Analisis Frontend Dalam Migrasi Multi Page Application Menjadi Single Page Application Dengan Metode Iterative Incremental Pada Aplikasi Sofi Modul Penjadwalan

1st Ataya Najla
Fakultas Rekayasa Industri
Universitas Telkom
Bandung, Indonesia
atayanj@student.telkomuniversity.ac.id

2nd Ekky Novriza Alam
Fakultas Rekayasa Industri
Universitas Telkom
Bandung, Indonesia
ekkynovrizalam@telkomuniversity.ac.id

3rd Tien Febrianti Kusumasari
Fakultas Rekayasa Industri
Universitas Telkom
Bandung, Indonesia
tienkusumasari@telkomuniversity.ac.id

***Abstrak* — Perkembangan teknologi di era sekarang mempengaruhi berbagai bidang salah satunya adalah pada bidang Pendidikan, yang mulai beradaptasi dengan aplikasi-aplikasi berbasis teknologi. Universitas Telkom menjadi salah satu, yang menggunakan kesempatan itu untuk melakukan perkembangan, khususnya pada Fakultas Rekayasa Industri salah satunya Aplikasi SOFI yang merupakan aplikasi manajemen sidang TA (Tugas Akhir). Namun aplikasi SOFI sekarang memiliki beberapa masalah dari sisi *frontend.* Oleh karna itu dilakukan migrasi ke *single page application* agar lebih *response time* dan meningkatkan intuitive. Penulis mengembangkan ini dengan metode *Iterative Incremental.* Metode tersebut mengembangkan aplikasi secara berulang dengan mengevaluasi dan melakukan perbaikan yang diperlukan pada setiap tahap, mengurangi risiko dan meningkatkan kualitas aplikasi. Hasil dari penelitian ini aplikasi SOFI menjadi responsif dan efisien saat digunakan oleh banyak pengguna secara bersamaan dan meningkatkan intuitivitas aplikasi serta memberikan pengalaman pengguna yang lebih baik.**

Kata kunci— Frontend, Multi Page Application, Single Page Application, Iterative Incremental

I. PENDAHULUAN

Perkembangan teknologi di era sekarang mempengaruhi berbagai bidang salah satunya adalah pada bidang pendidikan, yang mulai beradaptasi dengan aplikasi-aplikasi berbasis teknologi. Perkembangan teknologi juga terjadi saat pandemi, banyak institusi pendidikan yang mulai menggunakan aplikasi-aplikasi berbasisi teknologi. Covid-19 memberi dampak pada sekolah, perguruan tinggi, dan universitas di seluruh dunia untuk menunda kelas tatap muka dan menerapkan transformasi digital disemua sistem pendidikan [1].

Universitas Telkom menjadi salah satu yang menggunakan kesempatan itu untuk melakukan perkembangan, khususnya pada Fakultas Rekayasa Industri yang memiliki beberapa aplikasi berbasis digital. Satu dari beberapa aplikasi yang digunakan untuk membantu produktivitas Fakultas Rekayasa Industri yaitu aplikasi SOFI yang merupakan aplikasi manajemen sidang TA (Tugas Akhir). Aplikasi tersebut digunakan untuk registrasi sidang, penjadwalan sidang, penilaian sidang, revisi sidang, dan laporan sidang.

Di sisi *frontend*, aplikasi SOFI juga masih menggunakan *Multi-Page Application. Multi Page Application* (MPA) merupakan aplikasi web terdiri dari banyak halaman yang berbeda dan halaman web sepenuhnya dimuat ulang dari server setiap kali pengguna berinteraksi dengan aplikasi dan memuat konten baru. Meskipun MPA memiliki keunggulan dalam navigasi dan struktur yang lebih tradisional, ada beberapa kelemahan yang dapat mengurangi pengalaman pengguna. Salah satu kelemahan yaitu mengalami *error* saat memuat data pengguna. Ketika aplikasi perlu memproses data dalam jumlah besar, ada kemungkinan besar terjadi eror. Hal ini dapat mengganggu fungsionalitas aplikasi dan mengurangi pengalaman pengguna. MPA dapat mengalami respon yang lemah dan keterlambatan karena arsitektur tradisional yang merugikan dalam pengumpulan data baru setiap kali dibutuhkan [2]. Hal ini dapat menjadi masalah, terutama jika aplikasi perlu mengelola data pengguna dalam jumlah besar secara berkala. Selain itu, adanya potensi hilangnya data yang telah diinput oleh pengguna sebelum error terjadi. Pemuatan ulang yang berulang juga dapat mengakibatkan kehilangan data, jika pengguna secara tidak sengaja meninggalkan halaman atau mengalami kesalahan selama proses pemuatan ulang, informasi yang sudah di isi mungkin tidak akan tersimpan [3]. Ini menyebabkan pengguna harus menginput ulang dari awal, yang sangat menghambat dan menyulitkan pengguna. Dalam pengguna aplikasi SOFI dengan rentang pengguna yang dari mahasiswa dan dosen junior hingga dosen senior, terdapat peluang besar untuk meningkatkan intuitive aplikasi dan meningkatkan kepuasan pengguna. Selain itu, memperbaiki *error handling*

dan meningkatkan *response time* dapat sangat membantu dalam menyelesaikan masalah yang muncul, sehingga menghasilkan pengalaman pengguna yang lebih responsif dan efisien.

Untuk menghadapi masalah tersebut, aplikasi SOFI dapat beralih ke *Single Page Application. Single Page Application* (SPA) merupakan aplikasi web yang terdiri dari komponen-komponen individu yang dapat di perbarui secara dinamis tanpa perlu memuat ulang seluruh halaman setiap permintaan dari pengguna. Dengan menggunakan SPA, data yang diinput oleh pengguna tidak akan hilang sebelum eror terjadi. Hal ini dapat meningkatkan intuitive pengguna terhadap aplikasi agar lebih mudah digunakan, serta menyelesaikan permasalahan interaksi aplikasi dengan pengguna. Selain itu, saat mengalami *error* ketika memuat data pengguna karena terlalu banyak, SPA dapat mengurangi waktu respon. Dengan SPA memuat semua sumber daya sekaligus dan memperbarui konten secara dinamis, SPA memberikan pengalaman yang lebih mulus dan interaktif bagi pengguna [3]. Selain itu, SPA dapat meningkatkan responsivitas aplikasi. Masalah yang dialami pada aplikasi SOFI dapat diatasi dengan menggunakan SPA sebagai solusi untuk mengatasi permasalahan yang ada.

Dari penjelasan di atas, penelitian ini melakukan migrasi aplikasi SOFI dari MPA ke SPA dengan metode *iterative incremental*. Metode ini merupakan gabungan antara metode *iterative* dan metode *incremental*. Keduanya berfokus pada pengembangan bertahap dan berulang untuk meningkatkan kualitas dan menyesuaikan sistem sesuai dengan kebutuhan pengguna. Metode *iterative* memungkinkan revisi dan penyempurnaan berulang berdasarkan umpan balik dan pembelajaran dari iterasi dengan strategi penjadwalan pengerjaan ulang dimana waktu disisihkan untuk merevisi dan memperbaiki bagian sistem [4]. Metode ini mengembangkan aplikasi secara berulang dengan mengevaluasi dan melakukan perbaikan yang diperlukan pada setiap tahap, mengurangi risiko dan meningkatkan kualitas aplikasi.

II. KAJIAN TEORI

A. Front-end

Frontend merupakan teknologi pengembangan web yang berfokus pada desain antarmuka pengguna dan pengalaman pengguna. Dalam frontend terdapat dua jenis aplikasi yaitu *Multi Page Application* (MPA) dan *Single Page Application* (SPA).

1. Multi Page Application

Multi Page Application atau di singkat MPA merupakan situs web yang memiliki banyak halaman, masing-masing halamannya dimuat langsung dari server. *Multi Page Application* merupakan model aplikasi web, dimana setiap interaksi atau permintaan dari pengguna akan memuat halaman baru dari server [5]. MPA cocok untuk pengembangan aplikasi yang kecil dan aktivitas sederhana. Karna membuat halaman yang kompleks di server dan mentransfernya ke klien membutuhkan waktu yang signifikan dan menurunkan pengalaman pengguna [6].

2. Single Page Application

Single Page Application atau di singkat SPA merupakan aplikasi web yang memuat satu halaman HTML dan di perbarui secara otomatis tanpa perlu memuat ulang seluruh halaman setiap permintaan atau interakasi dari pengguna. SPA terdiri dari komponen individu yang dapat diganti/diperbarui secara mandiri, tanpa menyegarkan/memuat ulang seluruh halaman sehingga seluruh halaman tidak perlu dimuat ulang pada setiap tindakan pengguna [7]. SPA memiliki keunggulan yang berbeda dalam menyegarkan dan memperbarui halaman web dan kontennya hanya sekali. Saat pemuatan halaman awal, SPA memuat semua sumber daya yang dibutuhkan seperti CSS, gambar, skrip, dan komponen lainnya, dengan SPA dapat mengoptimalkan waktu pemuatan berikutnya dengan menggunakan sumber daya yang telah dimuat sebelumnya [3].

B. Incremental/Iterative Model

Metode *iterative* melibatkan pengembangan sistem melalui serangkaian siklus berulang atau iterasi. Metode *iterative* memungkinkan revisi dan penyempurnaan berulang berdasarkan umpan balik dan pembelajaran dari iterasi dengan strategi penjadwalan pengerjaan ulang dimana waktu disisihkan untuk merevisi dan memperbaiki bagian sistem [4]. Metode *incremental* fokus pada pengembangan sistem dalam bentuk komponen atau bagian kecil yang dapat dikelola. Setiap increment (penambahan) menambah fungsi atau fitur baru ke sistem yang ada. Kedua metode memiliki karakteristik dan manfaat untuk mengatasi kompleksitas dan ketidakpastian dalam proyek pengembangan. Keduanya berfokus pada pengembangan bertahap dan berulang untuk meningkatkan kualitas dan menyesuaikan sistem sesuai dengan kebutuhan pengguna [8]. Metode ini mengembangkan sistem melalui iterasi yang masing-masing menghasilkan increment baru. Setiap increment melalui siklus *iterative* untuk memastikan kualitas dan penyesuaian berdasarkan umpan balik.

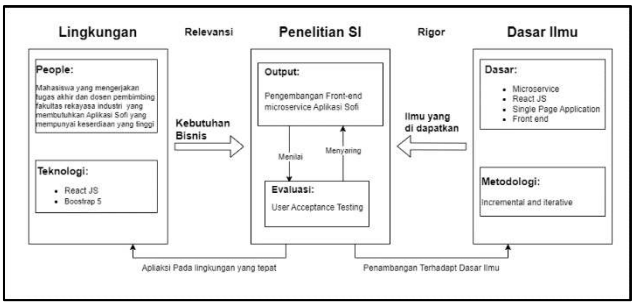
1. User Acceptance Testing

User Acceptance Testing (UAT) merupakan salah satu testing yang digunakan pada tahapan testing dalam alur pengembangan, yang terjadi di akhir proses saat sebelum sistem digunakan. Tahapan tersebut melibatkan pengguna akhir yang melakukan pengujian sistem di lingkungan operasional untuk memvalidasi apakah sistem tersebut memenuhi kebutuhan dan ekspektasi pengguna [9]. UAT memiliki beberapa kelebihan, termasuk kemampuannya untuk mengungkap logika bisnis atau fungsionalitas yang belum terdeteksi pada tahap pengujian unit dan sistem, mengukur kesesuaian sistem dengan kebutuhan pengguna, dan menentukan sejauh mana sistem telah selesai dikembangkan. UAT umumnya melibatkan dua mekanisme pengujian alpha testing salah satunya menggunakan metode *blackbox* yang berfokus pada pengujian fungsionalitas sistem [10]. Dengan menerapkan UAT, efektivitas pengujian dapat ditingkatkan, memastikan bahwa produk akhir sesuai dengan kebutuhan pengguna dan dapat diselesaikan tepat waktu.

III. METODE

A. Kerangka Berpikir

Kerangka berpikir dapat membantu untuk mengetahui atau mendeskripsikan tentang permasalahan kemudian mendeskripsikan tentang objek-objek yang terkait dengan penelitian. Dengan menggambarkan kerangka berpikir dapat mengetahui awal mula tentang penelitian yang dilakukan.

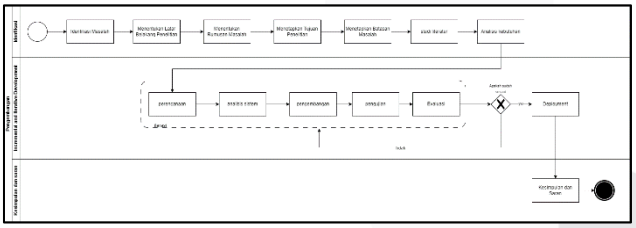


GAMBAR 1
Kerangak Berpikir

Pada Gambar 1 memaparkan model konseptual yang digunakan dalam penelitian ini. Pada model konseptual terdiri dari 3 bagian yaitu lingkungan, penelitian SI dan dasar ilmu. Bagian pertama yaitu lingkungan terdapat people yang dimana pengguna yang menggunakan aplikasi SOFI yaitu mahasiswa akhir dan dosen pembimbing fakultas rekayasa industri. Selanjutnya teknologi yang digunakan untuk mengembangkan aplikasi SOFI menggunakan React Js dan Boostrap 5. Bagian kedua yaitu penelitian SI yang dimana hasil akhirnya *frontend* aplikasi SOFI yang sudah menjadi SPA dengan evaluasinya menggunakan *User Acceptance Testing* dengan metode *Blackbox Testing*. Bagian ketiga terdapat dasar ilmu yang berdasarkan beberapa landasan teori yaitu *Microservice*, React js, *frontend*, dan *Single page application*. Serta menggunaka metodologi *Iterative Incremental* untuk menggembangan aplikasi SOFI.

B. Sistematika Penyelesaian Masalah

Penelitian ini dilakukan untuk pengembangan aplikasi SOFI. Pada sistematika penyelesaian masalah merupakan tahapan yang dipilih untuk perancanaan dalam penelitian ini agar dapat solusi untuk menyelesaikan masalah tersebut.



GAMBAR 2
Sistematika Penyelesaian Masalah

Pada Gambar 2 memiliki 2 tahapan pengembangan yaitu *incremental and iterative* dan kesimpulan dan saran.

a. Tahap Incremental and Iterative Development

Pada tahap ini terdapat perencanaan, Analisis Sistem, Pengembangan, Pengujian, Evaluasi dan *Deployment*.

1. Perencanaan

Aktifitas pertama dalam tahap ini yaitu perencanaan untuk menentukan apa saja yang dilakukan selama tahap pengembangan. Tahap ini memerlukan analisis dari informasi yang didapatkan dari kebutuhan pengguna aplikasi tersebut, bisa berupa fungsi yang dibutuhkan dan fitur pada aplikasi tersebut.

2. Analisis Sistem

Lalu analisi sistem untuk perancangan sistem sesuai dengan kebutuhan pengguna yang telah di tetapkan sebelumnya dan tahap ini menghasilan UML seperti *activity diagram* dan *use case*. Pada *use case* menggambarkan hubungan interaksi anatara aplikasi SOFI dan pengguna. Sedangkan *activity diagram* menggambarkan berbagai aktivitas yang terjadi pada aplikasi SOFI.

3. Pengembangan

Tahap pengembangan merupakan implementasi ke dalam bahasa pemograman berdasarkan rancangan desain diagram yang sudah dikembangkan sebelumnya.

4. Pengujian

Selanjutnya testing yaitu menguji sistem untuk mengetahui apakah ada kesalahan setelah melakukan proses implementasi. Untuk melakukan pengujian menggunakan UAT dengan metode *Blackbox testing*. UAT di gunakan untuk melakukan pengujian sistem di lingkungan operasional yang melibatkan pengguna akhir untuk memvalidasi apakah sistem tersebut memenuhi kebutuhan dan ekspektasi pengguna.

5. Evaluasi

Selanjutnya melakukan evaluasi untuk memastikan apakah aplikasi yang sudah di kembangkan sesuai atau masih ada yang kurang. Melakukan evaluasi juga untuk mengetahui fitur mana yang harus diperbaiki dan ditingkatkan. Jika masih terdapat kesalahan atau masalah dalam pengembanganya maka ada iterasi atau tahapan yang di ulang hingga aplikasi tersebut tercapat sesuai kebutuhn pengguna.

6. Deployment

Tahap terakhir dari tahap ini yaitu *deployment* yang Dimana tahap ini mempublikasikan aplikasi yang sudah dikembangkan atau sudah ditingkatkan.

b. Tahap Kesimpulan dan saran

Tahap ini merupakan feedback dari pengguna yang sudah menggunakan aplikasi SOFI agar mendapatkan saran dan kesimpulan dari aplikasi yang sudah dikembangkan. Hal tersebut juga bisa menjadi referensi untuk kedepannya.

IV. HASIL DAN PEMBAHASAN

Penelitian ini melakukan pengembangan dengan migrasi dari MPA ke SPA pada aplikasi SOFI. Pengembangan ini berfokus pada modul penjadwalan.

A. Tahap Perancangan

Pada tahap perancangan ini, langkah pertama yang dilakukan adalah menyusun sistem berdasarkan hasil identifikasi kebutuhan yang telah dilakukan sebelumnya. Setelah tahap ini dilakukan, hasil dari proses ini dirangkum dalam tabel yang menggambarkan rincian fitur yang dikembangkan.

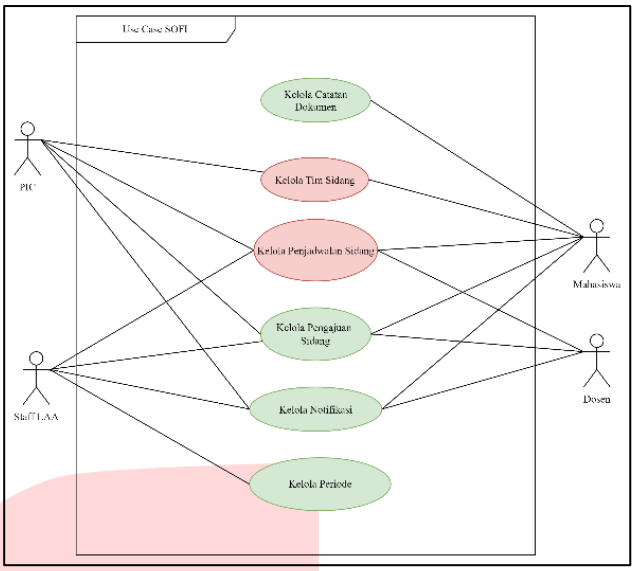
TABEL 1
Kebutuhan Fungsionalitas Iterasi Fase Pertama

ID	Fitur	*Method*	Deskripsi
REQ-02.01	Kelola Penjadwalan Sidang	Mendapatkan Seluruh Jadwal Sidang	Proses mendapatkan seluruh data penjadwalan sidang berdasarkan akses pengguna

ID	Fitur	*Method*	Deskripsi
REQ-02.02		Mendapatkan Detail Jadwal Sidang	Proses mendapatkan detail dari penjadwalan sidang

TABEL 2
Kebutuhan Fungsionalitas Iterasi Fase Kedua

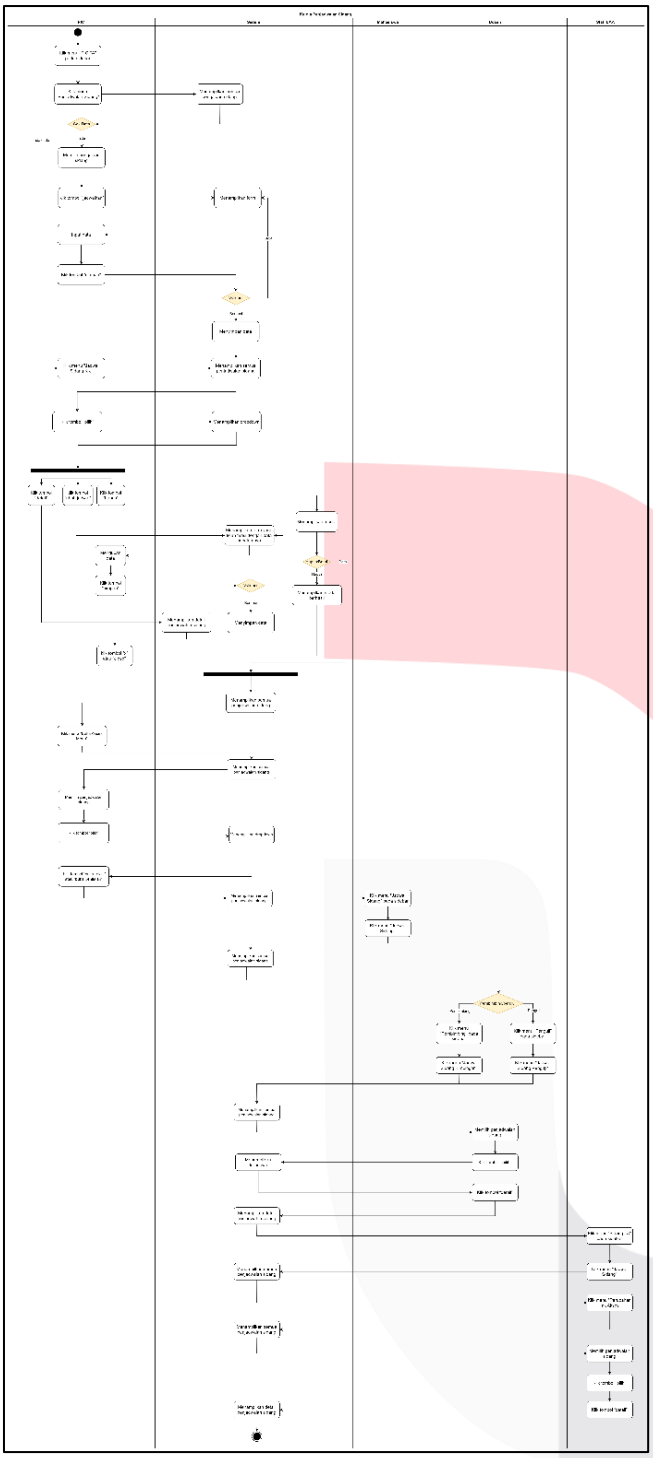
ID	Fitur	*Method*	Deskripsi
REQ-01.01	Kelola Tim Sidang	Mendapatkan Detail Tim Sidang	Proses dimana mendapatkan detail tim sidang
REQ-02.03	Kelola Penjadwalan Sidang	Membuat Jadwal Sidang	Proses membuat penjadwalan sidang
REQ-02.04		Mengubah Jadwal Sidang	Proses melakukan perubahan data pada penjadwalan sidang
REQ-02.05		Menandai Jadwal Sidang	Proses yang dimana untuk menandai beberapa kasus pada penjadwalan sidang
REQ-02.06		Menghapus Jadwal Sidang	Proses menghapus penjadwalan sidang



GAMBAR 3
Use Case SOFO

B. Tahap Analisis sistem

Pada tahap kedua, analisis dan desain sistem sesuai dengan persyaratan yang telah ditentukan pada tahap pertama dilakukan. Hasil dari tahap perancangan ini mencakup berbagai diagram dan model yang memberikan gambaran menyeluruh tentang bagaimana sistem bejalan dan berfungsi. Tahap ini mengahasilkan *usecase diagram* dan *activity diagram*.



GAMBAR 4
Activity Diagram Kelola Penjadwalan

C. Tahap Pengembangan

Pada tahap ketiga dalam proses pengembangan aplikasi, sistem dikembangkan berdasarkan perencanaan telah disiapkan pada langkah pertama. Pada tahap ini, aplikasi dibuat menggunakan React.js, dan tim pengembang bekerja sama dengan melalui *repository* GitHub. Hasil dari proses pengembangan aplikasi yang telah di migrasi ke SPA berdasarkan pada kebutuhan pengguna yang telah diidentifikasi tahap sebelumnya.

GAMBAR 5
Tampilan Halaman Jadwal Sidang PIC

GAMBAR 6
Tampilan Halaman Create Penjadwalan

D. Tahap Pengujian

Pada tahap keempat dalam proses pengembangan aplikasi adalah tahap pengujian. Pada tahap ini, aplikasi diuji untuk memastikan bahwa migrasi ke SPA berhasil dan sesuai dengan spesifikasi yang telah ditetapkan. Tahap pengujian di fase ini menggunakan UAT dengan metode *Blackbox testing*.

TABEL 3
Hasil UAT Fase Pertama

ID	Expected result	Actual result	Status
Mendapatkan Seluruh Jadwal Sidang			
REQ-02.01	Menampilkan halaman jadwal sidang KK dan seluruh penjadwalan yang di input PIC (PIC)	Menampilkan halaman jadwal sidang KK dan seluruh penjadwalan yang di input PIC (PIC)	Pass
	Menampilkan halaman jadwal sidang penguji dan seluruh penjadwalan yang jadi penguji (Dosen Penguji)	Menampilkan halaman jadwal sidang penguji dan seluruh penjadwalan yang jadi penguji (Dosen Penguji)	Pass
	Menampilkan halaman jadwal sidang pembimbing dan seluruh penjadwalan yang di bimbing (Dosen Pembimbing)	Menampilkan halaman jadwal sidang pembimbing dan seluruh penjadwalan yang di bimbing (Dosen Pembimbing)	Pass
	Menampilkan halaman jadwal sidang dan seluruh	Menampilkan halaman jadwal sidang dan seluruh	Pass

ID	Expected result	Actual result	Status
	penjadwalan (Staff LAA)	penjadwalan (Staff LAA)	
Mendapatkan Detail Jadwal Sidang			
REQ-02.02	Menampilkan halaman jadwal mahasiswa dan yang berisi detail penjadwalan sidang (Mahasiswa)	Menampilkan halaman jadwal mahasiswa dan yang berisi detail penjadwalan sidang (Mahasiswa)	pass
	Menampilkan halaman detail penjadwalan sidang mahasiswa yang di bimbing (Dosen Pembimbing)	Menampilkan halaman detail penjadwalan sidang mahasiswa yang di bimbing (Dosen Pembimbing)	Pass
	Menampilkan halaman detail jadwal sidang yang telah di tentukan (Dosen Penguji)	Menampilkan halaman detail jadwal sidang yang telah di tentukan (Dosen Penguji)	Pass
	Menampilkan modal detail jadwal sidang yang sebelum nya telah di buat (PIC)	Menampilkan modal detail jadwal sidang yang sebelum nya telah di buat (PIC)	Pass
	Menampilkan halaman detail jadwal sidang (Staff LAA)	Menampilkan halaman detail jadwal sidang (Staff LAA)	Pass

TABEL 4
Hasil UAT Fase Kedua

ID	Expected result	Actual result	Status
Mendapatkan Detail Tim Sidang			
REQ-01.01	Menampilkan halaman create penjadwalan dan berhasil menampilkan detail tim sidang yang di jadwalkan	Menampilkan halaman create penjadwalan dan berhasil menampilkan detail tim sidang yang di jadwalkan	Pass
Membuat Jadwal Sidang			
REQ-02.03	Berhasil menginput data penjadwalan dan menampilkan data penajdwalan sidang yang sebelumnya dibuat	Berhasil menginput data penjadwalan dan menampilkan data penajdwalan sidang yang sebelumnya dibuat	Pass
Mengubah Jadwal Sidang			
REQ-02.04	Menampilkan halaman update penjadwalan, berhasil mengubah data penjadwalan yang sudah dibuat dan menampilkan data penjadwalan sidang yang sudah diubah.	Menampilkan halaman update penjadwalan, berhasil mengubah data penjadwalan yang sudah dibuat dan menampilkan data penjadwalan sidang yang sudah diubah.	Pass
Menandai Jadwal Sidang			
REQ-02.05	Menampilkan Halaman Perubahan Hak Akses dan berhasil menambah menu akses revisi/penilaian pada salah satu penjadwalan sidang	Menampilkan Halaman Perubahan Hak Akses dan berhasil menambah menu akses revisi/penilaian pada salah satu penjadwalan sidang	Pass
Menghapus Jadwal Sidang			
REQ-02.06	Menampilkan modal delete dan berhasil menghapus salah satu penjadwalan sidang	Menampilkan modal delete dan berhasil menghapus salah satu penjadwalan sidang	Pass

E. Tahap Evaluasi

Setelah pengujian selesai, tahap kelima dalam proses pengembangan aplikasi adalah evaluasi. Tahap ini mengevaluasi hasil pengujian dan implementasi aplikasi untuk memastikan bahwa sistem yang dibangun memenuhi kebutuhan dan harapan pengguna serta sesuai dengan spesifikasi yang telah ditentukan. Berdasarkan tahap pengujian berhasil 100% pass secara keseluruhan.

F. Tahap Deployment

Setelah semua iterasi dalam metode *Iterative Incremental* selesai, selanjutnya tahap deployment yang merupakan tahap terakhir dari metode Iterative Incremental. Proses ini mencakup *hosting* aplikasi yang telah dikembangkan melalui layanan yang memungkinkan *deployment global*, sehingga pengguna dapat mengaksesnya. Aplikasi dihosting menggunakan platfom vercel. Vercel akan memproses dan meng-host aplikasi sehingga dapat diakses.

V. KESIMPULAN

Aplikasi SOFI, yang awalnya berbasis MPA, menjadi lebih lambat dan tidak responsif saat memproses banyak data pengguna. Dengan migrasi ke SPA, waktu respon dapat dikurangi, sehingga meningkatkan kinerja aplikasi, memastikan responsivitas yang lebih baik, dan meningkatkan efisiensi aplikasi. Selain itu SPA, memuat seluruh konten halaman web sekali saja dan kemudian mengelola pembaruan konten secara dinamis, proses pemuatan data dapat

dioptimalkan. Dengan ini membuat aplikasi SOFI tetap responsif dan efisien, yang dapat meningkatkan pengalaman pengguna.

Aplikasi SOFI dalam MPA menghadapi masalah *handling error*, yang menyebabkan data input pengguna hilang sebelum eror terjadi, yang membuat pengalaman pengguna buruk dan menimbulkan tantangan dalam memberikan kepuasan pengguna. Dengan menggunakan SPA, *handdling eror* dapat ditingkatkan karena penanganan kesalahan secara lokal tanpa perlu memuat ulang halaman secara keseluruhan. Hal ini membuat data input pengguna dapat dipertahankan meskipun terjadi kesalahan dan memberikan pengalaman pengguna yang lebih baik dan memuaskan.

REFERENSI

[1] D. Banerjee, D. Das, S. Pal, S. R. Paul, A. Debnath, and M. Reza, "Effect of covid-19 on digital transformations in teaching learning methodology and its consequences in society: A review," in *Journal of Physics: Conference Series*, IOP Publishing Ltd, Mar. 2021. doi: 10.1088/1742-6596/1797/1/012066.

[2] S. L. Jeng, W. H. Chieng, and Y. Chen, "Web-Based Human-Machine Interfaces of Industrial Controllers in Single-Page Applications," *Mobile Information Systems*, vol. 2021, 2021, doi: 10.1155/2021/6668843.

[3] A. Ghasemzadeh, H. Mahmoudi, and A. H. Mohseni, "Enhancing QoE (Quality of Experience) in Web Applications: The Role of SPA (Single Page Application) for Improved QoE," 2023.

[4] O. J. Okesola, A. A. Adebiyi, A. A. Owoade, O. Adeaga, O. Adeyemi, and I. Odun-Ayo, "Software Requirement in Iterative SDLC Model," in *Advances in Intelligent Systems and Computing*, Springer, 2020, pp. 26–34. doi: 10.1007/978-3-030-51965-0_2.

[5] A. Mesbah and A. Van Deursen, "Migrating multi-page web applications to single-page AJAX interfaces," in *Proceedings of the European Conference on Software Maintenance and Reengineering, CSMR*, 2007, pp. 181–190. doi: 10.1109/CSMR.2007.33.

[6] M. Kaluža and K. Troskot, "Vukelić: Comparison of Front-End Frameworks for Web Applications," 2018. [Online]. Available: https://www.json.org/

[7] Madhuri A Jadhav, B. R. Sawant, and A. Deshmukh, "Single Page Application using AngularJS," 2015. [Online]. Available: http://mydomain.com/myseo#key=value

[8] S. Saeed, N. Z. Jhanjhi, M. Naqvi, and M. Humayun, "Analysis of software development methodologies," *International Journal of Computing and Digital Systems*, vol. 8, no. 5, pp. 445–460, 2019, doi: 10.12785/ijcds/080502.

[9] I. Otaduy and O. Diaz, "User acceptance testing for Agile-developed web-based applications: Empowering customers through wikis and mind maps," *Journal of Systems and Software*, vol. 133, pp. 212–229, Nov. 2017, doi: 10.1016/j.jss.2017.01.002.

[10] I. Afrianto, A. Heryandi, A. Finadhita, and S. Atin, "User Acceptance Test For Digital Signature Application In Academic Domain To Support The Covid-19 Work From Home Program," *International Journal of Information System & Technology Akreditasi*, vol. 5, no. 3, pp. 270–280, 2021, [Online]. Available: https://tt-el.my.id/.