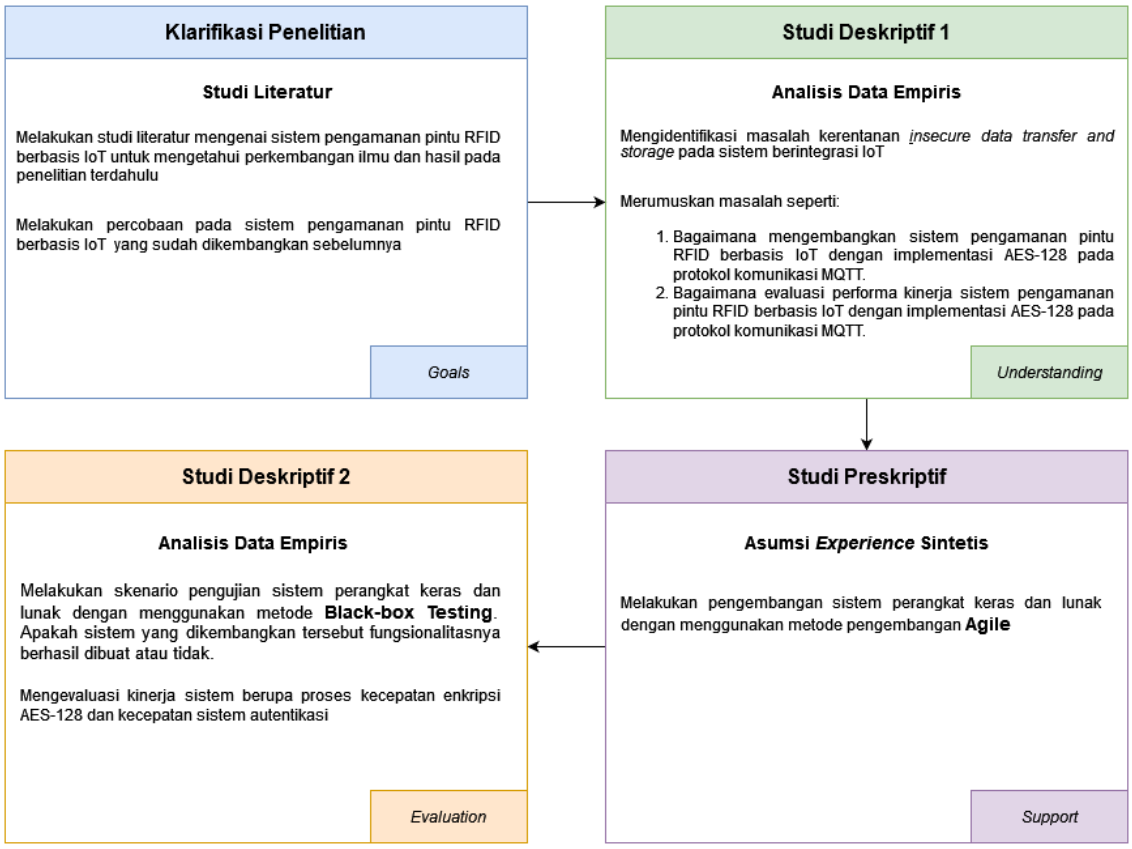
BAB III
METODE PENELITIAN

3.1 Desain Penelitian

Pada penelitian ini, penulis menggunakan desain metode penelitian DRM (*Development Research Methodology*). DRM adalah sebuah pendekatan penelitian yang lebih menekankan pada pengembangan atau perbaikan produk dari teknologi-teknologi pada penelitian sebelumnya yang sudah ada (Blessing & Chakrabarti, 2009). Tujuan utama DRM ini adalah mengidentifikasi masalah, merancang dan mengembangkan solusi, serta menguji implementasi solusi tersebut (Blessing & Chakrabarti, 2009). Hal tersebut selaras dengan tujuan utama dari penelitian ini, yaitu mengembangkan sebuah sistem yang sudah ada dengan merancang dan mengimplementasikan salah satu solusi teknologi kebaruan. Adapun ilustrasi skema yang dilakukan pada penelitian ini berdasarkan prinsip desain DRM dapat dilihat pada Gambar 3.1



Gambar 3.1 Desain Penelitian menggunakan DRM

3.1.1 Klarifikasi Penelitian

Pada tahapan pertama dalam desain metode penelitian DRM ini, Klarifikasi Penelitian terlebih dahulu dilakukan dengan *main outcomes*-nya yaitu *Goals*. *Goals* dimaksudkan dengan tujuan utama melakukan fokus studi literatur sebagai awal dalam mengetahui perkembangan ilmu terkini dari penelitian-penelitian sebelumnya. Dengan didapat beberapa teori berdasarkan buku, jurnal artikel dan sumber ilmiah lainnya, penulis dapat merangkai penelitian dan mengkaji kelebihan dan kekurangan dari penelitian-penelitian sebelumnya. Tahapan ini yang menjadikan titik acuan penulis dalam menemukan sebuah ide untuk mengembangkan sistem pengamanan pintu RFID berbasis IoT dengan mengimplementasikan ide kebaruan yaitu algorima enkripsi AES-128. Penulis melakukan studi literatur mengenai topik penelitian seperti sistem pengamanan pintu RFID berbasis IoT, *cyber security* pada sistem berintegrasi IoT, dan algoritma enkripsi AES-128.

3.1.2 Studi Deskriptif I

Selaras dengan *main outcomes*-nya yaitu *Understanding*, tahapan kedua dalam desain metode penelitian DRM ini merupakan proses analisis data empiris yang memfokuskan penulis untuk memperluas pemahaman dan mendalami lebih dalam kajian topik penelitian terdahulu agar dapat mengidentifikasi dan merumuskan sebuah masalah dalam penelitian. Dengan ini penelitian dapat menjawab permasalahan-permasalahan yang ada sesuai dengan tujuan dan relevansi yang ingin dicapai, serta penulis dapat mempersiapkan kebutuhan dalam mengembangkan sebuah kebaruan sistem.

3.1.3 Studi Preskriptif

Kemudian selanjutnya masuk ke dalam tahapan proses Studi Preskriptif. Tahapan ini dengan *main outcomes*-nya yaitu *Support* yang didalamnya penulis melakukan sebuah dukungan dari pemahaman yang didapat untuk mengimplementasikan kebaruan pada pengembangan sistem berupa perangkat keras atau lunak. Adapun penulis melakukan pengembangan sistem pengamanan pintu RFID berbasis IoT dengan algoritma enkripsi AES-128, serta metode pengembangan sistem di dalamnya yang menggunakan sebuah metode bernama *Agile*. Penulis dapat melakukan asumsi *experience* sintetis

yang melakukan sebuah pengembangan sistem tersebut karena telah berangkat dari proses sebelumnya, yaitu Klarifikasi Penelitian dan Studi Deskriptif I.

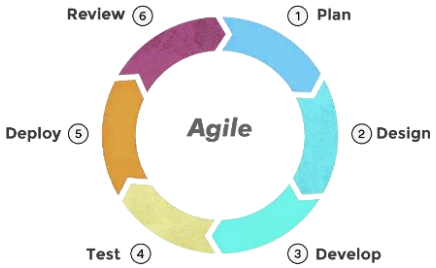
3.1.4 Studi Deskriptif II

Lalu terakhir, yaitu tahapan proses Studi Deskriptif II dalam melakukan analisis data empiris yang sama halnya dengan Studi Deskriptif I akan tetapi perbedaanya yang pada *main outcomes*-nya yaitu *Evaluation*. Pada tahapan ini penulis melakukan sebuah evaluasi berupa pengujian dari proses setelah melakukan sebuah pengembangan sistem, apakah hasil tersebut sesuai dengan tujuan dan relevansi yang diharapkan dalam menjawab rumusan masalah atau tidak. Adapun penulis dalam melakukan pengujian sistem tersebut dengan melakukan sebuah pendekatan metode bernama *Black-box Testing.*

3.2 Metode Pengembangan Sistem

Penulis menggunakan metode pengembangan sistem bernama *Agile* dalam Studi Preskriptifnya. Dibandingkan dengan pendekatan tradisional yang berorientasi pada tahapan pengembangan yang linear seperti *Waterfall*, *Agile* menekankan pada fleksibilitas, responsif terhadap perubahan, dan iterasi berulang (Binar Academy, 2023). Fleksibilitas *Agile* memungkinkan penulis untuk menyesuaikan rencana dan prioritas berdasarkan hasil dan perubahan yang terjadi selama penelitian. Selain itu, metode ini memungkinkan penulis untuk mendapatkan umpan balik yang cepat dari setiap iterasi.

Dalam konteks perangkat keras dan perangkat lunak, *Agile* memungkinkan penulis untuk lebih mudah menyesuaikan perubahan kebutuhan, *troubleshooting code*, dan tuntutan desain yang mungkin muncul selama proses pengembangan. Meskipun *Agile* acapkali digunakan oleh proyek manajemen tim, akan tetapi konsep-konsep seperti *backlog* tugas ketika ada perubahan yang menyesuaikan dapat digunakan oleh proyek individu seperti pada penelitian yang dilakukan oleh penulis (Binar Academy, 2023). Pada metode pengembangan sistem *Agile*, terdapat beberapa proses iterasi seperti *plan*, *design*, *develop*, dan lain sebagainya. Adapun ilustrasi dan tahapan iterasi secara lengkapnya yang digunakan penulis dalam mengembangkan sistem pengamanan pintu RFID berbasis IoT dengan algoritma enkripsi AES-128 pada Studi Preskriptif dapat dilihat pada Gambar 3.2.



Gambar 3.2 Metode Pengembangan Sistem *Agile*

3.2.1 *Plan*

Pada proses iterasi pertama *Agile* atau perencanaan *sprint*, yaitu *plan*, penulis melakukan persiapan dan menetapkan kebutuhan teknologi alat dan bahan yang dipakai dalam pengembangan sistem pengamanan pintu RFID berbasis IoT dengan algoritma enkripsi AES-128. Proses ini fleksibel dan dapat disesuaikan seiring berjalannya waktu untuk memenuhi kebutuhan dan tantangan yang muncul selama pengembangan sistem. Adapun *plan* kebutuhan teknologi yang sesuai dengan fungsionalitasnya dalam menunjang pengembangan sistem tersebut tersajikan pada Tabel 3.1 dan Tabel 3.2.

Tabel 3.1
Perencanaan Teknologi Perangkat Lunak yang Digunakan

Nama Teknologi	**Fungsi**
Arduino IDE dan VS Code	Sebagai teks editor C++ & Laravel untuk *develop* perangkat lunak dalam sistem tertanam perangkat keras, dengan didukung oleh beberapa *library* seperti enkripsi AES-128.
Adafruit IO	Sebagai MQTT broker yang menaungi *subscribe* dan *publish* data dalam protokol komunikasi IoT.
Laravel	Sebagai *framework* dalam proses pembuatan *web* dashboard
LAMP (Linux, Apache, MySQL, dan PHP)	Sebagai *stack* teknologi sistem operasi Linux yang berperan sebagai *database* dan web *server* dalam sistem pengamanan pintu RFID berbasis IoT.
DigitalOcean	Sebagai *Infrastructure as a Service* dalam pengembangan.
DrawIO, Fritzing, dan EasyEDA	Sebagai *software* untuk mendesain PCB, arsitektur, diagram, dan flowchart dalam pengembangan sistem.
GitLab	Sebagai repositori kode untuk automasi perubahan fitur.
FreeRTOS	Sebagai *kernel* sistem operasi pada *microcontroller*.

Tabel 3.2
Perencanaan Komponen Perangkat Keras yang Digunakan

Nama Komponen	Fungsi
Microcontroller ESP32	Sebagai otak pada sistem tertanam perangkat keras dalam menangani proses input dan *output* pada sistem pengamanan pintu RFID berbasis IoT.
RFID MRFC 522 HF	Sebagai inputan *contactless card* (eKTP) untuk menerima UID pengguna dalam autentikasi pembukaan kunci pintu.
Relay Module 5V	Sebagai sakelar dalam membuka atau menutup kunci pintu
Doorlock Solenoid 12V	Sebagai aktuator komponen kunci pintu yang mengubah energi listrik menjadi gerakan mekanis.
Sensor *Fingerprint* AS608	Sebagai alternatif inputan RFID yang memanfaatkan biometrik sidik jari pengguna.
Keypad PIN 4x4 *Membran*	Sebagai inputan untuk proses pendaftaran (mode *enrollment*) pengguna.
LCD I2C *LiquidCrystal* 16x2	Sebagai luaran dalam menampilkan pesan berupa visual teks yang disajikan kepada pengguna.
LED RGB KY-016	Sebagai luaran yang menampilkan visual warna sukses dan eror.
Mini Buzzer Alarm	Sebagai luaran yang menghasilkan suara peringatan sukses dan eror.
Button Module KY-004	Sebagai inputan untuk memilih mode *log history* atau *enrollment*, serta sebagai tombol untuk mengunci pintu.
Kabel *Jumper* F-M, M-M, dan F-F	Sebagai penghubung antar komponen perangkat keras sementara.
Catu Daya 5V dan 12V	Sebagai sumber daya dalam mengaktifkan komponen perangkat keras.

3.2.2 *Design*

Kemudian masuk ke dalam tahapan proses *design*, pada tahapan ini penulis melakukan rancangan dan desain sebelum dilakukannya proses *develop.* Tahapan ini bisa saja berubah-ubah dan dapat menyesuaikan dengan

iterasi sebelumnya yaitu *plan.* Penulis melakukan *design* pada sistem bagian perangkat keras dengan cara membuat *block* diagram, *wiring* diagram, arsitektur diagram, dan desain PCB (*Printed Circuit Board*). Sedangkan untuk *design* pada sistem bagian perangkat lunak, penulis membuat skenario protokol komunikasi MQTT pada sistem, flowchart sistem, *sequence* diagram, *activity* diagram, *use case* diagram, dan ERD (*Entity Relationship Diagram*). Adapun hasil temuan dan pembahasannya dalam perancangan desain pengembangan sistem dapat dilihat pada BAB IV.

3.2.3 *Develop*

Setelah itu masuk ke dalam tahapan proses *develop,* pada tahapan ini penulis melakukan serangkaian penggabungan teknologi alat dan bahan yang telah dipersiapkan dan didesain sebelumnya. Proses ini memakan waktu yang cukup lama ketimbang tahapan proses *Agile* lainnya. Untuk *develop* sistem pada bagian perangkat keras, penulis melakukan kegiatan konfigurasi *wiring* komponen *microcontroller* dengan penyesuaian *module,* sensor, aktuator, dan bentuk *casing*-nya, penulis pun melakukan pembuatan prototipe untuk simulasi miniatur pintunya sebelum akan diproduksi secara massal. Sedangkan untuk *develop* sistem pada bagian perangkat lunaknya, penulis melakukan kegiatan *ngoding* pada teks editor secara lokal dan konfigurasi IaaS untuk proses tahapan *deploy*. Adapun hasil temuan dan pembahasannya dalam pengembangan sistem dapat dilihat pada BAB IV.

3.2.4 *Test*

Lalu pada tahapan proses *test,* penulis melakukan pengujian sistem dengan menggunakan metode *Black-box Testing*. Tahapan ini merupakan proses yang terintegrasi secara menyeluruh sepanjang siklus pengembangan. Penulis melakukan pengujian secara terus-menerus selama setiap iterasi untuk memastikan bahwa setiap fungsionalitas yang dihasilkan memenuhi standar kualitas yang ditetapkan. Pendekatan ini memungkinkan deteksi dini terhadap *bug error* atau masalah lainnya, sehingga meminimalkan risiko penumpukan masalah di tahap akhir pengembangan.

3.2.5 *Deploy*

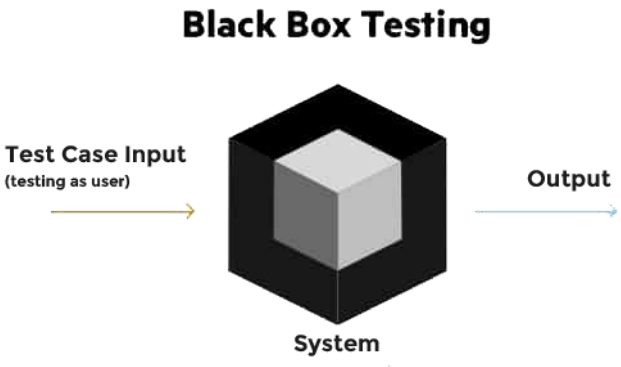
Proses *deploy* dalam metodologi *Agile* bertujuan untuk menyampaikan hasil dari pengembangan ke lingkungan produksi atau pengguna dengan cepat dan efisien. Penulis melakukan *deployment* secara berkala dalam IaaS DigitalOcean, penulis menggunakan GitLab dalam otomatisasi proses *deployment* yang acapkali diterapkan untuk mempercepat dan mengurangi risiko kesalahan dalam penyebaran perangkat lunak. Selain itu, pendekatan "*continuous deployment*" memungkinkan penulis untuk secara otomatis menerapkan perubahan langsung ke produksi *database* dan web server setelah lulus serangkaian pengujian yang ketat pada *development* secara lokal.

3.2.6 *Review*

Terakhir, proses ini untuk merefleksikan pengalaman pengguna atau si pembuat selama iterasi yang baru saja berakhir. Proses ini melibatkan evaluasi terhadap apa yang perlu ditingkatkan, dan bagaimana proses pengembangan dapat ditingkatkan ke depannya dengan kebaruan teori yang ada. Pendekatan ini memungkinkan penulis untuk terus mendapatkan masukan (*feedback)* dari pengembangan sistem yang telah dibuat, dan dapat menjadi acuan *backlog* ke iterasi *plan* berikutnya.

3.3 Metode Pengujian Sistem

Penulis pada metode pengujian sistem dalam Studi Deskriptif II-nya menggunakan pendekatan bernama *Black-box Testing* untuk mengidentifikasi dan memvalidasi fungsionalitas sistem secara keseluruhan tanpa perlu mengetahui rincian internal implementasinya. Pendekatan ini memungkinkan penulis sang pembuat pengembangan sistem untuk menguji sistem dari perspektif pengguna, fokus pada input dan *output* yang diharapkan, serta mengevaluasi respons sistem terhadap berbagai kondisi dan skenario (Wicaksono, 2021). Untuk ilustrasi gambaran dari proses cara kerja metode pengujian sistem *Black-box Testing* dapat dilihat pada Gambar 3.3.



Gambar 3.3 Ilustrasi Cara Kerja *Black-box Testing*

Dengan menggunakan teknik *Black-box Testing*, peneliti dapat menilai keefektifan sistem berdasarkan berfungsi atau tidaknya fitur dan berapa lamanya *delay* waktu yang didapatkan dalam memenuhi kebutuhan pengguna tanpa bergantung pada pengetahuan internal tentang struktur atau logika internal sistem (Wicaksono, 2021). Meskipun penulis sebagai pengembang sistem memiliki pengetahuan mendalam tentang struktur dan logika internal sistem yang dikembangkan, pengujian *Black-box* tetap penting karena memberikan sudut pandang dari perspektif pengguna. Dengan melakukan *Black-box Testing*, penulis dapat mengidentifikasi masalah atau kesenjangan dalam fungsionalitas sistem yang tidak penulis sadari selama proses pengembangan (Wicaksono, 2021). Hal ini memungkinkan penulis untuk membuat perbaikan yang diperlukan sebelum sistem dirilis ke pengguna akhir, sehingga meningkatkan kualitas dan keandalan sistem yang dihasilkan. Adapun hasil temuan dan pembahasannya dalam pengujian sistem dapat dilihat pada BAB IV.