

BAB III

METODE PENELITIAN

3.1 Metode Penelitian

Metode yang diterapkan dalam penelitian ini adalah metode *Design and Development* (D&D). D&D adalah pendekatan yang dirancang untuk melakukan studi sistematis mengenai proses desain, pengembangan, dan evaluasi. Menurut Ellis dan Levy (2010), "*Design and Development* merupakan investigasi disipliner dalam konteks pengembangan produk atau program dilakukan dengan tujuan untuk meningkatkan kualitas produk yang sedang dikembangkan maupun kompetensi pengembangnya. Metode D&D dipilih karena pendekatan ini menawarkan kerangka kerja yang terstruktur untuk merancang, mengembangkan, dan mengevaluasi solusi atau produk yang dihasilkan, seperti yang ditunjukkan dalam Gambar 3.1.



Gambar 3.1 Alur Metode Penelitian *Design and Development*

Berdasarkan gambar 3.1, tahapan ini diawali dengan analisis, di mana kebutuhan dan permasalahan diidentifikasi secara mendalam. Selanjutnya, pada fase desain, konsep dan solusi visual dikembangkan berdasarkan hasil analisis. Setelah itu, pengembangan adalah tahap di mana desain diimplementasikan menjadi produk nyata. Produk yang sudah jadi kemudian diuji pada tahap pengujian untuk memastikan fungsionalitas dan kualitasnya. Hasil dari pengujian tersebut dievaluasi pada tahap evaluasi, yang mungkin mengarah pada revisi atau perbaikan. Terakhir, seluruh proses dan hasil didokumentasikan dalam tahapan pelaporan.

Metode ini tidak hanya berfokus pada pembuatan produk, tetapi juga bertujuan untuk memperluas pemahaman dan pengetahuan mengenai produk yang dikembangkan melalui proses yang terstruktur dan sistematis. Metode *Design and Development* (D&D) dipilih dalam penelitian ini karena bertujuan untuk mengembangkan solusi teknologi dengan mengintegrasikan sistem validasi kode resi dan *monitoring* akses berbasis *web* ke dalam sebuah *smartbox* untuk

pengiriman paket. Kombinasi dari kedua komponen ini dirancang untuk meningkatkan efisiensi, keamanan, dan pemantauan proses penerimaan paket. Proses pengembangan sistem dilakukan melalui beberapa tahap, yaitu identifikasi kebutuhan, perancangan sistem, implementasi dan integrasi perangkat keras dan perangkat lunak, serta kegiatan evaluasi dan pengujian untuk memastikan fungsionalitas dan keamanan sistem.

3.2 Analisis Kebutuhan Sistem

Tahap analisis bertujuan untuk mengidentifikasi kebutuhan dan permasalahan yang melandasi pengembangan sistem *smartbox* otomatis. Masalah utama yang dihadapi adalah kurangnya sistem keamanan yang mampu memastikan proses penyimpanan paket secara aman, terutama dalam memastikan paket benar-benar ditempatkan di dalam *box*. Penelitian ini juga mengidentifikasi kebutuhan utama pengguna, yaitu adanya sistem penguncian otomatis berbasis kode resi. Pada tahap ini, dilakukan analisis terhadap spesifikasi teknologi yang akan digunakan, seperti sistem penguncian dengan verifikasi resi, dan kamera untuk monitoring barang kiriman. Selain itu, dilakukan evaluasi awal terhadap kebutuhan sistem *backend* dan *frontend* untuk mendukung konektivitas IoT dan antarmuka pengguna.

Tahap ini mencakup studi literatur dan pengumpulan data berdasarkan potensi serta permasalahan yang ditemukan di lapangan. Observasi langsung dilakukan untuk mengidentifikasi kendala dalam proses penerimaan paket, terutama ketika pemilik rumah tidak berada di tempat. Proses ini bertujuan untuk menganalisis permasalahan dan merumuskan solusi yang potensial. Analisis mencakup kebutuhan akan peningkatan keamanan serta kemampuan untuk memantau paket yang diterima secara efektif. Hasil dari analisis ini diharapkan menjadi dasar dalam pengembangan sistem yang mampu memenuhi kebutuhan pengguna terkait keamanan dan pemantauan paket secara optimal.

Website yang dirancang dalam sistem ini bertujuan untuk memudahkan pengguna dalam mengelola dan memantau paket yang diterima. Melalui *website* ini, pengguna dapat melakukan kontrol sepenuhnya menggunakan perangkat *mobile* maupun PC dari mana saja. Untuk memastikan hanya pengguna yang valid yang dapat mengakses *dashboard*, sistem dilengkapi dengan fitur *Sign Up* dan *Sign*

In. Pengguna juga tetap dapat mengganti akun pada *device* yang sama untuk mengakses *smartbox* lainnya. Setelah berhasil melakukan *Sign In*, pengguna akan diarahkan ke halaman *dashboard* yang menampilkan tabel berisi daftar paket beserta detailnya. Fitur utama pada *dashboard* meliputi tabel paket dengan kolom nama paket, nomor resi, status penerimaan, foto bukti penerimaan, tanggal dan jam penerimaan, serta tombol aksi untuk mengedit atau menghapus data. Selain itu, terdapat fitur *card* yang menampilkan informasi mengenai jumlah paket yang akan datang dan total paket yang sudah diterima. Fitur-fitur ini dirancang untuk mempermudah manajemen dan pemantauan paket oleh pengguna secara efisien dan terintegrasi.

Dari sisi manajemen data, diperlukan sebuah *database* yang mampu mengelola informasi pengguna serta data paket secara efisien. *Database* relasional menjadi pilihan yang tepat karena memiliki kemampuan untuk melakukan pembaruan data dengan cepat ketika terjadi perubahan, baik pada informasi pengguna maupun status paket. Dengan demikian, integritas dan konsistensi data dalam sistem dapat selalu terjaga.

Dari sisi komponen perangkat keras, pengelolaan seluruh sistem membutuhkan mikrokontroler yang memiliki kemampuan komputasi mumpuni, konsumsi daya rendah, serta konektivitas nirkabel yang mendukung integrasi IoT. ESP32 menjadi pilihan yang tepat karena dilengkapi prosesor dengan performa tinggi, konektivitas Wi-Fi bawaan, dan efisiensi daya yang baik. Selain itu, ESP32 memiliki jumlah pin GPIO yang cukup banyak, sehingga memungkinkan pemasangan berbagai modul tambahan seperti LCD TFT, *keypad*, dan *servo* tanpa mengurangi kinerja sistem. Keunggulan inilah yang menjadikan ESP32 sangat sesuai digunakan dalam pengembangan sistem *smartbox*.

3.3 Desain Sistem

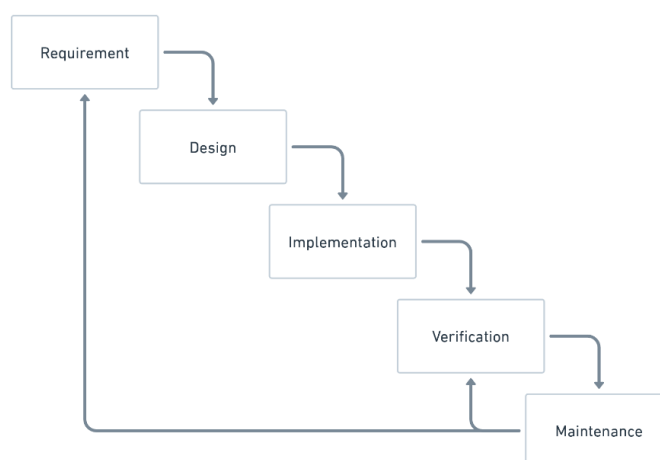
Pada tahap ini, dirancang sistem *smartbox* dengan fitur verifikasi resi dan monitoring visual untuk memastikan penerimaan paket yang aman, sesuai dengan kebutuhan yang telah dianalisis. Proses ini melibatkan penentuan desain sistem secara menyeluruh guna memastikan sistem yang dikembangkan memenuhi standar

keamanan dan fungsionalitas yang diharapkan. Sistem *smartbox* dirancang menjadi tiga komponen utama:

1. Perangkat Lunak: Sebuah *website* yang dirancang agar dapat diakses oleh pengguna untuk memantau dan mengelola pengiriman paket.
2. *Database*: Berfungsi sebagai penyimpanan data untuk mendukung pengelolaan informasi paket dan pengguna.
3. Perangkat Keras: Berperan sebagai komponen fisik utama untuk mendukung operasional sistem.

3.4 Pengembangan Sistem

Dalam pengembangan sistem, dibutuhkan suatu metode yang terstruktur agar proses dapat berjalan secara sistematis dan meminimalisasi kesalahan. Metode pengembangan ini berperan sebagai pedoman dalam mengatur tahapan mulai dari analisis kebutuhan hingga pemeliharaan sistem setelah implementasi. Salah satu metode yang umum digunakan adalah *Waterfall*. *Waterfall* merupakan pendekatan pengembangan perangkat lunak yang banyak diterapkan, bersifat *linear*, di mana setiap tahap harus diselesaikan sebelum melanjutkan ke tahap berikutnya. Pendekatan ini juga dikenal sebagai model sekuensial *linear* atau siklus hidup klasik. Model ini menyajikan alur hidup perangkat lunak secara berurutan, dimulai dari *requirement* atau analisis kebutuhan, perancangan (*design*), implementasi (*implementation*), verifikasi (*verification*), hingga pemeliharaan (*maintenance*) (Wahid, 2020) seperti yang ditampilkan pada Gambar 3.2.



Gambar 3.2 Tahapan Pengembangan Metode *Waterfall*

3.4.1 Requirement

Pada tahapan ini peneliti menganalisis terkait kebutuhan yang dibutuhkan dalam proses pengembangan sistem smartbox dengan fitur verifikasi resi dan monitoring visual barang kiriman. Berikut merupakan teknologi yang dibutuhkan dalam proses perancangan sistem yang ditampilkan pada Tabel 3.1 dan Tabel 3.2

Tabel 3.1 Kebutuhan Teknologi Perangkat Lunak

Nama Teknologi	Fungsi
Visual Studio Code	Visual Studio digunakan sebagai code editor untuk kebutuhan <i>website</i> dan IoT.
PlatformIO	<i>Environment</i> untuk pengembangan sistem embedded dan IoT yang berbasis Arduino
Fusion360	Membuat desain 3D dari box yang dirancang
Fritzing	Membuat skema <i>wiring</i> untuk perangkat keras
DrawIo dan Whimsical	Keduanya digunakan membuat diagram yang dibutuhkan
HTML, TailwindCSS dan JavaScript	Digunakan untuk membuat tampilan <i>frontend</i> dan logika <i>website</i>
Strapi	Digunakan sebagai backend untuk mengelola konten yang akan ditampilkan
MySQL	Digunakan untuk manajemen basis data relasional.

Tabel 3.2 Kebutuhan Teknologi Perangkat Keras

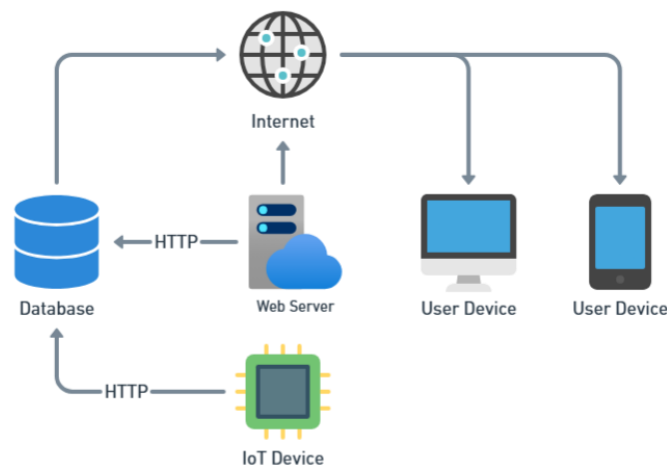
Nama Teknologi	Fungsi
ESP32 Devkit V1	Digunakan untuk menjalankan logika pemograman sistem yang dibangun
ESP32 Cam	Digunakan untuk melakukan proses foto paket
LCD TFT 1.9"	Menampilkan interface untuk <i>box</i>
Keypad 4x4	Digunakan untuk input angka dan huruf
Servo MG996R	Digunakan sebagai mekanisme buka tutup otomatis
<i>Stepdown</i> LM2596	Menurunkan tegangan sesuai dengan kebutuhan

Nama Teknologi	Fungsi
<i>Power Supply</i> 12V	Men-supply daya ke semua perangkat kelistrikan

3.4.2 Design

Sebagai dasar pemahaman dari system yang akan dibangun, pada bagian ini dipaparkan secara deskriptif mekanisme sistem *smartbox*. Sistem *smartbox* diawali dengan proses konfigurasi oleh pengguna (*owner*). Pengguna terlebih dahulu menghubungkan perangkat seperti *smartphone* ataupun PC ke *access point* yang dibuat ESP32 dan melakukan pengaturan pada *Wi-Fi Manager*. Setelah terhubung, pengguna mendaftar akun di *website* melalui halaman *sign up* dengan memasukkan *username*, *email*, dan *password*, lalu masuk (*sign in*) menggunakan akun yang telah dibuat. Pada halaman *profile*, pengguna memperoleh *Owner ID* yang kemudian disetel ke perangkat dengan menekan tombol pada *smartbox* tiga kali secara cepat, memasukkan *Owner ID*, dan menekan *enter*. Setelah konfigurasi selesai, pengguna dapat menambahkan data paket berupa nama barang dan nomor resi melalui *website*. Data tersebut ditampilkan dalam tabel yang dapat diubah atau dihapus sesuai kebutuhan. Dari sisi kurir, nomor resi dimasukkan melalui *keypad* pada *smartbox*. Jika valid, pintu akan terbuka untuk memasukkan paket dan menutup kembali secara otomatis. Modul *ESP32-CAM* kemudian mengambil foto sebagai bukti penerimaan, memperbarui status paket menjadi “sudah diterima” di sistem, serta menampilkan foto tersebut pada tabel *website* yang dapat diakses oleh pengguna. Sebagai tambahan, pemilik *smartbox* juga dapat membuka kotak untuk mengambil paket dengan menggunakan *Owner Code* yang dapat di-generate melalui halaman profil pada *website*. *Owner Code* tersebut kemudian dimasukkan ke *smartbox* melalui antarmuka yang tersedia, dan sistem akan membuka kotak sehingga pengguna dapat mengambil paket yang sudah diterima.

Untuk memberikan pemahaman yang lebih jelas mengenai struktur sistem yang akan dibangun, peneliti menyusun diagram arsitektur yang menggambarkan komponen utama serta hubungan antar komponennya. Melalui diagram yang ditampilkan pada Gambar 3.3, dapat terlihat gambaran umum mengenai kontribusi masing-masing komponen terhadap fungsionalitas sistem secara keseluruhan.

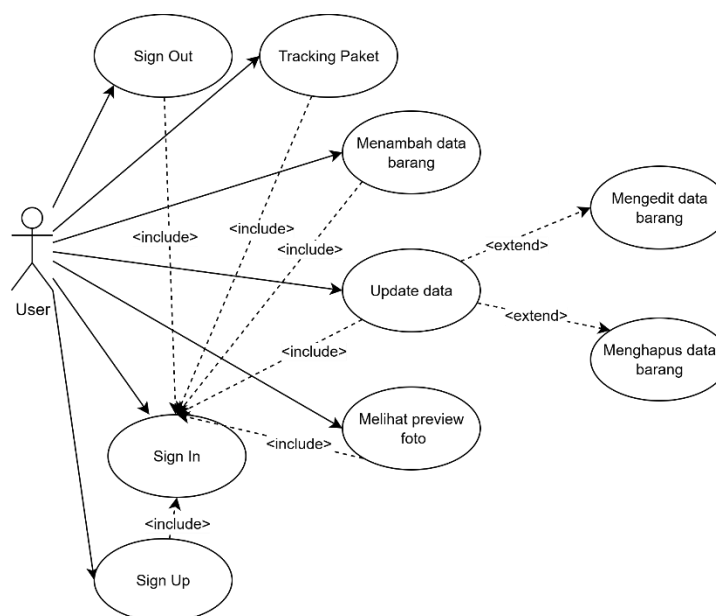


Gambar 3.3 Diagram Arsitektur Sistem

Berdasarkan diagram arsitektur sistem pada gambar 3.3, dapat dijelaskan bahwa ESP32 berperan sebagai unit pengendali utama (*main controller*) dalam sistem *smartbox* ini. Dari sisi perangkat keras, beberapa modul digunakan untuk mendukung fungsionalitas sistem. ESP32-CAM digunakan sebagai perangkat pengambil gambar untuk dokumentasi paket yang masuk. Modul input berupa *keypad* 4x4 digunakan untuk memasukkan data, baik angka maupun huruf yang dibutuhkan oleh sistem. Sebagai perangkat *output*, digunakan LCD TFT untuk menampilkan antarmuka (*interface*) kepada pengguna, sedangkan aktuator *servo* berfungsi sebagai mekanisme buka-tutup pintu *smartbox*. Dari sisi perangkat lunak, sistem dilengkapi dengan *website* dan *backend database*. *Website* dibangun menggunakan kombinasi HTML, Tailwind CSS, dan JavaScript untuk memberikan antarmuka pengguna yang responsif. Pada bagian *backend* menggunakan *framework* Strapi dan *database* MySQL sebagai tempat penyimpanan data pengguna dan paket. Untuk konektivitas antar komponen, digunakan protokol WebSocket antara ESP32 dan ESP32-CAM, sedangkan komunikasi antara ESP32 dan *Website* dengan *backend* dilakukan melalui REST API. Integrasi antara perangkat keras dan perangkat lunak ini membentuk sistem *smartbox* yang saling terhubung, efisien, dan mampu beroperasi secara otomatis. Untuk mempermudah perancangan dari *software* dan *hardware* akan dibagi menjadi 3 seperti dibawah ini.

a. *Software Design*

Untuk mendukung sistem *hardware* pada *smartbox*, diperlukan sebuah perangkat lunak berupa *website* yang berfungsi sebagai antarmuka pengguna. *Website* ini dirancang untuk memudahkan pengguna dalam memantau status paket yang dikirimkan. Melalui *website* ini, pengguna dapat dengan mudah mengetahui apakah paket telah diterima atau masih dalam proses pengiriman. Selain itu, *website* juga menyediakan fitur *preview* foto sebagai bukti penerimaan paket, sehingga pengguna dapat memastikan bahwa paket telah diterima dengan aman. Dalam perancangan perangkat lunak berupa *website*, diperlukan *use case diagram* untuk mempermudah proses pengembangan. Diagram ini berfungsi untuk memvisualisasikan hubungan antara aktor dan sistem, sehingga alur interaksi dapat dipahami dengan lebih jelas. Dengan adanya *use case diagram*, pengembang dapat merancang sistem sesuai dengan kebutuhan pengguna serta memastikan setiap fitur yang diperlukan telah terdefinisi dengan baik. Gambar 3.4 berikut menunjukkan *use case diagram* dari *website* yang dikembangkan.



Gambar 3.4 *Use Case Diagram Website*

Gambar 3.4 menunjukkan *use case diagram* untuk sistem *smartbox*, di mana *user* sebagai aktor utama memiliki akses ke berbagai fitur dalam sistem. Diagram ini menggambarkan hubungan antara pengguna dan sistem dalam melakukan proses *sign in*, pengelolaan data barang, serta bukti penerimaan paket melalui foto. Berikut adalah penjelasan masing-masing *use case* dalam sistem.

User adalah aktor utama yang memiliki akses ke semua fitur dalam sistem.

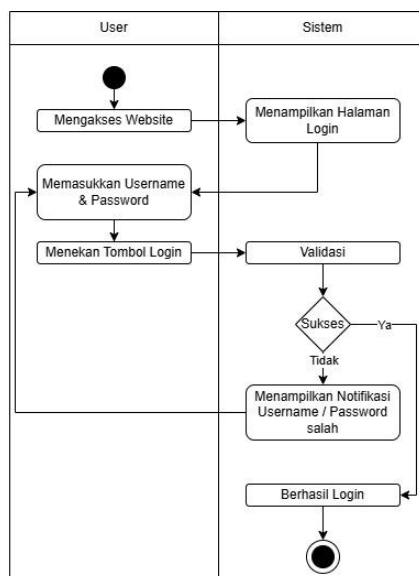
1. *Sign In* – Untuk dapat mengakses website utama pengguna harus melakukan proses *sign in* dengan memasukkan *username* dan *password* untuk mengakses fitur lainnya. Jika kredensial benar, pengguna diarahkan ke *dashboard* utama, jika salah, sistem menampilkan pesan *error*.
2. *Sign Up* – Jika user belum memiliki akun, user dapat melakukan proses *sign up* terlebih dahulu dengan mengisi data berupa *username*, email, dan *password*. Apabila data yang dimasukkan valid, pengguna akan langsung diarahkan ke halaman *Sign In* untuk melakukan *input* data. Namun, jika terdapat data yang tidak valid, misalnya email sudah terdaftar atau format *password* tidak sesuai ketentuan, maka sistem akan menampilkan pesan *error* yang sesuai untuk memberitahukan kesalahan *input* kepada pengguna.
3. *Sign Out* – Pengguna dapat melakukan proses *sign out* untuk dapat keluar dari akun yang sedang aktif sebelumnya
4. Menambah Data Barang – Pengguna dapat menambahkan barang dengan mengisi informasi seperti nama dan nomor resi. Setelah data dikirim, sistem menyimpannya ke dalam *database*. Jika ada kesalahan *input*, sistem meminta koreksi.
5. Menghapus Data Barang – Pengguna dapat menghapus barang dari sistem dengan memilihnya dari daftar dan mengonfirmasi penghapusan. Jika dikonfirmasi, data dihapus dari *database*.
6. Mengedit Data Barang – Pengguna dapat memperbarui informasi barang yang sudah tersimpan dengan memilihnya dari daftar, mengeditnya, dan menyimpan perubahan. Jika data tidak valid, sistem meminta revisi.

7. Melihat Foto Paket – Pengguna dapat melihat foto bukti penerimaan paket dengan memilih barang yang tersedia di daftar.
8. Men-*tracking* Paket - Pengguna dapat melakukan pelacakan (*tracking*) paket yang terdapat pada daftar tabel dengan 2 cara. Cara akses pertama adalah dengan menekan langsung nomor resi yang ingin dilacak pada tabel. Cara kedua adalah dengan masuk ke halaman Resi *Checker* yang dapat diakses melalui menu di sidebar.

Berdasarkan fitur-fitur yang terdapat pada *use case diagram*, dapat dibuat *activity diagram* yang memvisualisasikan alur proses dari setiap fitur dan aktivitas dalam sistem. *Activity diagram* ini berfungsi untuk memperjelas urutan langkah-langkah yang terjadi dalam suatu proses, mulai dari inisiasi hingga penyelesaian. Dengan adanya diagram ini, pengembang dapat memahami cara kerja setiap fitur secara lebih terstruktur dan efisien. Beberapa aktivitas utama yang digambarkan dalam *activity diagram* antara lain:

1. *Activity Diagram Sign In*

Activity Diagram berikut menggambarkan proses pengguna ke dalam *website*. Diagram ini menunjukkan alur kerja dari tahapan *sign in*, mulai dari *input* kredensial hingga hasil autentikasi oleh sistem. Proses ini dapat dilihat pada Gambar 3.5.

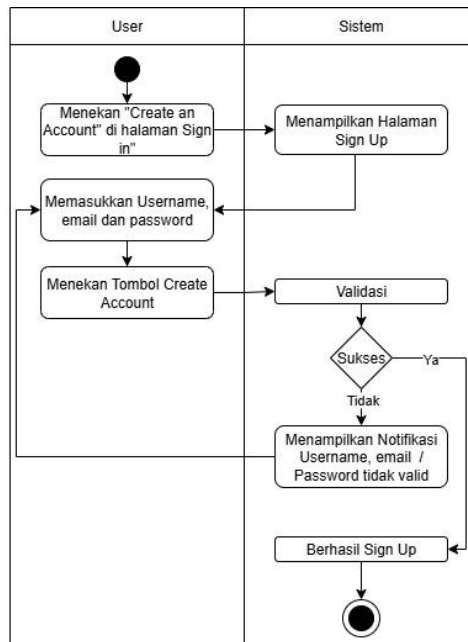


Gambar 3.5 *Activity Diagram Sign In*

Activity Diagram pada Gambar 3.5 menggambarkan proses *sign in* pengguna ke dalam *website*. Proses ini dimulai ketika pengguna membuka *website*, yang kemudian menampilkan halaman *sign in*. Pada tahap ini, pengguna dapat memasukkan *username* dan *password* yang sudah ada, lalu menekan tombol masuk. Setelah itu, sistem akan melakukan validasi dengan memeriksa keberadaan *username* dalam *database*. Jika *username* ditemukan dan kredensial sesuai, sistem akan menampilkan notifikasi berhasil *sign in*, dan pengguna akan *redirect* ke halaman utama *website*. Sebaliknya, jika validasi gagal, sistem akan menampilkan pesan kesalahan dan meminta pengguna untuk memasukkan kembali kredensial yang benar.

2. *Activity Diagram Sign Up*

Activity Diagram berikut menggambarkan proses *Sign Up* pengguna ke dalam *website*. Diagram ini menunjukkan alur kerja dari tahapan *sign up*, mulai dari *input* kredensial hingga hasil autentikasi oleh sistem. Proses ini dapat dilihat pada Gambar 3.6.



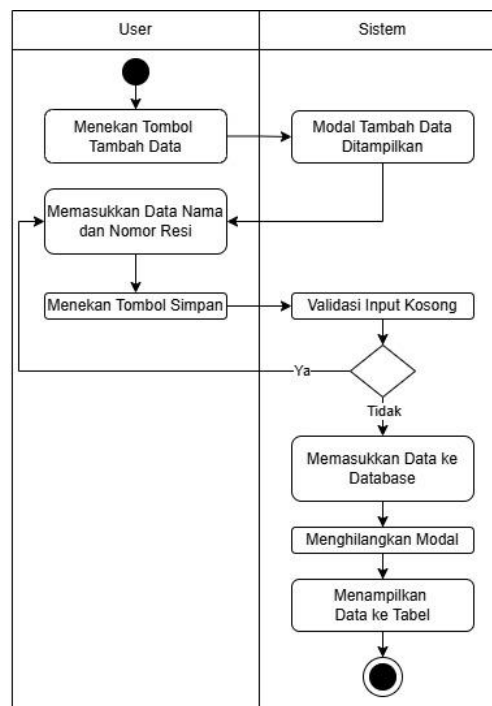
Gambar 3.6 *Activity Diagram Sign Up*

Activity Diagram pada Gambar 3.6 menggambarkan proses *Sign Up* atau pendaftaran akun pengguna. Proses ini dimulai ketika pengguna membuka

halaman *Sign Up* dan memasukkan *username*, email, serta *password* yang valid. Setelah menekan tombol *Sign Up*, sistem akan melakukan validasi data, seperti memastikan email berformat benar dan belum terdaftar, serta *password* memenuhi syarat. Apabila seluruh data dinyatakan valid, sistem akan menyimpan informasi pengguna ke dalam *database* dan menampilkan notifikasi bahwa pendaftaran berhasil, kemudian mengarahkan pengguna ke halaman *Sign In*. Sebaliknya, jika terdapat kesalahan, sistem akan menampilkan pesan *error* dan meminta pengguna untuk memperbaiki input yang diberikan.

3. Activity Diagram Tambah Data

Activity Diagram berikut menggambarkan proses tambah data yang dilakukan pengguna ke dalam *website*. Diagram ini menunjukkan alur kerja dari tahapan tambah data. Proses ini dapat dilihat pada Gambar 3.7.



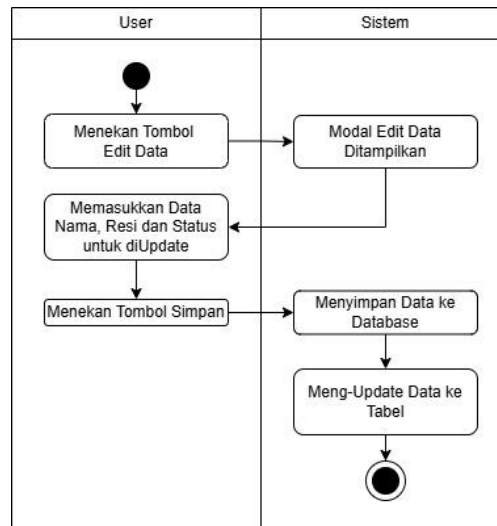
Gambar 3.7 Activity Diagram Tambah Data

Gambar 3.7 merupakan *Activity Diagram* yang menggambarkan proses penambahan data barang yang dilakukan oleh pengguna melalui *website*. Proses ini dimulai ketika pengguna membuka *website*, yang kemudian

menampilkan halaman utama. Pada tahap ini, pengguna dapat menekan tombol “Tambah Data”, sehingga sistem akan menampilkan modal *input* untuk memasukkan informasi barang. Pengguna diwajibkan mengisi nama barang dan nomor resi. Setelah data dimasukkan, sistem akan melakukan validasi untuk memastikan bahwa kedua kolom tidak kosong. Jika validasi berhasil, pengguna dapat menekan tombol Simpan, dan sistem akan menyimpan data ke dalam *database*. Setelah proses penyimpanan selesai, modal *input* akan tertutup secara otomatis, dan data barang yang baru ditambahkan akan muncul dalam tabel di halaman utama.

4. Activity Diagram Edit Data

Activity Diagram berikut menggambarkan proses pengeditan data barang yang sudah ada di dalam tabel melalui interaksi pengguna dengan *website*. Diagram ini menunjukkan alur kerja tahapan edit data, mulai dari pemilihan data hingga penyimpanannya setelah dilakukan perubahan. Proses ini dapat dilihat pada Gambar 3.8.



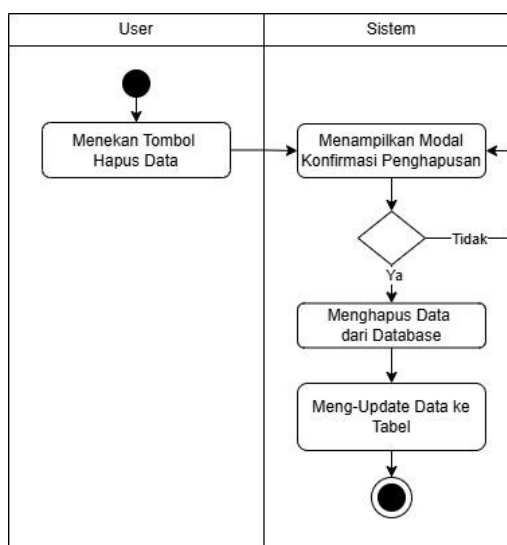
Gambar 3.8 Activity Diagram Edit Data

Activity Diagram pada gambar 3.8 menggambarkan proses pengeditan data barang yang dilakukan oleh pengguna melalui *website*. Proses ini dimulai ketika pengguna membuka *website*, yang kemudian menampilkan halaman utama. Pada tahap ini, pengguna dapat memilih data yang ingin diedit, lalu

menekan tombol Edit untuk menampilkan modal edit. Pada modal ini, terdapat tiga *input* yang dapat diperbarui, yaitu nama barang, nomor resi, dan status penerimaan. Setelah pengguna melakukan perubahan, mereka dapat menekan tombol Simpan, sehingga sistem akan memperbarui data di dalam *database*. Setelah proses pembaruan selesai, modal edit akan tertutup secara otomatis, dan tabel akan diperbarui untuk menampilkan hasil perubahan yang telah dilakukan.

5. Activity Diagram Hapus Data

Activity Diagram berikut menggambarkan proses penghapusan data barang yang sudah ada di dalam tabel melalui interaksi pengguna dengan *website*. Diagram ini menunjukkan alur kerja tahapan hapus data, mulai dari pemilihan data yang akan dihapus hingga proses penghapusan dari sistem. Proses ini dapat dilihat pada Gambar 3.9.



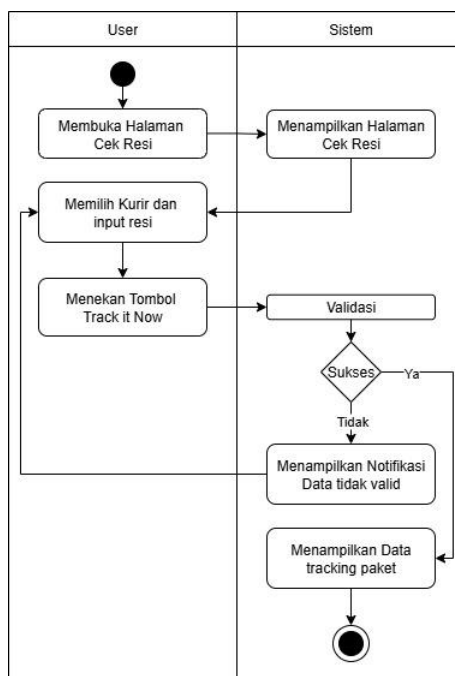
Gambar 3.9 Activity Diagram Hapus Data

Activity Diagram berikut menggambarkan proses penghapusan data barang yang dilakukan oleh pengguna melalui *website*. Proses ini dimulai ketika pengguna membuka *website*, yang kemudian menampilkan halaman utama. Pada tahap ini, pengguna dapat memilih data yang ingin dihapus, lalu menekan tombol hapus untuk menampilkan modal konfirmasi penghapusan. Pada modal ini, sistem akan meminta konfirmasi dari pengguna sebelum

menghapus data secara permanen. Jika pengguna menekan tombol hapus, sistem akan menghapus data dari *database*, menutup modal secara otomatis, dan memperbarui tabel untuk menampilkan kondisi terbaru setelah penghapusan dilakukan.

6. Activity Diagram Cek Resi

Activity Diagram berikut menggambarkan proses cek resi atau *tracking* paket pengguna punya. Diagram ini menunjukkan alur kerja tahapan dari mulai akses halaman hingga *input* yang harus dimasukkan pengguna. Proses ini dapat dilihat pada Gambar 3.10.



Gambar 3.10 Activity Diagram Cek Resi

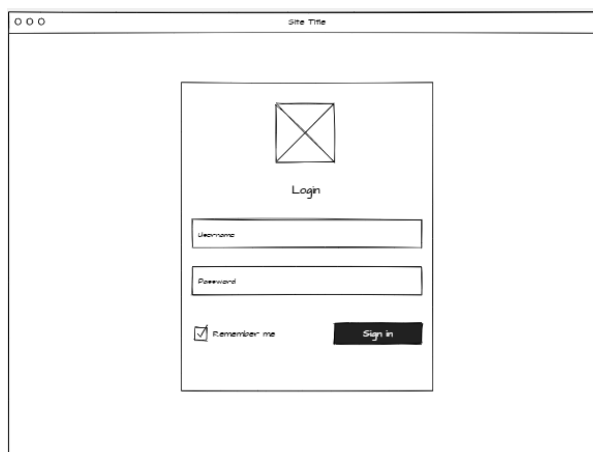
Proses cek resi dimulai ketika pengguna masuk ke halaman Resi Checker. Pada halaman ini, pengguna terlebih dahulu memilih jasa kurir yang digunakan dari daftar *dropdown* yang tersedia, kemudian memasukkan nomor resi pada *input field* yang disediakan. Setelah itu, pengguna menekan tombol “Track it Now” untuk memulai proses pencarian data. Sistem akan memproses permintaan tersebut dengan mengambil data berdasarkan jasa ekspedisi dan nomor resi yang dimasukkan. Apabila data valid dan ditemukan, sistem akan menampilkan detail informasi pelacakan paket di bawah tombol tersebut.

Namun, jika data tidak valid atau nomor resi tidak ditemukan, sistem akan menampilkan pesan pemberitahuan yang meminta pengguna untuk memeriksa kembali dan memasukkan *input* yang benar.

Berdasarkan hasil perancangan *website*, diperlukan desain antarmuka pengguna (*User Interface/UI*) untuk mendukung berbagai fungsionalitas dalam aplikasi. Desain UI ini bertujuan untuk memastikan pengalaman pengguna yang intuitif, efisien, dan mudah digunakan. Beberapa halaman utama yang dirancang dalam sistem ini meliputi:

1. Desain *Wireframe* Halaman *Sign In Website*

Halaman *Sign In* menampilkan *input field* untuk *username* dan *password* sebagai syarat agar pengguna dapat mengakses sistem. Desain *wireframe* untuk halaman *sign in* dapat dilihat pada Gambar 3.11.



Gambar 3.11 *Wireframe* Halaman *Sign In*

2. Desain *Wireframe* Halaman *Sign Up*

Halaman *Sign Up* menampilkan *input field* untuk *username* email dan *password* sebagai syarat agar pengguna membuat akun baru. Desain *wireframe* untuk halaman *Sign Up* dapat dilihat pada Gambar 3.12.

Gambar 3.12 *Wireframe* Halaman *Sign Up*

3. Desain *Wireframe* Halaman Utama *Website*

Halaman utama *website* menampilkan sebuah tabel data yang berisi informasi dari data yang telah di-*input* oleh pengguna. Data yang dimasukkan akan langsung ditampilkan dalam tabel untuk memudahkan pemantauan. Selain itu, terdapat tombol Tambah Data, yang memungkinkan pengguna menambahkan informasi baru ke dalam sistem. Desain *wireframe* halaman utama dapat dilihat pada Gambar 3.13.

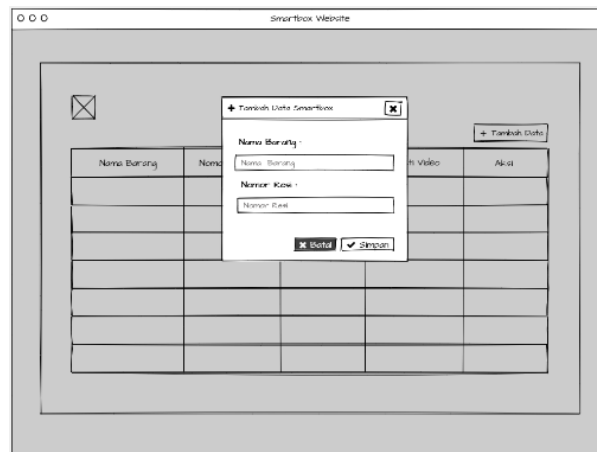
Nama Barang	Nomor Resi	Status	Bukti Video	Aksi

Gambar 3. 13 *Wireframe* Halaman Utama

4. Desain *Wireframe* Modal Tambah Data

Pada halaman utama *website*, terdapat sebuah tombol Tambah Data, yang ketika ditekan akan menampilkan modal untuk menambah data. Dalam modal tersebut, terdapat dua *input field*, yaitu Nama Barang dan Nomor Resi.

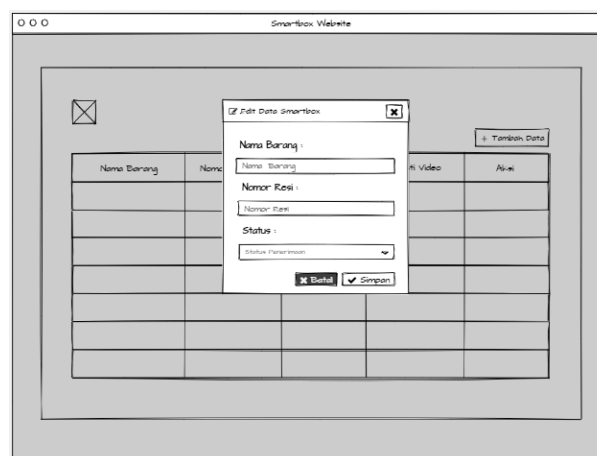
Pengguna diwajibkan mengisi kedua *input* ini, karena sistem akan melakukan validasi untuk memastikan data tidak kosong sebelum disimpan. Desain *wireframe* untuk fitur Tambah Data dapat dilihat pada Gambar 3.14.



Gambar 3.14 *Wireframe* Modal Tambah Data

5. Desain *Wireframe* Modal Edit Data

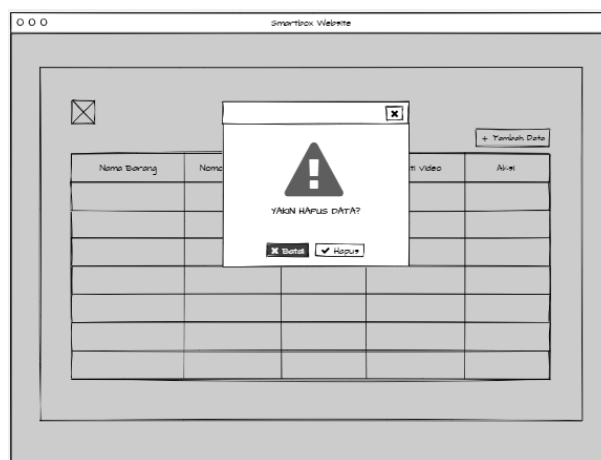
Pada halaman utama *website*, terdapat tabel data yang menampilkan barang yang telah diinput pengguna. Setiap entri dapat diedit dengan menekan tombol Edit, yang akan menampilkan modal edit data. Dalam modal ini, pengguna dapat mengubah Nama Barang, Nomor Resi, dan Status Pengiriman (*dropdown*). Setelah menekan tombol Simpan, data akan diperbarui di *database*, modal akan tertutup, dan tabel akan menampilkan perubahan. Desain *wireframe* fitur Edit Data dapat dilihat pada Gambar 3.15.



Gambar 3.15 *Wireframe* Modal Edit Data

6. Desain *Wireframe* Modal Hapus Data

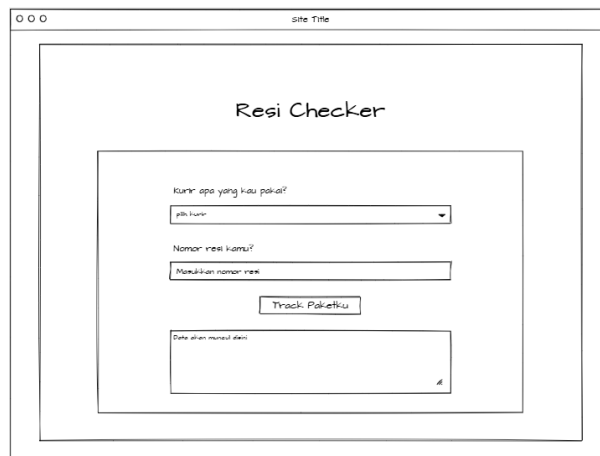
Pada halaman utama *website*, di baris entri yang diinginkan pengguna bisa melakukan proses hapus data dengan menekan ikon tempat sampah. Setelah itu, sistem akan menampilkan modal konfirmasi penghapusan. Dalam modal ini, terdapat dua tombol: Hapus untuk mengonfirmasi penghapusan dan tombol Batal untuk membatalkan aksi. Jika pengguna menekan Hapus, data akan dihapus dari *database*, modal akan tertutup, dan tabel akan diperbarui. Desain *wireframe* fitur Konfirmasi Hapus dapat dilihat pada Gambar 3.16.



Gambar 3.16 *Wireframe* Modal Hapus Data

7. Desain *Wireframe* Halaman Cek Resi

Pada halaman Cek Resi, pengguna dapat melakukan proses pelacakan paket dengan mudah. Halaman ini dilengkapi dengan *dropdown* untuk memilih jasa kurir yang digunakan, *input field* untuk memasukkan nomor resi, dan tombol “Track it Now” untuk memulai proses pengecekan. Setelah pengguna memilih jasa kurir dan memasukkan nomor resi, kemudian menekan tombol tersebut, sistem akan memproses permintaan sesuai data yang dimasukkan. Jika data valid dan nomor resi ditemukan, maka informasi detail pengiriman akan muncul di bagian bawah tombol tersebut. Namun, jika data tidak valid atau nomor resi tidak ditemukan, sistem akan menampilkan notifikasi yang meminta pengguna untuk memasukkan ulang data dengan benar agar proses pelacakan dapat dilakukan. Desain *wireframe* halaman cek resi dapat dilihat di gambar 3.17.



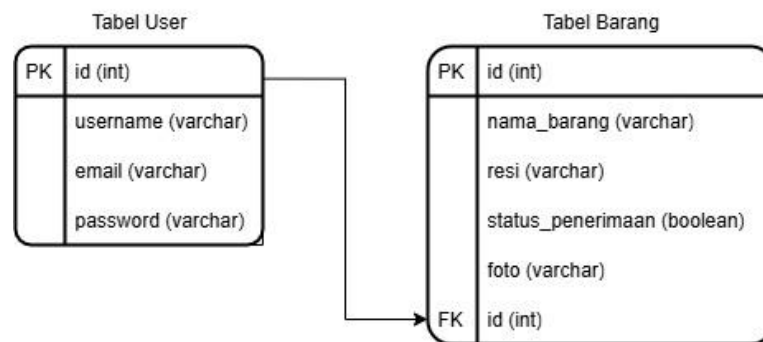
Gambar 3.17 *Wireframe* Halaman Cek Resi

Frontend yang dibangun tidak dapat berkomunikasi langsung dengan *database*, melainkan mengirimkan permintaan (*request*) ke *backend*. Pada sistem ini, *backend* menggunakan Strapi, yaitu *headless* CMS berbasis Node.js yang bertindak sebagai *server* API. Strapi berperan untuk menerima permintaan dari *frontend*, melakukan pemrosesan seperti validasi data dan autentikasi, serta menyediakan endpoint REST API untuk operasi CRUD (*Create*, *Read*, *Update*, *Delete*) yang dibutuhkan. Selain itu, Strapi juga mengatur koneksi ke *database* MySQL yang digunakan untuk menyimpan semua data paket secara terstruktur.

Meskipun Strapi telah menyediakan API *premade* secara otomatis berdasarkan struktur data yang dibuat, sistem ini juga mendukung penambahan *custom logic* melalui fitur *controller* dan *route customization*. Dengan demikian strapi dapat menambahkan fungsi khusus seperti pengolahan data tambahan, filter khusus, atau integrasi dengan layanan lain sebelum mengirimkan respons ke *frontend* sesuai kebutuhan sistem. Dengan adanya Strapi sebagai *backend*, *frontend* dapat mengakses data secara terstruktur melalui API *endpoint* yang telah disediakan, misalnya untuk menampilkan daftar paket yang tersimpan atau mengirim data *input*-an baru ke *database* melalui *backend*. Arsitektur ini memastikan pemisahan yang jelas antara tampilan (*frontend*) dan logika *server* serta *database* (*backend*), sehingga pengembangan sistem menjadi lebih modular, fleksibel, dan terintegrasi dengan baik.

b. Database Design

Sebagai pendukung sistem *software* dan *hardware*, diperlukan *database* yang dapat menyimpan serta mengelola data dengan efisien. Dalam hal ini, digunakan MySQL yang berjalan di lokal sebagai sistem manajemen basis data untuk menyimpan informasi terkait paket yang diproses oleh *smartbox*. Gambar 3.18 menampilkan relasi antar tabel user dan barang.



Gambar 3.18 Relasi Tabel *user* dan barang

Pada gambar 3.18 diperlihatkan relasi antara tabel user dan barang. Berdasarkan skema basis data yang disajikan, terdapat relasi satu ke banyak (*one-to-many*) antara tabel_user dan tabel_barang, yang menjadi arsitektur inti dari struktur data aplikasi. Relasi ini terjalin melalui kunci utama (id) pada tabel_user yang diacu oleh kunci asing (id) pada tabel_barang. Secara fungsional, relasi ini memungkinkan satu entri pengguna (*user*) dapat dihubungkan dengan banyak entri barang (barang), namun setiap barang hanya dapat terhubung dengan satu pemilik pengguna yang valid. Penerapan relasi ini memastikan integritas data, menghindari data barang yang tidak terasosiasi dengan pemiliknya, serta memfasilitasi pengambilan data yang efisien di mana informasi barang dapat dengan mudah dihubungkan dengan detail pengguna terkait tanpa adanya duplikasi data yang tidak perlu.

Pada tahap *prototyping* ini, digunakan MySQL *self-hosted* yang dijalankan secara lokal. Penggunaan MySQL lokal dipilih karena sejalan dengan kebutuhan sistem yang dirancang, khususnya dalam integrasinya dengan Strapi sebagai *backend*. Dengan menjalankan MySQL langsung di lokal, Strapi dapat

melakukan koneksi *database* secara cepat tanpa konfigurasi akses jaringan eksternal yang kompleks, sehingga proses pengembangan dan pengujian menjadi lebih efisien.

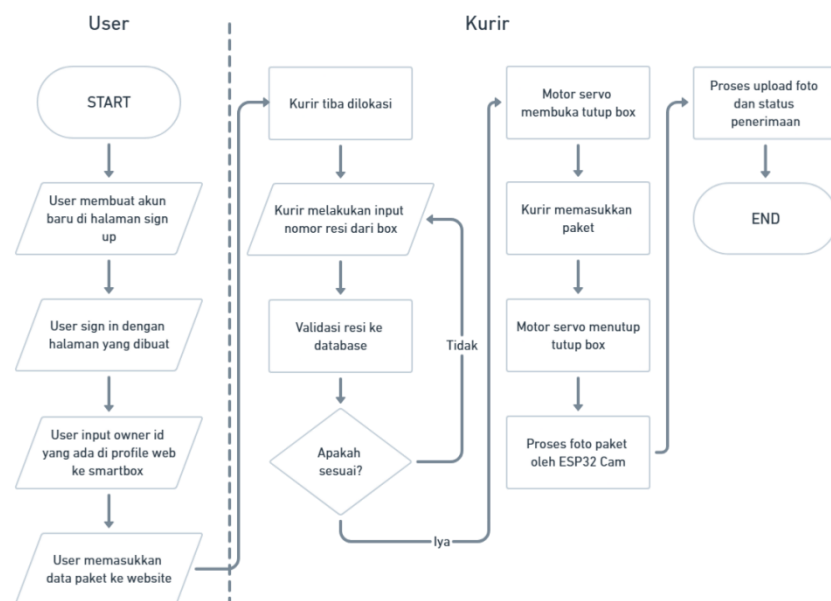
Selain itu, seluruh data tetap berada dalam satu ekosistem yang terkontrol tanpa ketergantungan pada layanan pihak ketiga. Pendekatan ini juga memungkinkan penyimpanan hasil foto secara langsung di *server* lokal, sehingga tidak diperlukan penyimpanan *cloud* eksternal. Hal ini tidak hanya mengurangi biaya tambahan, tetapi juga meningkatkan efisiensi karena akses ke foto dapat dilakukan lebih cepat tanpa latensi jaringan eksternal. Dengan *database* MySQL dan penyimpanan foto berada dalam satu lingkungan yang terintegrasi dengan Strapi, proses pengambilan dan pengelolaan data menjadi lebih optimal serta mendukung kinerja sistem *smartbox* secara keseluruhan, terutama pada tahap *prototyping* dan pengujian sebelum implementasi skala besar.

c. *Hardware Design*

Sebagai komponen utama dalam sistem *Smartbox* yang dikembangkan, diperlukan perangkat keras yang berfungsi untuk mendukung proses otomatisasi dan interaksi dengan sistem. Perangkat keras yang digunakan mencakup mikrokontroler ESP32, modul kamera ESP32-CAM, serta mekanisme buka-tutup *box* yang dikendalikan secara otomatis. Mikrokontroler ESP32 berperan sebagai pusat kendali yang menangani proses verifikasi nomor resi, aktivasi kunci *box*, serta komunikasi dengan *database*. Dengan struktur ini, sistem diharapkan dapat memberikan solusi yang efektif dan efisien untuk meningkatkan keamanan dan kemudahan dalam proses penerimaan paket. Untuk memberikan gambaran yang lebih komprehensif, peneliti menyertakan berbagai diagram yang mendukung, seperti *flowchart*, Diagram Blok, *wiring diagram*. Berikut ini adalah rancangan sistem yang telah disusun.

a. *Flowchart Sistem Smartbox*

Pembuatan *flowchart* bertujuan untuk meminimalkan kesalahan selama proses pengembangan produk. Dengan memiliki panduan visual yang jelas, setiap langkah dalam perancangan dan implementasi sistem dapat diidentifikasi dan diikuti dengan lebih mudah. Gambar 3.19 menampilkan flowchart sistem.



Gambar 3.19 Flowchart Sistem *Smartbox*

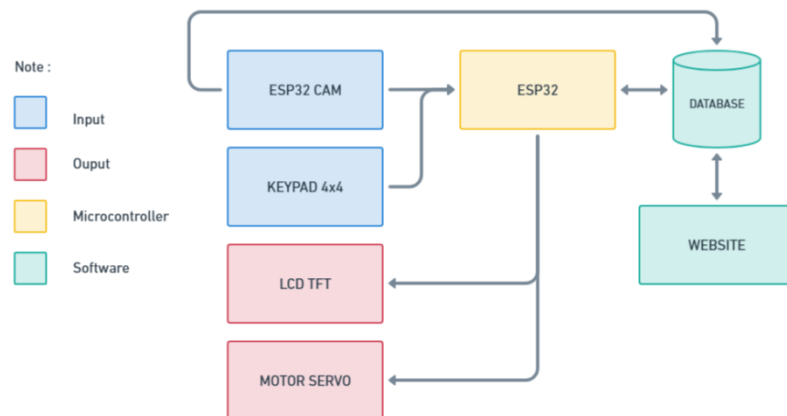
Flowchart membantu dalam merinci alur kerja, dari tahap inisiasi hingga penyelesaian, sehingga memungkinkan identifikasi potensi hambatan atau kesalahan sejak dini. Adapun alur sistem *smartbox* yang dibuat seperti pada gambar 3.19 sebagai berikut.

1. *Start*: Tahap awal proses, sistem siap untuk menerima *interaksi* dari pengguna.
2. User akan melakukan proses *sign up* ke *website* untuk membuat akun baru
3. Dengan akun baru yang dibuat *user* akan melakukan *sign in* untuk dapat mengakses *website*
4. User memasukkan nomor resi ke *Website*: User melakukan proses penginputan nomor resi yang digunakan sebagai “*password*” ke *Website*.

5. Pengiriman data ke *database*: Data yang sudah diinputkan dari *website* akan disimpan di *database*.
6. Kurir tiba di lokasi: Kurir akan melakukan penginputan nomor resi yang tertera di paket
7. Pengecekan *database*: Hasil *input* kurir akan melakukan verifikasi apakah nomor resi yang diinput *user* dengan yang ada di *database* sama atau tidak
8. Jika verifikasi berhasil (*Yes*): Sistem mengonfirmasi bahwa nomor resi sesuai, dan proses akan berlanjut.
9. Jika verifikasi gagal (*No*): Sistem akan menolak akses dan kembali ke tahap "Kurir melakukan *input* nomor resi".
10. *Motor servo* membuka tutup *box* : Setelah sistem memastikan bahwa nomor resi sama selanjutnya *motor servo* akan akan untuk otomatis membuka tutup *box*.
11. Kurir memasukkan paket ke dalam *box*
12. *Motor servo* menutup tutup *box* : Setelah semua selesai *motor servo* akan kembali aktif untuk menutup *box*.
13. ESP32-Cam : Proses foto akan dimulai untuk memberi bukti paket yang masuk.
14. *Upload* foto ke *database* dan *update* status pengiriman.
15. *End*: Tahap akhir proses, dimana sistem selesai melakukan tugasnya

b. Diagram Blok

Blok diagram sistem adalah gambaran visual dari sistem yang memperlihatkan interaksi dan fungsi antar komponen utamanya. Diagram ini menggunakan blok-blok untuk melambangkan komponen atau subsistem, sementara garis-garis menunjukkan hubungan atau aliran data antara komponen-komponen tersebut (Winata & Suweno, 2022). Pada Gambar 3.20 menampilkan blok diagram sistem.



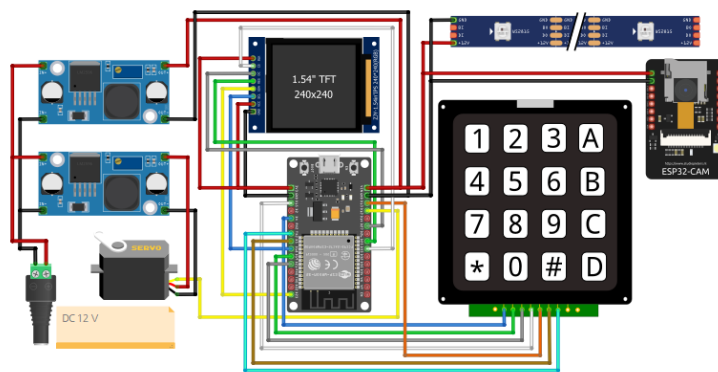
Gambar 3.20 Diagram Blok Sistem

Pada Gambar 3.20 menunjukkan diagram blok sistem terdiri dari beberapa komponen antara lain.

1. ESP32 adalah mikrokontroler yang berfungsi sebagai otak dari sistem. Ia menerima data dari *keypad* 4x4 dan ESP32CAM, memproses informasi, dan mengendalikan perangkat lain seperti *Motor Servo* dan *LCD TFT*. ESP32 juga menangani komunikasi dengan Strapi.
2. *Keypad* 4x4 adalah perangkat yang digunakan untuk melakukan *input* berupa angka dan huruf.
3. ESP32-Cam adalah perangkat kamera yang berbasis ESP32. ESP32-Cam berfungsi sebagai alat untuk memfoto paket dan akan berkomunikasi dengan ESP32 Utama dengan protokol WebSocket
4. LCD TFT ini digunakan untuk menampilkan informasi visual, seperti *input* angka dari *keypad* dan juga pesan informasi ketika inputan salah
5. *Database* MySQL adalah sistem manajemen basis data relasional (RDBMS) yang digunakan untuk menyimpan dan mengelola data dalam bentuk tabel.
6. *Website* ini terhubung dengan MySQL melalui Strapi *backend* dan berfungsi sebagai antarmuka pengguna untuk mengelola sistem. Melalui *website* ini, *user* dapat menginput nomor resi, dan juga melihat kondisi paketnya

c. *Wiring Diagram*

Wiring diagram menunjukkan jalur pengkabelan dari komponen-komponen yang digunakan. *Wiring diagram* akan mempermudah perencanaan kelistrikan dan juga bisa mencegah adanya kegagalan saat perakitan. Gambar 3.21 merupakan jaringan koneksi antar komponen.



Gambar 3.21 *Wiring Diagram* Sistem

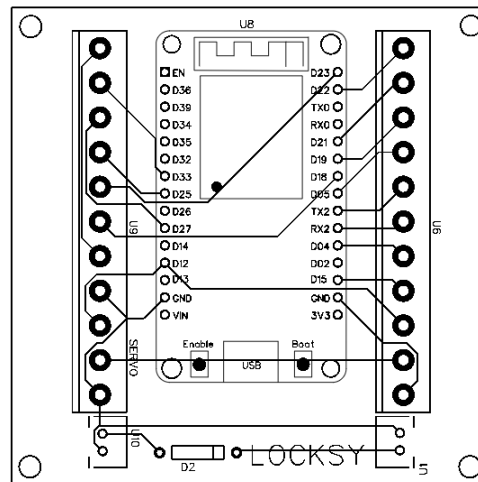
Dalam gambar 3.21 terdapat komponen-komponen yang dipakai untuk pengembangan sistem ini, antara lain.

1. ESP32
2. ESP32 Cam
3. LCD TFT 1.9"
4. Keypad 4x4
5. Motor Servo MG996R
6. Adaptor DC 12V
7. Module Stepdown LM2596
8. LED Strip

d. *Desain PCB Sistem*

Desain PCB dilakukan menggunakan *software* EasyEDA untuk mempermudah dan mempercepat proses pemasangan komponen pada papan sirkuit. Perancangan PCB ini dibuat berdasarkan *wiring diagram*

yang telah disusun sebelumnya, sehingga seluruh rangkaian dapat terintegrasi dengan rapi dan memiliki keandalan yang baik saat digunakan. Desain dari PCB dapat dilihat pada gambar 3.22



Gambar 3. 22 Desain PCB *Smartbox*

Untuk koneksi antar komponen pada rangkaian, digunakan beberapa jenis konektor sesuai kebutuhan. *Pin header* digunakan sebagaiudukan ESP32 agar mempermudah proses pemasangan dan pelepasan mikrokontroler saat diperlukan. Sementara itu, untuk koneksi GPIO ke modul LCD TFT, *keypad*, dan *servo*, digunakan *terminal block* sehingga memudahkan proses *prototyping* dan penggantian kabel. Adapun konektor yang berhubungan dengan suplai daya menggunakan konektor jenis XH, yang memiliki penguncian lebih kuat dan aman untuk sambungan tegangan.

e. Desain Box

Desain *box* pada sistem ini dibuat dengan dimensi 30 x 30 x 20 cm dan dirancang memiliki dua kompartemen. Kompartemen pertama berfungsi sebagai tempat untuk memasukkan paket, sedangkan kompartemen kedua digunakan untuk menyimpan komponen elektronik dan rangkaian sistem. Pembagian ini bertujuan agar paket yang

dimasukkan tidak mengganggu atau merusak komponen internal. Desain 3D dari *box* bisa dilihat pada gambar 3.23.



Gambar 3.23 Desain Box

3.4.3 Implementation

Setelah melalui tahapan perancangan desain sistem, langkah berikutnya adalah tahap implementasi yang diwujudkan melalui penulisan kode program. Pada penelitian ini, hasil rancangan diimplementasikan menjadi dua produk utama, yaitu perangkat keras (hardware) berupa SmartBox dan perangkat lunak (software) berupa aplikasi web yang terdiri atas *frontend*, *backend*, serta *database*. Implementasi perangkat keras dilakukan menggunakan bahasa pemrograman C++ dengan bantuan Visual Studio Code sebagai *code editor* dan PlatformIO sebagai *embedded system development environment*. Perangkat keras ini berfungsi sebagai pengendali utama *SmartBox*, yang mencakup pengolahan input, pengendalian mekanisme buka-tutup, serta komunikasi dengan sistem kamera. Sementara itu, pada sisi perangkat lunak, implementasi difokuskan pada pembangunan aplikasi web. Bagian *frontend* dikembangkan menggunakan HTML untuk struktur halaman, dipadukan dengan TailwindCSS untuk mempermudah proses *styling* antarmuka, serta JavaScript untuk menangani logika interaktif pada sisi klien. Seluruh proses implementasi perangkat lunak dilakukan menggunakan Visual Studio Code sebagai lingkungan pengembangan utama. Untuk sisi *backend* dan basis data, penelitian ini memanfaatkan Strapi sebagai *headless CMS* yang berperan dalam manajemen data serta penyediaan API, sedangkan penyimpanan data dilakukan menggunakan

MySQL. Integrasi antara *frontend*, *backend*, dan *database* menghasilkan sebuah sistem yang saling terhubung sehingga mampu mendukung fungsi utama SmartBox, yaitu verifikasi nomor resi serta pengelolaan bukti penerimaan berupa foto maupun rekaman kamera.

3.4.4 Verification

Pada tahap verification atau tahap pengujian, peneliti melakukan pengujian pada dua aspek utama, yaitu perangkat lunak (*website*) dan perangkat keras (*smartbox*). Dari sisi *website*, pengujian dilakukan menggunakan metode *black box testing*. Metode ini berfokus pada pengujian *fungsi sistem* tanpa memperhatikan bagaimana kode program dijalankan. Dengan pendekatan ini, peneliti dapat memastikan bahwa setiap fitur yang ada, seperti input data resi, pengelolaan status paket, dan tampilan dokumentasi foto, dapat berfungsi sesuai kebutuhan pengguna. Sementara itu, pada sisi *smartbox*, pengujian dilakukan secara bertahap. Pertama, setiap komponen perangkat keras diuji secara individual. Tujuan dari pengujian ini adalah memastikan bahwa masing-masing komponen bekerja sesuai dengan spesifikasinya. Setelah semua komponen berhasil diuji secara terpisah, dilakukan pengujian integrasi perangkat keras dalam bentuk sistem penuh. Pengujian ini mencakup alur lengkap, mulai dari input nomor resi, pembukaan kotak secara otomatis, pengambilan foto oleh kamera, hingga pembaruan status penerimaan pada sistem backend. Dengan serangkaian pengujian tersebut, dapat dipastikan bahwa baik perangkat keras maupun perangkat lunak dapat bekerja sesuai rancangan, serta saling terintegrasi untuk mewujudkan fungsi utama *Smartbox*.

3.4.5 Maintenance

Pada tahap terakhir dari metode pengembangan *Waterfall*, yaitu *maintenance*, dilakukan proses pemeliharaan terhadap sistem yang telah diimplementasikan dan diuji. Dalam penelitian ini, tahap *maintenance* mencakup kegiatan perbaikan *bug* pada perangkat lunak maupun perangkat keras apabila ditemukan kesalahan. Selain itu, tahap ini juga meliputi pengoptimalan fitur-fitur yang sudah ada agar sistem dapat berjalan lebih stabil, andal, serta sesuai dengan kebutuhan pengguna.

3.5 Pengujian Sistem

Pengujian sistem merupakan tahapan penting dalam proses pengembangan untuk memastikan bahwa sistem yang dibangun dapat berjalan sesuai dengan kebutuhan dan spesifikasi yang telah ditentukan pada tahap perancangan. Tujuan dari pengujian ini adalah untuk mendeteksi adanya kesalahan, memastikan fungsionalitas berjalan sebagaimana mestinya, serta memverifikasi integrasi antara perangkat keras dan perangkat lunak. Pada penelitian ini, pengujian sistem dilakukan dengan dua pendekatan utama, yaitu pengujian pada sisi perangkat keras (*hardware*) dan pengujian pada sisi perangkat lunak (*software*). Pengujian perangkat keras difokuskan pada pemeriksaan fungsionalitas setiap komponen serta kinerja sistem *smartbox* secara keseluruhan. Sementara itu, pengujian perangkat lunak dilakukan menggunakan metode *black box testing* dengan tujuan memastikan bahwa setiap fitur pada aplikasi web dapat digunakan sesuai fungsinya tanpa melihat alur kode di dalamnya.

3.5.1 Pengujian Perangkat Keras

Pada pengujian ini, dilakukan pengujian fungsionalitas komponen yang bertujuan untuk memastikan setiap komponen dalam sistem *smartbox* beroperasi sesuai dengan spesifikasi yang telah dirancang. Proses pengujian dilakukan dengan memeriksa setiap komponen secara terpisah guna mengevaluasi kinerjanya dalam kondisi operasional sebenarnya, sekaligus menentukan apakah diperlukan penyesuaian lebih lanjut. Melalui pendekatan ini, potensi kesalahan atau ketidaksesuaian dapat diidentifikasi sedini mungkin sebelum dilakukan proses integrasi sistem secara keseluruhan. Tabel 3.3 Menunjukkan skenario pengujian pada setiap komponen yang digunakan.

Tabel 3.3 Uji Fungsionalitas Komponen

No.	Pengujian	Skenario uji	Harapan
1	Kinerja Upload ESP32 Cam	mengupload dengan ukuran bervariasi	ESP32 Cam dapat mengupload dengan ukuran

No.	Pengujian	Skenario uji	Harapan
2	Kinerja LCD TFT	Menampilkan interface untuk dilihat user	LCD TFT menampilkan interface dengan baik
3	Kinerja <i>Keypad</i>	Menguji kesesuaian angka pada <i>keypad</i> dengan <i>output</i> di sistem	Angka yang keluar sesuai dengan <i>output</i> di sistem
4	Kinerja <i>Servo Motor</i>	Memberi perintah buka di pada sudut tertentu	<i>Servo</i> membuka ke sudut yang diinginkan

3.5.2 Pengujian Perangkat Lunak

Pengujian pada *website* akan dilakukan dengan menggunakan metode *Black Box Testing*, di mana pengujian berfokus pada interaksi antara *input* dan *output* tanpa memperhatikan proses internal yang terjadi di balik sistem. Metode ini digunakan untuk memastikan bahwa setiap fitur pada *website* berjalan sesuai dengan spesifikasi yang telah dirancang dan dapat menangani berbagai kemungkinan skenario penggunaan. Adapun beberapa skenario yang di uji antara lain ditampilkan pada Tabel 3.4, Tabel 3.5 dan Tabel 3.6.

1. Pengujian Halaman *Sign up* dan *Sign In*

Tabel 3. 4 Tabel Skenario Uji Halaman *Sign Up* dan *Sign In*

Fitur	Skenario Uji	Harapan
<i>Sign Up</i>	Pengguna mendaftar dengan semua <i>field</i> wajib (username, email, password) diisi dengan format yang valid.	Registrasi berhasil, pengguna diarahkan ke halaman <i>Sign In</i> .
	Pengguna mendaftar dengan nama atau email yang sudah terdaftar.	Sistem menampilkan pesan kesalahan
	Pengguna mendaftar dengan kata sandi (password) kurang dari 8 karakter.	Sistem menampilkan pesan kesalahan

Fitur	Skenario Uji	Harapan
<i>Sign In</i>	Pengguna <i>sign in</i> dengan email dan kata sandi (password) yang valid	Registrasi berhasil, pengguna diarahkan ke halaman <i>Sign In</i> .
	Pengguna <i>sign in</i> dengan email atau kata sandi yang salah.	Sistem menampilkan pesan kesalahan
	Pengguna mencoba <i>sign in</i> dengan <i>field</i> email kosong.	Sistem menampilkan pesan kesalahan
	Pengguna mencoba <i>sign in</i> dengan <i>field</i> kata sandi kosong.	Sistem menampilkan pesan kesalahan

2. Pengujian Halaman *Dashboard*

Tabel 3.5 Tabel Skenario Uji Halaman *Dashboard*

Fitur	Skenario Uji	Harapan
<i>Stats Cards</i>	Memverifikasi bahwa <i>card</i> menampilkan elemen data yang diharapkan	Semua elemen data yang diharapkan ditampilkan dengan benar.
<i>Modal Add Data</i>	Membuka modal " <i>Add New Package</i> " dan memverifikasi semua elemen (judul, label <i>input</i> , tombol) terlihat dengan benar.	Modal terbuka, semua elemen visual ditampilkan
	Memasukkan nama paket yang valid ke dalam <i>field</i> " <i>Package Name</i> ".	<i>Field</i> " <i>Package Name</i> " menerima <i>input</i> dengan benar.
	Memasukkan nomor resi yang valid ke dalam <i>field</i> " <i>Resi</i> ".	<i>Field</i> " <i>Resi</i> " menerima <i>input</i> dengan benar.

Fitur	Skenario Uji	Harapan
	Mencoba menyimpan tanpa mengisi <i>field</i> " <i>Package Name</i> " dan "Resi"	Sistem menampilkan pesan kesalahan
	Mengklik tombol " <i>Cancel</i> " tanpa mengisi data apapun.	Modal tertutup
	Mengklik tombol " <i>Cancel</i> " dengan <i>input field</i> terisi.	Modal tertutup
Tabel Data	Memuat halaman daftar paket dan memverifikasi semua kolom <i>header</i> (<i>PACKAGE NAME</i> , <i>RESI</i> , <i>STATUS</i> , <i>PHOTO</i> , <i>RECEIVED DATE</i> , <i>ACTION</i>) terlihat dengan benar.	Semua <i>header</i> kolom ditampilkan dengan jelas dan sesuai.
	Memverifikasi bahwa data di setiap baris tabel ditampilkan dengan benar sesuai dengan kolomnya.	Data di setiap baris ditampilkan di bawah kolom yang tepat.
	Memverifikasi bahwa nilai di setiap kolom sesuai dengan data yang diharapkan.	Nilai pada setiap kolom ditampilkan dengan benar.
	Mengklik ikon foto untuk memverifikasi fungsionalitasnya	Modal <i>preview</i> foto ditampilkan
	Mengklik ikon "Edit" / Pensil pada salah satu baris untuk memverifikasi fungsionalitasnya.	membuka modal edit dengan data paket sesuai

Fitur	Skenario Uji	Harapan
	Mengubah nilai di <i>field</i> "Package Name" dan "Resi" dengan <i>input</i> yang valid.	Field "Package Name" dan "Resi" menerima perubahan <i>input</i> dengan benar.
	Mengklik tombol "Update" dengan melakukan semua perubahan pada data.	Modal tertutup, perubahan yang disimpan, dan data yang sudah update muncul di tabel
	Mengklik tombol "Update" tanpa melakukan perubahan apapun pada data.	Modal tertutup, tidak ada perubahan yang disimpan, dan tidak ada pesan kesalahan.
	Mengklik tombol "Cancel" setelah melakukan beberapa perubahan.	Modal tertutup, perubahan yang dilakukan tidak disimpan.
	Mengklik tombol "Cancel" tanpa melakukan perubahan apapun.	Modal tertutup.
	Mengklik ikon "Delete" pada salah satu baris dan mengkonfirmasi penghapusan.	Paket berhasil dihapus dari daftar setelah konfirmasi, dan tidak lagi terlihat di tabel.
	Mengklik ikon "Delete" pada salah satu baris dan membatalkan penghapusan.	Paket tidak dihapus dari daftar setelah pembatalan.

3. Pengujian Halaman *Profile*

Tabel 3. 6 Tabel Skenario Uji Halaman *Profile*

Fitur yang diuji	Skenario Pengujian	Hasil Pengujian
<i>Profile Card</i>	Memverifikasi bahwa <i>card</i> menampilkan elemen data yang diharapkan	Semua elemen data yang ditampilkan dengan benar.
<i>Update Username</i>	Menyimpan dengan <i>field</i> "Username" dibiarkan kosong.	Sistem menampilkan pesan kesalahan.
	Mengisi <i>field</i> "Username" dengan <i>username</i> baru yang valid dan mengklik tombol "Update Username".	<i>Username</i> berhasil diperbarui.

3.5.3 Pengujian Integrasi *Software* dan *Hardware*

Selain pengujian fungsionalitas setiap komponen, pengujian sistem secara menyeluruh juga dilakukan untuk mengevaluasi kinerja keseluruhan *smartbox*. Pengujian ini bertujuan untuk memastikan bahwa sistem yang dibangun dapat berjalan tanpa memakan waktu yang panjang serta tidak menghambat proses `buka-tutup *smartbox*, hingga dokumentasi foto penerimaan paket, dapat berjalan sesuai dengan yang diharapkan. Sebagai rincian, dalam satu siklus penggunaan *smartbox* terdapat setidaknya lima tahap yang harus dilalui, mulai dari *verifikasi* resi nomor resi, mekanisme hingga *smartbox* tertutup dan status penerimaan paket ter-*update*. Penjelasan tahapan secara lengkap dapat dilihat pada Tabel 3.7.

Tabel 3.7 Tahapan Pengujian Kinerja Sistem

No	Tahapan Pengujian	Jumlah Pengujian
1	Verifikasi nomor resi di <i>database</i>	50
2	Proses pembukaan <i>box</i>	50
3	Proses penutupan <i>box</i>	50

No	Tahapan Pengujian	Jumlah Pengujian
4	Pengambilan foto dan upload oleh ESP32-CAM	50
5	Pembaruan status penerimaan di <i>database</i>	50

Berdasarkan Tabel 3.7, proses pengujian sistem dapat diuraikan melalui beberapa tahapan. Pengujian akan diawali memasukkan nomor resi menggunakan *keypad* namun tidak proses input ini tidak masuk parameter uji. Parameter uji yang pertama ketika sistem melakukan verifikasi di latar belakang untuk memeriksa ketersediaan resi tersebut di dalam basis data. Apabila resi ditemukan, mekanisme pembuka *box* akan aktif secara otomatis. Setelah *box* terbuka dan kemudian menutup, modul ESP32 Cam akan mengambil dan mengunggah foto sebagai dokumentasi penerimaan. Seluruh rangkaian proses ini diakhiri dengan pembaruan status penerimaan barang di basis data, menandai selesainya siklus pengujian.

Untuk menguji sistem verifikasi resi, dilakukan pengujian dengan menggunakan 50 nomor resi yang tidak terdaftar pada sistem. Pengujian ini bertujuan untuk memastikan bahwa *SmartBox* hanya akan terbuka apabila nomor resi yang diinputkan melalui *keypad* sesuai dengan data yang tersimpan pada basis data sistem. Proses pengujian dilakukan dengan cara memasukkan setiap nomor resi secara manual melalui *keypad*, kemudian mengamati respons sistem terhadap setiap *input*. Apabila nomor resi diterima padahal tidak terdaftar, maka sistem dinyatakan gagal dalam melakukan verifikasi. Sebaliknya, apabila nomor resi ditolak sesuai dengan harapan, maka sistem berhasil melakukan verifikasi dengan benar.

Selain itu, pengujian juga dilakukan terhadap penggunaan *owner code* sebagai kode khusus yang dimiliki pemilik untuk membuka *SmartBox*. Pada tahap ini, dilakukan percobaan dengan memasukkan *owner code* yang salah sebanyak 20 kali untuk memastikan bahwa sistem tidak akan membuka *SmartBox* apabila *owner code* yang dimasukkan tidak sesuai. Dengan demikian, sistem hanya dapat membuka kotak ketika *owner code* yang benar dimasukkan oleh pemilik.

3.6 Evaluasi Sistem

Tahapan evaluasi merupakan proses penilaian terhadap setiap langkah dalam pengembangan sistem *smartbox*. Sistem ini dilengkapi dengan validasi kode resi untuk penguncian dan pembukaan *box* secara aman, serta fitur *monitoring* berbasis *web* untuk memantau aktivitas penyimpanan paket. Evaluasi mencakup pengecekan fungsionalitas, kinerja, kegunaan, keamanan, serta identifikasi kelebihan dan kekurangan dari prototipe yang telah dirancang. Tujuan dari evaluasi ini adalah memastikan bahwa prototipe mampu beroperasi sesuai dengan desain awal, memberikan pengalaman pengguna yang optimal, serta mendukung pengembangan lebih lanjut untuk meningkatkan kualitas dan keamanan sistem.

3.7 Pelaporan

Hasil penelitian ini akan didokumentasikan dalam bentuk buku skripsi, yang merangkum seluruh temuan dan proses penelitian yang telah dilakukan. Laporan ini disusun secara sistematis dan terperinci, mencakup tahapan perancangan, implementasi, serta pengujian sistem. Dengan adanya dokumentasi ini, diharapkan dapat menjadi referensi ilmiah yang berguna bagi pembaca atau pengembang yang memiliki bidang minat serupa, serta memberikan kontribusi bagi pengembangan sistem sejenis di masa mendatang.