BAB II

LANDASAN TEORI

2.1 Tinjauan Terhadap Penelitian Terkait

Berikut ini adalah beberapa hasil penelitian yang terkait dan menjadi referensi pada penelitian ini:

1. Anderias Eko Wijaya dan Deni Alfian dalam penelitian ini menjelaskan tentang membuat sebuah sistem rekomendasi laptop menggunakan *Collaborative Filtering* dan *content based filtering*. Dalam penelitian ini memiliki masalah yaitu sulitnya dalam memilih ataupun membeli laptop yang tepat dan sesuai keinginan dari konsumen, sehingga peneliti merancang aplikasi tersebut dengan menerapkan metode *Collaborative Filtering* dan *content based filtering* dengan memanfaatkan data transaksi penjualan sehingga mempermudah pembeli dalam memilih ataupun membeli laptop yang sesuai dan sebagai salah satu alternatif untuk membantu memberikan rekomendasi dan saran laptop.
2. Albertus Bayu Aji Priyono dalam penelitian ini menjelaskan tentang membuat sebuah sistem rekomendasi bagi pengguna dan rekomendasi paket wisata bagi penyedia perjalanan wisata, sehingga didapatkan objek dan paket wisata yang tepat bagi pengguna. informasi yang diberikan pengguna yang diperoleh secara eksplisit dan implisit. Yang dimaksud secara eksplisit adalah informasi tersebut diberikan langsung oleh pengguna. Misalnya, memberikan *rating* terhadap film yang pernah ditonton. Sedangkan yang dimaksud secara implisit adalah informasi tersebut diperoleh tanpa diketahui oleh pengguna.

Pembuatan sistem rekomendasi lokasi pariwisata dengan menggunakan metode *Collaborative Filtering* dan Algoritma Apriori bertujuan untuk menghasilkan sebuah sistem yang secara otomatis melakukan perhitungan dalam memberikan rekomendasi objek pariwisata yang tepat bagi pengguna yang menjadi target rekomendasi serta menghasilkan paket perjalanan wisata berdasarkan pola kebiasaan wisatawan dalam mengunjungi tempat wisata favoritnya.

2.2 Sistem Rekomendasi

Sistem rekomendasi adalah sebuah aplikasi yang berfungsi untuk memprediksi suatu *item* yang menarik bagi user, contohnya rekomendasi film, musik, buku, berita dan lain sebagainya.

Ada dua tipe metode yang diterapkan pada sistem rekomendasi, yaitu collaborative filtering dan content based filtering. Collaborative filtering adalah algoritma sistem rekomendasi dimana rekomendasi diberikan berdasarkan pertimbangan data dari user yang lain. Sedangkan Content based filtering pemberian rekomendasi diberikan dengan mengeksplorasi isi dari profil user, deskripsi produk atau hal-hal yang berhubungan dengan pembentukan pilihan user atas sebuah *item* . Penelitian ini menggunakan metode content based filtering pada pembentukan item-*item* yang muncul pada rekomendasi yang diberikan pada user (Badriyah, dkk, 2018).

2.3 Data *Mining*

Data *Mining* merupakan proses untuk menggali (*mining*) pengetahuan dan informasi baru dari data yang berjumlah banyak pada data *warehouse*, dengan

menggunakan kecerdasan buatan (*Artificial Intelligence*), statistik dan matematika. Data *mining* merupakan teknologi yang diharapkan dapat menjembatani komunikasi antara data dan pemakainya.

Beberapa solusi yang diberikan data *mining* antara lain :

1. Menebak Target Pasar

Data Mining dapat mengelompokkan (*clustering*) model pembeli dan melakukan klasifikasi terhadap setiap pembeli terhadap setiap pembeli sesuai dengan karakteristik yang diinginkan.

2. Melihat Pola Beli Dari Waktu Ke Waktu

Data Mining dapat digunakan untuk melihat pola beli dari waktu ke waktu.

3. *Cross-Market Analysis*

Data Mining dapat dimanfaatkan untuk melihat hubungan antara satu produk dengan produk lainnya.

4. Profil Pelanggan

Data Mining bisa membantu pengguna untuk melihat profil pembeli sehingga dapat diketahui kelompok pembeli tertentu cenderung kepada satu produk apa saja.

5. Informasi *Summary*

Data *Mining* dapat membuat laporan *summary* yang bersifat multidimensi dan dilengkapi dengan informasi statistik lainnya.

2.4 Pengelompokan Data Mining

Data mining dibagi menjadi beberapa kelompok berdasarkan tugas yang dapat dilakukan, yaitu (Tampubolon dkk, 2013) :

1. Deskripsi

Terkadang peneliti dan analis secara sederhana ingin mencoba mencari data untuk menggambarkan pola dan kecenderungan yang terdapat dalam data. Sebagai contoh, petugas pengumpulan suara mungkin tidak dapat menentukan keterangan atau fakta bahwa siapa yang tidak cukup professional akan sedikit didukung dalam pemilihan presiden. Deskripsi dari pola dan kecenderungan sering memberikan kemungkinan penjelesan untuk suatu pola atau kecenderungan.

2. Estimasi

Estimasi hampir sama dengan klasifikasi, kecuali variable target estimasi lebih kearah numerik dari pada kearah kategori. Model dibangun menggunakan record lengkap yang menyediakan nilai dari variabel target sebagai prediksi. Selanjutnya, pada peninjauan berikutnya estimasi nilai dari variabel target dibuat berdasarkan nilai variabel predikasi.

3. Prediksi

Prediksi hampir sama dengan klasifikasi dan estimasi, kecuali bahwa dalam predikasi nilai dari hasik akan ada dimasa mendatang. Contoh prediksi bisnis dan penelitian adalah:

a. Prediksi harga beras dalam tiga bulan yang akan dating.

b. Prediksi persentasi kenaikan kecelakaan lalu lintas tahun depan jika batas bawah kecepatan dinaikkan.

Beberapa metode dan teknik yang digunakan dalam klasifikasi dan estimasi dapat pula digunakan (untuk keadaan yang tepat) untuk prediksi.

4. Klasifikasi

Dalam klasifikasi, terdapat target variabel kategori. Sebagai contoh, penggolongan pendapatan dapat dipisahkan dalam tiga kategori , yaitu pendapatan tinggi, pendapatan sedang, dan pendapatan rendah

5. Pengklusteran (*Clustering*)

Pengkluteran merupakan pengelompokan *record*, pengamatan, atau memperhatikan dan membentuk kelas objek-objek yang memiliki kemiripan. Kluster adalah kumpulan record yang memiliki kemiripan satu dengan yang lainnya dan memiliki ketidakmiripan dengan *record-record* dalam kluster lain. Pengklusteran berbeda dengan klasifikasi yaitu tidak adanya variabel target dalam pengklusteran. Pengklusteran tidak mencoba untuk melakukan klasifikasi, mengestimasi, atau memprediksi nilai dari variabel target. Akan tetapi, algoritma pengklusteran mencoba untuk melakukan pembagian terhadap keseluruhan data menjadi kelompok-kelompok yang memiliki kemiripan (*homogeny*), yang mana kemiripan dalam satu kelompok akan bernilai maksimal, sedangkan kemiripan dengan record dalam kelompok lain akan bernilai minimal.

6. Asosiasi

Tugas asosiasi dalam data mining adalah menemukan attribut yang muncul dalam satu waktu. Dalam dunia bisnis lebih umum disebut analisis keranjang belanja.

2.5 Pemograman Aplikasi

2.5.1 Website

Menurut (Rukiastiandari, 2019) "World Wide Web atau lebih sering dikenal sebagai web adalah suatu layanan sajian informasi yang menggunakan konsep hyperlink (tautan), yang memudahkan surfer (sebutan para pemakai komputer yang melakukan browsing atau penelusuran informasi melalui internet). Keistimewaan inilah yang telah menjadikan web sebagai service yang paling cepat pertumbuhannya."

2.5.2 HTML

Menurut (Setiawan, 2016) HTML merupakan singkatan dari *Hypertext Markup Language* merupakan bahasa pemrograman terstruktur yang dikembangkan untuk membuat halaman web yang dapat diakses atau ditampilkan menggunakan web browser. HTML sendiri secara resmi lahir pada tahun 1989 oleh Tim Berners Lee dan dikembangkan oleh W3C (*World Wide Web Consortium*) berupa tag-tag yang menyusun setiap elemen dari website. HTML berperan sebagai penyusun struktur halaman website yang menempatkan setiap elemen website sesuai layout yang diinginkan

HTML terdiri atas unsur-unsur yang membentuk struktur *script* (Rudjiono, D., & Jam tangantro, H. 2020), yaitu:

a. Tag

Tag adalah simbol khusus berupa dua karakter “<” dan “>” yang mengapit suatu tag.

b. Atribut

Atribut adalah *property* yang mengatur bagaimana elemen dari suatu tag akan ditampilkan. Atribut ditulis di dalam simbol tag setelah nama tag dengan di pisahkan oleh spasi.nilai suatu atribut ditulis di dalam tanda petik ganda (“...”), dipisahkan dengan simbol sama dengan (=) dari nama atribut.

c. Element

Element merupakan bagian dari skrip HTML yang terdiri dari *tag* pembuka, isi *element*, dan tag penutup.

2.5.3 CSS

CSS merupakan kependekan dari *Cascading Style Sheet* yang berfungsi untuk mengatur tampilan dengan kemampuan jauh lebih baik dari tag maupun atribut standar HTML (*Hypertext Markup Language*). CSS sebenarnya adalah suatu kumpulan atribut untuk fungsi format tampilan dan dapat digunakan untuk mengontrol tampilan banyak dokumen secara bersamaan. Keuntungan menggunakan CSS yaitu jika ingin mengubah format dokumen, maka tidak perlu mengedit satu persatu (Novendri, M. S, dkk. 2019).

2.5.4 JavaScript

Javascript dibuat dan didesain oleh Brandan Eich, seorang karyawan Netscape pada bulan September 1995. Awalnya bahasa pemrograman ini disebut Mocha, kemudian berganti nama lagi menjadi Mona, lalu berganti lagi menjadi *Livescript*, dan pada akhirnya menyandang nama menjadi *Javascript*. Pada akhirnya pada tahun 2006 *Javascript* telah beralih dari bahasa pemrograman

yang serba terbatas menjadi salah satu *tool* paling penting bagi *web developer*. *Javascript* adalah bahasa pemrograman atau kode script yang diletakan bersama kode HTML ataupun terpisah yang digunakan untuk membuat tampilan website lebih dinamis (Rudjiono, D., & Jam tangantro, H. 2020).

2.5.5 Bootstrap

Bootstrap adalah sebuah *framework* yang menyediakan set kelas CSS dan fungsi *JavaScript* untuk memudahkan proses pembangunan antarmuka halaman web. Mengaktifkan fitur design responsif dukungan untuk menampilkan desktop maupun mobile. Situs ini dikembangkan agar dapat bekerja dengan baik pada desktop maupun mobile. developer tidak harus bekerja dengan CSS untuk membuat website terlihat menarik atau mendukung prinsip desain responsive, kecuali diperlukan.

2.5.6 PHP

PHP merupakan bahasa *server-side scripting* yang menyatu dengan HTML untuk membuat halaman web yang dinamis. Karena itu sintaks dan perintah-perintah PHP akan dieksekusi di server kemudian hasilnya dikirimkan ke browser dalam format HTML.

Hampir seluruh aplikasi berbasis web dapat dibuat dengan menggunakan PHP. Namun, keunggulan paling utama adalah melekatnya integrasi dengan sistem *database*dalam web. Kelebihan dari PHP diantaranya adalah:

a. PHP mudah dibuat dan dijalankan.
b. Bersifat *open source* dan dapat didistribusikan kembali dibawah lisensi GPL (*GNU Public License*).

c. Dapat dioperasikan pada *platform* Linux maupun Windows.

d. Bahasa pemrogramannya sederhana, singkat dan mudah.

e. Didukung oleh banyak *database*, seperti MySQL, PostgreSQL, dan lainlain.

f. HTML-*embedded*, artinya bahasa pemrogramannya dapat ditulis dengan menempelkannya pada sintak-sintak HTML.

2.5.7 Database

*Database*sebagai kumpulan terorganisasi dari data-data yang berhubungan sedemikian rupa sehingga mudah disimpan, dimanipulasi serta dipanggil oleh pengguna. Terminologi hubungan berarti data mendeskripsikan domain (ranah) tertentu sehingga pengguna mudah untuk mendapatkan jawaban atas pertanyaan yang diajukan ke basis data tersebut. Sedangkan pengertian sistem basis data adalah sebagai koleksi dari data-data yang terorganisasi sedemikian rupa sehingga data mudah disimpan dan dimanipulasi (diperbarui, dicari, diolah dengan perhitungan-perhitungan tertentu, serta dihapus) (Novendri, dkk, 2019).

2.5.8 MySQL

MySQL merupakan RDBMS (atau server *database*) yang mengelola *database*dengan cepat menampung dalam jumlah sangat besar dan dapat diakses oleh banyak *user*. MySQL merupakan RDBMS (*Relational DatabaseManagement System)* server. RDBMS adalah suatu program yang memungkinkan pengguna *database*untuk membuat, mengelola, dan menggunakan data pada model relational. Dengan demikian, tabel-tabel yang ada pada *database*memiliki relasi antara satu tabel dengan tabel lainnya. Berikut ini merupakan keunggulan dari MySQL, yaitu:

a. Cepat, handal dan mudah dalam penggunaannya.

b. Didukung oleh berbagai bahasa.

c. Mampu membuat tabel berukuran sangat besar.

d. Bersifat open source dan didistribusikan dengan gratis.

e. Melekatnya integrasi PHP dan MySQL.

2.6 Algoritma Apriori

Algoritma apriori merupakan suatu bagian dari *data mining* yang memakai Teknik aturan asosiasi dalam mencari kombinasi *item* yang paling sering muncul bersama sesuai dengan kriteria yang diinginkan. Aturan yang menyatakan asosiasi antara beberapa atribut sering disebut affinity analysis atau market basket analysis. Analisis asosiasi atau association rule mining adalah teknik data mining untuk menemukan aturan suatu kombinasi *item* . Salah satu tahap analisis asosiasi yang menarik perhatian banyak peneliti untuk menghasilkan algoritma yang efisien adalah analisis pola frequensi tinggi(*frequent* pattern mining). Penting tidaknya suatu asosiasi dapat diketahui dengan dua tolok ukur, yaitu : *support* dan *confidence*. *Support* (nilai penunjang) adalah persentase kombinasi *item* tersebut dalam database, sedangkan *confidence* (nilai kepastian) adalah kuatnya hubungan antara-*item* dalam aturan asosiasi. (Tampubolon, dkk, 2013)

2.6.1 Minimum support dan *Minimum confidence*

Minimum support yaitu parameter yang digunakan sebagai batasan frekuensi kejadian atau *support* count yang harus dipenuhi suatu kelompok data untuk dapat dijadikan aturan. Untuk mencari nilai *support count* dapat ditentukan dengan rumus berikut (Masnur, 2015):

$$Support(A) = \frac{Jumlah\ Transaksi\ Berisi\ A}{Jumlah\ Total\ Transaksi} \quad (1)$$

Untuk nilai *support* dari 2 *item* atau lebih diperoleh dengan rumus:

$$Support(A \cup B) = \frac{Jumlah\ Transaksi\ Berisi\ A\ dan\ B}{Jumlah\ Total\ Transaksi} \quad (2)$$

Minimum confidence adalah sebuah parameter yang menjelaskan *minimum* level dari nilai *confidence* yang harus ditentukan agar dapat terbentuknya suatu aturan asosiasi yang berkualitas. Berikut rumus untuk menentukan nilai *confidence*:

$$Confidence\ P(B \mid A) = \frac{Jumlah\ Transaksi\ Mengandung\ A\ dan\ B}{Jumlah\ Transaksi\ Mengandung\ A} \quad (3)$$

Suatu aturan asosiasi dapat dibentuk dengan mencari kombinasi dari *item* yang dapat terbentuk dengan *support* lebih besar atau sama dengan *minimum support* yang ditentukan. Kuat tidaknya aturan asosiasi ditunjukkan berdasarkan *lift ratio* nya. *Lift ratio* merupakan ukuran seberapa pentingnya aturan asosiasi. *Lift ratio* dapat dihitung dengan membandingkan nilai *confidence* dengan nilai *benchmark confidence*. *Benchmark confidence* merupakan suatu perbandingan antara jumlah dari semua *item consequent* terhadap jumlah total dari seluruh transaksi.

$$Brenchmark\ Confidence = \frac{Item\ Consequent}{Total\ Transaksi} \quad (4)$$

$$Lift\ Ratio(A, B) = \frac{Confidence\ P(B|A)}{Support\ (B)} \quad (5)$$

Sebuah rule dapat dikatakan valid apabila mempunyai nilai *Lift ratio* > 1, artinya aturan tersebut memiliki manfaat. Nilai *lift ratio* yang semakin tinggi menunjukkan kekuatan asosiasi yang lebih besar.

2.6.2 Contoh Penggunaan Algoritma Apriori

Ada 10 data transaksi keranjang belanja dari sebuah grosir seperti pada tabel berikut

Tabel 2.1. Pola Penjualan

ID Transaksi	*Itemset*
1	Celana, Baju, Kacamata
2	Sandal, Topi, Baju, Celana, Sepatu
3	Celana, Sandal, Baju, Sepatu
4	Sepatu, Sandal, Topi, Kacamata
5	Baju, Celana, Topi, Sandal
6	Celana, Sandal, Baju
7	Celana, Sandal, Sepatu
8	Baju, Sepatu, Kacamata
9	Jam tangan, Celana, Baju
10	Baju, Topi, Jaket

a. Pembentukan *itemset*

Menentukan *itemset* kombinasi *itemset* yang sering muncul dalam dataset (data transaksi). sehingga aturan yang dibuat nantinya mampu menghasilkan nilai *Confidence* yang tinggi, pada percobaan ini jumlah *minimum support* yang digunakan adalah 5, yaitu:

1. Kombinasi 1 *Itemset* (K=1)

   Pembentukan 1 *itemset* dengan *minimum support* 5. Dapat ditentukan dengan rumus berikut:

$$Support(A) = \frac{Jumlah\ Transaksi\ Berisi\ A}{Jumlah\ Total\ Transaksi}\ x\ 100\%$$

Tabel 2.2. Nilai *Support* dari 1 *Itemset*

Itemset	*Support Count*	*Support*
Baju	8	80%
Sandal	6	60%
Celana	7	70%
Sepatu	5	50%
Topi	4	40%
Kacamata	3	30%
Jam tangan	1	10%
Jaket	1	10%

2. Kombinasi 2 *Itemset* (K=2)

Pembentukan 2 *itemset* dengan *minimum support* 5. Dapat ditentukan dengan rumus berikut:

$$Support(A, B) = \frac{Jumlah\ Transaksi\ Berisi\ A\ dan\ B}{Jumlah\ Total\ Transaksi}\ x\ 100\%$$

Tabel 2.3. Kandidat 2 *Itemset*

Itemset	*Support count*	*Support*
Baju, Sandal	4	40%
Baju, Celana	6	60%
Baju, Sepatu	3	30%

Baju, topi	3	30%
Baju, Kacamata	1	10%
Baju, Jam tangan	1	10%
Baju, Jaket	1	10%
Sandal, Celana	5	50%
Sandal, Sepatu	4	40%
Sandal, Topi	3	30%
Sandal, Kacamata	1	10%
Sandal, Jam tangan	0	0%
Sandal, Jaket	0	0%
Celana, Sepatu	3	30%
Celana, topi	2	20%
Celana, Kacamata	0	0%
Celana, Jam tangan	1	10%
Celana, Jaket	0	0%
Sepatu, topi	2	20%
Sepatu, Kacamata	2	20%
Sepatu, Jam tangan	0	0%
Sepatu, Jaket	0	0%
Topi, Kacamata	1	10%
Topi, Jam tangan	0	0%
Topi, baju	1	10%
Kacamata, Jam tangan	0	0%

Kacamata, Jaket	0	0%
Jam tangan, Jaket	0	0%

Dengan *minimum support* 5, jadi kombinasi dari 2 *itemset* yang tidak memenuhi *minimum support* akan eliminasi atau dihilangkan, dapat dilihat pada table berikut:

Tabel 2.4. *Minimum* 2 *itemset* 50%

Itemset	*Support count*	*Support*
Baju, Celana	6	60%
Sandal, Celana	5	50%

3. Kombinasi 3 *Item seet* (K=3)

Pembentukan 3 *itemset* dengan *minimum support* 5. Dapat ditentukam dengan rumus berikut:

$$Support(A, B, C) = \frac{Jumlah\ Transaksi\ Berisi\ A, B\ dan\ C}{Jumlah\ Total\ Transaksi}\ x\ 100\%$$

Tabel 2.5. Kombinasi 3 *Itemset*

Itemset	*Support Count*	*Support*
Baju, Celana, Kacamata	1	10%
Baju, Sandal, Celana	4	40%
Baju, Sandal, topi	1	10%
Baju, Sandal, Sepatu	2	20%
Sandal, Celana, topi	2	20%

Sandal, Celana, Sepatu	3	30%
Celana, topi, Sepatu	1	10%
Sandal, Sepatu, topi	1	10%
Sandal, Sepatu, Kacamata	1	10%
Sepatu, topi, Kacamata	1	10%
Baju, Celana, topi	1	10%
Baju, Sepatu, Kacamata	1	10%
Jam tangan, Celana, Baju	1	10%
Baju, topi, Jaket	1	10%

4. Kombinasi 4 *itemset*

Pembentukan 4 *itemset* dengan *minimum support* 5. Dapat ditentukam dengan rumus berikut:

$$Support(A,B,C,D) = \frac{Jumlah\ Transaksi\ Berisi\ A,B,C,D}{Jumlah\ Total\ Transaksi}\ x\ 100\%$$

Tabel 2.6. Kombinasi 4 *Itemset*

Itemset	*Support Count*	*support*
Baju, Sandal, Celana, Sepatu	2	20%
Baju, Sandal, Celana, topi	2	20%
Sandal, Sepatu, topi, Kacamata	1	10%

Karena hasil dari *itemset* 3 dan 4 tidak ada yang memenuhi *minimum support* nya, maka kombinasi 2 *itemset* lah yang nantinya dapat dilakukan pembentukan asosiasi.

b. Pembentukan aturan asosiasi

Setelah kombinasi *itemset* terbentuk, kita akan mencari aturan asosiasi berdasarkan *minimum support* dan *minimum confidence* yang ditentukan sebelumnya.

$$Confidence\ P(B \mid A) = \frac{Jumlah\ Transaksi\ Mengandung\ A\ dan\ B}{Jumlah\ Transaksi\ Mengandung\ A}$$

Dari kombinasi 2 *itemset* yang telah dibentuk sebelumnya, dapat dilihat nilai *support* dan *confidence* pada tabel berikut:

Tabel 2.7. Aturan Asosiasi

Itemset	*Confidence*	
Jika membeli Baju, maka akan membeli Celana	6/8	75%
Jika membeli Celana, maka akan membeli Baju	6/7	85.7%
Jika membeli Sandal, maka akan membeli Celana	5/6	83.3%
Jika membeli Celana, maka akan membeli Sandal	5/7	71.4%

2.7 Collaborative Filtering

Menurut Sarwar dalam Alfian (2018), *Collaborative Filtering* adalah suatu konsep dimana opini dari pengguna lain yang ada digunakan untuk memprediksi *item* yang mungkin disukai/diminati oleh seorang pengguna. Kualitas rekomendasi yang diberikan dari metode ini sangat bergantung dari opini pengguna lain (*neighbor*) terhadap suatu *item* . Belakangan diketahui bahwa melakukan reduksi neighbor (yaitu dengan memotong *neighbor* sehingga hanya beberapa pengguna yang memiliki kesamaan / *similiarity* tertinggi sajalah yang akan digunakan dalam perhitungan) mampu meningkatkan kualitas rekomendasi yang diberikan .

Collaborative Filtering memberikan rekomendasi berdasarkan kumpulan dari pendapat, minat dan ketertarikan beberapa *user* yang biasanya diberikan dalam bentuk *rating* yang diberikan *user* kepada suatu *item* . Untuk memperoleh data *rating* dari *user* yang digunakan dalam sistem rekomendasi. Pendekatan *Collaborative Filtering* pada dasarnya dibagi menjadi dua kategori yaitu *user-based Collaborative Filtering* disebut juga *memory-based*, dan *item based Collaborative Filtering* yang disebut juga *model-based*.

Pada pendekatan *user based Collaborative Filtering* sistem memberikan rekomendasi kepada *user* item-*item* yang disukai atau di *rating* oleh *user* – *user* lain yang memiliki banyak kemiripan dengannya. Misalnya, *user* a menyukai atau me*rating item* 1,2 dan 3, kemudian *user* b menyukai *item* 1,2 dan 4 maka sistem akan merekomendasikan *item* 3 kepada *user* b dan *item* 4 kepada *user* a. Kelebihan dari pendekatan *user* based *Collaborative Filtering* adalah dapat menghasilkan rekomendasi yang berkualitas baik. Sedangkan kekurangannya adalah kompleksitas perhitungan akan semakin bertambah seiring dengan bertambahnya pengguna sistem, semakin banyak pengguna (*user*) yang menggunakan sistem maka proses perekomendasian akan semakin lama.

Pendekatan *item based Collaborative Filtering* memberikan rekomendasi berdasarkan kemiripan antar *item* . Metode ini merupakan metode rekomendasi yang didasari atas adanya kesamaan antara pemberian *rating* terhadap suatu *item* dengan *item* yang pernah di *rating user* lain. *item* yang telah di *rating* oleh *user* akan menjadi patokan untuk mencari sejumlah *item* lainnya yang berkorelasi dengan *item* yang telah di *rating user*. Motivasi kunci dibalik metode ini adalah

user akan cenderung menyukai *item* yang sejenis atau mempunyai korelasi dengan *item* yang telah disukainya.

Secara umum proses pemberian rekomendasi pada *Collaborative Filtering* terdiri atas 2 langkah yaitu Penemuan *similar item* dan Penghitungan prediksi. Terdapat beberapa algoritma untuk menemukan *similar item* , yaitu:

1. Algoritma *Cosine-based Similarity* Pada kasus ini dua *item* dianggap sebagai 2 vektor. Kesamaan antara 2 *item* ini diukur dengan menghitung *cosinus* dari sudut antara 2 vektor *item* . *item* dibandingkan misalnya u dan v, dianggap sebagai sebuah vektor baris dengan anggotanya adalah nilai *rating* yang diberikan terhadap kedua *item* tersebut. Dua vektor dikatakan sama jika membentuk sudut atau nilai *cosinus* nya 1. Dengan kata lain dua *item* dikatakan mirip jika nilai *cosinus* dari perhitungan mendekati 1.

   Persamaan *cosine-based similarity*:

   $$sim(i,j) = \cos(\vec{i},\vec{j}) = \frac{\vec{i} * \vec{j}}{||\vec{i}||_2 * ||\vec{j}||_2}$$

   Dimana $\vec{i}$ dan $\vec{j}$ merupakan vektor vektor baris dengan anggota nilai *rating* pada *item* i dan *item* j. *Cos* $(\vec{i}\vec{j})$ merupakan nilai *cosinus* sudut yang dibentuk vektor baris *rating*itemi dan j.

2. Algoritma *Correlation-based Similarity* Pada algoritma ini kemiripan antara dua *item* i dan j diukur dengan menghitung korelasi *Pearson correlation*. Agar perhitungan korelasi yang diperoleh akurat, terlebih dahulu dilakukan pemisahan terhadap *co-rated items* (*item-item* yang kedua *item* i dan j nya di-*rating* oleh *user*).

Persamaan *correlation-based similarity:*

$$sim\ (i,j) = \frac{\Sigma ueU(R_{u,i} - \bar{R}_i)(R_{u,j} - \bar{R}_j)}{\sqrt{\Sigma ueU(R_{u,i} - \bar{R}_i)^2}\sqrt{\Sigma ueU(R_{u,j} - \bar{R}_j)^2}}$$

3. *Adjusted-cosine similarity* Persamaan *adjusted cosine similarity* digunakan untuk menghitung nilai kemiripan antar *item.* Perhitungan kemiripan ini merupakan modifikasi dari perhitungan kemiripan berbasis vektor dimana dengan melihat fakta bahwa setiap *user* memiliki skema *rating* yang berbeda-beda. Terkadang *user* memberi *rating* yang tinggi terhadap *item* a disisi lain *user* memberi *rating* yang sangat rendah pada *item* b. Maka dari itu untuk setiap *rating* dikurangi dengan rata-rata *rating* yang diberikan *user*.

$$sim(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \bar{R}_u)(R_{u,j} - \bar{R}_u)}{\sqrt{\sum_{u \in U}(R_{u,i} - \bar{R}_u)^2}\sqrt{\sum_{u \in U}(R_{u,j} - \bar{R}_u)^2}}$$

Keterangan :

sim(i,j) = Nilai kemiripan antara *item* i dan *item* j.

$u \in U$ = Himpunan *user* u yang me*rating item* i dan *item* j.

R_{ui} = *Rating user* u pada *item* i.

R_{ui} = *Rating user* u pada *item* j.

$\overline{R}_u$ = Nilai rata-rata *rating user* u

Untuk menghitung nilai kemiripan (*similarity*) antar 2 item, diperlukan himpunan *user* yang me-*rating item* tersebut. Nilai yang dihasilkan pada persamaan *adjusted-cosine similarity* adalah berkisar antara +1.0 dengan - 1.0. *item* dianggap

saling berkorelasi jika nilai *similarity* antara kedua *item* tersebut mendekati +1, begitu juga sebaliknya *item* dianggap tidak berkorelasi apabila nilai similarity-nya mendekati -1.

2.8 *Unified Modeling Language* (UML)

UML (*Unified Modelling Language*) adalah suatu metode dalam pemodelan secara visual yang digunakan sebagai sarana perancangan sistem berorientasi objek. Awal mulanya, UML diciptakan oleh *Object Management Group* dengan versi awal 1.0 pada bulan Januari 1997. UML juga dapat didefinisikan sebagai suatu bahasa standar visualisasi, perancangan, dan pendokumentasian sistem, atau dikenal juga sebagai bahasa standar penulisan *blueprint* sebuah *software.*

UML diharapkan mampu mempermudah pengembangan piranti lunak (RPL) serta memenuhi semua kebutuhan pengguna dengan efektif, lengkap, dan tepat. Hal itu termasuk faktor-faktor *scalability, robustness, security,* dan sebagainya. Sistem yang baik itu berawal dari perancangan dan pemodelan yang matang. Salah satu yang bisa kamu praktekkan, yaitu dengan menggunakan UML. Adapun tujuan dan fungsi perlu adanya UML yaitu sebagai berikut:

1. Dapat memberikan bahasa pemodelan visual atau gambar kepada para pengguna dari berbagai macam pemrograman maupun proses umum rekayasa.
2. Menyatukan informasi-informasi terbaik yang ada dalam pemodelan.
3. Memberikan suatu gambaran model atau sebagai bahasa pemodelan visual yang ekspresif dalam pengembangan sistem.

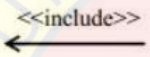
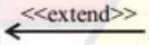
4. Tidak hanya menggambarkan model sistem *software* saja, namun dapat memodelkan sistem berorientasi objek.
5. Mempermudah pengguna untuk membaca suatu sistem.
6. Berguna sebagai *blueprint,* jelas ini nantinya menjelaskan informasi yang lebih detail dalam perancangan berupa *coding* suatu program.

2.8.1 Use case diagram

Menurut Tohari dalam Tabrani dan Aghnia (2019:47) menyimpulkan bahwa, "*use case* adalah rangkaian atau uraian sekelompok yang saling terkait dan membentuk sistem secara teratur yang dilakukan atau diawasi oleh sebuah aktor". Komponen-komponen pada *use case diagram* di antaranya sebagai berikut:

a. Sistem, menunjukkan batasan sistem antara relasi dengan *actor* yang menggunakannya dan fitur-fitur yang terdapat didalam sistem tersebut.
b. Aktor, merupakan seorang pengguna yang nantinya akan menjalankan sistem untuk melakukan sesuatu.
c. *Use case,* merupakan suatu gambaran fungsional dari sebuah system yang nantinya gambaran tersebut dapat menjelaskan fungsi-fungsi dari sistem.

Simbol-simbol pada *use case diagram* sebagai berikut:

Simbol	Keterangan
	Aktor : Mewakili peran orang, sistem yang lain, atau alat ketika berkomunikasi dengan *use case*
	*Use case* : Abstraksi dan interaksi antara sistem dan aktor
	Association : Abstraksi dari penghubung antara aktor dengan use case
	Generalisasi : Menunjukkan spesialisasi aktor untuk dapat berpartisipasi dengan use case
	Menunjukkan bahwa suatu use case seluruhnya merupakan fungsionalitas dari use case lainnya
	Menunjukkan bahwa suatu use case merupakan tambahan fungsional dari use case lainnya jika suatu kondisi terpenuhi

Gambar 2.1. Simbol *Use case diagram*

2.8.2 Activity diagram

Menurut Tohari dalam Tabrani dan Aghnia (2019:46) mendefinisikan bahwa, "*activity diagram* memodelkan *workflow* proses bisnis dan urutan aktivitas dalam sebuah proses. Diagram ini sangat mirip dengan *flowchart* karena memodelkan *workflow* dari suatu aktivitas lainnya atau dari aktifitas ke status". Komponen *Activity diagram* sebagai berikut:

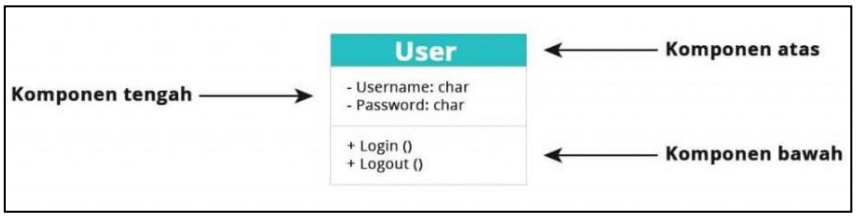
Simbol	Nama	Keterangan
	Status awal	Sebuah diagram aktivitas memiliki sebuah status awal.
	Aktivitas	Aktivitas yang dilakukan sistem,aktivitas biasanya diawali dengan kata kerja.
	Percabangan / Decision	Percabangan dimana ada pilihan aktivitas yang lebih dari satu.
	Penggabungan / Join	Penggabungan dimana yang mana lebih dari satu aktivitas lalu digabungkan jadi satu.
	Status Akhir	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
	Swimlane	Swimlane memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

Gambar 2.2. Komponen *Activity diagram*

2.8.3 Class diagram

Class diagram atau diagram kelas adalah salah satu jenis diagram struktur pada UML yang menggambarkan dengan jelas struktur serta deskripsi *class*, *atribut*, metode, dan hubungan dari setiap objek. Ia bersifat statis, dalam artian diagram kelas bukan menjelaskan apa yang terjadi jika kelas-kelasnya berhubungan, melainkan menjelaskan hubungan apa yang terjadi.

Komponen *Class diagram* sebagai berikut:



Gambar 2.3. Komponen *Class diagram*

a. Komponen atas, komponen ini berisikan nama *class*. Setiap *class* pasti memiliki nama yang berbeda-beda, sebutan lain untuk nama ini adalah *simple name* (nama sederhana).

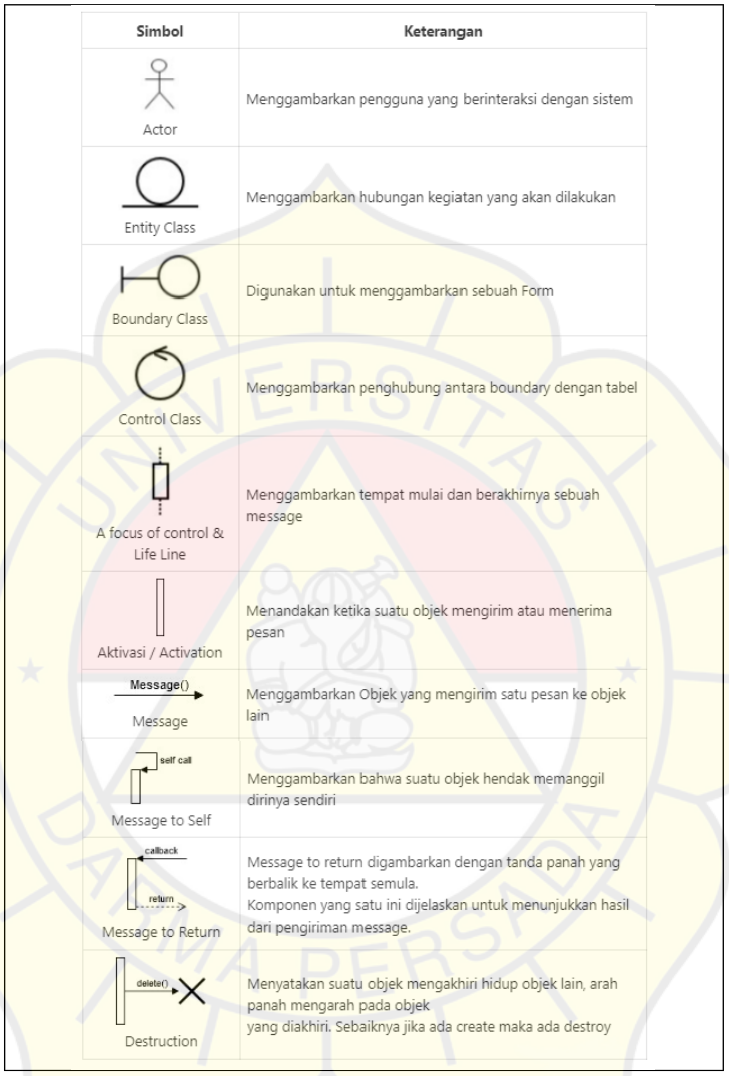
b. Komponen tengah, komponen ini berisikan atribut dari *class*, komponen ini digunakan untuk menjelaskan kualitas dari suatu kelas. Atribut ini dapat menjelaskan dapat ditulis lebih detail, dengan cara memasukan tipe nilai.

c. Komponen bawah, komponen ini menyertakan operasi yang ditampilkan dalam bentuk daftar. Operasi ini dapat menggambarkan bagaimana suatu *class* dapat berinteraksi dengan data.

2.8.4 Sequence diagram

Sequence diagram menggambarkan interaksi antar objek berupa pesan (*message*) yang digambarkan terhadap waktu. *Sequence diagram* terdiri antar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). *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 proses (Syafitri, 2016).

Simbol *Sequence diagram* sebagai berikut :



Simbol	Keterangan
Actor	Menggambarkan pengguna yang berinteraksi dengan sistem
Entity Class	Menggambarkan hubungan kegiatan yang akan dilakukan
Boundary Class	Digunakan untuk menggambarkan sebuah Form
Control Class	Menggambarkan penghubung antara boundary dengan tabel
A focus of control & Life Line	Menggambarkan tempat mulai dan berakhirnya sebuah message
Aktivasi / Activation	Menandakan ketika suatu objek mengirim atau menerima pesan
Message() Message	Menggambarkan Objek yang mengirim satu pesan ke objek lain
self call Message to Self	Menggambarkan bahwa suatu objek hendak memanggil dirinya sendiri
callback return Message to Return	Message to return digambarkan dengan tanda panah yang berbalik ke tempat semula. Komponen yang satu ini dijelaskan untuk menunjukkan hasil dari pengiriman message.
delete() Destruction	Menyatakan suatu objek mengakhiri hidup objek lain, arah panah mengarah pada objek yang diakhiri. Sebaiknya jika ada create maka ada destroy

Gambar 2.4. Simbol *Sequence diagram*

Komponen *Sequence diagram* sebagai berikut:

a. *Object* - adalah komponen berbentuk kotak yang mewakili sebuah *class* atau *object*. Mereka mendemonstrasikan bagaimana sebuah *object* berperilaku pada sebuah *system*.

b. *Activation boxes* - adalah komponen yang berbentuk persegi panjang yang menggambarkan waktu yang diperlukan sebuah *object* untuk menyelesaikan tugas. Lebih lama waktu yang diperlukan, maka *activation boxes* akan lebih panjang.

c. *Actors* - adalah komponen yang berbentuk *stick figure*. Komponen yang mewakili seorang pengguna yang berinteraksi dengan system.

d. *Lifeline* - adalah komponen yang berbentuk garis putus - putus. *Lifeline* biasanya memuat kotak yang berisi nama dari sebuah *object*. Berfungsi menggambarkan aktivitas dari *object*.

2.8.5 Deployment diagram

Diagram ini bersifat statis. Diagram ini memperlihatkan konfigurasi saat aplikasi dijalankan (saat *run time*). Dengan ini memuat simpul – simpul (node) beserta komponen – komponen yang ada di dalamnya. *Deployment diagram* berhubungan erat dengan diagram kompoen dimana *deployment diagram* memuat satu atau lebih komponen – komponen. Diagram ini sangat berguna saat aplikasi berlaku sebagai aplikasi yang dijalankan pada banyak mesin (*distributed computing*).

2.9 Metodologi Pengembangan Sistem

2.9.1 *Requirement* (Analisa Kebutuhan)

Analisis ini tahapan awal yang dilakukan dimana tahapan ini dilakukan untuk mengembangkan program yang akan dibuat. Dalam tahapan ini penulis mencari data pada koperasi, di mana pengumpulan data secara fisik ataupun *non* fisik.

2.9.2 *Design System* (Desain Sistem)

Dalam Pengembangan suatu perangkat lunak atau sistem pastinya tidak lepas dengan tahapan desain sistem atau sering disebut perancangan. Perancangan diperlukan untuk memberikan gambaran yang jelas dan rancang bangun yang lengkap sebagai pedoman bagi pemrogram dalam mengembangkan perangkat lunak guna menghasilkan suatu rancangan sistem yang baik. Desain sistem dapat diartikan bahwa suatu titik pemecahan masalah dengan melakukan penggambaran, perencanaan dan pembuatan sketsa dari beberapa elemen yang terpisah dalam satu kesatuan.

2.9.3 *Coding and Testing* (Penulisan Sinkode Program/Implementasi)

Dilakukan untuk pengujian terhadap program yang telah dibuat yang bertujuan untuk mengetahui kinerja pada program tersebut. Penulis melakukan pengujian program menggunakan teknik pengujian black box. Teknik pengujian ini berfokus pada fungsionalitas program.

2.9.4 Penerapan/Pengujian Program (*Integration and Testing*)

Dalam tahapan ini program diintegrasikan dan diuji sebagai sistem yang lengkap untuk menjamin kebutuhan sistem yang telah terpenuhi oleh program.

2.9.5 Pemeliharaan (*Operation and Maintenance*)

Pada tahap terakhir kita melakukan tahap maintenance atau pemeliharaan dan perbaikan terhadap program yang sudah dibuat.