PENGEMBANGAN SERVICE HARMONISASI PADA APLIKASI OMNIBUS LAW

(Skripsi)

Oleh

AHMAD JULIO RIZKI
1817051042

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2022

PENGEMBANGAN SERVICE HARMONISASI PADA APLIKASI OMNIBUS LAW

Oleh

AHMAD JULIO RIZKI

Skripsi

Sebagai Salah Satu Syarat untuk Mendapat Gelar
SARJANA KOMPUTER

Pada

Jurusan Ilmu Komputer
Fakultas Matematika dan Ilmu Pengetahuan Alam

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2022

ABSTRAK

PENGEMBANGAN SERVICE HARMONISASI PADA APLIKASI OMNIBUS LAW

Oleh

AHMAD JULIO RIZKI

Aplikasi Omnibus Law adalah aplikasi yang digunakan oleh seorang Legal Drafter yang berfungsi untuk membantu dalam pembuatan rancangan undang-undang. Saat ini undang-undang di Indonesia sudah sangat banyak dan saling tumpang tindih. Penelitian ini bertujuan untuk mengembangkan sebuah service yang dapat digunakan untuk membantu dalam penyusunan undang-undang baru agar tidak terjadi tumpang tindih peraturan. Metode dalam pengembangan penelitian ini menggunakan metode waterfall. Penelitian ini menerapkan gaya arsitektur REST saat mengembangkan API sebagai *backend* untuk aplikasi Omnibus Law dan menggunakan *Javascript Object Notation* (JSON) sebagai format standar untuk komunikasi dan transfer data. Service dibuat menggunakan bahasa pemrograman *python* dengan *framework* FastApi. Pada penelitian ini service yang dihasilkan akan dibuat dokumentasi yang bertujuan untuk mempermudah dan membantu developer dalam menggunakan API yang dihasilkan oleh service tersebut, dokumentasi akan dibuat menggunakan Swagger. Service yang dihasilkan akan diuji menggunakan metode Black-Box Testing dengan metode pengujian Equivalence Partitioning dan software atau tools yang akan digunakan dalam pengujian ini yaitu Postman. Penelitian ini menghasilkan sebuah layanan yang akan digunakan oleh aplikasi Omnibus Law yang dapat membantu Legal Drafter dalam mendeteksi adanya tumpang tindih peraturan dalam undang-undang.

Kata Kunci: REST, API, FastApi, JSON, Swagger.

ABSTRACT

HARMONIZATION SERVICE DEVELOPMENT IN OMNIBUS LAW APPLICATIONS

By

AHMAD JULIO RIZKI

The Omnilaw application is an application used by Legal Drafters to assist in the making and drafting of laws. There are many regulations in Indonesia today, and there are overlapping regulations. This study aims to develop a service that can be used to assist in the drafting of new laws to avoid overlapping regulations. The method in the development of this research uses the waterfall method. This study applies the REST architectural style when developing the API as a backend for the Omnilaw application and uses Javascript Object Notation (JSON) as the standard format for communication and data transfer. Service development is carried out using the Python programming language with the FastApi framework. In this study, the resulting service will be made documentation that aims to simplify and assist developers in using the API generated by the service, documentation will be created using Swagger. The resulting service will be tested using the Black-Box Testing method with Equivalence Partitioning as the test method and the software or tools that will be used in this test, namely Postman. This research produces a service that will be used by the Omnilaw application that can assist Legal Drafters in detecting overlapping regulations in the law.

Keywords: REST, API, FastApi, JSON, Swagger.

Judul Skripsi	: **PENGEMBANGAN SERVICE HARMONISASI PADA APLIKASI OMNIBUS LAW**
Nama Mahasiswa	: Ahmad Julio Rizki
Nomor Induk Mahasiswa	: **1817051042**
Program Studi	: **S1 Ilmu Komputer**
Jurusan	: **Ilmu Komputer**
Fakultas	: **Matematika dan Ilmu Pengetahuan Alam**

MENYETUJUI

1. Komisi Pembimbing

Aristoteles, S.Si., M.Si.
NIP 19810521 200604 1 002

2. Ketua Jurusan Ilmu Komputer

Didik Kurniawan, S.Si., M.T.
NIP 19800419 200501 1 004

MENGESAHKAN

1. Tim Penguji

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

   Penguji Pembahas : **Rudy, S.H., LL.M., LL.D.**

   Penguji Pembahas : **Drs. Rd. Irwan Adipribadi, M.Kom.**

2. Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam

Dr Eng. Suripto Dwi Yuwono, S.Si., M.T.
NIP 19740705 200003 1 001

Tanggal Lulus Ujian Skripsi : **11 Agustus 2022**

PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : Ahmad Julio Rizki

NPM : 1817051042

Dengan ini menyatakan bahwa skripsi saya yang berjudul **"Pengembangan Service Harmonisasi Pada Aplikasi Omnibus Law"** merupakan karya saya sendiri dan bukan karya orang lain. Semua tulisan yang tertuang dalam skripsi ini telah mengikuti kaidah penulisan karya ilmiah Universitas Lampung. Apabila di kemudian hari terbukti skripsi saya merupakan hasil penjiplakan atau dibuat orang lain, maka bersedia menerima sanksi berupa pencabutan gelar yang telah saya terima.

Bandar Lampung, 26 Agustus 2022

Ahmad Julio Rizki

NPM. 1817051042

RIWAYAT HIDUP

Penulis dilaharikan di Bandar Lampung, pada tanggal 22 Juli 2000, sebagai anak ketiga dari tiga bersaudara.

Penulis menyelesaikan pendidikan formal di SDN 1 Perumnas Way Halim dan selesai pada tahun 2012. Kemudian pendidikan menengah pertama di SMPN 20 Bandar Lampung yang diselesaikan pada tahun 2015, lalu melanjutkan ke pendidikan menengah atas di SMAN 5 Bandar Lampung yang diselesaikan pada tahun 2018.

Pada tahun 2018 penulis terdaftar sebagai mahasiswa Jurusan Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung melalui jalur SBMPTN. Selama menjadi mahasiswa, penulis melakukan beberapa kegiatan antara lain.

1. Menjadi anggota Adapter Himpunan Mahasiswa Jurusan Ilmu Komputer pada periode 2018/2019.
2. Menjadi Asisten Dosen Jurusan Ilmu Komputer tahun 2020 hingga 2021.
3. Mengikuti ujian sertifikasi dan mendapat sertifikat *Junior Web Developer* oleh Lembaga Sertifikasi Profesi Informatika pada tahun 2021.

4. Melaksanakan Kerja Praktek pada bulan Februari periode 2020/2021 di KSOP Kelas 1 Panjang.
5. Melaksanakan kegiatan Kuliah Kerja Nyata (KKN) di Desa Bumi Raya, Kecamatan Bumi Waras, Kota Bandar Lampung, Lampung pada tahun 2020/2021.

MOTTO

“Jika kamu tidak sanggup menahan lelahnya belajar maka kamu harus sanggup menahan perihnya kebodohan.”

(Imam Syafi'i)

“Jangan takut mencoba hal yang baru, gapailah impianmu. Tapi ingatlah, tak peduli kemana kamu pergi, keluarga tempatmu kembali.”

(Anonim)

“Jika kamu ingin hidup yang sulit maka kamu harus membuat pilihan yang mudah.”

(Anonim)

“Work in silence and let success make the noise.”

(Anonim)

PERSEMBAHAN

Alhamdulillahirobbilalamin

Puji dan syukur tercurahkan kepada Allah Subhanahu Wa Ta'alaa atas segala Rahmat dan Karunia-Nya sehingga saya dapat menyelesaikan skripsi ini. Shalawat serta salam selalu tercurahkan kepada Nabi Muhammad SAW.

Kupersembahkan karya ini kepada:

Kedua Orang Tuaku Tercinta

Yang senantiasa memberikan yang terbaik, dan melantunkan do'a yang selalu menyertaiku. Kuucapkan pula terima kasih sebesar-besarnya karena telah mendidik dan membesarkanku dengan cara yang dipenuhi kasih sayang, dukungan, dan pengorbanan yang belum bisa terbalaskan.

Seluruh Keluarga Besar Ilmu Komputer 2018

Yang selalu memberikan semangat dan dukungan.

Almamater Tercinta, Universitas Lampung dan Jurusan Ilmu Komputer

Tempat bernaung mengemban semua ilmu untuk menjadi bekal hidup.

SANWACANA

Puji syukur kehadirat Allah SWT atas berkah, rahmat dan hidayat-Nya, serta petunjuk dan pedoman dari Rasulullah Nabi Muhammad Sholallahu Alaihi Wasallam penulis dapat menyelesaikan skripsi yang berjudul "Pengembangan Service Harmonisasi Pada Aplikasi Omnibus Law" dengan baik dan lancar.

Terima kasih penulis ucapkan kepada semua pihak yang telah membantu dan berperan besar dalam menyusun skripsi ini, antara lain.

1. Kedua orang tua tercinta yang selalu memberi dukungan, do'a, semangat, motivasi, dan kasih sayang yang luar biasa tak terhingga. Semua yang telah kalian berikan tidak akan pernah mampu untukku balas. Semoga Allah SWT selalu memberikan kebahagiaan dan keberkahan dalam kehidupan kalian di dunia dan akhirat.
2. Bapak Aristoteles, S.Si., M.Si. sebagai pembimbing utama yang telah memberikan arahan, ide, kritik serta saran kepada penulis sehingga dapat menyelesaikan skripsi ini dengan baik.
3. Bapak Rudy, S.H., LL.M., LL.D. sebagai pembahas yang juga selalu dapat memberikan waktu untuk membimbing penulis dalam memberikan ide, kritik serta saran untuk dapat menyelesaikan skripsi ini.
4. Bapak Drs. Rd. Irwan Adipribadi, M.Kom. sebagai pembahas yang telah memberikan masukan yang bermanfaat dalam perbaikan skripsi ini.
5. Bapak Favorisen R. Lumbanraja, Ph.D. selaku pembimbing akademik penulis yang selalu mendukung peningkatan akademik penulis.
6. Bapak Didik Kurniawan, S.Si., M.T. selaku ketua Jurusan Ilmu Komputer FMIPA Universitas Lampung.

7. Bapak Dr. Eng. Suripto Dwi Yuwono, S.Si., M.T. selaku Dekan FMIPA Universitas Lampung.
8. Bapak Dr. rer. nat. Akmal Junaidi, M.Sc. selaku Sekretaris Jurusan Ilmu Komputer FMIPA Universitas Lampung.
9. Ibu Ade Nora Maela dan Bang Zainuddin yang telah membantu segala urusan administrasi penulis di Jurusan Ilmu Komputer.
10. Bapak dan Ibu Dosen Jurusan Ilmu Komputer FMIPA Universitas Lampung yang telah memberikan ilmu dan pengalaman dalam hidup untuk menjadi lebih baik.
11. Kakak Ayu Puspita Sari, dan Abang Robi Setiawan sebagai kakak kandung yang telah mendoakan, menyemangati, dan mendukung selama ini.
12. Teman-teman terdekat, Arsyi, Nabil, Lita, Rahma, Sasya, dan Amara yang telah memberikan banyak dukungan moril, segala bentuk bantuan, dan selalu menemani dari awal perkuliahan.
13. Keluarga Ilmu Komputer 2018 yang tidak bisa penulis sebut satu persatu. Keluarga kedua penulis, rekan kelompok, rekan diskusi, rekan bercanda, dan telah memberi arti dan warna serta pengalaman tak ternilai semasa duduk di bangku kuliah.
14. Seluruh kakak tingkat dan adik tingkat Ilmu Komputer yang tidak bisa disebutkan satu persatu yang telah menjadi warna selama masa perkuliahan penulis.

Penulis menyadari bahwa skripsi ini masih jauh dari kata sempurna, semoga skripsi ini membawa manfaat dan keberkahan bagi semua civitas Ilmu Komputer Universitas Lampung aamiin ya rabbal aalamiin.

Bandar Lampung, 26 Agustus 2022

Ahmad Julio Rizki

NPM. 1817051042

DAFTAR ISI

Halaman

DAFTAR TABEL

Tabel **Halaman**

DAFTAR GAMBAR

Gambar **Halaman**

I. PENDAHULUAN

1.1. Latar Belakang

Undang-Undang merupakan peraturan negara yang memiliki kekuatan hukum yang mengikat, Indonesia menerbitkan undang-undang baru tiap tahunnya, sehingga terjadinya penumpukan regulasi. Jumlah regulasi yang ada di Indonesia saat ini sudah terlalu banyak dan saling tidak harmonis, saling konflik, serta saling tumpang tindih antara regulasi satu dengan yang lainnya atau *overregulated* (Setiadi, 2020). Dampak dari *overregulated* itu sendiri mengakibatkan pergerakan ekonomi di Indonesia menjadi lamban sehingga kontraproduktif dengan kebijakan dalam lingkup ASEAN (Malau, 2014). Sebuah sistem dikatakan baik jika tidak terdapat suatu pertentangan antara bagian-bagian. Selain itu juga tidak boleh terdapat duplikasi atau tumpang tindih diantara bagian-bagian itu (Nurhardianto, 2015). Menurut (Fitryantica, 2019), harmonisasi merupakan proses untuk melepaskan peraturan yang mengganjal atau tumpang tindih, harmonisasi dapat dilakukan dengan menerapkan konsep *omnibus law* yang kemunculannya berasal dari kebiasaan hukum *common law*. Omnibus Law adalah aturan undang-undang baru yang memuat beragam subtansi aturan yang keberadaannya mengamademenkan beberapa undang-undang (Busroh, 2017).

Penelitian ini dilatar belakangi oleh penelitian-penelitian terdahulu. Berdasarkan penelitian yang dilakukan oleh Putra Saut Martua Sinaga (2021), yang berjudul "Pengembangan Sistem Informasi Omnibus Law Menggunakan Codeigniter", menghasilkan sebuah sistem informasi berbasis web yang bertujuan untuk membantu dalam pembuatan Omnibus Law, dapat juga digunakan untuk pencarian undang-undang sehingga memudahkan masyarakat dalam mencari undang-undang.

Kemudian penelitian yang dilakukan oleh Adinda Rizky Febiyanto (2021) yang berjudul "Implementasi Text Mining Terhadap Undang-Undang Berbasis Omnibus Law Menggunakan Word2Vec dan Soft Cosine Similarity", pada penelitian yang dilakukan Adinda Rizky Febiyanto menghasilkan algoritma *text mining* yang dapat menghitung tingkat similaritas antara RUU (Rancangan Undang-Undang) dengan undang-undang yang telah ada. Penelitian berhasil menghasilkan persentase dari similaritas dokumennya tetapi belum dapat menyajikan ataupun menampilkan dalam sebuah *interface*. Sehingga perlu dilakukan penelitian lebih lanjut yang dapat menampilkan hasil perhitungan similaritas RUU dan UU yang telah ada ke dalam sistem informasi Omnibus Law yang telah dibuat oleh peneliti terdahulu Putra Saut Martua Sinaga.

1.2. Rumusan Masalah

Berdasarkan latar belakang yang telah dikemukakan, maka permasalahan yang dapat dirumuskan pada penelitian ini yaitu bagaimana mengembangkan sebuah *service* yang dapat membantu aplikasi Omnibus Law menampilkan sebuah *interface* menggunakan *framework* FastApi.

1.3. Batasan Masalah

Agar penelitian lebih terfokus, berikut ini beberapa batasan masalah dalam penelitian ini:

1. *Service* yang dikembangkan menggunakan *framework* FastApi dan menggunakan basis data MariaDB.
2. Menggunakan Swagger untuk membuat dokumentasi API.
3. Menggunakan JSON sebagai standar format dalam komunikasi data.
4. Menggunakan metode *black-box testing* dengan *tools* postman untuk memastikan kesesuaian antara *request* dan *response*.

1.4. Tujuan Penelitian

Tujuan dilakukannya penelitian ini adalah untuk menghasilkan sebuah service yang dapat digunakan oleh aplikasi Omnibus Law sehingga aplikasi dapat menampilkan *interface* yang dibutuhkan.

1.5. Manfaat Penelitian

Manfaat dari penelitian ini adalah:

1. Mempermudah Legal Drafter dalam pembuatan rancangan undang-undang.
2. Memudahkan masyarakat dalam pencarian undang-undang.
3. Dapat menjadi referensi bagi penelitian-penelitian berikutnya yang berhubungan dengan *service*.
4. Memudahkan pencarian undang-undang yang relevan sehingga memudahkan dalam mengambil keputusan.

II. TINJAUAN PUSTAKA

2.1. Penelitian Terdahulu

Beberapa penelitian terdahulu yang digunakan sebagai referensi dalam penelitian ini adalah sebagai berikut.

2.1.1. Harmonisasi Peraturan Perundang-Undangan Indonesia melalui Konsep Omnibus Law.

Penelitian ini dilakukan oleh (Fitryantica, 2019), penelitian ini diharapkan sebagai jalan analisa terhadap jalan keluar dari dampak yang terjadi dalam peraturan perundang-undangan saat ini, yang akan menerapkan konsep omnibus law yang dipraktikkan oleh negara-negara *common law* dan Amerika Serikat.

Penelitian ini berkesimpulan, bahwa implementasi Konsep omnibus law di Indonesia dan hierarki tata urutan peraturan perundang-undangan di Indonesia sebagaimana diatur di dalam Undang-Undang Nomor 12 Tahun 2011 tentang Pembentukan Peraturan Perundang-Undangan, belum memasukkan konsep omnibus law sebagai salah satu asas dalam sumber hukum maupun sebagai kerangka metodologis untuk melakukan revisi peraturan perundang-undangan. Secara teori perundang-undangan di Indonesia, kedudukan UU dari konsep omnibus law belum diatur. Jika melihat sistem perundang-undangan di Indonesia, UU hasil konsep omnibus law bisa mengarah sebagai UU Payung karena mengatur secara menyeluruh dan kemudian mempunyai kekuatan terhadap aturan yang lain. Harmonisasi Omnibus law menjadi pelindung bagi pejabat daerah yang ingin melakukan inovasi dan kreasi untuk kemajuan ekonomi dan investasi.

2.1.2. Implementasi Text Mining Terhadap Undang-Undang Berbasis Omnibus Law Menggunakan Word2Vec dan Soft Cosine Similarity.

Penelitian ini dilakukan oleh Adinda Rizky Febianto (2021), penelitian ini mengimplementasikan algoritma text mining yang digunakan untuk membandingkan RUU dan UU yang telah ada. Perbandingan tersebut menghasilkan tingkat similaritas, untuk mengetahui tingkat similaritas antara dokumen, setiap kata pada dokumen diubah ke dalam vektor dengan menggunakan metode Word2Vec, kemudian kata yang telah diubah ke vektor tersebut diukur tingkat similaritasnya menggunakan metode Soft Cosine Similarity Measure yang dapat melakukan pemodelan sinonim kata.

Hasil penelitian ini, yaitu engine atau algortima text mining untuk membandingkan antar RUU (Rancangan Undang-Undang) dengan UU (Undang-Undang) yang telah ada, pada penelitian ini juga sudah diuji tingkat keakuratan dan kebenaran dari hasil perbandingan tersebut.

2.1.3. Pengembangan Sistem Informasi Omnibus Law Menggunakan Codeigniter.

Penelitian yang dilakukan oleh Putra Saut Martua Sinaga (2021), pada penelitian ini dikembangkan sebuah Sistem Informasi Omnibus Law yang bertujuan untuk membantu Legal Drafter dalam pencarian undang-undang terkait, dengan menggunakan *filter* dan pencarian menggunakan kata kunci. Penelitian ini dikembangkan menggunakan *framework* CodeIgniter 3, dan menggunakan MariaDB sebagai databasenya.

Penelitian ini berhasil mengembangkan Sistem Informasi Omnibus Law, sistem informasi yang dihasilkan mengimplementasikan algoritma *cosine similarity* yang digunakan dalam fitur pencarian undang-undang terkait menggunakan kata kunci.

2.2. Uraian Landasan Teori

2.2.1. API

Application programming interface (API) adalah suatu dokumentasi yang terdiri dari antarmuka, fungsi, kelas, struktur dan sebagainya untuk membuat sebuah perangkat lunak (Ramadhani, 2015). Dengan adanya API ini, maka memudahkan *programmer* untuk "membongkar" suatu *software*, kemudian dapat dikembangkan atau diintegrasikan dengan perangkat lunak yang lain. API dapat dikatakan sebagai koneksi antar aplikasi dengan aplikasi lain yang memungkinkan seorang *programmer* menggunakan sistem *function* (Pranata, Hijriani, & Junaidi, 2018). Proses ini dikelola melalui sistem operasi. Keunggulan dari menggunakan API adalah memungkinkan suatu aplikasi dengan aplikasi lainnya dapat saling berkomunikasi dan berinteraksi.

2.2.2. Aplikasi

Menurut (Abdurahman & Riswaya, 2014), aplikasi adalah sekumpulan fungsi siap pakai yang dapat digunakan untuk menjalankan perintah dari pengguna aplikasi tersebut. Tujuannya mendapatkan hasil yang lebih akurat sesuai dengan tujuan pembuatan aplikasi tersebut. Aplikasi mempunyai arti yaitu pemecahan masalah yang menggunakan salah satu teknik pemrosesan data aplikasi yang biasanya berpacu pada sebuah komputasi yang diinginkan atau diharapkan maupun pemrosesan data yang diharapkan.

2.2.3. Service

Service atau web service adalah aplikasi atau layanan berbasis web yang dibuat agar dapat dipanggil, diakses, serta digunakan oleh aplikasi lain melalui internet dengan menggunakan format pertukaran data sebagai media komunikasi (Kasman, 2016). Service menggunakan jaringan protokol HTTP, layanan tersebut dapat diakses dan dimanfaatkan oleh aplikasi dengan bahasa pemrograman, arsitektur, dan sistem

operasi yang berbeda. Service dibangun menggunakan arsitektur REST (Representational State Transfer), dan menggunakan JSON atau XML sebagai format pertukaran datanya.

2.2.4. Harmonisasi

Menurut (Fitryantica, 2019), harmonisasi merupakan proses untuk melepaskan peraturan yang mengganjal atau tumpang tindih, harmonisasi dapat dilakukan dengan menerapkan konsep *omnibus law* yang kemunculannya berasal dari kebiasaan hukum *common law*.

2.2.5. Basis Data

Menurut (Swara & Pebriadi, 2016), basis data atau *database* merupakan kumpulan data yang disusun secara sistematis di dalam sebuah perangkat keras (komputer) sehingga dapat diolah menggunakan perangkat lunak untuk menjadi sebuah informasi yang berguna.

MariaDB merupakan *Relational Database Management System* (RDBMS) yang dibangun oleh orang yang sama dengan yang membuat MySQL, sehingga basis data MariaDB memiliki kemiripan pada MySQL (Hendra & Andriyani, 2020). Adapun PHPMyadmin pendukung sebagai salah satu alat bantu yang berguna dalam proses administrasi kepada basis data dan sebagai pengguna melalui visual paltform yang berjalan berupa browser.

2.2.6. REST API

REST API adalah sebuah implementasi dari API (*Application Programming Interface*). REST (*Representational State Transfer*) adalah suatu arsitektur metode komunikasi yang menggunakan protokol HTTP untuk pertukaran data (Perdana, 2018).

a. Kelebihan

Menurut (Halili & Ramadani, 2018), berikut ini kelebihan dari REST API :

1. Mendukung komunikasi *stateless*.
2. *Bandwidth* yang ringan, dikarenakan menggunakan format data JSON.
3. Dapat digunakan oleh berbagai *client*.

b. Kekurangan

Menurut (Halili & Ramadani, 2018), berikut ini kekurangan dari REST API :

1. Tidak cocok untuk data dalam jumlah besar.
2. *Request* REST (terutama GET) tidak cocok untuk data dalam jumlah besar.
3. Latensi dalam waktu pemrosesan permintaan dan penggunaan *bandwidth*.

2.2.7. JSON

JSON (Javascript *Object Notation*) merupakan format teks yang tidak bergantung pada bahasa pemrograman apapun karena menggunakan gaya bahasa yang umum digunakan. Selain JSON, terdapat XML yang memiliki fungsi sama yaitu alat komunikasi antar aplikasi, integrasi data, dan komunikasi aplikasi eksternal dengan luaran. JSON lebih unggul dari XML, mulai dari kecepatan, penulisan yang lebih mudah dan *coding* untuk *parsing* yang lebih ringkas dan sederhana (Qibtiyah & Rahayu, 2017).

2.2.8. Python

Python adalah bahasa pemrograman tingkat tinggi yang dinamis, Python dapat melakukan eksekusi sejumlah instruksi multi guna secara langsung (interpretatif) dengan metode orientasi objek. Menurut (Nosrati, 2011) beberapa karakteristik yang menjadi ciri khas atau pembeda python adalah:

1. Python cepat dan kuat, Python menyediakan semua fasilitas yang dibutuhkan untuk pemrograman dari operasi dasar hingga fungsi lanjutan. Dengan bantuan

third-party tools memungkinkan Python melakukan segala hal. Misalnya, dengan menulis hanya 3 baris kode sudah bisa membuat web server.

2. Python bersifat portabel, Python dapat digunakan pada sistem operasi yang berbeda-beda seperti: Windows, Linux, UNIX, Amigo, Mac OS, dan OS lainnya.
3. Python sederhana dan indah, Python adalah bahasa tingkat tinggi yang memiliki banyak sumber untuk belajar. Selain itu, beragam *third-party tools* membuat bahasa ini menjadi mudah digunakan.
4. Python bersifat open source, dimana tidak ada batasan untuk digunakan, diubah, dan didistribusikan.

2.2.9. *Framework* FastApi

Pada tahun 2018, Sebastian Ramirez mengembangkan kerangka kerja web Python yang disebut FastAPI. Menurut pendapatnya, FastAPI seharusnya menjadi kerangka kerja terbaik untuk mengembangkan layanan REST API. Framework FastApi dijalankan secara asynchronous dan mudah diintegrasikan dengan OpenAPI-schema, yang membuatnya lebih mudah untuk bekerja dengan Swagger dan ReDoc (Kornienko, Mishina, & Melnikov, 2021). Meskipun FastApi belum lama dikembangkan, FastAPI memantapkan dirinya sebagai tools yang berkualitas. Banyak perusahaan besar mulai menggunakan untuk mengembangkan API mereka.

2.2.10. Unit Testting

Menurut (Rizkyana, Yunanto, Yoga, Herdian, & R., 2021), Unit testing merupakan salah satu metode pengujian perangkat lunak yang digunakan dengan cara menguji unit terkecil dari sebuah kode. Kegunaan unit testing bagi pengembang perangkat lunak adalah dapat memperkecil kesalahan atau bug yang ada pada sistem. Pengujian unit testing dilakukan hingga sistem sudah memenuhi syarat sesuai rancangan. Untuk membuktikan bahwa pengujian unit testing dapat efektif digunakan dalam pengembangan service, maka dalam penelitian ini dilakukan pengujian service dengan unit testing yaitu menggunakan Postman.

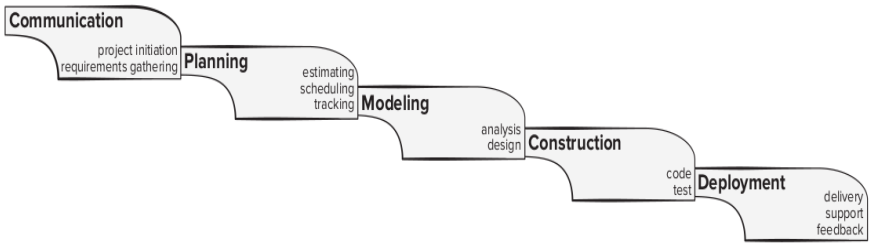
2.2.11. Swagger

Menurut (Dermawansyah & Syaryadhi, 2018), swagger adalah suatu *tools framework* untuk merancang, membuat dokumentasi dan mengakses Restful API berdasarkan spesifikasi dari Open API. Bahasa dan format *file* yang digunakan untuk membangun rancangan Restful API menggunakan swagger adalah YAML (*Yet Another Markup Language*). Swagger berperan dalam memberikan spesifikasinya kepada Open API *Initiative* (OAI) yang menjadi standar dalam pengembangan Restful API.

2.2.12. UML (Unified Modeling Language)

Unified Modeling Language (UML) adalah standar bahasa yang digunakan untuk menjelaskan dengan jelas *requirement*, pembuatan desain serta analisis dan arsitektur dari pemrograman yang berorientasi objek tersebut digambarkan dengan jelas dikarenakan dalam dunia industri standar bahasa ini banyak digunakan. UML ada dikarenakan dalam pemodelan visual sangat dibutuhkan untuk menspesifikasikan, menggambarkan, membangun, dan mendokumentasikan dari sisi sistem perangkat lunak. UML adalah salah satu standar bahasa yang banyak digunakan didunia industri untuk mendefinisikan requirement, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek (Sukamto & Shalahudin, 2014).

2.2.13. Metode Waterfall



Gambar 2.1 Metodologi *Waterfall* (Pressman & Maxim, 2020).

Metode *waterfall* adalah metode kerja yang menekankan fase-fase yang berurutan dan sistematis. Disebut *waterfall* karena proses mengalir satu arah "ke bawah" seperti air terjun. Metode *waterfall* ini harus dilakukan secara berurutan sesuai dengan tahap yang ada. Menurut (Pressman & Maxim, 2020), metode *waterfall* memiliki 5 fase pengembangan yaitu *Communication*, *Planning*, *Modeling*, *Construction*, dan *Deployment*. Secara *flow* dapat dilihat pada Gambar 2.1.

Communication adalah tahap penginisiasian proyek atau bagaimana proyek yang akan dibuat. *Planning* sendiri adalah membuat rencana kerja dan proyek yang telah diinisiasikan. Kemudian masuk ke dalam tahap *Modeling* yang dilakukan untuk membuat desain analisis dari proyek agar mendapatkan gambaran kasar dari proyek. Setelah selesai melakukan *Modeling*, maka dilakukan *Construction* untuk mengimplementasikan desain yang dibuat ke dalam kode program dan melakukan pengujian. Tahap akhir dari metode *waterfall* adalah *deployment* yang bertujuan untuk menyebarkan sistem yang sudah dibuat, sering juga disebut melakukan *hosting*. Saat proses *deployment* berjalan, pengembang sistem tetap harus memantau sistem untuk mengetahui apakah ada *error* pada sistem dikemudian hari, *error* sendiri bisa ditemukan oleh pengembang maupun para pengguna yang memberikan *feedback* kepada pengembang.

2.2.14. Black-Box Testing

Black-box testing merupakan pengujian perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program untuk mengetahui apakah fungsi, masukan dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan (Cholifah, Yulianingsih, & Sagita, 2018). Metode *Black-box testing* merupakan salah satu metode yang mudah digunakan karena hanya memerlukan batas bawah dan batas atas dari data yang diharapkan, Estimasi banyaknya data uji dapat dihitung melalui banyaknya *field* data *entry* yang akan diuji, aturan entri yang harus dipenuhi serta kasus batas atas dan batas bawah yang memenuhi. Metode ini dapat diketahui jika fungsionalitas masih dapat menerima masukan data yang tidak diharapkan maka menyebabkan data yang disimpan kurang valid.

III. METODE PENELITIAN

3.1. Waktu dan Tempat Penelitian

Penelitian dilakukan di jurusan ilmu komputer, fakultas matematika dan ilmu pengetahuan alam, Universitas Lampung. Penelitian ini dilakukan pada semester genap Tahun Ajaran 2021/2022.

3.2. Perangkat Penelitian

Spesifikasi perangkat yang digunakan pada penelitian ini adalah sebagai berikut.

3.2.1. Perangkat Keras

Perangkat keras yang digunakan dalam penelitian ini adalah sebuah laptop dengan spesifikasi sebagai berikut.

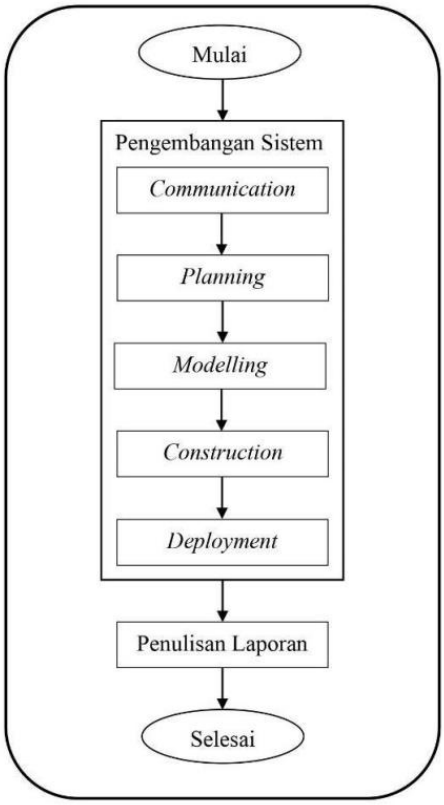
a. *System Manufacturer* : ASUS
b. *System* Model : ASUS VivoBook
c. *Processor* : Intel® Core™ i7-10510U CPU @ 1.80GHz (8 CPUs)
d. *Storage* : 500 GB SSD
e. *Installed* RAM : 8 GB
f. *System Type* : 64 bit

3.2.2. Perangkat Lunak

Perangkat lunak yang digunakan dalam penelitian ini adalah sebagai berikut.

a. Sistem Operasi Windows 10
b. Visual Studio Code *version* 1.63.2
c. Apache *version* 2.4.41
d. PHP *version* 7.3.28
e. MariaDB *version* 10.4.18
f. Chrome *Browser version* 101.0.4951.54
g. *Framework* FastApi *version* 0.46.0
h. Git *version control system version* 2.2.5.1

3.3. Tahapan Penelitian



Gambar 3.1 Diagram Alir Penelitian.

Gambar 3.1 merupakan diagram alir penelitian. Tahapan penelitian dibagi menjadi 3 bagian yaitu pengumpulan data, pengembangan sistem, dan penulisan laporan.

3.3.1. Studi Literatur

Tahapan yang pertama adalah studi literatur. Studi literatur adalah tahapan yang dilakukan untuk memahami bagaimana struktur dari aplikasi Omnibus Law dan juga mengidentifikasi permasalahan yang sering muncul.

3.3.2. Pengumpulan Data

Pengumpulan data pada penelitian ini didapatkan dari 2 sumber yaitu observasi dan studi pustaka.

a. Studi Pustaka

Studi pustaka merupakan langkah awal dalam metode pengumpulan data. Studi pustaka merupakan metode pengumpulan data yang diarahkan kepada pencarian data dan informasi melalui dokumen-dokumen, baik dokumen tertulis, foto-foto, gambar, maupun dokumen elektronik yang berkaitan dengan REST API maupun *service* yang dapat mendukung dalam proses pengembangan sistem.

b. Observasi

Observasi merupakan langkah kedua dalam pengumpulan data setelah penulis melakukan studi pustaka. Observasi dilakukan agar penulis dapat mengetahui dengan lebih pasti kondisi permasalahan yang sebenarnya pada aplikasi Omnibus Law. Observasi dilakukan dengan cara mempelajari aplikasi yang telah dibuat sebelumnya seperti arsitektur, *database*, struktur, fitur-fitur yang ada dan juga *gate* yang sering diakses oleh pengguna.

3.3.3. Pengembangan Sistem

Tahap selanjutnya yaitu melakukan pengembangan sistem, pada tahap ini menggunakan metode *waterfall*. Mulai dari *Communication, Planning, Modeling, Construction*, dan *Deployment*.

1. Communication

Communication adalah tahap inisialisasi proyek seperti menganalisis permasalahan yang dihadapi dan mengumpulkan data-data yang diperlukan, serta membantu mendefinisikan fitur dan fungsi dari aplikasi. Tahap ini dilakukan dengan cara berdiskusi dengan peneliti sebelumnya dan juga mengumpulkan data-data tambahan baik yang ada di jurnal, artikel, maupun dari internet. Dari tahapan ini, didapatkan hasil berupa analisis sistem yaitu:

a. Analisis Masalah

Berdasarkan penjelasan yang diberikan oleh peneliti sebelumnya, permasalahan yang ada pada aplikasi Omnibus Law adalah dibutuhkannya fitur untuk menghitung tingkat similaritas antara rancangan undang-undang dan undang-undang yang telah ada, dan dibutuhkannya fitur untuk mencari pasal yang berkaitan pada undang-undang yang telah ada, sehingga dapat membantu Legal Drafter dalam mencabut atau merubah kebijakan yang sedang berlaku.

b. Analisis Kebutuhan Sistem

Pada penelitian ini akan dikembangkan *service* harmonisasi yang digunakan untuk mengelola dan memanajemen data undang-undang yang akan digunakan pada aplikasi Omnibus Law. Kebutuhan fungsional merupakan kebutuhan aplikasi yang berhubungan dengan proses *input* dan *output* pada aplikasi. Kebutuhan fungsional dari pengembangan *service* harmonisasi ini adalah:

1) *Service* dapat menghitung tingkat similaritas dokumen.
2) *Service* dapat mengirimkan data ke aplikasi Omnibus Law.
3) Service dapat mencari pasal UU yang berkaitan dengan bunyi pasal UU yang dicari.

Kebutuhan fungsional *service* akan dijelaskan lebih jelas pada *use case* diagram yang akan dibahas pada tahap *modeling*.

Adapun kebutuhan secara non-fungsional yaitu:

1) Terdapat dokumentasi api yang dapat memudahkan *frontend engineer*.
2) Memiliki tingkat keamanan yang baik.

2. *Planning*

Tahap pengembangan sistem berikutnya adalah *planning*. *Planning* menjelaskan tentang estimasi tugas-tugas teknis yang akan dilakukan, produk kerja yang ingin dihasilkan, penjadwalan kerja yang akan dilaksanakan, dan *tracking* proses pengerjaan sistem. Tahapan ini akan menghasilkan *Gantt Chart* dari pengembangan sistem yang dapat dilihat pada Tabel 3.1.

Tabel 3.1 Rencana Pengembangan Sistem.

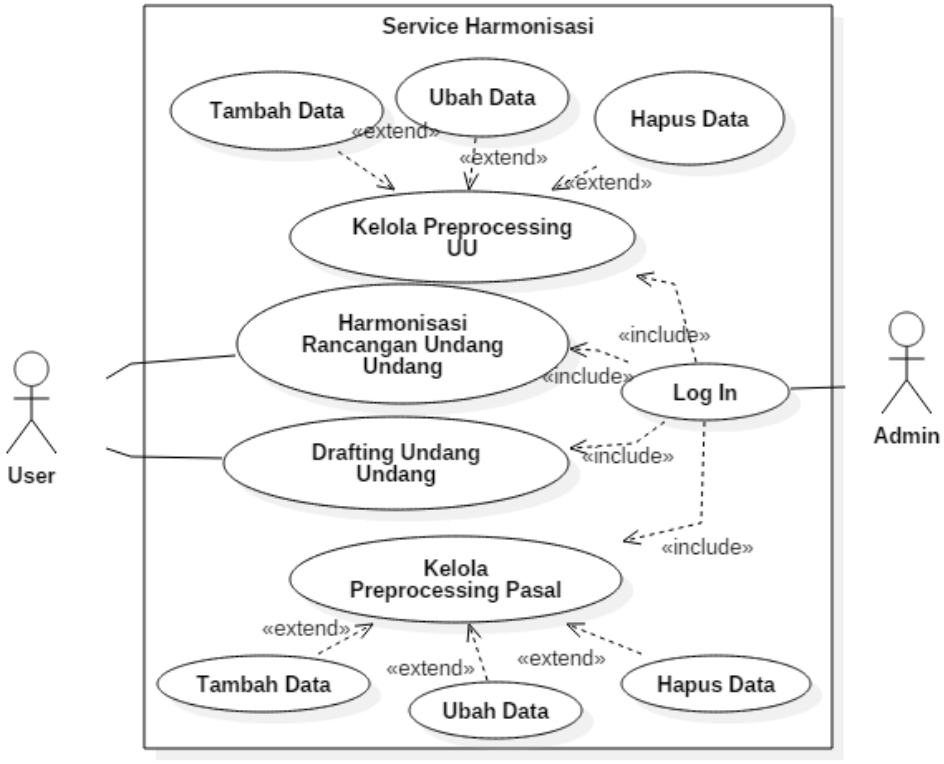
Kegiatan	Detail Kegiatan	Januari				Februari				Maret				April			
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Modeling	Desain Arsitektur																
	Use Case Diagram																
	Activity Diagram																
	Pengecekan																
	Pembuatan *Database*																
Construction	Pembuatan *endpoint preprocessing*																
	Pembuatan *endpoint* harmonisasi dokumen																
	Pembuatan *endpoint* pencarian pasal																
	Pembuatan dokumentasi API																
	Pengecekan																
Testing	Pembuatan skenario																
	Pengujian																
Deployment	*Hosting*																
	CI/CD																

3. *Modeling*

Modeling sendiri menjadi gambaran kasar sistem sebelum diimplementasi pada sebuah bahasa pemrograman. *Modeling* merupakan tahap perancangan dan pemodelan arsitektur sistem yang berfokus pada perancangan *Use Case* Diagram,

Activity Diagram, ER Diagram. Tujuannya untuk lebih memahami gambaran besar dari apa yang akan dikerjakan.

a. *Use Case* Diagram



Gambar 3.2 *Use Case* Diagram *Service* Harmonisasi.

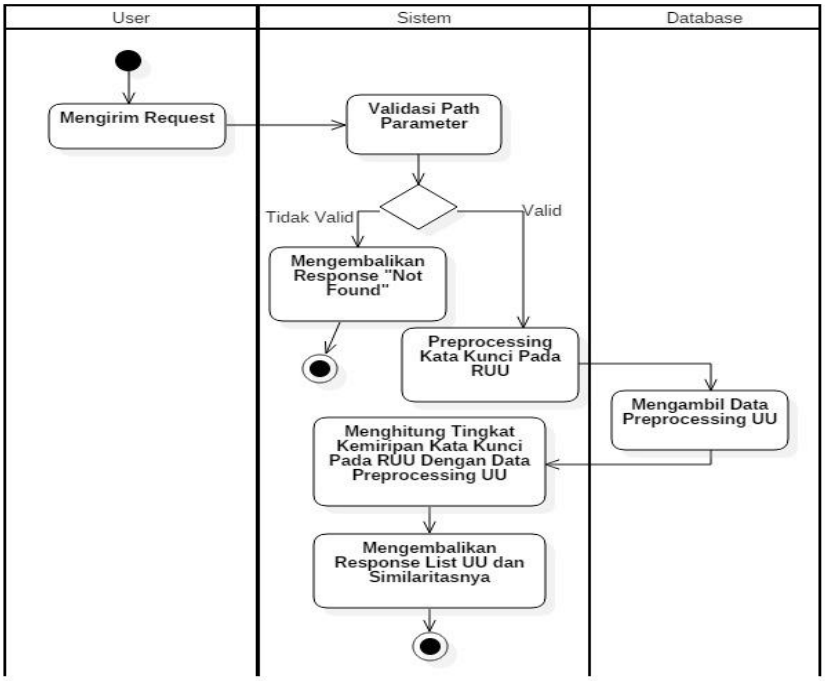
Gambar 3.2 merupakan *Use Case* Diagram dari *Service* Harmonisasi, Terdapat 2 aktor yaitu *user* dan *admin*. *Admin* dapat mengelola data preprocessing UU, dan mengelola data preprocessing pasal, termasuk menambah, mengubah, dan menghapus data. *Admin* juga dapat melakukan harmonisasi rancangan undang-undang, dan drafting undang-undang. Sedangkan *user* dapat melakukan harmonisasi rancangan undang-undang, dan drafting undang-undang.

***b.* *Activity* Diagram**

Activity diagram merupakan gambaran dari proses-proses yang terjadi pada sebuah sistem. *Activity* diagram adalah pengembangan dari *Use Case* diagram yang

memiliki alur aktivitas. *Activity* diagram pada *service* harmonisasi memiliki 3 fitur utama.

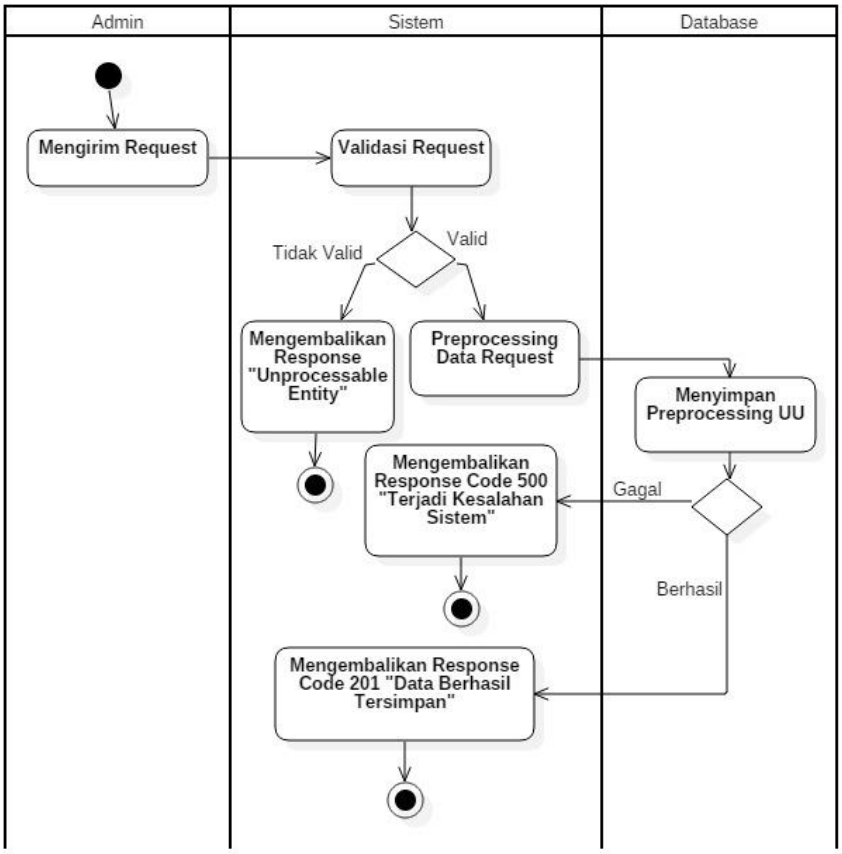
1) *Activity* Diagram Harmonisasi Rancangan Undang-Undang



Gambar 3.3 *Activity* Diagram Harmonisasi Rancangan Undang-Undang.

Gambar 3.3 merupakan proses untuk harmonisasi rancangan undang-undang. *User* mengirimkan *request* ke *service* harmonisasi kemudian pada *service* harmonisasi akan divalidasi parameter pada *path endpoint* berupa file rancangan undang-undang. Jika valid maka *service* akan melakukan preprocessing kata kunci yang ada pada rancangan undang-undang, kemudian mengambil data preprocessing undang-undang dari *database* kemudian dari data yang dibawa akan dibandingkan dengan kata kunci pada rancangan undang-undang dan dihitung tingkat kemiripannya, kemudian akan dikembalikan sebagai *response* dalam bentuk JSON. Namun jika tidak valid, maka *service* akan mengembalikan *response not found*.

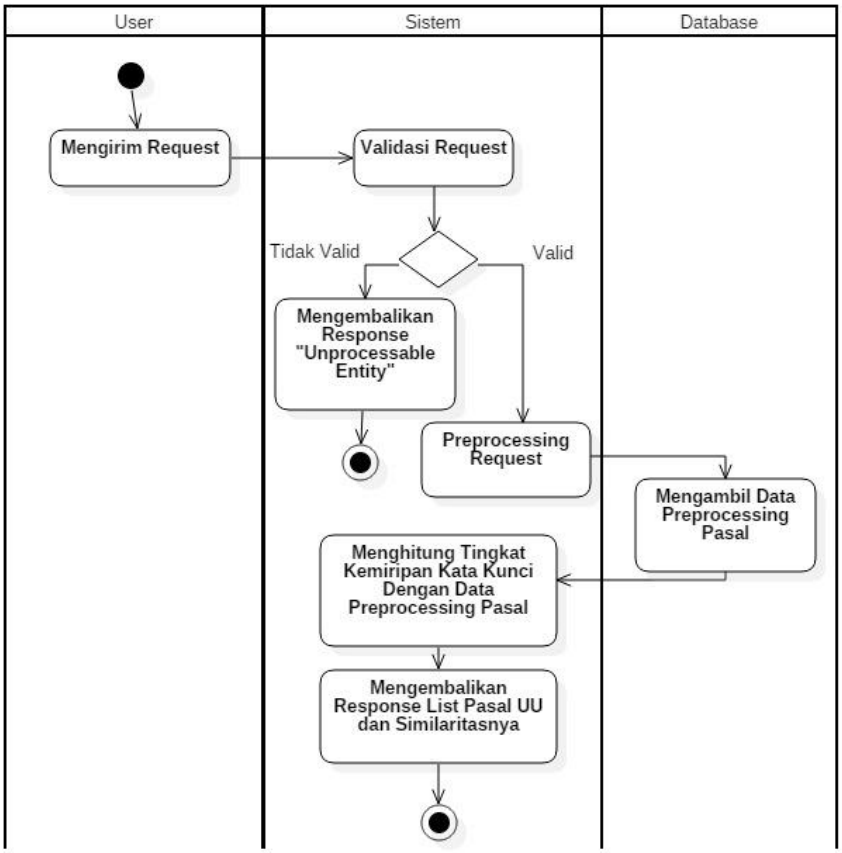
2) *Activity* Diagram Menambah Data Preprocessing Undang-Undang



Gambar 3.4 *Activity* Diagram Menambahkan Preprocessing UU.

Gambar 3.4 merupakan *Activity* diagram untuk menambahkan data preprocessing undang-undang. Proses menambahkan data undang-undang terdapat 2 pengecekan, yang pertama yaitu validasi *request*, jika tidak valid maka sistem akan mengembalikan *response unprocessable entity*, jika valid data *request* akan di preprocessing oleh *service*. Kemudian masuk kedalam pengecekan yang ke-2, jika data gagal tersimpan makan sistem akan mengembalikan *response code* 500 dengan pesan terjadi kesalahan, jika data berhasil tersimpan maka sistem akan mengembalikan *response code* 201 dengan pesan data berhasil tersimpan.

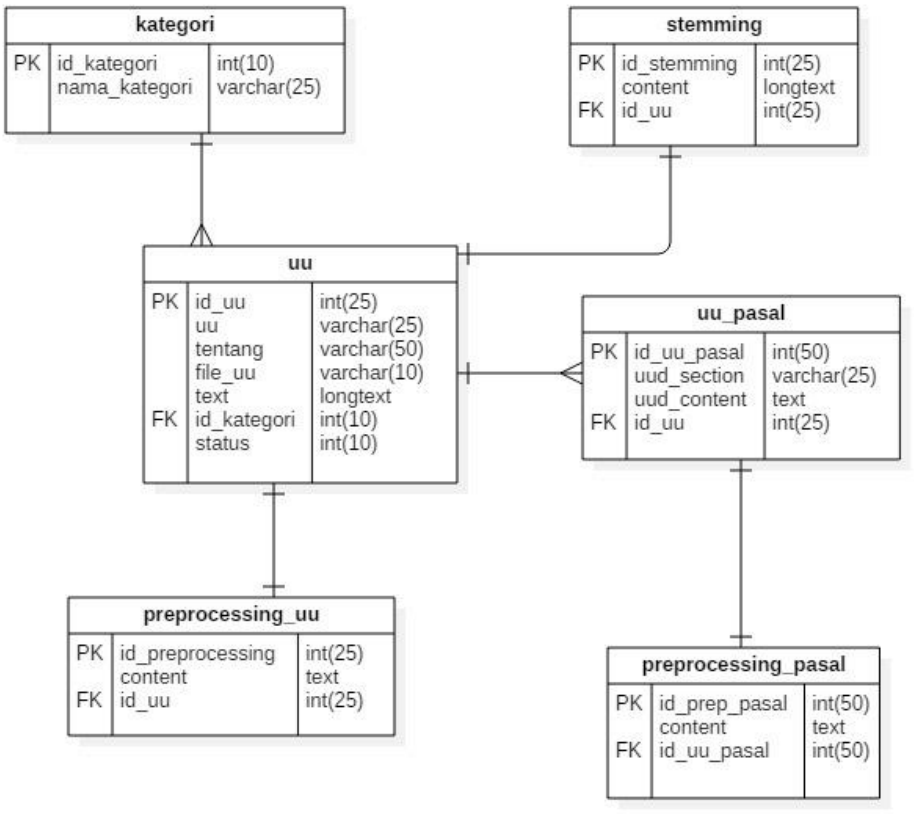
3) *Activity* Diagram Drafting Undang-Undang



Gambar 3.5 *Activity* Diagram Drafting Undang-Undang.

Gambar 3.5 merupakan proses untuk drafting undang-undang. *User* mengirimkan *request* ke *service* harmonisasi, kemudian pada *service* harmonisasi akan divalidasi *request* nya. Jika valid maka *service* akan melakukan preprocessing *request* tersebut, kemudian mengambil data preprocessing pasal dari *database* kemudian dari data yang dibawa akan dibandingkan dengan data pada request, dan dihitung tingkat kemiripannya, kemudian akan dikembalikan sebagai *response* dalam bentuk JSON. Namun jika tidak valid, maka *service* akan mengembalikan *response unprocessable entity.*

c. Entity Relationship Diagram



Gambar 3.6 *Entity Relationship* Diagram *Service* Harmonisasi.

Gambar 3.6 merupakan *Entity relationship* pada *service, Entity relationship* diagram merupakan suatu model atau rancangan untuk membuat *database* yang menggambarkan data yang memiliki hubungan atau relasi dalam bentuk sebuah desain. Terdapat 6 entitas atau tabel, yaitu kategori, stemming, uu, uu pasal, preprocessing uu, dan preprocessing pasal, untuk nama atribut dan *type* datanya dapat dilihat pada Gambar. Atribut pada setiap entitas akan dijelaskan pada kamus data.

d. Kamus Data

Kamus data digunakan untuk menjelaskan lebih detail mengenai atribut dari setiap entitas pada ERD, sehingga masing-masing atribut dapat terjelaskan maksudnya.

Berikut merupakan kamus data yang merupakan penjelasan lebih lanjut dari ERD yang sudah dibuat.

1) Kamus Data Entitas kategori

Kamus data entitas kategori menjelaskan masing-masing atribut yang terdapat pada entitas kategori. Kamus data kategori dapat dilihat pada Tabel 3.2.

Tabel 3.2 Kamus data entitas kategori

Nama Field	Tipe Data	Ukuran	Batasan
id_kategori	integer	10	primary key
nama_kategori	integer	25	not null

2) Kamus Data Entitas uu

Kamus data entitas uu menjelaskan masing-masing atribut yang terdapat pada entitas uu. Kamus data uu dapat dilihat pada Tabel 3.3.

Tabel 3.3 Kamus data entitas uu

Nama Field	Tipe Data	Ukuran	Batasan
id_uu	integer	25	primary key
uu	varchar	25	not null
tentang	varchar	50	not null
file_uu	varchar	10	not null
text	longtext		not null
id_kategori	integer	10	foreign key dari kategori, unique, not null
status	integer	10	not null

3) Kamus Data Entitas preprocessing_uu

Kamus data entitas preprocessing_uu menjelaskan masing-masing atribut yang terdapat pada entitas preprocessing_uu. Kamus data preprocessing_uu dapat dilihat pada Tabel 3.4.

Tabel 3.4 Kamus data entitas preprocessing_uu

Nama Field	Tipe Data	Ukuran	Batasan
id_preprocessing	integer	25	primary key
content	text		not null
id_uu	integer	25	foreign key dari uu, unique, not null

4) Kamus Data Entitas stemming

Kamus data entitas stemming menjelaskan masing-masing atribut yang terdapat pada entitas stemming. Kamus data stemming dapat dilihat pada Tabel 3.5.

Tabel 3.5 Kamus data entitas stemming

Nama Field	Tipe Data	Ukuran	Batasan
id_stemming	integer	25	primary key
content	longtext		not null
id_uu	integer	25	foreign key dari uu, unique, not null

5) Kamus Data Entitas uu_pasal

Kamus data entitas uu_pasal menjelaskan masing-masing atribut yang terdapat pada entitas uu_pasal. Kamus data uu_pasal dapat dilihat pada Tabel 3.6.

Tabel 3.6 Kamus data entitas uu_pasal

Nama Field	Tipe Data	Ukuran	Batasan
id_uu_pasal	integer	50	primary key
uud_section	varchar	25	not null
uud_content	text		not null
id_uu	integer	25	foreign key dari uu, unique, not null

6) Kamus Data Entitas preprocessing_pasal

Kamus data entitas preprocessing_pasal menjelaskan masing-masing atribut yang terdapat pada entitas preprocessing_pasal. Kamus data preprocessing_pasal dapat dilihat pada Tabel 3.7.

Tabel 3.7 Kamus data entitas uu_pasal

Nama Field	Tipe Data	Ukuran	Batasan
id_prep_pasal	integer	50	primary key
content	text		not null
id_uu_pasal	integer	50	foreign key dari uu_pasal, unique, not null

4. *Construction*

Construction merupakan proses penerjemahan bentuk desain atau model yang sudah dibuat pada tahap sebelumnya menjadi bentuk kode atau bahasa yang yang dapat dimengerti oleh mesin. Hasil dari *Construction* dijelaskan pada bab 4 hasil dan pembahasan. Setelah melakukan koding, maka dilakukan pengujian dengan metode *Black-Box Testing*. Skenario pengujian dapat dilihat pada poin 3.4 tentang Skenario Pengujian Sistem. Tujuannya untuk menemukan kesalahan yang mungkin terjadi untuk nantinya diperbaiki. Setelah dilakukan testing maka tahap selanjutnya yaitu membuat API *documentation* yang akan dibuat dengan Swagger.

5. *Deployment*

Tahapan terakhir dari metode *waterfall* adalah *deployment* yang merupakan tahapan membuat *service* yang sudah dikembangkan menjadi *live* atau biasa disebut dengan *hosting*, sehingga memudahkan *programmer service* lain dalam penggunaanya jika dibutuhkan dalam proses pengembanganya. Setelah melakukan tahap *deployment*, pengembang mengimplementasikan CI (*Continuous Integration*)/CD (*Continuous Delivery*) agar setiap terjadi perubahan kode pada *service* tidak langsung di-*push* ke dalam *production*, namun masuk ke dalam fase *review* dan *staging* terlebih dahulu. Dimana perubahan atau penambahan kode tersebut dapat dilakukan *test* sebelum kode diimplementasikan ke dalam fase *production*. Sehingga melalui tes otomatis ini akan lebih mudah mendeteksi *bug* pada tahap pengembangan awal, jadi tidak perlu khawatir dengan munculnya *error* pada detik-detik terakhir.

3.3.4. Skenario Pengujian Sistem

Pengujian sistem pada penelitian ini yaitu menggunakan metode *black-box testing*. Jenis *black-box testing* yang digunakan pada penelitian ini adalah *Equivalence Partitioning*. *Equivalence Partitioning* merupakan metode pengujian pada *black-box testing* dengan menggunakan skenario uji, hasil yang diharapkan dan hasil pengujian untuk melihat apakah sistem berjalan sesuai atau tidak (Pranata, Hijriani, & Junaidi, 2018). *Tools* yang akan digunakan dalam pengujian ini yaitu Postman. Berikut merupakan skenario dalam pengujian *service* harmonisasi.

Tabel 3.8 Skenario Pengujian *Equivalence Partitioning* pada *Service* Harmonisasi

Kode Uji	**Daftar Pengujian**	**Kasus Uji**	**Hasil yang Diharapkan**
Path	Validasi *path*	*Request method* GET pada *endpoint* /harmonisasi/file/{file}	Sukses mendapatkan list undang-undang, dan similaritas
		Request method POST pada *endpoint* /draft/pasal	Sukses mendapatkan list pasal yang dicari dan tingkat similaritas
		Request method GET pada *endpoint* /harmonisasi/file/{wrongFile}	*Error* dengan status kode 404 - *not found*
		Request method GET pada *endpoint* /salah	*Error* dengan status kode 404 - *not found*
Request method	Validasi *request method*	*Request method* POST pada *endpoint* /harmonisasi/file/{file}	*Error* dengan status kode 405 - *method not allowed*
		Request method PUT pada *endpoint* /draft/pasal	*Error* dengan status kode 405 - *method not allowed*
Request body	Validasi *request body*	*Request method* POST pada *endpoint* /draft/pasal tanpa *body*	*Error* dengan status kode 422 - *Unprocessable Entity*

V. SIMPULAN DAN SARAN

5.1. Simpulan

Adapun kesimpulan dari penelitian yang telah dilakukan adalah sebagai berikut.

1. Telah berhasil dikembangkan *service* harmonisasi pada aplikasi Omnibus Law dengan menggunakan *framework* FastApi dan basis data MariaDB.
2. *Service* harmonisasi merupakan sebuah *service* yang membantu aplikasi Omnibus Law mengolah data untuk proses draft dan harmonisasi.
3. Terdapat 10 *endpoint* yang dapat digunakan dan setiap *endpoint* tersebut sudah ada di dalam dokumentasi API yang telah dibuat menggunakan Swagger.
4. Hasil pengujian dari unit *test* menunjukkan bahwa terdapat 38 *test case* yang sudah sesuai dan telah berhasil diimplementasikan pada aplikasi Omnibus Law.

5.2. Saran

Adapun saran dari penelitian yang telah dilakukan adalah sebagai berikut.

1. Perlu ditambahkannya proses authorisasi menggunakan JWT (JSON Web Token) untuk meningkatkan keamanan pada *service* harmonisasi, sehingga *service* dapat dijadikan *public* API.
2. Diperlukan pengembangan untuk mempercepat waktu perhitungan draft dan harmonisasi.

3. Pada proses harmonisasi dapat membaca banyak jenis file, sehingga tidak terbatas pada file pdf saja.

DAFTAR PUSTAKA

Abdurahman, H., & Riswaya, A. R. (2014). Aplikasi Pinjaman Pembayaran Secara Kredit Pada Bank Yudha Bhakti STMIK Mardira Indonesia, Bandung. *Computech & Bisnis, 8(2)*, 61-69.

Busroh, F. F. (2017). Konseptualisasi Omnibus Law dalam Menyelesaikan Permasalahan Regulasi Pertahanan. *Arena Hukum*, 227-250.

Cholifah, W. N., Yulianingsih, Y., & Sagita, S. M. (2018). Pengujian Black Box Testing pada Aplikasi Action & Strategy Berbasis Android dengan Teknologi Phonegap. *STRING (Satuan Tulisan Riset Dan Inovasi Teknologi), 3(2)*, 206.

Dermawansyah, D., & Syaryadhi, M. (2018). Restful Web Service Untuk Pemantauan Dan Pengendalian Peternakan Ayam Broiler. *Jurnal Komputer, Informasi Teknologi, Dan Elektro, 3(2)*, 53-59.

Fitryantica, A. (2019). Harmonisasi Peraturan Perundang-Undangan Indonesia melalui Konsep Omnibus Law. *Jurnal Gema Keadilan, 6(III)*, 300-316.

Hendra, & Andriyani, W. (2020). Studi Komparasi Menyimpan dan Menampilkan Data Histori Antara Database Terstruktur MariaDB dan Database Tidak Terstruktur InfluxDB. *Jurnal Teknologi Technoscientia, Vol. 12*, 168-174.

Kasman, A. D. (2016). *Trik Kolaborasi ANDROID dengan PHP dan MySQL*. Yogyakarta: Lokomedia.

Kornienko, D. V., Mishina, S. V., & Melnikov, M. O. (2021). Principles of securing RESTful API web services developed with python frameworks. *Journal of Physics: Conference Series*, 1-11.

Malau, M. T. (2014). Aspek Hukum Peraturan dan Kebijakan Pemerintah Indonesia Menghadapi Liberalisasi Ekonomi Regional: Masyarakat Ekonomi ASEAN 2015. *Jurnal Rechts Vinding*, 171-172.

Nosrati, M. (2011). Python: An appropriate language for real world programming. *World Applied Programming, 1(2)*, 110-117.

Nurhardianto, F. (2015). Sistem Hukum dan Posisi Hukum Indonesia. *Jurnal Tapis*, 35-45.

Perdana, M. A. (2018). Pengembangan REST API Layanan Penyimpanan menggunakan Metode Rapid Application Development (Studi kasus PT. XYZ). *Jurnal Nasional Informatika Dan Teknologi Jaringan, 3(1)*, 100-104.

Pranata, B. A., Hijriani, A., & Junaidi, A. (2018). Perancangan Application Programming Interface (API) Berbasis WEB Menggunakan Gaya Arsitektur Representational State Transfer (REST) Untuk Pengembangan Sistem Informasi Administrasi Pasien Klinik Perawatan Kulit. *Jurnal Komputasi, 6(1)*, 33-42.

Pressman, R., & Maxim, B. (2020). *Software Engineering: A Practitioner's Approach* (9th ed.). New York: McGraw-Hill Education.

Putra, M. L., & Putera, M. A. (2019). Analisis Perbandingan metode SOAP dan REST yang digunakan pada Framework Flask untuk membangun Web Service. *Teknologi Informasi dan Komunikasi, XIV*, 1-7.

Qibtiyah, U. M., & Rahayu, S. (2017). Implementasi JSON Web Service pada Aplikasi Digital Library Politeknik Sukabumi. *Jurnal Teknologi Rekayasa, 2(1)*, 9.

Ramadhani, M. F. (2015). Pembangunan Aplikasi Informasi, Pengaduan, Kritik, Dan Saran Seputar Kota Cimahi Pada Platform Android. *Jurnal Ilmiah Komputer Dan Informatika (KOMPUTA)*, 9.

Rizkyana, M. A., Yunanto, Yoga, Herdian, C. A., & R., A. Y. (2021). Implementasi Unit Testing Menggunakan Metode Test-First Development. *JURNAL MULTINETICS, 7(1)*, 37-47.

Setiadi, W. (2020). Simplifikasi Regulasi Dengan Menggunakan Metode Pendekatan Omnibus Law. *Jurnal Rechts Vinding*, 39-52.

Standsyah, R. E., & Restu, I. S. (2017). Implementasi PHPMyAdmin Pada Rancangan Sistem Pengadministrasian. *Jurnal UJMC, 3(2)*, 38-44.

Sukamto, R., & Shalahudin, M. (2014). *Rekayasa Perangkat Lunak Terstruktur Dan Berorientasi Objek.* Bandung: Informatika Bandung.

Swara, G. Y., & Pebriadi, Y. (2016). Rekayasa Perangkat Lunak Pemesanan Tiket Bioskop Berbasis Web. *Jurnal TEKNOIF, 4(2)*, 27-39.