MJI

Media Jurnal Informatika
Vol. 17 No 1, Juni 2025
http://jurnal.unsur.ac.id/mjinformatika

p-issn : 2088-2114
e-issn : 2477-2542

Sistem Pengelolaan dan Pemantauan Data Layanan Pengaduan Publik Menggunakan Algoritma *String Matching*

Najla Putri Afifah[1], Medhanita Dewi Renanti[2], Irma Rasita Gloria Barus[3]
[1,2,3]*Program Studi Teknologi Rekayasa Perangkat Lunak, Sekolah Vokasi IPB University*
najlaputri@apps.ipb.ac.id[1], medhanita@apps.ipb.ac.id[2], irmabarus@apps.ipb.ac.id[3]

Abstract

Public complaint services in Tangerang City, which are currently managed using Microsoft Excel through the LAKSA application, face several challenges. The primary issue lies in data processing, which is time-consuming and prone to errors. This study aims to develop the Report Aspirations Communications and Sentiment (Re-Actions) application, featuring automated categorization using a string matching algorithm to enhance the complaint classification process and speed it up. The development of this application follows the Scrum method, ensuring a structured approach that can adapt to changing user needs. The system includes a real-time dashboard that allows administrators to monitor and analyze public complaint data easily. Testing using the black-box method indicates that all features—including login, data management, dashboard visualization, and categorization by label and subdistrict—function as expected. This system has significantly improved the complaint classification process by employing the string matching algorithm for categorization. The algorithm efficiently identifies and displays the position of patterns within the text, ultimately making public complaint management more efficient and accurate.

Keywords: blackbox testing, public complaint website, public service, scrum, string matching

Abstrak

Pelayanan pengaduan publik di Kota Tangerang yang selama ini dikelola menggunakan Microsoft Excel melalui aplikasi LAKSA mengalami berbagai kendala. Kendala utamanya dalam pengolahan data yang memerlukan waktu lama dan rentan kesalahan. Penelitian ini bertujuan mengembangkan aplikasi Report Aspirations Communications and Sentimen (Re-Actions) yang dilengkapi fitur kategorisasi otomatis menggunakan algoritma string matching untuk meningkatkan dan mempercepat proses klasifikasi pengaduan. Pengembangan aplikasi ini menggunakan metode Scrum, yaitu pendekatan terstruktur yang mampu beradaptasi dengan perubahan kebutuhan pengguna. Sistem ini dilengkapi dashboard real-time yang memudahkan admin memantau dan menganalisis data pengaduan masyarakat. Pengujian yang dilakukan menggunakan metode blackbox, hasil menunjukkan bahwa semua fitur berfungsi sesuai kebutuhan, mulai dari login, pengelolaan data, dashboard visualisasi, hingga kategorisasi label dan kecamatan. Dengan menggunakan algoritma string matching untuk kategorisasi, sistem ini telah meningkatkan proses klasifikasi pengaduan secara signifikan. Algoritma ini mampu mengidentifikasi dan menampilkan posisi pola dalam teks secara efisien, sehingga pengelolaan pengaduan masyarakat menjadi lebih efisien dan akurat.

Kata kunci: pelayanan publik, pengujian blackbox, scrum, string matching, website pengaduan masyarakat

I. PENDAHULUAN

Pelayanan publik sering kali mendapat kritik dari masyarakat, terutama terkait efektivitas dan transparansi dalam penanganannya. Untuk meningkatkan kualitas layanan, pemerintah perlu mengadopsi prinsip akuntabilitas dan memanfaatkan teknologi digital guna mempercepat proses pengelolaan pengaduan [1]. Salah satu bentuk inovasi yang diterapkan adalah *e-government*, yang memungkinkan interaksi langsung antara masyarakat dan pemerintah melalui media sosial atau sistem pengaduan yang disediakan [2]. Pengelolaan pengaduan pelayanan publik telah diatur dalam Peraturan Presiden Nomor 76 Tahun 2013, yang mewajibkan setiap instansi pemerintah menyediakan sarana pengaduan yang mudah diakses serta memastikan prosesnya berjalan secara transparan, akuntabel, dan tepat waktu. Sebagai bagian dari implementasi kebijakan tersebut, Pemerintah Kota Tangerang telah melaksanakan *e-government* karena telah mengembangkan aplikasi Layanan Aspirasi Kotak Saran Anda (LAKSA) sebagai wadah penyampaian pengaduan masyarakat terkait infrastruktur, pelayanan publik, keamanan, dan berbagai isu lainnya [3].

Aplikasi LAKSA merupakan bentuk pelayanan publik responsif dari Pemerintah Kota Tangerang dalam menanggapi pengaduan masyarakat. Setiap aduan yang masuk dikelola oleh admin sebelum diteruskan ke Organisasi Perangkat Daerah (OPD), namun proses pengolahan data masih dilakukan secara manual menggunakan Microsoft Excel [4].

Pada aplikasi LAKSA terdapat permasalahan terkait dengan jumlah admin yang terbatas, sementara jumlah aduan yang masuk setiap tahunnya terus meningkat dan

melebihi kapasitas yang dapat ditangani oleh admin, ini menjadi kendala utama yang menghambat efisiensi waktu pengelolaan data [5]. Sistem sebelumnya belum dilengkapi dengan fitur dan detail yang memadai untuk mengelola layanan publik, sehingga proses seperti kategorisasi penentuan label, dan kategorisasi lokasi kecamatan masih harus dilakukan secara manual. Berdasarkan permasalahan tersebut, dikembangkan aplikasi *Report Aspirations Communications and Sentimen* (Re-Actions) sebagai solusi yang diharapkan.

Dalam pengembangan sistem ini menggunakan metode scrum, yang dirancang untuk menangani proyek dengan kebutuhan yang kompleks dan dapat berubah sewaktu-waktu sesuai dengan kebutuhan pengguna. Scrum tidak hanya digunakan dalam pengembangan aplikasi, tetapi juga dalam manajemen proyek. *Project manager*, *scrum master*, dan tim pengembang berkolaborasi untuk mengontrol dan menyelesaikan tugas secara efisien. Metode ini Metode ini mengutamakan tiga pilar utama, yaitu transparansi yang memastikan seluruh proses dapat dipantau oleh semua pihak terkait, inspeksi yang dilakukan secara rutin untuk mengidentifikasi penyimpangan hasil, dan adaptasi yang mendorong perubahan cepat agar tujuan tetap tercapai [6] dengan begitu, memungkinkan kerja tim yang lebih terstruktur dan fleksibel, sehingga setiap anggota bekerja sebagai satu unit dalam mencapai tujuan bersama [7]. Sistem ini dibuat menggunakan bahasa pemrograman PHP *framework CodeIgniter* 4. *Framework* ini mempermudah dan mempercepat pengembangan aplikasi *web*. *Framework* ini gratis, ringan, dan mendukung PHP 4 serta versi terbaru. Dengan arsitektur *Model View Controller* (MVC), *CodeIgniter* memisahkan logika bisnis, tampilan, dan pengendali, sehingga menghasilkan kode yang terstruktur dan mudah dipelihara [8]. *Database Management System* yang digunakan dalam pembuatan sistem ini adalah MySQL.

Penelitian ini bertujuan untuk membuat sistem yang dapat meningkatkan efisiensi waktu dan mempercepat proses pengolahan data. Sistem ini dilengkapi dengan fitur *dashboard* yang memungkinkan admin memantau dan mengelola data secara *real-time* dan pengelompokan kategorisasi label, dan kecamatan dengan menerapkan algoritma *string matching* berdasarkan isi aduan yang masuk. Dengan adanya *sistem Report Aspirations Communications and Sentimen* (Re-Actions) diharapkan proses *preprocessing data* pengaduan dapat berlangsung lebih cepat sehingga pengaduan dapat segera diteruskan kepada Organisasi Perangkat Daerah (OPD) terkait untuk ditindaklanjuti.

II. METODE PENELITIAN

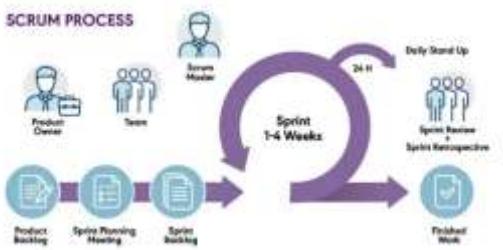
Pengelompokan label dan kecamatan dalam sistem ini menggunakan algoritma *string matching* berbasis *regular expressions*. *String matching* merupakan proses pencocokan pola dalam data teks untuk menemukan kemunculan substring yang relevan. Algoritma ini dapat digunakan untuk mencari satu atau lebih pola sekaligus dalam sebuah teks [9]. Salah satu pendekatan yang umum digunakan dalam *string matching* adalah pemanfaatan *regular expressions* (*regex*). Seperti dijelaskan oleh [10] fungsi manipulasi *string* berbasis *regex* secara luas digunakan dalam pencarian *substring* dan transformasi *string* karena kemampuannya dalam mendefinisikan pola secara deklaratif dan fleksibel. *Regex* sendiri merupakan format penulisan khusus untuk merancang pola pencarian dalam teks, sehingga memudahkan pengguna menemukan karakter atau kata sesuai kriteria tertentu [11] .

Gambar 1 merupakan bentuk formal dari *query* pencarian menggunakan *regular expressions* yang memungkinkan pencocokan pola fleksibel terhadap kemunculan kata kunci "*regular*" dan "*expression*" dalam berbagai urutan dan posisi. Seperti dijelaskan oleh [12], pola ini diterjemahkan dari *string* pencarian pengguna menjadi *regular expressions* yang secara otomatis menyeleksi dokumen relevan berdasarkan kecocokan terhadap kondisi pola tertentu, dengan pendekatan *state machine*, yaitu sistem yang membaca simbol demi simbol dari teks dan menentukan apakah urutan tersebut mengarah ke keadaan *accept* (diterima) atau *reject* (ditolak). Dengan demikian, hanya *string* atau dokumen yang mengandung susunan kata sesuai dengan pola dalam regex yang akan dianggap sebagai hasil pencarian yang valid. Pendekatan ini tidak hanya meningkatkan akurasi pencarian, tetapi juga memberikan fleksibilitas tinggi dalam menjangkau berbagai kemungkinan bentuk penulisan atau struktur kalimat dalam data teks.

$$(\Sigma^*\texttt{regular}\Sigma^*\texttt{expression}\Sigma^*)^* \cup (\Sigma^*\texttt{expression}\Sigma^*\texttt{regular}\Sigma^*)^*$$

Gambar 1: *Regular Expressions*

Metode yang digunakan adalah scrum yang merupakan kerangka kerja dalam metodologi Agile yang dirancang oleh Jeff Sutherland pada tahun 1990 [13] dan dikembangkan lebih lanjut oleh Schwaber dan Beedle. Kerangka ini membantu menyelesaikan masalah kompleks secara fleksibel, menghasilkan produk berkualitas secara efisien dan kreatif (Schwaber dan Sutherland 2020). Scrum memungkinkan tim untuk menghadapi tantangan yang terus berubah sekaligus memaksimalkan produktivitas dan kreativitas. Tahapan scrum dapat dilihat pada Gambar 2.



Gambar 2: Tahapan Metode Scrum

Menurut [14] ada beberapa tahapan dalam metode scrum, sebagai berikut:

1. *Product backlog*, tahap *product backlog* adalah proses mengumpulkan dan menyusun daftar prioritas kebutuhan yang diperlukan untuk pengembangan atau perubahan perangkat lunak. Daftar ini mencakup deskripsi singkat fitur atau fungsi yang dibutuhkan pengguna. Pemilik produk bertanggung jawab mengelola dan memprioritaskan daftar ini

berdasarkan tingkat pentingnya bagi pengguna dan dampaknya terhadap bisnis. Tujuannya adalah memastikan tim fokus pada tugas yang paling bernilai terlebih dahulu.

2. *Sprint planning*, *Sprint planning* adalah proses menentukan tujuan *sprint* (*sprint goals*) dan memilih item prioritas dari *product backlog* yang akan dikerjakan selama proses *sprint*. Kegiatan ini melibatkan kolaborasi antara scrum master, *product owner*, dan tim *developer* untuk meninjau, memahami, dan merencanakan tugas yang akan dimasukkan ke dalam *sprint backlog*. *Product owner* menyajikan item prioritas tertinggi berdasarkan masukan *stakeholder*, sementara tim *developer* memastikan pemahaman yang jelas tentang kebutuhan tersebut.
3. *Sprint*, *sprint* adalah periode waktu 1-4 minggu, biasanya 2 minggu untuk menyelesaikan *sprint backlog* dalam pengembangan perangkat lunak. Tim *developer* bekerja sama dengan *product owner* dan *scrum master* untuk mencapai tujuan *sprint* yang memberikan nilai bagi pengguna dan bisnis. Di akhir sprint, hasil kerja disajikan kepada *stakeholder* dalam tinjauan sprint. Jika tujuan tercapai, sprint selesai. Jika tidak, tim dan *product owner* mengevaluasi langkah berikutnya, seperti memperpanjang sprint atau memprioritaskan ulang *backlog*.
4. *Daily scrum* membantu tim meningkatkan komunikasi, koordinasi, dan kolaborasi. Rapat harian ini memungkinkan tim untuk mengidentifikasi dan mengatasi masalah atau hambatan secara cepat sebelum berkembang menjadi lebih besar.
5. *Sprint review*, adalah pertemuan di akhir sprint yang melibatkan tim scrum untuk menilai kemajuan produk, meninjau fitur baru atau perbaikan, dan mengumpulkan masukan dari pemangku kepentingan. Tujuannya adalah mengevaluasi produk.
6. *Sprint Retrospective* dilakukan di akhir sprint setelah *sprint review*. Pada tahap ini, tim scrum membahas perbaikan terkait orang dan hubungan, proses, serta *definition of done*. Tujuannya untuk mengatasi hambatan dan masalah yang ada, serta melakukan perbaikan untuk meningkatkan efisiensi dan produktivitas di *sprint* berikutnya. memenuhi kebutuhan pemangku kepentingan.
7. Pengujian
   Pengujian sistem Re-Actions menggunakan *blackbox testing*, metode ini bertujuan memverifikasi keluaran aplikasi berdasarkan data masukan untuk memastikan bahwa fungsionalitasnya sesuai dengan kebutuhan dan persyaratan yang telah ditentukan. Pengujian ini berfokus pada antarmuka aplikasi, fungsi yang tersedia, serta kesesuaian alur kerja dengan harapan pengguna. Tahapan dalam metode ini mencakup pembuatan *test case* untuk menguji berbagai fungsi aplikasi, pengembangan *test case* guna mengevaluasi apakah alur kerja setiap fungsi sesuai dengan kebutuhan pengguna, serta identifikasi *bug* atau kesalahan yang tampak pada antarmuka aplikasi [15].

III. HASIL PENELITIAN

1. *Product Backlog*
   Product backlog pada *website Re-Actions* berisi rancangan pekerjaan selama sprint berlangsung. *Product backlog* terdiri atas daftar fitur, estimasi waktu selesai, dan prioritas secara terurut dari skala prioritas tertinggi. Pada *product backlog website* Re-actions terdapat 6 *product backlog* yang dapat dilihat pada Tabel 1 berikut.

Tabel 1: Daftar *product backlog*

No	Fitur	Estimasi	Prioritas
1	Pembuatan diagram sistem	15 hari	Sedang
2	Autentikasi *login*	8 hari	Sedang
3	Tabel data pengaduan	15 hari	Tinggi
4	*Dashboard*	10 hari	Tinggi
5	Kategorisasi label pengaduan	14 hari	Tinggi
6	Kategorisasi kecamatan	10 hari	Tinggi
	Total	72 hari	

2. *Sprint planning*

Sprint planning yang dijelaskan pada panduan scrum adalah pertemuan yang dilakukan di awal setiap *sprint* untuk menentukan pekerjaan yang akan dilakukan dalam *sprint* tersebut. Dalam sesi ini, tim scrum berkolaborasi untuk memilih *item* dari *product backlog* yang akan dikerjakan dan menyusun rencana pekerjaan tersebut akan diselesaikan. Rancangan pada tahap *sprint planning* 1 merupakan hasil kolaborasi seluruh tim yang menghasilkan *sprint backlog* yang ditunjukkan pada Tabel 2.

Tabel 2: Daftar *Sprint Planning*

No	*Sprint*	*Backlog Item*	*Task*
1	*Sprint* 1	Pembuatan diagram sistem	Membuat visualisasi dan interaksi antara sistem dan aktor menggunakan *use case* diagram
			Pembuatan alur sistem dengan menggunakan *activity diagram*
		Autentifikasi *login*	Membuat *form username*, *password*, dan validasi *input*
2	*Sprint* 2	Tabel data pengaduan	*Import* data dan validasi *file* .xsxl
			Pembuatan filter tanggal dan filter data
			Tambah, edit, hapus data pengaduan
			Search, *pagination*, dan *sort*
3	*Sprint* 3	*Dashboard*	Filter tanggal
			Membuat akumulasi total

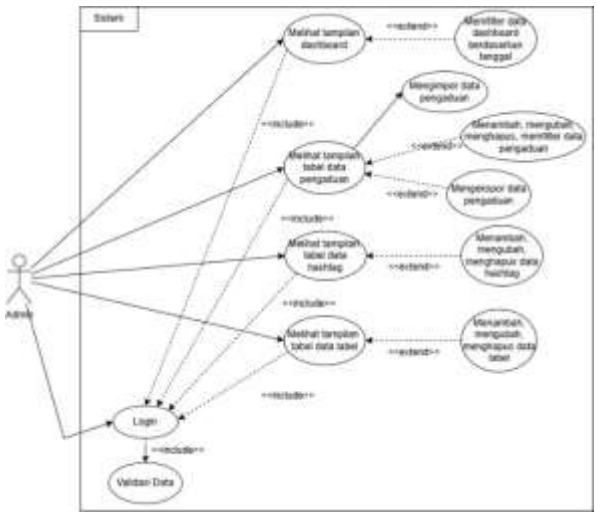
No	*Sprint*	*Backlog Item*	*Task*
			data pengaduan
			Membuat akumulasi total data sentimen positif, netral, dan negatif
			Membuat grafik sentimen pengaduan berdasarkan tanggal
			Membuat b*ar chart* 5 OPD dengan laporan tertinggi
			Membuat *pie chart* sentimen dan status laporan
			Membuat *pie chart* sentimen berdasarkan kecamatan
4	*Sprint* 4	Kategorisasi label	Membuat tambah, edit, hapus daftar OPD dan *hashtag*
			Search, pagination, dan *sort*
			Membuat *alert* ketika data OPD dan *hashtag* yang dimasukkan sudah tersedia
			Membuat tambah, edit, hapus data
			Membuat *Dynamic dropdown* untuk daftar OPD dan *hashtag*
			Membuat *multi keyword* untuk melakukan *string matching* berasarkan isi pengaduan
			Membuat kategorisasi label pada data pengaduan berdasarkan kecocokan OPD dan *hashtag*
		Kategorisasi kecamatan	Membuat kategorisasi kecamatan berdasarkan kecocokan dari lokasi

3. Hasil *Sprint*

Pada hasil *sprint* 1 ini, salah satu fungsionalitas yang berhasil diselesaikan adalah fitur *login* untuk admin (Gambar 3). Admin dapat mengakses sistem dengan memasukkan *username* dan *password* sesuai akun yang didaftarkan. Proses *login* ini memastikan hanya pengguna yang mempunyak akses yang dapat menggunakan sistem. Selain itu, sistem juga telah dilengkapi dengan mekanisme validasi yang akan menampilkan pesan peringatan secara jelas apabila admin memasukkan username atau *password* yang tidak sesuai dengan *database*. Dengan adanya peringatan ini, admin akan mendapatkan umpan balik yang informatif, sehingga dapat segera memperbaiki kesalahan *input* dan mencoba *login* kembali.

Gambar 3: Tampilan *Login*

Use case diagram adalah salah satu jenis diagram dalam *Unified Modeling Language* (UML) yang digunakan untuk memvisualisasikan interaksi antara sistem lain atau aktor pengguna dengan sistem yang sedang dikembangkan. Diagram ini menggambarkan hubungan antara pihak-pihak yang berinteraksi aktor dan serangkaian fungsi atau layanan *use case* yang disediakan oleh sistem. Melalui *use case* diagram, pengembang dan pemangku kepentingan dapat memahami ruang lingkup sistem, peran masing-masing aktor, serta aktivitas atau proses yang dapat dilakukan, sehingga membantu dalam merancang sistem yang sesuai dengan *requirement gathering* dan tujuan bisnis [16]. Gambar 4 merupakan hasil pengembangan pada *sprint* 1 yang menggambarkan *use case* diagram sistem, meliputi alur dan fitur utama seperti tampilan *dashboard*, manajemen data pengaduan, pengelolaan *hashtag*, dan label.

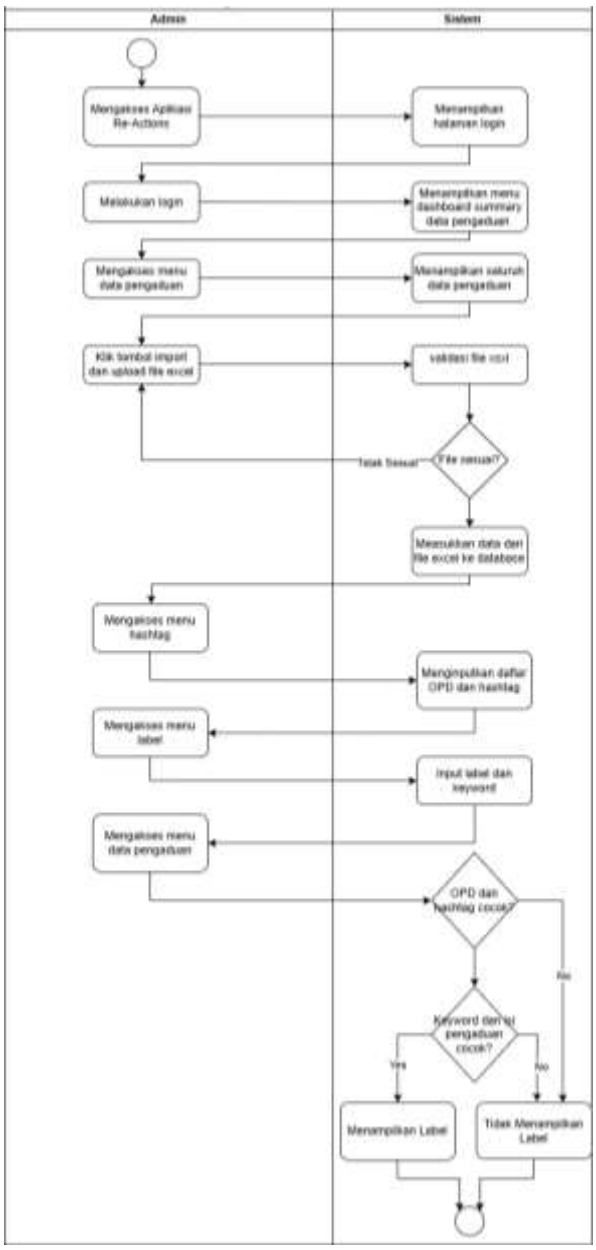


Gambar 4: *Use Case* Sistem Re-Actions

Activity diagram merupakan salah satu jenis diagram dalam UML yang digunakan untuk memodelkan alur aktivitas atau proses bisnis dalam sistem informasi. Diagram ini memberikan gambaran visual mengenai urutan aktivitas, awal dan akhir proses, serta transisi antar aktivitas yang terjadi dalam suatu *workflow* [17]. Gambar 5 menggambarkan alur aktivitas admin saat menggunakan sistem Re-Actions, mulai dari *login*, mengelola data pengaduan, mengimpor *file* Excel, mengakses menu *hashtag* dan label, hingga proses pencocokan OPD, *hashtag*, dan *keyword*. Sistem akan memvalidasi *file* yang diunggah dan menampilkan label jika data sesuai, atau tidak menampilkan label jika tidak ada kecocokan.

Halaman data pengaduan pada *sprint* 2 ditunjukkan pada Gambar 6, yang memuat fitur untuk menambah, mengedit, menghapus, serta mengimpor data pengaduan ke dalam sistem. Selain itu, pada halaman ini juga disediakan fitur *filtering* yang cukup lengkap, memungkinkan *admin* untuk menyaring data berdasarkan rentang tanggal, status laporan, sumber pengaduan, kecamatan, dan *hashtag*. Proses penyaringan ini menggunakan komponen *checkbox* yang memudahkan *admin* dalam memilih lebih dari satu kriteria secara bersamaan tanpa perlu mengetik secara manual. Dengan adanya fitur ini, pengelolaan data pengaduan menjadi lebih efisien, terstruktur, dan mendukung kebutuhan pencarian data secara spesifik sesuai kebutuhan analisis atau tindak lanjut laporan.

Dashboard yang dikembangkan pada *sprint* 3 ditunjukkan pada Gambar 7, bertujuan untuk membantu admin memantau perkembangan layanan publik secara menyeluruh. Fitur ini menyajikan data angka, grafik, dan progres layanan, seperti total pengaduan, total sentimen, diagram tren pengaduan, OPD dengan laporan terbanyak, serta distribusi sentimen dan status berdasarkan kecamatan dan kanal pengaduan. Dengan fitur ini, admin dapat melihat ringkasan informasi penting secara cepat dan akurat.



Gambar 5: *Activity Diagram* Sistem Re-Actions

Gambar 6: Tampilan Data Pengaduan

Gambar 7: Tampilan *Dashboard*

Berikut adalah hasil *sprint* 4 alur algoritma *string matching* berbasis *regex*:

1. Admin mengkases halaman hashtag dan menginputkan data hashtag yang terdiri dari daftar OPD dan hashtag yang sesuai. Di dalam halaman ini admin dapat melakukan input data, edit data, dan hapus data dan sudah dilengkapi dengan *filtering* dan *seraching* yang dapat ditunjukkan pada Gambar 8 (tampilan menu *hashtag*) dan Gambar 9 (Contoh Hasil Penginputan OPD dan Hashtag).

Gambar 8: Tampilan Menu *Hashtag*

Gambar 9: Contoh Hasil Penginputan OPD dan Hashtag

2. Setelah admin memasukkan data pada menu *hashtag*, *admin* dapat mengakses menu label untuk melakukan pengelompokan lebih lanjut. Pada menu ini, *admin* hanya perlu memilih OPD dan *hashtag* karena data tersebut sudah terhubung otomatis dengan menu *hashtag*. Sementara itu, untuk bagian label dan *keyword*, admin masih perlu memasukkan data secara manual. Pada kolom *keyword*, admin dapat memasukkan lebih dari satu *keyword* untuk satu label. Selain itu, admin juga dapat melakukan *input*, edit, dan hapus data sesuai kebutuhan. Menu ini juga telah dilengkapi dengan fitur *filtering* dan *searching* untuk memudahkan pencarian data, seperti ditunjukkan pada Gambar 10 (tampilan pemasukkan menu label) dan Gambar 11 (contoh hasil pemasukkan menu label).

Gambar 10: Tampilan Pemasukkan Menu Label

Gambar 11: Contoh Hasil Pemasukkan Menu Label

3. Setelah semua data disimpan, sistem akan memproses konfigurasi yang telah dibuat untuk digunakan dalam pencocokan pengaduan dengan menggunakan *string matching*. Setiap pengaduan yang masuk akan dianalisis dan dicocokkan dengan daftar *keyword*, *hashtag*, dan tujuan OPD yang telah ditentukan. Jika sesuai, pengaduan tersebut akan masuk ke dalam kategori label yang telah dibuat oleh admin, sehingga proses kategorisasi dapat berjalan lebih cepat dan efisien tanpa perlu dilakukan secara manual berulang kali.

Gambar 12 merupakan hasil dari Algoritma *String Matching* berbasis *Regex,* disini label yang dihasilkan sesuai karena *keyword*, *hashtag*, dan tujuan OPD sesuai dengan isi pengaduan.

Gambar 12: Tampilan Hasil *String Matching*

4. *Daily Scrum*

Tahapan *daily scrum* dilakukan setiap harinya selama 15 menit pada saat *sprint* belangsung. *Daily scrum* dilakukan bersama tim *developer* secara tatap muka atau daring. Tim *developer* sistem *Re-Actions* beranggotakan dua orang. Pertemuan *daily scrum* secara daring dilakukan jika terkendala untuk tatap muka. Dalam pertemuan ini, masing-masing anggota tim memberikan *update* mengenai kemajuan tugas yang telah dikerjakan, rencana pekerjaan selanjutnya, serta kendala yang dihadapi.

5. *Sprint Review*

Pada tahap *sprint review*, tim mengevaluasi hasil kerja setiap *sprint* untuk memastikan standar kualitas dan kesesuaian dengan kebutuhan proyek. Dalam sesi ini, tim mendemonstrasikan fitur atau produk kepada *product owner* dan *stakeholder* untuk mendapatkan masukan. Jika ada perbaikan yang diperlukan, tim mencatatnya sebagai pertimbangan untuk *sprint* berikutnya. Daftar *sprint review* ditampilkan pada Tabel 3.

Tabel 3: Daftar *Sprint Review*

No	*Sprint*	Evaluasi
1	*Sprint* 1	Proses *login* yang gagal belum disertai *alert* atau pesan yang menjelaskan penyebab kegagalan
2	*Sprint* 2	Belum tersedia *alert* ketika proses *import* data dilakukan, tidak ada konfirmasi status berhasil atau gagal
3	*Sprint* 3	Belum tersedia *bar chart* OPD dengan laporan tertinggi
4	*Sprint* 4	Teks *alert* pada menu *hashtag* masih menggunakan Bahasa Inggris

6. *Sprint Retrospective*

Tahap *sprint retrospective* dilakukan bersama oleh *product owner*, *scrum master*, dan tim *developer* untuk mengevaluasi terkait bagian yang perlu ditingkatkan pada sistem *Re-Actions.* Daftar *sprint restrospective* ditampilkan pada Tabel 4.

Tabel 4: Daftar *Sprint Retrospective*

No	*Sprint*	Evaluasi
1	*Sprint* 1	Menambahkan *alert* pada proses *login* untuk menampilkan pesan penyebab kegagalan login
2	*Sprint* 2	Menambahkan *alert import* data ketika data berhasil atau gagal
3	*Sprint* 3	Menambahkan *bar chart* OPD laporan tertinggi
4	*Sprint* 4	Mengubah kalimat pada *alert* menu hashtag menjadi bahasa indonesia

7. *Blackbox Testing*

Blackbox testing adalah pengujian data yang dijalankan perangkat lunak untuk memastikan bahwa perangkat lunak berfungsi dengan baik [18]. Tabel 5 merupakan hasil dari *blackbox testing* yang telah dilakukan. Berdasarkan hasil pengujian, seluruh unit berhasil diuji dan aplikasi berjalan dengan baik sesuai dengan fungsionalitas yang diharapkan.

Tabel 5: *Blackbox Testing*

Kode	Skenario Pengujian	Hasil yang diharapkan	Hasil
TU-01	Pengguna melakukan login dengan memasukkan *username* dan *password* dengan benar	Sistem menampilkan *landing page*	Sesuai
TU-02	Pengguna memasukkan *username* dan *password* salah	Menampilkan *alert* "*username* dan *password* salah"	Sesuai
TU-03	Melakukan *Import* data Excel	Data pengaduan tersimpan dan tampil sesuai kolom	Sesuai
TU-04	*Search, pagination*, dan *sort*	Data tampil sesuai pencarian, dapat melakukan *next, previous*, dan *sort*	Sesuai
TU-05	Melakukan *filter* data sesuai tanggal	Dapat mengambil data sesuai dengan data yang dimasukkan oleh pengguna	Sesuai
TU-06	Melakukan pemasukkan data, edit, dan hapus di menu data pengaduan	Jumlah data bertambah jika data ditambahkan, berubah jika dilakukan edit data, dan berkurang jika dihapus	Sesuai
TU-07	Melakukan filter data sesuai tanggal	Dapat mengambil data sesuai dengan data yang dimasukkan oleh pengguna	Sesuai
TU-08	Menambahkan data pengaduan dan sentimen	Menampilkan informasi jumlah pengaduan dan *sentimen*	Sesuai
TU-09	Menambahkan data pengaduan beserta tanggal	Menampilkan jumlah *sentimen* berdasarkan tanggal laporan berupa grafik	Sesuai
TU-10	Menambahkan data OPD	Menampilkan informasi berupa *bar chart* 5 OPD dengan laporan tertinggi	Sesuai

Kode	Skenario Pengujian	Hasil yang diharapkan	Hasil
TU-11	Menambahkan data status laporan, kecamatan, dan kanal pengaduan	Menampillkan status laporan, sentimen berdasarkan kecamatan dan kanal pengaduan dengan *pie chart*	Sesuai
TU-12	Melakukan tambah, edit, hapus daftar OPD dan *hashtag*	Jumlah data bertambah jika data ditambahkan ,berubah jika dilakukan edit data, dan berkurang jika dihapus	Sesuai
TU-13	Menginputkan OPD dan *hashtag* yang sudah ada	Menampilkan *alert* "data duplikat"	Sesuai
TU-14	Melakukan tambah dan edit data label, OPD, dan *hashtag*	Menampilkan *hashtag* yang masuk ke dalam kategori OPD	Sesuai
TU-15	Menambah dan mengedit *keywords* lebih dari satu kata	Menampilkan pencocokan label yang sesuai dengan isi pengaduan jika kalimat sesuai dengan *keywords*	Sesuai
TU-16	Tujuan OPD, *hashtag* sesuai dengan isi pengaduan	Menampilkan label yang sesuai dengan isi pengaduan	Sesuai
TU-17	Menambahkan data alamat lengkap dengan kecamatan	Menampilkan kecocokan nama kecamatan dalam kolom alamat	Sesuai

IV. PEMBAHASAN

Berdasarkan hasil yang diperoleh dari pengembangan sistem Re-Actions, seluruh fitur yang direncanakan pada setiap sprint berhasil diselesaikan dan diuji menggunakan metode *blackbox testing*. Hasil pengujian menunjukkan bahwa *fitur* login, pengelolaan data pengaduan, *dashboard monitoring*, dan kategorisasi label dan kecamatan berjalan sesuai dengan kebutuhan pengguna.

Fitur *login* mempermudah admin dalam mengakses sistem dengan validasi yang jelas melalui pesan peringatan ketika terjadi kesalahan *input*. Fitur data pengaduan memungkinkan admin untuk menambah, mengedit, menghapus, dan mengimpor data, yang memudahkan pengelolaan informasi secara terstruktur.

Dashboard memberikan ringkasan data dalam bentuk angka dan grafik yang membantu pemantauan layanan secara menyeluruh. Sementara itu, fitur kategorisasi label dan kecamatan menggunakan algoritma *string matching* berbasis *regex* yang digunakan dalam sistem Re-Actions terbukti efektif dalam mengidentifikasi *keyword* yang sesuai dari isi pengaduan. Algoritma ini memungkinkan sistem melakukan pencocokan cepat dan akurat terhadap daftar *keyword hashtag*, dan OPD yang telah ditentukan oleh admin, sehingga mempercepat proses klasifikasi aduan yang sebelumnya dilakukan secara manual dan membantu meminimalisir waktu yang dibutuhkan untuk mengelola data. Dengan demikian, penerapan algoritma ini menyediakan solusi yang tepat untuk mengatasi keterbatasan fitur pengelolaan yang belum terdapat di aplikasi LAKSA.

Di sisi lain, penerapan metode Scrum juga memberikan dampak positif terhadap efektivitas pengembangan sistem, karena prosesnya berjalan secara bertahap dan dapat beradaptasi dengan perubahan kebutuhan pengguna. Setiap *sprint* yang dilaksanakan menghasilkan peningkatan fitur dan perbaikan yang relevan, sesuai dengan evaluasi yang dilakukan dalam *sprint review* dan *retrospective*. Hal ini membuktikan bahwa scrum mendorong kolaborasi tim dan ketepatan penyelesaian tugas dalam waktu yang terbatas.

Sistem ini juga memiliki keterbatasan khususnya dalam mengenali aduan yang menggunakan bahasa asing hal ini dapat menyebabkan beberapa label tidak terkategorisasi secara optimal, meskipun kasus penggunaan bahasa asing terbilang jarang terjadi, namun potensi kendala ini tetap perlu diperhatikan. Oleh karena itu, diharapkan pengembang selanjutnya dapat mempertimbangkan penambahan fitur pendukung multi bahasa atau mekanisme sederhana lainnya yang memungkinkan sistem mengenali variasi bahasa yang lebih luas. Upaya ini dapat menjadi pengembangan lanjutan bagi *programmer* berikutnya yang akan melanjutkan sistem ini, agar kedepannya sistem Re-Actions mampu menjangkau dan mengelola aduan dari masyarakat yang menggunakan berbagai istilah dan bahasa yang lebih beragam.

V. KESIMPULAN

Berdasarkan hasil pengembangan dan pengujian, sistem Re-Actions berhasil dibangun sesuai dengan tujuan penelitian, yaitu menyediakan fitur pengelolaan dan pemantauan data layanan pengaduan yang lebih efisien. Penerapan *string matching* pada fitur kategorisasi berhasil mempercepat proses klasifikasi pengaduan, sedangkan dashboard *real time* memudahkan admin dalam memantau data secara keseluruhan.

Seluruh fitur yang dikembangkan telah berhasil diuji dan berfungsi sesuai harapan. Penerapan metode Scrum juga terbukti efektif dalam memastikan proses pengembangan berjalan secara terstruktur dan responsif terhadap kebutuhan yang muncul selama proyek berlangsung. Keberhasilan ini membuka peluang untuk pengembangan lebih lanjut, seperti penambahan fitur analisis tren pengaduan atau integrasi dengan layanan pemerintah lainnya guna meningkatkan kualitas pelayanan publik.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih yang sebesar kepada Pemerintah Kota Tangerang, khususnya Dinas Komunikasi dan Informatika Kota Tangerang, atas izin dan dukungan yang telah diberikan dalam pengembangan *website* Re-Actions. Semoga hasil penelitian ini dapat memberikan manfaat dan kontribusi positif bagi berbagai pihak yang membutuhkan.

VI. REFERENSI

[1] W. A. Yohanitas, "Media Pengembangan Ilmu dan Praktek Administrasi," *Jurnal Ilmu Administrasi (JIA)*, vol. 15, no. 1, pp. 103–104, 2018, doi: https://doi.org/10.31113/jia.v15i1.140.

[2] A. N. F. Muqoffa, Mawar, and N. Serikandi, "Manfaat Sosial Media Dalam E-Government di Indonesia," *Jurnal Ilmu Sosial, Pendidikan, dan Humaniora*, vol. 1, no. 3, pp. 34–35, 2022, doi: https://doi.org/10.56910.

[3] F. S. Isbandi, A. Sagiyanto, A. Rahmah, W. Apriani, and A. S. Utomo, "Implementasi Fitur Laksa Pada Aplikasi Tangerang Live Sebagai Layanan Aspirasi Masyarakat Tangerang," *Jurnal Komunikasi*, vol. 16, no. 1, pp. 89–90, 2022, doi: 10.21107/ilkom.v16i1.13118.

[4] M. Rizki, "Penggunaan Aplikasi Laksa Dalam Mewujudkan Transparansi dan Partisipasi Masyarakat di Kota Tangerang [skripsi]," Syarif Hidayatullah Jakarta Islamic State University, Jakarta, 2022.

[5] A. D. Aulia, "Aplikasi Laksa (Layanan Aspirasi Kotak Saran Anda) Dalam Meningkatkan Pelayanan Publik di Kota Tangerang [skripsi]," Institut Pemerintahan Dalam Negeri (IPDN), Tangerang, 2022.

[6] Warkim, M. H. Muslim, F. Harvianto, and S. Utama, "Penerapan Metode Scrum dalam Pengembangan Sistem Informasi Layanan Kawasan," *Jurnal Teknik Informatika dan Sistem Informasi*, vol. 6, no. 2, pp. 366–367, Aug. 2020, doi: 10.28932/jutisi.v6i2.2711.

[7] R. W. P. Pamungkas, A. N. Azizah, and B. S. Zebua, "Analisis Penerapan Metode Scrum untuk Meningkatkan Efektivitas dalam Pembuatan Aplikasi melalui Literature Review," *Jurnal Pendidikan Informatika dan Sains*, vol. 11, no. 2, pp. 156–164, Dec. 2022, doi: 10.31571/saintek.v11i2.4650.

[8] R. Kurniadi, C. Riki, and M. Nurkamilah, "Rancang Bangun Aplikasi Perpustakaan berbasis Web dengan Menggunakan Framework CodeIgniter," *Formosa Journal of Science and Technology*, vol. 1, no. 5, pp. 508–509, Sep. 2022, doi: 10.55927/fjst.v1i5.1209.

[9] M. Syarif, "Implementasi Algoritma String Matching Dalam Pencarian Surah Dan Ayat Dalam Al-Quran Berbasis Web," *Indonesian Journal on Networking and Security*, vol. 6, no. 2, pp. 70–72, 2017, [Online]. Available: https://ijns.org/journal/index.php/ijns/article/view/375

[10] N. Chida and T. Terauchi, "Repairing Regex-Dependent String Functions," in *Proceedings - 2024 39th ACM/IEEE International Conference on Automated Software Engineering, ASE 2024*, Association for Computing Machinery, Inc, Oct. 2024, pp. 294–305. doi: 10.1145/3691620.3695005.

[11] J. M. Bintang, M. F. Ashshidiq, and H. F. Dzakwan, "Penerapan Algoritma String Matching dan Regular Expression pada Aplikasi Kamus Besar Bahasa Indonesia (KBBI)," *BIOS : Jurnal Teknologi Informasi dan Rekayasa Komputer*, vol. 4, no. 1, pp. 34–41, Mar. 2023, doi: 10.37148/bios.v4i1.57.

[12] I. Onyenwe, S. Ogbonna, E. Onyedimma, O. Ikechukwu-Onyenwe, and C. Nwafor, "Developing Smart Web-Search Using RegEx," *https://arxiv.org/abs/2110.04767*, Oct. 2021, doi: https://doi.org/10.48550/arXiv.2110.04767.

[13] M. Hilmyansyah, Malabay, H. Simorangkir, and Yulhendri, "Implementasi Metode Scrum Pada Pembangunan Sistem Informasi Monitoring Progress Proyek Berbasis Web (Studi Kasus: PT Quatra Engineering Mandiri)," *Jurnal Komputer dan Informatika*, vol. 6, no. 3, pp. 31–32, 2022, [Online]. Available: https://journals.upi-yai.ac.id/index.php/ikraith-informatika/issue/archive

[14] F. A. El Hakim, A. Prayudi, K. Hanifati, A. Fariza, and H. Rante, "Scrum Framework Implementation for Building an Application of Monitoring and Booking E-Bus Based on QRCode," *Jurnal Teknik Informatika*, vol. 16, no. 1, pp. 102–104, May 2023, doi: 10.15408/jti.v16i1.29409.

[15] A. Jailani and M. Yaqin A, "Pengujian Aplikasi Sistem Informasi Akademik menggunakan Metode Blackbox dengan Teknik Boundary Value Analysis," *JACIS : Journal Automation Computer Information System*, vol. 4, no. 2, pp. 60–66, 2024, doi: 10.47134/jacis.v4i2.78.

[16] S. Nabila, A. R. Putri, A. Hafizhah, F. H. Rahmah, and R. Muslikhah, "Pemodelan Diagram UML Pada Perancangan Sistem Aplikasi Konsultasi Hewan Peliharaan Berbasis Android (Studi Kasus: Alopet)," *Jurnal Ilmu Komputer dan Bisnis*, vol. 12, no. 2, pp. 130–139, Nov. 2021, doi: 10.47927/jikb.v12i2.150.

[17] S. W. Ramdany, S. Aulia Kaidar, B. Aguchino, C. Amelia, A. Putri, and R. Anggie, "Penerapan UML Class Diagram dalam Perancangan Sistem Informasi Perpustakaan Berbasis Web," *Journal of Industrial and Engineering System*, vol. 5, no. 1, pp. 30–41, Jun. 2024.

[18] A. Fahrezi, F. N. Salam, G. M. Ibrahim, R. R. Syaiful, and A. Saifudin, "Pengujian Black Box Testing pada Aplikasi Inventori Barang Berbasis Web di PT. AINO Indonesia," *Jurnal Ilmu Komputer dan Pendidikan*, vol. 1, no. 1, pp. 1–2, 2022, [Online]. Available: https://journal.mediapublikasi.id/index.php/logic