BAB II
LANDASAN TEORI

2.1 Tinjaun Pustaka

Sistem absensi otomatis berbasis RFID dan ESP32 telah banyak dikembangkan sebagai solusi untuk menggantikan metode manual yang masih umum digunakan di berbagai institusi pendidikan dan perusahaan. RFID memungkinkan identifikasi dan pencatatan kehadiran secara otomatis tanpa kontak fisik, sementara ESP32 berfungsi sebagai *microcontroller* yang mendukung konektivitas Wi-Fi untuk pengiriman data secara real-time ke server atau basis data (Hidayat et al., 2022).

Penelitian oleh Chandra *and* Amrizal 2023 menunjukkan bahwa sistem absensi otomatis berbasis ESP32 dan RFID dapat meningkatkan efisiensi pencatatan kehadiran siswa di sekolah, menggantikan metode manual yang dinilai kurang efektif. Sistem ini terbukti mampu membaca data RFID dengan akurat, menampilkan informasi kehadiran secara real-time, dan menyimpan data di basis data, sehingga memudahkan proses monitoring dan pelaporan (Chandra et al., 2023).

Alat absensi menggunakan RFID dibuat menggunakan ESP8266 sebagai pengontrol dan RFID *Reader* sebagai pembaca RFID *tag* atau kartu. Alat absensi dikelompokkan berdasarkan ruangan sesuai dengan jadwal perkuliahan pada STMIK Lombok, pengiriman informasi absensi dari alat absensi ke basis data server melalui jaringan wireless dan ditampilkan pada sistem yang telah dibuat, data yang ditampilkan pada sistem berupa tanggal absen, informasi kuliah, kelas, jam-masuk, jam keluar, keterangan dan menit keterlambatan (Huda et al., 2022).

Sistem presensi manual yang masih digunakan di Pondok Pesantren Daarul Huda rentan terhadap kecurangan, seperti pencatatan kehadiran atas nama orang lain, serta keterbatasan dalam pemantauan kehadiran secara langsung, sehingga menyulitkan evaluasi kedisiplinan santri. Selain itu, tidak adanya sistem pencatatan terpusat membuat pembuatan laporan kehadiran menjadi lambat dan kurang akurat, sehingga menghambat pengambilan keputusan yang efektif. Sistem absensi yang masih dilakukan secara manual seperti pencatatan di buku atau lembar absen sering kali menimbulkan berbagai kendala. Pencatatan manual rentan terhadap kesalahan, seperti hilangnya data, pencatatan waktu tidak akurat, atau kesalahan penulisan, yang

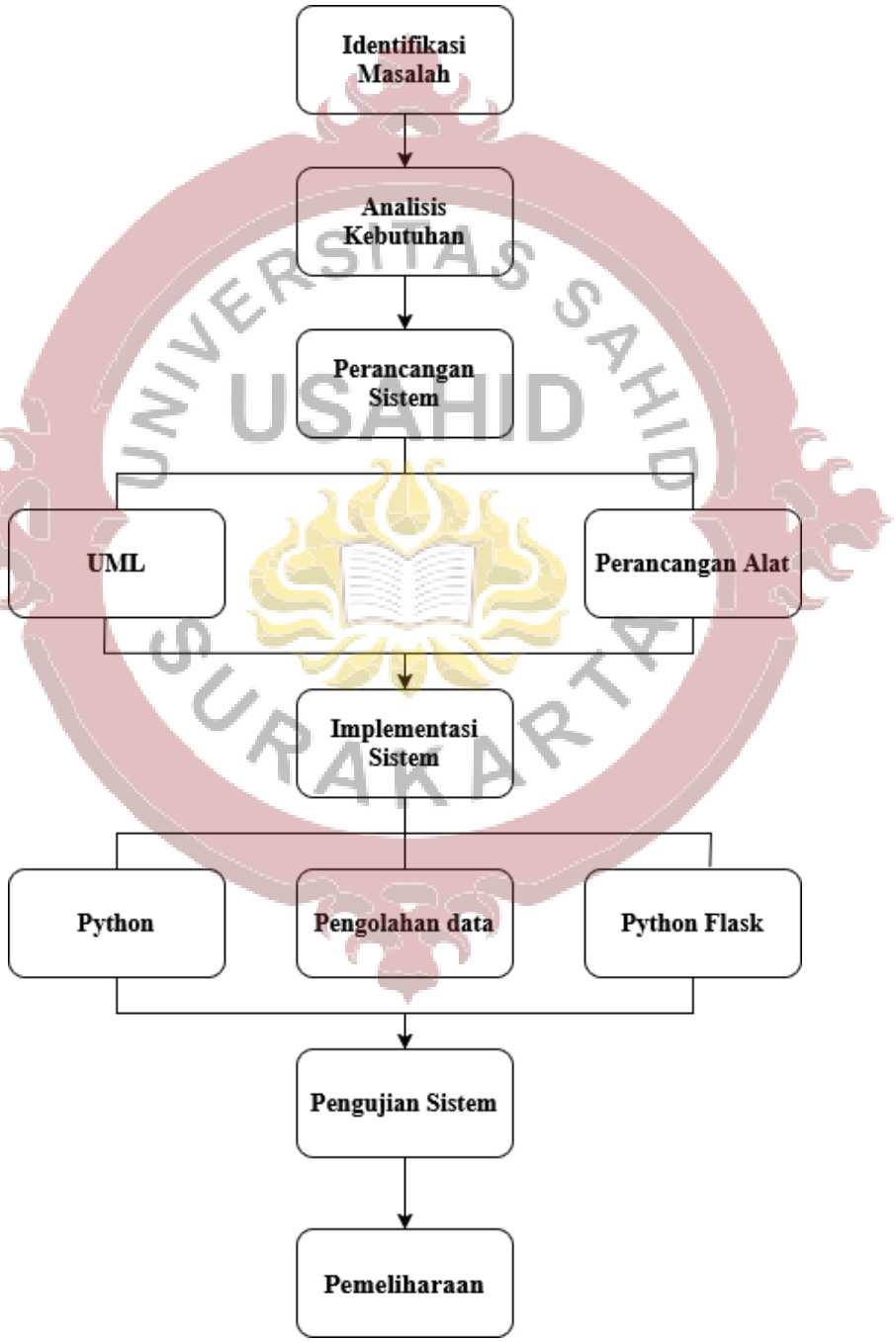
dapat memengaruhi keakuratan informasi kehadiran santri. Selain itu, proses pengolahan data absensi menjadi lambat dan memakan banyak waktu, terutama dengan jumlah santri yang terus bertambah (Kahar & Thamrin, 2025).

Perkembangan teknologi untuk pengontrolan peralatan elektronika telah sangat maju. Teknologi yang mencakup beberapa fungsi peralatan rumah tangga *(home appliance)* mendukung integrasi antar perangkat yang dimiliki oleh pengguna. Salah satu perkembangan teknologi pengontrolan perangkat elektronik adalah rancangan prototipe pengendali saklar beberapa perangkat elektronik dalam suatu rumah dengan perangkat *Handphone* melalui konsep otomasi rumah atau dikenal dengan *Smarthome.* Rumah pintar atau *smarthome* adalah sebuah tempat tinggal atau kediaman yang menghubungkan jaringan komunikasi dengan peralatan listrik yang dimungkinkan dapat dikontrol, dimonitor atau diakses dari jarak jauh. *Smarthome* juga dapat meningkatkan efisiensi, kenyamanan dan keamanan dengan menggunakan teknologi secara otomatis (Santiko, 2020).

Berdasarkan tinjauan pustaka dari beberapa penelitian terdahulu tersebut, terdapat celah penelitian (*research gap*) di mana sistem absensi berbasis IoT yang dikembangkan umumnya masih bergantung pada interaksi tombol fisik pada perangkat keras untuk menentukan jenis kegiatan, atau mewajibkan sistem autentikasi (*login admin*) yang rumit pada antarmuka web, sehingga kurang praktis jika diterapkan pada ekosistem pondok pesantren yang membutuhkan mobilitas tinggi. Kebaruan (*novelty*) dari penelitian ini terletak pada pengembangan sistem absensi santri yang dirancang secara minimalis dan otomatis tanpa tombol pilihan manual pada perangkat ESP32, dengan fokus murni pada perekaman data kode unik kartu (*Unique ID*) dan stempel waktu (*timestamp*) secara *real-time*. Selain itu, sistem ini menerapkan konsep antarmuka web dengan akses terbuka tanpa *login* khusus untuk memudahkan pengasuh maupun santri di Pondok Pesantren Daarul Huda dalam memantau data kehadiran secara transparan. Melalui pendekatan ini, kompleksitas operasional perangkat keras dapat dipangkas secara signifikan sehingga mempercepat proses adaptasi teknologi di lingkungan asrama.

2.2 Kerangka Pikiran

Kerangka pemikiran ini menjelaskan alur pemecahan masalah absensi di Ponpes Daarul Huda. Dimulai dari evaluasi kendala sistem manual, lalu dirancanglah sistem otomatis berbasis RFID dan ESP32 yang terhubung ke web. Hasilnya, kehadiran santri bisa tercatat otomatis ke *basis data* dan dipantau secara *real-time*. Alur lengkapnya dapat dilihat pada Gambar 2.1.



Gambar 2. 1 Bagan Alur Kerangka Fikiran

Uraian dari Kerangka berfikir sebagai berikut:

1. Identifikasi Masalah

   Langkah awal adalah memetakan kendala pada sistem absensi manual di Ponpes Daarul Huda, seperti risiko titip absen, data yang sering terselip, hingga lambatnya proses rekapitulasi kehadiran santri.

2. Analisis Kebutuhan

   Di sini ditentukan apa saja yang dibutuhkan untuk solusi, mulai dari perangkat keras (kartu RFID, sensor, ESP32) hingga kebutuhan perangkat lunak yang bisa mengolah data secara otomatis dan real-time.

3. Perancangan Sistem

   Tahap merancang arsitektur bagaimana alat dan aplikasi saling terhubung. Fokusnya adalah membangun jembatan komunikasi antara hardware di lapangan dengan basis data pusat melalui jaringan Wi-Fi.

4. UML (Unified Modeling Language)

   Penyusunan diagram (seperti Use Case atau Activity Diagram) untuk memvisualisasikan interaksi pengguna dengan sistem, sehingga alur kerja aplikasi web absensi menjadi lebih jelas dan terstruktur.

5. Perancangan Alat

   Proses merakit komponen fisik, yaitu mengintegrasikan RFID *Reader* dengan *microcontroller* ESP32 ke dalam satu case agar siap digunakan sebagai mesin absensi di lokasi.

6. Implementasi Sistem

   Tahap pemasangan dan penerapan sistem secara langsung. Di sini kode program dimasukkan ke dalam ESP32 agar bisa mulai membaca kartu RFID milik santri.

7. Python

   Digunakan sebagai bahasa pemrograman utama untuk membangun logika di sisi server (backend). Python bertugas mengelola alur data yang masuk dari alat sebelum disimpan.

8. Pengolahan Data

   Proses di mana setiap ID kartu yang di-tap oleh santri dicocokkan dengan data identitas di basis data, lalu status kehadirannya dicatat secara otomatis berdasarkan waktu saat itu.

9. Flask

   Penggunaan framework Flask untuk membangun aplikasi web. Flask berfungsi menyediakan antarmuka bagi pengurus pondok agar bisa memantau laporan absensi secara mudah lewat browser.

10. Pengujian Sistem

    Melakukan uji coba fungsi alat dan web dengan metode *performance testing*. Tujuannya untuk memastikan tidak ada bug dan sistem benar-benar layak digunakan oleh pengurus.

11. Pemeliharaan

    Langkah terakhir untuk menjaga performa sistem tetap stabil dalam jangka panjang, termasuk pembaruan data santri secara berkala dan pengecekan konektivitas alat.

Tabel 2. 1 Perbedaan Penelitian Terdahulu

Aspek / Sistem	**Sistem TA Mukhlis (Pondok Pesantren)**	**Sistem Chandra 2023 (PT Dharma Sentosa Marindo)**	**Sistem Ali dkk. 2021 (SMK Ma'arif NU Talang)**
Konteks & objek	Santri di pondok pesantren	Karyawan perusahaan galangan kapal	Siswa SMK Ma'arif NU Talang
Sistem absensi yang digunakan	Kartu RFID + ESP32 (*real-time*).	Kartu RFID + Web konvensional.	Kartu RFID + ESP32-Cam.
Pendekatan IoT / real-time	IoT *real-time* berbasis ESP32 untuk pengasuh pondok.	Web konvensional penerima input alat.	IoT berbasis kelas menggunakan ESP32-Cam.
Platform web / back-end	Menggunakan spesifikasi khusus *framework* Python Flask untuk *backend* dan manajemen basis data	Menggunakan sistem informasi berbasis web umum tanpa spesifikasi *framework backend* tertentu.	Berbasis web yang diintegrasikan dengan API gateway WhatsApp untuk notifikasi orang tua.
Antarmuka web	Laporan visual berbasis grafik dan tabel.	Menu standar perusahaan (CRUD karyawan)	Monitoring web sederhana
Fitur pengelolaan pengguna	Fokus pada santri dan pengelola pondok	Multi-aktor (Karyawan, Admin, HRD).	Peran untuk guru dan manajemen sekolah.
Sistem / platform pendukung	RFID RC522, ESP32, Python Flask, basis data.	RFID, ESP32, web standar.	ESP32-Cam, RFID, API WhatsApp
Metode pengembangan perangkat lunak	Model Waterfall	Metodologi Agile / Scrum	Kustom non-baku
Sistem pengujian	Tes performa (latensi/jarak) dan *Black Box*	*Black Box* fokus form input dan aktor.	Tes fungsi hardware dan kirim WhatsApp.
Sistem data & pengumpulan informasi	Wawancara, observasi, kualitatif di pondok	Studi literatur, alur kerja industri, wawancara pimpinan.	Observasi SMK, wawancara sekolah, studi hardware.
Bentuk laporan & pemantauan	Rekap digital *real-time* di web.	Laporan filterable yang bisa dicetak fisik.	Rekap web dan notifikasi instan via WhatsApp.

2.3 Landasan Teori

2.3.1 Radio Frequncy Identification (RFID)

Radio Frequency Identification (RFID) merupakan sebuah teknologi yang menggunakan metode auto-ID atau Automatic Identification. Auto-ID adalah metode pengambilan data dengan identifikasi objek secara otomatis tanpa ada ketelibatan manusia. Auto-ID bekerja secara otomatis sehingga dapat meningkatkan efisiensi dalam mengurangi kesalahan dalam memasukkan data. Pada RFID proses identifikasi oleh RFID *reader* dan RFID *tag*. RFID *tag* diletakkan pada suatu benda atau objek yang akan diidentifikasi (ID number) yang unik, sehingga tidak ada RFID *tag* yang memiliki ID number yang sama (Dzaky Naufalrochman, 2021).

Alat *Radio Frequency Identification* (RFID) merupakan peralatan dan teknologi yang menggunakan sinyal radio untuk memberikan data yang telah diidentifikasikan. *Radio Frequency Identification* (RFID) ini termasuk dalam bentuk *tag* atau kabel kecil yang dapat mengidentifikasi sebuah objek data diterima melalui sinyal radio, kemudian diterjemahkan kembali dalam bentuk angka atau informasi lainnya. Teknologi ini mampu mengidentifikasi berbagai objek secara bersamaan tanpa diperlukan kontak langsung. Sensor *Radio Frequency Identification* (RFID) adalah sensor yang mengidentifikasikan suatu barang dengan menggunakan frekuensi radio. Sensor ini terdiri dari dua bagian penting: *transceiver (reader)* dan *transponder (tag)*. Setiap *tag* tersimpan data yang berbeda. Data tersebut merupakan data identitas *tag*. *Reader* akan membaca data dari *Tag* dengan perantara gelombang radio. Pada *reader* biasanya berhubungan dengan suatu mikrokontroller. Mikrokontroller ini berfungsi untuk mengolah data yang didapat *reader* (Riskayani et al., 2023).

2.3.2 ESP32

ESP32 adalah *microcontroller* yang dikembangkan oleh Espressif Systems, yang dilengkapi dengan kemampuan Wi-Fi dan Bluetooth. ESP32 banyak digunakan dalam pengembangan sistem IoT karena kemampuannya untuk terhubung ke jaringan dan mengirimkan data secara real-time. Dalam konteks sistem absensi, ESP32 berfungsi sebagai pengolah data dari RFID dan mengirimkannya ke server atau basis data untuk penyimpanan (Kurnialensya & Saputra, 2023).

2.3.3 Internet Of Things (IoT)

Internet of Things (IoT) adalah konsep di mana perangkat fisik terhubung ke internet dan dapat saling berkomunikasi untuk mengumpulkan dan bertukar data. Dalam sistem absensi berbasis RFID dan ESP32, IoT memungkinkan pengiriman data kehadiran secara real-time ke server atau cloud, sehingga memudahkan pengelola dalam memonitor dan menganalisis data kehadiran. IoT juga mendukung integrasi dengan aplikasi web atau mobile untuk akses data yang lebih fleksibel (Nurendro Ilhamsyah et al., 2024).

2.3.4 HTML (HyperText Markup Language)

HyperText Markup Language (HTML) adalah bahasa penanda standar (*markup language*) yang digunakan secara luas untuk membuat, menyusun, dan menstrukturkan komponen-component dasar sebuah halaman web di sisi *front-end*. HTML berfungsi sebagai fondasi utama untuk meletakkan elemen-elemen struktural dokumen web, seperti teks, judul, paragraf, daftar (*list*), tombol, formulir (*form*), tabel, gambar, serta media lainnya. Sebagai kerangka dasar statis, HTML bekerja dengan menggunakan sistem penandaan berupa *Tags* (seperti <html>, <body>, <table()) untuk menginstruksikan peramban web (*browser*) tentang bagaimana sebuah konten harus diorganisasikan dan ditampilkan kepada pengguna. Penggunaan HTML standar memastikan bahwa dokumen web yang dibangun dapat dibaca dan diakses secara konsisten di berbagai platform perangkat, baik melalui komputer desktop maupun *smartphone*. Selain itu, HTML memungkinkan integrasi dengan elemen-elemen interaktif, seperti tombol, formulir, dan tabel, yang mendukung pengelola dalam mengakses dan mengelola data absensi dengan cepat dan akurat. Penggunaan HTML dalam sistem ini juga memastikan bahwa antarmuka dapat diakses melalui berbagai perangkat, seperti komputer dan smartphone, sehingga meningkatkan fleksibilitas dan kemudahan penggunaan sistem absensi (Teguh & Elizabeth, 2020).

2.3.5 CSS (Cascading Style Sheets)

Cascading Style Sheets (CSS) adalah bahasa desain lembar gaya (*style sheet language*) yang digunakan untuk mengatur, mengontrol, dan memformat tampilan visual dari suatu dokumen yang ditulis dengan bahasa penanda seperti HTML. Jika

HTML berfungsi untuk membangun struktur dan kerangka dasar halaman web, maka CSS berperan di sisi *front-end* untuk menentukan estetika, presentasi, dan tata letak (*layout*) elemen-elemen tersebut. Melalui CSS, pengembang web dapat memisahkan konten logis dari desain visualnya, sehingga pengaturan warna, jenis huruf (*font*), jarak antar-elemen (*margin* dan *padding*), hingga animasi dapat dikelola secara terpusat dan konsisten. Selain itu, CSS memiliki fitur *responsive web design* (seperti *media queries*) yang memungkinkan sebuah halaman web beradaptasi secara dinamis mengikuti berbagai ukuran resolusi layar perangkat yang mengoperasikannya, baik melalui komputer desktop, tablet, maupun perangkat *mobile* (Sari & Irfani, 2022).

2.3.6 Python Flask

Python Flask adalah sebuah kerangka kerja pengembangan web (*web framework*) berbasis bahasa pemrograman Python yang dirancang khusus untuk membangun komponen sisi belakang (*back-end*) suatu aplikasi. Flask dikategorikan sebagai *micro-framework* karena sifatnya yang minimalis, ringan, dan tidak bergantung pada pustaka (*library*) atau alat bantu pihak ketiga tertentu untuk dapat berfungsi dasar. Meskipun bersifat *micro*, Flask menyediakan fleksibilitas yang sangat tinggi bagi pengembang untuk mengelola logika bisnis aplikasi, mengatur rute URL (*routing*), menangani permintaan HTTP, serta menjembatani komunikasi data antara antarmuka pengguna (*front-end*) dengan server basis data melalui penyediaan API (*Application Programming Interface*). Kerangka kerja ini juga mendukung skalabilitas pengembangan melalui integrasi berbagai modul tambahan (*extensions*) secara mandiri, seperti sistem autentikasi keamanan, pengelolaan sesi, hingga konektivitas ke berbagai jenis DBMS (*Basis data Management System*) baik SQL maupun NoSQL (Sholeh et al., 2022).

2.3.7 Pengujian Sistem

Pada penelitian ini, *performance testing* digunakan sebagai metode pengujian untuk mengukur efisiensi transmisi data dari perangkat ESP32 ke *server* serta responsivitas sistem dalam mengolah data absensi secara *real-time*. Pengujian sistem dilakukan untuk menjamin kualitas dan reliabilitas perangkat. Dalam konteks IoT berfokus pada *performance testing* melalui tiga indikator utama, yaitu kecepatan

(*speed*) untuk mengukur waktu respon transmisi data dari ESP32 ke *server*, akurasi (*accuracy*) untuk memastikan tingkat kesuksesan pengiriman data (*success rate*) tanpa adanya paket yang hilang, serta kinerja (*stability*) untuk menguji ketahanan sistem dalam menangani trafik data dari pembacaan RFID secara berkelanjutan (Khatami et al., 2019).

Pada penelitian ini, *performance testing* digunakan sebagai metode pengujian teknis untuk menjamin kualitas sistem, yang berfokus pada pengukuran waktu respon transmisi data dari ESP32 ke *server* (kecepatan), tingkat keberhasilan pengiriman UID RFID tanpa kehilangan data (akurasi), serta stabilitas *server* Flask dalam menangani beban trafik absensi (kinerja). Selain itu, wawancara mendalam dilakukan dengan 5 pengguna kunci (2 pengelola pondok, 3 santri senior) untuk mendapatkan *feedback* kualitatif terkait proses absensi RFID secara keseluruhan, seperti kemudahan penggunaan dan kendala operasional di lapangan.

2.3.8 HTTP Request dan REST API

RESTful API merupakan gaya arsitektur layanan web yang memanfaatkan protokol HTTP dan operasi seperti GET dan POST untuk pertukaran data antara *client* dan server, dan banyak digunakan pada sistem IoT karena ringan dan mudah diintegrasikan dengan perangkat embedded. Dalam konteks sistem absensi santri, ESP32 berperan sebagai *client* IoT yang mengirimkan data hasil pembacaan RFID ke server melalui HTTP request menuju endpoint yang disediakan oleh aplikasi Python Flask, sehingga REST API menjadi jembatan komunikasi data antara perangkat IoT dan sistem web (Maurya, 2021).

2.3.9 Buzzer

RESTful API Buzzer dalam sistem absensi berbasis RFID di Pondok Pesantren Daarul Huda berfungsi sebagai perangkat *output* audio yang memberikan umpan balik fisik secara *real-time* kepada santri saat melakukan proses pemindaian kartu. Buzzer berperan krusial dalam menandakan status keberhasilan atau kegagalan transaksi data antara perangkat ESP32 dan *basis data*. Ketika santri melakukan *tap* kartu dan data berhasil divalidasi oleh sistem, buzzer akan mengeluarkan bunyi singkat sebagai indikator bahwa kehadiran telah tercatat. Sebaliknya, jika kartu tidak terdaftar, kartu salah, atau

terjadi kendala koneksi pada ESP32, buzzer akan mengeluarkan pola bunyi yang berbeda sebagai peringatan terjadinya kesalahan (*error*). Implementasi buzzer ini sangat efektif dalam memitigasi antrean saat jam keberangkatan atau kepulangan santri, karena memberikan konfirmasi instan tanpa mengharuskan santri melihat layar monitor. Selain sebagai alat bantu operasional, suara buzzer juga berfungsi sebagai indikator keamanan teknis untuk mendeteksi adanya kartu yang tidak dikenal atau percobaan akses yang tidak sah, sehingga meningkatkan integritas data pada sistem absensi (Pradiptayasa Agustana, 2025).

2.4.0 Unified Modeling Language (UML)

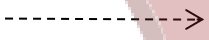
Unified Modeling Language (UML) merupakan sebuah bahasa standar yang digunakan untuk melakukan visualisasi, spesifikasi, konstruksi, serta pendokumentasian artefak dari sistem perangkat lunak. Dalam penelitian ini, UML digunakan pada tahap perancangan proses untuk memodelkan sistem absensi santri agar alur kerja antara perangkat keras (ESP32 & RFID) dan perangkat lunak (Web Flask) dapat terpetakan dengan jelas (Didit Suhartono et al., 2019).

Penggunaan UML dalam memodelkan sistem ini mencakup serangkaian diagram visual yang memiliki fungsi spesifik untuk menggambarkan arsitektur sistem dari berbagai sudut pandang. Pemanfaatan diagram-diagram tersebut bertujuan untuk menguraikan interaksi pengguna (*actor*), alur kerja fungsionalitas (*workflow*), serta struktur logis pertukaran data secara terstruktur. Adapun jenis-jenis diagram UML yang digunakan dalam perancangan sistem ini meliputi *Use Case Diagram* untuk mendefinisikan batas sistem dan hak akses pengguna, *Activity Diagram* untuk menggambarkan alur aktivitas sistem secara berurutan saat pemindaian kartu RFID hingga pemrosesan data, *Sequence Diagram* untuk menjelaskan alur pertukaran pesan antar-objek atau antarmuka secara *real-time*, serta *Class Diagram* untuk memetakan struktur basis data dan hubungan antar-entitas dalam sistem. Pembagian diagram yang komprehensif ini memastikan bahwa seluruh tahapan transisi data dari perangkat keras ke peladen (*server*) dapat diimplementasikan secara presisi dan minim potensi *error*:

a. Use Case Diagram

Use Case Diagram Digunakan untuk menggambarkan interaksi antara aktor (pengguna) dengan sistem. Dalam sistem absensi di Ponpes Daarul Huda, aktor yang terlibat biasanya meliputi Administrator (pengurus pondok) dan Santri. Diagram ini menjelaskan fungsi-fungsi apa saja yang bisa dilakukan oleh setiap aktor terhadap sistem.

Tabel 2. 2 Use Case Diagram

Simbol	Nama	Keterangan
 Actor	Aktor	Mewakili peran orang, sistem yang lain,atau alat Ketika berkomunikasi
	Use Case	Abstraksi dan interaksi antara sistem dan aktor.
	Association	Abstraksi dari penghubung antara aktor dengan use case.
	Generalisasi	Menunjukkan spesialisasi aktor untuk dapat berpartisipasi dengan use case.
	Anak panah include	Menunjukkan bahwa suatu use case seluruhnya merupakan fungsionalitas dari use case lainnya.
	Anak panah extend	Menunjukkan bahwa suatu use case seluruhnya merupakan fungsionalitas dari use case lainnya jika suatu kondisi terpenuhi.

b. Activity Diagram

Activity Diagram Digunakan untuk memodelkan alur kerja (*workflow*) atau aktivitas bisnis dari sebuah sistem. Diagram ini menunjukkan urutan aktivitas mulai dari santri melakukan *tapping* kartu RFID pada alat ESP32 hingga data tersebut diproses dan ditampilkan di aplikasi web. Simbol Activity Diagram bisa dilihat pada tabel 2.3.

Tabel 2. 3 Activity Diagram

Simbol	Nama	Keterangan
	Status Awal	Sebuah diagram aktivitas memiliki sebuah status awal.
	Aktivitas	Aktivitas yang dilakukan sistem, aktivitasi biasanya diawali dengan kata kerja
	Percabangan/ Decision	Percabangan dimana ada pilihan aktivitas yang lebih dari satu.
	Penggabungan/ Join	Penggabungan dimana yang mana lebih dari satu aktivitas digabungkan jadi satu.
	Status Akhir	Status akhir yang dilakukan sistem, sebuah diagaram aktivitas memiliki sebuah status akhir.

c. Class Diagram

Class Diagram Digunakan untuk menggambarkan struktur sistem dari sisi pendefinisian kelas-kelas yang akan membentuk basis data, termasuk atribut dan hubungan (*relationship*) antar tabel seperti data santri, data kartu RFID, dan log riwayat kehadiran. Simbol Class Diagram bisa dilihat pada Tabel 2.4

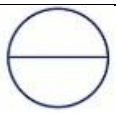
Tabel 2. 4 Simbol Class Diagram

Simbol	Nama	Keterangan
	Generalization	Hubungan dimana objek anak (descendent) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (ancestor).
	Nary Association	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
	Class	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
	Collaboration	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
	Realization	Operasi yang benar-benar dilakukan oleh suatu objek.
	Dependency	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (independent) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri.
	Association	Apa yang menghubungkan antara objek satu dengan objek lainnya.

d. Sequence Diagram

Sequence Diagram Digunakan untuk menjelaskan interaksi antar objek dalam sistem yang disusun berdasarkan urutan waktu. Diagram ini sangat penting untuk melihat bagaimana pesan (*message*) dikirimkan dari perangkat ESP32 menuju *basis data* melalui API Flask. Simbol Class Diagram bisa dilihat pada Tabel 2.5

Tabel 2. 5 Simbol Sequence Diagram

Simbol	**Nama**	**Keterangan**
	Entity Class	Gambaran sistem sebagi landasan dalam Menyusun basis data
	Boundary Class	Menangani komunikasi antar lingkungan sistem
	Control Class	Bertanggung jawab terhadap kelas-kelas terhadap objek yang berisi logika.
	Recursive	Pesan untuk dirinya.
	Activation	Mewakili proses durasi aktivasi sebuah operasi.
	Life Line	Komponen yang digambarkan garis putus-putus dengan objek.