BAB II
LANDASAN TEORI

2.1 Penelitian Terdahulu

Penelitian terdahulu merupakan penelitian yang digunakan untuk membandingkan penelitian yang akan dibuat dengan penelitian yang sebelumnya pernah ada. Penelitian terdahulu teriraikan sebagai berikut :

a. Penelitian yang dilakukan oleh Nabyla dan Sigita adalah penelitian yang membangun aplikasi sistem pendaftaran online pada rumah sakit. Sistem ini dibangun berbasiskan smartphone Android dengan metode pengembangan sistem yang digunakan adalah DSRM (*Design Science Research Methodology*). Sistem yang dibangun memiliki fitur registrasi pendaftaran layanan dan terdapat notifikasi antrian. Sistem ini dapat memberikan informasi nomer antrian dan perkiraan waktu kapan nomer antrian dilayani sehingga membuat proses antrian menjadi efektif karena memiliki fitur notifikasi (Nabyla and Sigitta, 2019).
b. Penelitian yang dilakukan oleh Yunianto adalah penelitian yang membangun sistem informasi antrian layanan PT. Multi Informatika Solusindo berbasis kiosk. Kiosk sendiri merupakan mesin atau terminal komputer (aplikasi *desktop*) yang dibuat untuk tujuan menyediakan informasi atau menyediakan pelayanan yang ada pada sebuah bisnis atau perusahaan yang biasanya diletakkan di bagian depan ruang pelayanan informasi. Kiosk dibangun dengan menggunakan Visual Basic .Net dan *database* MySQL. Peneliti menerapkan metode pengembangan sistem SDLC (*System Development Life Cycle*). *Tools* yang digunakan untuk merancang sistem adalah DFD dan ERD. Sistem antrian sendiri menerapkan metode FCFS (*First Come First Serve*). Adanya sistem ini dapat membantu kegiatan operasional perusahaan dalam memasukan dan memproses data agar bisa lebih efektif dan efisien (Yunianto, 2019).
c. Penelitian yang dilakukan oleh Rahayu dan Nurul adalah penelitian yang membangun sistem informasi pelayanan kesehatan pada Klinik Smart Medica berbasiskan *web*. Sistem dibangun dengan menggunakan bahasa pemograman

PHP, *database MySQL*, dan metode pengembangan sistem *waterfall*. Sistem ini berisi informasi jadwal praktik dokter sehingga user dapat mengetahui dokter-dokter yang sedang praktik pada hari itu maupun dokter yang diinginkan dan juga informasi-informasi lainnya. Dengan adanya informasi jadwal dokter yang akurat maka pasien dapat memilih waktu yang sesuai dengan jadwal dokter yang menangani penyakitnya dan juga tidak perlu menunggu lama di karenakan pada *website* tersebut sudah ada jam praktek doter (Amalia and Huda, 2020).

Tabel 2.1 Penelitian Terdahulu

Nama	**Judul**	**Fitur**	**Metode Analisis**	**Hasil**
Nabyla dan Sigita (2019)	Desain Aplikasi Sistem Pendaftaran Online Menggunakan Smartphone Untuk Meningkatkan Mutu Pelayanan pada Rumah Sakit	Berbasiskan smartphone Android dengan fitur registrasi pendaftaran layanan dan notifikasi nomor antrian	Metode pengembangan sistem DSRM (*Design Science Research Methodology*)	Sistem ini dapat memberikan informasi nomer antrian dan perkiraan waktu kapan nomer antrian dilayani sehingga membuat proses antrian menjadi efektif karena memiliki fitur notifikasi
Yunanto (2019)	Pengembangan Sistem Informasi Antrian Layanan Berbasis Kiosk di PT. Multi	Berbasiskan kiosk (desktop) dengan fitur pendaftarandan pencetakan nomor antrian	- Metode pengembangan sistem SDLC - Tools DFD dan ERD - Algoritma	Membantu kegiatan operasional perusahaan dalam memasukan dan memproses data

	Informatika Solusindo	dan laporan	antrian FCFS	agar bisa lebih efektif dan efisien

Tabel 2.1 Penelitian Terdahulu (Lanjutan)

Nama	**Judul**	**Fitur**	**Metode Analisis**	**Hasil**
Rahayu dan Nurul (2020)	Implementasi Sistem Informasi Pelayanan Kesehatan Pada Klinik Smart Medica	Berbasiskan wb dengan fitur informasi jadwal praktek dokter	- Metode pengembangan sistem waterfall - Tools use case diagram dan class diagram	Adanya informasi jadwal dokter maka pasien dapat memilih waktu yang sesuai dengan jadwal dokter yang menangani penyakitnya
Yuni Mawarni (2022)	Sistem Informasi Layanan Kesehatan Ibu Hamil dan Anak Usia Balita Berbasis Android di Rumah Sakit Umum Pringsewu	Berbasiskan Android dengan fitur registrasi pengguna, pendaftaran layanan poliklinik online, riwayat kesehatan beserta harga, dan notifikasi nomor antian pasien	- Metode pengambangan sistem Extreme Programming - Tools UML (use case diagram, activiity diagram, dan class diagram) - Algoritma	Diharapkan dapat memberikan kemudahan akses pendaftaran layanan dan mendapatkan nomor antrian serta prakiraan waktu akan dilayani.

			antrian queue FIFO	

2.2 Pengertian Sistem Informasi

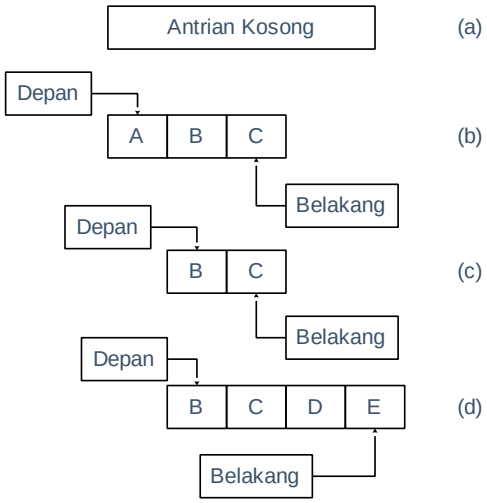
Sistem adalah kumpulan orang yang saling bekerja sama dengan ketentuan-ketentuan aturan sistematis dan terstruktur untuk membentuk satu kesatuan yang melaksanakan suatu fungsi untuk mencapai tujuan. Informasi berupa data yang diolah hingga berguna bagi penerimanya. Oleh karena itu, sistem informasi merupakan suatu kombinasi dari manusia, *hardware*, *software*, internet, dan sumber data yang diubah berupa informasi dalam sebuah organisasi (Yunaeti Anggraeni, 2017).

Sistem informasi adalah suatu sistem didalam suatu organisasi yang merupakan kombinasi dari manusia, teknologi, media, dan prosedur pengendalian yang ditujukan untuk mendapatkan jalur komunikasi penting, memproses tipe transaksi rutin tertentu, memberi sinyal kepada manajemen dan yang lainnya terhadap kejadian-kejadian internal dan eksternal yang penting dan menyediakan suatu dasar informasi untuk pengambilan keputusan yang cerdik. Dalam sistem informasi diperlukannya klasifikasi alur informasi, hal ini disebabkan keanekaragaman kebutuhan akan suatu informasi oleh pengguna informasi (Yunianto, 2019).

2.3 Metode *Queue FIFO* (*First In First Out*)

Antrian atau *queue* adalah salah satu struktur data yang memiliki sistem kerja pertama masuk dan pertama keluar seperti halnya antrian pada dunia nyata. Penambahan data pada *queue* dilakukan pada baris akhir elemen data kemudian dilakukan penghapusan elemen data pada ujung baris depan (Sindar, 2019).

Metode *queue* FIFO merupakan metode dimana orang yang datang lebih awal akan diberikan pelayanan terlebih dahul dan orang yang datang terakhir akan dilayani terakhir. Contoh : Pelayanan kasir supermarket.



Gambar 2.1 Algoritma Antrian FIFO

Gambar 1(a) menyatakan keadaan ketika antrian dalam keadaan kosong. Gambar 1(b) memperlihatkan ketika terdapat tiga item yang sedang antri. Depan menunjuk pada A yang menyatakan data yang akan dilayani duluan dan belakang menunjuk ke C yang menyatakan item yang terakhir kali masuk antrian. Apabila kemudian A dilayani, maka Atidak lagi berada pada antrian. Hal ini digambarkan pada Gambar 1(c). Dalam hal ini, depan sekarang menunjuk ke B. Adapun Gambar 1(d) menyatakan ketika B belum dilayani ternyata ada dua item yaitu D dan E yang ikut antri.

2.4 Pengertian *Android*

Android adalah sistem operasi berbasis *Linux* yang dirancang untuk *smartphone* dan komputer tablet. *Android* adalah sistem operasi *open source* yang memungkinkan perangkat lunak untuk menjadi dimodifikasi dan didistribusikan secara bebas oleh pembuat perangkat. Selain itu, *Android* mumnya ditulis dalam versi yang disesuaikan dari bahasa pemrograman Java (Karim and Agarina, 2019). *Android* adalah sebuah sistem operasi perangkat *mobile* berbasis *linux* yang mencangkup sistem operasi, *middleware*, dan aplikasi. *Android* menyediakan *platform* terbuka bagi para pengembang perangkat lunak dalam pembuatan aplikasi. *Android* menyertakan *kit development* perangkat lunak dalam pembuatan atau pemograman dan modul bagi pengguna *Android*. *Android* juga menyediakan

pasar untuk mendistribusikan aplikasi. secara keseluruhan, *Android* menyatakan ekosistem untuk aplikasi seluler (Karman, Mulyono and Martadinata, 2019).

2.5 Bahasa Pemograman *Kotlin*

Menurut artikel yang tertera pada *website* resmi *Android* yaitu developer.android.com, *Kotlin* adalah bahasa pemrograman modern yang digunakan lebih dari 60% *developer Android* profesional untuk membantu meningkatkan produktivitas, kepuasan, dan keamanan.

Kotlin adalah sebuah bahasa pemograman dengan *Statically typed* (tipe statis) yang berjalan pada *platform Java Virtual Machine* (JVM). Kotlin menggunakan dapat dikompilasi ke dalam kode *JavaScript*. *Kotlin* merupakan bahasa yang powerfull tentu cocok bagi *developer* dalam membuat aplikasi *Android*. Karena Kotlin adalah bahasa nomor satu untuk pengembangan aplikasi *Android*. Adapun kelebihan *Kotlin* adalah (Aljundi and Akbar, 2018) :

a. *Concise* : *Kotlin* mampu mengurangi *boilerplate of code* atau tingkat kerumitan dari kode yang biasa kita tulis, ketika menggunakan bahasa *Java*.
b. *Safe* : *Kotlin* mampu menjamin bahwa setiap *syntax* yang kita tulis secara proses kompilasi dapat mencegah kemungkinan terjadinya *error*, misalnya mampu mencegah terjadinya *NullPointerExceptions* ketika kita *coding* menggunakan bahasa *Java*.
c. *Versatile* : Kotlin sejatinya sama seperti *Java*, karena memang *kotlin* itu sendiri di turunkan dari bahasa induknya, yaitu *Java*. Sehingga *kotlin* juga dapat di pakai dalam pengembangan aplikasi di *web* maupun *mobile*.
d. *Interoperable* : *Kotlin* tidak sama seperti *Scala* ataupun *Clojure*. *Kotlin* mampu membaca *library* yang digunakan atau ditulis dengan bahasa *Java* dan sebaliknya.

2.6 MyQSL

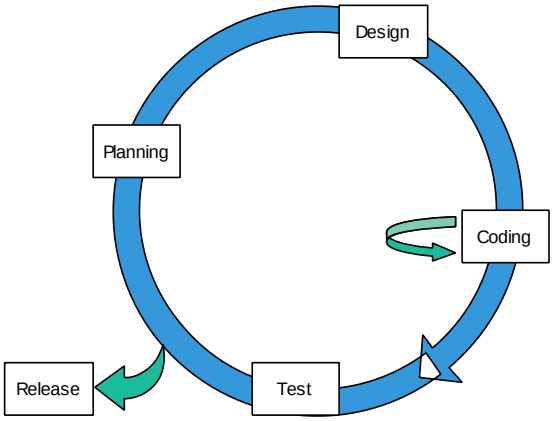
MySQL adalah *DBMS open source* dengan *free software* dan *shareware* Jadi, *MySQL* adalah *database server* dengan lisensi GNU yang gratis sehingga dapat dipakai untuk keperluan pribadi maupun komersial tanpa harus membayar lisensi yang ada (Fitri, 2020).

Pengelolaan DBMS yang digunakan dalam pembuatan aplikasi ini adalah MySQL. MySQL (*My Structure Query Language*) adalah suatu perangkat lunak *database* relasi atau *Relational Database Management System* (RDBMS) yang didistribusikan grastis dibawah *lisensi GPL* (*General Public License*). Dimana setiap orang bebas menggunakan *MySQL*, namun tidak boleh dijadikan produk turunan yang dijadikan *closed source* atau komersial (Muslihudin & Helmiyanto, 2020).

Seperti yang sudah disebutkan sebelumnya, *MySQL* masuk ke dalam jenis RDBMS (*Relational Database Management Sistem*). Maka dari itu, istilah semacam baris, kolom, tabel dipakai pada *MySQL*. *MySQL* merupakan *server* yang melayani *database*, kita dapat mempelajari pemograman khusus yang disebut *query* (perintah) *SQL*. *MySQL* merupakan *database engine* atau *server database* yang mendukung bahasa *SQL* sebagai bahasa interaktif dalam mengelola data. *MySQL* adalah sebuah perangkat lunak sistem manajemen basis data *SQL* atau *DBMS* yang *multithread, multi-user* (Enterprise, 2018).

2.7 Extreme Programming

Salah satu metodologi pengembangan perangkat lunak yang dapat digunakan adalah *Extreme Programming*. *Extreme Programming* (XP) adalah metodologi dalam pengembangan *agile software development methodologies* yang berfokus pada pengkodean (*coding*) yang menjadi aktivitas utama dalam semua tahapan pada siklus pengembangan perangkat lunak. Metode XP dapat diterapkan dengan waktu pembuatan aplikasi yang sebentar atau waktu yang singkat dan berulang pada bagian-bagian berbeda sesuai dengan kebutuhan yang dicapai (Gunawan *et al.*, 2020). Tahapan pengembangan perangkat lunak dengan XP meliputi: *planning* (perencanaan), *design* (perancangan), *coding* (pengkodean) dan *testing* (pengujian) (Borman, Priandika and Edison, 2020). Tahapan-tahapan yang dilakukan pada pengembangan sistem XP dapat dilihat pada gambar 2 berikut ini.



Gambar 2.2 Pemodelan *Extreme Programming*

Sumber: (Gunawan *et al.*, 2020)

Adapun penjelasan tiap tahapan pada metode *extreme programming* adalah sebagai berikut (Rusman, Ramanda and Syaleha, 2021) :

a. Tahap *Planning*

   Tahap *planning* dimulai dengan membuat *user stories* yang menggambarkan *output*, *fitur*, dan fungsifungsi dari *software* yang akan dibuat. *User stories* tersebut kemudian diberikan bobot seperti prioritas dan dikelompokkan untuk selanjutnya dilakukan proses *delivery secaraincremental*.

b. Tahap *Design*

   Untuk desain yang sulit, *Extreme Programming* akan menggunaan *spike solution* dimana pembuatan *design* dibuat langsung ke tujuannya. *Extreme Programming* juga mendukung adanya *refactoring* dimana *software system* diubah sedemikian rupa dengan cara mengubah stuktur kode dan menyederhanakannya namun hasil dari kode tidak berubah.

c. Tahap *Coding*

   Proses *coding* pada XP diawali dengan membangun serangkaian unit *test* dan kemudian diimplementasikan oleh pengembang. *Pair programming* dikenal dalam XP dimana proses pemograman dilakukan oleh *programmer* secara berpasangan dan saling bekerjasama pada satu komputer. Dengan melakukan ini akan didapat *real-time problem solving* dan *real-time quality assurance*.

d. Tahap *Testing*

Tahap ini dilakukan pengujian kode pada unit *test*. Dalam *Extreme Programming*, diperkenalkan XP *acceptance test* atau biasa disebut *customer test*. Tes ini dilakukan oleh *customer* yang berfokus kepada fitur dan fungsi sistem secara keseluruhan. *Acceptance test* ini berasal dari *user stories* yang telah diimplementasikan.

2.8 UML

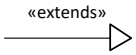
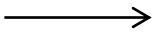
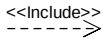
UML (*Unified Modelling Language*) adalah suatu metode pemodelan secara visual sebagai sarana perancangan sistem berorientasi objek atau dikenal juga sebagai bahasa standar penulisan *blueprint* sebuah *software* (Rosa and Shalahuddin, 2018). Jenis pemodelan *UML* yang digunakan dalam pengembangan sistem di dalam penelitian yang dilakukan adalah *use case diagram, activity diagram, squence diagram,* dan *class diagram.*

2.8.1 Use Case Diagram

Use case mendeskripsikan sebuah interaksi antara aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi dari sebuah sistem informasi dan siapa saja yang dapat menggunakan fungsi tersebut. Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *use case* yaitu pendefinisian aktor dan *use case*. Aktor dapat berupa orang atau sistem yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri.

Tabel 2.2 Simbol *Use Case Diagram*

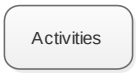
Simbol	**Deskripsi**
Use case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja diawal-awal frase nama *use case*
Aktor	Orang, proses atau sistem lain yang berinteraksi dengan sistem yang dibuat.

	
Asosiasi	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.
Ekstensi 	Relasi *use case* tambahan ke sebuah *use case*, dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan.
Generalisasi 	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.
Include 	*Include* berarti *use case* yang ditambahkan akan selalu dipanggil saat *use case* tambahan dijalankan

2.8.2 Activity Diagram

Activity diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Diagram aktivitas menggambarkan aktivitas sistem bukan proses yang dilakukan oleh aktor.

Tabel 2.3 Simbol *Activity Diagram*

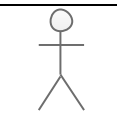
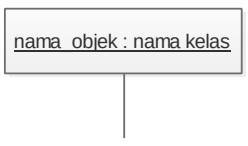
Simbol	**Deskripsi**
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
Percabangan 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
Penggabungan	Asosiasi penggabungan dimana lebih dari satu aktivitas

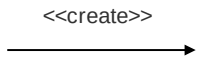
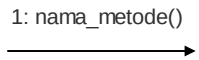
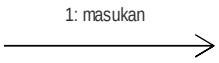
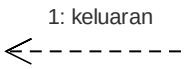
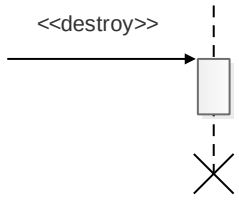
	digabungkan menjadi satu.
Swimlane 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.

2.8.3 Squence Diagram

Diagram skuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Banyaknya diagram skuen yang harus digambar adalah minimal sebanyak pendefinisian *use case.*

Tabel 2.4 *Squence Diagram*

Simbol	**Deskripsi**
	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem infromasi itu sendiri. Jadi walaupun aktor gambar orang tetapi aktor belum tentu merupakan orang
Garis hidup / *lifeline* 	Menyatakan kehidupan suatu objek
Objek 	Menyatakan objek yang berinteraksi pesan

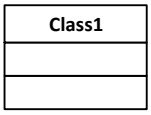
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi. Semua yang terhubng dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya
Pesan tipe *create* 	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat
Pesan tipe *call* 	Menyatakan suatu objek memanggil operasi/metode yang da pada objek lain atau dirinya sendiri
Pesan tipe *send* 	Menyatakan bahwa suatu objek mengirimkan data/masukan/infromasi ke objek lainnya. Arah panah mengarah pada objek yang dikirimi
Pesan tipe *return* 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu. Arah panah mengarah pada objek yang menerima kembalian
Pesan tipe *destroy* 	Menyatakan suatu objek mengakhiri suatu objek yang lain. Arah panah mengarah pada objek yang diakhiri

2.8.4 Class Diagram

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. kelas memiliki apa yang disebut atribut dan metode atau operasi. Atribut merupakan variabel-variabel yang dimiliki suatu kelas, sedangkan operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas (Rosa and Shalahuddin, 2018). Simbol yang ada pada diagram kelas adalah seperti pada Tabel 2.5.

Tabel 2.5 Simbol *Class Diagram*

Simbol	**Deskripsi**

Kelas 	Kelas pada struktur sistem.
Natarmuka/*interface* 	Sama dengan konsep *interface* dalam pemograman berorientasi objek.
Asosiasi 	Relasi antar kelas dalam makna umum, asosiasi biasanya juga disertai dengan *multiplicity*.
Asosiasi berarah 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*.
Generalisasi 	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum-khusus).
Kebergantungan 	Relasi antar kelas dengan makna kebergantungan antar kelas.
Agregasi 	Relasi antar kelas dengan makna semua bagian (*whole-part*).

2.9 Black Box Testing

Black box testing merupakan pengujian untuk mengetahui fungsi perangkat lunak yang telah berjalan sesuai dengan kebutuhannya (Jurnal (ABDI) and Rahardja, 2020). Pengujian dimaksudkan untuk mengetahui apakah fungsi-fungsi, masukan, dan pengeluaran perangkat lunak sesuai dengan spesifikasi yang dibutuhkan (Rosa and Shalahuddin, 2018).