PENCARIAN DOKUMEN PADA TEKS HASIL TES EPPS MENGGUNAKAN METODE *INDEXING*



Disusun Oleh:

N a m a : Kahfiara Krisna S.P

NIM : 16523234

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

2020

HALAMAN PENGESAHAN DOSEN PEMBIMBING

PENCARIAN DOKUMEN PADA TEKS HASIL TES EPPS MENGGUNAKAN METODE *INDEXING*

TUGAS AKHIR

Disusun Oleh:

N a m a : Kahfiara Krisna S.P

NIM : 16523234

Yogyakarta, 21 Juli 2020

Pembimbing,

(Sri Mulyati, S.Kom., M.Kom)

HALAMAN PENGESAHAN DOSEN PENGUJI

PENCARIAN DOKUMEN PADA TEKS HASIL TES EPPS MENGGUNAKAN METODE *INDEXING*

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 21 Juli 2020

Tim Penguji

Sri Mulyati, S.Kom., M.Kom.

Anggota 1

Arrie Kurniawardhani, S.Si., M.Kom.

Anggota 2

Lizda Iswari, S.T., M.Sc.

Mengetahui,
Ketua Program Studi Informatika – Program Sarjana
Fakultas Teknologi Industri
Universitas Islam Indonesia

(Dr. Raden Teduh Dirgahayu, S.T., M.Sc.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Kahfiara Krisna S.P

NIM : 16523234

Tugas akhir dengan judul:

PENCARIAN DOKUMEN PADA TEKS HASIL TES EPPS MENGGUNAKAN METODE *INDEXING*

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila dikemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung resiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 21 Juli 2020



(Kahfiara Krisna S.P)

HALAMAN PERSEMBAHAN

Allhamdulillahirobbil'alamin, puja dan puji syukur kehadirat Allah *Subhanahu wata'ala* atas segala nikmat yang diberikan kepada kita semua. Shalawat serta salam kepada junjungan kita Nabi Muhammad Sallallahu 'alaihi wasallam yang telah membimbing kita menuju Islam yang rahmatan lil 'alamin, dan yang kita nantikan syafa'atnya kelak di hari akhir. *Allhamdulillahirobbil'alamin,* atas rahmat Allah dan ridho-Nya peneliti berhasil menyelesaikan tugas sekaligus amanah terakhir dalam bangku perkuliahan ini. Karya ini saya persembahkan untuk:

1. Orang tuaku yang saya hormati dan cintai, Bapak Setya Graha dan Ibu Lita Yuswandani. Terima kasih atas segala bentuk dukungan, motivasi serta doa yang diberikan, karena tanpa restu dari Orang Tua tugas akhir ini tidak akan dapat saya selesaikan. Tugas akhir akhir ini saya persembahkan untuk Bapak dan Ibu sebagai wujud cinta dan terima kasih saya kepada kalian.
2. Adik-adik yang aku sayangi, Afiq Bima Alfajri dan Alfayza Yassinta Ramadhani sebagai penghilang lelah dan sumber semangat dan bahagia saya selama ini.
3. Dosen Pembimbing saya, ibu Sri Mulyati, S.Kom., M.Kom. yang telah membimbing, mengajar dan memberikan saya nasihat serta motivasi dalam pembuatan laporan tugas akhir ini.
4. Keluarga dan teman-temanku yang selalu memberi inspirasi dan motivasi kepada saya dalam menyelesaikan tugas akhir ini.
5. Dan kamu yang sedang membaca tugas akhir ini. Terimakasih telah meluangkan waktumu, semoga kesuksesan selalu ada padamu. Aamiin.

HALAMAN MOTO

“Wahai orang-orang yang beriman! Mohonlah pertolongan (kepada Allah) dengan sabar dan sholat. Sungguh, Allah beserta orang-orang yang sabar”.

(Q.S Al-Baqarah: 153)

“Karena sesungguhnya sesudah kesulitan itu ada kemudahan, sesungguhnya sesudah kesulitan itu ada kemudahan. Maka apabila kamu telah selesai (dari sesuatu urusan), kerjakanlah dengan sungguh-sungguh (urusan) yang lain, dan hanya kepada Tuhanmulah hendaknya kamu berharap”.

(Q.S Al-Insyirah: 5-8)

“Allah tidak membebani seseorang melainkan sesuai dengan kesanggupannya. Ia mendapat pahala (dari kebajikan) yang diusahakannya dan ia mendapat siksa (dari kejahatan) yang dikerjakannya. (Mereka berdoa): "Ya Tuhan kami, janganlah Engkau hukum kami jika kami lupa atau kami tersalah. Ya Tuhan kami, janganlah Engkau bebankan kepada kami beban yang berat sebagaimana Engkau bebankan kepada orang-orang sebelum kami. Ya Tuhan kami, janganlah Engkau pikulkan kepada kami apa yang tak sanggup kami memikulnya. Beri maaflah kami; ampunilah kami; dan rahmatilah kami. Engkaulah Penolong kami, maka tolonglah kami terhadap kaum yang kafir”.

(Q.S Al-Baqarah: 286)

“Sekarang Allah telah meringankan kepadamu dan dia telah mengetahui bahwa padamu ada kelemahan. Maka jika ada diantaramu seratus orang yang sabar, niscaya mereka akan dapat mengalahkan dua ratus orang kafir; dan jika diantaramu ada seribu orang (yang sabar), niscaya mereka akan dapat mengalahkan dua ribu orang, dengan seizin Allah. Dan Allah beserta orang-orang yang sabar.

(Q.S Al-Anfal: 66)

KATA PENGANTAR

Assalamualaikum Warahmatullahi Wabarakatuh.

Allhamdulillahirobbil'alamin, penulis memanjatkan rasa syukur yang setinggi-tingginya kepada Allah Subhanahu wata'ala atas segala nikmat yang telah diberikan, sehingga penulis bisa menyelesaikan laporan tugas akhir ini. Shalawat serta Salam terhaturkan kepada junjungan Nabi Muhammad *Sallallahu 'alaihi wasallam* yang telah membawa cahaya Islam ke muka bumi, dan yang dinantikan syafa'atnya di Yaumul Qiyamah.

Laporan ini disusun untuk memenuhi tugas akhir sebagai syarat untuk menyelesaikan pendidikan pada jenjang Strata 1 (S1) pada Jurusan Informatika Universitas Islam Indonesia. Penulis sadar laporan ini tidak akan dapat selesai dengan waktu yang cepat tanpa dukungan serta motivasi dari beberapa pihak. Oleh karena itu, penulis hendak menyampaikan apresiasi dan terimakasih setinggi-tingginya kepada pihak-pihak yang disebutkan di bawah ini:

1. Kepada Allah SWT, atas limpahan rahmat dan hidayah-Nya yang selalu ada di setiap langkah dalam memberikan kekuatan, kemampuan, dan menjaga semangat saya untuk dapat menyelesaikan Tugas Akhir ini dengan lancar.
2. Kedua orang tua, Bapak Setya Graha dan Ibu Lita Yuswandani, atas dukungan, motivasi, nasihat serta doa yang tiada henti-hentinya sebagai kekuatan saya dalam menyelesaikan tugas akhir ini.
3. Bapak Fathul Wahid, S.T., M.Sc., Ph.D., sebagai Rektor Universitas Islam Indonesia dan seluruh jajaran Rektorat Universitas Islam Indonesia.
4. Bapak Prof. Dr. Ir. Hari Purnomo, M.T., sebagai Dekan Fakultas Teknologi Industri Universitas Islam Indonesia.
5. Bapak Hendrik, S.T., M.Eng., sebagai Ketua Jurusan Informatika Universitas Islam Indonesia.
6. Bapak Dr. Raden Teduh Dirgahayu, S.T., M.Sc., sebagai Ketua Program Studi S1 Informatika.
7. Ibu Sri Mulyati, S.Kom., M.Kom., sebagai Dosen Pembimbing yang telah banyak memberikan pengarahan, bimbingan, serta masukan dalam menyelesaikan tugas akhir.
8. Dosen-dosen Jurusan Informatika beserta staff yang telah memberikan ilmu pengetahuan serta bantuannya.
9. Teman-teman Informatika, seluruh angkatan terutama Hexadecima atas dukungan serta pengalaman yang diberikan bersama sampai detik ini.

10. Teman-teman KKN unit 288, atas pengalaman serta dukungan yang diberikan sampai detik ini.
11. Teman-teman pabji learning yang aku cintai, Dwitiya, Citra, Fionna, Febri yang sudah menjadi sumber keceriaan pada hari-hari saya selama mengerjakan tugas akhir ini.
12. Keluarga Marching Band beserta teman-teman saya, Intan, Edgar, Asti, Mila, Iven, Vita,Sera yang tidak henti-henti memberikan dukungan, motivasi dan menjadi tempat berkeluh kesah saya sampai saat ini dalam menyelesaikan tugas akhir ini.
13. Semua pihak yang tidak bisa saya sebutkan satu per satu, terimakasih atas semua doa serta dukungan yang diberikan.

Semoga segala bentuk bantuan, dukungan, bimbingan, dan pengalaman yang telah diberikan kepada peneliti sampai saat ini mendapatkan kebaikan dari Allah SWT. Peneliti memohon maaf apabila selama melaksanakan Tugas Akhir terdapat kekhilafan dan kesalahan. Peniliti menyadari akan keterbatasan kemampuan yang dimiliki. Semoga laporan ini dapat bermanfaat bagi semua yang membaca dan menikmatinya.

Yogyakarta, 21 Juli 2020



(Kahfiara Krisna Selia Putri)

SARI

Tes EPPS (*Edward Personal Preference Schedule*) merupakan salah satu bentuk tes psikologi yang dilakukan baik secara personal maupun organisasi. Penggunaan tes EPPS digunakan dalam bidang pendidikan sebagai tahap penyeleksian pada universitas yang menyelenggarakan tes EPPS sebagai kebutuhan untuk penerimaan calon mahasiswanya. Menemukan kembali informasi yang dibutuhkan pada hasil tes EPPS sangatlah penting pada proses penyeleksian calon pendaftar. Data hasil tes EPPS yang dihasilkan tak sedikit jumlahnya mengingat universitas membuka penerimaan calon mahasiswa setiap tahun ajaran baru dan beberapa fakultas menerapkan tes EPPS sebagai tahapan untuk proses penerimaan calon mahasiswa selain tes penerimaan pada umumnya. Pada saat tertentu kendala yang kerap terjadi ketika ingin mencari letak dokumen apa saja yang memiliki informasi yang mengandung kata dalam aspek kebutuhan dan penyajian informasi yang mudah digunakan oleh pihak penyelenggara. Solusi atas permasalahn tersebut adalah menerapkan metode proses *indexing* pada *Information Retrieval* dimana mengambil indeks yang telah ditentukan untuk menemukan kembali informasi yang mengandung aspek kebutuhan pada dokumen tertentu dan mencari letak dokumen lain dengan indeks kata tersebut.

Dari hasil tes berbentuk pdf akan dikonversi kedalam bentuk csv, kemudian melalui tahap pengindeksan untuk mendapatkan informasi terkait dengan karakteristik calon pendaftar berdasarkan aspek kebutuhan pada EPPS. Dataset yang digunakan sebanyak 74 data berbentuk pdf. Secara keseluruhan, terdapat 60 dataset yang dijadikan acuan untuk mengambil frasa pada data yang ditentukan dan data uji sebanyak 14 data. Tipe data frasa terbagi menjadi dua bagian yaitu *all* dan *all reducted*. Hasil pengujian terbesar yang diperoleh oleh *all* dengan rata-rata pada *precision* sebesar 0,94, *recall* sebesar 0,97 dan *f-measure* sebesar 0,96.

Kata kunci: Aspek Kebutuhan, Frasa, *Indexing*, Tes EPPS, *Text Mining*, Tes Psikologi.

GLOSARIUM

Python	Bahasa pemrograman
Visual Code	Tools untuk melakukan pemrograman
Text Mining	Proses ekstraksi pola berupa informasi atau pengetahuan dari data
EPPS	Salah satu tes psikologi
Pre-processing	Proses membersihkan data menjadi data terstruktur
Dataset	Kumpulan data dalam pembentukan model
Data *training*	Data pelatihan yang dijadikan model
Data *test*	Data uji yang dijadikan untuk menghitung evaluasi
Indexing	Metode dalam menemukan informasi yang terkait pada dokumen

DAFTAR ISI

HALAMAN JUDUL 1
HALAMAN PENGESAHAN DOSEN PEMBIMBING 2
HALAMAN PENGESAHAN DOSEN PENGUJI 3
HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR 4
HALAMAN PERSEMBAHAN 5
HALAMAN MOTO 6
KATA PENGANTAR 7
SARI 9
GLOSARIUM 10
DAFTAR ISI 11
DAFTAR TABEL 13
DAFTAR GAMBAR 14
BAB I PENDAHULUAN 16
1.1 Latar belakang 16
1.2 Rumusan Masalah 17
1.3 Batasan Masalah 17
1.4 Tujuan Penelitian 17
1.5 Manfaat Penelitian 18
1.6 Metode Penelitian 18
1.7 Sistematika Penulisan 19
BAB II LANDASAN TEORI 21
2.1 Dasar Teori 21
2.1.1 Tes Psikologi 21
2.1.2 Edward Personal Preference Schedule 22
2.1.3 Text Mining 23
2.1.4 *Indexing* dan *Retrieval* 24
2.1.5 Penentuan Kesamaan Dokumen 25
2.1.6 *Performance Evaluation Measure* 25
2.2 Tinjauan Pustaka 27
BAB III METODOLOGI PENELITIAN 29
3.1 Studi Pustaka 29
3.2 Pengumpulan Data 30
3.3 Analisis Data 31
3.4 Perancangan Sistem 33
3.4.1 Konversi Dokumen 34
3.4.1.1 Preprocessing 35
3.4.2 *Split* Data 37
3.4.3 Pelabelan Manual 37
3.4.4 Pembentukan Daftar Frasa 37
3.4.5 Identifikasi Informasi 39
3.4.6 Pencarian Dokumen 41
3.4.7 Uji dan Evaluasi Model 43
BAB IV HASIL DAN PEMBAHASAN 44
4.1 Implementasi Program 44
4.1.1 Konversi Dokumen 44
4.1.1.1 *Preprocessing* 46
4.1.2 *Split* Data 49

4.1.3 Pelabelan Manual 50
4.1.4 Pembentukan Daftar Frasa 51
4.1.5 Identifikasi Informasi 53
4.1.6 Pencarian Dokumen 56
4.1.7 Hasil Pengujian dan Evaluasi 58
BAB V KESIMPULAN DAN SARAN 62
5.1 Kesimpulan 62
5.2 Saran 62
DAFTAR PUSTAKA 63
LAMPIRAN 66

DAFTAR TABEL

Tabel 2.1 Tabel Confusion Matrix 27
Tabel 3.1 Aspek Kebutuhan beserta Label 32
Tabel 3.2 Contoh Hasil *Casefolding* 36
Tabel 3.3 Contoh Hasil *Tokenization* 36
Tabel 4.1 Hasil pengujian pada data uji 59
Tabel 4.2 Hasil pengujian model tipe data frasa 60
Tabel 4.3 Hasil rata-rata evaluasi 61

DAFTAR GAMBAR

Gambar 3.1 Diagram Alur Metodologi Penelitian 29
Gambar 3.2 Hasil Psikotes EPPS 31
Gambar 3.3 Hasil Psikotes EPPS 33
Gambar 3.4 Diagram Alur Perancangan Sistem 33
Gambar 3.5 Alur proses konversi dokumen 34
Gambar 3.6 Contoh Teks Hasil tes EPPS 35
Gambar 3.7 Alur proses pembentukan daftar frasa 38
Gambar 3.8 Alur proses identifikasi informasi 40
Gambar 3.9 Alur proses pencarian dokumen 42
Gambar 4.1 Kode program *pdf to txt* 44
Gambar 4.2 Kode program untuk mengambil indeks uraian 45
Gambar 4.3 Contoh kalimat pada uraian 45
Gambar 4.4 Kode program untuk konversi dokumen 46
Gambar 4.5 Hasil konversi dokumen 46
Gambar 4.6 Kode program *case folding* 47
Gambar 4.7 Hasil *casefolding* 47
Gambar 4.8 Kode program tokenisasi 48
Gambar 4.9 Hasil tokenisasi 48
Gambar 4.10 Hasil keluaran konversi dokumen 49
Gambar 4.11 Kode program *split* data 49
Gambar 4.12 Hasil *split* data *train* 50
Gambar 4.13 Hasil *split* data *test* 50
Gambar 4.14 Hasil pelabelan manual dokumen 51
Gambar 4.15 Kode program mengambil frasa dari data 52
Gambar 4.16 Kode program pengurutan dan simpan daftar frasa 52
Gambar 4.17 Hasil pembentukan daftar frasa 53
Gambar 4.18 Kode program pemisah paragraf 53

Gambar 4.19 Kode program identifikasi informasi 54
Gambar 4.20 Contoh hasil proses identifikasi informasi 56
Gambar 4.21 Kode program pencarian dokumen 57
Gambar 4.22 Hasil implementasi file berdasarkan aspek kebutuhan “ACH” 57
Gambar 4.23 Hasil implementasi file berdasarkan aspek kebutuhan “NUR” 58
Gambar 4.24 Kode program uji dan evaluasi model 60

BAB I
PENDAHULUAN

1.1 Latar belakang

Tes EPPS (*Edward Personal Preference Schedule*) merupakan salah satu bentuk tes psikologi yang dilakukan baik secara personal maupun organisasi. Penggunaan tes EPPS digunakan dalam bidang pendidikan sebagai tahap penyeleksian pada universitas yang menyelenggarakan tes EPPS sebagai kebutuhan untuk penerimaan calon mahasiswa baru. File tes EPPS akan disimpan dalam bentuk pdf yang berisikan matriks psikogram beserta kesimpulan.

Setiap tahun ajaran baru, banyak calon pendaftar yang mendaftar pada universitas tersebut terlebih lagi dengan pilihan jurusan yang diinginkan. Tentunya jumlah data hasil tes EPPS yang dihasilkan akan sangat besar dan tersimpan setiap periodenya. Dengan banyaknya data calon pendaftar yang baru serta informasi baru tentu menyulitkan pihak penyelenggara untuk memberikan prediksi yang tepat pada pemilihan calon mahasiswanya karena informasi yang disajikan masih kurang efektif dan menyeluruh. Terlebih lagi pihak penyelenggara yang harus menyaring kembali sejumlah dokumen dan mencari hasil dengan kesesuaian penilaian yang ditentukan. Tidak adanya data rekam yang dapat menunjukkan informasi baru dengan data yang ada sebelumnya dan penyajian yang lebih efisien agar mudah dipahami baik pakar maupun pihak penyelenggara masih menjadi kendala saat ini. Pada saat tertentu kendala yang kerap terjadi ketika ingin mencari letak dokumen apa saja yang memiliki informasi yang mengandung kata dalam aspek kebutuhan. Untuk menemukan informasi tertentu dari sekumpulan dokumen biasanya dilakukan dengan mencari kesamaan kata yang terkandung pada suatu dokumen. Jika proses ini dilakukan secara manual akan memakan waktu yang cukup lama dalam menemukan dokumen tersebut terlebih lagi dokumen dalam jumlah yang besar.

Penemuan kembali dengan indeks berperan dalam menemukan indeks yang mengandung aspek kebutuhan yang diinginkan dalam hasil tes EPPS dan menemukan indeks kata yang di cari pada dokumen tersebut. Penentuan kesamaan dokumen dapat dilakukan melalui metode pengindeksan kata dari sekumpulan dokumen. Proses *indexing* bertujuan untuk mengubah dokumen yang tidak terstruktur menjadi dokumen yang terstruktur yang dalam prosesnya jika

menemukan indeks kata yang dicari maka dapat menampilkan dokumen tersebut (Hidayat, 2013).

Berdasarkan pemaparan di atas, dalam penelitian ini diharapkan dapat memudahkan para pihak penyelenggara untuk melihat banyak data dan menyaring banyaknya calon pendaftar dengan menemukan informasi penting yang diinginkan dari kesimpulan psikotes EPPS berdasarkan aspek kebutuhan yang telah ditentukan serta mampu menunjukkan posisi dokumen yang tersimpan. Sasaran dari penelitian ini yaitu anak sekolah menengah atas jurusan IPA yang akan mendaftar sebagai calon mahasiswa Fakultas Kedokteran UII. Pada kesempatan ini, penulis mencoba melakukan proses pencarian informasi dan kesamaan dokumen menggunakan metode *indexing* untuk menemukan kecocokan informasi mengenai aspek kebutuhan yang terdapat pada teks hasil kesimpulan tes EPPS serta mencari kesamaan informasi data rekam dengan informasi data baru dalam mendapatkan hasil kebutuhan tertinggi berdasarkan aspek kebutuhan yang dicari pihak penyelenggara.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah disebutkan, maka diperoleh rumusan masalah yang disusun adalah bagaimana mencari serta menyajikan informasi yang ada dalam hasil matriks psikogram tes EPPS untuk memudahkan pencarian kebutuhan yang diinginkan?

1.3 Batasan Masalah

Untuk menjaga fokus pada penelitian dalam TA ini, berikut beberapa batasan yang diperlukan:

1. Data yang digunakan adalah hasil tes EPPS calon pendaftar tahun 2018.
2. Tes kepribadian EPPS yang diimplementasikan melibatkan kalangan siswa-siswi SMA jurusan IPA yang mendaftar ke Fakultas Kedokteran UII.
3. Data keluaran berupa informasi meliputi token dengan label kelas sebanyak 15 aspek kebutuhan pada tes EPPS.
4. Jumlah *dataset* yang digunakan adalah 74 dokumen yang dipisah menjadi data *training* sejumlah 60 *dataset* dan data uji sejumlah 14 *dataset*.

1.4 Tujuan Penelitian

Berdasarkan identifikasi masalah, maka dapat disusun tujuan dari penerlitian ini adalah:

1. Memahami informasi secara terstruktur mengenai kebutuhan individu pada data psikotes EPPS berdasarkan 15 aspek kebutuhan agar memudahkan bagi pihak penyelenggara untuk mengetahui informasi yang dibutuhkan sebagai komposisi penilaian calon pendaftar.
2. Mengukur performa evaluasi sistem berdasarkan pola frasa yang terbentuk dalam proses pengindeksan pada tes EPPS dengan baik.

1.5 Manfaat Penelitian

Manfaat yang dihasilkan pada penelitian ini adalah membantu penanggungjawab yang ingin mengekstraksi informasi untuk mengenali kalimat kesimpulan pada psikotes EPPS yang termasuk kedalam 15 aspek kebutuhan yang telah ditentukan.

1.6 Metode Penelitian

Untuk menyelesaikan rumusan masalah pada penelitian ini, diperlukan beberapa tahapan yang dilakukan. Penelitian ini akan dilakukan dalam beberapa tahapan berikut:

A. Pengumpulan data

Dalam tahapan ini dilakukan dengan melakukan studi literatur, dengan mengumpulkan data melalui buku-buku, jurnal, situs internet, artikel terkait sebagai referensi terkait dengan permasalahan yang terjadi pada penelitian. Data yang digunakan dalam penelitian ini berupa data psikometri calon pendaftar Fakultas Kedokteran dari PMB tahun 2018. Data tersebut akan menjadi data latih maupun data uji untuk memeriksa sistem akan berjalan sesuai dengan ketentuan.

B. Analisis Kebutuhan

Dalam tahapan ini dilakukan analisis terhadap kebutuhan-kebutuhan yang akan digunakan untuk menemukan solusi atas permasalahan dari penelitaan terkait. Kebutuhan tersebut berupa data-data hasil tes EPPS. *Tools* pendukung untuk mengekstraksi informasi yaitu IDE *Visual Code* dan beberapa *Python package library* yang digunakan untuk membangun model.

C. Perancangan sistem

Dalam tahapan ini, perancangan sistem dilakukan untuk memberikan gambaran proses pada sistem ekstraksi informasi yang akan dibangun beserta dengan model apa yang digunakan dalam proses ekstraksi informasi agar lebih terstruktur dan mudah dipahami orang lain.

D. Implementasi sistem

Dalam tahapan ini, data yang diterima akan dilakukan proses implementasi ke dalam bahasa pemrograman untuk menemukan solusi atas permasalahan. Langkah yang akan dilakukan meliputi masukan data psikotes, membaca data, melakukan proses *preprocessing*, proses *training* dan *testing* sampai menghasilkan ekstraksi informasi beserta pengujian.

E. Pengujian dan evaluasi

Tahapan ini digunakan untuk mengintegrasikan program dan pengujian tujuan, apabila program yang dibuat berjalan sesuai dengan harapan. Pengujian ini dilakukan dengan menghitung *precision*, *recall*, dan *f-measure* dalam bentuk *Confusion Matrix* untuk mengetahui tingkat signifikasi atas penggunaan sistem.

F. Pembuatan Laporan

Dalam tahapan akhir ini, menyajikan hasil keseluruhan dari hasil penelitian serta penarikan kesimpulan terkait pengujian yang telah dilakukan. Diharapkan pada tahapan ini dapat berisikan masukan untuk pengembangan selanjutnya.

1.7 Sistematika Penulisan

Untuk mempermudah dalam penelitian maupun penyusunan laporan penelitian, maka dibuatlah sistematika dan runtutan proses penulisan penelitian sebagai berikut:

BAB I Pendahuluan

Pada bab ini membahas gambaran umum dari Tugas Akhir seperti latar belakang penelitian, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metode penelitian dan sistematika penulisan.

BAB II Landasan Teori

Pada bab ini membahas ulasan penelitian sebelumnya dengan melakukan perbandingan dengan penelitian yang dilakukan peneliti. Bab ini juga berisikan landasan-landasan teori yang terkait pada penelitian ini.

BAB III Metodologi Penelitian

Pada bab ini membahas tentang langkah-langkah dan analisis pada penelitian yang dilakukan.

BAB IV Hasil dan Pembahasan

Pada bab ini membahas tentang hasil dari implementasi serta pengujian kinerja sistem. Pada bagian implementasi membahas tentang *code programming* dan *printscreen* dari sistem yang akan dibangun. Sedangkan bagian pengujian membahas tentang pengujian validitas apakah sistem ekstraksi informasi dengan metode yang digunakan sudah tepat.

BAB V Kesimpulan dan Saran

Pada bab ini membahas tentang penjelasan hasil keseluruhan dari penelitian yang dilakukan dan saran yang diberikan dalam bentuk poin-poin yang dapat meningkatkan pengembangan sistem untuk peneliti selanjutnya.

BAB II
LANDASAN TEORI

2.1 Dasar Teori

2.1.1 Tes Psikologi

Tes merupakan suatu eksperimen yang tertulis berupa pertanyaan atau tugas untuk memperoleh informasi mengenai prestasi akademik atau kondisi psikologis dengan tujuan yang telah ditentukan (Kuncoro, 2012) . Psikologi dalam bahasa inggris "*psychology*" memiliki arti yaitu ilmu jiwa. Psikologi merupakan ilmu yang mempelajari tentang ekspresi pada kejiwaan atau mental individu untuk mendapatkan proses tingkah laku pada manusia (Waluya, 1970). Psikologi digunakan untuk mempelajari ekspresi tingkah laku manusia dari berbagai sudut pandang baik secara pendidikan, sosial, klinis, politik, kepribadian, perkembangan, rekayasa, industri dan lingkungan. Dengan kata lain, tes psikologi merupakan suatu alat penilaian yang tertulis yang digunakan oleh psikolog untuk memperoleh informasi mengenai tingkah laku individu dengan tujuan yang tertentu. Tes psikologi diinterpretasikan sebagai hasil tes yang terdiri dari parameter penilaian yang telah terstandar sebagai acuan dalam menetapkan tujuan yang cocok pada bidangnya. Dalam penggunaannya, tes psikologi dapat digunakan sebagai alat untuk memprediksi individu, diagnosis terhadap kondisi seseorang, mengamati individu, dan mengevaluasi berdasarkan hasil yang ditimbulkan.

Di Indonesia, terdapat berbagai macam tes psikologi yang telah dilakukan pada konteks pendidikan maupun pekerjaan antara lain tes grafis, DISC, tes papikostik, tes SSCT, EPPS, tes Rorschach, Kraplien test, dan tes Army Alpha Intelligent. Setiap tes memiliki parameter penilaian berbeda-beda sesuai dengan kebutuhan nya yang berupa pertanyaan pada lingkungan sekitar, menghitung kombinasi angka atau memakai kartu bergambar. Fungsi dari tes psikologi dapat digunakan sebagai berikut (Nurussakinah, 2014):

1. Seleksi: bertujuan untuk memutuskan individu yang terpilih pada suatu lembaga seperti pendidikan maupun pekerjaan
2. Klasifikasi: mengelompokkan individu berdasarkan kelompok yang sejenis dengan psikologisnya
3. Deskripsi: hasil tes psikologi telah dilakukan tanpa klarifikasi tertentu

4. Mengevaluasi: bertujuan untuk mengetahui tindakan tertentu terhadap seseorang atau sekolompok orang
5. Menguji suatu hipotesis: bertujuan untuk mengetahui kebenaran dibalik suatu hipotesis

2.1.2 Edward Personal Preference Schedule

Tes Edward Personal Preference Schedule atau yang disebut EPPS merupakan salah satu bentuk tes kepribadian yang terdiri dari beberapa pilihan jawaban yang akan dipilih oleh individu dari pertanyaan yang telah tersedia untuk mengetahui kebutuhan pribadi yang mencerminkan kepribadian individu (Haggard et al., 1983). Pilihan jawaban tersebut akan menggambarkan kondisi internal individu. Dalam tes ini diujikan untuk mengetes kepimpinan seseorang berdasarkan cara bersosialisasi ataupun memecahkan masalah. Selain menjawab tes tertulis, tes EPPS juga dilakukan secara wawancara. Dalam wawancara, subyek diminta untuk memilih salah satu jawaban dari beberapa pertanyaan yang dianggap paling cocok dengan dirinya.

Tes ini dikembangkan oleh Allen L Edward dengan memilih lima belas *needs* dari dua puluh *needs* yang telah disusun yang merujuk kepada konsep kepribadian Murray. Pada tes kepribadian ini terdapat 225 pertanyaan yang bersifat memaksa, objektif dan non-proyektif (Piedmont & Costa, 1992). Tujuan penggunaan tes EPPS ini adalah untuk mengamati kebutuhan-kebutuhan khusus yang dimiliki individu.

Murray (2017) menjelaskan bahwa kepribadian seorang individu tergambarkan melalui kebutuhan tertentu yang akan memotivasi perilaku tersebut muncul. Ketika kebutuhan telah terpenuhi maka individu akan memunculkan perilaku-perilaku yang menggambarkan karakteristik tersebut. Aspek kebutuhan yang diukur untuk memenuhi dalam standar tes EPPS sebagai berikut (Ferrara, 1996):

1. *Achievement*: Kebutuhan untuk melakukan usaha terbaik untuk menyelesaikan tugas-tugas yang sulit.
2. *Deference*: Kebutuhan untuk menyesuaikan diri dan mengikuti aturan.
3. *Order*: Kebutuhan untuk tertib dan bertanggungjawab dengan perencanaan sebelumnya.
4. *Exhibition*: Kebutuhan untuk mengekspresikan diri, optimis, dan percaya diri.
5. *Autonomy*: Kebutuhan untuk bersikap mandiri dan tidak tergantung pada orang lain dalam melakukan keputusan.
6. *Affilliation*: Kebutuhan untuk berbaik hati dan dapat bekerja sama dalam melakukan sesuatu dengan orang lain.

7. *Intraception*: Kebutuhan dalam menganalisa dan memahami motif-motif dan perasaan orang lain.
8. *Succorance*: Kebutuhan dalam menerima bantuan dari orang lain agar orang lain bersimpati pada dirinya
9. *Dominance*: Kebutuhan untuk mendominasi, memerintah, dan mempengaruhi orang lain.
10. *Abasemenet*: Kebutuhan untuk mengakui kesalahan diri.
11. *Nurturance*: Kebutuhan untuk menolong, merawat orang atau benda.
12. *Change*: Kebutuhan untuk melakukan perubahan.
13. *Endurance*: Kebutuhan untuk menyelesaikan sesuatu dengan tekun dan ulet dan tidak ingin diganggu selama bertugas.
14. *Heterosexual*: Kebutuhan terhadap kehidupan seksual.
15. *Aggression*: Kebutuhan untuk melawan pendapat orang lain secara agresif.

2.1.3 Text Mining

Text mining merupakan proses dari analisis teks yang dilakukan secara otomatis oleh komputer untuk penggalian data berupa teks yang tercatat dalam dokumen dan mencari kata-kata yang mewakili isi dari dokumen untuk menganalisa kaitan antar dokumen menggunakan metode statistik tertentu (Herdianto, 2010). Terdapat berbagai penyebutan untuk *text mining* seperti penambangan data teks (*text data mining)* atau *knowledge discovery* karena database tekstual yang merujuk pada penggalian atau pengambilan informasi penting dari teks yang tidak terstruktur. Struktur umum dari *text mining* melibatkan dua tahapan yaitu penggunaan struktur terhadap data teks sebagai masukan yang selanjutnya dilakukan ekstraksi informasi yang relevan dari data terstruktur dengan menggunakan teknik yang sama dengan *text mining. Text mining* adalah bidang multidisplin yang menyangkut pengambilan informasi, analisis teks, ekstraksi informasi, kategorisasi, pengelompokkan visualisasi, penambangan data, dan pembelajaran mesin (Dang & Ahmad, 2014). Tujuan dari *text mining* adalah untuk mendapatkan informasi yang relevan dari sekumpulan dokumen sehingga dapat dilakukan analisa keterkaitan antar dokumen. Sumber data yang digunakan pada *text mining* berupa sekumpulan teks yang memiliki format yang tidak terstruktur atau semi terstruktur. *Text mining* biasanya digunakan untuk mendapatkan hasil yang cepat dan dapat dilakukan pada sejumlah bidang aplikasi. *Text mining* adalah bidang penting yang menggunakan metode dan pendekatan kolektif dari berbagai bidang sebagai berikut (Eroglu et al., 2018):

1. *Information Retrieval (IR)*

Proses untuk memperoleh informasi yang telah tersimpan dari berbagai sumber yang relevan atau dibutuhkan.

2. *Information Extraction*

   Proses untuk mengekstrak informasi yang berguna dari data teks. IE digunakan untuk mengidentifikasi entitas, *events*, dan hubungan dari data semi-terstruktur dan tidak terstruktur.

3. *Categorization*

   Proses untuk melakukan pengelompokkan dokumen bahasa normal ke dalam topik yang telah ditentukan sesuai dengan kontennya.

4. *Clustering*

   Proses untuk menemukan struktur yang melekat dalam informasi dan mengaturnya menjadi sub-kelompok terkait untuk analisis lebih lanjut.

5. *Summarization*

   Proses untuk mengurangi panjang kalimat dengan mempertahankan poin utamanya dan makna keseluruhan seperti awalnya.

2.1.4 *Indexing* dan *Retrieval*

Proses dalam menemukan informasi dari suatu database dokumen dengan teks yang tidak testruktur yang berukuran besar sebagai jawaban atas kueri yang dibutuhkan (Heriyanto, 2015). *Indexing* dan *retrieval* termasuk ke dalam bagian *information retrieval*. Proses *indexing* dilakukan untuk menampilkan informasi yang diinginkan pada database dengan mengubah bentuk dokumen menjadi dokumen terstruktur sedangkan *retrieval* memudahkan pencarian dokumen berdasarkan informasi yang terkandung pada dokumen yang dicari. Tujuan utama pengindeksan dokumen adalah mengaitkan dokumen dengan informasi yang diwakili oleh fitur yang telah ditetapkan secara manual atau otomatis yang berasal dari kontennya serta memprediksi dokumen yang relevan dengan kueri yang diberikan (Broder et al., 2006).

Langkah-langkah pada proses *indexing* yaitu (Manning et al., 2004):

1. Segmentasi : Men-segmentasikan teks menjadi kata-kata.
2. *Stopword Removal* : Proses dalam menghilangkan kata yang diabaikan dalam pemrosesan dan kata tersebut tersimpan ke dalam file *stop list* pada library NLP.
3. Normalisasi : Proses dalam melakukan dua proses yaitu *stopwords* dan *stemming* (ekstraksi kata dasar).

Pada proses *indexing* dokumen terdapat beberapa tahapan yang dapat dilakukan dengan *inverted index* sebagai berikut (Hidayat, 2013):

1. *Parsing*

   Kata-kata yang ada pada dokumen akan dipecah menjadi satu kata terpisah yang akan disimpan dalam tabel bersama dengan keberadaan kata yang muncul pada dokumen tersebut.

2. *Sorting*

   Tabel yang diurutkan berdasarkan kata yang muncul.

3. *Merging*

   Jika dalam dokumen yang sama terdapat kata-kata yang sama munculnya maka digabungkan dalam satu file.

4. Hitung Frekuensi

   Kemunculan kata yang sama muncul akan dihitung frekuensi nya yang akan ditambahkan ke dalam tabel.

2.1.5 Penentuan Kesamaan Dokumen

Terdapat tiga macam metode yang digunakan untuk menentukan nilai kesamaan pada suatu dokumen antara lain (Yudhana et al., 2017):

1) *Distance-based similarity measure*

   Mengukur tingkat kesamaan pada dua dokumen yang memiliki variabel yang mencakup ke dalam kedua objek tersebut.

2) *Feature-based similarity measure*

   Mengukur tingkat kesamaan pada dokumen melalui fitur yang telah disediakan sebagai perbandingan. Biasanya, *feature-based* digunakan untuk mengklasifikasikan gambar atau teks.

3) *Probabilisatic-based similarity measure*

   Mengukur tingkat kesamaan dua dokumen menggunakan bentuk probabilitas sebagai alat untuk melakukan perbandingan.

2.1.6 *Performance Evaluation Measure*

Performance Evaluation Measure (PEM) atau pengukuran evaluasi performa. PEM dibagi menjadi dua makna yaitu pengukuran performa, proses berkelanjutan yang digunakan untuk memantau dan melaporkan kemajuan dan pencapaian program menggunakan performa yang telah dipilih sebelumnya. Sedangkan evaluasi program digunakan sebagai bentuk pengukuran dan analisis untuk menjawab pertanyaan spesifik tentang seberapa baik program mencapai hasil. PEM pada umumnya diperuntukkan dalam kasus training data untuk menganalisis model

yang sudah dibuat. Untuk mendapatkan hasil pada PEM dilakukan beberapa perhitungan seperti pada persamaan berikut (Sokolova & Lapalme, 2009):

1. *Precision*

   Alat ukur untuk melihat keakuratan pada model yang diprediksi positif dengan keseluruhan hasil yang diprediksi positif.

$$Precision = \frac{TP}{TP + FP} \quad (2.1)$$

2. *Recall*

   Recall berguna untuk menemukan kembali informasi yang telah tersimpan pada sistem

$$Recall = \frac{TP}{TP + FN} \quad (2.2)$$

3. *Accuracy*

   Akurasi berguna untuk memberikan informasi apakah suatu model yang sedang dilatih dilakukan secara benar dan bagaimana kinerja secara umum.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (2.3)$$

4. *F-Measure*

   Perbandingan rata-rata antara *precision* dan *recall* yang telah dibobotkan

$$F1 - Score = \frac{2xPrecisionxRecall}{Precision + Recall} \quad (2.4)$$

PEM dapat diinterpretasikan dalam bentuk tabel atau c*onfusion matrix*, tabel yang digunakan untuk menggambarkan kinerja model klasifikasi (atau "*classifier*") pada set data uji yang nilainya telah diketahui. Perhitungan menggunakan *confusion matrix* memberi informasi yang lebih baik tentang apa yang model klasifikasi sudah digunakan dengan benar dan jenis kesalahan apa yang dibuatnya. Jumlah prediksi yang benar dan salah dirangkum dengan nilai-nilai hitung yang dipecahkan oleh masing-masing kelas. Terdapat 4 istilah yang menjadi mewakili proses klasifikasi dapat dilihat pada Tabel 2.1. (Sokolova & Lapalme, 2009)

Tabel 2.1 Tabel Confusion Matrix

	Class 1: Predicted Positive	Class 2: Predicted Negative
Class 1 Actual	True Positive	False Negative
Actual 2 Actual	False Positive	True Negative

Keterangan:

True Positive (TP): Kata yang diekstrak ada dan secara manual ada

False Positive (FP): Kata yang diekstrak ada tapi secara manual tidak ada

False Negative (FN): Kata yang diekstrak tidak ada tapi secara manual ada

True Negative (TN): Kata yang diekstrak tidak ada dan tidak ada secara manual

2.2 Tinjauan Pustaka

Terdapat beberapa penelitian yang sudah pernah dilakukan oleh orang lain yang menjadi rujukan untuk dijadikan referensi dalam penelitian ini. Beberapa penelitian dapat dilihat sebagai berikut:

Penelitian pertama membahas tentang penggunaan metode *indexing* dalam proses menemukan dokumen yang dihasilkan dengan dokumen dengan hasil pencarian secara manual. *Inverted index* memiliki kelebihan pada penyimpanan data terstruktur sehingga mampu menyimpan berbagai dokumen dalam jumlah banyak. Proses yang dilakukan dengan membuat indeks kata dari kumpulan-kumpulan dokumen (Hidayat, 2013).

Penelitian kedua membahas pencarian dokumen teks arsip surat menggunakan metode *indexing* dan *query*. Penelitian ini membahas proses dalam menemukan kembali informasi dengan indeks agar dokumen dapat di cari kembali saat dibutuhkan dan dapat ditemukan. Metode yang digunakan yaitu metode indeks mengambil indeks yang telah tertera di buku untuk menemukan kalimat apa saja yang ditemukan. Dengan pengolahan yang lebih mendetail pada proses pengembangan *query* dapat melakukan pengelompokkan berdasarkan kata dapat dirangkum lebih simpel (Heriyanto, 2015).

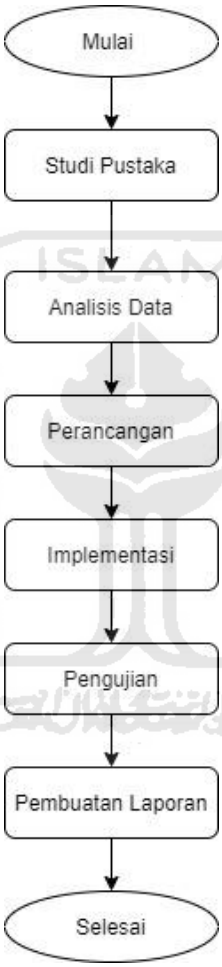
Penelitian ketiga berjudul *"Single Document Keywords Extraction in Bahasa Indonesia using Phrase Chunking"* (Trisna & Nurwidyantoro, 2020). Pada penelitian ini membahas menghasilkan kata kunci dari dokumen secara otomatis menggunakan *phrase chunking. Phrase chunking* dilakukan menggunakan pola *Part of Speech* (POS) untuk mengekstrak frasa dan kata kunci. Kata kunci dipilih dari kandidat berdasarkan jumlah kata dalam frasa dan beberapa tipe bentuk data frasa yaitu kata kunci terpendek dengan pengurangan *string* yang di

ekstraksi dari abstrak dan diurutkan berdasarkan frekuensi nilai tertinggi. Penelitian ini melakukan percobaan menggunakan tipe bentuk frasa seperti sumber kata kunci yang berbeda, dan variasi pola frasa serta memberikan perbandingan kinerja antara tipe bentuk frasa tersebut.

Penelitian yang keempat membahas tentang penggunaan melalui pendekatan *retrieval* yang berfokus untuk mengusulkan solusi masalah dengan mengandalkan konsep daripada kata kunci untuk pengindeksan dan pengambilan. Tujuan nya adalah mengambil dokumen yang relevan dengan permintaan pengguna yang telah diberikan. Pengindeksan dan pembobotan dokumen dan *query* menggunakan entitas semantik, konsep dan kata kunci. Dalam pendekatan ini, konsep yang diidentifikasi dari dokumen dan diukur menurut distribusi frekuensi dan kesamaan dengan konsep lain dalam dokumen (Boubekeur & Wassila, 2013).

BAB III
METODOLOGI PENELITIAN

Pada bab ini, peneliti akan menjelaskan tentang metodologi penelitian dalam pembuatan metode *indexing* berdasarkan hasil psikotes beserta tahapan-tahapan yang dilakukan mulai dari studi pustaka sampai tahap pembuatan laporan. Diagram alur metode penelitian dapat dilihat pada Gambar 3.1.



Gambar 3.1 Diagram Alur Metodologi Penelitian

3.1 Studi Pustaka

Pada tahapan ini dilakukan dengan cara mengumpulkan informasi melalui referensi terkait dengan penelitian ini melalui buku, jurnal, skripsi, artikel dari Internet seperti *google scholar*,

science direct, atau *researchgate* serta penelitian terdahulu sebagai referensi dalam pengembangan sistem pendukung keputusan.

3.2 Pengumpulan Data

Data yang digunakan dalam penelitan ini merupakan data sekunder. Data sekunder merupakan data yang tidak secara langsung didapatkan oleh peneliti, data tersebut didapatkan dari lembaga atau institusi. Institusi yang menjadi sumber pada penelitian ini yaitu Fakultas Kedokteran. Data yang digunakan pada penelitian ini berasal dari hasil tes psikometri EPPS calon pendaftar Fakultas Kedokteran tahun 2018. Hasil tes psikometri EPPS yang didapatkan berupa file berbentuk pdf. Data ini terdiri dari biodata diri (Kode ujian, UPCM, Nama, Durasi dan Konsistensi), Tabel psikometri dengan 15 aspek kebutuhan EPPS, dan kesimpulan yang didapatkan dari tabel psikometri EPPS. Selanjutnya, data tersebut akan diekstraksi informasinya untuk mendapatkan informasi mengenai kepribadian calon pendaftar yang sesuai dengan aspek kebutuhan. Jumlah data yang didapatkan yaitu 74 hasil psikotes yang nantinya akan di *split* ke dalam data latih dan data uji. Gambar 3.2 menunjukkan contoh data psikotes EPPS.



4/5/2018 Hasil Psikotest

Hasil Pemeriksaan Psikologi

Kode Ujian [redacted] Instansi
UPCM [redacted] Durasi 29 menit
Nama [redacted] Konsistensi 11/15

Psikogram

	Aspek Kebutuhan	Kategori				
		KS	K	S	T	TS
1	**Achievement** Kemauan untuk menunjukkan adanya suatu prestasi				V	
2	**Deference** Kemauan untuk menyesuaikan diri dan mengikuti aturan				V	
3	**Order** Kemauan untuk mengatur hidupnya, barang-barang, ide-ide dan tanggung jawab		V			
4	**Exhibition** Kemauan untuk mengkespresikan diri, optimistis, dan percaya diri				V	
5	**Autonomy** Kemauan untuk mandiri, dan tidak tergantung pada orang lain dalam hal pendirian/pendapat			V		
6	**Affiliation** Kebutuhan akan perhatian dari orang lain dan pergaulan yang harmonis			V		
7	**Intraception** Kebutuhan untuk berempati dan bersimpati kepada orang lain			V		
8	**Succorance** Kebutuhan untuk dimanjakan oleh orang lain			V		
9	**Dominance** Kebutuhan untuk mendominasi, memerintah, dan mempengaruhi orang lain		V			
10	**Abasement** Kebutuhan untuk mengakui kesalahan diri		V			
11	**Nurturance** Kebutuhan untuk menolong, merawat orang/ benda				V	
12	**Change** Kebutuhan untuk melakukan perubahan			V		
13	**Endurance** Kebutuhan untuk menyelesaikan sesuatu dengan tekun dan ulet			V		
14	**Heterosexual** Kebutuhan terhadap kehidupan seksual			V		
15	**Aggression** Kebutuhan untuk mendobrak sesuatu secara agresif	V				

Uraian

Hasil pemeriksaan psikologis menunjukkan bahwa saudara/i [redacted] memiliki kebutuhan yang tinggi untuk menyesuaikan diri dan mengikuti aturan/konvensi yang telah ditetapkan. Selain itu, saudara/i [redacted] memiliki kebutuhan yang tinggi untuk mencapai suatu prestasi atau memiliki dorongan meraih sesuatu dalam hidup serta memiliki kebutuhan yang tinggi untuk menolong dan merawat orang atau benda.

Namun, saudara/i [redacted] kurang memiliki dorongan untuk melawan dan menahan diri untuk berperilaku agresif. Selain itu, saudara/i [redacted] kurang menyukai situasi yang menuntutnya menjadi ketua kelompok atau yang mengatur orang lain serta kurang memiliki kebutuhan untuk mempengaruhi orang lain serta tidak terlalu suka menyalahkan diri sendiri, kurang menyukai keteraturan dalam bekerja dan cenderung kurang sistematis saat mengerjakan tugas, dan cukup tergantung pada orang lain dalam melakukan sesuatu termasuk dalam pengambilan keputusan.

Pemeriksa,

Uly Gusniarti, S.Psi., M.Si., Psikolog
0098-14-1-2

http://epps.tk/calculation/epps/1831654/print 1/1

Gambar 3.2 Hasil Psikotes EPPS

3.3 Analisis Data

Pada tahap ini, frasa yang telah didapatkan akan dikelompokkan ke dalam 15 aspek kebutuhan yang dijadikan sebagai acuan pada proses pencarian kebutuhan tertinggi dan terendah. Terdapat 15 kebutuhan atau *needs* beserta label yang dipasangkan dapat dilihat pada Tabel 3.1 di bawah ini:

Tabel 3.1 Aspek Kebutuhan beserta Label

Aspek Kebutuhan	**Kata Kunci**
Achievement (ACH)	Memperoleh sesuatu, mencapai prestasi, mencapai, prestasi, meraih sesuatu
Deference (DEF)	Menyesuaikan diri, mengikuti aturan, patuh terhadap aturan, patuh, aturan
*Order (*ORD)	Mengatur hidup, barang, ide, keteraturan, sistematis
Exhibition (EXH)	Mengekspresikan diri, kepercayaan diri, yakin dengan kemampuan, yakin, kemampuan
Autonomy (AUT)	Mandiri, tidak tergantung, pendirian/pendapat, tergantung pada orang, tergantung
Affiliation (AFF)	Memberikan perhatian, pergaulan harmonis, pergaulan, harmonis, lingkungan sosial, menaruh perhatian, berinteraksi
Intraception (INT)	Berempati, bersimpati, kepekaan, memahami, merasakan, memahami orang
Succorance (SUC)	Dimanjakan, diperhatikan, diurus
Dominance (DOM)	Mendominasi, memerintah, mempengaruhi, mengatur orang, menonjolkan diri, ketua kelompok
Abasement (ABS)	Mengakui kesalahan, perasaan bersalah, menyalahkan diri
Nurturance (NUR)	Menolong, merawat orang/benda,menjaga barang, merawat orang, manusia
Change (CHG)	Melakukan perubahan, menghindari rutin, sesuatu yang tetap, tetap
Endurance (END)	Menyelesaikan pekerjaan, ketekunan, keuletan, kegigihan, ketahanan, teratur
Heterosexual (HET)	Seksual, mengikuti pendapat,
Aggression (AGG)	Mendobrak, berperilaku agresif

Daftar frasa juga didapatkan berdasarkan level kategori sebagai patokan aspek kebutuhan pada komponen penilaian pada uraian hasil tes EPPS mulai dari Tinggi Sekali (TS) hingga Kurang Sekali (KS) dan diinterpretasikan ke dalam bentuk kesimpulan yang ada pada hasil EPPS sebagai kebutuhan tertinggi dan terendah yang nantinya dijadikan dikenali. Hasil kesimpulan pada tes EPPS berdasarkan level kategori yang didapatkan oleh calon pendaftar FK UII.

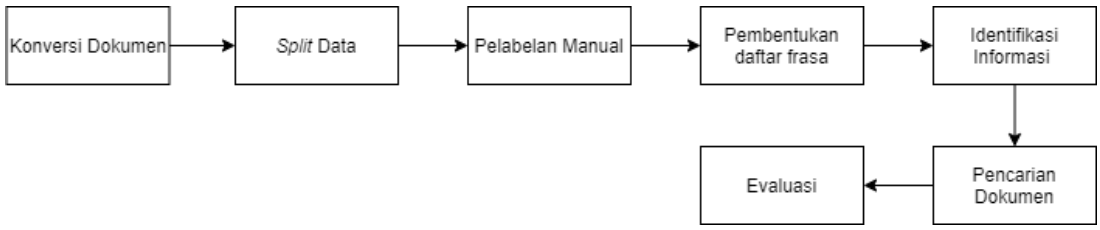
Gambar 3.3 menunjukkan salah satu contoh kesimpulan pada hasil psikometri tes EPPS yang mengandung jawaban terkait kepribadian individu untuk mengidentifikasi informasi mengenai kebutuhan tertinggi atau terendah calon pendaftar.

Hasil pemeriksaan psikologis menunjukkan bahwa saudara/i …. memiliki kebutuhan yang tinggi untuk **menyesuaikan diri** dan **mengikuti aturan/konvensi** yang telah ditetapkan. Selain itu, saudara/i …. memiliki kebutuhan yang tinggi untuk **mencapai suatu prestasi** atau memiliki dorongan **meraih sesuatu** dalam hidup serta memiliki kebutuhan yang tinggi untuk **menolong** dan **merawat orang** atau **benda**.

Gambar 3.3 Hasil Psikotes EPPS

3.4 Perancangan Sistem

Perancangan merupakan upaya untuk menemukan solusi atas permasalahan untuk menciptakan sesuatu melalui visual menggunakan konsep-konsep yang ideal. Diagram dibuat dalam bentuk *flowchart* yang menggambarkan proses pembuatan sistem mulai dari masukan data sampai pengujian. Gambar 3.4 menunjukkan alur sistem secara umum bekerja.



Gambar 3.4 Diagram Alur Perancangan Sistem

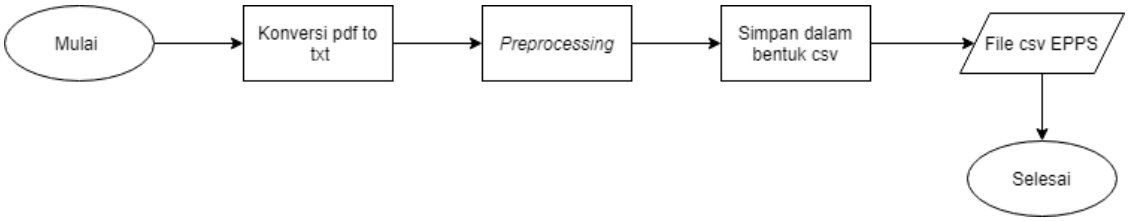
Langkah pertama yang dilakukan dalam pembuatan sistem ini adalah membaca data hasil tes EPPS dan mengambil indeks berupa paragraf ”uraian” pada tes EPPS. Setelah berhasil dilakukan teks tersebut di proses melalui *preprocessing* yang mengubah huruf kapital menggunakan *lower case* dan mendapatkan token-token setiap kata dengan tokenisasi. Hasil

preprocessing setiap dokumen file pdf akan disimpan dan diubah menjadi file .csv yang terdiri dari dua baris yaitu frasa dan label.

Langkah selanjutnya, file akan di *split* secara random menjadi data latih dan data uji. File csv dokumen akan dilabelkan manual oleh penulis. Pelabelan manual dilakukan dengan memberikan nama setiap pasangan label dengan frasa aspek kebutuhan yang sesuai. Selanjutnya, untuk mendapatkan hasil frasa beserta pasangan labelnya dilakukan melalui pembentukan daftar frasa yang akan disimpan dalam file csv bernama "frasa_list.csv". Selanjutnya dilakukan identifikasi informasi dalam dokumen dengan indeks daftar frasa untuk menemukan kecocokan kata berupa pasangan frasa dan label pada kebutuhan tertinggi dan terendah setiap dokumen serta dapat menyebutkan aspek kebutuhan yang muncul pada kalimat tersebut. Selanjutnya, melakukan pengujian dan evaluasi pada data uji untuk menentukan nilai derajat yang mirip terhadap data rekam.

3.4.1 Konversi Dokumen

Pada tahap ini, data hasil psikotes calon pendaftar telah terkumpulkan menjadi satu folder yang berbentuk pdf. Dokumen tersebut akan menjadi data masukan yang akan diolah. Hasil tes EPPS terdapat dua paragraf yang berisikan kalimat tentang kebutuhan tertinggi dan kebutuhan terendah calon pendaftar. Data pada hasil tes EPPS akan diambil melalui pengambilan indeks paragraf "uraian" yang muncul pada dokumen dan melakukan proses *preprocessing.* Gambar 3.5 menunjukkan alur tahapan pada konversi dokumen dan Gambar 3.5 menunjukkan contoh paragraf kesimpulan pada hasil tes EPPS.



Gambar 3.5 Alur proses konversi dokumen

Penjelasan *flowchart* alur proses konversi dokumen:

a. Konversi pdf *to text*: Pada proses ini data pdf akan diubah ke dalam bentuk teks untuk mengambil seluruh kata yang ada pada file pdf.

b. *Preprocessing*: Pada proses ini kata yang telah dikonversi akan diubah menggunakan *preprocessing* yaitu *casefolding* (mengubah huruf depan menjadi huruf kecil) dan tokenisasi (memisahkan setiap kata menjadi satu baris kata saja).
c. Simpan dalam bentuk csv: Pada proses ini file akan disimpan ke dalam bentuk file csv. File tersebut berisikan tabel frasa dan label.
d. File csv EPPS : File akan disimpan bersama dengan folder penyimpanan pdf nya.

Hasil pemeriksaan psikologis menunjukkan bahwa saudara/i …. memiliki kebutuhan yang tinggi untuk menyesuaikan diri dan mengikuti aturan/konvensi yang telah ditetapkan. Selain itu, saudara/i …. memiliki kebutuhan yang tinggi untuk mencapai suatu prestasi atau memiliki dorongan meraih sesuatu dalam hidup serta memiliki kebutuhan yang tinggi untuk menolong dan merawat orang atau benda.

Namun, saudara/i …. kurang memiliki dorongan untuk melawan dan menahan diri untuk berperilaku agresif. Selain itu, saudara/i …. kurang menyukai situasi yang menuntutnya menjadi ketua kelompok atau yang mengatur orang lain serta kurang memiliki kebutuhan untuk mempengaruhi orang lain serta tidak terlalu suka menyalahkan diri sendiri, kurang menyukai keteraturan dalam bekerja dan cenderung kurang sistematis saat mengerjakan tugas, dan cukup tergantung pada orang lain dalam melakukan sesuatu termasuk dalam pengambilan keputusan.

Gambar 3.6 Contoh Teks Hasil tes EPPS

3.4.1.1 Preprocessing

Pada tahap *preprocessing* ini merupakan proses tahap awal yang dipersiapkan untuk melakukan pengolahan lebih lanjut terhadap teks pada data sebelum masuk ke dalam proses pengindeksan. Proses dalam tahap *preprocessing* ini hanya meliputi beberapa tahap yaitu *casefolding* dan *tokenizing*. Tahap *stopwords* tidak dilakukan karena dapat menghilangkan kata yang punya makna satu-kesatuan. Hasil pada *prerocessing* nantinya akan disimpan dan dikonversi menjadi csv. File csv ini menyambung dengan pengambilan indeks kata pada paragraf “uraian” pada hasil tes EPPS. Adapun urutan langkah-langkah pada tahap *preprocessing* dapat dilihat sebagai berikut:

a. *Casefolding*

Proses mengubah semua karakter dalam dokumen ke dalam kasus yang sama, baik semua huruf yang besar maupun yang kecil untuk mempercepat perbandingan selama proses pengindeksan. Dalam kasus ini, karakter pada data akan diubah menjadi huruf kecil (*lower case).* Tabel 3.2 menunjukkan contoh penerapan proses *casefolding*.

Tabel 3.2 Contoh Hasil *Casefolding*

Sebelum	**Sesudah**
Hasil pemeriksaan psikologis menunjukkan bahwa saudara/i …. memiliki kebutuhan yang tinggi untuk menyesuaikan diri dan mengikuti aturan/konvensi yang telah ditetapkan. Selain itu, saudara/i …. memiliki kebutuhan yang tinggi untuk mencapai suatu prestasi atau memiliki dorongan meraih sesuatu dalam hidup serta memiliki kebutuhan yang tinggi untuk menolong dan merawat orang atau benda.	hasil pemeriksaan psikologis menunjukkan bahwa saudara/i …. memiliki kebutuhan yang tinggi untuk menyesuaikan diri dan mengikuti aturan/konvensi yang telah ditetapkan. Selain itu, saudara/i …. memiliki kebutuhan yang tinggi untuk mencapai suatu prestasi atau memiliki dorongan meraih sesuatu dalam hidup serta memiliki kebutuhan yang tinggi untuk menolong dan merawat orang atau benda.

b. Tokenisasi

Tokenisasi merupakan langkah dalam melakukan proses pemisahan *string* menjadi daftar token. Token adalah bagian dari kesuluruhan teks, sehingga kata merupakan token dalam kalimat dan kalimat merupakan token dalam paragraf. Token dapat dianggap menjadi satu bentuk kata, frasa atau elemen yang bermakna. Tahapan ini dilakukan setelah melewati proses *preprocessing.* Tabel 3.3 menunjukkan contoh proses tokenisasi.

Tabel 3.3 Contoh Hasil *Tokenization*

Sebelum	**Sesudah**
hasil pemeriksaan psikologis menunjukkan bahwa saudara/i …. memiliki kebutuhan yang tinggi untuk menyesuaikan diri dan mengikuti aturan/konvensi yang telah ditetapkan.	[“hasil”,”pemeriksaan”,”psikologis”, “menunjukkan”,”bahwa”,“saudara/i”,”memiliki”, “kebutuhan”,”yang”,”tinggi”,”untuk”, “menyesuaikan”,”diri”,”dan”,”mengikuti”, ”aturan/konvensi”,”yang”,“telah” “ditetapkan”,”.”]

3.4.2 *Split* Data

Pada tahap ini, dataset dibagi secara random menjadi data latih dan data uji dengan perbandingan rasio 80% : 20%. Data latih akan dijadikan sebagai sumber untuk pembentukan daftar frasa dan label yang nantinya akan di cari kecocokan nya dengan dokumen. Sedangkan, data uji berguna untuk mencari kemunculan dengan nilai yang mirip dengan data rekamnya.

3.4.3 Pelabelan Manual

Pada tahap ini, file csv yang telah di konversi akan berisikan tabel ‘frase’ dan ‘label’. Pada tabel frase telah di isi oleh sistem dari hasil konversi sedangkan pada tabel label masih kosong sehingga harus dilakukan pelabelan manual oleh penulis. Pada label tersebut akan dimasukkan secara manual berdasarkan nama singkatan aspek kebutuhan. Frasa yang labelkan berisikan inisial “*other”* atau “O” jika terdapat frasa tidak berkaitan dengan label dan berinisial dari 15 aspek kebutuhan jika frasa memiliki kaitan dengan labelnya.

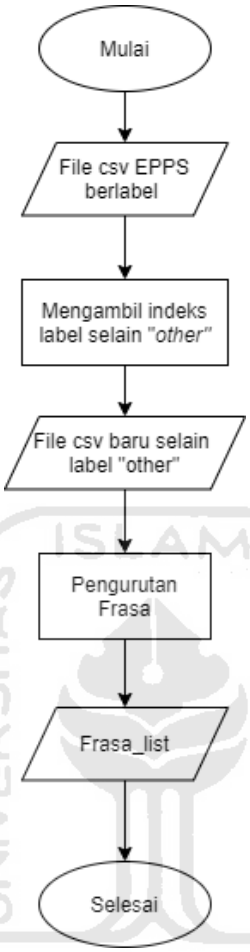
Nama singkatan dari aspek kebutuhan antara lain:

ACH: *Achievement*
DEF: *Deference*
ORD: *Order*
EXH: *Exhibition*
AUT: *Autonomy*
AFF: *Affiliation*
INT: *Intraception*
SUC: *Succorance*
DOM: *Dominance*
ABS: *Abasement*
NUR: *Nurturance*
CHG: *Change*
END: *Endurance*
HET: *Heterosexual*
AGG: *Aggresion*

3.4.4 Pembentukan Daftar Frasa

Pada tahapan ini, data latih yang telah dilabelkan akan mengambil indeks setiap kata yang mempunyai inisial selain “other” untuk memisahkan frasa yang akan disimpan dalam bentuk

file csv baru bernama ‘frasa_list.csv’. Frasa tersebut mengambil setiap pasangan frasa dan label yang telah diinputkan secara manual oleh pakar. Gambar 3.7 menunjukkan tahapan pembentukan daftar frasa.



Gambar 3.7 Alur proses pembentukan daftar frasa

Penjelasan *flowchart* pada tahapan pembentukan daftar frasa:

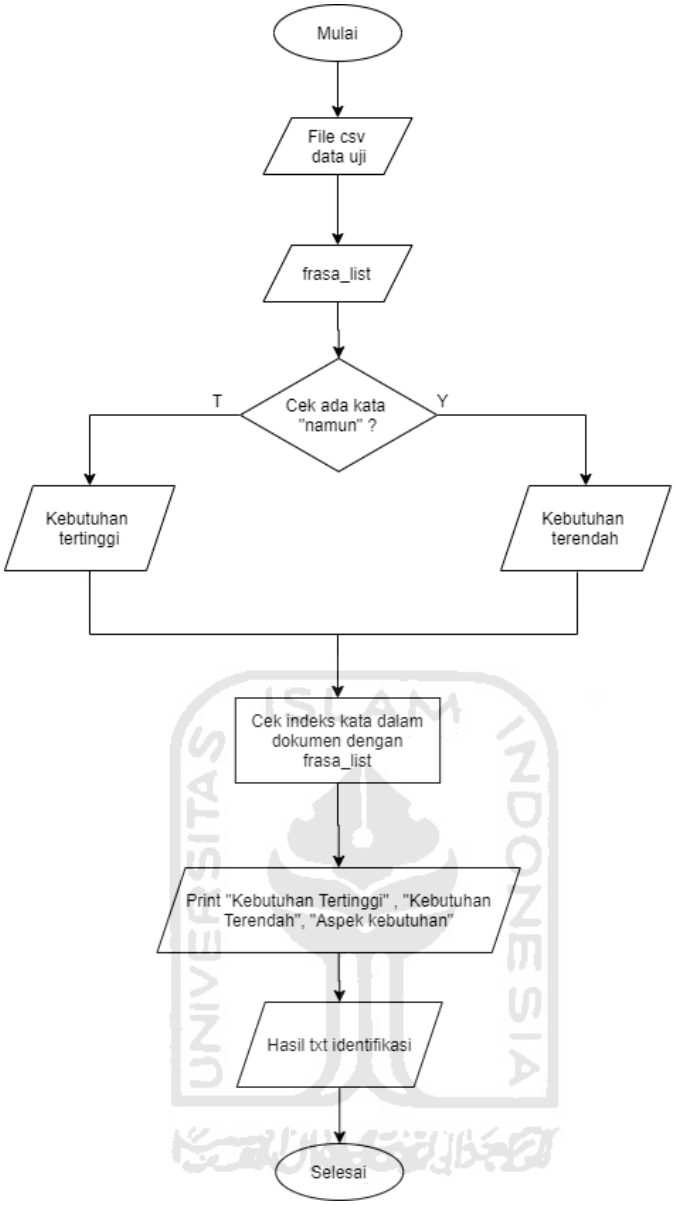
1. File csv EPPS berlabel : Sebagai inputan untuk mengambil data pada tabel frasa dan label setiap file.
2. Mengambil indeks label selain “*other*” : Label yang bukan berinisial ‘O’ atau *other* pada file akan diambil dan dipindahkan ke dalam file csv baru.
3. File csv baru selain label “O” : Pada proses ini file csv akan menyimpan label yang memiliki inisial aspek kebutuhan yang disebutkan pada pelabelan manual.
4. Pengurutan frasa : Frasa pada frasa_list akan diurutkan secara random berdasarkan kata yang dilabelkan secara panjang serta pendek yang dilabelkan oleh penulis pada proses pelabelan manual. Jika pada proses pelabelan manual, penulis melabelkan tiga atau lebih

kata / token pada file maka akan keluar pada urutan teratas sedangkan semakin sedikit kata yang dilabelkan maka akan keluar pada urutan terbawah pada frasa_list.

5. Frasa_list : *Output* yang dihasilkan akan disimpan ke dalam bentuk csv yang bernama "frasa_list.csv".

3.4.5 Identifikasi Informasi

Pada penelitian ini menggunakan metode pengindeksan kata untuk menemukan informasi terkait calon pendaftar. Informasi yang ingin dicari adalah label pada setiap filenya. Pada tahap ini sistem akan mengidentifikasi frasa dengan mencocokkan label yang telah ditentukan dalam frasa_list. Untuk melakukan proses ini, diperlukan data uji untuk mengetahui kesesuai model yang telah dibuat untuk menemukan informasi yang muncul beserta nama aspek kebutuhannya. Gambar 3.8 menunjukkan tahapan pada proses penemuan informasi.



Gambar 3.8 Alur proses identifikasi informasi

Penjelasan *flowchart* pada tahap identifikasi informasi:

a. File csv data uji : Sebagai inputan pada proses pengindeksan informasi.

b. Frasa_list : inputan sebagai acuan untuk mengindentifikasi frasa dalam dokumen yang dicocokkan labelnya pada frasa list.

c. Jika terdapat kata “namun” setelah kalimat tersebut, maka sistem akan memisahkan kalimat tersebut sebagai menjadi kebutuhan tertinggi dan kebutuhan terendah.

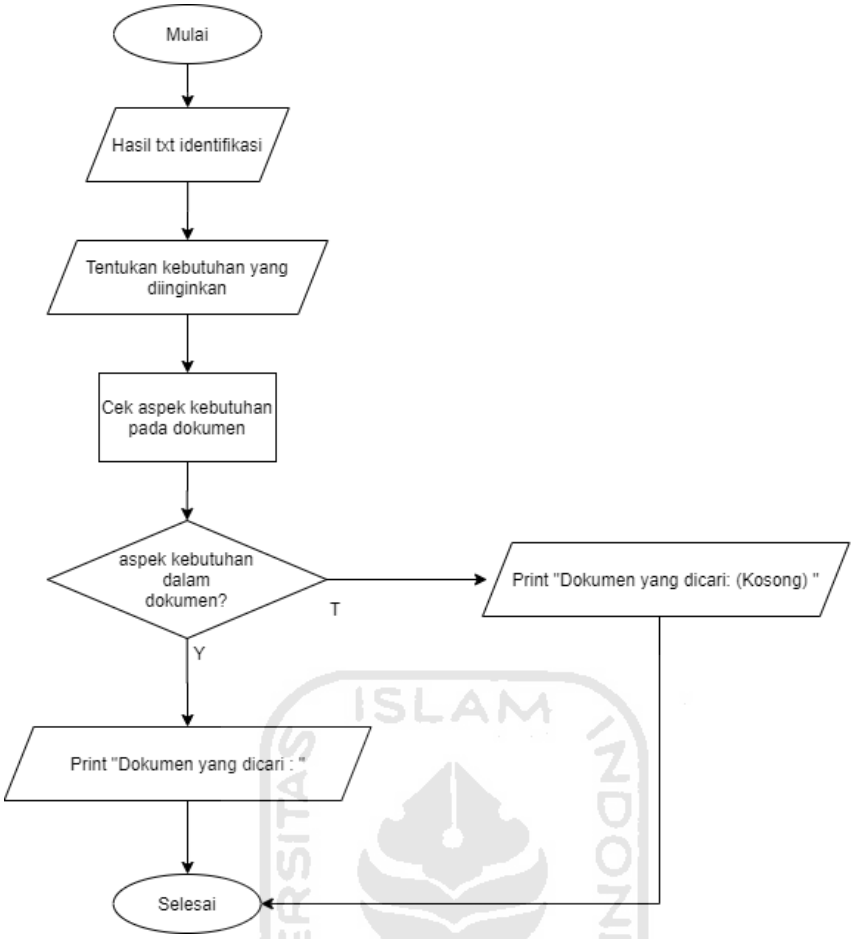
d. Cek indeks kata dalam dokumen dengan frasa : Mengecek indeks frasa apa saja yang muncul pada dokumen untuk mengidentifikasi kalimat yang merujuk pada label pada frasa_list.

e. Print “Kebutuhan Tertinggi”, “Kebutuhan Terendah”, “Aspek Kebutuhan” : Keluaran jika proses pencocokan selesai. Sistem akan menghasilkan nama frasa yang terdapat pada dokumen serta label yang sesuai dan terbagi ke dalam kebutuhan tertinggi dan terendah serta menampilkan inti aspek kebutuhan apa saja yang muncul.

f. Hasil txt identifikasi : Keluarannya akan memberikan keterangan frasa yang muncul pada setiap file beserta dengan labelnya dan rangkuman aspek kebutuhan yang keluar pada file tersebut. Hasil txt disimpan dalam folder yang berbeda dengan folder dokumennya.

3.4.6 Pencarian Dokumen

Pada tahap ini setelah mendapatkan hasil pencarian dari aspek kebutuhan yang diinginkan, hasil tersebut akan menjadi acuan untuk melihat mana saja calon pendaftar FK UII dengan kebutuhan tertinggi pada aspek tersebut. Dengan melakukan pencarian dokumen dapat memudahkan pihak penyelenggara untuk menemukan dokumen calon pendaftar FK UII antara 15 *needs* yang diinginkan sebagai komponen penilaian selanjutnya. Contoh, sistem dapat mengenali *needs* “ACH” sebagai kebutuhan tertinggi maka sistem akan menampilkan data calon pendaftar mana saja yang memiliki kebutuhan tertinggi pada *needs* tersebut. Gambar 3.9 menunjukkan tahapan pada pencarian dokumen calon pendaftar FK UII.



Gambar 3.9 Alur proses pencarian dokumen

Penjelasan *flowchart* pada tahapan pencarian dokumen:

1. Hasil txt identifikasi: Sebagai inputan sebelumnya untuk mencari dokumen dengan informasi tersebut.
2. Tentukan kebutuhan yang diinginkan: Pengguna dapat memilih dari 15 aspek kebutuhan untuk melihat dokumen yang dicari.
3. Cek aspek kebutuhan pada dokumen : melalui proses ini nanti mengecek manakah dalam dokumen yang dekat dengan label aspek kebutuhan yang dipilih oleh pengguna.
4. Jika ditemukan label aspek kebutuhan pada kalimat kebutuhan tertinggi maka akan disimpan sebagai array nama dokumen yang kebutuhan tertinggi nya memiliki label tersebut. Proses pencarian nama dokumen didapatkan ketika pengguna melakukan inputan aspek kebutuhan pada dokumen apa yang ingin dilihat. Informasi tersebut telah didapatkan dari proses identifikasi informasi sehingga dokumen dapat ditampilkan.
5. *Print* “Dokumen yang dicari :” : Ketika label yang dicari dalam array dokumen tersebut, maka sistem akan menjalankan perintah *print* indeks pada array dokumen dan

ditampilkan sebagai “Dokumen yang dicari:” beserta nama dokumen calon pendaftarnya dalam bentuk pdf.

6. *Print* “Dokumen yang dicari: (kosong) ” yang artinya hasil tersebut akan tetap dikeluarkan hanya saja kosong tanpa hasil apapun karena tidak ditemukan dokumen yang dicari.

3.4.7 Uji dan Evaluasi Model

Selanjutnya, dalam melakukan pengujian model dibutuhkan data asli dan data baru yang diujikan mendapatkan kemunculan nilai derajat yang mirip. Setelah mendapatkan dokumen yang berisikan informasi yang relevan, data uji akan melakukan evaluasi terhadap data aslinya terhadap data baru dengan mencari kemunculan nilai derajat yang mirip. Kemudian, mengevaluasi setiap tipe bentuk frasa menggunakan *precision, recall,* dan *f-measure. Precision* adalah ketepatan nilai antara permintaan pengguna dengan sistem, *recall* adalah ketepatan dalam menemukan kembali informasi yang telah tersimpan pada sistem dan *f-measure* adalah perbandingan rata-rata antara precision dan recall yang telah dibobotkan.Jumlah data yang digunakan sebagai data uji sebesar 20% sama dengan 0,20.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Implementasi Program

Pada tahap ini, tahap pembuatan model dari data yang didapatkan untuk menghasilkan sistem yang mampu mengidentifikasi setiap kata yang muncul yang berkaitan dengan 15 aspek kebutuhan yang telah ditentukan dan memberikan informasi aspek kebutuhan apa saja yang muncul pada kesimpulan tes EPPS tersebut. Dalam membantu pembuatan model ini, penelitian menggunakan bahasa pemrograman *python3* melalui terminal dan IDE *Visual Code.* Penjelasan proses dalam pembuatan model dapat dilihat sebagai berikut:

4.1.1 Konversi Dokumen

Pada tahap ini, data file pdf yang telah tersimpan akan di konversi ke dalam bentuk file csv dan mengambil paragraf pada indeks "uraian". Sebelum mengubah ke dalam bentuk file csv, data teks diubah terlebih dahulu menjadi txt untuk mendapatkan seluruh teks yang ada pada file pdf. Kode program dijalankan menggunakan terminal *python3.* Gambar 4.1 menunjukkan kode program untuk pengambilan kalimat pada indeks "uraian".

```
    def convert_pdf_to_txt(path):
        rsrcmgr = PDFResourceManager()
        retstr = StringIO()
        laparams = LAParams()
        device = TextConverter(rsrcmgr, retstr, laparams=laparams)
        fp = open(path, 'rb')
        interpreter = PDFPageInterpreter(rsrcmgr, device)
        password = ""
        maxpages = 0
        caching = True
        pagenos=set()

        for page in PDFPage.get_pages(fp, pagenos, maxpages=maxpages,
password=password,caching=caching, check_extractable=True):
            interpreter.process_page(page)

        text = retstr.getvalue()

        fp.close()
        device.close()
        retstr.close()
        return text
```

Gambar 4.1 Kode program *pdf to txt*

Selanjutnya, teks akan mengambil kalimat pada indeks “uraian” yang akan disimpan ke dalam bentuk file csv. Gambar 4.2 menunjukkan kode program pada proses ini dan Gambar 4.3 menunjukkan contoh kalimat yang akan di konversi. Kode program yang lengkap dapat dilihat pada bagian lampiran.

```
def get_main_text(text):
     text_list = text.split('\n')
     text_list=
    text_list[text_list.index('Uraian'):text_list.index('Pemeriksa,')][1:]
     return_text = ' '.join(text_list)
     return return_text.lower()
```

Gambar 4.2 Kode program untuk mengambil indeks uraian

Hasil pemeriksaan psikologis menunjukkan bahwa saudara/i …. memiliki kebutuhan yang tinggi untuk menyesuaikan diri dan mengikuti aturan/konvensi yang telah ditetapkan. Selain itu, saudara/i …. memiliki kebutuhan yang tinggi untuk mencapai suatu prestasi atau memiliki dorongan meraih sesuatu dalam hidup serta memiliki kebutuhan yang tinggi untuk menolong dan merawat orang atau benda.

Namun, saudara/i …. kurang memiliki dorongan untuk melawan dan menahan diri untuk berperilaku agresif. Selain itu, saudara/i …. kurang menyukai situasi yang menuntutnya menjadi ketua kelompok atau yang mengatur orang lain serta kurang memiliki kebutuhan untuk mempengaruhi orang lain serta tidak terlalu suka menyalahkan diri sendiri, kurang menyukai keteraturan dalam bekerja dan cenderung kurang sistematis saat mengerjakan tugas, dan cukup tergantung pada orang lain dalam melakukan sesuatu termasuk dalam pengambilan keputusan.

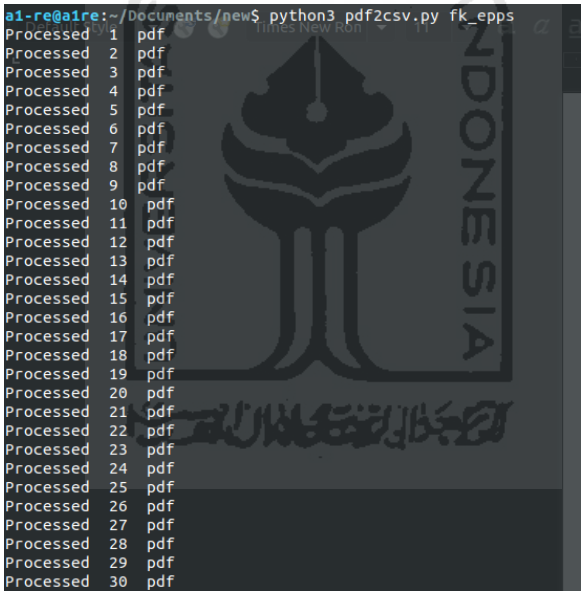
Gambar 4.3 Contoh kalimat pada uraian

Selanjutnya, file pdf akan disimpan ke dalam bentuk file csv dan tersimpan dalam satu folder dengan pdfnya. Kode program dapat dilihat pada Gambar 4.4 menunjukkan kode program untuk mengkonversi dokumen pdf.

```
if __name__ == '__main__':
    counter = 1
    for pdf in os.listdir(sys.argv[1]):
        print('Processed ',counter,' pdf')
        text = convert_pdf_to_txt('fk_epps//'+pdf)
        text = get_main_text(text)
        list_ = word_tokenize(text)
        table= pd.DataFrame(columns=['kata','label'])
        table['kata'] = list_
        table.to_csv(sys.argv[1]+'//'+pdf.strip('.pdf')+'.csv',index=None
)
        counter += 1
```

Gambar 4.4 Kode program untuk konversi dokumen

Pada Gambar 4.5 ini menunjukkan bahwa kode program jika konversi file pdf berhasil dilakukan dan tersimpan bersama file csv nya. Sistem dijalankan melalui terminal dalam bahasa pemrograman *python3*.



Gambar 4.5 Hasil konversi dokumen

4.1.1.1 *Preprocessing*

Tahap *preprocessing* merupakan tahap awal sebelum data diolah ke tahap selanjutnya. Pada penelitian ini hanya dilakukan dua tahap secara urut, yaitu:

a. *Case folding*

Tahap pada *case folding* adalah untuk menyamakan bentuk huruf dari keseluruhan teks menjadi huruf kecil. Dalam tahap ini, kode program *case folding* digabungkan dengan kode program pengambilan kalimat pada indeks teks agar secara otomatis berubah. Kode program

dijalankan menggunakan terminal *python3*. Gambar 4.6 dan Gambar 4.7 menunjukkan implementasi kode program *casefolding* beserta dengan hasilnya.

```
return return_text.lower()
```

Gambar 4.6 Kode program *case folding*

Sebelum	Sesudah
Hasil pemeriksaan psikologis menunjukkan bahwa saudara/i …. memiliki kebutuhan yang tinggi untuk menyesuaikan diri dan mengikuti aturan/konvensi yang telah ditetapkan. Selain itu, saudara/i …. memiliki kebutuhan yang tinggi untuk mencapai suatu prestasi atau memiliki dorongan meraih sesuatu dalam hidup serta memiliki kebutuhan yang tinggi untuk menolong dan merawat orang atau benda.	hasil pemeriksaan psikologis menunjukkan bahwa saudara/i …. memiliki kebutuhan yang tinggi untuk menyesuaikan diri dan mengikuti aturan/konvensi yang telah ditetapkan. Selain itu, saudara/i …. memiliki kebutuhan yang tinggi untuk mencapai suatu prestasi atau memiliki dorongan meraih sesuatu dalam hidup serta memiliki kebutuhan yang tinggi untuk menolong dan merawat orang atau benda.

Gambar 4.7 Hasil *casefolding*

b. Tokenisasi

Pada tahap tokenisasi adalah pemisah antara kata, simbol, frasa, dan entitas dari suatu teks. Dalam tahap ini, kode program *case folding* digabungkan dengan kode program pengambilan kalimat pada indeks teks agar secara otomatis berubah. Kode program dijalankan menggunakan terminal *python3*. Gambar 4.8 dan Gambar 4.9 menunjukkan implementasi kode program tokenisasi beserta hasilnya. Kode program yang lengkap dapat dilihat pada bagian lampiran.

```
list_ = word_tokenize(text)
```

Gambar 4.8 Kode program tokenisasi

Sebelum	Sesudah
hasil pemeriksaan psikologis menunjukkan bahwa saudara/i …. memiliki kebutuhan yang tinggi untuk menyesuaikan diri dan mengikuti aturan/konvensi yang telah ditetapkan.	[“hasil”,”pemeriksaan”,”psikologis”, “menunjukkan”,”bahwa”,“saudara/i”,”memiliki”, “kebutuhan”,”yang”,”tinggi”,”untuk”, “menyesuaikan”,”diri”,”dan”,”mengikuti”, ”aturan/konvensi”,”yang”,“telah” “ditetapkan”,”.”]

Gambar 4.9 Hasil tokenisasi

Gambar 4.10 menunjukkan hasil keluaran yang berisikan baris “kata” dan “label” yang telah di tokenisasi dan *preprocessing* yang tersimpan dalam bentuk file csv bersama dengan folder pdf hasil tes EPPS yang berjumlah 74 dokumen.

menyukai	
aktivitas	
yang	
berhubungan	
dengan	
manusia	
dan	
menaruh	
perhatian	
pada	
orang	
lain	
serta	
dapat	
bergaul	
secara	
harmonis	
serta	
memiliki	
perasaan	
bersalah	
yang	
tinggi	
dan	
cenderung	
menyalahkan	
diri	
sendiri	
.	
namun	
,	
saudara/i	
aura	
widad	
al	
addawiyah	


kurang
menunjukkan
keinginan
dan
kebutuhan
untuk
mencapai
prestasi
atau
memperoleh
sesuatu
yang
lebih
dari
orang
lain
dalam
kaitannya
dengan
pendidikan
dan
pekerjaan

Gambar 4.10 Hasil keluaran konversi dokumen

4.1.2 *Split* Data

Pada tahap ini, data teks yang tersimpan dalam satu folder data berisikan gabungan file csv dan file pdf. Kode program dijalankan menggunakan terminal *python3*. Gambar 4.11 menunjukkan implementasi kode program *split* data. Kode program yang lebih lengkap dapat dilihat pada bagian lampiran.

```
import os
import shutil
import random
import sys

files = [i for i in os.listdir(folder) if '.csv' in i]
test_data = random.sample(files,k=int(len(files)*test_size))
train_data = [i for i in files if i not in test_data]
```

Gambar 4.11 Kode program *split* data

Cara menjalankan kode program *split* data pelatihan dan data uji melalui terminal *python3*. Data uji telah di-set secara *default* 20% dari data latihnya. Folder yang di-*split* merupakan hasil konversi teks yang telah disimpan. Gambar 4.12 dan Gambar 4.13 menunjukkan hasil *split* untuk data *train* dan *test* bersama csv nya.



Gambar 4.12 Hasil *split* data *train*



Gambar 4.13 Hasil *split* data *test*

4.1.3 Pelabelan Manual

Pada tahap ini, data latih dan data uji yang telah dikonversikan akan dilabelkan manual oleh penulis dengan melihat beberapa kriteria dari aspek kebutuhan pada tes EPPS. Pelabelan dilakukan secara langsung dari file csv setiap dokumennya. Proses pelabelan manual dapat dilihat pada Gambar 4.14.

50	untuk	O
51	berempati	INT
52	,	O
53	merasakan	INT
54	apa	O
55	yang	O
56	orang	O
57	lain	O
58	rasakan	O
59	dan	O
60	bersimpati	INT
61	kepada	O

70	anggrila	O
71	sekar	O
72	fadhillah	O
73	kurang	O
74	yakin	EXH
75	dengan	EXH
76	kemampuan	EXH
77	diri	O
78	dan	O
79	tidak	O
80	suka	O
81	menonjolkan	DOM
82	diri	DOM
83	dalam	O
84	kelompoknya	O
85	.	O

92	menyukai	O
93	sesuatu	CHG
94	yang	CHG
95	tetap	CHG
96	dan	O
97	rutin	CHG
98	dalam	O
99	diri	O
100	dan	O
101	kehidupan	O
102	sehari-hari	O
103	serta	O
104	kurang	O
105	memiliki	O
106	keinginan	O
107	untuk	O
108	patuh	DEF
109	terhadap	O
110	aturan	DEF
111	dan	O
112	tidak	O

119	tergantung	AUT
120	pada	AUT
121	orang	AUT
122	lain	O
123	dalam	O
124	melakukan	O
125	sesuatu	O
126	termasuk	O
127	dalam	O
128	pengambilan	AUT
129	keputusan	AUT

112	berperilaku	AGG
113	agresif	AGG
114	,	O
115	tidak	O
116	terlalu	O
117	suka	O
118	menyalahkan	ABS
119	diri	ABS
120	sendiri	O
121	,	O
122	dan	O
123	cukup	O
124	tertarik	O
125	untuk	O
126	memberikan	INT
127	perhatian	INT

Gambar 4.14 Hasil pelabelan manual dokumen

4.1.4 Pembentukan Daftar Frasa

Pada tahap ini digunakan untuk mengambil dan memisahkan indeks frasa dan label yang banyak muncul pada setiap file untuk disimpan sebagai array ‘phrase’ dan ‘label’ ke dalam file csv bernama “frasa_list”. Gambar 4.15 menunjukkan kode program pengambilan kalimat pada frasa.

```
def get_txt_from_data(data):
    phrase_list = []
    start = 0
    for i in range(len(data.label))[1:]:
        if data.label[i] != data.label[i-1]:
            if data.label[start] != 'O' and data.label[start] != 'OO':
                phrase = ' '.join(data.kata[start:i])
                phrase = phrase.replace('.','').replace(',','')
                label = data.label[start]
                phrase_list.append((phrase.lower(),label))
            start = i
    return phrase_list
```

Gambar 4.15 Kode program mengambil frasa dari data

Tahap selanjutnya, ketika sudah mendapatkan teks dari data, teks tersebut dipisahkan dan disimpan ke dalam bentuk csv "frasa_list" dengan tabel yang berisi pasangan antara frasa dan labelnya. Proses pengurutan menggunakan fungsi "*sorted*" untuk mengurutkan frasa yang dilabelkan oleh penulis. Gambar 4.16 menunjukkan implementasi kode program simpan daftar frasa. Untuk kode program yang lebih lengkap dapat dilihat pada lampiran.

```
text = sorted(set(text),key=lambda x:len(x[0]),reverse=True)
phrase_list = pd.DataFrame(text,columns=['frase','label'])
phrase_list.to_csv('frasa_list.csv',index=None)
```

Gambar 4.16 Kode program pengurutan dan simpan daftar frasa

Gambar 4.17 menunjukkan hasil pada pasangan frasa dan label yang disimpan ke dalam bentuk file csv baru bernama 'frasa_list' setelah menghapuskan label yang berinisial "O" atau "*other*". Frasa pada frasa_list akan diurutkan berdasarkan frasa yang dilabelkan dengan tiga atau dua kata maupun hanya satu kata saja sehingga tersimpan sesuai dengan kata terpanjang serta terpendek yang diinputkan oleh penulis.

frase	label
mengikuti aturan/konvensi	DEF
mengekspresikan dirinya	EXH
yakin dengan kemampuan	EXH
tergantung pada orang	AUT
pengambilan keputusan	AUT
patuh terhadap aturan	DEF
memberikan perhatian	AFF
merawat orang/benda	NUR
berperilaku agresif	AGG
melakukan perubahan	CHG
mengikuti pendapat	HET
mempengaruhi orang	DOM
pendirian/pendapat	AUT
memperoleh sesuatu	ACH
sesuatu yang tetap	CHG
menaruh perhatian	AFF
perasaan bersalah	ABS
lingkungan sosial	AFF
kehidupan seksual	HET
menyesuaikan diri	DEF
mencapai prestasi	ACH
menyalahkan diri	ABS
tidak tergantung	AUT
menonjolkan diri	DOM
kepercayaan diri	EXH
mengatur orang	DOM
ketua kelompok	DOM
memahami orang	INT
meraih sesuatu	ACH
merawat orang	NUR
berinteraksi	AFF
diperhatikan	SUC
mendominasi	DOM
keteraturan	ORD
bersimpati	INT
memerintah	DOM
sistematis	ORD

memerintah	DOM
sistematis	ORD
dimanjakan	SUC
merasakan	INT
ketekunan	END
berempati	INT
ketahanan	END
kegigihan	END
menolong	NUR
mencapai	ACH
kepekaan	INT
agresif	AGG
harmonis	AFF
keuletan	END
prestasi	ACH
mandiri	AUT
bekerja	ORD
bergaul	AFF
aturan	DEF
diurus	SUC
rutin	CHG
benda	NUR
patuh	DEF
ide	ORD

Gambar 4.17 Hasil pembentukan daftar frasa

4.1.5 Identifikasi Informasi

Pada tahapan ini, sistem akan menyajikan makna yang terkandung pada psikogram tes EPPS agar lebih mudah diketahui aspek kebutuhan apa saja yang muncul dalam kalimat kesimpulan hasil tes EPPS calon pendaftar. Pertama, mencari indeks untuk memisahkan paragraf yang menjelaskan kebutuhan tertinggi dan terendah. Pada kasus ini, kata yang membedakan antara kebutuhan tertinggi dan terendah yaitu kata yang muncul sebagai "Namun" pada awal kalimat. Beberapa karakter yang muncul juga akan digantikan menggunakan library *regular expression* pada python. Implementasi kode program dapat dilihat pada Gambar 4.18 menunjukkan implementasi kode program pemisah paragraf.

```
   main_text = get_main_text(text)
   main_text = ' '+re.sub('(?<! )(?=[.,!?()])|(?<=[.,!?()])(?! )', r' ',
main_text)+' '

   positive_text = main_text.split('Namun ,')[0]
   egative_text = main_text.split('Namun ,')[1]
```

Gambar 4.18 Kode program pemisah paragraf

Tahap selanjutnya, informasi tersebut akan terbagi menjadi dua kondisi yaitu kebutuhan tertinggi dan terendah calon pendaftar beserta hasil frasa dan labelnya. Untuk melakukan pengindeksan informasi terkait kebutuhan tertinggi dan terendah tersebut dibutuhkan data latih untuk mengidentifikasi informasi. Gambar 4.19 menunjukkan implementasi kode program pengindeksan informasi.

```
print('=======Parameter(TS-S)=======')
    phrases = []
    labels = []
    for i in range(len(phrase_list.frase)):
        if ' '+phrase_list.frase[i]+' ' in positive_text:
            if reducted:
                if    bool([string    for    string    in    phrases    if
phrase_list.frase[i] in string]) == False:
                    phrases.append(phrase_list.frase[i])
                    labels.append(phrase_list.label[i])
            else:
                phrases.append(phrase_list.frase[i])
                labels.append(phrase_list.label[i])
    output = [(i,j) for i,j in zip(phrases,labels)]
    output = sorted(output, key=lambda x: main_text.index(x[0]))
    aspect = []
    if text_len != 'all':
        for i,j in output:
            if len(i.split()) == int(text_len):
                print(i+' -> '+j)
                aspect.append(j)
    else:
        for i,j in output:
            print(i+' -> '+j)
            aspect.append(j)
    print()
    print("Aspek    Kebutuhan    :    ",",    ".join(sorted(set(aspect),
key=aspect.index)))

    print()
```

Gambar 4.19 Kode program identifikasi informasi

Gambar 4.20 menunjukkan hasil keluaran dari identifikasi informasi berdasarkan frasa_list.csv. Keterangan “Kebutuhan tertinggi” menampilkan frasa yang berisikan kebutuhan tertinggi pada hasil tes EPPS sedangkan keterangan “Kebutuhan terendah” menampilkan frasa yang berisikan kebutuhan terendah calon pendaftar. Di dalamnya terdapat informasi mengenai frasa dan label yang tertulis dalam kalimat kesimpulan hasil tes EPPS tersebut. “Aspek kebutuhan” yang terdapat pada hasil keluaran ini menunjukkan secara singkat aspek kebutuhan berdasarkan dari indeks frasa dan labelnya yang telah ditentukan. Hasil dapat dilihat pada bagian lampiran.

```
1831150236.pdf
=======Kebutuhan Tertinggi=======
berempati -> INT
bersimpati -> INT
yakin dengan kemampuan -> EXH
menonjolkan diri -> DOM
ketua kelompok -> DOM
mengatur orang -> DOM
mempengaruhi orang -> DOM

Aspek Kebutuhan :  INT, EXH, DOM

=======Kebutuhan Terendah=======
patuh terhadap aturan -> DEF
patuh -> DEF
aturan -> DEF
mengikuti pendapat -> HET
berperilaku agresif -> AGG
agresif -> AGG
mencapai prestasi -> ACH
mencapai -> ACH
prestasi -> ACH
memperoleh sesuatu -> ACH
sesuatu yang tetap -> CHG
rutin -> CHG
ketahanan -> END
kegigihan -> END
ketekunan -> END

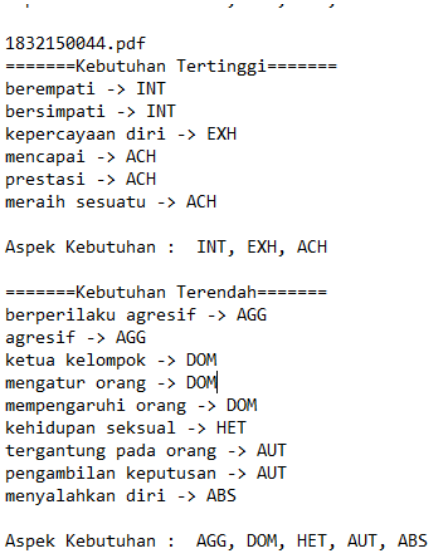
Aspek Kebutuhan :  DEF, HET, AGG, ACH, CHG, END


1831150091.pdf
=======Kebutuhan Tertinggi=======
mandiri -> AUT
tidak tergantung -> AUT
tergantung pada orang -> AUT
pendirian/pendapat -> AUT
menaruh perhatian -> AFF
bergaul -> AFF
harmonis -> AFF
menolong -> NUR
merawat orang -> NUR
benda -> NUR

Aspek Kebutuhan :  AUT, AFF, NUR

=======Kebutuhan Terendah=======
ketahanan -> END
kegigihan -> END
ketekunan -> END
mencapai prestasi -> ACH
mencapai -> ACH
prestasi -> ACH
memperoleh sesuatu -> ACH
patuh terhadap aturan -> DEF
patuh -> DEF
aturan -> DEF
mengikuti pendapat -> HET
keteraturan -> ORD
sistematis -> ORD
kehidupan seksual -> HET

Aspek Kebutuhan :  END, ACH, DEF, HET, ORD
```



Gambar 4.20 Contoh hasil proses identifikasi informasi

4.1.6 Pencarian Dokumen

Setelah informasi berhasil ditemukan, selanjutnya tahapan ini dilakukan untuk menemukan dokumen dari label yang ingin dicari yang mengandung elemen informasi tersebut. Gambar 4.21 menunjukkan implementasi kode program untuk pencarian dokumen berdasarkan kebutuhan yang telah diinginkan. Kode program lebih lengkap dapat dilihat pada bagian lampiran.

```
# generate positive label
     phrases = []
     labels = []
     for i in range(len(phrase_list.frase)):
         if ' '+phrase_list.frase[i]+' ' in positive_text:
             if reducted:
                if   bool([string   for   string   in   phrases   if
phrase_list.frase[i] in string]) == False:
                     phrases.append(phrase_list.frase[i])
                     labels.append(phrase_list.label[i])
             else:
                 phrases.append(phrase_list.frase[i])
                 labels.append(phrase_list.label[i])
     if set(labels)&set(symptomps) == set(symptomps):
        positive_doc_list.append(file_)

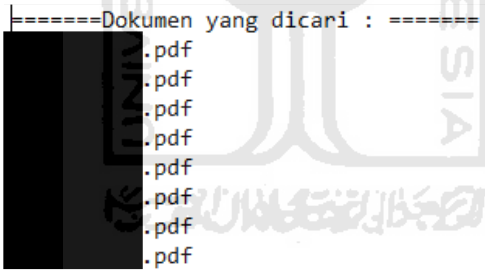
print('=======Parameter(TS-S)=======')
    for i in positive_doc_list:
        print(i)
```

Gambar 4.21 Kode program pencarian dokumen

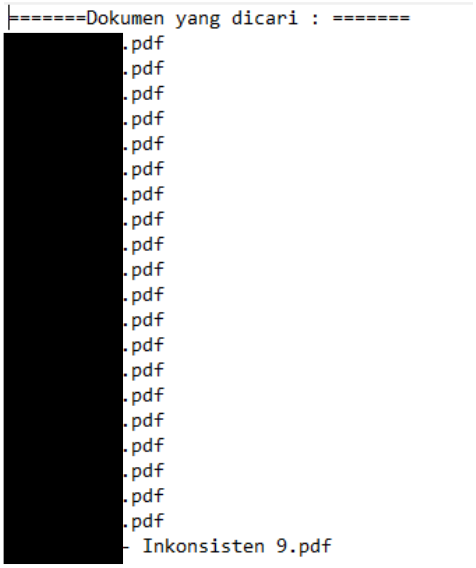
```
=======Dokumen yang dicari : =======
1831150091.pdf
1831150239.pdf
1832150031.pdf
1832150046.pdf
1831150104.pdf
1831150115.pdf
1832150084.pdf
1832150073.pdf
1832150027.pdf
1831150207.pdf
1831150214.pdf
1831150002.pdf
1831150202.pdf
1831150230.pdf
1831150177.pdf
1832150022.pdf
1832150069.pdf
1831150020.pdf
1832150091.pdf
1831150108.pdf
1831150090- Inkonsisten 9.pdf
```

Gambar 4.22 dan

Gambar 4.23 menunjukkan hasil implementasi kode program untuk mencari dokumen jika label yang diinginkan oleh pakar adalah “ACH” dan “DEF” yang dapat tersimpan dalam bentuk file txt.



Gambar 4.22 Hasil implementasi file berdasarkan aspek kebutuhan “ACH”



Gambar 4.23 Hasil implementasi file berdasarkan aspek kebutuhan “NUR”

4.1.7 Hasil Pengujian dan Evaluasi

a. Pengujian

Pada pembahasan ini akan menunjukkan hasil pengujian menggunakan data uji berdasarkan data yang sudah dilatih sebelumnya untuk mendapatkan pencarian dokumen yang diinginkan oleh pihak penyelenggara maupun pakar. Data uji berisikan 14 dokumen tes EPPS calon pendaftar. Tabel 4.1 menunjukkan kolom “data latih” sebagai data yang ingin dilatih untuk mendapatkan label yang sesuai dengan frasa dalam dokumen, kolom “hasil penemuan informasi” sebagai hasil setelah informasi berhasil ditemukan, kolom “hasil pencarian dokumen” untuk menampilkan dokumen yang dicari berdasarkan informasi yang ditemukan dan kolom “ketepatan” menyatakan apakah dokumen dan informasi di dalamnya sesuai dengan dokumen aslinya.

Tabel 4.1 Hasil pengujian pada data uji

Data Uji	Hasil Penemuan Informasi	Hasil Pencarian Dokumen	Ketepatan
Uraian Hasil pemeriksaan psikologis menunjukkan bahwa saudar[redacted]H memiliki kebutuhan yang tinggi untuk mandiri, dan tidak tergantung pada orang lain dalam hal memilih suatu pendirian/pendapat. Selain itu, saudara[redacted] memiliki keinginan yang kuat untuk mendominasi, memerintah, dan mempengaruhi orang lain serta memiliki kebutuhan yang tinggi untuk mencapai suatu prestasi atau memiliki dorongan meraih sesuatu dalam hidup. Namun, saudara[redacted] kurang yakin dengan kemampuan diri dan tidak suka menonjolkan diri dalam kelompoknya. Selain itu, saudara[redacted] tidak terlalu menyukai perilaku menolong dan merawat orang/benda serta kurang memiliki keinginan untuk diperhatikan dan dimanjakan oleh orang lain, kurang tertarik untuk memberikan perhatian pada orang lain dan kurang memiliki keinginan berinteraksi dengan lingkungan sosial, dan kurang memiliki kepekaan pada perasaan orang lain dan sulit memahami orang lain .	[redacted] =======Kebutuhan Tertinggi======= mandiri -> AUT tidak tergantung -> AUT tergantung pada orang -> AUT pendirian/pendapat -> AUT mendominasi -> DOM memerintah -> DOM mempengaruhi orang -> DOM mencapai -> ACH prestasi -> ACH meraih sesuatu -> ACH Aspek Kebutuhan : AUT, DOM, ACH =======Kebutuhan Terendah======= yakin dengan kemampuan -> EXH menonjolkan diri -> DOM menolong -> NUR merawat orang/benda -> NUR diperhatikan -> SUC dimanjakan -> SUC memberikan perhatian -> AFF berinteraksi -> AFF lingkungan sosial -> AFF kepekaan -> INT memahami orang -> INT Aspek Kebutuhan : EXH, DOM, NUR, SUC, AFF, INT	Ada dalam dokumen	Sesuai

b. Evaluasi

Tahap selanjutnya adalah melakukan uji dan evaluasi pada data *test* dengan data asli pada sistem untuk mencari kemunculan nilai derajat yang mirip. Terdapat 14 data uji yang telah di-*split* sebelumnya. Implementasi kode program proses menghitung dari nilai *precision, recall* dan *f-measure* dapat dilihat pada Gambar 4.24 di bawah ini. Kode program yang lebih lengkap dapat dilihat pada bagian lampiran.

```
            csv = pd.read_csv(folder+'//'+file_.strip('pdf')+'csv')
            real = get_txt_from_data(csv)
            try:
                precision   =   len(set([i[0]   for   i   in   real])   &
set(phrases))/len(phrases)
            except ZeroDivisionError:
                precision = 0
            try:
                recall   =   len(set([i[0]   for   i   in   real])   &
set(phrases))/len(real)
            except ZeroDivisionError:
                recall = 0
            try:
                fm = 2*precision*recall/(precision+recall)
            except ZeroDivisionError:
                fm = 0

            precisions.append(precision)
            recalls.append(recall)
            fms.append(fm)
```

Gambar 4.24 Kode program uji dan evaluasi model

Selanjutnya menghitung evaluasi performa dari proses yang telah dilakukan dan menampilkan nilai dari *precision, recall,* dan *f-measure.* Perhitungan akurasi tidak dijabarkan hasilnya karena dalam pencarian, hasil akurasi mendekati 100% atau hasil dari "*True Negative*" yang tak terhingga nilainya kenyataannya hasilnya tidak seperti itu.

Tabel 4.2 menunjukkan hasil dari nilai *precision, recall,* dan *f-measure* setiap dokumen bentuk frasa *all* (keseluruhan kata).

Tabel 4.2 Hasil pengujian model tipe data frasa

Dokumen	*Precision*	*Recall*	*F-Measure*
236.pdf	0,77	0,94	0,85
091.pdf	0,80	0,94	0,86
188.pdf	0,82	0,92	0,86
031.pdf	0,85	0,94	0,89
049.pdf	0,86	0,93	0,89
018.pdf	0,87	0,93	0,90
115.pdf	0,87	0,92	0,89
073.pdf	0,88	0,92	0,89

207.pdf	0,89	0,93	0,91
099.pdf	0,89	0,92	0,91
038.pdf	0,90	0,92	0,91
145.pdf	0,89	0,92	0,91
057.pdf	0,89	0,93	0,91
020.pdf	0,89	0,93	0,91

Berdasarkan dokumen-dokumen yang terdapat pada tabel diatas, diperoleh hasil akhir dari pengujian model dengan menghitung nilai rata-rata tiap dokumen. Pada pengujian di atas tidak terdapat akurasi karena akurasi lebih tepat digunakan untuk distribusi kelas sedangkan *f-measure*, metrik yang lebih baik ketika ada ketidakseimbangan dalam mengevaluasi model. Contoh yang terjadi seperti pada TN terdapat kata yang tidak perlu di cari dan seharusnya tidak di cari tetap dihitung melalui rumus akurasi yang mengakibatkan hasil tersebut mendekati 100% yang belum tentu data mendekati aslinya. Tabel 4.3 menunjukkan hasil rata-rata dari perhitungan evaluasi *precision*, *recall* dan *f-measure* dari setiap data uji.

Tabel 4.3 Hasil rata-rata evaluasi

Precision	*Recall*	*F-Meaasure*
0,89	0,93	0,91

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Setelah melalui tahapan-tahapan awal sampai akhir pada penelitian ini, dapat ditarik kesimpulan sebagai berikut:

a. Metode pencocokan informasi melalui proses *indexing* dapat diterapkan dalam menemukan informasi terkait aspek kebutuhan yang di cari pada tes EPPS calon pendaftar dengan baik dan terstruktur.
b. Dapat melakukan pencarian informasi dengan merujuk informasi yang telah terindentifikasi bersama frasa_list.
c. Perhitungan *f-measure* didapatkan sebesar 0,91 dari *precision* sebesar 0,89 dan *recall* sebesar 0,93.

5.2 Saran

Berdasarkan hasil penelitian yang dicapai, penelitian ini masih jauh dari sempurna karena masih terdapat kesalahan dalam mendeteksi kata pada tipe bentuk frasa. Oleh sebab itu, peneliti berharap penelitian ini bisa dikembangkan oleh peneliti-peneliti yang selanjutnya dengan beberapa saran dari peneliti seperti:

a. Menambah jumlah *dataset* dan daftar frasa yang selalu *update* agar mendapatkan hasil yang lebih baik.
b. Pengembangan dengan melakukan *stopword removal* untuk mendapatkan daftar frasa yang lebih baik.
c. Pelabelan manual pada hasil tes EPPS harus dilakukan oleh pakar.
d. Melakukan pengembangan dengan melakukan pengecekan dan pengoreksian kata kunci pada hasil konversi *file* PDF.

DAFTAR PUSTAKA

Ahmad, A. (2017). Mengenal Artificial Intelligence, Machine Learning, Neural Network, dan Deep Learning. *Jurnal Teknologi Indonesia*, (October), 3.

Basuki, S., Marthasari, G. I., & Arifandi, D. (2018). Penelitian Berbahasa Indonesia Berbasis Fitur. *Seminar Nasional Teknologi Dan Rekayasa (SENTRA)*, 15–20.

Boubekeur, & Wassila. (2013). *CONCEPT-BASED INDEXING IN TEXT INFORMATION RETRIEVAL*. *5*(1).

Broder, A. Z., Eiron, N., Fontoura, M., Herscovici, M., Lempel, R., McPherson, J., … Shekita, E. (2006). Indexing shared content in information retrieval systems. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *3896 LNCS*, 313–330. https://doi.org/10.1007/11687238_21

Cohen-swerdlik. (2010). Psychological:Testing and Assessment: An Introduction to Tests and Measurement. *McGraw-Hill*, (2), 423–424.

Dang, S., & Ahmad, P. H. (2014). Text Mining : Techniques and its Application. *International Journal of Engineering & Technology Innovations*, *1*(4), 22–25.

Darujati, C. (2016). Pemanfaatan Teknik Supervised Untuk Klasifikasi Teks Bahasa. *Jurnal Link*, *16*(February 2012), 8.

Eroglu, Ç., Allen, N. J., Susman, M. W., Rourke, N. A. O., Young, C., Özkan, E., … Barres, A. (2018). *Intelligent Natural Language Processing: Trends and Applications*. *740*(2), 380–392. https://doi.org/10.1007/978-3-319-67056-0

Ferrara, R. (1996). The Edwards Personal Preference Schedule as a Predictor of Success in a Collegiate Professional Pilot Training Program. *Journal of Aviation/Aerospace Education & Research*, *6*(2). https://doi.org/10.15394/jaaer.1996.1176

Haggard, L., Peterson-, S., Rhodewalt, F., Charles, P., Larsen, R. M., & Matarazzo, J. D. (1983). *REFERENCES Psychological Test Usage Patterns in Five Professional S e t t i n g s Kansas City*. 857–861.

Herdianto, A. (2010). *Pencarian Ayat-Ayat Al-Qur'an Berdasarkan Konten Menggunakan Text Mining Berbasis Aplikasi Desktop*. 1–14.

Heriyanto. (2015). *Pencarian Dokumen Teks Arsip Surat*. *2015*(November), 173–177.

Hidayat, W. (2013). Indexing and Retrieval Engine untuk Dokumen Berbahasa Indonesia dengan Menggunakan Inverted Index. *Seminar Nasional Informatika Dan Aplikasinya (SNIA) 2015*, (October). Retrieved from

https://www.researchgate.net/publication/284492748

Ismaya, A. (2014). Algoritma Ekstraksi Informasi Berbasis Aturan. *Jnteti*, *03*(04), 242–247. https://doi.org/10.22146/jnteti.v3i4.110

Kaiser, K., Miksch, S., Kaiser, K., & Miksch, S. (2005). *InformationExtraction_aSurvey*. (May).

Kuncoro. (2012). *EVALUASI KUALITAS TES PSIKOLOGI KEPRIBADIAN I*. *3*(4), 58–73. Retrieved from https://www.rics.org/south-asia/upholding-professional-standards/standards-of-conduct/ethics/

Manning, C. D., Raghavan, P., Schutze, H., Manning, C. D., Raghavan, P., & Schutze, H. (2004). Introduction to Modern Information Retrieval (2nd edition). *Library Review*, *53*(9), 462–463. https://doi.org/10.1108/00242530410565256

Muhammad Wahyu Kuncoro. (2012). *EVALUASI KUALITAS TES PSIKOLOGI KEPRIBADIAN I*. *3*(4), 58–73. Retrieved from https://www.rics.org/south-asia/upholding-professional-standards/standards-of-conduct/ethics/

Muludi, K., Kuspriyanto, Santoso, O. S., & Widyantoro, D. H. (2008). *Evaluasi Kinerja Algoritma Support Vector Machine dalam Ekstraksi Informasi Korpus Berbahasa Indonesia*.

Nurussakinah Daulay. (2014). Implementasi Tes Psikologi Dalam Bidang Pendidikan. *Jurnal Tarbiyah*, *21*(2), 402–421.

Piedmont, R. L., & Costa, P. (1992). *An Assessment of the Edwards Personal Preference Schedule From the Perspective of the Five-Factor Model*. (March). https://doi.org/10.1207/s15327752jpa5801

Sasmita, F., & Kinanti, K. P. (2017). *Ekstraksi Informasi Dokumen Karya Tulis Ilmiah Menggunakan Algoritma Learning Vector Quantization*.

Schultz, D. P., & Schultz, S. E. (2017). *Theories of Personality*.

Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing and Management*, *45*(4), 427–437. https://doi.org/10.1016/j.ipm.2009.03.002

Tang, J., Hong, M., Zhang, D., Liang, B., & Li, J. (2007). Information extraction: Methodologies and applications. *Emerging Technologies of Text Mining: Techniques and Applications*, 1–33. https://doi.org/10.4018/978-1-59904-373-9.ch001

Trisna, I. N. P., & Nurwidyantoro, A. (2020). Single document keywords extraction in Bahasa Indonesia using phrase chunking. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, *18*(4), 1917. https://doi.org/10.12928/telkomnika.v18i4.14389

Waluya, B. (1970). *Pengertian Dan Ruang Lingkup Psikologi Pendidikan*. Retrieved from https://s3.amazonaws.com/academia.edu.documents/51288090/Konsep_Dasar_Psikologi.pdf

Yudhana, A., Djayati, A. D., Sunardi, Yudhana, A., Djayati, A. D., & Sunardi. (2017). Sistem Deteksi Plagiarisme Dokumen Karya Ilmiah dengan Algoritma Pencocokan Pola. *Jurti*, Vol. 1, pp. 178–187.

LAMPIRAN

Lampiran script konversi dokumen dan *preprocessing*

```
    import os
    import pandas as pd
    from       pdfminer.pdfinterp       import       PDFResourceManager,
PDFPageInterpreter
    from pdfminer.converter import TextConverter
    from pdfminer.layout import LAParams
    from pdfminer.pdfpage import PDFPage
    from io import StringIO
    import sys
    from nltk import word_tokenize

    def convert_pdf_to_txt(path):
        rsrcmgr = PDFResourceManager()
        retstr = StringIO()
        laparams = LAParams()
        device = TextConverter(rsrcmgr, retstr, laparams=laparams)
        fp = open(path, 'rb')
        interpreter = PDFPageInterpreter(rsrcmgr, device)
        password = ""
        maxpages = 0
        caching = True
        pagenos=set()

        for page in PDFPage.get_pages(fp, pagenos, maxpages=maxpages,
password=password,caching=caching, check_extractable=True):
            interpreter.process_page(page)

        text = retstr.getvalue()

        fp.close()
        device.close()
        retstr.close()
        return text
```

```
    def get_main_text(text):
        text_list = text.split('\n')
        text_list                                                    =
text_list[text_list.index('Uraian'):text_list.index('Pemeriksa,')][1:]
        return_text = ' '.join(text_list)
        return return_text.lower()
    #%%
    if __name__ == '__main__':
        counter = 1
        for pdf in os.listdir(sys.argv[1]):
            print('Processed ',counter,' pdf')
            text = convert_pdf_to_txt('fk_epps//'+pdf)
            text = get_main_text(text)
            list_ = word_tokenize(text)
            table= pd.DataFrame(columns=['kata','label'])
            table['kata'] = list_

table.to_csv(sys.argv[1]+'//'+pdf.strip('.pdf')+'.csv',index=None)
            counter += 1
    def get_main_text(text):
        text_list = text.split('\n')
        text_list                                                    =
text_list[text_list.index('Uraian'):text_list.index('Pemeriksa,')][1:]
        return_text = ' '.join(text_list)
        return return_text.lower()
    #%%
    if __name__ == '__main__':
        counter = 1
        for pdf in os.listdir(sys.argv[1]):
            print('Processed ',counter,' pdf')
            text = convert_pdf_to_txt('fk_epps//'+pdf)
            text = get_main_text(text)
            list_ = word_tokenize(text)
            table= pd.DataFrame(columns=['kata','label'])
            table['kata'] = list_

table.to_csv(sys.argv[1]+'//'+pdf.strip('.pdf')+'.csv',index=None)
            counter += 1
```

Lampiran script *split* data

```
    import os
    import shutil
    import random
    import sys

    if __name__=="__main__":
        if 'test' not in os.listdir():
            os.makedirs('test')
        if 'train' not in os.listdir():
            os.makedirs('train')
        if bool(os.listdir('test')) and bool(os.listdir('test')):
            print('Both folder are already filled')
            exit()
        try:
            folder = sys.argv[1]
        except IndexError:
            print('Folder argument is missing')
            exit()
        try:
            test_size = float(sys.argv[2])
        except (IndexError,ValueError):
            print('Test size is set default of 20% of data')
            test_size = 0.2

        files = [i for i in os.listdir(folder) if '.csv' in i]
        test_data = random.sample(files,k=int(len(files)*test_size))
        train_data = [i for i in files if i not in test_data]

        if not bool(os.listdir('test')):
            for file_ in test_data:
                shutil.copyfile(folder+'//'+file_,'test//'+file_)

shutil.copyfile(folder+'//'+file_.strip('csv')+'pdf','test//'+file_.str
ip('csv')+'pdf')
        else:
            print('Test folder is already filled with data')
```

```
   if not bool(os.listdir('train')):
           for file_ in train_data:
               shutil.copyfile(folder+'//'+file_,'train//'+file_)

shutil.copyfile(folder+'//'+file_.strip('csv')+'pdf','train//'+file_.st
rip('csv')+'pdf')
       else:
           print('Train folder is already filled with data')
       if not bool(os.listdir('train')):
           for file_ in train_data:
               shutil.copyfile(folder+'//'+file_,'train//'+file_)

shutil.copyfile(folder+'//'+file_.strip('csv')+'pdf','train//'+file_.st
rip('csv')+'pdf')
       else:
           print('Train folder is already filled with data')
```

Lampiran script pengindeksan daftar frasa

```
    import sys
    import pandas as pd
    import numpy as np
    import os
    #%%
    def get_txt_from_data(data):
        phrase_list = []
        start = 0
        for i in range(len(data.label))[1:]:
            if data.label[i] != data.label[i-1]:
                if  data.label[start]  !=  'O'  and  data.label[start]  !=
 'OO':
                    phrase = ' '.join(data.kata[start:i])
                    phrase = phrase.replace('.','').replace(',','')
                    label = data.label[start]
                    phrase_list.append((phrase.lower(),label))
                start = i
        return phrase_list
```

```
if __name__ == "__main__":
    try:
        sys.argv[1]
    except IndexError:
        print('Training folder argument is missing')
        exit()
    files = [i for i in os.listdir(sys.argv[1]) if '.csv' in i]
    text = []
    for file_ in files:
        data = pd.read_csv(sys.argv[1]+'//'+file_)
        text += get_txt_from_data(data)
    text = list(set(text))
    removed_list = [('diperhatikan','AFF'),
                    ('ketua kelompok','DEF'),
                    ('memberikan perhatian','INT'),
                    ('menaruh perhatian', 'INT'),
                    ('mengatur','DEF'),
                    ('menjaga barang','ORD'),
                    ('rutin','NUR'),
                    ('teratur','ORD'),
                    ('tetap','NUR'),
                    ('aktivitas', 'NUR'),
                    ('sistematis', 'END'),
                    ('berinteraksi', 'NUR'),
                    ('dan', ''),
                    ('hal', ''),
                    ('itu', ''),
                    ]
    for word in removed_list:
        try:
            text.remove(word)
        except ValueError:
            pass

text = sorted(set(text),key=lambda x:len(x[0]),reverse=True)
    phrase_list = pd.DataFrame(text,columns=['frase','label'])
    phrase_list.to_csv('frasa_list.csv',index=None)
```

Lampiran script penemuan informasi

```
    import sys
    import pandas as pd
    from       pdfminer.pdfinterp       import       PDFResourceManager,
PDFPageInterpreter
    from pdfminer.converter import TextConverter
    from pdfminer.layout import LAParams
    from pdfminer.pdfpage import PDFPage
    from io import StringIO
    import re

    def convert_pdf_to_txt(path):
        rsrcmgr = PDFResourceManager()
        retstr = StringIO()
        laparams = LAParams()
        device = TextConverter(rsrcmgr, retstr, laparams=laparams)
        fp = open(path, 'rb')
        interpreter = PDFPageInterpreter(rsrcmgr, device)
        password = ""
        maxpages = 0
        caching = True
        pagenos=set()

        for page in PDFPage.get_pages(fp, pagenos, maxpages=maxpages,
password=password,caching=caching, check_extractable=True):
            interpreter.process_page(page)

        text = retstr.getvalue()
```

```
    fp.close()
        device.close()
        retstr.close()
        return text

    def get_main_text(text):
        text_list = text.split('\n')
        text_list                                                    =
text_list[text_list.index('Uraian'):text_list.index('Pemeriksa,')][1:]
        return_text = ' '.join(text_list)
        return return_text

    def get_txt_from_data(data):
        phrase_list = []
        start = 0
        for i in range(len(data.label))[1:]:
            if data.label[i] != data.label[i-1]:
                if  data.label[start]  !=  'O'  and  data.label[start]  !=
'OO':
                    phrase = ' '.join(data.kata[start:i])
                    phrase = phrase.replace('.','').replace(',','')
                    label = data.label[start]
                    phrase_list.append((phrase.lower(),label))
                start = i
        return phrase_list

    if __name__=='__main__':
        # arg 1 -> file name
        # arg 2 -> phrase length
        try:
            phrase_list = pd.read_csv('frasa_list.csv')
        except FileNotFoundError:
            print('phrase_list.csv is not available')
            exit()
```

```
    try:
            text_len = sys.argv[2]
        except IndexError:
            print('N-gram argument is missing')
            exit()
        try:
            reducted = bool(sys.argv[3])
        except IndexError:
            reducted = False
        try:
            text = convert_pdf_to_txt(file_)
        except FileNotFoundError:
            print(file_+' is not available')
            exit()

        main_text = get_main_text(text)
        main_text = ' '+re.sub('(?<! )(?=[.,!?()])|(?<=[.,!?()])(?! )',
r' ', main_text)+' '
        positive_text = main_text.split('Namun ,')[0]
        negative_text = main_text.split('Namun ,')[1]

        print('=======Parameter(TS-S)=======')
        phrases = []
        labels = []
        for i in range(len(phrase_list.frase)):
            if ' '+phrase_list.frase[i]+' ' in positive_text:
                if reducted:
                    if bool([string for string in phrases if
phrase_list.frase[i] in string]) == False:
                        phrases.append(phrase_list.frase[i])
                        labels.append(phrase_list.label[i])
                else:
                    phrases.append(phrase_list.frase[i])
                    labels.append(phrase_list.label[i])
        output = [(i,j) for i,j in zip(phrases,labels)]
        output = sorted(output, key=lambda x: main_text.index(x[0]))
try:
            file_ = sys.argv[1]
        except IndexError:
```

```
    aspect = []
        if text_len != 'all':
            for i,j in output:
                if len(i.split()) == int(text_len):
                    print(i+' -> '+j)
                    aspect.append(j)
        else:
            for i,j in output:
                print(i+' -> '+j)
                aspect.append(j)
        print()
        print("Aspek Kebutuhan : ","  ".join(sorted(set(aspect),
key=aspect.index)))

        print()
        print('=======Parameter(S-KS)=======')
        phrases = []
        labels = []
        for i in range(len(phrase_list.frase)):
            if ' '+phrase_list.frase[i]+' ' in negative_text:
                if reducted:
                    if bool([string for string in phrases if
phrase_list.frase[i] in string]) == False:
                        phrases.append(phrase_list.frase[i])
                        labels.append(phrase_list.label[i])
                else:
                    phrases.append(phrase_list.frase[i])
                    labels.append(phrase_list.label[i])
        output = [(i,j) for i,j in zip(phrases,labels)]
        output = sorted(output, key=lambda x: main_text.index(x[0]))
        aspect = []
        if text_len != 'all':
            for i,j in output:
                if len(i.split()) == int(text_len):
                    print(i+' -> '+j)
                    aspect.append(j)
```

```
    else:
            for i,j in output:
                print(i+' -> '+j)
                aspect.append(j)
        print()
        print("Aspek   Kebutuhan   :   ",",   ".join(sorted(set(aspect),
key=aspect.index)))    else:
            for i,j in output:
                print(i+' -> '+j)
                aspect.append(j)
        print()
        print("Aspek   Kebutuhan   :   ",",   ".join(sorted(set(aspect),
key=aspect.index)))
```

Lampiran script pencarian dokumen

```
    import pandas as pd
    import os
    import re
    import sys

    from       pdfminer.pdfinterp       import      PDFResourceManager,
PDFPageInterpreter
    from pdfminer.converter import TextConverter
    from pdfminer.layout import LAParams
    from pdfminer.pdfpage import PDFPage
    from io import StringIO

    def convert_pdf_to_txt(path):
        rsrcmgr = PDFResourceManager()
        retstr = StringIO()
        laparams = LAParams()
        device = TextConverter(rsrcmgr, retstr, laparams=laparams)
        fp = open(path, 'rb')
        interpreter = PDFPageInterpreter(rsrcmgr, device)
        password = ""
        maxpages = 0
        caching = True
        pagenos=set()
```

```
        for page in PDFPage.get_pages(fp, pagenos, maxpages=maxpages,
password=password,caching=caching, check_extractable=True):
            interpreter.process_page(page)

        text = retstr.getvalue()

        fp.close()
        device.close()
        retstr.close()
        return text

    def get_main_text(text):
        text_list = text.split('\n')
        text_list                                                    =
text_list[text_list.index('Uraian'):text_list.index('Pemeriksa,')][1:]
        return_text = ' '.join(text_list)
        return return_text.lower()
    #%%
    if __name__=='__main__':
        # arg 1 -> folder name
        # arg 2 -> phrase length
        try:
            phrase_list = pd.read_csv('frasa_list.csv')
        except FileNotFoundError:
            print('phrase_list.csv is not available')
            exit()
        try:
            folder_name = sys.argv[1]
        except IndexError:
            print('Folder argument is missing')
            exit()
    try:
            text_len = sys.argv[2]
        except IndexError:
            print('N-gram argument is missing')
            exit()
        if sys.argv[3] == 'reducted':
            reducted = True
            symptomps = sys.argv[4:]
        else:
            reducted = False
            symptomps = sys.argv[3:]
```

```
      positive_doc_list = []
        #negative_doc_list = []

        phrase_list = pd.read_csv('frasa_list.csv')
        files = [i for i in os.listdir(folder_name) if '.pdf' in i]
        for file_ in files:
            text = convert_pdf_to_txt(folder_name+'//'+file_)
            main_text = get_main_text(text)
            main_text = ' '+re.sub('(?<! )(?=[.,!?()])|(?<=[.,!?()])(?!
)', r' ', main_text)+' '
            positive_text = main_text.split('namun ,')[0]
        #    negative_text = main_text.split('namun ,')[1]

            # generate positive label
            phrases = []
            labels = []
            for i in range(len(phrase_list.frase)):
                if ' '+phrase_list.frase[i]+' ' in positive_text:
                    if reducted:
                        if bool([string for string in phrases if
phrase_list.frase[i] in string]) == False:
                            phrases.append(phrase_list.frase[i])
                            labels.append(phrase_list.label[i])
                    else:
                        phrases.append(phrase_list.frase[i])
                        labels.append(phrase_list.label[i])
            if set(labels)&set(symptomps) == set(symptomps):
                positive_doc_list.append(file_)
             else:
                        phrases.append(phrase_list.frase[i])
                        labels.append(phrase_list.label[i])
            if set(labels)&set(symptomps) == set(symptomps):
                positive_doc_list.append(file_)

    print('=======Calon pendaftar dengan hasil baik pada aspek berikut :
=======')
        for i in positive_doc_list:
            print(i)
```

Lampiran script pengujian

```
    import os
    import sys
    import pandas as pd
    from       pdfminer.pdfinterp       import       PDFResourceManager,
PDFPageInterpreter
    from pdfminer.converter import TextConverter
    from pdfminer.layout import LAParams
    from pdfminer.pdfpage import PDFPage
    from io import StringIO
    import re
    from numpy import mean

    def convert_pdf_to_txt(path):
        rsrcmgr = PDFResourceManager()
        retstr = StringIO()
        laparams = LAParams()
        device = TextConverter(rsrcmgr, retstr, laparams=laparams)
        fp = open(path, 'rb')
        interpreter = PDFPageInterpreter(rsrcmgr, device)
        password = ""
        maxpages = 0
        caching = True
        pagenos=set()

        for page in PDFPage.get_pages(fp, pagenos, maxpages=maxpages,
password=password,caching=caching, check_extractable=True):
            interpreter.process_page(page)

        text = retstr.getvalue()

        fp.close()
        device.close()
        retstr.close()
        return text
```

```
    def get_main_text(text):
        text_list = text.split('\n')
        text_list                                                    =
text_list[text_list.index('Uraian'):text_list.index('Pemeriksa,')][1:]
        return_text = ' '.join(text_list)
        return return_text

    def get_txt_from_data(data):
        phrase_list = []
        start = 0
        for i in range(len(data.label))[1:]:
            if data.label[i] != data.label[i-1]:
                if  data.label[start]  !=  'O'  and  data.label[start]  !=
'OO':
                    phrase = ' '.join(data.kata[start:i])
                    phrase = phrase.replace('.','').replace(',','')
                    label = data.label[start]
                    phrase_list.append((phrase.lower(),label))
                start = i
        return phrase_list
    def main():
        try:
            phrase_list = pd.read_csv('frasa_list.csv')
        except FileNotFoundError:
            print('phrase_list.csv is not available')
            exit()
        try:
            folder = sys.argv[1]
        except IndexError:
            print('Folder argument is missing')
            exit()
        try:
            text_len = sys.argv[2]
        except IndexError:
            print('N-gram argument is missing')
            exit()
```

```
        try:
            reducted = bool(sys.argv[3])
        except IndexError:
            reducted = False

        precisions = []
        recalls = []
        fms = []
        phrase_list = pd.read_csv('frasa_list.csv')
        files = [i for i in os.listdir(folder) if '.pdf' in i]
        for file_ in files:
            text = convert_pdf_to_txt(folder+'//'+file_)
            main_text = get_main_text(text)
            main_text = ' '+re.sub('(?<! )(?=[.,!?()])|(?<=[.,!?()])(?!
)', r' ', main_text)+' '
        phrases = []
        labels = []
        for i in range(len(phrase_list.frase)):
             if ' '+phrase_list.frase[i]+' ' in main_text:
                 if reducted:
                    if   bool([string   for   string   in   phrases   if
phrase_list.frase[i] in string]) == False:
                            phrases.append(phrase_list.frase[i])
                            labels.append(phrase_list.label[i])
                    else:
                        phrases.append(phrase_list.frase[i])
                        labels.append(phrase_list.label[i])
           output = [(i,j) for i,j in zip(phrases,labels)]
           output = sorted(output, key=lambda x: main_text.index(x[0]))
           phrases = []
           if text_len != 'all':
               for i,j in output:
                   if len(i.split()) == int(text_len):
                       phrases.append(i)
            else:
               for i,j in output:
                   phrases.append(i)
```

```
            csv = pd.read_csv(folder+'//'+file_.strip('pdf')+'csv')
            real = get_txt_from_data(csv)
            try:
                precision = len(set([i[0] for i in real]) &
set(phrases))/len(phrases)
            except ZeroDivisionError:
                precision = 0
            try:
                recall = len(set([i[0] for i in real]) &
set(phrases))/len(real)
            except ZeroDivisionError:
                recall = 0
            try:
                fm = 2*precision*recall/(precision+recall)
            except ZeroDivisionError:
                fm = 0
            precisions.append(precision)
            recalls.append(recall)
            fms.append(fm)

            print(file_)
            print('Precision: ',mean(precisions))
            print('Recall: ',mean(recalls))
            print('F-measure: ',mean(fms))
            print()

        print('Precision: ',mean(precisions))
        print('Recall: ',mean(recalls))
        print('F-measure: ',mean(fms))

    if __name__ == '__main__':
        main()
```

Hasil keluaran Pelabelan Manual Dokumen 002-091

15	untuk	O
16	menyesuaikan	DEF
17	diri	DEF
18	dan	O
19	mengikuti	DEF
20	aturan/konvensi	DEF
21	yang	O
22	telah	O
23	ditetapkan	O

110	menyalahkan	ABS
111	diri	ABS
112	sendiri	O

37	dengan	O
38	manusia	O
39	dan	O
40	menaruh	AFF
41	perhatian	AFF
42	pada	O
43	orang	O
44	lain	O
45	serta	O
46	dapat	O
47	bergaul	AFF
48	secara	O
49	harmonis	AFF

109	ketua	DOM
110	kelompok	DOM
111	atau	O
112	yang	O
113	mengatur	DOM
114	orang	DOM
115	lain	O
116	serta	O
117	kurang	O
118	memiliki	O
119	kebutuhan	O
120	untuk	O
121	mempengaruhi	DOM
122	orang	DOM

111	kepekaan	INT
112	pada	O
113	perasaan	O
114	orang	O
115	lain	O
116	dan	O
117	sulit	O
118	memahami	INT
119	orang	INT

101	menolong	NUR
102	dan	O
103	merawat	NUR
104	orang/benda	NUR
105	serta	O
106	kurang	O
107	menunjukkan	O
108	ketahanan	END
109	dan	O
110	kegigihan	END
111	dalam	O
112	menghadapi	O
113	tugas	O
114	yang	O
115	membutuhkan	O
116	ketekunan	END

15	untuk	O
16	mandiri	AUT
17	,	O
18	dan	O
19	tidak	AUT
20	tergantung	AUT
21	pada	O
22	orang	O
23	lain	O
24	dalam	O
25	hal	O
26	memilih	O
27	suatu	O
28	pendirian/pendapat	AUT

50	untuk	O
51	berempati	INT
52	,	O
53	merasakan	INT
54	apa	O
55	yang	O
56	orang	O
57	lain	O
58	rasakan	O
59	dan	O
60	bersimpati	INT
61	kepada	O

37	membutuhkan	O
38	ketekunan	END
39	dan	O
40	keuletan	END
41	serta	O
42	memiliki	O
43	kebutuhan	O
44	yang	O
45	tinggi	O
46	untuk	O
47	melakukan	CHG
48	perubahan	CHG

70	anggrila	O
71	sekar	O
72	fadhillah	O
73	kurang	O
74	yakin	EXH
75	dengan	EXH
76	kemampuan	EXH
77	diri	O
78	dan	O
79	tidak	O
80	suka	O
81	menonjolkan	DOM
82	diri	DOM
83	dalam	O
84	kelompoknya	O
85	.	O

79	untuk	O
80	berperilaku	AGG
81	agresif	AGG

	cukup	O
	sistematis	ORD
	saat	O
	mengerjakan	O

112	tergantung	AUT
113	pada	AUT
114	orang	AUT
115	lain	O
116	dalam	O
117	melakukan	O
118	sesuatu	O
119	termasuk	O
120	dalam	O
121	pengambilan	AUT
122	keputusan	AUT

37	mencapai	ACH
38	suatu	O
39	prestasi	ACH
40	atau	O
41	memiliki	O
42	dorongan	O
43	meraih	ACH
44	sesuatu	ACH
45	dalam	O

85	menjadikan	O
86	kehidupan	HET
87	seksual	HET
88	sebagai	O
89	kebutuhan	O

Hasil keluaran identifikasi informasi data uji

```
1831150091.pdf
=======Kebutuhan Tertinggi=======
mandiri -> AUT
tidak tergantung -> AUT
tergantung pada orang -> AUT
pendirian/pendapat -> AUT
menaruh perhatian -> AFF
bergaul -> AFF
harmonis -> AFF
menolong -> NUR
merawat orang -> NUR
benda -> NUR

Aspek Kebutuhan :  AUT, AFF, NUR

=======Kebutuhan Terendah=======
ketahanan -> END
kegigihan -> END
ketekunan -> END
mencapai prestasi -> ACH
mencapai -> ACH
prestasi -> ACH
memperoleh sesuatu -> ACH
patuh terhadap aturan -> DEF
patuh -> DEF
aturan -> DEF
mengikuti pendapat -> HET
keteraturan -> ORD
sistematis -> ORD
kehidupan seksual -> HET

Aspek Kebutuhan :  END, ACH, DEF, HET, ORD
```

```
1831150236.pdf
=======Kebutuhan Tertinggi=======
berempati -> INT
bersimpati -> INT
yakin dengan kemampuan -> EXH
menonjolkan diri -> DOM
ketua kelompok -> DOM
mengatur orang -> DOM
mempengaruhi orang -> DOM

Aspek Kebutuhan :  INT, EXH, DOM

=======Kebutuhan Terendah=======
patuh terhadap aturan -> DEF
patuh -> DEF
aturan -> DEF
mengikuti pendapat -> HET
berperilaku agresif -> AGG
agresif -> AGG
mencapai prestasi -> ACH
mencapai -> ACH
prestasi -> ACH
memperoleh sesuatu -> ACH
sesuatu yang tetap -> CHG
rutin -> CHG
ketahanan -> END
kegigihan -> END
ketekunan -> END

Aspek Kebutuhan :  DEF, HET, AGG, ACH, CHG, END
```

```
1831150188.pdf
=======Kebutuhan Tertinggi=======
kepercayaan diri -> EXH
ketekunan -> END
keuletan -> END
melakukan perubahan -> CHG

Aspek Kebutuhan :  EXH, END, CHG

=======Kebutuhan Terendah=======
diperhatikan -> SUC
dimanjakan -> SUC
kehidupan seksual -> HET
berperilaku agresif -> AGG
agresif -> AGG
patuh terhadap aturan -> DEF
patuh -> DEF
aturan -> DEF
mengikuti pendapat -> HET
keteraturan -> ORD
sistematis -> ORD

Aspek Kebutuhan :  SUC, HET, AGG, DEF, ORD
```

```
1832150031.pdf
=======Kebutuhan Tertinggi=======
menaruh perhatian -> AFF
bergaul -> AFF
harmonis -> AFF
ketekunan -> END
keuletan -> END
menolong -> NUR
merawat orang -> NUR
benda -> NUR

Aspek Kebutuhan :  AFF, END, NUR

=======Kebutuhan Terendah=======
berperilaku agresif -> AGG
agresif -> AGG
ketua kelompok -> DOM
mengatur orang -> DOM
mempengaruhi orang -> DOM
kehidupan seksual -> HET
kepekaan -> INT
memahami orang -> INT
sesuatu yang tetap -> CHG
rutin -> CHG

Aspek Kebutuhan :  AGG, DOM, HET, INT, CHG
```

```
1832150049.pdf
=======Kebutuhan Tertinggi=======
mengekspresikan dirinya -> EXH
kepercayaan diri -> EXH
mendobrak -> AGG
agresif -> AGG
mendominasi -> DOM
memerintah -> DOM
mempengaruhi orang -> DOM

Aspek Kebutuhan :  EXH, AGG, DOM

=======Kebutuhan Terendah=======
tergantung pada orang -> AUT
pengambilan keputusan -> AUT
menolong -> NUR
merawat orang/benda -> NUR
merawat orang/benda -> NUR
memberikan perhatian -> AFF
berinteraksi -> AFF
menyalahkan diri -> ABS
diperhatikan -> SUC
dimanjakan -> SUC

Aspek Kebutuhan :  AUT, NUR, AFF, ABS, SUC
```

```
1832150018.pdf
=======Kebutuhan Tertinggi=======
berempati -> INT
bersimpati -> INT
menyesuaikan diri -> DEF
mengikuti aturan/konvensi -> DEF
menaruh perhatian -> AFF
bergaul -> AFF
harmonis -> AFF

Aspek Kebutuhan :  INT, DEF, AFF

=======Kebutuhan Terendah=======
berperilaku agresif -> AGG
agresif -> AGG
ketua kelompok -> DOM
mengatur orang -> DOM
mempengaruhi orang -> DOM
kehidupan seksual -> HET
diperhatikan -> SUC
dimanjakan -> SUC
sesuatu yang tetap -> CHG
rutin -> CHG

Aspek Kebutuhan :  AGG, DOM, HET, SUC, CHG
```

1831150115.pdf
=======Kebutuhan Tertinggi=======
menolong -> NUR
merawat orang -> NUR
benda -> NUR
melakukan perubahan -> CHG
berempati -> INT
bersimpati -> INT

Aspek Kebutuhan : NUR, CHG, INT

=======Kebutuhan Terendah=======
berperilaku agresif -> AGG
agresif -> AGG
kehidupan seksual -> HET
ketahanan -> END
kegigihan -> END
ketekunan -> END
patuh terhadap aturan -> DEF
patuh -> DEF
aturan -> DEF
mengikuti pendapat -> HET
keteraturan -> ORD
sistematis -> ORD

Aspek Kebutuhan : AGG, HET, END, DEF, ORD

1832150073.pdf
=======Kebutuhan Tertinggi=======
menaruh perhatian -> AFF
bergaul -> AFF
harmonis -> AFF
menolong -> NUR
merawat orang -> NUR
benda -> NUR
ketahanan -> END
kegigihan -> END
ketekunan -> END

Aspek Kebutuhan : AFF, NUR, END

=======Kebutuhan Terendah=======
patuh terhadap aturan -> DEF
patuh -> DEF
aturan -> DEF
mengikuti pendapat -> HET
berperilaku agresif -> AGG
agresif -> AGG
sesuatu yang tetap -> CHG
rutin -> CHG
keteraturan -> ORD
sistematis -> ORD

Aspek Kebutuhan : DEF, HET, AGG, CHG, ORD

1831150207.pdf
=======Kebutuhan Tertinggi=======
menaruh perhatian -> AFF
bergaul -> AFF
harmonis -> AFF
dimanjakan -> SUC
diurus -> SUC
menolong -> NUR
merawat orang -> NUR
benda -> NUR

Aspek Kebutuhan : AFF, SUC, NUR

=======Kebutuhan Terendah=======
ketahanan -> END
kegigihan -> END
ketekunan -> END
patuh -> DEF
aturan -> DEF
mengikuti pendapat -> HET
ketua kelompok -> DOM
mengatur orang -> DOM
mempengaruhi orang -> DOM
sesuatu yang tetap -> CHG
rutin -> CHG
yakin dengan kemampuan -> EXH
menonjolkan diri -> DOM

Aspek Kebutuhan : END, DEF, HET, DOM, CHG, EXH

1831150099.pdf
=======Kebutuhan Tertinggi=======
dimanjakan -> SUC
diurus -> SUC
menaruh perhatian -> AFF
bergaul -> AFF
harmonis -> AFF
mengekspresikan dirinya -> EXH
kepercayaan diri -> EXH

Aspek Kebutuhan : SUC, AFF, EXH

=======Kebutuhan Terendah=======
sesuatu yang tetap -> CHG
rutin -> CHG
berperilaku agresif -> AGG
agresif -> AGG
kehidupan seksual -> HET
keteraturan -> ORD
sistematis -> ORD
ketua kelompok -> DOM
mempengaruhi orang -> DOM

Aspek Kebutuhan : CHG, AGG, HET, ORD, DOM

```
1831150038.pdf
=======Kebutuhan Tertinggi=======
kepercayaan diri -> EXH
dimanjakan -> SUC
diurus -> SUC
mandiri -> AUT
tidak tergantung -> AUT
tergantung pada orang -> AUT
pendirian/pendapat -> AUT

Aspek Kebutuhan :  EXH, SUC, AUT

=======Kebutuhan Terendah=======
patuh terhadap aturan -> DEF
patuh -> DEF
aturan -> DEF
mengikuti pendapat -> HET
ketahanan -> END
kegigihan -> END
ketekunan -> END
keteraturan -> ORD
sistematis -> ORD
menyalahkan diri -> ABS
ketua kelompok -> DOM
mengatur orang -> DOM
mempengaruhi orang -> DOM

Aspek Kebutuhan :  DEF, HET, END, ORD, ABS, DOM
```

```
1831150145.pdf
=======Kebutuhan Tertinggi=======
mencapai -> ACH
prestasi -> ACH
meraih sesuatu -> ACH
dimanjakan -> SUC
diurus -> SUC
tergantung pada orang -> AUT
pengambilan keputusan -> AUT

Aspek Kebutuhan :  ACH, SUC, AUT

=======Kebutuhan Terendah=======
sesuatu yang tetap -> CHG
rutin -> CHG
kehidupan seksual -> HET
berperilaku agresif -> AGG
agresif -> AGG
ketua kelompok -> DOM
mengatur orang -> DOM
mempengaruhi orang -> DOM
menolong -> NUR
merawat orang/benda -> NUR
merawat orang/benda -> NUR

Aspek Kebutuhan :  CHG, HET, AGG, DOM, NUR
```

```
1832150057.pdf
=======Kebutuhan Tertinggi=======
mengekspresikan dirinya -> EXH
kepercayaan diri -> EXH
ketekunan -> END
keuletan -> END
mendominasi -> DOM
memerintah -> DOM
mempengaruhi orang -> DOM

Aspek Kebutuhan :  EXH, END, DOM

=======Kebutuhan Terendah=======
sesuatu yang tetap -> CHG
rutin -> CHG
patuh terhadap aturan -> DEF
patuh -> DEF
aturan -> DEF
mengikuti pendapat -> HET
menyalahkan diri -> ABS
diperhatikan -> SUC
dimanjakan -> SUC
berperilaku agresif -> AGG
agresif -> AGG

Aspek Kebutuhan :  CHG, DEF, HET, ABS, SUC, AGG
```

```
1831150020.pdf
=======Kebutuhan Tertinggi=======
menaruh perhatian -> AFF
bergaul -> AFF
harmonis -> AFF
perasaan bersalah -> ABS
menyalahkan diri -> ABS
menolong -> NUR
merawat orang -> NUR
benda -> NUR

Aspek Kebutuhan :  AFF, ABS, NUR

=======Kebutuhan Terendah=======
ketua kelompok -> DOM
mengatur orang -> DOM
mempengaruhi orang -> DOM
kehidupan seksual -> HET
mencapai prestasi -> ACH
mencapai -> ACH
prestasi -> ACH
memperoleh sesuatu -> ACH
keteraturan -> ORD
sistematis -> ORD
diperhatikan -> SUC
dimanjakan -> SUC

Aspek Kebutuhan :  DOM, HET, ACH, ORD, SUC
```

Hasil Dokumen opsi “ACH”:

```
=======Dokumen yang dicari : =======
1831150030.pdf
1832150039.pdf
1832150072.pdf
1832150044.pdf
1831150145.pdf
1831150002.pdf
1831150068.pdf
1831150006.pdf
```

Hasil Dokumen opsi “DEF”:

```
=======Dokumen yang dicari : =======
1831150070.pdf
1832150018.pdf
1831150002.pdf
```

Hasil Dokumen opsi “ORD”:

```
=======Dokumen yang dicari : =======
1831150239.pdf
1831150007.pdf
1832150083.pdf
1831150006.pdf
```

Hasil Dokumen opsi “EXH”:

```
=======Dokumen yang dicari : =======
1831150228.pdf
1832150060-Inkonsisten 9.pdf
1831150070.pdf
1831150040.pdf
1832150046.pdf
1831150042.pdf
1831150104.pdf
1831150007.pdf
1832150027.pdf
1831150110.pdf
1831150004.pdf
1831150201.pdf
1831150025.pdf
1832150039.pdf
1831150175.pdf
1831150138.pdf
1831150109.pdf
1831150214.pdf
1832150083.pdf
1832150044.pdf
1831150198.pdf
1831150075.pdf
1831150163.pdf
1831150218.pdf
1832150065.pdf
1832150034.pdf
1831150044.pdf
1831150230.pdf
1831150060.pdf
1831150177.pdf
1831150227.pdf
1831150068.pdf
1831150216.pdf
1831150194.pdf
1831150006.pdf
1831150108.pdf
```

Hasil Dokumen opsi “AUT”:

```
=======Dokumen yang dicari : =======
1831150231.pdf
1831150228.pdf
1831150200.pdf
1831150030.pdf
1831150151.pdf
1831150130.pdf
1831150042.pdf
1831150175.pdf
1831150019.pdf
1831150027.pdf
1831150214.pdf
1832150083.pdf
1831150075.pdf
1831150164.pdf
1831150163.pdf
1831150060.pdf
1831150177.pdf
1831150148.pdf
1831150194.pdf
1832150002.pdf
```

Hasil Dokumen opsi “AFF”:

```
=======Dokumen yang dicari : =======
1831150228.pdf
1832150060-Inkonsisten 9.pdf
1831150239.pdf
1831150151.pdf
1832150040.pdf
1832150046.pdf
1831150042.pdf
1832150084.pdf
1831150004.pdf
1831150175.pdf
1831150019.pdf
1831150033.pdf
1831150138.pdf
1831150109.pdf
1831150155.pdf
1831150198.pdf
1831150092.pdf
1832150065.pdf
1831150202.pdf
1832150034.pdf
1831150044.pdf
1831150230.pdf
1831150012.pdf
1832150022.pdf
1832150069.pdf
1831150148.pdf
1831150216.pdf
1831150194.pdf
1832150091.pdf
1831150090- Inkonsisten 9.pdf
```

Hasil Dokumen opsi “INT”:

```
╞=======Dokumen yang dicari : =======
1831150200.pdf
1831150070.pdf
1832150040.pdf
1831150130.pdf
1832150027.pdf
1831150110.pdf
1831150201.pdf
1832150039.pdf
1831150033.pdf
1831150027.pdf
1831150155.pdf
1832150044.pdf
1831150198.pdf
1831150163.pdf
1832150065.pdf
1832150034.pdf
1831150012.pdf
1831150227.pdf
1832150022.pdf
1831150216.pdf
1832150002.pdf
```

Hasil Dokumen opsi “SUC”:

```
╞=======Dokumen yang dicari : =======
1831150231.pdf
1831150200.pdf
1831150151.pdf
1832150040.pdf
1831150040.pdf
1831150007.pdf
1832150072.pdf
1831150109.pdf
1831150164.pdf
1831150218.pdf
1831150202.pdf
1832150069.pdf
```

Hasil Dokumen opsi “DOM”:

```
╞=======Dokumen yang dicari : =======
1832150060-Inkonsisten 9.pdf
1831150030.pdf
1831150104.pdf
1831150110.pdf
1831150004.pdf
1832150072.pdf
1831150092.pdf
1831150164.pdf
1831150218.pdf
1831150044.pdf
1831150148.pdf
```

Hasil Dokumen opsi “ABS”:

```
=======Dokumen yang dicari : =======
1831150040.pdf
1831150155.pdf
1831150092.pdf
1831150012.pdf
1832150002.pdf
```

Hasil Dokumen opsi “NUR”:

```
=======Dokumen yang dicari : =======
1831150239.pdf
1832150046.pdf
1831150104.pdf
1832150084.pdf
1832150027.pdf
1831150214.pdf
1831150002.pdf
1831150202.pdf
1831150230.pdf
1831150177.pdf
1832150022.pdf
1832150069.pdf
1832150091.pdf
1831150108.pdf
1831150090- Inkonsisten 9.pdf
```

Hasil Dokumen opsi “CHG”:

```
=======Dokumen yang dicari : =======
1831150201.pdf
1831150025.pdf
1831150019.pdf
1831150027.pdf
1831150075.pdf
1831150227.pdf
1832150091.pdf
1831150090- Inkonsisten 9.pdf
```

Hasil Dokumen opsi “END”:

```
=======Dokumen yang dicari : =======
1831150231.pdf
1832150084.pdf
1831150025.pdf
1831150033.pdf
1831150138.pdf
1831150060.pdf
1831150068.pdf
```

Hasil Dokumen opsi “HET”:

```
=======Dokumen yang dicari : =======
1831150130.pdf
```

Hasil Dokumen opsi “AGG”:

```
=======Dokumen yang dicari : =======
```