

BAB III

DASAR TEORI

Pada Bab III akan membahas tentang Dasar Teori yang membahas untuk aplikasi *mobile* informasi lokasi memancing ikan *hampala* di Yogyakarta dalam kebutuhan tugas akhir.

3.1 Ikan *Hampala Macrolepidota*



Gambar 3.1 Ikan *Hampala Macrolepidota* (dokumentasi pribadi)

Ikan *Hampala* adalah sejenis ikan air tawar anggota suku *Cyprinidae* (kerabat ikan mas). Ikan ini menyebar luas di Asia Tenggara daratan dan Indonesia bagian barat. Ikan *Hampala* memiliki 2 jenis yang sering di dapat yaitu jenis yang mempunyai 1 *dots* (totol) dan yang memiliki 2 *dots* (totol) yang tersebar hampir di seluruh pulau di Indonesia. Di Yogyakarta ikan ini menjadi salah satu jenis yang paling di buru oleh para pemancing, mulai dari sebagai ikan konsumsi, ataupun ikan yang hanya di pancing dan di lepaskan kembali atau istilah yang sudah menyebar adalah *Catch and Release* .

Bukan hanya di Yogyakarta saja ikan ini di buru, melainkan hampir di seluruh wilayah Indonesia yang terdapat populasi ikan *hampala* dapat dilihat pada Gambar 3.1.

3.2 Memancing ikan dengan teknik *casting*



Gambar 3.2 Memancing dengan teknik *casting*
Macrolpidota(dokumentasi pribadi)

Memancing adalah suatu hobi yang telah lama berkembang di dunia, terkhususnya di Indonesia. Semakin banyak tayangan memancing di televisi yang mengenalkan cara memancing dengan umpan buatan "*lure*", yang memicu di Indonesia semakin berkembang dengan pesat teknik memancing ikan predator dengan teknik *Casting*. Teknik *Casting* adalah cara memancing dengan memanfaatkan umpan buatan yang biasa di sebut dengan "*lure*", dan cara memainkan "*lure*" adalah melempar umpan ke titik yang menjadi lokasi ikan berkumpul, lalu umpan tersebut di tarik kembali dengan gerakan umpan yang menyerupai ikan yang biasanya di makan oleh ikan predator yang menjadi target *casting* dapat dilihat pada Gambar 3.2.

3.3 Aplikasi Mobile

Saat ini telah ada tiga jenis Aplikasi *mobile* yang sudah berkembang yaitu :

a) *Native*



Gambar 3.3 Aplikasi *Native* (Codepolitan, 2018)

Aplikasi *Native* adalah aplikasi yang dibangun dengan bahasa pemrograman yang spesifik untuk platform tertentu. Contoh yang populer, penggunaan bahasa pemrograman *Objective-C* atau *Swift* untuk platform *iOS* (Apple), sedangkan untuk platform *Android* menggunakan bahasa pemrograman *Java*. Gambar 3.3 sebagai ilustrasi (Codepolitan, 2018). Tiga kelebihan dan kekurangan Aplikasi *Native* adalah sebagai berikut:

Kelebihan Aplikasi *Native*

1. Memiliki *user experience* yang baik, dan menghasilkan antarmuka look dan feel yang alami dengan baik.

2. Kualitas dan keamanannya terjamin karena di kontrol oleh *vendor* masing-masing.

3. Memiliki akses penuh ke perangkat *mobile*.

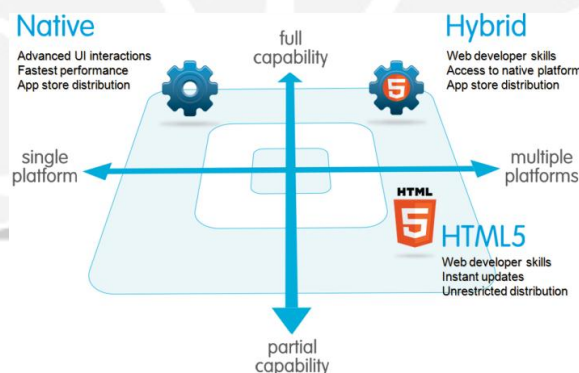
Kekurangan

1. Biaya pengembangan dan *maintenance* yang lebih mahal.

2. Pengembangan yang tidak mudah karena menggunakan Bahasa APis (*Application programming interface spesifik*).

3. Aplikasi hanya bekerja pada *platform* yang sudah dispesifikasikan di awal pengembangan.

b) *Hybird*



Gambar 3.4 Aplikasi *Hybird*(Codepolitan,2018)

Aplikasi *Hybird* adalah aplikasi yang ditransformasikan menjadi *code native* pada *platform* seperti *iOS* atau *Android*. Aplikasi *hybird* biasanya menggunakan *browser* untuk mengijinkan aplikasi *web* mengakses berbagai *fitur* di *device mobile* seperti *Push Notification*, *Contacts*, atau *offline data Storage* Gambar 3.4 sebagai ilustrasi(Codepolitan,2018). Tiga kelebihan dan kekurangan dari Aplikasi adalah sebagai berikut:

Kelebihan Aplikasi Hybrid

1. Memungkinkan pembuatan aplikasi secara *online* maupun *offline*.
2. Proses pengembangannya jauh lebih mudah.
3. Memiliki dukungan *multi-platform* yang tidak dimiliki *native* namun biayanya lebih mahal dibandingkan solusi *mobile web*.

Kekurangan Aplikasi Hybrid :

1. Kurang dukungan secara penuh untuk mengakses fitur asli bawaan dari *device*.
2. Performa aplikasi *hybird* masih belum bisa menyamai aplikasi *native*.
3. Memerlukan software bantu framework pengembangan aplikasi *mobile* berbasis web yang stabil dan mendukung lintas *platform*.

c) Web



Gambar 3.5 Aplikasi Web (Codepolitan, 2018)

Aplikasi web bisa menjadi pilihan yang terbaik untuk membangun produk yang diinginkan. Minimnya persyaratan dengan kata lain akses ke fitur dari perangkat *mobile* seperti *Push Notification* tidak diperlukan. Sayangnya meskipun menjadi pilihan murah dari sisi *budget*, aplikasi web tidak bisa didistribusikan melalui toko aplikasi *native* seperti *App Store* atau *Google Play* 3.5 sebagai ilustrasi (Codepolitan, 2018). Tiga kelebihan dan kekurangan Aplikasi Web adalah sebagai berikut:

Kelebihan Aplikasi Web:

1. Tidak perlu mempelajari bahasa baru karena menggunakan bahasa yang sudah familiar yaitu HTML5, CSS3, dan JavaScript.
2. Tahap pengembangannya sangat mudah karena menggunakan teknologi-teknologi web yang sudah ada.
3. Dapat berjalan baik di semua *browser* modern pada *platform mobile*

Kekurangan Aplikasi Web:

1. Sesuai karakteristiknya, aplikasi web *mobile* hanya tersedia secara *online*.
2. Harus mempunyai 2 buah *domain*.
3. Performanya kurang stabil dan bergantung pada koneksi yang ada

3.4 LBS (*Location Based Service*)

Location Based Service atau lebih dikenal dengan layanan berbasis lokasi merupakan suatu teknologi yang digunakan untuk menemukan suatu lokasi yang berbasis *map* pada perangkat yang digunakan. Perangkat tersebut biasa berupa PDA, atau *Smartphone* yang telah memiliki sistem GPS didalamnya. Sekarang ini, pengembangan LBS lebih banyak ke *smartphone*. Ini dapat dilihat dengan berkembangnya aplikasi-aplikasi yang berbasis LBS pada perangkat *mobile smartphone* (Neven Vrcek, Goran Bubas, and Neven Bosilj. 2009) .

3.5 GPS (*Global Positioning System*)

GPS adalah sebuah sistem telekomunikasi terbuka, tidak ada pemilikan (*non-proprietary*) melainkan kepemilikan hak cipta suatu perusahaan yang berkembang secara pesat dan konstan (Sunomo, 2004). Bagian utama dari sistem *GPS* adalah 24 satelit yang mengorbit Bumi di ketinggian 20.200 kilometer. *Orbit* satelit dirancang sehingga setiap titik di bumi dapat melihat paling sedikit empat satelit pada setiap saat Akurasi atau ketepatan perlu mendapat perhatian bagi penentuan koordinat sebuah titik/lokasi. Koordinat posisi ini akan selalu mempunyai 'faktor kesalahan', yang lebih dikenal dengan 'tingkat akurasi'. Misalnya, alat tersebut menunjukkan sebuah titik koordinat dengan akurasi 3 meter, artinya posisi sebenarnya bisa berada dimana saja dalam *radius* 3 meter dari titik koordinat (lokasi) tersebut. Makin kecil angka akurasi (artinya akurasi makin tinggi), maka posisi alat akan menjadi

semakin tepat. Harga alat juga akan meningkat seiring dengan kenaikan tingkat akurasi yang bisa dicapainya.

Pada pemakaian sehari-hari, tingkat akurasi ini lebih sering dipengaruhi oleh faktor sekeliling yang mengurangi kekuatan sinyal satelit. Karena sinyal satelit tidak dapat menembus benda padat dengan baik, maka ketika menggunakan alat, penting sekali untuk memperhatikan luas langit yang dapat dilihat.

Ketika alat berada disebuah lembah yang dalam (misal, akurasi 15 meter), maka tingkat akurasinya akan jauh lebih rendah daripada di padang rumput (misal, akurasi 3 meter). Di padang rumput atau puncak gunung, jumlah satelit yang dapat dijangkau oleh alat akan jauh lebih banyak daripada dari sebuah lembah gunung. Jadi, jangan berharap dapat menggunakan alat navigasi ini di dalam sebuah gua. Karena alat navigasi ini bergantung penuh pada satelit, maka sinyal satelit menjadi sangat penting. Alat navigasi berbasis satelit ini tidak dapat bekerja maksimal ketika ada gangguan pada sinyal satelit.

Setiap daerah di atas permukaan bumi ini minimal terjangkau oleh 3-4 satelit. Pada prakteknya, setiap GPS terbaru bisa menerima sampai dengan 12 *channel* satelit sekaligus. Kondisi langit yang cerah dan bebas dari halangan membuat *GPS* dapat dengan mudah menangkap sinyal yang dikirimkan oleh satelit. Semakin banyak satelit yang diterima oleh *GPS*, maka akurasi yang

diberikan juga akan semakin tinggi. Cara kerja *GPS* secara logik ada 5 langkah:

1. Memakai perhitungan "*triangulation*" dari satelit.
2. Untuk perhitungan "*triangulation*", *GPS* mengukur jarak menggunakan *travel time* sinyal radio.
3. Untuk mengukur *travel time*, *GPS* memerlukan akurasi waktu yang tinggi.
4. Untuk perhitungan jarak, kita harus tahu dengan pasti posisi satelit dan ketinggian pada orbitnya.
5. Terakhir harus mengoreksi *delay* sinyal waktu perjalanan di atmosfer sampai diterima *reciever*.

Satelit *GPS* berputar mengelilingi bumi selama 12 jam di dalam orbit yang akurat dia dan mengirimkan sinyal informasi ke bumi. *GPS reciever* mengambil informasi itu dan dengan menggunakan perhitungan "*triangulation*" menghitung lokasi *user* dengan tepat. *GPS reciever* membandingkan waktu sinyal di kirim dengan waktu sinyal tersebut di terima. Dari informasi itu didapat diketahui berapa jarak satelit. Dengan perhitungan jarak jarak *GPS reciever* dapat melakukan perhitungan dan menentukan posisi *user* dan menampilkan dalam peta elektronik.

3.6 Google Maps API

Google Maps API adalah *API* yang paling populer di internet. Pencatatan yang dilakukan pada bulan Mei tahun 2010 ini menyatakan bahwa 43% *mashup* (aplikasi dan situs web yang menggabungkan dua atau lebih sumber data) menggunakan *Google Maps API*. Beberapa tujuan dari penggunaan *Google Maps API* adalah untuk melihat lokasi, mencari alamat, mendapatkan petunjuk mengemudi dan lain sebagainya Iswara Halim. (2011) .

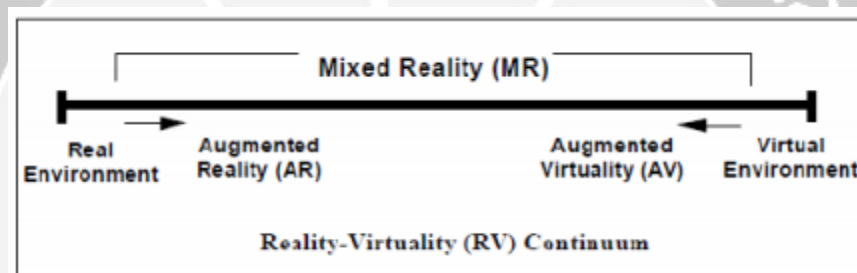
3.7 Augmented Reality

Augmented Reality merupakan upaya penggabungan dunia nyata ke dunia *virtual* melalui komputer sehingga batas antara keduanya sangat tipis. *Augmented Reality* (AR) adalah variasi dari *Virtual Environment* (VE) atau yang lebih dikenal dengan *Virtual Reality* (VR) . Sedangkan *virtual reality* memiliki arti sebuah situasi dimana pengguna secara keseluruhan berada di dalam lingkungan maya. Ketika berada di lingkungan itu pengguna sendiri tidak dapat melihat dunia nyata disekitarnya. Berbeda dengan AR yang masih dapat melihat dunia nyata dan objek maya hanya ditampilkan ke lingkungan nyata Azuma, Ronald T. (1997) .

Berdasarkan cara kerjanya AR dibagi menjadi 4 (Milgram et al, 1994), yaitu sebagai berikut:

1. Simple AR, objek dua dimensi atau tiga dimensi diciptakan oleh komputer dan diintegrasikan dengan objek atau lingkungan nyata atau fisik secara langsung maupun tidak langsung kemudian ditampilkan dalam layar. Simple AR ini adalah penerapan teknologi *Augmented reality* yang

paling sederhana dan paling awal, meskipun begitu masih digunakan secara luas hingga saat ini, salah satunya dalam acara olahraga di televisi. Sebagai contohnya pada olahraga sepakbola, dimana dalam lapangan terdapat garis yang membantu penonton untuk melihat ketika pemain melakukan kesalahan pada permainan, seperti *offside* atau yang lainnya yang ditayangkan di televisi. Dapat dilihat pada Gambar 3.6 diagram ilustrasi dari simple *Augmented Reality*.

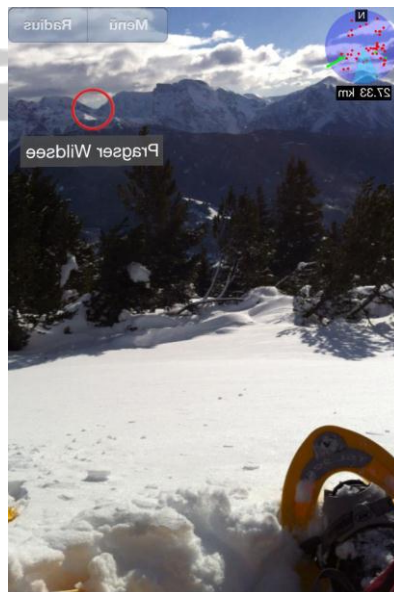


Gambar 3.6 Diagram Ilustrasi Simple AR (Milgram, 1994)

2. *Marker Based AR*, objek dua dimensi, objek tiga dimensi, teks, video maupun suara diproses menggunakan komputer dan *webcam* dan ditampilkan dalam layar maupun peralatan *display* khusus melalui pengenalan sebuah penanda, setelah penanda dikenali oleh komputer kemudian objek *virtual* yang sudah terdapat dalam *library* komputer ditampilkan diatas marker tersebut. Ada dua jenis marker yang digunakan saat ini yaitu QR (*Quick Response*) code dan *semacode*, adalah sejenis *barcode* dua dimensi yang memungkinkan kontennya untuk diterjemahkan dengan kecepatan tinggi. Sedangkan *semacode* adalah *barcode* yang

berjenis data *matriks*. *Marker based* AR telah menjadi tren sejak awal tahun 2010 hingga sekarang dan diimplementasikan pada berbagai media khususnya media cetak.

3. *Markerless* AR, umumnya diterapkan dalam *mobile device* seperti *smartphone*. Sesuai dengan namanya *markerless* AR tidak membutuhkan marker yang terlihat secara fisik untuk mengetahui posisi suatu objek. Sebagai gantinya, digunakan informasi dari kamera digital atau sensor optik lainnya, *accelerometers*, *GPS*, *gyroscopes*, kompas, *RFID*, sensor *wireless* dimana cara ini dikenal dengan nama *geotagging* dan *geolocation*. Melalui *geotagging* dan *geolocation* yang bias kita sebut sebagai marker yang tidak terlihat inilah konten seperti tulisan, video, maupun audio kemudian ditampilkan dilayar perangkat *mobile* tersebut. Seperti pada gambar 3.7 *Markerless Augmented Reality*.

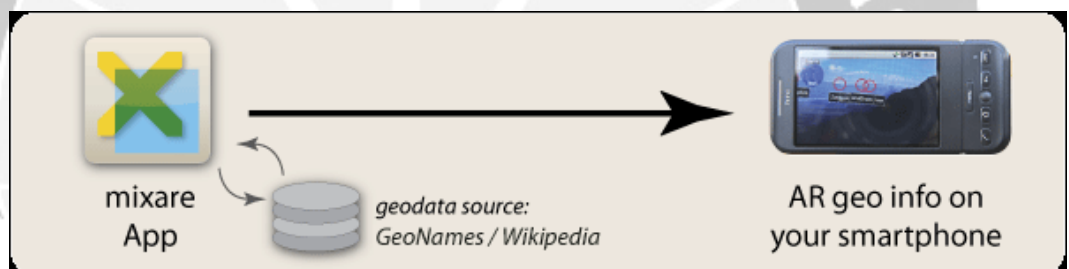


Gambar 3.7 *Markerless* AR (Mixare, 2018)

3.8 Mixare

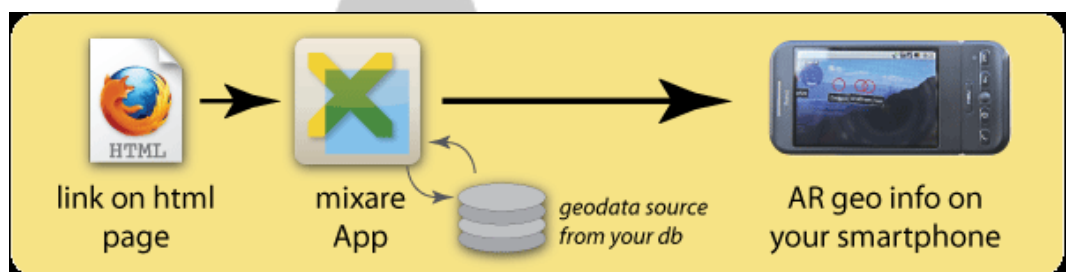
Mixare (Mix Augmented Reality Engine) adalah *open source augmented reality browser* yang diterbitkan dibawah GPLv3. *Mixare* tersedia untuk *Android* dan *iPhone 3GS* dan di atasnya. *Mixare* bekerja sebagai aplikasi yang sangat otonom dan tersedia untuk pengembangan dari implementasi sendiri. Ada beberapa cara dalam menggunakan *mixare*.

a) *Mixare* adalah sebuah aplikasi otonom yang menampilkan POI Wikipedia terhadap lingkungannya. *Mixare* dijalankan di perangkat *mobile* secara langsung. Seperti pada gambar 3.8 *Mixare* dengan Wikipedia sebagai database servernya.



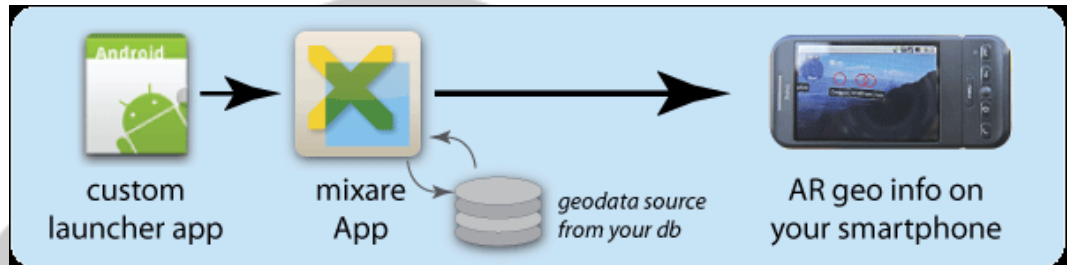
Gambar 3.8 *Mixare* dengan Wikipedia sebagai database servernya (Mixare,2018)

b) *Mixare* dapat diakses oleh link pada situs HTML, dimana sumber data ditransfer ke aplikasi. Seperti pada gambar 3.9 untuk gambaran ilustrasinya.



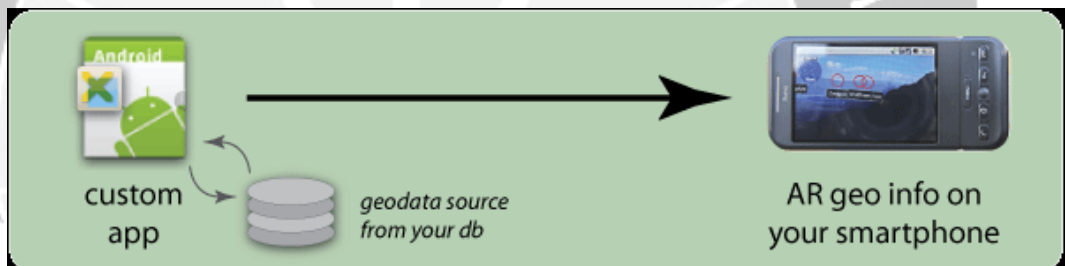
Gambar 3.9 *Mixare* diakses melalui sebuah link pada situs HTML (Mixare,2018)

c) *Mixare* dapat diakses oleh launcher-app kita sendiri, data source ditransfer ke aplikasi yang kita buat. Seperti pada gambar 3.10 untuk gambaran ilustrasinya.



Gambar 3.10 *Mixare* dapat diintegrasikan dengan aplikasi yang telah dibangun sendiri (Mixare,2018)

d) *Mixare* secara bebas di upgrade dan bahkan dapat dimodifikasi ke aplikasi individu. Seperti pada gambar 3.11 untuk gambaran ilustrasinya.



Gambar 3.11 *Mixare* dapat dikembangkan dan diubah menjadi aplikasi individu (Mixare,2018)

Framework untuk *Augmented reality* pada *Android* yang berbasis *geolocation* ini dikembangkan oleh Justin Wetherell. *Framework* ini bersifat *open source* yang dapat di download pada <https://code.google.com/p/android-augment-realityframework/>. *Framework* by Justin Wetherell ini dikembangkan dari *framework* dasar *Mixare* dengan bahasa *Java* sehingga cocok untuk membuat aplikasi *mobile* yang akan dijalankan di *smartphone* bersistem operasi *android*. *Framework* ini nantinya akan dijadikan sebagai *library*

yang dapat digunakan untuk membuat aplikasi *augmented reality* dengan citra virtual pada dunia nyata.

3.9 Web Service

Menurut Nuari (2014:2), *web service* adalah suatu sistem perangkat lunak yang dirancang untuk mendukung interoperabilitas dan interaksi antar sistem pada suatu jaringan. *Web service* bertujuan untuk meningkatkan kolaborasi antar pemograman dan perusahaan, yang memungkinkan sebuah fungsi di dalam *web service* dapat dipinjam oleh aplikasi lain tanpa perlu mengetahui detail pemograman yang ada di dalamnya.

Web Service merupakan suatu perangkat lunak atau aplikasi yang menyediakan layanan (*service*) bagi aplikasi-aplikasi lainnya atau aplikasi pengguna (*client*). *Web service* memungkinkan dikembangkan suatu aplikasi yang memanfaatkan layanan (aplikasi pengguna) yang disediakan oleh penyedia layanan (*server*), tanpa harus melihat pada platform perangkat keras dan sistem operasi yang digunakan. Baik server maupun pengguna dapat berkomunikasi dengan menggunakan protokol HTTP (*HyperText Transport Protocol*) (Nugroho & Mustofa, 2012). *Web Service* dapat menampilkan data informasi dalam *format* JSON atau XML, sehingga data ini dapat diakses oleh sistem lain walaupun berbeda *platform*, sistem operasi, dan bahasa pemrograman (Rahman, et al., 2013).

REST (*Representational State Transfer*) merupakan salah satu jenis arsitektur penerapan *web service* yang menerapkan konsep perpindahan antar state (Rahman, et al., 2013). State disini dapat digambarkan seperti

mengakses sebuah halaman situs. Perintah HTTP yang bisa digunakan dalam REST adalah fungsi GET, POST, PUT atau DELETE.

Retrofit memungkinkan tidak perlu lagi untuk membuat method-method sendiri untuk menggunakan *REST Client API* dari *backend*. Library ini menyediakan framework yang *powerfull* untuk authenticating dan berinteraksi dengan API dengan mengirimkan *request* menggunakan HTTP. *Retrofit* memarsing JSON response dari *Web API* menjadi *Plain Old Java Object* (POJO) sehingga akan memudahkan menggunakan *value-value* yang berada dalam response karena sudah otomatis terkonversi menjadi *object* (Yusuf, 2016).

3.10 MySQL (*My Structured Query Language*)

Menurut Nugroho (2004:98), MySQL (*My Structured Query Language*) adalah sebuah program pembuat dan pengelola database atau yang sering disebut dengan DBMS (Database Management System), sifat dari DBMS ini adalah open source. MySQL sebenarnya produk yang berjalan pada platform Linux, dengan adanya perkembangan dan banyaknya pengguna, serta lisensi dari database ini adalah Open Source, maka para pengembang merilis versi *Windows*.

MySQL merupakan program database yang mengakses datanya bersifat jaringan, sehingga dapat digunakan untuk aplikasi *Multi User* (banyak pengguna). Kelebihan lain dari MySQL adalah menggunakan bahasa *query* (permintaan) standar SQL (*Structured Query Language*). SQL adalah suatu bahasa permintaan yang terstruktur, SQL telah distandarkan untuk semua program pengakses

database seperti Oracle, PostgreSQL, SQL Server dan lain-lain.

3.11 JSON (*Java Script Object Notation*)

JSON adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh computer Taufik. (2012).

JSON (*JavaScript Object Notation*) adalah format pertukaran data ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat oleh komputer (Anonim, 2013). JSON merupakan format teks yang tidak bergantung pada bahasa pemrograman apapun karena menggunakan gaya bahasa yang umum, oleh karena itu JSON merupakan format yang sangat ideal sebagai bahasa pertukaran data. JSON terbuat dari dua struktur yaitu kumpulan pasangan nama atau nilai dan daftar nilaiurut atau array.

Sekian dari penjelasan BAB III tentang Dasar teori yang dipaparkan dari ikan *Hampala*, memancing ikan dengan teknik *casting*, Aplikasi *Mobile*, LBS(*Location Base Service*), GPS, Google Maps API, *Augmented Reality*, Mixare sebagai framework dari *Augmented Reality* MySQL (*My Structured Query Language*), *web service*, JSON(*Java Script Object Notation*). Selanjutnya pada BAB IV akan membahas tentang analisis dan perancangan sistem.