BAB 2
DASAR TEORI

www.itk.ac.id

Bab ini menjelaskan teori-teori yang menjadi dasar dalam melakukan penelitian, yang bersumber dari buku, jurnal ataupun artikel. Tujuannya agar konsep dan teori-teori penyelesaian masalah yang digunakan dapat dipahami.

2.1 Pangkalan Data Pendidikan Tinggi

Pangkalan Data Pendidikan Tinggi atau PD DIKTI adalah *database* yang digunakan untuk membantu penyelenggaraan kegiatan pendidikan tinggi di seluruh perguruan tinggi agar terintegrasi secara nasional. Seperti disebutkan pada UU No. 12 tahun 2012 pasal 56 ayat 2, menjelaskan bahwa Pangkalan Data Pendidikan Tinggi memiliki fungsi sebagai sumber informasi untuk:

1. Lembaga akreditasi, untuk melakukan akreditasi Program Studi dan Perguruan Tinggi;
2. Pemerintah, untuk melakukan pengaturan, perencanaan, pengawasan, pemantauan, dan evaluasi serta pembinaan dan koordinasi Program Studi dan Perguruan Tinggi; dan
3. Masyarakat, untuk mengetahui kinerja Program Studi dan Perguruan Tinggi.

Lalu disebutkan juga bahwa setiap perguruan tinggi wajib melaporkan data perguruan tinggi ke pangkalan data pendidikan tinggi sebagaimana diatur dalam Keputusan Menteri Pendidikan Nasional no 184/U/2001 pada pasal 5. Namun, apabila sebuah perguruan tinggi tidak melaporkan data ke PD DIKTI secara berkala maka perguruan tinggi tersebut akan dikenakan sanksi administratif ringan sebagaimana diatur dalam Peraturan Mentri Riset, Teknologi, dan Pendidikan Tinggi nomor 100 tahun 2016 pada pasal 26 ayat 1 poin (m). Yang kemudian disebutkan pada peraturan yang sama pasal 29 ayat 1 bahwa sanksi administratif ringan berupa peringatan secara tertulis yang diberikan ke perguruan tinggi. Apabila perguruan tinggi mengabaikan peringatan tersebut, maka sanksi administratif akan ditingkatkan ke sedang. Lalu apabila sanksi administratif sedang

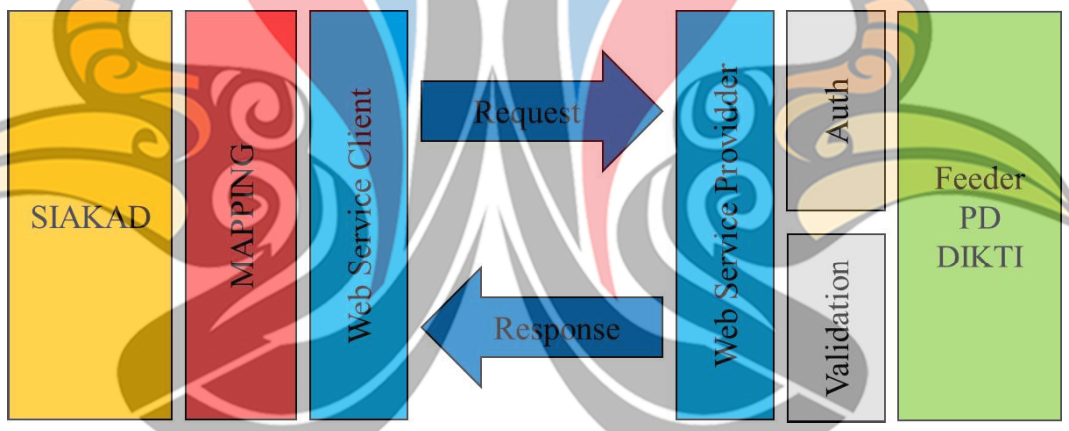
www.itk.ac.id

tetap diabaikan, maka perguruan tinggi dikenakan sanksi administratif besar (DIKTI, 2017).

www.itk.ac.id

2.2 *Feeder Importer*

Feeder Importer atau *Feeder* adalah sebuah aplikasi yang membantu perguruan tinggi dalam melakukan pelaporan data pendidikan tinggi. Aplikasi *Feeder* PD DIKTI merupakan aplikasi yang telah disediakan oleh PD DIKTI. Aplikasi ini memiliki fungsi untuk melakukan pengelolaan data mahasiswa dan data perkuliahan di setiap perguruan tinggi di Indonesia. Masing-masing perguruan tinggi wajib mengelola aplikasi *Feeder* PD DIKTI. Data yang telah diinputkan ke aplikasi *Feeder* ini dapat ditampilkan di sistem aplikasi *online report* (Forlap) yang dikelola oleh DIKTI (DIKTI, 2017). Aplikasi *Feeder* PD DIKTI juga menyediakan *web service* yang dapat diakses oleh setiap perguruan tinggi.



Gambar 2.1 Skema *Feeder* PD DIKTI (DIKTI, 2017)

Pada Gambar 2.1 dijelaskan bahwa *Feeder* PD DIKTI menyediakan layanan yang dapat digunakan oleh perguruan tinggi untuk dapat melakukan hubungan antara *Feeder* PD DIKTI dengan sistem informasi yang telah berjalan di lingkungan perguruan tinggi. Sumber data yang digunakan untuk memenuhi kebutuhan PD DIKTI dapat berasal dari sebuah sistem informasi yang sudah ada di setiap perguruan tinggi, tetapi data-data dari sistem informasi di tiap perguruan tinggi harus terlebih dahulu diformat melalui proses *mapping* dan menghasilkan data yang sesuai dengan format data PD DIKTI (DIKTI, 2017). Cara melakukan *mapping* data adalah dengan membandingkan data Sikantan dengan data PD DIKTI. Apabila terdapat data Sikantan yang tidak dibutuhkan oleh PD DIKTI maka data tersebut

www.itk.ac.id

tidak dilaporkan. Namun apabila terdapat data yang dibutuhkah oleh PD DIKTI namun tidak tersedia di Sikantan, maka perlu penambahan fungsi pada aplikasi untuk memenuhi pelaporan data ke PD DIKTI.

Dalam pengembangan aplikasi *Feeder*, beberapa *developer* telah mengembangkan aplikasi *Feeder* yang serupa dengan *Feeder* PD DIKTI namun memiliki fitur-fitur yang berbeda. Salah satu contohnya adalah Sevima GoFeeder. Sevima GoFeeder merupakan sebuah layanan sistem informasi akademik yang membantu perguruan tinggi untuk proses pelaporan ke PD DIKTI. GoFeeder juga memudahkan kampus dalam manajemen dan pengelolaan akademik (Sevima, 2017). Perbedaan yang ada pada pengembangan aplikasi *Feeder* data Pendidikan tinggi ITK dengan Sevime GoFeeder adalah, GoFeeder tidak dapat diubah sesuai kebutuhan ITK karena GoFeeder merupakan aplikasi yang telah selesai dikembangkan. Sedangkan, *Feeder* data pendidikan tinggi ITK dapat menyesuaikan kebutuhan dari user di ITK yaitu tenaga kependidikan di bidang akademik.

2.3 *Web Service*

Web service merupakan sebuah perangkat lunak yang dibuat untuk memudahkan komunikasi antar mesin melalui sebuah jaringan. Secara teknis, *web service* memiliki layanan yang bersifat terbuka untuk pengumpulan data dan informasi yang kemudian dapat diakses dengan internet oleh berbagai pihak menggunakan teknologi yang berbeda-beda sesuai dengan *user* dari *web service* tersebut. *Application Programming Interface* (API) memiliki fungsi yang sama dengan *web service*, namun *web service* memiliki kelebihan jika dibandingkan dengan API karena *web service* dapat diakses dari jarak yang jauh dengan hanya menggunakan internet. Dalam proses penggunaan data, *web service* dapat menggunakan bahasa pemrograman dan *platform* apapun (Sutrisno, et al., 2016).

PD DIKTI juga telah menyediakan *Web Service* yang dapat digunakan oleh setiap perguruan tinggi untuk membantu proses pelaporan data ke PD DIKTI. Total dari *method web service* yang disediakan oleh PD DIKTI sejumlah 148. Seluruh *method* tersebut dapat digunakan setiap perguruan tinggi dalam membuat aplikasi *Feeder* data pendidikan tinggi yang mendukung fungsi *web service client*. Fungsi

tersebut memungkinkan sebuah aplikasi dalam mengakses *method-method* yang telah disediakan oleh *web service* PD DIKTI.

www.itk.ac.id

Tabel 2.1 Daftar *method web service*

No	Nama Method	Keterangan
1	GetToken	Mendapatkan Token untuk dipakai sebagai parameter di fungsi *web service* lainnya.
2	GetDictionary	Mendapatkan struktur data dari suatu tabel.
3	GetListPrestasiMahasiswa	Mendapatkan daftar prestasi mahasiswa.
4	InsertPrestasiMahasiswa	*Insert* prestasi mahasiswa.
5	UpdatePrestasiMahasiswa	*Update* prestasi mahasiswa.
6	DeletePrestasiMahasiswa	*Delete* prestasi mahasiswa.
7	GetListAktivitasMahasiswa	Mendapatkan daftar aktivitas mahasiswa.
8	InsertAktivitasMahasiswa	*Insert* aktivitas mahasiswa.
9	UpdateAktivitasMahasiswa	*Update* aktivitas mahasiswa.
10	DeleteAktivitasMahasiswa	*Delete* aktivitas mahasiswa.
11	GetListAnggotaAktivitasMahasiswa	Mendapatkan daftar anggota aktivitas mahasiswa.
12	InsertAnggotaAktivitasMahasiswa	*Insert* anggota aktivitas mahasiswa.
13	DeleteAnggotaAktivitasMahasiswa	*Delete* anggota aktivitas mahasiswa.

www.itk.ac.id

No	Nama Method	Keterangan
14	GetListBimbingMahasiswa	Mendapatkan daftar bimbingan mahasiswa.
15	InsertBimbingMahasiswa	*Insert* bimbingan mahasiswa.
16	DeleteBimbingMahasiswa	*Delete* bimbingan mahasiswa.
17	GetListUjiMahasiswa	Mendapatkan daftar penguji mahasiswa.
18	InsertUjiMahasiswa	*Insert* penguji mahasiswa.
19	DeleteUjiMahasiswa	*Delete* penguji mahasiswa.
20	GetPembiayaan	Mendapatkan daftar pembiayaan.
21	GetJenisPrestasi	Mendapatkan daftar jenis prestasi.
22	GetTingkatPrestasi	Mendapatkan daftar tingkat prestasi.
23	GetJenisAktivitasMahasiswa	Mendapatkan daftar jenis aktivitas mahasiswa.
24	GetKategoriKegiatan	Mendapatkan daftar ketogori kegiatan.

Pada Tabel 2.1 terdapat 24 *method* dari total 148 *method* yang telah disediakan oleh *web service* PD DIKTI. Seluruh *method* tersebut dapat digunakan untuk mengambil data dan melaporkan data. *Method GetToken* berguna untuk mengambil nilai *token* yang kemudian dapat digunakan sebagai kunci agar *method* lain dapat digunakan. *Method* ini memberikan *username* dan *password* kepada *programmer* agar dapat mengembangkan aplikasi menyesuaikan dengan *web service* yang telah disediakan.

www.itk.ac.id

2.4 *Agile Software Development*

www.itk.ac.id

Metode pengembangan perangkat lunak dengan *Agile* adalah model pengembangan perangkat lunak yang digunakan ketika kebutuhan dari sistem biasanya berubah dengan cepat selama proses pengembangan perangkat lunak berlangsung. Metode ini sering sekali digunakan untuk menghilangkan dokumentasi formal agar proses pengembangan perangkat lunak menjadi lebih cepat. Untuk prinsip yang ada di metode pengembangan perangkat lunak dengan *Agile* dijelaskan dalam Tabel 2.2

Tabel 2.2 *The Principles of Agile Methods*

Principles	***Description***
Customer Involvement	Pelanggan harus terlibat dalam proses pengembangan. Peran mereka adalah menyediakan dan memprioritaskan persyaratan sistem baru dan untuk mengevaluasi iterasi sistem.
Incremental Delivery	Perangkat lunak ini dikembangkan secara bertahap dengan pelanggan yang menetapkan persyaratan untuk dimasukkan dalam setiap *increment*.
People Not Process	Keterampilan tim pengembangan harus diakui dan dieksploitasi. Anggota tim harus dibiarkan untuk mengembangkan cara kerja mereka sendiri tanpa proses preskriptif.
Embrace Change	Tim pengembang harus selalu bersiap-siap dengan segala perubahan yang diberikan oleh pelanggan.
Maintain Simplicity	Fokus pada kesederhanaan dalam perangkat lunak yang sedang dikembangkan dan dalam proses pengembangan. Sebisa mungkin menghilangkan kompleksitas dari sistem.

www.itk.ac.id

Dalam metode pengembangan perangkat lunak dengan menggunakan *Agile,* masih banyak contoh model proses yang dapat digunakan. Beberapa contoh yang model proses dari *Agile Development* adalah SCRUM, *Dynamic Systems Development Method* (DSDM), *Adaptive Software Development* (ASD), dan *Extreme Programming* (XP) (Sommerville, 2011).

Pengembangan perangkat lunak dengan *Agile* sesuai dengan pengembangan aplikasi *Feeder* data pendidikan tinggi ITK, karena pengembangan perangkat lunak dengan *Agile* dikembangkan secara cepat menyesuaikan waktu yang dibutuhkan dalam menyelesaikan penelitian selama empat bulan.

2.5 Scrum

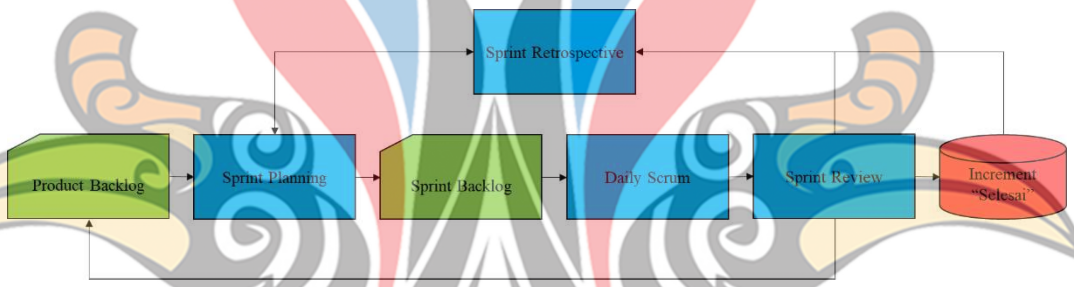
Dalam pengembangan perangkat lunak menggunakan metode *agile*, terdapat sebuah kerangka kerja yang dapat mengatasi masalah secara kompleks dan adaptif, dan pada saat yang sama kerangka kerja tersebut dapat menghasilkan produk dengan nilai yang tinggi secara produktif dan kreatif. Kerangka kerja ini dikenal dengan *Scrum*. Kerangka kerja ini memiliki sifat ringan, mudah untuk dipahami, dan sulit untuk dikuasai. *Scrum* bukanlah sebuah proses, teknik, ataupun metodologi. Akan tetapi *Scrum* adalah sebuah kerangka kerja yang dapat digabungkan dengan berbagai macam proses dan teknik pengerjaan di dalamnya (Schwaber & Sutherland, 2017).

Dalam menerapkan kerangka kerja dari *Scrum*, sekelompok orang yang terlibat dinamakan *Scrum Team*. Didalam *Scrum Team* terdiri dari *Product Owner*, *Development Team*, dan *Scrum Master*. Tim ini bersifat mandiri dan memilih cara yang terbaik untuk menyelesaikan pekerjaan mereka. Tim ini dirancang untuk mengoptimalkan fleksibilitas, kreativitas, dan produktivitas (Schwaber & Sutherland, 2017).

Seseorang yang bertanggung jawab terhadap produk yang dihasilkan dari *Scrum* dikenal dengan *Product Owner* (PO). PO adalah satu orang yang bertanggung jawab dalam pengelolaan dan pembuatan *Product Backlog.* Dalam *Scrum,* PO mewakili pihak lain dalam penyampaian *backlog* apa saja yang akan diisi di *Product Backlog*. Berikutnya ada *Development Team* (DT), DT terdiri dari sekelompok ahli profesi yang bekerja untuk menyelesaikan *backlog* yang kemudian dipindahkan ke *increment* “selesai” yang dirilis setiap akhir *sprint* dan wajib

tersedia pada saat *Sprint Review*. Lalu ada *Scrum Master* (SM), yang bertanggung jawab atas penggunaan *Scrum* di *Scrum Team*. SM merupakan pemimpin yang melayani *Scrum Team*, dan membantu beberapa pihak di luar *Scrum Team* untuk memahami interaksi mana yang bermanfaat dan tidak bermanfaat dalam pengisian *Product Backlog* (Schwaber & Sutherland, 2017).

Product Backlog adalah sebuah daftar terurut dari semua hal yang telah diketahui hingga saat ini yang harus ada di dalam produk. *Product Backlog* adalah satu-satunya sumber kebutuhan yang menjelaskan apa saja yang dapat dikerjakan oleh product. *Product Owner* bertanggung jawab atas pembuatan dari *Product Backlog*. Sebuah *Product Backlog* tidak pernah tuntas. Selama pengembangan produk, *Product Backlog* akan berubah ubah sesuai dengan kebutuhan produk dan lingkungan dari dimana produk tersebut digunakan (Schwaber & Sutherland, 2017).



Gambar 2.2 Metode Pengembangan *Scrum* (Schwaber & Sutherland, 2017)

Dalam penerapan *Scrum* terdapat acara-acara wajib yang berguna untuk menciptakan kerutinan dan mengurangi jumlah pertemuan lain yang bukan bagian dari *Scrum*. Seperti dijelaskan oleh Gambar 2.2, Tiap *sprint* terdiri dari *Sprint Planning*, *Daily Scrum*, *Sprint Review* dan *Sprint Retrospective*. Setiap acara yang dikerjakan memiliki batasan waktu sehingga setiap acara yang terdapat di *Scrum* memiliki durasi waktu yang maksimal. Acara selain *sprint* dapat berakhir kapanpun juga ketika tujuan acara telah tercapai, hal tersebut dapat dilakukan untuk memastikan waktu yang digunakan tidak ada yang terbuang selama proses *Scrum* berlangsung (Schwaber & Sutherland, 2017).

Sprint merupakan jantung dari *Scrum*, yang berarti setiap *sprint* memiliki batasan waktu maksimal satu bulan atau dapat kurang. *Sprint* memiliki durasi yang tetap selama daur hidup pengembangan produk. *Sprint* yang baru akan langsung dimulai ketika *sprint* sebelumnya selesai. Setiap *sprint* dapat dianggap sebagai

www.itk.ac.id

sebuah proyek dengan durasi yang telah dibatasi, ketika durasi *sprint* terlalu lama maka berdampak pada proyek yang telah dikerjakan mengakibatkan produk yang dikembangkan dapat berubah, tingkat kompleksitas dari produk dapat meningkat, dan resiko dalam mengembangkan produk juga dapat meningkat (Schwaber & Sutherland, 2017).

Sprint Planning merupakan perencanaan untuk pekerjaan apa saja yang akan dikerjakan pada saat *sprint* berlangsung. Perencanaan ini dilakukan oleh seluruh anggota yang terlibat di *Scrum Team. Sprint Planning* memiliki batasan waktu sebanyak delapan jam untuk durasi *sprint* maksimal. Apabila durasi *sprint* lebih cepat, maka durasi *Sprint Planning* juga lebih cepat mengikuti durasi *sprint*. Dalam proses ini, *Scrum Master* menjaga dan menjelaskan kepada *Scrum Team* agar durasi pengerjaan tetap terjaga. Pada *Sprint Planning*, menghasilkan sebuah *Sprint Backlog* yang berisikan *item-item backlog* yang akan dikerjakan selama *sprint* berlangsung. Beberapa *item* yang ada di *Sprint Backlog* diambil dari *Product Backlog* yang telah dibuat sebelum *sprint* berlangsung (Schwaber & Sutherland, 2017).

Daily Scrum merupakan suatu acara yang diadakan untuk *Development Team* dan memiliki batasan waktu selama 15 menit. Acara ini dilakukan selama *sprint* berlangsung, dalam acara ini *Development Team* akan membuat agenda untuk merencanakan apa yang akan mereka lakukan selama 24 jam kedepan. *Daily Scrum* akan dilakukan di waktu dan tempat yang sama dengan pengembangan produk agar mengefektifkan waktu pengerjaan *sprint*. *Daily Scrum* dapat meningkatkan komunikasi antar *Development Team,* mengidentifikasi masalah bareng, berdiskusi mengenai pengambilan keputusan, dan meningkatkan pemahaman pada *Development Team* (Schwaber & Sutherland, 2017).

Sprint Review merupakan suatu acara yang dilaksanakan di setiap akhir dari *sprint* untuk menginspeksi *Increment* dengan membandingkan *backlog* yang telah dikerjakan dengan *Sprint Backlog* yang telah direncanakan. *Sprint Review* memiliki durasi maksimal selama empat jam. Hasil dari *Sprint Review* adalah *Product Backlog* yang sudah direvisi yang menjabarkan *Product Backlog item* yang mungkin diimplementasikan di *Sprint* berikutnya. *Product Backlog* juga dapat

www.itk.ac.id

disesuaikan secara keseluruhan untuk mendapatkan peluang baru dalam pengembangan produk yang berupa aplikasi (Schwaber & Sutherland, 2017).

Sprint Retrospective adalah suatu acara yang dilaksanakan oleh *Scrum Team* yang bertujuan untuk menginspeksi dan membuat perencanaan terkait peningkatan yang akan dilakukan di *sprint* berikutnya. *Scrum Master* memastikan acara ini terselenggara dan setiap peserta memahami tujuan dari pertemuan ini (Schwaber & Sutherland, 2017).

Increment adalah sebuah pengelompokan terhadap *Product Backlog item* yang telah diselesaikan dalam *sprint* dan total dari nilai *Increment* dari setiap *sprint* sebelumnya. Sebuah *Increment* adalah hasil pekerjaan yang bisa diuji dan telah selesai guna mendukung bukti akhir dari *sprint*. *Increment* adalah sebuah langkah untuk mencapai tujuan atau visi dari pengembangan produk. Ketika *Increment* telah dinyatakan "selesai", maka *Increment* harus dapat digunakan dan berfungsi terlepas *Product Owner* akan menggunakan dalam aplikasinya atau tidak (Schwaber & Sutherland, 2017).

Dalam pengembangan aplikasi dengan framework *Scrum* akan cocok dalam proses pengembangan aplikasi *Feeder* data pendidikan tinggi karena *Scrum* memiliki alur kerja yang jelas dan dokumentasi pada pengembangan aplikasi yang berupa *user stories*.

2.6 Javascript

Express.js merupakan *framework* JavaScript yang popular untuk mengatur bagian *Back-End* suatu aplikasi. Bagian *Back-End* umumnya menangani bagian *server side* dari suatu aplikasi. Beberapa contoh lain yang dapat dilakukan oleh *framework* Express.js adalah mengatur mekanisme *routing* melalui aplikasi, dan juga *session management*. Express.js dapat membantu untuk mengembangkan sebuah aplikasi berbasis *web* secara cepat namun tetap aman saat digunakan (Poulter, et al., 2015).

Angular.js merupakan *framework* JavaScript yang digunakan untuk mengatur bagian *Front-End* suatu aplikasi. *Framework* yang bersifat *open-source* ini digunakan untuk meningkatkan fungsi dari pemrograman *web* yang biasa. Angular.js juga dapat membantu pengembangan aplikasi berbasis *web* yang menerapkan metode *Single-Page Application* (SPA) (Poulter, et al., 2015).

Node.js adalah sebuah *platform* dalam JavaScript yang dapat menangani fungsi-fungsi untuk mengakses *input-output server* secara cepat. Node.js sangat cocok untuk pengembangan *web service* yang menggunakan RESTful (*Representational State Transfer*) yang menangani beberapa parameter dan mendapatkan data dari jumlah *resource* yang besar, karena Node.js dapat memproses ribuan data yang dibutuhkan secara bersamaan (Poulter, et al., 2015).

Vue.js atau disebut juga dengan Vue merupakan sebuah *framework* JavaScript yang memiliki istilah *progresif framework* digunakan untuk membangun *user interfaces* atau tampilan antarmuka untuk pengguna. *Framework* Vue memiliki *core library* yang berfokus pada *view layer* atau bagian tampilan dalam sebuah aplikasi. Dalam pembuatan sebuah proyek dengan menggunakan *tools* dan *library* dari Vue dapat membantu pengembangan sebuah aplikasi *single page* dengan lancar. Vue telah menjadi sebuah *framework open source* yang paling popular sejak pertama rilis. Banyak programmer di GitHub yang menggunakan *framework* tersebut dikarenakan Vue memiliki fitur yang banyak dan mudah digunakan (Gore,2017).

2.7 Penelitian Terdahulu

Dalam pengembangan aplikasi *Feeder* data pendidikan tinggi, beberapa perguruan tinggi telah melakukan penelitian di bidang yang serupa. Berikut merupakan penelitian terdahulu yang pernah dilakukan dijelaskan pada Tabel 2.2.

Tabel 2.3 Penelitian Terdahulu

No	Nama	Studi Kasus	Aplikasi	Metode	Hasil
1	Sutrisno, 2016	Perguruan Tinggi	Sistem Informasi Akademik	Pengembangan Aplikasi	Sistem informasi akademik terintegrasi dengan data *Feeder* PD DIKTI dengan pemanfaatan *web service* dan *framework Laravel*
2	Siswanto, 2016	Perguruan Tinggi	Aplikasi *Feeder*	Pembangunan aplikasi dengan *Waterfall*	Pembangunan aplikasi sinkronisasi *database* antara sistem informasi akademik STIKOM Dinamika Bangsa Jambi

www.itk.ac.id

No	Nama	Studi Kasus	Aplikasi	Metode	Hasil
					dengan *Feeder* PDDIKTI dengan teknologi *web service.*
3	Pranatawijaya, 2016	Perguruan Tinggi	Aplikasi Migrasi Data	Pengembangan Aplikasi	Pembuatan perangkat lunak migrasi data ke *Feeder* PDDIKTI untuk membantu pelaporan data dengan memetakan table EPSBED dengan table *Feeder* PD DIKTI
4	Perwira, 2017	Perguruan Tinggi	Aplikasi *web service*	Pembangunan aplikasi dengan *Waterfall*	Berupa sebuah *web service* yang terdiri dari tiga fungsi utama *insert*, *update*, dan *delete* untuk mengakomodasi pelaporan data akademik ke PD DIKTI.
5	Widodo, 2018	Perguruan Tinggi	Aplikasi *Feeder*	Pembangunan aplikasi dengan *Waterfall*	Mengembangkan aplikasi *web service* yang dapat mengakomodasi kebutuhan pelaporan data akademik yang sudah berjalan ke DIKTI

Pada tabel tersebut dijelaskan mengenai beberapa hasil dari penelitian terdahulu yang memiliki keterkaitan dengan penelitian yang akan dilakukan. Dari penelitian Slamet Widodo, diperoleh informasi mengenai banyaknya data akademik yang bervariasi sehingga menjadi kendala dalam proses pelaporan data ke PD DIKTI. Kendala tersebut menyebabkan proses pelaporan data ke PD DIKTI memakan waktu yang lama dan rentan mengalami kesalahan. Dari penelitian Agus Siswanto, mengenai *Feeder* PD DIKTI tidak menyediakan fitur *import* sehingga perlu pembuatan aplikasi *Feeder* dan memetakan data sesuai kebutuhan tabel *Feeder* PD DIKTI. Dari penelitian Agung Sutrisno, diperoleh hasil berupa aplikasi *Feeder* yang mengintegrasikan Sistem Informasi Akademik (SIAKAD) dengan data PD DIKTI dengan memanfaatkan implementasi *web service* dan *Framework Laravel.*

www.itk.ac.id

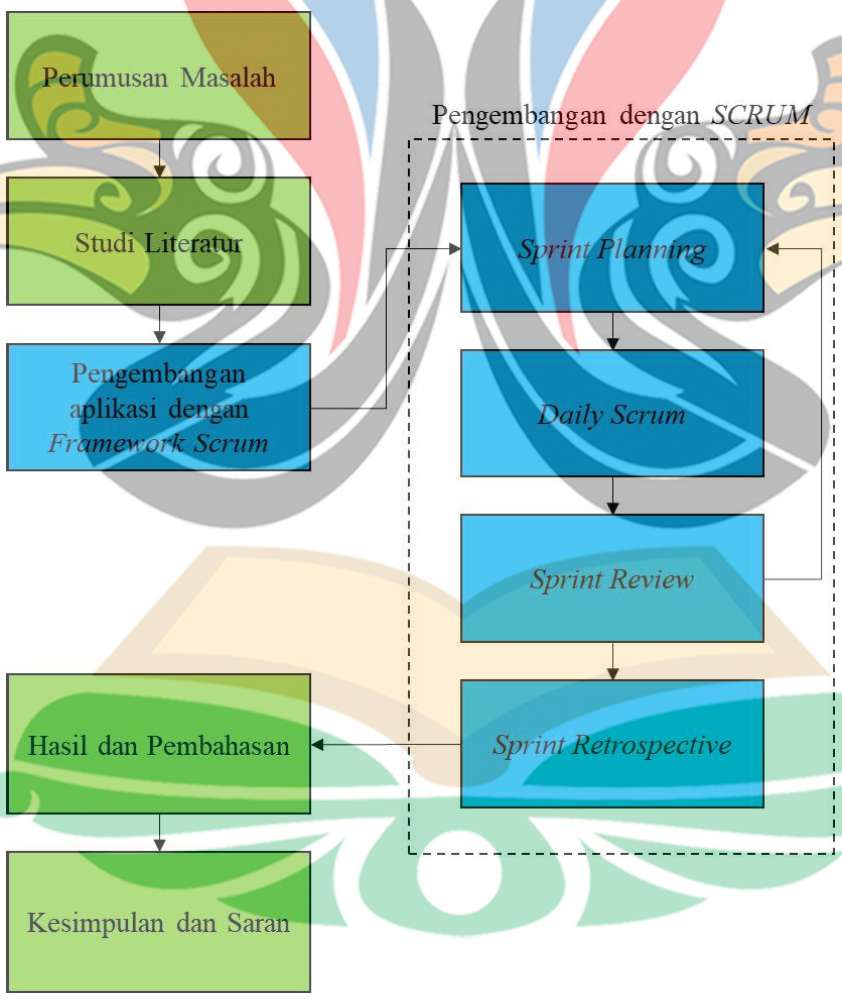
BAB 3
METODOLOGI

www.itk.ac.id

Bab ini menjelaskan mengenai metodologi yang digunakan dalam penelitian serta tahapan-tahapan yang dilakukan dalam pembuatan sistem informasi penerimaan mahasiswa baru.

3.1 Garis Besar Penelitian

Adapun garis besar alur penelitian ini dapat dilihat pada Gambar 3.1 sebagai berikut



www.itk.ac.id

Gambar 3.1 Prosedur Penelitian