Pengembangan Website Refactoring pada Aplikasi Berbasis Web UII Ops Monitoring Dashboard untuk Site Reliability Engineering di Badan Sistem Informasi UII

Arka'an Nurhuda
Jurusan Informatika
Universitas Islam Indonesia
Yogyakarta
17523007@students.uii.ac.id

Andhik Budi Cahyono
Jurusan Informatika
Universitas Islam Indonesia
Yogyakarta
andhikbudi@uii.ac.id

Pengembangan aplikasi berbasis web tidak terbebaskan dari kendala-kendala seperti *deadline* yang ketat, kekurangan pengalaman *dan mismanagement*. Dengan mempertimbangkan hal tersebut maka diperlukannya praktik khusus yang berfokus dalam meningkatkan kualitas dan *reliability* dari kode aplikasi berupa *refactoring*. Aplikasi UII Ops Monitoring Dashboard merupakan salah satu teknologi yang dimanfaatkan tim *Site Reliability Engineering* di Badan Sistem Informasi UII untuk membantu kinerja mereka dalam beroperasi. Bahasa pemrograman PHP yang digunakan dalam pengembangan aplikasi UII Ops merupakan salah satu bahasa pemrograman yang cukup populer, namun untuk menambah peforma, kesederhanaan, wawasan dan pemahaman terkait *trend* teknologi yang berkembang, diimplementasikanlah bahasa Go untuk mengganti bahasa PHP pada proyek *refactoring* UII Ops ini. Proyek *refactoring* yang dilakukan pada aplikasi telah meningkatkan *readability* dari struktur kode aplikasi secara keseluruhan dengan sentralisasi keseluruhan request di satu lokasi yang mudah dikenali. Performa dari aplikasi mengalam sedikit penurunan di beberapa sektor, seperti *loading* dan *scripting*. Penurunan tersebut tidak terlalu signifikan karena perbedaan waktu yang berupa ms(milisecond), namun perlu diperhatikan untuk pengembangan selanjutnya. Proyek *refactoring* aplikasi UII Ops memiliki kekurangan dan kelebihan, namun dengan perpindahan bahasa pemrograman dan peningkatan koherensi kode, *maintainability* dan *development* dari aplikasi UII Ops bisa dirasakan peningkatannya dalam jangka panjang.

Keywords—refactoring, UII Ops, Site Reliability Engineering

I. PENDAHULUAN

Tim *Site Reliability Engineering* memiliki berbagai macam program kerja yang berkaitan dengan proses-proses yang perlu dilakukan untuk menghasilkan dan menjaga sistem agar dapat diandalkan (*reliable*). Konsep *Site Reliability Engineering* datang dari kebutuhan untuk menerapkan sistem dan operasi yang handal dalam jangka panjang terutama di dalam dunia teknologi informasi. Kinerja dari *Site Reliability Engineering* tidak sepenuhnya terpisahkan dari kinerja entitas lain, yaitu DevOps. DevOps itu sendiri merupakan serangkaian praktik, *guideline*, dan kultur yang dirancang untuk membongkar *silo* (salah satu bagian dari sistem yang terisolasi dan terpisahkan dari bagian lain dalam asitektur[1]) di dalam pengembangan, operasi, jaringan, dan keamanan teknologi informasi[2].

Reliability merupakan salah satu atribut yang paling penting dalam teknologi informasi. Dalam kinerjanya, *Site Reliability Engineering* menerapkan prinsip-prinsip terkait *software engineering* pada infrastruktur sistem dalam rangka meningkatkan keamanan dan keandalan dari sistem perangkat lunak. Komponen-komponen yang diperhatikan dalam *Site Reliability Engineering* meliputi ketersediaan, performa, *latency*, efisiensi, kapasitas, dan *incident response*[3]. Perangkat lunak yang digunakan oleh *Site Reliability Engineering* tersedia dalam berbagai bentuk dan salah satu aplikasi yang paling populer dan mudah diakses adalah aplikasi berbasis web.

Tim SRE dari Badan Sistem Informasi UII telah mengembangkan aplikasi berbasis web UII Ops Monitoring Dashboard yang dapat digunakan untuk membantu kinerja mereka seperti *incident tracking*, *monitoring*, manajemen anggota tim, dan tugas-tugas lainnya. Dengan adanya satu aplikasi yang memuat banyak fitur tersebut di dalam satu lokasi, koordinasi kerja tim dapat berjalan dengan lebih terarah dan tidak terpisahkan antara satu sama lain. Kinerja dan pengembangan berbagai tugas dapat dicek antar sesama tim sesuai dengan *privilege* mereka masing-masing.

Dalam rangka untuk meningkatkan kualitas dan fungsionalitas dari aplikasi UII Ops, dilakukan aktivitas *refactoring* terhadap aplikasi tersebut untuk mengubah struktur kode internal dari aplikasi tersebut tanpa merubah perilaku eksternal. Refactoring itu sendiri dapat menghapus parameter atau variable yang tidak bermanfaat atau kurang optimal, kode yang terduplikasi bisa disederhanakan dan tanggung jawab antar entitas dapat didistribusikan dengan lebih baik[4].

Proyek *refactoring* ini juga dilakukan dengan mengubah bahasa pemrograman yang sebelumnya berupa bahasa PHP menjadi bahasa Go. Bahasa Go memiliki performa yang lebih unggul dibandingkan dengan bahasa PHP yang lebih sering digunakan[5]. Peningkatan performa ini akan berdampak positif dalam jangka panjang untuk kinerja dari tim *Site Reliability Engineering* di Badan Sistem Informasi UII. Dengan diterapkannya refactoring, diharapkan aplikasi UII Ops dapat

ditingkatkan dari segi pemahaman, kemudahan, dan keamanan.

II. LANDASAN TEORI

A. Site Reliability Engineering

Perusahaan dalam bidang IT menghasilkan dan mengimplementasikan berbagai macam aplikasi dalam beroperasi sehingga diperlukannya tim yang memiliki disiplin khusus yang dapat menjamin dan meningkatkan keandalan, pemahaman, dan keamanan dari berbagai macam aplikasi yang dimanfaatkan. *Site Reliability Engineering* akan mengatasi posisi tersebut dengan menerapkan *software engineering* untuk menerapkan *automation* pada berbagai macam tugas terkait teknologi informasi. Tugas-tugas tersebut meliputi manajemen sistem, *incident responds, incident tracking, emergency response,* dan *monitoring*[6].

B. Backend Development

Proses pengembangan aplikasi berbasis web dapat dibagi menjadi dua, yaitu *frontend development* dan *backend development. Backend development* itu sendiri merupakan jenis pengembangan aplikasi yang menyangkut pembuatan API, integrasi *database*, penggunaan *libraries*, implementasi komponen dalam sistem dan aktivitas lain yang tidak menyangkut antarmuka pengguna. Beberapa *skill* yang diperlukan dalam praktik backend development meliputi bahasa pemrograman web, *database* dan *cache*, API dan *server*[7].

C. Refactoring

Produk aplikasi menyangkut peran dari beberapa anggota yang memiliki keahlian yang berbeda-beda dan aplikasi sudah wajar untuk dilakukan pemberbaharuan dari waktu ke waktu. Teknik disiplin untuk mengubah struktur internal dari kode yang sudah ada tanpa mengubah perilaku eksternal biasa disebut dengan refactoring[8]. Tujuan dari aktivitas *refactoring* meliputi membersihkan kode, menyederhanakan struktur aplikasi, meningkatkan pemahaman dan meningkatkan efisiensi. Praktik pengembangan aplikasi dapat terkendala oleh beberapa faktor, seperti *deadline* yang ketat, kekurangan pengalaman, *mismanagement* dan berbagai kendala lainnya[9], sehingga dengan adanya teknik disiplin *refactoring*, peningkatan kualitas dan pembarbaharuan aplikasi dapat terus dijaga dalam jangka panjang.

D. Bahasa Go

Salah satu bahasa pemrograman yang menarik perhatian dari banyak perusahaan besar adalah bahasa Go. Bahasa Go merupakan bahasa pemrograman yang bersifat *open-source* dan dirancang untuk menjadi sederhana, dapat diandalkan, dan efisien dalam mengembangkan perangkat lunak. Bahasa pemrograman ini ditemukan oleh Rob Pike, Ken Thomson, dan Robert Griesemer pada tahun 2009. Salah satu motivasi mereka untuk mengembangkan bahasa pemrograman yang baru adalah karena pengalaman mereka dengan bahasa C++ yang dapat menghasilkan waktu *loading* yang sangat lama dan membatasi operasi pengembangan aplikasi. Bahasa Go dikembangkan dengan mengadopsi berbagai macam keunggulan dari bahasa C++, terutama pada bidang performa dan fitur-fitur keamanan, dan dikombinasikan dengan kecepatan dari bahasa python[10].

III. METODOLOGI

Metodologi yang digunakan dalam penulisan makalah ini meliputi perancangan, implementasi, dan pengujian. Tiap tahapan metodologi tersebut diperlukan untuk mengembangkan dan menyelesaikan proyek. Penjelasan secara detail dari tiap tahapan adalah sebagai berikut.

A. Perancangan

Pada tahap pertama ini, peneliti melakukan riset intensif terkait teknologi yang terkait dengan proyek *refactoring*. Teknologi tersebut menyangkut bahasa pemrograman dan *tools* yang perlu digunakan dan pembelajaran terkait teknologi tersebut tidak terpaku pada satu periode namun bisa berkembang dari waktu ke waktu seiring dengan perkembangan proyek. *Tools* yang digunakan di awal proyek bisa jadi menimbulkan kendala di masa berlangsungnya proyek, maka di kondisi tersebut perlu dipertimbangkan apakah memungkinkan untuk menemukan solusi dengan *tools* yang sama atau diperlukannya riset lanjut untuk mencari *tools* yang lebih cocok dan sesuai.

Eksplorasi terkait kode dan fungsionalitas dari aplikasi UII Ops itu sendiri tentunya juga harus dilakukan. Tiap *libraries, method*, dan fungsi dari setiap halaman dan bagaimana tiap komponen tersebut berinteraksi satu sama lain perlu dipelajari dengan seksama. Dikarenakan aktivitas *refactoring* berada di bidang *backend development*, maka berkas-berkas dan aset-aset terkait *frontend development* tidak perlu dirubah struktur internalnya. Pekerjaan *refactoring* dapat dibagi menjadi enam kategori penting yang perlu dipahami, yaitu [4]

1. *Composing Methods:* penyusunan *method* perlu dilakukan untuk memperbaiki *method* yang terlalu panjang dan bertele-tele. Penulisan *method* yang susah dipahami akan menghambat proses pengembangan aplikasi dalam jangka panjang.
2. *Simplifying Conditional Expressions: conditionals* dapat berkembang menjadi semakin kompleks dan panjang dalam pengembangan aplikasi, maka diperlukannya aksi penyederhanaan dari *conditionals* tersebut.
3. *Moving Features between Objects:* pitur yang ada dalam satu kelas bisa didistrbusikan ke kelas lain, namun bila fitur terlalu banyak digunakan di kelas lain dibandingkan di kelas asalnya, hal ini dapat menurunkan koherensi dan meningkatkan *dependency* antar kelas. Maka perlu dilakukannya perpindahan fitur ke kelas yang paling konsisten menggunakan fitur tersebut.
4. *Organizing Data:* pengorganisasian data meningkatkan *data handling* dan meluruskan asosiasi antar kelas, sehingga tiap kelas dapat menjadi lebih *portable* dan *reusable.*
5. *Dealing with Generalization:* mteode abstraksi perlu dilakukan salah satunya untuk menoptimalkan *class assistance hierarchy* dan membuat kelas dan *interface* yang lebih ideal.
6. *Making Method Calls Simpler:* menyederhanakan pemanggilan method akan membuatnya menjadi

lebih mudah dipahami dan dianalisis interaksi antar kelasnya.[9]

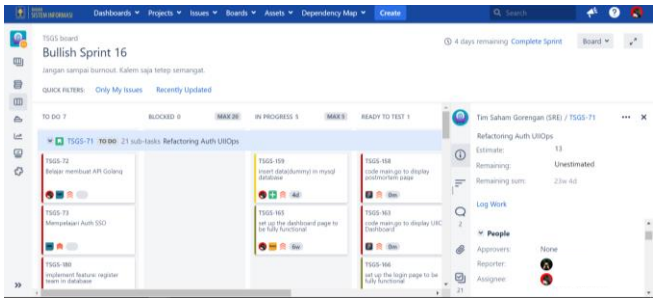
B. *Implementasi*

Tahap selanjutnya adalah implementasi dari teknologi yang sudah dipelajari dan direncanakan untuk mengembangkan proyek. Tingkat kerumitan dari implementasi ini beragam dari fungsi menampilkan halaman utama untuk seluruh pengguna hingga menampilkan status web monitoring secara *real-time* untuk *administrator*. Proses implementasi ini tentunya lebih ideal bila dikerjakan mulai dari tahap yang paling sederhana hingga tahap yang plaing rumit. Struktur internal dari sistem juga lebih bisa dipahami seiring dengan proses implementasi berlangsung, sehingga adaptasi terkait pendekatan dalam plaksanaan proyek bisa disesuaikan sesuai keperluan.

Aplikasi UII Ops memiliki halaman yang bisa diakses secara umum dan halaman yang hanya bisa diakses oleh administrator. Halaman utama ini merupakan halaman pertama yang diluncurkan oleh kode aplikasi, sehingga pengembangan aplikasi dengan bahasa Go bisa dimulai dari halaman tersebut. Bila pengembangan berjalan lancar, maka bisa dilanjutkan pada halaman yang lebih kompleks seperti halaman login, manajemen user&team, halaman dashboard, dan seterusnya. Pendekatan dalam pengembangan aplikasi perlu disesuaikan dengan bahasa pemrograman yang baru. Selain meningkatkan kualitas kode sebelumnya dengan *refactoring*, perlu juga dipahami cara kerja bahasa Go agar tidak justru menimbulkan masalah baru yang belum ada sebelumnya.

C. *Pengujian*

Tahap terakhir merupakan tahap pengujian terhadap hasil pengembangan proyek *refactoring*. Pengujian ini dilakukan secara bertahap agar hasil dari pengujian bisa dijadikan pertimbangan untuk proses pelaksanaan selanjutnya.



Gambar 1 Tampilan Task pada JIRA Software

Tim *Site Reliability Engineering* memanfaatkan JIRA sebagai *incident management tool* yang bisa digunakan untuk manajemen proyek, pelacakan *bug*, pelacakan masalah, dan *workflow*. Pada gambar di atas merupakan salah satu contoh pekerjaan yang dilacak oleh JIRA dalam jangka waktu tertentu. Seperti yang sudah dibahas sebelumnya, *assignee* dan *reporter* adalah dua orang yang berbeda, *assignee* merupakan pekerja yang bertanggung jawab untuk mengembangkan proyek sedangkan *reporter* adalah pekerja yang bertanggungjawab untuk melaksanakan testing dan laporan terhadap hasil proyek.

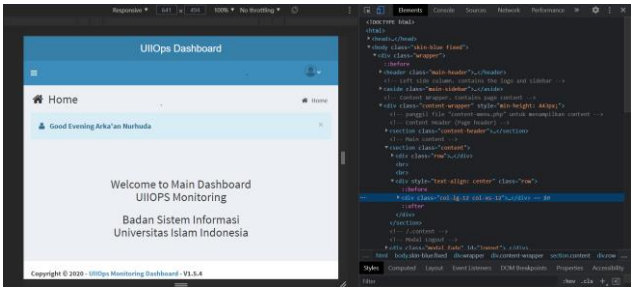
Peneliti memanfaatkan XAMPP Control Panel untuk membantu distribusi modul apache dan mysql dalam pengembangan aplikasi. Aplikasi UII Ops yang dikembangkan dengan bahasa Go dapat dikonfigurasikan ke port tersendiri. Aplikasi kemudian bisa diakses melalui browser dengan mengakses url localhost:7000. Halaman utama akan muncul pertama kali dan pengguna dapat mulai mengakses fitur-fitur lain dari aplikasi UII Ops. Tampilan dari *deployment* aplikasi adalah seperti berikut.

```
233
234     // Konfigurasi port 7000
235     fmt.Printf("Starting server at port 7000\n")
236     //log.Fatal(http.ListenAndServe(":7000", nil))
237     err := http.ListenAndServe(":7000", nil) // setup listening port
238     if err != nil {
239         log.Fatal("ListenAndServe: ", err)
240     }
241     appengine.Main()
242 }
243
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

LENOVO@LAPTOP-03AUQJ9M MINGW64 /c/xampp/htdocs/opsgo (master)
$ go run main.go
Starting server at port 7000
```

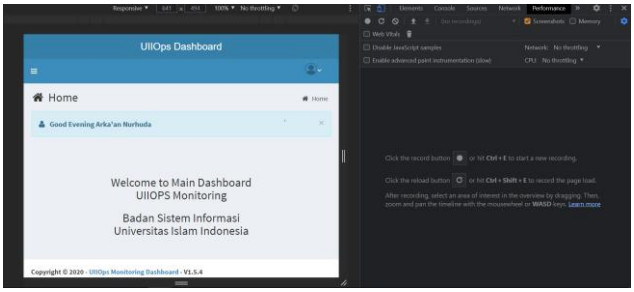
Gambar 2 Tampilan Deployment Aplikasi

Website refactoring dilakukan untuk meningkatkan kualitas dan maintainability dari *source code*, namun pengubahan struktur kode perlu dilakukan tanpa mengubah runtime dari aplikasi secara signifikan. Sehingga untuk menguji apakah refactoring telah dilaksanan dengan sukses, bisa dilakukan testing aplikasi web melalui browser. Pengujian bisa dilakukan dengan fitur inspect element yang tersedia di kebanyakan browser-browser populer. Berikut merupakan tampilan dari fitur tersebut.



Gambar 3 Tampilan Inspect Element pada Browser

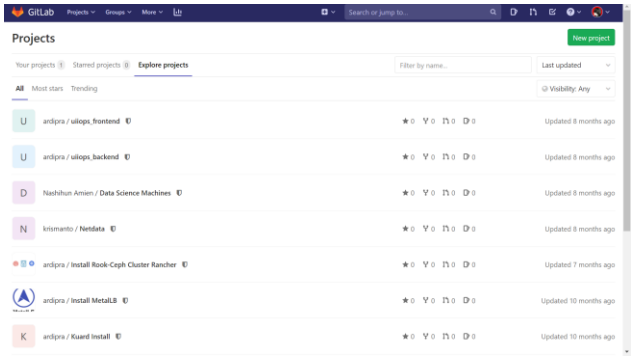
Dari halaman tersebut kita bisa beralih ke bagian performance dan di tab inilah kita bisa menguji performa dari suatu website. Di bagian tab performance kita bisa langsung melakukan pengujian melalui tombol *reload* atau dengan perintah Ctrl+Shift+E, seperti yang terlihat pada gambar berikut.



Gambar 4 Tampilan Tab Performance

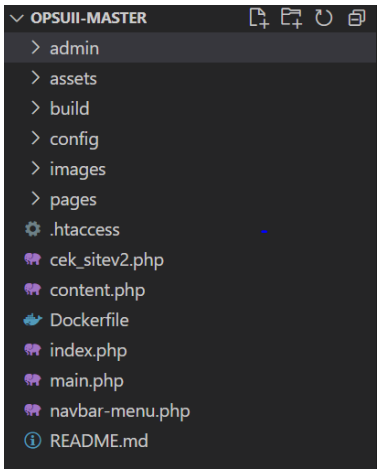
IV. HASIL DAN PEMBAHASAN

Untuk melakukan *refactoring*, diperlukannya analisis dan pemahaman terkait source code dari aplikasi UII Ops itu sendiri. Hasil pekerjaan dari Tim *Site Reliability Engineering* disimpan dalam *repository* GitLab yang bisa diakses oleh setiap anggota tim. Berikut merupakan tampilan dari *repository* hasil pekerjaan Tim SRE.



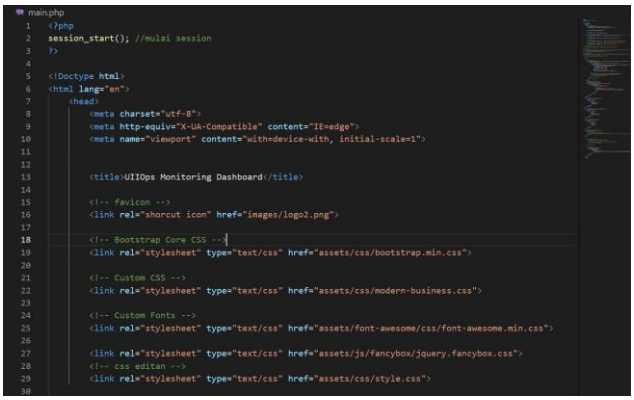
Gambar 5 Tampilan GitLab Tim SRE

Setiap anggota tim SRE bisa memeriksa hasil pekerjaan satu sama lain sehingga koordinasi dan *management* tim bisa dilakukan dengan lebih ideal dan terstruktur. Dari *repository* tersebut peneliti dapat mengakses *source code* dari aplikasi UII Ops. Berikut merupakan susunan *file* dan *folder* dari *source code* aplikasi UII Ops sebelum dilakukan *refactoring*.



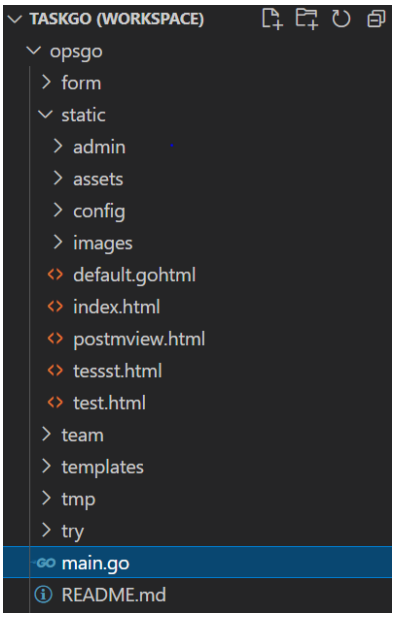
Gambar 6 Tampilan Struktur Kode Sebelum Refactoring

Susunan *file* dan *folder* tersebut sudah rapi dan tertata dengan baik, namun di dalam tiap *folder* tersebut mengandung banyak jumlah file yang bisa membingungkan anggota tim lain. File main.php mengandung kode html yang bertanggung jawab untuk tampilan halaman utama, namun struktur kode secara keseluruhan belum bisa ditelaah dan dianalis dari file tersebut, berikut tampilannya secara sekilas.



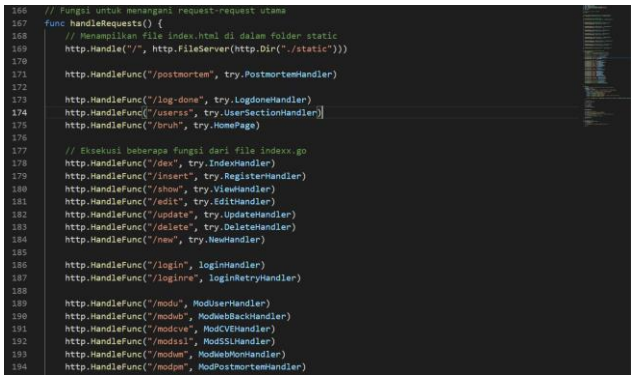
Gambar 7 Tampilan Kode main.php

Kode tersebut mengandung kode tampilan dari halaman utama. Hal ini masuk akal karena kode tersebut merupakan kode pertama yang dieksekusi, namun dalam melakukan refactoring yang berfokus pada sisi *backend development*, penggabungan kode *backend* dan *frontend* ini menjadi hambatan, sehingga hal pertama yang dilakukan peneliti adalah memisahkan berkas-berkas *frontend* dan berkas-berkas *backend* agar pekerjaan bisa dilakukan dengan lebih lancar dan fokus. Setelah dilakukan pekerjaan *refactoring*, berikut merupakan tampilan *file* dan *folder* dari *source code* aplikasi UII Ops.



Gambar 8 Tampilan Struktur Kode Setelah Refactoring

Bila dibandingkan dengan tampilan *file* dan *folder* sebelum dilakukan *refactoring*, susunan tersebut kurang terstruktur dan masih bisa dirapikan lagi. Hal ini dikarenakan peneliti mengembangkan aplikasi tersebut secara *native* dan belum menggunakan *framework* yang bisa merapikan susunan *file* dan *folder* tersebut. Perubahan struktur kode yang mendasar ada pada *file* utama main.go, berikut tampilannya.



Gambar 9 Tampilan Kode main.go

Di dalam berkas main.go, terdapat fungsi yang mengandung request dari keseluruhan aplikasi UII Ops. Hal ini membuat jumlah baris kode meningkat secara signifikan dari sebelumnya, namun ada manfaat lain yang sebelumnya tidak ada sebelum dilakukan *refactoring.* Berkat fungsi yang tersentralisasi tersebut, pengembang aplikasi dapat mengenali secara lebih seksama struktur kode dari keseluruhan aplikasi beserta berkas yang berkaitan dengan setiap request melalui satu berkas, yaitu main.go.

Hal ini meningkatkan *readability* dari struktur kode sehingga pengembang aplikasi bisa meneliti berapa banyak dan fitur apa yang ada dari aplikasi serta dimana kode perlu ditulis bila ada fitur yang perlu dikembangkan di masa mendatang. Adapun fitur-fitur dari aplikasi berbasis web UII Ops Monitoring Dashboard yang digunakan oleh Time *Site Reliability Engineering* adalah sebagai berikut.

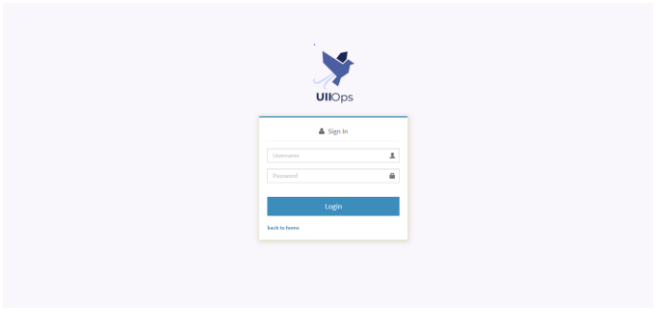
A. Halaman Utama

Halaman utama dari UII Ops Monitoring Dashboard merupakan halaman utama yang bisa diakses oleh seluruh pengguna ketika mengakses *domain url* UII Ops. Halaman ini memuat informasi dasar seperti *announcement* dan hasil *postmortem* yang ter-*update*. Terdapat juga fitur login yang akan memungkinkan penggunanya untuk masuk sebagai admin agar bisa mengakses *dashboard* dari UII Ops.

Gambar 10 Tampilan Halaman Utama

B. Halaman Login

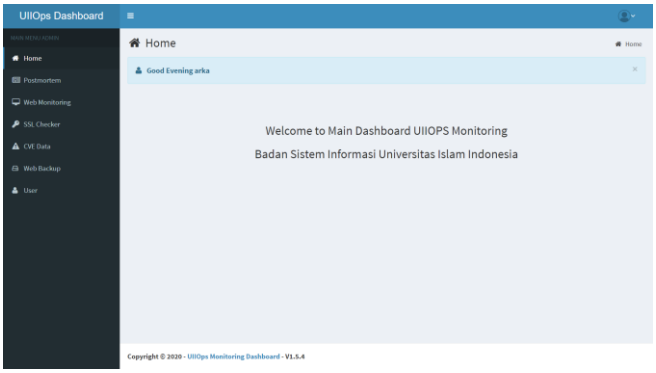
Halaman login ini memungkinkan pengguna untuk masuk sebagai admin selama *username* dan *password* sesuai dengan yang terdaftar di *database*. Pendaftaran anggota baru bisa dilakukan oleh admin yang telah terdaftar bila diperlukan.



Gambar 11 Tampilan Halaman login

C. Halaman Dashboard

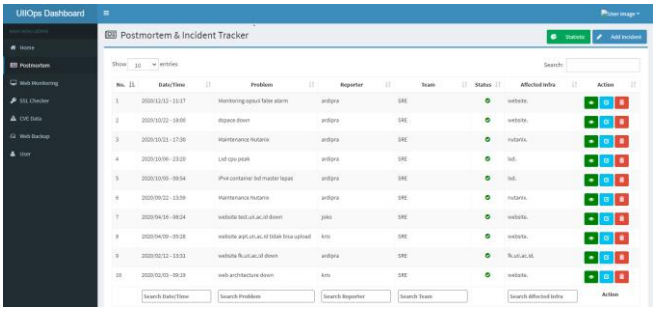
Halaman dashboard memuat akses terhadap fungsionalitas lengkap dari UII Ops. Pengguna dari UII Ops terbagi menjadi tiga kategori, yaitu *admin*, *user*, dan *viewer*. Ketiga kategori pengguna tersebut memiliki hak yang berbeda-beda dalam mengakses fitur-fitur dari UII Ops, *admin* memiliki akses terhadap keseluruhan fitur UII Ops, *user* memiliki akses terhadap keseluruhan fitur UII Ops kecuali panel manajemen *user*, dan *viewer* memiliki akses yang terbatas terhadap fitur UII Ops.



Gambar 12 Tampilan Halaman Dashboard

D. Halaman Postmortem

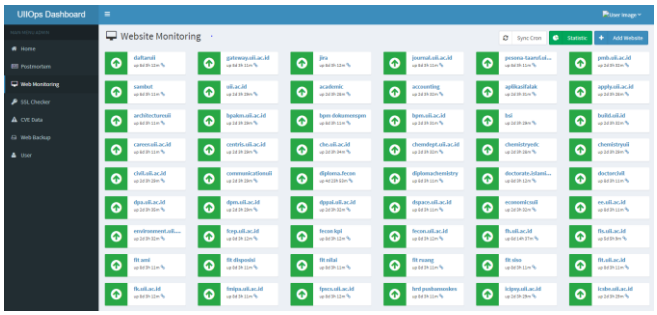
Halaman *Postmortem* bermanfaat untuk melaporkan berbagai macam *incident report* yang berkaitan dengan penggunaan web. Daftar laporan tersebut memungkinkan tim SRE untuk mendapat berbagai macam informasi penting terkait *incident* yang ada sehingga perbaikan bisa segera dilakukan. Selain memungkinkan untuk mengumpulkan berbagai macam *incident* tersebut untuk kemudian ditangani, *postmortem* dapat juga menjadi sarana pembelajaran bagi para pegawai untuk mengetahui lebih lanjut terkait berbagai macam *incident* yang telah terjadi sebelumnya.



Gambar 13 Tampilan Halaman Postmortem

E. *Halaman Website Monitoring*

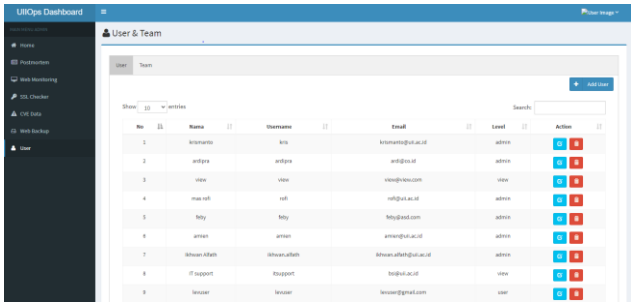
Halaman *website monitoring* memungkinkan untuk melakukan *monitoring* berbagai website yang berada di UII. Total dari website yang telah termonitoring melalui UII Ops berjumlah sekitar 140 *website*. Status dari *website* dan *alert* akan dapat diketahui melalui *dashboard* dan Slack. Laporan *monitoring website* tertentu dapat juga diekspor berupa *file* pdf bila diperlukan.



Gambar 14 Tampilan Halaman Website Monitoring

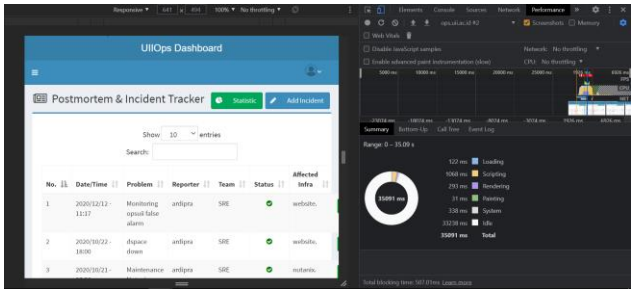
F. *Halaman User&Team*

Halaman User&Team menampilkan daftar pengguna dashboard UII Ops dan team yang terdaftar di Badan Sistem Informasi UII. Halaman ini menyediakan fitur *add*, *update*, *view* dan *delete*. Pengguna yang telah terdaftar di *database* akan tampil di halaman ini dan pengguna tersebut dapat *login* dengan *username* dan *password* mereka masing-masing untuk mengakses halaman dashboard UII Ops, sesuai dengan *privilege* yang telah terdaftar.



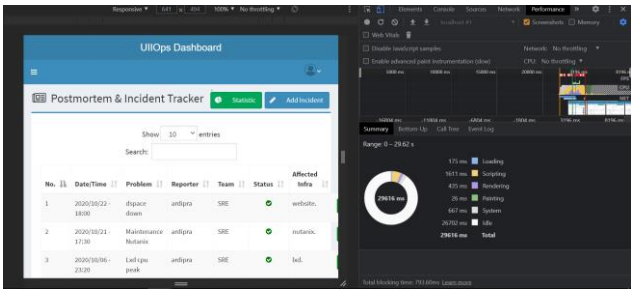
Gambar 15 Tampilan Halaman User&Team

Hasil refactoring berhasil dijalankan pada browser dan berfungsi dengan baik, namun perlu dilakukan pengujian terhadap performa website untuk menilai tingkat kesuksesan dari refactoring tersebut. Hasil performa dari website UII Ops sebelum dijalankan refactoring adalah sebagai berikut.



Gambar 16 Tampilan Hasil Performa Sebelum Refactoring

Hasil performa dari website UII Ops setelah dijalankan refactoring adalah sebagai berikut.



Gambar 17 Tampilan Hasil Performa Setelah Refactoring

Ada beberapa data bermanfaat yang disajikan melalu pengujian tersebut. Waktu loading dari website sebelum dilakukan refactoring adalah 122 ms, sedangkan waktu loading dari website setelah dilakukan *refactoring* adalah 175 ms. Perbedaan waktu tersebut menunjukkan waktu loading dari website refactoring telah meningkat, namun perbedaan waktu ini tidak terlalu signifikan karena masih dalam satuan ms(milisecond). Pengguna umum tidak akan langsung menyadari perbedaan load time, namun adanya perbedaan waktu loading time ini perlu diperhatikan dalam pengembangan selanjutnya dan masih bisa ditingkatkan lagi. Data-data tersebut bisa disajikan dengan tabel berikut.

	Waktu Sebelum Refactoring (ms)	Waktu Setelah Refactoring (ms)
Loading	122	175
Scripting	1068	1611
Rendering	293	435
Painting	31	26
System	338	667
Idle	33238	26702
Total	35091	29616

V. KESIMPULAN DAN SARAN

Proyek *refactoring* pada aplikasi UII Ops tidak akan langsung nampak hasilnya dari sisi pengguna, namun dapat disimpulkan dari pelaksanaan proyek ini bahwa struktur kode internal nampak lebih sederhana dan mudah dibaca, sehingga memiliki potensial untuk dilakukan pengembangan dan pemberbaharuan terhadap aplikasi di masa mendatang. Fungsionalitas aplikasi berfungsi dengan baik dan dapat memenuhi tujuan asal dari pengembangan aplikasi, yaitu untuk membantu kinerja tim SRE yang meliputi *monitoring*, *incident tracking*, manajemen anggota tim dan tugas-tugas lainnya Struktur kode refactoring telah meningkatkan readability dengan adanya fungsi yang menyimpan seluruh request dari aplikasi secara keseluruhan di tempat yang mudah dikenali, sehingga *development* dan *maintainability* terhadap aplikasi UII Ops dapat ditingkatkan dalam jangka panjang.

Berdasarkan dari hasil pengujian, ada beberapa data yang bisa dikaji dan dianalisis. Peningkatan pada waktu *loading*, *scripting*, dan *rendering* pada website setelah dilakukan *refactoring* menunjukkan bahwa pekerjaan *refactoring* masih bisa ditingkatkan lagi. Namun, peningkatan waktu tidak terlalu signifikan dari sisi pengguna, sehingga fungsionalitas dari *website* UII Ops

masih bisa diakses dan dimanfaatkan tanpa ada kendala yang signifikan. Fungsionalitas lengkap dari website UII Ops terbataskan aksesnya pada anggota tim *Site Reliability Engineering*, sehingga kendali terhadap pengembangan website bisa dijaga dengan lebih baik, mengingat pengguna website yang cukup sedikit. Setelah menelaah proyek secara keseluruhan, bisa disimpulkan bahwa proyek *refactoring* ini memiliki kelebihan dan kekurangan. Kekurangan dari proyek ini merupakan kekurangan dari peneliti sendiri yang masih pemula dalam menggunakan bahasa Go, namun kelebihannya meliputi peningkatan koherensi dan *readability* yang dapat meningkatkan *maintainability* dan *development* dari aplikasi UII Ops dalam jangka panjang.

Pelaksanaan proyek ini tentunya tidak terbebaskan dari berbagai macam hambatan maka saran yang diharapkan peneliti untuk meningkatkan pelaksanaan proyek selanjutnya adalah dengan menggunakan *framework* agar struktur kode bisa tersusun dengan baik yang akan memberikan manfaat terhadap kelancaran pengembangan aplikasi dalam jangka panjang. Diperlukan juga arahan yang lebih konkrit dalam aktivitas *refactoring* agar pelaksana proyek dapat lebih fokus dan mempelajari ilmu yang benar-benar sesuai dengan yang diperlukan. Selain itu, wawasan terkait bahasa Go belum sepenuhnya populer di kalangan pengembang, sehingga diperlukannya pembekalan lebih lanjut agar pengembangan aplikasi yang memanfaatkan bahasa Go tidak statis dan bisa terus dikembangkan lagi di masa mendatang.

REFERENSI

[1] "What is a Silo? - Definition from Techopedia." https://www.techopedia.com/definition/25939/silo (accessed Jun. 05, 2021).

[2] B. Beyer, N. Richard Murphy, D. K.Rensin, K. Kawahara, and S. Thorne, *The Site Reliability Workbook: Practical Ways to Implement SRE*. O'Reilly Media, Inc., 2018.

[3] D. Kececioglu, *Reliability Engineering Handbook*, no. v. 1. DEStech Publications, 2002.

[4] K. Omar Elish, "CLASSIFICATION OF REFACTORING METHODS BASED ON SOFTWARE QUALITY ATTRIBUTES," p. 1390 ,117-99 ص ;8 شماره.

[5] A. Agung *et al.*, "Rancang Bangun Sistem Informasi Instalasi Gawat," pp. 15–16, 2017.

[6] "What is Site Reliability Engineering (SRE) | IBM." https://www.ibm.com/cloud/learn/site-reliability-engineering (accessed Jun. 05, 2021).

[7] "What is Backend Developer? Skills Need for Web Development." https://www.guru99.com/what-is-backend-developer.html (accessed Jun. 05, 2021).

[8] M. Fowler, "Refactoring: Refactoring: Improving the Design Improving the Design of Existing Code of Existing Code What we will cover What we will cover q A simple example of refactoring â Blow by blow example of changes â Steps for illustrated refactorings," pp. 1–82, 1997, [Online]. Available: http://http//ourworldourworld..compuservecompuserve.com/homepages/Martin_Fowler.com/homepages/Martin_Fowler.

[9] "Refactoring: clean your code." https://refactoring.guru/refactoring (accessed Jun. 05, 2021).

[10] "Go language | Google Developers." https://developers.google.com/learn/topics/go (accessed Jun. 05, 2021).