BAB III

LANDASAN TEORI

3.1 Sistem Rekomendasi

Sistem Rekomendasi (SR) merupakan model aplikasi dari hasil observasi terhadap keadaan dan keinginan pelanggan. Sistem Rekomendasi memanfaatkan opini seseorang terhadap suatu barang dalam domain atau kategori tertentu, untuk membantu seseorang dalam memilih produk. Karena itu SR memerlukan model rekomendasi yang tepat agar apa yang direkomendasikan sesuai dengan keinginan pelanggan, serta mempermudah pelanggan mengambil keputusan yang tepat dalam menentukan produk yang akan dipilih (Purwanto, 2009).

Biasanya sistem rekomendasi terdapat pada penyedia layanan *e-commerce* seperti amazon, ebay, dan lain-lain. Penerapan rekomendasi ini biasanya melakukan prediksi sesuatu item, seperti rekomendasi film, musik, buku, berita dan lain sebagainya yang menarik *user*. Sistem ini berjalan dengan mengumpulkan data dari *user* secara langsung maupun tidak (Fadlil & Mahmudy, 2010).

Pengumpulkan data secara langsung dapat dilakukan sebagai berikut :

1. Meminta *user* untuk melakukan rating pada sebuah *item*.
2. Meminta *user* untuk melakukan rangking pada *item* favorit setidaknya memilih satu *item* favorit.

3. Memberikan beberapa pilihan *item* pada *user* dan memintanya memilih yang terbaik.
4. Meminta *user* untuk mendaftar *item* yang paling disukai atau *item* yang tidak disukainya.

Pengumpulan data dengan tidak langsung berhubungan dengan seorang *user*, dilakukan dengan cara seperti berikut:

1. Mengamati *item* yang dilihat oleh seorang *user* pada sebuah web *e-commerce*.
2. Mengumpulkan data transaksi pada sebuah toko online.

Data hasil pengumpulan, kemudian dilakukan perhitungan dengan algoritma tertentu yang kemudian hasil tersebut dikembalikan lagi kepada *user* sebagai sebuah rekomendasi *item* dengan parameter dari *user* tersebut. Sistem rekomendasi juga merupakan salah satu alternatif sebagai mesin pencari suatu item yang dicari oleh *user*.

3.2 *Item-based collaborative filtering*

Item -based collaborative filtering merupakan metode rekomendasi yang didasari atas adanya kesamaan antara pemberian rating terhadap suatu produk dengan produk yang dibeli. Tingkat kesamaan produk dihitung, kemudian dibagi dengan parameter kebutuhan pelanggan (yang membutuhkan rekomendasi) untuk memperoleh nilai kegunaan produk. Produk yang memiliki nilai kegunaan tertinggilah yang kemudian dijadikan rekomendasi (Purwanto, 2009).

3.3 *Customer Relationship Management*

Costumer Relationship Management (CRM) merupakan sebuah strategi bisnis yang berorientasi pada *customer*, dengan tujuan akhirnya memaksimalkan profit perusahaan dan kepuasan *customer*. CRM didefinisikan sebagai integrasi dari strategi penjualan, pemasaran, dan pelayanan yang terkoordinasi. CRM menyimpan informasi pelanggan dan merekam seluruh kontak yang terjadi antara pelanggan dan perusahaan, serta membuat profil pelanggan untuk staf perusahaan yang memerlukan informasi tentang pelangan tersebut (Istambul, 2010).

Salah satu contoh CRM yang bagus adalah situs www.amazon.com dimana pelanggan tidak hanya mendapat pengalaman yang menyenangkan selama membeli buku tetapi juga bisa melihat sejarah pembelian, melihat rekomendasi tentang sebuah buku, memonitor pengiriman, mendapat informasi terbaru (Danardatu, 2003).

3.4 Probabilitas

Probabilitas atau peluang adalah kemungkinan munculnya suatu kejadian. Untuk menghitung probabilitas bersyarat suatu kejadian dapat dilakukan dengan cara mencari banyaknya kejadian X dan Y muncul bersamaan, lalu dibandingkan dengan banyaknya kejadian X. sehingga dapat diketahui berapa kemungkinan Y muncul saat X muncul (Kreiner & Abraham, 2011).

$$P(Y \mid X) = \frac{f(X \mid Y)}{f(X)}$$

P(Y|X)= Peluang Y saat X muncul

f(X|Y)= Frekuensi X dan Y muncul bersamaan

f(X) = Frekuensi X

3.5 Web Aplikasi

Pada awalnya aplikasi web dibangun dengan hanya menggunakan bahasa yang disebut HTML (*HyperText Markup Langauge*). Pada perkembangan berikutnya, sejumlah skrip dan objek dikembangkan untuk memperluas kemampuan HTML seperti PHP dan ASP pada skrip dan *Apllet* pada objek. Aplikasi Web dapat dibagi menjadi dua jenis yaitu aplikasi web statis dan dinamis.

Web statis dibentuk dengan menggunakan HTML. Kekurangan aplikasi seperti ini terletak pada keharusan untuk memelihara program secara terus menerus untuk mengikuti setiap perkembangan yang terjadi. Kelemahan ini diatasi oleh model aplikasi web dinamis. Pada aplikasi web dinamis, perubahan informasi dalam halaman web dilakukan tanpa perubahan program tetapi melalui perubahan data. Sebagai implementasi, aplikasi web dapat dikoneksikan ke basis data sehingga perubahan informasi dapat dilakukan oleh operator dan tidak menjadi tanggung jawab dari *webmaster*.

Arsitektur aplikasi web meliputi klien, web *server*, *middleware* dan basis data. Klien berinteraksi dengan web *server*. Secara internal, web *server* berkomunikasi dengan *middleware* dan *middleware* yang berkomunikasi dengan basis

data. Contoh *middleware* adalah PHP dan ASP. Pada mekanisme aplikasi web dinamis, terjadi tambahan proses yaitu *server* menerjemahkan kode PHP menjadi kode HTML. Kode PHP yang diterjemahkan oleh mesin PHP yang akan diterima oleh klien (Kadir, 2002).

3.6 HTML

HTML (*HypertText Markup Language*) adalah suatu bahasa pemograman dalam bentuk *script* yang dapat digunakan untuk menyusun halaman *web*.

3.7 *Structured Query Language (SQL)*

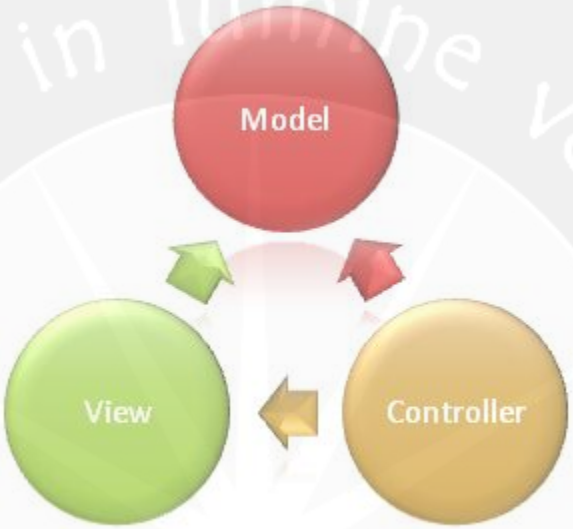
Structured Query Language (SQL) adalah bahasa yang dirancang secara khusus untuk berkomunikasi dengan basis data. SQL diciptakan dengan sedikit kata dan dirancang untuk memberikan efisiensi kepada pengguna untuk membaca dan menulis data dari suatu basis data.

3.8 *MVC (Model View Controller)*

MVC adalah arsitektur aplikasi yang memisahkan kode-kode aplikasi dalam tiga lapisan, *Model*, *View* dan *Control*. MVC termasuk dalam arsitektural *design pattern* yang menghendaki organisasi kode yang terstruktur dan tidak bercampur aduk. Ketika aplikasi sudah sangat besar dan menangani struktur data yang kompleks, harus ada pemisahan yang jelas antara domain *model*, komponen *view* dan *Control* yang mengatur penampilan *model* dalam *view*.

Secara sederhana dapat dijelaskan bahwa MVC merupakan sistem dasar pada *Codeigniter* yang

mengelompokkan fungsi-fungsi dalam *framework* tersebut berdasarkan tiga kategori menjadi *Model*, *View* dan *Controller* (MVC). Sebuah aplikasi MVC dirancang dan diterapkan menggunakan tiga atribut berikut



Gambar 3.1 MVC (Microsoft, 2014)

1. ***Model:*** *Model* mengandung informasi inti untuk aplikasi. *Model* mengandung data dan aturan-aturan validasi seperti akses data dan logika agregasi.
2. ***View:*** *View* membungkus tampilan (*presentation*) dari aplikasi, dan didalamnya secara umum berisi *markup* HTML.
3. ***Controller:*** *Controller* berisi logika pengaturan alur *(control-flow)* aplikasi. *Controller* ini yang menghubungkan antara *Model* dan *Views* untuk mengatur alur informasi dan jalannya aplikasi.

Pemisahan entitas memungkinkan lebih leluasa dalam membuat dan memelihara aplikasi Anda. Sebagai contoh, dengan pemisahan *views*, Anda dapat merubah bentuk tampilan tanpa harus menyentuh inti logika bisnis di dalamnya. Anda juga bisa memisahkan pekerjaan berdasarkan peran masing-masing, sehingga seorang perancang dapat bekerja pada *views* sedangkan pengembang lain bisa mengerjakan *model* (Microsoft, 2014).