REKOMENDASI BUKU DENGAN MENGGUNAKAN IMPLEMENTASI METODE *USER-BASED COLLABORATIVE FILTERING*

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan mencapai gelar Sarjana Komputer

Fanandi Prima Ratriansyah
NIM: 125150200111130

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG

2016

PENGESAHAN

REKOMENDASI BUKU DENGAN MENGGUNAKAN IMPLEMENTASI METODE *USER-BASED COLLABORATIVE FILTERING*

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan

memperoleh gelar Sarjana Komputer

Disusun Oleh:

Fanandi Prima Ratriansyah

NIM: 125150200111130

Skripsi ini telah diuji dan dinyatakan lulus pada

22 Desember 2016

Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Rekyan Regasari Mardi Putri, S.T, M.T	Agus Wahyu Widodo, S.T, M.Cs
NIK: 2011027704142001	NIP: 19740805 200112 1 001

Mengetahui

Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D

NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata di dalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan pasal 70).

Malang, 7 November 2016

Fanandi Prima Ratriansyah
NIM: 125150200111130

KATA PENGANTAR

Dengan menyebut nama Allah SWT Yang Maha Pengasih lagi Maha Penyayang. Puji syukur penulis panjatkan atas kehadirat Allah SWT karena berkat rahmat dan karunia-Nya penulis dapat menyelesaikan penyusunan skripsi berjudul "Rekomendasi Buku dengan Menggunakan Implementasi Metode *User-Based Collaborative Filtering*". Shalawat serta salam semoga senantiasa tercurahkan kepada Nabi Muhammad SAW, kepada keluarga, para sahabat dan umatnya hingga akhir zaman, amin. Penulisan skripsi ini diajukan untuk memenuhi salah satu syarat untuk mempeoleh gelar Sarjana pada Fakultas Ilmu Komputer Universitas Brawijaya (FILKOM UB). Dalam penyusunan dan penulisan skripsi ini tidak terlepas dari bantuan, bimbingan serta dukungan dari berbagai pihak. Oleh karena itu dalam kesempatan ini penulis dengan senang hati ingin menyampaikan terima kasih kepada:

1. Rekyan Regasari Mardi Putri, S.T, M.T selaku dosen pembimbing satu yang telah membimbing dan memberikan banyak masukan serta saran kepada penulis sehingga dapat menyelesaikan penulisan skripsi ini.
2. Agus Wahyu Widodo, S.T, M.Cs. selaku dosen pembimbing dua yang telah membimbing dan memberikan banyak masukan serta saran kepada penulis sehingga dapat menyelesaikan penulisan skripsi ini
3. Tri Astoto Kurniawan, S.T, M.T, Ph.D selaku Ketua Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya
4. Seluruh dosen FILKOM UB yang telah memberikan ilmunya kepada penulis selama masa menempuh studi di Fakultas Ilmu Komputer Universitas Brawijaya
5. Seluruh civitas akademik FILKOM UB yang telah memberikan bantuan selama penulis menempuh studi di Fakultas Ilmu Komputer Universitas Brawijaya
6. Pihak Perpustakaan yang telah membantu dengan memberikan data baik data buku, mahasiswa maupun transaksi yang digunakan penulis untuk *database*
7. Kedua orang tua penulis, Dondy Ariesdianto dan Titiek Sayekti, serta kedua adik penulis, Heral dan Aqsa dan seluruh keluarga besar penulis yang selalu memberikan dukungan dan doa untuk kelancaran skripsi penulis
8. Teman-teman kuliah anak-anak 11 Pria Tampan, teman mulai SD hingga kuliah satu jurusan, Claudia serta sahabat penulis Hersandi dan Haykal dan teman-teman lain yang tidak bisa disebutkan satu persatu, karena telah memberikan berbagai macam bantuan dan dukungan untuk menyelesaikan skripsi ini

Dengan segala kerendahan hati, penulis menyadari bahwa skripsi ini masih jauh dari sempurna. Oleh karena itu penulis sangat menerima bila ada masukan maupun kritikan agar skripsi ini dapat menjadikan penelitian ini

menjadi lebih baik. Bagi siapapun yang menjadikan skripsi ini referensi dan membutuhkan bantuan, dapat menghubungi penulis melalui email. Penulis berharap bahwa skripsi ini dapat bermanfaat bagi semua pihak termasuk penulis sendiri

Malang, 7 November 2016

Penulis

fanandiprimaratriansyah@gmail.com

ABSTRAK

Fanandi Prima Ratriansyah. 2016. Rekomendasi Buku dengan Menggunakan Implementasi Metode *User-Based Collaborative Filtering*. Program Studi Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya, Malang. Dosen Pembimbing: Rekyan Regasari Mardi Putri, S.T, M.T, Agus Wahyu Widodo, S.T, M.Cs

Daya baca masyarakat Indonesia secara keseluruhan termasuk rendah diantara negara-negara lain di dunia. *The World's Most Literate Nations* (WMLN) merilis daftar peringkat negara dengan daya baca tertinggi di dunia pada tahun 2016. Dari 61 negara yang di survey, Indonesia berada di peringkat 60, hanya setingkat dari Botswana, salah satu negara di Afrika. Bahkan dikalangan intelektual seperti mahasiswa, daya bacanya juga masih terhitung rendah. Salah satu faktornya adalah mahasiswa kekurangan referensi dan kesulitan untuk memilih mengenai buku apa yang sesuai dengan minat maupun kebutuhan mereka. Karena itu, terdapat beberapa upaya yang dapat dilakukan untuk membantu mahasiswa menemukan buku yang dibutuhkan, seperti dengan adanya rekomendasi buku. Rekomendasi bisa diberikan di katalog pencarian yang dimiliki oleh perpustakaan. Jika selama ini rekomendasi yang diberikan berdasarkan buku yang dipilih, maka penulis mencoba memberikan alternatif dengan memberikan rekomendasi berdasarkan kemiripan *user*. Metode yang sesuai dengan cara ini adalah *User-Based Collaborative Filtering*. Pada dasarnya, *User-Based Collaborative Filtering* memiliki asumsi, bahwa pengguna yang memiliki kesamaan di masa lampau, akan memiliki kesamaan di masa depan. Pengguna yang sama-sama meminjam buku tentang kedokteran, maka di masa depan akan meminjam buku kedokteran juga. Kecil kemungkinan mahasiswa kedokteran akan meminjam buku mengenai pemograman Java. Setelah kemiripan *user* ditemukan, barulah dihitung bobot buku-buku yang akan direkomendasikan. Berdasarkan hasil pengujian yang telah dilakukan, metode ini dapat memberikan rekomendasi jika kondisi lingkungannya ideal. Semakin ideal lingkungannya, semakin tinggi tingkat akurasi yang dihasilkan.

Kata Kunci: *User-Based Collaborative Filtering*, rekomendasi buku, perpustakaan

ABSTRACT

Literacy Indonesian society as a whole including the lowest among the other countries in the world. The World's Most Literate Nations (WMLN) released a ranking list of countries with the highest literacy in the world in 2016. Of the 61 countries surveyed, Indonesia ranked 60, just above that of Botswana, one of the countries in Africa. Even among intellectuals such as students, the power reading is still comparatively low. One factor for this problem is the lack of reference and difficulties students to choose the book what suits their interests and needs. Therefore, there are some efforts to be made to help students find the books that needed, like a book recommendations. Recommendations can be given in the catalog that is owned by a library searching systems. If during this recommendation given by the selected book, the author tries to provide alternatives to make recommendations based on user similarity. The method according to this method is the User-Based Collaborative Filtering. Basically, User-Based Collaborative Filtering assumes that users have in common in the past, will have a common future. Users were equally borrow books on medicine, then the future will borrow medical books as well. Medical students will be less likely to borrow books on Java programming. Once the user similarity is found, then calculated the weight of the books that will be recommended. Based on the results of testing that has been done, this method can provide a recommendation if environmental conditions are ideal. The more ideal environment, the higher the level of accuracy that is generated.

Keywords: *User-Based Collaborative Filtering, book recommendation, library*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR KODE PROGRAM

BAB 1 PENDAHULUAN

Pada bab ini akan menjelaskan apa yang akan dikerjakan dalam penelitian ini. Selain itu pada bab ini juga menjelaskan mengapa penelitian ini perlu dikerjakan

1.1. Latar Belakang

Daya baca masyarakat Indonesia bisa dibilang cukup rendah. Hal ini terlihat dari daftar peringkat negara dengan daya baca tertinggi di dunia yang dirilis oleh *The World's Most Literate Nations* (WMLN) pada tahun 2016. Dari 61 negara yang di survei, Indonesia berada di peringkat 60 dan hanya setingkat di atas Botswana, salah satu negara di Afrika. Indonesia berada di bawah negara ASEAN lainnya seperti Thailand (59), Malaysia (53) dan Singapura (36). Sedangkan peringkat pertama diduduki oleh Finlandia. Rendahnya daya baca masyarakat dapat diindikasi melalui dua hal (Kriswijayanti, 2009). Pertama adalah rendahnya jumlah buku yang terbit di Indonesia. Jika di Inggris bisa mencapai 100.000 judul per tahun, maka Indonesia hanya bisa menerbitkan antara 5.000 hingga 10.000 buku saja. Bahkan jika dibandingkan dengan negara tetangga seperti Malaysia, Indonesia masih kalah karena Malaysia bisa menerbitkan sekitar 15.000 buku setiap tahun. Tentu itu jumlah yang besar karena penduduk Malaysia jauh lebih sedikit dari penduduk Indonesia. Kedua, adalah pilihan masyarakat untuk mendapat informasi melalui media cetak (23,5%) kalah dengan televisi (85,9%) dan radio (40,3%). Bahkan untuk kalangan mahasiswa yang tergolong kaum intelektual pun daya bacanya masih rendah. Hal ini ditunjukkan oleh rendahnya transaksi yang dilakukan di perpustakaan yang ada di universitas. Di perpustakaan Universitas Brawijaya yang memiliki kurang lebih 60.000 mahasiswa, hanya ada sekitar 5.000-10.000 trasaksi per bulannya.

Banyak faktor yang melatarbelakangi rendahnya daya baca mahasiswa, seperti sistem pembelajaran di Indonesia yang kurang membuat mahasiswanya harus mencari informasi melalui buku, rasa malas membaca buku, kurang nyamannya perpustakaan yang tersedia dan kesulitan untuk menemukan buku yang tepat. Untuk alasan kurangnya sistem pembelajaran yang mengharuskan mahasiswa mencari informasi melalui buku, bisa diatasi dengan mewajibkan mahasiswa untuk melakukan citasi dari buku, bukan dari internet. Alasan malas membaca bisa diatasi dengan sosialisasi baik dari pemerintah maupun komunitas mengenai pentingnya membaca untuk bersaing dengan dunia global. Kurang nyamannya perpustakaan dapat diatasi dengan memperbaiki fasilitas yang dimiliki oleh perpustakaan, meskipun perpustakaan di Indonesia menduduki peringkat yang cukup tinggi di dunia, yakni peringkat 35, di atas negara-negara seperti Malaysia, Jerman, Singapura dan Selandia Baru. Lalu alasan mahasiswa kekurangan referensi mengenai buku apa yang sesuai dengan minat maupun kebutuhan mereka, dapat diatasi dengan adanya rekomendasi buku yang dapat membantu untuk menemukan referensi apa yang dibutuhkan. Masalah kesulitan

mencari buku terjadi karena beberapa faktor, seperti buta akan apa kebutuhan mereka, tidak mengetahui buku mana yang lebih baik untuk dijadikan referensi dan tidak tahu buku mana yang sesuai dengan topik permasalahan yang ingin dipecahkan. Masalah inilah yang diangkat oleh penulis dengan melakukan implementasi suatu metode untuk diterapkan pada rekomendasi buku. Rekomendasi bisa diletakkan di katalog pencarian yang dimiliki oleh perpustakaan. Jadi, *user* akan diberikan rekomendasi sesuai buku yang ia pilih ketika melakukan pencarian. Sebagai contoh, seorang *user* mencari buku dengan *keyword* Soeharto. Nantinya akan muncul beberapa buku dengan judul yang memiliki unsur kata Soeharto. Ketika *user* memilih satu buku, nanti di bawah ilustrasi buku akan muncul beberapa rekomendasi yang terkait dengan buku tersebut. Dengan mendapatkan literatur lain maka diharapkan *user* terbantu mendapatkan buku yang dibutuhkan.

Ada beberapa penelitian yang pernah dilakukan pada studi kasus seperti ini (Tsuji, 2014) (Uyun 2011). Pada penelitian yang pertama, parameter yang digunakan disini adalah catatan peminjaman yang dilakukan di Universitas T, kemiripan judul buku, kecocokan *Nippon Decimal Classification* (NDC) dan kemiripan *outline* dari sebuah buku di database. Penelitian ini menggunakan 32 mahasiswa Universitas T sebagai subyek penelitian. Metode yang digunakan adalah *Support Vector Machine* (SVM). Terdapat 4 kombinasi dalam menentukan rekomendasi buku, dimana tiga diantaranya memiliki tingkat akurasi yang lebih tinggi dari tingkat akurasi jika hanya menggunakan satu data saja. Ketiga kombinasi tersebut adalah NDC + Judul + Catatan Peminjaman (71.9%), NDC + Judul + Catatan Peminjaman + Database Buku (65.7%) dan Judul + Catatan Peminjaman (63.6%). Lalu pada penelitian yang kedua, parameter yang digunakan disini adalah kode buku, kode pelanggan yang telah terdaftar, angka I (jika pelanggan membeli buku) dan O (jika pelanggan tidak membeli buku), angka superskrip 1-5 yang merupakan rating dari pengguna dan skala rating mulai angka 1 (sangat tidak suka) hingga 5 (sangat suka). Menggunakan metode *Item Collaborative Filtering*, tidak disebutkan berapa tingkat akurasi dari penelitian ini.

Berdasarkan penelitian-penelitian yang telah disebutkan, penulis akan melakukan penelitian dengan judul "Rekomendasi Buku dengan Menggunakan Implementasi Metode *User-Based Collaborative Filtering*". Apabila di perpustakaan sering kali memberikan rekomendasi berdasarkan buku (*item*), maka disini akan diberikan alternatif. Alternatif tersebut adalah dengan memberikan rekomendasi dari sisi *user*, yakni berdasarkan kemiripan antar *user*. Metode *User-Based Collaborative Filtering* memiliki asumsi bahwa *user* yang memiliki kesamaan di masa lampau, akan memiliki kesamaan di masa depan pula. Penelitian ini nantinya akan menyajikan beberapa rekomendasi buku berdasarkan pencarian di katalog perpustakaan yang dilakukan oleh *user* perpustakaan. Adapun alasan penggunaan metode *User-Based Collaborative Filtering* adalah perhitungan kemiripan berdasarkan pengguna. Jadi, buku yang direkomendasikan adalah buku-buku yang pernah dipinjam oleh *user* yang memiliki kemiripan. Misalkan *user* x pernah meminjam buku 1 dan buku 2, maka sistem akan mencarikan *user-user* lain yang pernah meminjam buku 1 dan buku 2, lantas

mencarikan buku lain yang pernah dipinjam oleh *user-user* tersebut. Nantinya buku-buku tersebutlah yang akan direkomendasikan. Implementasi rekomendasi dengan metode seperti ini cocok untuk diterapkan di perpustakaan Universitas Brawijaya yang mayoritas peminjamnya dari mahasiswa berbagai fakultas dan jurusan, sehingga seorang mahasiswa dapat mendapatkan rekomendasi berdasarkan kemiripan dengan mahasiswa yang satu jurusan atau satu fakultas. Selain itu, metode *collaborative filtering* baik *user* maupun *item-based* dapat menerima *feedback* yang membuat metode ini dapat memprediksi keinginan dari *user.* Kualitas rekomendasi yang diberikan dari metode ini juga tinggi karena membandingkan dengan semua *user* yang ada.

1.2. Rumusan Masalah

Judul:

Rekomendasi Buku dengan Menggunakan Implementasi Metode *User-Based Collaborative Filtering*

Pertanyaan Penelitian:

1. Bagaimana metode *User-Based Collaborative Filtering* pada rekomendasi buku dapat diimplementasikan pada sistem katalog pencarian perpustakaan Universitas Brawijaya?
2. Bagaimana kondisi-kondisi yang dibutuhkan agar metode *User-Based Collaborative Filtering* dapat berjalan dengan baik?
3. Berapa akurasi dari penerapan metode *User-Based Collaborative Filtering* pada rekomendasi buku di sistem katalog pencarian perpustakaan Universitas Brawijaya?

1.3. Tujuan

Tujuan umum:

Tujuan secara umum dari penelitian ini adalah untuk membuat sebuah rekomendasi buku dengan mengimplementasikan metode *User-Based Collaborative Filtering* pada sistem katalog pencarian perpustakaan Universitas Brawijaya.

Tujuan khusus:

1. Menerapkan metode *User-Based Collaborative Filtering* pada rekomendasi buku di sistem katalog pencarian perpustakaan Universitas Brawijaya
2. Menentukan kondisi-kondisi yang dibutuhkan agar metode *User-Based Collaborative Filtering* dapat berjalan dengan baik
3. Menguji hasil pengujian akurasi dari penerapan metode *User-Based Collaborative Filtering* pada rekomendasi buku di sistem katalog pencarian perpustakaan Universitas Brawijaya

1.4. Manfaat

Manfaat yang dapat diperoleh dari penelitian ini adalah:

1. Dapat membantu mahasiswa agar dapat menemukan buku yang sesuai dengan kebutuhan
2. Meningkatkan transaksi peminjaman yang terjadi di perpustakaan karena adanya rekomendasi yang dapat menjadi stimulus kepada mahasiswa untuk membaca buku lebih banyak

1.5. Batasan Masalah

Berdasarkan latar belakang dan rumusan masalah yang telah disebutkan di atas, maka batasan-batasan yang dimiliki dari penelitian ini adalah:

1. Data-data yang terdapat dalam *database* adalah data-data yang telah diberikan dan disetujui oleh pihak perpustakaan Universitas Brawijaya
2. Data yang digunakan telah dinormalisasi sesuai kebutuhan penulis
3. Data *user* yang digunakan hanya data mahasiswa, tidak termasuk data dosen dan karyawan
4. *Input* dari sistem adalah Nomor Induk Mahasiswa (NIM) dari mahasiswa dan data buku yang dicari oleh mahasiswa melalui katalog perpustakaan
5. *Input* untuk setiap perhitungan adalah satu buku, yaitu buku yang dipilih dari hasil pencarian
6. Parameter yang digunakan untuk melakukan perhitungan rekomendasi hanya data transaksi *user* yang berisikan data mahasiswa (NIM dan nama) dan data buku (*register* dan *title*)

1.6. Sistematika Pembahasan

Sistematika pembahasan digunakan untuk menguraikan laporan secara umum yang dibagi menjadi beberapa bab, yaitu:

BAB I PENDAHULUAN

Bab ini berisi latar belakang, rumusan masalah, tujuan, manfaat, batasan masalah dari rekomendasi buku berdasarkan hasil pencarian *user* di katalog perpustakaan menggunakan implementasi metode *User-Based Collaborative Filtering*

BAB II LANDASAN KEPUSTAKAAN

Bab ini membahas penelitian-penelitian lain yang berkaitan dengan penelitian ini. Selain itu pada bab ini juga akan membahas dasar teori dan referensi baik yang digunakan maupun yang terkait dengan penyusunan penelitian yang sedang dilakukan

BAB III METODOLOGI

Bab ini menjelaskan metode dan alur yang digunakan dalam melakukan penelitian

BAB IV PERANCANGAN

Bab ini menjelaskan gambaran dari perancangan implementasi yang akan dibuat. Perancangan terdiri dari percancangan algoritma, perancangan *database* dan perancangan antarmuka

BAB V IMPLEMENTASI

Bab ini merupakan implementasi dari perancangan implementasi dari bab sebelumnya. Implementasi terdiri dari tiga bagian, yakni implementasi algoritma, implementasi antarmuka dan implementasi *database*.

BAB VI ANALISA DAN PENGUJIAN

Bab ini menguraikan hasil pengujian implementasi berdasarkan perancangan dari bab sebelumnya. Selanjutnya dilakukan analisis dari hasil pengujian tersebut, apakah sudah sesuai dengan yang diharapkan dan dapat menjawab rumusan masalah

BAB VII PENUTUP

Bab ini berisi kesimpulan dan saran baik terhadap implementasi maupun hasil dari penelitian

BAB 2 LANDASAN KEPUSTAKAAN

Bab ini akan menjabarkan kajian pustaka dari penelitian-penelitian sebelumnya yang memiliki keterkaitan dengan penelitian ini. Selain itu pada bab ini juga membahas mengenai dasar-dasar teori yang digunakan untuk menunjang penulisan penelitian ini.

2.1. Kajian Pustaka

Kajian pustaka merupakan penjabaran dari penelitian-penelitian sebelumnya yang terkait dengan rekomendasi buku berdasarkan hasil pencarian di katalog perpustakaan dengan implementasi metode *User-Based Collaborative Filtering*. Tujuan dari adanya kajian pustaka ini adalah sebagai referensi sekaligus perbandingan untuk membantu proses dari pengerjaan penelitian ini. Tabel 2.1 berikut merupakan referensi-referensi yang digunakan penulis pada penelitian ini.

Tabel 2.1 Kajian Pustaka

No	Judul	Penulis	Perbandingan	
			Kajian Pustaka	Skripsi Penulis
1	*Book Recommendation Based on Library Loan Records and Bibliographic Information*	Keita Tsuji, Nobuya Takizawa, Sho Sato, Ui Ikeuchi, Atsushi Ikeuchi, Fuyuki Yoshikane, Hiroshi Itsumura	Rekomendasi buku berdasarkan data peminjamanan dan menggunkaan metode *Support Vector Machine* (SVM)	Rekomendasi buku berdasarkan pencarian pengguna dan menggunakan metode *User-Based Collaborative Filtering*
2	*OWA based Book Recommendation Technique*	Shahab Saquib Sohail, Jamshed Siddiqui, Rashid Ali	Menggunakan Metode *Ordered Weighted Averaging (OWA)*	Menggunakan metode *User-Based Collaborative Filtering*
3	*Item Collaborative Filtering* untuk Rekomendasi Pembelian Buku Secara Online	Shofwatul 'Uyun. Imam Fahrurrozi, Agus Mulyanto	Digunakan pada toko buku *online* dan metode yang digunakan adalah *Item Collaborative Filtering*	Digunakan pada perpustakaan dan metode yang digunakan adalah *User-Based Collaborative Filtering*
4	Pengembangan Sistem Rekomendasi Penelusuran Buku dengan Penggalian *Association Rule*	Nugorho Wandi, Rully A. Hendrawan,	Rekomendasi penelusuran buku dan menggunakan	Rekomendasi buku berdasarkan hasil pencarian pengguna dan menggunakan metode *User-Based*

	Menggunakan Algoritma *Apriori*	Ahmad Mukhlason	metode *Association Rule*	*Collaborative Filtering*
5	Sistem Rekomendasi Buku Online dengan Metode *Collaborative Filtering*	Moh. Irfan, Andharini Dwi C. Fika Hastarita R.	Rekomendasi buku *online* dan menggunakan metode *Collaborative Filtering*	Rekomendasi buku di katalog perpustakaan dan menggunakan metode *User-Based Collaborative Filtering*

2.2 Buku dan Perpustakaan

Menurut Kamus Besar Bahasa Indonesia (KBBI), buku memiliki definisi "lembar kertas yang berjilid yang, berisi tulisan maupun kosong". Sejarah buku sendiri memang berawal dari sebuah kertas, yang terjadi pada ratusan tahun lalu, ketika Tsai Lun yang berasal dari Cina menciptakan kertas pada tahun 200 sebelum masehi (Hart, 1992). Kemudian sekitar tahun 1450 seorang penemu asal Jerman Johannes Gutenberg menemukan alat untuk mencetak buku (Hart, 1992). Konstribusi kedua orang tersebut membuat Michael M. Hart menempatkan Tsai Lun berada di peringkat 7 dan Johaness Gutenberg peringkat 8 dari 100 orang yang paling berpengaruh di dunia.

Perpustakaan adalah salah satu tempat yang bisa kita gunakan untuk menemukan buku. Dengan semakin meningkatkan kesadaran akan pentingnya membaca, kini setiap daerah memiliki perpustakaannya sendiri. Belum lagi adanya sekolah dan universitas yang memiliki perpustakaan untuk menunjang pembelajaran para siswa dan mahasiswanya. Terlebih lagi perpustakaan di universitas, yang memilki banyak sekali pengguna karena besarnya universitas tersebut. Tentu dengan banyaknya, tingkat transaksi yang terjadi terbilang tinggi, meskipun mungkin belum mencapai potensi maksimalnya.

Sebagai contoh di Universitas Brawaijaya, dengan mahasiswanya yang mencapai 60.000, tingkat transaksi peminjamannya hanya berkisar antara 5.000-10.000 per bulan. Artinya, hanya ada satu dari enam hingga 12 mahasiswa yang melakukan peminjaman di perpustakaan tiap bulannya. Padahal, bisa jadi dari 5.000 transakasi terdapat satu mahasiswa yang meminjam beberapa buku.

Salah satu upaya untuk meningkatkan hal tersebut adalah dengan adanya sistem rekomendasi. Jika rekomendasi yang sekarang berdasarkan buku (*item*), maka disini akan diberikan alternatif dimana rekomendasi diberikan berdasarkan kemiripan antar *user*. Nantinya implementasi ini akan ditampilkan dalam *Graphical User Interface* (GUI) dengan menggunakan bahasa pemograman *Java*.

2.2 *Dewey Decimal Classification* (DDC)

Buku memiliki beragam *genre* sehingga dibutuhkan klasifikasi untuk membagi *genre-genre* tersebut. Banyak cara dalam melakukan klasifikasi buku.

Salah satunya adalah pengelompokkan berdasarkan subyek. *Dewey Decimal Classification* (DDC) adalah salah satu sistem klasifikasi fundamental yang dapat melakukan hal tersebut. Pada pengembangannya, simbol notasi pada sistem klasifikasi ini menggunakan sistem desimal angka Arab (Subrata, 2009). Terdapat tiga komponen yang penting dalam sistem klasifikasi ini, yaitu bagan (*schedules*), indeks relatif dan tabel-tabel. Bagan adalah penggunaan bilangan desimal untuk membagi semua bidang ilmu pengetahuan menjadi sepuluh kelas utama. Bidang ilmu tersebut tertulis pada Tabel 2.2.

Tabel 2.2 Klasifikasi *Dewey Decimal Number* (DDC)

Kode	Keterangan
000	Komputer, Informasi dan Referensi Umum
100	Filsafat dan Psikologi
200	Agama
300	Ilmu Sosial
400	Bahasa
500	Sains dan Matematika
600	Teknologi
700	Kesenian dan Rekreasi
800	Sastra
900	Sejarah dan Geografi

Dari sepuluh kelas utama tersebut, nantinya masih akan terbagi-bagi lagi menjadi sub kelas untuk mengklasifikasikan suatu buku. Sebagai contoh, buku dengan tiga digit awal DDC-nya 901 menandakan bahwa buku tersebut termasuk buku filsafat dan teori sejarah.

Pada penelitian ini, DDC akan digunakan untuk melakukan proses pengujian. Caranya, hasil rekomendasi yang keluar akan dibandingkan, apakah memiliki tiga digit DDC awal yang sama dengan buku yang dipilih sebagai *input*. Selain itu juga akan dilakukan perbandingan dengan satu digit awal buku yang merupakan kelas utama dari buku tersebut. Semakin banyak buku hasil rekomendasi yang satu kelas dengan buku yang dipilih, maka semakin tinggi pula akurasi yang dihasilkan.

2.3 Sistem Rekomendasi

Sistem rekomendasi atau *Recommender Systems* (RSs) adalah teknik dan perangkat lunak yang menyediakan sebuah saran atau anjuran *item* yang akan digunakan atau dipilih oleh *user* (Ricci, 2011). Saran-saran yang dipilihkan

tergantung dari proses *decision-making*, seperti *item* apa yang dibeli, film apa yang dilihat atau *software* apa yang digunakan.

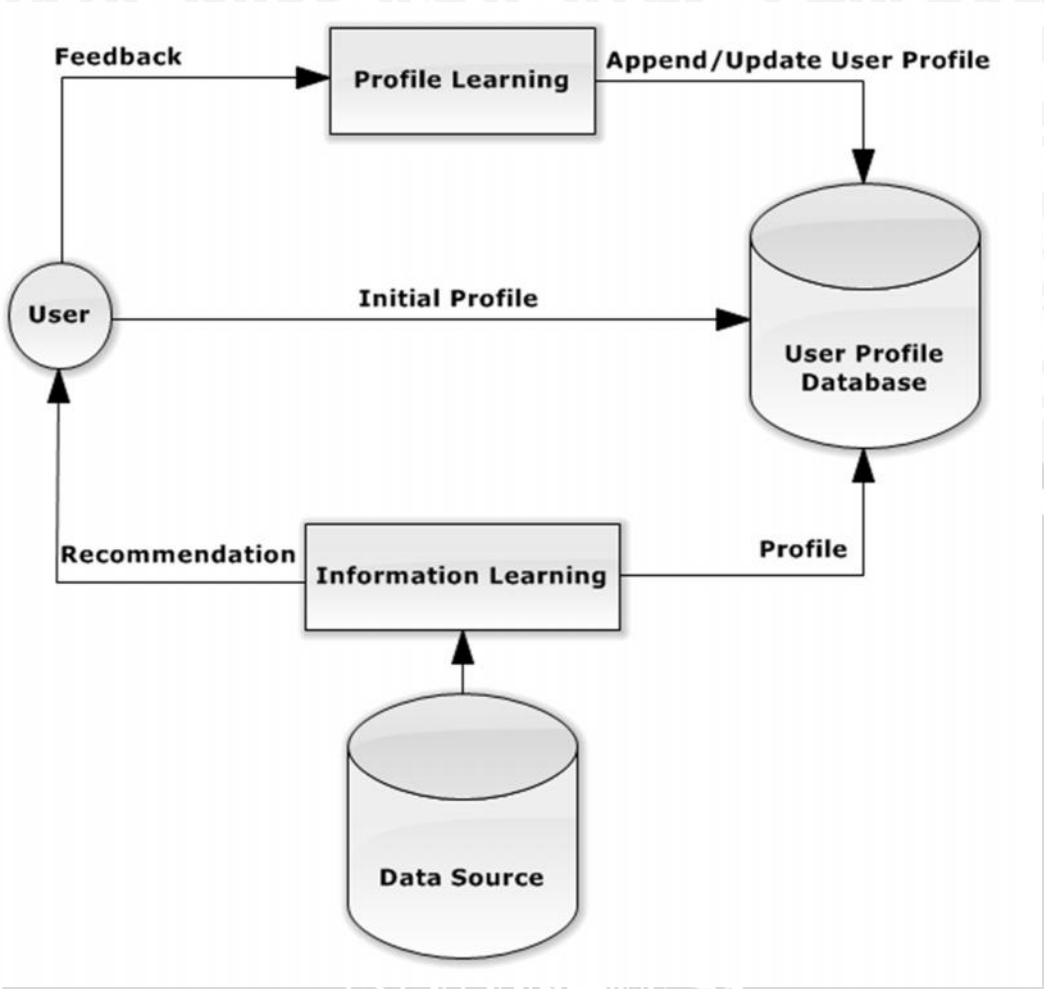
Item sendiri merupakan istilah umum untuk menunjukkan apa yang akan direkomendasikan oleh sistem. Pada umumnya, *item* yang disarankan oleh sistem rekomendasi berfokus pada satu *item* (seperti lagu, buku). Dalam bentuk yang paling sederhana, rekomendasi yang diberikan oleh sistem dapat berupa daftar *item* yang diurutkan berdasarkan peringkat. Tanpa disadari, sistem rekomendasi telah dilakukan dalam rutinitas sehari-hari. Misalkan seorang *user* ingin membeli sebuah laptop, tentu *user* tersebut akan meminta orang lain yang paham tentang laptop untuk memberikan rekomendasinya.

Dalam upaya meniru perilaku tersebut, algoritma pertama yang digunakan untuk membuat sistem rekomendasi adalah algoritma yang dihasilkan oleh komunitas pengguna terhadap pengguna aktif (pengguna yang membutuhkan rekomendasi). Rekomendasi diberikan berdasarkan kesamaan pengguna, atau kesamaan selera terhadap suatu *item*. *Item* yang direkomendasi melalui pendekatan seperti ini disebut dengan *collaborative filtering*.

Pada sebuah sistem rekomendasi terdapat tiga komponen, yakni *Input/Output*, Metode Rekomendasi dan Desain Rekomendasi (Masruri, 2007). *Input* yang dikomputasi oleh sistem rekomendasi merupakan hasil kombinasi dengan *input* dari *user-user* lain dan menghasilkan *output* berupa rekomendasi atau prediksi. Lalu terdapat beberapa metode yang dapat digunakan untuk memberikan rekomendasi, seperti secara manual, pendekatan statistik, menghitung kedekatan antar *user* (*User-based*) maupun antar *item* (*Item-based*). Sedangkan pada desain rekomendasi terbagi menjadi dua hal, yakni bagaimana rekomendasi ditampilkan dan bagaimana sifat dari rekomendasi atau tingkat personalisasinya. Rekomendasi dapat ditampilkan melalui 3 cara, yakni *push* (secara otomatis tanpa permintaan dari *user*), *pull* (harus ada permintaan dari *user*) dan pasif (menampilkan item lain yang memiliki hubungan dengan item yang dilihat pada saat itu juga). Sedangkan tingkat personalisasi rekomendasi dibagi menjadi dua macam, yakni *personalized* (rekomendasi yang diberikan tergantung profil *user*) dan *non personalized* (rekomendasi yang diberikan bersifat umum untuk semua profil *user*). Pada penelitian ini, rekomendasi ditampilkan dengan cara *pull* dan tingkat personalisasinya adalah *personalized*.

Pada penelitian ini, fokus dari sistem rekomendasi adalah memberikan suatu rekomendasi buku. Rekomendasi diberikan berdasarkan pencarian *user* di sistem katalog perpustakaan. Sebelum melakukan pencarian, *user* harus memasukkan Nomer Induk Mahasiswa (NIM) agar sistem mengetahui jurusan dari mahasiswa tersebut. Hal ini penting, mengingat sifat *User-Based Collaborative Filtering* yang membuat matriks dari keseluruhan data. Dengan menyaring hanya *user* yang hanya satu jurusan, hal ini akan menyingkat proses perhitungan. Setelah itu, *user* akan memilih salah satu buku dari hasil pencarian. Ketika *user* sudah memilih satu buku, buku tersebut akan dianggap sebagai *input* yang bersifat sementara dan sistem akan mulai melakukan proses perhitungan. Nantinya *output* dari hasil perhitungan akan muncul di bawah detail buku.

Sedangkan arsitektur sistem rekomendasi secara umum seperti yang terlihat pada Gambar 2.1.



Gambar 2.1 Arsitektur Sistem Rekomendasi

(Sumber: Putra, 2015)

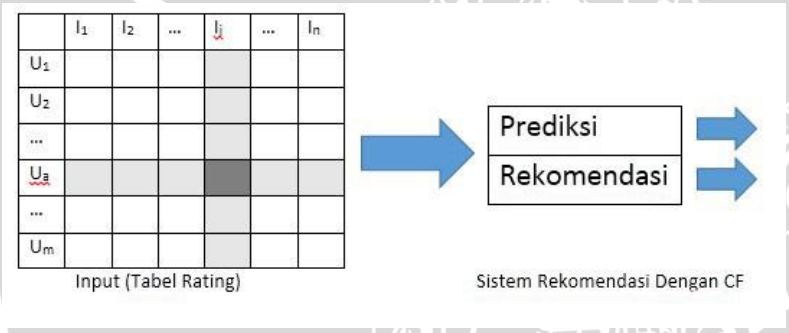
Dari Gambar 2.1 dapat ditarik kesimpulan bahwa sebuah sistem rekomendasi terdiri dua proses, yakni *Profile Learning* dan *Information Learning*. Pada *Profile Learning* sistem akan menerima *feedback* dari *user* yang digunakan untuk meng-*update database* profil *user*. Dari *database* ini proses *Information Learning* akan mempelajari data dan mencocokkan dengan *data source* untuk memberikan rekomendasi kepada *user*. Pada penelitian ini, *data source*-nya adalah data buku dan data transaksi. *User profile database*-nya adalah data mahasiswa. Sedangkan proses *profile learning* akan melihat buku apa saja yang pernah dipinjam oleh *user*. Nantinya data ini akan diproses pada *information learning* untuk mencari kemiripan antar *user* dan menentukan buku yang akan direkomendasikan.

2.4 *Collaborative Filtering*

Collaborative Filtering (CF) adalah sebuah metode yang mengumpulkan informasi dari *user-user* yang direpresentasikan dalam bentuk nilai peringkat dan digunakan untuk membuat suatu prediksi atau rekomendasi otomatis (Masruri, 2007). CF digunakan untuk memperkirakan ketertarikan atau selera seorang *user* terhadap suatu item tertentu. Jika ditarik garis besar, terdapat dua proses yang dilakukan dalam CF:

1. Mencari *user* lain yang memiliki kemiripan pola peringkat dengan *user* yang akan diberikan rekomendasi
2. Menggunakan nilai peringkat dari *user* lain yang diperoleh dari langkah satu tersebut untuk menghitung rekomendasi untuk *user* aktif

Rekomendasi sendiri memiliki definisi sebagai suatu daftar yang berisikan X *item* yang mempunyai kemungkinan terbesar akan disukai oleh *user* aktif dan *item* tersebut belum pernah diberi peringkat oleh *user* aktif. Namun pada dasarnya dalam memberikan suatu rekomendasi dibutuhkan suatu prediksi untuk menghitung tingkat ketertarikan *user* aktif terhadap item-item tersebut.



P_{aj} (Prediksi nilai *rating item* j untuk *user* aktif a)
$\{T_{i1}, T_{i2}, \ldots, T_{in}\}$ Top-N *item* untuk *user* aktif a

Gambar 2.2 Arsitektur *Collaborative Filtering*

(Sumber: Sarwar, 2001)

Secara umum, CF dibagi menjadi dua macam kategori, yaitu *User-based Collaborative Filtering* dan *Item-based Collaborative Filtering*.

2.4.1 *User-Based Collaborative Filtering*

Disebut juga *Memory-based Collaborative Filtering*. Algoritma ini menggunakan semua data peringkat untuk membuat suatu prediksi. Prediksi atau rekomendasi yang dihasilkan oleh metode ini adalah melalui kemiripan antar *user*. Sistem ini menggunakan teknik statistik untuk menentukan satu set pengguna (disebut *neighbors*) yang memiliki kesamaan dengan target pengguna (Sarwar, 2001). Setelah lingkungan terbentuk, metode ini menggunakan algoritma yang berbeda untuk menggabungkan preferensi *neighbors* untuk menghasilkan prediksi atau top N rekomendasi untuk pengguna aktif.

Pada dasarnya, *User-Based Collaborative Filtering* memiliki asumsi, bahwa pengguna yang memiliki kesamaan di masa lampau, akan memiliki kesamaan di

masa depan. Pengguna yang sama-sama meminjam buku tentang kedokteran, maka di masa depan akan meminjam buku kedokteran juga. Kecil kemungkinan mahasiswa kedokteran akan meminjam buku mengenai pemograman Java. Meskipun ada, hal-hal seperti ini akan teratasi oleh metode *User-Based Collaborative* Filtering karena banyaknya data yang dihitung. Dengan kata lain, *User-Based Collaborative Filtering* memiliki kestabilan akan *user preferences* dari waktu ke waktu (Jannach, 2011). Oleh karena itu, sistem dengan metode seperti ini cocok untuk diterapkan di perpustakaan Universitas Brawijaya karena peminjamnya adalah mahasiswa yang berasal dari beragam jurusan dan fakultas. Sebagai contoh, pada Tabel 2.3 akan ditunjukkan contoh dari data *rating* pengguna.

Tabel 2.3 Contoh Data Rating Pengguna

User\Item	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	0	1	0	0	1
User 2	0	0	1	1	1
User 3	1	1	0	0	0
User 4	1	0	1	0	1
User 5	1	1	1	X	X

Pada Tabel 2.3 terdapat *user* 1 hingga *user* 5 dan *item* 1 hingga *item* 5. *Item* 4 dan *item* 5 pada *user* 5 bertanda X untuk menunjukkan bahwa manakah dari kedua *item* tersebut yang lebih cocok untuk direkomendasikan kepada *user* 5.

Sebelum melakukan perhitungan kemiripan, perlu menghitung *Mean Adjusted Score* terlebih dahulu yang digunakan untuk mempermudah visualisasi perhitungan (Putra, 2015). Hal ini dilakukan karena kalkulasi akan susah dilakukan apabila angka yang digunakan hanya angka 0 dan 1. Setelah proses perhitungan *Mean Adjusted Score* selesai, akan dilakukan *transpose* matriks agar dapat dihitung dengan menggunakan *Person's Correlation Coefficient*. Perhitungan *Mean Adjusted Score* akan ditunjukkan oleh Persamaan 2.1 (Putra, 2015).

$$M_{(1,A)} = a_{(1,A)} - \frac{\sum a_{(1,n)}}{n}, \text{ dimana} \quad (2.1)$$

$M_{(1,A)}$ = *Mean adjusted score* dari mahasiswa 1 terhadap buku A

$a_{(1,A)}$ = Nilai buku A dari mahasiswa 1

$a_{(1,n)}$ = Nilai buku A dari seluruh mahasiswa berjumlah n

n = Jumlah buku yang ada

Cara menghitung kemiripan antar *user* dapat menggunakan *Person's Correlation Coefficient*, seperti yang akan ditunjukkan pada Persamaan 2.2 (Putra, 2015).

$$sim(a,b) = \frac{\sum_{i \in I}(R_{a,i} - \overline{R}_a)(R_{b,i} - \overline{R}_b)}{\sqrt{\sum_{i \in I}(R_{a,i} - \overline{R}_a)^2} \sqrt{\sum_{i \in I}(R_{b,i} - \overline{R}_b)^2}}$$, dimana (2.2)

sim(a,b) = similaritas *preference user a* dan *b*

$\overline{R}_a$ = rata-rata peringkat dari *user a*

$R_{a,i}$ = peringkat *user a* terhadap *item i*

Persamaan 2.2 menghasilkan nilai similaritas antara -1 dan +1. Korelasi negatif ditunjukkan oleh -1 dan korelasi positif ditunjukkan oleh +1. Hal ini menunjukkan adanya tingkat kesamaan *preferences* yang tinggi di antara kedua pengguna (Kwon, 2009). Sedangkan untuk menghitung prediksi nilai rating dapat menggunakan metode W*eighted Sum* seperti yang ditunjukkan oleh Persamaan 2.3 (Putra, 2015).

$$P_{u,i} = \frac{\sum_{allsimiliar\,users,N}(S_{u,N} * R_{N,i})}{\sum_{allsimiliar\,users,N}(S_{u,N})}$$, dimana (2.3)

$P_{u,i}$ = prediksi peringkat dari *user u* terhadap *item i*

$S_{u,N}$ = similaritas *user u* dengan *item N*

$R_{N,i}$ = peringkat *user N* terhadap *i*

2.5 *Mean Absolute Error* (MAE)

Sistem rekomendasi memiliki beberapa tipe pengujian untuk mengevaluasi kualitas dari sistem rekomendasi yang telah dibuat. Secara garis besar terdapat dua tipe pengujian, yakni *Statistical Accuracy Metrics* dan *Decision Support Accuracy Metrics* (Sarwar, 2001). Tipe pertama mengevaluasi akurasi dari sistem dengan membandingkan skor rekomendasi numeral dengan peringkat yang diberikan oleh user untuk pasangan *user-item* di dataset. Sedangkan tipe kedua mengevaluasi bagaimana efektivitas sebuah mesin prediksi membantu *user* untuk memilih *item* dengan kualitas tinggi dari seluruh *item* yang ada.

Mean Absolute Error (MAE) termasuk ke dalam *Statistical Accuracy Metrics*. MAE berupa matriks yang luas antara peringkat dan prediksi. MAE menghitung penyimpangan dari rekomendasi terhadap value *user-spesified* yang benar. Untuk setiap pasangan *rating*-prediksi <p_i, q_i> matriks ini memperlakukan *error* absolut diatara peringkat-prediksi | p_i, q_i |. MAE dihitung dari penjumlahan pertama dari *error* absolut tersebut dan N pasangan peringkat-prediksi yang memiliki hubungan dan menghitung rata-ratanya. Persamaannya dituliskan pada Persamaan 2.4 (Putra, 2015):

$$Akurasi = \frac{Nm}{N} \times 100\% \quad (2.4)$$

N_m = Banyaknya hasil rekomendasi yang cocok dengan data uji

N = Banyaknya data uji

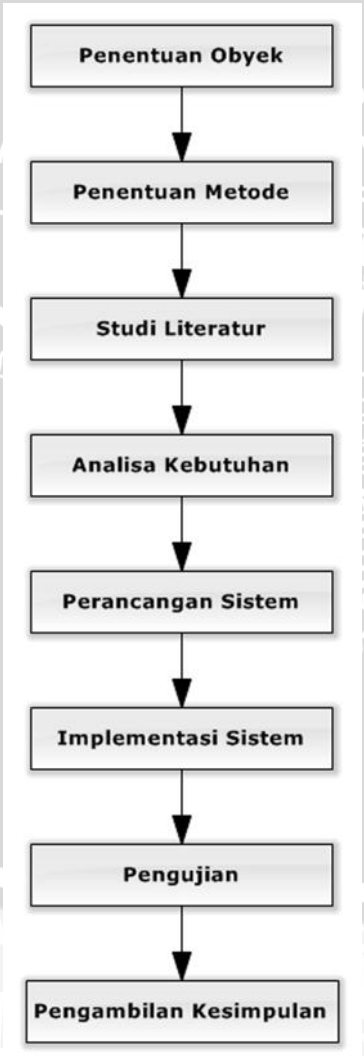
Pada penelitian ini, banyaknya hasil rekomendasi yang cocok adalah buku hasil rekomendasi yang memiliki digit DDC yang sama dengan buku yang dipilih. Sedangkan banyaknya data uji adalah keseluruhan data yang diuji pada tahap pengujian.

BAB 3 METODOLOGI

Bab ini akan membahas tahapan-tahapan yang diterapkan untuk mengimplementasikan metode *User-Based Collaborative Filtering* pada rekomendasi buku.

3.1. Tahapan Penelitian

Tahapan penelitian menjelaskan mengenai langkah-langkah yang harus ditempuh untuk menyelesaikan penelitian ini. Tahapan tersebut dimulai dari penentuan obyek, penentuan metode, studi literatur, analisa kebutuhan, perancangan sistem, implementasi sistem, pengujian dan pengambilan kesimpulan. Alur dari metode penelitan yang diterapkan terdapat pada Gambar 3.1.



Gambar 3.1 Diagram Alur Penelitian

3.1.1 Penentuan Obyek

Penentuan obyek merupakan awal dari penelitian, dimana penulis menentukan obyek apa yang akan diteliti. Obyek yang digunakan adalah implementasi metode yang dapat memberikan sebuah rekomendasi buku kepada pengunjung perpustakaan Universitas Brawijaya berdasarkan kemiripan *user*. Rekomendasi diberikan ketika pengguna menggunakan katalog pencarian. Pada tahap ini juga ditentukan mengenai lokasi yang dipilih untuk dijadikan sebagai studi kasus.

3.1.2 Penentuan Metode

Setelah menentukan obyek yang akan diteliti, hal yang harus ditentukan selanjutnya adalah metode yang akan digunakan. Setelah melakukan perbandingan beberapa metode, penulis memilih metode *User-Based Collaborative Filtering*. Alasan dari penggunaan metode ini adalah kesesuaian perhitungan algoritmanya dengan rekomendasi buku. Selain itu dengan tekniknya yang menghitung kemiripan antar pengguna cocok digunakan di perpustakaan Universitas Brawijaya, yang mayoritas pengunjungnya adalah mahasiswa dari berbagai jurusan dan fakultas. Diharapkan hasil yang ditampilkan melalui perhitungan metode ini akan menghasilkan hasil yang tinggi tingkat akurasinya.

3.1.3 Studi Literatur

Studi literatur digunakan untuk membandingkan penelitian-penelitan sebelumnya yang terkait dengan penelitian yang dilakukan oleh penulis. Dari perbandingan tersebut akan terlihat metode dan obyek yang digunakan pada penelitian-penelitian tersebut. Hasil perbandingan tersebut akan digunakan sebagai bahan referensi dalam penelitian ini.

3.1.4 Analisa Kebutuhan

Analisa kebutuhan digunakan untuk mengetahui kebutuhan apa saja yang dibutuhkan dalam penelitian, baik dari kebutuhan fungsional dan kebutuhan non fungsional

3.1.4.1 Kebutuhan Fungsional

a. Implementasi dapat menerima *input* dari buku yang dipilih dari daftar pencarian yang dilakukan oleh pengguna
b. Implementasi membutuhkan NIM dari *user* untuk mengetahui jurusan dari *user* tersebut
c. Implementasi membutuhkan data pengguna, data transaksi yang pernah dilakukan dan data buku
d. Implementasi dapat melakukan perhitungan dengan menggunakan metode *User-Based Collaborative Filtering*
e. Implementasi dapat mengeluarkan *output* berupa beberapa rekomendasi buku sesuai dengan buku yang dicari oleh pengunjung perpustakaan

3.1.4.2 Kebutuhan Non Fungsional

Dengan keterbatasan *hardware*, implementasi dapat menghitung di bawah 1 menit

3.1.5 Perancangan Implementasi

Perancangan implementasi digunakan agar penulis memiliki gambaran untuk melakukan implementasi metode yang digunakan. Nantinya hasil implementasi akan dibandingkan dengan perancangan, apakah sudah sesuai atau belum. Perancangan terdiri dari tiga bagian, yakni perancangan algoritma, perancangan antarmuka dan perancangan *database*.

3.1.6 Implementasi

Implementasi merupakan tahapan untuk mengimplementasikan perancangan yang telah dibuat sebelumnya. Pada implementasi ini, terdapat beberapa bahasa yang akan digunakan. Bahasa Java digunakan sebagai algoritma perhitungan metode. Proses-proses pada metode *User-Based Collaborative Filtering* akan dihitung disini. Nantinya hasil perhitungan akan ditampilkan dalam bentuk aplikasi *desktop*, sehingga dibutuhkan bahasa pemograman *Java* juga. Untuk *database* akan digunakan bahasa MySQL. Data yang digunakan dari perpustakaan Universitas Brawijaya adalah data pengguna, data transaksi pengguna dan data buku. Ada dua *input* yang dibutuhkan, yakni *input* untuk mengetahui jurusan *user* dan *input* untuk mengetahui buku yang dipilih. *Input* untuk mengetahui jurusan dapat diketahui ketika *user* memasukkan NIM. *Input* buku diperoleh dari pilihan buku hasil pencarian *user* di katalog pencarian. Setelah itu, implementasi melakukan perhitungan menggunakan metode *User-Based Collaborative Filtering* untuk menentukan rekomendasi buku yang sesuai dengan atribut-atribut tersebut. *Output* yang dihasilkan dari perhitungan tersebut adalah beberapa rekomendasi buku yang sesuai dengan buku yang dipilih oleh *user* dari hasil pencarian. Nantinya hasil rekomendasi tersebut akan terletak di bawah detail buku yang dipilih.

3.1.7 Pengujian

Pengujian implementasi dilakukan untuk mengetahui apakah implementasi sudah sesuai dengan yang direncanakan. Terdapat beberapa kondisi yang akan dianalisis untuk mengetahui pada kondisi apa metode *User-Based Collaborative Filtering* cocok digunakan dan kondisi seperti apa yang membuat metode ini tidak cocok untuk digunakan. Selain itu pengujian juga dilakukan untuk mengukur berapa tingkat keakurasian metode *User-Based Collaborative Filtering* jika diterapkan pada rekomendasi buku. Pengujian akan dilakukan dengan menggunakan metode *Mean Absolute Error (MEA)*. Nantinya akan dilakukan perbandingan, apakah hasil rekomendasi yang diberikan satu kelas dengan buku

yang dipilih dengan menggunakan tiga digit dan satu digit awal kode DDC dari buku tersebut.

3.1.8 Penarikan Kesimpulan

Tahap terakhir yang dilakukan adalah penarikan kesimpulan. Kesimpulan dapat ditarik dari hasil pengujian yang telah dilakukan terhadap implementasi yang telah dilakukan. Lalu setelah itu adalah pemberian saran yang digunakan untuk pengembangan metode yang telah diimplementasikan dapat menjadi lebih baik ke depannya.

3.2. Kebutuhan Implementasi

Kebutuhan implementasi akan membahas mengenai spesifikasi yang dibutuhkan untuk melakukan implementasi sistem. Kebutuhan sistem terbagi menjadi dua hal, kebutuhan perangkat keras (*hardware*) dan perangkat lunak (*software*).

3.2.1 Kebutuhan Perangkat Keras (Hardware)

Perangkat keras yang dibutuhkan dalam membangun implementasi ini adalah laptop dengan spesifikasi :

a. Prosesor Intel i5
b. VGA Nvidia 1 GB
c. RAM 4 GB
d. Harddisk 500 GB

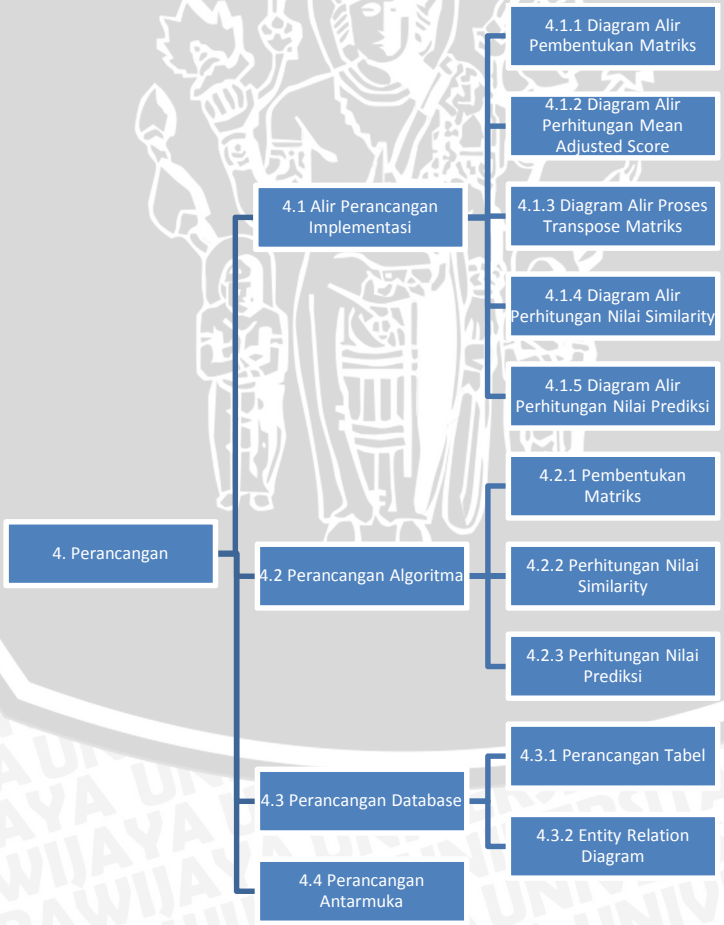
3.2.2 Kebutuhan Perangkat Lunak (Software)

Perangkat lunak yang dibutuhkan dalam membangun implementasi ini adalah:

a. *Windows 7* sebagai sistem operasi komputer
b. *Microsoft Word* sebagai penyusun laporan
c. *Microsoft Excel* sebagai perhitungan manualisasi
d. *Netbeans* sebagai penyunting bahasa pemograman *Java*
e. *SQLyog* sebagai pengolah *database*

BAB 4 PERANCANGAN

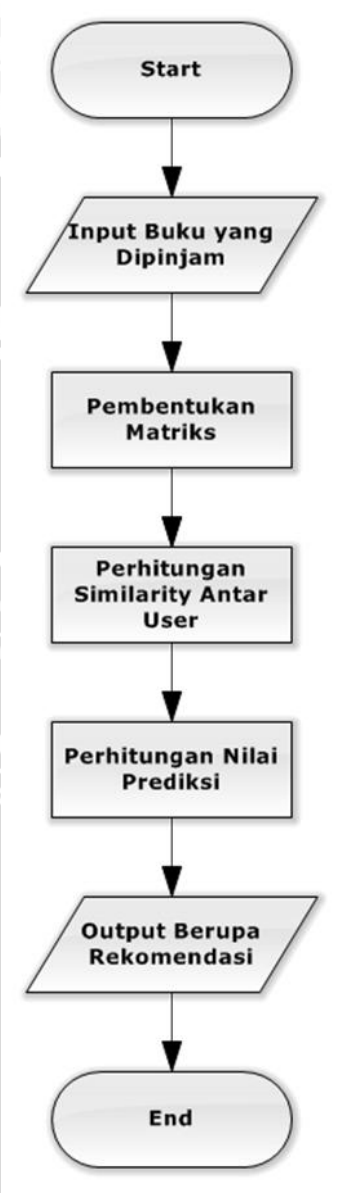
Pada bab ini akan dijelaskan mengenai perancangan dari implementasi rekomendasi buku dengan menggunakan metode *User-Based Collaborative Filtering*. Perancangan sistem sendiri terdiri dari 4 bagian, yakni alir perancangan implementasi, perancangan algoritma, perancangan *database* dan perancangan antarmuka. Alir perancangan implementasi digunakan untuk menggambarkan bagaimana aliran implementasi berjalan. Pertama dijelaskan aliran implementasi secara umum, lalu aliran dari tiap-tiap langkah perhitungan untuk menemukan rekomendasi. Lalu perancangan algoritma menjelaskan bagaimana implementasi melakukan perhitungan secara manual. Nantinya langkah-langkah perhitungan ini yang akan diimplementasikan dalam bentuk program. Perancangan *database* digunakan untuk merancang tabel-tabel apa saja yang akan digunakan pada implementasi. Agar lebih jelas, tabel-tabel ini akan dibuat *Entity Relation Diagram* untuk menjelaskan hubungan antara tabel. Lalu yang terakhir adalah perancangan antarmuka implementasi yang akan digunakan sebagai acuan dalam membuat antarmuka sistem. Untuk lebih jelasnya dapat dilihat pada Gambar 4.1.



Gambar 4.1 Pohon Perancangan

4.1. Alir Perancangan Implementasi

Rekomendasi buku dengan menggunakan implementasi metode *User-Based Collaborative Filtering* ini sendiri memiliki alir perancangan implementasi seperti yang akan ditunjukkan pada Gambar 4.2.



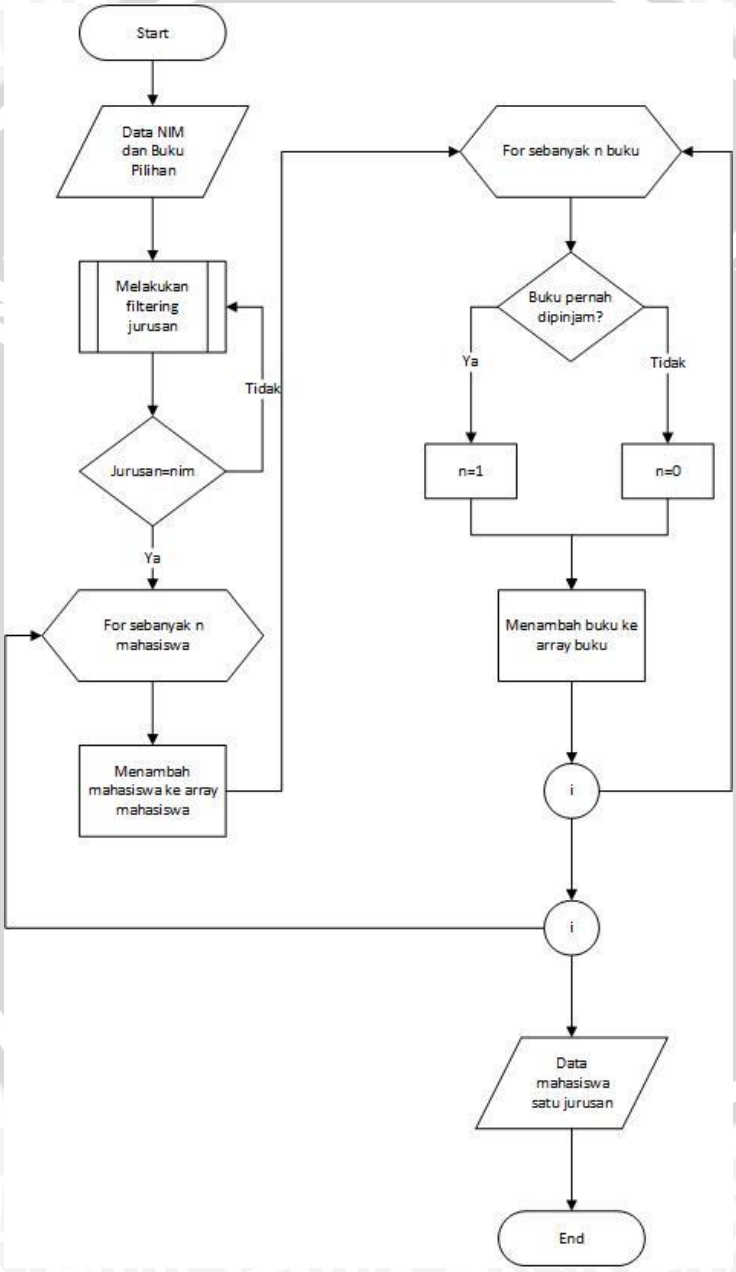
Gambar 4.2 Diagram Alir Perancangan Implementasi

Alir perancangan implementasi dimulai ketika adanya *input* yang masuk ke dalam sistem. *Input* disini berupa buku yang dicari pada katalog perpustakaan. Setelah diketahui buku yang dicari, sistem akan membuat matriks yang berisikan data buku dan data mahasiswa yang satu jurusan dengan *user*. Nantinya data dari matriks ini akan digunakan untuk menghitung kemiripan antar *user*. Oleh karena metode yang digunakan adalah *User-Based Collaborative Filtering*, maka nantinya sistem akan menghitung kemiripan berdasarkan kesamaan buku yang pernah dipinjam dengan *user* lain. Setelah menemukan kemiripan antar *user*, maka sistem

akan menghitung nilai prediksi untuk menentukan buku apa saja yang sesuai untuk direkomendasikan menggunakan *Weigthed Sum*. Sistem selesai ketika sistem mengeluarkan *output* berupa rekomendasi yang akan terletak di bawah detail buku yang dicari.

4.1.1 Alir Perancangan Pembentukan Matriks

Proses yang dilakukan pertama setelah melakukan *input* adalah pembentukan matriks dari *input* tersebut. Diagram alir dari pembentukan matriks tersebut akan ditunjukkan pada Gambar 4.3.

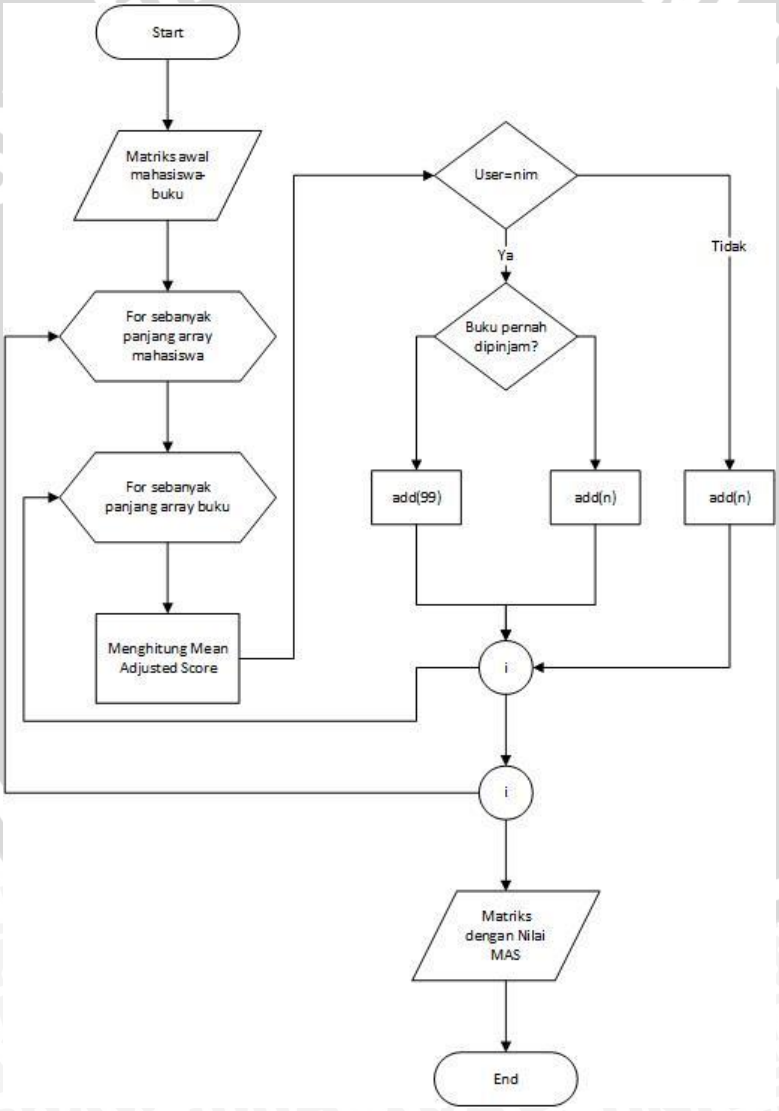


Gambar 4.3 Diagram Alir Pembentukan Matriks

Terdapat dua *input* yang dibutuhan dalam membentuk matriks, yakni NIM mahasiswa dan buku yang dipilih. Proses awal yang dilakukan adalah melakukan proses *filtering* jurusan terhadap mahasiswa, sehingga mahasiswa yang akan dimasukkan ke dalam matriks adalah mahasiswa yang satu jurusan dengan *user*. Setelah itu barulah dilakukan perulangan untuk memasukkan data mahasiswa tersebut ke dalam *array*. Di dalam perulangan ini terdapat perulangan lagi untuk memasukkan data buku ke dalam *array*. Terdapat pengkondisian untuk mengisi nilai buku, dimana 1 artinya pernah dipinjam dan 0 artinya belum pernah dipinjam. Setelah seluruh proses ini selesai, maka akan dihasilkan matriks awal.

4.1.2 Alir Perancangan Perhitungan Mean Adjusted Score

Seperti yang telah dijelaskan pada bab sebelumnya, perhitungan dengan metode *Mean Adjusted Score* digunakan untuk mempermudah visualisasi perhitungan. Diagram alir dari proses ini ditunjukkan pada Gambar 4.4.

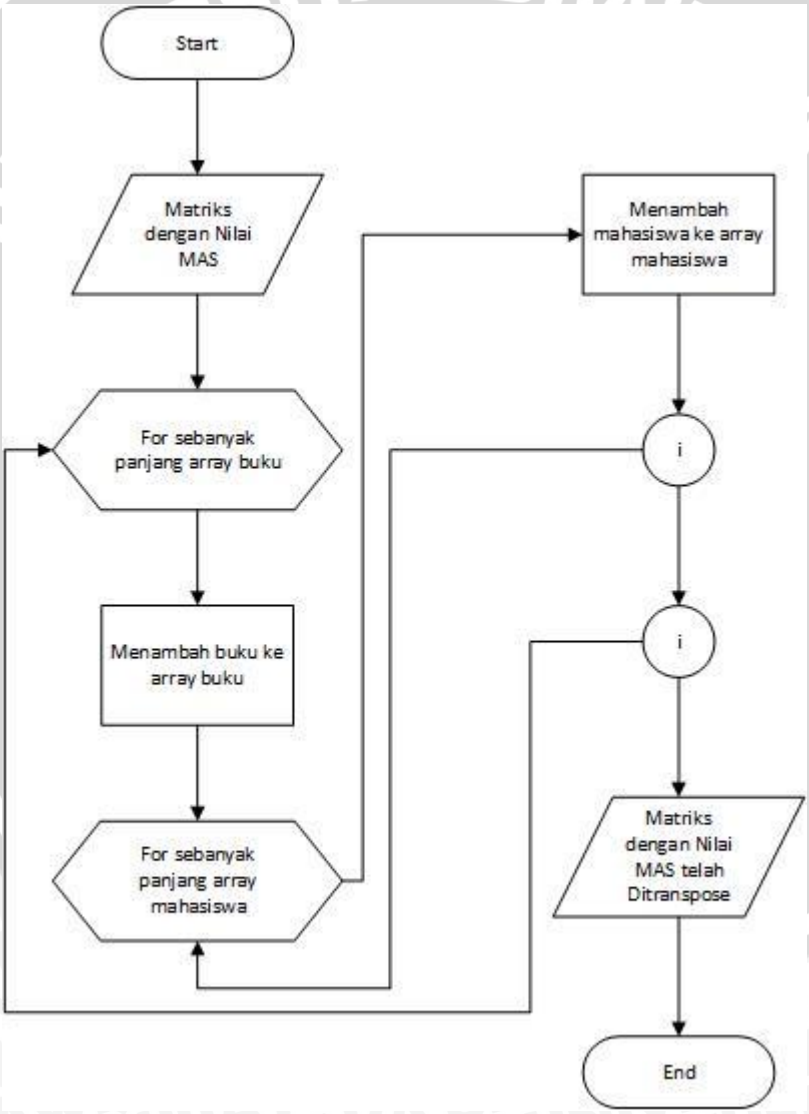


Gambar 4.4 Diagram Alir Perhitungan *Mean Adjusted Score*

Ketika menghitung *Mean Adjusted Score*, sistem akan mengecek apakah *user* yang dihitung merupakan *user* yang memasukkan NIM-nya. Jika iya, maka akan dilakukan pengecekan apakah *user* tersebut pernah meminjam buku yang dipilih. Jika pernah dipinjam, maka nilai dari perhitungan akan dimasukkan. Jika belum, maka akan dilambangkan dengan angka 99. Jika *user* bukan *user* yang memasukkan NIM, maka sistem akan memasukkan hasil perhitungan *Mean Adjusted Score*. *Output* dari proses ini adalah matriks yang telah diketahui nilai *Mean Adjusted Score*-nya.

4.1.3 Alir Perancangan Proses *Transpose* Matriks

Proses yang dilakukan setelah menghitung *Mean Adjusted Score* adalah melakukan *transpose* ke dalam matriks. Diagram alir dari proses ini ditunjukkan pada Gambar 4.5.

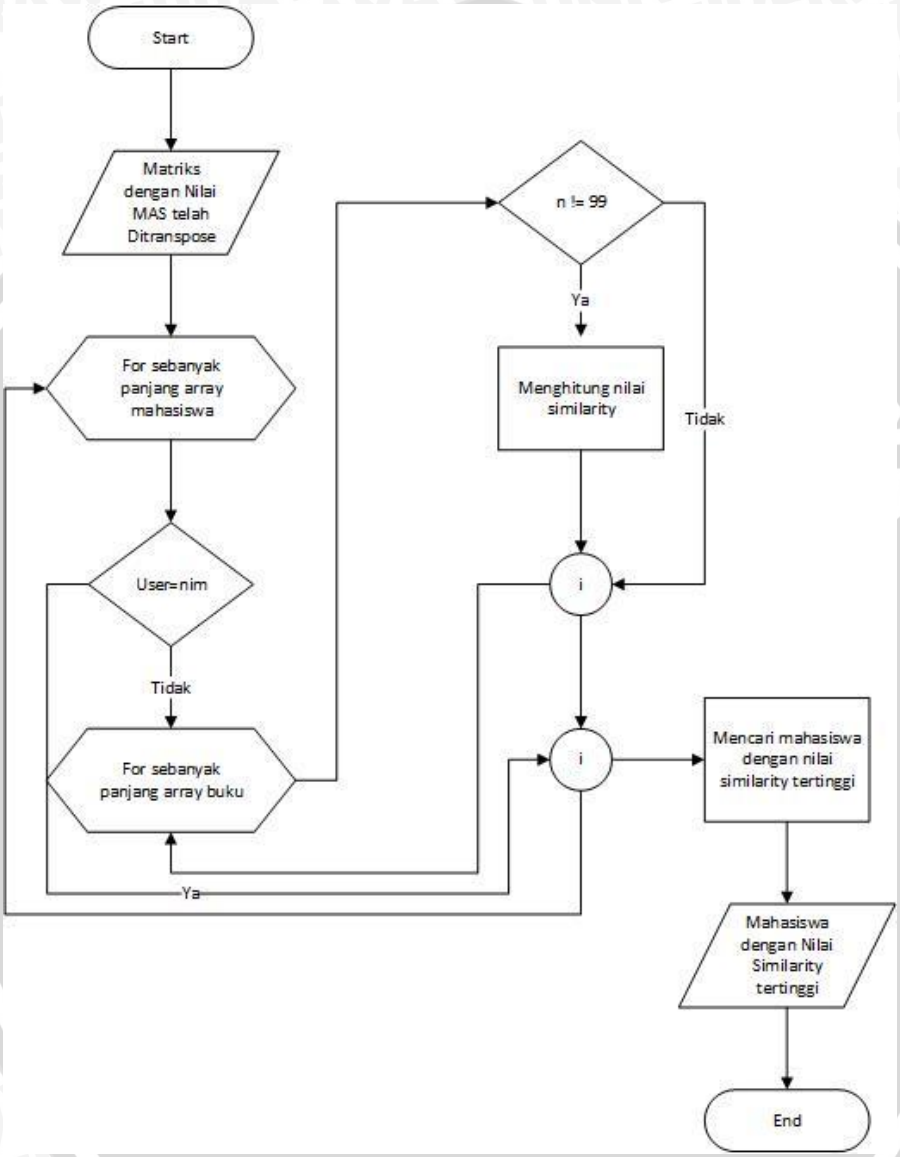


Gambar 4.5 Diagram Alir Proses *Transpose* Matriks

Proses dari *transpose* ini terbilang sederhana, hanya membalik proses memasukkan data ke *array* antara data mahasiswa dan buku.

4.1.4 Alir Perancangan Perhitungan Nilai *Similarity*

Setelah matriks selesai di-transpose, langkah yang dilakukan selanjutnya adalah menghitung nilai *similarity* antar *user*. Diagram alir dari proses perhitungan nilai *similarity* ditunjukkan pada Gambar 4.6.

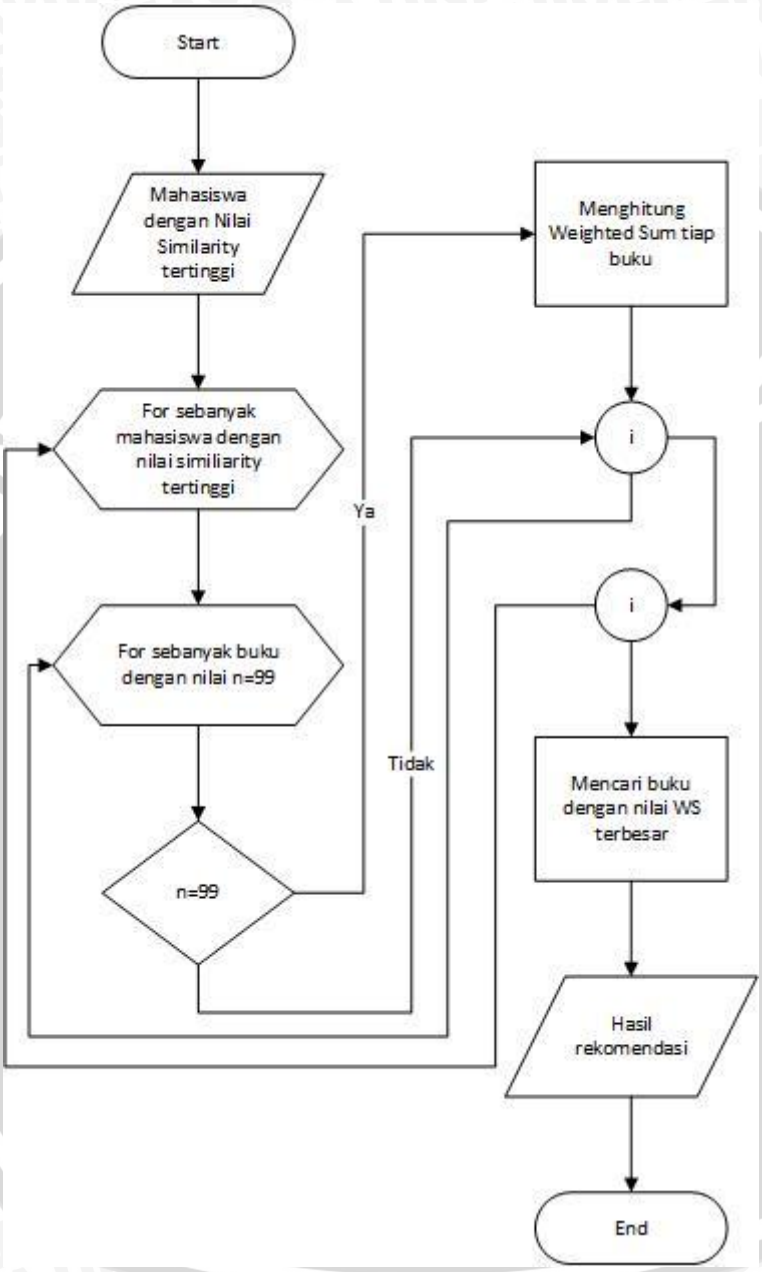


Gambar 4.6 Diagram Alir Perhitungan Nilai *Similarity*

User yang dihitung disini adalah *user* selain yang memasukkan NIM. Setelah itu, akan dilakukan pengulangan untuk mengetahui buku-buku yang nilainya di *user* yang memasukkan NIM bukan 99 atau pernah dipinjam. Dari sinilah dapat diketahui kemiripan antar *user*. Setelah seluruh perulangan selesai, akan diketahui *user* dengan nilai *similarity* tertinggi.

4.1.5 Alir Perancangan Perhitungan Nilai Prediksi

Setelah menemukan *user* dengan kemiripan paling besar, langkah selanjutnya adalah menghitung nilai prediksi dari *item* yang akan direkomendasikan. Diagram alir dari perhitungan nilai prediksi akan ditunjukkan pada Gambar 4.7.



Gambar 4.7 Diagram Alir Perhitungan Nilai Prediksi

Perhitungan nilai prediksi buku dari *user-user* yang memiliki nilai *similarity* tertinggi menggunakan metode *Weighted Sum*. Caranya, dengan mencari buku yang belum pernah dipinjam oleh *user* yang memasukkan NIM. Setelah dihitung, maka akan dicari buku-buku dengan nilai *Weighted Sum* tertinggi. Inilah yang akan direkomendasikan kepada *user*.

4.2. Perancangan Implementasi Rekomendasi

Pada bagian ini akan dijelaskan mengenai perancangan tentang implementasi rekomendasi yang akan dibangun. Perancangan implementasi rekomendasi ini terbagi menjadi tiga bagian, perancangan perhitungan, perancangan *database* dan perancangan antarmuka.

4.2.1 Perancangan Perhitungan

Perancangan perhitungan akan menjelaskan mengenai manualisasi implementasi rekomendasi buku menggunakan metode *User-Based Collaborative Filtering*. Disini akan dijelaskan secara bertahap bagaimana metode tersebut akan membuat rekomendasi. Tahapan-tahapan metode *User-Based Collaborative Filtering* sendiri adalah pembentukan matriks, perhitungan nilai *similarity* dan perhitungan nilai prediksi.

4.2.1.1 Pembentukan Matriks

Tabel matriks disini digunakan untuk mengetahui apakah suatu buku pernah dipinjam oleh mahasiswa atau tidak. Buku yang pernah dipinjam bernilai 1 dan buku yang tidak pernah dipinjam bernilai 0. Angka 1 dan 0 ini berfungsi sebagai peringkat dari buku tersebut. Nantinya peringkat tersebut berfungsi untuk mengetahui kemiripan antar *user*.

Sebagai contoh, Tabel 4.1 adalah sebuah tabel antara mahasiswa dengan buku yang pernah dipinjam:

Tabel 4.1 Contoh Tabel Mahasiswa dan Buku yang Pernah Dipinjam

Mahasiswa	Buku	Pernah Dipinjam
1	A	0
1	B	1
1	C	0
1	D	1
1	E	0
2	A	1
2	B	0
2	C	0
2	D	1
2	E	0
3	A	0
3	B	1
3	C	1
3	D	0
3	E	1
4	A	0

4	B	1
4	C	1
4	D	0
4	E	0
5	A	1
5	B	0
5	C	1
5	D	0
5	E	1

dari Tabel 4.1, maka informasi tersebut akan dikonversi menjadi Tabel 4.2.

Tabel 4.2 Matriks Hubungan Antara Mahasiswa dan Buku

Mahasiswa	Buku				
	A	B	C	D	E
1	0	1	0	1	0
2	1	0	0	1	0
3	0	1	1	0	1
4	0	0	1	1	0
5	1	0	1	0	1

Tabel 4.2 akan digunakan untuk menghitung *similarity* antar *user*. Setelah itu, akan ada mahasiswa X yang akan mendapatkan rekomendasi.

Tabel 4.3 Tambahan Mahasiswa X yang Akan DIberikan Rekomendasi

Mahasiswa	Buku				
	A	B	C	D	E
1	0	1	0	1	0
2	1	0	0	1	0
3	0	1	1	0	1
4	0	0	1	1	0
5	1	0	1	0	1
X	1	1	1	X	X

4.2.1.2 Perhitungan Nilai *Similarity*

Menghitung nilai *similarity* merupakan langkah selanjutnya dalam metode *User-Based Collaborative Filtering*. Pertama yang harus dilakukan adalah menghitung nilai *Mean Adjusted Score*. Tabel 4.3 akan dihitung dengan menggunakan Persamaan 2.1. Jika semua telah dihitung, maka hasilnya akan tampak pada Tabel 4.4.

Tabel 4.4 *Mean Adjusted Score*

Mahasiswa	Buku				
	A	B	C	D	E
1	-0.4	0.6	-0.4	0.6	-0.4
2	0.6	-0.4	-0.4	0.6	-0.4
3	-0.6	0.4	0.4	-0.6	0.4
4	-0.4	-0.4	0.6	0.6	-0.4
5	0.4	-0.6	0.4	-0.6	0.4
X	0.4	0.4	0.4	X	X

Karena menggunakan persamaan *Mean Adjusted Score*, maka tabel 4.4 akan di-*transpose* menjadi tabel 4.5

Tabel 4.5 *Mean Adjusted Score* yang Telah Di *Transpose*

Buku	Mahasiswa					
	1	2	3	4	5	X
A	-0.4	0.6	-0.6	-0.4	0.4	0.4
B	0.6	-0.4	0.4	-0.4	-0.6	0.4
C	-0.4	-0.4	0.4	0.6	0.4	0.4
D	0.6	0.6	-0.6	0.6	-0.6	X
E	-0.4	-0.4	0.4	-0.4	0.4	X

Setelah itu, akan dihitung nilai *similarity* antar *user* dengan menggunakan Persamaan 2.2. Perhitungan dilakukan satu per satu antara mahasiswa X dan mahasiswa lainnya. Sebagai contoh, akan dihitung perhitungan nilai *similarity* antara mahasiswa 1 dengan mahasiswa X.

$$Sim(X,1) = \frac{\sum_{i \square I}(R_{a,i}-\overline{R}_a)(R_{b,i}-\overline{R}_b)}{\sqrt{\sum_{i \square I}(R_{a,i}-\overline{R}_a)^2}\sqrt{\sum_{i \square I}(R_{b,i}-\overline{R}_b)^2}}$$

$$= \frac{-0.4 \square\ 0.4\ \ + 0.6 \square 0.4\ + (-0.4 \square (0.4)}{\sqrt{(-0.4)^2+0.6^2+(-0.4)^2}\ \sqrt{0.4^2+0.4^2+0.4^2}}$$

= -0.140028008

Apabila diringkas, maka nilai *similarity* antara mahasiswa X dengan mahasiswa lainnya adalah sebagai berikut:

Sim (X, Mahasiswa 1) : -0.140028008
Sim (X, Mahasiswa 2) : -0.140028008
Sim (X, Mahasiswa 3) : 0.140028008
Sim (X, Mahasiswa 4) : -0.140028008
Sim (X, Mahasiswa 5) : 0.140028008

Sehingga dapat disimpulan bahwa mahasiswa 3 dan mahasiswa 5 adalah mahasiswa yang paling memiliki kemiripan dengan mahasiswa X.

4.2.1.3 Perhitungan Nilai Prediksi

Setelah melakukan perhitungan untuk mencari nilai *similarity* antar *user*, maka selanjutnya adalah menghitung nilai prediksi, manakah di antara buku D dan E yang akan menjadi rekomendasi. Perhitungan akan menggunakan Persamaan 2.3. Sebagai contoh, akan dihitung bobot buku D menggunakan Persamaan 2.3.

$$P_{u,i} = \frac{\sum_{all\ similiar\ users,N}(S_{u,N} \cdot R_{N,i})}{\sum_{all\ similiar\ users,N}(S_{u,N})}$$

$$= \frac{(0.140028008 \cdot 0 + \cdots + (0.140028008 \cdot 1)}{|(0.140028008 + \cdots + (0.140028008)|}$$

Setelah buku D dan E telah dihitung, maka dengan menggunakan Persamaan (2.3), akan didapatkan hasil sebagai berikut:

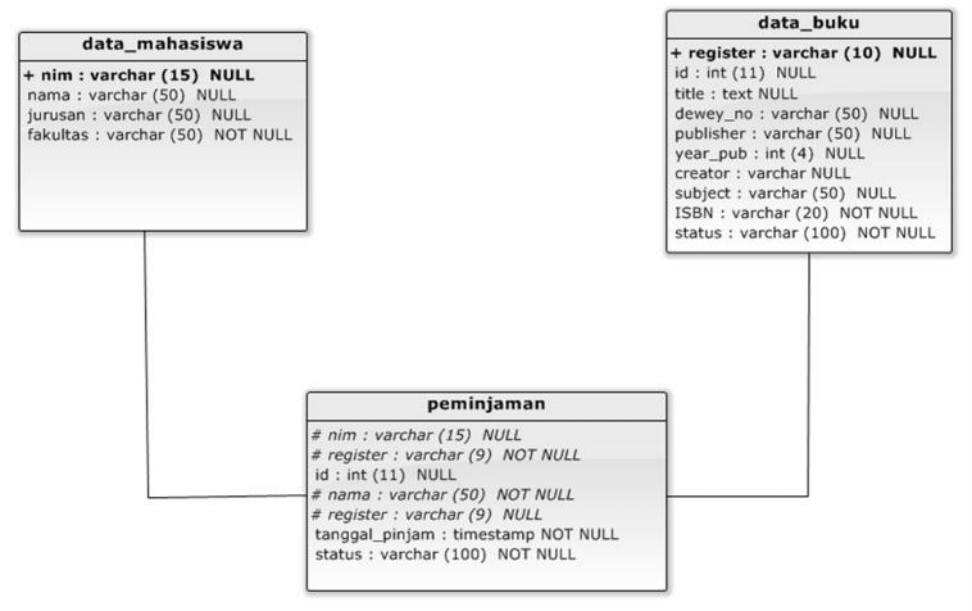
P(X, Buku D) : 0

P(X, Buku E) : 1

Sehingga dapat disimpukan bahwa buku yang akan lebih direkomendasikan kepada mahasiswa X adalah buku E dibandingkan buku D, karena memiliki nilai prediksi yang lebih besar. Seandainya nilai prediksi kedua buku sama, maka kedua buku tersebut akan sama-sama direkomendasikan.

4.2.2 Perancangan *Database*

Database yang akan digunakan disini terdiri dari 3 tabel, yakni tabel Mahasiswa, tabel Buku dan tabel Peminjaman. Secara umum *database* di sistem ini bertujuan sebagai data *training* yang digunakan untuk menghitung *similarity* antar *user*. Perancangan *database* yang terdiri dari tabel dan isi tabel terdapat pada Gambar 4.8.



Gambar 4.8 Perancangan *Database*

Tabel 4.6 adalah tabel yang berisikan data mahasiswa yang pernah melakukan transaksi di perpustakaan. Atribut-atribut dari tabel ini adalah sebagai berikut:

Tabel 4.6 Kolom Tabel Data Mahasiswa

No.	Nama Kolom	Tipe Data (Panjang)	Keterangan
1.	nim	Varchar (15)	Nomor Induk Mahasiswa (NIM)
2	nama	Varchar (50)	Nama dari mahasiswa
3	jurusan	Varchar (50)	Jurusan dari mahasiswa
4	fakultas	Varchar (50)	Fakultas dari mahasiswa

Tabel 4.7 merupakan tabel yang berisikan data-data buku yang ada di perpustakaan. Atribut-atribut dari tabel ini adalah:

Tabel 4.7 Kolom Tabel Data Buku

No.	Nama Kolom	Tipe Data (Panjang)	Keterangan
1.	id	Int (11)	Nomor identitas buku di *database*
2	register	Varchar (10)	Kode registrasi dari buku yang akan digunakan di tabel peminjaman
3	title	Text	Judul buku
4	dewey_no	Varchar (50)	Nomor *Dewey Decimal Classification* (DDC) dari buku
5	publisher	Varchar (50)	Penerbit buku

6	year_pub	Int (4)	Tahun buku diterbitkan
7	creator	Varchar (50)	Penulis buku
8	subject	Varchar (50)	Subyek atau topik dari buku
9	ISBN	Varchar (20)	Nomor *International Standard Book Number* (ISBN) dari buku
10	Status	Varchar (100)	Digunakan untuk menambahkan status *temp* ketika buku dipilih oleh *user* di pencarian

Tabel 4.8 merupakan tabel yang berisikan data peminjaman yang pernah dilakukan mahasiswa. Atribut-atribut dari tabel ini adalah:

Tabel 4.8 Tabel Data Peminjaman

No.	Nama Kolom	Tipe Data (Panjang)	Keterangan
1.	id	Int (11)	Nomor identitas peminjaman di *database*
2	nim	Varchar (15)	Nomor Induk Mahasiswa (NIM)
3	Nama	Varchar (50)	Nama dari mahasiswa
4	register	Varchar (10)	Kode registrasi dari buku yang akan digunakan di tabel peminjaman
5	title	Text	Judul buku
6	tanggal_pinjam	Timestamp	Waktu transaksi dilakukan
7	status	Varchar (100)	Digunakan untuk menambahkan status *temp* ketika buku dipilih oleh *user* di pencarian

4.2.3 Perancangan Antarmuka

Antarmuka yang akan digunakan pada sistem terdiri dari satu halaman. Sebelum memasukkan buku yang akan dipinjam, data yang harus dimasukkan terlebih dahulu adalah NIM dari peminjam. Setelah itu user akan memasukkan *keyword* pencarian. Nantinya *user* akan memilih satu buku dan dari buku yang dipilihlah sistem akan menghitung kemiripan antar *user* untuk menentukan rekomendasi. Untuk perancangannya terdapat pada Gambar 4.9.

Gambar 4.9 Perancangan Antarmuka Sistem

BAB 5 IMPLEMENTASI

Bab ini akan membahas mengenai proses implementasi dari perancangan Rekomendasi Buku dengan Menggunakan Implementasi Metode *User-Based Collaborative Filtering* yang telah disusun pada bab sebelumnya. Pembahasan yang dilakukan disini adalah implementasi algoritma, *database* dan antarmuka.

5.1 Implementasi Algoritma

Dari perancangan implementasi yang telah dibahas pada bab 4, maka disini akan dijelaskan mengenai implementasi algoritma dengan metode *User-Based Collaborative Filtering* sesuai dengan yang ada pada perancangan tersebut. Implementasi algoritma menggunakan bahasa pemograman Java dan berbasis *desktop*.

5.1.1 Implementasi Pembuatan Matriks

Tahap awal dari metode ini adalah penyusunan matriks yang berisikan data mahasiswa dan buku. Relasi antara mahasiswa dengan buku yang pernah dipinjam digambarkan dengan angka 0 dan 1, dimana 0 artinya tidak pernah dipinjam dan 1 artinya pernah dipinjam. Untuk mempercepat proses perhitungan, ketika membentuk matriks dilakukan penyaringan berdasarkan jurusan, sehingga *user* hanya akan dibandingkan dengan *user* lain yang satu jurusan. Tahapan ini akan ditunjukkan pada Kode Program 5.1.

```
1.  public class Matriks {
2.      public String register,nim,nim_ada="false";
3.      ArrayList<ArrayList>        matriks        =        new
4.  ArrayList<ArrayList>();
5.      ArrayList<ArrayList> buku = new ArrayList<ArrayList>();
6.      ArrayList<Integer> nilai = new ArrayList<Integer>();
7.      ArrayList<Integer> cari_buku = new ArrayList<Integer>();
8.      ArrayList<ArrayList> mas = new ArrayList<ArrayList>();
9.      ArrayList<ArrayList>       transpose       =       new
10. ArrayList<ArrayList>();
11.     ArrayList<String> arr_nim = new ArrayList<String>();
12.     ArrayList<String> arr_buku = new ArrayList<String>();
13.     ArrayList<String> semua_buku = new ArrayList<String>();
14.     ArrayList<String> subject = new ArrayList<String>();
15.     ArrayList<String> hasil_buku = new ArrayList<String>();
16.     ArrayList<String> filter_subj = new ArrayList<String>();
17.     ArrayList<Double> nilai_sim = new ArrayList<Double>();
18.     ArrayList<Double> weight_sum = new ArrayList<Double>();
19.
20.     // JDBC driver name and database URL
21.    static       final       String       JDBC_DRIVER       =
22. "com.mysql.jdbc.Driver";
23.    static        final        String        DB_URL        =
24. "jdbc:mysql://localhost:3307/dummy?autoReconnect=true&amp;v
25. erifyServerCertificate=false&amp;useSSL=true&amp;requireSSL
26. =true";
27.
28.    //  Database credentials
```

```
29.    static final String USER = "root";
30.    static final String PASS = "password";
31.
32.     public   ArrayList   getMatriks(String   gui_nim,String
33. gui_reg){
34.         Connection conn = null;
35.         Statement stmt = null,stmt2 = null,stmt3 = null,stmt4
36. = null,stmt4_2 = null,stmt5 = null,stmt6 = null,stmt7 = null;
37.    try{
38.
39.       //STEP 2: Register JDBC driver
40.       Class.forName("com.mysql.jdbc.Driver");
41.
42.       //STEP 3: Open a connection
43.       System.out.println("Connecting to database...");
44.       conn = DriverManager.getConnection(DB_URL,USER,PASS);
45.
46.       //STEP 4: Execute a query
47.       System.out.println("Creating statement...");
48.       stmt = conn.createStatement();
49.       stmt4 = conn.createStatement();
50.       stmt4_2 = conn.createStatement();
51.       stmt5 = conn.createStatement();
52.       stmt6 = conn.createStatement();
53.       stmt7 = conn.createStatement();
54.
55.       String sql,sql2,sql3,sql4,sql5,sql6,sql7;
56.       sql4 = "SELECT * FROM data_mahasiswa where nim =
57. '"+gui_nim+"'";
58.       sql5 = "SELECT * FROM data_buku where register =
59. '"+gui_reg+"'";
60.
61.       ResultSet rs4 = stmt4.executeQuery(sql4);
62.       ResultSet rs4_2 = stmt4_2.executeQuery(sql4);
63.       ResultSet rs5 = stmt5.executeQuery(sql5);
64.        if(rs4_2.next()){
65.          nim_ada = "true";
66.      }
67.       String nama="";
68.       String jurusan="";
69.
70.       while(rs4.next()){
71.       nama = rs4.getString("nama");
72.       jurusan = rs4.getString("jurusan");
73.
74.       }
75.        System.out.println(jurusan);
76.       String title="";
77.       while(rs5.next()){
78.       title = rs5.getString("title");
79.       String str_sbj = rs5.getString("subject");
80.       for(String retval: str_sbj.split(";")) {
81.           if(retval.trim().equals("")){
82.           }else{
83.           subject.add(retval);
84.
85.           }
86.       }
87.       }
```

```
88.       sql6          =          "INSERT          INTO
89.  peminjaman(nim,nama,register,title,status)VALUES('"+gui_nim
90.  +"','"+nama+"','"+gui_reg+"','"+title+"','temp')";
91.
92.       stmt6.executeUpdate(sql6);
93.
94.       sql  =  "SELECT  *  FROM  data_mahasiswa  where  nim  !=
95.  '"+gui_nim+"' and jurusan='"+jurusan+"'";
96.
97.       ResultSet rs = stmt.executeQuery(sql);
98.
99.        while(rs.next()){
100.         //Retrieve by column name
101.          String nim = rs.getString("nim");
102.
103.         arr_nim.add(nim);
104.
105.       }
106.        sql2 = "SELECT * FROM data_buku ";
107.        arr_nim.add(gui_nim);
108.        stmt2 = conn.createStatement();
109.        ResultSet rs2 = stmt2.executeQuery(sql2);
110.
111.        while(rs2.next()){
112.
113.          String reg = rs2.getString("register");
114.
115.          arr_buku.add(reg);
116.          }
117.
118.       for(int i=0;i<arr_nim.size();i++){
119.            matriks.add(new ArrayList());
120.            buku.add(new ArrayList());
121.          for(int ii=0;ii<arr_buku.size();ii++){
122.               int n;
123.          sql3  =  "SELECT  *  FROM  peminjaman  where  nim  =
124. '"+arr_nim.get(i)+"' and register = '"+arr_buku.get(ii)+"'";
125.          stmt3 = conn.createStatement();
126.          ResultSet rs3 = stmt3.executeQuery(sql3);
127.          if(rs3.next()){
128.              n = 1;
129.               String x = rs3.getString("register");
130.               ((ArrayList)buku.get(i)).add(x);
131.          }
132.          else{
133.              n = 0;
134.          }
135.
136.          ((ArrayList)matriks.get(i)).add(n);
137.        }
138.        }
139.         sql7  =  "DELETE  FROM  peminjaman  where  status  =
140. 'temp'";
141.          stmt7 = conn.createStatement();
142.          stmt7.execute(sql7);
143.
144.      //STEP 5: Extract data from result set
145.
146.      //STEP 6: Clean-up environment
```

```
147.       rs.close();
148.       stmt.close();
149.       conn.close();
150.
151.    }catch(SQLException se){
152.       //Handle errors for JDBC
153.       se.printStackTrace();
154.
155.    }catch(Exception e){
156.       //Handle errors for Class.forName
157.       e.printStackTrace();
158.    }finally{
159.       //finally block used to close resources
160.       try{
161.          if(stmt!=null)
162.             stmt.close();
163.       }catch(SQLException se2){
164.       }// nothing we can do
165.       try{
166.          if(conn!=null)
167.             conn.close();
168.       }catch(SQLException se){
169.          se.printStackTrace();
170.       }//end finally try
171.    }//end try
172.
173.          return matriks;
174.     }
175.
176.     public ArrayList getBuku(){
177.     Connection conn = null;
178.         Statement stmt = null,stmt2 = null,stmt3 = null;
179.    try{
180.       //STEP 2: Register JDBC driver
181.       Class.forName("com.mysql.jdbc.Driver");
182.
183.       //STEP 3: Open a connection
184.       //System.out.println("Connecting to database...");
185.       conn = DriverManager.getConnection(DB_URL,USER,PASS);
186.
187.       //STEP 4: Execute a query
188.      // System.out.println("Creating statement...");
189.       stmt = conn.createStatement();
190.       String sql,sql2,sql3;
191.       sql = "SELECT * FROM data_buku limit 1000";
192.       ResultSet rs = stmt.executeQuery(sql);
193.
194.        while(rs.next()){
195.          //Retrieve by column name
196.           String register = rs.getString("register");
197.
198.          // matriks.add(new ArrayList());
199.
200.          semua_buku.add(register);
201.
202.       }
203.
204.       //STEP 5: Extract data from result set
205.
```

```
206.       //STEP 6: Clean-up environment
207.       rs.close();
208.       stmt.close();
209.       conn.close();
210.    }catch(SQLException se){
211.       //Handle errors for JDBC
212.       se.printStackTrace();
213.    }catch(Exception e){
214.       //Handle errors for Class.forName
215.       e.printStackTrace();
216.    }finally{
217.       //finally block used to close resources
218.       try{
219.          if(stmt!=null)
220.             stmt.close();
221.       }catch(SQLException se2){
222.       }// nothing we can do
223.       try{
224.          if(conn!=null)
225.             conn.close();
226.       }catch(SQLException se){
227.          se.printStackTrace();
228.       }//end finally try
229.    }//end try
230.
231.        return semua_buku;
232.     }
```

Kode Program 5.1 Proses Pembentukan Matriks

Baris 2-18 merupakan deklarasi variabel-variabel yang digunakan pada program. Baris 21-44 merupakan fungsi untuk melakukan koneksi terhadap database. Baris 47-87 merupakan fungsi *query* untuk pengambilan data, dimana baris 55-59 merupakan *query* untuk mengambil data masukan berupa NIM *user* dan register buku yang dipilih. Lalu pada baris 64-87, adalah pengambilan data dari tabel data_mahasiswa dan data_buku. Data yang diambil dari data_mahasiswa adalah "nama" dan "jurusan". Sedangkan dari data_buku yang diambil adalah "*title*" dan "*subject*". Nantinya data tersebut akan digunakan pada baris 88-95 untuk di *insert* ke database peminjaman. Tujuan dari proses *insert* ini adalah untuk melakukan *update*, sehingga jika *user* memilih buku, buku tersebut akan dianggap pernah dipinjam oleh *user*, namun akan dihapus ketika proses selesai.

Baris 97-105 digunakan untuk menyimpan NIM mahasiswa termasuk *user*. Hal ini dilakukan untuk menentukan berapa jumlah mahasiswa yang akan dihitung dalam proses pembentukan matriks. Baris 106-116 hampir sama dengan baris 97-105, hanya yang disimpan adalah register dari bukunya. Baris 118-138 digunakan untuk pembentukan matriks antara mahasiswa dengan buku. Baris 139-142 digunakan untuk menghapus data dari tabel peminjaman yang statusnya "*temp*". Baris 147-149 digunakan untuk menutup koneksi. Baris 151-174 merupakan *handler* untuk mengatasi jika ada *error* yang terjadi.

Baris 176-232 digunakan untuk mengambil semua data buku untuk menyimpan register buku. Nantinya register yang disimpan tersebut akan

digunakan untuk mengetahui register buku mana saja yang akan ditampilkan sebagai rekomendasi.

5.1.2 Implementasi *Mean Adjusted Score*

Telah dijelaskan diperancangan algoritma, *Mean Adjusted Score* digunakan untuk mempermudah visualisasi perhitungan. Hal ini terjadi karena perhitungan dengan hanya menggunakan angka 0 dan 1 akan lebih susah untuk dikalkulasi. Tahapan ini akan ditunjukkan pada Kode Program 5.2.

```
1.  public ArrayList MeanAdjustedScore(){
2.
3.          for(int i=0;i<arr_nim.size();i++){
4.              mas.add(new ArrayList());
5.               int sumMatriks=0;
6.              for(int jj=0;jj<arr_buku.size();jj++){
7.
8.              int                 tempNilai                   =
9.  Integer.parseInt(""+matriks.get(i).get(jj));
10.             sumMatriks = sumMatriks + tempNilai;
11.
12.             }
13.             //System.out.print("[");
14.             for(int j=0;j<arr_buku.size();j++){
15.
16.             double                  tempNilai               =
17. Integer.parseInt(""+matriks.get(i).get(j));
18.               //              System.out.print(tempNilai+"-
19. ("+sumMatriks+"/"+((ArrayList)matriks.get(i)).size()+"),");
20.                 double    tempHitung    =         (double)
21. sumMatriks/((ArrayList)matriks.get(i)).size();
22.
23.             double n = tempNilai-tempHitung;
24.             int a = arr_nim.size()-1;
25.             if(i==a){
26.                 int                 b                   =
27. Integer.parseInt(""+matriks.get(i).get(j));
28.                 if(b==0){
29.                    ((ArrayList)mas.get(i)).add(99);
30.
31.                 }
32.                 else{
33.                 ((ArrayList)mas.get(i)).add(n);
34.                 }
35.             }
36.             else{
37.             ((ArrayList)mas.get(i)).add(n);
38.
39.             }
40.             }
41.             //System.out.println("]");
42.         }
43.         return mas;
44.     }
```

Kode Program 5.2 Proses Perhitungan *Mean Adjusted Score*

Baris 3-12 digunakan untuk membentuk matriks dengan melakukan perulangan sejumlah *array* NIM dan *array* buku. Baris 14-21 digunakan untuk melakukan perhitungan *Mean Adjusted Score*. Baris 23-44 digunakan untuk mengetahui buku mana yang belum pernah dipinjam oleh *user* dan dilambangkan dengan angka "99".

5.1.3 Implementasi *Transpose* Matriks

Setelah perhitungan *Mean Adjusted Score*, metode *User-Based Collaborative Filtering* akan melakukan *transpose* terhadap matriks tersebut. Tahap tersebut akan dijelaskan pada Kode Program 5.2.

```
1.  public ArrayList transpose(){
2.          for(int i = 0;i<arr_buku.size();i++){
3.              transpose.add(new ArrayList());
4.              for(int ii = 0;ii<arr_nim.size();ii++){
5.                  double                      n                   =
6.  Double.parseDouble(""+mas.get(ii).get(i));
7.                  ((ArrayList)transpose.get(i)).add(n);
8.                  }
9.          }
10.         return transpose;
      }
```

Kode Program 5.3 Proses Transpose Matriks

Baris 1-10 digunakan untuk melakukan *transpose* terhadap matriks yang telah dihitung menggunakan *Mean Adjusted Score*. Cara melakukan *transpose* adalah dengan membalik baris dan kolom matriks ketika melakukan perulangan.

5.1.4 Implementasi Perhitungan Nilai *Similarity* Antar *User*

Tahap selanjutnya adalah perhitungan nilai *similarity* antara *user* dengan *user* lainnya. Pada tahap inilah inti dari metode *User-Based Collaborative Filtering*, dimana perhitungan kemiripan dilihat dari perbandingan pengalaman antar *user.* Pengalaman disini adalah pernah tidaknya *user* meminjam suatu buku. Nanti kemiripan akan dicari melalui berapa tingginya buku yang sama-sama pernah dipinjam pada masa lalu. Tahapan ini akan ditunjukkan pada Kode Program 5.4.

```
1.  public ArrayList sim(){
2.          for(int i = 0;i<arr_nim.size();i++){
3.              String status="false";
4.              double nilaiSim=0;
5.             int x = arr_nim.size()-1;
6.              if(i!=x){
7.                 double sumSimA= 0;
8.                 double sumSimB= 0;
9.                 double sumSimC= 0;
10.               // System.out.print("[");
11.             for(int ii = 0;ii<arr_buku.size();ii++){
12.
13.                 double                      m                   =
14.  Double.parseDouble(""+transpose.get(ii).get(i));
15.                  //System.out.println(""+m);
16.                 double                      n                   =
17.  Double.parseDouble(""+transpose.get(ii).get(x));
```

```
18.                 //System.out.println("m = "+m+"& n = "+n);
19.                if(n!=99.0&&i!=x&&m!=0.0){
20.                    double simA = m*n;
21.                    double simB = pow(n,2.0);
22.                    double simC = pow(m,2.0);
23.
24.                    sumSimA = sumSimA+simA;
25.                    sumSimB = sumSimB+simB;
26.                    sumSimC = sumSimC+simC;
27. //                   System.out.println("simA"+sumSimA);
28. //                   System.out.println("simB"+sumSimB);
29. //                   System.out.println("simC"+sumSimC);
30.                    status = "true";
31.                }
32.            }
33.           // System.out.println("]");
34.
35.            if(status=="true"){
36.
37.            nilaiSim                                          =
38. sumSimA/((pow(sumSimB,0.5))*(pow(sumSimC,0.5)));
39.            }
40.            else if(status=="false"){
41.            nilaiSim = -1.0;
42.            }
43.
44.               //System.out.println(""+nilaiSim);
45.            if(nilaiSim<0.9999999999999999){
46.            nilai_sim.add(nilaiSim);
47.
48.            }
49.        }
50.        }
51.        for(int i=0;i<mas.size();i++){
52.            for(int
53. ii=0;ii<((ArrayList)mas.get(i)).size();ii++){
54.                double                    x                =
55. Double.parseDouble(""+mas.get(i).get(ii));
56.                if(x==99.0){
57.                cari_buku.add(ii);
58.                }
59.            }
60.        }
61.       return nilai_sim;
62.    }
```

Kode Program 5.4 Proses Perhitungan Nilai *Similarity* Antar *User*

Baris 2-10 digunakan untuk melakukan perulangan perhitungan sejumlah *array* NIM. Baris 11-18 digunakan untuk melakukan perulangan perhitungan sejumlah *array* buku. Baris 19-32 merupakan inisialisasi variabel rumus yang akan digunakan. Baris 35-50 merupakan rumus untuk mencari nilai *similarity* antar *user*. Apabila terdapat *user* yang nilai *similarity*-nya 1, maka sistem akan mencari *user* lain yang nilainya tertinggi setelah *user* tersebut. Baris 51-62 digunakan untk mencari buku-buku yang bernilai "99" atau tidak pernah dipinjam oleh *user* lain yang memiliki nilai *similarity* tertinggi.

5.1.5 Implementasi Perhitungan *Weighted Sum*

Langkah terakhir dari metode *User-Based Collaborative Filtering* adalah perhitungan rekomendasi buku yang akan diberikan kepada *user*. Tahap tersebut menggunakan perhitungan *Weighted Sum*. Buku yang akan direkomendasikan adalah buku yang tidak pernah dipinjam oleh *user*, tapi pernah dipinjam oleh *user* lain yang nilai *similarity*-nya paling tinggi. Tahapan ini akan dijelaskan pada Kode Program 5.5.

```
1.  public ArrayList hasil(){
2.          int x = 0;
3.          Object max = Collections.max(nilai_sim);
4.          double nilaiMax = Double.parseDouble(""+max);
5.          //System.out.println("nilaiSimMax : "+nilaiMax);
6.          ArrayList<Integer>       rekomen       =       new
7.  ArrayList<Integer>();
8.          for(int i=0;i<nilai_sim.size();i++){
9.              double                tempNilai                =
10. Double.parseDouble(""+nilai_sim.get(i));
11.             if(nilaiMax==tempNilai){
12.                 rekomen.add(i);
13.                 System.out.println("rekomen : "+i);
14.             }
15.         }
16.         System.out.println("cari buku : "+cari_buku.size());
17.         for(int i=0;i<cari_buku.size();i++){
18.             double nilai_ws=0.0;
19.             double tempHitung=0.0;
20.             double tempHitung2=0.0;
21.             //ArrayList<Integer>    tempHitung    =    new
22. ArrayList<Integer>();
23.                 for(int ii=0;ii<rekomen.size();ii++){
24.                     double          tempSim             =
25. nilai_sim.get(rekomen.get(ii));
26.                     System.out.println("tempSim:"+tempSim);
27.
28. //System.out.println(""+rekomen.get(ii));
29.                     double          tempNilai           =
30. Double.parseDouble(""+matriks.get(rekomen.get(ii)).get(cari
31. _buku.get(i)));
32.                     System.out.println("cari           buku
33. :"+cari_buku.get(i));
34.
35. System.out.println("tempNilai:"+tempNilai);
36.                     double tempKali = (tempSim*tempNilai);
37.                     tempHitung = tempHitung+tempKali;
38.                     tempHitung2 = tempHitung2+abs(tempSim);
39.                 }
40.                 nilai_ws = tempHitung/tempHitung2;
41.                 //System.out.println("nilai ws :"+nilai_ws);
42.                 weight_sum.add(nilai_ws);
43.
44.         }
45.
46.         Object max2 = Collections.max(weight_sum);
47.         double max_ws = Double.parseDouble(""+max2);
48.        System.out.println("max ws: "+max_ws);
```

```
49.         for(int i=0;i<weight_sum.size();i++){
50.             double                  tempNilai                   =
51. Double.parseDouble(""+weight_sum.get(i));
52.
53.             //System.out.println("ws : "+tempNilai);
54.             if(tempNilai==max_ws){
55.                 x = Integer.parseInt(""+cari_buku.get(i));
56.                 //System.out.println("indeks buku : "+x);
57.                 hasil_buku.add(semua_buku.get(x));
58.             }
59.         }
60.         return hasil_buku;
61.
62.     }
```

Kode Program 5.5 Proses Perhitungan *Weighted Sum*

Baris 2-15 digunakan untuk mencari indeks dari *user* yang memiliki nilai *similarity* tertinggi. Baris 16-33 digunakan untuk membuat matriks baru antara *user* lain yang memiliki nilai *similarity* tertinggi dengan buku yang belum pernah dipinjam oleh *user*. Baris 35-44 digunakan untuk menghitung rumus *Weighted Sum*. Baris 46-62 digunakan untuk mencari buku-buku dengan nilai *Weighted Sum* tertinggi. Buku dengan nilai *Weighted Sum* tertinggi lah yang akan menjadi rekomendasi.

5.1.6 Implementasi Filter Subjek

Ini merupakan fitur opsional yang digunakan jika *user* hanya menghendaki rekomendasi buku yang diberikan memiliki subjek yang sama dengan buku yang dicarinya. Tahapan ini akan dijelaskan pada Kode Program 5.6.

```
1.       for(int i=0;i<hasil_buku.size();i++){
2.       sql   =   "SELECT   *   FROM   data_buku   where   register
3. ='"+hasil_buku.get(i)+"'";
4.       ResultSet rs = stmt.executeQuery(sql);
5.       String ada_subjek = "false";
6.        while(rs.next()){
7.          //Retrieve by column name
8.           String subj = rs.getString("subject");
9.           String reg = rs.getString("register");
10.
11.          for (String retval: subj.split(";")) {
12.                for(int ii=0;ii<subject.size();ii++){
13.                    if(retval.trim().equals("")){
14.                    }else{
15.
16. System.out.println(retval+"#"+subject.get(ii));
17.                    if(retval.equals(subject.get(ii))){
18.                      ada_subjek = "true";
19.                          }
20.                    }
21.             }
22.         // matriks.add(new ArrayList());
23.         }
24.            System.out.println(ada_subjek);
25.         if(ada_subjek.equals("true")){
26.         filter_subj.add(reg);
27.              }
```

```
28.         }
29.          rs.close();
30.         }
```

Kode Program 5.6 Proses Penyaringan Berdasarkan Subyek

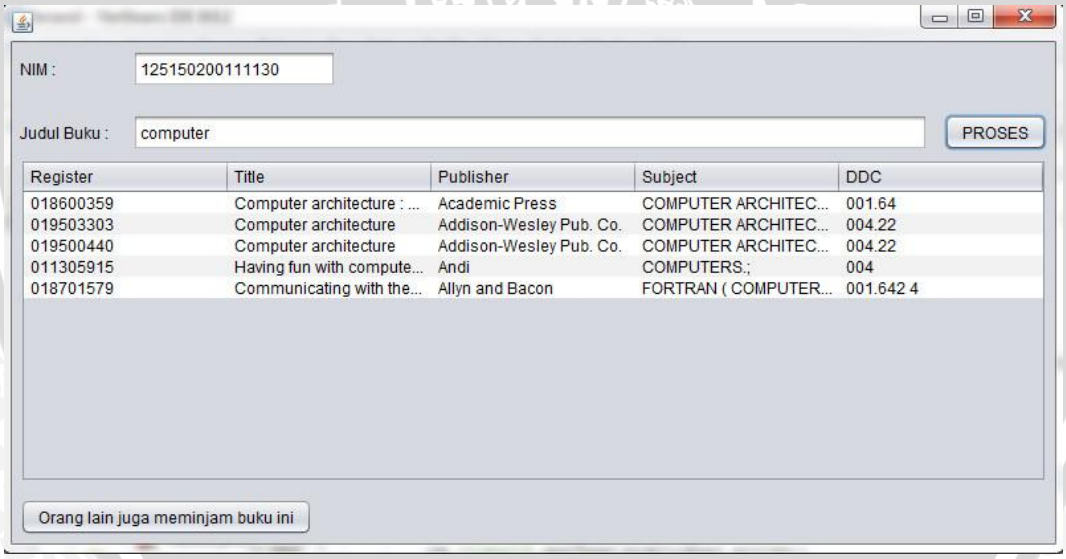
Baris 1-9 digunakan untuk mengambil data *string* dari *database* berupa "*subject*" dari buku yang direkomendasikan. Baris 11-21 digunakan untuk melakukan pemisahan subyek buku. Pemisahan subyek menggunakan tanda baca titik koma (;). Baris 24-30 digunakan untuk menampilkan hasil filter berdasarkan subyek buku.

5.2 Implementasi Antarmuka

Antarmuka implementasi menggunakan *Graphical User Inteface* (GUI) dari Java. Antarmuka terdiri dari dua bagian, yakni tampilan pencarian dan tampilan rekomendasi.

5.2.1 Implementasi Antarmuka Pencarian

Antarmuka adalah tampilan awal dari implementasi ketika *user* ingin melakukan pencarian buku. Terdapat beberapa komponen pada antarmuka katalog buku seperti yang terdapat pada Gambar 5.1.



Gambar 5.1 Tampilan Awal Antarmuka Implementasi

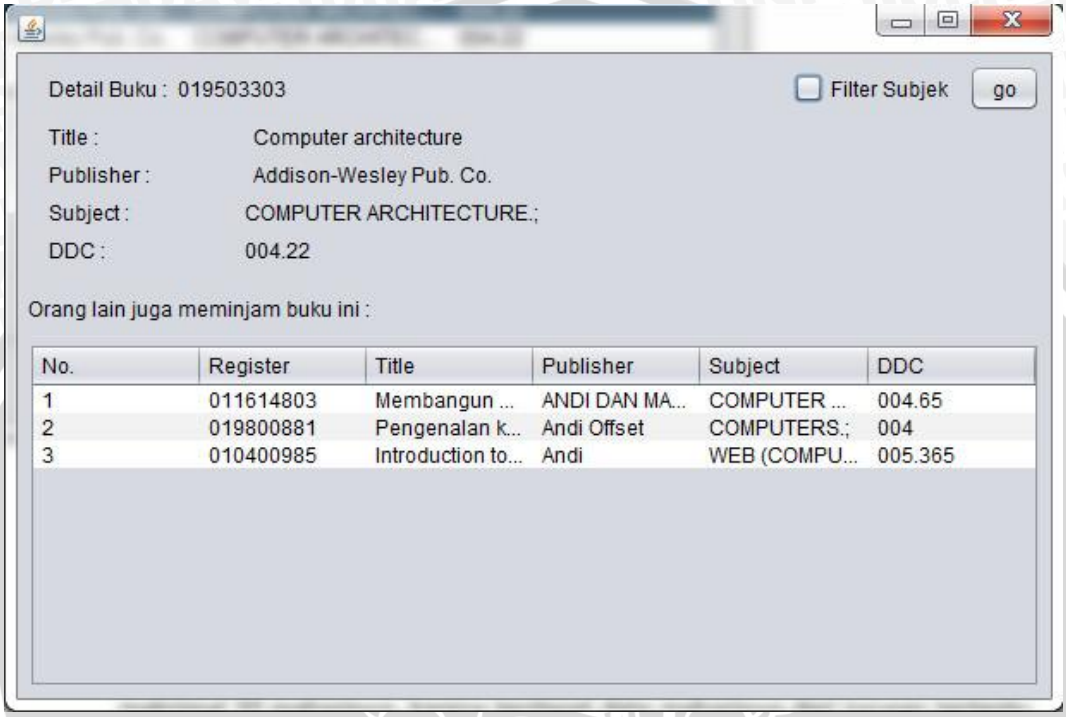
Di bagian atas implementasi, terdapat dua kolom, dimana kolom bagian atas digunakan untuk memasukkan NIM dari mahasiswa, dan kolom bagian bawah merupakan kolom pencarian. Pencarian dilakukan berdasarkan judul buku. Kedua hal tersebut harus dimasukkan oleh *user* dan sistem akan menampilkan pesan *error* apabila salah satu atau keduanya tidak di diberi *input*. Pesan *error* juga akan tampil apabila NIM yang dimasukkan tidak ada di dalam *database*.

Apabila *user* sudah memasukkan kedua hal tersebut, maka *user* akan menekan tombol proses agar sistem dapat menampilkan hasil pencarian. Ketika hasil pencarian telah ditampilkan, maka *user* dapat mengetahui buku lain yang

pernah dipinjam oleh *user* lain yang memiliki kemiripan dengannya. Caranya adalah dengan menekan tombol yang terdapat di bawah kolom hasil pencarian.

5.2.2 Implementasi Antarmuka Rekomendasi

Rekomendasi akan tampil apabila *user* telah memasukkan NIM dan memilih buku berdasarkan *keyword* yang dimasukkan ke kolom pencarian. Tampilan jendela rekomendasi terdapat pada Gambar 5.2.



Gambar 5.2 Tampilan Hasil Rekomendasi

Bagian atas dari implementasi merupakan detail buku yang dipilih oleh *user*. Di bawah detail buku tersebut, adalah buku-buku lain yang pernah dipinjam oleh *user* lain yang memiliki kemiripan dengan *user* tersebut. Apabila *user* merasa rekomendasi yang diberikan terlalu banyak atau kurang relevan, *user* dapat melakukan *filtering* berdasarkan subyek buku yang letaknya ada di pojok kanan atas.

5.3 Implementasi Database

Data yang digunakan pada *database* disini adalah data-data yang telah diberikan dan disetujui oleh pihak perpustakaan Universitas Brawijaya. Namun demi efisiensi dan keterbatasan perangkat yang digunakan, tidak semua data digunakan. Untuk data mahasiswa, dari sekitar 27.000 data hanya diambil 767 dari 86 jurusan di 15 fakultas yang ada di Universitas Brawijaya. Masing-masing diambil maksimal 10 mahasiswa, karena terdapat data mahasiswa dari jurusan tertentu yang datanya kurang dari 10. Sebagai contoh, jurusan Perpajakan dari Fakultas Ekonomi dan Bisnis hanya memiliki satu mahasiswa, yakni Satrio Bagus Prokoso dengan NIM 125030407111085. Tentu bukan berarti mahasiswa jurusan

Perpajakan hanya satu mahasiswa. Juga bukan berarti hanya ada satu mahasiswa dari jurusan Perpajakan yang pernah meminjam buku di perpustakaan. Hal ini terjadi karena ketidaklengkapan data yang diberikan oleh pihak perpustakaan Universitas Brawijaya, sehingga hanya ada satu mahasiswa dari jurusan Perpajakan yang datanya tersimpan.

Untuk data buku, pihak perpustakaan Universitas Brawijaya memberikan data kurang lebih 15.000 buku. Data yang diambil adalah 713 buku dimana pemilihan buku berdasarkan subyek buku. Dari banyaknya subyek buku yang ada, diambil 103 subyek dimana setiap subyek berkisar antara 4-5 buku. Pengecualian untuk buku-buku umum seperti "*Management*" dan "*Entrepreneur*" diambil hingga 10 buku, karena hampir di semua jurusan memiliki mata kuliah tersebut.

Untuk data peminjaman buku oleh mahasiswa, dari data yang diberikan oleh pihak perpustakaan Universitas Brawijaya memiliki banyak ketidakcocokan. Banyak buku yang ada di data peminjaman tidak ada di data buku. Begitupun data mahasiswa. Oleh karena itu, untuk data peminjaman digunakan data *dummy* dengan berbagai kondisi. Nantinya hasil pengujian dari berbagai kondisi tersebut akan ada di bab 6, Pengujian dan Analisis.

Data-data tersebut diimplementasikan ke dalam database yang berisikan tiga tabel, yakni tabel data_buku, data_mahasiswa dan peminjaman. Gambar 5.3 berikut menunjukkan kolom-kolom yang ada di tabel data_buku.

Column Name	Data Type	Length
id	int	11
register	varchar	10
title	text	
dewey_no	varchar	50
publisher	varchar	50
year_pub	int	4
creator	varchar	50
subject	varchar	50
ISBN	varchar	20
status	varchar	100

Gambar 5.3 Tabel data_buku

Data-data tersebut merupakan identitas buku. *ID* merupakan identitas buku di database. *Register* merupakan nomer registrasi buku di perpustakaan. *Title* adalah judul buku. *Dewey_no* merupakan nomor *Dewey Decimal Classification* (DDC) dari buku tersebut. *Publisher* merupakan penerbit dari buku. *Year_pub* merupakan tahun terbit buku. *Creator* adalah penulis dari buku. *Subject* adalah subyek atau topik dari buku. *ISBN* merupakan nomor *International Standard Book Number* dari buku. Status adalah kolom untuk melakukan *update* data *temporary* ketika *user* memilih salah satu buku di hasil pencarian.

Gambar 5.4 menunjukkan kolom-kolom yang terdapat pada tabel data_mahasiswa. Total terdapat 4 kolom pada tabel ini.

Column Name	Data Type	Length
nim	varchar	15
nama	varchar	50
jurusan	varchar	50
fakultas	varchar	50

Gambar 5.4 Tabel data_mahasiswa

Tabel data_mahasiswa berisikan empat kolom. Kolom-kolom tersebut merupakan identitas mahasiswa berupa Nomor Induk Mahasiswa (NIM), nama, jurusan dan fakultas.

Gambar 5.5 menunjukkan kolom-kolom yang terdapat pada tabel peminjaman. Total terdapat 7 kolom pada tabel ini.

Column Name	Data Type	Length
id	int	11
nim	varchar	15
nama	varchar	50
register	varchar	9
title	text	
tanggal_pinjam	timestamp	
status	varchar	100

Gambar 5.5 Tabel peminjaman

Tabel peminjaman merupakan tabel yang menyimpan data-data buku yang pernah dipinjam oleh mahasiswa. Tabel ini berisikan 7 kolom, dimana beberapa kolom diambil dari tabel data_buku (*register* dan *title*) dan tabel data_mahasiswa (nim dan nama).

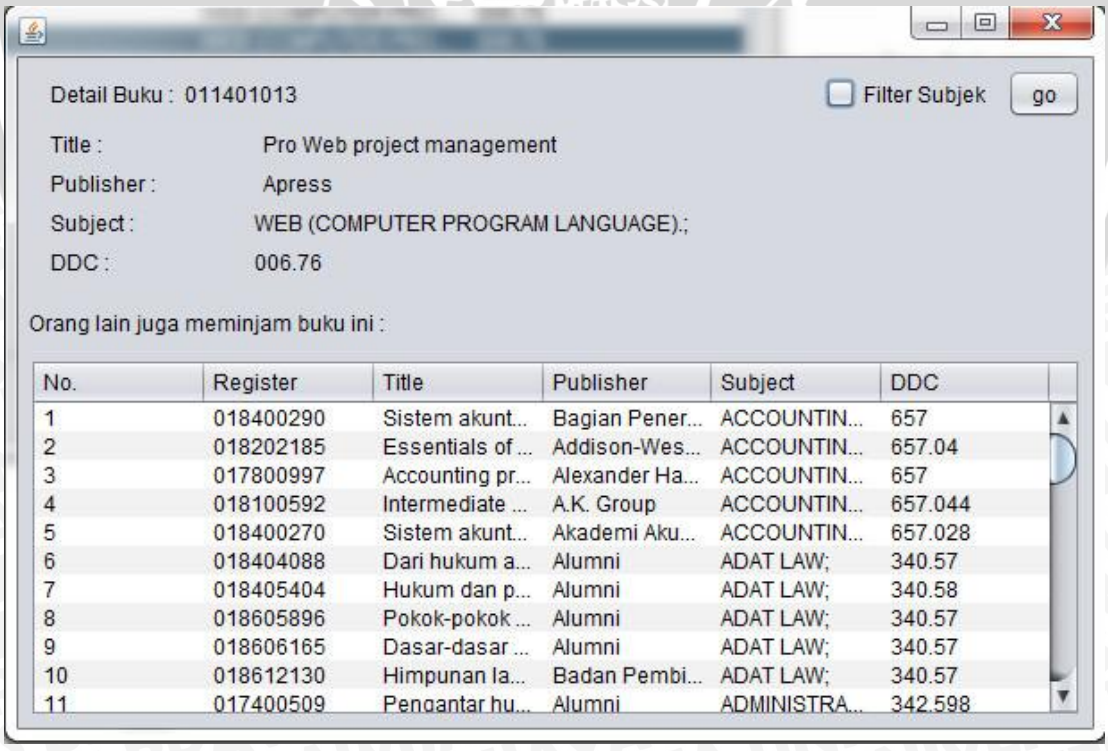
BAB 6 ANALISIS DAN PENGUJIAN

Bab ini menjelaskan mengenai hasil analisa dan pengujian dari proses implementasi percancangan yang telah dilakukan. Analisa dilakukan dengan cara mencari hasil rekomendasi yang dihasilkan dengan berbagai macam kondisi. Pengujian dilakukan dengan cara membandingkan DDC buku yang dipilih dengan DDC buku hasil rekomendasi.

6.1 Pengujian Implementasi dengan Berbagai Macam Kondisi

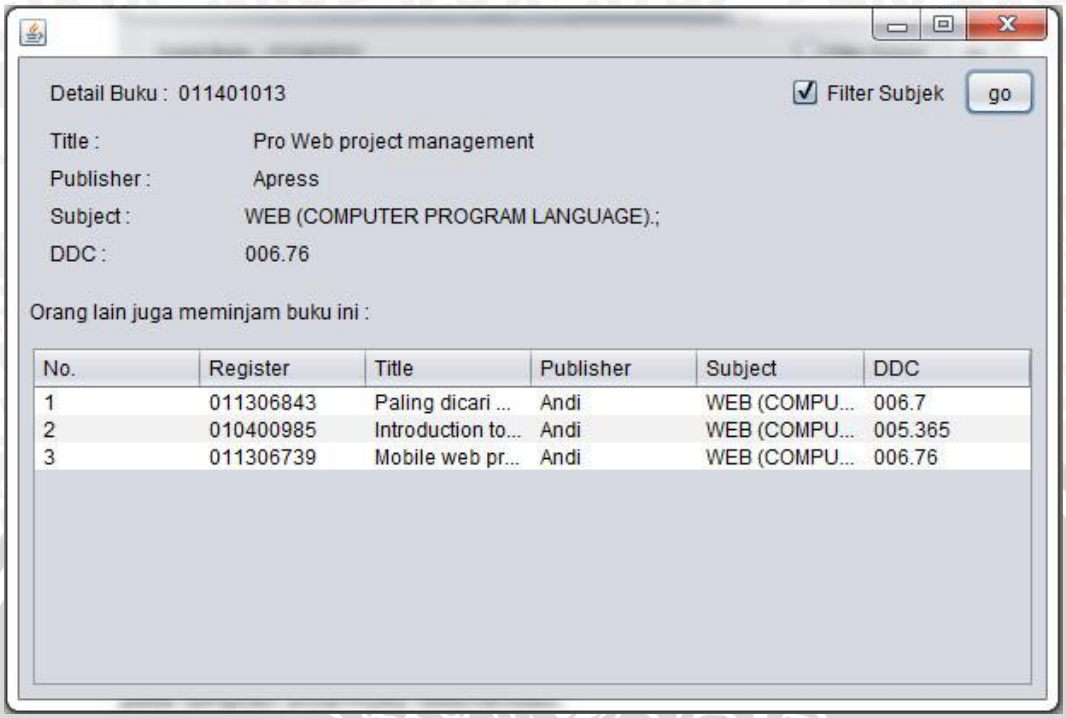
Seperti yang telah disebutkan di Bab 3, pengujian ini bertujuan untuk mengetahui berbagai macam kondisi yang mungkin terjadi pada implementasi. Sehingga dapat diketahui bagaimana kondisi ideal agar rekomendasi dapat dihasilkan dan mengetahui juga kondisi-kondisi seperti apa yang membuat implementasi tidak dapat menghasilkan rekomendasi.

Data peminjaman yang digunakan disini merupakan data *dummy*, dimana data tersebut dibuat sedemikian rupa sehingga menghasilkan berbagai macam kondisi. Waktu eksekusi berkisar antara 45-50 detik untuk tiap proses. Apabila sistem memproses lebih dari itu, maka kemungkinan tidak ditemukan *user* yang mirip atau kondisi-kondisi lainnya. Tidak ditemukannya *user* yang mirip mengakibatkan tidak munculnya rekomendasi. Jika hal tersebut terjadi, maka sistem akan menampilkan semua buku yang ada seperti pada Gambar 6.1.



Gambar 6.1 Contoh Ketika Menampilkan Semua Rekomendasi

Untuk mengantisipasi hal tersebut, maka ditambahkan fungsi filter subyek pada tampilan antarmuka rekemendasi seperti Gambar 6.2.



Gambar 6.2 Contoh Ketika DIlakukan Filter Subyek

Terdapat 12 kondisi yang ada pada analisa ini dan terbagi menjadi empat sub-bab.

6.1.1 Kondisi dari Sisi Mahasiswa

Kondisi dari sisi mahasiswa menunjukkan beberapa kondisi yang mungkin terjadi dari sisi mahasiswa seperti tidak pernah meminjam atau hanya pernah sekali meminjam.

1. Kondisi Mahasiswa Tidak Pernah Meminjam Buku

   Kondisi ini terjadi apabila ada *user* yang sebelumnya belum pernah melakukan peminjaman buku sama sekali. Pada kondisi ini digunakan data mahasiswa bernama Frans Agum Gumelar dari jurusan Teknik Informatika. Hasilnya adalah munculnya semua buku yang ada di *database*. Hal ini terjadi karena tidak ditemukannya *user* yang memiliki kemiripan dengan dirinya. Ini merupakan salah satu kekurangan metode *User-Based Collaborative Filtering*.

2. Kondisi Mahasiswa Hanya Pernah Meminjam Satu Buku

   Kondisi ketika *user* telah melakukan peminjaman sebelumnya setidaknya satu buku. Pada kondisi ini digunakan data mahasiswa bernama Dio Saputra Kudori dari jurusan Teknik Informatika dan *user* tersebut pernah meminjam buku bersubyek *Computer* berjudul Pengenalan Komputer. Dari beberapa percobaan yang telah dilakukan, sistem berhasil

menghasilkan rekomendasi bergantung pada buku yang dipilih. Percobaan pertama menghasilkan satu rekomendasi, percobaan kedua menghasilkan dua rekomendasi dan percobaan ketiga menghasilkan tiga rekomendasi. Ini membuktikan bahwa buku yang dipilih dimasukkan ke dalam perhitungan sistem.

3. Kondisi Mahasiswa Sering Meminjam Buku yang Subyeknya Sama

   Kondisi dimana *user* melakukan banyak peminjaman namun buku yang dipinjam memiliki subyek yang sama. Pada kondisi ini digunakan data mahasiswa bernama Fanandi Prima Ratriansyah yang pernah meminjam lima buku bersubyek *Java (Computer Program Languange)*. Hasilnya sama seperti sebelumnya, menampilkan beberapa rekomendasi yang berbeda sesuai dengan buku yang dipilih. Namun karena semua buku yang bersubyek *Java (Computer Program Languange)* sudah pernah dipinjam, maka buku yang direkomendasikan adalah buku yang bersubyek lain. Selain itu karena buku yang dipinjam bersubyek sama, maka rekomendasi yang diberikan seringkali sama.

4. Kondisi Mahasiswa Sering Meminjam Buku yang Subyeknya Berbeda

   Kondisi dimana mahasiswa sering meminjam buku namun subyek buku yang dipinjam berbeda-beda. Pada kondisi ini digunakan data mahasiswi bernama Edelin Yoda Bernadifta dari jurusan Teknik Informatika yang pernah meminjam lima buku dari lima subyek yang berbeda. Dengan variatifnya buku yang pernah dipinjam, maka perhitungan nilai *similarity* antar *user* makin variatif.

6.1.2 Kondisi dari Sisi Buku

Kondisi dari sisi buku merupakan beberapa kondisi dilihat dari sisi buku seperti buku tersebut belum pernah dipinjam atau hanya pernah sekali dipinjam.

1. Kondisi Buku Belum Pernah Dipinjam

   Kondisi dimana terdapat buku yang belum pernah dipinjam sama sekali oleh siapapun. Pada kondisi ini digunakan data buku berjudul “Profil Perusahaan Berbadan Hukum Jawa Timur” yang sebelumnya belum pernah dipinjam. Dari beberapa percobaan, dapat disimpulkan bahwa pada kondisi buku belum pernah dipinjam, sistem tetap dapat menampilkan rekomendasi asalkan *user* tersebut masih memiliki *user* lain yang memiliki nilai *similarity*.

2. Kondisi Buku Hanya Pernah Dipinjam Sekali

   Kondisi dimana terdapat buku yang hanya pernah dipinjam satu kali. Pada kondisi ini digunakan data buku berjudul “BASIC” yang hanya pernah sekali dipinjam oleh mahasiswa bernama Abdul Malik Mukhtar dari jurusan Teknik Komputer. *User* lain dikondisikan ingin meminjam buku tersebut. Hasilnya, sistem dapat menampilkan rekomendasi jika *user* pernah meminjam buku lain dan buku tersebut juga pernah dipinjam oleh *user* lain.

Hal tersebut terjadi karena apabila buku lain yang pernah dipinjam oleh *user* juga hanya pernah dipinjam sekali, maka tidak akan ditemukan *user* dengan nilai *similarity* tertinggi.

3. Kondisi Buku Sering Dipinjam

   Kondisi dimana terdapat sebuah buku yang sering dipinjam oleh *user* yang memiliki jurusan yang sama. Pada kondisi ini digunakan data buku berjudul "Algoritma & Pemrograman Menggunakan Java". Dari percobaan dapat disimpulkan bahwa rekomendasi akan muncul apabila ada *user* lain yang memiliki kemiripan. Seringnya buku dipinjam oleh *user* lain akan tidak berpengaruh apabila *user* tidak memiliki kemiripan.

6.1.3 Kondisi dari Sisi Jurusan dan Subyek Buku

Kondisi ini berkaitan dengan relasi antara jurusan dari mahasiswa dan subyek dari buku, seperti subyek buku apa yang dipinjam oleh mahasiswa satu jurusan.

1. Kondisi Satu Jurusan Tidak Pernah Meminjam

   Kondisi dimana tidak pernah ada mahasiswa dari satu jurusan yang pernah melakukan peminjaman. Pada kondisi ini digunakan data-data mahasiswa dari jurusan Teknologi Hasil Pertanian. Hasilnya adalah tidak ada satupun rekomendasi yang dihasilkan. Hal ini seperti yang sudah dijelaskan sebelumnya, terjadi karena tidak adanya *user* lain yang memiliki kemiripan.

2. Kondisi Satu Jurusan Meminjam Hanya Satu Subyek Buku yang Sama

   Kondisi dimana mahasiswa pada satu jurusan hanya pernah meminjam buku dengan subyek yang sama. Pada kondisi ini digunakan data-data mahasiswa dari jurusan Fisika dan hanya pernah meminjam buku dengan subyek *Physic*. Dalam kondisi seperti ini, DDC buku antara yang dipilih dan yang direkomendasikan hampir selalu sama. Nilai DDC yang sama menunjukkan keakuratan buku yang direkomendasikan dengan buku yang dipilih oleh *user*. Pengecualian terjadi apabila terdapat buku-buku yang satu subyek, namun nomor DDC-nya berbeda. Sebagai contoh di subyek *Computer*, ada buku dengan nomor DDC 004 dan ada juga buku dengan nomor DDC 001.

3. Kondisi Satu Jurusan Meminjam Banyak Subyek Buku yang Berbeda

   Kondisi dimana mahasiswa pada satu jurusan meminjam buku dari berbagai macam subyek buku yang ada. Pada kondisi ini digunakan data-data mahasiswa jurusan Hukum dengan lima subyek buku yang berbeda. Dengan variatifnya baik subyek, maka makin besar kemungkinan nomor DDC buku yang direkomendasikan pun berbeda dengan buku yang dipilih. Namun jika buku tersebut masih memiliki keterkaitan, misal seputar hukum, maka nomor DDC buku yang direkomendasikan akan berdekatan. Sebagai contoh, buku yang dipilih memiliki nomor DDC 345, maka nomor DDC buku yang direkomendasikan adalah 346 atau 347. Dari sini juga dapat

disimpulkan bahwa semakin tinggi transaksi yang dilakukan, semakin variatif buku yang direkomendasikan.

4. Kondisi Satu Jurusan Meminjam Subyek Buku yang Semuanya Berbeda

   Kondisi dimana mahasiswa pada satu jurusan tidak pernah meminjam buku dengan subyek yang berbeda. Pada kondisi ini digunakan data-data mahasiswa dari jurusan Teknologi Industri Pertanian dan meminjam 10 subyek buku yang berbeda. Hasilnya, sistem tidak dapat menampilkan rekomendasi karena tidak ditemukannya *user* yang memiliki kemiripan.

6.1.4 Kondisi Lain-Lain

1. Kondisi Satu Jurusan Tidak Ada yang Pernah Meminjam Buku yang Sama

   Kondisi dimana mahasiswa di satu jurusan pernah meminjam buku, namun semua buku yang dipinjam berbeda. Pada kondisi ini digunakan data mahasiswa-mahasiswa jurusan Teknik Sipil. Hasil penelitiannya tidak menghasilkan rekomendasi karena tidak ditemukannya *user* yang mirip.

2. Kondisi Dimana Terdapat *User* yang Nilai *Similarity*-nya 1

   Kondisi dimana terdapat *user* menemukan *user* lain yang nilai *similarity*-nya 1. Artinya, buku-buku yang pernah dipinjam oleh kedua *user* tersebut sama. Pada kondisi ini digunakan dua data mahasiswa jurusan Teknik Informatika, yakni Sofri Nada Cristanto dan Fendy Yulianto. Kemiripan yang sempurna mengakibatkan ketika salah satu dari *user* tersebut memilih satu buku, maka akan ditemukan *user* satunya memiliki nilai *similarity* paling tinggi. Padahal, jumlah buku yang pernah dipinjam sama persis. Akibatnya, tidak ada buku yang direkomendasikan.

3. Kondisi Satu Jurusan Meminjam Buku yang Tidak Berhubungan dengan Jurusannya

   Kondisi dimana terdapat *user* yang meminjam buku di luar bidangnya. Di luar bidang disini dalam artian meminjam buku yang subyeknya sama sekali tidak berhubungan dengan jurusannya. Misalkan mahasiswa jurusan Teknik Informatika meminjam buku bersubyek *Botany*. Selama *user* memiliki *user* lain yang mirip, hal ini bukan masalah. Hanya saja, buku yang direkomendasikan sama sekali tidak memiliki keterkaitan dengan buku yang dipilih oleh *user*.

4. Kondisi Dimana *User* Memilih Buku yang Pernah Dipinjam

   Kondisi ini terjadi ketika *user* memilih buku yang pernah dipinjam di hasil pencarian. Pada kondisi ini digunakan data mahasiswa bernama Fanandi Prima Ratriansyah dari jurusan Teknik Informatika yang pernah meminjam lima buku bersubyek *Java (Computer Program Languange)*. Di tampilan pencarian, *user* tersebut memilih buku Java yang pernah dipinjamnya. Sistem tetap dapat menampilkan rekomendasi, hanya saja buku-buku yang pernah dipinjamnya tidak akan muncul lagi.

6.2 Pengujian Akurasi dengan Metode *Mean Absolute Error*

Seperti yang sudah dijelaskan pada bab 2, pengujian akurasi dilakukan dengan menggunakan metode *Mean Absolute Error*. Terdapat dua variabel yang diperlukan dalam menghitung metode ini, yakni N_m dan N. N_m merupakan variabel yang melambangkan rekomendasi yang cocok sesuai dengan data uji, sedangkan N melambangkan jumlah keseluruhan data. Variabel N_m disini merupakan hasil rekomendasi yang nomor DDC-nya sama dengan buku yang dipilih. Hasil pengujian akurasi terdapat pada Tabel 6.1.

Tabel 6.1 Tabel Perbandingan Nomor DDC

User	Jurusan	Tiga Digit Awal DDC Buku yang Dipilih	Tiga Digit Awal DDC Hasil Rekomendasi
DIO SAPUTRA KUDORI	S1 TEKNIK INFORMATIKA	005	004
		001	005
			629
		004	005
FANANDI PRIMA RATRIANSYAH		004	004
			004
		004	005
			005
		004	004
			004
			005
		005	005
		005	005
		005	005
EDELIN YODA BERNADIFTA		005	005
			005
			005
		005	629
ZIYA EL ARIEF		005	629

			004
DEVITA FITRI TOUFANI	S1 ILMU DAN TEKNOLOGI PANGAN	641	641
			641
RIZKI MAHARDIAN	S1 ILMU PEMERINTAHAN	352	352
LIKA BEACTRIK SIAHAAN	S1 KETEKNIKAN PERTANIAN	581	630
			630
			581
			581
HALIMATUS SAKDIYAH	S1 ADMINISTRASI PERPAJAKAN	342	345
MUAMMAR ARDLI HAFIID	S1 ARSITEKTUR	720	720
DEVINA STELLA NABIGHAH	S1 AKUNTANSI	657	657
RADITYO ARI NUGROHO	S1 BIOLOGI	574	574
CHRISTIAN RICHARD	S1 BISNIS INTERNASIONAL	510	650
MARISA ALFIA	S1 EKONOMI PEMBANGUNAN	330	510
FITRIA NING ROSITA	S1 FISIKA	530	530
			530
REYGAMA DWISETYO	S1 HUBUNGAN INTERNASIONAL	327	341
			327
IRFAN MAULANA	S1 ILMU ADMINISTRASI PUBLIK	350	350
ARIA YUNIARTI	S1 ILMU HUKUM	340	346
			347
			340
NURUL ISLAMIYAH	S1 ILMU KOMUNIKASI	380	617
			380
BRENDA SAMPURNO PUTRI	S1 KIMIA	541	541
			540

SELVI SETYA EZI	S1 MANAJEMEN	658	658
			658
THERESIA AYU K L	S1 MATEMATIKA	510	510
			510
RARA LINGGA MARDIANI PRAGASIWI	S1 PSIKOLOGI	150	155

Dari Tabel 6.1, dapat dilihat hasil dari beberapa rekomendasi yang telah diberikan. Satu kali rekomendasi bisa menghasilkan beberapa buku tergantung nilai *similarity* dan bobot buku. Seperti yang telah disebutkan pada Bab 2, pengujian akan membandingkan DDC buku hasil rekomendasi dengan buku yang telah dipilih oleh *user*. Pengujian ini akan dibandingkan mulai dari perbandingan satu digit, dua digit dan tiga digit, dimana semakin banyak digit yang digunakan maka semakin rinci klasifikasi buku yang dibandingkan.

Dari 50 data hasil rekomendasi, jika digit pertama DDC yang dibandingkan, terdapat 42 buku hasil rekomendasi yang sama dengan buku yang dipilih oleh *user*. Jika dihitung dengan menggunakan perhitungan *Mean Absolute Error*, maka akan menjadi:

$$Akurasi = \frac{\mathrm{Nm}}{N} \times 100\%$$
$$= \frac{42}{50} \times 100\%$$
$$= 84\%$$

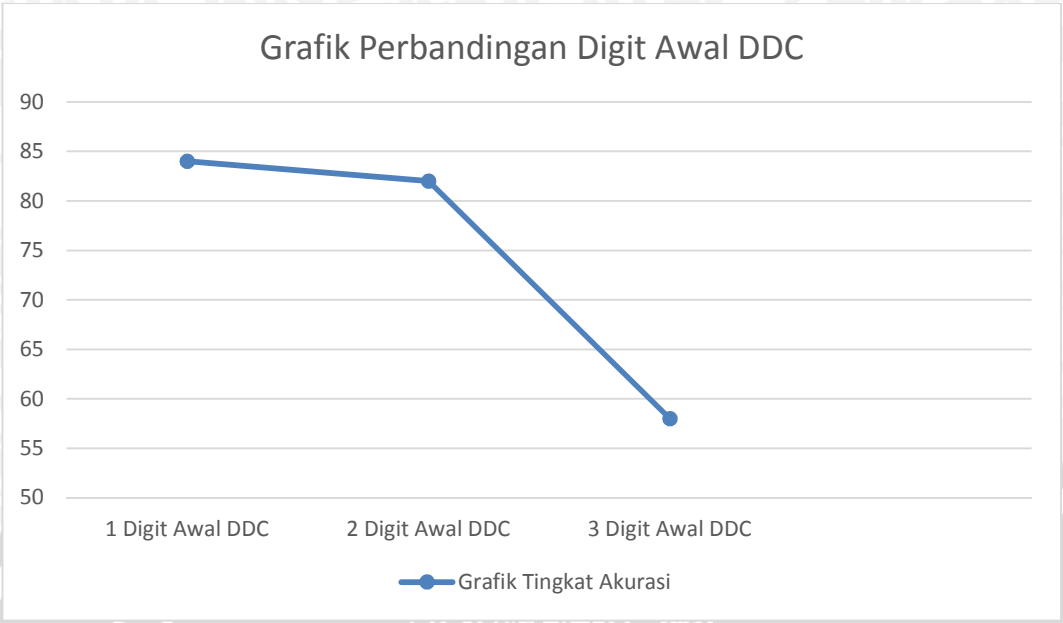
Jika yang dibandingkan adalah dua digit awal, maka terdapat 41 buku hasil rekomendasi yang sama dengan buku yang dipilih oleh *user*. Jika dihitung dengan menggunakan perhitungan *Mean Adjusted Score*, maka akan menjadi:

$$Akurasi = \frac{\mathrm{Nm}}{N} \times 100\%$$
$$= \frac{41}{50} \times 100\%$$
$$= 84\%$$

Lalu yang terakhir adalah membandingkan tiga digit awal DDC. Dari hasil pengujian, terdapat 29 buku hasil rekomendasi yang memiliki kesamaan dengan buku yang dipilih. Artinya jika kita masukkan ke dalam perhitungan *Mean Absolute Error*, maka akan menjadi:

$$Akurasi = \frac{\mathrm{Nm}}{N} \times 100\%$$
$$= \frac{29}{50} \times 100\%$$
$$= 58\%$$

Jika digambarkan dalam bentuk tabel, maka tingkat akurasi dari perbandingan DDC akan tampak seperti pada Gambar 6.3.



Gambar 6.3 Grafik Tingkat Akurasi dengan Perbandingan Digit Awal DDC

Dari Gambar 6.3 dapat disimpulkan bahwa semakin banyak digit awal yang dilakukan untuk melakukan perbandingan, maka akan semakin rendah hasil akurasinya. Hal ini terjadi karena semakin rinci digit DDC-nya, maka klasifikasi bukunya pun akan semakin rinci. Sehingga, banyak buku yang memiliki kelas utama yang sama (ditunjukkan oleh satu digit awal DDC), akan tetapi memiliki klasifikasi lebih detail yang berbeda.

Hasil perhitungan akurasi juga dipengaruhi oleh banyak faktor. Akurasi akan lebih tinggi jika mahasiswa satu jurusan hanya pernah meminjam satu buku subyek yang sama. Bisa juga berbeda subyek namun memilki tiga digit awal nomor DDC yang sama. Nilai akurasi yang dihasilkan terjadi karena metode ini sama sekali tidak melihat sisi *item*, hanya melihat dari sisi *user*. Hal tersebut menyebabkan ketika metode ini diimplementasikan dalam kondisi mahasiswa yang meminjam berbagai subyek seperti pada beberapa pengujian yang telah dilakukan, maka hasil akurasinya tidak terlalu tinggi.

BAB 7 PENUTUP

Bab ini terdiri dari 2 subbab, yakni kesimpulan dan saran. Kesimpulan menjelaskan mengenai apa yang bisa ditarik dari analisis dan pengujian dari implementasi yang telah dilakukan. Saran menjelaskan mengenai apa yang akan disarankan oleh penulis untuk ke depannya agar penelitian ini bisa lebih berkembang.

7.1 Kesimpulan

Terdapat beberapa hal yang dapat disimpulkan dari proses analisis dan pengujian dari implementasi metode *User-Based Collaborative Filtering* terhadap rekomendasi buku di perpustakaan.

1. Metode *User-Based Collaborative Filtering* telah diimplementasikan sebagai rekomendasi buku di perpustakaan. Rekomendasi yang diberikan berdasarkan kemiripan *user*, sehingga nanti rekomendasi dapat diberikan sebagai "Orang lain juga membaca buku ini".
2. Kondisi yang dapat mendukung implementasi metode ini adalah adanya *user* yang memiliki nilai *similarity*. Buku maupun subyek buku tidak mempengaruhi hasil rekomendasi yang muncul, asalkan ada *user* yang memiliki kemiripan. Sedangkan kondisi yang membuat implementasi metode ini tidak menghasilkan rekomendasi adalah tidak adanya *user* yang memiliki kemiripan. Pada kondisi seperti ini implementasi metode akan menampilkan semua buku yang nantinya bisa di-*filter* sesuai subyek buku yang dipilih.
3. Akurasi diperoleh dari perbandingan nomor DDC antara buku yang dipilih dengan buku yang direkomendasikan. Jika yang dibandingkan adalah satu digit awal maka tingkat akurasinya adalah 84%. Jika yang dibandingkan adalah dua digit awal maka tingkat akurasinya adalah 82%. Jika yang dibandingkan adalah tiga digit awal maka tingkat akurasinya adalah 58%. Hal ini menunjukkan semakin detail digit DDC yang dibandingkan, maka semakin rendah tingkat akurasinya karena klasifikasinya juga semakin detail.

7.2 Saran

Saran dituliskan disini dengan harapan apabila di masa depan ada orang lain yang tertarik dengan topik ini, dapat mengembangkan penelitian ini agar menjadi lebih baik.

1. Rata-rata sistem mengeksekusi satu kali proses adalah 47 detik. Inipun dikarenakan adanya proses *filtering* jurusan di awal proses. Ke depannya diharapkan adanya improvisasi agar implemetnasi metode dapat lebih optimal.

2. Adanya penambahan metode yang melihat dari sisi *item* untuk mengatasi kekurangan-kekurangan metode *User-Based Collaborative Filtering* yang telah disebutkan di kesimpulan.

DAFTAR PUSTAKA

Adomavicius, G. (2005). Toward the Next Generation of Recommender Systems : A Survey of the State-of-the-Art and Possible Extensions. *IEEE*, 734-749.

Crespo, R.G., et.al. (2011). Recommendation System based on User Interaction Data Applied to Intelligent Electronic Books. *Elsevier*, 1445-1449.

Fernandez, Y. B., et.al. (2011). Explorin Synergies Between Content-Based Filtering and Spreading Activation Techniques in Knowledge-Based Recommender Sytems. *Elsevier*, 4823-4840.

Hart, M. (2012). *100 Tokoh Paling Berpengaruh di Dunia.* Jakarta: Noura Books.

Irfan, M., Dwi, A., Hastarita, F. (2014). Sistem Rekomendasi : Buku Online dengan Metode Collaborative Filtering. *Jurnal Teknologi Technoscientia*, 76-84.

Jannach, D. e. (2011). Recommender Systems : An Introduction. *Cambridge University Press*.

Kriswijayanti, R. (2009). Menumbuhkan Minat Baca Pada Mahasiswa. *S1*, Universitas Diponegoro.

Kwon, H. J. (2009). Improved Memory-Based Collaborative Filtering Using Entropi-Based Similiarity Measures. *2009 International Symposium on Web Information Systems and Applications.*

Masruri, F., Mahmudy, W. F. (2007). Personalisasi Web E-Commerce Menggunakan Recommender System dengan Metode Item-Based Collaborative Filtering. *Kursor*, 1-12.

Oktoria, R., Maharani, W., Firdaus, Y. (2010). Content-based Recommender System Mengunakan Algoritma Apriori. *Konferensi Nasional Sistem dan Informatika*, 124-129.

Putra, A. H. (2015). Sistem Rekomendasi Mata Kuliah Pilihan Mahasiswa dengan Content-Based Filtering dan Collaborative Filtering (Studi Kasus : Universitas Brawijaya). *S1*, Universitas Brawijaya.

Ricci, F., Rokach, L., Shapira, B. (2011). Introduction to Recommender Systems Handbook. *Springer Science+Business Media*, 1-35.

Safavian, S. R. (1990). A Survey of Decision Tree Classifier Methodology. *IEEE Trans. Systems, Man & Cyvernetics*, 1-44.

Sarwar, B., et.al. (2001). Item-based Collaborative Filtering Collaboration Filtering Recommendation Algorithms. *WWW10*, 1-15.

Sarwar, B.M., et.al. (2000). Application of Dimensionality Reduction in Recommender System -- A Case Study. *Department of Computer Science and Engineering, University of Mennesota*, 1-14.

Sohail, S.S., Siddiqui, J., Ali, R. (2015). OWA based Book Recommendation Technique. *Elsevier*, 126-133.

Subroto, G. (2009). Klasifikasi Bahan Pustaka. *Pustakawan Perpustakaan UM*, 1-13.

Suyanto. (2014). *Artificial Intelligence.* Bandung: Informatika.

Tsuji, K. et.al. (2014). Book Recommendation Based on Library Loan Records and Bibliographic Information. *Elsevier*, 478-486\.

Uyun, S., Fahrurrozi, I., Mulyanto, A. (2011). Item Collaborative Filtering untuk Rekomendasi Pembelian Buku Secara Online. *JUSI*, 63-70.

Wandi, N., Hendrawan, R.A., Mukhlason, A. (2012). Pengembangan Sistem Rekomendasi Penelusuran Buku dengan Penggalian Association Rule Menggunakan Algoritma Apriori. *Jurnal Teknis ITS Vol 1*.