BAB II

LANDASAN TEORI

2.1 Teori Pengumpulan Data

2.1.1 *Web Crawler*

Web Crawler atau yang di sebut *spider* adalah robot internet yang menjelajahi internet melalui *World Wide Web* (WWW). Biasanya bertujuan untuk pengindeksan halaman website. Artinya, *Web Crawler* secara otomatis menemukan dan mengumpulkan sumber daya yang berbeda secara tertib dari internet sesuai dengan kebutuhan pengguna. (Singh & Singh, 2014)

Peneliti menerapkan metode *web crawler* ini untuk mendapatkan informasi salon yang terdapat di daerah Banten dengan cara mendapatkan tag HTML dari website salon tersebut. Setelah nantinya tag HTML dari *website* didapatkan, akan ditentukan data yang ingin diambil dari *website* tersebut. Setelah mendapatkan tag dari data yang ingin didapatkan dari suatu *website* selanjutnya akan dilanjutkan dengan melakukan metode *web scraping*.

2.1.2 *Web Scraping*

Web scraping (*web harvesting* atau *web data extraction*) adalah data *scraping* yang digunakan untuk mengekstrak data dari situs web. *Web scraping software* dapat mengakses *World Wide Web* (WWW) secara langsung dengan menggunakan *Hypertext Transfer Protocol* (HTTP) , atau melalui *web browser*.

Ini adalah bentuk penyalinan, di mana data spesifik dikumpulkan dan disalin dari suatu web, dan disimpan ke database lokal atau spreadsheet pusat, yang nanti nya dapat kita gunakan lagi di proses selanjutnya. (Renita & Vanita, 2015)

Pendapat lain menyatakan bahwa *Web scraping* adalah sebuah proses yang memanfaatkan dokumen berbentuk *semi – structured* yang didapatkan dari internet, yang dimana dokumen tersebut berbentuk sebuah halaman website yang dibangun oleh bahasa markup seperti HTML ataupun XHTML yang kemudian dianalisis untuk mendapatkan informasi yang berguna yang dapat dilakukan untuk konteks lain. (Nikhit, 2015)

Implementasi *web scraping* dapat diterapkan untuk mendapatkan informasi tertentu dari sebuah *website* yang nantinya akan diolah untuk dapat ditampilkan ke dalam bentuk lain yang menghasilkan informasi yang berguna untuk para penggunanya. Pada perancangan aplikasi yang dilakukan peneliti ini *web scraping* yang dilakukan adalah dengan mengambil data dari salon-salon di daerah Banten yang memiliki *website*. Setelah berhasil memperoleh data dari *website* tersebut, aplikasi yang dibuat akan menampilkan lokasi serta jasa yang disediakan sehingga output yang di hasilnyakan nantinya adalah aplikasi untuk merekomendasikan salon yang sesuai dengan lokasi dan jasa yang user inginkan.

2.2 Teori Pengolahan Data

2.2.1 *Data Mining*

Data Mining adalah ekstraksi pola yang menarik dari data dalam jumlah besar . Suatu pola dikatakan menarik apabila pola tersebut tidak sepele, implisit, tidak diketahui sebelumnya, dan berguna. Pola yang disajikan haruslah mudah dipahami, berlaku untuk data yang akan diprediksi dengan derajat kepastian tertentu, berguna, dan baru. Penggalian data memiliki beberapa nama alternatif, meskipun definisi eksaknya berbeda, seperti KDD (*knowledge discovery in database*), analisis pola, arkeologi data, pemanenan informasi, dan intelegensia bisnis. Penggalian data diperlukan saat data yang tersedia terlalu banyak (misalnya data yang diperoleh dari sistem basis data perusahaan, *e-commerce*, data saham, dan data bioinformatika), tetapi tidak tahu pola apa yang bisa didapatkan. (Nawal, Rasha, & Somia, 2016)

Data mining yang dilakukan oleh peneliti dalam melakukan perancangan aplikasi adalah lokasi salon serta jasa salon dengan menggali informasi dari sejumlah sumber data. Sehingga nanti nya dalam aplikasi yang akan dibuat, user dapat mencari salon yang mereka inginkan sesuai dengan lokasi dan jasa yang tersedia.

Untuk mendapatkan hasil informasi sesuai dengan yang dibutuhkan, sebelumya kita perlu melakukan preprocessing data. Di dalam preprocessing data terdapat 2 aktifitas inti yang dilakukan yaitu *data cleaning* dan *data transformation*. Terkadang dalam aktifitas pengambilan data terdapat data yang

nilainya tidak konsisten, terdapat redundansi data, ataupun data yang informasinya tidak relevan.

2.2.2 *Data Cleaning dan Data Transformation*

Data cleaning, disebut juga *cleaning* atau *scrubbing* adalah mendeteksi data dan menghapus kesahalan dan ketidak konsistenan data agar dapat meningkatkan kualitas data (Kobus, Eeckels, & Argeseanu, 2010). Dalam proses pengambilan data terdapat data yang memberikan informasi yang tidak relevan, nilai dari data yang tidak konsisten, dan redudansi data. Maka dari itu untuk dapat mendapatkan data dengan kualitas yang valid dan akurat perlu dilakukan data cleaning sebelum melakukan penarikan data.

Setelah data yang diambil telah selesai melalui tahap *data cleaning*, untuk memberikan informasi data yang valid diperlukan adanya penentuan format untuk data tertentu agar nilai yang dihasilkan konsisten. Proses penentuan format untuk setiap data inilah yang disebut dengan *tahap Data Transfromation* (M. A., Effenberger, Becker, & Adikaram, 2015). Output yang dihasilkan dari tahapan *data transformation* inilah yang dapat dijadikan sumber data oleh aplikasi yang dirancang oleh peneliti.

2.3 Teori *Mobile Based Application*

2.3.1 Pengertian *Mobile Application*

Aplikasi *mobile* adalah perangkat lunak aplikasi yang dirancang untuk berjalan pada perangkat mobile seperti smartphone dan komputer tablet. Sebagian besar perangkat tersebut dijual dengan beberapa aplikasi yang dibundel sebagai

perangkat lunak pra-instal, seperti *browser web*, klien email, kalender, program pemetaan, dan aplikasi untuk membeli musik atau media lainnya atau lebih banyak aplikasi. (Rakestaw, Eunni, & Kasuganti, 2012)

Aplikasi yang di rancang oleh peneliti juga berbasis *mobile* , sehingga pengguna dapat mengaksesnya dimana saja dan kapan saja hanya lewat *handphone* mereka.

2.3.2 *Database*

Database merupakan kumpulan dari data-data yang mempunyai relasi logis dan penjelasan tentang data yang saling terhubung tersebut dan perancangan data sampai sedemikian rupa sehingga dapat dianalisa untuk menghasilkan informasi yang berguna (Connolly, 2010).

Pada perancangan aplikasi yang dilakukan *database* yang dipilih adalah MongoDB. Salah satu alasan peneliti memili MongoDB untuk *database* yang dipakai dalam perancangan aplikasi adalah MongoDB menggunakan konsep NOSQL. NOSQL merupakan konsep penyimpanan *database* dinamis yang tidak terikat pada relasi-relasi tabel yang kaku seperti RDBMS. Selain lebih *scalable*, NoSQL juga memiliki performa pengaksesan yang lebih cepat.

2.3.3 *Android Studio*

Android Studio adalah sebuah IDE untuk pengembangan aplikasi di platform Android. Sama seperti kombinasi antara Eclipse dan Android *Developer Tools* (ADT). Android Studio merupakan pengembangkan dari Eclipse

IDE, dan dibuat berdasarkan IDE Java populer, yaitu IntelliJ IDEA. Android Studio direncanakan untuk menggantikan Eclipse ke depannya sebagai IDE resmi untuk pengembangan aplikasi Android. (Rajput, 2015)

Software ini dipakai oleh peneliti dalam tahap *coding* pada saat perancangan aplikasi yang dilakukan dimana aplikasi yang dibuat adalah aplikasi berbasis *mobile*.

Gambar 2.1 Android Studio Logo

Sumber: www.developer.android.com

2.4 Tools Yang Digunakan

2.4.1 MongoDB

MongoDB adalah *database* berorientasi dokumen (*Document-Oriented Database*). Jika pada RDBMS kita mengenal tabel, maka di mongoDB ada istilah

collection dan *Document* di dalam *collection* ada bentuk dokumen-dokumen yang sepadan dengan baris pada tabel. Dan nama-nama kolom pada tabel bisa dijumpai sebagai *field* di MongoDB. (Minsoo, 2014)

Data modelnya sendiri disebut BSON dengan struktur mirip dengan JSON. MongoDB sendiri sudah dikembangkan oleh 10gen sejak Oktober 2007, namun baru dipublikasikan pada Februari 2009. Selain karena performanya 4 kali lebih cepat dibandingkan MySQL serta mudah diaplikasikan, karena telah tergabung juga sebagai modul PHP.

Gambar 2.2 Logo MongoDB

Sumber: www.mongoDB.com

2.4.2 Aplikasi Node.Js

Node.js adalah sebuah *platform software* yang dipakai untuk membangun aplikasi *serverside* yang fleksibel di sebuah network. Node.js menggunakan JavaScript sebagai bahasa pemrogaman dan dapat dengan mudah menghasilkan *throughput* / pemrosesan tingkat tinggi melalui non-blocking I/O. Node.js memiliki *fitur built-in HTTP server library* yang menjadikannya mampu menjadi sebuah *web server* tanpa bantuan *software* lainnya seperti Apache atau Nginx. (Mahajan & Kumar, 2015)

Node.js pertama kali dibuat oleh Ryan Dahl pada tahun 2009 yang kemudian berkembang pesat di bawah licensi *Open Source* MIT oleh sebuah perusahaan bernama Joyent Inc. Node.js dikembangkan berdasarkan teknologi Google V8 JavaScript engine serta berisi kompilasi skrip inti dan banyak modul siap pakai yang bermanfaat sehingga pengguna (dalam hal ini *web developer*) tidak perlu melakukan *coding* dan mendesain segalanya dari awal.

2.4.3 NPM (*Node Private Manager*)

NPM adalah *package manager* untuk JavaScript. Jadi npm adalah sebuah *tool*/aplikasi kecil untuk mengatur *package*/aplikasi JavaScript yang mengunakan Node.js. NPM dapat digunakan untuk membuat *project* baru, menginstal modul atau *library*, menjalankan *script command line*. Apabila kita membutuhkan modul atau *library*, kita bisa menyuruh NPM untuk menginstalnya dengan perintah "npm install <nama modul>".

Node.js memiliki pustaka *server* HTTP sendiri sehingga memungkinkan untuk menjalankan *server web* tanpa menggunakan program *server web* seperti Apache atau Nginx. (Shah & Soomro, 2017)

2.4.4 Heroku

Heroku adalah sebuah *cloud platform* yang menjalankan bahasa pemrograman tertentu, Heroku mendukung bahasa pemrograman seperti Ruby, Node.js, Python, Java, PHP, dan lain-lain.

Heroku termasuk ke dalam kriteria Platform As A Service (PaaS), sehingga bagi anda yang ingin melakukan deploy aplikasi ke heroku cukup hanya dengan melakukan konfigurasi aplikasi yang ingin di deploy dan menyediakan platform yang memungkinkan pelanggan untuk mengembangkan, menjalankan, dan mengelola aplikasi tanpa kompleksitas membangun dan memelihara infrastruktur yang biasanya terkait dengan pengembangan dan peluncuran aplikasi.

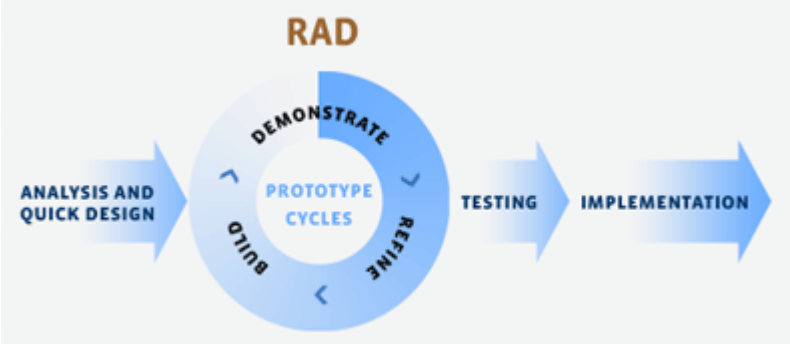
Gambar 2.3 Logo Heroku

Sumber: www.heroku.com

Gambar 2.3 merupakan logo dari heroku. Heroku server menyediakan kapasitas gratis sebesar 5MB.

2.5 *Rapid Application Development Model*

Metode yang akan digunakan untuk merancang dan mengembangkan aplikasi ini adalah metode RAD (*Rapid application Development*), RAD adalah suatu pendekatan berorientasi objek terhadap pengembangan sistem yang mencakup suatu metode pengembangan serta perangkat-perangkat lunak. RAD bertujuan mempersingkat waktu yang biasanya diperlukan dalam siklus hidup pengembangan sistem tradisional antara perancangan dan penerapan suatu sistem informasi. Pada akhirnya, RAD sama-sama berusaha memenuhi syarat-syarat bisnis yang berubah secara cepat (Kendall & Jullie, 2013). adapun langkah – langkah dalam RAD model adalah :



Gambar 2.4 Tahapan RAD (Kendall, 2010)

Sumber: www.researchgate.net

Gambar 2.4 merupakan tahapan yang dilakukan dalam metode RAD yang terdiri atas *Requirements Planning*, *Design* dan *Implementation*.

2.6 Teori *Unified Modeling Language* (UML)

UML merupakan singkatan dari "*Unified Modelling Language*" yaitu suatu metode permodelan secara visual untuk sarana perancangan sistem berorientasi objek, atau definisi UML yaitu sebagai suatu bahasa yang sudah menjadi standar pada visualisasi, perancangan dan juga pendokumentasian sistem *software*. Saat ini UML sudah menjadi bahasa standar dalam penulisan blue print software.

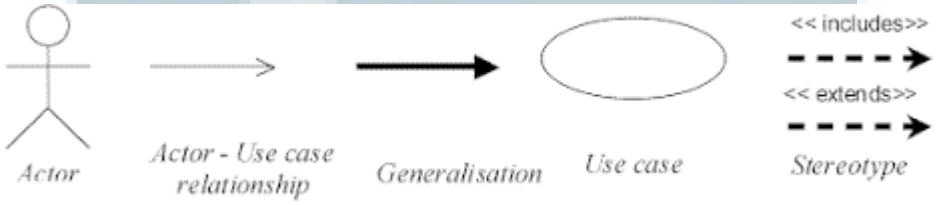
UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak. UML hanya berfungsi untuk melakukan pemodelan. Jadi penggunaan UML tidak terbatas pada metodologi tertentu, meskipun pada kenyataannya UML paling banyak digunakan pada metodologi berorientasi objek (Rosa & Shalahudin, 2014)

2.6.1 *Diagram* UML

UML merupakan sintak umum untuk membuat model logika dari suatu sistem dan digunakan untuk menggambarkan sistem agar dapat dipahami selama fase analisis dan desain. UML biasanya disajikan dalam bentuk diagram/gambar yang meliputi *class* beserta atribut dan operasinya, serta hubungan antar *class* yang meliputi *inheritance, association* dan komposisi. (Cepeda Porras & Gueh´ eneuc, 2010)

2.6.1.1 *Use Case Diagram*

Use case atau *diagram use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. (Rosa & Shalahudin, 2014)



Gambar 2.5 *Use Case Components*

Gambar 2.5 Merupakan *use case components* yang digunakan dalam membuat *use case diagram*.

2.6.1.2 *Activity Diagram*

Activity diagram menggambarkan berbagai alur aktifitas dalam sebuah sistem yang sedang dirancang, bagaimana masing-masing alur berawal, *decision* yang mungkin terjadi dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. *Activity diagram* tidak menggambarkan sifat internal dari sebuah sistem dan interaksi antara beberapa sub sistem secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum. (Bhattacharjee & Shyamasundar, 2009)

2.6.1.3 *Class Diagram*

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan method atau operasi (Rosa & Shalahudin, 2014). Berikut penjelasan atribut dan method :

1. Atribut merupakan variable-variabel yang dimiliki oleh suatu kelas.

2. Operasi atau method adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

2.6.1.4 *Sequence Diagram*

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, display, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). Sequence diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah event untuk menghasilkan output tertentu.

Message digambarkan sebagai garis berpanah dari satu objek ke objek lainnya. Pada fase desain berikutnya, message akan dipetakan menjadi operasi/metoda dari class. Activation bar menunjukkan lamanya eksekusi sebuah