BAB II

TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Dalam penelitian yang dilakukan oleh Hasan (2011) tentang aplikasi pencarian lokasi kuliner di Yogyakarta. Penelitian tersebut telah menghasilkan sebuah sistem pencarian lokasi kuliner berbasis *mobile web* untuk wilayah Yogyakarta yang memberikan informasi kedekatan jarak lokasi pengguna dengan lokasi tempat, rute jalan, dan penunjuk arah jalan.

Selain Penelitian di atas, juga pernah dilakukan oleh Sugiarto (2010) tentang Pemanfaatan *Google Map Service* untuk Sistem Informasi Pariwisata Kabupaten Gunungkidul. Penelitian tersebut telah menghasilkan sebuah aplikasi sistem informasi geografis yang dapat menampilkan informasi obyek-obyek wisata di Kabupaten Gunungkidul melalui *website*. Aplikasi tersebut digunakan untuk mencari obyek-obyek wisata dan memberikan informasi lengkap tentang obyek wisata yang ditemukan. Pengguna dapat juga memperoleh rute perjalanan dari lokasi pengguna ke lokasi obyek wisata.

Penelitian lain juga dilakukan oleh Hunarso (2014) tentang aplikasi pencarian lokasi rumah sakit di wilayah Yogyakarta. Penelitian menghasilkan sebuah aplikasi berbasis *mobile Android* yang dapat membantu dalam mencari Rumah Sakit yang ada di Wilayah Yogyakarta, serta memberikan detail informasi mengenai rumah sakit – rumah sakit tersebut dan info rute perjalanan menuju lokasi tersebut.

Sedangkan Aplikasi yang akan dikembangkan digunakan untuk mencari lokasi kafe hotspot dan memiliki fitur untuk memberikan kritik dan masukan pada kafe tersebut, sehingga tidak hanya berguna untuk *user* tetapi juga untuk pengelola kafe untuk meningkatkan fasilitas kafe mereka melalui *review* dan *feedback* dari pengunjung.

Untuk perbandingan selanjutnya akan disajikan dalam bentuk tabel 2.1 berikut ini :

Tabel 2.1 Data Penelitian Tentang Sistem Pencarian Lokasi

Parameter / Penulis	Objek	Metode	Bahasa pemprograman	Interface
Taufik Hasan (2011)	Pencarian Lokasi Kuliner	MAP API	JAVA	GUI
Sugiarto (2010)	Sistem Informasi Pariwisata	MAP API	JAVA	GUI
Hunarso (2014)	Pencarian Lokasi Rumah Sakit	MAP API	JAVA	GUI
Ahmad (2016)	Pencarian cafe berhotspot	MAP API	Java Script	Web

2.2 Dasar Teori

2.2.1 Web Development

Web development adalah proses pembuatan sebuah *website* untuk internet atau sebuah *intranet*. Dalam pembuatan sebuah *website* digunakan berbagai macam bahasa pemrograman. Diantaranya adalah :

2.2.1.1 HTML

HTML (*Hyper Text Mark Up Language*) merupakan bahasa yang digunakan untuk mendeskripsikan struktur sebuah halaman *web* HTML berfungsi untuk mempublikasi dokumen *online*(W3.org). *Statement* dasar dari HTML disebut *tags*. Sebuah *tag* dinyatakan dalam sebuah kurung siku (<>). *Tags* yang ditujukan untuk sebuah dokumen atau bagian dari suatu dokumen haruslah dibuat berupa pasangan. Terdiri dari *tag* pembuka dan tag penutup. Dimana *tag* penutup menggunakan tambahan tanda garis miring (/) di awal nama *tag*. Contohnya <html> merupakan *tag* pembuka dan </html> merupakan *tag* penutupnya (Henderson..2009 : 232).

2.2.1.2 HTML 5

HTML 5 merupakan pengembangan terbaru dari HTML. *Syntax* pada HTML5 dibuat lebih sederhana untuk memudahkan *webdeveloper*. HTML 5 memperkenalkan fitur- fitur baru seperti animasi, audio, transisi, tipografi, dan masih banyak lagi. Dengan begitu, HTML 5 dapat menggantikan permasalahan - permasalahan pada teknologi web sebelumnya. Salah satu contohnya adalah menggantikan penggunaan *flash*

pada *web* yang membutuhkan waktu lebih lama untuk melakukan load sebuah halaman (Granell. 2012 :6).

2.2.1.3 CSS

Css adalah bahasa-bahasa yang merepresentasikan halaman web. Seperti warna, *layout*, dan huruf. Dengan menggunakan CSS, seorang *web developer* dapat membuat halaman web yang dapat beradaptasi dengan berbagai macam ukuran layar. Pembuatan CSS biasanya terpisah dengan halaman HTML. Meskipun CSS dapat disisipkan di dalam halaman HTML. Hal ini ditujukan untuk memudahkan pengaturan halaman HTML yang memiliki rancangan yang sama (W3.org).

2.2.1.4 JavaScript

Sebuah *script* adalah kode program yang tidak butuh pra-proses (*compiling*) sebelum dijalankan. Dalam konteks pembuatan web, kode programnya ditulis dalam *Javascript*. *Javascript* dieksekusi oleh *web browser* pada saat mengunduh halaman web atau pada saat sebuah *event* dilakukan oleh pengguna. *Javascript* merupakan salah satu bahasa pemrograman populer yang mampu membuat halaman web dapat berinteraksi dengan penggunanya. *Javascript* pertama kali dikembangkan pada pertengahan dekade 90an. *Javascript* dapat disisipkan di dalam dokumen HTML ataupun dijadikan dokumen tersendiri yang kemudian diasosiasikan dengan dokumen lain yang dituju (Flanagan, 2006)

2.2.2 NodeJs

Node.js merupakan salah satu peranti pengembang yang bisa digunakan untuk membuat aplikasi berbasis *Cloud*. *Node.js* dikembangkan dari *engine JavaScript* yang dibuat oleh *google* untuk *browser Chrome / Chromium (V8)* ditambah dengan libUV serta beberapa pustaka internal lainnya. Dengan menggunakan *Node.js*, semua pengembangan akan dilakukan menggunakan *JavaScript*, baik pada sisi klien maupun *server*. *Node.js* dibuat pertama kali oleh Ryan Dahl (twitter.com/ryah) dan sampai saat ini dikembangkan oleh komunitas sebagai perangkat lunak bebas dengan pendanaan utama dari Joyent, perusahaan tempat Ryan Dahl bekerja (Purnomosidi, 2013 : 15).

2.2.3 Web Browser

Web browser adalah aplikasi perangkat lunak yang berfungsi untuk mentransfer informasi pada internet. Dengan begitu, pengguna dapat mengakses halaman *web* pada internet. Selain itu *web browser* juga dapat digunakan untuk mengakses data dalam *web server* di dalam sebuah jaringan yang tertutup. *Web browser* tidak hanya diperuntukan untuk komputer *desktop* saja. Hadir pula *mobile web browser* yang digunakan untuk telepon selular yang memiliki kemampuan untuk mengakses internet. *Web browser* mengirimkan *request* sebuah halaman *web* ke *webserver* dengan menggunakan alamat *web* tersebut. Kemudian *server* bertugas untuk mengirimkan halaman *web* yang sesuai dengan *request* untuk selanjutnya ditampilkan di *web browser*. Pada umumnya, *web browser* menyimpan *file* dan dokumen yang *direquest* pada *local cache. Cache* berfungsi untuk mengurangi

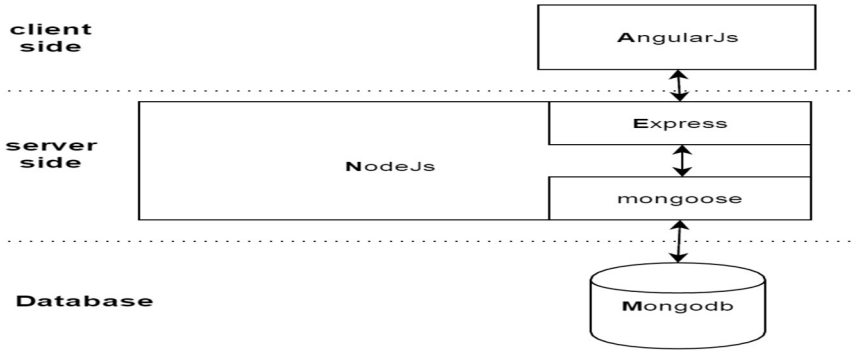
jumlah data yang harus dikirim kembali jika *web browser* akan melakukan *request* terhadap halaman *web* yang pernah diakses (Henderson.2009 :503).

2.2.4 Arsitektur *MEAN Stack*

MEAN Stack adalah kependekan dari *MongoDB, ExpressJS, AngularJS* dan *Node.JS.* Semua *tools* yang tergabung dalam *MEAN Stack* tersebut saling melengkapi satu sama lain, *MongoDB* sebagai media penyimpanan basis data yang bersifat *NoSQL,* berbeda dengan *MS SQL, MySQL, PostgreSQL* ataupun *Oracle* yang merupakan jenis *relational database*, *MongoDB* tidak menggunakan relasi, sehingga tidak menggunakan metode penggabungan untuk menggabungkan hasil *query* dari beberapa *collection / table,* dengan menggunakan *MongoDB* sangat memudahkan jika suatu saat sistem yang dibuat berkembang dan membutuhkan data yang lebih kompleks, karena tidak menggunakan *join* maka *query* untuk memanggil *tabel / collection* yang banyak cukup mudah jika dibandingkan dengan traditional database. *Tools* yang kedua dalam susunan *MEAN Stack* adalah *ExpressJS* yaitu sebuah *web framework* untuk memudahkan membuat aplikasi dibagian *server* yang berfungsi untuk mengatur *routing*, menghandel *request* dari klien dan *views* atau yang mengatur sistem menjadi *Model View Controller (MVC)* yang berjalan diatas *platform* NodeJS. Yang ketiga adalah *AngularJS* yang merupakan *framework* untuk membangun *front-end* atau tampilan *user interface* yang mengarah ke klien, dan yang terakhir adalah *NodeJs* yang merupakan *platform* tempat dimana *MongoDB, ExpressJS,* dan *AngularJS* berjalan. Berikut adalah beberapa keuntungan menggunakan teknologi *MEAN Stack* :

1. Menggunakan satu bahasa pemrograman untuk *server* dan klien nya yaitu *Javascript*.
2. Memiliki *performance* yang baik karena *NodeJs* menggunakan teknologi *non-blocking io* dan menggunakan *engine Chromium (V8)* dari *google.*
3. Menggunakan teknologi *Single Page Application* (*AngularJS*) yang menggunakan satu halaman web saja sebagai tampilan dari aplikasinya. Semua aksi klik atau pun penyajian data tidak akan membuat halaman secara utuh dimuat ulang (*reload*), tetapi hanya sebagian-sebagian saja yang *diupdate* dari *server* atau dari hasil proses aplikasi di sisi klien.
4. Memiliki komunitas aktif yang besar sehingga memudahkan pengembang untuk mencari *tools* maupun *library NodeJs*.
5. Didukung oleh perusahaan *cloud computing* raksasa seperti *Amazon Web Services* dan juga google (*AngularJS*).
6. Banyaknya perusahaan IT terkemuka yang telah sukses menggunakan teknologi *MEAN Stack*.

Agar lebih jelas tentang arsitektur MEAN Stack bisa dilihat pada gambar 2.1 berikut :



Gambar 2.1 Arsitektur *MEAN Stack*

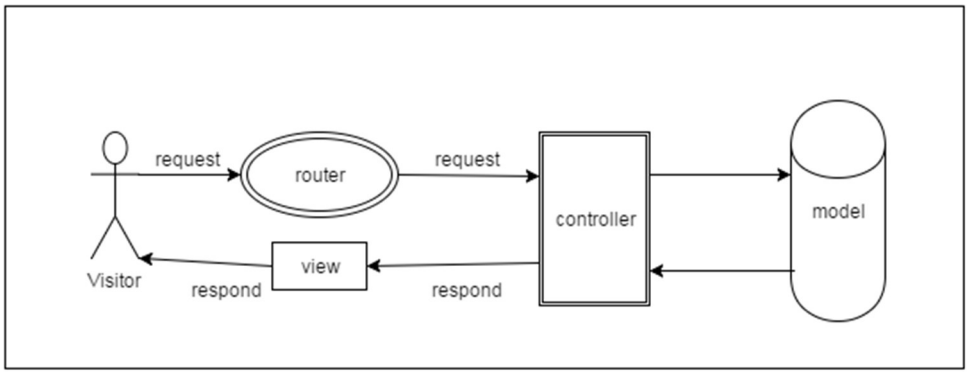
Pada Gambar 2.1 menjelaskan mengenai relasi antara klien dan *server* melalui jaringan *internet*, klien mengakses antar muka yang dibuat menggunakan *AngularJS* yang diteruskan ke *ExpressJS*, kemudian *ExpressJS merouting* permintaan dari klien ke *Mongoose* yang menghubungkan ke *MongoDB* untuk mengambil data yang diminta dan merespon permintaan yang akhirnya disampaikan kembali ke klien.

2.2.5 Arsitektur *Model View Controller* (MVC)

MVC adalah arsitektur aplikasi yang memisahkan kode-kode aplikasi dalam tiga lapisan yaitu, *Model, View* dan *Control* yang bertujuan untuk memisahkan logika bisnis dari pertimbangan antar muka pengguna agar lebih mudah mengubah setiap bagian tanpa mempengaruhi yang lain. Dalam *MVC*, model menggambarkan informasi (data) dan aturan bisnis; *view* (tampilan) berisi elemen antar muka pengguna seperti teks, *input form*, sementara *controller* mengatur komunikasi antar *model* dan *view.*

- **Model** adalah level paling bawah pada arsitektur MVC yang bertanggung jawab mengatur data.
- **View** bertanggung jawab untuk menampilkan seluruh data kepada *user*.
- **Controller** berisi logika untuk mengontrol interaksi antara *model* dan *view*

Untuk lebih jelasnya bisa dilihat pada gambar 2.2 berikut :

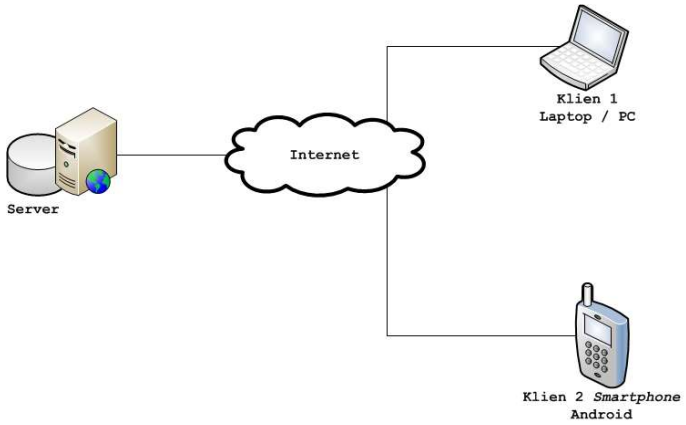


Gambar 2.2 Arsitektur *Model-View-Controller (MVC)*

Pada gambar di atas dapat kita lihat untuk menampilkan suatu *request* dari pengguna, pertama akan melali sebuah *router (express*) kemudian dalam *router* tersebut akan memberikan *request* kepada *controller* yang kemudian diambilkan dari *model*. Setelah itu *request* akan mendapatkan respon.

2.2.6 Struktur Jaringan

Pada bagian struktur jaringan ini menggambarkan relasi antara klien dan *server*, untuk membuktikan bahwa aplikasi berbasis web ini bersifat *cross platform* maka klien menggunakan *smartphone android* dan laptop atau *PC*.



Gambar 2.3 Struktur Jaringan

Pada Gambar 2.3 menjelaskan mengenai relasi antara klien dan *server* melalui jaringan *internet*, klien mengakses antar muka yang dibuat menggunakan *AngularJS* yang diteruskan ke *ExpressJS*, kemudian *ExpressJS merouting* permintaan dari klien ke *MongoDB* untuk mengambil data yang diminta dan merespon permintaan yang akhirnya disampaikan kembali ke klien.

2.2.7 NPM (*Node Package Manager*)

Npm adalah kependekan dari *Node Package Manager* adalah sebuah *online repository* yang digunakan untuk mem-*publish project NodeJs* yang *berbasis open-source*, npm juga merupakan *command-line utility* yang digunakan untuk berinteraksi dalam instalasi paket - paket yang berbasis *nodejs*(Purnomosidi, 2013 : 17)