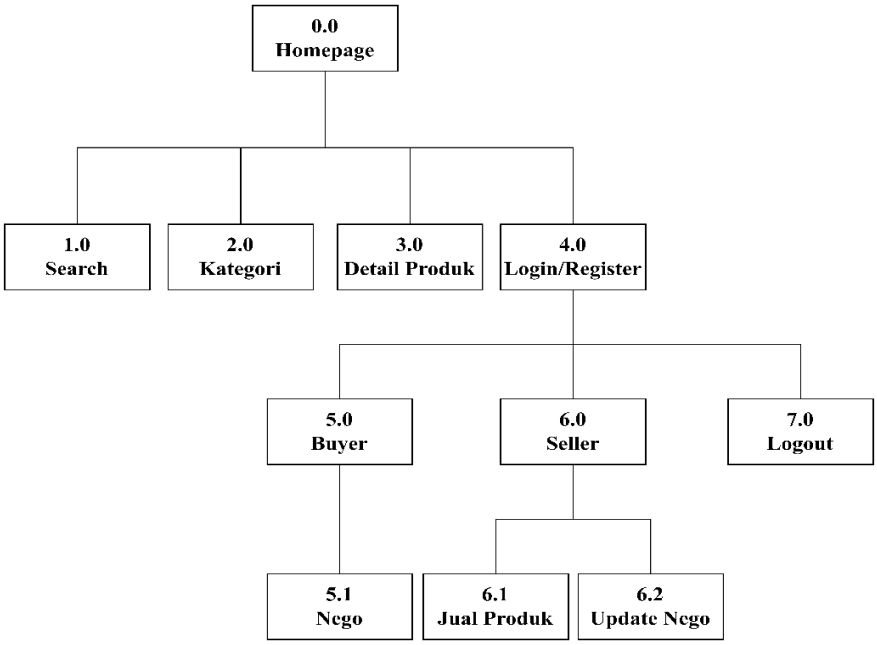
Dalam kasus terburuknya (*worst case*) :

a. Elemen yang dicari ada diposisi terakhir atau tidak ada sama sekali dalam *array*.
b. Pencarian berakhir sebagai kegagalan dengan *n* perbandingan.
c. Algoritma *Linear/ Sequential Search* mengambil operasi *O(n)*.

Linear search dapat digunakan untuk mencari elemen pada berbagai jenis data seperti *array*, *list*, atau data yang berstruktur lainnya. Pencarian *Linear/Sequential* memiliki kelebihan ketika data yang dicari berada di depan, sehingga dapat ditemukan dengan cepat. Namun, kekurangannya adalah jika data yang dicari berada di belakang atau paling akhir, waktu yang dibutuhkan dalam proses pencarian akan menjadi lebih lama[16].

3. METODOLOGI PENELITIAN

3.1 Hierarchy plus Input Process Output



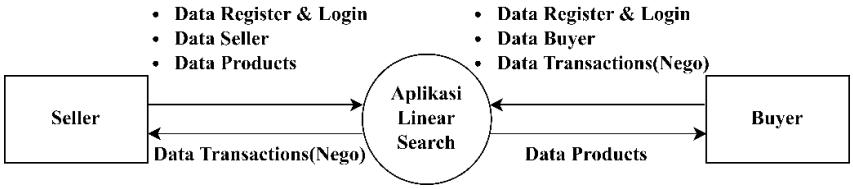
Gambar 2. *Hierarchy plus Input Process Output*

HIPO (*Hierarchy plus Input Process Output*) memberikan penjelasan yang komprehensif mengenai input yang akan digunakan, proses yang akan dilakukan, dan output yang diinginkan pada aplikasi SecondHand. Pada gambar 2 Halaman *Homepage* merupakan halaman utama dalam aplikasi SecondHand. Dihalaman ini, *users* dapat memilih keempat menu yang ditawarkan pada halaman utama atau *Homepage*. Setelah *users* memilih salah satu dari keempat menu tersebut, sistem akan memulai proses untuk menuju ke halaman menu yang dipilih oleh *users* tersebut. Ketika *users* memilih menu *Login/Register*, sistem akan memulai proses untuk menuju ke

halaman *Login/Register*. Pada halaman *Login/Register*, *users* dapat memilih tiga sub menu *Login/Register* yang berbeda.

1) Buyer
    a. Nego
2) Seller
    a. Jual Produk
    b. Update Nego
3) Logout

3.2 Diagram Konteks

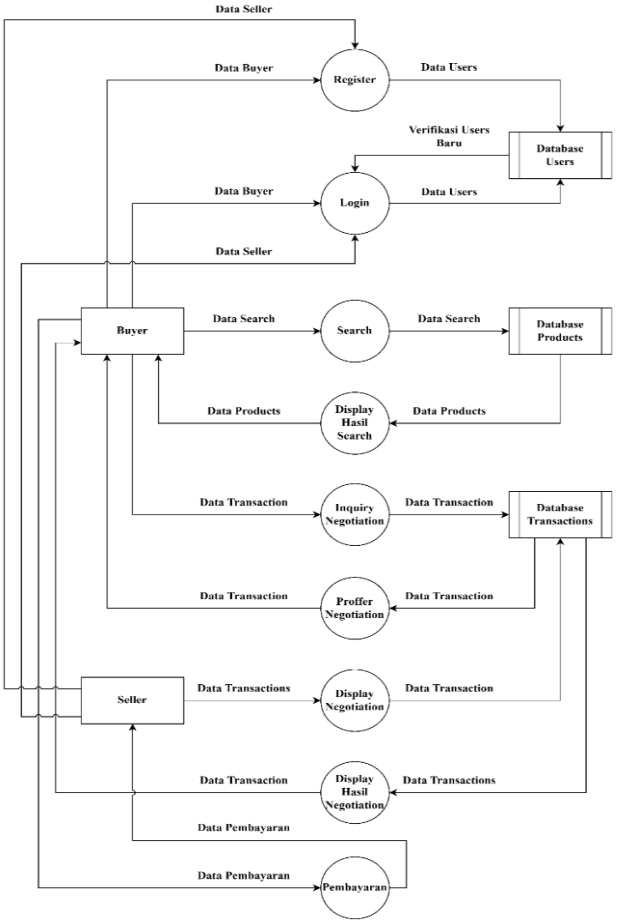


Gambar 3. *Diagram Konteks*

Diagram Konteks aplikasi SecondHand akan menampilkan sistem aplikasi SecondHand sebagai kotak tengah dengan input dan output yang berhubungan dengan aktor eksternal atau sistem lain di sekitarnya. Diagram Konteks aplikasi SecondHand akan memberikan gambaran yang jelas tentang bagaimana *users* dan elemen lain berinteraksi dengan sistem, serta bagaimana sistem tersebut berintegrasi dengan sistem lain. Berikut keterangan dari gambar 3:

1). *Request* : *Users (Seller dan Buyer)* menginputkan informasi pada sistem berupa *Register&Login*, Indentitas *Users* untuk mendapatkan data dari sistem *Database* pada Aplikasi. Peran penjual atau *Seller* menginputkan *Products* yang dijual, dan peran pembeli atau *Buyer* melakukan *Transactions* berupa *Nego* kesuatu *Product* yang ingin dibeli.
2). *Response* : Aplikasi memberi tanggapan yang *Users (Seller dan Buyer)* butuhkan. Dimana penjual atau *Seller* diberi tanggapan berupa *Transactions (Nego)*, sedangkan pembeli atau *Buyer* diberi tanggapan berupa deretan *Product*s.

3.3 DFD *(Data Flow Diagram)*



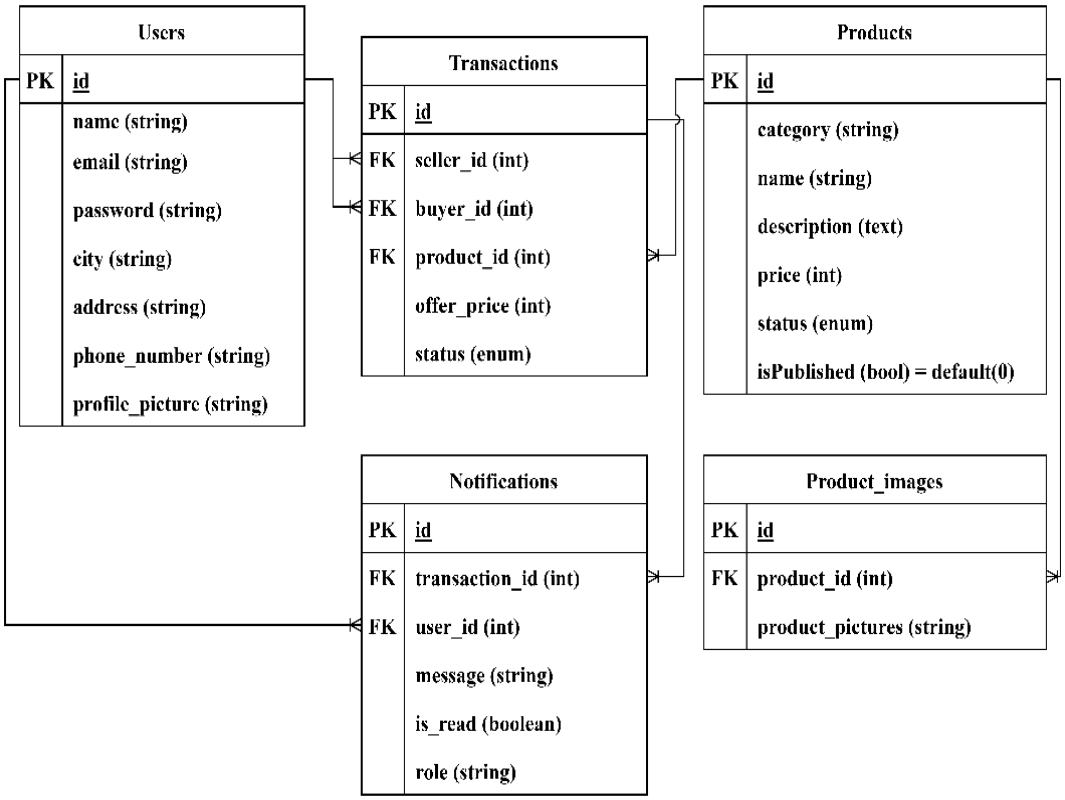
Gambar 4. *DFD lv 1*

Data Flow Diagram level 1 mendeskripsikan hal rinci dari diagram konteks, arus data yang dieksekusikan dalam diagram konteks menunjukkan sistem penyimpanan menuju database. Pada proses di gambar 4, DFD pada level 1, memiliki proses urutan yakni :

a). Register : *Users* mendaftar kedalam aplikasi.
b). Login : *Users* bisa masuk kedalam aplikasi.
c). Search : *Search* yakni *Users* mencari *product* yang ada di dalam aplikasi.
d). Display Hasil Search : *Display Hasil Search* yakni data *product* ditampilkan di aplikasi.
e). Inquiry Negotiation : *Inquiry Negotiation* yakni *Buyer* menawar barang yang ingin di beli.
f). Proffer Negotiation : *Proffer Negotiation* yakni *Buyer* melihat barang yang telah di nego.
g). Display Negotiation : *Display Negotiation* yakni *Seller* melihat barang yang telah di nego.
h). Display Hasil Negotiation : *Display Hasil Negotiation* yakni tampilan *Seller* menerima penawaran.
i). Pembayaran : *Pembayaran* yakni *Buyer* melakukan pembayaran ke *Seller*.

3.4 Relasi Tabel

Relasi tabel merupakan cara untuk mengaitkan data antar tabel dalam basis data. Dalam relasi tabel, setiap baris dalam satu tabel dapat digabungkan dengan baris dalam tabel lain yang sesuai melalui kunci yang digunakan untuk mengaitkan kedua tabel tersebut.



Gambar 5. *Relasi Tabel*

3.5 Pengujian Sistem

Pengujian sistem adalah proses validasi dan verifikasi suatu sistem atau aplikasi dengan memastikan bahwa sistem tersebut memenuhi spesifikasi dan persyaratan yang ditentukan. Tujuan dari pengujian sistem adalah untuk memastikan bahwa sistem atau aplikasi dapat digunakan dengan aman, efisien, dan memenuhi kebutuhan pengguna akhir atau *users*. Berikut pengujian yang akan dilakukan :

3.5.1 *White Box*

White box testing (pengujian kotak putih) adalah pengujian perangkat lunak dengan cara menganalisa dan meneliti struktur internal dan kode dari sistem atau aplikasi. Berikut tahapan pengujian menggunakan *White Box* :

1. Analisis Kode: Pertama, mempelajari dan memahami struktur internalnya.
2. Identifikasi Jalur Eksekusi: Temukan jalur eksekusi yang melibatkan pemahaman tentang kondisi, perulangan, percabangan, dan aliran kontrol dalam kode.
3. Desain Kasus Uji: Rancang kasus uji yang mencakup skenario pengujian yang berbeda untuk mencakup semua jalur eksekusi.
4. Pembuatan Data Uji: Buat data uji yang mencakup kondisi yang mengaktifkan berbagai jalur eksekusi.
5. Eksekusi dan Verifikasi: Jalankan kasus uji yang telah dirancang dan periksa hasilnya, kemudian bandingkan apakah berfungsi dengan benar atau tidak.
6. Analisis dan Perbaikan: Analisa dan temukan kekurangan dalam kode. Lakukan iterasi pengujian jika perbedaan yang dilakukan mempengaruhi jalur logika internal.
7. Penyimpanan Ulang: Lakukan uji kembali komponen yang telah diperbaiki untuk memastikan *bug* telah diperbaiki.

3.5.2 Postman

Aplikasi *postman* berfungsi sebagai *client* untuk melakukan uji coba pada *Rest API* yang sedang diimplementasikan. Berikut adalah tahap-tahap pengujian menggunakan *Postman* :

1. Buat koleksi: Klik *New Collection* dan beri nama yang sesuai untuk menyimpan permintaan *API*.
2. Buat permintaan baru: Klik *New Request* dan pilih metode *HTTP* yang sesuai (*GET, POST, PUT, DELETE, dll.*). Masukkan URL *API* yang akan diuji.
3. Tambahkan header: Jika permintaan memerlukan header, tambahkan header ke bagian *Headers*.
4. Tambahkan parameter: Jika permintaan memerlukan parameter, tambahkan parameter ke bagian *Params*.
5. Tambahkan body: Jika permintaan memerlukan body, tambahkan body ke bagian *Body*.
6. Kirim permintaan: Klik tombol *Send* untuk mengirim permintaan dan memeriksa *respons* dari *API*.
7. Analisis *respons*: Lihat respon dari *API* dalam bagian *Response*. Pastikan bahwa *respons* memenuhi harapan dan persyaratan yang ditentukan.
8. Ubah permintaan: Ubah permintaan dan ulangi langkah 5 hingga 7 untuk melakukan pengujian dengan cobaan berbeda.
9. Simpan permintaan: Klik tombol *Save* untuk menyimpan permintaan dan menggunakannya lagi di kemudian hari.

3.5.3 Algoritma *Linear Search*

Teknik pencarian dengan Algoritma *Linear/Sequential* adalah sebuah pendekatan yang membandingkan setiap elemen data dengan data yang dicari. Pencarian dilakukan dengan membandingkan elemen data yang dicari dengan elemen pertama hingga elemen terakhir. Jika data yang dicari ditemukan, pencarian akan dihentikan.

Terdapat dua kemungkinan dalam *Linear/Sequential Search*:

1) Kemungkinan Terbaik (*Best Case*): Jika data yang dicari terletak pada indeks array terdepan (elemen array pertama), waktu yang dibutuhkan untuk pencarian data akan sangat cepat (minimal).
2) Kemungkinan Terburuk (*Worst Case*): Jika data yang dicari terletak pada indeks array terakhir (elemen array terakhir), waktu yang dibutuhkan untuk pencarian data sangat lama (maksimal).

4. PENGUJIAN DAN PEMBAHASAN

4.1 Implementasi Database

Merujuk pada gambar 5, Tabel Relasi. Tabel-tabel pada gambar menunjukkan bagaimana sistem terhubung disetiap aspek yang melibatkan proses jual beli. Beberapa tabel utama di antaranya adalah:

1) *Users*: Menggambarkan informasi tentang pengguna aplikasi. Mereka mungkin merupakan penjual, pembeli, atau keduanya.
2) *Products*: Menyimpan informasi mengenai produk yang dijual melalui aplikasi. Produk memiliki kategori, nama, deskripsi, harga, dan status (mis. tersedia, habis).
3) *Transactions*: Merepresentasikan setiap transaksi yang telah dilakukan melalui aplikasi. Transaksi tersebut terhubung dengan penjual, pembeli, dan produk yang terlibat.

4) *Notifications*: Menggambarkan notifikasi yang dikirim kepada pengguna, terkait dengan transaksi atau produk.

5) *Product images*: Menggambarkan gambar yang terkait dengan produk yang dijual.

Selain tabel utama ini, beberapa tabel juga menunjukkan hubungan antar tabel yang mengidentifikasikan *Foreign Key* (FK). Contohnya, tabel *Transactions* menunjukkan hubungan antara penjual, pembeli, dan produk yang terlibat melalui *Foreign Key* yang mengarah ke tabel *Users* (*buyer_id*, *seller_id*) dan *Products* (*product_id*). Berikut ini adalah contoh salah satu hasil dari menjalankan database lokal pada *PostgreSQL*. Tabel yang terbentuk adalah seperti yang terlihat pada gambar 6.

id	name	description	price	status	category	isPublished
1	Kipas Angin	Lorem	125,000	DRAFT	Hobi	[v]
2	Mesin Cuci	Lorem	2,000,000	DRAFT	Fashion	[v]

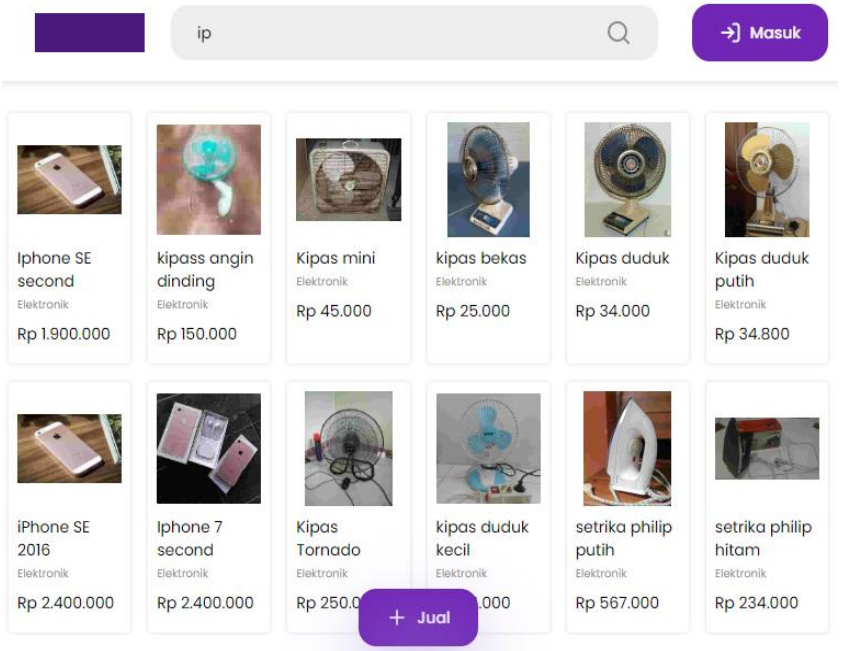
Gambar 6. *Hasil Implementasi Tabel Product Dalam Database*

4.2 Implementasi Algoritma Linear Search

```
6.  const { q, category } = req.query;
7.
8.  const options = {}
18.   if (category) {
19.       options.where.category = category
20.   }
21.   if (q) {
22.       options.where.name = { [Op.iLike]: `%${q}%`
    }
23.   }
24.
25.   const products = await
    Product.findAll(options);
26.
27.    const result = products.map((eachProduct) => {
28.        const image = eachProduct.ProductImages[0]
    ? eachProduct.ProductImages[0].product_pictures
    : null;
```

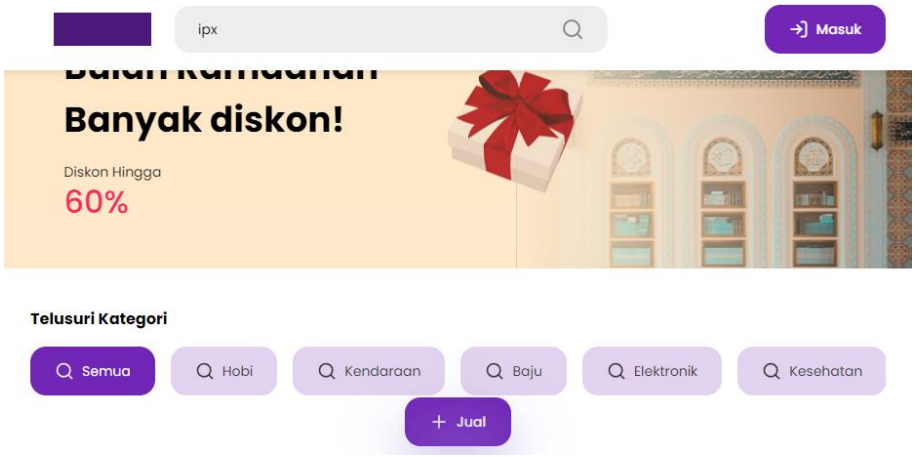
Gambar 7. *Implementasi Code Algoritma Linear/ Sequential Search*

Dalam gambar 7 terlihat bahwa saat melakukan pencarian variable *query* akan mengambil data dari variable *name* dan *category*. Variabel *name* dan *category* pada baris ke-6 berperan dalam pencarian produk pada tabel *Product* berdasarkan nama produk dan kategori produk. Kemudian pada baris ke-22 *[Op.iLike]* digunakan untuk mencari nilai yang cocok secara *case-insensitive* (tidak memperhatikan huruf besar atau kecil). Lalu nilai *%${q}%* berperan dalam mencocokkan bagian dari nilai kolom *name*. Dengan menggunakan *logic* ini, pencarian akan menghasilkan data yang memiliki bagian dari nilai name yang cocok dengan kata-kata pencarian yang dimasukkan.



Gambar 8. Interface Hasil Pencarian

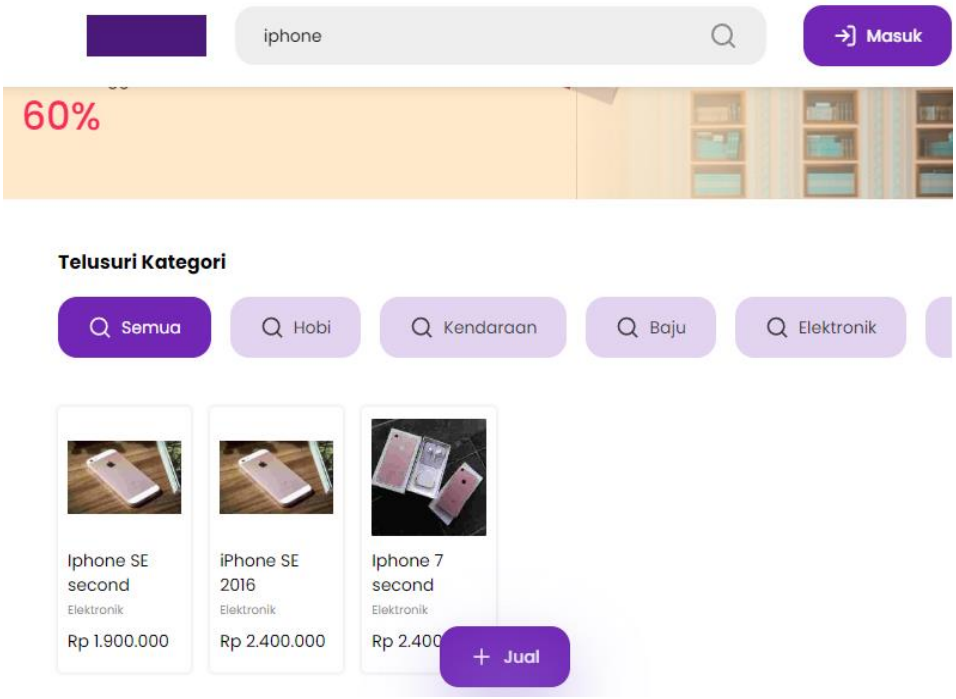
Sebagai contoh ketika kita mencari *keyword* "Ip" maka *keyword* akan masuk kedalam variabel *q* yang mana *q* akan masuk kedalam baris ke-22. Lalu dengan metode *map* pada baris ke-27 digunakan untuk melakukan iterasi pada setiap elemen array *products* dan mengembalikan array baru yang hasilnya akan disimpan dalam variabel *result*. Sehingga hasil dari baris ke-28 akan ditugaskan ke variabel *image*. Jika elemen pertama dalam array gambar produk tidak ada, maka nilai *image* akan menjadi *null*.



Gambar 9. *Interface Hasil Pencarian Jika Data Tidak Ada/Null*

Contohnya, ketika kita mencari *keyword* "Ipx", pencarian tidak akan menemukan produk yang sesuai. Hal ini disebabkan karena dalam database belum ada nama produk yang memiliki unsur yang sesuai dengan *keyword* yang dimasukkan.

Namun, pada implementasi Algoritma *Linear/Sequential Search* pada gambar 7 ketika melakukan *search* dengan *keyword* "Ip" seperti pada gambar 8 akan muncul banyak barang yang namanya mengandung *keyword* tersebut. Sehingga, terdapat banyak rekomendasi dalam pencarian nama *product*. Untuk mencari barang yang lebih spesifik, penginputan *keyword* harus lebih spesifik lagi seperti pada gambar 10.



Gambar 10. Interface Hasil Pencarian spesifik

4.3 Pengujian Sistem

Pengujian sistem dalam pengembangan aplikasi SecondHand merupakan proses krusial yang dilakukan secara menyeluruh dan terintegrasi. Tujuannya adalah untuk memastikan fungsionalitas, performa, kegunaan, dan keamanan aplikasi tersebut sesuai dengan persyaratan *users*. Pengujian dilakukan pada level unit, integrasi, hingga sistem secara keseluruhan, dengan fokus pada pengujian komponen, interaksi antar-komponen, dan pengalaman pengguna. Dengan melakukan pengujian yang komprehensif, pengembang dapat memastikan bahwa aplikasi SecondHand berkualitas tinggi, dapat diandalkan, dan memberikan nilai tambah kepada *users*. Selain itu, pengujian sistem juga membantu mengidentifikasi dan memperbaiki potensi *bug* atau kekurangan dalam aplikasi SecondHand sebelum dirilis kepada *users*.

4.3.1 White Box

Pengujian integrasi adalah jenis pengujian yang menggunakan metode *white box testing* untuk menguji interaksi dan integrasi antara komponen-komponen yang berbeda dalam suatu aplikasi SecondHand. Merujuk pada gambar 4.4 pada baris ke-25 dan baris ke-27. Pengujian yang diinputkan adalah *q* dan *category*. Tabel 1 berisi daftar kasus uji yang akan dilakukan pada pengujian integrasi *searching* aplikasi SecondHand.

Tabel 1. *Identifikasi dan Kasus Uji Pengujian Integrasi*

Nama Class	Nama Method	Tujuan
Products	FindAll()	Mengambil nilai dari *array* yang tersimpan dalam *database* dengan menggunakan Algoritma *Linear/Sequential Search*
Result	Map()	

Berikut merupakan hasil dari identifikasi pada pengujian integrasi yang telah dilakukan dapat dilihat pada tabel 2 pada tabel tersebut merupakan hasil dari pengujian integrasi.

Tabel 2. *Hasil Pengujian Integrasi*

Input Pertama	q:Treadmill

	category:Kesehatan
Method dari class Products	FindAll()
Output pertama atau input kedua	q:Treadmill category:Kesehatan
Method dari *class Result*	Map()
Expected Result	Diharapkan muncul hasil Treadmill di Homepage pada category Kesehatan
Result	Muncul hasil Treadmill di Homepage pada category Kesehatan
Status	Valid

Hasil pengujian integrasi antara *class Products* dan *class Result* menunjukkan bahwa kedua kelas tersebut berintegrasi dengan baik. Input yang diberikan dari *class Products* dapat diproses oleh *class Result*, sehingga menghasilkan *output* yang sesuai dengan Expected Result dan Result.

Berdasarkan hasil pengujian integrasi yang dilakukan, pengujian dari dua class Berdasarkan hasil pengujian integrasi yang dilakukan, pengujian dari dua *class* tersebut memperoleh status *valid*. Hubungan antara kedua *class* ini berhasil mengimplementasikan fitur pencarian yang diinginkan. Pengujian integrasi antara dua *class* ini memberikan jawaban terkait penerapan Algoritma *Linear/ Sequential Search* pada aplikasi SecondHand. Dengan demikian, dapat dipastikan bahwa penggunaan Algoritma *Linear/ Sequential Search* pada aplikasi SecondHand berjalan dengan efektif dan menghasilkan hasil pencarian yang akurat.

4.3.2 Postman

Pengujian sistem dengan metode *Grey box testing* menggabungkan elemen dari pengujian *Black box testing* dan *White box testing*. Dalam metode ini, pengujian dilakukan dengan pemahaman yang lebih dalam tentang struktur dan logika fungsionalnya. Pada pengujian *Grey box*, pengujian dilakukan dari perspektif *users*. Tabel 4.3 berisi hasil pengujian yang dilakukan pada pengujian sistem *searching* Aplikasi SecondHand.

Tabel 3. *Hasil Pengujian Sistem Searching Aplikasi SecondHand*

Http Request			
1	Verb/Method	:	Get
2	URL	:	https://secondhand-be-production.up.railway.app/api/ ?q=Sepeda&category=Kendaraan
3	Query Params	:	q:Sepeda category:Kendaraan
4	Request Header	:	-
5	Request Body	:	-

Http Response			
1	Status/Response Code	:	200 OK
2	Response Header	:	x-powered-by: Express vary: Origin content-type: application/json; charset=utf-8 content-length: 480 etag: W/"1e0-O3NOrXalINIYtFxWmdBt3MQmAjQ" date: Sun, 28 May 2023 09:17:00 GMT server: railway

3	Response Body	:	{
			"status": "success",
			"msg": "homepage",
			"data": [
			{
			"id": 22,
			"name": "Sepeda Gunung Exotic 2655",
			"description": "Jual Sepeda Gunung Exotic 2655 dari Pacific, Groupset Shimano, Kondisi Full std Original no modif, mulus bagus jarang dipakai. harga 1.2jt nego tipis minat silahkan chat dibawah Terimakasih",
			"price": 1200000,
			"status": "DRAFT",
			"category": "Kendaraan",
			"isPublished": true,
			"ProductImage": "https://res.cloudinary.com/dfhjxw9zd/image/upload/v1685014345/NaN/reepmwbzayu1fqyrdnzb.jpg"
			}
			]
			}

Pada tabel 3 data yang berhasil terpanggil yaitu menunjukkan bahwa permintaan REST API (*Representational State Transfer Application Programming Interface*) pada bagian *Homepage* SecondHand berhasil dilakukan. Dengan mendapatkan *response code 200* dan respons yang diberikan mengindikasikan keberhasilan yakni *response body status success*.

4.3.3 Algoritma *Linear Search*

Uji *run time* pencarian barang bekas pada *searching* aplikasi SecondHand akan di lakukan sebanyak 5 kali berdasarkan *category* dan *keyword* didapatkan hasil sebagai berikut :

Tabel 4. *Rician Hasil Run Time Pencarian Barang Bekas*

Pengujian	Category	Keyword	Hasil Yang Diperoleh	Run Time
1	Hobi	Lukisan	LUKISAN SHIO DOMBA CINA	210 ms
2	Kendaraan	Sepeda	sepedah gunung, sepeda jadul, sepeda tahun 1922, sepeda vintage, Sepeda Gunung Exotic 2655	227 ms
3	Baju	Dress	dress midi, dress bunga korea, Dress muslim korea, Gamis Dress Muslim, Mini Dress	223 ms
4	Elektronik	Iphone	Iphone SE second, iPhone SE 2016, Iphone 7 second	225 ms
5	Kesehatan	Treadmill	Treadmill Elektrik no incline bfs-607, X8 Motorized Treadmill	222 ms

Kompleksitas waktu dalam pencarian barang bekas berdasarkan *category* dan *keyword* secara *Linear/Sequential* pada aplikasi bisa dijelaskan sebagai berikut :

1) $T\ min\ (n)$: *Run time* pencarian data untuk kasus terbaik (*best case*).
   $T\ min\ (n) = 1$
   $T\ min\ (n)$: Pengujian ke satu mencari *category* Hobi dengan *keyword* "Lukisan", di dapatkan hasil *run time* = 210 ms =0,210 s.
2) $T\ max\ (n)$: *Run time* pencarian data untuk kasus terburuk (*worst case*).
   $T\ max\ (n) = n$
   $T\ max\ (n)$: Pengujian ke dua dengan pencarian dengan kasus terburuk ada pada saat pencarian dengan *category* Kendaraan dengan *keyword* "Sepeda", dengan *run time* adalah 227 ms= 0,227 s.
3) $T\ avg\ (n)$: *Run tine* pencarian data untuk kasus rata-rata (*average case*).
   $n = T\ max\ (n)$

$$T\ avg\ (n) = \frac{(n+1)}{2}$$

$$T\ avg\ (n) = \frac{(0{,}227+1)}{2} = 0{,}6135$$

Berdasarkan hasil pencarian dan perhitungan diatas, rata rata waktu yang dibutuhkan dalam mencari barang bekas sebanyak 613,5 ms = 0,6135 s.

5. KESIMPULAN

Berdasarkan hasil pengujian menggunakan metode *White box* dan *Postman/Grey Box*, aplikasi menunjukkan bahwa seluruh fungsi berjalan sesuai dengan alur dan logika yang telah ditentukan sebelumnya. Terbukti dari waktu eksekusi yang sangat singkat, sekitar 613,5 ms atau 0,6135 detik berdasarkan hasil lima kali pengujian menggunakan Algoritma *Linear/Sequential Search*. Hasil pengujian juga menunjukkan bahwa fitur dan fungsi pencarian berjalan dengan baik dan menghasilkan output yang *valid*. Dengan mempertimbangkan hasil analisis dan perancangan yang telah dijelaskan sebelumnya, serta pengujian dengan menerapkan Algoritma *Linear/Sequential Search* pada sistem pencarian SecondHand. Dapat disimpulkan bahwa

penerapan Algoritma *Linear/Sequential Search* pada sistem pencarian berhasil diimplementasikan dan mampu melakukan pencarian dengan baik.

DAFTAR PUSTAKA

[1] A. Feranika and D. Haryati, "Strategi Kebijakan Fiskal Terhadap Output dan Inflasi pada Perekonomian Indonesia dalam Menghadapi Dampak Virus Covid 19," *Bus. Innov. Entrep. J.*, vol. 2, no. 3, pp. 146–152, 2020, doi: 10.35899/biej.v2i3.154.

[2] TPIP, "Analisis Inflasi Desember 2020 Tim Pengendalian Inflasi Pusat (TPIP)," pp. 1–21, 2021.

[3] TPIP, "Analisis Inflasi Desember 2021 Tim Pengendalian Inflasi Pusat (TPIP)," pp. 1–17, 2022.

[4] K. K. B. P. R. INDONESIA, "Realisasi Inflasi Indonesia November Melandai Ditopang Harga Pangan yang Stabil pasca Penyesuaian Harga BBM," in *SIARAN PERS*, 2022, pp. 1–2.

[5] M. I. . Tuasikal and S. Susianti, "Analisis Perubahan Indeks Harga Saham Gabungan dan Harga Saham Perusahaan BUMN Konstruksi Tahun 2020-2022 di Pasar Modal Indonesia", jpe, vol. 17, no. 4, pp. 763-774, Dec. 2022, doi: 10.22437/jpe.v17i4.23907.

[6] Y. Rachmawati, "Pengaruh Inflasi dan Suku Bunga Terhadap Harga Saham Pada Perusahaan Perbankan Yang Terdaftar Di LQ45 Bursa Efek Indonesia," *J. Media Akunt.*, vol. 1, no. 1, pp. 66–79, 2019, doi: 10.31851/jmediasi.v1i1.2368.

[7] A. S. Permatasari, S. Rahmadhan, W. J. Firdausy, and H. L. Meidianti, "Pengaruh Komunikasi Pemasaran Thrift Shop terhadap Tingkat Konsumsi Fashion di Masa Pandemi," *J. Ilmu Komun.*, vol. 11, no. 1, pp. 93–107, 2021, doi: 10.15642/jik.2021.11.1.93-107.

[8] Hasriani, S. Arwati, and S. Khadijah, "Edukasi Food Preparation Yang Tepat Dalam Menghadapi Era New Normal Pandemi Covid-19 Pada Kelompok Rumah Tangga Di Kelurahan Bontomanai, Kec. Bontomarannu, Kab. Gowa," *J. Abdimas Indones.*, vol. 2, no. 1, pp. 46–53, 2022, doi: 10.53769/jai.v2i1.165.

[9] E. P. Nimasari et al., "Incorporating computer-assisted language learning for standardized test of academic English proficiency (STAcEP) in the post-Covid-19-era: A quantitative method," in AIP Conference Proceedings, AIP Publishing, 2023.

[10] A. K. Darmawan, M. B. Setyawan, A. F. Cobantoro, F. Masykur, A. Anwari, and T. Yulianto, "Knowledge Management System Analysis of Smart Regency Mobile-Apps Service with Software Usability Measurement Inventory (SUMI) Approach," in 2021 International Conference on ICT for Smart Society (ICISS), 2021, pp. 1–6. doi: 10.1109/ICISS53185.2021.9533212.

[11] A. Kisnu Darmawan, M. Bhanu Setyawan, A. Fajaryanto Cobantoro, F. Masykur, A. Komarudin and M. Waail al Wajieh, "Adaptation of the meCUE 2.0 Version for User Experience(UX) Measurement Approach into Indonesian Context," 2021 Sixth International Conference on Informatics and Computing (ICIC), Jakarta, Indonesia, 2021, pp. 1-6, doi: 10.1109/ICIC54025.2021.9633008.

[12] A. B. Nugroho and S. Mandala, "Study the Best PenTest Algorithm for Blind SQL Injection Attacks", ijoict, vol. 5, no. 2, pp. 1-10, Jun. 2020, doi: 10.21108/IJOICT.2019.52.268.

[13] I. K. S. Buana, H. Setiawan, and P. A. W. Putro, Pemrograman Terstruktur. Syiah Kuala University Press, 2022. [Online]. Available: https://books.google.co.id/books?id=aLVsEAAAQBAJ

[14] R. Toyib, Y. Darnita, and A. R. S. Deva, "Penerapan Algoritma Binary Search Pada Aplikasi E-Order," *J. Media Infotama*, vol. 17, no. 1, pp. 30–37, 2021, doi: 10.37676/jmi.v17i1.1314.

[15] S. B. H. M. I. S. F. T. Waruwu, "Perancangan Aplikasi Kamus Istilah Programming Dengan Metode Interpolation Search Berbasis Website," KOMIK (Konferensi Nas. Teknol. Inf. dan Komputer), no. Vol 6, No 1 (2022): Challenge and Opportunity For Z Generation in Metaverse Era, pp. 75–80, 2022, [Online]. Available: http://ejurnal.stmik-

budidarma.ac.id/index.php/komik/article/view/5790/3277
[16] Mhd Furqan, Armansyah, and Riski Askia Kurniawan, "Analisis Algoritma Sequential Search Pada Aplikasi Pencarian Berita," STIKOM Tunas Bangsa Pematangsiantar, vol. Vol 8, No 2 (2023): Edisi Agustus, 2023, [Online]. Available: https://tunasbangsa.ac.id/ejurnal/index.php/jurasik
[17] Y. Rahmanto, J. Alfian, D. Damayanti, and R. Indra, "Penerapan Algoritma Sequential Search pada Aplikasi Kamus Bahasa Ilmiah Tumbuhan," J. Buana Inform., vol. 12, p. 21, May 2021, doi: 10.24002/jbi.v12i1.4367.
[18] A. A. Rismayadi and L. Jamaliah, "Implementasi Algoritma Sequential Searching Pada Aplikasi E-Office," J. Nas. Riset, Apl. dan Tek. Inform., vol. 1, no. 1, 2019, [Online]. Available: https://naratif.sttbandung.ac.id/index.php/naratif/article/view/naratif.v1i1.21