

PENGEMBANGAN APLIKASI QISCUS MEET MENGGUNAKAN JITSI OPEN SOURCE FRAMEWORK



Disusun Oleh:

N a m a
NIM

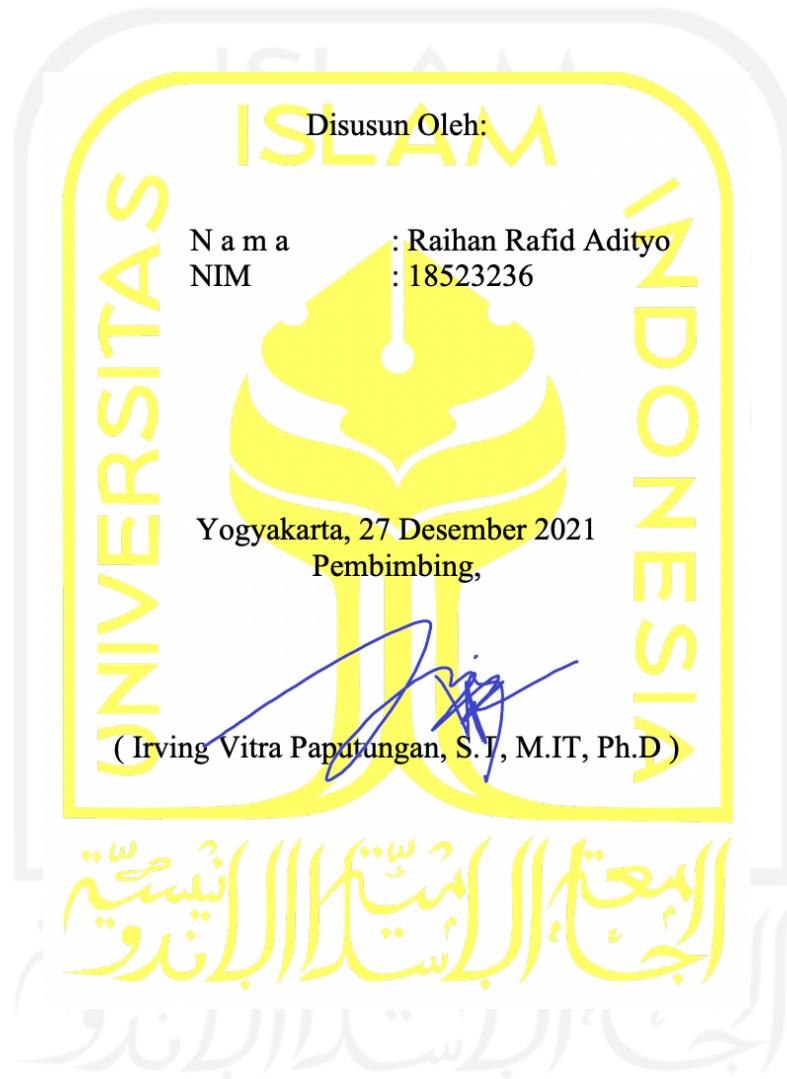
: Raihan Rafid Adityo
: 18523236

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
2022

HALAMAN PENGESAHAN DOSEN PEMBIMBING

**PENGEMBANGAN APLIKASI QISCUS MEET
MENGGUNAKAN JITSI OPEN SOURCE FRAMEWORK**

TUGAS AKHIR



HALAMAN PENGESAHAN DOSEN PENGUJI**PENGEMBANGAN APLIKASI QISCUS MEET
MENGGUNAKAN JITSI OPEN SOURCE FRAMEWORK**

ISLAM
TUGAS AKHIR

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

Yogyakarta, 23 May 2022

Tim Penguji

Irving Vitra Paputungan, S.T., M.IT., Ph.D

Anggota 1

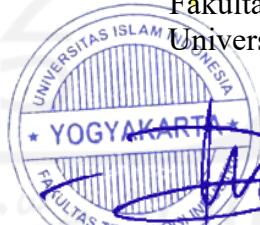
Hanson Prihantoro Putro, S.T., M.T.

Anggota 2

Nur Wijayaning Rahayu, S.Kom., M.Cs.

Mengetahui,

Ketua Program Studi Informatika – Program Sarjana
Fakultas Teknologi Industri
Universitas Islam Indonesia



(Dr. Raden Teduh Dirgahayu, S.T., M.Sc.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Raihan Rafid Adityo
NIM : 18523236

Tugas akhir dengan judul:

PENGEMBANGAN APLIKASI QISCUS MEET MENGGUNAKAN JITSI OPEN SOURCE FRAMEWORK

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila dikemudian 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 resiko dan konsekuensi apapun.

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

Yogyakarta, 27 Desember 2021



(Raihan Rafid Adityo)

HALAMAN PERSEMBAHAN

Alhamdulillahi Rabbil 'Alamin, puji syukur kepada Allah atas segala rahmat yang dilimpahkan-Nya serta sholawat serta salam selalu terucap kepada Rasulullah Muhammad. Alhamdulillah, setelah menjalani seluruh proses perkuliahan, sampailah kepada laporan tugas akhir yang disusun sebagai penuntasan apa yang telah dimulai. Laporan tugas akhir ini saya persembahkan khususnya kepada orang tua saya, Bapak dan Ibu, yang selalu memberikan dukungan yang tak terhitung khususnya dalam perjalanan kuliah saya. Juga kepada seluruh pihak yang menjadi *support system* selama proses penggerjaan baik sebagai rekan narasumber pengumpulan data, rekan kerja, hingga rekan konsultasi. Kemudian laporan tugas akhir ini juga saya persembahkan untuk siapa saja yang nantinya akan membaca laporan ini sebagai referensi yang dibutuhkan.

HALAMAN MOTO

“Any fool can write code that a computer can understand. Good programmers write code that humans can understand. Cause programming is mostly about reading, not writing”

Martin Flower, From the Book Refactoring : Improving the Design of Existing Code

“Most good programmers do programming not because they expect to get paid or get adulation by the public, but because it is fun to program”

Linus Torvalds

“It is not enough for code to work”

Robert C. Martin, From the Book Clean Architecture

“Success is the product of daily habits—not once-in-a-lifetime transformations”

James Clear, From the Book Atomic Habits

KATA PENGANTAR

Assalamu'alaikum Warrahmatullahi Wabarakatuh,

Alhamdulillah, segala puji dan syukur dihaturkan kepada Allah *Subhanahu Wata'ala*, berkat rahmah limpah-Nya yang senantiasa diberikan. Shalawat serta salam senantiasa kitaucapkan kepada Nabi Muhammad *Shallallahu 'Alaihi Wasallam*, yang telah membawa umat Islam dari zaman kebodohan menuju zaman yang terang benderang dengan Islam dan ilmu pengetahuan. Alhamdulillah, atas berkah dan izin-Nya, proses kuliah hingga magang dapat berjalan dengan baik dan lancar hingga saat ini.

Adapun dari berlangsungnya kegiatan magang di Qiscus Pte. Ltd, adapun laporan ini disusun untuk memenuhi persyaratan tugas jalur magang di Fakultas Teknologi Industri Jurusan Informatika Universitas Islam Indonesia. Dalam penyusunan laporan magang ini, tidak lepas dari arahan dan bimbingan dari berbagai pihak. Penulis mengucapkan rasa hormat dan terima kasih kepada semua pihak yang telah membantu. Pihak-pihak yang terkait antaranya sebagai berikut:

1. Allah Subhanahu Wata'ala yang telah memberikan kehidupan, keselamatan, Kesehatan jasmani dan rohani serta kesempatan untuk dapat melaksanakan magang dengan baik
2. Kedua orang tua tercinta yang selalu mendukung dan memberikan doa, bantuan positif baik secara moral maupun material dalam kegiatan magang ini.
3. Bapak Dr. Raden Teduh Dirgahayu, S.T., M.Sc., selaku Ketua Program Studi Informatika Program Sarjana Fakultas Teknologi Industri Universitas Islam Indonesia Harapan penulis terhadap penelitian tugas akhir yang telah diselesaikan.
4. Bapak Irving Vitra Paputungan, S.T., M.T., Ph.d selaku Dosen Pembimbing yang telah bersedia membimbing dan mengarahkan penyusunan laporan ini.
5. Bapak Ganjar Widiatmansyah selaku Engineering Manager yang telah menerima dan membimbing penulis dengan baik.
6. Bapak Faris Saeful Ilmi selaku Product Manager yang telah membantu penulis selama bekerja.

Pengalaman bekerja di Qiscus menjadi pengalaman yang membuka pemikiran penulis bahwa menuntut ilmu itu harus dilakukan secara terus menerus, karena khususnya ilmu yang ada di bidang teknologi itu terus berkembang mengikuti perkembangan zaman.

Laporan ini telah ditulis dengan usaha terbaik dari penulis, tetapi masih jauh dari kata sempurna. Sehingga diperlukan saran dan kritik yang membangun dari pembaca untuk penyempurnaan laporan ini. Akhir kata, penulis berharap dengan dibuatnya laporan ini dapat memberikan banyak manfaat bagi para pembaca.

Wassalamu'alaikum Warrahmatullahi Wabarakatuh.

Yogyakarta, 27 Desember 2021



Raihan Rafid Adityo

SARI

Kondisi pandemi COVID-19 membuat beberapa perusahaan harus menghentikan kegiatan secara tatap muka, sehingga menyebabkan perekonomian menurun dan membuat beberapa karyawan terkena PHK. Setelah beberapa bulan berlalu, banyak perusahaan memutuskan menjalankan operasional secara *online* dan membuat kebutuhan terkait *video call* atau *video conference* menjadi tinggi. Kebutuhan *video call* yang dibutuhkan adalah sebuah aplikasi *video call* yang dapat di-*embed* di aplikasi sebagai aplikasi pihak ke 3 yang dapat dimodifikasi sesuai dengan alur bisnis dalam aplikasi tersebut.

Qiscus mengembangkan sebuah produk terbaru yaitu aplikasi konferensi video yang bernama Qiscus Meet SDK. Qiscus Meet SDK dibangun di atas sebuah platform Web RTC *Open Source* bernama Jitsi Meet. Jitsi Meet sendiri terbagi atas beberapa komponen pembangun seperti Jitsi sebagai frontend yang menggunakan framework ReactJS, Jibri sebagai sistem untuk melakukan perekaman video dan audio konferensi, Jicofo sebagai pengelola hubungan antara partisipan konferensi video dari seorang partisipan konferensi video dengan komponen JVB, JVB (Jitsi Video Bridge) sebagai *video router* yang aktifitas utamanya adalah menerima audio dan video dari seorang partisipan ke partisipan lain yang ada di ruangan yang sama, dan Prosody yaitu sebuah XMPP server yang menghubungkan semua komponen lain melalui protokol XMPP. Oleh karena lisensi Jitsi yang bersifat *open source*, membuat platform ini bebas digunakan siapapun dan dapat dimodifikasi sesuai kebutuhan, sehingga dinilai cocok untuk digunakan Qiscus sebagai mesin utama dalam rangka mengembangkan aplikasi konferensi video. Tulisan ini menjelaskan tahap pengembangan aplikasi Qiscus Meet SDK seperti integrasi pada Jibri, migrasi backend dari bahasa PHP ke Go dan integrasi dengan Qiscus Multichannel sebagai fitur *call adds-on*. Hasil dari pengembangan ini adalah peningkatan performa backend, fitur *recording* pada Qiscus Meet SDK yang *scalable* dan integrasi Qiscus Meet SDK dengan Qiscus Multichannel.

Semoga dengan diadakannya pengembangan aplikasi ini dapat membantu perusahaan dalam mengatasi kebutuhan pengguna terkait dengan fitur *video call* atau *video conference* pada sebuah aplikasi berbasis *website* maupun *mobile*.

Kata kunci: Qiscus Meet Web SDK, Open Source, Jitsi Meet, Konferensi Video, Web RTC

GLOSARIUM

Frontend	<i>Interface</i> aplikasi untuk digunakan <i>end user</i> .
Jibri	Aplikasi khusus untuk melakukan <i>recording</i> di dalam Jitsi Meet.
Jicofo	Aplikasi untuk mengelola hubungan antara partisipan konferensi video dengan komponen Jitsi Meet.
Jitsi	Aplikasi front-end untuk melakukan konferensi video atau <i>video call</i> .
JVB	Merupakan <i>video router</i> yang berfungsi menerima audio dan video dari sebuah partisipan dan meneruskannya ke partisipan lain yang berada di ruangan yang sama.
<i>Peer-to-peer</i>	Model komunikasi dua arah antar pengguna PC melalui jaringan komputer tanpa melalui sebuah <i>server</i> .
Prosody	Sebuah XMPP <i>server</i> yang menghubungkan semua komponen lain melalui protokol XMPP.
Qiscus Meet Web SDK	<i>Software Development Kit</i> milik Qiscus yang menyediakan fitur utama untuk melakukan konferensi video yang dibangun di atas aplikasi <i>open source</i> bernama Jitsi Meet.
SDK	<i>Software Development Kit</i> adalah kumpulan perangkat lunak yang digunakan untuk mengembangkan aplikasi untuk perangkat atau sistem operasi tertentu.
XMPP	Sebuah standar komunikasi <i>real-time</i> berbasis teks, audio maupun video dengan teknologi Open XML.

DAFTAR ISI

HALAMAN JUDUL	i
HALAMAN PENGESAHAN DOSEN PEMBIMBING	ii
HALAMAN PENGESAHAN DOSEN PENGUJI	iii
HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR	iv
HALAMAN PERSEMBAHAN	v
HALAMAN MOTO	vi
KATA PENGANTAR	vii
SARI	ix
GLOSARIUM	x
DAFTAR ISI	xi
DAFTAR TABEL	xiii
DAFTAR GAMBAR	xiv
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Ruang Lingkup Magang	2
1.3 Tujuan	3
1.4 Manfaat	3
1.5 Sistematika Penulisan	3
BAB II DASAR TEORI	5
2.1 Tools	5
2.1.1 Visual Studio Code	5
2.1.2 Postman	6
2.1.3 Trello	7
2.1.4 Slack	8
2.1.5 Metabase	9
2.2 Jitsi Meet	10
2.2.1 Jitsi	12
2.2.2 Jibri	15
2.2.3 Jitsi Video Bridge (JVB)	16
2.2.4 Jicofo	17
2.2.5 Prosody	18
2.2.6 Web Server (Nginx)	19

2.2.7 <i>Database</i> (PostgreSQL)	20
2.3 Software Development Kit (SDK)	21
2.4 Open Source	22
2.5 AWS MinIO	23
2.6 Qiscus Meet SDK.....	24
BAB III PELAKSANAAN MAGANG	26
3.1 Gambaran Tim dan Projek	26
3.2 Metodologi Pengembangan.....	27
3.2.1 Permintaan Klien.....	27
3.2.2 Analisis Kebutuhan	27
3.2.3 Desain Solusi.....	27
3.2.4 Validasi Klien	28
3.2.5 Implementasi	28
3.2.6 Pengetesan Hasil	35
3.2.7 Evaluasi dan Perbaikan	35
BAB IV REFLEKSI PELAKSANAAN MAGANG	36
4.1 Manfaat Magang untuk Mahasiswa	36
4.2 Pemanfaatan Aplikasi Open Source dalam Pengembangan Produk	37
4.3 Penggunaan SDK Video Conference Dibanding Menggunakan Aplikasi Video Conference.....	37
4.4 Implementasi Qiscus Meet SDK pada Qiscus Multichannel <i>Call Adds-on</i>	37
4.5 <i>Performance Testing</i> Qiscus Meet Web SDK menggunakan Apache JMeter.....	38
BAB V KESIMPULAN DAN SARAN	41
5.1 Kesimpulan	41
5.2 Saran.....	41
DAFTAR PUSTAKA.....	43
LAMPIRAN	45

DAFTAR TABEL

Tabel 3.1 Timeline Pengembangan Proyek Qiscus Meet SDK	26
Tabel 4.1 Hasil Pengujian <i>Performance Testing</i> menggunakan Apache JMeter.....	40

DAFTAR GAMBAR

Gambar 2.1 Aplikasi Visual Studio Code.....	5
Gambar 2.2 Aplikasi Postman	7
Gambar 2.3 Trello Tim RND Qiscus.....	8
Gambar 2.4 Gambar Slack Qiscus	9
Gambar 2.5 Visualisasi Grafik Metabase	10
Gambar 2.6 Jitsi Meet.....	11
Gambar 2.7 Visualisasi Arsitektur Sistem Jitsi Meet	12
Gambar 2.8 Laman Github Jitsi Frontend.....	13
Gambar 2.9 Code Jitsi Framework ReactJS	14
Gambar 2.10 Desain Clean Architecture	15
Gambar 2.11 Laman Github Jibri	15
Gambar 2.12 Laman Github Jitsi Video Bridge (JVB).....	16
Gambar 2.13 Laman Github Jicofo.....	17
Gambar 2.15 Laman Github Prosody Jitsi Meet.....	18
Gambar 2.16 Konfigurasi Web Server Nginx.....	19
Gambar 2.17 Tampilan PG Admin 4	20
Gambar 2.18 Ilustrasi <i>Software Development Kit</i>	21
Gambar 2.19 Aplikasi <i>Open Source</i>	22
Gambar 2.20 Logo AWS MinIO	23
Gambar 2.21 Arsitektur Qiscus Meet SDK	24
Gambar 2.22 Tampilan <i>Website Amazon Web Services</i>	25
Gambar 3.1 Struktur Tim dan Projek Qiscus Meet SDK	26
Gambar 3.2 Metodologi Pengembangan Aplikasi	27
Gambar 3.3 Qiscus Meet Web SDK.....	28
Gambar 3.4 Dasbor Klien Qiscus Meet	29
Gambar 3.5 Dasbor Superadmin Qiscus Meet.....	30
Gambar 3.6 Pengembangan Go REST API	31
Gambar 3.7 Mekanisme Penyimpanan AWS Minio.....	32
Gambar 3.8 Dokumentasi Pengembangan <i>Scalable Recording System</i> Qiscus Meet SDK	32
Gambar 3.9 Mekanisme Penyimpanan AWS Minio.....	33
Gambar 3.10 Menu Video Call pada Qiscus Multichannel	34
Gambar 3.11 Tampilan <i>Video Call</i> Qiscus Multichannel <i>Call Adds-on</i>	34

Gambar 4.1 Hasil <i>Performance Testing</i> dengan 10 <i>threads</i>	38
Gambar 4.2 Hasil <i>Performance Testing</i> dengan 100 <i>threads</i>	39
Gambar 4.3 Hasil <i>Performance Testing</i> dengan 1000 <i>threads</i>	39



BAB I

PENDAHULUAN

1.1 Latar Belakang

Qiscus merupakan perusahaan teknologi penyedia platform *Multichannel Chat*, namun target pasar utamanya bukanlah untuk perorangan, melainkan untuk bisnis (Atmoko, 2021). Qiscus memiliki beberapa produk yaitu Multichannel Chat, Chat SDK, CRM, Chatbot dan yang terbaru yaitu Qiscus Meet SDK. Berdiri sejak tahun 2013, Qiscus sudah memiliki ribuan klien dari berbagai negara, solusi yang diberikan Qiscus dirasa sangat berdampak bagi perkembangan bisnis, khususnya untuk klien Qiscus.

Seiring berjalanannya waktu dan meningkatnya jumlah klien yang ada, Qiscus melihat banyaknya kebutuhan klien untuk menggunakan sistem konferensi video atau *video call*, ditambah dengan kondisi pandemi seperti saat ini yang semuanya serba daring. Konferensi video adalah teknologi yang memungkinkan pengguna yang berada pada lokasi yang berbeda, untuk mengadakan pertemuan tatap muka tanpa harus pindah ke satu lokasi Bersama (Anonim et al., 2020). Hal ini membuat Qiscus merasa hal tersebut dapat menjadi potensi bisnis baru, untuk mengembangkan sebuah produk sistem konferensi video yang dapat dipasang secara fleksibel ke berbagai jenis aplikasi. Qiscus memutuskan untuk menggunakan platform Web RTC *Open Source* bernama Jitsi Meet. Web RTC adalah spesifikasi HTML5 yang membuat pengguna dapat menambahkan komunikasi *real time*, langsung dari *browser* ke perangkat-perangkat lainnya (Santoso, 2017).

Jitsi Meet merupakan aplikasi yang digunakan untuk melakukan konferensi video. Selain konferensi video, Jitsi Meet memiliki fitur *chat*, *share desktop* dan *share URL* video dari Youtube, serta efek *blur* pada latar belakang (Asfar et al., 2020). Jitsi Meet dianggap cocok, dan memenuhi ekspektasi baik dari sisi bisnis maupun fitur. Jitsi Meet juga memiliki komunitas pengembang yang cukup besar, sehingga pada akhirnya dianggap sebagai pilihan yang tepat untuk Qiscus.

Jitsi Meet terdiri dari beberapa komponen pembangun yaitu yang pertama Jitsi, berperan sebagai front-end yang menggunakan *framework* ReactJS, lalu Jibri berperan sebagai sistem untuk melakukan perekaman audio dan video konferensi, Jicofo sebagai pengelola hubungan antara partisipan konferensi video dengan komponen JVB, lalu JVB (Jitsi Video Bridge) sebagai video *router* yang aktifitas utamanya adalah menerima audio dan video dari seorang partisipan ke partisipan lain yang ada di ruangan yang sama, dan yang terakhir yaitu Prosody,

sebuah XMPP *server* yang menghubungkan semua komponen lain melalui protokol XMPP. XMPP adalah sebuah standar komunikasi *real time* berbasis teks, suara maupun video dengan teknologi open XML (Anonim, 2021).

Tulisan ini menjelaskan pengembangan aplikasi Qiscus Meet SDK di atas sebuah platform Web RTC *Open Source* bernama Jitsi Meet dalam sebuah magang yang dilaksanakan di Qiscus Pte. Ltd. Aplikasi ini ditujukan untuk memberikan solusi terkait kebutuhan bisnis di era sekarang yang membutuhkan sistem *video call* dan konferensi video yang dapat dipasang di aplikasi masing-masing secara fleksibel. Diharapkan tulisan ini dapat memberikan gambaran terkait pengembangan aplikasi *video call* menggunakan basis aplikasi *open source*.

1.2 Ruang Lingkup Magang

Qiscus Pte Ltd adalah sebuah perusahaan *Business to Business*, yang bergerak pada bidang teknologi, khususnya teknologi telekomunikasi. Qiscus Pte Ltd memiliki kantor utama di Yogyakarta, tepatnya di Jl. Timoho No. 43, Muja Muju, Yogyakarta, lalu kantor Jakarta di CoHive 101 8th Floor, Jakarta Selatan dan kantor Singapura di 1 Coleman Street, #10-06 The Adelphi, Singapore. Adapun aktifitas yang dilakukan selama magang adalah sebagai berikut:

- a. Mempelajari aplikasi Jitsi Meet beserta seluruh komponennya (Jitsi, JVB, Prosody, Jibri dan Jicofo).
- b. Melakukan kustomisasi dan pengembangan fitur untuk front-end Qiscus Meet Web SDK berbasis ReactJS sesuai kebutuhan perusahaan.
- c. Melakukan pengembangan aplikasi web untuk dasbor superadmin dan klien yang berfungsi untuk melakukan manajemen sistem Qiscus Meet SDK beserta *monitoring analytics usage* menggunakan Framework Laravel dan Metabase.
- d. Melakukan Dev Ops untuk melakukan update versi Qiscus Meet Web SDK secara berkala menggunakan Amazon Web Service.
- e. Melakukan migrasi backend autentikasi JWT pengguna dari awalnya bahasa PHP menjadi Go.
- f. Melakukan desain ulang *database* Qiscus Meet SDK menjadi lebih terstruktur.
- g. Melakukan *meeting* dan *support* klien dari sisi teknis yang ingin mengimplementasi Qiscus Meet SDK.
- h. Melakukan pengembangan fitur *recording* agar dapat diimplementasi secara *scalable* untuk setiap *app id* yang dimiliki klien.
- i. Melakukan pengembangan Multichannel *Call Adds-on* tahap riset.

1.3 Tujuan

Sebagai bagian dari ruang lingkup magang, tujuan dari pekerjaan yang diangkat dalam laporan ini adalah sebagai berikut:

- a. Membuat sebuah aplikasi, untuk dapat digunakan klien sebagai SDK untuk membuat fitur *video call* atau *video conference*.
- b. Membuat sebuah aplikasi yang dapat terintegrasi dengan aplikasi dasbor superadmin, yang digunakan oleh *internal* Qiscus, untuk mengelola data pengguna Qiscus Meet SDK.
- c. Membuat sebuah aplikasi yang dapat terintegrasi dengan aplikasi dasbor klien, yang digunakan oleh klien, untuk melakukan melihat data penggunaan Qiscus Meet SDK.
- d. Membuat sebuah aplikasi yang dapat terintegrasi dengan Qiscus Multichannel untuk menjadi salah satu *adds-on* kedepannya.

1.4 Manfaat

Manfaat dari pengembangan aplikasi Qiscus Meet Web SDK di Qiscus Pte. Ltd. adalah sebagai berikut:

- a. Membantu klien yang membutuhkan fitur *video call* atau *video conference* yang dapat dipasang di aplikasi klien masing-masing.
- b. Digunakan oleh internal Qiscus untuk melakukan konferensi video dan melakukan *live streaming* ke Youtube.

1.5 Sistematika Penulisan

Untuk memahami lebih jelas laporan ini, materi-materi yang tertera pada laporan ini dikelompokkan menjadi beberapa sub bab dengan sistematika penyampaian sebagai berikut:

ABSTRAK

BAB I PENDAHULUAN

Bab ini berisi tentang latar belakang, ruang lingkup magang, tujuan dan manfaat penelitian, dan sistematika penulisan.

BAB II DASAR TEORI

Bab ini berisi teori yang berupa pengertian dan definisi yang diambil dari kutipan buku yang berkaitan dengan penyusunan laporan serta beberapa literature review.

BAB III PELAKSANAAN MAGANG

Bab ini berisi dokumentasi dari kegiatan dan tugas yang dilaksanakan selama menjalani

proses magang

BAB IV REFLEKSI PELAKSANAAN MAGANG

Bab ini menjelaskan mengenai hasil dari pengembangan Qiscus Meet SDK di Qiscus khususnya di tim R&D.

BAB V KESIMPULAN DAN SARAN

Bab ini berisi kesimpulan dan saran yang berkaitan berdasarkan yang telah diuraikan pada bab-bab sebelumnya.

DAFTAR PUSTAKA

LAMPIRAN

BAB II

DASAR TEORI

Qiscus Meet SDK dikembangkan diatas sebuah platform Web RTC *Open Source* bernama Jitsi. Alasan penggunaan Jitsi sebagai basis dari Qiscus Meet SDK adalah fitur yang sudah sesuai dengan kebutuhan Qiscus seperti dapat dijadikan sebagai sebuah SDK yang dapat diimplementasi pada aplikasi lain sebagai fitur, arsitekturnya yang *scalable*, lalu komunitas yang sudah besar dan aktif, serta *technology stack* yang sesuai dengan sumber daya manusia yang ada, membuat Qiscus mantap untuk menggunakan Jitsi.

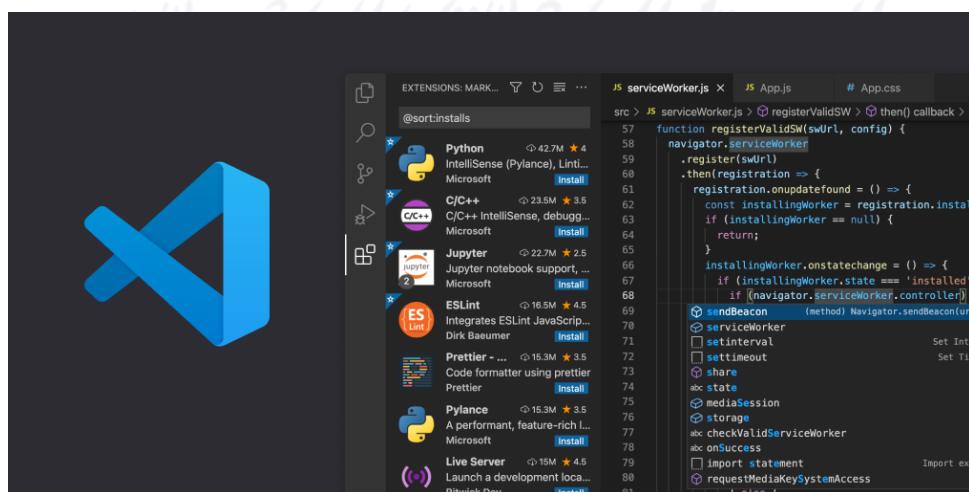
Jitsi juga dapat dikustomisasi dengan mudah karena dapat dilakukan instalasi di server internal, selain itu implementasinya yang dapat menjadi aplikasi pihak ketiga di dalam aplikasi lain, menjadikan penggunaan Jitsi yang fleksibel dan dapat menyesuaikan proses bisnis yang dimiliki aplikasi tertentu.

2.1 Tools

Dalam proses pengembangan Qiscus Meet SDK, digunakan beberapa *tools* pendukung seperti Visual Studio Code, Postman, ReactJS, Slack, dan Metabase untuk analitik.

2.1.1 Visual Studio Code

Visual Studio Code adalah aplikasi *code editor* untuk melakukan pengembangan aplikasi atau program. Visual Studio Code dibuat oleh Microsoft dan telah menjadi aplikasi *open source* yang memiliki banyak *extensions* yang membantu proses *Software Engineer* dalam melakukan pengembangan aplikasi. Tampilan dari Visual Studio Code tertera di Gambar 2.1.



Gambar 2.1 Aplikasi Visual Studio Code

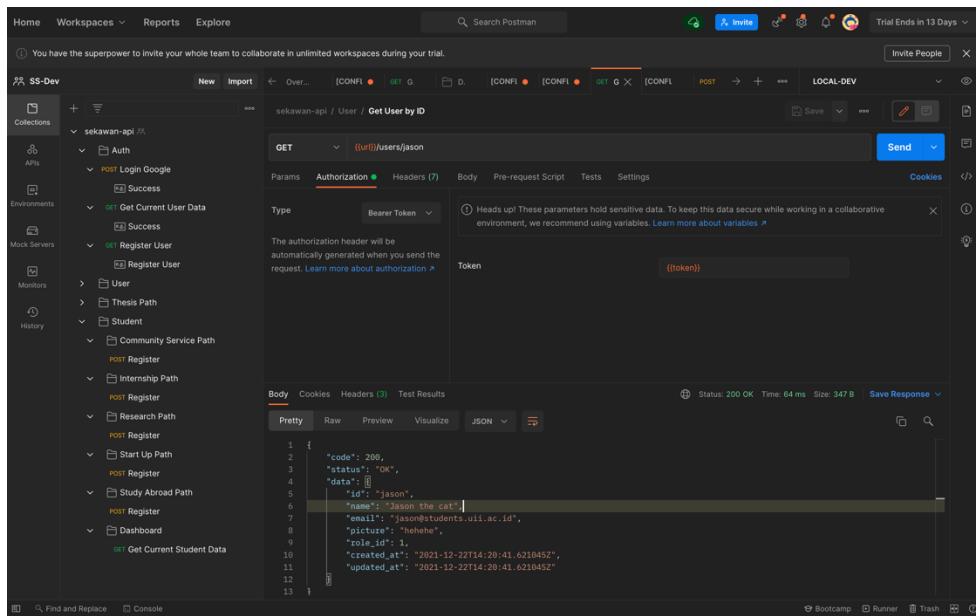
Penulis memilih menggunakan Visual Studio Code selain karena lengkapnya *extension* dan aplikasinya yang *open source*, melainkan juga karena *user interface*-nya yang mudah digunakan dan sederhana. Visual Studio Code juga sudah mendukung untuk digunakan banyak jenis bahasa pemrograman.

Kelebihan lain yang sangat membantu *Developer* yaitu Visual Studio Code sudah terintegrasi dengan git, jadi akan lebih mudah dalam menyelesaikan *conflict* atau mengetahui mana baris yang berubah atau ditambahkan, dampaknya akan lebih memudahkan dalam kerja tim (Gustopa, 2020). Git adalah sistem pengontrol versi yang biasanya digunakan pada proyek perangkat lunak, Qiscus sendiri menggunakan platform Git Bitbucket untuk penyimpanan *repositories* internal dan Github untuk *public repositories* atau yang bisa digunakan juga oleh klien.

Hal ini sangat memudahkan penulis dalam melakukan pengembangan Qiscus Meet SDK karena dalam prakteknya Git sangat dibutuhkan saat melakukan pembaharuan versi antara Qiscus Meet SDK dengan Jitsi Meet, yang posisinya berbeda *repository*, dan harus dilakukan komparasi yang mendalam saat melakukan pembaharuan. Dengan menggunakan Visual Studio Code hal ini menjadi lebih mudah dan efisien.

2.1.2 Postman

Postman adalah aplikasi yang berfungsi sebagai REST klien untuk uji coba REST API yang telah dibuat di sisi backend. Postman dengan sejumlah fitur dan kemudahan kerjanya, telah digunakan oleh jutaan penguji. Desain antarmuka yang sederhana dan ramah pengguna, memudahkan penguji untuk mengirim permintaan, cukup isi data yang diperlukan, pilih metode HTTP dan tekan tombol "Kirim" (Fathurrahman, 2021). Tampilan dari Postman tertera di Gambar 2.2.



Gambar 2.2 Aplikasi Postman

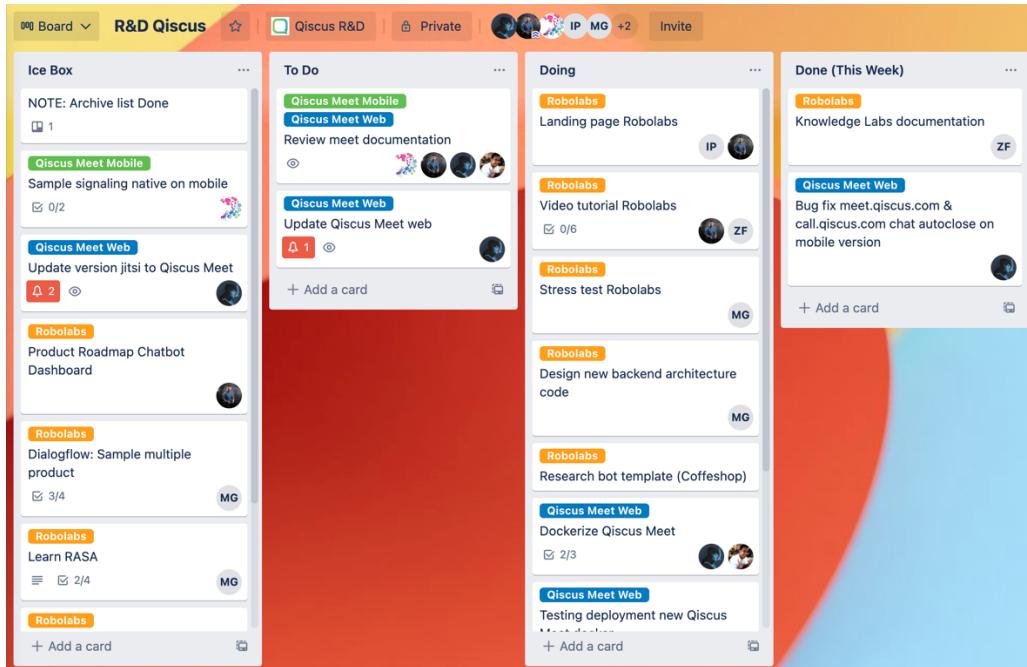
Dengan menggunakan Postman sebagai media untuk melakukan uji coba API, penulis merasa sangat terbantu dan lebih mudah dalam melakukan *testing* terkait API yang sudah dibuat di backend, sebelum akhirnya diimplementasi di sisi frontend.

Postman mendukung untuk melakukan *automation test* karena sudah dapat dilakukan pembuatan *test case* melalui aplikasi ini. Selain itu kemudahan untuk melakukan *import* dan *export collection* beserta *test case*-nya, juga sangat membantu ketika bekerja di dalam suatu tim yang memiliki banyak anggota.

Postman juga sudah mendukung untuk dilakukan pembuatan *environment* secara berbeda-beda sesuai dengan proyek yang sedang dikerjakan. Sehingga dalam prakteknya, ketika ada pengembang lain yang akan berpartisipasi dalam penggerjaan, dapat langsung *import environment* Postman yang sama, tanpa perlu melakukan persiapan dari awal yang dapat dibilang memakan waktu cukup banyak.

2.1.3 Trello

Trello adalah aplikasi manajemen pekerjaan yang memungkinkan penggunaanya untuk dapat mengatur berbagai pekerjaan yang dikerjakan dalam satu tempat (board). Alasan penggunaan Trello untuk manajemen *task* di tim RND Qiscus adalah karena kemudahan penggunaan dan gratis (Adiguna, 2020). Tampilan Trello tim RND Qiscus tertera pada Gambar 2.3.



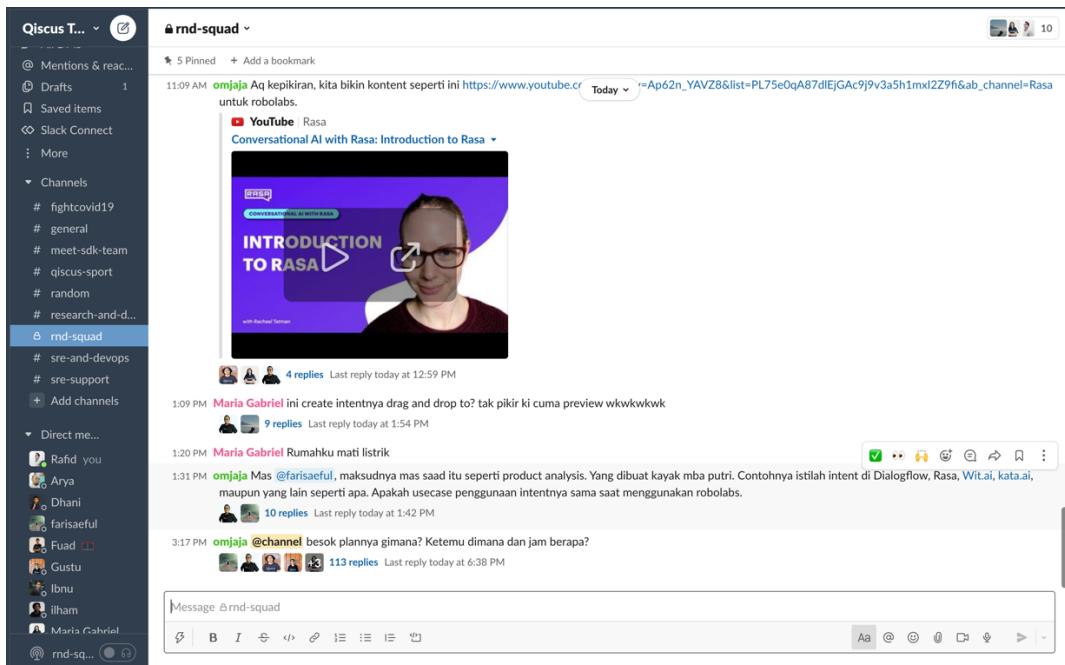
Gambar 2.3 Trello Tim RND Qiscus

Pengelompokan *progress task* dibagi menjadi 4 tahap yaitu *Ice Box*, *To Do*, *Doing* dan *Done*. *Ice Box* berarti *task* tersebut masih baru dalam proses perencanaan dan belum siap untuk dikerjakan oleh tim *Engineer*. *To Do*, berarti *task* tersebut sudah siap untuk dikerjakan oleh *Engineer* dan menjadi *task* untuk 1 pekan kedepan. *Doing*, berarti *task* tersebut sudah diterima oleh tim *Engineer* dan sudah dalam proses pengerjaan. Sedangkan *Done*, berarti *task* tersebut sudah selesai dikerjakan oleh *Engineer* dan siap untuk dipresentasikan ke tim pada *weekly syncup* di pekan tersebut.

Board yang digunakan di Trello akan diperbarui setiap satu bulan sekali oleh *Product Manager*, *card* dan *task* yang sudah selesai dilakukan di bulan yang lalu, akan diarsipkan sehingga tidak akan tampil di halaman *board* bulan selanjutnya.

2.1.4 Slack

Slack adalah sebuah aplikasi pesan dan juga dapat melakukan konferensi video, Slack memiliki fitur yang sangat lengkap untuk menjadi media utama dalam melakukan komunikasi tim yang memiliki banyak anggota, sehingga biasanya digunakan oleh banyak perusahaan khususnya perusahaan *startup*. Tampilan dari Slack Qiscus tertera di Gambar 2.4.



Gambar 2.4 Gambar Slack Qiscus

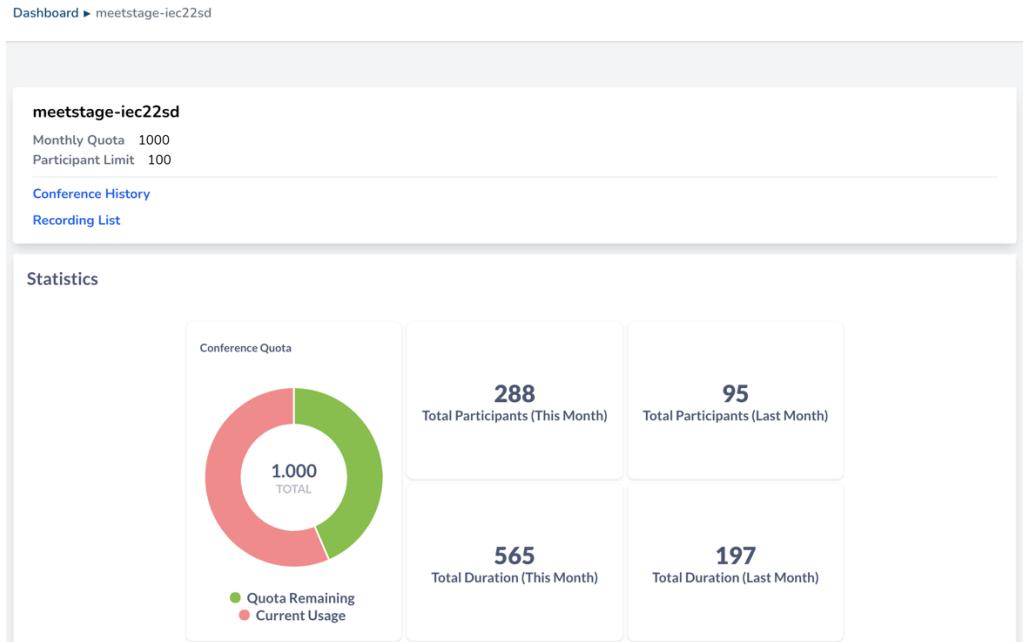
Slack menggabungkan UI yang intuitif dengan perpesanan grup dan orang-ke-orang. Ini juga memungkinkan perusahaan memiliki ukuran kendali atas siapa yang dapat menggunakan melalui sistem undangan (Marsyaf, 2021). Qiscus merasa penggunaan Slack untuk internal perusahaannya memberikan dampak yang sangat positif, baik untuk komunikasi privat maupun kelompok, karena di Slack sendiri juga dapat membuat berbagai *channel*, untuk membagi setiap tim yang ada di Qiscus. Fitur *thread* juga memudahkan tim dalam melihat pesan sesuai dengan konteks tertentu, sehingga dalam penggunaanya pesan-pesan yang ada tidak tercampur dan dapat dimanajemen dengan baik.

Pada dasarnya Slack bisa digunakan secara gratis, namun memiliki beberapa keterbatasan seperti ada batasan pesan yang disimpan dan beberapa fitur *premium* yang tidak dapat digunakan. Oleh karena itu Qiscus memutuskan untuk menggunakan Slack versi *premium* sejak tahun 2021 dan menjadikannya media komunikasi utama untuk komunikasi internal perusahaan.

2.1.5 Metabase

Metabase adalah sebuah aplikasi yang sering digunakan untuk keperluan *business intelligence*. Dengan menggunakan Metabase, dapat dilakukan visualisasi data melalui grafik, sehingga proses analisis dapat dengan mudah dilakukan. Biasanya, sumber data untuk diolah sebuah perusahaan tidak hanya berasal dari satu tempat, tetapi bermacam-macam. Oleh karena

itu, dibutuhkan data warehouse yang bisa menyimpan dan menyajikan data lebih mudah (Rahmalia, 2020). Tampilan dari visualisasi grafik Metabase tertera di Gambar 2.5.



Gambar 2.5 Visualisasi Grafik Metabase

Metabase dapat dipasang secara *on-premise* di server internal, sehingga penggunaanya dapat menyesuaikan spesifikasi server yang dibutuhkan perusahaan dengan lebih mudah. Qiscus sendiri melakukan instalasi Metabase Internalnya di AWS.

Metabase juga memiliki fitur untuk *di-embed* di aplikasi masing-masing sebagai iFrame. Sehingga dalam prakteknya dapat diimplementasikan kedalam dasbor klien dan superadmin bahkan dengan aplikasi eksternal milik klien dengan mudah. Metabase juga sudah mendukung beberapa bahasa pemrograman seperti PHP, Javascript, Golang dan Python.

2.2 Jitsi Meet

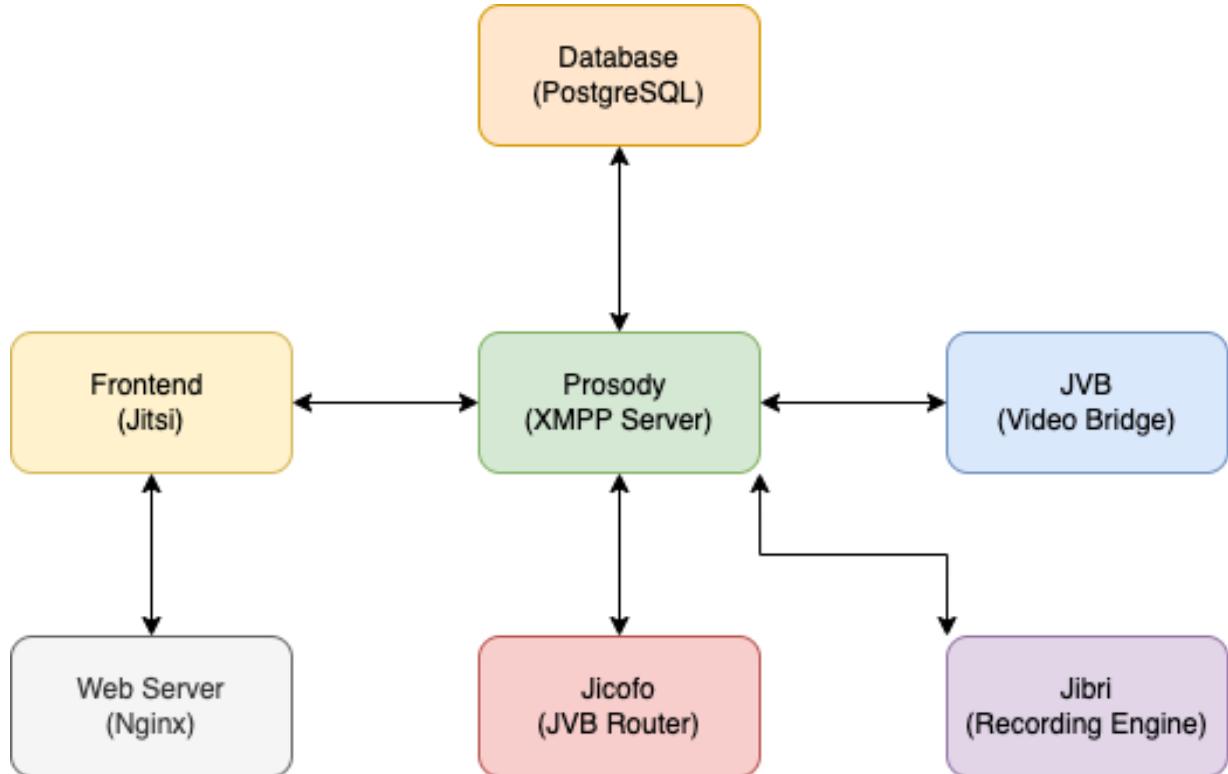
Jitsi Meet adalah sebuah platform Web RTC *Open Source* yang dapat digunakan untuk melakukan konferensi video atau *video call* dan dapat dipasang secara individual di *server* masing-masing. Jitsi Meet juga dapat diterapkan sebagai aplikasi pihak ketiga untuk memberikan fitur *video call* konferensi video di aplikasi masing-masing. *Website* dari Jitsi Meet dapat diakses di <https://jitsi.org>. Ilustrasi dari Jitsi Meet sepertinya tertera di Gambar 2.6.



Gambar 2.6 Jitsi Meet

Beberapa keunggulan Jitsi Meet yaitu panggilan video yang dienkripsi penuh, lalu penggunaan yang gratis, bahkan tidak memerlukan akun sama sekali. Jitsi Meet juga mengklaim mampu memberikan kualitas audio lebih tinggi dengan bekerja sama dengan Opus (Herfianto, 2020).

Jitsi Meet terdiri dari beberapa komponen pembangun utama yaitu: Jitsi, Jibri, Jicofo, JVB (Jitsi Video Bridge) dan Prosody. Visualisasi interaksi antar-komponen dalam Jitsi Meet diilustrasikan dalam Gambar 2.7.



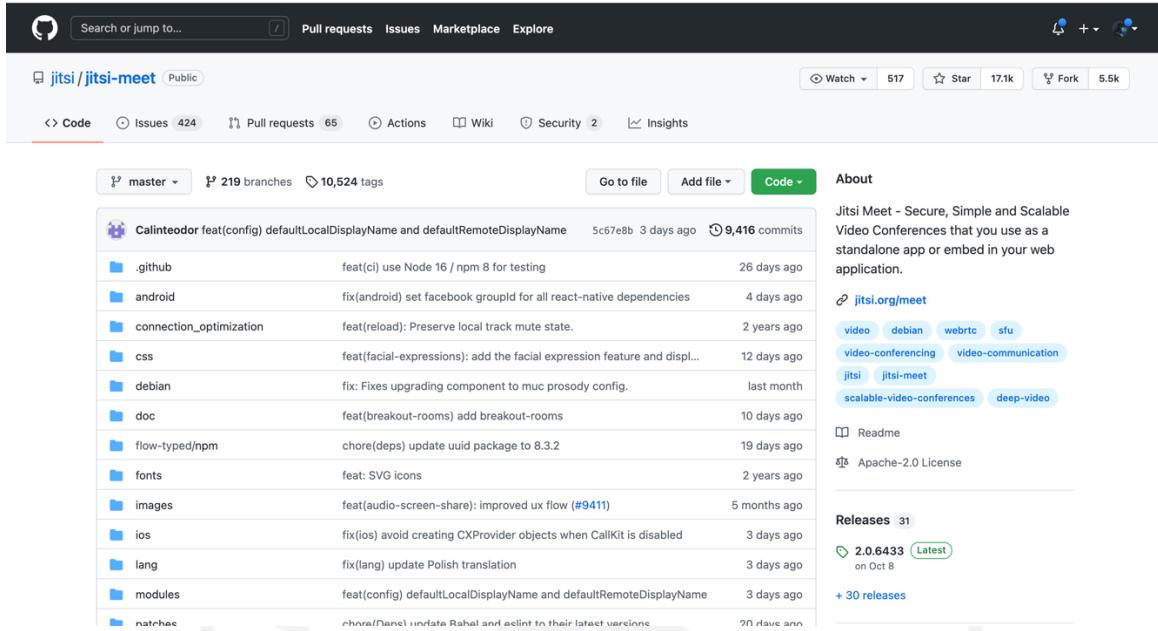
Gambar 2.7 Visualisasi Arsitektur Sistem Jitsi Meet

Seluruh komponen Jitsi Meet saling berkomunikasi melalui perantara Prosody atau XMPP *server* dari aplikasi Jitsi Meet itu sendiri, sehingga memudahkan dalam melakukan manajemen komunikasi antar komponen karena sudah tersentralisasi melalui satu komponen.

Komponen JVB dan Jibri secara *default* juga sudah dapat dilakukan *scaling* secara horizontal, yang berarti bisa menambah *instance* baru sesuai dengan kebutuhan penggunaan yang meningkat dan secara otomatis akan terkoneksi dengan komponen Jitsi lainnya selama konfigurasi yang dilakukan sudah sesuai, hal ini sangat memudahkan *Software Engineer* dalam melakukan konfigurasi *server* untuk Jitsi Meet.

2.2.1 Jitsi

Jitsi adalah antarmuka konferensi video berbasis *website* dan merupakan salah satu komponen Jitsi Meet yang berperan sebagai frontend yang diimplementasi menggunakan *framework* ReactJS. Bertugas mengelola halaman dan tampilan dari aplikasi Jitsi Meet, Jitsi Meet dikembangkan menggunakan *framework* ReactJS dengan menggunakan Javascript *flow type-annotation*. Laman Github dari Jitsi frontend tertera dalam Gambar 2.8.



Gambar 2.8 Laman Github Jitsi Frontend

Sedangkan ReactJS, ia adalah salah satu *framework* frontend yang menggunakan bahasa pemrograman Javascript. Javascript adalah bahasa pemrograman yang digunakan dalam pengembangan *website*, tujuannya adalah untuk membuat halaman *website* menjadi dinamis dan interaktif. *Browser* masa kini memungkinkan sebuah web menjadi lebih interaktif dan dinamis. Logika yang sebelumnya harus dilakukan di sisi server, kini sepenuhnya dapat di-handle pada sisi klien, semua berkat Javascript (Anonim, 2020).

Hampir semua *framework* frontend *website* yang ada, dibangun diatas bahasa pemrograman Javascript, seperti ReactJS, VueJS dan Angular, karena *behavior* dari Javascript itu sendiri yang secara desain, mudah untuk berinteraksi dengan *web browser*. Javascript juga merupakan *high-level programming language*, sehingga lebih mudah untuk dipelajari dan digunakan.

Namun, Javascript bukanlah bahasa yang memiliki *type system*, dan semakin besar basis kode yang dimiliki, kebutuhan akan *type system*, juga menjadi lebih besar. Beberapa opsi bahasa pemrograman front-end yang dapat melakukan *type system* yaitu Typescript, Reason atau dapat juga menggunakan *flow type annotation* untuk Javascript. Contoh kode Jitsi dari *framework* ReactJS tertera dalam Gambar 2.9.

277 lines (241 sloc) | 7.69 KB

Raw Blame    

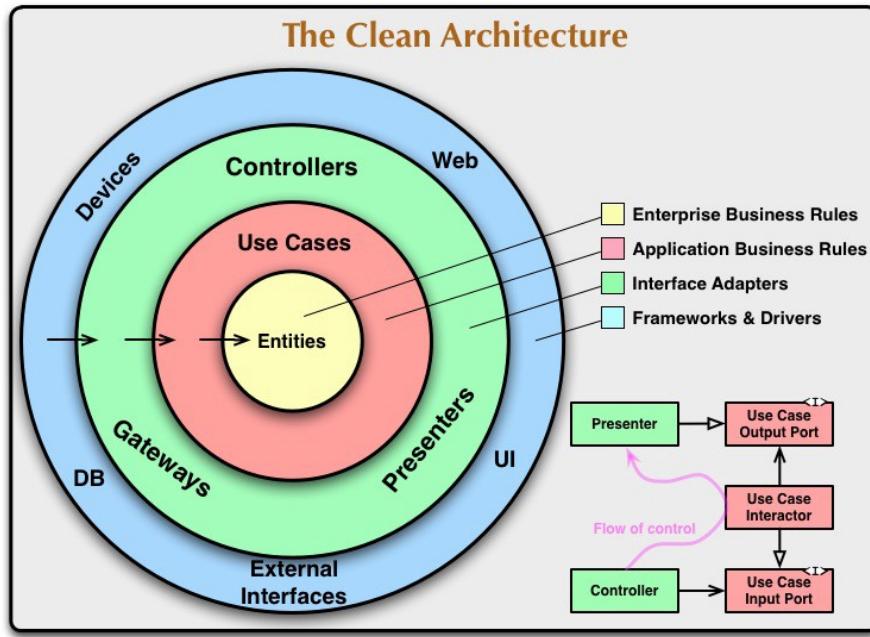
```
1 // @flow
2
3 import { generateRoomWithoutSeparator } from '@jitsi/js-utils/random';
4 import { Component } from 'react';
5 import type { Dispatch } from 'redux';
6
7 import { createWelcomePageEvent, sendAnalytics } from '../../../../../analytics';
8 import { appNavigate } from '../../../../../app/actions';
9 import isInsecureRoomName from '../../../../../base/util/isInsecureRoomName';
10 import { isCalendarEnabled } from '../../../../../calendar-sync';
11 import { isRecentListEnabled } from '../../../../../recent-list/functions';
12
13 /**
14 * {@code AbstractWelcomePage}'s React {@code Component} prop types.
15 */
16 export type Props = {
17
18     /**
19      * Whether the calendar functionality is enabled or not.
20      */
21     _calendarEnabled: boolean,
22
23     /**
24      * Whether the insecure room name functionality is enabled or not.
25      */
26     _enableInsecureRoomNameWarning: boolean,
27
28     /**
29      * URL for the moderated rooms microservice, if available.
30      */
31     _moderatedRoomServiceUrl: ?string,
32
33     /**
34      * Whether the recent list is enabled.
35      */
36     _recentListEnabled: Boolean,
37
38     /**
39      * Room name to join to.
40      */
41     _room: string,
42
43     /**
44      * The current settings.
45      */
46     _settings: Object,
```

Gambar 2.9 Code Jitsi Framework ReactJS

Pada Jitsi Meet, *framework* frontend yang digunakan yaitu menggunakan ReactJS dengan implementasi Javascript *flow type annotation*. Jitsi Meet juga mengimplementasi metode *Clean Architecture*.

Clean Architecture adalah arsitektur perangkat lunak yang berfokus untuk memisahkan logika bisnis dari *framework* atau teknologi eksternal. Dengan begitu, logika bisnis akan mudah untuk dilakukan perubahan, ketika terjadi perubahan terhadap *framework* atau teknologi yang digunakan.

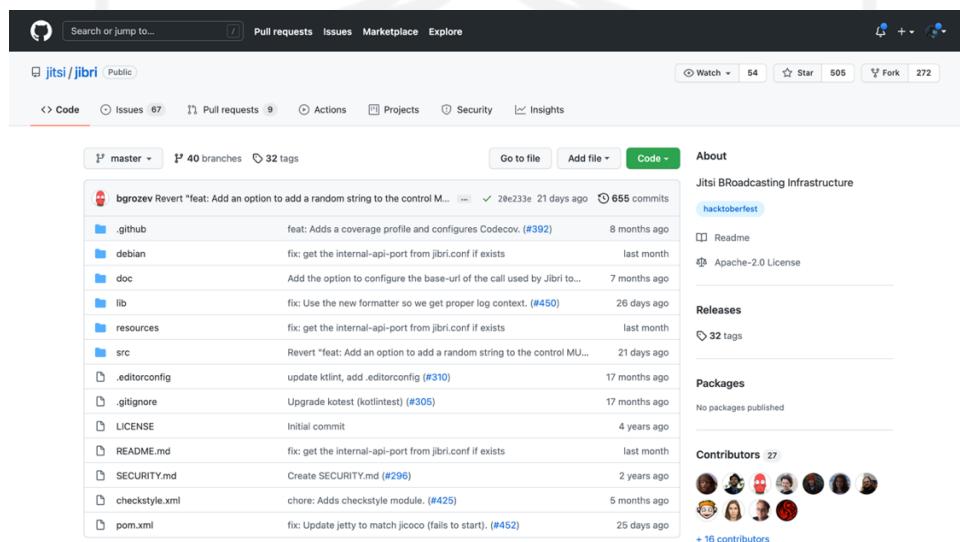
Intinya konsep ini bukan salah satu jenis *design pattern*, tetapi sebagai solusi pelengkap *design pattern* yang saat ini digunakan, sehingga kode yang yang ditulis, bisa atau mudah dibaca, dimaintain, dites dan digunakan kembali (Ahmad, 2022). Desain dari metode *Clean Architecture* tertera pada Gambar 2.10.



Gambar 2.10 Desain Clean Architecture (Sumber: <https://cybernauts.web.id>)

2.2.2 Jibri

Jibri adalah *engine* yang digunakan Jitsi Meet untuk melakukan *streaming* dan *recording*. Jibri dibangun diatas sistem operasi ubuntu 16.04 (Xenial) dan dikembangkan menggunakan bahasa pemrograman kotlin dan telah dilakukan pengetesan dengan *pre-built kernel* dan *extra kernel modules* (linux-image-extra-virtual package). Jibri bekerja dengan cara meluncurkan *instance chrome* yang dirender dalam *framebuffer virtual* dan menangkap serta menyandikan keluaran dengan ffmpeg. Laman Github dari Jibri tertera dalam Gambar 2.11.



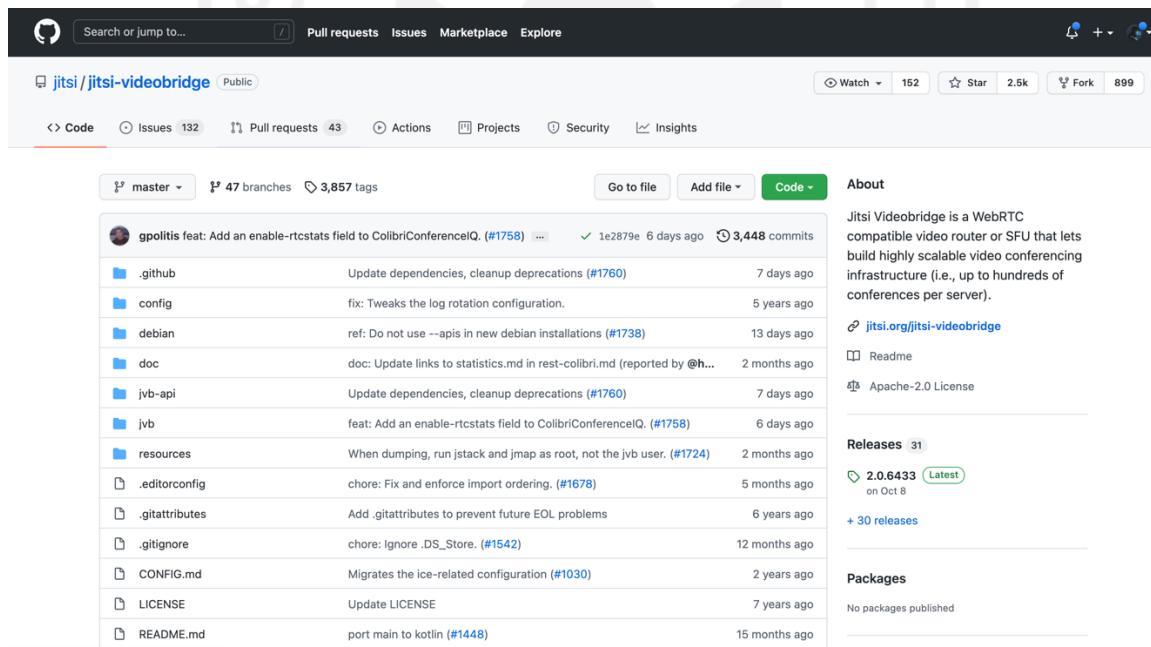
Gambar 2.11 Laman Github Jibri

Satu *instance* Jibri hanya dapat melakukan satu kali *recording* atau *streaming* video dan audio pada satu waktu. Sehingga ketika ingin melakukan *multiple recording*, perlu dilakukan *scaling* dan pengembangan lebih lanjut secara mandiri sesuai dengan kebutuhan.

Kebutuhan yang ada pada Qiscus Meet SDK yaitu dibutuhkan adanya fitur *multiple recording* yang terikat pada suatu *app id* tertentu milik klien. Data *file* yang ada, akan disimpan di satu *instance* AWS Minio dan dapat diakses melalui dasbor klien.

2.2.3 Jitsi Video Bridge (JVB)

Jitsi Video Bridge atau dapat disingkat sebagai JVB, bertugas sebagai *video router* atau *selective forwarding unit (SFU)*, aktivitas utamanya adalah menerima *audio* dan *video* dari sebuah partisipan dan meneruskannya ke partisipan lain yang berada di ruangan yang sama. JVB didesain untuk menjalankan ribuan *video stream* dari sebuah *server* JVB dan sudah kompatibel dengan WebRTC. Laman Github dari JVB tertera dalam Gambar 2.12.

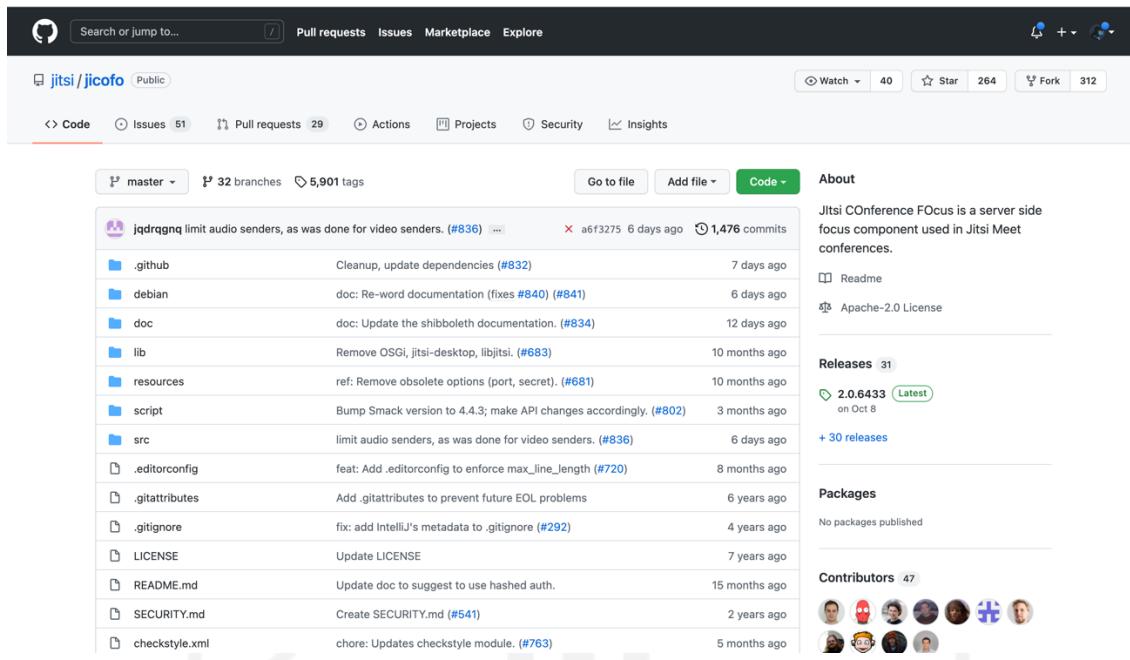


Gambar 2.12 Laman Github Jitsi Video Bridge (JVB)

Selain itu, dengan menggunakan JVB, dapat dilakukan *scaling* terkait *video channel* yang ada pada aplikasi Jitsi Meet, sehingga pada implementasi penggunaannya, setiap klien tertentu dapat memiliki *dedicated server* untuk JVB-nya sendiri, sehingga performa *video call* yang terjadi tidak akan terganggu oleh penggunaan oleh klien lain.

2.2.4 Jicofo

Jicofo bertugas mengelola hubungan antara partisipan konferensi video dengan komponen Jitsi Video Bridge (*JVB*). Jicofo merupakan komponen di sisi *server* dari Jitsi Meet. Jicofo mengelola hubungan antara partisipan konferensi video dengan komponen Jitsi Video Bridge (*JVB*). Laman Github dari Jicofo tertera dalam Gambar 2.13.



Gambar 2.13 Laman Github Jicofo

Salah satu contoh aktivitas sederhana yang dikelola oleh Jicofo adalah ketika membuat sebuah konferensi atau ruangan, maka yang akan membuatkan sesi tersebut dan memilihkan Jitsi Video Bridge (*JVB*) yang akan melayaninya adalah Jicofo. Jika terjadi kesalahan terkait konfigurasi atau adanya *error* pada komponen *JVB*, maka peserta konferensi video tidak dapat saling berkomunikasi baik melalui audio maupun video.

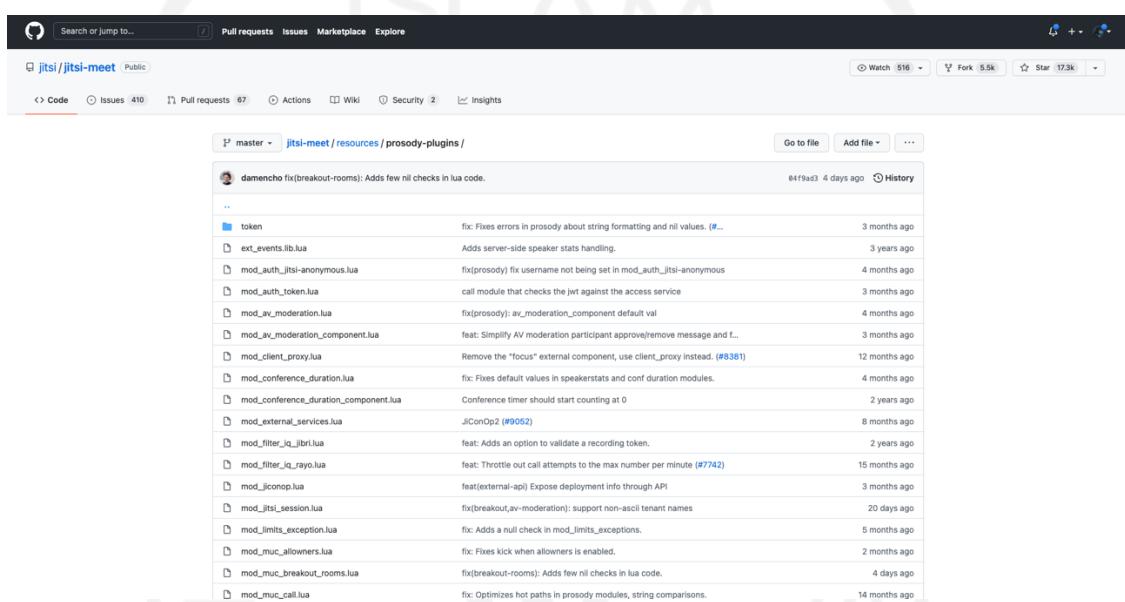
Seperti yang sudah dijelaskan sebelumnya, *JVB* juga secara *default* sudah dapat dilakukan *scaling* secara horizontal dengan cara menambahkan sebuah *instance* baru. Setelah dikonfigurasi dengan benar dan telah terkoneksi ke satu Prosody yang sama, maka secara otomatis Prosody dapat melakukan *load balancing* terkait penggunaan terhadap beberapa komponen *JVB* yang telah dibuat.

Load balancing adalah proses pembagian beban atau *traffic* sebuah aplikasi atau server. Dengan *load balancer*, beban atau *traffic* tidak akan dibebankan kepada beberapa jalur koneksi.

Hal ini mempercepat waktu respon server dan mencegahnya dari *overloading*. Dengan begini, kinerja server akan lebih maksimal tidak peduli berapa banyak beban atau *traffic* yang didapatkan (Safira, 2020).

2.2.5 Prosody

Prosody adalah sebuah XMPP *server* yang menghubungkan semua komponen lain pada Jitsi Meet melalui protokol XMPP. Laman Github dari Prosody tertera dalam Gambar 2.14.



Gambar 2.14 Laman Github Prosody Jitsi Meet

Dalam prosesnya, semua komponen Jitsi Meet yang ada berkomunikasi antar satu dengan lainnya dengan cara terkoneksi terlebih dahulu ke Prosody, dan setelah itu baru dapat saling berkomunikasi melalui *channel XMPP server*. Komponen Jicofo adalah yang pertama memasuki ruangan dan mengatur konferensi, mengirim undangan, membuka *channel* di Jitsi Video Bridge (JVB) dan seterusnya.

Prosody merupakan komponen utama yang menghubungkan komponen-komponen lain untuk dapat bekerja sesuai dengan fungsinya. Oleh karena itu, harus selalu dipastikan bahwa Prosody sudah terkonfigurasi dengan benar dan harus selalu dilakukan pembaharuan secara berkala terkait versinya, karena setiap komponen yang ada, harus saling memiliki kompatibilitas dan *up to date*, agar dapat menjalankan fungsionalitas masing-masing dengan baik.

2.2.6 Web Server (Nginx)

Web Server adalah sebuah perangkat lunak yang memberikan layanan berupa data. Berfungsi untuk menerima permintaan HTTP atau HTTPS dari klien atau yang dikenal dengan web browser (Chrome, Firefox). Selanjutnya ia akan mengirimkan respon atas permintaan tersebut kepada klien dalam bentuk halaman web (Juliarto, 2021). Contoh kode dari konfigurasi Web Server Nginx dalam Gambar 2.15.

```

1  http {
2      upstream localhost {
3          server 127.0.0.1:8000;
4          server 127.0.0.1:9000;
5      }
6
7      # This server accepts all traffic to port 80 and passes it to the upstream.
8      # Notice that the upstream name and the proxy_pass need to match.
9
10     server {
11         listen 80;
12
13         location / {
14             proxy_pass http://localhost;
15         }
16     }
17 }
18
19 events {
20     worker_connections 1024;
21 }
```

Gambar 2.15 Konfigurasi Web Server Nginx

Sedangkan Nginx, ia adalah salah satu jenis web server yang memiliki performa dan beberapa fitur yang canggih dan mudah untuk dilakukan konfigurasi. Karena alasan ini, Jitsi Meet menggunakan Nginx sebagai Web Servernya.

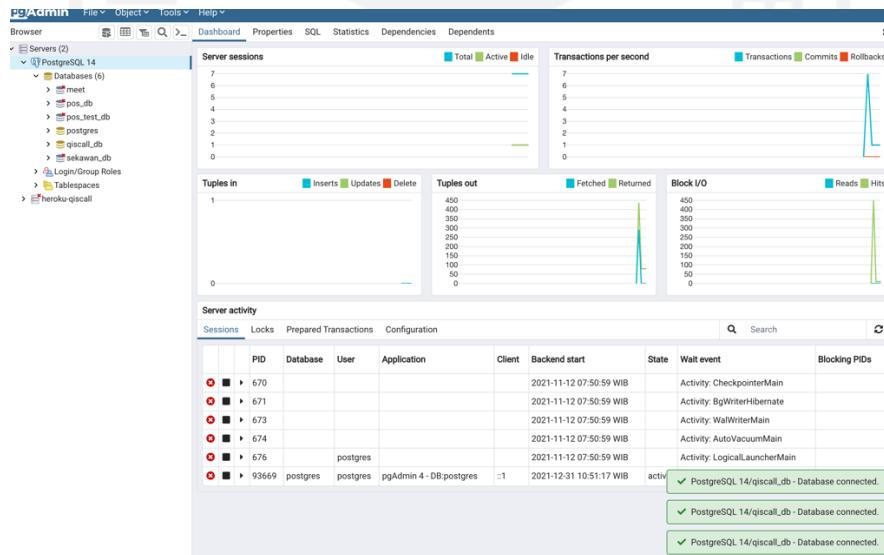
Dalam prakteknya dalam pengembangan Qiscus Meet SDK, penulis merasakan kelebihan-kelebihan yang diberikan dengan menggunakan Web Server Nginx, seperti keamanan yang baik, performa yang cepat, dan penulis juga merasa lebih mudah untuk melakukan instalasi Web Server saat melakukan *deployment* aplikasi menggunakan Nginx karena implementasinya yang mudah dan sederhana.

Kekurangan yang dirasakan oleh penulis selama menggunakan Web Server Nginx adalah komunitas pengembang yang belum cukup besar, dan ketika mengalami kendala terkait *error* atau suatu hal lain, masih sulit untuk menemukan solusi dari permasalahannya di internet.

2.2.7 Database (PostgreSQL)

Database adalah kumpulan data yang dikelola sedemikian rupa berdasarkan ketentuan tertentu yang saling berhubungan sehingga mudah dalam pengelolaannya. Melalui pengelolaan tersebut pengguna dapat memperoleh kemudahan dalam mencari informasi, menyimpan informasi dan membuang informasi. Dalam pengembangan Qiscus Meet SDK digunakan RDBMS PostgreSQL untuk melakukan penyimpanan data penggunaan konferensi video, seperti partisipan, lama penggunaan konferensi video, data *file recording* dan data pengguna beserta *app id*-nya.

Sedangkan RDBMS adalah cara mengolah suatu *database* yang terintegrasi untuk menghubungkan tabel yang satu dengan yang lainnya. Pada saat pengembangan Qiscus Meet SDK penulis menggunakan RDBMS PG Admin 4. PG Admin 4 adalah salah satu RDBMS untuk PostgreSQL. Tampilan dari PG Admin 4 tertera pada Gambar 2.16.



Gambar 2.16 Tampilan PG Admin 4

2.3 Software Development Kit (SDK)

SDK adalah kumpulan perangkat lunak yang digunakan untuk mengembangkan aplikasi untuk perangkat atau sistem operasi tertentu (Appkey, 2020). SDK dapat mempercepat proses pengembangan suatu aplikasi dikarenakan pengembang tidak perlu mengembangkan sistem atau fitur tertentu dari nol. Ilustrasi dari *Software Development Kit* dalam Gambar 2.17.



Gambar 2.17 Ilustrasi *Software Development Kit*

Sebagai contoh, ketika dibutuhkan sebuah aplikasi konsultasi dokter dan membutuhkan fitur *video call* untuk dokter dan pasien, pengembang dapat menggunakan Qiscus Meet SDK untuk mempercepat pembuatan fitur *video call* tersebut dikarenakan SDK yang memiliki sifat *plug and play*, tanpa perlu melakukan pengembangan lagi dari nol.

Dengan perkembangan perusahaan teknologi yang sangat pesat, potensi pasar untuk penjualan produk SDK cukup menjanjikan. Hampir setiap perusahaan pasti membutuhkan aplikasi baik itu untuk internal perusahaan atau jika perusahaan tersebut adalah *Software House*, maka perusahaan tersebut diharuskan melakukan pengembangan aplikasi untuk klien.

Proses pengembangan yang terus menerus dan banyaknya fitur yang harus dikembangkan, tentu pilihan untuk menggunakan SDK dalam pengembangannya sangatlah baik untuk dilakukan. Selain dapat mengurangi sumber SDM untuk merekrut programmer, hal ini juga memangkas biaya pengembangan dan pemeliharaan *server*, karena hal tersebut sudah ditanggung dan menjadi tanggung jawab perusahaan pemilik SDK.

2.4 Open Source

Open Source atau Sumber Terbuka adalah sistem pengembangan yang tidak dikoordinasi oleh suatu individu / lembaga pusat, tetapi oleh para pelaku yang disini adalah *Software Engineer* yang bekerja sama dengan memanfaatkan kode sumber (*source-code*) yang tersebar dan tersedia bebas (biasanya menggunakan fasilitas komunikasi internet). Beberapa contoh Aplikasi *Open Source* tersedia dalam Gambar 2.18.



Gambar 2.18 Aplikasi *Open Source*

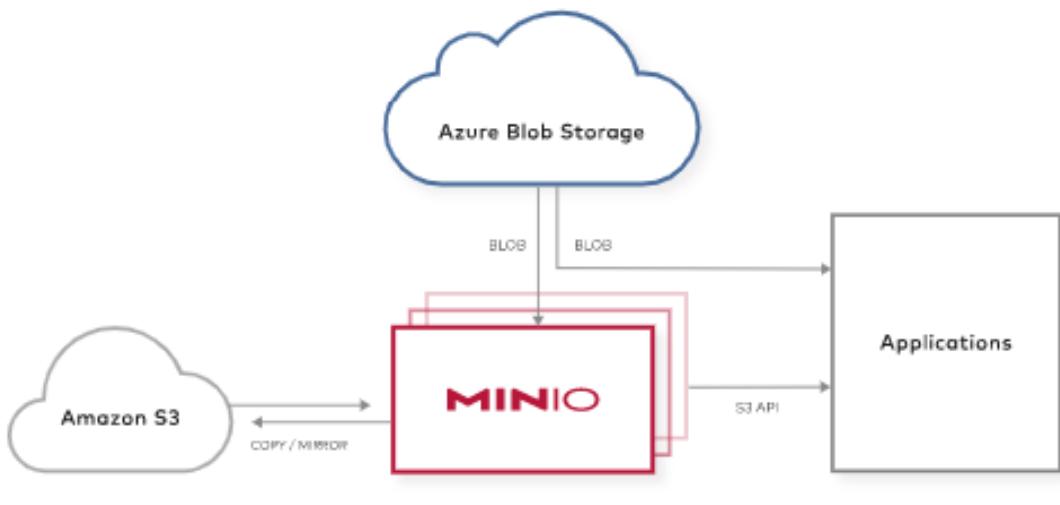
Pola pengembangan ini mengambil model ala bazaar, sehingga aplikasi *open source* ini memiliki ciri bagi komunitasnya yaitu adanya dorongan yang bersumber dari budaya memberi, yang artinya ketika suatu komunitas menggunakan sebuah program *open source* dan telah menerima sebuah manfaat kemudian akan termotivasi untuk menimbulkan sebuah pertanyaan apa yang bisa pengguna berikan balik kepada orang banyak. Contoh dari aplikasi *Open Source* yaitu seperti bahasa dan framework pemrograman, *database management system*, *linux operating system*, Jitsi dan lain sebagainya.

Aplikasi yang bersifat *open source*, biasanya tidak terikat terkait lisensi tertentu dalam penggunannya, bahkan dapat dimodifikasi atau dikembangkan secara independen dan dapat dikomersilkan ke pasar, tanpa harus membayar atau berlangganan.

2.5 AWS MinIO

Amazon Web Services atau biasa dikenal dengan AWS, merupakan salah satu platform *cloud* terbaik yang sangat komprehensif dan telah digunakan secara global. Fitur yang diberikan pun sangatlah lengkap untuk digunakan dalam kebutuhan *cloud server management*.

Salah satu fitur yang diberikan oleh AWS yaitu *storage server*, biasanya jika ingin menggunakan fitur ini, pilihannya dapat menggunakan AWS S3. Namun jika kebutuhannya adalah sebuah penyimpanan objek yang dapat dijalankan di jaringan pribadi atau lokal, AWS juga menyediakan sebuah *service* yang bernama AWS Minio. Ilustrasi arsitektur dari AWS MinIO dalam Gambar 2.19.



Gambar 2.19 Ilustrasi Arsitektur AWS MinIO

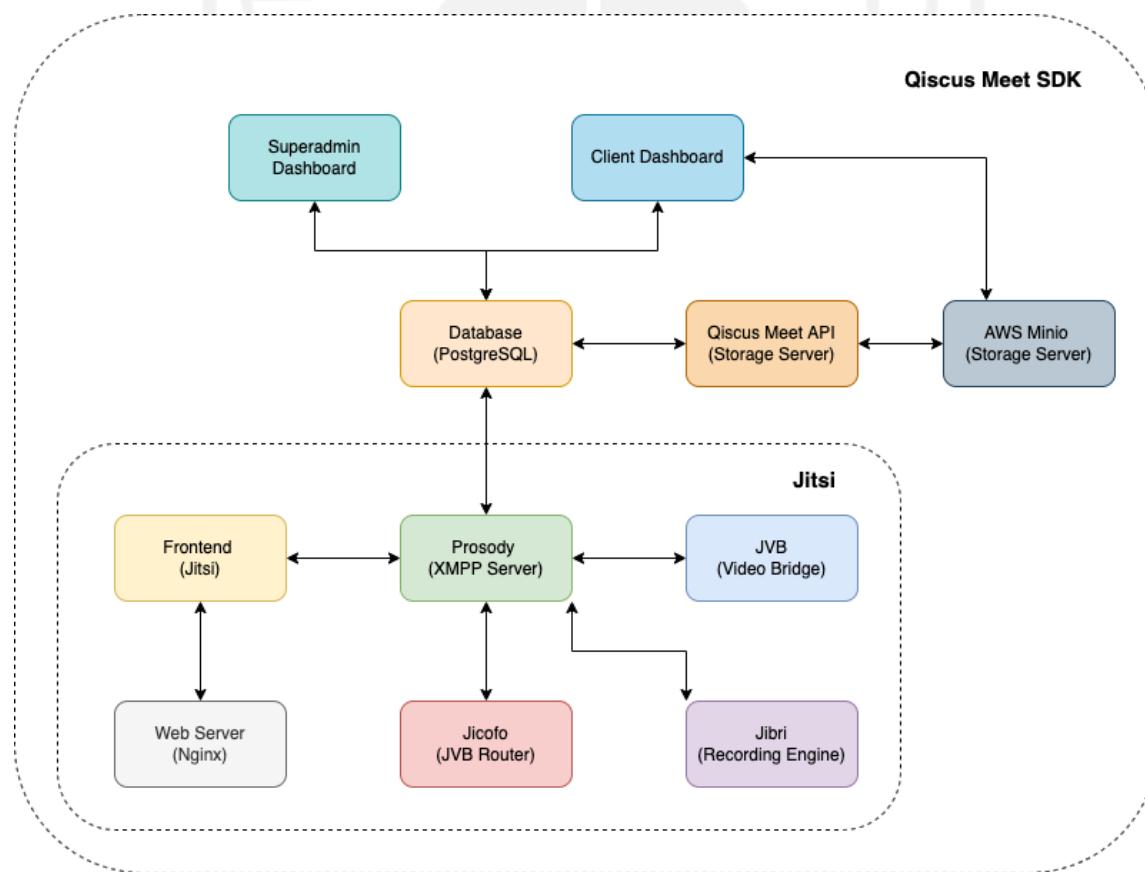
AWS MinIO memiliki API yang sudah kompatibel dengan Amazon S3, sehingga memungkinkan untuk mengembangkan aplikasi secara lokal yang reliabel dan aman. Penggunaan AWS MinIO dalam Qiscus Meet SDK digunakan sebagai *storage server* lokal, untuk melakukan penyimpanan hasil *file recording*.

Alur penyimpanan data *file recording* adalah dengan cara menghubungkan *server* Jibri dengan *server* Minio dengan menggunakan aplikasi RClone. RClone bertugas melakukan sinkronisasi data antara *server* Jibri dengan *server* MinIO, sehingga setiap kali ada *file recording* baru yang dibuat di *server* Jibri, secara otomatis data tersebut akan tersinkronisasi dengan data yang ada di *server* MinIO.

Selain itu, dengan menggunakan *storage server* dari AWS dapat mengurangi biaya sewa, karena *storage server* merupakan *server* khusus yang diberikan oleh AWS untuk melakukan penyimpanan data, sehingga dari sisi spesifikasi, jauh lebih rendah dibanding jenis *server* yang lainnya, dan harganya pun jauh lebih murah.

2.6 Qiscus Meet SDK

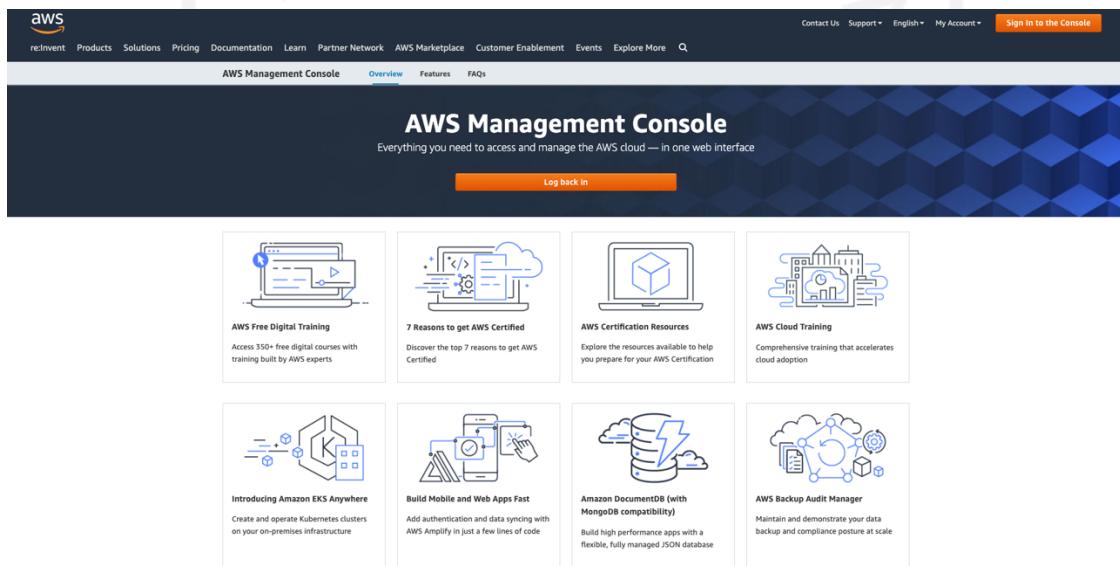
Qiscus Meet SDK adalah sebuah aplikasi *video call*, yang dibangun di atas sebuah *framework open source* bernama Jitsi Meet. Dengan menggunakan SDK dalam pengembangan suatu aplikasi, dapat mengurangi beban pengembangan aplikasi, baik dari sisi biaya maupun sumber daya manusia atau *programmer* karena *programmer* hanya perlu mengimplementasi SDK pada aplikasi yang sedang dikembangkan, tanpa harus membuatnya dari nol. Arsitektur lengkap dari Qiscus Meet SDK terdapat pada Gambar 2.20.



Gambar 2.20 Arsitektur Qiscus Meet SDK

Qiscus Meet SDK memiliki beberapa pengembangan fitur yang belum dimiliki oleh Jitsi Meet, seperti manajemen pengguna dengan menggunakan *app id*, manajemen *file recording*, dan integrasi dengan dasbor analitik dan dasbor superadmin.

Seluruh komponen yang ada di Qiscus Meet SDK dijalankan di server AWS atau Amazon Web Services dikarenakan seluruh arsitektur server Qiscus sebelumnya sudah menggunakan AWS sebagai *cloud server*. AWS adalah platform *cloud* yang dikembangkan oleh Amazon. AWS memberikan banyak fitur untuk melakukan manajemen *server cloud*, sehingga memudahkan tim *Site Reliability Engineer* (SRE) dalam melakukan manajemen dan pemeliharaan server. Tampilan dari Web AWS seperti yang tertera di Gambar 2.21.



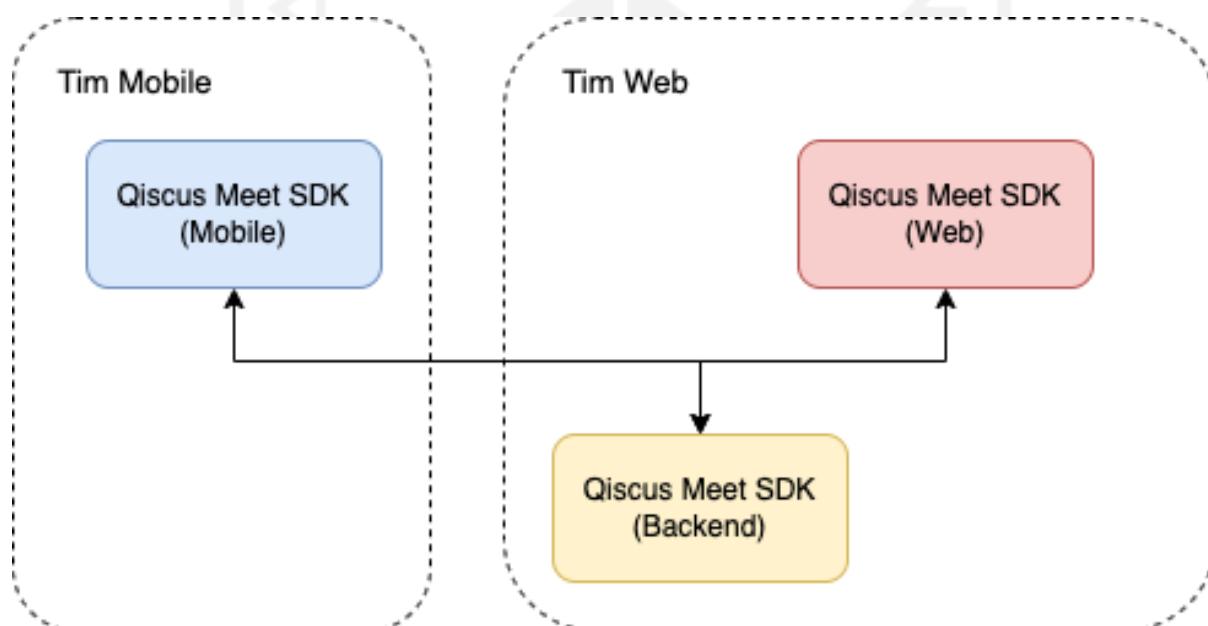
Gambar 2.21 Tampilan Website Amazon Web Services

Kelebihan yang dimiliki AWS selain dari sisi keamanan yang sudah sangat baik, yaitu dapat menggunakan fitur tertentu milik AWS pada saat sedang dibutuhkan saja, dan ketika sudah tidak membutuhkan, dapat langsung dimatikan kapan saja. Kelebihan tersebut sangat membantu perusahaan dalam mengurangi biaya sewa untuk *server*.

BAB III PELAKSANAAN MAGANG

3.1 Gambaran Tim dan Projek

Pengembangan Qiscus Meet SDK terbagi menjadi 3 bagian yaitu *mobile*, web dan backend. Penulis mengerjakan bagian web dan backend, sedangkan untuk bagian *mobile* dikerjakan oleh anggota tim Developer yang lain. Tim pengembang Qiscus Meet SDK berjalan dibawah supervisi dari *Engineering Manager* beserta *Product Manager* tim RND. Tim RND terdiri dari 7 orang dan terbagi menjadi 2 divisi, yaitu divisi Qiscus Meet dan Qiscus Robolabs. Gambaran dari tim untuk projek Qiscus Meet SDK dapat dilihat di Gambar 3.1.



Gambar 3.1 Struktur Tim dan Projek Qiscus Meet SDK

Pengembangan Aplikasi Qiscus Meet SDK dilakukan dalam 7 tahap dengan waktu selama kurang lebih 10 bulan. Selama proses penggerjaan, tim Qiscus Meet SDK dibantu oleh tim *Business Development* untuk membantu menghubungkan dengan klien beserta tim *Site Reliability Engineer* untuk membantu dalam proses *deployment*. Beberapa tahapan yang dilakukan terlampir pada Tabel 3.1.

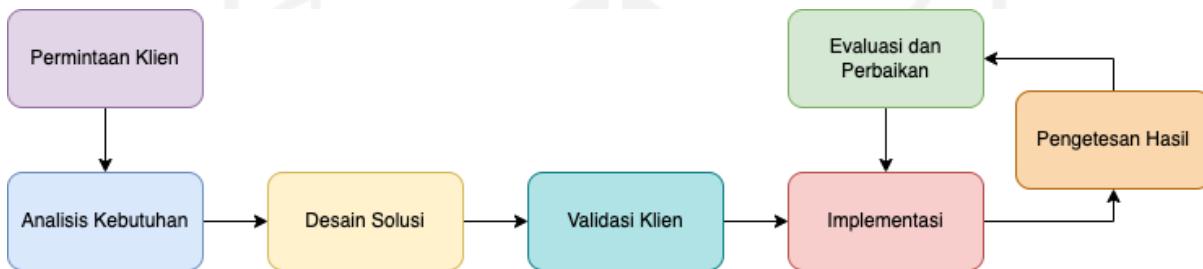
Tabel 3.1 Timeline Pengembangan Proyek Qiscus Meet SDK

No	Aktifitas
1	Meeting dan menerima permintaan dari Klien.
2	Analisis Desain

3	Desain Solusi
4	Validasi Klien
5	Implementasi
6	Pengetesan Hasil
7	Evaluasi dan Perbaikan

3.2 Metodologi Pengembangan

Pada saat penulis melakukan magang di Qiscus dan tergabung dengan tim RND. Tim RND sendiri pada saat itu merupakan divisi baru yang belum lama dibentuk, sehingga belum ada metodologi khusus yang digunakan dalam proses pengembangan aplikasi. Gambaran dari metodologi yang digunakan di tim RND Qiscus dapat dilihat pada Gambar 3.2.



Gambar 3.2 Metodologi Pengembangan Aplikasi

3.2.1 Permintaan Klien

Pada tahap ini, tim RND beserta tim *Business Development* melakukan *meeting* dengan klien untuk membahas kebutuhan aplikasi atau fitur dari klien tersebut.

3.2.2 Analisis Kebutuhan

Pada tahap analisis kebutuhan, *Product Manager* bersama dengan tim *Business Development* melakukan analisis kebutuhan terkait hal-hal yang dibutuhkan oleh klien beserta menentukan estimasi waktu pengerjaan dan biaya yang dibutuhkan untuk pengembangan. Dari hasil tersebut *Product Manager* membuat *Product Requirements Document* (PRD) untuk kedepannya digunakan oleh tim *Developer*.

3.2.3 Desain Solusi

Pada tahap desain solusi, *Product Manager* melakukan koordinasi bersama dengan tim *Developer* untuk membuat desain solusi dari hasil PRD dalam bentuk arsitektur sistem dan *flow chart*. Pembuatan desain aplikasi juga dilakukan pada tahap ini oleh tim *UI Designer*.

3.2.4 Validasi Klien

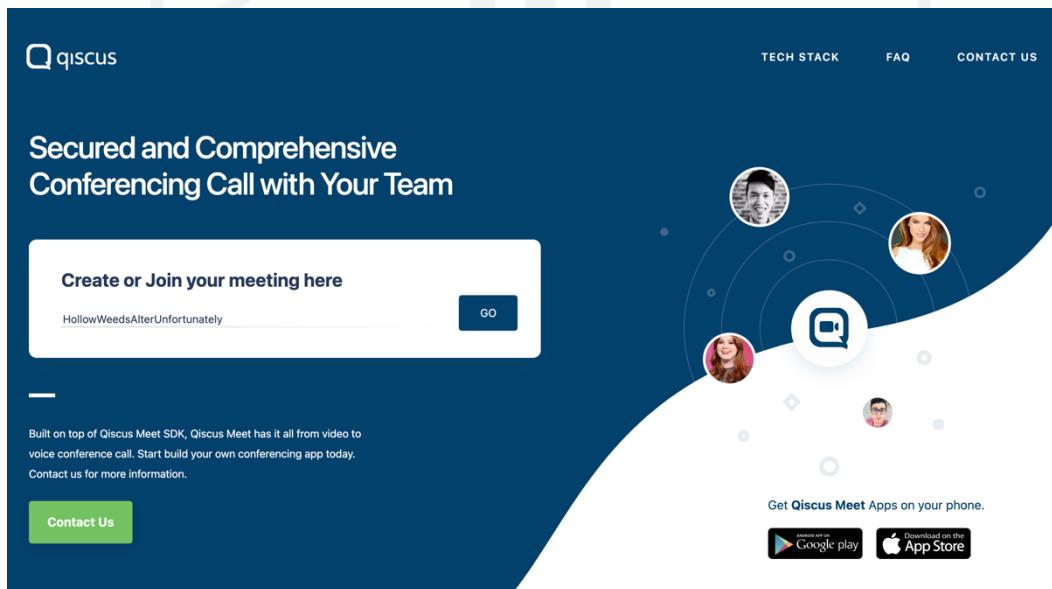
Setelah desain solusi telah berhasil dibuat, dilakukan konfirmasi kepada klien terkait hasil desain arsitektur sistem, waktu penggeraan, beserta biaya yang dibutuhkan, untuk selanjutnya dilakukan implementasi.

3.2.5 Implementasi

Pada tahap ini, dilakukan implementasi dari hasil PRD dan desain yang sudah dibuat oleh tim *UI Designer* oleh tim *Developer*. Hasil implementasi yang sudah dibuat, sebelum dilakukan *deployment* ke skala produksi, dilakukan pengetesan terlebih dahulu oleh tim QA untuk memastikan fitur yang dibuat sudah sesuai dan terbebas dari *bug*. Berikut beberapa pengembangan yang dilakukan dalam proses implementasi Qiscus Meet SDK.

On-Boarding Sistem Qiscus Meet Web SDK

Qiscus Meet Web SDK merupakan aplikasi konferensi video yang dibangun di atas platform Web RTC *Open Source* bernama Jitsi Meet. Jitsi Meet sendiri terdiri atas berbagai komponen pembangun seperti Jitsi, Jibri, Jicofo, JVB (Jitsi Video Bridge) dan Prosody. Halaman *landing page* dari Qiscus Meet Web SDK tertera di Gambar 3.3.



Gambar 3.3 Qiscus Meet Web SDK

Oleh karena kompleksnya sistem yang akan dikembangkan, penulis bersama *Engineering Manager* melakukan *on-boarding* terkait detail pengembangan sistem Qiscus Meet Web SDK. *On-boarding* dilakukan mulai dari pengenalan tim, *tools* yang digunakan

untuk komunikasi antar anggota tim dan teknologi yang digunakan untuk melakukan pengembangan Qiscus Meet SDK.

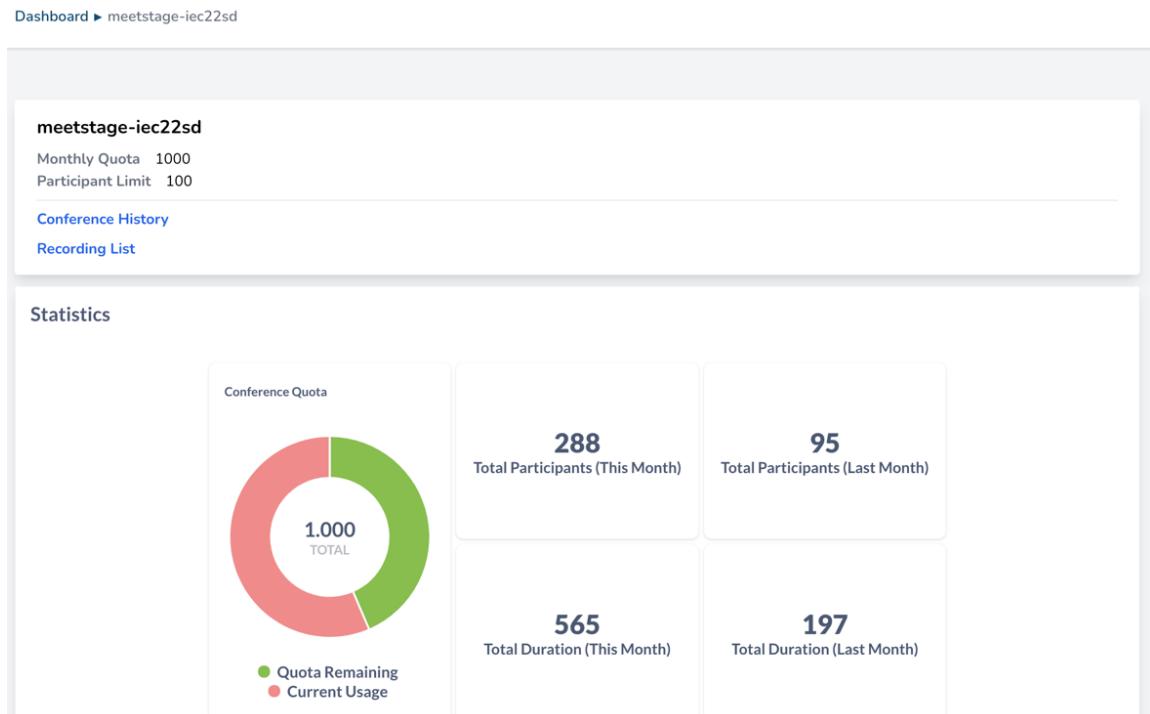
Sebelum dilakukan *on-boarding* terkait teknis pengembangan sistem Qiscus Meet Web SDK, penulis dikenalkan terlebih dahulu terkait *tools* pendukung selama bekerja di dalam tim. *Tools* yang digunakan antara lain Visual Studio Code, Postman, ReactJS, Slack, Chrome, Trello, Jitsi Meet dan Metabase untuk analitik.

Pengembangan Dasbor Superadmin dan Dasbor Klien

Seiring bertambahnya jumlah klien dan data yang semakin besar, diperlukan sebuah dasbor untuk melakukan manajemen *app id*, serta sebuah dasbor untuk menampilkan *analytics usage* untuk para klien. Pengembangan kedua dasbor ini dilakukan selama kurang lebih 1 bulan.

Dasbor Klien

Dasbor dibuat dengan *Framework* Laravel dengan Livewire sebagai front-end. Dasbor ini ditujukan untuk digunakan klien yang menggunakan Qiscus Meet SDK untuk melihat analitik penggunaan bulanan serta manajemen *app id*. Tampilan dasbor tertera di Gambar 3.4.



Gambar 3.4 Dasbor Klien Qiscus Meet

Dasbor Superadmin

Dasbor Superadmin dibuat dengan *Framework* Laravel dengan Livewire sebagai front-end-nya. Dasbor Superadmin bertujuan untuk membantu tim Qiscus Meet SDK melakukan manajemen klien yang menggunakan Qiscus Meet SDK, seperti pembuatan *app id*, pembuatan akun dan lain sebagainya. Tampilan Dasbor Superadmin tertera di Gambar 3.5.

The screenshot shows the Qiscus Meet Superadmin dashboard. At the top, there's a navigation bar with the Qiscus logo, Dashboard, Admin Management, App Platform Management, and Logout. Below the navigation, a breadcrumb path shows Dashboard > User Details. The main area starts with a user profile for 'Klikdokter' with an email 'it@klikdokter.com' and a 'RESET PASSWORD' button. Two large statistics are displayed: '284 Total Conference' and '1696 Total Conference Time (Minutes)'. Below these, there's a section for 'App IDs' with a '+ CREATE' button. A table lists two entries:

#	App ID	Quota	Max Participants	Recording Concurrent	Status	Action
1	klikdokter-0c490df	1000	100	ACTIVE 2	ACTIVE	[Edit] [View] [Delete]
2	kdstaging-EskemFdkWvhfj1B	1000	100	UNACTIVE	ACTIVE	[Edit] [View] [Delete]

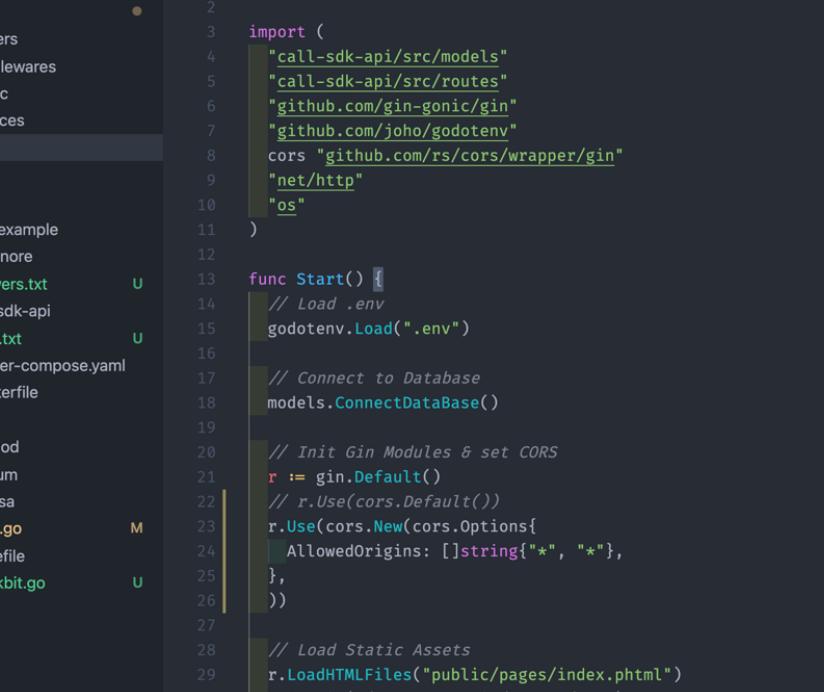
Gambar 3.5 Dasbor Superadmin Qiscus Meet

Dasbor Superadmin hanya memiliki satu akun superadmin, namun memiliki beberapa admin menyesuaikan jumlah klien yang ada. Superadmin dapat melakukan manajemen terhadap seluruh klien atau akun. Sedangkan admin, hanya dapat melakukan manajemen terhadap akun tertentu yang sudah diberikan kewenangan oleh superadmin.

Migrasi Backend Autentikasi dari Bahasa PHP ke Go

Back-end yang digunakan oleh Qiscus Meet SDK awalnya menggunakan bahasa pemrograman PHP namun struktur kode dan *library* yang digunakan sudah *outdated*, dan PHP sendiri yang sudah dianggap kurang *powerful* di masa sekarang, khususnya dibagian performa, oleh karena itu *Engineering Manager* tim RND merekomendasikan untuk dilakukan *recode* untuk migrasi ke bahasa Go. Bahasa Go sendiri saat ini dinilai memiliki performa yang paling

cepat dibanding dengan bahasa pemrograman lain yang digunakan untuk pengembangan aplikasi Back-end khususnya dibandingkan dengan PHP. Tampilan back-end autentikasi saat dibuka di VS Code tertera di Gambar 3.6.



The screenshot shows the VS Code interface with the following details:

- EXPLORER**: Shows the project structure with files like `app`, `helpers`, `middlewares`, `public`, `services`, `src`, `utils`, `.env`, `.env.example`, `.gitignore`, `answers.txt`, `call-sdk-api`, `case.txt`, `docker-compose.yaml`, `Dockerfile`, `env`, `go.mod`, `go.sum`, `key.rsa`, `main.go`, `makefile`, and `stockbit.go`.
- EDITOR**: Displays the `app.go` file content. The code is written in Go and defines a `Start` function that initializes Gin modules, sets CORS, loads static assets, and handles a moderator page.
- STATUS BAR**: Shows the file name as `app.go` and the current line number as `34`.

```
app > app.go > Start
  1 package app
  2
  3 import (
  4     "call-sdk-api/src/models"
  5     "call-sdk-api/src/routes"
  6     "github.com/gin-gonic/gin"
  7     "github.com/joho/godotenv"
  8     "github.com/rs/cors/wrapper/gin"
  9     "net/http"
 10    "os"
 11 )
 12
 13 func Start() {
 14     // Load .env
 15     godotenv.Load(".env")
 16
 17     // Connect to Database
 18     models.ConnectDataBase()
 19
 20     // Init Gin Modules & set CORS
 21     r := gin.Default()
 22     // r.Use(cors.Default())
 23     r.Use(cors.New(cors.Options{
 24         AllowedOrigins: []string{"*", "*"},
 25     },
 26 ))
 27
 28     // Load Static Assets
 29     r.LoadHTMLFiles("public/pages/index.phtml")
 30     r.Static("/css", "public/assets/css")
 31     r.Static("/js", "public/assets/js")
 32
 33     // Moderator Page
 34     r.GET("/", func(c *gin.Context) {
```

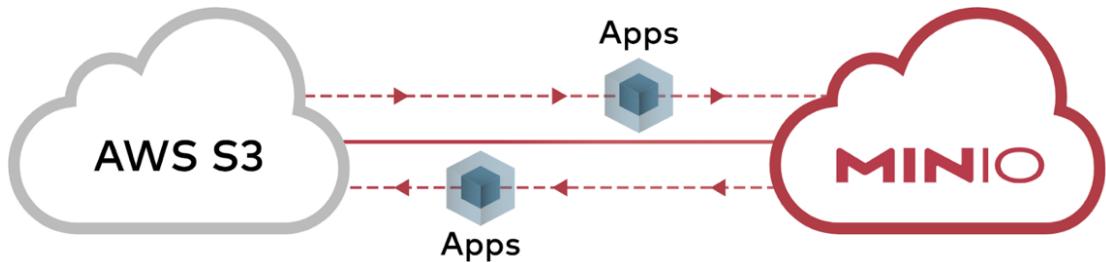
Gambar 3.6 Pengembangan Go REST API

Pengembangan Scalable Recording Sistem Qiscus Meet SDK

Default dari Jitsi Meet, fitur *recording* yang ada tak terbatas untuk siapapun yang ingin menggunakan, selama *engine* Jibri telah aktif, siapapun bisa melakukan *recording*. Hal ini menimbulkan sebuah masalah, karena tidak semua klien membeli Qiscus Meet SDK beserta dengan fitur *recording*-nya.

Oleh karena itu dilakukanlah *scaling* untuk penggunaan fitur *recording* berdasarkan *app id* klien. Pengembangan dilakukan dari sisi back-end *recording* (Jibri), Front-End Qiscus Meet Web SDK, serta menggunakan AWS Minio sebagai penyimpanan *file*. Pengembangan dari sisi back-end *recording* atau Jibri adalah dilakukannya *consume API* saat awal user melakukan *recording* serta saat user selesai melakukan *recording*, tujuannya supaya status penggunaan *recording* oleh setiap *app id* dapat termonitor oleh sistem. Ketika user selesai melakukan

recording, sistem akan melakukan pemanggilan API untuk dilakukan penyimpanan data ke *database* dan mengunggah *file* hasil *recording* ke dalam *server* AWS Minio. Skema hubungan antara aplikasi dengan AWS Minio tertera di Gambar 3.7.



Gambar 3.7 Mekanisme Penyimpanan AWS Minio

Sedangkan dari sisi front-end *Qiscus Meet Web*, dilakukan validasi *app id* setiap kali pengguna membuat ruangan, jika status *recorder* dari suatu *app id* itu sibuk, maka pengguna tidak akan bisa melakukan *recording*, pada kasus ini tombol *start recording* akan disembunyikan.

Untuk *app id* yang status *recorder*-nya masih tersedia, maka pengguna dapat melakukan *recording* dengan cara menekan tombol *start recording*. Dokumentasi pengembangan *scalable recording system* Qiscus Meet SDK tertera di Gambar 3.8.

› **RClone - Management file recording Qiscus Meet to AWS S3**

Source

1. <http://rclone.org>
2. <https://community.jitsi.org/t/how-to-set-up-a-working-setup-of-google-drive-one-drive-or-other-cloud-services-in-jibri-my-comprehensive-tutorial-for-the-beginner/42228>

Purpose

1. Menyimpan file recording lebih rapi dan bisa diakses oleh client
2. Dapat menggunakan AWS minio untuk package onpremise

Installation

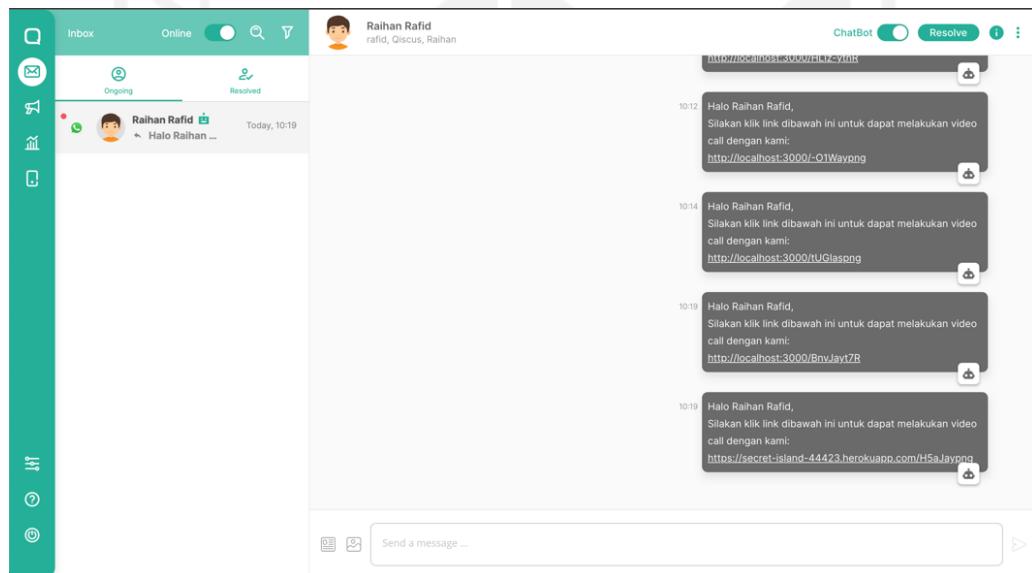
Rclone dapat dilakukan instalasi di local maupun langsung di server. Dalam kasus ini untuk instalasi dilakukan dilakukan di dalam server *jibri*

1. Persiapkan credential AWS S3 seperti
 - a. Access key ID
 - b. Secret Key
 - c. Region
2. Instal rclone di server *jibri* dengan script dibawah ini

Gambar 3.8 Dokumentasi Pengembangan *Scalable Recording System* Qiscus Meet SDK

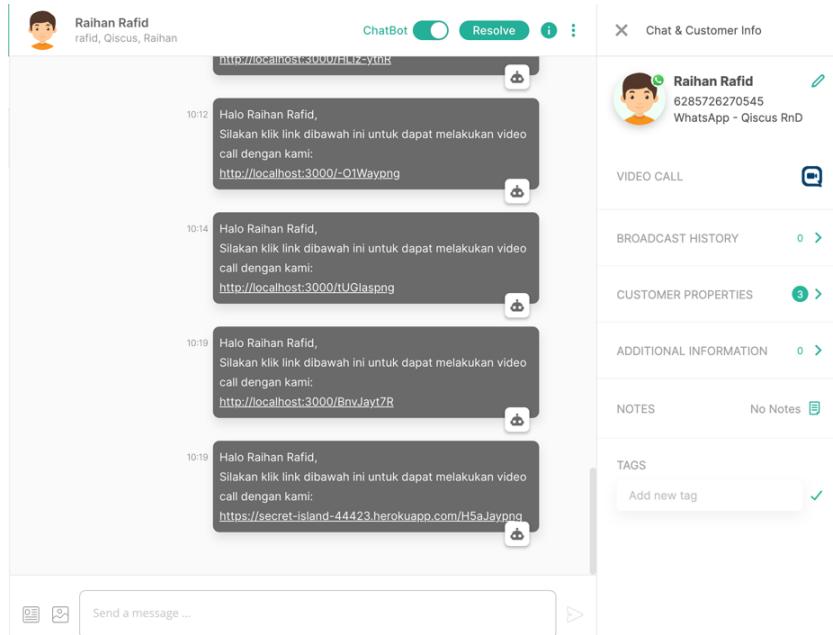
Pengembangan *Call Adds-on* Tahap Riset

Multichannel adalah sebuah aplikasi yang dapat mengelola pesan dari berbagai aplikasi chat di dalam sebuah dasbor. Dalam rangka meningkatkan *user experience* dan menyesuaikan kebutuhan klien yang kian hari makin banyak, Qiscus mencoba untuk mengimplementasi Qiscus Meet SDK miliknya sendiri untuk menjadi salah satu *adds-on* yang dapat dibeli oleh klien. Tampilan utama dari Qiscus Multichannel terdiri dari *sidebar menu* di sebelah paling kiri, list chat pengguna pada bagian tengah dan menu untuk melakukan *chat* pada bagian kanan. Pengiriman tautan *video call* yang dikirimkan oleh admin, akan masuk ke halaman menu chat dan dapat langsung diakses oleh kustomer yang menerima pesan, tampilan tersebut tertera pada Gambar 3.9.



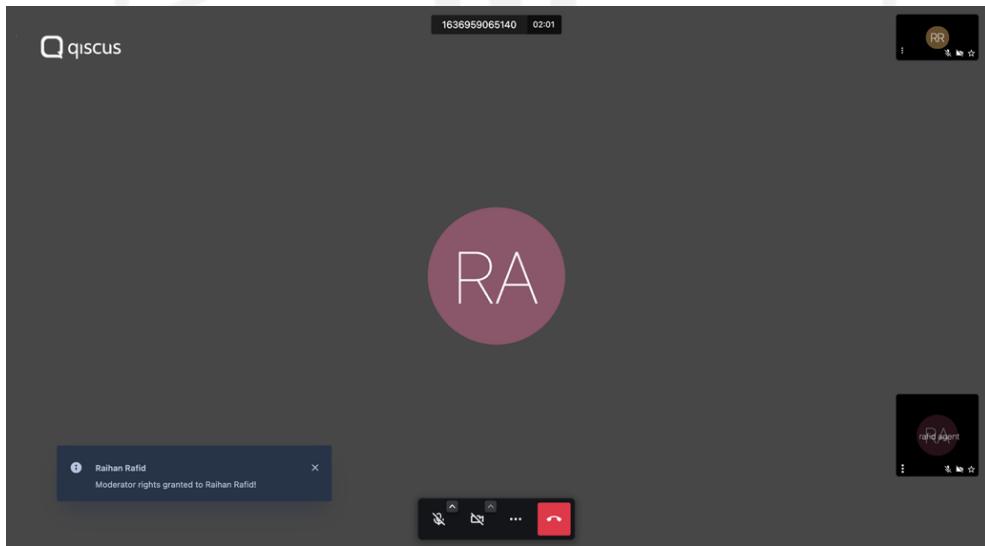
Gambar 3.9 Tampilan Qiscus Multichannel Chat

Dalam prakteknya, ketika seorang agen ingin melakukan *video call* dengan kustomer, agen dapat membuat sebuah tautan *video call*, dengan cara menekan tombol *video call* di menu bagian kanan. Ketika salah satu dari agen atau klien mengakhiri panggilan, maka url *video call* tersebut akan kadaluarsa, dan agen diharuskan membuat sebuah url baru, ketika ingin melakukan *video call* kembali. Tampilan dari menu *video call* pada aplikasi Qiscus Multichannel tertera pada Gambar 3.10.



Gambar 3.10 Menu Video Call pada Qiscus Multichannel

Setelah agen klik tombol *video call*, agen akan diarahkan ke halaman website *video call*. Dan ketika agen masuk, customer dapat masuk ke dalam ruang *video call* melalui tautan yang telah dikirim oleh agen di halaman *chat*. Tampilan dari *video call* di dalam Qiscus Multichannel Adds-on tertera pada Gambar 3.11.



Gambar 3.11 Tampilan Video Call Qiscus Multichannel Call Adds-on

Karena masih dalam pengembangan tahap riset, pengembangan yang dilakukan belum diimplementasikan langsung di Aplikasi Qiscus Multichannel di tahap *production*, namun

masih di tahap *staging*, pada tahap ini masih dilakukan penilaian dan pengembangan secara terus menerus.

Diharapkan dengan dikembangkannya *Call adds-on* untuk aplikasi Multichannel, kelak dapat meningkatkan *customer experience* agar menjadi lebih baik lagi, dan membantu klien Qiscus yang memiliki kebutuhan untuk agen atau kustomernya melakukan *video call* melalui aplikasi Qiscus Multichannel.

3.2.6 Pengetesan Hasil

Pada tahap ini, dilakukan pengetesan terkait hasil fitur yang telah diimplementasi oleh tim QA bersama dengan tim RND. Hasilnya, tim QA akan membuat sebuah dokumen *test case* untuk setiap skenario pengujian yang akan digunakan sebagai *feedback* kepada tim *Developer*. Dokumen *test case* akan divalidasi terlebih dahulu oleh *Product Manager*, sebelum akhirnya diberikan kepada tim *Developer*.

3.2.7 Evaluasi dan Perbaikan

Setelah dilakukan pengetesan oleh tim QA dan tim *Developer* sudah menerima hasil *feedback* yang diberikan. Tim *Developer* dapat langsung melakukan perbaikan sesuai dengan *feedback* yang diterima dan dilakukan evaluasi bersama tim sebelum dilakukan *deployment* kembali.

BAB IV

REFLEKSI PELAKSANAAN MAGANG

4.1 Manfaat Magang untuk Mahasiswa

Selama bekerja kurang lebih selama 10 bulan di Qiscus Pte. Ltd., beberapa manfaat yang dirasakan penulis seperti:

1. Pengalaman bekerja pada sebuah perusahaan secara profesional yang disana penulis diberi tanggung jawab untuk mengembangkan sistem yang benar-benar langsung digunakan oleh klien.
2. Pengalaman melakukan *meeting* dengan klien dan mengetahui keinginan dan kebutuhan klien. Penulis juga diberi tugas untuk membantu secara teknis dalam proses implementasi Qiscus Meet Web SDK yang dilakukan oleh klien.
3. Pengalaman mengikuti kegiatan internal di sebuah perusahaan seperti *meeting OKR* yang diadakan setiap 3 bulan sekali, yang membahas mengenai refleksi dan *review* kuarter sebelumnya dan rencana perusahaan untuk kuarter depan. Pelajaran penting yang penulis dapat pada bagian ini adalah pentingnya menentukan perencanaan secara terstruktur terkait apa yang akan dilakukan dalam rangka mengembangkan bisnis yang sudah ada, serta pentingnya untuk melakukan refleksi dan *review* terkait apa yang sudah dikerjakan selama satu kuarter sebelumnya dalam rangka terus meningkatkan dan memperbaiki proses bisnis yang ada di perusahaan.
4. Peningkatan kemampuan *public speaking*, yang didapat ketika melakukan *meeting* bersama tim internal maupun dengan klien
5. Peningkatan kemampuan *team work*, tentang bagaimana sebuah tim dapat memberikan hasil yang maksimal sesuai dengan permintaan dan harapan klien dan pemahaman mengenai bagaimana sebuah perusahaan itu dapat berjalan dengan baik dan dapat memenuhi kebutuhan pasar.

4.2 Pemanfaatan Aplikasi Open Source dalam Pengembangan Produk

Perkembangan teknologi yang pesat serta jumlah pengembang teknologi yang semakin banyak, membuat banyak terbentuknya aplikasi-aplikasi *open source*. Aplikasi *open source* sendiri adalah sistem pengembangan aplikasi yang tidak dikoordinasi oleh suatu individu / lembaga pusat, tetapi oleh para pengembang yang bekerja sama dengan memanfaatkan kode sumber (*source-code*) yang tersebar dan tersedia bebas (biasanya menggunakan fasilitas komunikasi internet).

Beberapa hal yang harus diperhatikan jika ingin menggunakan aplikasi *open source*, apalagi jika tujuan penggunaannya untuk membuat produk yang ingin dikomersilkan, yang pertama yaitu apakah aplikasi tersebut memiliki komunitas yang baik, sehingga ketika menemukan sebuah *bug* atau *error*, menjadikan dapat dengan mudah mencari solusi atas permasalahan tersebut ke dalam forum untuk meminta bantuan dari komunitas yang ada. Yang kedua yaitu apakah aplikasi tersebut dikembangkan secara baik, baik dari sisi kode, metode pengembangan dan teknologi yang digunakan, apakah relevan dengan kondisi yang ada di masa sekarang atau tidak, sehingga dapat dengan mudah melihat kualitas dari aplikasi tersebut.

4.3 Penggunaan SDK Video Conference Dibanding Menggunakan Aplikasi Video Conference

Penggunaan *SDK Video Conference* tentu memiliki kelebihan-kelebihan khusus, apalagi jika ingin membuat aplikasi *video conference* yang terikat ke sebuah aplikasi atau menjadi salah satu fitur dari aplikasi tersebut. Ketika menggunakan aplikasi *video conference* yang *dedicated* seperti zoom atau google meet, pengguna hanya dapat mencantumkan *link* dari *conference* tersebut, dan pengguna diharuskan membuka halaman baru jika menggunakan website atau harus membuka aplikasi lain jika menggunakannya di aplikasi *mobile*. Penggunaan *SDK Video Conference* akan lebih fleksibel dan dapat dikustomisasi sesuai kebutuhan aplikasi dan proses bisnis yang ada, sehingga dapat mempermudah pengguna dan membuat *user experience* yang dirasakan pengguna menjadi lebih baik.

4.4 Implementasi Qiscus Meet SDK pada Qiscus Multichannel *Call Adds-on*

Manfaat Qiscus Meet SDK sebagai sebuah *third-party application*, tak hanya dirasakan oleh klien Qiscus saja. Namun, Qiscus Meet SDK juga sangat berdampak kepada perusahaan Qiscus sendiri, khususnya untuk produk Qiscus Multichannel. Qiscus Multichannel yang

sebelumnya hanya berfokus pada komunikasi melalui media *chat*, kini sedang dilakukan proses pengembangan untuk dapat melakukan *one on one video call* selain melalui media *chatting*.

Tentu dengan dikembangkannya fitur *one on one call* untuk Qiscus Multichannel antara agen dengan customer, dapat meningkatkan *user experience* dan memenuhi kebutuhan klien yang perlu melakukan *video call* dengan kustomernya. Dengan dikembangkannya Qiscus Multichannel *Call Adds-on* dapat menjadi solusi bagi klien yang membutuhkan fitur video call dengan kustomernya serta meningkatkan *user experience* customer.

4.5 Performance Testing Qiscus Meet Web SDK menggunakan Apache JMeter

Untuk melihat performa dari Qiscus Meet Web SDK, dilakukanlah *performance testing* menggunakan Apache JMeter. Apache JMeter adalah aplikasi *open source* berbasis Java yang dapat dipergunakan untuk melakukan *performance test* (Yogi, 2018).

Penulis melakukan *performance testing* terhadap Qiscus Meet Web SDK dengan menggunakan 3 jumlah tes, yang pertama yaitu dengan 10 pengguna, 100 pengguna dan 1000 pengguna. Setelah dilakukan tes, akan menghasilkan waktu *response time* dalam satuan *millisecond*, termasuk rata-rata dan deviasinya. Nilai dari standar deviasi yang baik harus dibawah 5 % dari rata-rata keseluruhan data (Hamdani, 2014).

Pada saat dilakukan menggunakan 10 *threads* atau pengguna, didapatkan rata-rata *response time* sebesar 280 ms dan deviasi sebesar 77. Berikut hasil *performance testing* dengan *test case* 10 *threads* pada Gambar 4.1.

View Results in Table									
Name:	View Results in Table								
Comments:									
Write results to file / Read from file		<input type="button" value="Browse..."/>		Log/Display Only: <input type="checkbox"/> Errors <input type="checkbox"/> Successes		<input type="button" value="Configure"/>			
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time
1	19:24:53.615	Users 1-1	Homepage	410	✓	69269	236	107	
2	19:24:53.720	Users 1-2	Homepage	385	✓	69269	236	108	
3	19:24:53.820	Users 1-3	Homepage	360	✓	69269	236	93	
4	19:24:53.920	Users 1-4	Homepage	281	✓	69269	236	88	
5	19:24:54.020	Users 1-5	Homepage	341	✓	69269	236	129	
6	19:24:54.120	Users 1-6	Homepage	368	✓	69269	236	124	
7	19:24:54.219	Users 1-7	Homepage	275	✓	69269	236	86	
8	19:24:54.320	Users 1-8	Homepage	363	✓	69269	236	113	
9	19:24:54.420	Users 1-9	Homepage	538	✓	69269	236	67	
10	19:24:54.520	Users 1-10	Homepage	483	✓	69269	236	148	

Gambar 4.1 Hasil *Performance Testing* dengan 10 *threads*

Lalu saat menggunakan 100 *threads*, didapatkan rata-rata *response time* sebesar 1288 ms dan deviasi sebesar 655. Berikut hasil *performance testing* dengan *test case* 10 *threads* pada Gambar 4.2.

View Results in Table																				
Name:	<input type="button" value="View Results in Table"/>																			
Comments:																				
Write results to file / Read from file																				
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(…	Time(…										
1	19:26:01.261	Users 1-2	Homepage	300	✓	69269	236	115	7	7										
2	19:26:01.247	Users 1-1	Homepage	340	✓	69269	236	129	7	7										
3	19:26:01.280	Users 1-4	Homepage	356	✓	69269	236	126	5	5										
4	19:26:01.351	Users 1-11	Homepage	329	✓	69269	236	132	3	3										
5	19:26:01.340	Users 1-10	Homepage	434	✓	69269	236	140	3	3										
6	19:26:01.420	Users 1-18	Homepage	361	✓	69269	236	60	3	3										
7	19:26:01.300	Users 1-6	Homepage	484	✓	69269	236	211	7	7										
8	19:26:01.293	Users 1-5	Homepage	498	✓	69269	236	208	5	5										
9	19:26:01.440	Users 1-20	Homepage	367	✓	69269	236	61	3	3										
10	19:26:01.380	Users 1-14	Homepage	443	✓	69269	236	158	7	7										
11	19:26:01.361	Users 1-12	Homepage	470	✓	69269	236	52	3	3										
12	19:26:01.273	Users 1-3	Homepage	558	✓	69269	236	58	2	2										
13	19:26:01.471	Users 1-23	Homepage	366	✓	69269	236	61	3	3										
14	19:26:01.490	Users 1-25	Homepage	409	✓	69269	236	63	2	2										
15	19:26:01.390	Users 1-15	Homepage	509	✓	69269	236	163	8	8										
16	19:26:01.510	Users 1-27	Homepage	392	✓	69269	236	57	2	2										
17	19:26:01.430	Users 1-19	Homepage	519	✓	69269	236	206	9	9										
18	19:26:01.560	Users 1-32	Homepage	412	✓	69269	236	85	9	9										
19	19:26:01.591	Users 1-35	Homepage	411	✓	69269	236	76	2	2										
20	19:26:01.451	Users 1-21	Homepage	572	✓	69269	236	56	1	1										
21	19:26:01.267	Users 1-20	Homepage	440	✓	69269	236	60	1	1										

Gambar 4.2 Hasil *Performance Testing* dengan 100 *threads*

Lalu saat menggunakan 1000 *threads*, didapatkan rata-rata *response time* sebesar 20543 ms dan deviasi sebesar 7306. Berikut hasil *performance testing* dengan *test case* 10 *threads* pada Gambar 4.3.

View Results in Table																				
Name:	<input type="button" value="View Results in Table"/>																			
Comments:																				
Write results to file / Read from file																				
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(…	Time(…										
1	19:26:38.879	Users 1-30	Homepage	337	✓	69269	236	57	28	28										
2	19:26:38.872	Users 1-22	Homepage	373	✓	69269	236	56	27	27										
3	19:26:38.881	Users 1-40	Homepage	484	✓	69269	236	55	27	27										
4	19:26:38.857	Users 1-9	Homepage	523	✓	69269	236	97	57	57										
5	19:26:38.886	Users 1-34	Homepage	508	✓	69269	236	58	28	28										
6	19:26:38.860	Users 1-12	Homepage	535	✓	69269	236	64	30	30										
7	19:26:38.857	Users 1-8	Homepage	541	✓	69269	236	106	66	66										
8	19:26:38.887	Users 1-37	Homepage	523	✓	69269	236	57	27	27										
9	19:26:38.872	Users 1-24	Homepage	551	✓	69269	236	58	25	25										
10	19:26:38.908	Users 1-50	Homepage	793	✓	69269	236	59	28	28										
11	19:26:38.938	Users 1-82	Homepage	1731	✓	69269	236	60	25	25										
12	19:26:38.878	Users 1-25	Homepage	2649	✓	69269	236	58	30	30										
13	19:26:38.861	Users 1-18	Homepage	3453	✓	69269	236	118	60	60										
14	19:26:38.897	Users 1-45	Homepage	4409	✓	69269	236	159	56	56										
15	19:26:39.686	Users 1-831	Homepage	4760	✓	69269	236	386	179	179										
16	19:26:39.784	Users 1-922	Homepage	4706	✓	69269	236	369	201	201										
17	19:26:38.935	Users 1-86	Homepage	5967	✓	69269	236	60	28	28										
18	19:26:39.688	Users 1-816	Homepage	5234	✓	69269	236	1359	330	330										
19	19:26:38.887	Users 1-39	Homepage	6324	✓	69269	236	57	27	27										
20	19:26:39.299	Users 1-441	Homepage	6145	✓	69269	236	451	186	186										
21	19:26:39.688	Users 1-60	Homepage	6160	✓	69269	236	446	177	177										

Gambar 4.3 Hasil *Performance Testing* dengan 1000 *threads*

Hasil dari ketiga test cases tersebut ditampilkan pada Tabel 4.1 berikut. *Test Cases* merupakan jenis tes yang dilakukan, lalu *average* yaitu rata-rata *response time*, *standard deviation* berarti standar deviasi dan *percentage* yaitu persentase *standard deviation* terhadap *average response time*.

Tabel 4.1 Hasil Pengujian *Performance Testing* menggunakan Apache JMeter

<i>Test Cases (threads)</i>	<i>Average (ms)</i>	<i>Standard Deviation</i>	<i>Percentage (%)</i>
10	280	77	28
100	1288	655	50
1000	20543	7306	35

Berdasarkan tiga *test cases* tersebut, dapat disimpulkan bahwa dari ketiga *test cases* yang dilakukan, semuanya menghasilkan persentase di atas 25 persen, yang artinya performa dari Qiscus Meet Web SDK belum cukup baik, jika digunakan oleh banyak pengguna secara bersamaan. Penulis telah melaporkan dan memberi saran kepada manajer untuk dilakukan improvisasi dan perbaikan secara berkala terhadap Qiscus Meet SDK, baik dari sisi *server* maupun SDK itu sendiri, dalam rangka meningkatkan performa aplikasi supaya menjadi lebih baik.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Produk Qiscus Meet Web SDK ini merupakan produk terbaru dari Qiscus yang baru diluncurkan resmi 2 bulan setelah magang dilaksanakan. Dikarenakan Qiscus Meet SDK merupakan produk terbaru dari Qiscus dan masih dalam proses pengembangan, maka fitur yang ada belum lengkap dan belum dapat memenuhi semua kebutuhan klien. Sehingga diharapkan penulis beserta tim Qiscus Meet dapat untuk selalu melakukan *improvement* terkait fitur maupun reliabilitasnya. Beberapa hal yang dapat disimpulkan selama bekerja di Qiscus yaitu:

- a. Qiscus Meet SDK dapat digunakan oleh klien sebagai SDK untuk digunakan sebagai fitur *video call* atau *video conference* di aplikasi klien, dan dapat dikustomisasi sesuai kebutuhan klien baik dari sisi alur proses bisnis aplikasinya maupun desain tampilan penggunanya karena sifatnya yang berdiri sebagai sebuah SDK.
- b. Qiscus Meet SDK sudah dapat terintegrasi dengan aplikasi dasbor superadmin dan dapat diakses melalui website, dari dasbor ini tim internal Qiscus bisa melakukan manajemen dan *monitoring* klien.
- c. Qiscus Meet SDK sudah dapat terintegrasi dengan aplikasi dasbor klien dan dapat diakses melalui website, dari dasbor ini klien yang sudah menggunakan Qiscus Meet SDK dapat melakukan *monitoring* terkait penggunaan SDK di aplikasi klien.
- d. Qiscus Meet SDK sudah diintegrasikan dengan Qiscus Multichannel untuk menjadi salah satu *adds-on*, walaupun masih dalam proses pengembangan tahap awal.
- e. Peningkatan kemampuan *public speaking*
- f. Pengalaman bekerja sebagai seorang professional di sebuah perusahaan *startup*

5.2 Saran

Setelah melewati beberapa bulan bekerja di Qiscus, terdapat beberapa poin yang dapat penulis berikan sebagai saran, yaitu:

- a. Sebelum memilih penjaluran magang, sebaiknya mahasiswa terlebih dahulu melakukan riset terkait spesialisasi teknis yang ingin dipelajari untuk karir kedepannya. Setelah itu lakukan belajar secara mandiri terlebih dahulu, sebelum masuk penjaluran magang, sehingga proses eksplorasi saat magang dapat lebih maksimal dan tidak ada *stuck* pada hal-hal yang bersifat dasar.

- b. Tidak ada batasan dalam penjaluran magang di kurikulum baru, artinya adalah bagi mahasiswa yang ingin mengambil jalur magang, tidak harus mendaftar ke sebuah perusahaan sebagai pemagang, namun jika dirasa sudah memiliki pengalaman dan ilmu yang cukup, dapat langsung melamar menjadi karyawan *fulltime* di perusahaan tersebut.
- c. Dalam melaksanakan kegiatan magang, perlu sungguh-sungguh dalam bekerja dan memberikan kontribusi yang maksimal. Karena hal tersebut akan membuat rekan kerja merasa mahasiswa tersebut memiliki dedikasi tinggi, sehingga kemungkinan untuk diangkat sebagai karyawan di perusahaan tersebut tinggi.
- d. Dalam melakukan pengembangan sebuah produk atau fitur diharuskan untuk dapat berkomunikasi dengan baik dengan tim maupun klien, karena hal tersebut akan sangat berdampak kepada hasil produk atau fitur yang akan dikembangkan.
- e. Ketika mengembangkan sebuah aplikasi dengan basis aplikasi *open source*, perlu dilakukan pemahaman mendalam terkait aplikasi tersebut, mulai dari arsitektur sistem sampai penggunaan teknologi yang digunakan, sehingga ketika sudah melakukan terjun pengerjaan projek menjadi lebih mudah dan terstruktur.
- f. Perlunya dilakukan pengujian terhadap Qiscus Meet SDK secara lebih mendalam, termasuk perbandingan dengan aplikasi *video conference* lain.
- g. Perlu dilakukannya penelitian terkait penerapan arsitektur *Clean Architecture* dalam pengembangan sebuah aplikasi Qiscus Meet SDK ini beserta perbandingannya dibandingkan menggunakan arsitektur lainnya.

DAFTAR PUSTAKA

- Atmoko, B. D. (2014). Mengenal Qiscus, Platform Multichannel Chat dengan Fitur Canggih untuk Bisnis. <https://gizmologi.id/aplikasi/qiscus-multichannel-chat/>
- Anonim (2020). Video Conference : Pengertian, Fungsi, Manfaat, dan Rekomendasi Aplikasi. <https://idcloudhost.com/video-conference-pengertian-fungsi-manfaat-dan-rekomendasi-aplikasi/>
- Santoso, B. A. (2017). Mengenal Web RTC. <https://www.codepolitan.com/mengenal-webrtc-59c0cf77151c3>
- Gustopa (2020). Kelebihan dan Kekurangan VS Code.
<https://www.plimbi.com/article/176010/kelebihan-dan-kekurangan-vs-code>
- Fathurrahman (2021). Apa itu Postman dan Cara Penggunaannya.
<https://www.matawebsite.com/blog/apa-itu-postman-dan-bagaimana-menggunakan-postman-untuk-menguji-api>
- Adiguna, S. P. (2020). Cara Menggunakan Trello Untuk Manajemen Pekerjaan.
<https://whello.id/tips-digital-marketing/cara-menggunakan-trello>
- Marsyaf, M. I. (2021). Apa itu Slack dan Alasan Banyak Orang Menyukainya.
<https://tekno.sindonews.com/read/332790/207/apa-itu-slack-dan-mengapa-banyak-orang-menyukainya-1613088130>
- Rahmalia, N. (2020). Inilah Metabase, Tool Open Source untuk Kebutuhan Business Intelligence. <https://whello.id/tips-digital-marketing/cara-menggunakan-trello>
- Asfar, A. M. I. T., & Asfar, A. M. I. A. (2020). Video Conference Menggunakan Aplikasi Jitsi Meet.
https://www.researchgate.net/publication/340682919_Video_Conference_Menggunakan_Aplikasi_Jitsi_Meet
- Anonim. (2016). Mengenal XMPP dan Kegunaannya.
<https://achmadlutfi.wordpress.com/2016/06/27/mengenal-xmpp-dan-kegunaannya/>
- Herfianto, P. (2020). Mengenal Jitsi Meet, Alternatif Konferensi Video.
<https://gizmologi.id/aplikasi/jitsi-meet-alternatif-layanan-konferensi-video/>
- Anonim (2020). 5 Alasan Kenapa Javascript Penting untuk Web Programmer.
<https://www.dicoding.com/blog/javascript-penting-untuk-web-programmer/>
- Ahmad, Z. (2022). Berkenalan dengan Konsep Clean Architecture.
<https://cybernauts.web.id/2022/02/04/berkenalan-dengan-konsep-clean-architecture/>

- Safira, S. P. (2020). Pengertian Load Balancing dan Manfaatnya untuk Server.
<https://www.goldenfast.net/blog/pengertian-load-balancing/>
- Juliarto, R. (2021). Apa itu Web Server dan Fungsinya. <https://www.dicoding.com/blog/apa-itu-web-server-dan-fungsinya>
- Appkey. (2020). SDK adalah? Pengertian Software Development Kit
<https://achmadlutfi.wordpress.com/2016/06/27/mengenal-xmpp-dan-kegunaannya/>
- Yogi. (2018). Pengertian jMeter dan Contoh Performance Test Menggunakan jMeter
<https://edusoftcenter.com/pengertian-jmeter-dan-contoh-performace-test-menggunakan-jmeter/>
- Hamdani, Z. M. (2014). Standard Deviation
<https://stackoverflow.com/questions/26898060/jmeter-deviation-vs-throughput>

LAMPIRAN

