

BAB II

TINJAUAN PUSTAKA

2.1 Teori Terkait

Penelitian yang dilakukan oleh Alex Prasetio Sitepu, Nenna Irsa Syahputri, dan Mufida Khairani (2024) merancang sebuah sistem ketersediaan tempat parkir secara *online* berbasis *web* dengan memanfaatkan *Arduino IDE* dan *Arduino Cloud*. Sistem ini menampilkan informasi kapasitas, ruang terisi, serta slot kosong pada empat lantai parkir. Data dari perangkat keras dikirimkan ke *Google Sheet* menggunakan metode *webhook* sebelum diolah dan ditampilkan melalui *website* berbasis *WordPress*. Walaupun mampu memberikan informasi secara daring, penelitian ini masih mengandalkan alur pengolahan data melalui *Google Sheet* yang berpotensi mengalami keterlambatan pembaruan dan duplikasi data, sehingga integrasi *real-time* belum sepenuhnya optimal. Selain itu, tampilan *website* menggunakan *WordPress* membuat fleksibilitas pengembangan fitur monitoring menjadi terbatas dibandingkan *framework* berbasis *MVC*. Penelitian ini menjadi salah satu acuan penting dalam pengembangan sistem monitoring parkir berbasis *web* yang lebih efisien dan *responsive* [5].

Penelitian yang dilakukan oleh Anisya Caty Praniffa dkk. merancang sebuah sistem informasi parkir berbasis *web* menggunakan metode *Extreme Programming (XP)*. Penelitian ini berfokus pada permasalahan parkir di lingkungan kampus yang belum tertata dengan baik, seperti tidak adanya pencatatan kendaraan secara terkontrol, sulitnya mengetahui ketersediaan

lahan parkir, dan tingginya risiko kehilangan kendaraan. Sistem yang dikembangkan menyediakan fitur pencatatan kendaraan masuk dan keluar, tampilan informasi ketersediaan slot, serta pemantauan melalui *website*. Meskipun mampu meningkatkan ketertiban parkir di area kampus, sistem ini belum menekankan aspek integrasi dengan perangkat *IoT* secara *real-time* karena masih mengandalkan proses *input* manual seperti pemindaian *barcode* tiket dan pencatatan oleh petugas. Penelitian ini relevan sebagai pembanding untuk pengembangan sistem monitoring parkir berbasis *website* yang lebih terotomatisasi dan responsif [6].

Penelitian oleh Abd. Rahman dan Handrie Noprisson (2025), menjelaskan pengembangan sebuah aplikasi *dashboard* parkir berbasis *web* yang mampu mengintegrasikan data transaksi, keanggotaan, monitoring slot parkir, serta pelaporan pendapatan secara *real-time*. Aplikasi ini dikembangkan menggunakan *framework CodeIgniter 3* serta dilengkapi dengan fitur *login admin* dan operator, pengelolaan data member, *input* transaksi kendaraan, dan *dashboard statistik* untuk menampilkan ringkasan data harian. Namun penelitian ini masih berfokus pada manajemen parkir secara umum dan belum secara khusus membahas sistem deteksi otomatis untuk ketersediaan slot parkir menggunakan sensor *IoT*, sehingga perlu dilakukan pengembangan lebih lanjut agar monitoring slot parkir dapat berjalan secara mandiri tanpa *input* manual [7].

Penelitian oleh Sandy Yudha, Yuri Rahmanto, dan Styawati (2024) membahas implementasi teknologi berbasis *web* yang dikombinasikan dengan

sensor *IR* dan mikrokontroler *ESP8266* untuk memberikan informasi ketersediaan lahan parkir secara *real-time*. Sistem yang dikembangkan mampu menampilkan kondisi slot parkir melalui *website* serta mengirimkan data dari sensor ke *server* secara otomatis. Hasil penelitian menunjukkan adanya peningkatan efisiensi waktu pencarian lahan parkir dari 29 detik menjadi 16 detik setelah pengguna mengetahui slot tujuan. Namun penelitian ini masih terbatas dan tampilan *website* sederhana, sehingga belum memiliki manajemen data atau fitur tambahan seperti pengelolaan pengguna maupun integrasi *framework* pengembangan *web modern* [8].

Penelitian oleh Silvia Permata Perangin-angin dkk. (2023) mengembangkan sebuah sistem parkir kampus berbasis *website* untuk mengatasi permasalahan parkir manual yang masih menggunakan kartu dan pencatatan buku tulis. Sistem yang dibangun menerapkan metode prototipe dan memanfaatkan teknologi *web* seperti *HTML*, *CSS*, dan *JavaScript* untuk menampilkan informasi parkir secara lebih efisien. Hasil penelitian menunjukkan bahwa sistem mampu mengurangi waktu tunggu di area masuk dan keluar kampus, serta memberikan notifikasi kepada pengendara setelah melakukan pemindaian *barcode* ketika keluar. Namun, penelitian ini belum dilengkapi dengan fitur pemantauan ketersediaan slot secara *real-time*, sehingga pengguna masih belum dapat mengetahui kapasitas parkir sebelum memasuki area kampus. Selain itu, sistem belum menggunakan *framework* pengembangan *web* seperti *CodeIgniter* yang dapat meningkatkan struktur *website* dan kemudahan pengelolaan data. Dengan demikian, penelitian

tersebut memberikan gambaran awal terkait digitalisasi parkir, namun masih memerlukan pengembangan lebih lanjut agar dapat menyediakan informasi ketersediaan slot parkir secara akurat dan terintegrasi dalam satu sistem monitoring yang komprehensif [9].

Penelitian yang dilakukan oleh Steffanuel Pranatalie Krispyanto dkk. (2025) merancang sistem parkir cerdas *U-Park* berbasis *QR Code* yang terintegrasi dengan Kartu Tanda Mahasiswa (KTM) di lingkungan kampus UPN “Veteran” Jawa Timur. Penelitian ini memfokuskan pengembangan autentikasi digital menggunakan *QR Code* dinamis untuk menggantikan proses verifikasi manual dengan STNK, sehingga meningkatkan efisiensi dan keamanan akses kendaraan. Sistem yang dibangun mampu melakukan pencatatan aktivitas masuk dan keluar secara otomatis serta menghasilkan kecepatan verifikasi rata-rata kurang dari tiga detik. Meskipun demikian, penelitian ini masih berfokus pada pengelolaan akses parkir dan belum menyediakan fitur monitoring ketersediaan slot parkir secara *real-time*, sehingga pengguna belum dapat mengetahui lokasi parkir kosong sebelum memasuki area kampus. penelitian ini belum memanfaatkan teknologi IoT atau sensor pendeteksi kendaraan untuk memberikan informasi ketersediaan slot parkir secara otomatis, sehingga masih terdapat peluang pengembangan sistem *monitoring* parkir yang lebih komprehensif [10].

2.2 Landasan Teori

Adapun landasan teori pada “Sistem informasi ketersediaan slot parkir berbasis *website* menggunakan *framework codeigniter 3*” berpacuan dengan berbagai penelitian terdahulu di antaranya :

2.2.1 Internet Of Things

Internet of Things (IoT) merupakan konsep jaringan perangkat fisik yang dapat mengumpulkan dan bertukar data melalui internet. perangkat *IoT* mendeteksi keadaan slot parkir dan mengirimkan data tersebut ke *server*. Data yang telah diterima kemudian ditampilkan pada *website* sebagai informasi monitoring. Integrasi ini memungkinkan administrator dan pengguna memperoleh pembaruan status slot parkir secara cepat, akurat, dan *real-time* [11].

2.2.2 Framework CodeIgniter 3

CodeIgniter 3 adalah *framework PHP* yang menggunakan arsitektur *Model-View-Controller* (MVC). *Framework* ini dikenal ringan, cepat, dan mudah digunakan, sehingga sangat cocok untuk pengembangan aplikasi berbasis *web* yang membutuhkan performa tinggi seperti sistem monitoring *smart parking*. Struktur MVC memungkinkan pengelolaan kode yang lebih terorganisir dan memudahkan pemisahan antara logika aplikasi, tampilan, dan data. *CodeIgniter 3* juga mendukung integrasi *API*, sehingga memudahkan pengambilan data dari *server* atau perangkat *IoT* [12].



Gambar 2. 1 CodeIgniter

2.2.3 Laragon

Laragon merupakan lingkungan pengembangan lokal yang menyediakan paket lengkap untuk menjalankan aplikasi *web*, termasuk *web server*, *PHP*, dan *MySQL*. *Laragon* dipilih karena stabil, ringan, dan mendukung konfigurasi otomatis untuk berbagai *framework*, termasuk *CodeIgniter* 3. Lingkungan ini mempermudah proses pengembangan *website* secara lokal sebelum diimplementasikan ke *server* yang sebenarnya [13].



Gambar 2. 1 Laragon Server

2.2.4 phpMyAdmin

MySQL adalah sistem manajemen basis data relasional yang berfungsi menyimpan dan mengelola seluruh data yang digunakan

dalam sistem. *phpMyAdmin* digunakan sebagai antarmuka berbasis *web* untuk memudahkan proses administrasi *database*. Dalam sistem *smart parking*, *database* menyimpan berbagai informasi seperti data pengguna, status slot parkir, riwayat masuk dan keluar kendaraan, hasil *scan QR Code*, serta laporan transaksi. Semua data tersebut diolah dan ditampilkan di *website* untuk keperluan *monitoring* [14].



Gambar 2. 2 phpMyAdmin

2.2.5 PHP

PHP adalah bahasa pemrograman *server-side* yang digunakan untuk membangun aplikasi *web* dinamis. *PHP* berperan untuk memproses data yang dikirim oleh perangkat *IoT*, mengelola logika aplikasi, serta menghubungkan antara *website* dan *database*. *PHP* memungkinkan data seperti status slot, riwayat kendaraan, hingga laporan transaksi dapat diproses dan ditampilkan secara *real-time* kepada pengguna [15].



Gambar 2. 3 PHP Bahasa Pemrograman

2.2.6 Bootstrap

Bootstrap adalah *framework* CSS yang digunakan untuk mempercepat proses pengembangan tampilan antarmuka *website*. *Framework* ini menyediakan kumpulan komponen siap pakai seperti *grid system*, tombol, kartu, *form*, navigasi, serta berbagai elemen *UI* lainnya yang responsif dan mudah dikustomisasi. Penggunaan *Bootstrap* memungkinkan tampilan *website* tetap rapi dan konsisten pada berbagai ukuran layar, baik desktop maupun perangkat *mobile*. Dalam sistem informasi monitoring, *Bootstrap* digunakan untuk membangun antarmuka yang intuitif, responsif, serta mudah digunakan oleh pengguna maupun administrator saat mengakses informasi slot parkir, riwayat kendaraan, dan data pendukung lainnya [16].



Gambar 2. 4 Bootstrap

2.2.7 UML (Unified Modeling Language)

Unified Modeling Language adalah bahasa standar yang digunakan untuk membuat diagram atau denah perangkat lunak. Memungkinkan untuk memvisualisasikan, mendefinisikan, membangun, dan

mendokumentasikan sistem perangkat lunak. Analogi yang sering digunakan adalah seperti seorang arsitek bangunan yang membuat denah untuk perusahaan konstruksi; begitu pula seorang arsitek perangkat lunak membuat diagram agar pengembang dapat membangun sistem dengan lebih terstruktur. Pemahaman terhadap notasi dan kosakata serta mempermudah seseorang dalam memahami sistem, serta menjelaskan desain kepada pihak lain.

Unified Modeling Language sendiri termasuk salah satu metode pemodelan visual yang banyak digunakan dalam perancangan perangkat lunak berbasis objek. berfungsi sebagai standar penulisan atau semacam blueprint, yang di dalamnya mencakup bisnis proses, penulisan kelas, dan berbagai elemen sistem lainnya dalam bahasa yang spesifik. Terdapat beragam diagram yang sering digunakan dalam pengembangan sistem untuk menggambarkan struktur, perilaku, dan interaksi antar elemen sistem.

1. Use Case Diagram

Adalah gambaran dari fungsionalitas yang diharapkan dari sebuah sistem dan merepresentasikan sebuah interaksi antara aktor dan sistem.

Didalam *Use Case Diagram* terdapat (7 tujuh aktor) yang merupakan sebuah gambaran entitas dari manusia atau sebuah sistem yang melakukan pekerjaan di sistem.

Tabel 2. 1 *Use Case Diagram*

No.	Gambar	Simbol	Keterangan
1.		Aktor	Mewakili peran seorang <i>system</i> yang lain, atau alat ketika berkomunikasi dengan <i>Use Case</i>
		<i>Use Case</i>	Mwakiili Abstraksi dan interaksi antara sebuah <i>System</i> dan <i>Actor</i>

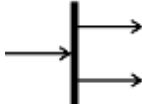
3.		<i>Directed Association</i>	Abstraksi penghubung antar <i>Actor</i> dengan <i>Use Case</i>
4.		<i>Generalisasi</i>	Menunjukkan spesialisasi actor untuk dapat berpartisipasi dengan <i>Use Case</i>
5.		<i>Include</i>	Menunjukkan bahwa suatu <i>Use Case</i> seluruhnya merupakan fungsionalitas dari <i>Use Case</i> lainnya
6.		<i>Extend</i>	Menunjukkan bahwa suatu <i>Use Case</i> merupakan tambahan fungsional dari <i>Use Case</i> lainnya jika suatu kondisi terpenuhi.

7		<i>Association</i>	Menunjukkan antara aktor dan <i>use case</i> , digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengindikasikan data..
---	---	--------------------	--

2. Activity Diagram

Menggambarkan alur kerja dari suatu proses dan urutan aktivitas, yang dimodelkan sebagai alur kerja dari satu aktivitas ke aktivitas berikutnya atau dari satu aktivitas ke aktivitas lainnya.

Tabel 2. 2 Activity Diagram

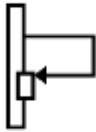
No.	Gambar	Simbol	Keterangan
1.		<i>Initial node</i>	Merupakan awal aktivitas yang akan dibuat
2.		<i>Activity</i>	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi
3.		<i>Decision</i>	Menggambarkan suatu keputusan atau tindakan yang harus diambil pada kondisi tertentu.
4.		<i>Fork node</i>	Satu aliran pada tahap yang berubah menjadi beberapa aliran.
5.		<i>Fork Join node</i>	Digunakan untuk menunjukan kegiatan yang dilakukan secara paralel atau untuk menggabungkan dua kegiatan paralel menjadi satu

6.		<i>Activity Final node</i>	Merupakan akhir aktivitas yang telah dibuat.
----	---	----------------------------	--

3. Sequence Diagram

Menggambarkan interaksi antar objek didalam dan disekitar sistem yang berupa *message* yang digambarkan terhadap waktu.

Tabel 2. 3 Sequence Diagram

No.	Gambar	Simbol	Keterangan
1.		<i>Entity Class</i>	Menggambarkan awal dari sebuah sistem, menjadi dasar penyusunan basisdata.
2.		<i>Boundary Class</i>	Menggambarkan komunikasi antara sistem seperti <i>Form</i>
3.		<i>Control Class</i>	Penghubung antara <i>boundary</i> dengan <i>table</i> .
4.		<i>Recursive</i>	Mengirimkan pesan yang akan dikirim ke dirinya sendiri

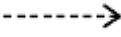
5.		<i>Activation</i>	Menghubungkan proses durasi aktivasi sebuah proses
6.		<i>Life Line</i>	Menghubungkan tiap-tiap objek yang mana terdapat <i>activation</i>
7.		<i>Message</i>	Sebagai pengirim pesan ke tiap-tiap <i>Class</i>

4. Class Diagram

Mengambaran struktur dan deskripsi dari *class*, *package* dan objek yang saling berhubungan seperti diantaranya pewarisan, asosiasi dan lainnya

Tabel 2. 4 *Class Diagram*

No.	Gambar	Simbol	Keterangan
1.		<i>Class</i>	Menggambarkan kumpulan objek-objek dengan atribut dan operasi yang sama
2.		<i>Nary Association</i>	Sebagai upaya menghindari asosiasi dengan lebih dari 2 objek

3.		<i>Collaboration</i>	Urutan aksi-aksi yang ditampilkan oleh sistem yang menghasilkan suatu hasil yang tertukar bagi suatu <i>actor</i>
4.		<i>Generalization</i>	Menggambarkan dimana hubungan objek <i>descendent</i> berperilaku dan struktur data objek yang di atasnya objek induk <i>ancestor</i>
5.		<i>Realization</i>	Operasi yang benar-benar dilakukan oleh suatu objek.
6.		<i>Dependency</i>	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri akan mempengaruhi elemen yang tidak mandiri.