

## Aplikasi "Find Food" Di Wilayah Bandung Berbasis APEX Menggunakan *Google Maps*

<sup>1</sup>Francisca Lady Nice R Sp, <sup>2</sup>Suryatiningsih, S.T., M.T., OCA, <sup>3</sup>Asniar, S.T., M.T.

Program Studi D3 Manajemen Informatika, Fakultas Ilmu Terapan, Universitas Telkom

<sup>1</sup>francisca.lady@gmail.com, <sup>2</sup>[suryatiningsih@tass.telkomuniversity.ac.id](mailto:suryatiningsih@tass.telkomuniversity.ac.id), <sup>3</sup>asniar@tass.telkomuniversity.ac.id

### Abstrak

Bandung adalah salah satu tempat tujuan wisata yang sering dikunjungi karena wisata kulinernya, bukan hanya dari masyarakat Bandung sendiri, tetapi dari luar kota Bandung. Pengetahuan yang kurang terhadap lokasi tempat makan membuat kebanyakan masyarakat bingung dalam menemukan tempat makan. Penelusuran *google* dirasakan kurang efektif untuk mencari dan menemukan tempat makan yang dibutuhkan. Berdasar latar belakang tersebut, juga melihat dari fitur yang dimiliki *google map*, muncul gagasan untuk membangun aplikasi alternatif bagi masyarakat dengan tujuan membantu masyarakat dalam mencari dan menemukan tempat makan di Bandung. Pembuatan aplikasi ini dilakukan dengan metode *Rapid Application Development (RAD)*. Aplikasi berbasis *desktop* dan di implementasikan pada *Application Express (APEX)* dengan fitur *Google Map* untuk penunjuk lokasi tempat makan dan *database Oracle* sebagai media penyimpanan data. Aplikasi ini telah diuji menggunakan metodologi *BlackBox Testing*. Berdasarkan hasil pengujian tersebut, disimpulkan bahwa aplikasi dapat menyimpan, mengolah dan menampilkan data tempat makan dengan menggunakan fitur *Google Map* sebagai penunjuk lokasi tempat makan. Disediakan juga fitur untuk melakukan kelola tempat makan, kelola menu, kelola komentar dan kelola *rating* sebagai fitur yang akan didapatkan apabila pengguna mendaftarkan diri menjadi *member* aplikasi ini.

**Kata Kunci:** Tempat Makan Bandung, aplikasi, *Application Express (APEX)*, *Google Maps*, *Rapid Application Development (RAD)*, *BlackBox Testing*

### Abstract

*Bandung is one of the most visited tourist destination for culinary tourism, not only of the people of Bandung itself, but from outside the city. Less knowledge of the location of culinary place, make most people are confused in finding a culinary place. Google searching felt less effective to seek and find needed culinary place. Based on this background, and the features of the google map, the idea coming to build an alternative application for peoples with the purpose to helping peoples for seeking and finding a culinary place in Bandung. This application was made by the Rapid Application Development (RAD) method. This application based on Desktop and implemented in Application Express (APEX) with Google Map feature to show the location of culinary place and Oracle database as data storage media. This application has been tested using BlackBox Testing methodology. Based on these test results, it was concluded that the application can store, process and display culinary place's data using Google Map as feature to show location of culinary place. Also, provided a feature to manage culinary place, menu's management, and manage comments, manage rating as a additional feature if user registered to be a member of this application.*

**Key words:** Culinary Place Of Bandung, application, *Application Express (APEX)*, *Google Maps*, *Rapid Application Development (RAD)*, *BlackBox Testing*

### 1. Pendahuluan

*Oracle Application Express (OAE)* adalah *host* pengembangan deklaratif lingkungan untuk mengembangkan dan menggunakan *database* sentris aplikasi web, juga menciptakan atau memodifikasi metadata yang disimpan didalam tabel *database*. *Oracle Database XE* mencakup antarmuka berbasis *browser* pengguna yang intuitif dan dibangun di dalam *Oracle Application Express*. *Google Maps API (Application Programming Interface)* adalah sebuah *library* yang berbentuk *JavaScript*. Atau sederhananya bisa diartikan sebagai kode program yang merupakan antarmuka atau penghubung antara aplikasi atau *web* yang dibuat dengan fungsi-fungsi yang dikerjakan dimana *programmer* bisa menggunakannya sebagai data peta dunia sehingga dapat membantu dalam membuat sebuah aplikasi. Dengan melihat bermacam fitur yang dimiliki *Google Maps*, maka terdapat sebuah gagasan untuk membuat sebuah aplikasi yang dapat menerapkan fitur yang dimiliki *Google Maps* ke dalam *Application Express (APEX)*.

Bandung adalah salah satu kota metropolitan terbesar di Jawa Barat, yang juga menjadi ibu kota provinsi. Kota Kembang adalah sebutan lain untuk kota ini, begitu juga dengan sebutan *Parijs Van Java* karena keindahannya. Selain itu kota Bandung juga dikenal sebagai kota belanja, pariwisata, pendidikan, sampai wisata kuliner. Salah satu yang akan dibahas adalah wisata kulinernya. Melihat dari tingkat ketertarikan masyarakat yang tinggi untuk mencicipi kuliner kota Bandung, baik itu masyarakat dari luar bandung ataupun bandung sendiri, mereka datang untuk mencicipi aneka ragam makanan dan minuman ataupun jajanan daerah Bandung, namun hanya saja minat masyarakat yang besar

itu tidak dibarengi dengan pengetahuan akan tempat-tempat makan yang ada di Bandung, atau bahkan terkadang juga kebingungan untuk memilih tempat makan yang tepat dan sesuai dengan selera ataupun sesuai dengan kantong mereka. Dewasa ini penggunaan teknologi telah menjadi bagian hidup bagi masyarakat, dengan menelusuri pencarian menggunakan *google* untuk menemukan daftar tempat makan di Bandung. Hanya saja aktivitas seperti demikian dirasakan kurang maksimal. Berdasarkan masalah di atas, dibutuhkan sebuah aplikasi yang dapat membantu masyarakat dalam menemukan tempat makan di Bandung.

Aplikasi "Find Food" merupakan suatu perangkat lunak yang dibuat dengan tujuan sebagai aplikasi yang mampu menyajikan informasi tempat makan di Bandung kepada masyarakat. Penggunaan aplikasi ini sebagai pemandu masyarakat untuk menemukan tempat makan yang dicari berdasarkan lokasi, nama tempat makan dan subkategori tempat makan, yang akan dibantu menggunakan *google map* untuk menampilkan peta tempat makan yang tepat dan akurat. Juga aplikasi ini dapat melakukan *input* tempat makan dan menu-menu yang dimiliki tempat makan tersebut yang selanjutnya tempat makan akan muncul di peta pencarian. Fungsi tambahan yang dimiliki aplikasi ini adalah *member* dapat melakukan *input* komentar dan *input rating* tempat makan.

Diharapkan pembuatan aplikasi "Find Food" ini bukan hanya sebagai sebuah aplikasi alternatif pencarian tempat makan yang dapat membantu masyarakat dalam menemukan tempat makan di Bandung, tetapi juga merupakan aplikasi yang mampu menerapkan fitur yang dimiliki *Google Maps* ke dalam *Application Express (APEX)*.

## 2. Landasan Teori

### 2.1 Aplikasi

Aplikasi berasal dari kata *Application* yang artinya penerapan lamaran atau penggunaan. Secara istilah, aplikasi adalah program siap pakai yang direka untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain dan dapat digunakan oleh sasaran yang dituju. Contoh aplikasi adalah program pemroses kata. Aplikasi merupakan sekumpulan elemen yang saling berinteraksi dan saling berketerkaitan antara satu dengan yang lainnya dalam melakukan suatu kegiatan secara bersama-sama untuk mencapai tujuan tertentu. Sebagai contoh, aplikasi komputer yang terdiri dari *software*, *hardware* dan *brainware*. Ketiga unsur tersebut saling berkaitan dan bertanggung jawab dalam memproses *input* (masukan) dan menghasilkan *output* (keluaran). Aplikasi dapat diartikan juga sebagai program komputer yang dibuat untuk menolong manusia dalam melaksanakan tugas tertentu. Aplikasi *software* yang dirancang untuk penggunaan praktis khusus dan klasifikasi luas ini dapat dibagi menjadi 2 (dua), yaitu :

1. Aplikasi *software* spesialis, yaitu program dengan dokumentasi terdapat yang dirancang untuk menjalankan tugas tertentu.
2. Aplikasi paket, yaitu suatu program dengan dokumentasi terdapat yang dirancang untuk jenis masalah tertentu [4].

### 2.2 Oracle Application Express

Basis data *Oracle* adalah basis data relasional yang terdiri dari kumpulan data dalam suatu sistem manajemen basis data RDBMS. Perusahaan perangkat lunak. *Oracle* memasarkan jenis basis data ini untuk bermacam-macam aplikasi yang bisa berjalan pada banyak jenis dan merk perangkat keras komputer (*platform*). Basis data *Oracle* ini pertama kali dikembangkan oleh Larry Ellison, Bob Miner dan Ed Oates lewat perusahaan konsultasinya bernama *Software Development Laboratories* (SDL) pada tahun 1977. Pada tahun 1983, perusahaan ini berubah nama menjadi *Oracle Corporation* sampai sekarang. *Oracle* merupakan RDBMS yang paling banyak digunakan oleh perusahaan-perusahaan di dunia. *Database Oracle* adalah produksi dari *Oracle Corporation*, sebuah perusahaan komputer raksasa yang bermarkas di Redwood City, California. Pengembangan *Oracle* dimulai pada tahun 1977 ketika Larry Ellison melihat ada peluang bagus yang dimanfaatkan perusahaan-perusahaan *software* kala itu. Tahun 1979 versi pertama dijual kepada umum. Versi pertama menyertakan *interface SQL* untuk berinteraksi dengan *database*. Tahun 1983, mereka mengubah nama perusahaan menjadi *Oracle Corporation*. Pada tahun itu juga, *Oracle Corporation* meluncurkan versi ketiga. Tahun 1984 versi keempat diluncurkan, *Oracle* telah mendukung beberapa sistem operasi yang ada waktu itu. Pengembangan terus dilakukan sesuai dengan perkembangan teknologi komputer. Tahun 1998, *Oracle* mulai mengadopsi konsep Orientasi Objek (*Object Oriented*). Konsep orientasi objek pada *database* sedikit berbeda dengan konsep yang dikenal di lingkungan internet sehingga mereka membubuhkan huruf 'i' yang merupakan huruf awal "internet". Sampai saat ini *Oracle* telah mengeluarkan versi terbarunya, yakni *Oracle 11g*.

Keunggulan *Oracle Application Express* antara lain :

1. *Scalability*, memiliki kemampuan menangani banyak *user* yang melakukan koneksi secara bersamaan tanpa berkurangnya *performance*.
2. *Reliability*, memiliki kemampuan untuk melindungi data dari kerusakan jika terjadi kegagalan fungsi pada sistem seperti *disk failure*.
3. *Service ability*, memiliki kemampuan untuk mendeteksi masalah, kecepatan dalam mengoreksi kesalahan, dan kemampuan melakukan konfigurasi ulang struktur data.
4. *Stability*, memiliki kemampuan untuk *crash* karena beban *load* yang tinggi. Hal ini berkaitan dengan *scalability*.

5. *Availability*, yaitu kemampuan dalam penanganan *crash* atau *failure* agar *service* dapat tetap berjalan.
6. *Multiplatform*, dapat digunakan pada banyak sistem operasi seperti *windows*, *unix*, *linux*, dan *solaris*.
7. Mendukung data yang sangat besar.
8. Sistem keamanan yang cukup handal.
9. Dapat menampung hampir semua tipe data seperti data teks, *image*, *sound*, *video*, dan *time series*. [5].

### 2.3 Google Maps

*Google Maps* adalah layanan pemetaan *online* yang paling banyak digunakan di seluruh penjuru dunia. Merupakan perangkat lunak berbasis internet yang berisi peta atas sebuah wilayah atau lokasi. Peta tersebut menunjukkan gambaran peta konvensional yang dibuat secara cetak [6].

#### 2.3.1 Google Static Maps API

*Google Static Maps API* memungkinkan Anda menanamkan *Google Maps* gambar pada halaman web Anda tanpa memerlukan *JavaScript* atau halaman dinamis setiap pemuat. *Google Static Map* layanan menciptakan peta Anda berdasarkan parameter URL yang dikirim melalui permintaan HTTP standar dan mengembalikan peta dengan gambar Anda dapat menampilkan pada halaman web Anda. Semua aplikasi *Static Maps API* harus menggunakan kunci API.

Untuk menentukan kunci dalam permintaan Anda, memasukkannya sebagai nilai parameter kunci. Contoh dalam dokumen ini mengecualikan parameter kunci sehingga mereka akan bekerja untuk orang-orang yang memotong dan menyisipkan kode.

*Google Static Maps API* mengembalikan gambar (baik GIF, PNG atau JPEG) dalam menanggapi permintaan HTTP melalui URL. Untuk setiap permintaan, Anda dapat menentukan lokasi peta, ukuran gambar, tingkat *zoom*, jenis peta dan penempatan tanda opsional di lokasi pada peta. Anda dapat menambahkan label penanda Anda menggunakan karakter alfanumerik. *Static Maps API* gambar tertanam dalam *src* atribut sebuah *<img>* tag, atau setara dalam bahasa pemrograman lain. Jika API gambar *static Maps* digunakan di luar aplikasi berbasis web (seperti *browser*) maka link harus disertakan menunjuk ke lokasi yang ditampilkan dalam browser web atau *native* aplikasi *Google Maps*.

Parameter lokasi adalah *center* (diperlukan jika penanda tidak ada) mendefinisikan tengah peta, berjarak sama dari semua sisi dari peta. Parameter ini mengambil lokasi baik sebagai {*lintang*, *bujur*} pasangan dipisahkan koma (misalnya "40,714728, -73,998672") atau alamat *string* (misalnya "balai kota, new york, ny") mengidentifikasi lokasi yang unik di wajah bumi. *Zoom* (diperlukan jika penanda tidak ada) mendefinisikan tingkat *zoom* peta, yang menentukan tingkat pembesaran peta. Parameter ini mengambil nilai numerik sesuai dengan tingkat *zoom* daerah yang diinginkan.

Gaya (opsional) mendefinisikan gaya kustom untuk mengubah presentasi dari fitur tertentu (jalan, taman, dll) dari peta. Parameter ini mengambil fitur dan elemen argumen mengidentifikasi fitur untuk memilih dan seperangkat operasi gaya untuk menerapkan seleksi itu. Anda dapat menyediakan beberapa gaya dengan menambahkan parameter gaya tambahan. *Google Maps API* untuk pengguna Kerja harus mencakup klien dan tanda tangan yang *valid* parameter dengan permintaan mereka [2].

### 2.3.2 Google Maps JavaScript V3

Dokumentasi ini dirancang untuk orang yang terbiasa dengan pemrograman *JavaScript* dan konsep pemrograman berorientasi objek. Anda juga harus akrab dengan *Google Maps* dari sudut pengguna pandang. Ada banyak tutorial *JavaScript* yang tersedia di web. Dokumentasi konseptual ini dirancang untuk membiarkan Anda dengan cepat mulai mengeksplorasi dan mengembangkan aplikasi dengan *Google Maps API*. *Google* menerbitkan *Google Maps API Reference*. Memeroleh *Key API*. Semua *Maps* aplikasi API harus memuat *Maps API* menggunakan kunci API. Menggunakan kunci API memungkinkan Anda untuk memonitor penggunaan *Maps API* aplikasi Anda, dan memastikan bahwa *Google* dapat menghubungi Anda tentang aplikasi Anda jika perlu. Jika *Maps* penggunaan API aplikasi Anda melebihi batas penggunaan, Anda harus memuat *Maps API* menggunakan kunci API untuk membeli kuota tambahan. Memuat *Google Maps API*. URL yang terkandung dalam *tag script* adalah lokasi *file JavaScript* yang memuat semua simbol dan definisi yang Anda butuhkan untuk menggunakan *Google Maps API*. *Tag script* ini diperlukan. Parameter kunci berisi kunci API aplikasi Anda. HTTPS atau HTTP adalah kami pikir keamanan di web cukup penting, dan menyarankan menggunakan HTTPS bila memungkinkan. Sebagai bagian dari upaya kami untuk membuat web lebih aman, kami telah membuat semua *Google Maps API* tersedia melalui HTTPS. Menggunakan enkripsi HTTPS membuat situs Anda lebih aman, dan lebih tahan terhadap gangguan atau intaian.

*Library* adalah ketika memuat *JavaScript Maps API* melalui URL Anda opsional dapat memuat *library* tambahan melalui penggunaan parameter perpustakaan URL. *Library* adalah modul kode yang menyediakan fungsionalitas tambahan untuk utama *JavaScript API* tapi tidak dimuat kecuali Anda secara khusus meminta mereka. *Asynchronous* memuat API, Anda mungkin ingin memuat kode *Maps API JavaScript* demi halaman telah selesai dimuat, atau sesuai permintaan. Untuk melakukannya, Anda dapat menambahkan sendiri `<script>` Anda *tag* dalam menanggapi peristiwa *window.onload* atau panggilan fungsi, tetapi Anda perlu tambahan menginstruksikan *Maps JavaScript API bootstrap* untuk menunda eksekusi kode aplikasi Anda sampai kode *API Maps JavaScript* diberi beban penuh. Anda dapat melakukannya dengan menggunakan parameter *callback*, yang mengambil sebagai argumen fungsi untuk mengeksekusi setelah menyelesaikan memuat API. Kode berikut menginstruksikan aplikasi untuk memuat *Maps API* setelah halaman telah terisi penuh (menggunakan *window.onload*) dan menulis *Maps JavaScript API* menjadi `<script>` *tag* di dalam halaman [3].

### 2.4 Javascript Object Notation(JSON)

*Javascript Object Notation (JSON)* adalah format data *interchange* ringan berbasis teks yang bersifat *language-independent*. Relatif sangat mudah bagi manusia untuk membaca dan menulis JSON. Relatif sangat mudah pula bagi mesin (komputer) untuk mengurai dan menghasilkan JSON. JSON dibuat dengan berbasiskan *JavaScript*, standar ECMA-262 Edisi 3-Desember 1999. JSON merupakan format teks yang benar-benar bahasa independen tetapi menggunakan konvensi yang akrab bagi *programmer* dari keluarga bahasa C, termasuk C, C++, C#, Java, JavaScript, Perl, Python, dan banyak lainnya. Properti ini membuat JSON sebagai bahasa data *interchange* yang ideal [10].

### 2.5 Flowmap

*Flowmap* adalah penggambaran secara grafik dari langkah-langkah dan urutan prosedur dari suatu program. *Flowmap* berguna untuk membantu analisis dan *programmer* untuk memecahkan masalah kedalam segmen yang lebih kecil dan menolong dalam menganalisis alternatif pengoperasian. Biasanya *flowmap* mempermudah penyelesaian suatu masalah khususnya masalah yang perlu dipelajari dan dievaluasi lebih lanjut.

Fungsi *flowmap* mendefinisikan hubungan antar bagian (pelaku proses), proses (manual/berbasis komputer) dan aliran data (dalam bentuk dokumen *input* dan *output*). Untuk membuat sebuah analisis menggunakan *flowmap* seorang analisis dan *programmer* memerlukan beberapa tahapan, diantaranya:

1. *Flowmap* digambarkan dari halaman atas ke bawah dan dari kiri ke kanan.
2. Aktivitas yang digambarkan harus didefinisikan secara hati-hati dan definisi ini harus dapat dimengerti oleh pembacanya.
3. Kapan aktivitas dimulai dan berakhir harus ditentukan secara jelas.
4. Setiap langkah dari aktivitas harus diuraikan dengan menggunakan deskripsi kata kerja.
5. Setiap langkah dari aktivitas harus berada pada urutan yang benar.
6. Ruang lingkup dan *range* dari aktivitas yang sedang digambarkan harus ditelusuri dengan hati-hati. Percabangan-percabangan yang memotong aktivitas yang sedang digambarkan tidak perlu digambarkan pada *flowchart* yang sama. Simbol konektor harus digunakan dan percabangannya diletakkan pada halaman yang terpisah atau hilangkan seluruhnya bila percabangannya tidak berkaitan dengan sistem.
7. Gunakan simbol yang terdapat dalam *flowmap* yang sudah tersedia [7].

### 2.6 Data Flow Diagram (DFD)

Informasi yang ada di dalam perangkat lunak dimodifikasi dengan beberapa transformasi yang dibutuhkan. *Data Flow Diagram (DFD)* atau dalam bahasa Indonesia menjadi Diagram Alir Data (DAD) adalah representasi grafik yang menggambarkan aliran informasi dan transformasi informasi yang diaplikasikan berbagai data yang mengalir dari masukan (*input*) dan keluaran (*output*). DFD dapat digunakan untuk mempresentasikan sebuah sistem atau perangkat lunak pada beberapa level abstraksi. DFD dapat dibagi menjadi beberapa level yang lebih detail untuk mempresentasikan aliran informasi atau fungsi yang lebih detail. DFD menyediakan mekanisme untuk pemodelan fungsional ataupun pemodelan aliran informasi. Oleh karena itu, DFD lebih sesuai digunakan untuk memodelkan fungsi-fungsi perangkat lunak yang akan diimplementasikan menggunakan pemrograman terstruktur karena pemrograman terstruktur membagi-bagi bagiannya dengan fungsi-fungsi dan prosedur-prosedur.

DFD tidak sesuai untuk memodelkan sistem perangkat lunak yang akan dibangun menggunakan pemrograman berorientasi objek. Paradigma pemrograman terstruktur dan pemrograman berorientasi objek merupakan hal yang berbeda. Jangan mencampur adukkan pemrograman terstruktur dan pemrograman berorientasi objek.[1]

### 2.7 Entity Relationship Diagram (ERD)

*Entity Relationship Diagram (ERD)* atau diagram ER sebagai salah satu pemodelan konseptual basis data. Di dalam membuat sebuah basis data, hal pertama yang harus dilakukan adalah mendesain tabel-tabel yang akan digunakan untuk menyimpan data sesuai bisnis proses yang kita inginkan. Proses konseptual adalah pandangan secara konsep tentang basis data. Pandangan konseptual ini tentunya harus bisa diimplementasikan ke dalam bentuk tabel, karena basis data relasional hanya mengenal tabel.

ER adalah salah satu pemodelan basis data konseptual yang menggambarkan basis data ke dalam bentuk entitas-entitas dan relasi yang terjadi di antara entitas-entitas yang ada. Entitas diartikan sebagai 'objek' di dunia nyata yang bisa dibedakan dengan 'objek' yang lain. Relasi diartikan sebagai hubungan yang terjadi di antara satu entitas dengan entitas yang lainnya. Diagram ER merupakan diagram model konseptual untuk menggambarkan struktur *logis* dari basis data berbasis grafis [8].

### 2.8 Kamus Data

Kamus data (*data dictionary*) dipergunakan untuk memperjelas aliran data yang digambarkan pada DFD. Kamus data adalah kumpulan daftar elemen data yang mengalir pada sistem perangkat lunak sehingga masukan (*input*) dan keluaran

(output) dapat dipahami secara umum (memiliki standar cara penulisan) [9].

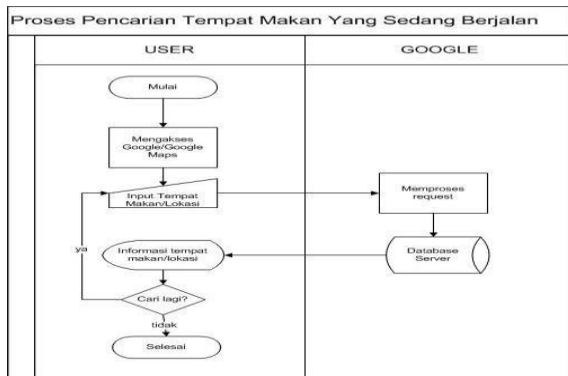
**2.9 Black Box Teasting**

Spesifikasi adalah unit program memenuhi kebutuhan (requirement). Pada *black box testing*, cara pengujian hanya dilakukan dengan menjalankan atau mengeksekusi unit atau modul, kemudian diamati apakah hasil dari unit itu sesuai dengan proses bisnis yang diinginkan. Jika ada unit yang tidak sesuai outputnya maka untuk menyelesaikannya, diteruskan pada pengujian selanjutnya yaitu *white box testing* [1].

**3. Analisis dan Perancangan**

**3.1 Gambaran Sistem Saat Ini**

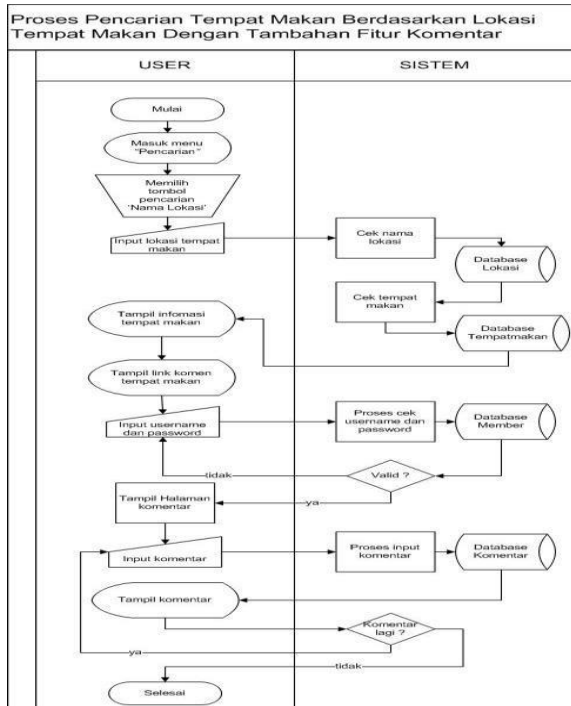
Berikut adalah *flowmap* dari proses yang sedang berjalan untuk proses pencarian tempat makan di Bandung :



**Gambar 1**  
Proses Pencarian Tempat Makan Yang Sedang Berjalan

**3.2 Gambaran Sistem yang Diusulkan**

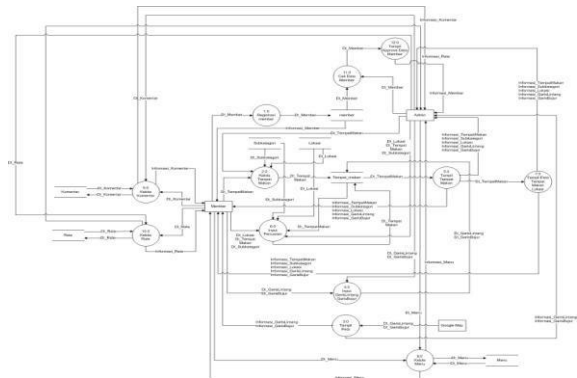
Berikut adalah *flowmap* sistem usulan untuk proses pencarian tempat makan berdasarkan lokasi tempat makan dengan tambahan fitur komentar pada aplikasi yang akan dibangun :



**Gambar 2**  
Flowmap Proses Pencarian Tempat Makan Berdasarkan Lokasi Tempat Makan Dengan Tambahan Fitur Komentar

**3.3 Data Flow Diagram**

Berikut merupakan *data flow diagram* (DFD) level 1, yang merupakan modul-modul dari sistem yang akan dibuat yaitu berupa proses hasil dari *breakdown* diagram konteks. Berikut adalah *data flow diagram* (DFD) level 1 untuk aplikasi yang akan dibuat :



**Gambar 3**  
Data Flow Diagram (DFD) Level 1

Pada DFD Level 1 untuk Aplikasi “Find Food” Di Wilayah Bandung Berbasis APEX Menggunakan *Google Map* digambarkan memiliki 12 proses yaitu :

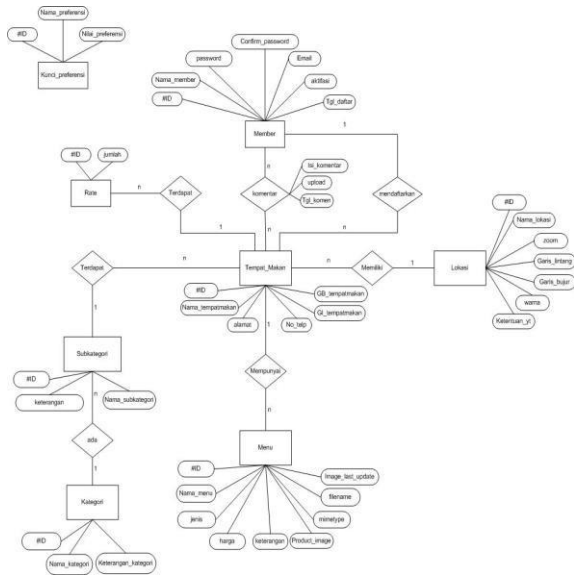
1. Registrasi *Member*. Pada proses ini, *user* yang memiliki tempat makan dapat mendaftarkan tempat makan dengan melakukan pendaftaran menjadi *member* terlebih dahulu. *User* melakukan *input* data yang *valid*, kemudian data *member* disimpan ke dalam *database member*. Kemudian sistem kirim notifikasi kepada *e-mail member* yang mendaftar. Selanjutnya akan dilakukan *approve* menjadi *member* aktif oleh admin.
2. Kelola Tempat Makan. Pada proses ini, *user* yang telah terdaftar menjadi *member* yang dapat melakukan *input* tempat makan. Admin dan *member* dapat mengelola data tempat makan, baik itu untuk keperluan *insert* data baru, *read* data, *update* dan *delete*.
3. Tampil Peta. Pada proses ini, menjelaskan ketika proses *member* melakukan *input* data tempat makan, *member* akan dialihkan ke *website Google Maps*, dimana *google maps* berfungsi untuk menampilkan data garis lintang dan garis bujur atau posisi dimana *user* dapat menentukan posisi tempat makan miliknya. Yang tahapan selanjutnya adalah *user* melakukan salin data garis lintang dan garis bujur secara manual.
4. *Input* Garis Lintang, Garis Bujur. Pada proses ini, menjelaskan proses *member* setelah dialihkan ke *website Google Maps*. Yang selanjutnya *member* menyalin data garis lintang dan garis bujur secara manual ketika *user* melakukan pendaftaran tempat makan dengan mengisi kolom garis lintang dan garis bujur pada aplikasi dengan data garis lintang dan garis bujur yang telah didapatkan dari halaman *google maps*. Hal ini dilakukan untuk *user* dapat menampilkan posisi tempat makan miliknya pada aplikasi “Find Food” sesuai dengan koordinat yang didapatkan dari nilai yang dimiliki garis lintang dan garis bujur di peta *google maps*.
5. Tampil Tempat Makan. Pada proses ini, data tempat makan yang telah di *input* dengan *valid*, sistem simpan data tempat makan ke dalam *database*. Data tempat makan akan ditampilkan menjadi informasi tempat makan kepada *user*. Kemudian, koordinat yang terdapat pada masing-masing tempat makan ditampilkan menjadi *marker* atau *pin* ke dalam peta.
6. *Input* Pencarian. Pada proses ini, *user* melakukan *input* pencarian tempat makan. Aplikasi menampilkan tiga pilihan

penarikan tempat makan, yaitu pencarian berdasarkan nama tempat makan, lokasi tempat makan dan subkategori tempat makan. Setelah *user* melakukan *input* pencarian, aplikasi akan menampilkan informasi tempat makan dan peta lokasi tempat makan sesuai dengan *input* dari *user*.

7. Tampil Peta Tempat Makan Lokasi. Pada proses ini, *user* melakukan *input* pencarian, kemudian aplikasi menampilkan informasi lokasi tempat makan dalam bentuk peta dengan *marker* atau *pin* sebagai posisi tempat makan. *User* dapat memilih lokasi tempat makan berdasarkan daftar lokasi yang telah disediakan aplikasi. Selain itu, pada proses ini juga *member* dapat melihat lokasi tempat makan miliknya.
8. Kelola Menu. Pada proses ini, *user* yang telah mendaftarkan tempat makan yang dapat melakukan input menu. Admin dan *member* dapat mengelola data menu, baik itu untuk keperluan *insert* data baru, *read* data, *update* dan *delete*.
9. Kelola Komentar. Pada proses ini, hanya *member* yang dapat melakukan *input* data komentar. *Member* memilih tempat makan, kemudian *input* data komentar. Sistem selanjutnya menyimpan data komentar ke dalam *database*.
10. Kelola Rate. Pada proses ini, hanya *member* yang dapat melakukan *input* data *rate*. *Member* memilih tempat makan, kemudian *input* data *rate*. Sistem selanjutnya menyimpan data *rating* ke dalam *database*.
11. Cek Data *Member*. Pada proses ini, data *member* akan dilakukan *approve* oleh admin. Data *member* yang telah disimpn di *database*, ditampilkan kepada admin.
12. Tampil *Approve* Data *Member*. Pada proses ini, data *member* yang telah ditampilkan kepada admin, selanjutnya akan dilakukan *approve*, dengan merubah status *member* tidak aktif menjadi *member* aktif oleh admin. Sistem kirim notifikasi berisi data *member* yang terpilih dan dicantumkan *link* untuk melakukan *login* di halaman aplikasi find food pada *e-mail member* tersebut.

**3.4 Entity Relationship Diagram (ERD)**

Berikut ini merupakan *entity relationship* diagram (ERD) dari aplikasi yang dibuat :



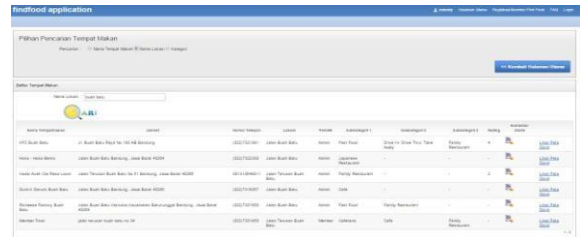
**Gambar 4**  
Entity Relationship Diagram

**4. Implementasi**

Berikut adalah implementasi halaman-halaman yang ada di aplikasi pencarian tempat makan.

**4.1 Implementasi Halaman Pencarian Tempat Makan Berdasarkan Lokasi Tempat Makan Dengan Tambahan Fitur Komentar**

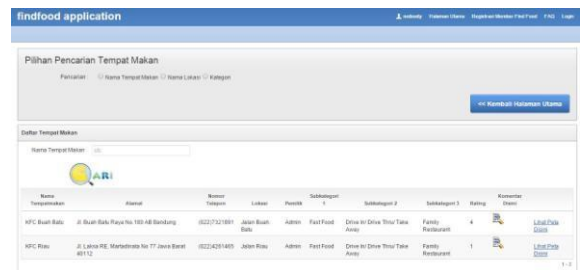
Halaman pencarian tempat makan dengan menggunakan pencarian berdasarkan lokasi tempat makan. *User* melakukan *input* lokasi tempat makan. Jika *valid*, daftar informasi tempat makan pada lokasi tersebut akan muncul dan disediakan tombol untuk melakukan *input* komentar. Komentar dapat dilakukan hanya bagi *user* yang telah menjadi *member*. Berikut adalah tampilan antar muka nya :



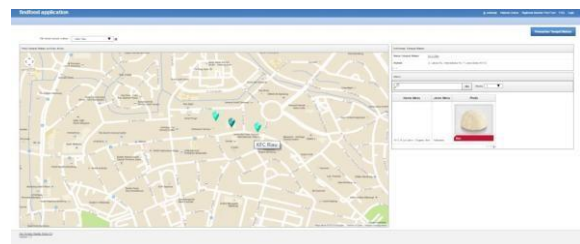
**Gambar 5**  
Implementasi Halaman Pencarian Tempat Makan Berdasarkan Lokasi Tempat Makan Dengan Tambahan Fitur Komentar

**4.2 Implementasi Halaman Informasi Tempat Makan Dengan Peta**

Halaman pencarian disediakan dengan menampilkan pencarian tempat makan yang sebelumnya telah melakukan *input* pencarian berdasarkan nama tempat makan atau lokasi tempat makan atau subkategori tempat makan yang kemudian menampilkan informasi tempat makan. Dapat juga dengan melakukan pilih daftar lokasi yang telah disediakan aplikasi untuk menemukan tempat makan. Setelah *user* memilih lokasi, akan tampil semua daftar tempat makan yang terdapat pada lokasi tersebut. Kemudian *user* memilih dan klik *marker* atau pin pada peta untuk melihat informasi tempat makan tersebut. Menu-menu pada tempat makan akan muncul ketika *user* telah memilih tempat makan. Berikut adalah tampilan antar muka nya :



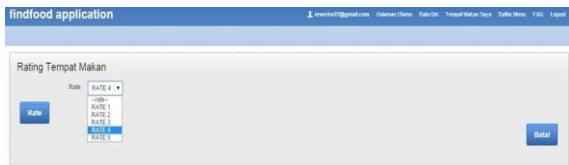
**Gambar 6**  
Implementasi Antar Muka Pencarian Sebelum Ke Halaman Peta



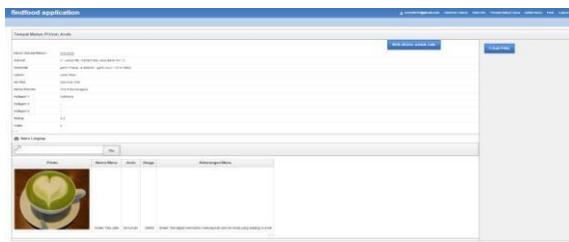
**Gambar 7**  
Implementasi Antar Muka Informasi Tempat Makan Dengan Peta

4.3 Implementasi Halaman Rating Tempat Makan

Halaman yang menyediakan kelola rating tempat makan untuk dilihat oleh user. Halaman ini hanya dapat diakses bagi member. Berikut adalah tampilan antar muka nya :



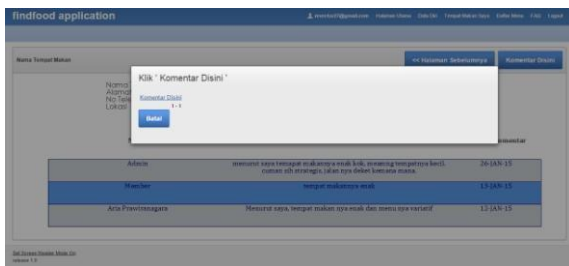
Gambar 8 Implementasi Halaman Input Rating



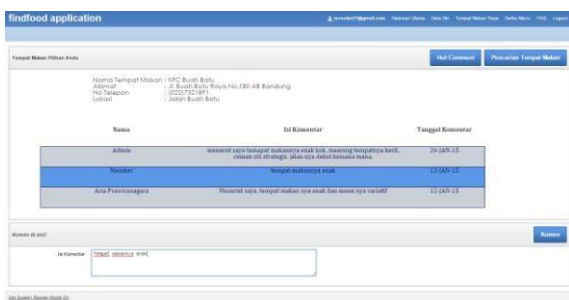
Gambar 9 Implementasi Halaman Tampil Rating

4.4 Implementasi Halaman Kelola Komentar Tempat Makan

Halaman yang menyediakan kelola komentar tempat makan untuk dilihat oleh user. Halaman ini hanya dapat diakses bagi member. Berikut adalah tampilan antar muka nya :



Gambar 10 Implementasi Halaman Link Ke Halaman Tampil Komentar



Gambar 11 Implementasi Halaman Input Komentar



Gambar 12 Implementasi Halaman Tampil Komentar

5. Pengujian

Pengujian yang dilakukan penulis ini menggunakan metode pengujian black box. Pengujian ini dilakukan dengan cara menguji aplikasi dari segi fungsionalitasnya, yaitu memberikan inputan berdasarkan suatu kondisi, lalu mengamati apakah keluaran yang dihasilkan sesuai dengan keluaran yang diharapkan sebelumnya dan memberikan kesimpulan dari hasil pengujian tersebut. Pada aplikasi ini, penulis melakukan pengujian terhadap 13 fungsionalitas yang tersedia. Adapun fungsionalitas yang diuji adalah proses login, registrasi member, pencarian tempat makan, tambah data tempat makan, ubah dan hapus tempat makan, tambah data menu tempat makan, ubah dan hapus menu tempat makan, ubah data diri member, tambah komentar tempat makan, tambah rating tempat makan, dan aktivasi member oleh Admin. Dari 13 fungsionalitas yang diuji, keluaran yang diharapkan dan keluaran yang dihasilkan dari masing-masing fungsionalitas sudah mencapai 100% sama (sudah sesuai).

6. Kesimpulan dan Saran

6.1 Kesimpulan

Berdasarkan analisis dan pembuatan aplikasi ini, maka dapat disimpulkan bahwa hasil analisis dan pembuatan aplikasi ini adalah :

- Berhasil dibuat aplikasi yang dapat menerapkan fitur yang dimiliki Google Maps ke dalam Application Express (APEX). Fitur yang dimiliki Google Maps dan digunakan pada aplikasi ini adalah Static Maps API dan Google Maps JavaScript API V3.
- Berhasil dibuat aplikasi yang menjadi alternatif bagi user dalam menemukan tempat makan di Bandung dengan menampilkan peta sebagai penunjuk lokasi yang lebih jelas dan informasi tempat makan yaitu alamat, nomor telepon, subkategori tempat makan, juga menu yang dimiliki tempat makan.
- Berhasil dibuat aplikasi yang dapat menyediakan fitur pencarian berdasarkan lokasi, nama tempat makan dan subkategori tempat makan untuk memudahkan pengguna dalam mendapatkan informasi tempat makan di Bandung.
- Berhasil dibuat aplikasi yang dapat menyediakan fitur input tempat makan bagi member, dimana tempat makan milik member dapat tampil di peta tempat makan Bandung. Fungsi input hanya dapat dilakukan oleh user yang telah mendaftar menjadi member.
- Berhasil dibuat aplikasi yang dapat menyediakan fitur kelola komentar tempat makan bagi member. Kelola komentar adalah fungsi input komentar, hanya dapat dilakukan oleh user yang telah mendaftar menjadi member.
- Berhasil dibuat aplikasi yang dapat menyediakan fitur input rating tempat makan bagi member. Fungsi input hanya dapat dilakukan oleh user yang telah mendaftar menjadi member.

6.2 Saran

Berdasarkan hasil pengerjaan yang diperoleh maka penulis mempunyai saran-saran untuk meningkatkan kinerja aplikasi dan pengembang aplikasi sebagai berikut:

- Diharapkan pada aplikasi ini nantinya dapat ditambahkan fitur untuk menampilkan petunjuk arah yaitu rute (direction map).
- Diharapkan pada aplikasi ini nantinya dapat ditampilkan ke dalam bentuk mobile Application.



**Daftar Pustaka**

- [1] A. Fatta. *Analisis dan Perancangan Sistem Informasi untuk Keunggulan Bersaing Perusahaan dan Organisasi Modern*. Yogyakarta : ANDI, 2007.
- [2] *Developers Google*.(2014) *Developers Google* [Online].  
HYPERLINK  
"<https://developers.google.com/maps/documentation/static-maps/>".
- [3] *Developers Google*.(2014) *Developers Google* [Online].  
HYPERLINK  
"<https://developers.google.com/maps/documentation/javascript/reference/>".
- [4] Febrian, J. *Kamus Komputer dan Teknologi Informasi*. Bandung : Informatika, 2007.
- [5] G. Rick, *Begginning Oracle Application Express*. Canada: Wiley Publishing, 2009.
- [6] *Google Maps*.(2014) *Google Maps* [Online].  
HYPERLINK  
"<https://www.google.com/intx/id/work/mapsearth/products/mapsapi.html>".
- [7] P. S.Roger, *Rekayasa Piranti Lunak Pendekatan Praktisi*. Yogyakarta: Andi, 1997.
- [8] Rahman, Dedy. *Perancangan Basis Data Relasional*. Bandung : Politeknik Telkom, 2009.
- [9] Rosa and Shalahudin, M.*Rekayasa Perangkat Lunak*. Bandung : Modula, 2011.
- [10] Soeherman, Bonie and Pinontoan, Marion. *Designing Information System*. Jakarta : PT Alex Media Komputindo, 2008.
- [11] Westriningsih , Ed., *Mudah & Cepat Membuat Website dengan CodeIgniter*. Yogyakarta: Andi, 2011.

