PENGEMBANGAN MODUL *DRAFTING* PADA SISTEM INFORMASI *OMNIBUS LAW*

(Skripsi)

Oleh

Muhammad Umaruddin Syam

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS LAMPUNG

BANDAR LAMPUNG

2024

PENGEMBANGAN MODUL *DRAFTING* PADA SISTEM INFORMASI *OMNIBUS LAW*

Oleh

Muhammad Umaruddin Syam

Skripsi

Sebagai Salah Satu Syarat untuk Mencapai Gelar

SARJANA ILMU KOMPUTER

Pada

Jurusan Ilmu Komputer

Fakultas Matematika dan Ilmu Pengetahuan Alam

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS LAMPUNG

BANDAR LAMPUNG

2024

ABSTRAK

PENGEMBANGAN MODUL *DRAFTING* PADA SISTEM INFORMASI *OMNIBUS LAW*

Oleh

MUHAMMAD UMARUDDIN SYAM

Omnibus Law merupakan salah satu usaha Pemerintah untuk mengurangi pertumbuhan regulasi khususnya undang-undang dengan melakukan simplifikasi atau persingkat aturan untuk menyederhanakannya. Salah satu metode omnibus menurut pasal 64 ayat 1b undang-undang (UU) No. 13 Tahun 2022 adalah mencabut peraturan perundang-undangan yang jenis dan hierarkinya sama. Mencabut peraturan membutuhkan sumber aturan yang dicabut sehingga dalam pembuatannya peraturan yang dibuat tidak menimbulkan tumpang tindih antar aturan.

Metode mencabut peraturan membuat *legal drafter* memerlukan kemampuan dalam memahami peraturan sejenis maupun hierarkinya sehingga diperlukannya bantuan berupa basis data dan pengecekan kemiripan. Pengecekan kemiripan norma dengan UU dilakukan menggunakan algoritme *cosine similarity* yang menghitung kesamaan antar dua vektor. Vektor didapat dari jumlah kata yang muncul dalam satuan gramatikal dengan menggunakan metode *term frequency-inverse document frequency* (TF-IDF). Data UU yang dipakai pada penelitian ini dari tahun 1945 – 2022 dengan jumlah 1705 UU dari 43 kategori UU dan 60186 pasal dan/atau ayat.

Percobaan menggunakan *cosine similarity* menghasilkan nilai *recall* sebesar 90.10% dari 500 data uji. Pengembangan sistem informasi dilakukan sehingga *end-user* dapat menggunakan pengecakan kemiripan norma. Metode pengembangan sistem informasi dilakukan dengan model pengembangan *waterfall* dengan 4 tahapan yaitu *analysis*, *design*, *implementation*, dan testing. Algoritme diterapkan menggunakan FastAPI dengan 2 tes skenario sukses. Tampilan pengguna menggunakan Laravel dengan 3 tes skenario sukses.

Kata kunci: *Cosine Similarity*, FastAPI, Laravel, *Omnibus Law*

ABSTRACT

DRAFTING MODULE DEVELOPMENT IN OMNIBUS LAW SYSTEM INFORMATION

By

MUHAMMAD UMARUDDIN SYAM

Omnibus law is one of the government's efforts to reduce growth of regulations, especially Undang-Undang (UU), by simplifying or shortening rules to simplify them. One of omnibus methods according to Article 64 paragraph 1b of UU No. 13 of 2022 is to revoke laws and regulations of the same type and hierarchy. Revocation of regulations requires the source of the revoked regulations so that the enacted regulations do not cause overlapping of regulations. Method of revoking regulations makes the drafters of legislation need the ability to understand similar regulations and their hierarchy, so assistance in the form of databases and checking similarities is needed. Checking the similarity of rules with laws is done using the cosine similarity algorithm, which calculates the similarity between two vectors. Vectors are obtained from the number of words appearing in grammatical units using term frequency-inverse document frequency (TF-IDF) method. The legal data used in this research is from 1945 - 2022 with a total of 1705 laws from 43 categories of laws and 60186 articles and/or paragraphs. Experiments using cosine similarity resulted in a recall value of 90.10% from 500 test data. Information system development was carried out in such a way that end-users can use the norm similarity check. Information system development method was carried out using a waterfall development model with 4 stages, namely analysis, design, implementation, and testing. Algorithm is implemented using FastAPI with 2 success scenario tests. User interface is implemented using Laravel with 3 successful scenario tests.

Key words : Cosine Similarity, FastAPI, Laravel, Omnibus Law

Judul Skripsi	: PENGEMBANGAN MODUL *DRAFTING* PADA SISTEM INFORMASI *OMNIBUS LAW*
Nama Mahasiswa	: Muhammad Umaruddin Syam
Nomor Pokok Mahasiswa	: 1857051006
Program Studi	: Ilmu Komputer
Fakultas	: Matematika dan Ilmu Pengetahuan Alam

MENYETUJUI

1. Komisi Pembimbing

Aristoteles, S.Si., M.Si.

NIP. 198105212006041002

2. Ketua Jurusan Ilmu Komputer

Didik Kurniawan, S.Si., M.T.

NIP. 198004192005011004

MENGESAHKAN

1. Tim Penguji

   Ketua : Aristoteles, S.Si., M.Si. .....

   Penguji I

   Penguji Pembahas : Tristiyanto, S.Kom., M.I.S., Ph.D. .....

   Penguji II

   Penguji Pembahas : Bambang Hermanto, S.Kom., M.Cs .....

2. Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam

Dr. Eng. Heri Satria, S.Si., M.Si.

NIP. 197110012005011002

Tanggal Lulus Ujian Skripsi : 28 Februari 2024

PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : Muhammad Umaruddin Syam

NPM : 1857051006

Dengan ini menyatakan bahwa skripsi saya yang berjudul "PENGEMBANGAN MODUL *DRAFTING* PADA SISTEM INFORMASI *OMNIBUS LAW*" adalah benar hasil karya sendiri dan bukan orang lain. Seluruh tulisan yang tertuang dalam skripsi ini telah mengikuti kaidah penulisan karya ilmiah Universitas Lampung. Jika di kemudian hari terbukti skripsi saya adalah hasil penjiplakan atau dibuat oleh orang lain, maka saya bersedia menerima sanksi sesuai dengan peraturan yang berlaku

Bandar Lampung, 29 Februari 2024

Penulis

Muhammad Umaruddin Syam

NPM. 1857051006

RIWAYAT HIDUP

Muhammad Umaruddin Syam adalah penulis skripsi yang berjudul "Pengembangan Modul Drafting pada Sistem Informasi Omnibus Law". Penulis dilahirkan di Lampung pada tanggal 30 Desember 1999 sebagai anak kedua dari tiga bersaudara.

Pendidikan yang Penulis dimulai dari Taman Kanak-kanak (TK) Aisyiyah Bustanul Athfal 1 Pringsewu pada tahun 2004. Pada tahun 2006 Penulis melanjutkan pendidikan dasar di Sekolah Dasar (SD) Muhammadiyah Pringsewu. Tahun 2012 Penulis melanjutkan pendidikan menengah di Sekolah Menengah Pertama (SMP) Negeri 1 Pringsewu, dan dilanjutkan ke Sekolah Menengah Atas (SMA) Negeri 1 Pringsewu.

Tahun 2018 penulis diterima sebagai mahasiswa Jurusan Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung melalui jalur Seleksi Mandiri Masuk PTN-Barat (SMM PTN-Barat). Selama menjadi mahasiswa, penulis berpartisipasi dalam berbagai kegiatan, antara lain.

1. Anggota Adapter Himpunan Mahasiswa Ilmu Komputer pada tahun 2018.

2. Ketua divisi lomba Esai Nasional Pekan Raya Jurusan VIII.
3. Staff Ahli Keuangan Himpunan Mahasiswa Ilmu Komputer periode 2019/2020.
4. Asisten Dosen Jurusan Ilmu Komputer untuk mata kuliah Sistem Operasi dan Basis Data tahun ajaran 2019/2020 dan 2020/2021.
5. Mahasiswa magang di PLN UID Lampung sebagai Performance Analyst pada tahun 2021.
6. Kuliah Kerja Nyata (KKN) pada periode II tahun 2021 di Pekon Ambarawa, Kec. Ambarawa, Kab. Pringsewu.

MOTTO

“Be Legendary”

(Kobe Bryant)

“The most important thing is to try and inspire people so that they can be great at whatever they want to do”

(Kobe Bryant)

PERSEMBAHAN

Dengan menyebut nama Allah SWT yang telah memberikan kekuatan dan memberkahi hidup penulis, skripsi ini saya persembahkan kepada kedua orang tua dan keluarga penulis yang telah memberikan dukungan tidak terbatas, kepada semua dosen yang telah menginspirasi, kepada semua teman, dan negara tercinta Indonesia.

SANWACANA

Puji syukur penulis ucapkan kehadirat Allah SWT, karena atas rida dan rahmat-Nya skripsi ini dapat diselesaikan. Skripsi dengan Judul "Pengembangan Modul *Drafting* pada Sistem Informasi *Omnibus Law*" dapat menjadi tambahan wawasan pagi penulis maupun pembaca dikemudian harinya mengenai *omnibus law* maupun penerapan metode *cosine similarity* dengan bantuan FastAPI.

Selama proses penulisan skripsi ini tidak terlepas dari dukungan banyak pihak yang telah membimbing, membantu dan memberi semangat kepada saya, sehingga pada kesempatan ini saya ingin menyampaikan ungkapan terima kasih kepada:

1. Orang Tua, Kakak, Adik, dan Keluarga yang yang senantiasa memberikan doa, dukungan, dan kasih sayang.
2. Bapak Aristoteles, S.Si., M.Si. sebagai pembimbing utama yang telah membimbing, memberikan kritik dan saran serta membina dalam menyelesaikan skripsi ini yang dapat diselesaikan dengan baik.
3. Bapak Tristiyanto, S.Kom., M.I.S., Ph.D. pembahas pertama yang telah membimbing saya dalam memberikan ide, kritik, saran sehingga penulisan skripsi ini dapat diselesaikan dengan baik.

4. Bapak (Alm) Drs. Rd. Irwan Adi Pribadi, M.Kom dan Bapak Bambang Hermanto, S.Kom., M.Cs. sebagai pembahas kedua yang telah membimbing saya dalam memberikan ide, kritik, saran sehingga penulisan skripsi ini dapat diselesaikan dengan baik.
5. Bapak Prof. Dr. Eng. Admi Syarif sebagai pembimbing akademik saya yang telah membimbing saya selama perkuliahan saya di Jurusan Ilmu Komputer.
6. Bapak Didik Kurniawan, S.Si., M.T., selaku Ketua Jurusan Ilmu Komputer Universitas Lampung.
7. Ibu Anie Rose Irawati, ST, M.Cs., selaku sekretaris Jurusan Ilmu Komputer FMIPA Universitas Lampung.
8. Bapak dan Ibu Dosen Jurusan Ilmu Komputer FMIPA Universitas Lampung yang telah memberikan ilmu dan pengalaman dalam hidup untuk menjadi lebih baik.
9. Ibu Ade Nora Maela, Bang Zainuddin, dan Mas Nofal yang telah membantu segala urusan administrasi dan segala jenis izin penulis di Jurusan Ilmu Komputer.
10. Teman-teman Heemahoora yang telah menemani saya dalam menjalani hari-hari dan saling memberi semangat selama masa perkuliahan.
11. Keluarga Ilmu Komputer 2018 yang tidak bisa penulis sebut satu persatu. Keluarga kedua penulis yang telah memberikan pengalaman tak ternilai semasa duduk di bangku kuliah.

12. Semua pihak yang telah berpartisipasi baik secara langsung maupun tidak langsung dalam membantu penyusunan skripsi ini.

Akhir kata, penulis menyadari bahwa penelitian ini masih jauh dari kesempurnaan. Oleh karena itu, saran dan kritik yang bersifat membangun sangat diharapkan guna perbaikan di masa mendatang.

Semoga hasil penelitian ini dapat memberikan manfaat dan kontribusi positif bagi perkembangan ilmu pengetahuan. Semoga Allah SWT senantiasa memberikan rahmat dan hidayah-Nya kepada kita semua.

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Gambar Halaman

DAFTAR TABEL

Tabel Halaman

I. PENDAHULUAN

1.1 Latar Belakang

Sebagai negara hukum, Indonesia memiliki banyak peraturan di dalamnya. Presiden Jokowi mengatakan bahwa peraturan di negara Indonesia sudah terlalu banyak dan negara kita bukan "negara peraturan" (Setiadi, 2020). Dilansir dari laman peraturan.go.id, pada 19 Mei 2022 Indonesia memiliki 50.538 dengan rincian pada Tabel 1.

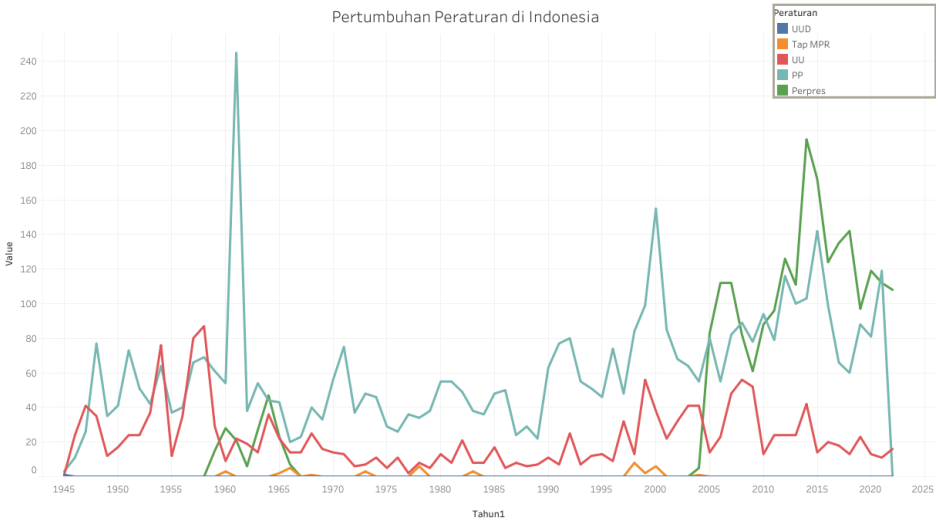
Tabel 1. Jumlah Peraturan di Indonesia dengan *heatmap*

Jenis Regulasi	Jumlah
Instruksi Presiden	387
Keputusan Presiden	5,182
Ketetapan Majelis Permusyarawatan Rakyat	41
Otoritas Jasa Keuangan	429
Penetapan Presiden	75
Peraturan Badan Pemeriksa Keuangan	31
Peraturan Bank Indonesia	185
Peraturan Daerah	15,982
Peraturan Kolonial	121
Peraturan Lembaga Pemerintah Non Kementrian	469
Peraturan Menteri	17,315
Peraturan Pemerintah	4,756
Peraturan Pemerintah Pengganti Undang-Undang	182
Peraturan Presiden	2,254
Putusan Mahkamah Agung	202
Putusan Mahkamah Konstitusi	1,031
Undang-Undang	1,715
Undang-Undang Darurat	177
Undang-Undang Dasar	2
Undang-Undang Dasar Sementara	2

Jumlah 2 17,315

Jumlah regulasi pada Tabel 1 dapat bertambah seiring berjalannya waktu. Tingkat pertumbuhan regulasi di Indonesia jika dilihat dari median pertumbuhan untuk 5 hierarki teratas menurut pasal 7 ayat 1 Undang-Undang (UU) Nomor 12 Tahun

2011, UUD memiliki median 0, Tap MPR memiliki median 0, UU memiliki median 16, PP memiliki median 55, dan Perpres memiliki median 0. Rincian pertumbuhan regulasi dapat dilihat pada Gambar 1.



Gambar 1. Pertumbuhan Peraturan di Indonesia.

Penambahan regulasi dilakukan untuk mengatur dan menertibkan kehidupan masyarakat, dalam berbangsa maupun bernegara. Banyaknya regulasi dapat menimbulkan dampak negatif, secara dampak negatif akibat banyaknya regulasi dikelompokkan ke dalam 4 kelompok, yang meliputi (1) Konflik regulasi yaitu situasi apabila ada regulasi atau pasal yang dengan jelas berbenturan dengan regulasi yang sudah ada, (2) Konsistensi regulasi yang tidak konsisten yaitu jika terdapat regulasi atau ketentuan yang inkonsisten dalam satu perundang-undangan beserta turunannya, (3) Penafsiran regulasi yang beragam yaitu ketidakjelasan tentang tujuan dan subyek regulasi. Hal ini menyebabkan perumusan kebahasaan yang tidak jelas (sulit dipahami) dan memiliki sistematika yang tidak tepat, (4) Non-operasional, yaitu regulasi tidak berpengaruh, tetapi regulasi tersebut masih berlaku atau tidak ada regulasi penegakan (Sadiawati, 2015).

Pemerintah berusaha mengurangi pertumbuhan regulasi khususnya undang-undang dengan melakukan simplifikasi atau persingkat aturan untuk menyederhanakannya. Pengurangan regulasi diimplementasikan menggunakan pendekatan omnibus *law*. Menurut (Muqsith, 2020), Omnibus *Law* merupakan

suatu bentuk hukum yang mengatur berbagai persoalan yang kompleks dan kemudian ditempatkan dalam satu kerangka hukum. Omnibus *law* cocok diterapkan di negara yang regulasinya saling tumpang tindih, hiper regulasi, dan disharmoni, selain itu omnibus *law* menggabungkan banyak peraturan yang dibahas dalam satu peraturan, sehingga menjawab dua hal sekaligus: efisiensi hukum dan keselarasan hukum (Setiadi, 2020). Penyederhanaan banyak regulasi dalam satu regulasi berdampak juga kepada efisiensi anggaran negara dalam proses pembentukan undang-undang.

Salah satu metode omnibus menurut pasal 64 ayat 1b UU No. 13 Tahun 2022 adalah mencabut peraturan perundang-undangan yang jenis dan hierarkinya sama. Mencabut peraturan membutuhkan sumber aturan yang dicabut sehingga dalam pembuatannya peraturan yang dibuat tidak menimbulkan tumpang tindih antar aturan. Metode mencabut peraturan membuat *legal drafter* memerlukan kemampuan dalam memahami peraturan sejenis maupun hierarkinya. Basis data mengenai UU baik pasal maupun ayat dapat membantu pekerjaan *legal drafter*, selain itu diperlukan juga pengecekan kemiripan norma dengan UU sehingga dapat meminimalkan perbedaan bahasa.

Algoritme yang dapat diterapkan dalam melakukan pengecekan kemiripan, salah satunya *cosine similarity*. *Cosine Similarity* memiliki kelebihan dibanding beberapa algoritme lain karena tidak bergantung pada panjang pendeknya suatu dokumen dan memiliki keakuratan yang cukup tinggi (Riyani et al., 2019). Algoritme *cosine similarity* dapat dipadukan dengan algoritme *soft cosine similarity* yang mencari kata yang berhubungan dengan kata kunci di dalam set data yang tersedia. Oleh karena itu, penelitian ini dirancang agar para pembuat undang-undang atau penulis undang-undang dapat mencari kata kunci (norma) pada undang-undang secara menyeluruh dan mengimplementasikan pada pembuatan undang-undang dengan bantuan mesin pencarian sehingga dapat memangkas waktu dalam memahami tiap undang-undang yang ada.

1.2 Rumusan Masalah

Berdasarkan latar belakang, penelitian ini memiliki rumusan masalah sebagai berikut.

1. Bagaimana cara mengukur tingkat kesamaan antara kata kunci pembuatan rancangan undang-undang dan undang-undang yang sudah ada?
2. Bagaimana cara menerapkan model tingkat kesamaan antara kata kunci pembuatan rancangan undang-undang dan undang-undang yang sudah ada agar dapat digunakan oleh *legal drafter*?

1.3 Batasan Masalah

Penelitian harus memiliki ruang lingkup dan keterbatasan, sehingga hanya dapat fokus pada masalah, sehingga dapat menemukan hasil dari pemecahan masalah. Oleh karena itu ruang batasan masalah didefinisikan sebagai berikut.

1. Pencarian undang-undang terkait menggunakan *keyword* atau norma hukum menggunakan metode *cosine similarity* untuk melihat tingkat kemiripannya dengan set data.
2. Set data yang digunakan pada pengukuran kesamaan adalah Undang-undang dari tahun 1945 – 2021 yang sudah diekstraksi sesuai pasal dan ayatnya.
3. Perancangan dan pembuatan modul menggunakan FastAPI yang menggunakan bahasa pemrograman python sebagai *backend* dan Laravel yang menggunakan bahasa pemrograman PHP sebagai *frontend*.

1.4 Tujuan Penelitian

Adapun tujuan yang ingin dicapai dalam penelitian ini adalah sebagai berikut.

1. Untuk mengetahui cara mengukur tingkat antara kata kunci rancangan undang-undang dan undang-undang yang sudah ada.
2. Untuk mengetahui cara menerapkan model tingkat kesamaan antara rancangan undang-undang dan undang-undang yang sudah ada agar dapat digunakan oleh *legal drafter*.

1.5 Manfaat Penelitian

Manfaat dari penelitian ini adalah.

1. Mengetahui tingkat kesamaan kata antara kata kunci rancangan undang-undang dan undang-undang yang sudah ada.
2. Mengetahui akurasi metode *cosine similarity* untuk set data undang-undang.
3. Hasil *similarity* dapat dimanfaatkan *legal drafter* dalam penyusunan rancangan undang-undang.

II. TINJAUAN PUSTAKA

2.1 Penelitian Terkait

Penulis melakukan penelitian ini berdasarkan penelitian-penelitian terdahulu yang telah dilakukan sebelumnya. Penulis menggunakan penelitian sebelumnya sebagai referensi dasar penelitian. Penelitian yang dirujuk oleh penulis adalah.

Pertama, penelitian tentang "Pencarian Perangkingan Obat Tradisional Berdasarkan Gejala Penyakit Menggunakan Metode *Cosine Similarity"* (Hima Pandalu, 2014). Penelitian yang dilakukan (Hima Pandalu, 2014) menggunakan *cosine similarity* untuk mengurutkan atau mengurutkan jenis obat yang paling mirip dengan kata kunci yang dimasukkan. Penggunaan pencarian ini yaitu dengan cara pengguna menginput tanda-tanda awal penyakit yang diderita oleh pengguna kemudian akan diproses dengan mencari nilai kemiripan tanda-tanda awal dengan khasiat obat tradisional dengan *cosine similarity*. Dari penelitian yang dilakukan didapatkan nilai *recall* dan *precision* untuk pemeringkatan obat tradisional dengan menggunakan metode *cosine similarity*, nilai rata-rata *recall* yang di dapat dalam lima kali uji mencapai 91%, sedangkan nilai rata-rata *precision* dalam lima uji coba adalah 100%. Hasil *recall* dan *precision* ini menunjukkan bahwa metode yang digunakan dapat diterapkan untuk pencarian obat dengan input kata kunci berupa tanda-tanda awal penyakit.

Kedua, penelitian tentang "Deteksi Plagiarisme pada Artikel Jurnal Menggunakan Metode *Cosine Similarity"* oleh (Pratama et al., 2019). Berdasarkan penelitian tersebut dalam menentukan kemiripan dokumen dengan membandingkan artikel jurnal yang ada di repository menggunakan *cosine similarity* diperoleh nilai *recall* 13%, dan *precision* 8%.Ketiga, penelitian tentang "*Analysis of Thesis Plagiarism*

Document Level with Cosine Similarity Method and TF-IDF Weighting" oleh (Azmi, 2022). Berdasarkan penelitian tersebut algoritme yang dipakai menggunakan *Cosine Similarity* dan TF/IDF. Penelitian tersebut Hasil penelitian ini menunjukkan bahwa *stemming* pada proses penerapan model meningkatkan sensitivitas *cosine similarity* dalam memutuskan kesamaan dokumen. Hal ini tercermin dari kecenderungan nilai kemiripan yang lebih tinggi dibandingkan dengan proses non-*stemming*.

Keempat, penelitian tentang "Perancangan dan Pembuatan Aplikasi Pencarian Informasi Beasiswa dengan Menggunakan *Cosine Similarity"* (Kurniawan et al., 2014). Berdasarkan penelitian tersebut pencarian data beasiswa yang relevan satu sama lain yang dibutuhkan oleh *user* berdasarkan input yang dimasukkan. Dalam penelitian ini menggunakan metode *Information Retrieval* (IR) yang digunakan untuk menjelajahi dokumen yang terdapat di dalam *database*. *Information Retrieval System* untuk melakukan pencarian berdasarkan tingkat kemiripan pada setiap dokumen. Untuk mendapatkan informasi beasiswa yang relevan terhadap input yang dimasukkan, digunakan metode *cosine similarity* untuk menghitung seberapa besar nilai kemiripan antara dokumen dan input. Metode *cosine similarity* memungkinkan adanya perangkingan dokumen. Perangkingan dokumen disesuaikan dengan kesamaan atau relevansi input pencarian yang dimasukkan.

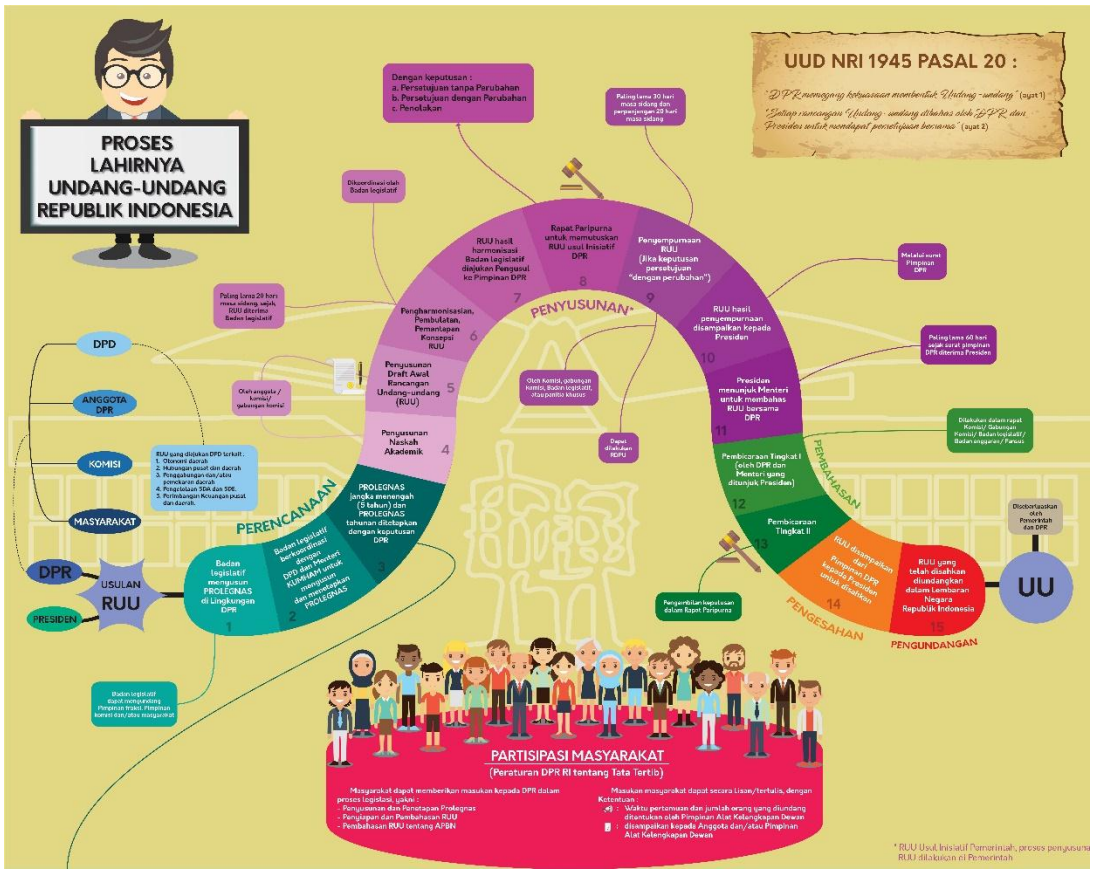
2.2 Landasan Teori

Landasan teori yang digunakan peneliti dalam penelitian ini adalah sebagai berikut.

2.2.1 Undang-Undang

Undang-undang, dalam arti formil, merujuk pada keputusan penguasa yang ditentukan berdasarkan bentuk dan proses pembuatannya. Sementara itu, dalam arti materiil, undang-undang mengacu pada keputusan atau ketetapan penguasa yang dilihat dari substansinya, dan ketetapan ini disebut sebagai undang-undang yang mengikat setiap orang secara umum (Ruslan, 2011).

Undang-undang (UU) merupakan salah satu jenis peraturan perundang-undangan. Undang-undang disahkan melalui proses legislatif yang melibatkan badan legislatif atau parlemen, seperti DPR di Indonesia. Menurut pasal 1 UU Nomor 12 Tahun 2011, pembentukan undang-undang terdiri dari lima tahapan utama, yaitu perencanaan, penyusunan, pembahasan, pengesahan, dan pengundangan. Proses pembuatan UU dapat dilihat pada Gambar 2.



Gambar 2. Infografis Proses Pembuatan Undang-Undang.
Sumber : https://www.dpr.go.id/dokhumas/publication/Infografis-Proses-Pembuatan-UU-di-DPR-RI.pdf

Proses pembuatan berdasarkan info grafis pada Gambar 2 tahapan pembuatan UU yang memiliki lima tahapan utama diuraikan menjadi lima belas tahapan. Tahapan awal yaitu perencanaan yang akan menghasilkan Program Legislasi Nasional (prolegnas) jangka menengah dan prolegnas tahunan yang ditetapkan dengan keputusan DPR. Tahapan kedua dari proses pembuatan UU yaitu Penyusunan,

penyusunan berawal dari pembuatan naskah akademik oleh anggota/komisi/gabungan komisi yang dibantu oleh *legal drafter*. Naskah akademik akan dijadikan dasar dalam penyusunan *draft* awal Rancangan Undang-undang (RUU). RUU akan diharmonisasi, dibulatkan, dan dimantapkan konsepsinya sehingga akan diusulkan dalam rapat paripurna. Rapat paripurna dilakukan untuk memutuskan RUU usul inisiatif DPR, RUU akan disempurnakan jika keputusan persetujuan "dengan perubahan". Hasil Penyempurnaan akan disampaikan kepada Presiden melalui surat Pimpinan DPR. Presiden akan menunjuk Menteri untuk membahas RUU bersama DPR. Tahapan ketiga yaitu pembahasan, tahap pembahasan akan dimulai dengan pembicaraan tingkat I yang dilakukan oleh DPR dan menteri yang ditunjuk Presiden. Pembicaraan tingkat II akan dilakukan apabila pembicaraan I sudah selesai dan diambil keputusan dalam rapat paripurna. Tahapan keempat yaitu pengesahan, tahap ini dilakukan apabila RUU sudah disetujui oleh pihak yang terlibat dalam pembicaraan baik tingkat I maupun tingkat II. RUU yang disetujui akan disampaikan dari Pimpinan DPR kepada Presiden untuk disahkan. Tahap kelima yaitu RUU yang telah disahkan akan diundangkan dalam Lembaran Negara Republik Indonesia. UU yang sudah disahkan akan disebarluaskan oleh Pemerintah dan DPR.

Undang-undang dianggap sebagai salah satu dari tiga fungsi utama pemerintahan yang berasal dari prinsip pemisahan kekuasaan. Kelompok yang memiliki wewenang formal dalam membuat undang-undang disebut legislator (pembuat undang-undang), sementara badan yudikatif pemerintahan memiliki wewenang formal dalam menafsirkan undang-undang, dan badan eksekutif pemerintahan hanya dapat bertindak sesuai dengan batasan kekuasaan yang telah ditetapkan oleh hukum perundang-undangan.

2.2.2 Omnibus *Law*

Omnibus *law* adalah metode atau konsep pembuatan peraturan yang dapat menyelesaikan masalah dalam berbagai UU yang menyangkut perizinan dengan membuat satu UU baru yang merevisi pasal dalam beberapa Undang-Undang (Azhar, 2019). Menurut (Mayasari, 2020) omnibus *law* adalah cara yang tepat untuk mengembangkan kerangka hukum untuk perizinan proses bisnis Indonesia.

Metode ini memungkinkan pembuatan peraturan yang mencakup beberapa materi substantif, atau beberapa yang lebih kecil dalam satu aturan guna menciptakan ketertiban, kepastian hukum, dan kemanfaatan.

Menurut UU Nomor 13 Tahun 2022, omnibus memiliki 3 metode yaitu (1) memuat materi muatan baru, (2) mengubah materi muatan yang memiliki keterkaitan dan/atau kebutuhan hukum yang diatur dalam berbagai peraturan Perundang-undangan yang jenis dan hierarkinya sama, (3) mencabut peraturan Perundang-undangan yang jenis dan hierarkinya sama. Metode omnibus dilakukan dengan menggabungkan ke dalam satu Peraturan Perundang-undangan untuk mencapai tujuan tertentu.

Omnibus *law* digunakan untuk menyelesaikan masalah hukum yang kompleks atau untuk mengelola banyak hal sekaligus. Pembuatan omnibus *law* memiliki tujuan untuk mempermudah proses legislatif dengan cara membuat satu kebijakan yang mencakup banyak materi atau bidang hukum. Selain mempermudah proses legislatif, omnibus *law* yang mencakup banyak materi atau bidang hukum dalam satu kebijakan dapat menjamin tingkat konsistensi dalam kebijakan hukum.

Pembuatan omnibus *law* melibatkan beberapa pihak yaitu parlemen atau lembaga legislatif, pemerintah, *stakeholder*, dan ahli hukum. Parlemen atau lembaga legislatif memegang tanggung jawab dalam pembuatan undang-undang, mulai dari membahas rancangan sampai menyetujui atau menolak kebijakan yang dirancang. Pemerintah memiliki pernah sebagai penanggung jawab dalam implementasi undang-undang. Dalam proses pembuatan omnibus *law*, pemerintah akan memberi masukan dan membuat peraturan pelaksanaan atau peraturan turunan yang lebih spesifik untuk menjelaskan bagaimana omnibus *law* tersebut harus dilaksanakan. *Stakeholder* sebagai pihak yang terdampak oleh kebijakan hukum yang tercantum dalam omnibus *law* dapat memberi masukkan atau saran materi

Pemerintah akan menggunakan omnibus *law* ketika terdapat masalah hukum yang kompleks atau terdapat banyak hal yang perlu dikelola sekaligus. Omnibus *law* dianggap sebagai cara yang efisien dalam menyederhanakan proses pembuatan

undang-undang dan mempercepat proses legislatif. Omnibus *law* juga dianggap sebagai cara untuk mengintegrasikan berbagai aspek hukum dalam satu kebijakan dan menjamin konsistensi dalam kebijakan hukum. Namun, omnibus *law* juga dapat dianggap kontroversial karena banyak dianggap sebagai upaya untuk mengeliminasi pembatasan atau kontrol yang ada dalam hukum yang ada, terutama dalam bidang yang terkait dengan hak asasi manusia atau lingkungan.

2.2.3 *Legal Drafting*

Legal drafting adalah konsep dasar tentang penyusunan peraturan perundang-undangan yang berisi tentang naskah akademik hasil kajian ilmiah beserta naskah awal peraturan perundang-undangan yang diusulkan (Sholichah, 2021). Menurut Kamus Besar Bahasa Indonesia (KBBI), legal drafting mengacu pada kepatuhan terhadap peraturan perundang-undangan atau hukum. Sementara itu, draf didefinisikan sebagai rancangan atau konsep. Oleh karena itu, legal drafting merujuk pada proses merancang naskah hukum, termasuk pembuatan rancangan peraturan, keputusan, atau perjanjian (Sholichah, 2021). Proses *legal drafting* penting dilakukan karena untuk memastikan bahwa dokumen hukum yang dibuat memenuhi semua persyaratan hukum yang berlaku dan dapat digunakan dalam proses pengadilan atau transaksi bisnis yang sah. *Legal drafting* juga dapat diartikan sebagai proses pembuatan peraturan perundang-undangan yang meliputi tahapan perencanaan, penyusunan, pembahasan, pengesahan atau penetapan, dan pengundangan(Supriyanti et al., 2020).

Penyusunan peraturan perundang-undangan dilakukan oleh advokat atau juru tulis hukum (*legal drafter)* yang memiliki keahlian dalam menulis dan menyusun dokumen hukum yang tepat dan sesuai dengan hukum yang berlaku. Pengetahuan yang mendalam mengenai hukum dan peraturan yang berlaku di negara serta kemampuan menulis dengan jelas dan tegas menjadi tolak ukur dalam *legal drafting*. Pemahaman terhadap *legal drafting* sangat penting, mengingat Pasal 1 ayat (3) Undang Undang Dasar Negara Republik Indonesia tahun 1945, yang menyatakan bahwa Indonesia adalah negara hukum. Oleh karena itu, segala hal yang ada di Indonesia haruslah didasarkan atau dilandasi oleh aturan. Dengan demikian, peraturan yang baik dapat dibuat melalui pemahaman dan proses *legal*

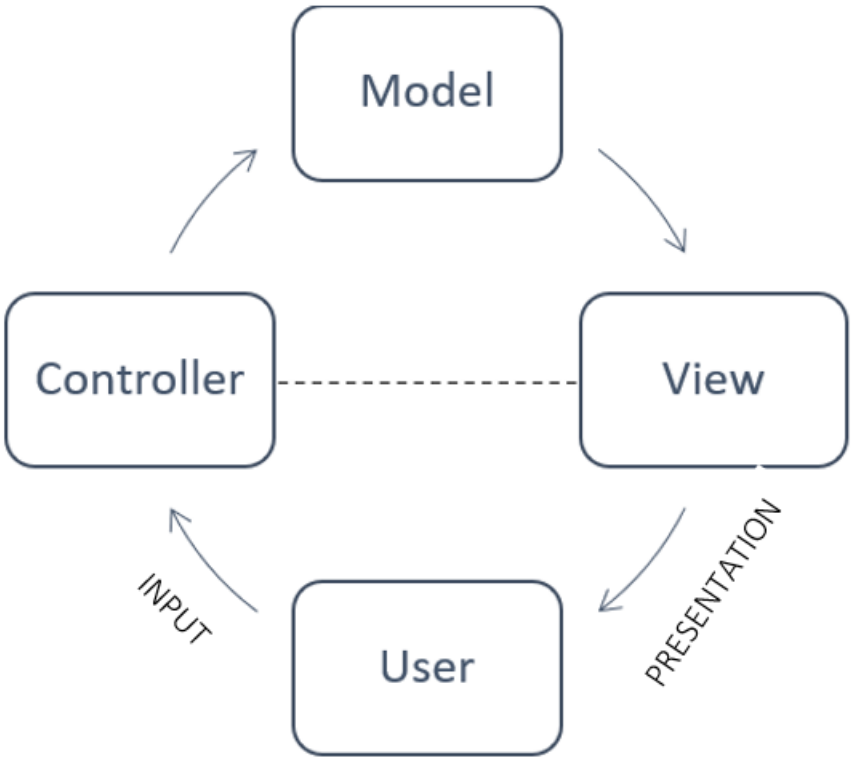
drafting yang baik. Fungsi *legal drafting* dalam proses pembuatan peraturan perundang-undangan memiliki peran yang sangat penting, karena hal ini membantu secara signifikan dalam pembuatan produk hukum yang akurat dan lengkap (Al Hidayat, 2017).

2.2.4 Sistem Informasi

Sistem informasi adalah sistem dalam organisasi yang merangkum fungsi manajemen operasi organisasi dan persyaratan pemrosesan transaksi sehari-hari yang mendukung kegiatan strategis organisasi untuk menyediakan pihak luar tertentu dengan informasi yang diperlukan untuk memudahkan dalam membuat keputusan (Anggraeni & Irviani, 2017). Komponen yang terdapat dalam sistem informasi yaitu perangkat keras (*hardware*), perangkat lunak (*software*), data, dan orang-orang yang terorganisasi untuk memproses, menyimpan, dan menyampaikan informasi. Namun, dalam realitasnya, tidak semua sistem informasi melibatkan semua komponen tersebut, seperti halnya sistem informasi pribadi yang fokus pada jaringan telekomunikasi. Menurut (Zakiyudin, 2011) Sistem informasi adalah sebuah sistem di dalam suatu organisasi yang menghubungkan kebutuhan pengolahan transaksi, mendukung operasi, manajerial, dan kegiatan strategis organisasi dengan laporan yang diperlukan, baik dari internal maupun eksternal.

2.2.5 Laravel

Laravel adalah sebuah kerangka kerja web dibangun menggunakan bahasa pemrograman PHP yang diciptakan untuk pengembangan situs web dengan pola *Model-View-Controller* (MVC) (Purnama Sari & Wijanarko, 2020). Tahun 2011 Taylor Otwell memulai proyek kerangka kerja laravel karena Taylor Otwell tidak dapat menemukan kerangka kerja yang mengikuti pembaruan dari bahasa pemrograman PHP.



Gambar 3. Konsep *Model - View - Controller* pada Laravel.

Gambar 3 menunjukkan konsep MVC pada Laravel yang dimulai dari *user* yang mengirimkan *input* lalu diterima oleh *controller*. Komponen *Controller* berupa *bussiness logic* menjalankan perintah yang diberikan oleh *user* baik menuju *model* atau langsung menuju *view*. *Controller* dapat menjalankan perintah langsung menuju *view* dikarenakan tidak ada data yang harus dipanggil dalam *database*.

Framework Laravel menawarkan sebuah fitur terbaru yang dikenal sebagai migration untuk berkomunikasi dengan basis data. Pengembang dapat dengan mudah melakukan modifikasi terhadap basis data pada platform yang digunakan secara independen, karena skema basis data diwakili dalam sebuah *class* menggunakan fitur migration. Migration dapat diimplementasikan dengan berbagai jenis basis data yang didukung oleh Laravel, termasuk MySQL, PostgreSQL, MSSQL, dan SQLite. Untuk mengimplementasikan *Active Record* pada Laravel, digunakan fitur yang disebut sebagai Eloquent, yang mengadopsi standar OOP modern.

Laravel juga menyediakan *Command Line Interface* (CLI) yang disebut sebagai Artisan, yang memungkinkan pengembang berinteraksi dengan aplikasi untuk melakukan berbagai tindakan seperti migrasi, pengujian, atau pembuatan *controller* dan model. Selain itu, Laravel juga menyertakan mesin template Blade yang memberikan tampilan yang estetis dan kode yang terstruktur pada bagian tampilan (*view*) (Luthfi, 2017).

2.2.6 FastAPI

FastAPI adalah sebuah kerangka kerja web yang digunakan untuk membuat RESTful *Application Programming Interface* (API) dengan menggunakan bahasa pemrograman python. FastAPI dirancang dengan tujuan untuk memberikan kemudahan penggunaan, dan memiliki desain yang mirip dengan kerangka kerja populer lainnya, seperti Flask, yang juga sangat populer di kalangan pengembang (Azhari, 2022). FastAPI juga dikenal sebagai salah satu framework paling cepat dalam bahasa pemrograman Python, dan sebanding dengan performa framework NodeJS dan Go (Samuel Rajiv & K, 2022).

Fitur andalan dari FastAPI adalah validasi otomatis terhadap data masukan dengan menggunakan anotasi tipe dan fitur validasi Pydantic, FastAPI secara otomatis memvalidasi data masukan, menghasilkan respons kesalahan yang jelas jika data tidak sesuai dengan skema yang ditentukan. Ini membantu dalam mengurangi kesalahan pemrograman dan meningkatkan keamanan aplikasi. Pendekatan asinkron dan dukungan untuk HTTP (*Hypertext Transfer Protocol*), FastAPI dapat menangani banyak permintaan secara bersamaan dengan efisiensi tinggi. Ini membuatnya cocok untuk membangun aplikasi yang perlu menangani lalu lintas tinggi dan skala yang besar.

Pembangunan FastAPI menggunakan standar dari OpenAPI yang dikenal sebagai Swagger dan JSON Schema (Maholtra, 2022). *Request* data dari browser dalam FastAPI dijawab dengan 3 digit angka yaitu HTTP *status code*. HTTP *status code* memiliki nama untuk mengenali respons apa yang diberikan oleh server, beberapa respons yang dijawab oleh server terdapat pada Tabel 2.

Tabel 2. HTTP *status code* (Kode Status)

No	Kode Status	Arti
1	100	Informasi
2	200	Sukses
3	300	*Redirection*
4	400	Eror klien
5	500	Eror Server

2.2.7 *Text Preprocessing*

Preprocessing adalah tahapan awal yang dilakukan sebelum sebuah data diproses. Tahap *preprocessing* bertujuan untuk menyamakan set data agar lebih mudah diproses (Silvi Lydia et al., 2018). Data latih perlu melalui tahap *preprocessing* terlebih dahulu sebelum digunakan untuk proses selanjutnya. Adapun langkah-langkah *preprocessing* yang digunakan pada penelitian ini adalah sebagai berikut.

1. *Case Folding*

   Case Folding adalah proses penyamaan *case* dalam sebuah dokumen (Alita & Rahman, 2020). Dalam sebuah teks data yang disajikan tidak semuanya memiliki *case* huruf yang sama. Perbedaan *case* tersebut berpengaruh ketika memasuki proses TF dan TF/IDF, maka dari itu dibutuhkan *preprocessing case folding* untuk merubah *case* huruf menjadi sama yaitu menjadi huruf kecil.

2. *Tokenizing*

   Tokenizing adalah proses pemotongan sebuah dokumen menjadi bagian-bagian, yang disebut token (Alita & Rahman, 2020). Pemotongan pada *tokenizing* dilakukan berdasarkan unsur pemisah seperti spasi, enter, dan koma. Proses *tokenizing* bukan hanya memotong dokumen menjadi *token*, tetapi proses ini juga dapat membuang karakter tertentu yang dianggap tanda baca.

3. *Filtering*

   Filtering adalah proses menghilangkan beberapa kata yang dianggap kurang perlu atau tidak penting berdasarkan *stopwords* (Hermawan et al.,

2020). *Stopwords* biasanya berisi kata ganti orang, kata hubung, dan masih banyak lagi.

2.2.8 *Term Frequency and Inverse Document Frequency* (TF-IDF)

Algoritme TF-IDF digunakan dalam konten apa pun untuk menimbang kata kunci dan atribut penting untuk kata kunci itu berdasarkan berapa kali muncul dalam dokumen. TF-IDF atau *term frequency and inverse document frequency* merupakan salah satu metode pemberian bobot pada suatu hubungan kata yang menggabungkan dua cara yaitu *term frequency* dan *inverse document frequency* (Hormansyah & Utama, 2018). *Term frequency* (TF) melakukan pembobotan berdasarkan jumlah kemunculan sebuah kata dalam dokumen, sedangkan *inverse document frequency* (IDF) melakukan pengurangan *term* yang sering muncul dalam dokumen karena *term* yang sering muncul dianggap kata umum atau tidak penting. Menurut (Hormansyah & Utama, 2018) formula yang digunakan untuk menghitung TF dan IDF terdapat pada persamaan (1) - (3).

$$TF = term\ frequency \tag{1}$$

$$IDF = \log \frac{N}{df} \tag{2}$$

$$TF - IDF = tf\ x\ IDF \tag{3}$$

Pada persamaan (3) menunjukkan bahwa cara mencari nilai TF-IDF adalah dengan cara melakukan perkalian antara TF dan IDF. Nilai TF didapat dari banyak kata yang dicari pada sebuah dokumen. Nilai IDF didapat dari hasil logaritma pembagian total dokumen dibagi dengan banyak dokumen yang mengandung kata yang dicari. Semakin besar TF-IDF dari sebuah kata maka semakin besar juga tingkat similaritas dokumen terhadap kata yang dicari, begitu pun sebaliknya.

2.2.9 *Cosine Similarity*

Cosine Similarity adalah model yang dipakai dalam mencari tingkat *similarity* antara objek data dengan kata kunci (Hima Pandalu, 2014). Metode ini didasari pada *vector space similarity measure*, dalam kata lain metode *cosine* menghitung nilai *cosinus* antara dua vektor dengan menentukan kemiripan antara dua buah objek yang dibandingkan dengan nilai terbesar adalah 1 dan nilai terkecil adalah

0. Apabila nilai dari sebuah proses mendekati 1 menandakan bahwa dokumen yang dibandingkan dengan data memiliki kemiripan yang tinggi, dan semakin angka mendekati nilai 0 maka dokumen tersebut memiliki tingkat kemiripan yang kecil.

Setiap vektor dalam *cosine similarity* akan mewakili setiap kata dalam setiap dokumen yang dibandingkan dan akan membentuk segitiga sehingga hukum *cosinus* dapat diterapkan untuk mengatakan bahwa sebuah dokumen identik atau tidak identik . (Hima Pandalu, 2014) mendefinisikan rumus fungsi perhitungan *cosine similarity* pada persamaan (4).

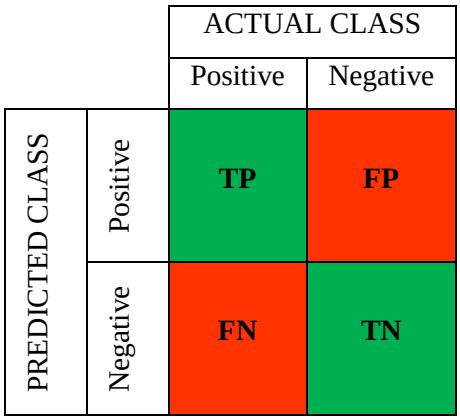
$$\cos(A, B) = \frac{A \cdot B}{|A||B|} = \frac{\sum_{i=1}^{n} A_i \; x \; B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \; x \; \sqrt{\sum_{i=1}^{n} (AB_i)^2}} \qquad (4)$$

Pada persamaan (4) mendefinisikan cara menghitung sudut antar 2 objek menggunakan *cosinus* dengan cara mencari *dot product* antara vektor A dan vektor B kemudian dibagi dengan hasil perkalian dari panjang vektor A dan panjang vektor B.

2.2.10 *Confusion Matrix*

Confusion matrix didefinisikan sebagai matriks yang menyediakan perbandingan dari prediksi dengan kenyataan yang digunakan sebagai pengukur performa dari sebuah algoritme klasifikasi (Markoulidakis et al., 2021).

Dimensi yang dipakai dalam *confusion matrix* untuk *binary classification problem* adalah matriks berdimensi 2 * 2 dengan label positif dan negatif. Kolom dalam matriks merepresentasikan kejadian dari prediksi, sedangkan baris merepresentasikan kenyataan. Elemen dalam *confusion matrix* berdasarkan dari label prediksi (positif, negatif) dan hasil komparasi prediksi dengan kejadian kenyataan dapat dilihat pada Gambar 4.



Gambar 4. *Confusion Matrix*.

Gambar 4 terdapat 4 elemen yaitu *True Positive* (TP), *False Positive* (FP), *False Negative* (FN), dan *True Negative* (TN). TP adalah nilai kejadian kenyataan dan prediksi bernilai positif. FP adalah nilai kejadian kenyataan bernilai negatif tetapi bernilai positif pada prediksi. FN adalah nilai kejadian kenyataan bernilai positif tetapi bernilai negatif pada prediksi. TN adalah nilai kejadian kenyataan dan prediksi bernilai negatif. Gambar 5 mengilustrasikan bagaimana keempat elemen dalam *confusion matrix* bekerja.

		ACTUAL CLASS	
		Positive	Negative
PREDICTED CLASS	Positive		
	Negative		

Gambar 5. Ilustrasi Elemen *Confusion Matrix.*
Sumber : https://yassineelkhal.medium.com/confusion-matrix-auc-and-roc-curve-and-gini-clearly-explained-221788618eb2

Keempat elemen *confusion matrix* digunakan untuk mengukur performa dari klasifikasi model menggunakan beberapa metrik (Tharwat, 2018). Beberapa metrik yang dapat diterapkan menggunakan nilai dari *confusion matrix* adalah *sensitivity*, *specificity*, *precision*, *accuracy*, *f1 score*, dll (Markoulidakis et al., 2021). Metrik performa memiliki nilai dari rentang 0 sampai dengan 1 dapat dilihat pada Tabel 3.

Tabel 3. Metrik Performa

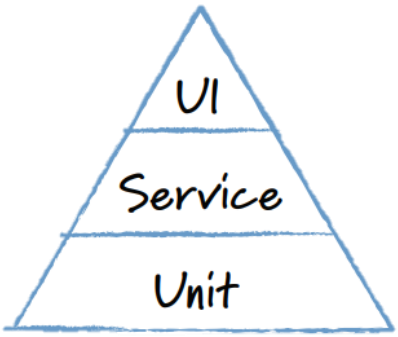
Simbol	Metrik	Formula
ACC	Accuracy	$\frac{TP + TN}{TP + FN + TN + FP}$
SNS	Sensitivity/Recall	$\frac{TP}{TP + FN}$

Simbol	Metrik	Formula
PRC	Precision	$\frac{TP}{TP + FP}$
SPS	Specificity	$\frac{TN}{TN + FP}$
F_1	F_1 score	$2\frac{PRC \times SNS}{PRC + SNS}$

Pada Tabel 3 menunjukkan 5 metrik pengukuran performa sebuah model klasifikasi. Metrik pertama yaitu *accuracy*, *accuracy* mewakili jumlah dari data yang diklasifikasikan dengan benar dibandingkan dengan jumlah total data. *Sensitivity* disebut juga sebagai *recall* atau *true positive rate* mewakili hasil perbandingan antara TP dengan jumlah data yang kejadian sebenarnya yang bernilai positif. *Precision* atau *positive predictive value* mewakili hasil perbandingan antara TP dengan jumlah data yang diprediksi bernilai positif. *Specificity* merupakan metrik evaluasi keefektifan F_1 *score* mencerminkan keseimbangan antara *precision* dan *recall*.

2.2.11 *Unit Test*

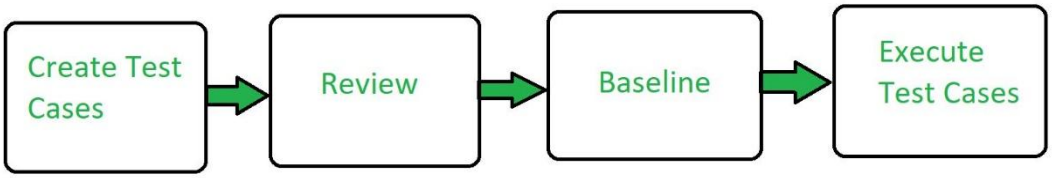
Unit Test adalah proses verifikasi bagian terkecil dari sebuah *code* (dikenal sebagai *unit*) dilakukan secara cepat dan terisolasi (Khorikov, 2020). Proses *unit test* memastikan bahwa setiap fungsi dalam *code* menjalankan tugasnya masing-masing. (Cohn, 2010) mengemukakan *unit test* merupakan dasar *pyramid* pengujian otomatis lihat pada Gambar 6. Piramida yang dikemukakan Mike Cohn berawal dari *unit test*, *service test*, dan *UI test*.



Gambar 6. Piramida Pengujian Otomatis.
Sumber : (Cohn, 2010)

Keuntungan dari penggunaan *unit test* adalah memungkinkan pengujian dan *debugging* pada bagian terkecil sehingga memberikan cara yang lebih baik untuk mengintegrasikan unit ke dalam skala yang lebih besar. *Unit test* dapat memfasilitasi pengujian otomatis karena *behaviour* dari unit terkecil dapat direkam dan diputar ulang dengan penggunaan kembali yang lebih maksimal. Dukungan pengujian secara matematis pengujian logika dari *code* memungkinkan pengujian dapat dilakukan menggunakan skenario yang lebih sedikit.

Cara kerja *unit test* pada Gambar 7 diawali dengan pembuatan *test case*, tinjau ulang *case*, menetapkan standar atau *baseline*, dan eksekusi *test case*. Penggunaan *unit test* pada tiap *code* bergantung kepada bahasa yang digunakan. Pada penelitian ini menggunakan PHP dan Python dapat menggunakan PHPUnit dan testclient.



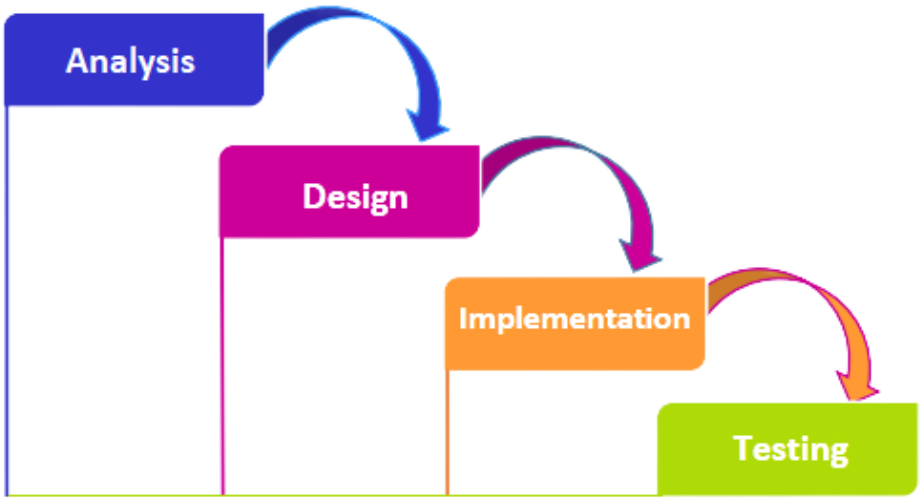
Gambar 7. Alur Kerja *Unit Testing*.

Proses *unit test* menggunakan *terminal* atau *command line* untuk mengeksekusi skenario atau *case* pengujian. *Output* yang dihasilkan dari *unit test* baik menggunakan PHPUnit maupun testclient adalah HTTP Status code, lihat pada Tabel 2.

III. METODOLOGI PENELITIAN

3.1 Desain Penelitian

Penelitian ini dilaksanakan menggunakan salah satu model pengembangan perangkat lunak yaitu *waterfall model*. *Waterfall model* terbagi menjadi 4 tahapan yaitu *analysis*, *designing*, *implementation*, dan *testing*. Gambar 8 mengilustrasikan desain penelitian yang menggunakan *waterfall model*.



Gambar 8. Desain Penelitian.

3.1.1 Tahap Analisis (*Analysis Phase*)

Tahap analisis lebih dikenal dengan *software requirements spesification* (SRS) merupakan deskripsi lengkap dan *comprehensive* dari perangkat lunak yang akan dikembangkan. Analisis melibatkan sistem dan bisnis analisis untuk mendefinisikan kedua persyaratan fungsional maupun non fungsional.

Persyaratan fungsional mendefinisikan tujuan, ruang lingkup, perspektif, fungsi, atribut perangkat lunak, karakteristik pengguna, spesifikasi fungsionalitas, persyaratan antarmuka, dan persyaratan basis data. Persyaratan non-fungsional merujuk kepada berbagai kriteria, kendala, batasan dan persyaratan yang diberlakukan terhadap desain dan pengoperasian perangkat lunak, ketimbang perilaku spesifik yang diinginkan. Cakupan dari persyaratan non-fungsional berupa keandalan, *scalability*, *testability*, ketersediaan, pemeliharaan, kinerja, dan standar kualitas.

Langkah yang diperlu dilakukan untuk tahap analisis adalah mencari kebutuhan fungsi, dan kebutuhan data dari perangkat lunak. Pencarian kebutuhan dari perangkat lunak dapat dilakukan dengan melakukan observasi, wawancara, dan analisis dokumen. Kebutuhan fungsi dan kebutuhan data yang sudah siap dapat mempermudah tahapan selanjutnya dalam metode *waterfall*.

Data mentah yang sudah terkumpul akan diubah atau dilakukan *preprocessing* untuk menjadi data matang yang siap diproses. Proses *preprocessing* dilakukan dengan 5 cara diantaranya.

1. *Case Folding*

   Tahap *case folding* bertujuan untuk membuat semua karakter dalam undang-undang menjadi huruf kecil (*lowercase*), menghapus angka, menghapus tanda baca, dan menghapus *whitespace*. Ilustrasi tahap *case folding* terdapat pada Tabel 4.

Tabel 4. Ilustrasi *Case Folding*

Sebelum *case folding*	**Sesudah *case folding***
Kepala Daerah berwenang menunjuk Pejabat untuk penagihan pajak daerah.	kepala daerah berwenang menunjuk pejabat untuk penagihan pajak daerah

Pada Tabel 3, terdapat data sebelum dan sesudah *case folding*. Perbedaan dari kedua kolom tersebut adalah kolom pertama terdapat huruf kapital, sedangkan kolom kedua sudah tidak ada huruf kapital.

2. *Tokenizing*

   Tahap *tokenizing* bertujuan untuk memetakan kata berdasarkan penyusunannya ke dalam sebuah *index array*. Ilustrasi tahap *tokenizing* terdapat pada Tabel 5.

Tabel 5. Ilustrasi *Tokenizing*

Sebelum *tokenizing*	**Sesudah *tokenizing***
kepala daerah berwenang menunjuk pejabat untuk penagihan pajak daerah	[kepala,daerah,berwenang,menunjuk,pejabat,untuk,penagihan,pajak,daerah]

Pada Tabel 5, terdapat data sebelum dan sesudah *tokenizing*. Perbedaan dari kedua kolom tersebut adalah kolom pertama belum dipetakan ke dalam sebuah *index array*, sedangkan kolom kedua sudah dipetakan ke dalam sebuah *index array* yang dipisahkan oleh tanda baca koma.

3. *Filtering*

   Tahap *filtering* bertujuan untuk membuang kata yang sering muncul dan tidak memiliki makna seperti “yang”, “dan”, “untuk” dll. Ilustrasi tahap *filtering* terdapat pada Tabel 6.

Tabel 6. Ilustrasi *Filtering*

Sebelum *filtering*	**Sesudah *filtering***
[kepala,daerah,berwenang,menunjuk,pejabat,untuk,penagihan,pajak,daerah]	[kepala,daerah,berwenang,pejabat,penagihan,pajak,daerah]

Pada Tabel 6, terdapat data sebelum dan sesudah *filtering*. Perbedaan dari kedua kolom tersebut jika pada kolom pertama terdapat kata “untuk”, pada kolom kedua “untuk” dihapuskan karena termasuk kata yang umum.

3.1.2 Tahap Desain (*Design Phase*)

Tahap desain merupakan rencana dan *problem solving* dari sebuah solusi perangkat lunak. Tahap desain melibatkan pengembang perangkat lunak untuk mendefinisikan rencana dari solusi yang termasuk desain algoritme, desain arsitektur perangkat lunak, skema konseptual basis data dan desain diagram logika, desain konsep, tampilan grafis antarmuka, dan definisi struktur data.

Pembuatan desain algoritme, dan desain arsitektur dibantu menggunakan perangkat lunak pengolah gambar seperti Paint, Adobe Photoshop, maupun Corel Draw. Pembuatan skema konseptual, definisi struktur data dapat dilakukan menggunakan ERD draw maupun perangkat lunak pembuat *chart* lainnya, untuk perancangan antarmuka dapat menggunakan Balsamiq.

3.1.3 Tahap Implementasi (*Implementation Phase*)

Tahap implementasi merupakan tahapan realisasi dari persyaratan bisnis dan desain spesifikasi menjadi konkret program yang bisa dieksekusi, basis data, *website*, atau komponen perangkat lunak melalui *programming*. Tahap implementasi menggunakan *real code* yang ditulis dan disusun menjadi aplikasi operasional, yang mana basis data dan teks sudah dibuat, dalam kata lain tahap implementasi merupakan proses mengonversi seluruh persyaratan dan tinta biru menjadi *production environment*.

3.1.4 Tahap Pengujian (*Testing Phase*)

Tahap pengujian dikenal dengan verifikasi dan validasi yang mana proses untuk pengecekan bahwa perangkat lunak sudah memenuhi persyaratan dan spesifikasi untuk mencapai tujuan dari pembuatan perangkat lunak. Verifikasi digunakan untuk mengevaluasi perangkat lunak untuk menentukan produk dari fase pengembangkan memenuhi kondisi yang diberlakukan pada awal tahapan model *waterfall*. Validasi digunakan untuk mengevaluasi perangkat lunak selama atau pada akhir proses pengembangan untuk menentukan apakah memenuhi persyaratan yang ditentukan. Tahapan pengujian ditujukan untuk menjadi jalan keluar untuk *debugging* yang memungkinkan untuk menemukan *bug* dan gangguan dalam sistem sehingga dapat diperbaiki, dan disempurnakan.

3.2 Peralatan Penelitian

Peralatan yang digunakan dalam penelitian ini sebagai berikut.

1. Perangkat Keras
   a. *System Manufactured* : Acer
   b. *System Model* : Swift 3 SF314-41
   c. *Processor* : AMD Ryzen 3500U
   d. *Installed RAM* : 12288 MB
   e. *System Type* : 64-bit *operating system*
2. Perangkat Lunak
   a. Sistem Operasi : Windows 11 Home
   b. *Code Editor* : Visual Studio Code

3.3 Waktu dan Tempat Penelitian

Penelitian ini dilakukan di Jurusan Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung pada semester genap Tahun Ajaran 2021/2022.

V. SIMPULAN DAN SARAN

5.1 Simpulan

Berdasarkan hasil penelitian mengenai pengembangan modul *drafting* pada sistem informasi omnibus *law*, maka penulis menarik simpulan sebagai berikut.

1. Tingkat kesamaan antara kata kunci pembuatan rancangan undang-undang dan undang-undang yang sudah ada, dapat diukur menggunakan algoritme *cosine similarity*, dengan hasil *recall* dari 500 data uji sebesar 90,10%.
2. Penerapan model tingkat kesamaan antara kata kunci dan undang-undang yang sudah ada agar dapat digunakan oleh *user* (*legal drafter*) dengan menggunakan FastAPI sebagai *back-end* dan Laravel sebagai *front-end*. Penerapan pada FastAPI dan Laravel sudah dilakukan uji coba menggunakan *unit test* yang menghasilkan hasil tes sukses pada tiap skenario pengujiannya.

5.2 Saran

Berdasarkan hasil penelitian yang diperoleh , penulis mengajukan beberapa saran untuk pengembangan lebih lanjut dalam topik penelitian ini.

1. Penelitian mendukung penerapan metode lain seperti *jaccard*, *path*, dan k-nearest *neighbor*. Oleh karena itu, penelitian lanjutan dapat difokuskan untuk meningkatkan hasil *recall* dari model dengan menerapkan beberapa metode yang sudah disebutkan.
2. Mengingat temuan bahwa FastAPI dan Laravel dapat digunakan untuk menerapkan model tingkat kesamaan antar kata walaupun dengan data yang besar, disarankan untuk melakukan optimalisasi kinerja fitur dengan menambahkan *caching* atau sejenisnya.

DAFTAR PUSTAKA

Al Hidayat, N. (2017). Implementasi Legal Drafting dalam Proses Penyusunan Peraturan Daerah Kabupaten/Kota (Studi Pada Sekretariat Dewan Perwakilan Rakyat Daerah (DPRD) Kabupaten Bungo). *Jurnal Serambi Hukum*, *11*(01), 69–95.

Alita, D., & Rahman, A. (2020). Pendeteksian Sarkasme pada Proses Analisis Sentimen Menggunakan Random Forest Classifier. *Jurnal.Fmipa.Unila.Ac.Id*, *8*. https://jurnal.fmipa.unila.ac.id/komputasi/article/view/2615

Anggraeni, E. Y., & Irviani, R. (2017). *Pengantar Sistem Informasi*. CV. Andi Offset. https://books.google.co.id/books?id=8VNLDwAAQBAJ&printsec=frontcover#v=onepage&q&f=false

Azhar, M. (2019). Omnibus Law sebagai Solusi Hiperregulasi Menuju Sonkronisasi Peraturan Per-Undang-undangan di Indonesia. *Administrative Law and Governance Journal*, *2*(1), 170–178. https://doi.org/10.14710/alj.v2i1.170-178

Azhari, R. Y. (2022). Web Service Framework : Flask DAN FastAPI. *Technology and Informatics Insight Journal*, *1*(1), 58–65. https://doi.org/10.32639/TIIJ.V1I1.54

Azmi, M. (2022). Analisis Tingkat Plagiasi Dokumen Skripsi Dengan Metode Cosine Similarity Dan Pembobotan Tf-Idf. *TEKNIMEDIA: Teknologi Informasi Dan Multimedia*, *2*(2), 90–95. https://doi.org/10.46764/teknimedia.v2i2.51

Cohn, M. (2010). *Succeding With Agile : Software Development Using Scrum* (K. Gettman (ed.); 2nd ed.). Pearson Education, Inc.

Hermawan, L., Ismiati, M. B., Bangau, J., 60, N., & Charitas, M. (2020). Pembelajaran text preprocessing berbasis simulator untuk mata kuliah information retrieval. *156.67.218.228*, *17*(2), 188–199. https://156.67.218.228/index.php/transformatika/article/view/1705

Hima Pandalu, P. K. (2014). *Pencarian dan Perankingan Obat Tradisional Berdasarkan Gejala Penyakit Menggunakan Metode Cosine Similarity*. Universitas Islam Negeri Maulana Malik Ibrahim Malang.

Hormansyah, D. S., & Utama, Y. P. (2018). Aplikasi Chatbot Berbasis Web pada Sistem Informasi Layanan Publik Kesehatan di Malang dengan Menggunakan Metode TF-IDF. *Jip.Polinema.Ac.Id*, *4*. http://jip.polinema.ac.id/ojs3/index.php/jip/article/view/211

Khorikov, V. (2020). *Unit Testing : Principles, Practices, and Patterns*. Manning Publications Co.

Kurniawan, A., Solihin, F., & Hastarita, F. (2014). Perancangan Dan Pembuatan Aplikasi Pencarian Informasi Beasiswa dengan Menggunakan Cosine Similarity. *Jurnal Simantec*, *4*(2). https://doi.org/10.21107/SIMANTEC.V4I2.1392

Luthfi, F. (2017). Penggunaan Framework Laravel dalam Rancang Bangun Modul Back-End Artikel Website Bisnisbisnis.ID. *JISKA (Jurnal Informatika Sunan Kalijaga)*, *2*(1), 34–41. https://doi.org/10.14421/jiska.2017.21-05

Maholtra, M. (2022). *Internship Report At Paxcom India Pvt*.

Markoulidakis, I., Rallis, I., Georgoulas, I., Kopsiaftis, G., Doulamis, A., & Doulamis, N. (2021). Multiclass Confusion Matrix Reduction Method and Its Application on Net Promoter Score Classification Problem. *Technologies*, *9*(4). https://doi.org/10.3390/technologies9040081

Mayasari, I. (2020). Kebijakan Reformasi Regulasi Melalui Implementasi Omnibus Law Di Indonesia. *Ima Mayasari*, *9*(1). https://www.doingbusiness.org/en/rankings

Muqsith, M. A. (2020). UU Omnibus law yang Kontroversial. *'Adalah : Buletin Hukum & Keadilan*, *4*(3), 109–115. https://doi.org/10.15408/adalah.v4i3.17926

Pratama, R. P., Faisal, M., & Hanani, A. (2019). Deteksi Plagiarisme pada Artikel Jurnal Menggunakan Metode Cosine Similarity. *SMARTICS Journal*, *5*(1), 22–26. https://doi.org/10.21067/smartics.v5i1.2848

Purnama Sari, D., & Wijanarko, R. (2020). Implementasi Framework Laravel pada Sistem Informasi Penyewaan Kamera (Studi Kasus di Rumah Kamera Semarang). *Jurnal Informatika Dan Rekayasa Perangkat Lunak*, *2*(1), 32. https://doi.org/10.36499/jinrpl.v2i1.3190

Riyani, A., Naf'an, M. Z., & Burhanuddin, A. (2019). Penerapan Cosine Similarity dan Pembobotan TF-IDF untuk Mendeteksi Kemiripan Dokumen. *Jurnal Linguistik Komputasional*, *2*(1), 23–27. https://doi.org/10.26418/jlk.v2i1.17

Ruslan, A. (2011). *Teori dan Panduan Praktik Pembentukan Peraturan Perundang-undangan di Indonesia*. Rangkang Education.

Sadiawati, D. (2015). *Strategi Nasional Reformasi Regulasi: Mewujudkan Regulasi yang Tertib dan Sederhana*. Kementerian Perencanaan dan Pembangunan Nasional/ Bappenas.

Samuel Rajiv, C., & K, P. (2022). A Platform to Help in Generating Code for

Machine Learning and Data Science Projects. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.4033072

Setiadi, W. (2020). Simplifikasi Regulasi dengan Menggunakan Metode Pendekatan Omnibus Law. *Jurnal Rechts Vinding: Media Pembinaan Hukum Nasional*, *9*(1), 39. https://rechtsvinding.bphn.go.id/ejournal/index.php/jrv/article/view/408

Sholichah, I. U. (2021). Eksistensi Legal Drafting Hukum Islam Di Indonesia. *Syar'Ie*, *4*(2), 95–107.

Silvi Lydia, M., Dara Fadillah, S., Miftahul Huda VOLUME, dan, & Rekayasa Elektrika, J. (2018). Perbandingan Metode Klaster dan Preprocessing Untuk Dokumen Berbahasa Indonesia. *Pdfs.Semanticscholar.Org*. https://doi.org/10.17529/jre.v14i1.9027

Supriyanti, D., Purnomo, A. C., & Ismudiarti, A. (2020). Rancang Bangun Sistem Manajemen Legal Drafting Untuk Mendukung Tertib Administrasi. *Journal CERITA*, *6*(1), 12–25. https://doi.org/10.33050/cerita.v6i1.883

Tharwat, A. (2018). Classification assessment methods. *Applied Computing and Informatics*, *17*(1), 168–192. https://doi.org/10.1016/j.aci.2018.08.003

Zakiyudin, A. (2011). *Sistem Informasi Manajemen*. Mitra Wacana Media.