BAB 3
METODE PENELITIAN

Penelitian ini adalah penelitian rancang-bangun. Penelitian berawal dari latar belakan permasalahan yang ada, memetakan proses-proses, mencari sumber permasalahan, dan akhirnya merancang dan mengembangkan suatu sistem yang dapat digunakan untuk mereduksi atau mengeliminasi permasalahan yang ada.

3.1 BAHAN DAN ALAT PENELITIAN

Bahan penelitian yang akan dibutuhkan adalah data buku berlabel dari Google Books berjumlah 8702 baris data dan untuk menghitung kemiripan kata dengan *Cosine Similarity* membutuhkan bagian data dari judul buku, dan jenis buku untuk proses klasifikasi daripada langkah berikutnya.

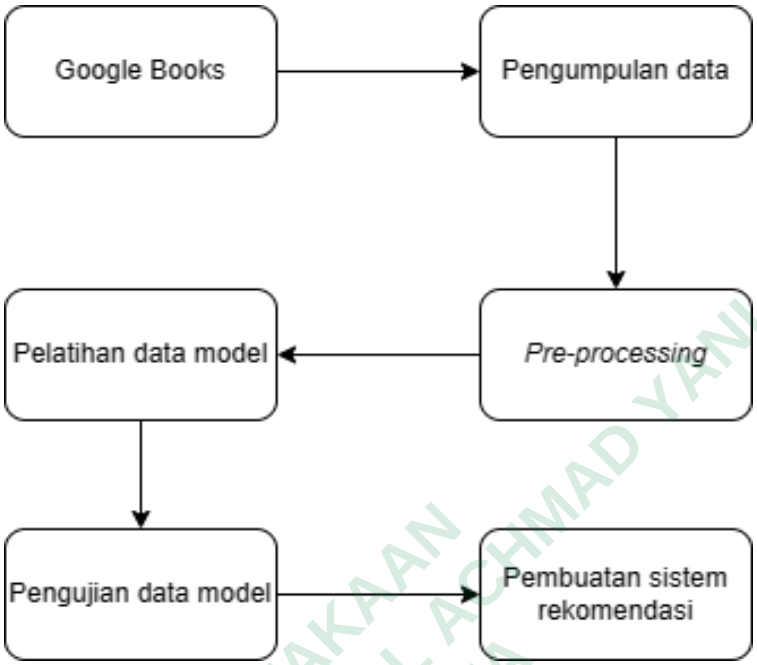
Alat yang digunakan dalam penelitian ini adalah komputer dengan spesifikasi cukup untuk menjalankan sistem operasi dan perangkat lunak pengembangan serta koneksitas Internet.

Sistem Operasi dan program-program aplikasi yang dipergunakan dalam dalam pengembangan aplikasi ini adalah:

1. Sistem Operasi: Windows 11
2. Microsoft Excel 2013
3. Database engine: MongoDB versi 4.4.6
4. Python versi 3.7
5. NodeJS versi 10.14.0
6. Visual Studio Code versi 1.76.0
7. Jupyter Notebook v2023.2.1000592019.

3.2 JALAN PENELITIAN

Penelitian ini menggunakan bahasa pemrograman Python dan Jupyter Notebook untuk melakukan pengolahan data dan penyesuaian model. Koleksi data disimpan ke dalam Microsoft Excel 2013 yang akan dimuat oleh *library* Pandas. Berikut alur dari penelitian ini, pada Gambar 3.1



Gambar 3.1. Jalan penelitian

Berikut merupakan tahapan-tahapan yang akan digunakan dalam penelitian ini.

1. Pengumpulan data

Pengumpulan data adalah tahap mengumpulkan data buku dari Google Books sejumlah 8702 data dari 319 kategori bukua melalui *platform* Kaggle (TEBBA, 2020). Data yang diperoleh disimpan dalam bentuk ekstensi (.csv) supaya mudah dalam pemuatan data menggunakan modulasi pandas oleh Python. Tabel 3.1 merupakan sampel dari tahapan ini

Tabel 3.1 Data tahap pengumpulan

Dokumen	Title	Category
d1	*Theories of Development: Concepts and Applications*	*Medical Books*
d2	*A History of Civilizations*	*History*
d3	*A River Runs Through It*	*Fiction*

2. *Pre-processing*

Pre-processing atau pra-pemrosesan adalah proses untuk mempersiapkan data menjadi lebih terukur dan merubah format data menjadi yang diperlukan untuk tahap berikutnya.

a. *Case folding* yaitu merubah bentuk awal ke bentuk standar seperti *lowercase*
b. *Number removal* yaitu menghilangkan unsur angka pada kalimat
c. *Punctuation functional* yaitu menghilangkan karakter yang tidak memiliki unsur signifikan
d. *Tokenizing* yaitu pemecahan kata dalam suatu kalimat yang didefinisikan sebagai pemisah kata atau bukan
e. *Stopword removal* yaitu menghilangkan kata sambung atau kata yang memiliki informasi rendah terhadap kalimat.
f. Transformasi data menggunakan TF-IDF untuk merubah bentuk data teks melalui pembobotan.

Selain urutan pra-pemrosesan yang telah diuraikan terdapat pra-pemrosesan lain yaitu mempersempit jumlah kategori buku menjadi tiga kategori dari sebelumnya kumpulan data ini memiliki 319 kategori. Logika alur dari pra-pemrosesan dalam mempersempit jumlah kategori sebagai berikut

a. Mengumpulkan korpus dari tiga kategori yang ditentukan. Misalnya untuk kategori *sains-tech* perlu judul buku yang memiliki pola *RegExp* yaitu *antropoda*, *social-humanaria* yaitu *business*, sedangkan *general* yaitu *marvel.*
b. Apabila kumpulan data yang dimiliki sudah melalui proses pertama, maka dilakukan logika kondisional yaitu memeriksa data pada kolom *category*. Misalnya untuk kategori *sains-tech* yaitu *Medical Books*, *social-humanaria* yaitu *history* sedangkan *general* yaitu *fiction.*
c. Kemudian apabila proses pertama dan kedua berakhir maka data disimpan ke dalam tipe data list untuk pendataan label buku dan menambah kolom baru yaitu *category_type*

Tabel 3.2 merupakan sampel dari hasil pra-pemrosesan sebelum sampai kepada tahapan transformasi data menggunakan TF-IDF

Tabel 3.2 Data setelah pra-pemrosesan

Dokumen	*Title*	*Text_cleaning*	*Category_type*
d1	*Theories of Development: Concepts and Applications*	*theory development concept application*	*Sains-tech*
d2	*A History of Civilizations*	*history civilization*	*Social-humanaria*
d3	*A River Runs Through It*	*river run*	*General*

Contoh dari penerapan TF, IDF dan TF-IDF dari persamaan (1), persamaan (2) dan persamaan (3) sebagai berikut.

$$Tf_{development} = 1/1 = 1$$

$$Idf_{development} = \log(\frac{3}{1}) = 0.477$$

Sehingga perolehan TF-IDF yaitu berdasarkan kepada persamaan (3) dengan perhitungan sebagai berikut.

$$Tf - idf_{development} = 1 \text{ X } 0.477 = 0.477$$

Tabel 3.3 merupakan implementasi dari perubahan bentuk teks oleh kolom text_cleaning yang terdapat di Tabel 3.2 kepada pembobotan TF-IDF.

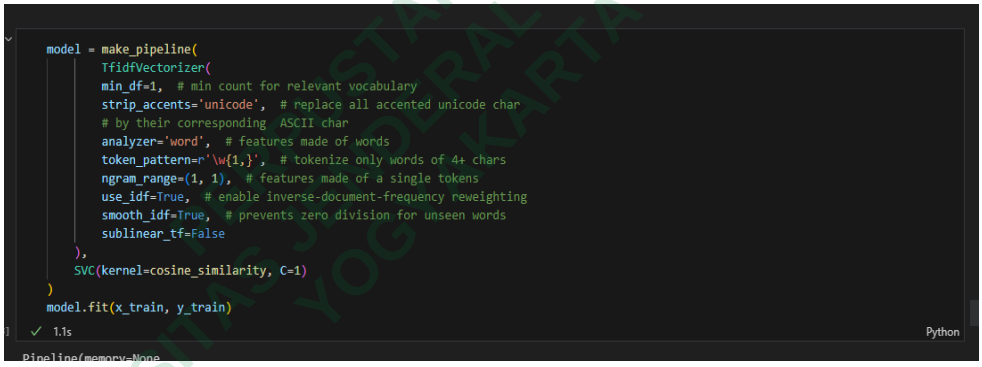
Tabel 3.3 Kalkulasi TF-IDF

Corpus	TF			IDF			TF-IDF		
	d1	d2	d3	d1	d2	d3	d1	d2	d3
development	1	0	0	0.477	0	0	0.477	0	0
concept	1	0	0	0.477	0	0	0.477	0	0
application	1	0	0	0.477	0	0	0.477	0	0
history	0	1	0	0	0.477	0	0	0.477	0
civilization	0	1	0	0	0.477	0	0	0.477	0
run	0	0	1	0	0	0.477	0	0	0.477

theory	1	0	0	0.477	0	0	0.477	0	0
river	0	0	1	0	0	0.477	0	0	0.477

3. Pelatihan data

Pelatihan data adalah penyesuaian data yang sudah dilakukan dari pemrosesan sebelumnya. Di dalam penelitian ini, metode yang digunakan adalah *Cosine Similarity* sebagai kernel dari algoritma klasifikasi dari salah satu implementasi modul *libsvm* yaitu C-Support Vector Classification (SVC). Pada pelatihan data, peneliti memisahkan hasil data pra-pemrosesan yaitu 8637 menjadi 6477 data latih. Kemudian peneliti membangun *Pipeline* untuk mengatur beberapa langkah yang dapat divalidasi dengan mengatur beberapa parameter yang berbeda. Dalam kasus penelitian ini, *pipeline* dibangun seperti Gambar 3.2



Gambar 3.2. *Pipeline* data

Gambar 3.2 menjelaskan bahwa model dibangun atas proses *pipeline* yang mengatur dan mengotomatisasi pergerakkan data pelatihan yang kemudian dikelola sesuai dengan parameter yang dilengkapi seperti pada Gambar 3.2. *Feature* dan *label* dari penyesuaian *pipeline* oleh data latih dengan parameter pertama yaitu mentransformasikan *feature* menjadi pembobotan TF-IDF. Kemudian membangun model SVC menggunakan kernel Cosine Similarity.

Selain kernel, membangun model dapat dipengaruhi dengan menentukan nilai dari parameter regulasi-L2 yaitu c. Penelitian ini membandingkan nilai c=0.05 dan c=1.0 untuk memilih hasil akurasi dari prediksi yang lebih baik. Hasil dari

transformasi atau pembobotan *feature* yang awalnya adalah bentuk teks yaitu judul buku diubah menjadi vektor dan ditransformasikan kembali oleh *Cosine Similarity* menggunakan seperangkat formulasi matematika, karena fungsi utama dari kernel pada SVC adalah menjadikan data sebagai masukkan dan dimanipulasi sehingga pemrosesan yang dibutuhkan terpenuhi. Kemudian parameter regulasi-L2 yaitu c dari SVC akan menentukan *hyperplane* dengan *margin* yang optimal. SVC sangat bergantung kepada besaran dari nilai parameter c, karena nilai ini meningkatkan kompleksitas kelas hipotesa dan penanganan *outlier*, mencegah *overfitting* dan pemilihan *hyperplane* tergantung dengan besaran tersebut.

Berikut skenario dari kalkulasi berdasarkan persamaan (5) *Cosine Similarity* setelah data ditransformasi menjadi pembobotan oleh TF-IDF maka dijelaskan pada Tabel 3.4 apabila terdapat masukkan judul buku ke sistem yaitu *Chemistry: Concepts & Applications, Student Edition*.

Tabel 3.4 Dokumen masukkan

Dokumen	Masukkan pengguna	Hasil
D4	*Chemistry: Concepts & Applications, Student Edition*	*chemistry concept application student edition*

Tabel 3.5 merupakan implementasi perubahan bentuk masukkan pengguna yaitu teks menjadi pembobotan TF-IDF. Pada masukkan pengguna, banyaknya dokumen yang mengandung kata tertentu terjadi peningkatan seperti contoh sebagai berikut

$$Tf_{development} = 1/1 = 1$$

$$Idf_{development} = \log(\frac{4}{2}) = 0.6$$

Sehingga perolehan TF-IDF yaitu berdasarkan kepada persamaan (3) dengan perhitungan sebagai berikut.

$$Tf - idf_{development} = 1 \text{ X } 0.6 = 0.60$$

Tabel 3.5 Kalkulasi TF-IDF masukkan

Corpus	TF	IDF	TF-IDF
	d4	d4	d4
development	0	0	0
concept	1	0.6	0.6
application	1	0.6	0.6
history	0	0	0
civilization	0	0	0
run	0	0	0
theory	0	0	0
river	0	0	0

Dari Tabel 3.3 dan Tabel 3.5 maka pembobotan TF-IDF seperti pada Tabel 3.6 dengan penjelasan bahwa kolom Data merupakan pembobotan TF-IDF dari beberapa judul buku sedangkan kolom Masukkan merupakan judul buku yang menjadi masukkan pengguna.

Tabel 3.6 Hasil pembobotan

Data	Masukkan
[[0.477, 0.477, 0.477, 0, 0, 0, 0.477, 0], [0, 0, 0, 0.477, 0.477, 0, 0, 0], [0, 0, 0, 0, 0, 0.477, 0, 0.477]]	`[[0, 0.6, 0.6, 0, 0, 0, 0, 0]]`

Setelah memperoleh hasil pembobotan kata dari masukkan pengguna, langkah berikutnya adalah menghitung sudut *kosinus* dari pembobotan tersebut. Contoh penerapan dari persamaan (5) sebagai berikut.

a. Menentukan nilai A dan B

   A = [0.477, 0.477, 0.477, 0, 0, 0, 0.477, 0]

   B = [0, 0.6, 0.6, 0, 0, 0, 0, 0]

b. Menentukan perkalian dari A x B

   A x B = 0.477 x 0 + 0.477 x 0.6 + …..

   A x B = 0.5724 atau 0.572

c. Menentukan nilai norma dari || A || , || B || dan || A || x || B ||

$|| A || = \sqrt{0.477^2 + 0.477^2 + \cdots} = 0.954$

$|| B || = \sqrt{0^2 + 0.6^2 + \cdots} = 0.848$

|| A || x || B || = 0.954 x 0.848 = 0.809

d. Menentukan nilai *kosinus* menggunakan persamaan (5)

$cos_{(\partial)}$ = 0.572 / 0.809

$cos_{(\partial)}$ = 0.707

α = 45.00865◦

Dari contoh penerapan tersebut maka *kosinus* dari setiap judul dengan masukkan pengguna dapat dilihat pada Tabel 3.7 Hasil *Cosine Similarity* sebagai berikut.

Tabel 3.7 Hasil *Cosine Similarity*

Text_cleaning	Hasil *kosinus*
theory development concept application	0.707
history civilization	0.0
river run	0.0

Berdasarkan Tabel 3.7, nilai masukkan judul buku yang diberikan yaitu *Chemistry: Concepts & Applications, Student Edition* memiliki keserupaan dengan judul buku *Theories of Development: Concepts and Applications* dengan nilai *Cosine Similarity* dari keduanya adalah 0.707 atau apabila diubah ke sudut *kosinus* maka nilai tersebut memiliki sudut 45.00865◦. Setelah mengubah bentuk pembobotan menjadi hasil *kosinus* dari tiga sampel yang digunakan, langkah berikutnya adalah menghitung *hyperplane* dari metode SVC dengan menggunakan persamaan (6) melalui contoh penerapan sebagai berikut.

a. Memberi pemisalan dari tiga kategori. Misalnya kategori *general* ditandai oleh angka 0, *sains-tech* ditandai oleh angka 1, sedangkan *social-humanaria* ditandai oleh angka 2

b. Menentukan *features* sebagai x dan *label* sebagai y dari data yang telah ditransformasikan ke dalam dimensi yang diperlukan seperti pada Tabel 3.8

Tabel 3.8 Menentukan x dan y

Features (x)	*Label* (y)
0.707	1
0	2
0	0

c. Mencari *hyperplane* menggunakan persamaan (6) dengan syarat yaitu perhitungan $w^t + b \geq 1$ sehingga diperoleh persamaan sebagai berikut
   (1) $0.707w_1 + b \geq 1$
   (2) $0w_1$ + b ≥ 1
   (3) -$0w_1 - b \geq 1$

d. *Subtitusi* dan eliminasi persamaan yang didapatkan dari tahapan sebelumnya yaitu menjumlahkan (1) dan (3) sehingga diperoleh nilai dari w_1 yaitu 2.8288 atau 2.829

e. *Subtitusi* nilai w_1 dengan persamaan (2) dan persamaan (3) dari tahapan c yaitu $0w_1$ + b ≥ 1 atau $0w_1$ - b ≥ 1 sehingga diperoleh nilai b yaitu 1 atau -1

f. Menentukan persamaan *hyperplane* dari nilai w_1 dan b yang telah dihitung sebelumnya menggunakan rumus $w_1x_1 + b = 0$ yaitu $2.829x_1$ + 1 = 0 atau $2.829x_1$ - 1 = 0

g. Menentukan hasil klasifikasi melalui persamaan *hyperplane* melalui Tabel 3.9 dan 3.10 dengan hasil klasfikasi dibulatkan dari belakang koma seperti berikut.

Tabel 3.9 Hasil klasifikasi dengan b=1

Data uji (x)	Kelas = Sign($2.829x_1$ + 1)
0.707	3 (tidak valid)
0	1
0	1

Tabel 3.10 Hasil klasifikasi b= -1

Data uji (x)	Kelas = Sign(2.829x_1 - 1)
0.707	1
0	-1 (tidak valid)
0	-1 (tidak valid)

Berdasarkan Tabel 3.9 dan Tabel 3.10, maka kecenderungan kelas 1 atau *sains-tech* menjadi hasil klasifikasi dari tiga sampel data yang telah dilakukan dari metode penelitian ini.

4. Pengujian data

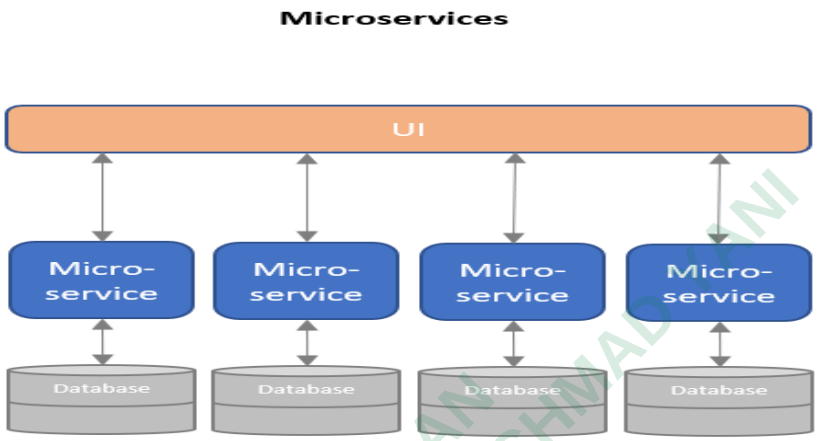
Pengujian data adalah tahapan untuk mengetahui tingkat keakuratan pemodelan yang dibangun pada tahap pelatihan data yang digunakan untuk memprediksi label atau kelas dari data uji yang tersedia. Jumlah dari data uji yang digunakan adalah 2160 data uji. Model yang sudah diperoleh kemudian dihitung menggunakan beberapa metode pada *confusion matrix* untuk mengetahui persentase setiap dilakukan pengujian. Metode yang digunakan seperti

a. *Accuracy* untuk mengetahui jumlah klasifikasi dibagi dengan total sampel testing yang diuji
b. *Precision* untuk mengetahui klasifikasi dari kategori dibagi dengan total sampel klasifikasi kategori tersebut
c. *Recall* untuk mengetahui sampel yang diklasifikasikan dalam kategori tertentu dibagi total sampel dalam testing yang berkategori tertentu
d. *F-measure* untuk menghitung rata-rata dari *precision* dan *recall*

5. Pembuatan sistem rekomendasi

Pembuatan sistem rekomendasi dilakukan setelah proses pengujian model. Pada tahap ini, model yang sudah dilatih dan diuji dapat diimplementasi ke dalam sistem dan berinteraksi dengan pengguna melalui masukkan dan memberikan keluaran hasil berupa buku yang direkomendasikan.

Dalam pembuatan sistem rekomendasi dari penelitian ini terdapat dua bagian yang menjadi arsitektur layanan mikro yaitu membangun layanan API dan antarmuka pengguna seperti pada Gambar 3.3.



Gambar 3.3 Arsitektur sistem

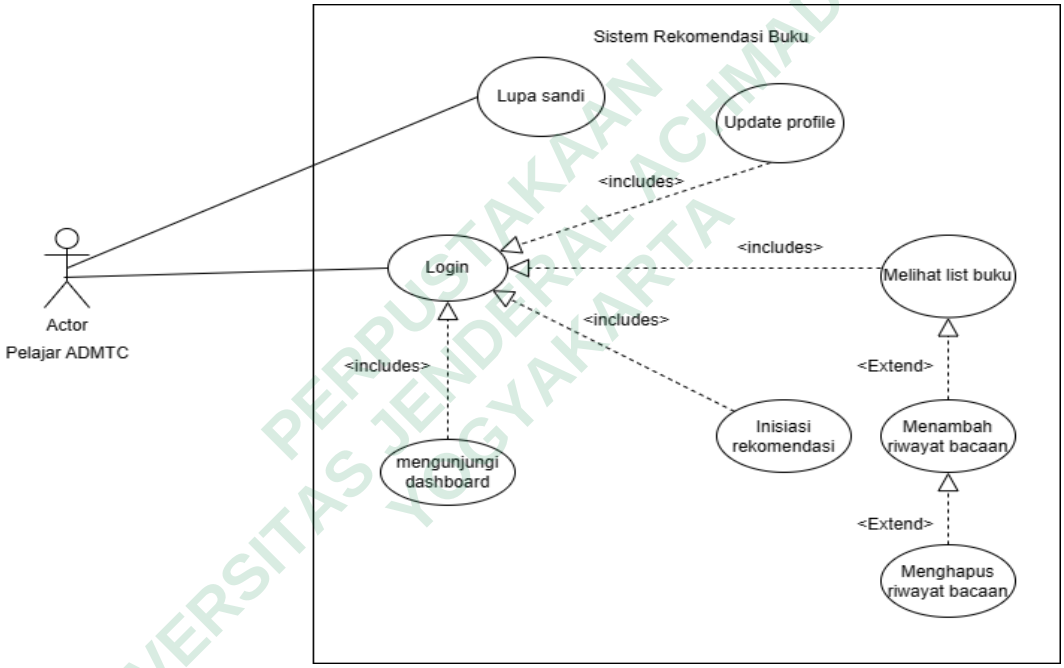
Pada Gambar 3.3 Arsitektur Sistem, bagian UI, *microservice* dan basis data terpisah. Untuk mengambil data, *microservice* bertanggungjawab atas komunikasi data oleh sistem seperti menambah, mengubah, menghapus dan menampilkan data. Di penelitian ini, layanan data disajikan oleh *microservice* GraphQL yang merupakan layanan API terbaru di dalam dunia pengembangan perangkat lunak. Untuk mengakses layanan API tersebut yaitu dengan cara mengetahui URL dari wadah penyimpanan API, menyiapkan skenario otentikasi dan otorisasi serta *header* apa saja yang diperlukan. Misalnya wadah dari layanan API GraphQL dapat diakses melalui "<*url_target*>*/graphql*", kemudian pengembang diarahkan kepada tampilan GraphQL dan memanfaatkan API tersebut untuk melakukan komunikasi data. GraphQL menggunakan konsep *query* untuk mengakses datanya, apabila ingin menampilkan keseluruhan data cukup memanggil *query* dan fungsinya beserta variabel data yang ingin ditampilkan, sehingga untuk pengembangan yang berfokus kepada tampilan *User Interface* (UI) maka data tersebut dapat dimanfaatkan sesuai kebutuhan.

3.3 METODE LAINNYA (MENYESUAIKAN DENGAN TOPIK)

Bagian ini menguraikan perancangan *Unified Modeling Language* (UML) dari sistem rekomendasi seperti *Use Case Diagram*, *Activity Diagram* dan *Class Diagram.*

3.3 1 Use Case Diagram

Gambar 3.4 merupakan *Use Case Diagram* yang digunakan dalam membagi aktor dan sistem. Rancangan ini menjelaskan bagaimana setiap *Use Case* memiliki dependensi satu sama lain.



Gambar 3.4 *Use Case* Diagram antar *Use Case* lain

Menurut penjelasan Gambar 3.4, keterangan *actor* dan *use case* dapat dijelaskan melalui Tabel 3.11 Keterangan *use case*

Tabel 3.11 Keterangan *use case*

Actor	*Use case*	Deskripsi
Pelajar	*Login*	Untuk melakukan proses otentikasi dan otorisasi

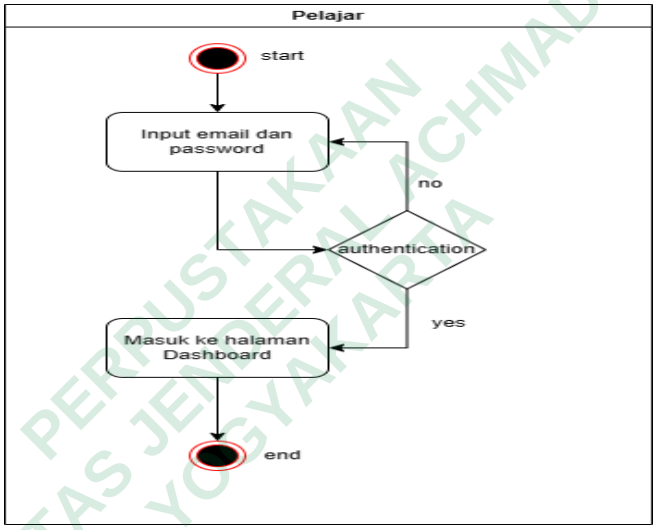
	Lupa sandi	Untuk mengubah kata sandi pengguna sebelum melakukan otentikasi dan otorisasi.
	Update profile	Untuk mengubah identitas pengguna seperti yang dikirimkan kepada formulir kepada layanan API
	Mengunjungi *dashboard*	Untuk menampilkan grafik, tabel hasil pemrosesan kata dan melakukan prediksi berdasarkan judul masukkan pengguna
	Inisiasi rekomendasi	Untuk melakukan rekomendasi buku berdasarkan buku bacaan matakuliah atau riwayat bacaan pelajar
	Melihat *list* buku	Untuk menampilkan keseluruhan dari data buku
	Menambah riwayat bacaan	Untuk menambah riwayat bacaan pengguna ketika pengguna membaca salah satu buku yang tersedia
	Menghapus riwayat bacaan	Untuk menghapus riwayat bacaan pengguna ketika pengguna tidak memerlukan buku tersebut.

3.3 2 Activity Diagram

Activity diagram yang dirancang dalam sistem rekomendasi ini meliputi serangkaian *Use Case* yang terdapat pada Gambar 3.11, dan diuraikan penjelasannya bagaimana sistem bekerja dari awal hingga akhir melalui Gambar 3.5 hingga Gambar 3.11

a. *Login*

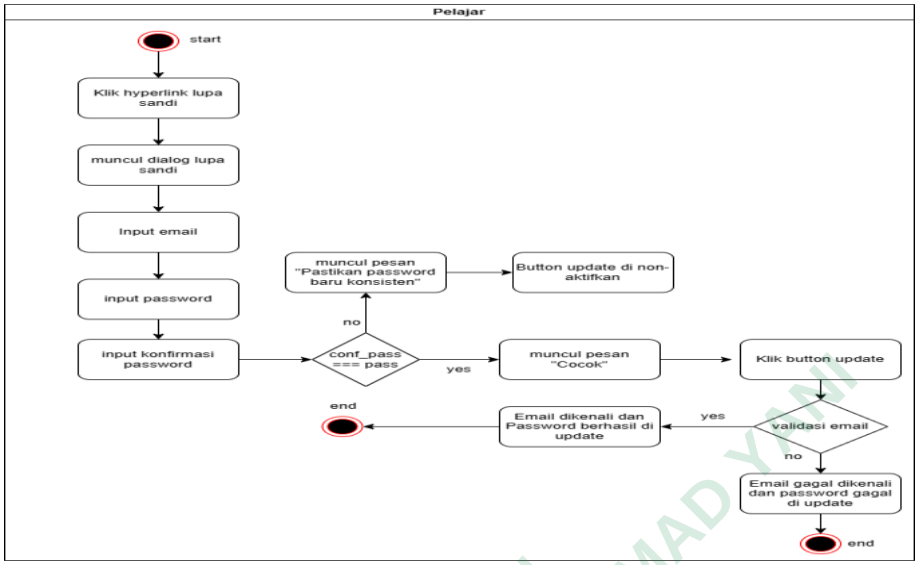
Fitur *login* dibutuhkan untuk pengguna memasukkan *email* dan *password.* Hal ini bertujuan untuk proses otentikasi dan otorisasi yang memberikan akses kepada pengguna dalam validasi bahwa pengguna terdaftar di dalam basis data sistem dan pengguna memperoleh token setelah proses otentikasi berhasil dilakukan. Apabila pengguna tidak dikenali oleh sistem, maka sistem akan memberikan pesan bahwa pengguna tidak dikenali. *Activity diagram* dari *login* dapat dilihat dari Gambar 3.5



Gambar 3.5 *Activity Diagram Login*

b. Lupa kata sandi

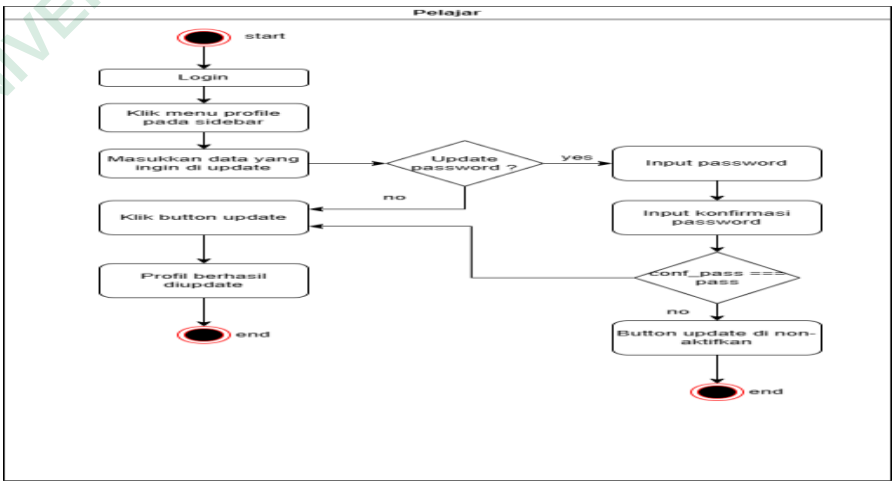
Fitur ini tanpa melalui proses otentikasi dan otorisasi karena pengguna melupakan kata sandinya tetapi *email* dapat dikenali oleh sistem. Letaknya dibawah form dari *email* dan *password.* Ketika lupa kata sandi ditekan oleh pengguna, dialog akan muncul dan sistem meminta masukkan seperti *email, password* dan konfirmasi *password. Activity diagram* dari lupa kata sandi dapat dilihat dari Gambar 3.6



Gambar 3.6 *Activity Diagram* Lupa Kata Sandi

c. Ubah profil pengguna

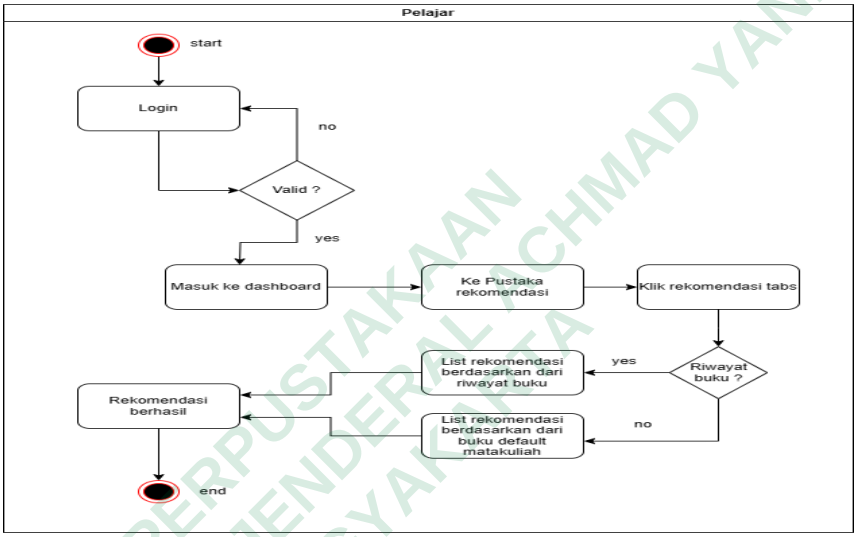
Fitur ini perlu melalui proses otentikasi dan otorisasi. Setelah pengguna *login* maka pengguna dapat dikenali dan memperoleh token untuk mengakses data yang hanya dapat dilihat oleh pengguna tersebut. Pengguna dapat mengubah profil dari fitur *Profile* yang terdapat pada menu sebelah kanan pojok (*sidebar*). *Activity diagram* dari ubah profil pengguna dapat dilihat dari Gambar 3.7



Gambar 3.7 *Activity Diagram* Ubah Profil

d. Inisiasi rekomendasi

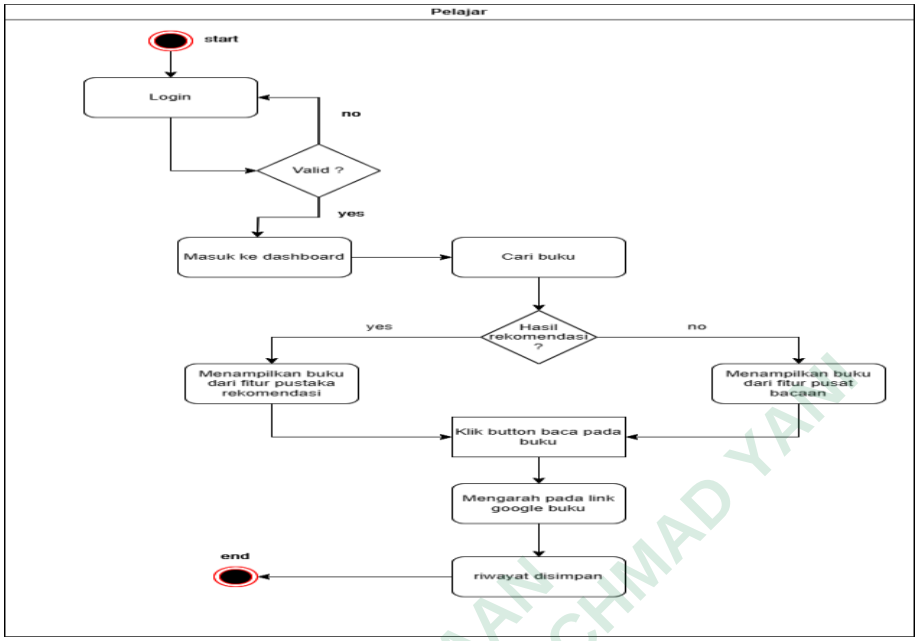
Fitur ini perlu melalui proses otentikasi dan otorisasi. Inisiasi rekomendasi dilakukan berdasarkan buku dari matakuliah (buku *default*) dan riwayat buku (jika ada). Untuk rekomendasi buku dapat dilihat pada fitur Pustaka Rekomendasi tepatnya *tabs* rekomendasi. *Activity diagram* dari inisiasi rekomendasi dapat dilihat dari Gambar 3.8



Gambar 3.8 *Activity Diagram* Inisiasi Rekomendasi

e. Menambah riwayat bacaan

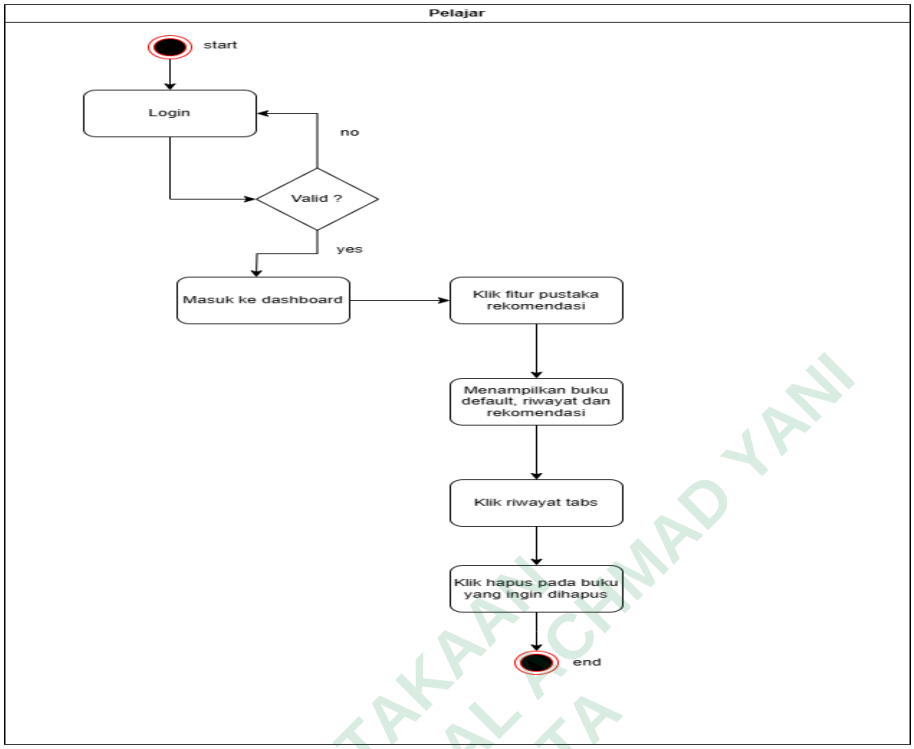
Fitur ini perlu melalui proses otentikasi dan otorisasi untuk melakukan penambahan data riwayat bacaan. Penambahan ini dilakukan ketika pengguna membaca buku pada fitur Pusat bacaan, dan Pustaka rekomendasi. *Activity diagram* dari menambah riwayat bacaan dapat dilihat dari Gambar 3.9



Gambar 3.9 *Activity Diagram* Menambah riwayat

f. Menghapus riwayat bacaan

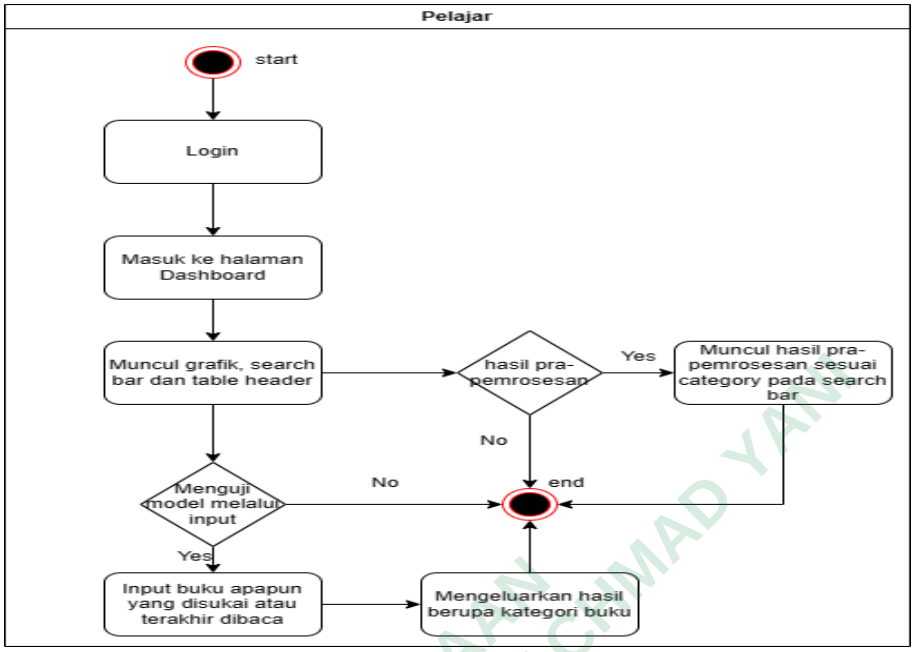
Fitur ini perlu melalui proses otentikasi dan otorisasi untuk menghapus riwayat bacaan. Riwayat tersebut dapat dilihat ketika pengguna berada di fitur Pustaka rekomendasi *tabs* buku riwayat serta pengguna dapat melakukan penghapusan riwayat dengan menekan tombol hapus pada *list* buku riwayat bacaan. *Activity diagram* dari menghapus riwayat bacaan dapat dilihat dari Gambar 3.10



Gambar 3.10 *Activity Diagram* Menghapus Riwayat

g. Mengunjungi *dashboard*

Fitur ini perlu melalui proses otentikasi dan otorisasi. Tujuan dari fitur ini adalah melihat Tabel berdasarkan label yang dipilih pengguna seperti judul buku dan hasil pra-pemrosesannya, grafik dari jumlah buku berdasarkan label, dan masukkan prediksi judul buku sesuai masukkan pengguna. *Activity diagram* dari mengunjungi *dashboard* dapat dilihat dari Gambar 3.11



Gambar 3.11 *Activity Diagram Dashboard*

3.3 3 Class Diagram

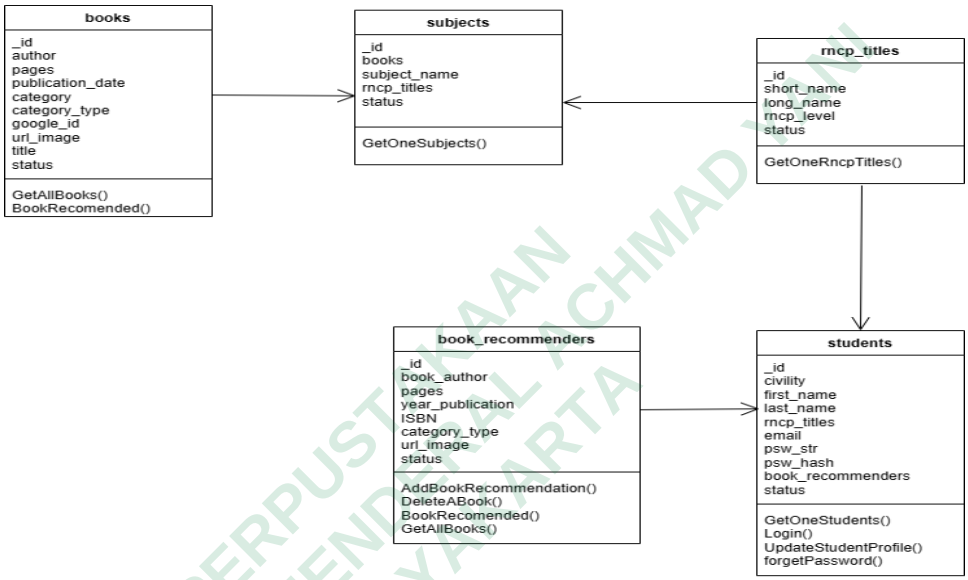
Pada Gambar 3.12 merupakan implementasi UML yaitu *Class Diagram* yang menjelaskan perancangan penyimpanan data pada sistem rekomendasi yang dibangun beserta hubungan dari setiap kelasnya dan operasi penggunaan data dari setiap kelas.

Sistem rekomendasi buku yang dikembangkan menggunakan basis penyimpanan data MongoDB. Setiap data yang disimpan tidak berelasi dan berbentuk dokumen. Maka dari itu *Class Diagram* mengGambarkan bagaimana setiap entitas memiliki referensi terhadap entitas lain, dan dari setiap entitas dimanfaatkan oleh fungsi yang memanggilnya seperti penjelasan berikut yang menguraikan koleksi dari MongoDB

a. *books* memiliki fungsi *GetAllBooks*(), *BookRecomended*()
b. *subjects* memiliki referensi dari entitas *books* dan *rncp_titles* dan memiliki fungsi *GetOneSubjects*()
c. *rncp_titles* memiliki fungsi *GetOneRncpTitles()*

d. *students* memiliki referensi dari entitas *book_recommenders* dan memilliki fungsi *GetOneStudents(), Login(), UpdateStudentProfile(), ForgetPassword()*

e. *book_recommenders* memiliki fungsi *AddBookRecommendation(), DeleteABook(), BookRecommended(), GetAllBooks().*



Gambar 3.12 *Class Diagram*