Platform Big Data Analytic Berbasis Apache Spark Bagi Pemula Dalam Menyusun Data Analysis Workflow

Daniel Jeremia, Henry Novianus Palit, Andre Gunawan
Program Studi Informatika Fakultas Teknologi Industri Universitas Kristen Petra
Jl. Siwalankerto 121 – 131 Surabaya 60236 Telp. (031) – 2983455, Fax. (031) – 8417658
E-Mail: jeremia.daniel98@gmail.com, hnpalit@petra.ac.id, andre.gunawan@petra.ac.id

ABSTRAK

Data merupakan dasar yang konkrit untuk pengambilan keputusan. Seiring dengan berkembangnya teknologi, jumlah dan kompleksitas data menjadi masalah sebab diperlukan metode khusus untuk melakukan analisis data. Oleh sebab itu lah big data analytics diperlukan. Kebutuhan untuk melakukan analisis data yang cepat, sederhana, dan kokoh sangat tinggi, terutama bagi pemula. Diperlukannya keahlian dan pengalaman yang tinggi di bidang pemrograman menjadi masalah bagi pengguna pemula.

Untuk menghadapi hal ini, sebuah platform untuk melakukan big data analysis yang ramah bagi pengguna pemula diusulkan dalam penelitian ini. Platform ini dibuat dengan tujuan pengguna yang tidak memiliki pengalaman pemrograman dapat melakukan analisis data dengan mudah. Manipulasi data dibuat dalam bentuk diagram/workflow yang dapat dirancang secara drag-and-drop, sehingga ramah untuk pengguna pemula. Selebihnya, platform ini menggunakan teknologi terkemuka di industri saat ini untuk menghadapi permasalahan big data analytic bernama Apache Spark tanpa diketahui oleh pengguna sama sekali.

Dilakukan survei/demo terhadap 12 orang dari 3 macam latar belakang, yaitu awam, pemula, dan ahli. Hasil yang diperoleh mengindikasikan pengalaman yang positif dalam melakukan analisis data tanpa menggunakan pemrograman. Rata – rata peserta survei menilai platform ini memudahkan pekerjaan analisis data sebanyak 4.4 dari 5. Platform big data analytic berpotensi besar baik bagi pengguna pemula maupun profesional.

Kata Kunci: analisis *big data*, alur kerja *data science*, *apache spark,* analisis data pemula

ABSTRACT

Data is a concrete foundation for decision-making. The development of technology, in turn, creates a problem in the number and complexity of data as it requires sophisticated methods to analyze. This calls for the need of big data analytics. Analyzing data quickly, simply, and robustly is now a very high requirement, especially for beginners.

To combat this problem, a platform for big data analytics that is beginner-friendly is proposed in this research. This platform is created with the purpose of simplifying the process of analyzing data easily without the use of programming for beginners. Diagrams/workflows are designed to manipulate data in a drag-and-drop fashion to make it easier for beginners. Furthermore, this platform uses industry-leading technology such as Apache Spark to deal with the problems of big data analytic without being known by the user at all.

A survey/demo of 12 people with 3 different backgrounds, namely commoners, beginners, and experts, is held. The obtained result indicates a positive experience in doing data analysis without programming. An average score of 4.4 out of 5 is given by the participants for how much this platform can simplify the work of data analysis. This big data analytic platform has a huge potential for beginners and professionals alike.

Keywords: *big data analytics, data science workflow, apache spark, beginner-friendly data analytics*

1. PENDAHULUAN

Data memiliki peranan yang penting dalam perolehan informasi. Analisis data menyediakan dasar yang konkrit atas pengambilan keputusan. Namun, sebagian besar data yang dihasilkan setiap harinya merupakan data tidak terstruktur (*unstructured data)*.

Analisis big data merupakan sebuah bidang yang menjelaskan bagaimana mengumpulkan, menyiapkan, mengolah, hingga menganalisis data yang berjumlah sangat besar dan bervariasi. Big data mendefinisikan data yang besar dengan konsep "4 V's of Big Data", yaitu *volume, velocity, variety,* dan, *veracity.*

Salah satu perusahaan ternama yang telah menggunakan analisis big data untuk pengambilan keputusan yang strategis adalah Coca Cola. Coca Cola merupakan perusahaan minuman yang terbesar di dunia dengan lebih dari 500 macam produk yang dijual di lebih dari 200 negara. Setiap harinya, lebih dari 1.9 miliar produk mereka yang dikonsumsi. Menurut artikel Forbes, Coca Cola merupakan salah satu perusahaan pertama di luar sektor teknologi yang memanfaatkan analisis big data [5].

Di tengah perkembangan big data yang begitu pesat — dan diperkirakan untuk semakin berkembang, timbul sebuah masalah. Semakin banyak dan cepat data yang dihasilkan, semakin sulit dan kompleks untuk melakukan proses analisis big data. Salah satu alasan utamanya adalah kurangnya orang - orang dengan kemampuan analisis data yang memadai untuk menghasilkan informasi yang dibutuhkan dari data [2].

Hingga saat ini, belum ada sebuah *platform* untuk melakukan proses analisis big data menggunakan *framework* seperti *Apache Spark* yang ramah untuk orang awam. *Apache Spark* masih membutuhkan kemampuan pemrograman yang cukup tinggi, sehingga diperlukan ahli untuk menggunakannya. Program - program yang sudah ada membantu membuat *workflow* dan mempermudah pelaksanaan *job Spark,* namun belum ada *tool* yang khusus untuk mempermudah penggunaan Spark hingga tidak memerlukan pengalaman *coding*.

Untuk menghadapi hal ini, ide yang diusulkan adalah membuat sebuah *platform* untuk melakukan *big data analysis workflow* menggunakan HDFS dan Spark yang mudah untuk digunakan tanpa memerlukan *skill* pemrograman. Platform ini dapat mendesain sebuah *workflow* analisis data yang dapat

1. Melakukan EDA (Exploratory Data Analysis),
2. Melakukan *preprocessing* dan *cleaning* data,
3. Membuat visualisasi dan grafik hasil analisis data, dan
4. Membuat model machine learning.

Platform ini dibuat dengan harapan dapat memudahkan pekerjaan *data analysis* khususnya bagi pengguna pemula yang minim pengalaman.

2. DASAR TEORI

2.1. Big Data Analysis

Istilah *big data* adalah hasil dari meledaknya jumlah data dari berbagai macam sumber. Adanya perkembangan data yang begitu besar mendorong peneliti - peneliti untuk melakukan eksperimen berbasis data. Metode - metode analisis secara konvensional sudah tidak lagi dapat mengolah data tersebut dengan baik. Sekarang, big data bukan hanya berbicara mengenai jumlah atau volume dari data [4].

1. Karakteristik Big Data

   Menurut Gandomi dan Haider, big data memiliki 4 karakteristik khusus yaitu *volume, velocity, variety,* dan *veracity*. *Volume* berbicara mengenai banyaknya jumlah data yang harus diolah, ini merupakan hal yang paling terlihat dari big data. *Velocity* adalah cepatnya data baru dibuat. *Variety* artinya data yang ada bermacam - macam dan berbeda - beda bentuk nya, mulai dari *structured data, semi-structured data,* hingga *unstructured data*. Terakhir, veracity menjelaskan bagaimana tidak semua data dapat dipercaya, terutama data yang bervolume sangat besar.

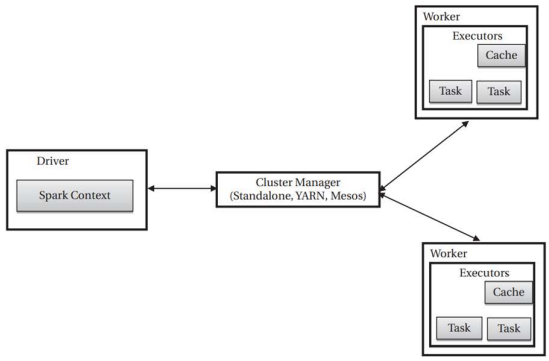
2. Masalah dalam Big Data

   Inilah yang menjadi masalah utama dalam *big data.* Metode – metode yang digunakan dalam *big data* adalah untuk menghadapi 4 masalah tersebut. Rumitnya data dalam big data yang dikarakteristikan menjadi *4 V's of Big Data* membuat proses analisis big data tidak semudah menganalisis data pada umumnya. Data yang biasanya digunakan dalam big data dapat mencapai *terabyte* hingga *petabyte.* Tidak banyak mesin komputasi di dunia yang dapat melakukan proses komputasi menggunakan data sebanyak itu. Solusinya adalah melakukan *horizontal scaling. Horizontal scaling* artinya meningkatkan kinerja dengan cara menambah mesin komputasi yang digunakan. Konsep ini disebut *Distributed Computing*. Inilah yang membuat analisis big data berbeda dari analisis data konvensional.

2.2. Apache Spark

Apache Spark adalah sebuah *framework* untuk melakukan proses analisis *big data* yang *open source*, cepat, dan disimpan dalam memori. *Apache Spark* menyediakan *API* yang bersifat *high-level* untuk melakukan proses pengolahan data dalam bahasa *Java, Scala, Python*, dan *R*.

Apache Spark memiliki arsitektur berupa *master* dan *slave* (disebut *driver* dan *worker*) yang diperantarai oleh sebuah *cluster manager*. *Cluster manager* dapat menggunakan *standalone, yarn,* atau *mesos*. Secara garis besar, arsitektur *Apache Spark* dapat dilihat pada Gambar 1.



Gambar 1. Arsitektur Apache Spark [1]

2.3. Data Science Workflow

Proses analisis data mencangkup beberapa langkah yang harus dilewati data yang sedang dianalisis. Sekuen yang terbentuk disebut dengan Data Science Workflow. Workflow ini mengatur bagaimana alur data mulai dari akuisisi hingga menghasilkan informasi berupa visualisasi data.

Sebuah contoh workflow sederhana paling tidak dapat dibagi menjadi 3 bagian. Tahap yang pertama disebut sebagai data acquisition. Proses ini mencangkup mengumpulkan data dari berbagai macam tempat dan dikumpulkan untuk diolah.

Selanjutnya, data yang sudah dikumpulkan harus diproses sebelum dapat dianalisis lebih dalam. Data yang terkumpul bisa saja dalam bentuk yang berbeda - beda mulai teks, numerik, gambar, video, hingga audio. Untuk melakukan proses analisis data dengan baik, perlu dilakukan *preprocessing* dan *cleaning data.*

Terakhir, setelah data sudah siap, masuklah ke tahap analisis data. Ada banyak metode yang tersedia untuk menganalisis data antara lain *machine learning*, analisis statistik, dan *data mining* [3].

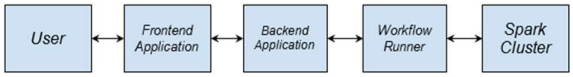
3. DESAIN SISTEM

Platform analisa big data berbasis *Spark* dibuat dalam bentuk aplikasi web yang dapat dibagi menjadi 3 bagian, yaitu sebagai berikut.

1. Workflow Runner
2. *Backend* aplikasi website
3. *Frontend* aplikasi website

Masing - masing bagian tersebut saling berhubungan secara *microservice* menggunakan *API*. *Workflow Runner* berfungsi untuk mengubah *workflow* menjadi *Spark job*. *Backend* aplikasi website digunakan sebagai perantara antara *Frontend* dengan *Workflow Runner*. Pengguna platform dapat menggunakan platform melalui antarmuka dari aplikasi *Frontend.*

Secara keseluruhan alur penggunaan platform dijabarkan menjadi Gambar 2 di bawah ini.

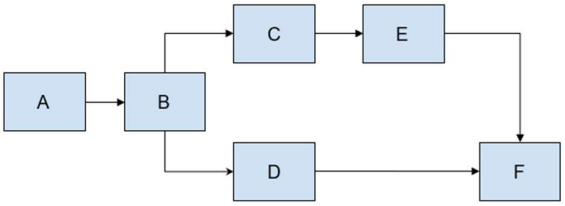


Gambar 2. Alur Keseluruhan Platform Analisa Big Data

3.1. Desain Sistem Workflow Runner

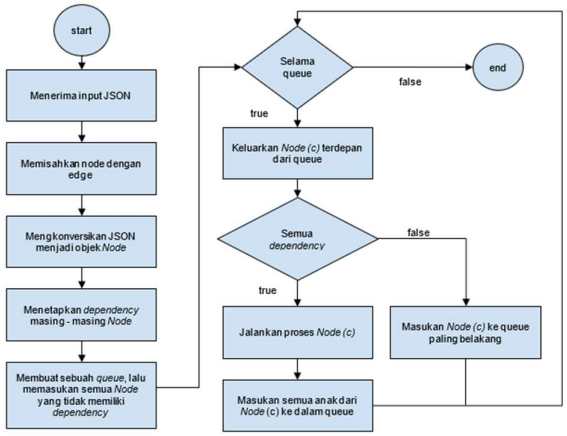
Workflow Runner merupakan sebuah *package* Python yang dibuat dalam penelitian skripsi ini untuk menjalankan *Spark job* berdasarkan input *workflow*. *Package* ini terdiri dari 2

komponen yaitu *DAG Parser* dan *Nodes*. Komponen *DAG Parser* berisi implementasi bagaimana masing - masing *node* dalam *workflow* akan dikerjakan. *DAG Parser* menerima input berupa sebuah *Directed Acyclic Graph* yang disimpan dalam bentuk *JSON*. *JSON* tersebut berisi seluruh *node* yang ada dalam graf dan seluruh hubungan antara *node - node* tersebut. Contoh sebuah *DAG* dapat dilihat pada Gambar 3 berikut ini.



Gambar 3. Contoh sebuah DAG

Cara kerja *DAG Parser* dijelaskan pada flowchart pada Gambar 4 berikut ini.



Gambar 4. Proses Kerja Workflow

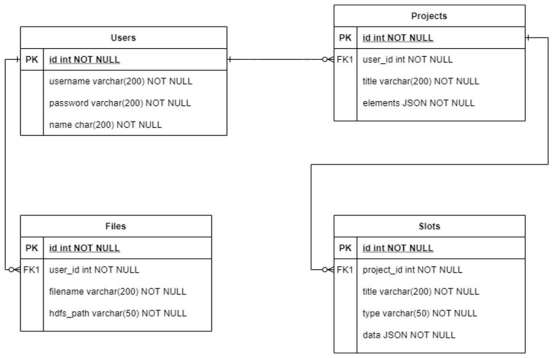
Sistem ini juga memuat seluruh *node* hasil pemetaan dengan Apache Spark.

3.2. Desain Sistem Backend Aplikasi

Aplikasi *Backend* website merupakan sebuah server yang berjalan menggunakan *Python Flask* untuk melayani *HTTP requests* serta *Google Firestore* untuk database *NoSQL*. Aplikasi ini digunakan sebagai perantara antara aplikasi *frontend* dan *Workflow Runner*. Semua layanan seperti mengirim workflow dari *frontend* ke *Workflow Runner*, mengunggah dataset ke *HDFS*, mengambil skema dari sebuah dataset dalam *HDFS*, dan mendapatkan data untuk visualisasi akan melalui aplikasi *backend* ini. Aplikasi ini memiliki beberapa *endpoint* yang dapat diakses.

3.2.1. Database

Database yang digunakan adalah *Google Firestore*. *Google Firestore* adalah sebuah database yang disediakan oleh *Google Cloud Platform* sebagai database *document-based NoSQL*. Gambar 5 berikut ini adalah *data modelling diagram* yang digunakan dalam pengimplementasian aplikasi *backend* website.



Gambar 5. Data Modelling Diagram

3.3. Desain Sistem Frontend Aplikasi

Frontend aplikasi website merupakan tampilan antarmuka utama yang akan ditemui oleh pengguna. Aplikasi ini dibuat menggunakan *Next.JS*, sebuah *framework* untuk *React* yang berdasarkan *Javascript*. Untuk mempermudah, aplikasi *frontend* akan disebut sebagai *Workflow Designer*. Menggunakan *Workflow Designer*, pengguna dapat menyusun *node - node* fungsi *Spark* dalam bentuk *DAG*. Setiap *node* yang ada pada sistem *Workflow Runner* telah dipetakan di dalam aplikasi ini. Setelah selesai menyusun *workflow*, pengguna dapat menekan tombol *Run* yang akan mengirimkan *workflow* melalui aplikasi *backend* menuju *Workflow Runner*.

Sistem ini juga memuat bagian tampilan seluruh *node* yang ada pada *Workflow Runner*.

4. IMPLEMENTASI SISTEM

Implementasi sistem dilakukan pada sebuah klaster *Apache Spark* yang berjumlah 27 komputer (1 *master* dan 26 *worker*). Seluruh komputer tersebut sudah dilengkapi dengan

1. Python 3.8,
2. Apache Spark dan PySpark,
3. HDFS, dan
4. Yarn.

Library – library Python yang dibutuhkan adalah sebagai berikut:

5. Flask
6. Redis
7. Google Firestore
8. PySpark
9. Flask_cors

Aplikasi *frontend* dapat dijalankan pada *service* yang terpisah. Aplikasi ini menggunakan *framework* bernama NextJS.

5. ANALISA DAN PENGUJIAN

Pengujian akan dilakukan dengan cara melakukan survei terhadap beberapa kelompok tertentu yaitu awam, pemula, dan ahli dalam bidang *data analysis*. Peserta survei harus mengisi form kuesioner untuk menentukan kategori peserta, lalu melakukan analisa data menggunakan *Workflow Designer* sesuai dengan skenario yang diberikan. Hasil survei akan diambil dari waktu pengerjaan analisa dari masing - masing peserta dan *feedback* dari peserta mengenai penggunaan aplikasi yang dibuat dalam penelitian skripsi ini.

5.1. Hasil Pengujian Aplikasi

Pengujian skripsi berhasil dilakukan oleh 12 subjek survei dengan hasil yang positif. Dari 12 subjek survei 3 orang dikategorikan sebagai pengguna awam, 6 orang dikategorikan sebagai pengguna pemula, 3 orang dikategorikan sebagai

pengguna ahli. Masing - masing subjek mengerjakan 2 skenario menggunakan aplikasi yang dibuat pada skripsi ini seperti yang tertulis di sub bab sebelumnya, dengan tambahan 3 orang pengguna ahli mengerjakan kedua skenario tersebut juga dengan *code* secara manual menggunakan *PySpark.* Dalam pengujian yang dilakukan, beberapa hal menjadi kriteria penilaian yang penting yaitu:

- Waktu pengerjaan
- Hasil kode (*running time* dan panjang kode)
- Penilaian kualitatif dari subjek survei

5.1.1. Hasil Pengujian Waktu Pengerjaan Skenario

Pengujian waktu pengerjaan skenario dilakukan dengan cara membandingkan lama waktu pengerjaan skenario antara menggunakan aplikasi yang dibuat dalam skripsi ini dengan menggunakan kode *PySpark* secara manual. Disini, diambil 2 orang dari kategori peserta survei ahli untuk melakukan kembali skenario menggunakan *PySpark* dalam *Jupyter Notebook*.

Waktu pengerjaan skenario pertama menggunakan aplikasi yang dibuat dalam skripsi ini dapat dilihat pada Tabel 1 di bawah dan waktu pengerjaan skenario pertama menggunakan PySpark dapat dilihat pada Tabel 2.

Tabel 1. Waktu Pengerjaan Skenario Pertama Menggunakan SparkFlow

Awam		Pemula		Ahli	
Nomor Peserta	Durasi (Menit)	Nomor Peserta	Durasi (Menit)	Nomor Peserta	Durasi (Menit)
C-1	35 (Tidak Selesai)	B-1	55 (Tidak Selesai)	E-1	48
C-2	50 (Tidak Selesai)	B-2	60	E-2	78
C-3	41	B-3	79	E-3	45
		B-4	70		
		B-5	30		
		B-6	50		

Tabel 2. Waktu Pengerjaan Skenario Pertama Menggunakan PySpark

Ahli	
Nomor Peserta	Durasi (Menit)
E-2	98
E-3	48

Selanjutnya, data - data untuk skenario kedua ditampilkan dalam Tabel 3 dan Tabel 4.

Tabel 3. Waktu Pengerjaan Skenario Kedua Menggunakan SparkFlow

Awam		Pemula		Ahli	
Nomor Peserta	Durasi (Menit)	Nomor Peserta	Durasi (Menit)	Nomor Peserta	Durasi (Menit)
C-1	-	B-1	30 (Tidak Selesai)	E-1	11
C-2	29 (Tidak Selesai)	B-2	-	E-2	26
C-3	11	B-3	30	E-3	21
		B-4	-		
		B-5	35		
		B-6	25		

Tabel 4. Waktu Pengerjaan Skenario Kedua Menggunakan PySpark

Nomor Peserta	Durasi (Menit)
E-2	27
E-3	13

Dapat disimpulkan bahwa rata - rata pengerjaan skenario pertama adalah 53.4 menit sedangkan untuk skenario kedua adalah 23.5 menit. Jika dibandingkan, peserta yang ahli memerlukan waktu 98 menit dan 48 menit untuk skenario pertama dan 27 menit dan 13 menit untuk skenario kedua.

5.1.2. Hasil Pengujian Hasil Kode Skenario

Pengujian berikutnya dilihat dari efektivitas pengerjaan *data analysis* jika dibandingkan antara penggunaan aplikasi workflow dengan PySpark. Efektivitas akan dilihat dari jumlah *node* yang digunakan dalam aplikasi workflow dibandingkan dengan jumlah baris kode dalam PySpark. Tabel.5 dan Tabel 6 berikut ini menampilkan data jumlah node dalam pengerjaan skenario pertama dan kedua menggunakan aplikasi workflow yang dibuat oleh peserta survei.

Tabel 5. Jumlah Node dalam Pengerjaan Skenario Pertama

Awam		Pemula		Ahli	
Nomor Peserta	Jumlah Node	Nomor Peserta	Jumlah Node	Nomor Peserta	Jumlah Node
C-1	12 (Tidak Selesai)	B-1	9 (Tidak Selesai)	E-1	29
C-2	14 (Tidak Selesai)	B-2	31	E-2	33
C-3	25	B-3	32	E-3	27
		B-4	30		
		B-5	27		
		B-6	27		

Tabel 6. Jumlah Node dalam Pengerjaan Skenario Kedua

Awam		Pemula		Ahli	
Nomor Peserta	Jumlah Node	Nomor Peserta	Jumlah Node	Nomor Peserta	Jumlah Node
C-1	-	B-1	-	E-1	17
C-2	19 (Tidak Selesai)	B-2	-	E-2	19
C-3	19	B-3	20	E-3	18
		B-4	-		
		B-5	15		
		B-6	18		

Selanjutnya, data - data di atas dibandingkan dengan jumlah baris yang diperlukan untuk menyelesaikan skenario pertama dan kedua menggunakan PySpark oleh 2 orang peserta survei. Data - data tersebut ditampilkan dalam Tabel 7 dan Tabel 8 di bawah ini.

Tabel 7. Jumlah Baris dalam Pengerjaan Skenario Pertama

Nomor Peserta	Jumlah Baris
E-2	95
E-3	67

Tabel 8. Jumlah Baris dalam Pengerjaan Skenario Kedua

Nomor Peserta	Jumlah Baris
E-2	46
E-3	30

Tabel 9. Jumlah Baris Esensial Skenario Pertama

Nomor Peserta	Jumlah Baris
E-2	38
E-3	32

Tabel 10. Jumlah Baris Esensial Skenario Kedua

Nomor Peserta	Jumlah Baris
E-2	23
E-3	12

Jumlah baris dan jumlah node tentu bukanlah parameter yang definitif untuk mengukur tingkat efektivitas suatu algoritma

analisis data, namun dengan menghitung hal tersebut dapat dilakukan perkiraan seberapa efektif aplikasi workflow dalam mengerjakan skenario analisis data. Supaya dapat melakukan berbandingan dengan lebih adil, Tabel 9 dan Tabel 10 menampilkan jumlah baris yang esensial saja (baris kode yang melakukan sebuah operasi/manipulasi data). Dalam hal ini, aplikasi workflow lebih unggul untuk membuat proses yang lebih efektif.

5.1.3. Hasil Pengujian Penilaian Kualitatif dari Subjek Survei

Pada akhir survei, peserta diberikan sebuah *feedback form* untuk menilai pengalaman pengguna dan potensi dari aplikasi SparkFlow.

Dari ke-12 peserta survei, secara garis besar memberikan respon yang positif. Rata - rata peserta survei menjawab 4.4 dari 5 dan 4.1 dari 5 untuk kemudahan menggunakan aplikasi SparkFlow untuk mengerjakan skenario pertama dan kedua secara berturut - turut.

Bagi peserta yang menjawab telah memiliki pengalaman sebelumnya, SparkFlow sangat membantu dalam mempersingkat waktu dan mengurangi waktu untuk membuka dokumentasi kode.

Semua peserta yang menjawab tidak memiliki pengalaman analisis data sebelumnya merasa dimudahkan dan tertarik untuk mempelajari lebih dalam dengan adanya aplikasi SparkFlow. Menurut peserta survey, dari segi pengalaman pengguna (*user experience)* aplikasi ini sudah ramah bagi pengguna awam dan pemula, hanya diperlukan waktu untuk membiasakan diri saja.

Salah satu kekurangan yang banyak disebut adalah halaman *Project* dan *Dashboard* yang terpisah sehingga harus bolak - balik antara kedua halaman. Selain itu, bagi pengguna awam lebih baik jika ada panduan penggunaan pada aplikasinya.

Aplikasi SparkFlow disebut sangat berpotensi jika dikembangkan secara terus - menerus. Pemula hingga profesional dapat diuntungkan dengan kemudahan dan kesederhanaan dari penggunaan aplikasi SparkFlow.

6. KESIMPULAN

Dari hasil survei aplikasi, dapat diambil kesimpulan sebagai berikut:

1. Waktu yang dibutuhkan bagi pengguna ahli untuk menyelesaikan sebuah skenario di antara menggunakan aplikasi workflow sedikit lebih cepat dari pada menggunakan kode PySpark dengan rata – rata waktu penyelesaian skenario pertama 53.4 menit, sedangkan peserta ahli membutuhkan waktu 98 menit dan 48 menit untuk E-2 dan E-3 secara berturut – turut. Hal ini semakin benar untuk pengguna pemula yang tidak memiliki pengalaman dalam melakukan *coding* dalam PySpark. Waktu yang dibutuhkan untuk belajar dan membiasakan diri dengan aplikasi workflow jauh lebih cepat sebab pengalaman pengguna dibuat mudah, sederhana, dan ramah untuk pengguna pemula.
2. Workflow yang dihasilkan menggunakan aplikasi workflow jauh lebih efektif daripada hasil kode PySpark untuk melakukan skenario yang sama. Hal ini diukur menggunakan jumlah *node* dalam workflow dibandingkan dengan jumlah baris dalam kode. Satu *node* dalam sebuah workflow dapat melingkupi beberapa proses sekaligus, sehingga dapat mempersingkat panjang workflow.
3. Tampilan workflow beserta dengan pengalaman pengguna yang bagus berdampak positif dalam pengerjaan *big data analysis* khususnya bagi pengguna pemula.
4. Aplikasi kurang berguna bagi peserta yang terlalu awam, sebab tetap dibutuhkan pengetahuan dasar dalam beberapa istilah - istilah seperti "*Group By"* dan *"Join"*. Selain itu, bagi orang yang belum pernah melakukan analisis data sebelumnya akan merasa kesulitan menyusun logika yang benar. Aplikasi akan mulai terasa mudah dan sederhana jika pengguna paling tidak memiliki pengetahuan dasar dalam bidang *data analysis*.
5. Aplikasi berupa workflow berpotensi untuk menjadi platform bagi pengguna pemula untuk memulai melakukan *big data analysis* dengan mudah. Tampilan yang sederhana dan kemudahan untuk melakukan manipulasi data dengan *drag-and-drop* tanpa perlu melakukan *coding* membuat pengguna pemula dapat langsung melakukan analisis tanpa perlu belajar *coding* terlalu lama. Aplikasi ini juga dapat menjadi platform untuk edukasi bagi pengguna awam.

Berikut ini adalah beberapa saran yang dapat digunakan untuk mengembangkan penelitian ini lebih lanjut:

1. *Node - node* dibuat lebih *high-level* sehingga masing - masing *node* melakukan sesuatu secara spesifik dan bukan hanya memetakan satu fungsi dalam PySpark. Hal ini dikarenakan limitasi yang dihadapi untuk memanipulasi data yang lebih kompleks tanpa *coding*.
2. Menerapkan *error handling* yang baik sehingga pengguna dapat memahami error apa yang terjadi di dalam *workflow*.
3. Membuat *node* dapat dijalankan satu - persatu, sehingga jika ada sedikit perubahan dalam workflow tidak perlu menjalankan keseluruhan workflow.

7. DAFTAR PUSTAKA

[1] Chellappan S., and Ganesan D. 2018. Introduction to Apache Spark and Spark Core. In *Practical Apache Spark* (Berkeley, CA, December 13, 2018). Apress, Berkeley, CA. DOI= https://doi.org/10.1007/978-1-4842-3652-9_3

[2] Ford, A. 2017. *Big Data Workflow Automation is Closer Than You Think*. URI= https://www.nintex.com/blog/big-data-workflow-automation-closer-think/

[3] Kashyap, V. 2019. *Data Science Workflow: From Research Experiments to Business Use-Cases*. URI= https://www.progresstalk.com/threads/progress-news-progress-openedge-abl-data-science-workflow-from-research-experiments-to-business-use-cases.192803/

[4] Silva, B. N., Diyan, M., and Han, K. 2018. Big Data Analytics. In *Deep Learning: Convergence to Big Data Analytics* (Springer, Singapore, December 31, 2018). SpringerBriefs in Computer Science. Springer, Singapore. DOI= https://doi.org/10.1007/978-981-13-3459-7_2

[5] Ulunma. 2020. *Coca Cola Leverages Data Analytics to Drive Innovation*. URI= https://digital.hbs.edu/platform-digit/submission/coca-cola-leverages-data-analytics-to-drive-innovation/