BAB 2

TINJAUAN PUSTAKA

2.1 Landasan Teori

Landasan teori mencakup konsep dan prinsip yang mendukung analisis serta pemecahan masalah dalam proyek akhir. Sebagai dasar ilmiah, landasan ini berfungsi sebagai kerangka acuan dalam merumuskan solusi dan menerapkan metodologi yang tepat guna mencapai tujuan secara sistematis dan terukur.

2.1.1 PTPN IV Regional III

PT. Perkebunan Nusantara IV (PTPN IV) Regional III merupakan salah satu entitas di bawah naungan Badan Usaha Milik Negara (BUMN) yang bergerak di sektor perkebunan kelapa sawit. Wilayah operasional Regional III mencakup beberapa unit kerja di Sumatera Utara dengan fokus pada pengelolaan produksi, pengolahan, dan distribusi komoditas sawit. Dalam menjalankan perannya, PTPN IV tidak hanya terlibat dalam aktivitas industri, tetapi juga mendukung program pengembangan sumber daya manusia dan pendidikan melalui kemitraan dengan institusi akademik:

Sebagai bagian dari komitmen tersebut, PTPN IV Regional III secara rutin menerima peserta magang, kunjungan edukatif, dan penelitian dari berbagai universitas dan lembaga pendidikan. Aktivitas ini memerlukan sistem pengelolaan administrasi yang terstruktur, transparan, dan terdokumentasi dengan baik untuk menjamin kelancaran proses, baik dari sisi internal perusahaan maupun pemohon. Oleh karena itu, pemahaman terhadap struktur organisasi, proses kerja, dan kebijakan internal PTPN IV menjadi landasan penting dalam merancang sistem informasi pengajuan kegiatan akademik yang tepat guna, khususnya dari sisi integrasi dan alur otorisasi yang sesuai dengan tata kelola perusahaan.

2.1.2 Sistem Informasi

Sistem informasi merupakan gabungan dari berbagai komponen teknologi dan manusia yang bekerja bersama untuk mengumpulkan, mengelola, menyimpan, dan mendistribusikan informasi yang relevan dan berguna untuk mendukung

kegiatan organisasi. Menurut Jonny Seah (2020), *sistem informasi* memungkinkan pertukaran data yang efisien antara berbagai pihak di dalam organisasi, sehingga mendukung pengambilan keputusan yang lebih cepat dan berbasis data yang akurat.

Sebagai sistem yang terintegrasi, *sistem informasi* membantu meningkatkan kinerja organisasi dengan menyediakan jalur komunikasi yang terstruktur. Dengan begitu, setiap elemen dalam organisasi bisa mengakses informasi yang dibutuhkan sesuai dengan peran dan tanggung jawab mereka. Di PTPN IV, penerapan *sistem informasi* yang efisien akan sangat membantu dalam mengelola pengajuan kegiatan akademik dengan lebih terstruktur.

2.1.3 Sistem Informasi Manajemen

Sistem Informasi Manajemen (SIM) merupakan sistem terstruktur yang mengintegrasikan komponen-komponen seperti prosedur kerja, manusia, teknologi, dan informasi untuk mendukung proses pengambilan keputusan dalam organisasi. Menurut James Alter (1992), SIM berfungsi sebagai penghubung antara aliran informasi, aktivitas operasional, teknologi informasi, dan kontribusi sumber daya manusia dalam rangka mencapai sasaran strategis institusi. Sementara itu, Bodnar dan Hopwood (1993) mendefinisikan SIM sebagai kombinasi antara perangkat keras dan lunak yang bekerja untuk mengolah data menjadi informasi yang bermanfaat bagi manajemen dalam mengambil keputusan yang tepat.

Gambar 2. 1 Hirarki Manajemen

Dalam implementasinya, SIM memiliki struktur hirarkis yang mencerminkan peran informasi pada setiap tingkat manajemen. Gambar 2.1 menggambarkan

Hirarki Manajemen SIM, di mana informasi mengalir dari data operasional di level bawah, menuju laporan-laporan terstruktur di tingkat menengah, hingga ringkasan analitis untuk manajemen tingkat atas. Pada konteks penelitian ini, SIM berperan penting dalam pengelolaan pengajuan kegiatan akademik, karena memungkinkan distribusi tugas administratif ke unit subarea berdasarkan otoritas yang ditentukan, serta menyajikan informasi yang akurat untuk keperluan monitoring dan pengambilan keputusan oleh pihak manajemen.

2.1.4 Sistem Manajemen Pengajuan

Sistem manajemen pengajuan adalah sistem yang dirancang untuk mempermudah dan mengotomatiskan pengelolaan permintaan yang masuk. Sistem ini melibatkan proses pengajuan, verifikasi, persetujuan, serta pemantauan status pengajuan. Menurut Alfarisy dan Sutabri (2020), *sistem manajemen pengajuan* bertujuan untuk meminimalisir kesalahan input data dan meningkatkan efisiensi proses, dengan mengandalkan alur kerja yang sudah terstandarisasi.

Pada sistem pengajuan kegiatan akademik di PTPN IV, sistem ini akan digunakan untuk mengelola pengajuan magang, kunjungan, dan penelitian. Dengan menggunakan sistem berbasis *web* yang terintegrasi, seluruh data akan lebih mudah dipantau dan dikelola, serta mendukung proses pengambilan keputusan yang lebih cepat dan tepat

2.1.5 Pengembangan Back-End

Pengembangan *back-end* adalah proses pembangunan komponen aplikasi yang berjalan di sisi *server*. Pengembangan *back-end* mencakup pengelolaan basis data, pengolahan data, otorisasi pengguna, dan pembuatan *API* untuk menghubungkan *server* dengan *front-end*. Menurut Nadiyono (2021), pengembangan *back-end* memiliki peran yang sangat penting dalam keberhasilan suatu aplikasi *web*, karena *back-end* bertanggung jawab atas manajemen data dan logika bisnis yang menggerakkan aplikasi.

Selain itu, keberhasilan pengembangan *back-end* sangat dipengaruhi oleh pemilihan arsitektur yang tepat dan implementasi struktur kode yang efisien. Salah satu pendekatan yang umum digunakan adalah pemisahan logika bisnis dan pengelolaan data melalui arsitektur *Model-View-Controller (MVC)*. Struktur ini

memungkinkan pengembang untuk mengelola tanggung jawab sistem secara modular, mempermudah proses pemeliharaan, serta mempercepat deteksi dan perbaikan kesalahan (*debugging*). Dalam konteks sistem yang membutuhkan skalabilitas dan integrasi yang tinggi seperti layanan pengajuan kegiatan akademik berbasis *REST API*, penerapan prinsip modular dan *separation of concerns* pada sisi *back-end* menjadi krusial untuk menjaga konsistensi, keamanan, dan performa sistem secara keseluruhan.

2.1.6 Framework

Framework yang digunakan dalam pengembangan *back-end* sangat menentukan efisiensi dan keandalan sistem yang dibangun. Menurut Putra dan Wulandari (2022), *Laravel* adalah *framework PHP* berbasis arsitektur *MVC* yang mempermudah pengembangan *back-end* dengan menyediakan banyak fitur untuk *routing*, *middleware*, validasi, dan *ORM* (*Object Relational Mapping*). Selain itu, *framework* ini juga mempermudah pengembangan *REST API* dengan menyediakan fitur seperti *API Resource* dan *API Authentication*.

Di sisi lain, *Express.js* untuk *Node.js* juga menjadi pilihan populer dalam pengembangan *back-end*. *Framework* ini menyediakan kerangka kerja minimalis yang memungkinkan pengembangan *API* secara cepat dan fleksibel, serta mendukung pembuatan aplikasi *web* yang *scalable.*

2.1.7 Laravel

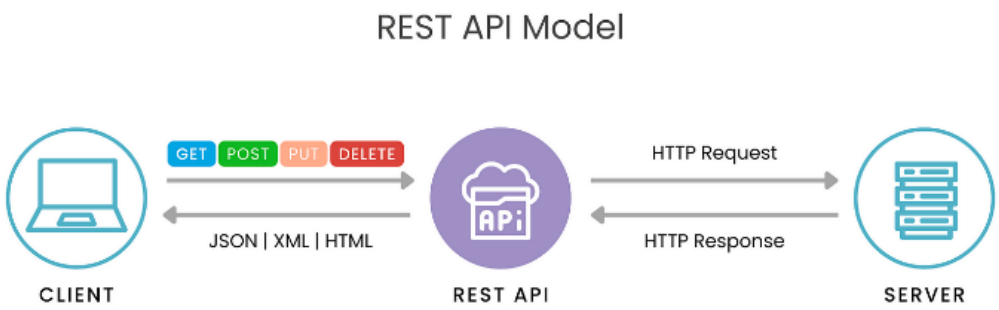
Laravel adalah *framework* open-source berbasis PHP yang dirancang untuk mempercepat dan menyederhanakan proses pengembangan aplikasi web modern. Framework ini mengadopsi pola arsitektur *Model-View-Controller (MVC)* yang memisahkan logika aplikasi dari tampilan dan manajemen data, sehingga struktur kode menjadi lebih terorganisir dan mudah untuk dikelola. Laravel juga menyediakan berbagai fitur *built-in* seperti sistem *routing*, *middleware*, manajemen sesi, otentikasi, serta *template engine* Blade yang mendukung pembuatan antarmuka pengguna yang dinamis. Dengan dokumentasi yang lengkap dan komunitas yang aktif, Laravel telah menjadi salah satu framework PHP yang paling banyak digunakan dalam pengembangan sistem berbasis web.

Dalam konteks pengembangan *back-end* berbasis *REST API*, Laravel menyediakan dukungan penuh melalui komponen HTTP client, *controller* berbasis *resource*, dan kemampuan untuk mendefinisikan *endpoint* dengan fleksibel. Selain itu, Laravel juga dilengkapi dengan Eloquent ORM (Object-Relational Mapping) yang memudahkan interaksi dengan basis data secara deklaratif dan aman. Melalui fitur *migration* dan *seeder*, pengembang dapat mengelola struktur database secara versioning dan otomatis. Oleh karena itu, Laravel menjadi pilihan yang relevan dan strategis untuk membangun sistem pengajuan kegiatan akademik yang membutuhkan fleksibilitas, skalabilitas, dan struktur modular dalam implementasinya

2.1.8 Rest API

REST API (*Representational State Transfer*) adalah arsitektur komunikasi web yang menggunakan protokol HTTP untuk melakukan pertukaran data antara *client* dan *server*. Narwastu, dkk. (2022) menjelaskan bahwa REST API memanfaatkan metode standar seperti GET, POST, PUT, dan DELETE dalam mengelola sumber daya yang diidentifikasi melalui URL. Keuntungan utama dari penggunaan REST API adalah kemudahan integrasi antar sistem serta efisiensi pengolahan data, yang membuat aplikasi berjalan lebih fleksibel. REST API juga menggunakan format data ringan seperti JSON, sehingga komunikasi menjadi lebih cepat dan hemat *bandwidth* dibandingkan dengan format lain seperti XML.

Untuk mendukung pemahaman terhadap konsep tersebut, ilustrasi model komunikasi REST API ditampilkan pada Gambar 2.2 Rest Api Model. Gambar tersebut diambil dari sumber terbuka di internet dan digunakan sebagai visualisasi pendukung terhadap penjelasan teori yang diuraikan. Visual ini menggambarkan bagaimana proses permintaan dan respons terjadi antara *client*, API, dan *server* melalui protokol HTTP secara terstruktur dan efisien.



Gambar 2. 2 REST API Model

2.1.9 Postman

Postman adalah platform populer yang digunakan untuk mengembangkan, menguji, dan mendokumentasikan *API*. Dengan antarmuka grafis yang ramah pengguna, *Postman* memungkinkan pengembang dan penguji untuk mengirim permintaan *HTTP*, memeriksa respons, dan mengotomatiskan pengujian *API* tanpa menulis kode yang rumit. Menurut Chandrika ARRN (2022), *Postman* menyediakan fitur seperti pembuatan koleksi permintaan, pengelolaan variabel lingkungan, dan kemampuan *scripting* menggunakan *JavaScript* untuk menulis skrip pengujian. Fitur-fitur ini memungkinkan pengujian fungsionalitas *API* secara manual maupun otomatis, serta mendukung pengujian integrasi dan regresi dalam siklus pengembangan perangkat lunak.

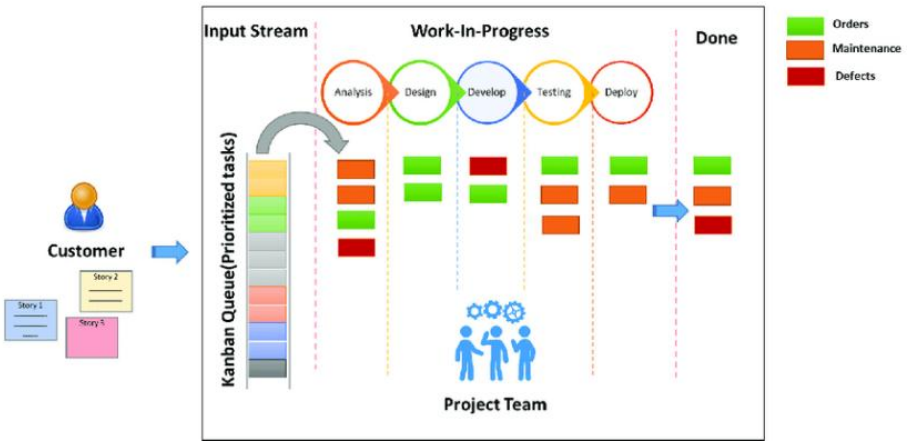
Dalam konteks pengembangan sistem manajemen pengajuan di PTPN IV, *Postman* dapat digunakan untuk memastikan bahwa *REST API* yang dikembangkan berfungsi sesuai dengan spesifikasi yang diinginkan. Dengan menggunakan *Postman*, tim pengembang dapat melakukan pengujian terhadap berbagai *endpoint API*, memverifikasi status kode respons, dan memastikan bahwa data yang dikirim dan diterima sesuai dengan format yang diharapkan. Hal ini penting untuk menjaga kualitas dan keandalan sistem yang dibangun.

Gambar 2. 3 Postman

2.1.10 Kanban

Kanban adalah metodologi manajemen proyek yang memfokuskan pada visualisasi alur kerja dan pengelolaan tugas dengan tujuan untuk meningkatkan efisiensi dan mengurangi pemborosan. Menurut Huda dan Zainudin (2021), *Kanban* membantu tim untuk mengidentifikasi hambatan dalam alur kerja dan memprioritaskan pekerjaan dengan menggunakan papan *Kanban* yang terdiri dari kolom seperti *Backlog*, *In Progress*, dan *Done*. Dengan pendekatan visual ini, anggota tim dapat dengan mudah memantau progres proyek dan menyesuaikan prioritas tugas secara fleksibel.

Metode *Kanban* sangat cocok diterapkan dalam pengembangan *REST API* karena sifatnya yang iteratif, memfasilitasi pengembangan secara bertahap dan memastikan bahwa setiap fitur yang selesai segera dapat dipindahkan ke tahap pengujian. Kanban Board dapat dilihat pada Gambar 2. 4 Kanban Board.



Gambar 2. 4 Kanban Board

2.1.11 Pengujian

Pengujian sistem adalah tahapan krusial dalam pengembangan perangkat lunak untuk memastikan bahwa sistem berfungsi sesuai dengan yang diharapkan dan memenuhi kebutuhan pengguna. Pengujian ini melibatkan beberapa jenis uji, seperti pengujian fungsionalitas, pengujian keamanan, dan pengujian performa.

Menurut Waskitho Wibisono dan Fajar Baskoro (2002), pengujian sistem dilakukan untuk mencari kesalahan perangkat lunak yang dikembangkan. Tahap analisis, desain, dan implementasi perangkat lunak tidak menjamin bahwa perangkat lunak bebas kesalahan. Oleh karena itu, diperlukan suatu tahap pengujian untuk menemukan kesalahan-kesalahan yang ada pada perangkat lunak.

2.1.12 Blackbox Testing

Black-box testing adalah jenis pengujian perangkat lunak di mana penguji tidak mengetahui struktur internal dari aplikasi yang diuji. Fokus dari pengujian ini adalah untuk memverifikasi apakah aplikasi memberikan *output* yang benar berdasarkan *input* yang diberikan. Suwirmayanti dan Ady (2020) menjelaskan bahwa *black-box testing* sangat efektif untuk menguji fungsionalitas aplikasi, di mana penguji akan memeriksa apakah aplikasi berfungsi sesuai dengan yang diharapkan tanpa mempertimbangkan kode atau arsitektur internal sistem.

Selain digunakan untuk memastikan kesesuaian antara input dan output sistem, *blackbox testing* juga memungkinkan pengujian berbagai skenario yang merepresentasikan perilaku pengguna akhir secara nyata. Metode ini sangat cocok diterapkan pada sistem berbasis *REST API*, karena pengujian dapat dilakukan terhadap setiap *endpoint* tanpa perlu mengetahui struktur kode internal. Dengan demikian, pengembang dapat mengevaluasi respons sistem terhadap berbagai jenis permintaan, validasi parameter, serta menangani kemungkinan *error* yang muncul, sehingga memastikan kestabilan dan keandalan fungsi aplikasi sebelum digunakan oleh pengguna akhir.

2.1.13 Unit Testing

Unit testing adalah jenis pengujian yang dilakukan untuk memverifikasi bahwa setiap unit terkecil dalam aplikasi (seperti fungsi atau metode) berfungsi dengan baik. Agustinus (2021) menjelaskan bahwa *unit testing* sangat penting

dalam memastikan bahwa kode yang ditulis berfungsi sesuai dengan yang diinginkan sebelum digabungkan menjadi aplikasi yang lebih besar. Pengujian unit sering kali dilakukan secara otomatis menggunakan alat pengujian seperti *JUnit* untuk *Java* atau *pytest* untuk *Python*.

Lebih jauh, *unit testing* juga mendukung prinsip *test-driven development (TDD)*, yaitu pendekatan di mana pengujian ditulis terlebih dahulu sebelum pengembangan fungsionalitas dilakukan. Pendekatan ini tidak hanya membantu menjaga konsistensi logika sistem selama proses pengembangan, tetapi juga memungkinkan deteksi dini terhadap perubahan yang dapat menyebabkan gangguan fungsional. Dalam konteks sistem berbasis *modular API*, penerapan *unit testing* penting untuk memastikan setiap fungsi atau komponen berjalan secara independen, terutama dalam pengolahan data, otentikasi, dan pengiriman notifikasi.

2.1.14 Performance Testing

Performance testing merupakan jenis pengujian yang berfokus pada evaluasi kinerja sistem dalam menangani beban kerja tertentu, baik dalam kondisi normal maupun ekstrem. Pengujian ini mencakup metode seperti *load testing*, yang mengukur kemampuan sistem dalam menangani banyak permintaan secara bersamaan, serta *stress testing*, yang dilakukan untuk mengetahui batas maksimum kapasitas sistem sebelum mengalami kegagalan. Menurut Ginasari, Wibawa, dan Wirdiani (2021), pengujian performa pada layanan *API* menggunakan Apache JMeter dapat membantu mengidentifikasi *throughput*, waktu tanggap rata-rata, dan kestabilan sistem dalam berbagai skenario pengaksesan. Sementara itu, Hendayun, Ginanjar, dan Ihsan (2022) menekankan bahwa pengujian ini penting dilakukan untuk mengurangi risiko *bottleneck* dan memastikan sistem tetap responsif saat digunakan dalam lingkungan operasional nyata.

Performance testing akan digunakan untuk menilai keandalan berbagai *endpoint* saat menerima permintaan secara bersamaan, seperti proses pengajuan, verifikasi, dan notifikasi. Pengujian akan dilakukan menggunakan alat seperti Apache JMeter atau k6, dengan indikator yang diuji antara lain *response time*, *latency*, dan tingkat kegagalan permintaan. Hasil dari pengujian ini diharapkan dapat menjadi dasar dalam optimalisasi konfigurasi server, perbaikan logika pemrosesan, serta peningkatan

2.2 Penelitian Terdahulu

Penelitian terdahulu merupakan landasan penting dalam perancangan dan pengembangan sistem *back-end* pengajuan kegiatan akademik berbasis *REST API* di PT. Perkebunan Nusantara IV Regional III. Dengan mengkaji berbagai studi sebelumnya, diperoleh wawasan terkait desain arsitektur *API*, perancangan basis data, mekanisme autentikasi, dan pendekatan pengembangan sistem yang relevan untuk kebutuhan pengelolaan akademik secara digital. Di samping itu, keterbatasan pada masing-masing penelitian menjadi pijakan dalam menyusun sistem yang lebih terintegrasi dan adaptif terhadap kebutuhan pengguna.

Penelitian oleh L. Lestari, dkk. (2019), yang mengembangkan sistem informasi pendaftaran magang berbasis *web* dengan metode *Waterfall*. Sistem ini menggunakan *PHP* dan *MySQL*, serta menyediakan fitur manajemen data peserta magang, penyimpanan hasil seleksi, dan pengelolaan divisi. Meskipun bermanfaat dalam konteks operasional, sistem ini tidak memanfaatkan pendekatan *REST API*, sehingga menyulitkan integrasi dengan layanan lain dan tidak mendukung komunikasi lintas antarmuka.

Penelitian oleh N. Fitria, dkk. (2024), membahas sistem pengajuan judul tugas akhir dengan dukungan alur validasi dan persetujuan digital. *Back-end* sistem dibangun menggunakan *PHP* dan *MySQL*, serta dikembangkan dengan pendekatan *Prototyping*. Walaupun cukup lengkap dari sisi alur akademik, sistem ini belum menerapkan konsep *API*-berbasis layanan, sehingga interoperabilitas dengan aplikasi lain belum terjamin.

Penelitian oleh E. Jamasnia, dkk. (2021), memfokuskan pengembangan sistem berbasis *RESTful Web Service* untuk mendukung komunikasi dua arah antara mahasiswa dan program studi. Meskipun sudah menerapkan konsep *REST*, penelitian ini belum membahas lebih jauh aspek pengamanan seperti autentikasi *token* atau pengelolaan hak akses, serta tidak menguraikan modularitas sistem yang diperlukan untuk skala pengembangan lebih besar.

Penelitian oleh Harlina, dkk. (2024), yang membangun sistem pengelolaan magang dengan fitur *login* multi-peran dan validasi data input. Sistem dikembangkan menggunakan *PHP* dan *MySQL*. Meskipun mencakup manajemen peran pengguna, sistem ini belum menerapkan pemisahan antara logika bisnis dan

kontrol data (seperti pada arsitektur *MVC*), dan tidak menyediakan *API endpoint* untuk komunikasi antar layanan.

Penelitian oleh L. Suswitasari, dkk. (2022), menggunakan *framework* Laravel untuk membangun sistem informasi pengajuan magang dengan fitur autentikasi, manajemen akun multi-peran, serta pengunggahan dokumen. Penggunaan *Eloquent ORM* memberikan kemudahan dalam mengelola data secara efisien. Namun, *REST API* belum menjadi inti komunikasi dalam sistem ini, sehingga fleksibilitas integrasi dengan modul *front-end* lain masih terbatas.

Tabel 2. 1 Penelitian Terdahulu

Penulis	**Jenis Perbandingan**			
	Judul	**Metode**	**Bahasa Pemograman**	**Hasil**
Ayu Lestari, Fanani Bastian A., Mega Novita (*Riady, dkk.,* 2024)	*Sistem Informasi Magang Berbasis Website pada Dinas Kesehatan Provinsi Jawa Tengah*	*Waterfall*	*PHP, MySQL*	Mempermudah pendaftaran dan pengelolaan data magang secara *online* oleh *admin* dan *user*.
Nazla Fitria, Fitri Handayani N., Aldimas (*Widayana, dkk.,* 2021)	*Perancangan Sistem Informasi Pengelolaan Judul Proposal Tugas Akhir Berbasis Web*	*Prototyping*	*PHP, MySQL*	Meningkatkan efisiensi dan transparansi pengelolaan judul tugas akhir secara digital.

Penulis	Jenis Perbandingan			
	Judul	**Metode**	**Bahasa Pemograman**	**Hasil**
Erwin Jamasnia, Muhammad Assidiq, Ul Khairat	*Sistem Informasi Pengajuan Judul Skripsi Berbasis Web Service*	*Kualitatif*	*PHP, Web Service (REST API)*	Memfasilitasi komunikasi dua arah antara mahasiswa dan ketua program studi dalam pengajuan judul.
MS Harlina, MS Herawati, Kosdiana	*Perancangan Sistem Informasi Pendaftaran Magang Mahasiswa di CV. Bolist Art GRC*	*Waterfall*	*PHP, MySQL, HTML, JavaScript*	Sistem mempercepat proses input data dan validasi pendaftaran magang serta mempermudah pengelolaan arsip secara digital.
Linda Suswitasari	*Rancang Bangun Aplikasi Pengajuan PKL pada Biro Kesejahteraan Rakyat SETDA Provinsi Sumsel*	*Waterfall*	*Laravel (PHP), MySQL*	Sistem mendukung multi-role user dengan validasi dan otomatisasi surat balasan, sehingga mempercepat alur administrasi pengajuan PKL.

Berdasarkan telaah penelitian terdahulu, perancangan dan implementasi sistem *back-end* pengajuan kegiatan akademik dengan pendekatan *REST API* di PT. Perkebunan Nusantara IV Regional III memiliki nilai strategis dalam mendukung digitalisasi administrasi akademik. Pendekatan ini memungkinkan layanan yang modular, fleksibel, dan mudah dikembangkan sehingga mempercepat proses verifikasi pengajuan serta meningkatkan akurasi data. Penelitian oleh E. Jamasnia, dkk. menunjukkan efektivitas arsitektur *REST* dalam membangun komunikasi dua arah secara *real-time* antara mahasiswa dan pengelola akademik. Studi oleh N. Fitria, dkk. menekankan pentingnya desain *back-end* yang mendukung persetujuan otomatis dan pengelolaan data berbasis validasi berlapis, meskipun sistem mereka belum berbasis layanan terbuka (*API*). Selain itu, penggunaan *framework* Laravel dalam penelitian L. Suswitasari, dkk. memperlihatkan keunggulan penerapan prinsip *MVC* dan *ORM* yang selaras dengan konsep *REST API* modern.

Berbeda dari sistem yang dikembangkan pada penelitian sebelumnya, proyek ini menghadirkan fitur tambahan seperti endpoint khusus untuk pemantauan status pengajuan, integrasi notifikasi berbasis email, dan pengelolaan peran yang lebih adaptif terhadap kebutuhan pengguna internal maupun eksternal. Dengan mengadopsi layanan terbuka dan pola kerja yang terstruktur, sistem ini diharapkan mampu mengatasi keterbatasan sistem konvensional yang masih sering ditemukan serta menjadi solusi tangguh untuk meningkatkan efisiensi proses administrasi dan keterpaduan data di lingkungan PT. Perkebunan Nusantara IV Regional III