BAB II
TINJAUAN PUSTAKA

2.1 Teori Dasar

2.1.1 System *Development Life Cycle* (SDLC)

System Development Life Cycle (SDLC) ialah suatu hubungan antara faktor-faktor yang tertentu atau disebut dengan model konseptual yang dapat dimanfaatkan dan digunakan untuk membangun proses manajemen proyek dalam realisasi fase-fase yang terkait dengan proyek perancangan dan pengembangan suatu sistem informasi, berikut ini beberapa tahapan dalam SDLC menurut (Rayeb, Maulana & Zhuhryanto, 2017) yaitu:

1. *System Planning*

   Tahap awal yaitu (*system planning)* perencanaan sistem dengan pengumpulan kebutuhan dan informasi pengguna sistem yang nantinya sebagai dasar dalam pengembangan sistem.

2. *System Analysis*

   Tahap selanjutnya (*system analysis*) analisis sistem yaitu melakukan analisis sistem yang sedang berjalan, hasil dari analisis digunakan sebagai acuan dalam *system design*.

3. *System Desain*

   Tahap system desain yaitu diagram digunakan sebagai representasi dari desain proses dan desain database dan gambar digunakan sebagai representasi dari desain visual (*mockup*) pada sistem yang dirancang atau dikembangkan.

4. *Implementation*

   Proses implementasi sistem merupakan proses perancangan perangkat dalam melakukan pekerjaan sistem seperti penulisan kode program yang didasarkan dari hasil *system design*.

5. *Maintenance*

tahapan maintenance merupakan perawatan sistem untuk mengatasi masalah yang muncul agar langsung diperbaiki pada perangkat lunak dan pengembangan sistem dimulai kembali dari tahap perencanaan sistem.

2.1.2 *Metodology Iterative* SDLC

Dengan Methodology *Iterative* pengerjaan proyek dapat dibagi menjadi sub proyek-sub proyek, Hal ini mendukung suatu kerja sama oleh pihak terkait pengembangan dan perancangan untuk melihat hasil sebelumnya dan didapatkan suatu umpan balik yang berharga dari seorang pengguna sistem. Pendekatan ini membangun implementasi parsial dari keseluruhan sistem dan kemudian secara perlahan menambah fungsionalitas (Setyoningrum,2016). Metode *Iterative* SDLC yang terdiri dari beberapa bagian yaitu :

1. *Planning*

   Langkah pertama dalam merancang sistem adalah mengamati suatu proses bisnis pada perusahaan, selanjutnya menentukan sistem yang diperlukan dan mempelajari suatu kelayakan pada sistem yang akan dikembangkan. Studi kelayakan sistem mencakup bidang teknis, ekonomi dan operasional.

2. *Analysis*

   Setelah lolos uji kelayakan dan mendapat persetujuan dari perusahaan, selanjutnya masuk ke tahapan analisis, yang melingkupi kebutuhan *fungsional* dan *non-fungsional*, pemodelan dengan bentuk *use case*, dan pemodelan struktural dalam bentuk diagram dan diagram aktivitas. Dengan menggunakan UML.

3. *Design*

   Tahap *Design* adalah dimulai dengan perancangan database, normalisasi data awal, kemudian membuat diagram kelas dan menjelaskan setiap struktur tabel, kemudian merancang antarmuka termasuk form dan laporan.

4. *Implementation* dan *Testing*

   Tahap terakhir adalah implementasi perancangan ke dalam bentuk aplikasi yang sudah dikerjakan, membuat buku petunjuk dan melakukan percobaan dengan pengguna untuk menghindari *bug* dan kesalahan dalam perancangan sistem (Rahayu & Makinto, 2017).

2.1.3 Pengertian *Framework*

Framework adalah kerangka kerja pada suatu pemrograman yang mencangkup library pada sebuah kode program, model perangkat lunak, API dan berbagai elemen lain yang dapat mempermudah proses pemrograman. (Hafizulhaq, 2021). Dengan adanya konsep kerangka kerja dapat memudahkan pengguna untuk merancang pekerjaan aplikasi yang dibangun.

Framework dibangun dengan menggunakan pendekatan struktural MVC, yaitu *Model*, *View*, dan *Controller*, memungkinkan pengembang untuk mengelompokkan fungsi seperti *input*, pemrosesan, dan *output* dari aplikasi yang dirancang, dengan adanya *framework* pada suatu program pengerjaan *code* pada program dalam membangun program akan lebih terstruktur dan tidak adanya *script* pengulang seperti biasanya dilakukan oleh program yang bersifat *native*.

2.1.4 Pengertian Node.JS

Node.JS adalah suatu perangkat lunak yang dirancang untuk melakukan pengembangan aplikasi berbasis pada web, dengan dirancang dalam sintaks pemrograman dengan bahasa JavaScript yang berjalan disisi *client*/browser, dari hal ini Node.JS akan melengkapi Bahasa pemrograman javascript sebagai bahasa yang berjalan di sisi *server*, seperti PHP, Ruby, dan Perl. Node.JS mendukung sistem operasi Windows, Mac OS X dan Linux tanpa mengubah kode program. Node.JS memiliki perpustakaan *server*

HTTP sendiri untuk menjalankan *server* web tanpa menggunakan program *server* web seperti *apache* (Lutfi, 2017). Dari hal ini penggunaan Node.JS menjadi suatu *platform* javascript yang sangat dapat diandalkan penggunaannya pada teknik *asynchronous*.

2.1.5 Pengertian Javascript

Javascript adalah bahasa pemrograman *scripting* dinamis yang dapat Anda gunakan untuk membuat interaksi yang biasa digunakan pada halaman HTML yang dapat dilakukan secara statis. Oleh karena itu, menambahkan pemrograman ke blok kode Javascript memudahkan pengelolaan data langsung dari sisi *client* atau browser pengguna (Sianipar, 2017).

Penggunaan Bahasa pemrograman javascript biasa digunakan pengembang dalam membuat aplikasi berbasis web yang berjalan pada sisi browser atau *client*. Namun, sejak ditemukannya Node.JS Bahasa pemrograman javacript dapat digunakan pada sisi server seperti PHP, C+, Ruby dan ASP, dari hal ini dengan adanya Bahasa pemrograman javascript dapat mendukung perancangan aplikasi yang berjalan pada sisi server dengan memanfaatkan Node.JS untuk mengembangkan dan merancang aplikasi (Equan,2019).

2.1.6 Pengertian Database

Database adalah kumpulan data atau informasi yang akan dilakukan penyimpanan secara sistematis dalam sistem komputer dan dapat dilakukan pemeriksaan dalam memperoleh informasi dari basis data. Perangkat lunak digunakan untuk mengelola dan memanggil *query* basis data yang dikenal dalam sistem basis data sebagai sistem manajemen basis data (DBMS) dan dapat dipelajari di bidang ilmu informasi (Setiawan, 2021). Dikarenakan peran database sangat penting dalam pengelolaan sistem informasi, dari hal ini terdapat banyak pilihan software pada *Database*

Management System ada beberapa contoh DBMS yang sering digunakan yaitu: Oracle, MySQL, MS SQL Server dan Database Desktop yaitu Paradox. Dalam menjalankan Perintah pada umumnya ditentukan oleh user, adapun Bahasa yang digunakan dibagi dua macam diantaranya sebagai berikut:

1. DDL (*Data Definition Language*)

   Menggambarkan desain pada basis data secara menyeluruh dan dapat digunakan dalam membuat tabel baru, dan mengubah tabel, hasil dari kumpulan *Data Definition Language* akan disimpan pada kamus data.

2. DML (*Data Manipulation Language*)

   DML digunakan dalam memanipulasi dan pengambilan suatu data pada sebuah basis data, seperti menambahkan data baru ke dalam database, mengubah data dalam database dan menghapus data dalam database (Syafnidawaty, 2020). Dari penerapan DML yang dijelaskan penulis dapat memanfaatkan pengelolaan database seperti perancangan untuk melakukan backup pada database dan restore pada database, hal ini sangat berguna dalam mengembangkan Data *manipulation Language* pada suatu fitur yang dirancang dan diteliti oleh penulis.

2.1.7 Pengertian JSON

JSON (*Javascript Object Notation*) adalah format untuk pertukaran data dan penyimpanan data. Json berasal dari bahasa pemrograman JavaScript, tetapi struktur ini mendukung dalam banyak bahasa, termasuk PHP, Ruby, Python, dan Java. Dikarenakan dalam pembacaan pada suatu komputer sangat ringan sehingga JSON memberikan kemudahan yang lebih baik untuk XML dan lebih sedikit pemformatan (Santoso, 2017). Implementasi JSON dalam pemrograman biasanya digunakan sebagai bentuk standar pertukaran data antar aplikasi. Berikut adalah beberapa implementasi JSON:

1. JSON sebagai format untuk pertukaran data klien dan *server* atau aplikasi seperti *Restful* API.
2. JSON sebagai suatu penyimpanan data seperti database mongodb.
3. JSON digunakan untuk menyimpan konfigurasi proyek seperti composer.Json di PHP dan package.Json.
4. JSON digunakan untuk menyimpan konfigurasi proyek di Node.JS (Muhardian, 2017).

2.1.8 Pengertian *Object Oriented Analysis Design* (OOAD)

Menurut Mathiassen, Diagram (OOAD) merupakan sebuah pendekatan yang digunakan untuk merancang maupun menganalisa suatu sistem dengan menggunakan pendekatan secara objek (Purwaningtias, 2018). Metode penggunaan pada OOAD yaitu UML Merupakan suatu *tools* yang digunakan dalam perancangan berorientasi objek berbasis UML, seperti *use case diagram*, *activity diagram*, dan *sequence diagram*. Di bawah ini adalah deskripsi dari setiap diagram UML dan notasi yang paling umum digunakan.:

1. *Use Case Diagram*

   Use case diagram adalah diagram yang menggambarkan hubungan antara aktor dan sistem. *Use case* diagram menjelaskan interaksi antara satu atau lebih aktor dan sistem yang dirancang. (Hutauruk, 2021).

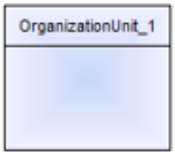
Tabel 2.1. Simbol Dalam *Use Case* Diagram

Simbol	Fungsi
	Menggambarkan seorang pengguna yang berinteraksi dengan sistem
 Notasi Use Case	Deskripsi urutan aksi yang ditampilkan sistem yang memberikan hasil bagi aktor
Notasi Interaction	Perilaku yang dilakukan aktor terhadap sistem untuk menyelesaikan suatu fungsi tertentu
<<Include>> Relationship	Menggambarkan suatu use case termasuk di dalam use case lain
<<Extend>> Relationship	Digunakan untuk menggambarkan satu use case membutuhkan use case lain jika kondisi atau syarat tertentu terpenuhi

2. *Activity Diagram*

Activity Diagram adalah Memodelkan sistem dan menggambarkan aktivitas sistem yang sedang berjalan, diagram aktivitas digunakan untuk menjelaskan aktivitas program tanpa pemrograman atau memberikan tampilan sistem yang sedang dikerjakan.(Novitasari, 2020).

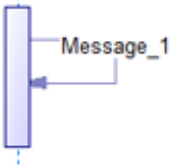
Tabel 2.2. Simbol Dalam *Activity Diagram*

Simbol	Fungsi
 Action State	Menggambarkan langkah-langkah dalam sebuah aktivitas
 Start State	Menunjukan aliran kerja dimulai
 End State	Menunjukan aliran kerja berakhir
 Decission	Menunjukkan keputusan yang perlu dibuat dalam aliran kerja
Synchronization	Menunjukan dua atau lebih aliran kerja berjalan secara bersama
 Swimlane	Menunjukan aktivitas yang dilakukan pada objek tertentu

3. *Sequence Diagram*

Sequence Diagram adalah diagram yang menunjukkan serangkaian pesan interaksi yang dikirim antar objek dan interaksi antar objek (Pratama, 2019). Dapat dilihat pada tabel di bawah ini :

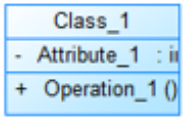
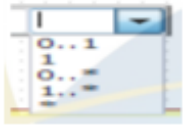
Tabel 2.3. Simbol Dalam *Sequence Diagram*

Simbol	Fungsi
 *Lifeline*	Menggambarkan entitas tunggal dalam sequence diagram. Entitas ini memiliki nama *stereotype* atau berupa *instance*
 Actor	Entitas yang berada di luar sistem dan dapat berupa manusia, atau perangkat sistem lainnya
 *Self Message*	Menunjukkan bahwa suatu objek hendak memanggil dirinya sendiri
 *Boundary*	Menggambarkan interaksi atau antarmuka yang dilakukan oleh aktor.
 *Synchronization*	Digunakan untuk memanggil operasi atau metode yang dimiliki oleh suatu objek dengan *message* yang mengharuskan untuk menyelesaikan proses baru kemudian memanggil proses berikutnya.

4. *Class Diagram*

Class Diagram merupakan kumpulan dari suatu *class* dan hubungan struktural, dengan adanya *class* diagram menjadi penghubung antara alur jalannya database pada sebuah sistem, dapat dilihat pada tabel di bawah ini:

Tabel 2.4. Simbol dalam *class diagram*

Simbol	Fungsi
	Terdiri dari nama kelas, atribut yang digunakan untuk memberikan deskripsi properti dari sebuah *class*, dan *operation* untuk menjelaskan sesuatu yang dapat dilakukan pada *class* tersebut
	*Cardinality* dan *multiplicity* untuk menunjukkan jenis hubungan dari satu *class* dengan *class* lain
 Association	Garis lurus yang digunakan untuk menghubungkan 2 (Dua) *class* berbeda

2.1.9 Pengertian *Object Oriented Programming*

Object Oriented Programming (OOP) adalah sebuah strategi perancangan pada perangkat lunak yang dalam mengatur dan menyusun perangkat lunak sebagai kumpulan objek yang berisikan suatu data atau operasi yang diberlakukan. *Object Oriented Programing* (OOP) merupakan pandangan atau suatu teknik dalam metode pemrograman berorientasi kepada objek (Rais,2019). Dengan adanya suatu metode pemrograman OOP dapat dengan mudah menangani kesulitan dalam melakukan pemrograman, hal ini dapat mengatasi permasalahan yang ada dalam merancang program.

2.2 Tinjauan Studi

Tinjauan studi dilakukan penulis untuk mendukung penulisan tugas akhir, Penelitian yang dilakukan sebelumnya akan sangat bermanfaat bagi penulis untuk menjadi pedoman dalam mendapatkan teori-teori hasil penelitian yang sesuai untuk pembuatan *framework* pada dalam merancang fitur *backup* dan *restore* dalam mempermudah proses database. Berikut ini beberapa referensi untuk mendukung penelitian ini sebagai berikut:

1. Jurnal penelitian yang dilakukan oleh Rahmanda, Pramukantoro dan Yahya di tahun 2017 dengan judul "PERANCANGAN DAN IMPLEMENTASI KELAS VIRTUAL FILKOM UNIVERSITAS BRAWIJAYA DENGAN MEMANFAATKAN TEKNOLOGI WEB RTC (WEB *REAL-TIME COMMUNICATION*)". Jurnal penelitian ini membahas mengenai perancangan dan implementasi suatu sistem *e-learning* atau dikenal dengan kelas virtual yang diterapkan pada FILKOM UB dengan menggunakan Web RTC, Dari hal ini dalam memberikan suatu solusi kepada pelajar untuk melakukan pembelajaran jarak jauh di suatu tempat dan kapan saja melalui web secara *real-time* dalam melakukan suatu proses eksekusi. Komunikasi secara *real-time* akan ditangani oleh suatu tindakan pengguna atau disebut dengan *event-driven* hal ini akan mempermudah proses tanpa menggunakan banyak memori, hal ini akan mempermudah pengembang aplikasi agar dapat memanfaatkan waktu dalam membangun fungsionalitas dari suatu aplikasi, pada hal ini penggunaan *server* menjadi sangat ringan karena penggunaan bahasa pemrograman yaitu Javascript dapat menghemat kapasitas bandwidth dan beban *server*, dari hal ini node.JS merupakan *runtime environment* yang berguna untuk merancang suatu aplikasi yang bersifat *server-side* Arsitektur yang berbasis pada *event-driven* yang dikombinasikan dengan suatu arsitektur *asyncronous* atau dikenal dengan *non-blocking* I/O, dari hal ini sangat tepat dalam membangun aplikasi dengan jumlah pengguna yang sangat banyak (Rahmanda, Pramuktoro & Yahya, 2018). Berdasarkan penjelasan tersebut dapat dijadikan bahan dan

referensi pendukung penulis untuk mengangkat pengembangan *framework* Aurora.JS berbasis Node.js, dengan mengembangkan suatu fitur *backup* dan *restore* dalam pembuatan proses database, sehingga kekurangan yang terjadi di *framework* Aurora.JS dapat disempurnakan dan interaksi database dapat lebih mudah.

2. Jurnal penelitian yang dilakukan oleh Diovianto Putra Rakhmadani, Yudithia Dian Putra dan Ine Luna Dianti di tahun 2020 dengan judul "TRANSFORMASI DIGITAL PADA BISNIS UMKM DENGAN PENERAPAN DBMS". Jurnal penelitian ini membahas mengenai penerapan DBMS pada bisnis UMKM yang mengaruskan para pelaku usaha untuk menjalankan bisnis dengan melakukan beberapa modifikasi pada usahanya di masa revolusi industri 4.0, perubahan yang dilakukan yaitu melakukan transformasi ke bidang digital dalam memanfaatkan teknologi informasi untuk membantu pekerjaan bisnis dalam segi kompetitif maupun maupun segi kompetitor bisnis usaha sejenis, hal ini untuk membantu usaha yang dilakukan untuk melakukan persaingan secara sehat. Pada jurnal ini menggunakan pendekatan *Database Management System* (DBMS) merupakan sistem perangkat lunak yang memungkinkan pengguna dalam menjelaskan, membuat, mengelola, dan mengontrol akses ke database, dengan penggunaan DBMS dapat mendeskripsikan basis data dengan data *Definition Language* (DDL) untuk menentukan suatu tipe maupun struktur batasan data yang akan disimpan dalam database nantinya, pada pengguna dapat melakukan suatu perubahan, melakukan penambahan, melakukan penghapusan dan pengambilan data dari database yang telah dibuat penggunaan pada *Data Manipulation Language* (DML). Tujuan penggunaan DBMS dalam penggunaan suatu jaringan pada komputer ialah untuk menghadapi masalah saat melakukan pemprosesan data dalam jumlah yang sangat besar. Dari hal ini menimbulkan keraguan mengenai kegagalan pada operasional. Penggunaan DBMS menjadi suatu perantara antara pengguna dan database, tentu hal ini pengguna dapat melakukan interaksi dengan DBMS dan pada sisi pengguna diharuskan

dapat menggunakan Bahasa pada database yang telah disesuaikan dan ditentukan. Adanya suatu perancangan pada DBMS merupakan suatu langkah transformasi di bidang digital pada UMKM, dalam penyimpanan data ke dalam DBMS yang dijadikan sebagai suatu pondasi untuk menjalar ke arah digital dengan memanfaatkan data yang sudah terkomputerisasi (Diovanto, 2020). Hasil dari penelitian tersebut dengan menerapkan metodologi DBMS dapat dijadikan sebagai pertimbangan penulis untuk memastikan bahwa modul database pada *framewor*k Aurora.JS dapat digunakan untuk membantu menganalisa perancangan fitur *backup* dan *restore* dapat mempermudah proses pekerjaan programmer dalam memanfaatkan metode DBMS untuk merancang data yang terdapat di database.

3. Jurnal penelitian yang dilakukan oleh Endang Retnoningsih, Jafar Shadiq, Dony Oscar di tahun 2017 dengan judul "PEMBELAJARAN PEMROGRAMAN BERORIENTASI OBJEK (*OBJECT ORIENTED PROGRAMMING*) BERBASIS PROJECT *BASED LEARNING*". Jurnal penelitian ini membahas pembelajaran dalam mempelajari pemrograman yang dibangun dapat sesuai dengan yang diharapkan, maka diperlukan suatu pembelajaran untuk dapat berpikir secara algoritmik. Pemrograman berorientasi objek atau pemrograman berorientasi (OOP) yang sangat berguna saat menulis struktur program. Hal ini dikarenakan pemrograman OOP sudah menggunakan konsep modularitas objek dan kelas. Pemrograman berorientasi objek, atau pemrograman berorientasi (OOP), adalah pendekatan pemrograman yang menggunakan objek dan kelas. OOP berguna saat melakukan penulisan dalam membangun struktur program. Keuntungan membuat pemrograman berorientasi objek atau dikenal dengan *object oriented programming* (OOP) adalah penggunaan kembali kode yang dibuat disebut *reusability*, programmer dapat membuat metode baru tanpa menulis kode dari Anda dapat membuat atau memodifikasi metode yang sudah ada. Scratch diperlukan disebut *extensibility*, kode yang ditulis lebih mudah dikelola ketika aplikasi ditulis dalam skala besar dan dapat

menyebabkan kesalahan selama pengembangan. Hal ini dapat diatasi dalam OOP karena pemrograman OOP sudah menggunakan konsep modularitas disebut dengan *maintainanility* (Retnoningsih et al., 2017). Berdasarkan hasil dari penjelasan tersebut penerapan pemrograman berorientasi OOP dapat mempermudah proses pekerjaan code dapat dikelola oleh perancang apabila aplikasi yang dibuat berskala besar, hal ini sangat baik dilakukan untuk melakukan penambahan modul pada *framework* Aurora.JS.

4. Jurnal penelitian yang dilakukan oleh Ajeng Ananda, Ary Budi Warsito, Dian Triyanjaya di tahun 2017 dengan judul "PENERAPAN DATA JSON UNTUK MENDUKUNG PENGEMBANGAN APLIKASI PADA PERGURUAN TINGGI DENGAN TEKNIK *RESTFULL* DAN WEB *SERVICE*" Jurnal penelitian ini membahas data dalam suatu aplikasi dibutuhkan suatu database yang dapat menyimpan suatu data secara besar dengan mendistribusikan data yang tersedia pada *server* kepada *client*, banyaknya aplikasi yang membutuhkan suatu kapasitas *bandwith* yang sangat besar dalam menyajikan pengambilan data sebagai sistem informasi. Hal ini sangat berguna untuk berbagai untuk berbagai tujuan. *Web service* memainkan peran penting dalam hal ini, *client* yang perlu mengakses database melalui jalur lain yang akan diteruskan ke layanan web selama proses akuisisi data, memfasilitasi pekerjaan database yang perlu mengulangi permintaan dari *client*. pada dasarnya Arsitektur *rest* beroperasi melalui *Hypertext Transfer Protocol*. Dalam membaca halaman tertentu yang berisi *file* JSON. Saat membuat *file* JSON yang mendukung pengembangan aplikasi, solusi masalahnya adalah dengan menerapkan metode *Restfull* API, karena data yang diambil melalui database benar-benar terpisah dari database. Data yang terkumpul diubah menjadi *file* yang dapat diunduh atau di download pada siswa tanpa harus terhubung langsung ke dalam database. Mahasiswa tidak perlu lagi melakukan akses secara manual, menawarkan banyak keuntungan, seperti mempermudah proses pendataan mahasiswa, memberikan kemudahan akses *web service* yang

sudah tersedia, Anda dapat melakukan ndownload semua data tentang universitas. Kapan saja dan dimana saja. *File* hasil konversi JSON tidak terhubung ke database, jadi jika data rusak atau sistem yang diretas oleh pihak yang tidak bertanggung jawab, data asli akan disimpan dengan aman di database tanpa gangguan sedikit pun. (Warsito et al., 2017). Dari hasil penelitian tersebut dalam menerapkan fitur *backup* dan *restore* dalam mengerjakan proses database dengan mempelajari dan memanfaatkan file JSON dalam menyimpan dan mengambil data sehingga meringankan kerja database, karena data disimpan dalam *file* JSON.

5. Jurnal Penelitian yang dilakukan oleh Ilham Arnomo di tahun 2019 dengan judul "SIMULASI *BACKUP* DAN *RESTORE* DATABASE *REPOSITORY* INSTITUSI BERBASIS APLIKASI DSPACE". Pada jurnal penelitian ini membahas tentang analisa suatu kinerja *backup* dan *restore* pada database dengan fitur penyimpanan koleksi repositori institusional milik aplikasi DSpace. Menggunakan metodologi pendekatan eksperimental dan prosedur penelitian: Kumpulkan data dan informasi tentang menginstal aplikasi DSpace dari referensi situs web. Unduh aplikasi DSpace dan perangkat lunak pendukungnya dari situs referensi. Instal aplikasi DSpace dan perangkat lunak pendukung di komputer pribadi Anda. Uji *backup* dan *restore* database di aplikasi DSpace. Analisis integritas data yang disimpan. Tidak ada perubahan data yang dilakukan setelah data dikembalikan. Penelitian tentang pencadangan dan pemulihan data pernah dilakukan oleh (Handrini et al., 2018) metode *full backup restore* dapat mendukung strategi pemulihan bencana. Metode pemulihan penuh mencadangkan dan memulihkan semua *file*, sehingga memudahkan pebisnis untuk menemukan semua data mereka. Hasil pengujian integritas data, pencadangan, dan pemulihan dengan perangkat lunak Bacula menggunakan *hash* MD5 dan tanda tangan digital terus dipertahankan demi keamanan dan keaslian data (Arnomo, 2019).
6. Penelitian tugas akhir yang dilakukan oleh Calvin Johannes pada tahun

2019 dengan judul "PENGEMBANGAN MODUL DATABASE PADA FRAMEWORK AURORA.JS UNTUK MEMUDAHKAN PENGEMBANGAN APLIKASI BERBASIS NODE.JS" jurnal penelitian ini membahas tentang pengembangan database yang dirancang pada *Framework* Aurora.JS berbasis Node.JS. Penggunaan Node.js masih sedikit, hal ini disebabkan belum banyaknya *framework* yang mendukung Node.JS, sehingga terjadinya ide pada peneliti dalam membangun *framework* Aurora.JS. Pengembangan modul database pada *framework* Aurora.JS dengan mendukung pada database MySQL. Tujuan dari modul database yang dikembangkan yaitu untuk menambah pilihan *framework* Node.JS untuk memiliki fitur database, dan menyediakan fungsi untuk menggantikan penggunaan syntax SQL dalam modul database (Johannes, 2019).

7. Penelitian tugas akhir yang dilakukan oleh Thomas Dermawan pada tahun 2019 dengan judul "PENGEMBANGAN MODUL CREATE DAN UPDATE PADA FRAMEWORK AURORA.JS UNTUK MEMUDAHKAN PENGEMBANGAN APLIKASI BERBASIS NODE.JS". Penelitian tugas akhir ini membahas Pengembangan fitur untuk manipulasi data yaitu *create* dan *update*. Oleh karena itu, pengembang membutuhkan teknik pengembangan yang sederhana dan dapat membuat aplikasi yang memenuhi kebutuhan aktivitas masyarakat. Teknologi pengembangan aplikasi telah ditingkatkan secara signifikan dalam hal kemudahan penggunaan, seperti bentuk kerangka kerja. Di sini, Node.JS merupakan platform yang memudahkan pengembang untuk menjalankan proses I/O menggunakan sisi *server* yang ringan. Dari jurnal penelitian ini pengembang melakukan implementasi perancangan fitur manipulasi data *create* dan *update* pada *framework* Aurora.JS yang dirancang melalui platform Node.JS. Dengan adanya fitur ini mempermudah pengembang aplikasi dalam penulisan *syntax* serta modifikasi kode untuk modul *create* dan *update* data pada *framework* Aurora.JS (Dermawan, 2019).
8. Penelitian tugas akhir yang dilakukan oleh Muhammad Russel pada

tahun 2019 dengan judul "PENGEMBANGAN MODUL MANIPULASI BASIS DATA (*READ* DAN *DELETE*) PADA *FRAMEWORK* AURORA.JS UNTUK MYSQL DATABASE" jurnal penelitian ini membahas tentang Pengembangan fitur *read* dan *delete* menggunakan Node.JS yang merupakan suatu perangkat lunak yang didesain untuk melakukan pengembangan sistem pada aplikasi yang berbasis pada web dan ditulis dalam sintaks Bahasa suatu pemrograman yaitu javascript yang memiliki konsep *non-blocking* I/O. Konsep ini dapat memproses dua permintaan secara bersamaan sehingga dapat mempercepat pelayanan, hal ini yang menjadi dasar pada peneliti tersebut untuk mengembangkan fitur *read* dan *delete* berbasis Node.JS. dengan adanya fitur ini pada penelitian ini penulis melakukan pengembangan *framework* yang diberi nama Aurora.JS yang memiliki tujuan yaitu untuk membantu developer dalam pembuatan modul manipulasi basis data *read* dan *delete* pada sistem yang dirancang peneliti menggunakan web service berbasis Node.JS yaitu *framework* Aurora.JS (Russel, 2019).

UNIVERSITAS PEMBANGUNAN JAYA