

**PENGEMBANGAN FRONTEND SISTEM INFORMASI
MANAJEMEN REKRUTMEN DAN KOMUNITAS
MENGUNAKAN FRAMEWORK Nuxt**



Disusun Oleh:

N a m a : Muhammad Alwan Abdurra'uf
NIM : 20523076

**PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
2024**

HALAMAN PENGESAHAN DOSEN PEMBIMBING

**PENGEMBANGAN FRONTEND SISTEM INFORMASI
MANAJEMEN REKRUTMEN DAN KOMUNITAS
MENGUNAKAN FRAMEWORK Nuxt**

TUGAS AKHIR JALUR MAGANG



Pembimbing,

(Erika Ramadhani, S.T., M.Eng.)

HALAMAN PENGESAHAN DOSEN PENGUJI**PENGEMBANGAN FRONTEND SISTEM INFORMASI
MANAJEMEN REKRUTMEN DAN KOMUNITAS
MENGUNAKAN FRAMEWORK NUXT****TUGAS AKHIR JALUR MAGANG**

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika – Program Sarjana di Fakultas Teknologi Industri, Universitas Islam Indonesia

Yogyakarta, 22 Juli 2024

Tim Penguji

Erika Ramadhani, S.T., M.Eng.

Anggota 1

Ari Sujarwo, S.Kom., M.I.T.

Anggota 2

Feri Wijayanto, S.T., M.T.

Mengetahui,

Ketua Program Studi Informatika – Program Sarjana

Fakultas Teknologi Industri

Universitas Islam Indonesia



(Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini :

Nama : Muhammad Alwan Abdurra'uf
NIM : 20523076

Tugas akhir dengan judul :

PENGEMBANGAN FRONTEND SISTEM INFORMASI MANAJEMEN REKRUTMEN DAN KOMUNITAS MENGUNAKAN FRAMEWORK NUXT

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila di kemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung risiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 04 Juli 2024


(Muhammad Alwan Abdurra'uf)

HALAMAN PERSEMBAHAN

Puji dan syukur penulis panjatkan kepada Allah SWT yang telah memberi kemudahan, rahmat, dan petunjuk-Nya yang telah membimbing penulis melewati setiap tantangan selama perjalanan perkuliahan hingga mencapai tahap akhir, yakni menyelesaikan tugas akhir sebagai bagian dari perjalanan untuk mendapatkan gelar sarjana. Shalawat dan salam penulis sampaikan kepada Nabi Muhammad SAW yang menjadi teladan bagi kita semua dan juga membimbing menuju jalan lurus dan benar yang diridhoi Allah SWT.

Tugas akhir ini, yang disajikan di hadapan pembaca, penulis persembahkan kepada semua pihak yang telah mendukung saya selama ini khususnya kedua orang tua penulis yang selalu memberikan dukungan, doa, dan motivasi selama menjalani perkuliahan. Semoga segala upaya dan doa yang telah kalian curahkan sepanjang perjalanan ini menjadi ladang amal jariyah yang tak terputus.

HALAMAN MOTO

“Maka Sesungguhnya bersama kesulitan ada kemudahan”

Q.S. Al-Insyirah ayat 5

“One day or day one?, You decide” - Unknown

KATA PENGANTAR

Segala puji dan syukur penulis panjatkan kehadirat Allah SWT yang telah melimpahkan rahmat, taufik dan hidayah-Nya sehingga penulis dapat menyelesaikan tugas akhir ini dengan lancar walaupun jauh dari kata sempurna. Laporan ini dibuat untuk memenuhi salah satu persyaratan kelulusan jalur magang di Program Studi Teknik Informatika, Fakultas Teknologi Industri, Universitas Islam Indonesia.

Penyusunan tugas akhir ini tidak terlepas dari bantuan berbagai pihak, baik secara langsung maupun tidak. Oleh karena itu, penulis mengucapkan terima kasih kepada semua pihak yang telah membantu, diantaranya :

1. Kedua orang tua yang selalu memberikan doa serta dukungan selama proses magang berlangsung hingga penyelesaian tugas akhir.
2. Bapak Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D. selaku Ketua Program Studi Informatika – program sarjana.
3. Ibu Erika Ramadhani, S.T., M.Eng. selaku Dosen Pembimbing yang telah memberikan saran, nasehat dan bimbingan dalam penyusunan laporan ini.
4. Mas Gigih Prasetya selaku mentor, juga seluruh staf dan karyawan Geekgarden Software House yang telah memberikan kepercayaan, arahan, dan pengalaman selama magang berlangsung.
5. Seluruh dosen dan staff Program Studi Informatika yang memberikan ilmu pengetahuan selama perkuliahan.

Laporan ini tidak luput dari kesalahan, baik dalam proses pembuatan maupun dari hasil laporan. Untuk itu, penulis menerima kritik dan saran guna menyempurnakan laporan ini. Akhir kata, penulis berharap semoga tugas akhir ini bermanfaat bagi penulis serta pembaca.

Yogyakarta, 04 Juli 2024



(Muhammad Alwan Abdurra'uf)

SARI

PT. Komuri Indonesia GeekGarden adalah sebuah perusahaan pengembang perangkat lunak (*software house*) yang berfokus pada pemberian layanan pengembangan aplikasi berskala kecil hingga menengah di Yogyakarta. Seiring dengan perkembangan perusahaan, GeekGarden mulai mengembangkan beberapa aplikasi internal, salah satunya adalah JoinGeek. JoinGeek merupakan aplikasi yang digunakan untuk manajemen rekrutmen karyawan dan pemagang, serta manajemen komunitas yang dibentuk oleh GeekGarden. Pengembangan JoinGeek menerapkan metodologi scrum yang melibatkan proses pengembangan frontend dan backend. Untuk *frontend*, teknologi utama yang digunakan adalah Nuxt. Pemilihan Nuxt didasarkan pada kemudahannya dalam membangun antarmuka pengguna, menyediakan fitur yang diperlukan, dan mendukung *Server Side Rendering* (SSR) untuk integrasi dengan *backend* melalui *REST API*. Setiap fitur yang dibuat akan dilakukan blackbox testing hingga aplikasi layak digunakan oleh pengguna. Diharapkan dengan adanya aplikasi ini akan memudahkan manajemen rekrutmen dan komunitas di perusahaan.

Kata kunci: Nuxt, Frontend, library, Scrum

GLOSARIUM

Agile	Sekumpulan metode pengembangan perangkat lunak yang dilakukan secara bertahap dan berulang (iterasi).
Backlog	Daftar tugas, pekerjaan serta fitur yang harus bisa diselesaikan dalam suatu proyek kerja.
Browser	Perangkat lunak yang digunakan untuk mengakses dan menampilkan informasi di internet.
Dashboard	Tampilan visual yang menyajikan data dan informasi penting secara ringkas dan mudah dipahami.
Framework	Kerangka kerja yang digunakan untuk mengembangkan aplikasi berbasis desktop atau aplikasi berbasis web.
Frontend	Bagian dari aplikasi yang dapat dilihat dan berinteraksi langsung dengan pengguna.
Koping	Respons pikiran dan perilaku terhadap situasi penuh tekanan yang bertujuan untuk mengatasi konflik yang muncul akibat situasi tersebut, baik konflik internal maupun eksternal
Library	Kumpulan kode yang telah ditulis sebelumnya (berisi <i>function</i> , <i>class</i> , modul) dan dapat digunakan kembali untuk menyelesaikan tugas-tugas tertentu.
Repository	Tempat di mana informasi, dokumen, kode sumber, dan artefak digital lainnya dapat disimpan dan diakses.
Slicing	Mengonversi tampilan desain web berupa gambar menjadi bagian-bagian yang kecil
Sprint	Batasan waktu yang berisi periode kerja yang sudah ditentukan.

DAFTAR ISI

HALAMAN JUDUL.....	i
HALAMAN PENGESAHAN DOSEN PEMBIMBING.....	ii
HALAMAN PENGESAHAN DOSEN PENGUJI	iii
HALAMAN PERSEMBAHAN.....	v
HALAMAN MOTO.....	vi
KATA PENGANTAR.....	vii
SARI.....	viii
GLOSARIUM.....	ix
DAFTAR ISI.....	x
DAFTAR TABEL.....	xii
DAFTAR GAMBAR	xiii
BAB 1 PENDAHULUAN	1
1.1. Latar Belakang	1
1.2. Ruang Lingkup.....	3
1.3. Tujuan	4
1.4. Manfaat	4
1.5. Sistematika Penulisan.....	4
BAB 2 LANDASAN TEORI DAN TINJAUAN PUSTAKA.....	6
2.1. Sistem Informasi Manajemen.....	6
2.2. Vue js	6
2.3. Nuxt.....	6
2.4. Scrum	7
2.5. Blackbox Testing	8
2.6. Tinjauan Pustaka	8
BAB 3 PELAKSANAAN MAGANG.....	12
3.1. Manajemen Proyek.....	12
3.2. Pengembangan Frontend JoinGeek.....	19
3.3. Pengujian.....	74
3.4. Perencanaan Pengerjaan Versi Selanjutnya	77
3.5. Penutupan Proyek.....	80
BAB 4 REFLEKSI PELAKSANAAN MAGANG	81
4.1. Relevansi Akademik	81
4.2. Pembelajaran Magang	82

	xi
BAB 5 PENUTUP	84
5.1. Kesimpulan.....	84
5.2. Saran.....	84
DAFTAR PUSTAKA	86
LAMPIRAN.....	87

DAFTAR TABEL

Tabel 2.1 Tabel perbandingan penelitian terkait	11
Tabel 3.1 Aktivitas pemegang dalam proyek pengembangan aplikasi JoinGeek	12
Tabel 3.2 Perbandingan antara Vue, React, dan Angular	13
Tabel 3.3 Tabel Sprint.....	14
Tabel 3.4 Pengujian pada fitur sidebar dashboard member	75
Tabel 3.5 Rangkuman pengujian setiap fitur	75

DAFTAR GAMBAR

Gambar 2.1 Metodologi Scrum.....	7
Gambar 3.1 Tampilan Jira proyek pengembangan aplikasi JoinGeek	17
Gambar 3.2 Repository GitHub aplikasi frontend JoinGeek	18
Gambar 3.3 Group telegram pengembangan aplikasi JoinGeek	19
Gambar 3.4 Halaman landing page	20
Gambar 3.5 Tombol untuk membuka fitur autentikasi	21
Gambar 3.6 Fitur login pengguna member	21
Gambar 3.7 Fitur registrasi pengguna member	22
Gambar 3.8 Fitur forgot password pengguna member.....	23
Gambar 3.9 Tampilan email reset password	23
Gambar 3.10 Halaman login pengguna author dan admin.....	24
Gambar 3.11 Halaman detail job salah satu pekerjaan	24
Gambar 3.12 Halaman form apply job.....	25
Gambar 3.13 Halaman form apply internship	26
Gambar 3.14 Halaman fitur author	27
Gambar 3.15 Halaman utama fitur admin.....	28
Gambar 3.16 Halaman utama fitur user management.....	28
Gambar 3.17 Halaman utama fitur roles	29
Gambar 3.18 Halaman utama fitur member.....	30
Gambar 3.19 Halaman utama job listing.....	31
Gambar 3.20 Halaman fitur job available	31
Gambar 3.21 Halaman fitur job applicants	32
Gambar 3.22 Halaman fitur certificate.....	33
Gambar 3.23 Halaman fitur course	33
Gambar 3.24 Halaman fitur internship.....	34
Gambar 3.25 Halaman utama fitur blog.....	35
Gambar 3.26 Halaman utama fitur question	36
Gambar 3.27 Halaman utama fitur account	36
Gambar 3.28 Halaman fitur profile.....	37
Gambar 3.29 Halaman fitur job applied.....	38
Gambar 3.30 Halaman fitur saved job	38
Gambar 3.31 Halaman fitur setting.....	39
Gambar 3.32 Halaman fitur change password	40

Gambar 3.33 Tampilan fitur help & support.....	40
Gambar 3.34 Tampilan fitur kirim email support	41
Gambar 3.35 Struktur folder proyek JoinGeek	42
Gambar 3.36 Tampilan implementasi tabel	43
Gambar 3.37 Fungsi timeout.....	44
Gambar 3.38 Fitur login untuk pengguna author dan admin	45
Gambar 3.39 Controller register user.....	46
Gambar 3.40 Route register user.....	47
Gambar 3.41 Fungsi untuk register user dari frontend.....	48
Gambar 3.42 Kode program pembuatan tampilan sidebar.....	50
Gambar 3.43 Sidebar salah satu dashboard dalam posisi terbuka.....	51
Gambar 3.44 Sidebard salah satu dashboard dalam posisi tertutup	52
Gambar 3.45 Kode program pembuatan dropdown	54
Gambar 3.46 Kode program toggle ubah status.....	55
Gambar 3.47 Kode program filter by step.....	57
Gambar 3.48 Kode program layout edit dan tambah question.....	59
Gambar 3.49 Kode Program submit data apply pekerjaan.....	61
Gambar 3.50 Array untuk menu pada dashboard member.....	62
Gambar 3.51 Fungsi untuk mengunduh data dalam format excel.....	65
Gambar 3.52 State untuk menyimpan data jenis form autentikasi.....	65
Gambar 3.53 Kode program profil member.....	66
Gambar 3.54 Kode program fitur bookmark.....	68
Gambar 3.55 Menampilkan data pengguna pada form input	69
Gambar 3.56 Kode program filter kategori pekerjaan	71
Gambar 3.57 Kode program implementasi kloning fitur	72
Gambar 3.58 Contoh kode program memunculkan ikon	74
Gambar 3.59 Tombol login menggunakan google.....	77

BAB 1

PENDAHULUAN

1.1. Latar Belakang

PT. Komuri Indonesia GeekGarden adalah sebuah *software house* yang sekarang memfokuskan jasa dalam pengembangan aplikasi berskala kecil menengah. GeekGarden sebagai sebuah perusahaan terpanggil untuk membantu perusahaan-perusahaan berinvestasi pada suatu sistem atau aplikasi. Hal ini bertujuan untuk meningkatkan efisiensi dan efektivitas dalam membangun suatu perusahaan atau instansi. Layanan yang Geekgarden sediakan yaitu *jasa Mobile App Development, Web App Development, E-commerce Website Development, Geek Enterprise System* dan *Manpower Sharing* (GeekGarden, 2022). Beberapa produk yang penulis kembangkan selama magang antara lain Voteroom, Yada Telemarketer, dan JoinGeek.

Voteroom merupakan salah satu dari tiga aplikasi pada ekosistem milik partai. Ekosistem ini dibuat untuk menunjang perhitungan suara pada pemilu 2024. Tiga aplikasi pada ekosistem ini antara lain aplikasi Hitung Mandiri pada perangkat seluler untuk penginputan suara pada setiap TPS, Warroom digunakan untuk manajemen *user* untuk aplikasi *mobile* dan Voteroom, dan Voteroom sebagai bagian dari ekosistem berperan sebagai dashboard untuk monitoring suara-suara yang telah masuk melalui aplikasi Hitung Mandiri.

Penulis ikut mengembangkan aplikasi Voteroom selama empat bulan, mulai dari bulan September hingga Desember 2023. Tahap-tahap pengembangan yang diikuti oleh penulis mulai dari tahap implementasi, hingga penyelesaian aplikasi, tanpa mencakup tahap testing dan peningkatan kecepatan aplikasi. Pada proyek ini, pemagang bekerja di bawah mentor dengan mengikuti arahan yang diberikan.

Yada adalah aplikasi *marketplace* inovatif dan transparan untuk bahan bangunan, mengusung model S2B2C yang memberikan solusi lengkap bagi kebutuhan material konstruksi. Yada, sebagai sebuah ekosistem bisnis, memiliki dua modul utama: modul *back office* untuk manajemen bisnis dan aplikasi pelanggan untuk pengalaman belanja yang efisien. Modul *back office* ini terdiri dari dua aplikasi, yaitu Yada Admin dan Yada Telemarketer. Yada Admin berfungsi untuk mengelola segala sesuatu yang berkaitan dengan aplikasi pelanggan. Yada Telemarketer adalah aplikasi yang digunakan untuk mengelola transaksi yang telah dibayar dan menawarkannya kepada mitra, hingga proses transaksi selesai.

Pemagang ikut mengembangkan aplikasi Yada Telemarketer selama satu bulan, mulai dari bulan Desember 2023 hingga Januari 2024. Tahap yang diikuti oleh pemagang yaitu tahap penyelesaian dengan tugas memperbaiki *bug-bug* yang ada, dan menambahkan fitur-fitur tambahan. Pada proyek ini, pemagang bekerja dibawah proyek manajer yang mana tugas-tugas yang akan diberikan ketika ada bug-bug yang muncul dan jika ada perubahan yang diminta oleh klien.

JoinGeek merupakan salah satu aplikasi berbasis web yang disediakan oleh GeekGarden. Aplikasi ini berfungsi sebagai platform untuk manajemen pelamar lowongan dan pelamar pekerjaan di perusahaan ini, baik sebagai karyawan tetap, *freelance*, maupun peserta magang, serta manajemen komunitas GeekGarden. JoinGeek menyajikan informasi mengenai lowongan pekerjaan yang tersedia, spesifikasi pekerjaan, keahlian yang diharapkan, formulir pendaftaran, serta berbagai aspek lain yang terkait dengan proses perekrutan di GeekGarden. Untuk manajemen komunitas, JoinGeek juga menyediakan informasi mengenai acara-acara komunitas, pelatihan-pelatihan, dan lain sebagainya.

Pengembangan aplikasi JoinGeek yang diikuti oleh penulis berlangsung selama empat bulan, dari bulan Desember 2023 hingga Maret 2024. Tahapan pengembangan yang diikuti oleh penulis meliputi slicing ui, integrasi api, hingga penyelesaian aplikasi, termasuk pengujian aplikasi. Pengembangan aplikasi ini telah menggunakan metode pengembangan aplikasi yaitu scrum, dan pemagang bekerja langsung dibawah proyek manajer. Proyek ini dikerjakan dalam beberapa tahap dan untuk tahap pertama telah berhasil diselesaikan oleh pemagang.

Dalam laporan ini, akan dibahas mengenai pengembangan *frontend* sistem informasi manajemen rekrutmen dan komunitas menggunakan *framework* Nuxt, yaitu aplikasi JoinGeek. Pemilihan ini didasarkan pada proses pengembangan aplikasi yang telah menggunakan *framework* pengembangan proyek yaitu Scrum, sehingga manajemen pengerjaan proyek menjadi lebih terstruktur dan aplikasi bisa diselesaikan meskipun baru sampai tahap yang pertama. Hal lain yang menjadi dasar pemilihan proyek ini adalah penggunaan teknologi Nuxt sebagai *framework frontend*. Nuxt merupakan teknologi baru yang dipelajari oleh pemagang, sehingga diharapkan dengan adanya laporan ini, pemahaman pemagang mengenai *framework* ini dapat bertambah, dan pembaca juga bisa mendapatkan pembelajaran dari perspektif seseorang yang baru belajar mengenai suatu teknologi dengan langsung mengimplementasikannya dalam pembuatan aplikasi.

1.2. Ruang Lingkup

Pelaksanaan magang sebagai *Frontend Developer* di GeekGarden Software House berlangsung selama 6 bulan, dimulai dari 18 September 2023 hingga 18 Maret 2024. Selama periode magang, pemagang aktif berkontribusi dalam pengembangan tiga proyek aplikasi, yakni Voteroom, Yada Telemarketer, dan JoinGeek.

Pengembangan aplikasi Voteroom yang diikuti oleh penulis berlangsung selama empat bulan, dimulai dari bulan September hingga Desember 2023. Tahap-tahap pengembangan yang diikuti oleh pemagang mulai dari tahap implementasi, hingga penyelesaian aplikasi, tanpa mencakup tahap testing dan peningkatan kecepatan aplikasi. Pekerjaan yang dilakukan oleh pemagang dalam pengembangan aplikasi ini antara lain pembuatan komponen-komponen, pembuatan *dashboard*, integrasi dengan API, serta redesain seluruh *dashboard*, dan masih banyak lagi.

Pengembangan aplikasi Yada Telemarketer yang diikuti oleh penulis berlangsung selama satu bulan, dari bulan Desember 2023 hingga Januari 2024. Tahap yang diikuti oleh pemagang adalah tahap penyelesaian, dengan tugas memperbaiki *bug-bug* yang ada dan menambahkan fitur-fitur tambahan yang diminta oleh klien. Tugas yang dilakukan oleh penulis dalam pengembangan aplikasi ini antara lain memperbaiki tampilan pada seluruh tabel, menampilkan produk bundel yang dibeli oleh pelanggan, memperbaiki perhitungan total belanja pada fitur detail pesanan dan nota barang, serta memperbaiki seluruh templat pesan WhatsApp dengan menambahkan produk bundel yang dibeli (jika ada) dan memperbaiki perhitungan di dalamnya.

Pengembangan aplikasi JoinGeek yang diikuti oleh pemagang berlangsung selama empat bulan, dari bulan Desember 2023 hingga Maret 2024. Tahapan pengembangan yang diikuti oleh pemagang meliputi *slicing ui*, implementasi hingga penyelesaian aplikasi, termasuk pengujian aplikasi. Pekerjaan yang dilakukan oleh pemagang dalam pengembangan aplikasi ini antara lain pembuatan fitur autentikasi pengguna, integrasi *dashboard author* dengan API, *slicing* tampilan *dashboard* admin ke web dan mengintegrasikan dengan API, *slicing* tampilan *dashboard member* ke web dan mengintegrasikan dengan API, serta pekerjaan-pekerjaan lainnya.

1.3. Tujuan

Tujuan dibuatnya tugas akhir ini adalah untuk mendalami serta memahami dengan lebih baik proses pengembangan sistem informasi manajemen rekrutmen dan komunitas versi pertama. Fokus utama pada pembahasan ini adalah pada pengembangan *frontend* aplikasi JoinGeek menggunakan Framework Nuxt.

1.4. Manfaat

Manfaat yang diperoleh antara lain :

- a. Pemahaman yang lebih mendalam tentang proses pengembangan sistem informasi manajemen rekrutmen dan komunitas, khususnya pada pengembangan *frontend*.
- b. Pengimplementasian dan pemahaman yang lebih dalam tentang teknologi dan alat yang digunakan, terutama *Framework Nuxt*.
- c. Pengimplementasian dan pemahaman metode pengembangan aplikasi yang digunakan yaitu Scrum.
- d. Pemahaman cara bekerja dalam tim dan pembagian tugas

1.5. Sistematika Penulisan

Sistematika penulisan disusun untuk memberikan gambaran umum terhadap keseluruhan isi laporan. Laporan tugas akhir ini terdiri dari lima bab dengan susunan sistematika penulisan sebagai berikut :

- a. Bab 1 Pendahuluan
Bab ini membahas mengenai latar belakang, ruang lingkup magang, tujuan, manfaat, dan sistematika penulisan.
- b. Bab 2 Landasan Teori dan Tinjauan Pustaka
Bab ini membahas teori-teori yang memiliki keterikatan dengan topik laporan sebagai landasan teori dalam penyusunan laporan dan juga berisi penelitian/pekerjaan apa saja yang sudah dilakukan dari berbagai referensi makalah yang relevan dengan topik laporan.
- c. Bab 3 Pelaksanaan Magang
Bab ini menguraikan gambaran aktivitas selama proses pelaksanaan magang khususnya penjelasan mengenai pengembangan *frontend* untuk sistem informasi manajemen rekrutmen dan komunitas menggunakan *Framework Nuxt*.

d. Bab 4 Refleksi Pelaksanaan Magang

Bab ini menguraikan hasil refleksi yang penulis dapat selama pelaksanaan magang di GeekGarden Software House.

e. Bab 5 Penutup

Bab ini berisi kesimpulan dari hasil pengembangan *frontend* untuk sistem informasi manajemen rekrutmen dan komunitas menggunakan *Framework* Nuxt dan juga berisi saran agar tujuan/manfaat dari pengembangan aplikasi dapat tercapai pada pengembangan berikutnya

BAB 2

LANDASAN TEORI DAN TINJAUAN PUSTAKA

2.1. Sistem Informasi Manajemen

Sistem Informasi Manajemen (SIM) merupakan sebuah sistem yang dirancang untuk mendukung suatu perusahaan atau organisasi dalam mengelola, menyimpan, mengakses, dan menganalisis informasi yang diperlukan untuk proses pengambilan keputusan manajemen. Aspek-aspek dalam sistem informasi manajemen meliputi perangkat keras (*hardware*), perangkat lunak (*software*), sumber daya manusia (*brainware*), prosedur, basis data, dan komunikasi data serta jaringan komputer (Farhan Saputra & Franciscus Dwikotjo Sri Sumantyo, 2023). Sistem Informasi Manajemen bekerja melalui aspek-aspek tersebut dengan mengumpulkan data dari berbagai sumber, kemudian mengolahnya sehingga dapat disajikan dalam bentuk informasi yang telah divisualisasi. Dengan hal tersebut, perusahaan dapat mengembangkan perencanaan program dengan lebih maksimal dan efektif.

2.2. Vue js

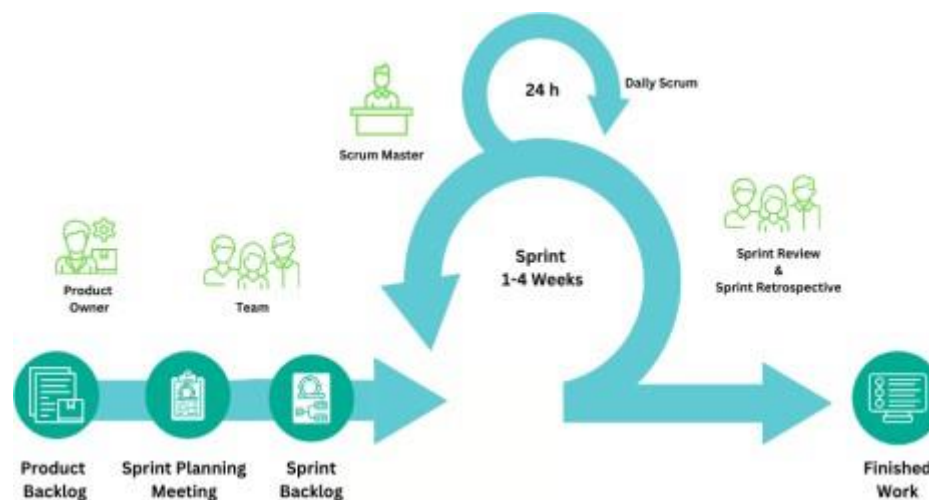
Vue adalah *framework* JavaScript untuk membangun antarmuka pengguna. *Framework* ini dibangun di atas standar HTML, CSS, dan JavaScript, serta menyediakan model pemrograman deklaratif berbasis komponen yang membantu untuk mengembangkan antarmuka pengguna dengan efisien, tanpa mengabaikan kompleksitasnya. Vue diciptakan oleh karyawan Google saat itu, Evan You pada tahun 2014 (You, n.d.). Kelebihan vue dibandingkan dengan *library* atau *framework* serupa lainnya adalah lebih mudah dipelajari dan digunakan. *Library* ini dikenal dan digunakan oleh Alibaba, GitLab, Baidu, 9GAG, dan diapresiasi oleh para pengembang dan desainer secara global.(Pšenák & Tibenský, 2020)

2.3. Nuxt

Nuxt adalah sebuah *framework open-source* tingkat tinggi yang dibangun di atas Vue. Dengan kata lain, pengguna dapat membangun tampilan web dalam JavaScript menggunakan sistem komponen dari Vue. *Framework* ini mengabstraksi sebagian besar konfigurasi yang rumit seperti sinkronisasi, *middleware*, dan *routing*. Selain itu, Nuxt juga menyediakan kemampuan penting untuk melakukan proses rendering di sisi server, yang dikenal dengan Server Side Rendering (SSR)(Helda & Suryadi, 2023). SSR akan meningkatkan performa web, memperbaiki optimasi mesin pencari (SEO), dan mempercepat muatan halaman awal.

2.4. Scrum

Scrum merupakan salah satu *framework* dari metode Agile yang berfokus pada pengembangan produk untuk mengatasi segala macam isu secara kreatif dan adaptif. Scrum membagi rencana jangka panjang menjadi beberapa periode kerja jangka pendek yang disebut *sprint*. Tujuannya adalah menjaga komunikasi antar tim dan bekerja berkelanjutan untuk membuat *software* berkualitas yang sesuai dengan kebutuhan klien.



Gambar 2.1 Metodologi Scrum

Sumber : (Muay et al., 2024)

Pada gambar 2.1, terdapat ilustrasi proses pengembangan produk menggunakan metodologi scrum. Metodologi ini dijalankan dengan mengisi beberapa posisi untuk menjalankan perannya masing-masing seperti *Product Owner* yang bertanggung jawab atas arah pengembangan produk dan memastikan mencapai hasil semaksimal mungkin, *Scrum Master* yang bertanggung jawab agar implementasi metode scrum dalam pengerjaan proyek oleh tim dapat berjalan dengan baik, serta anggota tim yang bertanggung jawab untuk mengerjakan produk dan menghasilkan produk yang berkualitas. Pihak-pihak tersebut akan saling berkolaborasi dan bekerja sama agar tujuan yang direncanakan tercapai.

Selama fase ini, ide-ide diimplementasikan menjadi hasil yang berharga dalam bentuk produk (Muay et al., 2024). Ide-ide tersebut kemudian diuraikan menjadi fitur-fitur yang tercatat dalam *product backlog*. Setelah itu, dilakukan *sprint planning meeting* untuk menetapkan daftar pekerjaan yang akan dilakukan dan batas waktu untuk menyelesaikannya (*sprint*), yang kemudian tercatat dalam *sprint backlog*. Dalam *sprint* yang ditentukan, dilakukan *daily scrum* yang membahas kemajuan pekerjaan sebelumnya, hari ini, dan mengidentifikasi hambatan dalam pengerjaan. Setelah sprint selesai, dilakukan *sprint review* untuk mengevaluasi hasil dari sprint yang telah selesai.

2.5. Blackbox Testing

Black Box Testing adalah suatu metode pengujian perangkat lunak dan aplikasi yang bertujuan untuk mengevaluasi apakah perangkat lunak dan aplikasi tersebut berfungsi dengan optimal atau tidak, tanpa memperhatikan rincian internal dari kode program atau struktur internalnya. Pengujian ini berfokus pada aspek eksternal dari aplikasi, seperti antarmuka pengguna, fungsi-fungsi yang tersedia, dan kecocokan alur fungsi dengan proses bisnis. Pengujian dilakukan dengan mencoba program yang telah dibuat dengan memasukkan data ke dalam setiap formulir yang tersedia untuk menguji fungsionalitasnya.(Febriyantia et al., 2021). Pengujian ini penting untuk memastikan bahwa program beroperasi sesuai dengan kebutuhan yang diinginkan oleh perusahaan atau organisasi.

2.6. Tinjauan Pustaka

Penelitian ini tidak dapat diselesaikan tanpa merujuk kepada penelitian terdahulu yang relevan dengan topik yang diangkat. Penelitian terdahulu yang berhubungan dengan pengembangan *frontend* sistem informasi manajemen rekrutmen dan komunitas menggunakan framework Nuxt akan menjadi sumber referensi utama dan bahan pembandingan dalam penelitian ini. penelitian-penelitian terdahulu tersebut akan memberikan landasan yang kokoh untuk pemahaman yang lebih mendalam mengenai pengembangan sistem yang diusulkan.

Helda & Suryadi (2023) melakukan sebuah penelitian berjudul “Koneksi Tanpa Batas: Membangun Portfolio Web Interaktif dengan Vue, Nuxt, Dan API”. Penelitian ini dilatarbelakangi oleh kekurangan sistem informasi sebelumnya dalam mempublikasikan program kegiatan yayasan serta kurangnya layanan donasi terbuka di situs web untuk menarik dermawan yang ingin memberikan bantuan untuk mendukung program yayasan tersebut. Peneliti menggunakan metode *Research and Development* untuk menyempurnakan desain aplikasi sebelumnya dan menambahkan fitur-fitur baru. Pengerjaan aplikasi dilakukan dengan menggunakan *framework Nuxt* untuk menciptakan antarmuka yang responsif. Hasil dari penelitian ini adalah sebuah aplikasi berupa web yang menampilkan beberapa informasi seperti program yayasan, struktur organisasi, galeri, dan layanan donasi terbuka.

Penelitian serupa dilakukan oleh (Hanafie et al., 2023) dalam laporan berjudul “Pengembangan Website Yayasan Al-Hizam Menggunakan Framework Nuxt JS”, yang dilatarbelakangi oleh eksplorasi teknologi modern seperti *Vue*, *Nuxt*, dan *API* untuk membangun portofolio web interaktif. Peneliti menggunakan metode *waterfall* dalam pembuatan web tersebut, dengan langkah-langkah pengerjaan yang mencakup instalasi dan konfigurasi *Nuxt*, desain antarmuka dengan *Vue*, pengaturan rute dan navigasi, integrasi dengan *API*, serta *deployment*. Hasil dari penelitian ini adalah sebuah web portofolio yang dibuat dengan mengimplementasikan *Nuxt*, *Vue*, dan integrasi dengan *API* eksternal. Web portofolio yang dihasilkan memiliki desain menarik, performa optimal, dan integrasi yang efektif dengan sumber daya eksternal, sehingga meningkatkan kualitas web sehingga menarik perhatian calon klien atau perekrut.

Rafif Wicaksana Putra,(2023)melakukan penelitian berupa Pengembangan Aplikasi Jurnal Emosi Berbasis Progressive Web App. Permasalahan yang menjadi fokus penelitian berasal dari tekanan dan tuntutan yang dapat menyebabkan stres pada seseorang. Untuk mengatasi masalah tersebut, akan dibangun sebuah sistem sebagai alat koping untuk penanganan stres, yang membantu pengguna dalam mengelola emosi dan suasana hati. Pengembangan aplikasi ini menggunakan *Vue.js* sebagai teknologi utama, serta menerapkan *Progressive Web App (PWA)* Hasil dari penelitian ini adalah sebuah aplikasi Jurnal Emosi yang telah mengimplementasikan *Progressive Web App* yang dapat digunakan di berbagai jenis perangkat secara *offline* yang mana memungkinkan pengguna untuk mencatat dan mengelola emosi mereka dengan lebih efektif.

Penelitian yang dilakukan oleh (Aryasta, 2022) dalam laporan berjudul “Pengembangan Front End Sistem Informasi Menggunakan Kerangka Kerja Vue js” yang mana membahas mengenai pengembangan aplikasi sebagai solusi untuk meningkatkan efisiensi aktivitas akuntansi pada sebuah perusahaan klien dari GeekGarden. Dalam laporannya, pengembangan frontend aplikasi ini menggunakan framework Vue sebagai teknologi utama sertam menggunakan balck box testing sebagai metode pengujian. Hasil akhir dari laporan tersebut menunjukkan bahwasanya penggunaan vue sangat membantu proses pengembangan aplikasi karena menyediakan hal-hal untuk mendukung kebutuhan pengmebangan aplikasi tersebut.

Dalam penelitian oleh Albarra Naufala Erdanto (2021), membahas tentang peningkatan efektivitas pengelolaan *state* menggunakan pola *Prop Drilling* dalam Vue.js pada fitur finansial Jala Tech. Dalam kasus tersebut, peneliti mencoba melakukan penulisan ulang kode pada fitur-fitur tertentu untuk mengubah pengelolaan *state* dari pola *event bus* menjadi pola *prop drilling*. Metode pengembangan yang digunakan oleh peneliti adalah Scrum, yang melibatkan beberapa tahap seperti *sprint planning*, *development*, *sprint review*, dan *sprint retrospective*. Hasil dari penelitian ini berupa peningkatan efektivitas pengelolaan *state* pada fitur finansial yang telah menerapkan pola *prop drilling* pada kode *frontend*.

Adapun perbandingan antara penelitian terdahulu dengan penelitian ini telah disajikan dalam bentuk tabel yang dapat dilihat pada tabel 2.1.

Tabel 2.1 Tabel perbandingan penelitian terkait

No	Peneliti (Tahun)	Judul	Metode Pengembangan	Teknologi	Pengujian
1	Nurul Helda, Suryadi (2023)	Koneksi Tanpa Batas: Membangun Portfolio Web Interaktif dengan Vue, Nuxt, Dan API	Waterfall	Framework Nuxt dan Vue js	Pagespeed
2	Ahmad Hanafie, Amaliah Chintami D.A, Nur Isnayanti B, Sulfitra	Pengembangan Website Yayasan Al-Hizam Menggunakan Framework Nuxt JS	Research and Development	Framework Nuxt dan Vue js	Blackbox Testing
3	Rafif Wicaksana Putra (2022)	Pengembangan Aplikasi Jurnal Emosi Berbasis Progressive Web App	Rapid Application Development (RAD)	Vue js dan Firebase	Blackbox testing
4	Andhika Rizky Aryasta (2022)	Pengembangan Front End Sistem informasi Akuntansi menggunakan Kerangka Kerja Vue.js	-	Vue js	Blackbox Testing
5	Albarra Naufala Erdanto (2022)	Peningkatan Efektivitas Pengelolaan State menggunakan Pola Prop Drilling Vuejs pada Fitur finansial Jala Tech	Scrum	Vue js	-

BAB 3

PELAKSANAAN MAGANG

3.1. Manajemen Proyek

3.1.1. Aktivitas Magang

Pengerjaan proyek pengembangan aplikasi JoinGeek telah dimulai sejak Agustus 2023 dan berlanjut hingga saat ini. Penulis secara resmi telah bergabung sebagai pengembang *frontend* dalam proyek ini sejak Januari 2024. Rangkuman aktivitas yang telah dilakukan oleh pemangang dalam menjalankan tugasnya di proyek pengembangan aplikasi JoinGeek dapat ditemukan di Tabel 3.1.

Tabel 3.1 Aktivitas pemangang dalam proyek pengembangan aplikasi JoinGeek

No	Aktivitas	Waktu
1	Belajar teknologi yang digunakan dalam proyek JoinGeek sambil mengerjakan proyek Yada Telemarketer	4 Minggu
2	Belajar mengenai proyek JoinGeek	1 Minggu
3	Pengerjaan <i>dashboard author</i>	1 Minggu
4	Pengerjaan <i>dashboard member</i>	1 Minggu
5	Pengerjaan <i>dashboard user</i>	1 Minggu
6	Pengerjaan <i>dashboard admin</i>	1 Minggu
7	Pengerjaan <i>front office</i>	1 Minggu
8	Perbaikan <i>bug-bug</i> dan juga pemindahan tugas	1 Minggu

3.1.2. Penggunaan Framework Nuxt

Nuxt merupakan salah satu framework pengembangan frontend JavaScript yang dibangun di atas Vue.js. Pemilihan Nuxt sebagai framework dibandingkan dengan framework frontend lainnya, seperti Angular atau framework yang dibangun di atas React, didasarkan pada keunggulan Vue.js itu sendiri. Perbandingan lebih lanjut dapat dilihat pada Tabel 3.2.

Tabel 3.2 Perbandingan antara Vue, React, dan Angular

	Vue	React	Angular
Kemudahan Penggunaan	Mudah digunakan karena struktur yang sederhana.	Memerlukan pemahaman JSX dan konsep state management	kompleksitas dan banyaknya fitur built-in.
Kemudahan Belajar	Kurva belajar yang mudah	Kurva belajar moderat	Kurva belajar yang lebih sulit
Komunitas	Komunitas yang aktif namun lebih kecil dibandingkan React dan Angular.	Komunitas yang sangat besar dan aktif dengan banyak library dan alat tambahan.	Komunitas yang besar dengan dukungan resmi dari Google.
Ekosistem	Ekosistem yang berkembang dengan banyak plugin dan alat tambahan, tetapi masih terbatas dibandingkan React dan Angular.	kosistem yang luas dengan banyak library, alat tambahan, dan integrasi yang tersedia.	ramework yang sangat komprehensif dengan banyak fitur built-in, tetapi terkadang kurang fleksibel
Fleksibilitas	Fleksibel dan dapat diintegrasikan secara bertahap dengan proyek yang ada.	Sangat fleksibel dengan banyak pilihan untuk library tambahan dan alat pengembangan.	Kurang fleksibel dibandingkan Vue dan React karena merupakan framework yang lebih terstruktur.

Dari beberapa poin yang terdapat pada Tabel 3.2, alasan utama pemilihan Nuxt, yang dibangun di atas Vue.js, adalah kemudahan dalam belajar dan penggunaan. Pendekatan sederhana dan terstruktur dari Vue.js membantu pengembang baru untuk dengan cepat memahami dan mulai bekerja dengan Nuxt yang didasarkan atas teknologi ini. Hal ini menjadikan Nuxt sebagai pilihan yang tepat untuk proyek seperti JoinGeek, di mana pengembang dapat memperoleh pengalaman praktis dan memperdalam pemahaman mereka mengenai pengembangan frontend tanpa terlalu terbebani oleh kompleksitas teknologi yang ada.

Aplikasi JoinGeek, yang merupakan salah satu aplikasi internal perusahaan, menempatkan pemegang sebagai pengembang frontend utama dalam proyek ini. Hal ini memungkinkan pemegang untuk lebih leluasa bereksperimen dengan risiko yang lebih kecil. Selain itu, pertimbangan lainnya adalah agar pemegang tidak hanya fokus pada aspek teknis penulisan kode dan teknologi, tetapi juga memahami proses pengembangan proyek secara keseluruhan.

3.1.3. Metodologi Pengembangan Aplikasi

Dalam pengembangan proyek aplikasi ini, digunakan sebuah metode yaitu metode Scrum. Dalam metode ini, tim proyek dibagi menjadi beberapa peran, yaitu produk manager sebagai *Product Owner*, Proyek Manager sebagai *Scrum Master*, serta UI/UX Designer, Frontend Developer, dan Backend Developer masuk di dalam tim pengembang. Setiap anggota tim bertanggung jawab untuk mencapai *sprint goals* dan memastikan kelancaran proses pengembangan.

Langkah awal yang dilakukan adalah membuat *product backlog* yang dikerjakan oleh *product owner* dan juga *scrum master* dengan berdiskusi dengan *stackholder* terkait. Setelah *product backlog* selesai, mereka menyusun *sprint planning* sesuai dengan batas waktu pengerjaan yang ditentukan sejak awal menjadi sebuah *sprint backlog*. *Sprint backlog* tersebut didiskusikan kembali dengan tim pengembang untuk menentukan kesepakatan *sprint*. Sprint yang telah disepakati akan dijalankan dengan dipimpin oleh *scrum master* termasuk juga revisi sampai proses *sprint* selesai dan pekerjaan telah diselesaikan. Langkah-langkah ini akan diulangi kembali dari awal pada fase-fase selanjutnya. Tabel waktu pengerjaan sprint dapat dilihat pada Tabel 3.3.

Tabel 3.3 Tabel Sprint

No	Tanggal	Kegiatan
1	Selasa, 23 Januari 2024	Belajar Repositori Join Geek
2	Rabu, 24 Januari 2024	Belajar Repositori Join Geek
3	Kamis, 25 Januari 2024	Belajar Repositori Join Geek
4	Jumat, 26 Januari 2024	Belajar Repositori Join Geek
5	Senin, 29 Januari 2024	Belajar Repositori Join Geek
6	Selasa, 30 Januari 2024	Belajar Repositori Join Geek
7	Rabu, 31 Januari 2024	Belajar Repositori Join Geek
8	Kamis, 01 Februari 2024	Libur
9	Jumat, 02 Februari 2024	Pembuatan REST API dan mengintegrasikannya di halaman Internship pada Dashboard author
10	Senin, 05 Februari 2024	Pembuatan REST API ubah status dan mengintegrasikannya di halaman Job Listing pada Dashboard author
11	Selasa, 06 Februari 2024	Pembuatan REST API dan mengintegrasikannya di halaman Job Available pada Dashboard author
12	Rabu, 07 Februari 2024	Pembuatan REST API di halaman Job Applicants pada Dashboard author

No	Tanggal	Kegiatan
13	Kamis, 08 Februari 2024	Mengintegraskan REST API di halaman Job Applicants pada Dashboard author
14	Jumat, 09 Februari 2024	Libur
15	Senin, 12 Februari 2024	Pembuatan REST API dan mengintegraskannya di halaman Blog pada Dashboard author
16	Selasa, 13 Februari 2024	Pembuatan REST API dan mengintegraskannya di halaman Create/Update Blog pada Dashboard author
17	Rabu, 14 Februari 2024	Libur Pemilu
18	Kamis, 15 Februari 2024	Perbaikan Bug-bug minor
19	Jumat, 16 Februari 2024	Pembuatan REST API dan mengintegraskannya di halaman Question pada Dashboard author
20	Senin, 19 Februari 2024	Pembuatan REST API Question pada apply job dan mengintegraskannya di halaman di halaman Apply Career
21	Selasa, 20 Februari 2024	Slicing UI dashboard member dan halaman Profile, Job Applied, Change Password, dan Help & Support
22	Rabu, 21 Februari 2024	Pembuatan REST API dan mengintegraskannya di halaman Account pada Dashboard author
23	Kamis, 22 Februari 2024	Slicing UI halaman Setting pada dashboard member
24	Jumat, 23 Februari 2024	Slicing UI dashboard admin dan halaman user management, role, dan member
25	Senin, 26 Februari 2024	Pembuatan REST API serta mengintegraskannya di halaman Profile dan Job Applied pada Dashboard Member
26	Selasa, 27 Februari 2024	Slicing UI halaman Saved Job pada dashboard member
27	Rabu, 28 Februari 2024	Pembuatan REST API dan mengintegraskannya di halaman role dan user management pada dashboard admin
28	Kamis, 29 Februari 2024	Pembuatan REST API dan mengintegraskannya di halaman member pada dashboard admin
29	Jumat, 01 Maret 2024	Pembuatan REST API serta mengintegraskannya di halaman Change Password dan Help&Support pada Dashboard Member
30	Sabtu, 02 Maret 2024	Slicing UI untuk modal registrasi, login dan reset password untuk member
31	Senin, 04 Maret 2024	Pembuatan REST API dan mengintegraskannya modal registrasi, login dan reset password untuk member
32	Selasa, 05 Maret 2024	Menampilkan profile member yang telah login di navbar, membuat fitur bookmark di halaman detail job
33	Rabu, 06 Maret 2024	Menampilkan data pada form Apply Job bagi member yang telah login dan membuat filter untuk daftar pekerjaan di landing page
34	Kamis, 07 Maret 2024	Menyalin fitur-fitur halaman pada dashboard author ke dashboard admin
35	Jumat, 08 Maret 2024	Menyalin fitur-fitur halaman pada dashboard author ke dashboard admin
36	Senin, 11 Maret 2024	Pemindahan tugas ke orang lain dan Pembuatan REST API serta mengintegraskannya di halaman Saved Job pada Dashboard Member
37	Selasa, 12 Maret 2024	Pembuatan REST API serta mengintegraskannya di halaman Settings pada Dashboard Member

No	Tanggal	Kegiatan
38	Rabu, 13 Maret 2024	Pemindahan tugas ke orang lain an memperbaiki bug pada halaman Job Applicant di dashboard author dan admin
39	Kamis, 14 Maret 2024	Pemindahan tugas ke orang lain dan memperbaiki bug-bug ikon yang hilang
40	Jumat, 15 Maret 2024	Perpisahan

Pengeimplementasian metode scrum dalam pengembangan proyek JoinGeek adalah membuat *product backlog*, yang dilakukan oleh *product owner* dan *scrum master* dengan berdiskusi dengan *stackholder* terkait. Setelah *product backlog* selesai, mereka menyusun perencanaan *sprint* sesuai dengan batas waktu yang telah ditetapkan sejak awal, menjadi *sprint backlog*. Kemudian *sprint backlog* tersebut didiskusikan dengan tim pengembang untuk menetapkan sprint sesuai dengan kemampuan tim pengembang. Setelah itu, *sprint* akan dijalankan dengan dipimpin oleh *scrum master*, juga dengan *scrum review* hingga proses sprint selesai dan pekerjaan telah diselesaikan. Langkah-langkah ini akan diulangi dari awal pada fase-fase selanjutnya.

3.1.4. Kegiatan Rutinitas

Sprint merupakan tahapan utama dalam metode *Scrum*, yang mana tim pengembangan mulai bekerja pada pengembangan proyek sesuai dengan tenggat waktu *sprint* yang ditentukan, yang mana dalam proyek ini disetujui dengan tenggat waktu yaitu seminggu. Selain pengerjaan tugas-tugas dalam sprint tersebut, terdapat agenda-agenda lain yang harus dilakukan untuk menunjang proses pengerjaan proyek seperti *Sprint Planning*, *Daily Scrum*, dan *Sprint Review* di waktu yang telah dijadwalkan.

Sprint Planning dijadwalkan setiap hari Senin, dimulai pukul 09.00 hingga selesai. Pertemuan ini dilakukan secara daring dan dihadiri oleh *Product Owner*, Proyek Manajer, serta tim pengembang yang dipimpin oleh Proyek Manajer. Agenda ini bertujuan untuk mendiskusikan *backlog* yang telah disusun, menentukan tugas yang harus dilakukan oleh setiap individu, dan mengatur prioritas pekerjaan ke dalam *sprint*.

Daily Scrum atau *Daily Standups* diadakan setiap hari Selasa, Rabu, dan Kamis, dimulai pukul 09.00 hingga selesai. Pertemuan ini dilakukan secara daring dan dihadiri oleh *Product Owner*, Proyek Manajer, serta tim pengembang yang dipimpin oleh Proyek Manajer. Agenda ini bertujuan untuk membahas rencana kerja harian tim serta mendiskusikan hambatan yang dihadapi beserta solusinya.

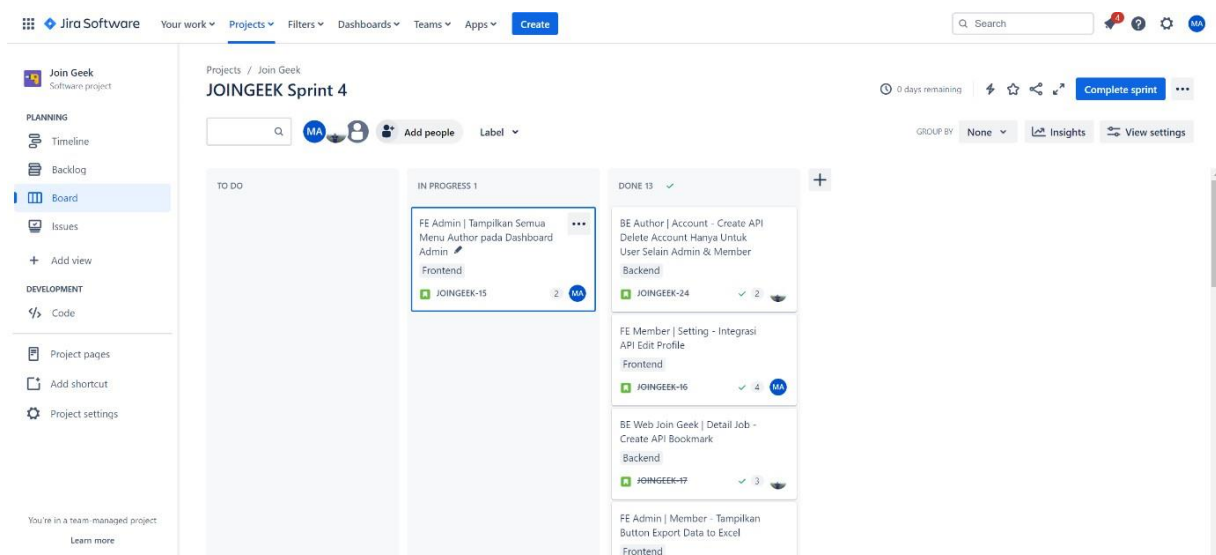
Sprint Review diadakan setiap hari Jumat, dimulai pukul 15.00 hingga selesai. Pertemuan ini dilakukan secara daring yang dihadiri product owner, proyek manajer, dan juga tim pengembang yang dipimpin oleh proyek manajer. Agenda ini bertujuan untuk mendiskusikan *progress* pengerjaan dalam *sprint* yang berlangsung serta mengevaluasi hasil *sprint* tersebut yang diadakan pada akhir setiap *sprint*.

3.1.5. Pemantauan dan Pengendalian Proyek

Proyek Pengembangan yang berlangsung memerlukan kerjasama dan komunikasi yang baik di dalam tim. Oleh karena itu, digunakanlah beberapa aplikasi untuk mendukung hal tersebut, antara lain:

a. Jira

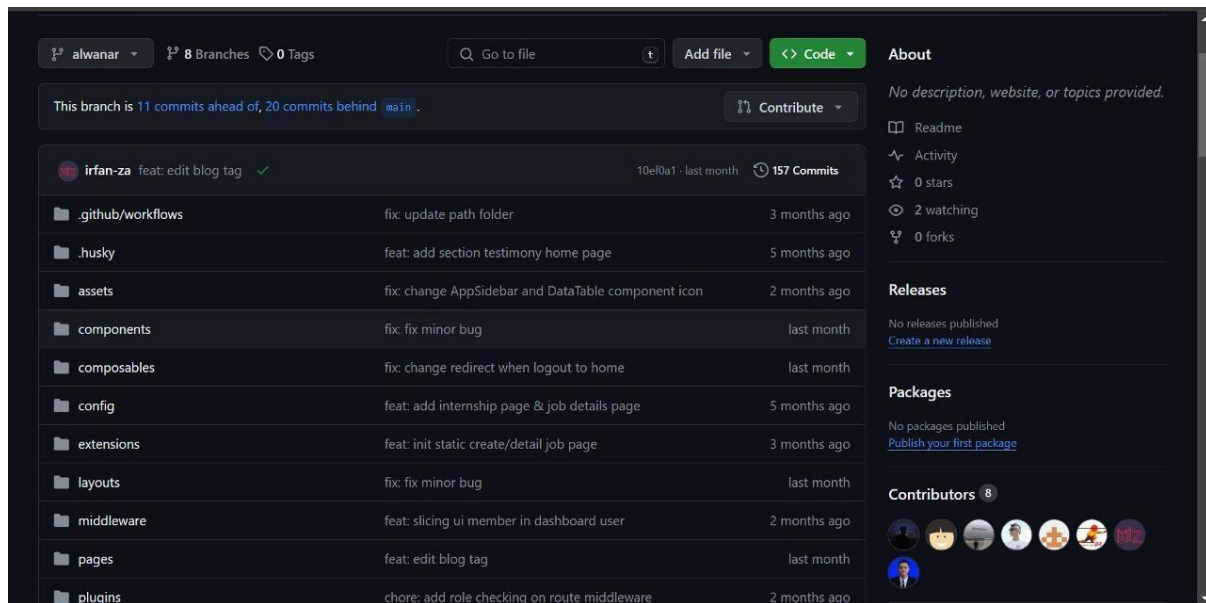
Jira adalah aplikasi yang digunakan untuk manajemen tim dalam proyek pengembangan aplikasi JoinGeek. Jira dipilih dibandingkan dengan perangkat lunak serupa lainnya karena fitur-fitur dan alat yang tersedia di dalamnya sangat mendukung pengembangan proyek yang menggunakan metode Scrum. Beberapa fungsi yang dapat dilakukan dalam Jira meliputi pembuatan *backlog*, penyusunan sprint, penjadwalan dan alokasi tugas, dokumentasi proyek, pelacakan kemajuan tugas, serta pemantauan progres keseluruhan proyek. Tampilan Jira untuk pengembangan aplikasi JoinGeek dapat dilihat pada gambar 3.1.



Gambar 3.1 Tampilan Jira proyek pengembangan aplikasi JoinGeek

b. Github

Github digunakan sebagai layanan untuk menyimpan *repository* proyek yang berbasis web. Dengan adanya github ini, para pengembang dapat menyimpan perubahan kode program secara realtime serta dapat mengakses *repository* secara *remote* dimana saja melalui internet sehingga memudahkan kolaborasi antar pengembang. Tampilan *repository* Github aplikasi *frontend* JoinGeek dapat dilihat pada gambar 3.2.



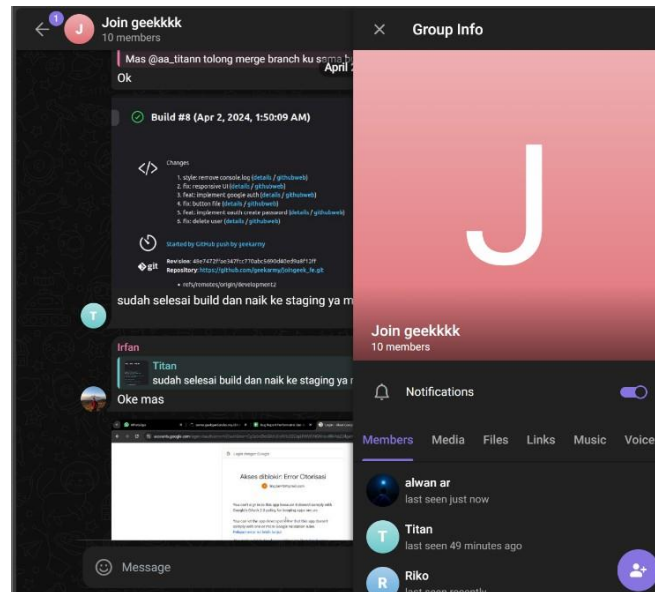
Gambar 3.2 Repository GitHub aplikasi frontend JoinGeek

c. Google Meet

Google Meet digunakan sebagai platform untuk pertemuan daring dalam agenda proyek seperti *Sprint Planning*, *Daily Scrum*, dan *Sprint Review*. Google Meet dipilih dibandingkan dengan perangkat lunak serupa karena aplikasi ini gratis dan mudah digunakan karena berbasis *web*.

d. Telegram

Telegram merupakan aplikasi yang digunakan untuk komunikasi antar tim dalam pengerjaan sprint. Grup Telegram digunakan untuk mempercepat penerimaan informasi dan *update* mengenai proyek dan melaporkan kemajuan ataupun kendala dan pengerjaan. Tampilan grup telegram tim proyek pengembangan aplikasi JoinGeek dapat dilihat pada gambar 3.3.



Gambar 3.3 Group telegram pengembangan aplikasi JoinGeek

3.2. Pengembangan Frontend JoinGeek

3.2.1. Aplikasi JoinGeek

JoinGeek merupakan salah satu aplikasi internal berbasis web milik GeekGarden. Aplikasi ini berbentuk sebuah platform yang bertujuan untuk manajemen pelamar pekerjaan, baik sebagai karyawan *full-time*, *freelance*, maupun sebagai pemagang, serta sebagai platform manajemen komunitas GeekGarden. Tujuan lain dari aplikasi ini adalah untuk mengumpulkan individu yang tertarik dengan GeekGarden sehingga menjadi sebuah komunitas, agar individu-individu tersebut dapat memperkenalkan GeekGarden kepada lebih banyak orang, dan memperluas jangkauan kehadiran GeekGarden ke khalayak yang lebih luas.

Proyek ini awalnya direncanakan sebagai pengembangan aplikasi yang telah dibuat sebelumnya dengan nama yang sama, yaitu JoinGeek. Pengerjaan proyek ini dimulai sebagai inisiasi perusahaan untuk menuju fase selanjutnya, yaitu menyediakan *Software as a Service* (SaaS), dengan cara memperbaiki dan mengembangkan aplikasi internal terlebih dahulu. Setelah berbagai pertimbangan dan analisis, akhirnya diputuskan untuk membangun aplikasi ini dari awal. Beberapa pertimbangan untuk memulai dari awal antara lain penggunaan versi teknologi yang sudah usang, kompleksitas kode program yang sulit dipelajari karena pengembang aslinya tidak lagi bekerja di perusahaan, dan juga kebutuhan untuk mengadopsi teknologi terbaru.

3.2.2. Fitur-Fitur Aplikasi

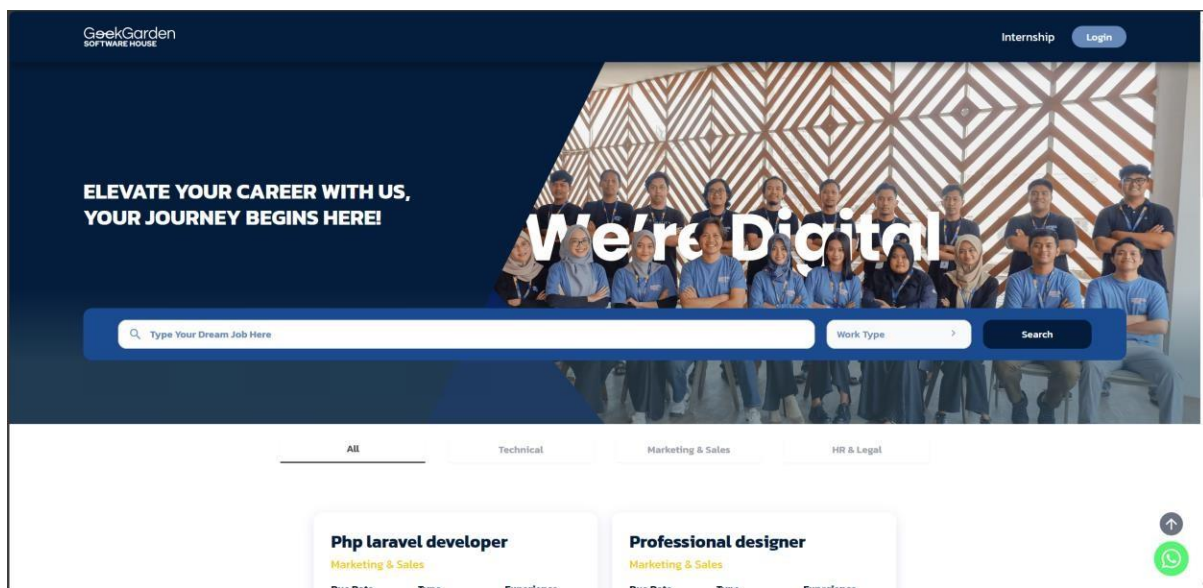
Untuk mewujudkan fungsi dari aplikasi JoinGeek, fitur-fitur dalam aplikasi ini dibagi menjadi dua jenis berdasarkan aksesnya, yaitu *Front Office* dan *Back Office*.

Front Office

Front Office adalah kumpulan halaman-halaman atau fitur-fitur yang dapat diakses langsung oleh pengunjung tanpa memerlukan proses autentikasi pengguna. Fitur-fitur ini mencakup berbagai layanan dan informasi yang tersedia untuk umum. Fitur-fitur tersebut antara lain ;

a. Landing Page

Landing Page merupakan halaman utama dari web JoinGeek yang akan pertama kali muncul saat web diakses. Halaman ini berisi informasi umum mengenai perusahaan dan aplikasi, serta informasi lainnya berupa daftar pekerjaan yang tersedia dan terbuka untuk pendaftaran. Informasi pekerjaan tersebut mencakup informasi dasar seperti jenis pekerjaan, batas waktu pendaftaran, dan lain-lain. Pencarian pekerjaan juga dipermudah dengan adanya fitur pencarian dan filter. Tampilan halaman ini dapat dilihat pada gambar 3.4.



Gambar 3.4 Halaman landing page

b. Autentikasi

Fitur ini memungkinkan pengunjung untuk memverifikasi identitas sebagai pengguna *member* agar dapat mengakses *dashboard member*. Fitur ini dapat diakses dengan menekan tombol *login* pada *navbar* di halaman *Front Office*, yang kemudian akan menampilkan modal untuk fitur tersebut. Tampilan tombol tersebut dapat dilihat pada gambar 3.5. Fitur ini juga memiliki sub-fitur seperti *login*, *register*, dan *forgot password*.



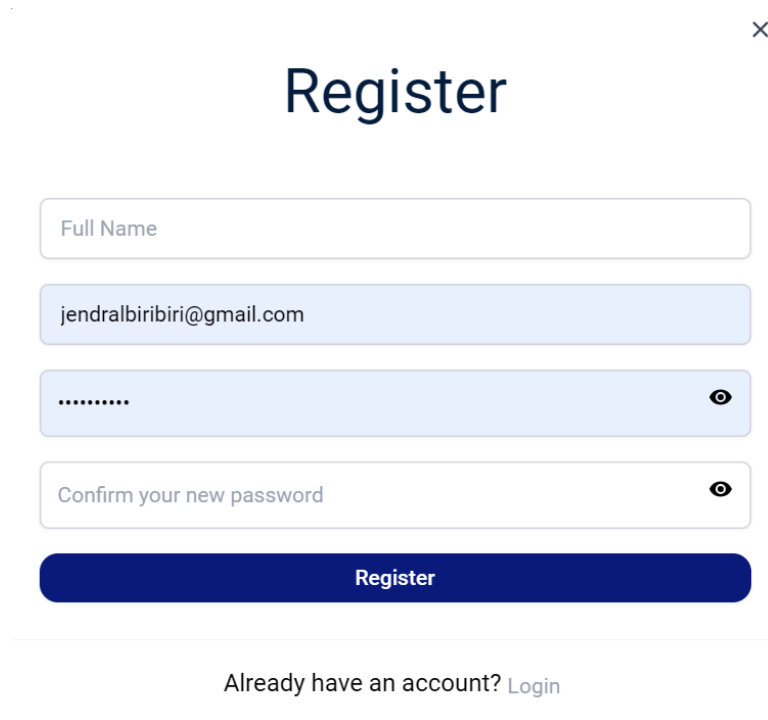
Gambar 3.5 Tombol untuk membuka fitur autentikasi

Login merupakan sebuah fitur di mana pengunjung mengisi formulir agar sistem dapat mengenali dan memberikan akses mereka ke *dashboard member* sesuai data pengguna terkait. *Login* dilakukan dengan mengisi *email* dan *password* akun, kemudian pengguna dialihkan ke *dashboard member*. Tampilan fitur ini dapat dilihat pada gambar 3.6.

 A login modal window with a white background and a dark blue header. The header contains the word 'Login' in large blue font and a close button (X) in the top right corner. Below the header, there are two input fields: the first contains the email 'jendralbiribiri@gmail.com' and the second contains a masked password '.....' with an eye icon for toggling visibility. To the right of the password field is a link 'Forgot Password?'. Below the input fields is a large dark blue button labeled 'Login'. Underneath this button is the text 'OR' and a button with the Google logo and the text 'Continue with Google'. At the bottom of the modal, there is a link 'Don't have an account? Register'.

Gambar 3.6 Fitur login pengguna member

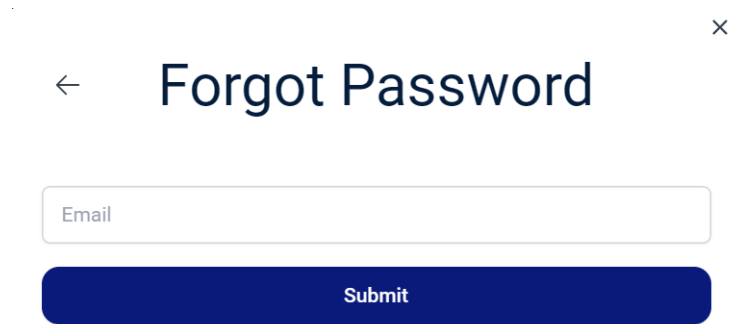
Register merupakan sebuah fitur di mana pengunjung dapat mendaftarkan diri dan membuat akun sebagai pengguna *member*. Pendaftaran dapat dilakukan dengan mudah oleh pengunjung dengan mengisi *input* pada formulir yang tersedia, seperti nama lengkap, *email*, dan *password*. Tampilan fitur ini dapat dilihat pada gambar 3.7.



The image shows a web registration form titled "Register" with a close button (X) in the top right corner. The form contains four input fields: "Full Name", "Email" (containing "jendralbiribiri@gmail.com"), "Password" (masked with dots and having an eye icon), and "Confirm your new password" (also having an eye icon). Below the fields is a dark blue "Register" button. At the bottom, there is a link that says "Already have an account? Login".

Gambar 3.7 Fitur registrasi pengguna member

Forgot Password merupakan fitur yang digunakan apabila pengguna *member* lupa *password* akun yang telah dibuat. Fitur ini dapat diakses dengan mengisi alamat *email* pada input yang tersedia. Selanjutnya, sistem akan mengirimkan *email* ke alamat tersebut untuk proses *reset password*. Tampilan fitur ini dapat dilihat pada gambar 3.8. Tampilan *email* yang diterima setelah 'submit' dapat dilihat pada gambar 3.9

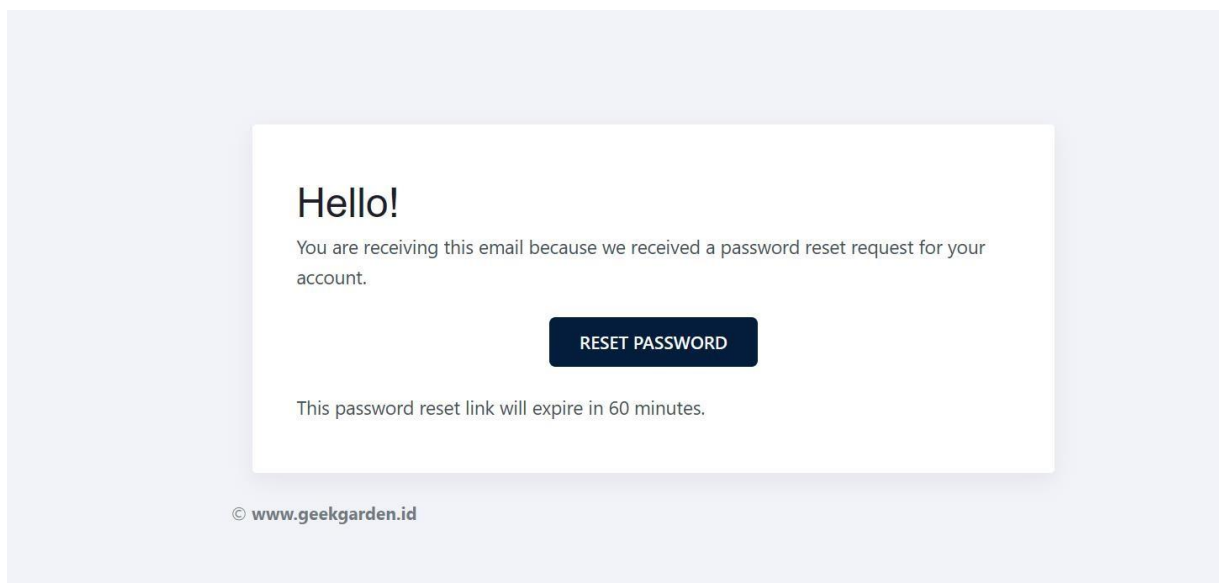


← Forgot Password

Email

Submit

Gambar 3.8 Fitur forgot password pengguna member



Gambar 3.9 Tampilan email reset password

c. Login untuk pengguna *author* dan *admin*

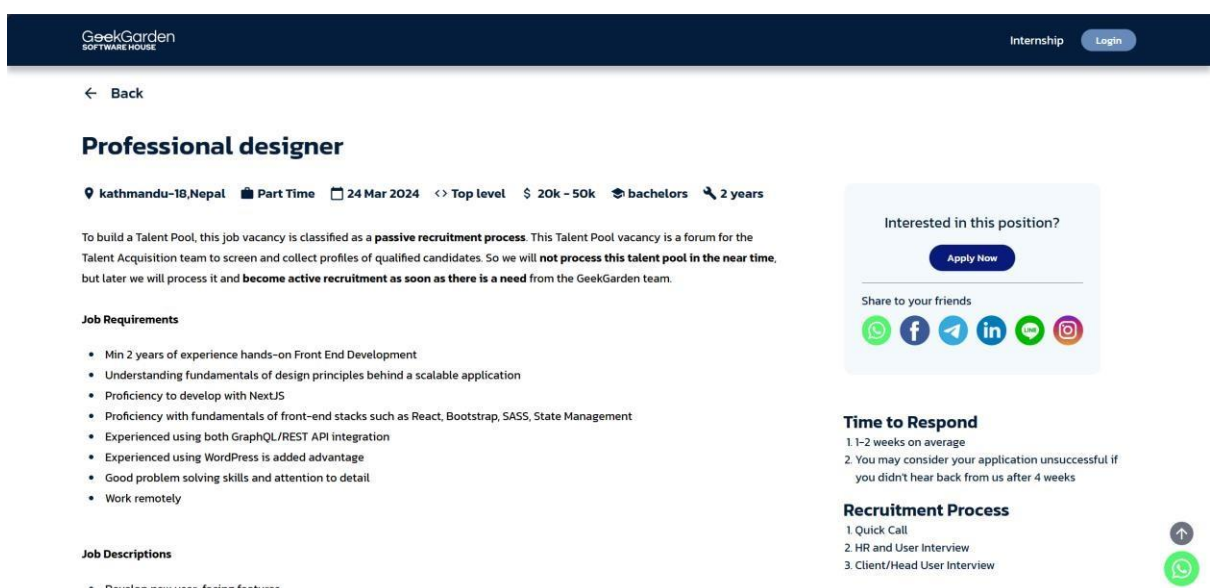
Login merupakan sebuah fitur di mana pengunjung mengisi formulir agar sistem dapat mengidentifikasi dan memberikan akses ke *dashboard author* atau *admin* dengan data pengguna yang sesuai. Halaman ini hanya dapat diakses dengan memasukkan *URL '/login'* pada *browser*. Setelah proses *login* dengan mengisi *email* dan *password* akun berhasil, pengguna akan diarahkan ke *dashboard author* atau *admin*, tergantung pada akun pengguna yang digunakan. Tampilan halaman fitur ini dapat dilihat pada gambar 3.9.



Gambar 3.10 Halaman login pengguna author dan admin

d. Detail Job

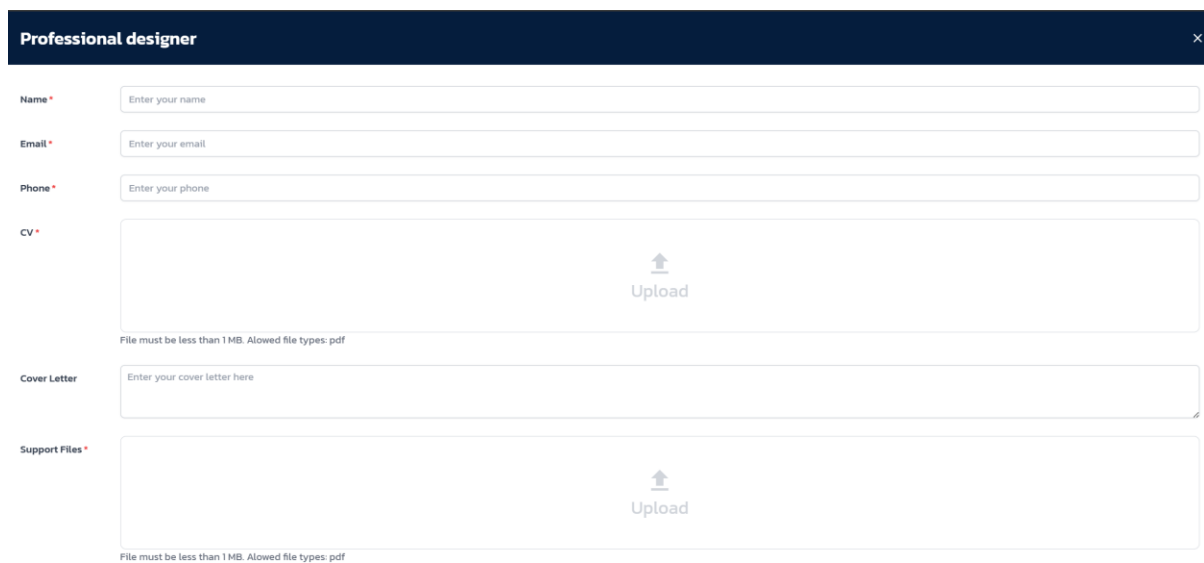
Detail Job adalah sebuah fitur yang menyajikan informasi terperinci mengenai pekerjaan yang lowongannya dibuka oleh perusahaan. Halaman ini memuat informasi tambahan seperti gaji, lokasi, level pekerjaan, syarat-syarat pekerjaan, deskripsi tugas yang akan diemban, dan proses yang harus dilalui untuk melamar. Pengguna dapat mengakses halaman ini dengan mengeklik pekerjaan yang diminati pada halaman *landing page*. Tampilan halaman fitur ini terlihat pada gambar 3.11.



Gambar 3.11 Halaman detail job salah satu pekerjaan

e. Form Apply Job

Form Apply Job adalah sebuah fitur yang berisi formulir untuk melamar pekerjaan. Fitur ini dapat diakses dengan menekan tombol ‘*Apply Job*’ pada halaman *Detail Job*, yang akan memunculkan *modal* yang berisi formulir. Informasi yang perlu diisi meliputi identitas pribadi yang dibutuhkan oleh perusahaan, seperti nama, *email*, CV, dan lain-lain. Tampilan halaman ini dapat dilihat pada gambar 3.12.



The image shows a modal window titled "Professional designer" with a close button (X) in the top right corner. The form contains the following fields:

- Name ***: A text input field with the placeholder "Enter your name".
- Email ***: A text input field with the placeholder "Enter your email".
- Phone ***: A text input field with the placeholder "Enter your phone".
- CV ***: A file upload area with an "Upload" button and a note: "File must be less than 1 MB. Allowed file types: pdf".
- Cover Letter**: A text area with the placeholder "Enter your cover letter here".
- Support Files ***: A file upload area with an "Upload" button and a note: "File must be less than 1 MB. Allowed file types: pdf".

Gambar 3.12 Halaman form apply job

f. Form Apply Internship

Form Apply Internship merupakan sebuah fitur yang berisi formulir untuk melamar sebagai pemagang. Fitur ini dapat diakses dengan menekan tombol ‘*Internship*’ pada *navbar*. Informasi yang perlu diisi meliputi identitas pribadi yang dibutuhkan oleh perusahaan, seperti nama, *email*, CV, dan lain-lain. Perbedaannya dengan formulir untuk *apply* pekerjaan adalah bahwa posisi yang dilamar pemagang akan diisikan secara manual, seperti input untuk durasi magang, dan masih banyak lagi. Tampilan fitur ini dapat dilihat pada gambar 3.13.

GeekGarden
SOFTWARE HOUSE

Internship Login

← Back

Internship Form

Full Name *
E.g: Ridzky Kurniawan

Current Address *
E.g: Jl. Brawijaya, Tamantirto, Kec. Kasihan, Kab.Bantul, DIY 55183

Phone *
E.g: 089977086644

Email *
E.g: kurniawanrr@gmail.com

School/University *
E.g: SMA N 1 Sleman/Universitas Negeri Yogyakarta

What is your major? *
E.g: Multimedia/Teknik Informatika

What semester are you in? *
E.g: 5

WhatsApp icon

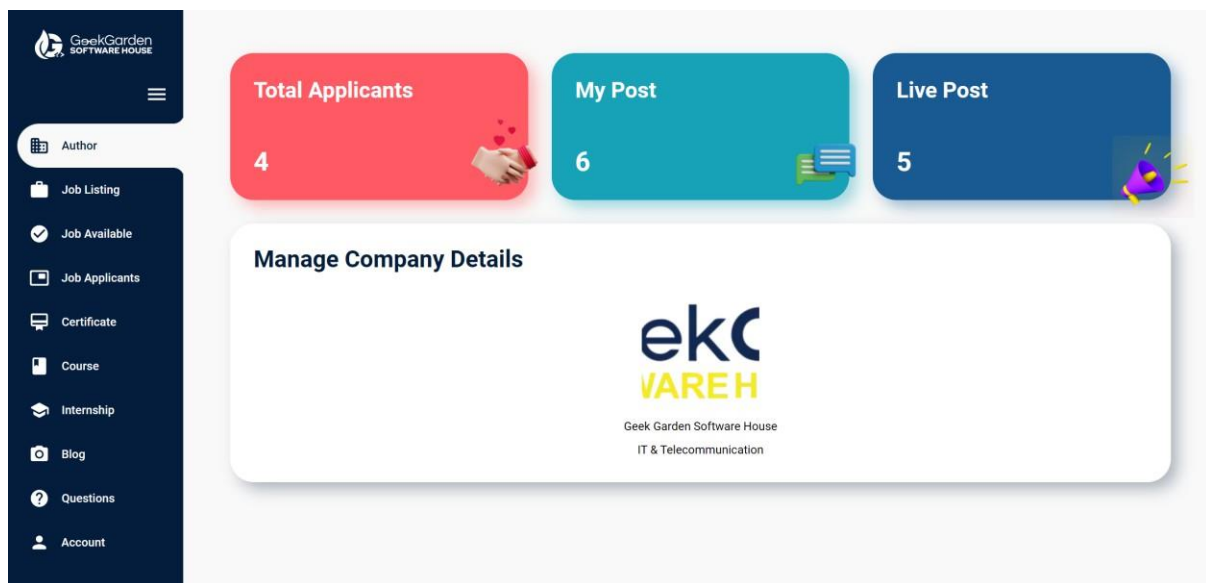
Gambar 3.13 Halaman form apply internship

Back Office

Back Office adalah kumpulan halaman-halaman atau fitur-fitur yang dikelompokkan ke dalam dashboard-dashboards dan hanya dapat diakses oleh pengunjung setelah mereka melewati proses autentikasi. Sistem ini membedakan tiga jenis pengguna: *member*, *author*, dan *admin*, yang masing-masing diberikan akses ke *dashboard* mereka sendiri. *Member* merupakan jenis pengguna yang memberikan akses kepada pengunjung umum dan dapat memberikan mereka kemungkinan untuk mendaftar sebagai pengguna member. *Author*, di sisi lain, adalah jenis pengguna yang diberikan wewenang untuk mengelola lowongan, pelamar dan mengelola konten di *front office*, seperti blog, lowongan pekerjaan, dan lainnya. Sedangkan *admin* adalah jenis pengguna yang memiliki akses penuh ke fitur-fitur *author* dan tambahan untuk bertanggung jawab atas pengaturan jenis pengguna dan aksesnya, informasi perusahaan serta pengaturan terhadap pengguna lainnya. Fitur-fitur yang ada di *back office* adalah sebagai berikut :

a. Author

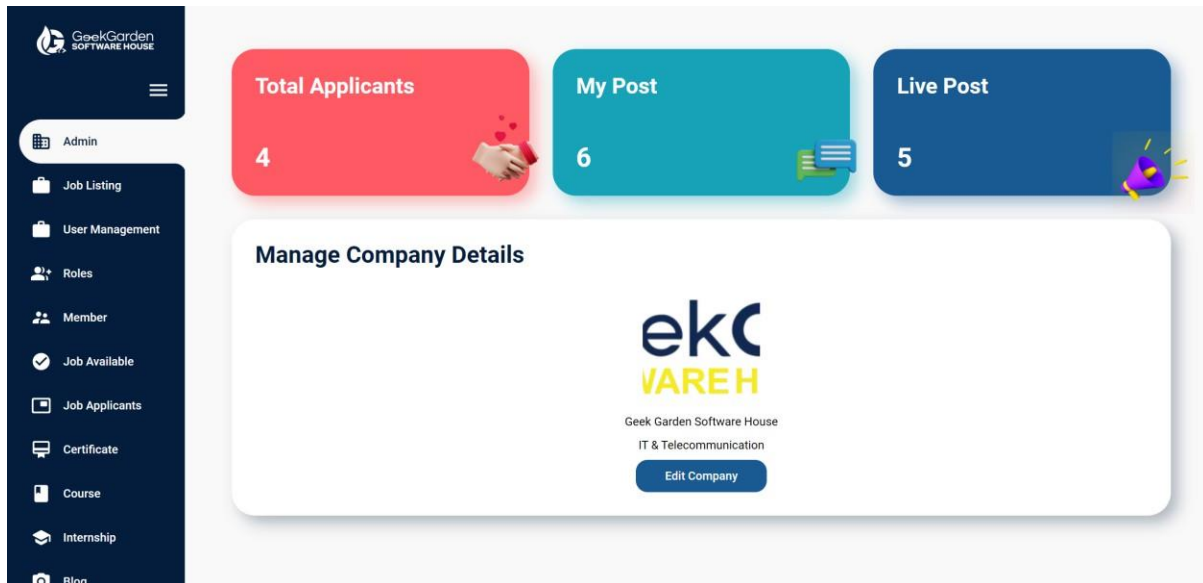
Author merupakan fitur atau halaman yang pertama kali diakses setelah *login* sebagai pengguna *author* di *dashboard author*. Halaman ini menyajikan rangkuman informasi penting dari *dashboard*, seperti jumlah pelamar, postingan, dan postingan yang telah dipublikasikan. Selain itu, halaman ini juga menyediakan informasi singkat mengenai perusahaan, memberikan gambaran umum kepada pengguna tentang identitas perusahaan. Tampilan halaman utama fitur ini dapat dilihat pada gambar 3.14.



Gambar 3.14 Halaman fitur author

b. Admin

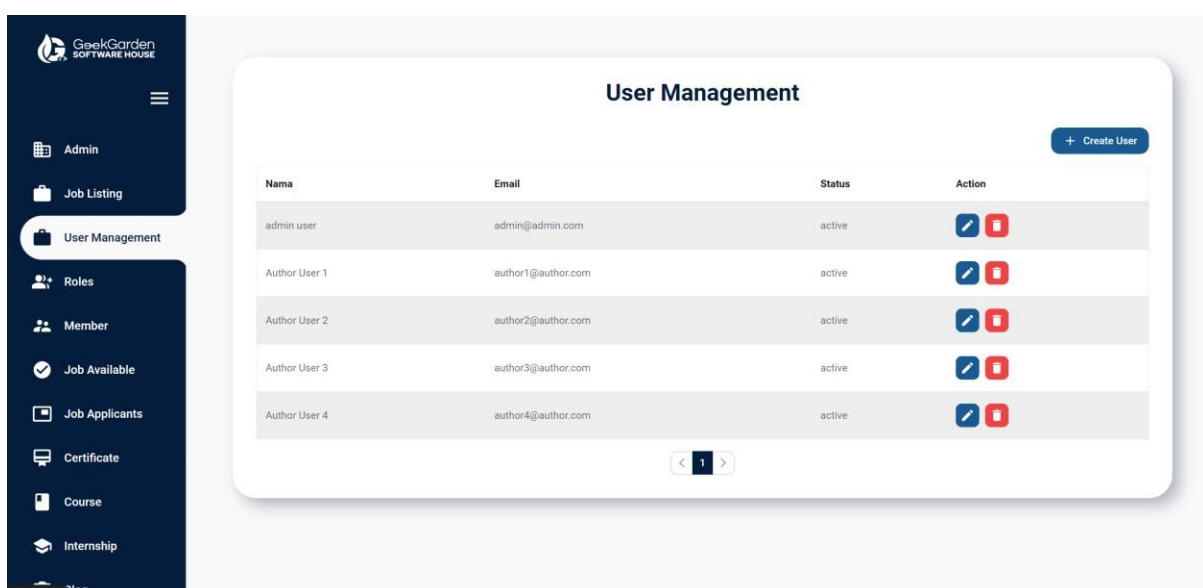
Admin adalah fitur atau halaman yang pertama kali diakses setelah *login* sebagai pengguna *admin* di *dashboard admin*. Fungsi dari fitur ini mirip dengan fitur "*Author*" pada *dashboard author*. Perbedaannya adalah bahwa dalam fitur ini, pengguna *admin* diberikan wewenang untuk mengubah atau menambah informasi perusahaan, memastikan bahwa informasi yang tersedia selalu terkini dan akurat. Tampilan halaman utama fitur ini dapat dilihat pada gambar 3.15.



Gambar 3.15 Halaman utama fitur admin

c. User Management

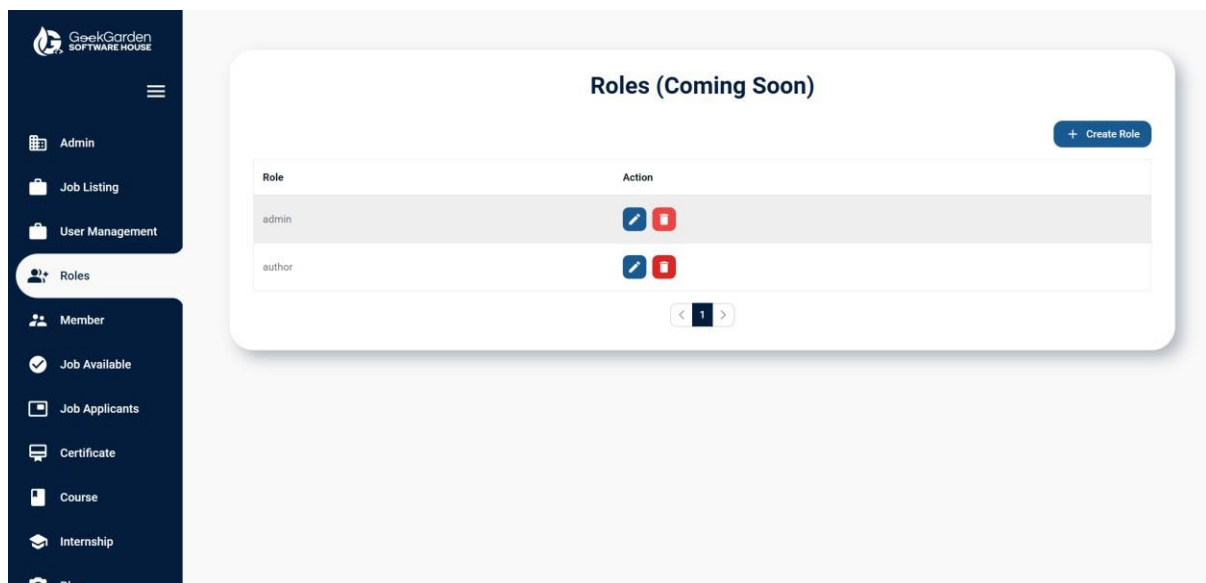
User Management adalah fitur yang digunakan untuk mengelola akun pengguna *author* dan *admin*. Fungsinya memungkinkan pengguna untuk menambah, mengubah, dan menghapus akun-akun tersebut. Ini memberikan kontrol penuh kepada pengguna *admin* atas manajemen pengguna, memastikan bahwa hak akses terhadap *dashboard author* dan *admin* tetap terorganisir dan terkendali. Fitur ini hanya dapat diakses melalui *dashboard admin*. Tampilan utama halaman fitur ini dapat dilihat pada gambar 3.16.



Gambar 3.16 Halaman utama fitur user management

d. Roles

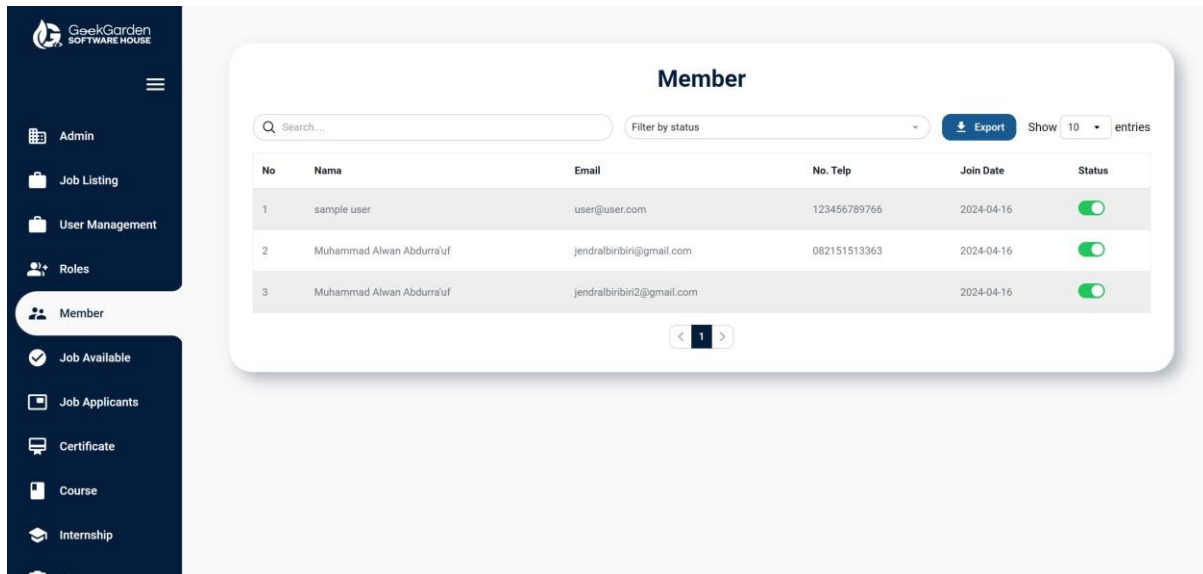
Roles adalah fitur yang berfungsi untuk mengelola jenis pengguna pada aplikasi ini. Pada fitur ini, pengguna dapat membuat, mengubah, dan menghapus jenis pengguna yang ada. Fitur ini juga memungkinkan untuk menentukan hak akses yang diberikan kepada setiap jenis pengguna, sehingga pengguna dapat mengatur dengan cermat fitur-fitur yang dapat diakses oleh masing-masing jenis pengguna yang dibuat atau diubah. Dengan demikian, *Roles* memberikan fleksibilitas dan kontrol yang lebih besar terhadap manajemen akses dalam aplikasi. Fitur ini hanya dapat diakses melalui *dashboard admin*. Tampilan utama halaman fitur ini dapat dilihat pada gambar 3.17.



Gambar 3.17 Halaman utama fitur roles

e. Member

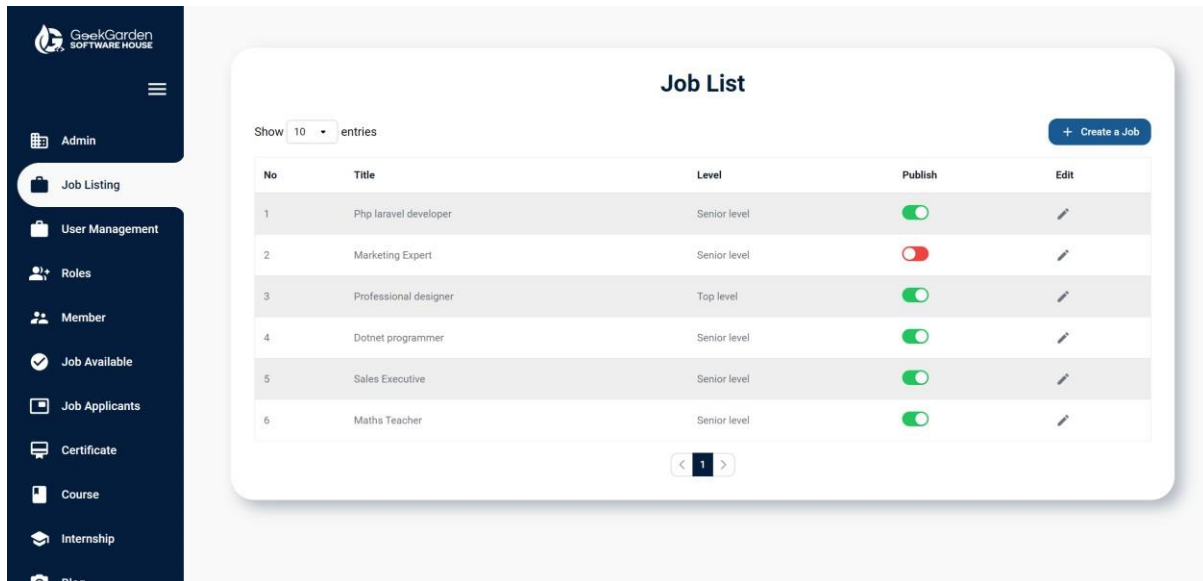
Member merupakan sebuah fitur yang bertujuan untuk mengelola pengguna-pengguna *member* yang telah terdaftar dalam aplikasi ini. Selain itu, pengguna juga diberikan kemampuan untuk menonaktifkan status akun pengguna jika dibutuhkan, serta dapat mengunduh data-data pengguna untuk keperluan tertentu. Fitur ini hanya dapat diakses melalui *dashboard admin*. Tampilan utama halaman fitur ini dapat dilihat pada gambar 3.18.



Gambar 3.18 Halaman utama fitur member

f. Job Listing

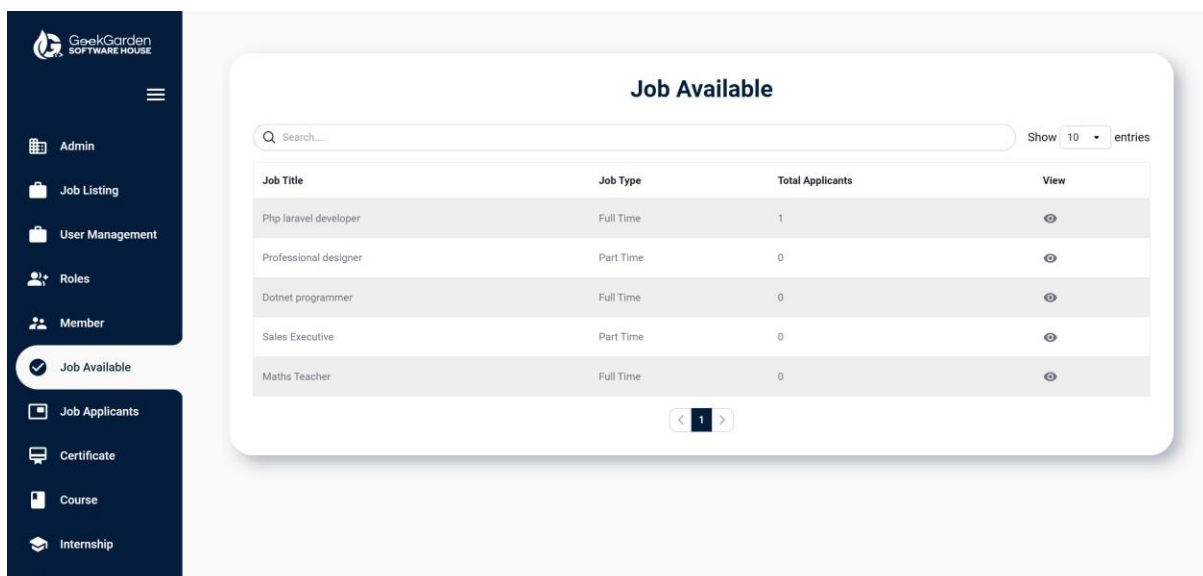
Job Listing adalah fitur yang berfungsi untuk mengelola pekerjaan-pekerjaan yang tersedia di perusahaan. Dalam fitur ini, pengguna memiliki kemampuan untuk menambah, mengubah, dan menghapus pekerjaan-pekerjaan tersebut. Selain itu, pengguna juga dapat memutuskan untuk mempublikasikan atau tidak mempublikasikan pekerjaan tersebut, yang menandakan apakah lowongan tersebut dibuka atau tidak. Fitur ini dapat diakses melalui *dashboard author* dan *admin*. Tampilan utama halaman fitur ini dapat dilihat pada gambar 3.19.



Gambar 3.19 Halaman utama job listing

g. Job Available

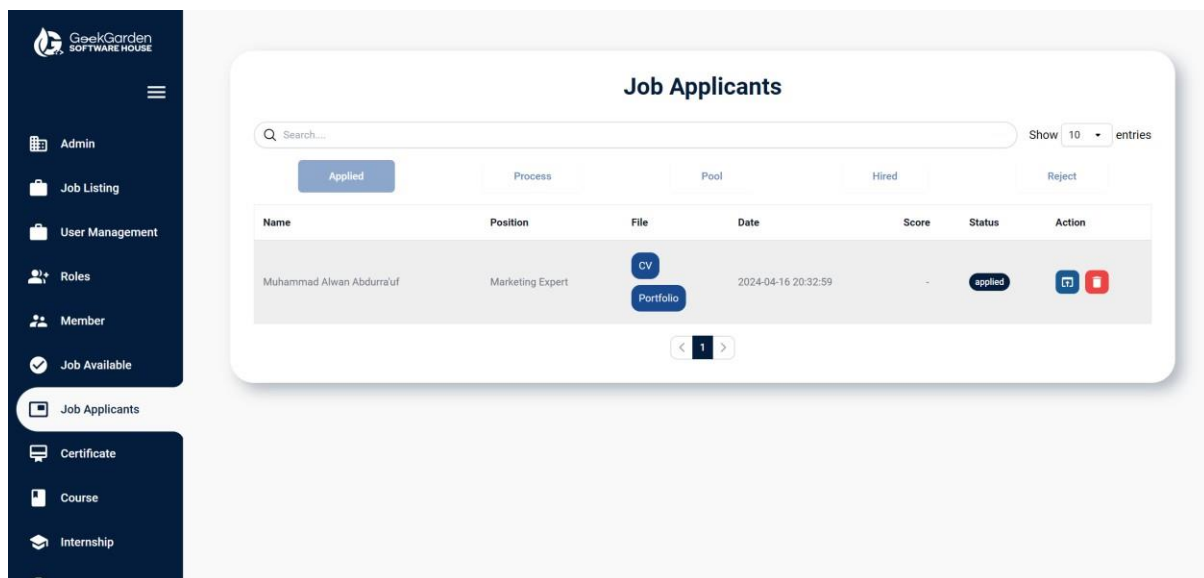
Job Available adalah fitur yang berfungsi untuk menampilkan daftar pekerjaan yang lowongannya telah dibuka oleh perusahaan. Dalam fitur ini, pengguna dapat melihat daftar pekerjaan yang tersedia dan mendapatkan detail informasi lengkap mengenai setiap lowongan tersebut. Fitur ini dapat diakses melalui *dashboard author* dan *admin*. Tampilan utama halaman fitur ini dapat dilihat pada gambar 3.20.



Gambar 3.20 Halaman fitur job available

h. Job Applicants

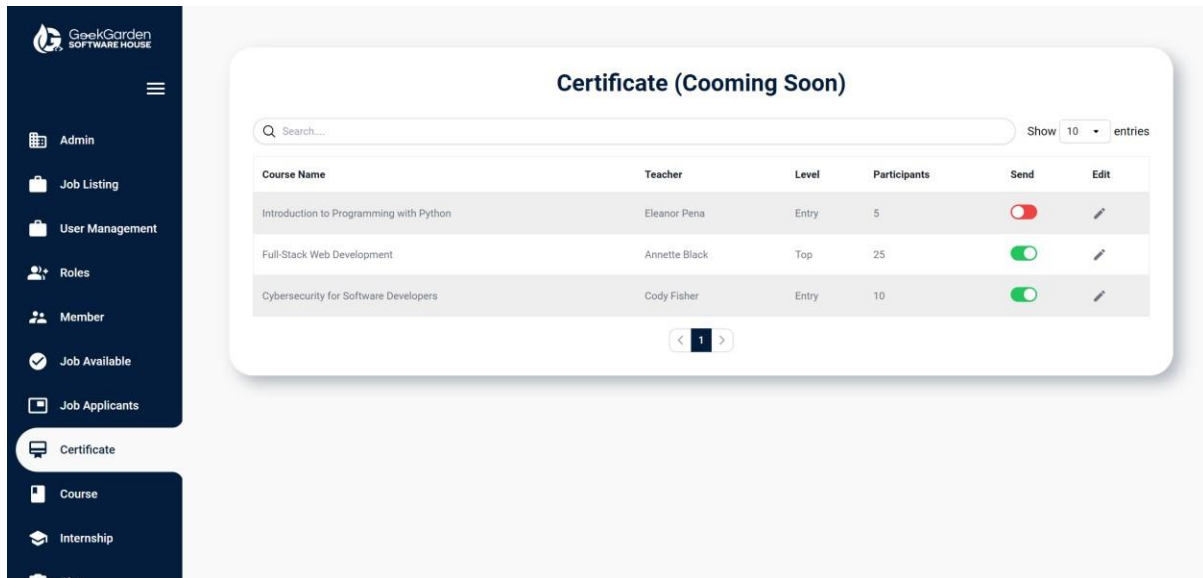
Job Applicants adalah sebuah fitur untuk mengelola semua pelamar pekerjaan pada lowongan pekerjaan yang dibuka oleh perusahaan. Dalam fitur ini, pengguna memiliki kemampuan untuk mengelola informasi lamaran, seperti melihat detail informasi setiap pelamar, menghapus lamaran, dan menentukan langkah-langkah selanjutnya dalam proses perekrutan. Fitur ini juga dilengkapi dengan filter untuk memudahkan pengguna dalam mengelompokkan lamaran berdasarkan tahap-tahap dalam proses penerimaan karyawan. Fitur ini dapat diakses melalui *dashboard author* dan *admin*. Tampilan utama halaman fitur ini dapat dilihat pada gambar 3.21.



Gambar 3.21 Halaman fitur job applicants

i. Certificate

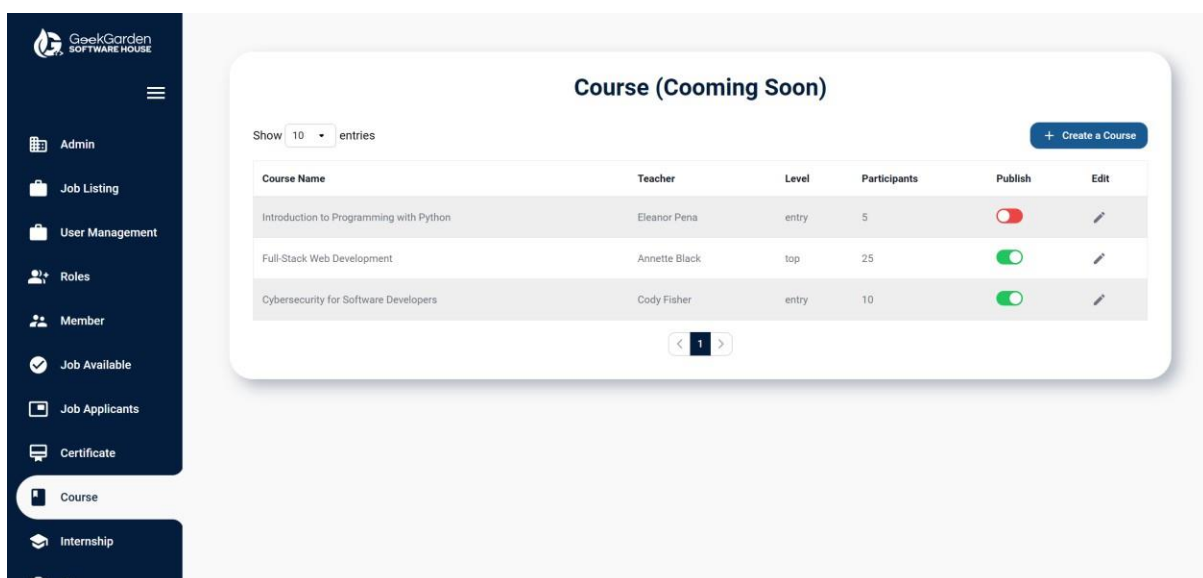
Certificate adalah sebuah fitur untuk mengelola sertifikat yang diterima dari kursus, webinar, dan kegiatan lainnya.. Dalam fitur ini, pengguna dapat mengirimkan sertifikat kegiatan dengan menekan tombol toggle "*send*" yang terdapat pada daftar kegiatan yang tersedia. Fitur ini memungkinkan pengguna untuk dengan mudah mengelola sertifikat kegiatan dan mendukung proses administratif terkait dengan pelatihan atau kegiatan lainnya. Fitur ini dapat diakses melalui *dashboard author* dan *admin*. Tampilan utama halaman fitur ini dapat dilihat pada gambar 3.22.



Gambar 3.22 Halaman fitur certificate

j. Course

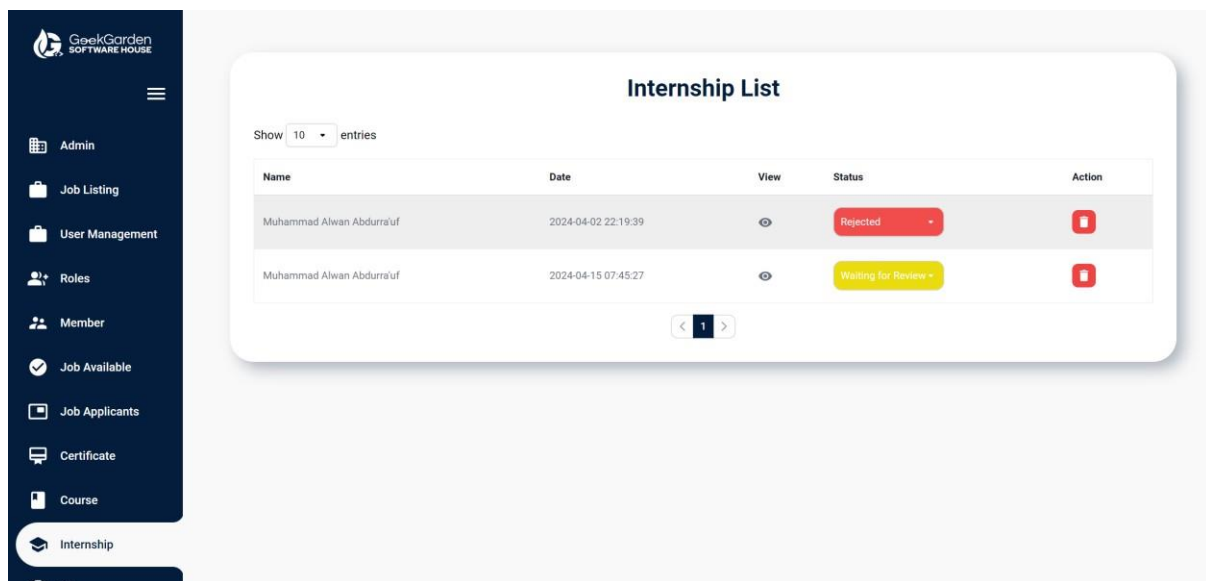
Course adalah sebuah fitur yang mengelola kursus, webinar, dan kegiatan lainnya untuk komunitas GeekGarden. Dalam fitur ini, pengguna memiliki keputusan untuk mempublikasikan atau tidak mempublikasikan kegiatan tersebut, yang memungkinkan mereka untuk mengatur visibilitas dan ketersediaan kursus atau webinar yang dikelola. Fitur ini dapat diakses melalui *dashboard author* dan *admin*. Tampilan utama halaman fitur ini dapat dilihat pada gambar 3.23.



Gambar 3.23 Halaman fitur course

k. Internship

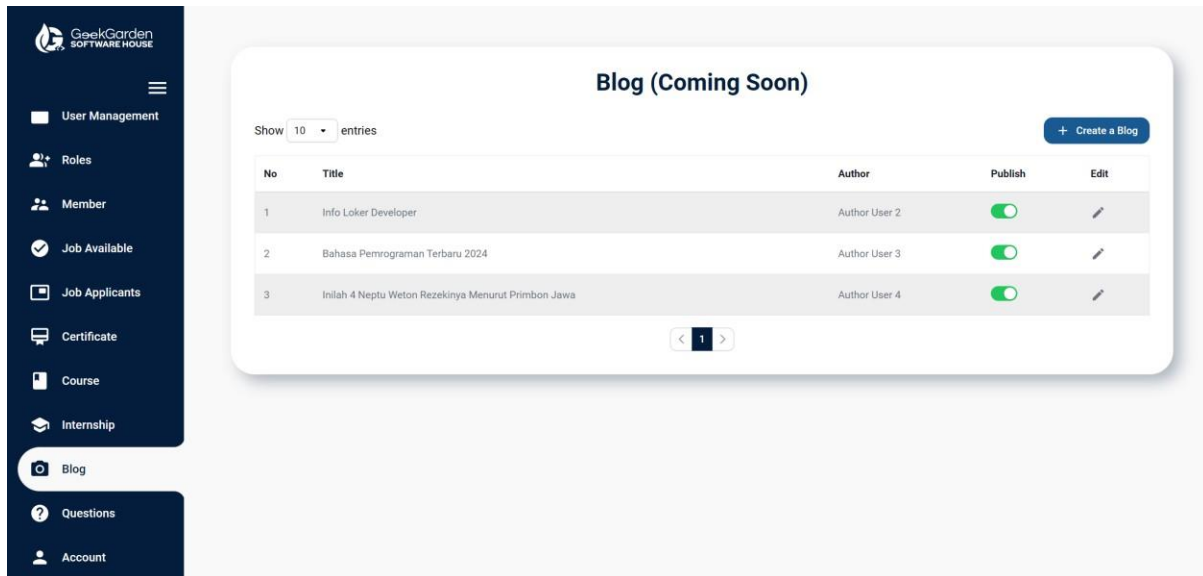
Internship adalah sebuah fitur yang mengelola seluruh pendaftar magang di perusahaan. Di dalam fitur ini, pengguna dapat melihat daftar pendaftar, detail informasi pendaftar, menghapus entri, dan menentukan langkah-langkah selanjutnya dalam proses perekrutan. Fitur ini dapat diakses melalui *dashboard author* dan *admin*. Tampilan utama halaman fitur ini dapat dilihat pada gambar 3.24.



Gambar 3.24 Halaman fitur internship

l. Blog

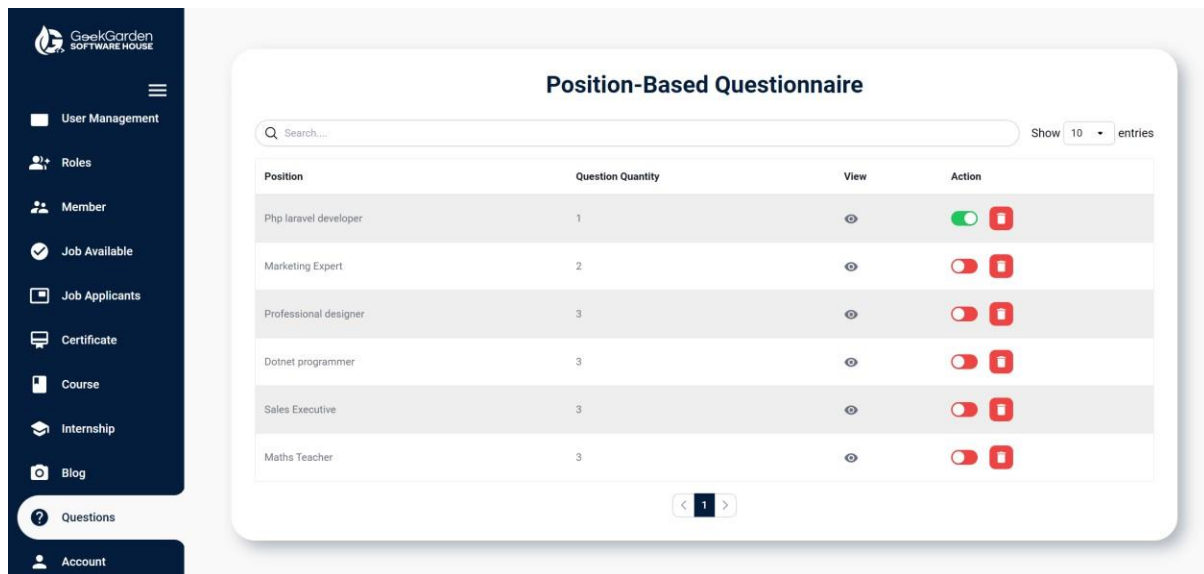
Blog adalah sebuah fitur untuk mengelola postingan artikel untuk disajikan kepada pengunjung web. Di dalam fitur ini, pengguna dapat membuat, mengubah, dan menghapus postingan artikel. Selain itu, pengguna memiliki keputusan untuk mempublikasikan atau tidak mempublikasikan artikel tersebut, memberikan kontrol penuh atas konten yang akan ditampilkan kepada pengunjung situs perusahaan. Fitur ini dapat diakses melalui *dashboard author* dan *admin*. Tampilan utama halaman fitur ini dapat dilihat pada gambar 3.25.



Gambar 3.25 Halaman utama fitur blog

m. Question

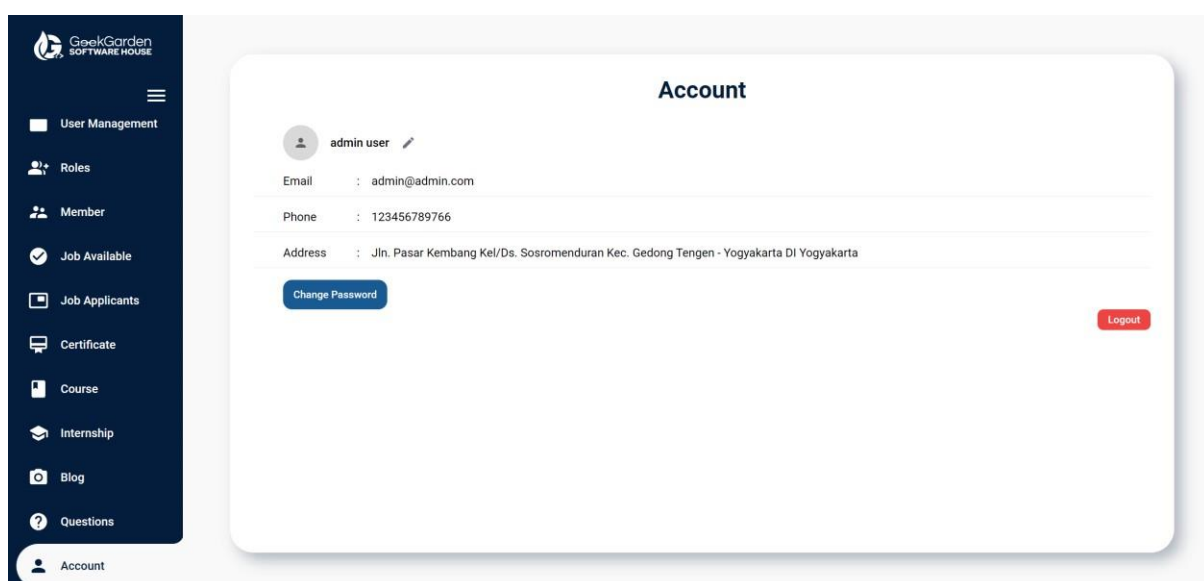
Question adalah sebuah fitur yang difungsikan untuk mengelola pertanyaan-pertanyaan yang akan ditampilkan pada formulir pendaftaran pekerjaan. Di dalam fitur ini, pengguna dapat melihat daftar pertanyaan pada setiap lowongan yang dibuka, serta menambah, mengubah, dan menghapus pertanyaan. Selain itu, pengguna memiliki keputusan untuk membuat pertanyaan-pertanyaan tersebut tampil di formulir pendaftaran pekerjaan atau tidak. Fitur ini dapat diakses melalui *dashboard author* dan *admin*. Tampilan utama halaman fitur ini dapat dilihat pada gambar 3.26.



Gambar 3.26 Halaman utama fitur question

n. Account

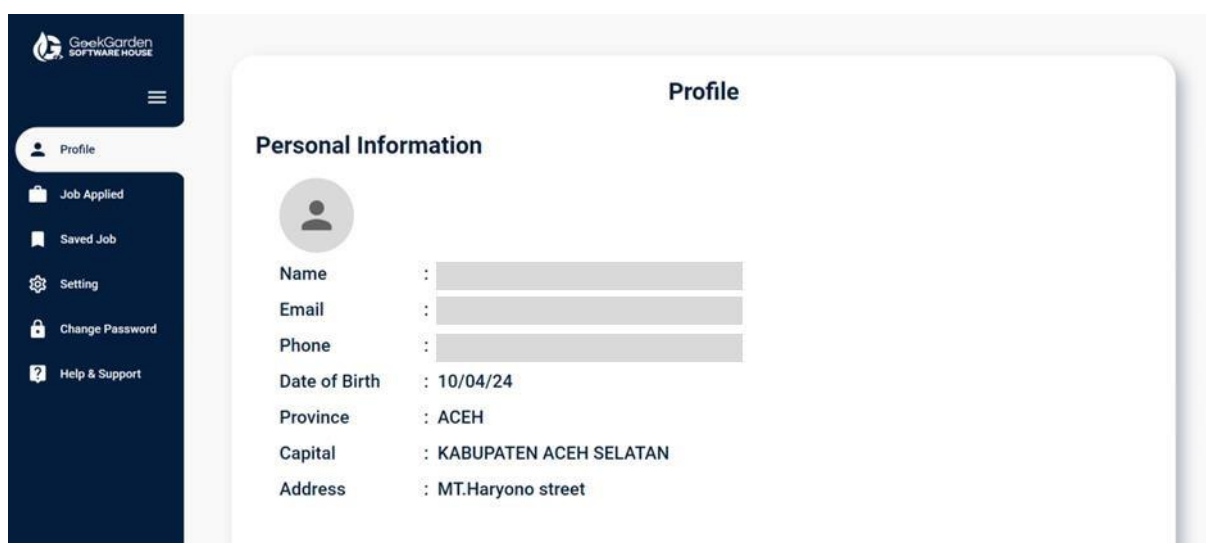
Account adalah sebuah fitur yang digunakan untuk mengelola data diri akun pengguna. Di dalam fitur ini, pengguna dapat melihat, mengubah data diri, dan juga mengubah *password* akun mereka. Selain itu, fitur ini menyediakan opsi untuk *logout*, sehingga pengguna dapat keluar dari dashboard dengan mudah. Tampilan utama halaman fitur ini dapat dilihat pada gambar 3.27.



Gambar 3.27 Halaman utama fitur account

o. Profile

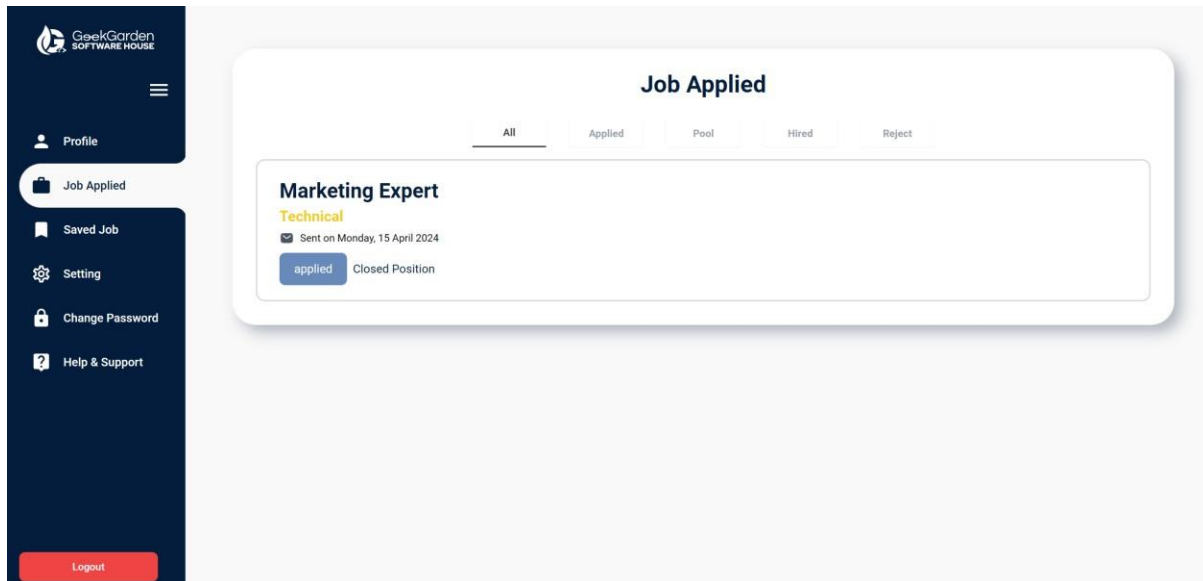
Profile adalah fitur atau halaman yang pertama kali diakses setelah login sebagai pengguna *member* di *dashboard member*. Fitur ini dirancang untuk menampilkan data diri pengguna secara lengkap, yang sangat dibutuhkan oleh perusahaan untuk keperluan administratif dan komunikasi. Pada fitur ini, pengguna dapat melihat informasi pribadi mereka, seperti riwayat pendidikan, pengalaman kerja, sertifikasi, dan informasi kontak lainnya. Tampilan halaman fitur ini dapat dilihat pada gambar 3.28.



Gambar 3.28 Halaman fitur profile

p. Job Applied

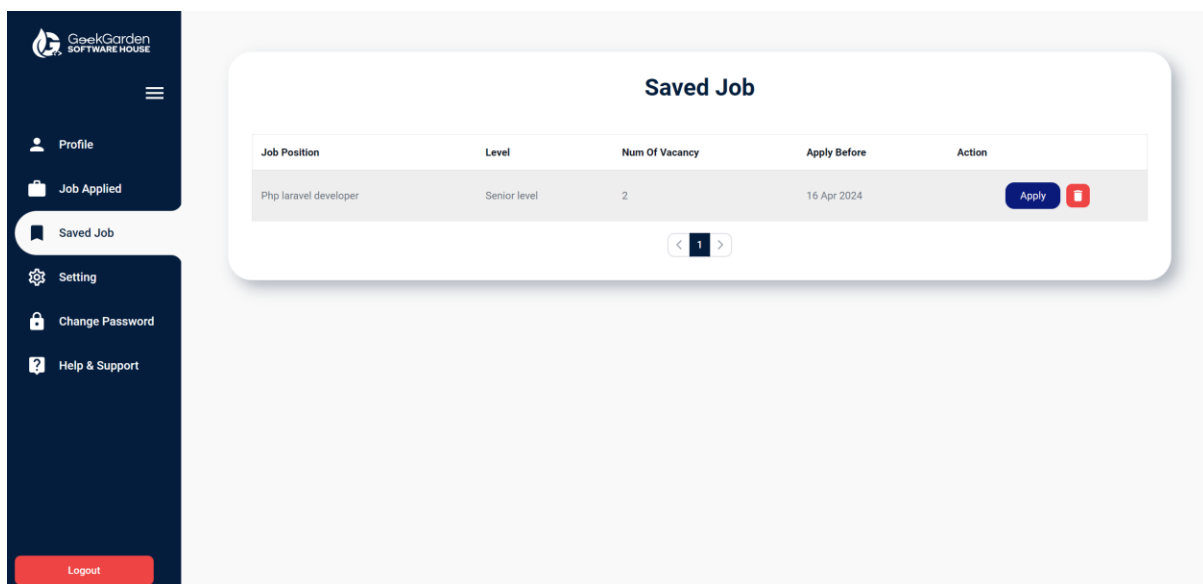
Job Applied adalah sebuah fitur yang akan menampilkan semua lowongan pekerjaan yang telah di-*apply* pengguna terkait di Geekgarden. Data yang ditampilkan meliputi nama pekerjaan, waktu pengiriman lamaran, status apakah lowongan masih dibuka atau tidak, serta status langkah dari lowongan yang diikuti oleh pengguna. Fitur ini dilengkapi dengan *filter* untuk memudahkan pengguna dalam mengelompokkan lamarannya berdasarkan tahap lamaran yang sedang dijalani. Tampilan halaman fitur ini dapat dilihat pada gambar 3.29.



Gambar 3.29 Halaman fitur job applied

q. Saved Job

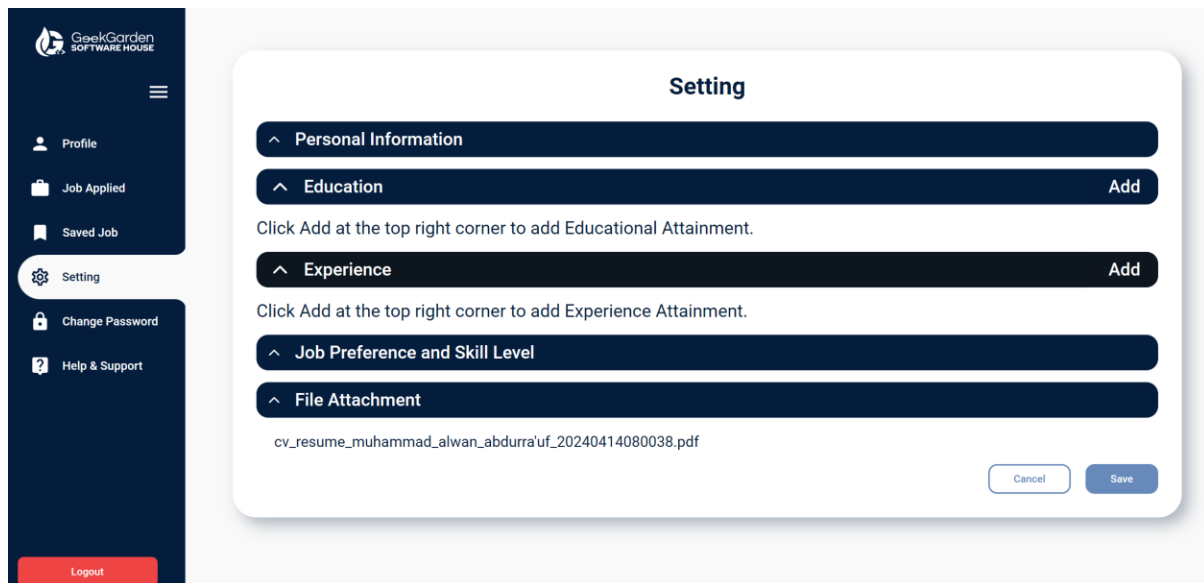
Saved Job adalah sebuah fitur yang menampilkan pekerjaan yang telah ditandai sebagai *bookmark* oleh pengguna sebagai pengingat. Fitur ini memudahkan pengguna untuk membuka formulir pendaftaran lowongan pekerjaan dari daftar pekerjaan yang telah ditandai dengan menekan tombol '*Apply*' pada pekerjaan yang diminati. Tampilan halaman fitur ini dapat dilihat pada gambar 3.30.



Gambar 3.30 Halaman fitur saved job

r. Setting

Setting merupakan sebuah fitur yang memungkinkan pengguna untuk mengubah maupun menambah data diri mereka sesuai kebutuhan. Dengan adanya fitur ini, pengguna dapat dengan mudah mengelola informasi profil mereka untuk memastikan keakuratan dan kelengkapan data yang terkait dengan akun mereka. Tampilan halaman fitur ini dapat dilihat pada gambar 3.31.



Gambar 3.31 Halaman fitur setting

s. Change Password

Change Password merupakan sebuah fitur yang digunakan pengguna untuk mengganti *password* dari akun mereka. Fitur ini memberikan pengguna kendali penuh atas keamanan akun mereka dengan memberi mereka kemampuan untuk secara berkala mengubah *password* sesuai dengan kebutuhan keamanan dan privasi yang mereka inginkan. Dengan fitur ini, pengguna dapat memastikan bahwa akun mereka tetap aman dan terlindungi dari akses yang tidak sah. Tampilan halaman fitur ini dapat dilihat pada gambar 3.32.

Gambar 3.32 Halaman fitur change password

t. Help & Support

Help & Support adalah sebuah fitur yang menyediakan berbagai informasi dan bantuan kepada pelamar pekerjaan. Di dalamnya, terdapat daftar pertanyaan umum (FAQ) yang sering diajukan oleh para pelamar. Selain itu, fitur ini juga menyediakan formulir untuk memungkinkan pelamar atau anggota komunitas untuk mengajukan pertanyaan lebih lanjut melalui *email*, memastikan bahwa mereka mendapatkan bantuan yang sesuai dengan kebutuhan mereka. Tampilan pertanyaan umum (FAQ) dapat dilihat pada gambar 3.33 dan Tampilan formulir *support* dapat dilihat pada gambar 3.34.

Gambar 3.33 Tampilan fitur help & support

Support

Name

Email

jendralbiribiri@gmail.com

Question

Send

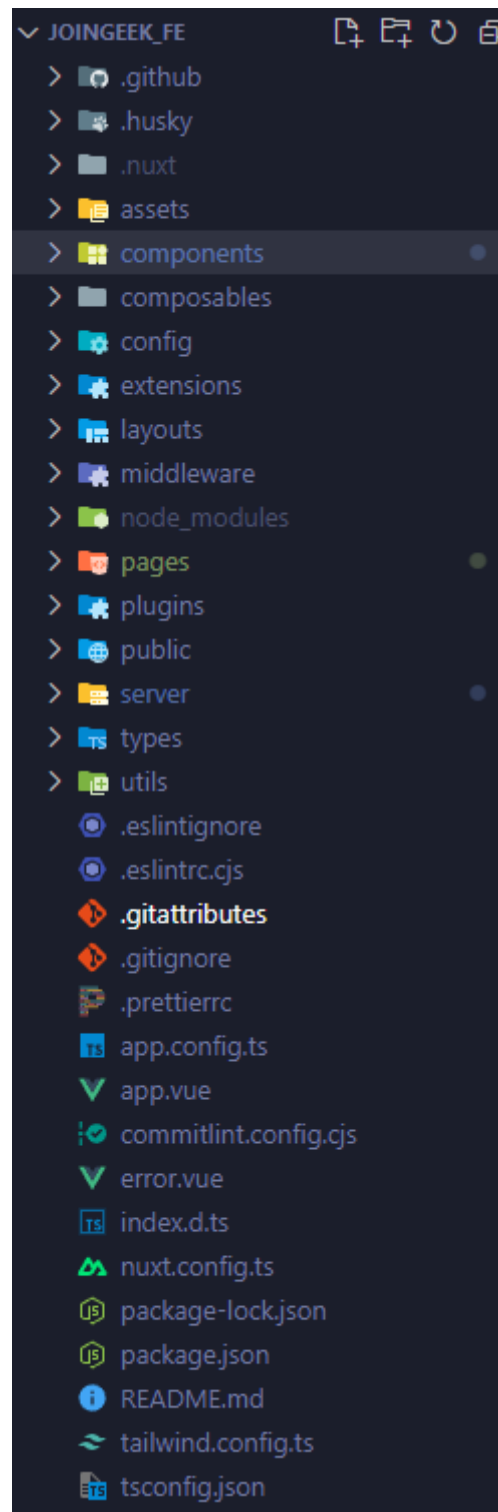
Gambar 3.34 Tampilan fitur kirim email support

3.2.3. Pengerjaan Proyek

Hal-hal yang telah dikerjakan sebelumnya

a. Pembuatan Struktur folder

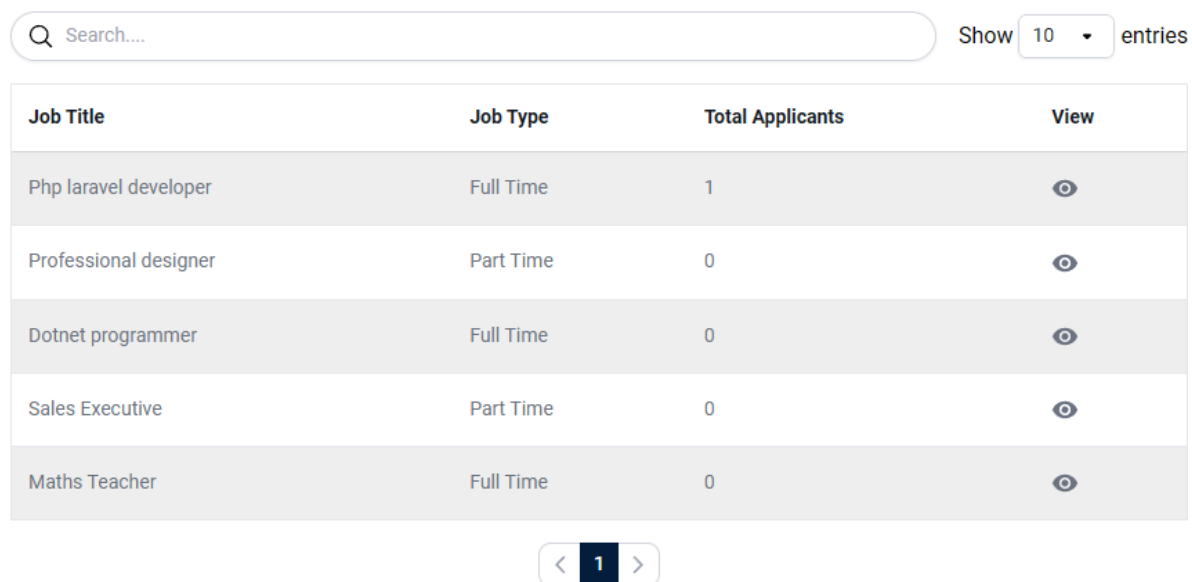
Proyek ini memiliki struktur folder yang mirip dengan struktur folder proyek nuxt pada umumnya. Namun, perbedaannya terletak pada penggunaan server yang dibuat sendiri dan tidak menggunakan server *backend* bawaan milik nuxt. Sebagai gantinya, server *backend* dibuat dengan teknologi H3. Struktur folder untuk server *backend* ini akan serupa dengan server pada proyek yang menggunakan Express.js. Dalam implementasinya, *frontend* akan melakukan permintaan data kepada server yang telah dibuat, kemudian server akan mengirimkan permintaan tersebut ke server eksternal yang dibuat dengan Laravel yang terhubung dengan *database*. Struktur folder proyek dapat dilihat pada gambar 3.35.



Gambar 3.35 Struktur folder proyek JoinGeek

b. Pembuatan komponen tabel

Tabel merupakan komponen yang digunakan hampir pada setiap fitur di setiap *dashboard*. Tabel yang digunakan memiliki tampilan yang seragam dengan beberapa fitur tambahan. Tabel ini dibuat dengan *header* yang dapat disesuaikan tergantung pada data yang akan ditampilkan. Terdapat juga fitur *fetch* data yang menampilkan *loading* secara *default* saat mengambil data dari *database*. Selain itu, tabel ini dilengkapi dengan fitur tambahan seperti tombol untuk mengganti halaman, pencarian, filter, dan pilihan untuk menampilkan jumlah entri. Tabel ini juga dapat dikustomisasi untuk menambahkan fitur lainnya yang diperlukan. Contoh tampilan implementasi tabel dapat dilihat pada Gambar 3.36.



Job Title	Job Type	Total Applicants	View
Php laravel developer	Full Time	1	
Professional designer	Part Time	0	
Dotnet programmer	Full Time	0	
Sales Executive	Part Time	0	
Maths Teacher	Full Time	0	

< 1 >

Gambar 3.36 Tampilan implementasi tabel

c. Pembuatan *utility function*

Beberapa fungsi telah dibuat sebelumnya sebelum pengerjaan fitur-fitur dilakukan. fungsi-fungsi ini dibuat untuk memudahkan pengerjaan proyek karena penggunaan fungsi-fungsi ini sangat repetitif digunakan. Fungsi-fungsi tersebut dibuat dari awal maupun memanfaatkan *library* bawaan vue dan nuxt maupun *library* lainnya. Adapun fungsi-fungsi tersebut memiliki banyak jenis dengan kegunaan yang berbeda seperti memformat tanggal, menyimpan data pengguna yang telah *login*, *timeout*, dan lain sebagainya.

```

/**
 * Time out composable
 * @param ms - number of milliseconds
 * @param callback - Callback function
 * @returns
 */
export function useTimeoutPromise(ms: number = 500, callback: () => void =
() => {}) {
  return new Promise((resolve) => {
    setTimeout(() => {
      callback();
      resolve(true);
    }, ms)
  });
}

```

Gambar 3.37 Fungsi timeout

Gambar 3.37 menunjukkan salah satu fungsi yang telah dibuat. Fungsi ini digunakan pada fungsi untuk mendapatkan data dari *database* ketika endpoint api belum dibuat. Fungsi tersebut memanfaatkan promise sehingga dapat mensimulasikan keadaan loading, mirip seperti saat menunggu akses data dari *database*. Dengan demikian, tampilan loading dapat ditampilkan meskipun api belum dibuat. Fungsi ini, `useTimeoutPromise`, menerima dua parameter: `ms`, yang menentukan jumlah milidetik untuk menunggu, dan `callback`, yang merupakan fungsi yang akan dijalankan setelah waktu tunggu berakhir. Fungsi ini mengembalikan sebuah *promise* yang akan diselesaikan setelah jangka waktu yang ditentukan, sehingga memungkinkan untuk menyimulasikan penundaan yang terjadi saat mengakses data dari *database*.

d. Pembuatan fitur login bagi pengguna author dan admin

Fitur *login* ini digunakan oleh pihak GeekGarden, yaitu pengguna *author* dan admin. Pengguna-pengguna tersebut berbeda dengan pengguna *member* karena mereka tidak dapat mendaftarkan sendiri. pendaftaran mereka dilakukan oleh pengembang melalui *database*, yang mana kedepannya, pengguna admin juga akan bisa mendaftarkan pengguna baru untuk pengguna *author* dan admin. halaman *login* ini juga tidak dapat diakses melalui halaman utama akan tetapi pengguna harus mengetik `/login` secara manual di pencarian. Tampilan dari fitur ini dapat dilihat pada Gambar 3.38.



Gambar 3.38 Fitur login untuk pengguna author dan admin

e. Alur untuk mendapatkan data dari *database*

Seperti yang dijelaskan sebelumnya, proyek ini memiliki server *backend* sendiri. Hal ini bertujuan untuk memberikan keamanan data melalui URL API server *backend* yang terpisah dan menghindari tereksposnya *endpoint url server eksternal* di browser yang digunakan, serta untuk memanfaatkan server agar proses *rendering* lebih cepat. Alur pengambilan data dari *database* hingga ditampilkan di *frontend*, pemagang akan mengambil contoh pada fitur registrasi pengguna.

```

import type { H3Event } from 'h3';
import { z } from 'zod';

interface Register {
  fullname: string;
  email: string;
  password: string;
  password_confirm: string;
}

async function register(event: H3Event) {
  try {
    const validations: Record<keyof Register, any> = {
      fullname: z.string().trim().min(1, { message: 'Field is required' }),
      email: z.string().trim().min(1, 'Email is required'),
      password: z.string().trim().min(1, { message: 'Field is required' }),
      password_confirm: z
        .string()
        .trim()
        .min(1, { message: 'Field is required' })
        .refine((value) => value !== '', { message: 'Field is required' })
    };

    const { data: body } = await useValidateBody<Register>(event, { schema:
z.object(validations) });
    const result = await doRequest({
      path: '/api/register',
      method: 'POST',
      body,
      event
    });

    return {
      success: result.success,
      message: result.message
    };
  } catch (err: any) {
    throw createError(err);
  }
}

export default {
  register: defineEventHandler(register),
};

```

Gambar 3.39 Controller register user

Pada Gambar 3.39 diperlihatkan kode untuk pembuatan *controller* di server. *Controller* tersebut berisi fungsi-fungsi yang berguna untuk mengambil data dari *endpoint* yang disediakan oleh *server eksternal* yang terhubung langsung ke *database*. Hal pertama yang harus dilakukan adalah mengimpor *library* yang akan digunakan. *Library* wajib yang harus digunakan adalah H3, yang kita gunakan dalam pembuatan server. Lalu dibuatlah sebuah fungsi yang menerima parameter berupa H3 bernama *Register*.

Dalam fungsi register tersebut, dilakukan validasi data yang akan dikirimkan dengan menggunakan *library* Zod yaitu data pengguna yang ingin mendaftar. Ini memastikan bahwa format data yang dikirimkan ke server eksternal valid dan akan mengirimkan pesan *error* jika data yang dikirimkan tidak valid. Validasi data ini tidak selalu digunakan pada setiap permintaan, tergantung pada *http method* yang digunakan apakah memerlukan pengiriman data atau tidak.

Langkah selanjutnya adalah melakukan *http request* ke *server eksternal* dengan menggunakan fungsi *doRequest* yang telah dibuat sebelumnya. Fungsi ini menerima beberapa parameter seperti path yang berisi *endpoint* dari *server eksternal*, method yang menentukan metode yang digunakan dalam *http request*, body yang berisi data yang dikirimkan ke *server*, *query* yang berisi parameter kueri pada *endpoint*, dan beberapa parameter lainnya. Fungsi ini juga mengembalikan parameter seperti pesan dan juga data (jika ada). Kemudian function ini akan *diexport* agar bisa digunakan untuk membuat route.

```
user from '../controller/user';

const router = createRouter();
const h = (...handlers: EventHandler[]) => createApp().use(handlers).handler

router.post('/register', auth.register);

export default useBase('/api', router.handler);
```

Gambar 3.40 Route register user

Gambar 3.40 diperlihatkan kode untuk *route handlers* di server. Fitur route ini merupakan kumpulan route yang telah dibuat. Fitur ini berfungsi untuk menginisiasi route berdasarkan *controller* yang telah dibuat. Cara mendeklarasikan route baru adalah dengan menggunakan `route.post()`. Kata "post" dapat diganti dengan *http method* yang kita pilih sesuai dengan kebutuhan. Route tersebut menerima tiga parameter, parameter pertama berupa *endpoint* baru yang dibuat dan harus bersifat unik, parameter kedua berupa jenis pengguna yang bisa mengakses *endpoint* tersebut yang dituliskan dalam *array* dan saat ini tidak digunakan karena *endpoint* ini dapat digunakan seluruh pengguna, dan parameter terakhir berupa controller yang telah diimpor ke halaman ini.

```

async function onSubmitRegister() {
  try {
    loading.open();
    await useRequest('/register', {
      method: 'POST',
      body: formModelRegister.value
    });

    formModelRegister.value = {
      fullname: '',
      email: '',
      password: '',
      password_confirm: ''
    };
    loading.close();
    dialog.success({ message: 'Register Success' });
    modalType.value = 'login';
  } catch (err: unknown) {
    useRequestError(err, formRegisterRef);
  } finally {
    loading.close();
  }
}

```

Gambar 3.41 Fungsi untuk register user dari frontend

Gambar 3.41 memperlihatkan kode untuk *fetch* data pada *frontend*. Fungsi `onSubmitRegister` bertanggung jawab untuk menangani proses *submit* saat pengguna mencoba untuk melakukan registrasi. Fungsi yang akan dipakai pada setiap *request* ke server adalah fungsi `useRequest`. Sama seperti `doRequest`, Fungsi ini adalah fungsi yang telah dibuat sebelumnya untuk melakukan HTTP request. Perbedaannya adalah fungsi ini ditujukan untuk server internal yang telah dibuat.

Fungsi `useRequest` menerima parameter, seperti metode HTTP yang akan digunakan (misalnya POST), dan *endpoint* yang sesuai dengan rute yang telah didefinisikan di pengaturan rute internal server, yakni `'/register'`. Selain itu, fungsi juga menerima data yang akan dikirimkan, yang merupakan data formulir registrasi dari pengguna. Hasil dari permintaan ini akan sesuai dengan apa yang dikembalikan oleh *controller* yang terhubung dengan rute tersebut di server.

f. Pembuatan front office

Halaman ataupun fitur pada front office yang telah dibuat seperti landing page, dan *form apply job*, dan *form apply intership*. Dan fitur-fitur lainnya. Pengerjaan setiap halaman dan fitur tersebut melibatkan beberapa tahap penting: *slicing* UI, memastikan desain responsif, dan integrasi dengan API. Pengerjaan ini memastikan implementasi fitur-fitur yang telah direncanakan sebelumnya, meskipun masih dalam tahap dasar. Perbaikan dan penyempurnaan fitur-fitur ini akan dilakukan seiring berjalannya proyek.

g. Slicing sidebar dashboard

Sidebar merupakan sebuah komponen UI yang akan digunakan pada setiap *dashboard*, baik *dashboard member*, *author*, maupun admin. Komponen ini terdiri dari logo GeekGarden yang berfungsi sebagai tombol untuk mengarahkan pengguna ke halaman utama, daftar menu dari *dashboard* tersebut, serta ikon hamburger untuk membuka atau menutup *sidebar*. *Sidebar* ini juga dilengkapi dengan tombol *logout* yang responsif dan hanya muncul pada *dashboard member*.

```
<template>
  <div
    class="fixed flex flex-col top-0 bottom-0 z-20 min-h-screen bg-main w-
full max-w-[250px] pl-6 pt-8 transition-all"
    :class="isOpen ? 'left-0' : '-left-[180px]'"
  >

    <NuxtLink class="flex items-center w-full space-x-2 mx-6 text-white mb-
4" to="/">
      <div v-html-safe="LogoGeekGarden" class="w-10 h-10" :class="['isOpen ?
'ml-0' : 'ml-36']"></div>
      <div v-html-safe="LogoCaptionGeekGarden" class="w-28" :class="['isOpen ?
'' : 'hidden']"></div>
    </NuxtLink>

    <div class="flex w-full pr-5 py-4">
      <AppIcon
        name="ic:baseline-menu"
        class="!w-8 !h-8 text-white ml-auto cursor-pointer"
        @click="onToggleMenu"
      ></AppIcon>
    </div>

    <div
      class="space-y-2 py-2 h-[calc(100vh_-_8rem)] overflow-x-hidden
overflow-y-hidden hover:overflow-y-auto scroll-mini flex flex-col justify-
between pb-4"
    >

      <div>
        <div v-for="(item, key) in sidebar.data" :key="`menu-${key}`">
```

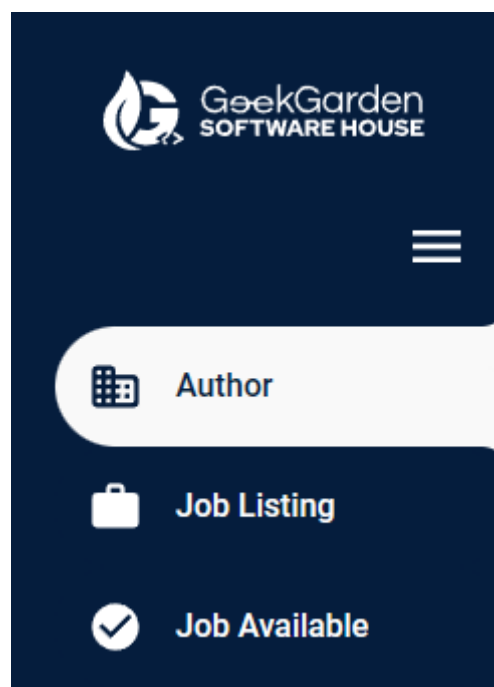
```

        <NuxtLink
          :to="item.path"
          class="flex items-center space-x-4 nav-menu p-4"
          :class="[activeClass(item.pages), ...[isOpen ? 'ml-0' : 'ml-
40']]"]
        >
          <AppIcon :name="item.icon" class="!w-7 !h-7"></AppIcon>
          <span class="font-medium" :class="isOpen ? '' : 'hidden'">{{
item.label }}</span>
        </NuxtLink>
      </div>
    </div>
    <UButton
      v-if="isLogout && isOpen"
      color="red"
      class="rounded-lg w-10/12 text-center mr-3 mb-8 flex justify-center"
      size="lg"
      @click="
        dialog.delete({
          message: 'Are you sure, you want to log out???' ,
          buttonTextSubmit: 'Yes, Logout',
          onSubmit: onLogout
        })
      "
    >
      <span class="font-medium" :class="isOpen ? '' :
'hidden'">Logout</span>
    </UButton>
    <UButton
      v-if="isLogout && !isOpen"
      color="red"
      class="rounded-lg text-center mr-3 mb-8 flex justify-center justify-
self-end self-end"
      size="lg"
      @click="
        dialog.delete({
          message: 'Are you sure, you want to log out???' ,
          buttonTextSubmit: 'Yes, Logout',
          onSubmit: onLogout
        })
      "
    >
      <UIcon name="i-heroicons-power-16-solid" />
    </UButton>
  </div>
</div>
</template>

```

Gambar 3.42 Kode program pembuatan tampilan sidebar

Pada Gambar 3.42 ditampilkan kode program untuk pembuatan tampilan *sidebar*. Pada kode tersebut, banyak diimplementasikan fungsi-fungsi yang berguna seperti membuat tampilan *sidebar* yang responsif, baik dalam keadaan terbuka maupun tertutup, sesuai dengan ukuran layar perangkat. Selain itu, kode ini juga mengatur agar saat ikon hamburger ditekan, *sidebar* akan terbuka atau tertutup. Terdapat juga fungsi-fungsi untuk mengarahkan ke menu tertentu saat salah satu tombol menu ditekan, serta membuat warna menu aktif berbeda dengan menu yang tidak aktif, dan lain sebagainya. Tampilan *sidebar* dapat dilihat pada Gambar 3.43 dan Gambar 3.44.



Gambar 3.43 Sidebar salah satu dashboard dalam posisi terbuka



Gambar 3.44 Sidebard salah satu dashboard dalam posisi tertutup

h. Pengerjaan dashboard author

Dashboard author yang telah dikerjakan menampilkan daftar menu yang dapat diakses pada halaman tersebut melalui *sidebar*, seperti yang telah dijelaskan sebelumnya. Selain itu, seluruh menu dan fitur yang ada telah melalui proses *slicing* UI. Pengerjaan juga memastikan *dashboard* yang dibuat telah responsif.

Selain melakukan *slicing* seluruh UI pada *dashboard author*, hal lain yang dikerjakan sebelumnya adalah membuat API untuk fitur author dan mengintegrasikannya dengan API tersebut. Langkah ini dilakukan terlebih dahulu karena halaman ini merupakan halaman pertama yang diakses ketika pengguna *author* berhasil *login*. Alasan lainnya adalah untuk menyediakan contoh cara pengerjaan sebuah fitur sehingga penulis dapat mencontohnya dan mengaplikasikannya pada fitur lainnya.

Transfer Pengetahuan

Pengembangan *frontend* aplikasi ini awalnya dilakukan oleh mentor penulis sebagai *frontend* developer, sebelum kemudian dialihkan kepada penulis untuk melanjutkan pengerjaannya. Peralihan ini terjadi karena mentor penulis harus memprioritaskan beberapa proyek aplikasi lain pada waktu tersebut. Pada tahap transisi ini, berbagai langkah diambil untuk memastikan penulis dapat melanjutkan proyek tersebut dengan lancar.

Namun, proyek ini belum memiliki dokumentasi terkait penulisan kode programnya. Oleh karena itu, sebagai persiapan sebelum bergabung dalam tim, penulis mempelajari teknologi utama yang digunakan dalam pengembangan *frontend* aplikasi tersebut, yaitu Nuxt, selama satu bulan sembari mengerjakan proyek lain. Setelah itu, penulis menerima kode dari proyek tersebut dan segera mempelajari *library-library* pendukung yang digunakan.

Setelah resmi bergabung sebagai anggota tim pengembang, penulis menerapkan pendekatan "*learning by doing*" dengan dasar pembelajaran berupa teknologi Nuxt dan *library-library* pendukung yang telah dipelajari sebelumnya. Sebagaimana telah dijelaskan dalam pembahasan sebelumnya, mentor telah menyelesaikan beberapa pekerjaan dan mempersiapkan proyek sebelum penulis mengambil alih. Mentor hanya memberikan penjelasan mengenai alur dan aturan penulisan kode program tersebut dan penulis mempelajari lebih dalam penulisan kode sambil mengerjakan tugas yang diberikan. Dalam melanjutkan pengerjaan proyek, penulis memanfaatkan kode yang telah disediakan oleh mentor untuk mengembangkan fitur-fitur baru, dengan bimbingan dari mentor hingga penulis memahami seluruh proses pengerjaan proyek tersebut.

Pengerjaan Sprint

Sprint pertama

- a. Pembuatan API serta mengintegrasikannya dengan fitur Internship di *dashboard author*

Fitur ini hanya memiliki satu halaman yang berisi daftar pelamar magang. Di halaman ini, pemagang membuat empat API yang terdiri dari pembuatan *controller* dan *route* baru. *Controller* dan *route* yang dibuat berfungsi untuk menampilkan daftar pelamar magang, menampilkan detail informasi tiap pelamar, mengubah status lamaran, dan menghapus lamaran. Pembuatan API ini mirip dengan yang telah dibuat sebelumnya, sehingga penulis hanya perlu sedikit menyesuaikan sesuai kebutuhan.

Integrasi API yang telah dibuat juga kurang lebih sama dengan fitur lain yang telah dikerjakan. Integrasi api memanfaatkan komponen yang telah dibuat seperti tabel, *pop up delete*, *pop up detail* yang hanya memerlukan perubahan sesuai kebutuhan. Tidak ada kendala yang dihadapi ketika mengerjakan fitur ini, akan tetapi terdapat satu hal yang perlu dibuat dari awal yaitu *dropdown* untuk mengubah status dikarenakan fitur yang telah dibuat belum ada yang mengimplementasikannya. Kode program dari pembuatan *dropdown* tersebut dapat dilihat pada gambar 3.45.

```
<template #status-data="{ row }">
  <div class="dropdown-container relative w-[120px] lg:w-fit">
    <select
      v-model="row.status"
      :class="classes.selectMenu"
      :style="{ background: getStatus(row.status)?.color }"
      @update:model-value="onSelectDropdown(row.id, row.status)"
    >
      <option
        v-for="(item, k) in statusOptions"
        :key="`status-${k}`"
        :value="item.id"
        :hidden="item.id == 'waiting for review'"
      >
        {{ item.name }}
      </option>
    </select>
    <AppIcon
      name="ic:round-arrow-drop-down"
      class="absolute right-2 top-2.5 text-white w-5 h-5"
    ></AppIcon>
  </div>
</template>

<script setup lang="ts">
const statusOptions = ref([
  { id: 'waiting for review', name: 'Waiting for Review', color: '#EBDC0F' },
  { id: 'on process', name: 'On Process', color: '#29C949' },
  { id: 'reject', name: 'Rejected', color: '#F24D4D' }
]);
</script>
```

Gambar 3.45 Kode program pembuatan dropdown

Sprint Kedua

- a. Pembuatan API 'ubah status' serta mengintegrasikannya dengan fitur Job Listing di dashboard author

API ini berfungsi untuk mengubah status dari sebuah pekerjaan untuk dipublikasikan atau tidak. Pembuatan API dilakukan dengan menduplikasi dan mengubah *controller* serta *route* sesuai kebutuhan. Selanjutnya, penulis membuat komponen berupa toggle untuk integrasi ke API yang dibuat. Kode program pembuatan toggle dan integrasi API dapat dilihat pada gambar 3.46.

```
<template #publish-data="{ row }">
  <UToggle
    v-model="row.publish"
    color="green"
    :ui="{
      inactive: 'bg-red-500'
    }"
    @update:model-value="(val: any) => changeStatus(row.id, val)"
  ></UToggle>
</template>

<script setup lang="ts">
async function changeStatus(id: number, status: boolean) {
  try {
    const result = await useRequest(`/back/jobs/${id}/status`, {
      method: 'PUT',
      body: {
        status: status ? 'true' : 'false'
      }
    });
    toast.add({ title: result.message, color: 'green' });
  } catch (err: any) {
    useRequestError(err);
  }
}
</script>
```

Gambar 3.46 Kode program toggle ubah status

- b. Pembuatan API serta mengintegrasikannya dengan fitur Job Available di dashboard author

Fitur ini memiliki dua halaman, yaitu halaman utama untuk menampilkan daftar pekerjaan yang telah dipublikasikan beserta jumlah pelamar, dan halaman untuk melihat detail dari lowongan tersebut. Penulis membuat dua API yang melibatkan pembuatan *controller* dan *route* baru, yang akan diintegrasikan di kedua halaman tersebut. Pembuatan API ini mirip dengan yang telah dibuat sebelumnya, sehingga penulis hanya perlu menyesuaikan sedikit sesuai kebutuhan. Tidak ada kesulitan dalam pengerjaan fitur ini karena tidak ada perubahan tampilan, sehingga integrasi API mudah untuk diimplementasikan.

- c. Pembuatan API serta mengintegrasikannya dengan fitur Job Applicants di dashboard author

Fitur ini hanya memiliki satu halaman yang berfungsi untuk menampilkan data dari pelamar pekerjaan. Penulis membuat tiga API yang melibatkan pembuatan *controller* dan *route* baru, yang akan diintegrasikan di kedua halaman tersebut yang berfungsi untuk menampilkan pelamar pekerjaan, menghapus data pelamar, dan juga mengubah status pelamar sesuai step lamaran yang sedang dijalani. Pembuatan API ini mirip dengan yang telah dibuat sebelumnya, sehingga pemegang hanya perlu menyesuaikan sedikit sesuai kebutuhan.

Terdapat sedikit kustomisasi dalam tampilan tabel pada fitur ini, yaitu penambahan filter berupa daftar langkah-langkah (step) dari lamaran yang akan menampilkan daftar pelamar sesuai dengan langkah yang dipilih. Ada lima langkah dalam filter ini, dan saat memilih langkah 'process', akan muncul filter baru yaitu 'filter by status'. Kode program untuk implementasi tampilan filter tersebut dapat dilihat pada gambar 3.47.

```

<template>
  <template #top-left>
    <USelectMenu
      v-if="status == 'process'"
      v-model="label.category"
      :options="filterOptions"
      class="flex-grow"
      placeholder="Filter by status"
      :ui="{
        rounded: 'rounded-full'
      }"
      option-attribute="name"
      @update:model-value="onFilter"
    ></USelectMenu>
  </template>
  <template #under-top>
    <div class="flex justify-around items-center my-4">
      <UButton
        v-for="(item, k) in statusCategories"
        :key="`status-${k}`"
        class="px-[42px] py-3 bg-white text-[#678ABB] disabled:bg-[#678ABB]
disabled:text-white"
        :disabled="status === item.id"
        @click="onSelectStatus(item.id)"
      >
        {{ item.name }}
      </UButton>
    </div>
  </template>
</template>

<script setup lang="ts">
const filterOptions = ref([]);
const statusCategories = ref([]);

</script>

```

Gambar 3.47 Kode program filter by step

Sprint Ketiga

- a. Pembuatan API serta mengintegrasikannya dengan fitur Blog di dashboard author

Fitur ini memiliki tiga halaman, yaitu halaman utama untuk menampilkan daftar blog juga pembuat blog tersebut, serta halaman edit dan tambah blog untuk menambah atau mengubah pertanyaan, dengan *layout* yang sama. Semua *controller* API pada fitur ini ditulis dalam satu file agar memudahkan pencarian.

Halaman utama membutuhkan dua API yang masing-masing berfungsi untuk menampilkan daftar blog dan pembuat blog tersebut, dan mengubah status publikasi dari blog tersebut. Halaman tambah blog memiliki satu API dan halaman edit blog terdapat 3 API yang masing-masing berfungsi untuk menampilkan detail blog pada input form, hapus blog, dan juga edit blog. Halaman utama berhasil diintegrasikan ke UI yang telah dibuat, dan halaman yang lain telah juga diintegrasikan tetapi dibutuhkan perbaikan kembali kedepannya untuk menyesuaikan kebutuhan.

b. Pembuatan API serta mengintegrasikannya dengan fitur Question di dashboard author

Fitur ini memiliki empat halaman, yaitu halaman utama untuk menampilkan pekerjaan dan jumlah pertanyaan untuk lamaran, halaman detail untuk melihat daftar pertanyaan pada suatu lamaran pekerjaan, serta halaman edit dan tambah pertanyaan untuk menambah atau mengubah pertanyaan, dengan *layout* yang sama. Semua *controller* API pada fitur ini ditulis dalam satu file agar memudahkan pencarian.

Halaman utama membutuhkan tiga API yang masing-masing berfungsi untuk menampilkan daftar pekerjaan dan pertanyaan yang telah dibuat untuk lamaran pada posisi tersebut, menghapus seluruh pertanyaan pada posisi tersebut, dan mengubah status publikasi pertanyaan tersebut pada formulir lamaran. Halaman detail membutuhkan dua API yang masing-masing berfungsi untuk menampilkan daftar pertanyaan yang telah dibuat pada suatu lamaran pekerjaan dan menghapus pertanyaan tersebut.

Halaman tambah dan ubah pertanyaan membutuhkan dua API yang masing-masing berfungsi untuk menambah pertanyaan pada suatu lamaran pekerjaan dan untuk mengubah pertanyaan. Pembuatan API ini mirip dengan yang telah dibuat sebelumnya, sehingga penulis hanya perlu menyesuaikannya sedikit sesuai kebutuhan. Tantangan dalam pengerjaan fitur ini terletak pada penyesuaian tampilan pada halaman edit dan tambah pertanyaan agar sesuai dengan halaman yang dituju oleh pengguna.


```

<template>
  <BoxContainer
    :title="isDetail ? 'Edit Question' : `Question ${positionTitle}`"
    title-position="center"
    has-arrow-back
  >
    <UForm :state="formModel" class="space-y-4" @submit="onSubmit">
      <div v-for="(item, k) in formModel" :key="`question-${k}`">
        <UFormGroup label="Question" size="xl">
          <UTextarea v-model="item.question" placeholder="Question"
resize autoresize></UTextarea>
        </UFormGroup>
        <UFormGroup>
          <USelectMenu
            v-model="answerLabels[k].label"
            class="mt-4"
            placeholder="Select Answer Key"
            :options="answerOptions"
            option-attribute="name"
            @update:model-value="(val : any) =>
onSelectDropdownAnswer(val, k)"
          ></USelectMenu>
          <div
            v-if="!isDetail"
            class="flex justify-center w-8 h-8 rounded-lg mt-4 ml-auto"
            :class="
              isDisabledRemoveQuestion
                ? 'cursor-not-allowed bg-gray-400'
                : 'cursor-pointer bg-red-500 hover:bg-red-600'
            "
            @click="onRemoveQuestion(k)"
          >
            <AppIcon name="ic:baseline-delete" class="text-white w-5 h-5
self-center"></AppIcon>
          </div>
        </UFormGroup>
      </div>
      <div v-if="!isDetail" class="flex items-center space-x-4">
        <UButton
          color="light-blue"
          class="rounded-xl px-4"
          icon="ic:round-add-box"
          size="lg"
          @click="onAddQuestion"
        >
          Add Question
        </UButton>
        <UButton type="submit" color="green" class="rounded-xl px-4"
size="lg">Save</UButton>
      </div>
      <div v-else type="submit" color="light-blue" class="rounded-xl
px-4"> Update </UButton>
    </UForm>
  </BoxContainer>
</template>

<script setup lang="ts">
const isDetail = computed(() => route.params.question !== 'add');
</script>

```

Gambar 3.48 Kode program layout edit dan tambah question

Gambar 3.48 menunjukkan kode program untuk membedakan kondisi antara menampilkan halaman edit atau tambah pertanyaan. Dalam hal ini, *params* digunakan sebagai patokan. Jika *params* mengandung kata 'add', maka yang akan ditampilkan adalah halaman tambah pertanyaan; sebaliknya, halaman edit pertanyaan akan ditampilkan. Kondisi ini disimpan dalam variabel *isDetail* untuk memudahkan penggunaan secara kondisional, seperti menentukan nama halaman, menampilkan tombol, menerapkan fungsi, dan lain sebagainya. Pembuatan API ini mirip dengan yang telah dibuat sebelumnya, sehingga pemegang hanya perlu menyesuaikannya sedikit sesuai kebutuhan.

Sprint Keempat

- a. Pembuatan API serta mengintegrasikannya dengan fitur Question di apply pekerjaan

Pembuatan fitur ini dilakukan dengan membuat API yang berfungsi untuk menampilkan pertanyaan berdasarkan jenis pekerjaan yang dilamar. Hal yang dilakukan hanya menampilkan pertanyaan bila daftar pertanyaan berstatus *publish*. Hal yang dilakukan selanjutnya adalah mengumpulkan jawaban dari pertanyaan tersebut dalam sebuah variabel, mengubah nilainya menjadi *boolean*, lalu menambahkanannya dalam *body* untuk di-*submit*. Kode program untuk penjelasan tersebut dapat dilihat pada gambar 3.49.

```

async function onSubmit() {
  try {
    loading.open();
    const { question, ...data } = formModel.value;
    const question_id = question.map((questionItem) => questionItem.id);
    const answer = question.map((questionItem) => (questionItem.answer ?
questionItem.answer == 'true' ? true : false : null));
    const hasNull = question.some(questionItem => questionItem.answer ===
null);
    if (hasNull) {
      throw new Error('Answer the question!!');
    }

    const newData = {
      ...data,
      question_id,
      answer
    };
    const response = await useRequest(`/careers/${route.params.slug}`, {
      method: 'POST',
      body: useFormDataBuilder(newData)
    });

    if(response.success == 'false'){
      throw new Error(response.message);
      return;
    }

    reset();
    loading.close();
    formRef.value?.clear();
    openModalSuccess.value = true;
  } catch (err: any) {
    useRequestError(err, formRef);
    loading.close();
  }
}
}

```

Gambar 3.49 Kode Program submit data apply pekerjaan

b. Slicing user interface pada dashboard member

Hal yang dilakukan adalah membuat *array* yang berisi daftar menu yang dapat diakses pada dashboard tersebut. Array ini akan diimplementasikan pada *sidebar* tergantung pada peran pengguna (role) yang menggunakan dashboard, dalam hal ini pengguna member. Kode program dari array tersebut dapat dilihat pada gambar 3.50.

```

export const userMenus: Menu[] = [
  {
    icon: 'ic:round-person',
    path: '/user',
    label: 'Profile',
    pages: ['user']
  },
  {
    icon: 'ic:baseline-work',
    path: '/user/job',
    label: 'Job Applied',
    pages: ['user-job']
  },
  {
    icon: 'ic:baseline-bookmark',
    path: '/user/saved-job',
    label: 'Saved Job',
    pages: ['user-saved-job']
  },
  {
    icon: 'ic:outline-settings',
    path: '/user/setting',
    label: 'Setting',
    pages: ['user-setting']
  },
  {
    icon: 'ic:round-lock',
    path: '/user/change-password',
    label: 'Change Password',
    pages: ['user-change-password']
  },
  {
    icon: 'ic:round-live-help',
    path: '/user/help-support',
    label: 'Help & Support',
    pages: ['user-help-support']
  }
];

```

Gambar 3.50 Array untuk menu pada dashboard member

Array tersebut merupakan *array of objects* yang terdiri dari empat pasangan key-value, yaitu: `icon` yang berisi ikon untuk menu tersebut, `path` yang berisi URL yang dituju saat tombol ditekan, `label` yang berisi nama menu tersebut, dan `pages` yang berisi menu tambahan yang diletakkan setelah *path* dan menandakan bahwa menu pada *sidebar* akan tetap aktif ketika halaman tersebut dibuka. Langkah selanjutnya adalah membuat *file* untuk halaman yang sesuai dengan *path* dan *pages* pada *array* yang telah dibuat, kemudian memulai *slicing* desain tampilan sesuai dengan desain yang telah dibuat oleh UI/UX desainer. Halaman yang dikerjakan terlebih dahulu adalah fitur Profile, Job Applied, Settings, Change Password, dan Help & Support.

- c. Pembuatan API serta mengintegrasikannya dengan fitur Account di dashboard author

Fitur ini terdiri dari tiga halaman, yaitu halaman utama untuk menampilkan data pribadi pengguna, halaman edit untuk mengubah data pribadi pengguna, dan halaman change password untuk mengubah password akun pengguna. Halaman utama membutuhkan tiga API yang masing-masing berguna untuk menampilkan data pengguna, menghapus akun, dan keluar akun. Sementara itu, halaman *edit* dan halaman *change password* masing-masing hanya membutuhkan satu API sesuai dengan kebutuhannya. Integrasi dengan UI yang telah dibuat tidak mengalami kendala, karena tidak ada perubahan yang diperlukan sehingga hanya perlu penyesuaian.

- d. Slicing user interface pada dashboard admin

Langkah yang dilakukan hampir sama dengan saat membuat UI pada *dashboard member*, yaitu membuat *array* yang berisi daftar menu yang dapat diakses pada *dashboard* ini. *Array* tersebut akan diimplementasikan pada *sidebar* tergantung pada peran pengguna yang menggunakan *dashboard*, dalam hal ini adalah pengguna admin. Langkah selanjutnya adalah membuat file untuk halaman yang sesuai dengan *path* dan *pages* pada *array* yang telah dibuat. Perbedaannya adalah menu pada dashboard admin mencakup seluruh menu yang dapat diakses pada dashboard author, ditambah beberapa menu tambahan khusus. Slicing UI hanya dilakukan pada fitur khusus pada dashboard ini yaitu halaman *user management*, *role*, dan *member*. Menu yang sama dengan *dashboard author* akan diimplementasikan setelah pengerjaan *dashboard author* selesai, dengan beberapa penyesuaian.

Sprint Kelima

- a. Pembuatan API serta mengintegrasikannya dengan fitur Profile, Job Applied, Change Password, dan Help&Support di dashboard member

Pengerjaan pada sprint ini diawali dengan mengerjakan beberapa fitur yang ada pada *dashboard member*. Setiap fitur ini hanya memiliki satu halaman saja, sehingga masing-masing hanya membutuhkan satu API, kecuali fitur Help & Support. Fitur Help & Support membutuhkan API untuk menampilkan FAQ dan juga untuk mengirimkan *email* ke *customer support*.

Integrasi API yang telah dibuat dengan tampilan fitur yang sudah ada sangat mudah. Fitur Profile, Job Applied, dan FAQ hanya memerlukan penampilan data saja. Fitur Change Password sudah dibuat pada *dashboard author*, sehingga hanya perlu menyalin kode. Dan pembuatan fitur pengiriman email ke *customer support* hanya memerlukan sedikit modifikasi dari kode-kode yang telah dibuat sebelumnya.

b. Slicing halaman Saved Job pada dashboard member

Sebelumnya terjadi kesalahan desain yang dilakukan oleh UI/UX designer dengan membuat fitur ini bagi pengguna author. Setelah penulis merasakan keganjalan ini akhirnya ia melaporkan dan telah dipindah ke dashboard member. Slicing tampilan fitur ini mudah karena hanya terdiri atas satu halaman serta menggunakan tabel dan komponen yang ada, dengan sedikit modifikasi.

c. Pembuatan API serta mengintegrasikannya dengan fitur role, user management, dan member di dashboard admin

Ketiga fitur ini merupakan fitur khusus yang ada pada *dashboard admin*. Seperti sebelumnya, langkah pertama yang dilakukan adalah pembuatan API untuk masing-masing fitur. Untuk kali ini, API yang telah dibuat untuk fitur *role* belum digunakan karena pengerjaannya akan dilanjutkan pada versi selanjutnya. Integrasi API pada fitur lainnya mudah karena hanya perlu menyalin dan melakukan sedikit modifikasi dari kode-kode fitur lainnya.

Namun, terdapat hal baru dalam fitur *member*, yaitu adanya fitur *export* data. Langkah pertama adalah membuat API yang berguna untuk mengambil data seluruh pengguna member dan mengembalikan data tersebut dalam format base64. Pada *frontend*, setelah tombol *export* ditekan, data tersebut akan diambil dengan *fetch*, kemudian diubah ke format Excel dan diunduh ke komputer lokal. Kode program untuk mengubah data ke format Excel dan mengunduhnya dapat dilihat pada gambar 3.51.

```

async function exportData() {
  const result = await useRequest('/admin/member/export', {
    method: 'GET',
  });
  const currentDate = new Date().toISOString().split('T')[0];
  // Convert Base64 to Blob
  const byteCharacters = atob(result.data);
  const byteNumbers = new Array(byteCharacters.length);
  for (let i = 0; i < byteCharacters.length; i++) {
    byteNumbers[i] = byteCharacters.charCodeAt(i);
  }
  const byteArray = new Uint8Array(byteNumbers);
  const blob = new Blob([byteArray], { type:
'application/vnd.openxmlformats-officedocument.spreadsheetml.sheet' });

  // Create temporary URL
  const url = URL.createObjectURL(blob);
  const filename = `data_member_joingeeek_${currentDate}.xlsx`;
  // Create a link element
  const link = document.createElement('a');
  link.href = url;
  link.setAttribute('download', filename);

  // Simulate click to trigger download
  link.click();

  // Clean up
  URL.revokeObjectURL(url);
}

```

Gambar 3.51 Fungsi untuk mengunduh data dalam format excel

d. Slicing user interface fitur autentikasi

Fitur ini bukan berupa halaman, melainkan sebuah modal yang muncul ketika tombol login pada navbar ditekan. Implementasi slicing untuk fitur ini sangat mudah, namun kesulitan yang ada terletak pada bagaimana satu modal tersebut dapat beralih ke tiga *form* yang berbeda, yaitu *form login*, *register*, dan *forgot password*.

Dalam *modal* tersebut, untuk menampilkan form tertentu, pengguna akan menekan tombol yang disediakan. Logikanya adalah tombol tersebut akan mengubah nilai *state* `modalType`. *Form* yang ditampilkan akan sesuai dengan nilai dari *state* tersebut, dan nilai *input* dari *form* sebelumnya akan di-*reset*. Kode program untuk implementasi ini dapat dilihat pada gambar 3.52.

```

<script setup lang="ts">
  const modalType = ref<'login' | 'register' | 'forgotPassword' |
string>('login');
</script>

```

Gambar 3.52 State untuk menyimpan data jenis form autentikasi

Sprint Keenam

a. Pembuatan API serta mengintegrasikannya dengan fitur autentikasi

Pembuatan fitur ini diawali dengan pembuatan tiga API yang masing-masing berfungsi untuk login, register, dan forgot password. Pembuatan API dan integrasinya ke tampilan yang telah dibuat sangat mudah, dengan menyalin serta memodifikasi sesuai kebutuhan. Meskipun prosesnya relatif sama, perbedaannya terletak pada jumlah input di setiap form dan validasi yang berlaku. Nilai pada form-form tersebut akan dihapus setelah berhasil disubmit.

b. Menampilkan profil member

Fitur ini berupa tombol yang menampilkan foto profil dan nama pengguna, serta menyembunyikan tombol *login*. Fitur ini berfungsi untuk menandakan bahwa pengguna telah login, dan ketika ditekan, akan langsung mengarahkan pengguna ke *dashboard member*. Tidak ada kesulitan dalam mengerjakan fitur ini karena logika yang digunakan sangat sederhana, dan data pengguna yang sudah *login* telah disimpan pada *state* `useStateUser`, yang datanya dapat langsung digunakan. Kode program untuk pembuatan fitur ini dapat dilihat pada gambar 3.53.

```
<template>
<li>
  <div v-if="user?.role == 'user'" class="cursor-pointer">
    <ULink to="/user" class="flex lg:flex-row flex-row-reverse justify-
between gap-2 items-center">
      <span>{{ user.name }}</span>
      <div v-if="!user.image" class="bg-[#D9D9D9] flex justify-center w-8
h-8 rounded-full">
        <AppIcon name="ic:baseline-person" class="text-black/55 w-5 h-5
self-center"></AppIcon>
      </div>
      
    </ULink>
  </div>
  <div v-else>
    <AuthModal></AuthModal>
  </div>
</li>
</template>
<script setup lang="ts">
const { data } = useStateUser();
</script>
```

Gambar 3.53 Kode program profil member

c. Pembuatan fitur Bookmark

Fitur ini merupakan fitur khusus bagi pengguna member pada halaman detail pekerjaan. Fungsinya adalah untuk menandai sebuah pekerjaan agar disimpan dan dapat dilihat pada dashboard masing-masing pengguna. Pembuatan fitur ini dimulai dengan membuat API yang berguna untuk menambah dan menghapus *bookmark*.

Tantangan dalam mengerjakan fitur ini adalah membuat tombol yang digunakan berupa *toggle* yang akan berubah *style*-nya ketika ditekan, yaitu saat pekerjaan telah di-*bookmark* maupun sebaliknya. Awalnya, rencananya adalah membuat satu tombol dengan *style* dan fungsi yang bergantung pada kondisi apakah pekerjaan sudah di-*bookmark* atau belum. Namun, untuk memudahkan implementasi dan pembacaan kode, dibuat dua tombol dengan tugas dan *style* yang berbeda. Kode program untuk pembuatan fitur bookmark ini dapat dilihat pada gambar 3.54.

```

<template>
<div class="flex justify-center mb-4 gap-2">
  <AuthModal :is-apply="true" @show-modal="openForm"></AuthModal>
  <div v-if="user?.role == 'user'" class="">
    <UButton
      v-if="data?.is_bookmark == true"
      icon="i-heroicons-bookmark"
      class="rounded-xl"
      @click="deleteBookmark"
    ></UButton>
    <UButton
      v-else
      variant="outline"
      icon="i-heroicons-bookmark"
      class="rounded-xl"
      @click="addBookmark"
    ></UButton>
  </div>
</div>
</template>

<script setup lang="ts">
async function addBookmark() {
  try {
    const reqBody = {
      post_id: data.value?.id
    };
    await useRequest('/careers/bookmark', {
      method: 'POST',
      body: reqBody
    });
    data.value !== null && data.value !== undefined ?
    (data.value.is_bookmark = true) : false;
  } catch (err: any) {
    useRequestError(err);
  }
  // data?.value?.is_bookmark = true;
}

async function deleteBookmark() {
  try {
    await useRequest(`/careers/bookmark/${data.value?.id}`, {
      method: 'DELETE'
    });
    data.value !== null && data.value !== undefined ?
    (data.value.is_bookmark = false) : false;
  } catch (err: unknown) {
    useRequestError(err);
  }
}
</script>

```

Gambar 3.54 Kode program fitur bookmark

d. Penambahan fitur form apply job

Ada tiga hal yang ditambahkan dalam fitur ini. Pertama, menampilkan modal autentikasi ketika menekan tombol ‘Apply now’ pada halaman detail pekerjaan jika pengguna belum *login* sebagai member. Kedua, mengimplementasikan fitur pertanyaan dengan menampilkan *form* pertanyaan pada *form apply job* berdasarkan pekerjaan yang dilamar.

Terakhir, menampilkan data pengguna pada *form apply job*. Ketika *form apply* dibuka dan pengguna member sudah *login*, data seperti nama, email, dan nomor HP akan diambil dari *state user* dan disimpan pada *state formModel*. Data tersebut akan menjadi nilai *default* yang ditampilkan sebagai *input* pada *form apply job*. Kode program pengimplementasiannya dapat dilihat pada gambar 3.55.

```
<script setup lang="ts">

const { data } = useStateUser();

const formModel = ref({
  name: '',
  email: '',
  phone: '',
  curriculum_vitae: null as File | null,
  cover_letter: '',
  portfolio: null as File | null,
  question: [] as QuestionType[]
});

onMounted(() => {
  isOpen.value = true;

  if (data.value?.role == 'user') {
    formModel.value.name = data.value.name;
    formModel.value.email = data.value.email;
    formModel.value.phone = data.value.phone ?? '';
  }
  fetchQuestion();
});
</script>
```

Gambar 3.55 Menampilkan data pengguna pada form input

Kesulitan yang dihadapi adalah dalam mengimplementasikan *form* yang sudah ada dengan tambahan *form* pertanyaan. Kedua *form* tersebut memiliki kondisi yang berbeda, di mana setiap pekerjaan belum tentu memiliki pertanyaan yang sama atau bahkan tidak ada pertanyaan sama sekali. Oleh karena itu, diperlukan penyesuaian dalam logika kode untuk menangani kondisi ini.

e. Penambahan filter pada daftar pekerjaan di landing page

Filter yang akan ditambahkan adalah filter berdasarkan kategori pekerjaan. Daftar kategori akan didapatkan dari database karena kategori-kategori tersebut dapat berubah. Langkah pertama adalah membuat API untuk mendapatkan daftar kategori. Kemudian, dibuat sebuah fungsi untuk melakukan fetch data dan menyimpan data kategori pada *state* bernama 'statusCategory' yang akan di-*render* setiap halaman dibuka.

Selanjutnya, dibuat sebuah UI untuk menampilkan daftar kategori tersebut. Daftar ini berupa tombol-tombol dengan tombol 'All' sebagai tombol aktif secara default. Ketika tombol lain ditekan, fungsi onSelectStatus akan dipicu untuk mengubah nilai filter berdasarkan pilihan sebelumnya dan menjalankan kembali fungsi fetchData untuk meminta data terbaru dari server. Kode program pengimplementasiannya dapat dilihat pada gambar 3.56.

```

<template>
<div class="flex items-center mt-14 gap-8 mx-auto justify-center">
  <UButton
    v-for="(item, k) in statusCategories"
    :key="`status-${k}`"
    class="flex justify-center w-[200px] text-center rounded-none bg-white
text-gray-400 font-semibold disabled:text-[#000] disabled:border-0
disabled:border-b-2 disabled:border-b-black"
    :disabled="filter.category === item.id"
    @click="onSelectStatus(item.id)"
  >
    {{ item.name }}
  </UButton>
</div>

</template>

<script setup lang="ts">

interface Categories {
  id: number;
  name: string;
}
const statusCategories = ref<Categories[]>([]);

async function fetchCategory() {
  try {
    const data = await useRequest('/back/job-categories', {
      method: 'GET'
    });
    statusCategories.value = [
      {
        id: 0,
        name: 'All'
      },
      ...data
    ];
  } catch (err: any) {
    useRequestError(err);
  }
}

async function onSelectStatus(id: number) {
  try {
    loadingAll.open();
    filter.value.category = id;
    await fetchJob();
  } catch (err) {
    useRequestError(err);
  } finally {
    loadingAll.close();
  }
}

onMounted(async () => {
  await fetchJob();
  await fetchCategory();
});
</script>

```

Gambar 3.56 Kode program filter kategori pekerjaan

Ada beberapa kesulitan dalam penyelesaian fitur ini. Fitur ini sudah memiliki dua filter lainnya, yaitu pencarian berdasarkan nama pekerjaan dan tipe pekerjaan. Kesulitan yang dihadapi adalah bagaimana cara meletakkan UI serta mengimplementasikan logika ketika harus menambahkan filter baru pada fitur yang sudah memiliki filter lainnya.

f. Kloning fitur dari dashboard author ke dashboard admin

Setelah menyelesaikan fitur khusus pada *dashboard admin* dan fitur-fitur pada *dashboard author*, langkah selanjutnya adalah mengkloning fitur tersebut ke *dashboard admin*. Proses kloning dilakukan dengan menyalin keseluruhan kode dan melakukan beberapa perubahan kecil, seperti mengubah rute navigasi yang sebelumnya berawalan `/author` menjadi `/admin`, mengubah *middleware* menjadi `admin`, dan mengubah judul halaman yang awalnya berakhiran `author` menjadi `admin`. Contoh kode program untuk mengimplementasikan perubahan tersebut dapat dilihat pada gambar 3.57.

```
<script setup lang="ts">

definePageMeta({
  key: (r) => r.fullPath,
  layout: 'author',
  middleware: ['admin']
});

useHead({
  title: 'Applicant List - admin'
});

</script>
```

Gambar 3.57 Kode program implementasi kloning fitur

Sprint Ketujuh

a. Pengalih kerjaan tugas

Kurang dari sebulan sebelum waktu magang saya selesai, perusahaan merekrut pemegang baru dengan posisi yang sama seperti saya. Tujuan perekrutan ini adalah memberikan kesempatan magang kepada mahasiswa yang ingin mendapatkan pengalaman praktis, serta untuk memastikan kelanjutan pekerjaan yang telah saya mulai. Dengan demikian, transisi pekerjaan dapat berjalan lancar, dan proyek yang sedang berlangsung dapat diselesaikan dengan baik oleh pemegang baru.

Penulis kemudian memberikan arahan kepada pemegang baru mengenai teknologi dan *tools* yang perlu dipelajari sebelum terjun dalam pengerjaan proyek. Selanjutnya, penulis memberikan instruksi untuk meng-*instal* dan menjalankan proyek di komputer lokal, serta memberikan pemahaman mengenai kode program dan alur proyek. Penulis juga memberikan kesempatan kepada pemegang baru untuk praktik langsung dalam proyek, sehingga kesulitan yang dihadapi segera diketahui dan dapat segera diatasi.

- b. Pembuatan API serta mengintegrasikannya dengan fitur Saved Job, dan setting di dashboard member

Dua fitur ini merupakan fitur baru yang dikerjakan pada *dashboard member* di *sprint* terakhir. Fitur Saved Job baru dikerjakan karena *slicing* baru dilakukan pada sprint sebelumnya, sedangkan fitur *Setting* baru dikerjakan karena lebih rumit dibanding fitur lainnya. API yang dibuat terdiri dari satu API untuk fitur *Setting* dan dua API untuk fitur *Saved Job*.

Pengerjaan fitur Saved Job relatif mudah meskipun memiliki lebih banyak API, karena hanya untuk menampilkan data dalam tabel yang memerlukan sedikit modifikasi sesuai kebutuhan serta menghapus data. Fitur Setting lebih rumit meskipun hanya memiliki satu API. Fitur ini berupa *form* dengan lebih dari 15 *input* yang berbeda-beda jenisnya. *Input-input* tersebut juga terdiri dari beberapa kelompok yang memerlukan validasi berbeda-beda dan berupa *array of objects*. Oleh karena itu, tampilan yang sudah dibuat sering kali harus diubah untuk menyesuaikan dengan kebutuhan yang baru diketahui saat integrasi dengan API.

- c. Perbaikan bug di Job Applicant

Terdapat kendala berupa email yang tidak terkirim ketika pengguna mengubah status lamaran pelamar pada fitur ini. Pemegang mencari sumber masalah ini, namun tidak ditemukan pada kode program frontend. Setelah itu, pemegang meminta pengembang backend untuk memeriksa apakah ada kesalahan dalam kode program backend yang menyebabkan bug tersebut. Ditemukan bahwa masalahnya ada di backend, sehingga pemegang menunggu perbaikan dari tim backend untuk diimplementasikan di frontend.

d. Perbaikan bug ikon yang sering tidak muncul

Penggunaan ikon sangat masif dalam proyek ini. Kode program diatur untuk memunculkan ikon melalui server sehingga ikon hanya di-*render* saat dibutuhkan. Kendala yang sering dihadapi adalah ikon yang dibutuhkan tidak muncul, sehingga perlu dilakukan perbaikan untuk mengatasi masalah tersebut.

```
<UIcon    name="i-ic-baseline-delete"    class="text-white    w-5    h-5    self-center"></UIcon>
```

Gambar 3.58 Contoh kode program memunculkan ikon

Gambar 3.58 menunjukkan kode program yang dibuat untuk menggantikan kode sebelumnya. Kode baru ini dirancang untuk memunculkan ikon menggunakan metode bawaan dari library ikon yang digunakan. Dengan pendekatan ini, diharapkan masalah ikon yang tidak muncul dapat teratasi secara lebih efektif. Kode program ini hanya diimplementasikan pada ikon yang belum muncul. Jika terdapat ikon di fitur lain yang juga belum muncul, kode program yang sama akan diterapkan.

3.3. Pengujian

Pengujian dilakukan setelah setiap fitur diselesaikan menggunakan metode *black box testing* oleh QA/QC. Tujuannya adalah memastikan bahwa semua fungsi dalam sistem berjalan sesuai dengan proses bisnis dan kebutuhan perusahaan. Pengujian dilakukan dengan menambahkan data dan menguji setiap fitur di setiap halaman menggunakan beberapa skenario uji secara manual.

Berdasarkan hasil keluaran yang ditampilkan pada sistem, tim QA/QC dapat menilai apakah hasil tersebut sesuai dengan alur proses bisnis yang diharapkan. Jika masih ditemukan isu pada keluaran, isu tersebut akan dicatat dan diserahkan kepada proyek manajer untuk disampaikan kepada tim pengembang. Isu atau bug yang muncul akan diselesaikan oleh pengembang agar keluaran yang dihasilkan tidak mengandung cacat atau *bug*. Jika isu belum dapat diselesaikan, akan didiskusikan kembali pada pengembangan aplikasi di versi selanjutnya. Contoh hasil pengujian yang dilakukan dapat dilihat pada Tabel 3.4.

Tabel 3.4 Pengujian pada fitur sidebar dashboard member

Halaman Umum				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan logo pada <i>sidebar</i>	Masuk ke halaman <i>landing page</i>	Masuk ke halaman <i>landing page</i>	Terpenuhi
2	Menekan <i>Hamburger</i> menu di <i>sidebar</i>	<i>Sidebar</i> akan terbuka seluruhnya ketika keadaan awal <i>sidebar</i> hanya menampilkan ikon saja, dan sebaliknya	<i>Sidebar</i> akan terbuka seluruhnya ketika keadaan awal <i>sidebar</i> hanya menampilkan ikon saja, dan sebaliknya	Terpenuhi
3	Menekan tombol ' <i>Logout</i> ' di <i>sidebar</i>	Memunculkan <i>pop up</i> dialog dengan pertanyaan " <i>Are you sure, you want to log out???</i> " dengan pilihan ' <i>No</i> ' dan ' <i>Logout</i> '	Memunculkan <i>pop up</i> dialog dengan pertanyaan " <i>Are you sure, you want to log out???</i> " dengan pilihan ' <i>No</i> ' dan ' <i>Logout</i> '	Terpenuhi
4	Menekan tombol pilihan pada <i>pop up</i> dialog <i>Logout</i>	Menutup <i>pop up</i> jika tombol ' <i>No</i> ' ditekan dan diarahkan ke <i>landing page</i> jika tombol ' <i>Logout</i> ' ditekan	Menutup <i>pop up</i> jika tombol ' <i>No</i> ' ditekan dan diarahkan ke <i>landing page</i> jika tombol ' <i>Logout</i> ' ditekan	Terpenuhi

Kesimpulan hasil pengujian dari setiap fitur yang dikerjakan dapat dilihat pada Tabel 3.5. Tabel tersebut memberikan penjelasan singkat tentang fitur serta jumlah pengujian yang berhasil dan gagal. Pengujian yang diuraikan adalah pengujian yang dilakukan pada fitur yang melibatkan pemegang dan juga fitur yang telah selesai. Informasi lebih rinci mengenai pengujian fitur akan disajikan dalam lampiran. Mayoritas hasil pengujian berhasil pada keseluruhan pengujian menandakan bahwa aplikasi ini secara teknis telah berhasil diselesaikan dan memenuhi kebutuhan dari *stackholder*.

Tabel 3.5 Rangkuman pengujian setiap fitur

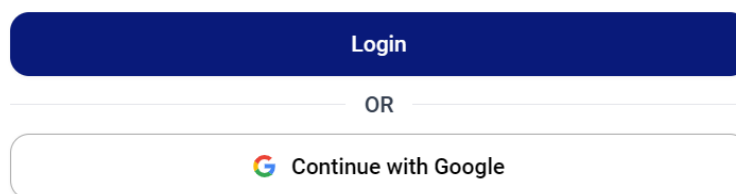
No	Halaman	Sub No	Fitur	Hasil Pengujian Berhasil	Hasil Pengujian Gagal
Front Office					
1	Autentikasi	a	Register	6	0
		b	Login	3	0
		c	Forgot Password	4	0
2	Umum	a	Landing Page	4	1
		b	Detail Job	6	0
		c	Apply Job	2	0
Dashboard Member					
3	Sidebar	a	Sidebar	4	0

No	Halaman	Sub No	Fitur	Hasil Pengujian Berhasil	Hasil Pengujian Gagal
4	Profile	a	Profile	3	0
5	Job Applied	a	Job Applied	3	0
6	Saved Job	a	Saved Job	3	0
7	Setting	a	Setting	4	0
8	Change Password	a	Change Password	6	0
9	Help & Support	a	Help & Support	4	0
Dashboard Author					
10	Sidebar	a	Sidebar	2	0
11	Author	a	Author	1	0
12	Job Listing	a	Job Listing	6	0
		b	Edit Job	4	0
		c	Create Job	2	0
13	Job Available	a	Job Available	5	0
14	Job Applicants	a	Job Applicants	16	0
15	Internship	a	Internship	8	0
16	Questions	a	Questions	6	0
		b	Detail Question	5	0
		c	Create New Question	4	0
		d	Edit Question	2	0
17	Account	a	Account	7	0
		b	Edit Account	3	0
		c	Change Password	5	0
Dashboard Admin					
18	Sidebar	a	Sidebar	2	0
19	Admin	a	Admin	4	0
20	Member	a	Member	7	0

3.4. Perencanaan Pengerjaan Versi Selanjutnya

3.4.1. Login dengan Google

Fitur ini merupakan opsi tambahan bagi pengguna untuk *login* sebagai pengguna *member*, selain dari proses registrasi dan login menggunakan email dan password. Proses pengembangan fitur ini telah mencapai tahap desain tombol yang disematkan ke dalam halaman web, yang terletak dalam modal di bawah formulir *login* dengan *email* dan *password*. Foto dari tampilan tombol *web* tersebut dapat dilihat pada gambar 3.59. Pengembangan fitur ini akan dilanjutkan dalam versi berikutnya karena implementasinya memerlukan waktu yang cukup lama, dan juga dikarenakan fitur ini dapat dianggap sebagai fitur tambahan yang fungsinya sebagian dapat digantikan oleh fitur *login* menggunakan *email* dan *password* yang telah ada sebelumnya.



Gambar 3.59 Tombol login menggunakan google

3.4.2. Menampilkan foto profil bagi member yang telah login

Fitur ini terintegrasi dengan fitur menampilkan nama *member* yang telah login. Data pengguna yang telah masuk (*login*) diambil dari sebuah api yang tersedia, kemudian setiap pengguna yang melakukan permintaan ke server akan menyimpan data tersebut dalam sebuah *state*. Data pengguna yang ada dalam *state* digunakan untuk otorisasi informasi bagi pengguna, termasuk untuk menampilkan foto profil dan nama *member* yang telah *login*.

Saat pengguna baru membuat akun dan melakukan login pada aplikasi ini, data yang dimiliki oleh pengguna hanya berupa *email* dan *username*. Oleh karena itu, secara *default* hanya gambar profil statis dan nama pengguna yang ditampilkan. Setelah pengguna mengisi data diri, *server* seharusnya memberikan *respons* yang memuat data pengguna yang diperlukan. Namun, ada kesalahan dari *backend* di mana foto profil tidak disertakan dalam *respons* tersebut. Akibatnya, foto profil yang ditampilkan pada *navbar* bagi pengguna yang telah *login* tetap menggunakan foto profil *default* meskipun pengguna telah mengisi data diri dengan lengkap.

Pemagang mengetahui saat program magang hampir selesai dilakukan. Kesalahan terjadi pada *response api* yang belum menyediakan foto profil yang merupakan tugas dari *backend*. Sebagai seorang pengembang *frontend*, pemagang tidak memiliki kewenangan untuk mengatasi masalah tersebut dan hanya bisa menunggu hingga *backend* memperbaikinya. Keputusan yang diberikan oleh proyek manajer bahwa pengerjaan fitur ini akan dilanjutkan pada versi aplikasi selanjutnya.

3.4.3. Fitur Course

Seperti yang telah dijelaskan sebelumnya, dalam versi pertama, pengerjaan proyek akan difokuskan pada manajemen rekrutmen karena terbatas oleh waktu. Fitur *course* tidak termasuk dalam fitur manajemen rekrutmen, melainkan masuk ke dalam fitur manajemen komunitas. Oleh karena itu, fitur tersebut belum menjadi prioritas dalam pengerjaan. Meskipun demikian, *slicing* desain *user interface* ke bentuk *web* telah diselesaikan yang berkemungkinan besar akan mengalami perubahan di masa mendatang. Pengerjaan lanjutan dari fitur ini akan dilakukan pada versi selanjutnya.

3.4.4. Fitur Certificate

Seperti yang telah dijelaskan sebelumnya, dalam versi pertama, pengerjaan proyek akan difokuskan pada manajemen rekrutmen karena terbatas oleh waktu. Sama seperti menu *course*, menu *Certificate* tidak termasuk dalam fitur manajemen rekrutmen, melainkan masuk ke dalam fitur manajemen komunitas. Oleh karena itu, fitur tersebut belum menjadi prioritas dalam pengerjaan. Meskipun demikian, *slicing* desain *user interface* ke bentuk *web* telah diselesaikan yang berkemungkinan besar akan mengalami perubahan di masa mendatang. Pengerjaan lanjutan dari fitur ini akan dilakukan pada versi selanjutnya.

3.4.5. Fitur Blog

Fitur *Blog* tersedia bagi pengguna *author* dan *admin* di masing-masing *dashboard*. Pengerjaan fitur ini telah selesai di *dashboard author*, termasuk proses *slicing* ke bentuk *web* serta integrasi dengan API. Namun, masalah semakin muncul saat mencoba mengimplementasikan fitur ini di *dashboard admin*.

Fitur ini tidak bermasalah pada pengerjaannya, akan tetapi bermasalah pada alur pengguna pada fitur ini. Pada saat pengerjaan di *dashboard author*, terjadi kebimbangan untuk menampilkan seluruh *blog* yang telah dibuat oleh pengguna *author* dengan resiko pengguna bisa saling memanipulasi *blog* milik pengguna lain atau hanya menampilkan *blog* milik pengguna yang bersangkutan. Proyek manajer mengambil keputusan dalam fitur ini hanya menampilkan *blog* milik pengguna yang bersangkutan.

Kemudian, saat mengimplementasikan fitur ini pada *dashboard admin*, muncul masalah lagi. Sebagai pengguna *admin*, pertanyaannya adalah apakah semua *blog* harus ditampilkan atau tidak, serta tugas apa yang diemban oleh pengguna *admin* dalam fitur ini dan batasan otorisasi bagi *admin* terkait dengan *blog* yang ada. Mengingat masalah yang timbul dan mempertimbangkan bahwa fitur ini merupakan fitur utama yang belum disediakan tempat untuk menampilkannya di bagian *front office*, diputuskan pengerjaan fitur ini akan dilanjutkan pada versi selanjutnya.

3.4.6. Halaman Roles

Halaman *Roles* merupakan salah satu fitur yang sekarang dapat diakses pada *dashboard admin*. Pengerjaan ini telah mencapai tahap slicing desain *user interface* ke bentuk *web* dan integrasi dengan API, namun dengan fungsi otorisasi fitur-fitur belum berjalan. Fitur ini muncul di pertengahan proses pengembangan aplikasi setelah pengerjaan *dashboard author* selesai.

Sebelumnya, telah dijelaskan bahwa aplikasi ini memiliki tiga jenis pengguna, yaitu *member*, *author*, dan *admin*. Fitur ini seharusnya akan menghilangkan peran *admin* dan juga *author*, kemudian membuat jenis pengguna baru dengan akses fitur-fitur yang dipilih berdasarkan halaman yang berada di *dashboard author* dan *admin* sebelumnya. Namun, karena struktur *folder*, otorisasi fitur, dan juga *sidebar* telah disesuaikan dengan menggunakan tiga jenis pengguna, maka dibutuhkan perombakan sebagian besar kode program yang akan memakan waktu yang lumayan lama. Oleh karena itu, pengerjaan lanjutan untuk fitur ini akan didiskusikan kembali pada versi selanjutnya.

3.5. Penutupan Proyek

Penutupan proyek pengembangan aplikasi Joingeek belum dilakukan karena aplikasi ini adalah salah satu aplikasi internal yang dimiliki oleh GeekGarden. Oleh karena itu, pengerjaan proyek ini akan terus berlanjut dan aplikasi akan terus dikembangkan sesuai dengan kebutuhan perusahaan. Namun, pengerjaan versi pertama telah selesai, dan tugas-tugas selanjutnya telah dialihkan dari pemegang ke pemegang lain yang menjabat sebagai *frontend developer* di GeekGarden.

BAB 4

REFLEKSI PELAKSANAAN MAGANG

4.1. Relevansi Akademik

Dalam pelaksanaan magang, terutama dalam pengerjaan proyek Pengembangan *Frontend* Sistem Informasi Manajemen Rekrutmen dan Komunitas Menggunakan *Framework* Nuxt ini terdapat tiga hal yang relevan dengan pembelajaran di kuliah. Hal-hal tersebut antara lain teknologi yang digunakan, manajemen pengembangan aplikasi, dan metode pengembangan aplikasi.

Mengenai teknologi yang dipakai dalam proyek ini, tidak ditemukan perbedaan dengan teori yang diajarkan di bangku perkuliahan. Pada mata kuliah Fundamen Pengembangan Aplikasi dan Pengembangan Aplikasi Berbasis Web telah diajarkan mengenai pembuatan web menggunakan html, css, dan php. Perbedaannya adalah pada pembelajaran di kuliah, lebih diajarkan menggunakan teknologi dasar untuk membuat web sehingga memperkuat konsep pemograman web. Sedangkan pada pengerjaan proyek ini, digunakan teknologi terbaru yang didasarkan pada pembelajaran di kuliah, yaitu menggunakan vue js yang merupakan sebuah *framework* untuk pembuatan *frontend* aplikasi berbasis web.

Proses manajemen pengembangan aplikasi pada proyek ini, sedikit berbeda dengan teori yang diajarkan di bangku perkuliahan. Pada mata kuliah Manajemen Pengembangan Teknologi Informasi diajarkan mengenai manajemen sebuah proyek secara konsep dan pembuatan dokumen-dokumen yang diperlukan. Dalam proyek ini lebih diajarkan untuk menggunakan alat-alat ataupun aplikasi tertentu agar proses manajemen yang dilakukan lebih dimudahkan dengan mengikuti arahan dan aturan yang berlaku.

Metode pengembangan aplikasi yang digunakan dalam proyek ini, memiliki perbedaan dengan apa yang dipelajari di bangku perkuliahan terutama yang di praktikkan saat pengerjaan tugas besar. Dalam pengerjaan tugas besar, biasanya tidak terlalu memperhatikan dan mempraktikkan metode pengembangan aplikasi yang telah diajarkan, hal ini dikarenakan lingkup proyek yang masih kecil, waktu pengerjaan yang pendek, dan juga kemampuan yang tidak merata antar anggota kelompok sehingga hanya berfokus pada penyelesaian tugas saja. Sedangkan pada pengerjaan proyek JoinGeek digunakan metode scrum yang telah diajarkan di kuliah, yang mana dengan metode ini membantu pengembangan aplikasi dalam pembagian tugas, manajemen waktu, dan juga goals yang akan dicapai.

Metode scrum yang digunakan dalam proyek ini lebih fleksibel dalam penggunaannya dibandingkan teori yang dipelajari sebelumnya. Fleksibilitas metodologi Scrum terlihat dalam pengerjaan item-item pada *sprint planning*. Terkadang, beberapa tugas perlu dikerjakan meskipun tidak terdaftar dalam *product backlog*. Hal ini dapat mempercepat penyelesaian aplikasi. Namun, fleksibilitas tersebut tidak selalu membawa dampak positif. Jika tugas pada sprint tersebut sudah sulit, penambahan tugas baru bisa membuat pekerjaan menjadi lebih sulit bahkan bisa tidak sempat dikerjakan. Fleksibilitas dalam proses pengembangan ini dapat menjadi pedang bermata dua jika implementasi yang dilakukan tidak memperhitungkan situasi dan kondisi.

4.2. Pembelajaran Magang

Selama menjalani kegiatan magang, penulis mendapatkan banyak pembelajaran baik itu yang langsung diajarkan oleh mentor maupun dipelajari sendiri, baik dari kendala yang didapat ketika berproses maupun tantangan ketika belajar. Beberapa pembelajaran yang didapat selama mengikuti program ini antara lain :

- a. Memperluas pengetahuan mengenai teknologi dan *tools* terkini

Framework Vue maupun Nuxt bukanlah hal yang baru bagi pemegang. Sebelumnya, pemegang telah mengenal teknologi tersebut, namun belum mempelajarinya secara mendalam. Dalam proyek ini, penulis diberi kesempatan untuk mengerjakan proyek dengan teknologi tersebut sambil belajar lebih dalam. Dengan arahan dari mentor, penulis menjadi lebih memahami teknologi tersebut serta *library-library* pendukung lainnya yang dapat mempercepat pengerjaan proyek.

Dengan banyaknya teknologi yang tersedia, bukan berarti tidak ada kendala yang dihadapi. Dalam pengerjaan proyek yang melibatkan banyak pihak, penulisan kode program sering kali berbeda satu sama lain karena tidak ada dokumentasi atau aturan yang diberlakukan oleh perusahaan. Hal ini membuat proses belajar menjadi sedikit lebih berat dan terkesan lambat. Namun, ini bukan berarti buruk. Terkadang, terdapat banyak cara untuk mencapai sesuatu, dan tidak ada yang salah dalam penulisan kode selama hal tersebut dapat menyelesaikan masalah. Dengan mempelajari berbagai persepsi dan gaya penulisan kode dari orang lain, penulisan kode program oleh penulis menjadi lebih baik dan efisien.

b. Mempelajari dan mempraktikkan metode scrum dalam sebuah manajemen proyek

Metode-metode manajemen proyek sebenarnya sudah diajarkan di bangku perkuliahan. Namun, sering kali metode-metode tersebut belum sempat atau belum bisa dipraktikkan dalam pengerjaan tugas besar. Pada proyek ini, digunakan metode scrum sebagai metode pengembangan proyek. Melalui proyek ini, penulis mulai diajarkan menggunakan aplikasi-aplikasi pendukung metode ini, seperti Jira, Trello, dan aplikasi-aplikasi lainnya.

Kendala yang dihadapi termasuk adaptasi terhadap penggunaan alat-alat baru tersebut dan implementasi prinsip-prinsip Scrum dalam tim. Penulis harus belajar menavigasi fitur-fitur dalam aplikasi manajemen proyek, memahami cara mengelola tugas-tugas dengan efisien, serta beradaptasi dengan ritme sprint dan iterasi yang cepat. Tantangan lainnya adalah komunikasi dan kolaborasi dalam tim yang terdiri dari anggota dengan beragam tingkat keahlian dan pengalaman. Hal ini merupakan pengalaman belajar yang berharga, meskipun prosesnya bisa menjadi cukup menantang pada awalnya.

c. Memahami peran dan komunikasi antar individu

Sebagai pemegang dalam sebuah proyek dengan banyak anggota tim, saya memahami bahwa tidak perlu menguasai semua bidang. Keberagaman peran dalam tim memungkinkan setiap anggota untuk fokus pada spesialisasi mereka masing-masing. Dalam hal ini, saya dapat berkonsentrasi pada peran yang telah ditentukan oleh perusahaan, yaitu sebagai *frontend developer*, dengan mengasah keterampilan spesifik yang relevan dengan posisi tersebut. Setiap anggota tim memiliki peran masing-masing dan memiliki kemampuan yang lebih dalam bidang tertentu dibandingkan saya. Ketergantungan satu sama lain serta arahan dan kesempatan dari teman-teman lain membuat saya semangat untuk belajar agar dapat mengimbangi kecepatan mereka dalam menyelesaikan tugas-tugas yang diberikan.

Kendala yang mungkin saya hadapi adalah kurangnya pengalaman dalam berkolaborasi dan terjun langsung di dalam proyek, sehingga kosakata dan cara penyampaian yang digunakan dalam komunikasi kadang-kadang kurang dimengerti oleh tim yang dapat menyebabkan kesalahpahaman pada anggota tim. Untuk mengatasi kendala tersebut, saya mulai meningkatkan keterampilan komunikasi dengan cara aktif bertanya dan meminta klarifikasi jika ada istilah atau instruksi yang kurang jelas. Selain itu, saya juga meminta masukan secara berkala untuk memastikan pemahaman saya sudah sesuai dengan harapan tim. Dengan pendekatan ini, saya berharap dapat lebih efektif berkontribusi dan berkolaborasi dalam tim.

BAB 5

PENUTUP

5.1. Kesimpulan

Dari hasil pelaksanaan magang dengan ikut serta dalam pengembangan *frontend* Sistem Informasi Manajemen Rekrutmen dan Komunitas menggunakan *framework* Nuxt, didapatkan kesimpulan sebagai berikut:

- a. Penulis telah berhasil menyelesaikan tugas magang untuk mengembangkan *frontend* aplikasi JoinGeek menggunakan *framework* Nuxt untuk fase pertama. Hasil pengujian fitur pada pembahasan sebelumnya menunjukkan bahwa sebagian besar fitur telah lolos uji yang menandakan bahwa aplikasi ini secara teknis telah berhasil diselesaikan dan memenuhi kebutuhan dari *stackholder*, sedangkan fitur yang belum lolos akan dikerjakan pada fase berikutnya oleh pengembangan lain sesuai kesepakatan.
- b. *Framework* Nuxt mempermudah pengembangan *frontend* karena memiliki fitur bawaan yang lengkap, didukung oleh komunitas yang aktif, serta memiliki dukungan dari banyak *library*.
- c. Pengimplementasian metode Scrum dalam proyek sangat membantu dalam pengerjaan proyek. Pemilihan metode Scrum sangat tepat karena proyek ini akan terus dikembangkan sesuai kebutuhan perusahaan.
- d. Manajemen proyek yang baik menghasilkan pembagian tugas yang efektif, sehingga manajemen waktu menjadi lebih maksimal dan komunikasi antar tim menjadi sangat baik.

5.2. Saran

Terdapat beberapa saran yang diberikan untuk membantu pengembangan proyek ini, antara lain :

- a. Perencanaan proyek dalam satu fase seharusnya sudah mencapai tahap final sebelum tahap pengembangan, sehingga pengembang dapat membuat kode program yang mudah dipahami dan dikembangkan di masa mendatang.
- b. Perencanaan proyek untuk setiap fitur harus detail, sehingga selama pengerjaan tidak terjadi hambatan yang tidak perlu, dan waktu tidak terbuang.

- c. Dokumentasi kode program dalam pengembangan proyek harus segera disusun, karena kode program yang ditulis sangat kustom dan pengembang yang mengerjakannya sering berganti.

DAFTAR PUSTAKA

- Aryasta, A. R. (2022). *Pengembangan Front End Sistem Informasi Akuntansi Mengguanakan Kerangka Kerja Vue.js*.
- Erdanto, A. N. (2021). *Peningkatan Efektivitas Pengelolaan State menggunakan Pola Prop Drilling Vuejs pada Fitur finansial Jala Tech*.
- Farhan Saputra, & Franciscus Dwikotjo Sri Sumantyo. (2023). Pengaruh Sistem Informasi Manajemen: Kepuasan Konsumen dan Keputusan Pembelian Tiket MPL Mobile Legend di Aplikasi Blibli.com. *Jurnal Kewirausahaan Dan Manajemen Bisnis: Cuan*, 1(2), 98–105. <https://doi.org/10.59603/cuan.v1i2.18>
- Febriyantia, N. M. D., Sudana, A. A. K. O., & Piarsa, I. N. (2021). Implementasi Black Box Testing pada Sistem Informasi Manajemen Dosen. *JITTER- Jurnal Ilmiah Teknologi Dan Komputer*, 3, 1–10.
- GeekGarden. (2022). *GeekGarden Company Profile*.
- Hanafie, A., Chintami D.A, A., B, N. I., & Sulihin, S. (2023). Pengembangan Website Yayasan Al-Hizam Menggunakan Framework Nuxt JS. *Jurnal Teknologi Dan Komputer (JTEK)*, 3(01), 246–251. <https://doi.org/10.56923/jtek.v3i01.114>
- Helda, N., & Suryadi, S. (2023). Koneksi Tanpa Batas: Membangun Portfolio Web Interaktif dengan Vue, Nuxt, Dan API. *Jurnal Minfo Polgan*, 12(1), 1557–1568. <https://doi.org/10.33395/jmp.v12i1.12892>
- Muay, N. T., Sedyono, E., & Tambotoh, J. (2024). Integration Waterfall and Scrum Methodology in The Development of SIMARGA Web Application. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, 8(2), 223–233. <https://doi.org/10.29207/resti.v8i2.5652>
- Pšenák, P., & Tibenský, M. (2020). The usage of Vue JS framework for web application creation. *Mesterséges Intelligencia*, 2(2), 61–72. <https://doi.org/10.35406/MI.2020.2.61>
- Putra, R. W. (2023). *Pengembangan Aplikasi Jurnal Emosi Berbasis Progressive Web App*.
- You, E. (n.d.). *Frequently Asked Questions - Vue.js*.

LAMPIRAN

Lampiran 1. Sertifikat Magang



Lampiran 2. Pengujian Aplikasi

Lampiran 2.1 Front Office

Register Member				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan tombol "Register" pada <i>pop up login</i>	Menampilkan <i>form register</i> pada <i>pop up</i> autentikasi	Menampilkan <i>form register</i> pada <i>pop up</i> autentikasi	Terpenuhi
2	Mengisi nama lengkap, <i>email</i> , <i>password</i> , dan konfirmasi <i>password</i> dengan benar	Berhasil <i>register akun</i> serta muncul <i>toast</i> dengan pesan "Register Success" dan dan memunculkan lagi tampilan <i>form login</i>	Berhasil <i>register akun</i> serta muncul <i>toast</i> dengan pesan "Register Success" dan dan memunculkan lagi tampilan <i>form login</i>	Terpenuhi
3	Mengisi <i>input</i> yang tidak sesuai dengan formatnya	Memunculkan pesan <i>error</i> di bawah setiap <i>input</i> yang salah	Memunculkan pesan <i>error</i> di bawah setiap <i>input</i> yang salah	Terpenuhi

Register Member				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
4	Mengisi <i>email</i> yang <i>valid</i> dan sudah terdaftar	Memunculkan pesan <i>error</i> di bawah <i>input email</i> dengan pesan " <i>The email has already been taken.</i> "	Memunculkan pesan <i>error</i> di bawah <i>input email</i> dengan pesan " <i>The email has already been taken.</i> "	Terpenuhi
5	<i>Password</i> tidak sesuai dengan pola yang telah ditentukan	Memunculkan pesan <i>error</i> di bawah <i>input password</i> dengan pesan " <i>The password must contain uppercase letter, lowercase letter, number.</i> "	Memunculkan pesan <i>error</i> di bawah <i>input password</i> dengan pesan " <i>The password must contain uppercase letter, lowercase letter, number.</i> "	Terpenuhi
6	<i>Password</i> dan juga konfirmasi <i>password</i> tidak sama	Memunculkan pesan <i>error</i> di bawah <i>input konfirmasi password</i> dengan pesan " <i>The password confirm and password must match.</i> "	Memunculkan pesan <i>error</i> di bawah <i>input konfirmasi password</i> dengan pesan " <i>The password confirm and password must match.</i> "	Terpenuhi

Login Member				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan tombol " <i>Login</i> " pada <i>navbar</i>	Menampilkan <i>pop up</i> autentikasi berupa <i>form login</i> member	Menampilkan <i>pop up</i> autentikasi berupa <i>form login</i> member	Terpenuhi
2	Mengisi <i>email</i> dan <i>password</i> dengan benar.	Berhasil <i>login</i> dan tampil <i>dashboard member</i> .	Berhasil <i>login</i> dan tampil <i>dashboard member</i> .	Terpenuhi
3	Salah mengisi <i>email</i> dan mengisi <i>password</i>	Tampil <i>toast</i> dengan pesan " <i>These credentials do not match our records</i> "	Tampil <i>toast</i> dengan pesan " <i>These credentials do not match our records</i> "	Terpenuhi

Forgot Password Member				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan tombol " <i>Forgot Password</i> " pada <i>pop up login</i>	Menampilkan <i>form forgot password</i> pada <i>pop up</i> autentikasi	Menampilkan <i>form forgot password</i> pada <i>pop up</i> autentikasi	Terpenuhi
2	Mengetikkan <i>email</i> yang tidak <i>valid</i> .	Memunculkan pesan <i>error</i> di bawah <i>input email</i> dengan pesan " <i>The email must be a valid email address.</i> "	Memunculkan pesan <i>error</i> di bawah <i>input email</i> dengan pesan " <i>The email must be a valid email address.</i> "	Terpenuhi
3	Mengisi <i>email</i> yang <i>valid</i> tetapi belum terdaftar	Tampil <i>toast</i> dengan pesan " <i>User does not exist</i> "	Tampil <i>toast</i> dengan pesan " <i>User does not exist</i> "	Terpenuhi
4	Mengisi <i>email</i> yang <i>valid</i> dan sudah terdaftar	Mengirimkan <i>email</i> (ke <i>mailtrap</i>) untuk <i>reset password</i> serta muncul <i>toast</i> dan memunculkan lagi tampilan <i>form login</i>	Mengirimkan <i>email</i> (ke <i>mailtrap</i>) untuk <i>reset password</i> serta muncul <i>toast</i> dan memunculkan lagi tampilan <i>form login</i>	Terpenuhi

Halaman Utama				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Member belum <i>login</i>	Menampilkan tombol <i>Login</i> pada <i>navbar</i> bagian kanan	Menampilkan tombol <i>Login</i> pada <i>navbar</i> bagian kanan	Terpenuhi
2	Member telah <i>login</i>	Menampilkan foto dan nama <i>member</i> pada <i>navbar</i> bagian kanan	Menampilkan nama member dan belum menampilkan foto (hanya default image kosong) pada <i>navbar</i> bagian kanan	Belum terpenuhi
3	Menekan foto member di <i>navbar</i>	Membuka <i>dashboard member</i>	Membuka <i>dashboard member</i>	Terpenuhi
4	Menekan tombol pada <i>filter</i> kategori pekerjaan	Menampilkan daftar pekerjaan yang tersedia berdasarkan kategori yang dipilih	Menampilkan daftar pekerjaan yang tersedia berdasarkan kategori yang dipilih	Terpenuhi

Halaman Detail Pekerjaan				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Membuka halaman <i>Detail Job</i> bagi <i>member</i> yang belum <i>login</i>	Tidak menampilkan tombol <i>bookmark</i>	Tidak menampilkan tombol <i>bookmark</i>	Terpenuhi
2	Membuka halaman <i>Detail Job</i> bagi <i>member</i> yang telah <i>login</i>	Menampilkan tombol <i>bookmark</i>	Menampilkan tombol <i>bookmark</i>	Terpenuhi
3	Menekan tombol <i>bookmark</i>	Tombol akan berwarna biru serta data pekerjaan yang dipilih akan disimpan, dan jika ditekan kembali, tombol akan kembali berwarna putih serta data akan dikeluarkan dari penyimpanan	Tombol akan berwarna biru serta data pekerjaan yang dipilih akan disimpan, dan jika ditekan kembali, tombol akan kembali berwarna putih serta data akan dikeluarkan dari penyimpanan	Terpenuhi
4	Menekan tombol " <i>Apply Now</i> " bagi <i>member</i> yang belum <i>login</i>	menampilkan <i>pop up</i> untuk autentikasi dan tombol ' <i>Apply directly</i> '	menampilkan <i>pop up</i> untuk autentikasi dan tombol ' <i>Apply directly</i> '	Terpenuhi
5	Menekan tombol " <i>Apply Now</i> " bagi <i>member</i> yang telah <i>login</i>	Menampilkan <i>pop up form</i> untuk melamar pekerjaan berdasarkan pekerjaan yang dipilih	Menampilkan <i>pop up form</i> untuk melamar pekerjaan berdasarkan pekerjaan yang dipilih	Terpenuhi
6	Menekan tombol ' <i>Apply Directly</i> ' pada <i>pop up form</i> autentikasi	Menampilkan <i>pop up form</i> untuk melamar pekerjaan berdasarkan pekerjaan yang dipilih	Menampilkan <i>pop up form</i> untuk melamar pekerjaan berdasarkan pekerjaan yang dipilih	Terpenuhi

Halaman Apply Job				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Membuka halaman <i>Apply Job</i> bagi member yang belum <i>login</i>	Memunculkan <i>form</i> untuk melamar pekerjaan dan beberapa pertanyaan yang harus dijawab (jika ada)	Memunculkan <i>form</i> untuk melamar pekerjaan dan beberapa pertanyaan yang harus dijawab (jika ada)	Terpenuhi
2	Membuka halaman <i>Apply Job</i> bagi member yang telah <i>login</i>	<i>Form</i> untuk melamar pekerjaan dengan <i>input</i> ' <i>name</i> ' dan ' <i>email</i> ' telah terisi berdasarkan data <i>member</i>	<i>Form</i> untuk melamar pekerjaan dengan <i>input</i> ' <i>name</i> ' dan ' <i>email</i> ' telah terisi berdasarkan data <i>member</i>	Terpenuhi

Lampiran 2.2 Dashboard Member

Halaman Umum				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan logo pada <i>sidebar</i>	Masuk ke halaman <i>landing page</i>	Masuk ke halaman <i>landing page</i>	Terpenuhi
2	Menekan <i>Hamburger</i> menu di <i>sidebar</i>	<i>Sidebar</i> akan terbuka seluruhnya ketika keadaan awal <i>sidebar</i> hanya menampilkan ikon saja, dan sebaliknya	<i>Sidebar</i> akan terbuka seluruhnya ketika keadaan awal <i>sidebar</i> hanya menampilkan ikon saja, dan sebaliknya	Terpenuhi
3	Menekan tombol ' <i>Logout</i> ' di <i>sidebar</i>	Memunculkan <i>pop up</i> dialog dengan pertanyaan " <i>Are you sure, you want to log out???</i> " dengan pilihan ' <i>No</i> ' dan ' <i>Logout</i> '	Memunculkan <i>pop up</i> dialog dengan pertanyaan " <i>Are you sure, you want to log out???</i> " dengan pilihan ' <i>No</i> ' dan ' <i>Logout</i> '	Terpenuhi
4	Menekan tombol pilihan pada <i>pop up</i> dialog <i>Logout</i>	Menutup <i>pop up</i> jika tombol ' <i>No</i> ' ditekan dan diarahkan ke <i>landing page</i> jika tombol ' <i>Logout</i> ' ditekan	Menutup <i>pop up</i> jika tombol ' <i>No</i> ' ditekan dan diarahkan ke <i>landing page</i> jika tombol ' <i>Logout</i> ' ditekan	Terpenuhi

Halaman Profile				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan menu ' <i>Profile</i> ' pada <i>sidebar</i>	Menampilkan halaman <i>profile</i> yang jika data diri utama pengguna sudah terisi, akan menampilkan data diri <i>member</i> . jika data diri yang tidak wajib sudah terisi maka akan muncul tanda '-'	Menampilkan halaman <i>profile</i> yang jika data diri utama pengguna sudah terisi, akan menampilkan data diri <i>member</i> . jika data diri yang tidak wajib sudah terisi maka akan muncul tanda '-'	Terpenuhi
2	Menekan menu ' <i>Profile</i> ' pada <i>sidebar</i>	Menampilkan halaman <i>profile</i> yang jika data diri utama belum terisi maka akan muncul <i>pop up</i> dengan pesan ' <i>Please Fill in Your Personal Details</i> ' dan tombol ' <i>OK</i> '	Menampilkan halaman <i>profile</i> yang jika data diri utama belum terisi maka akan muncul <i>pop up</i> dengan pesan ' <i>Please Fill in Your Personal Details</i> ' dan tombol ' <i>OK</i> '	Terpenuhi
3	Menekan tombol ' <i>OK</i> ' pada <i>pop up</i>	membuka halaman <i>Setting</i>	membuka halaman <i>Setting</i>	Terpenuhi

Halaman Job Applied				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan menu ' <i>Job Applied</i> ' pada <i>sidebar</i>	Menampilkan halaman Job Applied dan menampilkan data <i>detail Job</i> yang telah di- <i>apply</i> oleh <i>member</i> tersebut.	Menampilkan halaman Job Applied dan menampilkan data <i>detail Job</i> yang telah di- <i>apply</i> oleh <i>member</i> tersebut.	Terpenuhi
2	Menekan menu ' <i>Job Applied</i> ' pada <i>sidebar</i>	Menampilkan halaman <i>Job Applied</i> yang jika tidak ada pekerjaan yang di- <i>apply</i> , maka akan muncul ikon database dan pesan ' <i>No Items</i> '	Menampilkan halaman <i>Job Applied</i> yang jika tidak ada pekerjaan yang di- <i>apply</i> , maka akan muncul ikon database dan pesan ' <i>No Items</i> '	Terpenuhi
3	Melakukan <i>Sorting</i> berdasarkan status	Menampilkan <i>detail Job</i> untuk pekerjaan yang telah di- <i>apply</i> oleh <i>member</i> tersebut berdasarkan status yang dipilih	Menampilkan <i>detail Job</i> untuk pekerjaan yang telah di- <i>apply</i> oleh <i>member</i> tersebut berdasarkan status yang dipilih	Terpenuhi

Halaman Saved Job				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan menu ' <i>Saved Job</i> ' pada <i>sidebar</i>	Menampilkan halaman <i>Saved Job</i> dan menampilkan data pekerjaan yang telah disimpan oleh <i>member</i> tersebut dalam bentuk tabel dengan tambahan action berupa tombol ikon 'tempat sampah' dan tombol ' <i>Apply</i> '. jika kosong maka akan muncul ikon database dan pesan ' <i>No Items</i> '	Menampilkan halaman <i>Saved Job</i> dan menampilkan data pekerjaan yang telah disimpan oleh <i>member</i> tersebut dalam bentuk tabel dengan tambahan action berupa tombol ikon 'tempat sampah' dan tombol ' <i>Apply</i> '. jika kosong maka akan muncul ikon database dan pesan ' <i>No Items</i> '	Terpenuhi
2	Menekan ikon ' <i>tempat sampah</i> ' pada daftar item di tabel	Menampilkan <i>pop up</i> dengan pesan ' <i>Are you sure, you want to remove this bookmark?</i> ' dan muncul tombol ' <i>No</i> ' dan ' <i>Yes, Remove</i> '. jika tombol ' <i>No</i> ' ditekan maka akan menutup <i>pop up</i> . Jika tombol ' <i>Yes, Remove</i> ' ditekan maka data akan terhapus lalu <i>pop up</i> akan tertutup	Menampilkan <i>pop up</i> dengan pesan ' <i>Are you sure, you want to remove this bookmark?</i> ' dan muncul tombol ' <i>No</i> ' dan ' <i>Yes, Remove</i> '. jika tombol ' <i>No</i> ' ditekan maka akan menutup <i>pop up</i> . Jika tombol ' <i>Yes, Remove</i> ' ditekan maka data akan terhapus lalu <i>pop up</i> akan tertutup	Terpenuhi
3	Menekan tombol ' <i>Apply</i> ' pada daftar item di tabel	Menampilkan halaman <i>detail Job</i> pada <i>front office</i> sesuai dengan pekerjaan yang dipilih	Menampilkan halaman <i>detail Job</i> pada <i>front office</i> sesuai dengan pekerjaan yang dipilih	Terpenuhi

Halaman Help & Support				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan menu 'Help & Support' pada sidebar	Menampilkan halaman <i>Job Applied</i> yang berisi FAQ (<i>Frequently Asked Questions</i>) dan <i>form</i> untuk mengirim <i>email support</i> ke perusahaan	Menampilkan halaman <i>Job Applied</i> yang berisi FAQ (<i>Frequently Asked Questions</i>) dan <i>form</i> untuk mengirim <i>email support</i> ke perusahaan	Terpenuhi
2	Menekan menu 'Help & Support' pada sidebar	Menampilkan <i>email</i> akun pada <i>input email</i>	Menampilkan <i>email</i> akun pada <i>input email</i>	Terpenuhi
3	Menekan pertanyaan pada FAQ	Menampilkan jawaban pertanyaan yang dipilih tepat dibawah pertanyaan dan menutup jawaban pertanyaan lain jika ada jawaban yang muncul, atau menutup jawaban jika jawaban dari pertanyaan yang dipilih telah muncul sebelumnya	Menampilkan jawaban pertanyaan yang dipilih tepat dibawah pertanyaan dan menutup jawaban pertanyaan lain jika ada jawaban yang muncul, atau menutup jawaban jika jawaban dari pertanyaan yang dipilih telah muncul sebelumnya	Terpenuhi
4	Mengisi <i>input name</i> , <i>email</i> , dan <i>question</i> dengan benar dan menekan tombol 'send'	Memunculkan <i>pop up</i> pesan "Email Support Sent Successfully" sebagai tanda <i>email</i> telah berhasil terkirim.	Memunculkan <i>pop up</i> pesan "Email Support Sent Successfully" sebagai tanda <i>email</i> telah berhasil terkirim.	Terpenuhi
5	Mengosongkan <i>input name</i> , <i>email</i> , dan <i>question</i> dan menekan tombol 'send'	Memunculkan pesan error "Please enter a valid name" dibawah <i>input name</i> , pesan error "Please enter a valid email address" dibawah <i>input email</i> , dan pesan error "Please enter a valid question" dibawah <i>input question</i>	Memunculkan pesan error "Please enter a valid name" dibawah <i>input name</i> , pesan error "Please enter a valid email address" dibawah <i>input email</i> , dan pesan error "Please enter a valid question" dibawah <i>input question</i>	Terpenuhi

Halaman Change Password				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan menu 'Change Password' pada sidebar	Menampilkan halaman <i>Change Password</i> dan memunculkan <i>form</i> untuk mengganti <i>password</i> akun.	Menampilkan halaman <i>Change Password</i> dan memunculkan <i>form</i> untuk mengganti <i>password</i> akun.	Terpenuhi
2	Menekan menu 'Change Password' pada sidebar	Menampilkan <i>email</i> akun pada <i>input email</i> yang mana nilainya tidak bisa dirubah	Menampilkan <i>email</i> akun pada <i>input email</i> yang mana nilainya tidak bisa dirubah	Terpenuhi
3	Mengisi <i>input Current Password</i> , <i>New Password</i> dan <i>Confirm New Password</i> dengan benar dan menekan tombol 'Save'	Memunculkan <i>pop up</i> pesan "Password Changed Successfully" sebagai tanda <i>password</i> telah berhasil dirubah	Memunculkan <i>pop up</i> pesan "Password Changed Successfully" sebagai tanda <i>password</i> telah berhasil dirubah	Terpenuhi

Halaman Change Password				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
4	Mengosongkan <i>input Current Password</i> , <i>New Password</i> dan <i>Confirm New Password</i> dan menekan tombol 'Save'	Memunculkan pesan error " <i>Please enter a valid current password</i> " dibawah <i>input current password</i> , pesan error " <i>New password is required</i> " dibawah <i>input New Password</i> , dan pesan error " <i>Confirmation password is required</i> " dibawah <i>input Confirmation Password</i>	Memunculkan pesan error " <i>Please enter a valid current password</i> " dibawah <i>input current password</i> , pesan error " <i>New password is required</i> " dibawah <i>input New Password</i> , dan pesan error " <i>Confirmation password is required</i> " dibawah <i>input Confirmation Password</i>	Terpenuhi
5	Mengisi input <i>Current Password</i> tidak sesuai dengan password yang ada dan menekan tombol 'Save'	Memunculkan pesan error " <i>Invalid Password</i> " dibawah <i>input current password</i>	Memunculkan pesan error " <i>Invalid Password</i> " dibawah <i>input current password</i>	Terpenuhi
6	Mengisi <i>input New Password</i> dan <i>confirm New Password</i> dengan nilai yang berbeda dan menekan tombol 'Save'	Memunculkan pesan error " <i>The confirm password and new password must match.</i> " dibawah <i>input Confirmation Password</i>	Memunculkan pesan error " <i>The confirm password and new password must match.</i> " dibawah <i>input Confirmation Password</i>	Terpenuhi

Halaman Setting				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan menu 'Setting' pada <i>sidebar</i>	Menampilkan halaman <i>Setting</i> dan memunculkan <i>form</i> untuk mengubah dan menambahkan data pengguna	Menampilkan halaman <i>Setting</i> dan memunculkan <i>form</i> untuk mengubah dan menambahkan data pengguna	Terpenuhi
2	Menekan menu 'Setting' pada <i>sidebar</i>	Menampilkan data pengguna pada setiap <i>input</i> berdasarkan data yang telah ada, ditambahkan, atau diubah sebelumnya	Menampilkan data pengguna pada setiap <i>input</i> berdasarkan data yang telah ada, ditambahkan, atau diubah sebelumnya	Terpenuhi
3	Mengisi input data pada form dengan benar dan menekan tombol 'Save'	Memunculkan <i>pop up</i> pesan " <i>Changes Saved Successfully!</i> " sebagai tanda data pengguna telah berhasil ditambah atau diubah	Memunculkan <i>pop up</i> pesan " <i>Changes Saved Successfully!</i> " sebagai tanda data pengguna telah berhasil ditambah atau diubah	Terpenuhi
4	Mengosongkan <i>input</i> pada bidang yang diwajibkan ataupun salah <i>input</i> nilai pada bidang yang bersifat <i>optional</i> dan wajib serta menekan tombol 'Save'	Menampilkan pesan error dibawah <i>input</i> yang memiliki nilai yang salah	Menampilkan pesan error dibawah <i>input</i> yang memiliki nilai yang salah	Terpenuhi

Lampiran 2.3 Dashboard Author

Halaman Umum				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan logo pada <i>sidebar</i>	Masuk ke halaman <i>landing page</i>	Masuk ke halaman <i>landing page</i>	Terpenuhi
2	Menekan <i>Hamburger</i> menu di <i>sidebar</i>	<i>Sidebar</i> akan terbuka seluruhnya ketika keadaan awal <i>sidebar</i> hanya menampilkan ikon saja, dan sebaliknya	<i>Sidebar</i> akan terbuka seluruhnya ketika keadaan awal <i>sidebar</i> hanya menampilkan ikon saja, dan sebaliknya	Terpenuhi

Fitur Author				
Author				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan menu ' <i>Author</i> ' pada <i>sidebar</i>	Menampilkan halaman ' <i>Author</i> ' yang berisi informasi umum perusahaan dan juga informasi total pelamar,dll	Menampilkan halaman ' <i>Author</i> ' yang berisi informasi umum perusahaan dan juga informasi total pelamar,dll	Terpenuhi

Fitur Job Listing				
Job Listing				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan menu ' <i>Job Listing</i> ' pada <i>sidebar</i>	Menampilkan halaman ' <i>Job Listing</i> ' berisi tabel yang menampilkan data pekerjaan yang disediakan perusahaan	Menampilkan halaman ' <i>Job Listing</i> ' berisi tabel yang menampilkan data pekerjaan yang disediakan perusahaan	Terpenuhi
2	Memilih maksimal jumlah data yang ditampilkan antara 10, 20, 50, dan 100	Menampilkan data pekerjaan yang disediakan perusahaan dengan maksimal n data perhalaman	Menampilkan data pekerjaan yang disediakan perusahaan dengan maksimal n data perhalaman	Terpenuhi
3	Menekan daftar halaman atau menekan tombol panah sebelum atau sesudah	Menampilkan data pekerjaan yang disediakan perusahaan yang berada pada halaman yang dipilih	Menampilkan data pekerjaan yang disediakan perusahaan yang berada pada halaman yang dipilih	Terpenuhi
4	Menekan <i>toggle</i> pada tabel daftar pekerjaan	<i>toggle</i> berwarna hijau jika sebelumnya <i>toggle</i> berwarna merah yang menandakan status publish berubah dari ' <i>true</i> ' menjadi ' <i>false</i> ' yang menandakan bahwa lowongan tersebut tidak aktif dan sebaliknya. keduanya menampilkan notifikasi yang berisi pesan ' <i>Post successfully updated</i> '	<i>toggle</i> berwarna hijau jika sebelumnya <i>toggle</i> berwarna merah yang menandakan status publish berubah dari ' <i>true</i> ' menjadi ' <i>false</i> ' yang menandakan bahwa lowongan tersebut tidak aktif dan sebaliknya. keduanya menampilkan notifikasi yang berisi pesan ' <i>Post successfully updated</i> '	Terpenuhi

Job Listing				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
5	Menekan ikon ' <i>pencil</i> ' pada tabel daftar pekerjaan	Membuka halaman ' <i>Edit Job</i> ' dan menampilkan data pekerjaan yang ingin diubah pada setiap <i>input</i> berdasarkan data yang telah ada, ditambahkan, atau diubah sebelumnya	Membuka halaman ' <i>Edit Job</i> ' dan menampilkan data pekerjaan yang ingin diubah pada setiap <i>input</i> berdasarkan data yang telah ada, ditambahkan, atau diubah sebelumnya	Terpenuhi
6	Menekan tombol ' <i>Create a Job</i> ' pada tabel daftar pekerjaan	Menampilkan halaman ' <i>Create Job</i> ' yang berisi <i>form</i> untuk menambahkan pekerjaan	Menampilkan halaman ' <i>Create Job</i> ' yang berisi <i>form</i> untuk menambahkan pekerjaan	Terpenuhi
Edit Job				
1	Mengisi input data pada form dengan benar dan menekan tombol ' <i>Update Job</i> '	Memunculkan <i>pop up</i> pesan " <i>Job Updated Successfully!</i> " sebagai tanda data pekerjaan telah berhasil ditambah atau diubah dan mengarahkan ke halaman ' <i>Job Listing</i> '	Memunculkan <i>pop up</i> pesan " <i>Job Updated Successfully!</i> " sebagai tanda data pekerjaan telah berhasil ditambah atau diubah dan mengarahkan ke halaman ' <i>Job Listing</i> '	Terpenuhi
2	Mengosongkan <i>input</i> pada bidang yang diwajibkan ataupun salah <i>input</i> nilai pada bidang yang bersifat <i>optional</i> dan wajib serta menekan tombol ' <i>Update Job</i> '	Menampilkan pesan <i>error</i> dibawah <i>input</i> yang memiliki nilai yang salah atau kosong	Menampilkan pesan <i>error</i> dibawah <i>input</i> yang memiliki nilai yang salah atau kosong	Terpenuhi
3	Menekan tombol ' <i>Delete Job</i> '	Memunculkan <i>pop up</i> dialog dengan pertanyaan " <i>Are you sure, you want to delete this blog???</i> " dengan pilihan ' <i>No</i> ' dan ' <i>Yes, Remove</i> '.	Memunculkan <i>pop up</i> dialog dengan pertanyaan " <i>Are you sure, you want to delete this blog???</i> " dengan pilihan ' <i>No</i> ' dan ' <i>Yes, Remove</i> '	Terpenuhi
4	Menekan tombol pilihan pada <i>pop up</i> dialog ' <i>Delete Job</i> '	Menutup <i>pop up</i> jika tombol ' <i>No</i> ' ditekan dan memunculkan <i>pop up</i> pesan " <i>Job Deleted Successfully</i> " sebagai tanda berhasil menghapus pekerjaan dan mengarahkan ke halaman ' <i>Job Listing</i> ' jika tombol ' <i>Yes, Remove</i> ' ditekan	Menutup <i>pop up</i> jika tombol ' <i>No</i> ' ditekan dan memunculkan <i>pop up</i> pesan " <i>Job Deleted Successfully</i> " sebagai tanda berhasil menghapus pekerjaan dan mengarahkan ke halaman ' <i>Job Listing</i> ' jika tombol ' <i>Yes, Remove</i> ' ditekan	Terpenuhi

Create Job				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Mengisi input data pada form dengan benar dan menekan tombol 'Create a Job'	Memunculkan <i>pop up</i> pesan "Job Created Successfully!" sebagai tanda berhasil menambahkan pekerjaan dan mengarahkan ke halaman 'Job Listing'	Memunculkan <i>pop up</i> pesan "Job Created Successfully!" sebagai tanda berhasil menambahkan pekerjaan dan mengarahkan ke halaman 'Job Listing'	Terpenuhi
2	Mengosongkan <i>input</i> pada bidang yang diwajibkan ataupun salah <i>input</i> nilai pada bidang yang bersifat <i>optional</i> dan wajib serta menekan tombol 'Create a Job'	Menampilkan pesan <i>error</i> dibawah <i>input</i> yang memiliki nilai yang salah atau kosong	Menampilkan pesan <i>error</i> dibawah <i>input</i> yang memiliki nilai yang salah atau kosong	Terpenuhi

Fitur Job Available				
Job Availbale				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan menu 'Job Available' pada <i>sidebar</i>	Menampilkan halaman 'Job Availbale' yang berisi tabel yang menampilkan data pekerjaan yang telah di- <i>publish</i>	Menampilkan halaman 'Job Availbale' yang berisi tabel yang menampilkan data pekerjaan yang telah di- <i>publish</i>	Terpenuhi
2	Mencari data menggunakan input <i>search</i>	Menampilkan data pekerjaan yang telah di- <i>publish</i> dengan 'Job Title' sesuai inputan pengguna	Menampilkan data pekerjaan yang telah di- <i>publish</i> dengan 'Job Title' sesuai inputan pengguna	Terpenuhi
3	Memilih maksimal jumlah data yang ditampilkan antara 10, 20, 50, dan 100	Menampilkan data pekerjaan yang telah di- <i>publish</i> dengan maksimal <i>n</i> data perhalaman	Menampilkan data pekerjaan yang telah di- <i>publish</i> dengan maksimal <i>n</i> data perhalaman	Terpenuhi
4	Menekan ikon 'eye' pada tabel daftar pekerjaan yang di- <i>publish</i>	Membuka halaman 'Detail Job' dan menampilkan detail data pekerjaan yang dipilih	Membuka halaman 'Detail Job' dan menampilkan detail data pekerjaan yang dipilih	Terpenuhi
5	Menekan ikon 'arrow' dengan tulisan 'back' pada halaman 'Detail Job'	Kembali halaman 'Job Available'	Kembali halaman 'Job Available'	Terpenuhi

Fitur Job Applicants				
Job Applicants				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan menu ' <i>Job Applicants</i> ' pada <i>sidebar</i>	Menampilkan halaman ' <i>Job Applicants</i> ' yang berisi tabel yang menampilkan data pelamar pekerjaan dengan <i>default status 'Applied'</i> . jika kosong maka akan muncul ikon database dan pesan ' <i>No Items</i> '	Menampilkan halaman ' <i>Job Applicants</i> ' yang berisi tabel yang menampilkan data pelamar pekerjaan dengan <i>default status 'Applied'</i> . jika kosong maka akan muncul ikon database dan pesan ' <i>No Items</i> '	Terpenuhi
2	Melakukan <i>Sorting</i> berdasarkan <i>step</i> proses lamaran	Menampilkan data pelamar pekerjaan berdasarkan <i>step</i> yang dipilih. jika kosong maka akan muncul ikon database dan pesan ' <i>No Items</i> '	Menampilkan data pelamar pekerjaan berdasarkan <i>step</i> yang dipilih. jika kosong maka akan muncul ikon database dan pesan ' <i>No Items</i> '	Terpenuhi
4	Mencari data menggunakan input <i>search</i>	Menampilkan data pelamar pekerjaan dengan ' <i>Name</i> ' sesuai <i>input</i> -an pengguna	Menampilkan data pelamar pekerjaan dengan ' <i>Name</i> ' sesuai <i>input</i> -an pengguna	Terpenuhi
5	Memilih maksimal jumlah data yang ditampilkan antara 10, 20, 50, dan 100	Menampilkan data pelamar pekerjaan dengan maksimal <i>n</i> data perhalaman	Menampilkan data pelamar pekerjaan dengan maksimal <i>n</i> data perhalaman	Terpenuhi
6	Saat menampilkan data dengan <i>step 'Process'</i> , Memilih filter yang tersedia	Menampilkan data pelamar pekerjaan yang statusnya sesuai dengan pilihan	Menampilkan data pelamar pekerjaan yang statusnya sesuai dengan pilihan	Terpenuhi
7	Menekan daftar halaman atau menekan tombol panah sebelum atau sesudah	Menampilkan data pelamar pekerjaan yang berada pada halaman yang dipilih	Menampilkan data pelamar pekerjaan yang berada pada halaman yang dipilih	Terpenuhi
8	Menekan tombol ' <i>CV</i> ' pada daftar <i>item</i> di table	Membuka <i>file</i> CV milik pelamar di <i>tab</i> baru	Membuka <i>file</i> CV milik pelamar di <i>tab</i> baru	Terpenuhi
9	Menekan tombol ' <i>Portfolio</i> ' pada daftar <i>item</i> di table	Membuka <i>file</i> Portofolio milik pelamar di <i>tab</i> baru	Membuka <i>file</i> Portofolio milik pelamar di <i>tab</i> baru	Terpenuhi
10	Menekan ikon ' <i>tempat sampah</i> ' pada daftar item di tabel	Memunculkan <i>pop up</i> dialog dengan pertanyaan " <i>Are you sure, you want to remove this applicants???</i> " dengan pilihan ' <i>No</i> ' dan ' <i>Yes, Remove</i> '	Memunculkan <i>pop up</i> dialog dengan pertanyaan " <i>Are you sure, you want to remove this applicants???</i> " dengan pilihan ' <i>No</i> ' dan ' <i>Yes, Remove</i> '	Terpenuhi

Job Applicants				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
11	Menekan tombol pilihan pada <i>pop up</i> dialog ' <i>Delete applicants</i> '	Menutup <i>pop up</i> jika tombol ' <i>No</i> ' ditekan dan memunculkan <i>pop up</i> pesan " <i>Job applicant successfully deleted</i> " sebagai tanda berhasil menghapus pelamar pekerjaan jika tombol ' <i>Yes, Remove</i> ' ditekan	Menutup <i>pop up</i> jika tombol ' <i>No</i> ' ditekan dan memunculkan <i>pop up</i> pesan " <i>Job applicant successfully deleted</i> " sebagai tanda berhasil menghapus pelamar pekerjaan jika tombol ' <i>Yes, Remove</i> ' ditekan	Terpenuhi
12	Menekan ikon ' <i>arrow</i> ' pada daftar item di tabel	Membuka <i>pop up form invitaton</i> dengan nilai <i>input email, applied position</i> dan <i>phone</i> telah terisi berdasarkan data dari pelamar dan bersifat <i>readonly</i> . dan tambahan berupa <i>dropdown</i> untuk memilih step selanjutnya	Membuka <i>pop up form invitaton</i> dengan nilai <i>input email, applied position</i> dan <i>phone</i> telah terisi berdasarkan data dari pelamar dan bersifat <i>readonly</i> . dan tambahan berupa <i>dropdown</i> untuk memilih step selanjutnya	Terpenuhi
13	Menekan ikon ' <i>x</i> ' pada <i>pop up form</i> atau diluar <i>pop up</i>	Menutup <i>pop up form</i>	Menutup <i>pop up form</i>	Terpenuhi
14	Mengganti <i>step</i> pada <i>form</i>	Terbuka <i>input</i> tambahan lain yang berbeda berdasarkan <i>step</i> yang dipilih	Terbuka <i>input</i> tambahan lain yang berbeda berdasarkan <i>step</i> yang dipilih	Terpenuhi
15	Mengosongkan <i>input</i> tambahan setelah memilih <i>step</i> atau memasukkan nilai yang tidak sesuai, lalu menekan tombol ' <i>Submit</i> '	Menampilkan pesan <i>error</i> dibawah <i>input</i> yang memiliki nilai yang salah atau kosong	Menampilkan pesan <i>error</i> dibawah <i>input</i> yang memiliki nilai yang salah atau kosong	Terpenuhi
16	Mengisi input data pada form dengan benar dan menekan tombol ' <i>Submit</i> '	Menutup <i>pop up form</i> sebagai tanda berhasil dan mengirimkan <i>email</i> ke pelamar pekerjaan berdasarkan <i>step</i> yang dipilih	Menutup <i>pop up form</i> sebagai tanda berhasil dan mengirimkan <i>email</i> ke pelamar pekerjaan berdasarkan <i>step</i> yang dipilih	Terpenuhi

Fitur Internship				
Internship				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan menu ' <i>Internship</i> ' pada <i>sidebar</i>	Menampilkan halaman ' <i>Internship</i> ' yang berisi tabel yang menampilkan data pelamar magang, jika kosong maka akan muncul ikon database dan pesan ' <i>No Items</i> '	Menampilkan halaman ' <i>Internship</i> ' yang berisi tabel yang menampilkan data pelamar magang, jika kosong maka akan muncul ikon database dan pesan ' <i>No Items</i> '	Terpenuhi
2	Memilih maksimal jumlah data yang ditampilkan antara 10, 20, 50, dan 100	Menampilkan data pelamar magang dengan maksimal <i>n</i> data perhalaman	Menampilkan data pelamar magang dengan maksimal <i>n</i> data perhalaman	Terpenuhi
3	Memilih nilai dari <i>dropdown</i> pada status di item tabel	Muncul notifikasi dengan pesan ' <i>Internship successfully update</i> ', sebagai tanda status pelamar magang telah berubah	Muncul notifikasi dengan pesan ' <i>Internship successfully update</i> ', sebagai tanda status pelamar magang telah berubah	Terpenuhi
4	Menekan ikon ' <i>tempat sampah</i> ' pada daftar item di tabel	Memunculkan <i>pop up</i> dialog dengan pertanyaan " <i>Are you sure, you want to remove this applicants???</i> " dengan pilihan ' <i>No</i> ' dan ' <i>Yes, Remove</i> '	Memunculkan <i>pop up</i> dialog dengan pertanyaan " <i>Are you sure, you want to remove this applicants???</i> " dengan pilihan ' <i>No</i> ' dan ' <i>Yes, Remove</i> '	Terpenuhi
5	Menekan tombol pilihan pada <i>pop up</i> dialog ' <i>Delete internship</i> '	Menutup <i>pop up</i> jika tombol ' <i>No</i> ' ditekan dan memunculkan <i>pop up</i> pesan " <i>Internship successfully deleted</i> " sebagai tanda berhasil menghapus data pelamar magang jika tombol ' <i>Yes, Remove</i> ' ditekan	Menutup <i>pop up</i> jika tombol ' <i>No</i> ' ditekan dan memunculkan <i>pop up</i> pesan " <i>Internship successfully deleted</i> " sebagai tanda berhasil menghapus data pelamar magang jika tombol ' <i>Yes, Remove</i> ' ditekan	Terpenuhi
6	Menekan ikon ' <i>eye</i> ' pada tabel daftar pekerjaan yang di- <i>publish</i>	Membuka <i>pop up</i> ' <i>Detail pelamar magang</i> ' dan menampilkan detail data pelamar yang dipilih	Membuka <i>pop up</i> ' <i>Detail pelamar magang</i> ' dan menampilkan detail data pelamar yang dipilih	Terpenuhi
7	Menekan tombol ' <i>Letter</i> ' pada <i>pop up</i> ' <i>Detail pelamar magang</i> '	Membuka <i>file Letter</i> milik pelamar di <i>tab</i> baru	Membuka <i>file Letter</i> milik pelamar di <i>tab</i> baru	Terpenuhi
8	Menekan tombol ' <i>CV/Portfolio</i> ' pada daftar <i>item</i> di tabel	Membuka <i>file CV/Portofolio</i> milik pelamar di <i>tab</i> baru	Membuka <i>file CV/Portofolio</i> milik pelamar di <i>tab</i> baru	Terpenuhi

Fitur Questions				
Questions				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan menu 'Questions' pada sidebar	Menampilkan halaman 'Questions' yang berisi tabel yang menampilkan data pekerjaan dan jumlah pertanyaan yang telah dibuat. jika kosong, maka akan muncul ikon database dan pesan 'No Items'	Menampilkan halaman 'Questions' yang berisi tabel yang menampilkan data pekerjaan dan jumlah pertanyaan yang telah dibuat. jika kosong, maka akan muncul ikon database dan pesan 'No Items'	Terpenuhi
2	Memilih maksimal jumlah data yang ditampilkan antara 10, 20, 50, dan 100	Menampilkan data pekerjaan dan jumlah pertanyaan yang telah dibuat dengan maksimal n data perhalaman	Menampilkan data pekerjaan dan jumlah pertanyaan yang telah dibuat dengan maksimal n data perhalaman	Terpenuhi
3	Menekan ikon 'eye' di item pada tabel	Menampilkan halaman 'Detail Questions' yang berisi tabel untuk menampilkan daftar pertanyaan setiap pekerjaan untuk pelamar	Menampilkan halaman 'Detail Questions' yang berisi tabel untuk menampilkan daftar pertanyaan setiap pekerjaan untuk pelamar	Terpenuhi
4	Menekan toggle pada tabel daftar pekerjaan	toggle berwarna hijau jika sebelumnya toggle berwarna merah yang menandakan status publish berubah dari 'true' menjadi 'false' yang membuat pertanyaan tidak muncul saat mengisi form lamaran dan sebaliknya. keduanya menampilkan notifikasi yang berisi pesan 'Question status successfully updated'.	toggle berwarna hijau jika sebelumnya toggle berwarna merah yang menandakan status publish berubah dari 'true' menjadi 'false' yang membuat pertanyaan tidak muncul saat mengisi form lamaran dan sebaliknya. keduanya menampilkan notifikasi yang berisi pesan 'Question status successfully updated'	Terpenuhi
5	Menekan ikon 'tempat sampah' pada daftar item di tabel	Memunculkan pop up dialog dengan pertanyaan "Are you sure, you want to delete this???" dengan pilihan 'No' dan 'Yes, Remove'	Memunculkan pop up dialog dengan pertanyaan "Are you sure, you want to delete this???" dengan pilihan 'No' dan 'Yes, Remove'	Terpenuhi
6	Menekan tombol pilihan pada pop up dialog 'Delete Question'	Menutup pop up jika tombol 'No' ditekan dan memunculkan pop up pesan "All question for the job successfully deleted" sebagai tanda berhasil menghapus pertanyaan pada pekerjaan terkait jika tombol 'Yes, Remove' ditekan	Menutup pop up jika tombol 'No' ditekan dan memunculkan pop up pesan "All question for the job successfully deleted" sebagai tanda berhasil menghapus pertanyaan pada pekerjaan terkait jika tombol 'Yes, Remove' ditekan	Terpenuhi

Detail Questions				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan menu ' <i>Detail Questions</i> ' pada <i>sidebar</i>	Menampilkan halaman ' <i>Detail Questions</i> ' yang berisi tabel yang menampilkan daftar pertanyaan dari pekerjaan yang dipilih. jika kosong, maka akan muncul ikon database dan pesan ' <i>No Items</i> '	Menampilkan halaman ' <i>Detail Questions</i> ' yang berisi tabel yang menampilkan daftar pertanyaan dari pekerjaan yang dipilih. jika kosong, maka akan muncul ikon database dan pesan ' <i>No Items</i> '	Terpenuhi
2	Menekan tombol 'Create New Question' pada <i>sidebar</i>	Menampilkan halaman ' <i>Create New Question</i> ' yang berisi <i>form</i> untuk menambah pertanyaan pada pekerjaan terkait	Menampilkan halaman ' <i>Create New Question</i> ' yang berisi <i>form</i> untuk menambah pertanyaan pada pekerjaan terkait	Terpenuhi
3	Menekan ikon ' <i>pencil</i> ' di <i>item</i> pada tabel	Menampilkan halaman ' <i>Edit Question</i> ' yang berisi <i>form</i> untuk merubah pertanyaan pada pekerjaan terkait	Menampilkan halaman ' <i>Edit Question</i> ' yang berisi <i>form</i> untuk merubah pertanyaan pada pekerjaan terkait	Terpenuhi
4	Menekan ikon ' <i>tempat sampah</i> ' pada daftar item di tabel	Memunculkan <i>pop up</i> dialog dengan pertanyaan " <i>Are you sure, you want to delete this???</i> " dengan pilihan ' <i>No</i> ' dan ' <i>Yes, Remove</i> '	Memunculkan <i>pop up</i> dialog dengan pertanyaan " <i>Are you sure, you want to delete this???</i> " dengan pilihan ' <i>No</i> ' dan ' <i>Yes, Remove</i> '	Terpenuhi
5	Menekan tombol pilihan pada <i>pop up</i> dialog ' <i>Delete Question</i> '	Menutup <i>pop up</i> jika tombol ' <i>No</i> ' ditekan dan memunculkan <i>pop up</i> pesan " <i>Question successfully deleted</i> sebagai tanda berhasil menghapus pertanyaan pada pekerjaan yang dipilih jika tombol ' <i>Yes, Remove</i> ' ditekan	Menutup <i>pop up</i> jika tombol ' <i>No</i> ' ditekan dan memunculkan <i>pop up</i> pesan " <i>Question successfully deleted</i> sebagai tanda berhasil menghapus pertanyaan pada pekerjaan yang dipilih jika tombol ' <i>Yes, Remove</i> ' ditekan	Terpenuhi
Create New Question				
1	Menekan ikon ' <i>arrow</i> ' dengan tulisan ' <i>back</i> ' pada halaman ' <i>Create New Question</i> '	Kembali halaman ' <i>Detail Questions</i> '	Kembali halaman ' <i>Detail Questions</i> '	Terpenuhi
2	Menekan tombol ' <i>Add Question</i> '	Menambah <i>form</i> untuk pertanyaan yang akan dibuat	Menambah <i>form</i> untuk pertanyaan yang akan dibuat	Terpenuhi
3	Menekan ikon ' <i>tempat sampah</i> ' di bawah setiap <i>form</i>	Menghapus <i>form</i> tambah pertanyaan terkait. jika <i>form</i> hanya satu maka tombol akan <i>disable</i>	Menghapus <i>form</i> tambah pertanyaan terkait. jika <i>form</i> hanya satu maka tombol akan <i>disable</i>	Terpenuhi
4	Mengisi <i>input</i> data pada <i>form</i> dengan benar dan menekan tombol ' <i>save</i> '	Memunculkan <i>pop up</i> pesan " <i>Question Created Successfully!</i> " sebagai tanda pertanyaan baru berhasil dibuat dan mengarahkan ke halaman ' <i>Detail Question</i> '	Memunculkan <i>pop up</i> pesan " <i>Question Created Successfully!</i> " sebagai tanda pertanyaan baru berhasil dibuat dan mengarahkan ke halaman ' <i>Detail Question</i> '	Terpenuhi
5	Mengosongkan <i>input</i> , lalu menekan tombol ' <i>Save</i> '	Memunculkan notifikasi ' <i>Both question and answer fields must be filled in.</i> '	Memunculkan notifikasi ' <i>Both question and answer fields must be filled in.</i> '	Terpenuhi

No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
Edit Question				
1	Menekan ikon 'arrow' dengan tulisan 'back' pada halaman 'Edit Question'	Kembali halaman 'Detail Questions'	Kembali halaman 'Detail Questions'	Terpenuhi
2	Merubah <i>input</i> data pada form dengan benar dan menekan tombol 'Update'	Memunculkan <i>pop up</i> pesan "Question Updated Successfully" sebagai tanda pertanyaan telah berhasil dirubah dan mengarahkan ke halaman 'Detail Question'	Memunculkan <i>pop up</i> pesan "Question Updated Successfully" sebagai tanda pertanyaan telah berhasil dirubah dan mengarahkan ke halaman 'Detail Question'	Terpenuhi

Fitur Account				
Account				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan menu 'Account' pada sidebar	Menampilkan halaman 'Account' yang berisi data diri pengguna	Menampilkan halaman 'Account' yang berisi data diri pengguna	Terpenuhi
2	Menekan ikon 'pencil' pada tabel daftar pekerjaan	Membuka halaman 'Edit Account' dan menampilkan data pengguna yang ingin diubah pada setiap <i>input</i> berdasarkan data yang telah ada, ditambahkan, atau diubah sebelumnya	Membuka halaman 'Edit Account' dan menampilkan data pengguna yang ingin diubah pada setiap <i>input</i> berdasarkan data yang telah ada, ditambahkan, atau diubah sebelumnya	Terpenuhi
3	Menekan tombol 'Change Password'	Menampilkan halaman <i>Change Password</i> dan memunculkan <i>form</i> untuk mengganti <i>password</i> akun. Nilai <i>input email</i> merupakan <i>email</i> dari pengguna yang bersifat <i>readonly</i>	Menampilkan halaman <i>Change Password</i> dan memunculkan <i>form</i> untuk mengganti <i>password</i> akun. Nilai <i>input email</i> merupakan <i>email</i> dari pengguna yang bersifat <i>readonly</i>	Terpenuhi
4	Menekan tombol 'Delete Account'	Memunculkan <i>pop up</i> dialog dengan pertanyaan "Are you sure, you want to delete this account???" dengan pilihan 'No' dan 'Yes, Delete'.	Memunculkan <i>pop up</i> dialog dengan pertanyaan "Are you sure, you want to delete this account???" dengan pilihan 'No' dan 'Yes, Delete'	Terpenuhi
5	Menekan tombol pilihan pada <i>pop up</i> dialog 'Delete Account'	Menutup <i>pop up</i> jika tombol 'No' ditekan dan memunculkan <i>pop up</i> pesan "Account successfully deleted" sebagai tanda berhasil menghapus akun dan mengarahkan ke <i>landing page</i> jika tombol 'Yes, Delete' ditekan.	Menutup <i>pop up</i> jika tombol 'No' ditekan dan memunculkan <i>pop up</i> pesan "Account successfully deleted" sebagai tanda berhasil menghapus akun dan mengarahkan ke <i>landing page</i> jika tombol 'Yes, Delete' ditekan	Terpenuhi

Account				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
6	Menekan tombol 'Logout'	Memunculkan <i>pop up</i> dialog dengan pertanyaan "Are you sure, you want to log out???" dengan pilihan 'No' dan 'Yes, Logout'	Memunculkan <i>pop up</i> dialog dengan pertanyaan "Are you sure, you want to log out???" dengan pilihan 'No' dan 'Yes, Logout'	Terpenuhi
7	Menekan tombol pilihan pada <i>pop up</i> dialog 'Logout'	Menutup <i>pop up</i> jika tombol 'No' ditekan dan membuka halaman login untuk <i>author</i> jika tombol 'Yes, Logout' ditekan	Menutup <i>pop up</i> jika tombol 'No' ditekan dan membuka halaman login untuk <i>author</i> jika tombol 'Yes, Logout' ditekan	Terpenuhi
Edit Account				
1	Menekan tombol 'Cancel'	Kembali halaman 'Account'	Kembali halaman 'Account'	Terpenuhi
2	Mengisi atau mengganti nilai input data pada form dengan benar dan menekan tombol 'Submit'	Memunculkan <i>pop up</i> pesan "Changes Saved Successfully!" sebagai tanda berhasil merubah atau menambah data pengguna dan mengarahkan ke halaman 'Account'	Memunculkan <i>pop up</i> pesan "Changes Saved Successfully!" sebagai tanda berhasil merubah atau menambah data pengguna dan mengarahkan ke halaman 'Account'	Terpenuhi
3	Mengosongkan <i>input</i> atau mengganti nilai dengan nilai yang tidak sesuai, lalu menekan tombol 'Save Changes'	Menampilkan pesan <i>error</i> dibawah <i>input</i> yang memiliki nilai yang salah atau kosong	Menampilkan pesan <i>error</i> dibawah <i>input</i> yang memiliki nilai yang salah atau kosong	Terpenuhi
Change Password				
1	Menekan tombol 'Cancel'	Kembali halaman 'Account'	Kembali halaman 'Account'	Terpenuhi
2	Mengisi <i>input Current Password</i> , <i>New Password</i> dan <i>Confirm New Password</i> dengan benar dan menekan tombol 'Change Password'	Memunculkan <i>pop up</i> pesan "Password Changed Successfully" sebagai tanda <i>password</i> telah berhasil dirubah	Memunculkan <i>pop up</i> pesan "Password Changed Successfully" sebagai tanda <i>password</i> telah berhasil dirubah	Terpenuhi
3	Mengosongkan <i>input Current Password</i> , <i>New Password</i> dan <i>Confirm New Password</i> dan menekan tombol 'Change Password'	Memunculkan pesan <i>error</i> "Please enter a valid current password" dibawah <i>input current password</i> , pesan <i>error</i> "New password is required" dibawah <i>input New Password</i> , dan pesan <i>error</i> "Confirmation password is required" dibawah <i>input Confirmation Password</i>	Memunculkan pesan <i>error</i> "Please enter a valid current password" dibawah <i>input current password</i> , pesan <i>error</i> "New password is required" dibawah <i>input New Password</i> , dan pesan <i>error</i> "Confirmation password is required" dibawah <i>input Confirmation Password</i>	

Change Password				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
4	Mengisi <i>input Current Password</i> tidak sesuai dengan <i>password</i> yang ada dan menekan tombol ' <i>Change Password</i> '	Memunculkan pesan <i>error "Invalid Password"</i> dibawah <i>input current password</i>	Memunculkan pesan <i>error "Invalid Password"</i> dibawah <i>input current password</i>	Terpenuhi
5	Mengisi <i>input confirm/New Password</i> dengan nilai yang berbeda dan menekan tombol ' <i>Change Password</i> '	Memunculkan pesan <i>error "The confirm password and new password must match."</i> dibawah <i>input Confirmation Password</i>	Memunculkan pesan <i>error "The confirm password and new password must match."</i> dibawah <i>input Confirmation Password</i>	Terpenuhi

Lampiran 2.1 Dashboard Admin

Halaman Umum				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan logo pada <i>sidebar</i>	Masuk ke halaman <i>landing page</i>	Masuk ke halaman <i>landing page</i>	Terpenuhi
2	Menekan <i>Hamburger</i> menu di <i>sidebar</i>	<i>Sidebar</i> akan terbuka seluruhnya ketika keadaan awal <i>sidebar</i> hanya menampilkan ikon saja, dan sebaliknya	<i>Sidebar</i> akan terbuka seluruhnya ketika keadaan awal <i>sidebar</i> hanya menampilkan ikon saja, dan sebaliknya	Terpenuhi

Halaman Admin				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan menu ' <i>Admin</i> ' pada <i>sidebar</i>	Menampilkan halaman ' <i>Admin</i> ' yang berisi informasi umum perusahaan dan juga informasi total pelamar,dll	Menampilkan halaman ' <i>Admin</i> ' yang berisi informasi umum perusahaan dan juga informasi total pelamar,dll	Terpenuhi
2	Menekan tombol ' <i>Edit Company</i> '	Menampilkan halaman ' <i>Edit Company</i> ' yang berisi <i>form</i> untuk mengubah atau menambahkan data perusahaan	Menampilkan halaman ' <i>Edit Company</i> ' yang berisi <i>form</i> untuk mengubah atau menambahkan data perusahaan	Terpenuhi
3	Menekan tombol ' <i>Edit Company</i> '	Menampilkan <i>Company Title</i> , <i>category</i> , dan <i>description</i> pada masing-masing input <i>Company Title</i> , <i>category</i> , dan <i>description</i>	Menampilkan <i>Company Title</i> , <i>category</i> , dan <i>description</i> pada masing-masing input <i>Company Title</i> , <i>category</i> , dan <i>description</i>	Terpenuhi
4	Mengisi input dan Menekan tombol ' <i>Update</i> '	Menampilkan <i>pop up</i> pesan ' <i>Company Updated Successfully</i> ' yang meandakan data perusahaan berhasil berubah dan diarahkan ke halaman admin	Menampilkan <i>pop up</i> pesan ' <i>Company Updated Successfully</i> ' yang meandakan data perusahaan berhasil berubah dan diarahkan ke halaman admin	Terpenuhi

Halaman Member				
No	Kasus Pengujian	Keluaran yang diharapkan	Keluaran yang didapatkan	Hasil Pengujian
1	Menekan menu <i>'Member'</i> pada <i>sidebar</i>	Menampilkan halaman <i>'Member'</i> yang berisi tabel yang menampilkan data pengguna dengan <i>role 'member'</i>	Menampilkan halaman <i>'Member'</i> yang berisi tabel yang menampilkan data pengguna dengan <i>role 'member'</i>	Terpenuhi
2	Mencari data menggunakan input Search	Menampilkan data pengguna yang memiliki <i>role 'member'</i> dengan nama sesuai inputan pengguna	Menampilkan data pengguna yang memiliki <i>role 'member'</i> dengan nama sesuai inputan pengguna	Terpenuhi
3	Memilih maksimal jumlah data yang ditampilkan antara 10, 20, 50, dan 100	Menampilkan data pengguna yang memiliki <i>role 'member'</i> dengan maksimal <i>n</i> data perhalaman	Menampilkan data pengguna dengan <i>role 'member'</i> dengan maksimal <i>n</i> data perhalaman	Terpenuhi
4	Memilih filter yang bernilai all, Active, Inactive	Menampilkan data pengguna dengan <i>role 'member'</i> yang statusnya active, inactive ataupun seluruhnya	Menampilkan data pengguna dengan <i>role 'member'</i> yang statusnya active, inactive ataupun seluruhnya	Terpenuhi
5	Menekan daftar halaman atau menekan tombol panah sebelum atau sesudah	Menampilkan data pengguna dengan <i>role 'member'</i> yang berada pada halaman yang dipilih	Menampilkan data pengguna dengan <i>role 'member'</i> yang berada pada halaman yang dipilih	Terpenuhi
6	Menekan <i>toggle</i> pada tabel daftar pengguna	<i>toggle</i> berwarna hijau jika sebelumnya <i>toggle</i> berwarna merah yang menandakan status berubah dari <i>'Active'</i> menjadi <i>'Inactive'</i> dan sebaliknya. keduanya menampilkan notifikasi yang berisi pesan <i>'Member user successfully updated'</i>	<i>toggle</i> berwarna hijau jika sebelumnya <i>toggle</i> berwarna merah yang menandakan status berubah dari <i>'Active'</i> menjadi <i>'Inactive'</i> dan sebaliknya. keduanya menampilkan notifikasi yang berisi pesan <i>'Member user successfully updated'</i>	Terpenuhi
7	Menekan tombol <i>'Export'</i>	mengunduh seluruh data pengguna dengan <i>role 'member'</i> dalam bentuk <i>file excel</i>	mengunduh seluruh data pengguna dengan <i>role 'member'</i> dalam bentuk <i>file excel</i>	Terpenuhi