BAB III

PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

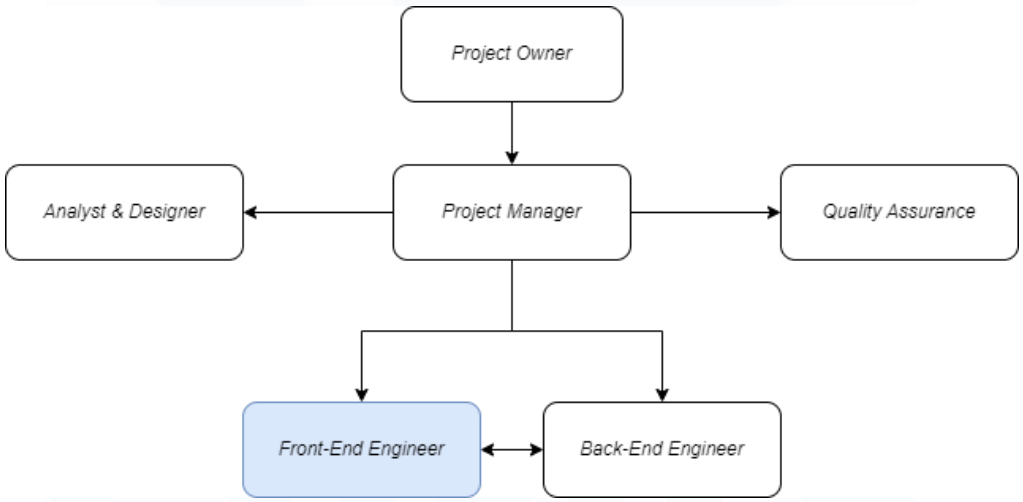
Selama masa pelaksanaan kerja magang yang berlangsung selama lima bulan di PT Sagara Asia Teknologi, penulis ditempatkan sebagai *Front-End Engineer* dalam tim Sagara Group Web. Proyek Sagara Group Web yang dipimpin oleh Gina Septiyani Putri selaku *Project Manager* ini berada di bawah pengawasan Bapak Diifa dan Bapak Dion yang merupakan *Project Owner* proyek Sagara Group Web. Selama berjalannya proyek ini, peran *Project Owner* adalah untuk mengawasi jalannya proyek, memberikan masukan terhadap pekerjaan yang dilakukan, serta memberikan bantuan dalam pemenuhan data serta kebutuhan lainnya yang diperlukan agar proyek dapat berjalan dengan baik.

Seluruh koordinasi dalam pengembangan proyek Sagara Group Web dilakukan secara daring yaitu melalui aplikasi komunikasi digital seperti Discord, Whatsapp, Zoom, dan Google Meet. Koordinasi pada pengembangan proyek Sagara Group Web ini berpusat pada *Project Manager*, sehingga semua komunikasi dengan *Project Owner* umumnya hanya dilakukan oleh *Project Manager* dan kemudian diarahkan kepada anggota tim yang terlibat. Sehingga dengan begitu, semua koordinasi dalam pengembangan proyek ini berasal dari satu pintu, yaitu melalui *Project Manager*.

Setelah *Project Manager* melakukan koordinasi dengan *Project Owner*, hal pertama yang dilakukan di awal pengembangan Sagara Group Web adalah *Project Manager* mengarahkan *System Analyst & Business Analyst* untuk membuat *Product Requirement Document* (PRD). Setelah PRD mendapat persetujuan dari *Project Owner*, dilanjutkan dengan *Project Manager* mengarahkan *UI/UX Designer* untuk merancang tampilan antarmuka pengguna Sagara Group Web. Ketika desain sudah disetujui oleh *Project Owner*, maka selanjutnya akan masuk ke tahap pengembangan yang dimana *Project Manager* akan mengarahkan tim pengembang yaitu *Front-End Engineer* dan *Back-End Engineer* untuk mulai

mengembangkan Sagara Group Web. Koordinasi antara kedua tim pengembang dapat terjalin secara mandiri karena membutuhkan banyak koordinasi antara hasil pekerjaan *Back-End Engineer* dan hasil pekerjaan *Front-End Engineer* yang nantinya harus di satukan oleh *Front-End Engineer* kedalam Sagara Group Web. Setelah tim pengembang menyelesaikan pekerjaannya, *Project Manager* akan mengarahkan *Quality Assurance* untuk melakukan *testing* terhadap halaman *website* yang telah dibuat dan jika masih terdapat kekurangan maka *Project Manager* akan mengarahkannya kembali kepada tim pengembang untuk melakukan perbaikan.

Gambar 3.1 di bawah ini merupakan bagan yang menggambarkan alur koordinasi yang terjadi antara *Project Owner, Project Manager, System Analyst, Business Analyst, UI/UX Designer, Quality Assurance, Front-End Engineer,* dan *Back-End Engineer* selama program proyek Sagara Group Web ini berlangsung:



Gambar 3. 1 Alur Koordinasi Tim Sagara Group Web

Sumber: *Project Manager* (2022)

3.2 Tugas dan Uraian Kerja Magang

Selama program kerja magang berlangsung, terdapat beberapa rangkaian dan alur kegiatan kerja magang di PT Sagara Asia Teknologi, yaitu dimulai dengan acara *Onboarding* dan pengalan perusahaan, dilanjutkan dengan pembekalan, dan

terakhir para pekerja magang akan dibagi kedalam tim untuk memulai proyeknya masing-masing. Dalam masa pengembangan proyek Sagara Group Web yang dijalani oleh penulis, terdapat tiga tahapan utama antara lain pengembangan *web* utama, pengembangan *web* admin, dan pengembangan *web investor*. Pada tabel 3.1 di bawah ini, telah diperinci tanggal pelaksanaan dari masing-masing rangkaian kegiatan yang dijalankan selama program kerja magang ini berlangsung, yaitu selama 22 minggu atau selama lima bulan mulai dari tanggal 13 Juni 2022 hingga tanggal 13 November 2022.

Tabel 3. 1 Realisasi Kerja Magang

No	Pekerjaan Yang Dilakukan	Minggu	Tanggal Mulai	Tanggal Selesai
1.	*Onboarding* dan Pengenalan Perusahaan	1	13 Juni 2022	17 Juni 2022
2.	Kegiatan Pembekalan *Student Trainee*	2-8	20 Juni 2022	5 Agustus 2022
3.	Pembagian Tim dan Pemahaman Proyek	9-10	8 Agustus 2022	19 Agustus 2022
4.	Pengembangan Proyek Tahap 1 (*Web* Utama):	11-15	22 Agustus 2022	23 September 2022
	4.1 Persiapan Pengembangan	11	22 Agustus 2022	26 Agustus 2022
	4.2 Halaman O*ur Business*	12	29 Agustus 2022	2 September 2022
	4.3 Halaman *Services*	13	5 September 2022	9 September 2022
	4.4 *Consume API*	13-14	5 September 2022	16 September 2022
	4.5 Finalisasi	15	19 September 2022	23 September 2022
5.	Pengembangan Proyek Tahap 2 (*Web* Admin):	14-17	12 September 2022	7 Oktober 2022
	5.1 Persiapan Pengembangan	14-15	12 September 2022	23 September 2022
	5.2 Halaman *Our Business*	16	26 September 2022	30 September 2022
	5.3 Halaman *News*	17	3 Oktober 2022	7 Oktober 2022
	5.4 *Consume API*	16-17	26 September 2022	7 Oktober 2022
6.	Pengembangan Proyek Tahap 3 (*Web Investor*):	18-22	10 Oktober 2022	11 November 2022

No	Pekerjaan Yang Dilakukan	Minggu	Tanggal Mulai	Tanggal Selesai
	6.1 Halaman *Business Total Revenue*	18-20	10 Oktober 2022	28 Oktober 2022
	6.2 Halaman *Our Business*	21-22	31 Oktober 2022	11 November 2022

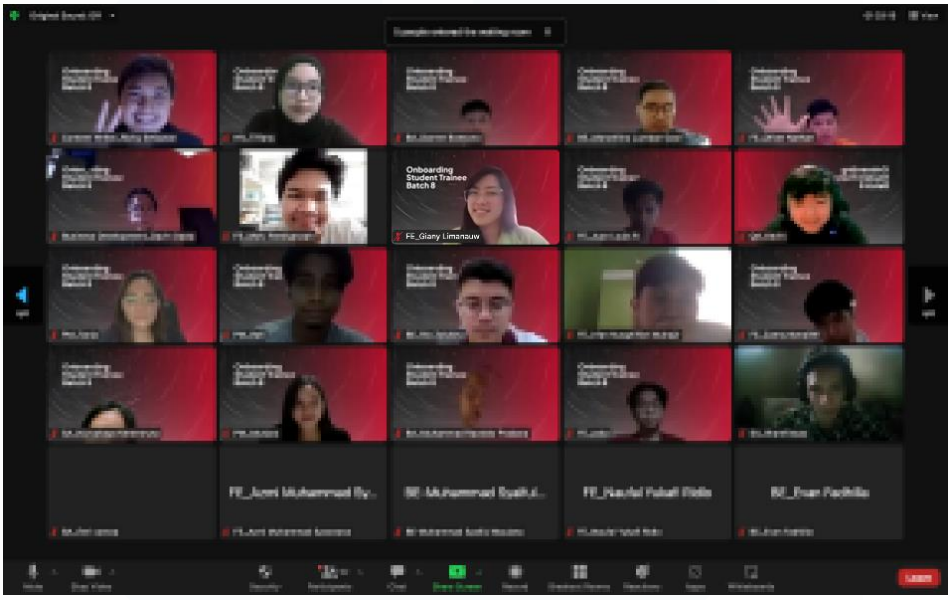
Berikut merupakan uraian seluruh pekerjaan yang dilakukan selama masa program kerja magang berlangsung di PT Sagara Asia Teknologi:

3.2.1 *Onboarding* dan Pengenalan Perusahaan

Penulis yang telah menerima surat penerimaan kerja magang di PT Sagara Asia Teknologi diundang untuk mengikuti acara pembukaan program magang PT Sagara Asia Teknologi *batch* ke 8 atau disebut dengan acara *Onboarding*. Acara ini diperuntukan untuk para pekerja magang atau disebut sebagai *Student Trainee* sebagai bentuk pengenalan perusahaan dan penyuluhan bagaimana rangkaian kegiatan pada program kerja magang ini akan dilaksanakan. Acara *Onboarding* ini diadakan pada hari pertama kontrak kerja dimulai yaitu pada tanggal 13 Juni 2022 melalui *platform* Zoom seperti pada gambar 3.2 di bawah. Pada acara ini bukan sekedar pengenalan perusahaan dan penyuluhan, melainkan ajang untuk saling mengenal sesama *Student Trainee* di *batch* ke 8 ini. Pengenalan ini terjalin melalui permainan-permainan daring yang menyenangkan dan membangun suasana hingga yang membuat acara ini menjadi sebuah kesuksesan.

Selama minggu pertama program kerja magang dimulai, selain pengenalan kegiatan program kerja magang dan perusahaan, penulis juga dikenalkan kepada mentor yang bertanggung jawab untuk membantu dan mengarahkan penulis selama bekerja di PT Sagara Asia Teknologi yaitu Bapak Muhammad Iqbal. Kemudian penulis juga mendapatkan *introduction* mengenai kultur kerja di PT Sagara Asia Teknologi yaitu kolaboratif, komunikatif, *positive vibes*, profesional, dan tentunya harus selalu berusaha untuk memberikan yang terbaik. Selain itu, selama minggu pertama ini juga penulis mengikuti pertemuan online bersama para *student*

trainee di divisi *Front-End* dan *Back-End* untuk saling mengenal satu sama lain serta diberikan *mentoring* oleh para mentor mengenai hal-hal yang perlu dipersiapan sebelum masa pembekalan.



Gambar 3. 2 Acara *Onboarding Student Trainee Batch 8*

3.2.2 Kegiatan Pembekalan *Student Trainee*

Masa pembekalan *Student Trainee* ini berlangsung selama hampir dua bulan yang dimulai pada tanggal 20 Juni 2022 hingga tanggal 5 Agustus 2022. Pada masa pembekalan ini, penulis diberikan beberapa materi yang perlu dipelajari baik secara mandiri maupun melalui kelas daring yang disediakan oleh PT Sagara Asia Teknologi. Tujuan dilakukannya kegiatan pembekalan ini adalah untuk memudahkan *Student* Trainee dalam menyelesaikan pekerjaan-pekerjaan yang diberikan selama keberlangsungan program kerja magang ini.

Masing-masing divisi diberikan materi yang berbeda-beda sesuai dengan pekerjaannya. Selain yang berhubungan dengan pekerjaan, *Student Trainee* juga diberikan materi-materi yang *up to date* dengan industri teknologi saat ini yang mungkin akan berguna pada masa mendatang seperti *blockchain technology* pada *web 3.0*. Berikut di bawah ini

merupakan uraian mengenai materi-materi yang dipelajari selama kegiatan pembekalan *Student Trainee* berlangsung:

1) *Web 3.0*

Berdasarkan materi yang dibawakan oleh Bapak Diifa Agamas pada kelas daring *course 1 – Student Trainee intensive learning, web 3.0* merupakan sebuah teknologi berbasis *web* generasi berikutnya yang memiliki fokus utama pada pengembangan penggunaan *machine learning* serta *artificial intelligence. Web 3.0* memiliki tiga karakteristik yaitu *open, trustless,* dan *permissionless*. *'Open'* karena *web 3.0* dibangun menggunakan perangkat lunak *open-source* yang *accessible* dan terbuka. *Web 3.0* juga memiliki karakteristik *'Trustless'* karena *web 3.0* dapat memungkinkan pengguna untuk berinteraksi baik secara publik ataupun pribadi tanpa menggunakan *third-party* yang perlu dipercaya. Selain itu, *web 3.0* juga merupakan sebuah teknologi yang memungkin penggunanya dapat berpartisipasi tanpa memerlukan otoritas dari badan pengatur mana pun atau disebut dengan *'Permissionless'*.

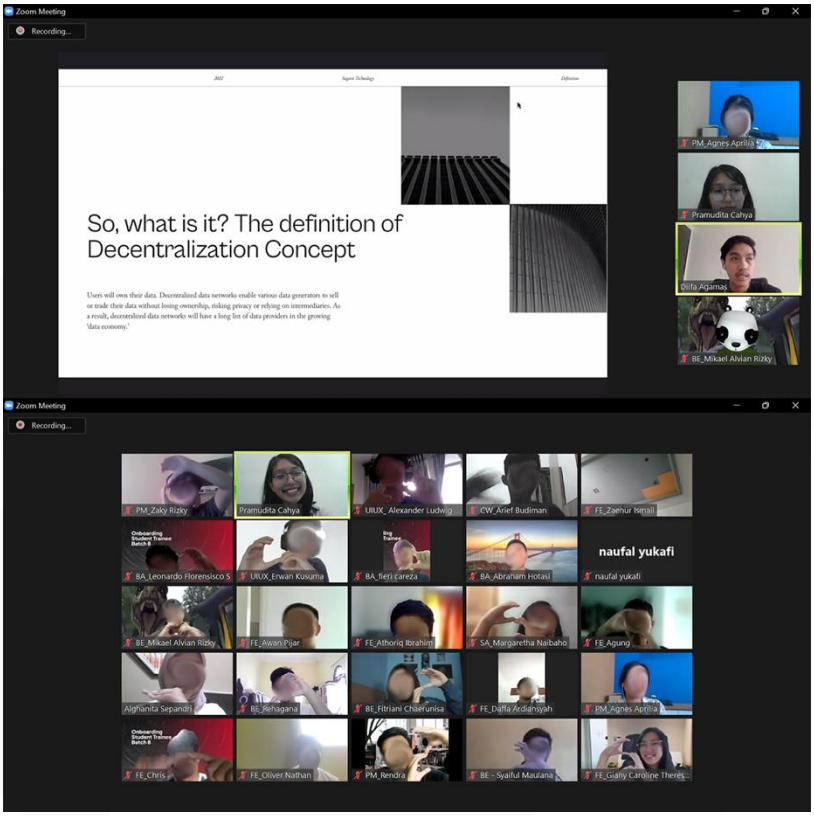
Secara fundamental, *web 3.0* dapat memperluas ruang lingkup interaksi antara manusia dengan teknologi atau mesin dan bahkan dapat melampaui apa yang dapat kita bayangkan sejauh ini. Interaksi yang dapat dilakukan menggunakan keunggulan *web 3.0* seperti melakukan pembayaran tanpa batas dan bahkan dapat melakukan transfer data dengan tingkat keamanan yang jauh lebih tinggi. Dengan menggunakan *web 3.0,* pengguna juga dapat melakukan intersaksi kesesama pengguna dan bahkan dapat berinteraksi dengan mesin apapun di seluruh dunia tanpa memerlukan perantara.

Dalam penggunannya, *web 3.0* memungkinkan para penghasil data untuk melakukan barter data tanpa

mengkhawatirkan kehilangannya kendali atas kepemilikan datanya dan melepas privasi atau memiliki ketergantungan terhadap pihak perantara. Hal ini dapat dilakukan karena *web 3.0* memiliki kelebihan yaitu jaringan data yang terdesentralisasi. Maka dengan kehadiran *web 3.0*, jaringan-jaringan data yang terdesentralisasi ini dapat mengumpulkan seluruh penghasil data kedalam *data economy*.

Dengan adanya *web 3.0* pengguna juga dapat menggunakan sebuah teknologi yang disebut dengan *smart contracts* yang dapat digunakan untuk melakukan transasksi melalui *blockchain technology*. Transaksi yang dilakukan menggunakan *smart contracts* yang berupa kode digital tanpa kertas ini dapat memberikan penawaran mengenai serangkaian janji dengan kondisi yang telah di tentukan dan disepakati oleh pihak yang bersangkutan. Dengan menggunakan *smart contracts*, transaksi dapat dilakukan tanpa memerlukan otoritas pihak perantara untuk melakukan verifikasi dan tentunya aman.

Menurut pengajar yang membawakan materi, teknologi *web 3.0* ini sudah bukan lagi teknologi yang terbilang sangat baru, melainkan sudah lahir selama beberapa tahun silam. Penerapan *web 3.0* pun sudah cukup banyak beredar, terutama pada *website-website* yang digunakan untuk melakukan transaksi Cryptocurrency. Maka dengan adanya kelas daring ini, *Student Trainee* diharapkan dapat terus mengikuti perkembangan di dunia teknologi dan mempelajarinya. Gambar 3.3 di bawah ini merupakan foto-foto yang diambil selama kegiatan kelas daring *course 1 – Student Trainee intensive learning* dengan topik *web 3.0*



Gambar 3. 3 *Course* 1 – *Student Trainee Intensive Learning*

2) HTML & CSS

Pada masa pembekalan ini, penulis yang bekerja sebagai *Front-End Engineer* disarankan oleh mentor divisi *Front-End Engineer* untuk mengulas kembali materi seputar HTML dan CSS. Hal ini dilakukan agar kedepannya, penulis dapat lebih mudah meresap materi-materi yang lebih kompleks dan memudahkan penulis ketika pengembangan proyek dimulai nantinya. Berikut di bawah ini merupakan beberapa pembelajaran yang diterima selama mendalami dan mengulas kembali kedua materi ini.

HTML merupakan singkatan dari *HyperText Markup Language*. Jika dipecah maka *HyperText* merupakan potongan-potongan teks yang dapat ditaukan ke dokumen lain di dalam *website*. Sedangkan *Markup Languange* dapat digambarkan seperti kode-kode atau bahasa yang dapat digunakan untuk

memberikan tanda bahwa bagian tersebut perlu di *bold* atau bagian tersebut perlu di *italic* atau lain sebagainya. Dalam HTML, *Markup Languange* yang dimaksudkan di atas disebut juga dengan HTML *tags*.

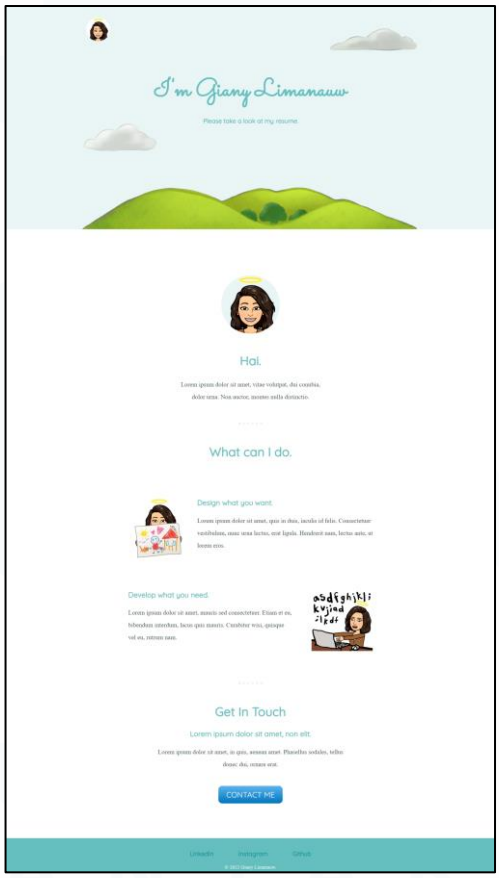
CSS atau *Cascading Style Sheets* merupakan sebuah bahasa yang dipakai untuk mengatur tampilan pada halaman sebuah *website*. Tidak terlepas dari HTML, CSS juga dapat berfungsi untuk mengatasi keterbatasan dalam mengatur tampilan pada HTML. Dengan menggunakan CSS, kode yang ditulis akan lebih singkat sehingga *loading* akan terasa lebih singkat. Selain itu, CSS juga dapat membantu pengelolaan kode serta memungkinkan tampilan *website* menjadi lebih *responsive* dengan ukuran layar pengguna. Gambar 3.4 di bawah ini merupakan logo dari bahasa HTML dan CSS yang dipelajari pada masa pembekalan ini.

Gambar 3. 4 Logo HTML dan CSS

Dalam masa pembekalan materi HTML dan CSS ini, penulis juga diberikan tantangan untuk membuat sebuah *basic web design* dengan sekreatif mungkin. Dalam mengerjakan tugas ini, penulis diharapkan untuk menggunakan sebanyak mungkin materi yang telah dipelajari seperti *tag*, elemen, atribut, dan lain sebagainya. Melalui tugas ini, mentor juga dapat melihat sejauh mana kemampuan dan kreatifitas *Student Trainee* di divisi *Front-End Engineer*.

Tugas yang diberikan ini memberikan dampak yang cukup besar dalam masa pembekalan materi HTML dan CSS. Selama mengerjakan tugas *basic web design*, banyak ilmu non-teori yang membuat pembelajaran menjadi lebih mudah untuk dipahami. Hal yang dipelajari seperti perbedaan jenis-jenis CSS dan mana yang terbaik untuk digunakan, bagaimana cara kerja CSS dan format yang baik dalam penulisannya, perbedaan rinci dari masing-masing *positioning* pada CSS, dan sepenting apa pemilihan *font* dalam mendesain sebuah *website*.

Pada gambar 3.5 di bawah ini merupakan hasil pengerjaan penulis untuk tugas *basic web design* dengan menggunakan HTML dan CSS sesuai dengan apa yang telah dipelajari selama masa pembekalan materi HTML dan CSS.



Gambar 3. 5 Hasil Pengerjaan Tugas *Basic Web Design*

3) JavaScript

Gambar 3. 6 Logo JavaScript

Tahap selanjutnya dalam masa pembekalan ini adalah mengulas kembali materi mengenai JavaScript secara mandiri. Pada gambar 3.6 di atas merupakan logo dari bahasa pemrograman JavaScript. Berikut ini merupakan rangkuman materi yang telah dipelajari dan dipahami kembali selama masa pembekalan materi JavaScript:

JavaScript pada umumnya tidak dapat terlepas dari HTML dan CSS. Masing-masing dari bahasa ini digunakan untuk fungsi yang berbeda dan bertujuan untuk saling melengkapi satu dan lainnya. HTML bertanggung jawab atas isi dari *website* tersebut atau konten di dalamnya. Sedangkan untuk CSS bertanggung jawab atas bagaimana penampilan dari *website* tersebut. Lalu, JavaScript digunakan untuk memungkinkan *website* tersebut melakukan sesuatu.

JavaScript merupakan salah satu bahasa pemrograman yang paling populer hingga saat ini. Bahasa ini sering digunakan untuk membangun *website* dengan konten yang dinamis seperti animasi, *slideshow*, dan lain sebagainya [4]. JavaScript memiliki beberapa keunggulan yang tidak dimiliki bahasa lain, antar lain:

a) Mudah dipelajari karena struktur penulisan pada JavaScript terbilang cukup sederhana dan *error* yang muncul lebih mudah untuk diidentifikasi.

b) Lebih cepat dalam mengeksekusi *script* karena JavaScript dapat melakukannya langsung di browser tanpa terkoneksi pada server.
c) Kompatibel dengan berbagai bahasa lain selain HTML seperti PHP, Java, dan lainnya.
d) JavaScript dapat mengurangi beban pada server karena dapat mengeksekusi *script* langsung pada browser, maka permintaan yang dikirim ke server pun akan berkurang.
e) Selalu *up to date*, JavaScript secara rutin terus dikembangkan dan mengeluarkan *framework* serta *library* baru [5].

4) Git dan GitHub

Gambar 3. 7 Logo Git dan GitHub

Tahap selanjutnya dalam masa pembekalan ini adalah mempelajari cara penggunaan Git dan GitHub secara mandiri. Pada gambar 3.7 di atas merupakan logo dari *platform* Git dan GitHub. Berikut ini merupakan rangkuman materi yang telah dipelajari dan dipahami selama masa pembekalan materi Git dan GitHub:

Git adalah sebuah *software* yang berfungsi untuk menyimpan segala perubahan yang terjadi pada sebuah proyek. Sedangkan GitHub adalah sebuah layanan berbasis *cloud* yang digunakan menyimpan *file* proyek tersebut kedalam sebuah tempat penyimpanan yang disebut *repository*. Pada dasarnya, keduanya memiliki konsep yang sama yaitu menulis dan

mengelola *source code* yang dapat dilakukan baik secara individu maupun secara tim. Namun, yang dapat membedakannya adalah antara lain:

a) Git merupakan sebuah *software* yang artinya memerlukan sebuah instalasi ke penyimpanan lokal, sedangkan GitHub merupakan sebuah layanan berbasis *cloud* yang tidak memerlukan penyimpanan lokal.
b) Git memiliki fungsi yang berfokus pada *version control* dan *code sharing*, sedangkan GitHub berfokus pada *source code hosting* secara terpusat.
c) Secara sederhana dapat dilihat juga bahwa Git hanya dapat diakses secara *offline* dan GitHub hanya dapat diakses secara *online* [6].

Dalam penggunaannya, Git dan GitHub sangat bermanfaat untuk melakukan pengembangan sistem secara tim. Kemampuan *code sharing* dan *version control* memungkinkan penggunanya untuk mengerjakan sebuah proyek secara daring. Terdapat beberapa istilah dan fitur yang dipelajari selama mempelajari cara kerja Git dan GitHub:

a) *Remote Repository* adalah sebuah *repository* yang digunakan untuk menyimpan *code* dan dapat menggunakan *repository* milik pribadi, pengguna lain, atau bahkan server lain [7]. Dalam penggunaanya, *remote repository* dapat menyimpan perubahan-perubahan yang terjadi pada *local repository* ke dalam penyimpanan *cloud* di GitHub sehingga seluruh tim atau pengguna lain dapat mengakses segala perubahan dan *file* yang ada di dalamnya.

b) *Gitignore* adalah sebuah *file* yang berisi aturan-aturan yang digunakan untuk mencegah terjadinya *committing* terhadap beberapa *file* ke dalam *repository*.
c) *Clone* adalah cara yang dapat dilakukan ketika ingin menarik sebuah *remote repository* ke komputer lokal. Dengan melakukan *cloning*, seluruh *commit* yang pernah terjadi atau seluruh versi dari *repository* tersebut akan ikut masuk ke komputer lokal.
d) *Commit* adalah perintah yang digunakan untuk menyimpan perubahan yang terjadi pada *file*.
e) *Branching* adalah membuat cabang dari hasil pengerjaan cabang utama dan dapat digunakan sebagai cabang untuk melakukan eksperimen. Dengan melakukan *branching*, perubahan yang terjadi pada cabang eksperimen ini tidak akan berpengaruh pada cabang utama sehingga dapat meminimalisir terjadinya kerusakan pada cabang utama pengerjaan proyek.
f) *Merge* adalah sebuah perintah yang digunakan untuk menggabungkan dua *branch* menjadi satu. Seluruh *commit* yang terjadi pada kedua cabang tersebut akan tetap tercatat dan terkumpul pada satu cabang utama.
g) *Push* adalah sebuah perintah yang digunakan untuk mendorong perubahan-perubahan yang ada atau seluruh *commit* yang terjadi pada *local repository* ke dalam *remote repository*.
h) *Pull* adalah kebalikan dari perintah *push* yang dimana berfungsi untuk menarik segala perubahan yang terjadi pada *remote repository* ke dalam *local repository*.

5) React JS

Gambar 3. 8 Logo React JS

Tahap selanjutnya dalam masa pembekalan ini adalah mempelajari mengenai *framework* ReactJS secara mandiri. Pada gambar 3.8 di atas merupakan logo dari *framework* ReactJS. Berikut ini merupakan rangkuman materi yang telah dipelajari dan dipahami selama masa pembekalan materi ReactJS:

ReactJS merupakan sebuah JavaScript *library* yang dapat digunakan untuk membangun *user interface* (UI) yang indah dengan lebih mudah. Dalam penggunaannya, isi dari ReactJS yang merupakan kode-kode JavaScript yang sudah ditulis sedemikian rupa sehingga ketika ingin menggunakannya, hal yang perlu dilakukan adalah mengambilnya dari dalam *libarary* tersebut. Sehingga ketika ingin membangun sebuah aplikasi berbasis *web* menggunakan ReactJS, hal yang akan terjadi adalah *framework* ini akan membongkar struktur UI yang kompleks menjadi sebuah komponen yang berdiri sendiri.

6) Tailwind CSS

Gambar 3. 9 Logo Tailwind CSS

Tahap selanjutnya dalam masa pembekalan ini adalah mempelajari mengenai Tailwind CSS secara mandiri. Pada

gambar 3.9 di atas merupakan logo dari Tailwind CSS. Berikut ini merupakan rangkuman materi yang telah dipelajari dan dipahami selama masa pembekalan materi Tailwind CSS:

Tailwind CSS merupakan sebuah *framework* bersifat *utility first* yang artinya digunakan untuk membangun tampilan antarmuka dengan lebih cepat dan sederhana. Dalam penggunaannya, Tailwind CSS juga lebih ramah terhadap *framework* ReactJS yang merupakan *component-base framework*. Hal tersebut dikarena Tailwind CSS dapat mengekstraksi kelas komponen yang berasal dari pola-pola *utility* berulang [8].

Penggunaan Tailwind CSS terbilang cukup sederhana seperti *framework* lainnya. Menurut dokumentasi yang sertakan pada situs resminya, Talwind CSS dapat diunduh melalui NPM atau *Node Package Manager* pada terminal dan kemudian dapat digunakan pada proyek. Ketika menggunakan Tailwind CSS, kode yang digunakan akan dituliskan langsung pada *class* seperti melakukan *inline* CSS. *Class* yang dapat digunakan pada Tailwind CSS pun sangat lengkap dan beragam, seperti *class* untuk mengatur *layout, grid, spacing, sizing, typography, backgrounds, borders, effects, filters, tables, transition, transforms,* dan *interactivity* [9].

7) *Application Programming Interface* (API)

Tahap selanjutnya dalam masa pembekalan ini adalah mempelajari mengenai API secara mandiri. Berikut ini merupakan rangkuman materi yang telah dipelajari dan dipahami selama masa pembekalan materi API:

API atau *Application Programming Interface* adalah sebuah antarmuka yang dapat digunakan sebagai perantara yang menghubungkan sebuah aplikasi dengan aplikasi lain walau berada di *platform* yang berbeda. API terdiri dari empat jenis,

yaitu *public, private, partner,* dan *composite*. Sebagai contoh, API *public* biasanya berupa data cuaca yang dapat diakses oleh semua pihak. API *private* biasanya berupa data internal yang tidak dapat diakses secara umum. API *partner* bisa dikatakan sebagai gabungan antara *public* dan *private* yang dimana dapat digunakan untuk kepentingan umum namun hanya untuk yang memiliki izin. Sedangkan API *composite* digunakan untuk menyimpan data yang berasar dari server yang berbeda-beda kedalam satu tempan penyimpanan.

3.2.3 Pembagian Tim dan Pemahaman Proyek

Student Trainee yang telah berhasil melalui masa pembekalan selama tujuh minggu atau hampir dua bulan pada akhirnya dibagi kedalam beberapa tim yang masing-masing akan bertanggung jawab terhadap sebuah proyek. Setiap *Project Manager* pada masing-masing tim diberikan kesempatan untuk memilih proyek mana yang akan dikerjakan bersama dengan tim. Terdapat 10 pilihan proyek yang diberikan, beberapa diantaranya merupakan proyek milik PT Sagara Asia Teknologi yang baru akan dimulai dan ada beberapa yang sudah sementara berjalan, serta ada juga proyek yang berasal dari *client* PT Sagara Asia Teknologi. Pilihan proyek yang diberikan antara lain:

1) Hackathon

   Proyek ini merupakan bentuk kolaborasi antara PT Sagara Asia Teknologi dengan perusahaan teknologi lain yang tidak ingin disebutkan namanya. Proyek kolaborasi ini ditujukan untuk pembangunan situs resmi yang akan digunakan sebagai penyalur informasi dari acara reguler Hackathon. Acara ini merupakan sebuah ajang perlombaan pemrograman yang diikuti oleh *programmer* atau bisa juga diikuti oleh orang

yang berasal dari industri lain namun ingin melakukan kolaborasi dengan dunia teknologi.

2) Indonesia Future

Proyek Indonesia Future merupakan pengembangan situs *job portal* yang akan dikhususkan untuk penyedia dan pencari kerja dibidang IT, terutama untuk pekerja lepas atau sering disebut dengan *freelance* dibidang IT.

3) Sagara Foundation

Sagara Foundation merupakan sebuah proyek pengembangan aplikasi berbasis *mobile* dan *web* milik PT Sagara Asia Teknologi. Proyek ini digunakan sebagai situs resmi yayasan yang dibangun oleh PT Sagara Asia Teknologi yaitu Sagara Foundation. Yayasan ini dibangun sebagai bentuk *Corporate Social Responsibility* (CSR) dari PT Sagara Asia Teknologi yang memiliki tujuan mulia yaitu memberi bantuan kepada anak-anak yatim piatu dibidang pendidikan teknologi. Hal ini dilakukan karena PT Sagara Asia Teknologi yang memiliki dasar dibidang teknologi juga memiliki mimpi untuk meningkatkan pendidikan teknologi di generasi muda yang akan sangat berdampak terhadap kemajuan negara Indonesia.

4) Total Politics

Proyek ini merupakan proyek yang berasal dari *client* PT Sagara Asia Teknologi yaitu Total Politics. Pada proyek ini, Total Politics ingin membangun sebuah *website* resmi yang dapat digunakan untuk bertukar dan mencari informasi serta berita-berita terkini yang berhubungan dengan dunia politik.

5) Techspace

Proyek ini merupakan sebuah proyek pengembangan *platform* berbasis *website* dan *mobile apps* untuk mencari berita seputar teknologi yang berskala internasional.

Techspace tidak hanya menyuguhkan berita-berita populer dan terkini, Techspace juga menyediakan jurnal-jurnal penelitian akademik yang berhubungan dengan teknologi. Dengan kehadiran Techspace, *Digital Talent* beserta orang-orang yang ingin mencari informasi terkini dan relevan dengan dunia teknologi akan jauh dipermudah.

6) Mom & Kids Apps

Proyek ini merupakan proyek yang dibangun untuk keperluan *client* dari PT Sagara Asia Teknologi. Mom & Kids Apps adalah sebuah aplikasi berbasis *mobile* yang ditujukan untuk *memonitoring* kesehatan ibu dan bayi. Aplikasi ini dapat *memonitoring* kondisi sang ibu dan buah hatinya mulai dari masa kehamilan hingga pasca melahirkan. Selain fitur *monitoring,* aplikasi ini juga memiliki fitur konsultasi *real-time* dengan dokter kandungan secara *virtual* dan juga aplikasi ini dapat dikoneksikan pada sebuah *wearable device* yang dapat digunakan untuk deteksi dini jika terdapat kejanggalan pada sang ibu.

7) Sagara Group Web

Proyek Sagara Group Web merupakan sebuah *website* resmi yang dimiliki oleh Sagara Group. Pengembangan *website* ini ditujukan untuk memberikan informasi terkini mengenai pergerakan dan perkembangan unit bisnis yang berada di bawah Sagara Group kepada *investor-investor* yang terlibat. Informasi yang ditampilkan mulai dari profil, data keuangan perusahaan, hingga laman investasi.

8) HRIS

Human Resource Integrated System atau disingkat dengan HRIS merupakan sebuah *platform* yang digunakan untuk membantu mengelola pekerjaan dari departemen *People & Culture* (HR). Pekerjaan yang dikelola oleh *platform* HRIS ini

dimulai dari kegiatan *recruitment*, seluruh *database* karyawan beserta performanya, dan *platform* ini juga mengelola evaluasi performa karyawan.

9) Ateeat

Proyek ini adalah pembangunan sebuah situs yang menyediakan layanan pesan antar *web 3.0* yang menggunakan NFT sebagai token *rewards* untuk para pengguna baik *seller* maupun *customer*. Ateeat juga merupakan sebuah situs yang dipesan oleh *client* PT Sagara Asia Teknologi yang berasal dari Amerika Serikat, sehingga proyek ini juga dibangun dengan tujuan penggunaan di area Amerika Serikat.

10) Speadgear

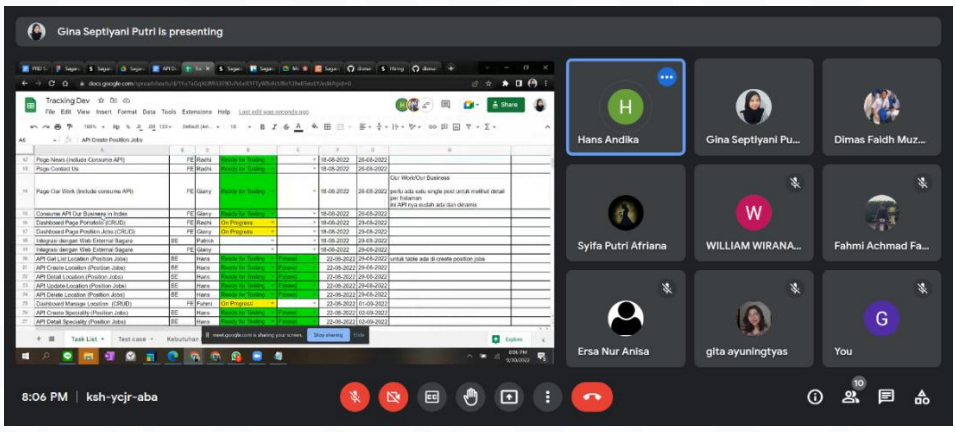
Proyek ini merupakan sebuah *platform* digital yang dibangun dengan tujuan sebagai media penyalur NFT dan juga Metaverse yang saat ini masih ramai diperbincangkan. NFT dan Metaverse juga dipercaya akan menjadi dunia digital yang mengubah masa depan dunia akan dunia maya.

Pada kesempatan kali ini, tim kami memilih untuk mengerjakan proyek Sagara Group Web. Selama dua minggu ini, tim Sagara Group Web rutin melakukan pertemuan secara daring untuk membahas tuntas detail-detail mengenai keperluan pengembangan proyek Sagara Group Web. Proyek ini merupakan sebuah proyek pembangunan *website* internal perusahaan yang digunakan untuk menyalurkan informasi kepada para *investor*. Informasi yang dibagikan melalui *website* ini merupakan informasi mengenai keungan dan perkembangan setiap unit bisnis yang berada di bawah naungan Sagara Group atau PT Sagara Asia Teknologi.

Website ini terbagi menjadi tiga bagian yaitu *website utama* yang merupakan tampilan awal ketika *user* memasuki *website*. Kemudian, setelah *user* melakukan *login* pada *website* utama, *user* akan diberkan pilihan untuk *login* sebagai *investor* atau sebagai admin. Jika *user* terdaftar

sebagai admin maka *user* akan masuk ke *website* admin. Namun, apabila *user* terdaftar sebagai *investor* maka *user* akan masuk ke *website investor*. *Website* admin hanya bisa digunakan oleh admin dari masing-masing unit bisnis dan untuk *website investor* hanya dapat digunakan oleh *investor-investor* untuk mengakses informasi mengenai perkembangan unit bisnis yang diinvestasikan.

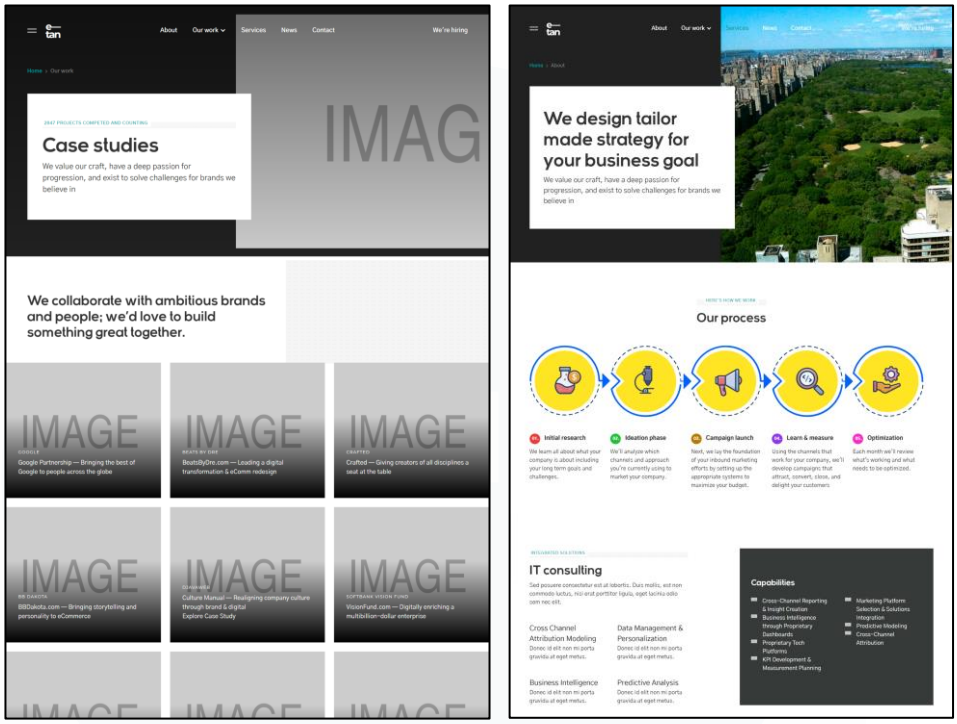
Project Manager memimpin dan memulai proyek ini dengan memberikan tugas untuk masing-masing divisi setelah semua detail mengenai Sagara Group Web telah terkumpul. *Business Analyst* dan *System Analyst* mulai menyusun *Product Requirement Document* (PRD) untuk menjadi fondasi pembangunan proyek Sagara Group Web. Kemudian, *UI/UX Designer* juga mulai mengerjakan desain antarmuka yang akan digunakan dalam *website* Sagara Group. Lalu, untuk tim *developer* yaitu *Front-End Engineer* dan *Back-End Engineer* mulai untuk mempelajari dan mengembangkan *website* Sagara Group. Selanjutnya, untuk *Quality Assurance* dapat memulai melakukan *testing* terhadap halaman *website* yang sudah dikerjakan oleh tim *developer*. Semua tugas ini akan di-*monitoring* setiap minggu oleh *Project Manager* dan dilaporkan kepada *Product Owner* setiap progress yang dicapai oleh tim Sagara Group Web. *Monitoring* setiap minggu ini dilakukan melalui *platform* Google Meet seperti yang ditampilkan pada gambar 3.10 di bawah ini.



Gambar 3. 10 *Weekly Meeting* Tim Sagara Group Web

Pada masa pemahaman proyek, *Front-End Engineer* dibekali dengan *design markup* yang berisi kumpulan *file* HTML untuk menggambarkan tampilan antarmuka yang telah disiapkan oleh *Project Owner*. Sebelum memulai pengembangan proyek Sagara Group Web ini, *design markup* diberikan untuk digunakan sebagai pedoman dasar apa yang dikerjakan nantinya dalam masa pengembangan proyek. Pada *design markup* terdapat 10 *file* HTML, yaitu *about.html, contact-us.html, hiring.html, index.html, login.html, news.html, our-work.html, our-work-2.html, services.html,* dan *single-post.html*. Semua *file* HTML ini bukan merupakan desain tampilan antarmuka final yang diinginkan oleh *Project Owner* karena desain pada setiap *file* masih belum konsisten. Maka, dengan ini *Front-End Engineer* pun ditugaskan untuk memilih desain yang dapat diikuti dan desain yang masih perlu disesaikan lagi agar Sagara Group Web dapat memberikan *user interface* dan *user expirence* yang terbaik.

Berikut gambar 3.11 di bawah ini merupakan beberapa tampilan dari dokumen *markup* yaitu *file our-work.html* dan *services.html* yang diberikan oleh *Project Owner* melalui *Project Manager* kepada *Front-End Engineer* sebelum memulai pengembangan proyek tahap satu yaitu *web* utama.



Gambar 3. 11 Desain *Markup* Halaman *Our Business* dan *Services*

3.2.4 Pengembangan Proyek Tahap 1 (*Web* Utama):

3.2.4.1 Persiapan Pengembangan

Pada masa pengembangan proyek tahap pertama, langkah awal yang diambil adalah melakukan persiapan pengembangan. Persiapan yang dimaksudkan adalah memahami *file web* utama yang diberikan oleh *Project Owner* melalui *repository* GitHub. Dalam *repository* tersebut telah diberikan instruksi dan informasi terkait pengerjaan proyek ini melalui README *file*. Instruksi yang diberikan adalah langkah yang harus dilakukan untuk menjalankan *website* di dalam *repository* tersebut pada komputer lokal. Sedangkan untuk informasi yang diberikan melalui README *file* adalah mengenai *atomic design methodoly repository* agar struktur *folder* tetap teratur dan sesuai dengan ketentuan.

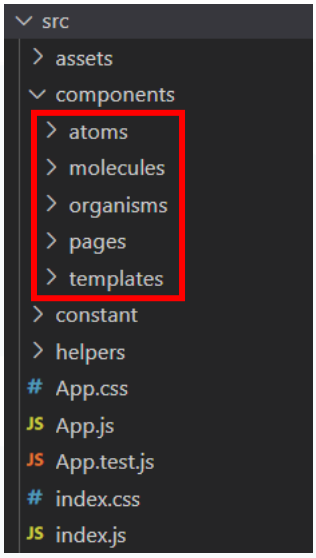
Berdasarkan instruksi yang diberikan oleh *Project Owner* pada README *file*, hal pertama yang harus dilakukan adalah melakukan *cloning file repository* tersebut. Selanjutnya, agar *file*

yang sudah di *cloning* dapat terhubung langusng dengan *repository* di GitHub, perlu menambahkan yang namanya *remote* melalui Git. Dengan menambkan *remote* kedalam *repository* lokal maka ketika melakukan pemrograman, perubahan yang dilakukan dapat langsung di dorong kedalam *repository* di GitHub dan *developer* lain bisa langsung mengakses perubahan yang dibuat. Selanjutnya, agar *website* dapat dijalankan, perlu melakukan instalasi Node Modules dengan menggunakan terminal *repository* tersebut di komputer lokal. Setelah selesai melakukan instalasi Node Modules, *website* sudah bisa dijalankan dengan menuliskan 'npm start' pada terminal.

Setelah *file* proyek yang diberikan oleh *Project Owner* sudah siap untuk dilakukan pengembangan, selanjutnya penulis melakukan analisa dan pemahaman akan *code* yang telah ada yaitu *code landing page*. Pada *code landing page*, hal yang dipelajari adalah koneksi-koneksi antara beberapa *file* yang terhubung karena menggunakan struktur *folder atomic design methodoly repository*. Dikarenakan strutur *folder* tersebut, untuk membangun satu tampilan *web* ini memerlukan beberapa *file* yang tersebar di berbagai *folder* sesuai dengan fungsinya.

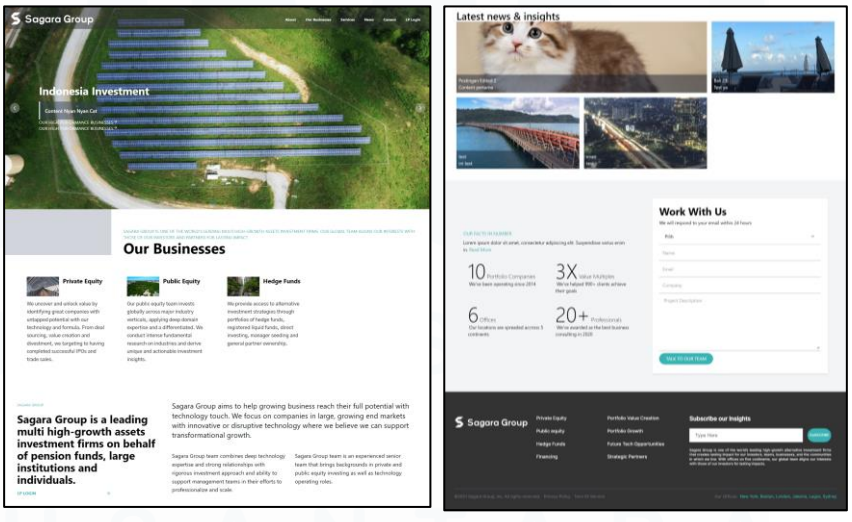
Berdasarkan informasi yang diberikan melalui README *file* pada *repository* GitHub, *atomic design methodoly repository* merupakan metode penyimpanan komponen-komponen penyusun tampilan *website* yang terdiri dari lima tingkatan *folder* penyimpanan, yaitu *atoms, molecules, organisms, templates*, dan *pages* seperti pada gambar 3.12 di bawah. Setiap *folder* memiliki fungsi penyimpanannya masing-masing, fungsi yang dimaksud adalah sebagai berikut:

1) *Folder atoms* untuk menyimpan komponen-komponen HTML sederhana seperti *button, form, label, input,* dan masih banyak lagi.
2) *Folder molecules* digunakan untuk menyimpan komponen-komponen yang merupakan gabungan dari elemen sederhana pada *folder atoms* yang dapat berfungsi bersama sebagai sebuah unit. Komponen yang dapat di simpan pada *folder molecules* seperti *card* yang di dalamnya terdapat elemen *button* dan *form*.
3) *Folder organisms* digunakan untuk menyimpan komponen-komponen tampilan antarmuka *website* yang jauh lebih kompleks. Seluruh isi yang ingin ditampilkan dalam *website* akan dimasukan kedalam *file* yang disimpan pada *folder* ini mulai dari kumpulan *molecules* dan *atoms* hingga struktur dan isi tampilan *website*.
4) *Folder templates* berfungsi untuk memposisikan komponen-komponen yang dipakai berulang-ulang dan konsisten pada beberapa atau semua halaman. Pada *folder* ini, komponen yang dapat disimpan adalah seperti *navbar, footer, loader, sidebar,* dan lain sebagainya.
5) *Folder pages* merupakan tempat untuk menyimpan *file* yang digunakan sebagai penggabung tampilan antarmuka. *File* yang telah dibuat pada *folder organisms* dan *templates* digabungkan hingga akhirnya dapat menghasilkan tampilan *web* yang diinginkan.



Gambar 3. 12 *Atomic Design Methodology Repository*

Sesuai dengan ketentuan yang sudah ditetapkan oleh *Project Owner*, para proyek Sagara Group Web ini bahasa pemrograman yang digunakan adalah JavaScript dan *framework* yang digunakan adalah React JS. Kemudian, pada proyek ini juga menggunakan *open-source CSS framework* yaitu Tailwind CSS. Berikut pada gambar 3.13 di bawah ini merupakan tampilan dari *landing page* pada *repository* GitHub yang telah disiapkan oleh *Project Owner* untuk proyek Sagara Group Web.



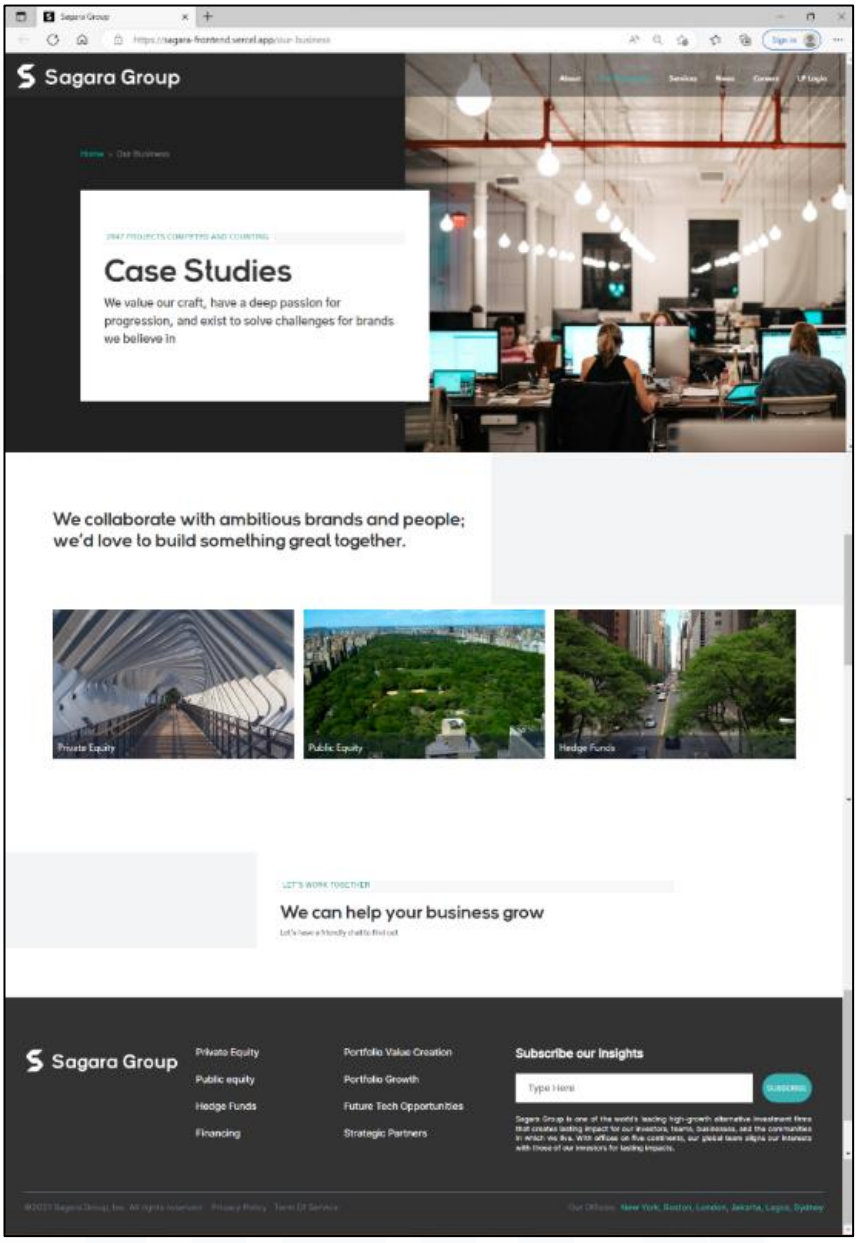
Gambar 3. 13 Hasil *Running Code Landing Page*

3.2.4.2 Halaman *Our Business*

Tugas pertama yang diberikan oleh *Project Manager* setelah divisi *Front-End Engineer* selesai melakukan persiapan adalah membangun halaman-halaman yang dibutuhkan pada *web* utama. Pada *web* utama, halaman-halaman yang akan dibangun selain *landing page* adalah halaman *about, our business, services, news, careers,* dan halaman *login*. Halaman-halaman *website* ini dibangun berdasarkan gambaran tampilan antarmuka yang telah diberikan melalui *file markup* pada masa pemahaman proyek.

Pada saat pembagian tugas yang dilakukan oleh *Project Manager*, halaman-halaman yang harus dikerjakan oleh penulis adalah halaman *our business* dan halaman *services*. Halaman *our business* merupakan halaman yang dikerjakan terlebih dahulu pada masa pengembangan proyek tahap satu ini. Fungsi dari halaman ini adalah sebagai sumber informasi mengenai unit bisnis yang berada di bawah naungan PT Sagara Asia Teknologi. Dengan memberikan data mengenai unit bisnis perusahaan, maka akan lebih mudah untuk menarik *investor* untuk memahami skala perusahaan seperti apa yang akan mereka investasikan.

Proses pengerjaan halaman *our business* dimulai dengan memuat struktur serta isi dari tampilan halaman pada *folder organisms*. Mulai dari isi konten, warna, penempatan objek, font, dan lain sebagainya. Kemudian, pada *folder pages* dilakukan penggabungan antara *file* pada *folder organisms* halaman *our business* dengan *templates* yang sudah disediakan oleh *Project Owner* yaitu *template navbar, footer,* dan *loader*. Berikut pada gambar 3.14 di bawah ini merupakan hasil pembuatan halaman *our business* pada masa pengembangan proyek tahap satu selama satu minggu tepatnya pada minggu ke-12.



Gambar 3. 14 Hasil Pembuatan Halaman *Our Business*

3.2.4.3 Halaman *Services*

Tugas berikutnya yang diberikan oleh *Project Manager* adalah untuk mengerjakan halaman *services* untuk *web* utama Sagara Group Web. Pengerjaan halaman *service* berlangsung selama hampir satu minggu, yaitu pada minggu ke 13. Dalam pengerjaannya, halaman *service* tidak terlalu rumit karena tidak

menggunakan API untuk mengambil data. Halaman *service* ini juga terbilang sederhana karena hanya berupa *web* statis yang artinya isi dari *web* tersebut tidak mudah untuk diubah karena *script* dalam pembuatan *web* statis tidak mendukung untuk melakukan perubahan [4].

Halaman *services* dibangun dengan tujuan untuk dijadikan pusat informasi mengenai layanan apa saja yang disediakan oleh PT Sagara Asia Teknologi. Selain itu, dalam pengembangan halaman *service* juga berisi bagaimana cara kerja dan proses yang harus dilalui ketika menggunakan jasa yang ditawarkan. Dengan adanya informasi mengenai layanan yang ada, PT Sagara Asia Teknologi juga memberikan informasi ini agar para *investor* dan calon *investor* mengetahui pasti layanan apa saja yang ditawarkan sebagai nilai jual oleh perusahaan. Hal ini juga tentunya akan memberikan dampak baik kepada PT Sagara Asia Teknologi yaitu dapat meningkatkan keyakinan dan daya minat para *investor*.

Isi dari halaman ini merupakan konten yang di dapatkan melalui *file markup* yang telah dipelajari sebelumnya pada masa pemahaman proyek. Sama halnya dengan tampilan pada halaman *our business* yang dibuat berdasarkan *file markup*. Pengerjaan halaman *service* ini juga menggunakan bahasa pemrograman JavaScript dan *framework* yang digunakan adalah React JS sesuai dengan standar pengembangan proyek Sagara Group Web. Kemudian, pada halaman *services* ini juga menggunakan *open-source CSS framework* yaitu Tailwind CSS. Berikut pada gambar 3.15 di bawah ini merupakan hasil pembuatan halaman *services*.

Gambar 3.15 Hasil Pembuatan Halaman *Services*

3.2.4.4 *Consume API*

Tidak hanya membangun tampilan antarmuka *website*, tugas *Front-End Engineer* juga menghubungkan data yang sudah disiapkan oleh *Back-End Engineer* dengan halaman *website*. Kemudian, data yang sudah dihubungkan tersebut akan di tampilkan sebisa mungkin sesuai dengan yang diharapkan pada *file markup*. Hal tersebut dapat dilakukan dengan melakukan *consume API* hasil *Back-End Engineer* kedalam kode *front-end website* melalui React App API URL pada *file environment.json* dengan *code* seperti pada gambar 3.16 di bawah ini.

```
{
  "REACT_APP_API_URL": "https://backend-sgw.herokuapp.com/api/v1/"
}
```

Gambar 3. 16 Struktur Kode React App API URL

Data pada API yang sudah di hubungkan melalui *environment* tersebut kemudian akan dihubungkan melalui *folder pages* dengan *file service.js*. Selanjutnya, agar data yang berada di dalam API dapat digunakan diperlukan fungsi *fetch* untuk menarik data sesuai dengan kebutuhannya. Berikut pada gambar 3.17 merupakan struktur kode fungsi *fetch* yang digunakan pada halaman *our business*.

```
const fetchOurBusiness = async () => {
  setLoading(true);
  try {
    const response = await GlobalGet({
      url: `${REACT_APP_API_URL}business`,
    });
    if (response?.status === 200) {
      setLoading(false);
      setDataBusiness(response?.data?.post);
    } else {
      setLoading(false);
      alertMessage("error", response?.message, "Retry", () => {});
      setDataBusiness([]);
    }
  } catch (err) {
    setLoading(false);
    alertMessage("error", "Network Error", "Retry", () => {});
    setDataBusiness([]);
  }
}
```

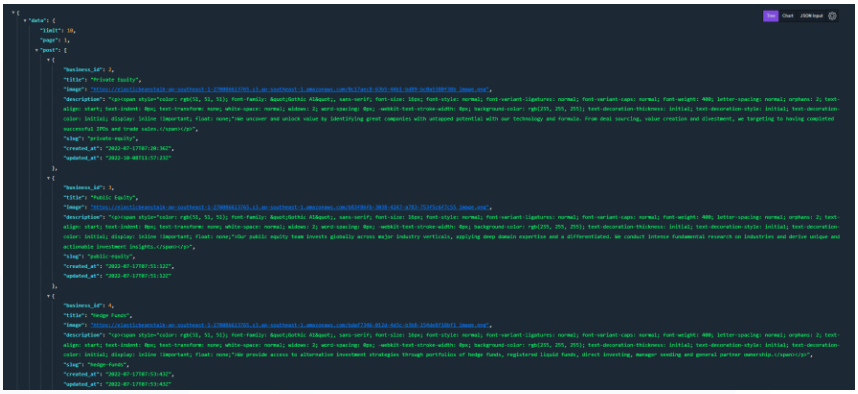
Gambar 3. 17 Struktur Kode Fungsi *Fetch Our Business*

Agar data pada API dapat dipanggil, diperlukan sebuah *function* yang disebut *map*. Dengan menggunakan *function* tersebut, data yang begitu banyak dalam API dapat dipilah sesuai dengan item dan index yang ingin dipanggil. Seperti pada kode *mapping* halaman *our business* pada gambar 3.18 di bawah, *item* yang dipanggil untuk tampil pada halaman *web* adalah *image* dan *title*.

```
{dataBusiness?.map((item, index) => (
  <a href="#" className="max-w-full inline-block">
  <div key={index} className="col-span-1 h-[250px] bg-dark">
    <Banner
      imgUrl={item?.image}
      classes={'bg-no-repeat bg-cover bg-center h-full w-full'}
    >
      <div className="flex items-end w-full h-full text-white">
        <div className="flex flex-col bg-dark/50 w-full p-2">
          <span>{item?.title}</span>
        </div>
      </div>
    </Banner>
  </div>
  </a>
))}
```

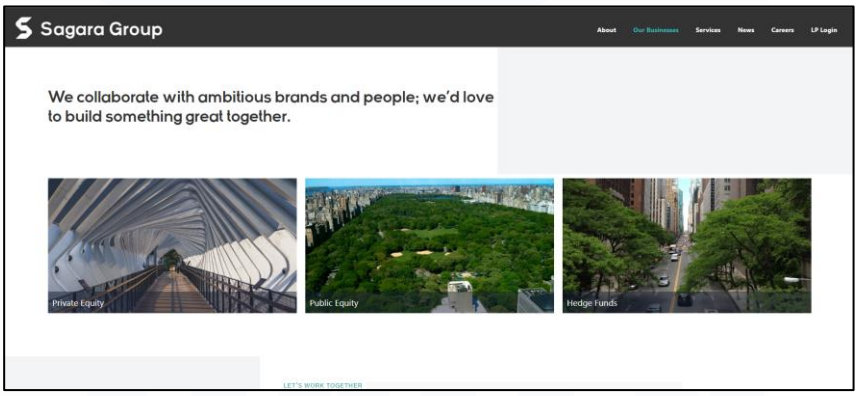
Gambar 3. 18 Struktur Kode *Function Map Consume API Business*

Berikut pada gambar 3.19 ini merupakan struktur dan data-data yang berada di dalam API yang digunakan pada halaman *our business*. Struktur pada data tersebut terdiri dari *business_id*, *title*, *image*, *description*, *slug*, *created_at*, dan *updated_at*. Dalam mengakses data pada API ini, *end point* yang digunakan adalah *business*, sehingga link API-nya adalah https://backend-sgw.herokuapp.com/api/v1/business.



Gambar 3. 19 Struktur dan Isi Data pada API *Business*

Setelah API terhubung, data yang berada di dalam API dengan *end point business* akan tampil tiga gambar. Dimana ketiga gambar tersebut berasal dari tiga data buatan atau data *dummy* yang berada di dalam API yaitu *private equity*, *public equity*, dan *hedge funds*. Gambar 3.20 di bawah ini merupakan hasil dari *consume* API pada halaman *our business*.



Gambar 3. 20 Hasil *Consume API* pada Halaman *Our Business*

3.2.4.5 Finalisasi

Finalisasi pada pengembangan *web* utama merupakan salah satu tugas yang diberikan oleh *Project Manager* seteleh semua halaman pada *web* utama selesai dikerjakan oleh tim *developer*. Selain *landing page*, *web* utama memiliki enam halaman yaitu halaman *about*, *our business*, *services*, *news*, *careers*, dan *LP login*. Semua halaman tersebut dikerjakan oleh tiga *Front-End Engineer*

yang berada di dalam tim Sagara Group Web. Dua *Front-End Engineer* lainnya yang berada dalam tim adalah Radhi dan Fahmi. Pembagian yang diberikan oleh *Project Manager* adalah halaman *about, news,* dan *careers* dikerjakan oleh Radhi dan halaman *LP login* dikerjakan oleh Fahmi. Sedangkan halaman yang tersisa halaman *our business* dan *services* dikerjakan oleh penulis.

Pada tahap ini, tugas pertama yang harus dikerjakan adalah menggabungkan seluruh halaman yang telah dibuat oleh *Front-End Engineer*. Penggabungan ini seharusnya hanya perlu melakukan *pull request* dari masing-masing *branch* yang dibuat oleh masing-masing *Front-End Engineer*. Namun, salah satu *Front-End Engineer* yang dipekerjakan terlebih dahulu yaitu Radhi mengerjakan tugasnya pada *repository* miliknya pribadi. Sehingga, pekerjaan yang harus dilakukan adalah menggabungkan halaman *web* yang dikerjakannya melalui *repository* pribadinya kedalam *repository* yang sudah telah disiapkan oleh *Project Owner*.

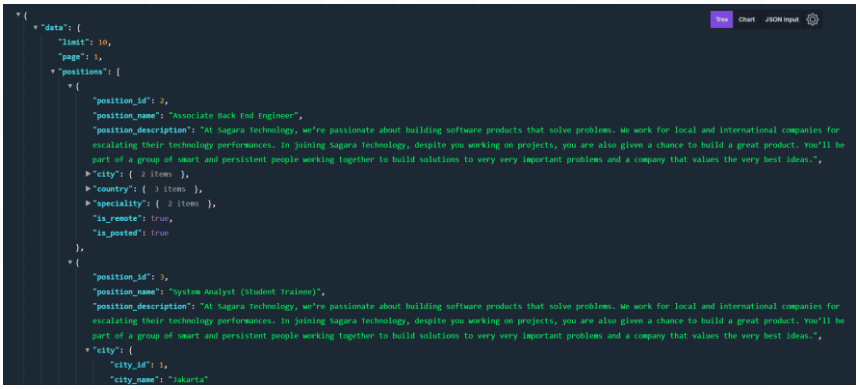
Dalam proses penggabungan, terdapat kendala yang dialami yaitu kendala dalam menggabungkan dua *repository* yang berbeda. Setelah melakukan penelusuran cara melalui internet, berdiskusi dengan teman tim, serta mentor, akhirnya ditentukan cara termudah untuk melakukannya adalah dengan memindahkannya secara manual. Sebelum memutuskan untuk memindahkan secara manual, penulis telah mencoba berbagai cara yang disarankan, akan tetapi terjadi *conflict* pada saat menggabungkan kedua *repository*. Hal tersebut terjadi karena begitu banyak data pada folder yang berbeda-beda dan banyak data yang bertabrakan dengan perubahan yang tidak seharusnya dirubah.

Penggabungan secara manual dilakukan dengan menyalin *file* yang perlu untuk dipindahkan saja. *File* yang dipindahkan adalah *index.js* pada *folder aboutPage, hiringPage,* dan *newsPage* dalam *folder organisms*. Selain itu, dalam *folder pages* juga terdapat

beberapa *file* yang harus dipindahkan yaitu *screen.js* dan *service.js* pada *folder* news, serta *file hiring.js* dan *about.js*. Semua *file* tersebut kemudian ditempankan ke dalam *folder*-nya masing-masing sesuai dengan penempatan yang telah ditetapkan.

Saat semua kode halaman *web* pada *web* utama telah digabungkan, selanjutnya adalah mengecek kembali seluruh halaman sudah saling terhubung, dapat diakses dan sesuai dengan yang diharapkan. Setelah melakukan pengecekan, ditemukan bahwa pada halaman *careers* dan halaman *about* yang dikerjakan oleh Radhi terdapat bagian-bagian yang memerlukan konsumsi API. Sehingga, tugas selanjutnya yang harus dikerjakan adalah melakukan *consume* API seperti yang telah dilakukan pada tahap *consume* API sebelumnya.

Pada halaman *careers*, terdapat sebuah bagian pada halaman yang digunakan untuk menampilkan lowongan pekerjaan apa saja yang tersedia di PT. Sagara Asia Teknologi. Data lowongan yang tersedia tersebut diambil melalui API Sagara Group Web dengan menggunakan *end point position*, sehingga link yang digunakan menjadi https://backend-sgw.herokuapp.com/api/v1/position. Gambar 3.21 di bawah ini merupakan tampilan struktur data yang berada di dalam API dengan *end point position*.



Gambar 3. 21 Struktur dan Isi Data pada API *Position*

Data yang ditampilkan pada halaman *careers* adalah data *position* yang tersedia, *speciality* dari posisi tersebut, dan *location* sesuai dengan kantor cabang mana yang membuka lowongan tersebut. Dengan demikian, maka API tersebut harus dapat dipanggil dengan menggunakan *GlobalGet* sebagai *fetcher* seperti pada gambar 3.22 di bawah ini.

```
const getDataPosition = async () => {
  try {
    const respons = await GlobalGet ({url:`${REACT_APP_API_URL}position`});
    setPosition(respons?.data?.positions);
  } catch (error) {
    console.log(error);
  }
}

const getDataSpeciality = async () => {
  try {
    const respons = await GlobalGet ({url:`${REACT_APP_API_URL}speciality`});
    setSpeciality(respons?.data?.specialities);
  } catch (error) {
    console.log(error);
  }
}

const getDataLocation = async () => {
  try {
    const respons = await GlobalGet ({url:`${REACT_APP_API_URL}location/city`});
    setLocation(respons?.data?.cities);
  } catch (error) {
    console.log(error);
  }
}
```

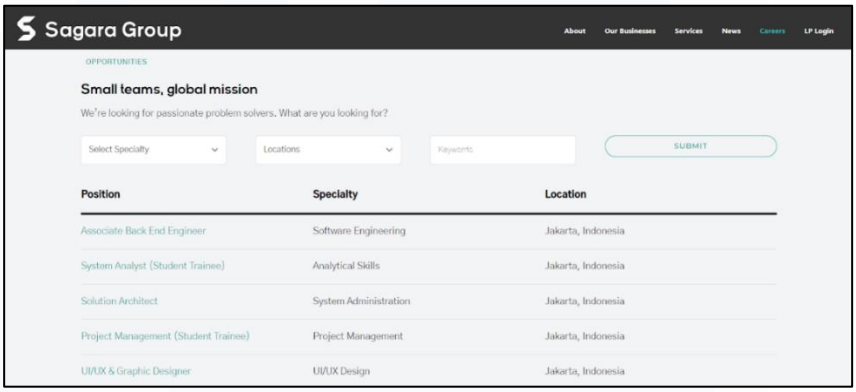
Gambar 3. 22 Struktur Kode Fungsi *Fetch Position*

Pada saat API yang diinginkan telah terhubung, selanjutnya *mapping* dapat dilakukan dalam *folder organisms* halaman *careers* agar data *position* dapat ditampilkan pada halaman *careers* seperti pada gambar 3.23 berikut ini.

```
{position?.map((item) => (
  <>
    <a
      href="#"
      className="py-4 font1 font-light text-blueLink inline-block max-w-full border-solid border-b-2 "
    >
      <div>{item.position_name}</div>
    </a>
    <div className="py-4 font1 font-light inline-block max-w-full border-solid border-b-2">
      {item.speciality.speciality_name}
    </div>
    <div className="py-4 font1 font-light inline-block max-w-full border-solid border-b-2">
      {item.city.city_name}, {item.country.country_name}
    </div>
  </>
))}
```

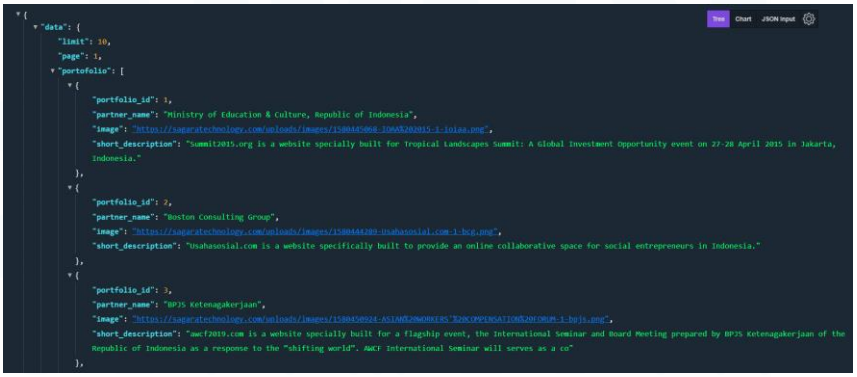
Gambar 3. 23 Struktur Kode *Function Map Consume API Position*

Berikut pada gambar 3.24 di bawah ini merupakan hasil dari *consume* API pada halaman *careers* dengan menggunakan *end point position*. Setelah data berhasil ditampilkan, data yang tampil adalah data *dummy* yang telah disiapkan oleh *Back-End Engineer*.



Gambar 3. 24 Hasil *Consume API* pada Halaman *Careers*

Pada bagian akhir halaman *about*, terdapat sebuah bagian yang digunakan untuk menampilkan *portfolio* yang telah dikumpulkan oleh PT. Sagara Asia Teknologi. Data *portfolio* tersebut diambil melalui API Sagara Group Web dengan menggunakan *end point portfolio*, sehingga link yang digunakan menjadi https://backend-sgw.herokuapp.com/api/v1/portfolio. Gambar 3.25 di bawah ini merupakan tampilan struktur data yang berada di dalam API dengan *end point portfolio*.



Gambar 3. 25 Struktur dan Isi Data pada API *Portfolio*

Data yang ditampilkan pada bagian *portfolio* memerlukan sebuah fungsi *fetcher* yaitu *GlobalGet* sehingga API dapat dipanggil. Berikut gambar 3.26 di bawah ini merupakan struktur kode yang digunakan untuk menghubungkan API *portfolio* ke dalam halaman *about*.

```
const getDataPortfolio = async () => {
  try {
    const respons = await GlobalGet ({url:`${REACT_APP_API_URL}portfolio`});
    setPortfolio(respons?.data?.portofolio);
  } catch (error) {
    console.log(error);
  }
}
```

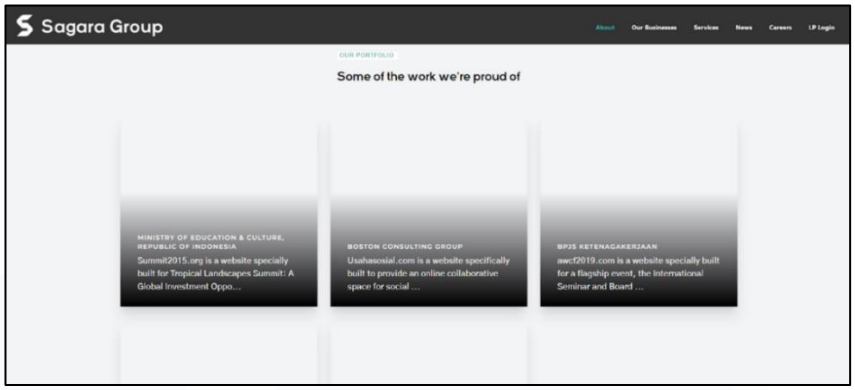
Gambar 3. 26 Struktur Kode Fungsi *Fetch Portfolio*

Pada saat API yang diinginkan telah terhubung, selanjutnya *mapping* dapat dilakukan dalam *folder organisms* halaman *about* agar data *portfolio* dapat ditampilkan pada halaman *about.* Berikut pada gambar 3.27 di bawah ini merupakan struktur kode yang digunakan untuk melakukan *mapping* data API dengan *end point portfolio*.

```
{portfolio?.map((data) => (
  <a href="#" className="card-zoom   ">
    <div className="c-grid1__content-2 z-[5]">
      <div className="mb-2 font3 text-xs font-semibold tracking-widest text-[#fff] uppercase">
        {data.partner_name}
      </div>
      <p className="font1 text-[#fff]">
        {data.short_description.substring(0,100)}...
        <br />
      </p>
    </div>
    <div className="card-zoom-image" style={{backgroundImage : 'url("${data.image}")'}} />
  </a>
))}
```

Gambar 3. 27 Struktur Kode *Function Map Consume API Portfolio*

Berikut gambar 3.28 di bawah merupakan hasil dari *consume* API pada halaman *about* dengan menggunakan *end point portfolio*. Setelah data berhasil ditampilkan, data yang tampil adalah data *dummy* yang telah disiapkan oleh *Back-End Engineer*.



Gambar 3. 28 Hasil *Consume API* pada Halaman *About*

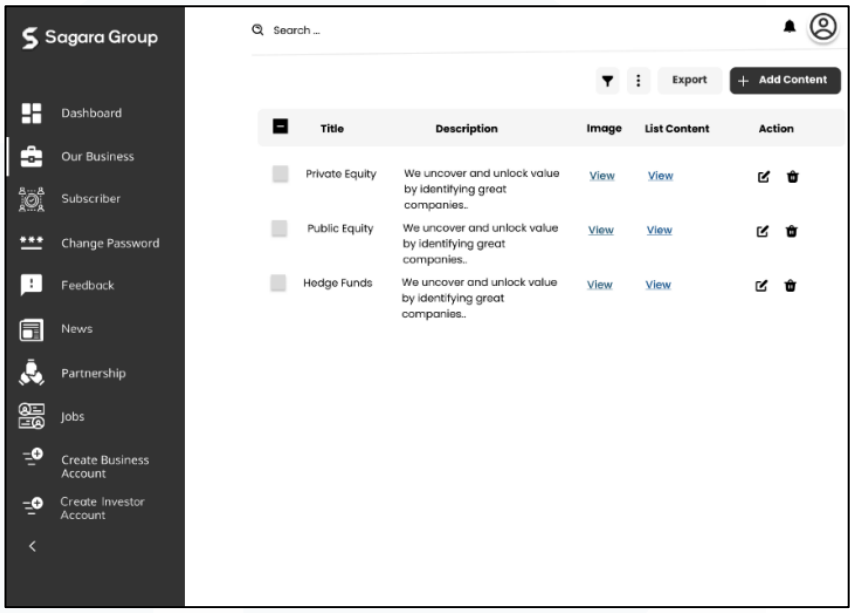
3.2.5 Pengembangan Proyek Tahap 2 (*Web* Admin):

3.2.5.1 Persiapan Pengembangan

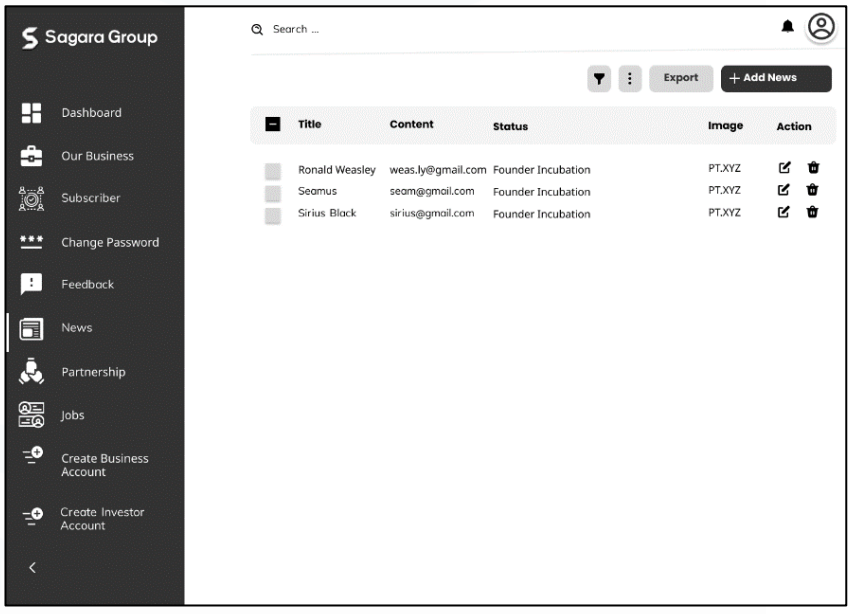
Pada masa pengembangan proyek tahap dua yaitu tahap pengembangan *web* admin, hal pertama yang dilakukan adalah melakukan persiapan sebelum memulai pengembangan. Persiapan yang dilakukan adalah mempelajari desain tampilan antarmuka yang telah dibuat oleh UI/UX *Designer*. Berdasarkan desain tampilan antarmuka, *web* admin memiliki 10 *menu* yaitu *dashboard*, *our business*, *subscriber*, *change password*, *feedback*, *news*, *partnership*, *jobs*, *create business account*, dan *create investor account*. Namun, berdasarkan informasi yang diberikan oleh *Project Manager*, *menu partnership* tidak perlu dikerjakan karena tidak disetujui oleh *Project Owner*.

Setelah mempelajari desain tampilan antarmuka, *Project Manager* akhirnya melakukan pembagian tugas yang dimana penulis mendapat bagian untuk mengerjakan *menu our business* dan *news*. Kedua *menu* ini memiliki tampilan dan fitur yang cukup serupa. Dimana halaman utama *menu* digunakan untuk menampilkan *list* data yang ada dan memiliki fitur *add*, *edit*, dan *delete*. Berikut pada gambar 3.29 dan gambar 3.30 di bawah ini

merupakan hasil desain tampilan antarmuka dari halaman utama *menu our business* dan *news*.



Gambar 3. 29 Desain Tampilan Antarmuka Halaman Utama *Menu Our Business*



Gambar 3. 30 Desain Tampilan Antarmuka Halaman Utama *Menu News*

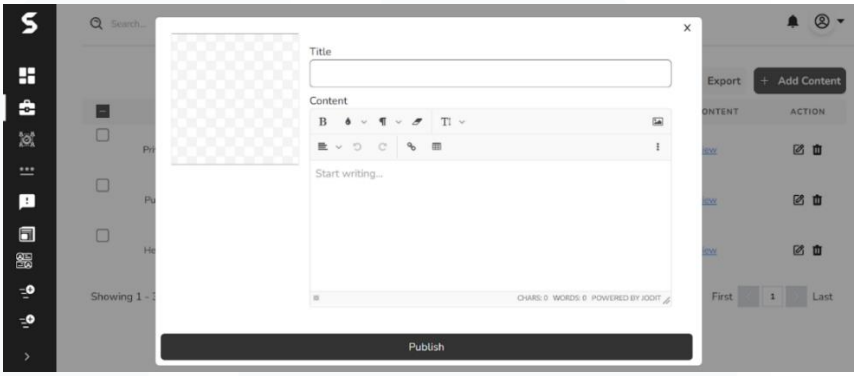
3.2.5.2 Halaman *Our Business*

Pada *menu our business*, terdapat satu halaman utama dan dua halaman *form* yang harus dikerjakan yaitu *add content* dan *edit content*. Halaman utama berisi *sidebar* dan tabel yang menampilkan daftar unit-unit bisnis yang berada di bawah PT Sagara Asia Teknologi. Dalam pengerjaannya, semua halaman pada *web* admin akan menampilkan *sidebar* yang telah dibuat oleh *Front-End Engineer* lain sehingga hanya perlu dipanggil ketika ingin digunakan pada halaman lainnya. Kemudian, tabel yang digunakan untuk menampilkan daftar unit bisnis juga dibuat menggunakan *file* yang berbeda. Setelah semua bagian dibuat, selanjutnya adalah semuanya pada *folder pages* sehingga hasil yang didapatkan menyerupai dengan desain tampilan antar muka. Gambar 3.31 berikut ini merupakan hasil dari pembuatan halaman *our business* pada *web* admin.

Gambar 3. 31 Hasil Pembuatan Halaman *Our Business* pada *Web* Admin

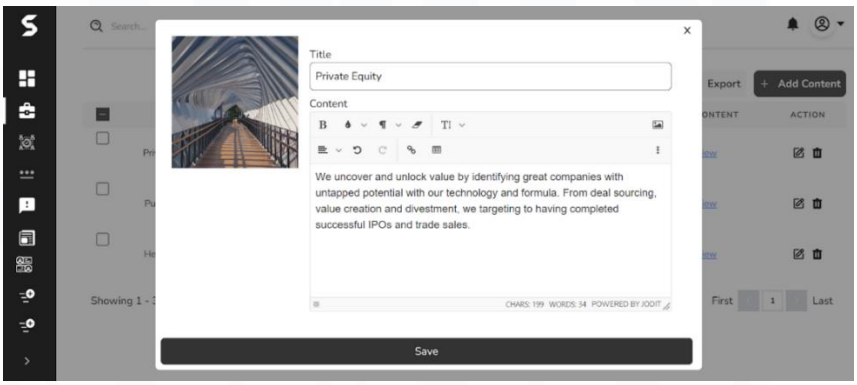
Fitur *add content* pada halaman *our business* akan muncul ketika pengguna menekan tombol *add content* pada ujung kanan atas halaman *web* yang dibuat pada *file* yang terpisah. Fitur ini akan tampil menimpa halaman utama sehingga fitur ini juga digabungkan pada *folder pages* milik *menu our business*. Sehingga, ketika tombol *add content* ditekan maka *show form add content*

akan ditetapkan sebagai *true* dengan menggunakan *handler* dan *false* ketika ditutup. Gambar 3.32 di bawah ini merupakan hasil pembuatan fitur *add content* pada halaman *our business*.



Gambar 3. 32 Hasil Pembuatan Fitur *Add Content*

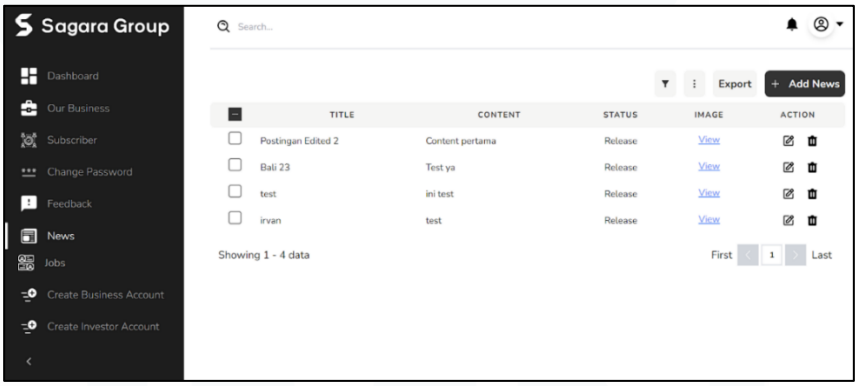
Fitur *edit content* ini dapat diakses ketika pengguna menekan tombol *edit* yang berada di kolom paling kanan pada tabel yaitu kolom *action*. Sama seperti fitur *add content*, fitur *edit content* juga dibuat pada *file* yang terpisah dan gabungkan pada *folder pages* halaman utama *menu our business*. Selain itu, fitur ini juga menggunakan *handler* sehingga ketika tombol *edit* ditekan maka *show form edit content* akan ditetapkan sebagai *true* dan ketika tombol silang ditekan akan ditetapkan sebagai *false* untuk menutup *form*. Gambar 3.33 di bawah ini merupakan hasil pembuatan fitur *edit content* pada halaman *menu our business*.



Gambar 3. 33 Hasil Pembuatan Fitur *Edit Content*

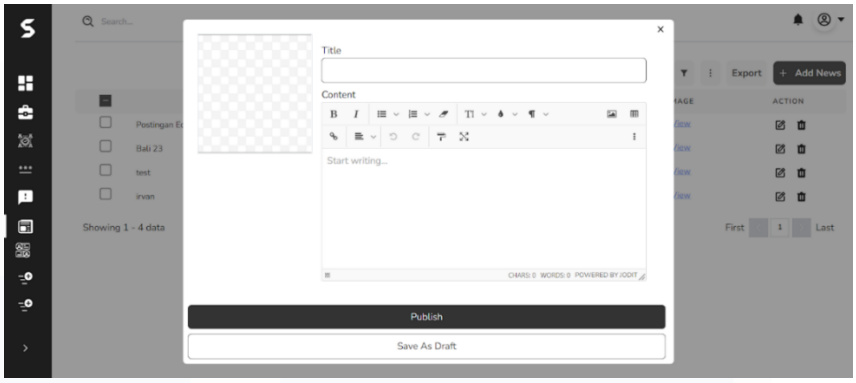
3.2.5.3 Halaman *News*

Pada *menu news* juga terdapat satu halaman utama dan dua halaman *form* yang harus dikerjakan yaitu *add content* dan *edit content*. Halaman utama berisi *sidebar* dan tabel yang menampilkan daftar berita-berita yang berada ingin ditampilkan pada *website*. Sama seperti pembuatan halaman *our business*, tabel yang digunakan untuk menampilkan daftar berita juga dibuat menggunakan *file* yang berbeda. Kemudian, pada *folder pages* digunakan untuk menggabungkan *sidebar* dan juga tabel yang telah dibuat sehingga hasil yang didapatkan menyerupai dengan desain tampilan antar muka. Gambar 3.34 merupakan hasil dari pembuatan halaman *news* pada *web* admin.



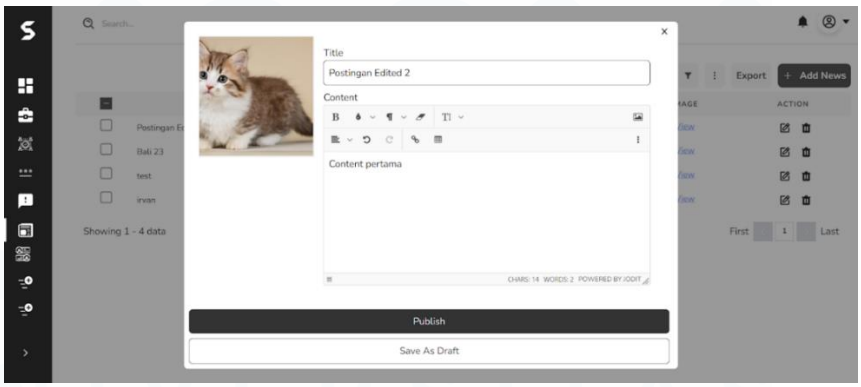
Gambar 3. 34 Hasil Pembuatan Halaman *News* pada *Web* Admin

Fitur *add news* pada halaman *news* akan muncul ketika pengguna menekan tombol *add news* pada ujung kanan atas halaman *web*. Sama seperti pada halaman *our business*, fitur ini akan ditampilkan dengan menimpa halaman utama maka nantinya akan digabungkan pada *folder pages* milik *menu news*. Sehingga, ketika tombol *add news* ditekan maka *show form add news* akan di tetapkan sebagai *true* dengan menggunakan *handler* dan *false* ketika ditutup. Gambar 3.35 di bawah ini merupakan hasil pembuatan fitur *add news* pada halaman *news*.



Gambar 3. 35 Hasil Pembuatan Fitur *Add News*

Fitur *edit news* ini dapat diakses ketika pengguna menekan tombol *edit* yang berada di kolom paling kanan pada tabel yaitu kolom *action*. Sama seperti fitur *add news*, fitur *edit news* juga dibuat pada *file* yang terpisah dan gabungkan pada *folder pages* halaman utama *menu news*. Selain itu, fitur ini juga menggunakan *handler* sehingga ketika tombol *edit* ditekan maka *show form edit news* akan ditetapkan sebagai *true* dan ketika tombol silang ditekan akan ditetapkan sebagai *false* untuk menutup *form*. Gambar 3.36 di bawah ini merupakan hasil pembuatan fitur *edit news* pada halaman *menu news*.



Gambar 3. 36 Hasil Pembuatan Fitur *Edit News*

3.2.5.4 *Consume API*

Data-data yang digunakan pada Sagara Group Web dapat dimanipulasi melalui *web* admin ini karena data tersebut terhubung melalui API yang telah dihubungkan melalui *file environment*. Pada *web* admin ini, terdapat beberapa fitur yang perlu dihubungkan dengan API agar data tersebut dapat dimanipulasi yaitu tambah data, ubah data, dan hapus data. Namun, sebelum dapat melakukan itu semua hal yang perlu dilakukan terlebih dahulu adalah melakukan *fetch* data sama seperti pada saat menampilkan data di *web* utama.

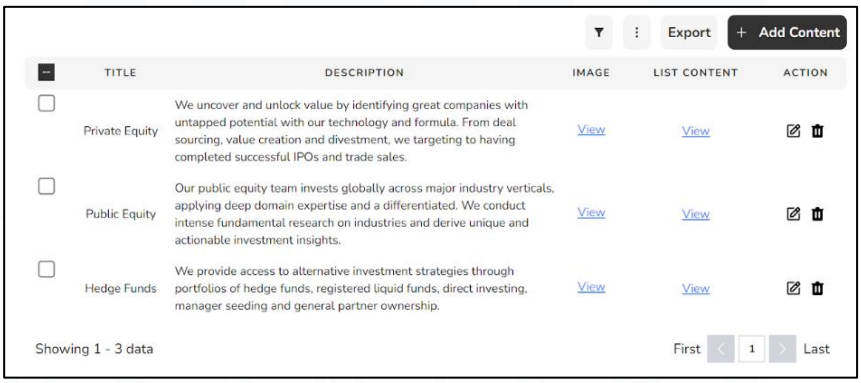
Dalam melakukan tambah data, fungsi yang digunakan adalah fungsi *GlobalPost* yang dimana dapat digunakan untuk memasukan data kedalam tautan API yang digunakan. Sedangkan dalam melakukan ubah data, fungsi yang digunakan adalah fungsi *GlobalPut* yang digunakan untuk melakukan *update* pada data yang berada di dalam API. Lalu untuk melakukan hapus data, fungsi yang digunakan adalah *GlobalDelete* yang dapat menghapus data pada API yang digunakan. Untuk halaman *our business*, *end point* API yang digunakan adalah *business*. Sedangkan, *end point* untuk halaman *news* adalah *post*.

Gambar 3.37 dan gambar 3.38 merupakan isi dari data yang berada di dalam API yang digunakan pada masa pengembangan *web* admin yaitu API dengan *end point business* dan *post*.

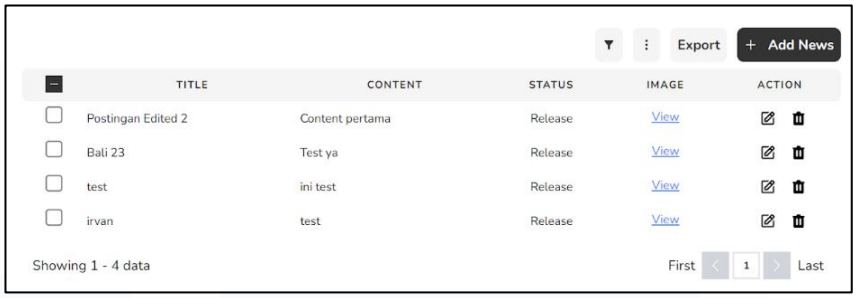
Gambar 3. 37 Struktur dan Isi Data API *Business*

Gambar 3. 38 Struktur dan Isi Data API *Post*

Setelah API terhubung dengan semua fungsi, data yang berada di dalam API dengan *end point business* dan *post* dapat digunakan sesuai dengan fungsinya masing-masing. Gambar 3.39 dan gambar 3.40 di bawah ini merupakan hasil dari *consume* API pada halaman utama *our business* dan *news*.



Gambar 3. 39 Hasil *Consume API* pada Halaman *Our Business Web* Admin



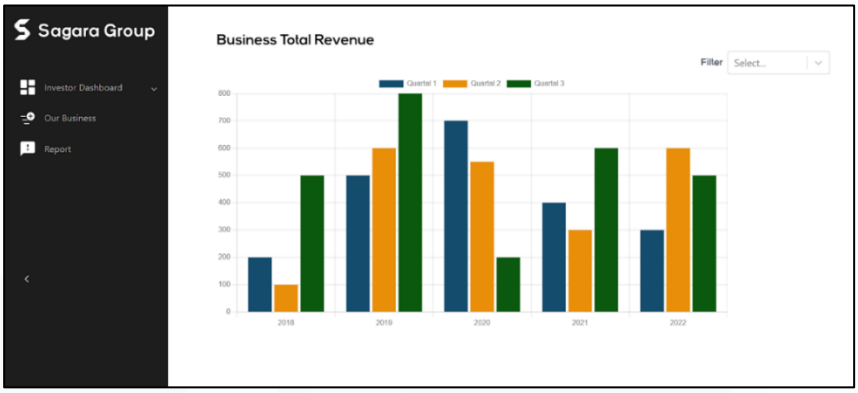
Gambar 3. 40 Hasil *Consume API* pada Halaman *News Web* Admin

3.2.6 Pengembangan Proyek Tahap 3 (*Web Investor*):

3.2.6.1 Halaman *Busienss Total Revenue*

Web investor merupakan bagian dari Sagara Group Web yang hanya dapat diakses oleh para *investor*. Pada tahap terakhir ini, halaman pertama yang ditugaskan oleh *Project Manager* adalah untuk membuat tampilan dari halaman *business total revenue* pada *menu investor*. Desain tampilan antar muka yang diberikan cukup menyerupai dengan tampilan dari *web* admin, dimana keduanya sama-sama menggunakan *sidebar* untuk menampilkan *menu* pada halaman *web*.

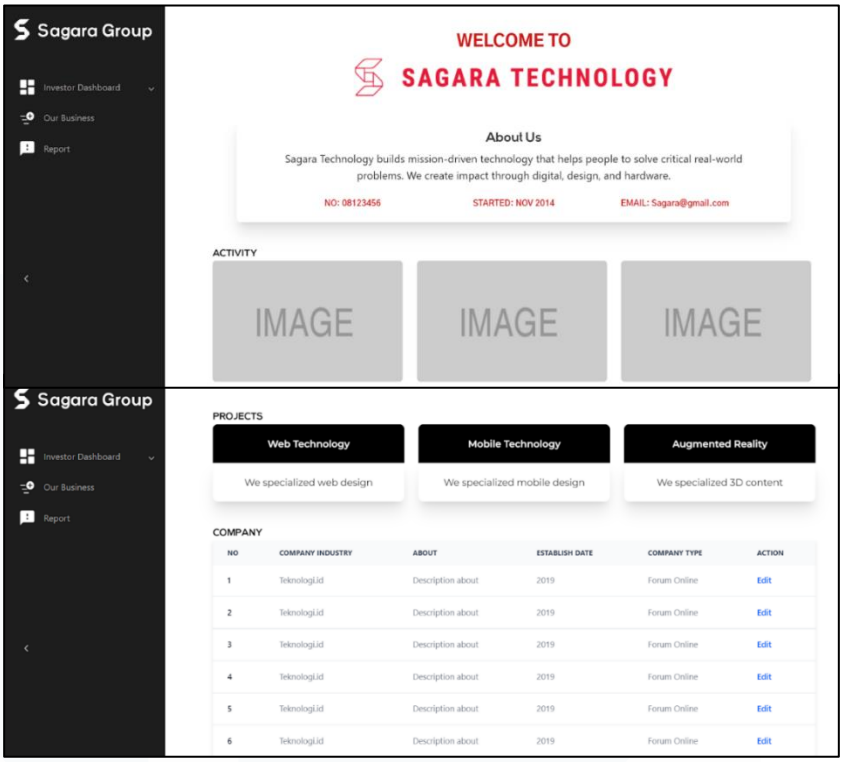
Dalam pengembangannya, *sidebar* yang digunakan merupakan kode yang telah dibuat oleh *Front-End Engineer* lain sehingga ketika akan digunakan hanya perlu dipanggil kedalam halaman *web* tersebut. Kemudian, pembuatan *bar chart* yang digunakan untuk menampilkan data keuntungan unit bisnis dibuat dengan memakai *open-source library* JavaScript yaitu ChartJS. Dikarenakan API yang digunakan belum siap, maka dalam masa pengembangan ini hanya menggunakan data *dummy*. Gambar 3.41 di bawah ini merupakan hasil pembuatan fitur *business total revenue* pada *web investor*.



Gambar 3. 41 Hasil Pembuatan Fitur *Business Total Revenue* pada *Web Investor*

3.2.6.2 Halaman *Our Business*

Tugas berikutnya pada masa pengembangan proyek *web investor* yang diberikan oleh *Project Manager* adalah pembuatan halaman *our business*. Halaman ini digunakan untuk menampilkan beberapa informasi seperti *about us*, *activity*, *projects*, dan *list company* yang berada di bawah naungan PT Sagara Asia Teknologi. Sama seperti pada halaman *business total revenue*, halaman *our business* juga masih mengunakan data *dummy* untuk menampilkan perkiraan tampilan yang akan terlihat ketika data sudah dimasukan nantinya. Gambar 3.42 di bawah ini merupakan hasil pembuatan halaman *our business* pada *web investor*.



Gambar 3. 42 Hasil Pembuatan Halaman *Our Business* pada *Web Investor*

3.3 Kendala yang Ditemukan

Selama masa pelaksanaan program kerja magang sebagai *Student Trainee* pada divisi *Front-End Engineer* di PT Sagara Asia Teknologi, adapun beberapa kendala atau kesulitan yang ditemukan antara lain:

1) Adanya keterbatasan dalam berkomunikasi dengan tim Sagara Group Web, *Project Owner*, dan *Supervisor* karena selama program kerja magang dilakukan di rumah atau *work from home*.
2) Terdapat beberapa *tools, platform* dan *framework* yang belum pernah dipelajari semasa kuliah yaitu ReactJS, API, GitHub, dan Tailwind CSS. Semua yang belum pernah dipelajari tersebut akhirnya harus dipelajari secara mandiri selama masa pembekalan yang cukup singkat. Hal ini mengakibatkan proses pengerjaan proyek menjadi terhambat karena belum terbiasa menggunakannya.

3.4 Solusi atas Kendala yang Ditemukan

Berdasarkan kendala-kendala yang ditemukan selama program kerja magang sebagai *Student Trainee* pada divisi *Front-End Engineer* di PT Sagara Asia Teknologi, adapun solusi yang dilakukan guna mengatasi kendala yang telah ditemukan antara lain:

1) Melakukan komunikasi melalui *platform* WhatsApp, Discord, dan juga rutin menghadiri *weekly meeting* yang dilakukan melalui *platform* Google Meet atau Zoom dengan seluruh pihak yang terlibat selama program kerja magang berlangsung.
2) Membagi waktu selama masa pembekalan untuk dapat mempelajari semua *tools, platform* dan *framework* yang belum pernah dipelajari, dimana setiap minggu difokuskan untuk mempelajari satu *tools* atau *platform* atau *framework*. Kemudian, ketika menemukan kendala selama masa pengerjaan proyek dan tidak dapat menyelesaikannya sendiri, maka hal yang dilakukan adalah berkomunikasi dan berkonsultasi kepada teman tim, *Project Manager*, serta *Supervisi* untuk menyelesaikannya.