BAB I
PENDAHULUAN

1.1 Latar Belakang

Ditengah menghadapi Revolusi Industri 4.0 yang serba digital. Seluruh aspek kehidupan manusia telah dibantu atau digantikan oleh hadirnya teknologi. Salah satu yang berperan penting dalam kehidupan yang serba digital ini adalah para pengembang aplikasi. Sumber daya manusia ini sangat dibutuhkan untuk mengakselerasi Indonesia menuju dunia digital.

Dasar dari pemikiran pengambilan pengembangan dan implementasi rest api web developer untuk toko belanja menggunakkan NODE JS adalah judul yang diambil sebagai tugas akhir kegiatan studi independent dan yang sudah di kerjakan. Alasan untuk memutuskan judul tersebut menjadi judul tugas akhir adalah yang dimana hal tersebut bersinggungan dengan bidang ilmu yang diambil.

Tujuan dari pengembangan dan implementasi REST API web developer aplikasi toko belanja ini bertujuan sebagai untuk meningkatkan efisiensi dan efektivitas, dikarenakan NODE JS memiliki sendiri memiliki kelebihan *fast execution*, mendukung proses *single-thread*, dan *non-blocking*.

Manfaat yang didapatkan dari pengembangan produk ini dalam segi Pendidikan dapat dijadikan bahan referensi untuk melakukan pengembangan *Rest* API dan juga dapat digunakkan sebagai bahan pertimbangan untuk pembuatan produk *Rest* API lainnya. Manfaat lain yang dapat dirasakan yaitu pengembangan ini dapat meningkatkan berbagai website Indonesia baik dari efektifitas dan efisiensi pengolahan data.

1.2 Profil

1.2.1 Struktur Organisasi (Studi Independent)

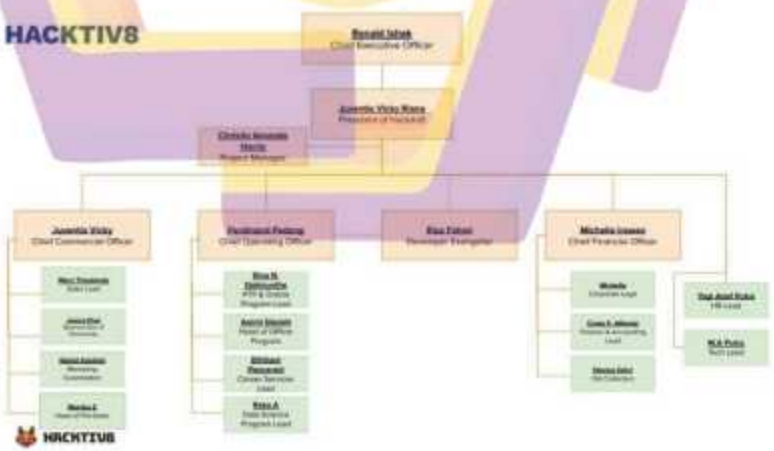
Hacktiv8 adalah lembaga pendidikan teknologi yang berdiri pada tahun 2016. Hacktiv8 didirikan oleh Roland Ishak dan Riza Fahmi, yang didasari oleh

keresahan mereka melihat banyaknya kejadian saling bajak talent-talent pengembang (developer) di antara perusahaan-perusahaan teknologi.

Hacktiv8 secara resmi diluncurkan pada tahun 2016 untuk menjembatani developer Indonesia dengan kebutuhan dan permintaan pasar yang semakin kompetitif. Hacktiv8 hadir sebagai platform pendidikan teknologi yang membantu menghasilkan talenta digital berstandar global. Semua demi mengakselerasi Indonesia agar menjadi yang terdepan.

Saat ini, lebih dari 1450+ lulusan developer di Hacktiv8. Terdapat banyak pembelajar yang telah dan sedang terdaftar dalam kelas yang disediakan oleh Hacktiv8. Terdaftar ada sekitar 51-200 karyawan professional yang ada dinaungan hacktiv8.

Adapun struktur organisasi merupakan sebuah garis penugasan formal yang menunjukkan alur tugas dan tanggung jawab setiap anggota perusahaan, perusahaan serta hubungan antar pihak dalam organisasi yang bekerja sama untuk mencapai suatu tujuan organisasi. Struktur organisasi dari PT. Hacktivate Teknologi Indonesia.



1.2.2 lokasi Kegiatan

Kegiatan studi independent batch-3 mitra Hacktiv8 berlokasi di Jl. Arteri Pd. Indah No.7, RT.5/RW.9, Kby. Lama Sel., Kec. Kby. Lama, Kota Jakarta Selatan, Daerah Khusus Ibukota Jakarta 12240 dan dilaksanakan secara online / daring dirumah masing-masing.

1.2.3 Lingkup kegiatan

Dalam program MSIB Batch ke-3 Kampus Merdeka, Hacktiv8 menjadi salah satu mitra yang menyediakan berbagai kegiatan dalam program tersebut. Salah satu kegiatannya adalah Studi Independent Pengembang Node JS for Back-End Web Developer. Dalam kegiatan ini Hacktiv8 menyajikan berbagai kelas pada peserta yang mengikutinya. Tidak hanya kelas, sesi pertemuan secara online juga disajikan oleh Hacktiv8 untuk dapat berkomunikasi secara tatap muka dengan peserta. Adapun rician sistem pembelajaran yang diberikan oleh Hacktiv8 kepada pesertanya dalam kegiatan ini, antara lain sebagai berikut :

1. Belajar Mandiri

   Belajar mandiri merupakan metode belajar yang diberikan oleh Dicoding Indonesia kepada pesertanya yang dilakukan secara online. Dalam proses belajar mandiri, peserta disajikan berbagai kelas, dalam kelas yang disajikan terdapat berbagai modul mengenai materi pembelajaran pada kegiatan Node JS for Back-End Web Developer.

   Materi yang disajikan berupa penjelasan teori, dan praktikum. Penjelasan teori yang diberikan berupa penjelasan melalui bentuk teks dan juga bentuk video. Dalam setiap kelas terdapat ujian akhir, kuis. Hal ini diberikan kepada peserta dengan tujuan untuk dapat mengukur tingkat pemahaman, kemampuan serta kemajuan peserta sebelum mengikuti proses pembelajaran dan setelah mengikuti proses pembelajaran pada kelas yang sedang diakses.

   Bagi peserta yang sudah berhasil menyelesaikan seluruh modul,

kuis, ujian akhir dalam suatu kelas, peserta akan disajikan *project submission* yang harus diselesaikan agar dapat melanjutkan ke kelas selanjutnya. *Project submission* yang disajikan kepada peserta merupakan suatu implementasi dari materi dalam kelas yang sedang diakses. Sehingga untuk dapat mengerjakan *project submission* tersebut peserta harus paham betul mengenai materi yang sedang dipelajari dalam kelas yang sedang diakses.

2. Sesi Tatap Muka Bersama Expert (ILT tech)

   Sesi tatap muka bersama *Expert* merupakan sesi meeting yang dihadiri oleh seluruh peserta yang mengikuti kegiatan serupa (Pengembang Node JS for Back-End Web Developer). Sesi ini diadakan dalam 3 jam setiap dua kali dalam satu minggunya (delapan kali dalam satu bulan). Dalam sesi meeting ini Hacktiv8 akan menghadirkan Mentor *Expert* sesuai pada bidangnya. Mentor *Expert* merupakan seseorang narasumber yang akan membantu menjelaskan materi kepada peserta yang mengikuti kegiatan tersebut.

   Materi yang dibawakan merupakan materi yang sedang dipelajari oleh peserta pada kelas yang telah diberikan. Materi tersebut telah dijadwalkan oleh Tim Hacktiv8 dalam *timeline* kegiatan. Dalam sesi ini terdapat pembagian waktu dalam setiap beberapa menitnya. Pembagian waktu tersebut meliputi :

   - Sesi Penjelasan Materi

     Sesi penjelasan materi merupakan sesi yang membahas mengenai materi yang sudah dijadwalkan. Dalam penjelasan materi biasanya dilakukan dengan penjelasan secara teori dan penjelasan secara praktikum.

   - Sesi Q&A

     Sesi Q&A merupakan sesi tanya jawab antara peserta dengan narasumber. Bagi peserta yang memiliki topik yang ingin ditanyakan kepada narasumber, peserta dapat bertanya dalam sesi

Q&A ini.

- Sesi Kuis

  Setelah pembelajaran selesai dijelaskan, peserta diharapkan dapat mengikuti kuis mengenai materi yang telah dibahas dalam sesi ILT *Tech* yang sedang dihadiri.

3. Sesi ILT *soft Skill*

   Sesi ILT *Soft Skill* merupakan sesi yang diadakan oleh Hacktiv8 selama 3 jam sekali setiap satu bulannya. Dalam sesi ILT *Soft Skill* peserta akan dilatih oleh mentor tentang berbagai pelatihan mengenai kepribadian diri, bagaimana cara membangun pribadi agar menjadi lebih baik, hingga pelatihan persiapan dalam memasuki dunia kerja. Dalam sesi ini terdapat pembagian waktu dalam setiap beberapa menitnya. Pembagian waktu tersebut meliputi :

   - Sesi Penjelasan Materi

     Sesi penjelasan materi merupakan sesi yang membahas mengenai materi yang sudah dijadwalkan. Dalam penjelasan materi biasanya dilakukan dengan penjelasan secara teori.

   - Sesi Q&A

     Sesi Q&A merupakan sesi tanya jawab antara peserta dengan mentor. Bagi peserta yang memiliki topik yang ingin ditanyakan kepada mentor, peserta dapat bertanya dalam sesi Q&A ini.

4. Sesi Konsultasi Bersama Mentor

Sesi konsultasi bersama mentor merupakan sesi yang diadakan oleh Hacktiv8 selama 3 jam dalam setiap minggunya. Sesi konsultasi ini disajikan kepada peserta dengan dimentori oleh mentor dalam grup kelasnya masing-masing. Adapun tujuan dari sesi konsultasi ini antara lain

sebagai berikut :

- Memonitoring progress kegiatan belajar peserta dalam setiap minggunya,
- Menyebarkan informasi mengenai kegiatan yang sedang diikuti oleh peserta.
- Membantu peserta yang memiliki kendala terkait kegiatan yang sedang diikutinya.

Dengan diadakannya sesi konsultasi bersama mentor, peserta dapat mengetahui berbagai informasi mengenai kegiatan yang sedang diikutinya, serta peserta yang memiliki kendala dalam kegiatan yang sedang diikutinya dapat mendapatkan solusi dari mentor dalam memecahkan kendala yang sedang dihadapinya.

1.2.4 Deskripsi kegiatan

Aktivitas Studi Independen Pengembang Node JS for Back-End Web Developer meliputi pembelajaran individu dan *project* akhir dalam bentuk tim. Pada pembelajaran individu, setiap peserta akan mengikuti kelas dalam bentuk *asynchronous* (online melalui modul belajar di Hacktiv8) dimana peserta dapat berkonsultasi dengan *expert* terkait materi yang dipelajarinya melalui forum diskusi.

Peserta akan mendapatkan materi intensif dari instruktur *profesional* yang ahli di bidangnya selama 2 bulan (16 sesi, 3 jam setiap sesi). Metode pelatihan terdiri atas *live lecture* dan *self-paced learning*. Setelah belajar mengenai teori dasar, peserta akan diberi tugas untuk membuat *final project* dan diberi waktu selama 4 bulan untuk menyelesaikan *final project* tersebut secara berkelompok. Peserta akan diminta untuk membuat REST API dengan *framework* nodeJS yaitu expressJS dan database PostgreSQL. REST API yang dibangun harus memiliki authentifikasi yang juga dilengkapi dengan implementasi JWT (Json Web Token)

dan harus bisa Create-Read-Update-Delete dengan menerapkan query SQL storing, *Manipulation* & *retrieve Data* pada Postgres Database. Hasil akhir untuk *source code* diletakkan pada repository Github/Bitbucket dan harus di lakukan *deployment online* via Heroku/Netlify. Hasil tugas akhir akan diperiksa oleh instruktur/mentor yang mengajar di kelas. Selama pengerjaan final project, peserta akan mendapatkan sesi mentorship sehingga peserta bisa mendapatkan *feedback* dari mentor untuk menyempurnakan final projectnya.

Selain itu, setiap peserta akan memiliki pembimbing sebagai tempat konsultasi jika ditemui kesulitan non-akademik dalam mengikuti pembelajaran. Peserta akan memperoleh sertifikat kompetensi di setiap kelas di dalam *Learning Path* Node JS for Back-End Web Developer jika peserta berhasil lulus dari setiap ujian/penilaian yang diadakan untuk setiap kompetensi.

Pada project akhir, peserta akan dibagi menjadi kelompok, dimana satu kelompok terdiri atas-3 orang dengan tema yang ditentukan oleh masing-masing kelompok dan harus mendapatkan persetujuan dari pembimbing atau *expert*. *Project* akhir di Hacktiv8 terdapat 4 *final project*, *final project* ini adalah *rest* API dan juga testing. Peserta yang mengikuti program ini diharapkan mampu:

1. Memahami serta mengetahui fundamental dasar seorang back-end web developer
2. Memahami serta mengimplementasikan framework Nodejs terutama backend untuk keperluan web application
3. Mendesain dan membuat database untuk keperluan web application
4. Mengamankan dan mengatur user authentication dan access control untuk application backend
5. Men-debug dan mengoptimalisasi kinerja web application yang dibuat

1.2.5 Jadwal Kegiatan

Kegiatan studi independen di Hacktiv8 pada program MSIB *batch* ke-3 dilaksanakan pada tanggal, 11 agustus 2022 sampai dengan , 31 desember 2022 adapun rincian jadwal dari kegiatan tersebut adalah sebagai berikut :

Tabel 1. 1 Jadwal Kegiatan

Bulan	Pekan	Tanggal (2022)	Sesi konsultasi	Kelas	Target Milestone	Cut-off 2022
Agustus	3	23/Agustus	1	introduction NodeJS (Hello World) Modules	Dapat membuat hello word dengan menggunakan node js .Menguasai penggunaan modul Dapat membuat modul sendiri	1/ September
	3	25/Agustus	2	Event Handler NodeJS Buffer & Streams	memahami cara kerja event menguasai handling event dapat menangani error pada event	
	4	30/Agustus	3	File Systems HTTP Server	Mampu berinteraksi dalam sistem file dengan cara yang sama dengan fungsi standar POSIX Memahami perbedaan fungsi file sistem	

					synchronous dan asynchronous Dapat membuat server Dapat mengkonfigurasikan keamanan pada http header	
septem ber	4	1/ September	4	Pengenalan ExpressJs Routes	Dapat menginstall dan mengkonfigurasi express Memahami route methods, paths, parameters, handlers Menguasai response methoods Mengerti bagaimana titik akhir aplikasi (URIs) merespons permintaan klien.	4 / Oktober
	5	6/Septemb er	5	Static Files MiddleWare	Memahami route methods, paths, parameters, handlers Menguasai response methoods Mengerti bagaimana titik akhir aplikasi (URIs) merespons permintaan klien	

					dapat melakukan akses req, res Dapat melakukan perubahan pada permintaan dan objek respons Dapat mengakhiri siklus req-res	
	5	8/September	6	Templates & Template Engines Querystring & Post Parameters	Dapat memanipulasi template file statis menjadi variabel-variabel yang kemudian dapat dikirimkan ke klien untuk memudahkan design halaman HTML Dapat memparsing data dengan menggunakan querystring Dapat mengirimkan informasi ke server melalui url	
	6	13/September	7	RDBMS Pengenalan PostgreSQL	Memahami konsep database Memahami jenis-jenis database Dapat melakukan instalasi dan konfigurasi	

					Mengetahui fitur yang terdapat di PostgreSQL/MongoDB	
	6	15/September	8	Persiapan Database Query Melalui Terminal	Mengetahui cara install postgre Dapat membuat query database dengan menggunakan terminal / CLI (Create, Delete, View, Update)	
	7	20/September	9	PgAdmin Tipe Data Integer Tipe Data Character	Dapat melakukan interaksi postgres menggunakan GUI Dapat mengetahui perbedaan masing-masing tipe data dan penggunaan nya	
	7	22/September	10	Create & Delete Database Create & Delete Table Insert & Update Table	Mengetahui macam-macam DDL dan DML Mampu membuat dan menghapus database Menampilkan data menggunakan order dan limit Mengetahui macam-macam DDL dan DML	

					Mampu membuat dan menghapus database Dapat mengisi dan memodifikasi isi tabel Menampilkan data menggunakan order dan limit	
	8	27/September	11	Perkenalan GraphQL Perkenalan REST API Persiapan	Mengetahui fungsi penggunaan GraphQL Dapat menjelaskan kembali apa itu REST API dan cara kerja nya Dapat mempersiapkan environment dan yang dibutuhkan untuk membuat API	
	8	29/September	12	Endpoint Create Endpoint Read Endpoint Update Endpoint Delete	Dapat membuat endpoint untuk membuat data baru pada database Dapat membuat endpoint untuk membaca data pada database Dapat membuat endpoint untuk memperbarui data pada database	

					Dapat membuat endpoint untuk menghapus data pada database	
Oktober	9	4 / Oktober	13	JWT Integrasi	Dapat membuat token dan autentikasi pada REST API yang telah dibuat untuk meningkatkan keamanan Dapat melakukan integrasi REST API dengan Aplikasi menggunakan JWT, dan melempar datanya ke front-end	16 / Oktober
		6 / Oktober	14	Perkenalan Testing	Mengetahui pentingnya melakukan testing dalam sebuah aplikasi	
		13 / Oktober	15	Tipe Testing Unit Testing Tools Continous Integration	Mampu menentukan jenis testing apa yang diperlukan / cocok untuk aplikasi yang sedang dibangun Menguasai tools yang ada pada unit testing Mampu menentukan tools yang digunakan sesuai dengan bahasa pemrograman pada sebuah	

					aplikasi yang akan dilakukan pengujian Dapat menggunakan Unit Testing	
		16 / Oktober	16	Deployment Final Project	Memahami konsep heroku dan dapat deploy aplikasi yang telah dibuat pada Heroku Membuat Database dan API web service Reflection API	

1.3 Landasan Teori

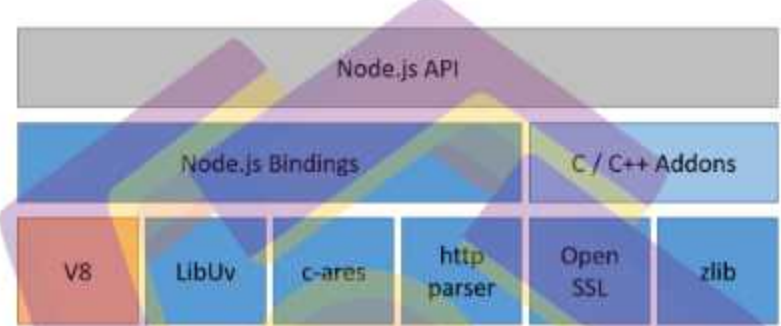
1.3.1 Node JS

Node.js merupakan salah satu platform pengembang yang dapat digunakan untuk membuat aplikasi berbasis *Cloud*. Node.js dikembangkan dari engine JavaScript yang dibuat oleh Google untuk browser Chrome ditambah dengan libuv serta beberapa pustaka lainnya. Node.js menggunakan JavaScript sebagai bahasa pemrograman dan *event-driven*, *non-blocking* I/O (asynchronous) model yang membuatnya ringan dan efisien. Node.js memiliki fitur *built-in* HTTP *server library* yang menjadikannya mampu menjadi sebuah web server tanpa bantuan software lainnya seperti *Apache* dan *Nginx* [1].

Pada dasarnya, Node.js adalah sebuah *runtime environment* dan *script library*. Sebuah *runtime environment* adalah sebuah software yang berfungsi untuk mengeksekusi, menjalankan dan mengimplementasikan fungsi-fungsi serta cara kerja inti dari suatu bahasa pemrograman. Sedangkan *script library* adalah kumpulan, kompilasi atau bank data berisi skrip/kode-kode pemrograman.

Node.js dibangun menggunakan JavaScript dan C++, terdapat arsitektur serta fungsi dari Google V8 di dalamnya yang berfungsi sebagai compiler ditulis dalam C++ dan library Libuv bekerja untuk menangani operasi *asynchronous* I/O dan *main event loop*.

Berikut merupakan gambar arsitektur Node.js :



Gambar 1. 1Asitektur Node.js

1.3.2 NPM *(Node Package Manager)*

NPM merupakan paket *manager* untuk Node.js, yang ditemukan pada tahun 2009 sebagai suatu proyek terbuka untuk membantu pemgembang Javascript saling berbagi kode paket modul. NPM juga merupakan sebuah kode perintah untuk memungkin pengembang menggunakan dan mempublikasi paket modul[2].

1.3.3 PostgreSQL

PostgreSQL adalah sebuah sistem basis data yang disebarluaskan secara bebas menurut Perjanjian lisensi BSD. Peranti lunak ini merupakan salah satu basis data yang paling banyak digunakan saat ini, selain MySQL dan Oracle. PostgreSQL menyediakan fitur yang berguna untuk replikasi basis data.

PostgreSQL (atau dikenal juga sebagai Postgres) adalah sebuah

RDBMS *open-source* (didistribusikan secara *free*) yang menekankan pada pemenuhan standar teknis dan fleksibilitas (keluwesan) data. PostgreSQL didistribusikan dengan lisensi bebas/gratis, sehingga dapat digunakan, dimodifikasi, dan didistribusikan kembali kepada publik secara bebas/gratis untuk tujuan pribadi, komersial, ataupun akademik. PostgreSQL dirancang untuk menangani beban kerja terhadap data dari sebuah mesin menuju layanan web yang diakses banyak orang secara bersamaan[3].

1.3.4 **ExpressJS**

ExpressJS merupakan minimal framework yang sangat fleksibel, memungkin pengguna untuk membut laman *server* HTML, statik file, layanan *web* dengan akses REST API atau aplikasi *hybrid* yaitu selain pengguna mempunyai akses melalui REST API juga dapat memiliki akses pada laman HTML[4].

1.3.5 **JWT *(JsonWebToken)***

JWT merupakan kunci kewajiban untuk mengamankan *authorization* API (*Aplication Programing Interface*) ketika digunakan di *end user*. JSON Web Token (JWT) adalah standar terbuka (RFC 7519) yang mendefinisikan sederhana dan mandiri untuk mengirimkan informasi antar pihak secara aman dalam format JSON.

1.3.6 REST (Representational State Transfer)

1.3.6.1 Definisi REST

REST (*Representational State Transfer*) merupakan seperangkat prinsip arsitektur yang melakukan transmisi data melalui antarmuka yang terstandarisasi seperti HTTP. REST API bekerja layaknya seperti aplikasi *web* biasa. *Client* dapat mengirimkan permintaan kepada *server* melalui protokol HTTP dan kemudian *server* memberikan respons balik kepada klien. REST dikembangkan oleh Roy Fielding yang merupakan *co-founder* dari Apache HTTP *Server Project*. Arsitektur REST menjelaskan enam batasan. Adapun keenam batasan arsitektur

REST adalah sebagai berikut [5]:

a. *Client Server*

Batasan *client server* menjelaskan suatu antarmuka yang memisahkan bagian *client* dan *server*. Melalui pemisahan antarmuka, REST memberikan keuntungan yaitu client tidak perlu berurusan dengan masalah penyimpanan. Hal tersebut meningkatkan portabilitas antarmuka pengguna di berbagai platform dan meningkatkan skalabilitas dengan menyederhanakan komponen *server*. Pemisahan antarmuka pengguna memungkinkan komponen dapat terus berkembang secara independen. *Uniform Interface* yang menghubungkan antara *client* dan *server*.

b. *Stateless*

Batasan ini menjadi hal penting pada arsitektur REST. Batasan ini menjelaskan bahwa *server* tidak menyimpan *state* atau penanda *client*. Maksudnya adalah setiap pesan yang dikirimkan oleh client bersifat *self-descriptive* atau dengan kata lain setiap pesan yang dikirimkan memiliki informasi atau konteks yang cukup untuk *server* dalam memeroses pesan tersebut. Setiap *state* atau penanda session tersimpan pada pihak *client*.

c. *Cacheable*

Batasan ini menjelaskan bahwa respons *server* bersifat *cacheable*. Setiap respons *server* dapat disimpan secara implisit, eksplisit, atau *negotiated*.

d. *Uniform Interface*

Unform interface menjelaskan antarmuka antara *client* dan *server*. Hal ini menyederhanakan dan memisahkan arsitektur kedua pihak, yang memungkinkan setiap bagian dapat berkembang secara independen. Batasan ini merupakan hal fundamental untuk desain RESTful. RESTful menggunakan HTTP method (GET, POST, PUT, DELETE) untuk menjelaskan metode permintaan, URI (*Uniform Resource Identifier*) untuk mengidentifikasi nama sumber, HTTP *response* (*status*, *body*) untuk menjelaskan informasi yang dikembalikan oleh *server*.

e. *layered System*

Batasan ini menjelaskan bahwa pihak *client* tidak bisa secara langsung terhubung ke *server*. Terdapat perantara antara *server* dan *client*. Hal tersebut berkaitan dengan poin pemisahan antara *client* dan *server*. Perantara tersebut meningkatkan skalabilitas dengan memungkinkan *load-balancing* dan dengan menyediakan *cache* bersama. Serta *layered system* dapat menerapkan kebijakan keamanan.

f. *Code On Demand* (Opsional)
Batasan ini menjelaskan bahwa *server* dapat memberikan atau menyesuaikan fungsionalitas *client* secara sementara dengan mentransferkan *logic* ke dalamnya sehingga dapat dijalankan. Contohnya yaitu komponen yang dikompilasi seperti Java applet dan Javascript.

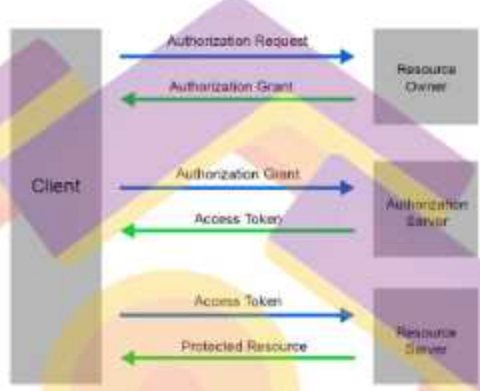
1.3.6.2 Keunggulan REST

Adapun beberapa keunggulan yang dimiliki oleh REST adalah sebagai berikut [6]:

a. REST menyediakan infrastruktur yang bagus dalam proses *caching* melalui metode HTTP GET. Hal ini dapat meningkatkan performa jika informasi tidak diubah dan tidak dinamis.
b. REST menyediakan infrastruktur yang bagus dalam proses *caching* melalui metode HTTP GET. Hal ini dapat meningkatkan performa jika informasi tidak diubah dan tidak dinamis.
c. REST dapat mengembalikan *respons* dalam format yang beragam dan sesuai dengan permintaan client.
d. REST dapat dikembangkan menggunakan bahasa pemrograman manapun selama bahasa tersebut dapat membuat permintaan berbasis *web* melalui HTTP.
e. REST cocok digunakan pada aplikasi perangkat bergerak.

1.3.7 Alur Protokol

Alur protokol merupakan alur yang memuat langkah – langkah client untuk mengakses sumber data. Terdapat aliran abstrak yang menggambarkan interaksi antar 4 peran dan mencakup beberapa langkah di antaranya yang ditunjukkan pada Gambar 1.2 adalah sebagai berikut [1]:



Gambar 1. 2 Alur Protokol

a. Client melakukan permintaan otorisasi dari resource owner. Permintaan otoritas dapat dilakukan secara langsung melalui resource owner ataupun secara tidak langsung melalui authorization server sebagai perantara.
b. Client menerima wewenang otorisasi dari resource owner, yang merupakan kredensial yang mewakili otorisasi resource owner.
c. Client meminta token akses dengan melakukan otentikasi terhadap authorization server dengan menyajikan wewenang otorisasi yang didapatkan pada langkah sebelumnya.
d. Authorization server melakukan otentikasi client dan validasi terhadap wewenang otorisasi. Jika hasil proses tersebut bernilai valid, maka token akses akan diterbitkan.
e. Client melakukan permintaan sumber data terlindungi dari resource server

dan melakukan otentikasi dengan memberikan token akses.

f. Resource server melakukan validasi terhadap token akses yang diberikan client. Apabila hasilnya valid, maka resource server mengembalikan respons terhadap permintaan client.