BAB II
TINJAUAN PUSTAKA

2.1 Penelitian Terdahulu

Tabel 2.1 *Milestone* Penelitian Terdahulu

No.	Penulis	Judul	Metode	Hasil
1.	Azmi Sahid Fillah, Ishartono dan Muhammad Fedryansayah	Program Penanggulangan Bencana Oleh Disaster Management Center Dompet Dhuafa	Metode penelitian yang digunakan dalam penulisan jurnal ini adalah deskriptif dengan cara studi pustaka melalui dokumen-dokumen pendukung berupa data dari buku, jurnal, dokumen elektronik maupun internet.	Hasil analisis dari jurnal ini yaitu sebuah konsep berpikir untuk menggambarkan proses pelayanan yang dilakukan oleh Disaster Management Center (DMC) Dompet Dhuafa dalam penanggulangan bencana, sehingga dapat meminmalisir dampak yang dirasakan. Selain itu, proses manajemen bencana terdiri dari tiga tahapan diantaranya pra bencana (mitigasi), tanggap darurat (terjadi bencana) dan pasca bencana (*recovery)*.
2.	Didin Lukmanudin, Fahmi Yusuf	Sistem Informasi Mitigasi Bencana BPBD Kabupaten Kuningan	Perancangan sistem informasi ini menggunakan	Hasil dari penelitian ini sistem berjalan dengan baik sehingga dapat menampilkan

	dan Iwan Lesmana	Berbasis Android	perancangan *Realtional Unified Process* (RUP) yang memanfaatkan konsep *Object Oriented.* Metode ini menggunakan pengembangan model *Unified Modelling Language* (UML).	notifikasi pada *smartphone* kepada *user* ketika ada informasi bencana alam dan peringatan dini secara *realtime*. Selain itu, *user* juga dapat melaporkan bencana alam yang terjadi di lokasi dan melakukan diskusi dengan pihak BPBD setempat.
3.	Fadel Muhammad, Ahmaddul Hadi dan Dedy Irfan	Pengembangan Sistem Informasi Panduan Mitigasi Bencana Alam Provinsi Sumatera Barat Berbasis Android	Metode perancangan aplikasi ini menggunakan pemodelan UML.	Hasil dari pengembangan sistem informasi panduan mitigasi bencana alam Provinsi Sumatera Barat yaitu penggunaan bahasa pemrograman PHP dengan *framework* CI pada sisi *server* dan java dari sisi *client* sehingga menerapkan paradigma *Client-Server*. Selain itu, sistem ini juga dilengkapi dengan panduan mitigasi bencana, berita, artikel, *event* seputar mitigasi, layanan interaksi lapor bencana dan tanya jawab bencana serta peta

				penunjuk lokasi BPBD.
4.	Sri Mulyani, Muh Rifai Katili dan Rampi Yusuf	Sistem Informasi Mitigasi Bencana Banjir Berbasis Android Pada Badan Penanggulangan Bencana Daerah Kota Gorontalo	Metode yang digunakan dalam pengembangan sistem ini adalah menggunakan model *prototype* dengan tahapan *communication, quick plan and modelling quick design, contruction of prototype* dan *deployment delivery and feedback.*	Hasil analisis yang didapatkan dari jurnal ini yaitu sistem informasi mitigasi banjir dapat mempermudah BPBD dalam pengelolaan bencana banjir khususnya di Kota Gorontalo. Selain itu, aplikasi ini menghasilkan informasi berupa notifikasi peringatan dini banjir, informasi banjir *realtime*, jalur evakuasi, dan *GIS* rawan banjir di Kota Gorontalo. Dengan demikian, pengujian sistem ini menggunakan *blackbox* dan *whitebox* dengan hasil sistem sudah memenuhi kriteria sesuai tujuan dari penelitian ini.
5.	Ardhana Januar Mahardhani, Iqbal Akbar Imamudin dan	Upaya Mitigasi Bencana Melalui Aplikasi Dayakan *Mitigation Center*	Metode pelaksanaan dalam kegiatan pengabdian masyarakat ini yaitu sosialisasi program,	Hasil dari penelitian ini berupa aplikasi berbasis *Android* dengan fitur kenal bencana, cegah becana, lapor bencana, berita BPBD,

	Fendy Eko Hardiawan		pembentukan tim, pembuatan aplikasi, pengenalan penggunaan aplikasi, lokarya dan *launching* aplikasi serta evaluasi.	SIG Desa Dayakan, hubungi admin, dan fitur *panic button* (pelaporan cepat). Pada sisi admin peneliti menggunakan *website* untuk mengatur aplikasi Dayakan *Mitigation Center* secara *realtime.*
6.	Efri Tri Ardianto dan Alinea Dwi Elisanti	Pengembangan Aplikasi Penanggulangan Bencana *Ship, Handle & Drive* Berbasis *Android* dan *Web*	Metode penelitian yang digunakan dalam jurnal ini yaitu metode *waterfall* dengan pendekatan sistematis melalui tahapan *analysis design, coding, testing* dan evaluasi.	Hasil analisis dari penelitian ini yaitu sebuah aplikasi penanggulangan bencana yang mempunyai tiga *role* user yaitu penyintas, petugas dan admin dengan fungsi masing-masing. Pada *role* penyintas terdapat fitur lokasi korban dan petugas, SOS bantuan, *push to talk* dan lapor bencana secara *realtime*.
7.	Achmad Affandi dan Radityo Prasetianto Wibowo	Rancang Bangun Aplikasi Bergerak Pendataan Kondisi Bencana (Studi Kasus : MDMC Jawa Timur)	Metodologi penelitian yang digunakan dalam jurnal ini diantaranya studi literatur, analisis kebutuhan sistem, desain sistem,	Hasil analisis aplikasi yaitu aplikasi dapat berjalan dengan baik di *Android* versi Marshmallow+ dan aplikasi dapat menyimpan data pada saat tidak ada koneksi internet serta secara

			pembangunan sistem, pengujian sistem dan pembuatan laporan penelitian.	langsung data terkirim pada *database* ketika terkoneksi internet.
8.	Sularno, Renita Astri dan Deni Irda Mazni	Sistem Informasi Geografis Pemetaan Jalur Evakuasi dan Edukasi Bencana Tsunami Berbasis Pada Kecamatan Padang Barat Kota Padang	Metode penelitian yang digunakan yaitu metode *deskriptif* dengan cara mengumpulkan data dan informasi rute evakuasi dan *shelter* tsunami	Hasil analisis dari jurnal ini yaitu mengangkat topik tentang pemetaan jalur evakuasi dan edukasi bencana tsunami bagi warga yang tinggal di dekat pantai kecamatan Padang Barat. *System* ini memiliki fitur jalur alternative menuju *shelter* terdekat dan memiliki informasi detail mengenai daya tampung. Selain itu, *system* ini memiliki fitur informasi kontak darurat seperti pemadam kebakaran, BPBD, kantor polisi dan informasi terbaru mengenai bencana. Sedangkan dari sisi *administrator*, admin dapat melakukan kelola data titik evakuasi (*shelter)*, mengelola informasi dan

				mengirim pesan.
9.	Luthfiya Anggraini dan Faisal Akib	Aplikasi Edukasi Mitigasi dan Pelaporan Kejadian Bencana Berbasis Android	Metode perancangan aplikasi yang digunakan yaitu *Software Development Life Cycle* (SDLC) dengan ciri khas pengerjaan menggunakan fase *waterfall* sedangkan teknik pengujian menggunakan *blackbox*	Hasil dari penelitan ini sebuah aplikasi *Android* yang membantu masyarakat melaporkan kondisi disekitar lokasi bencana dan edukasi mengenai mitigasi bencana. Hal tersebut diimplementasikan dengan nama fitur panduan mitigasi, pelaporan bencana dan daftar laporan bencana.
10.	Amir Alkodri, Burhan Ishanto dan Sujono	Aplikasi Pengaduan Masyarakat Untuk Pelaporan Kejadian dan Bencana di Basarnas Bangka Belitung	Metode penelitian ini menggunakan *waterfall* dengan model penulisan analisis, desain, pengodean dan pengujian.	Hasil dari penelitian ini yaitu aplikasi *Android* sebagai media pelaporan masyarakat kejadian masyarakat maupun kejadian bencana alam di Bangka Belitung, sehingga dengan adanya aplikasi ini Basarnas dalam menanggapi pelaporan dari masyarakat lebih efektif dan efisien.

2.2 Teori Terkait

2.2.1 Mitigasi Bencana

Mitigasi bencana adalah rangkaian tindakan untuk mengurangi resiko bencana dengan melibatkan pembangunan infrastruktur, peningkatan kesadaran masyarakat, dan pengembangan kapasitas dalam menghadapi berbagai ancaman, baik yang bersifat alamiah maupun bukan alamiah (BPBD, 2018). Hal ini juga diatur pada pasal 1 ayat 6 PP No 21 tahun 2008 tentang penyelenggaraan penaggulangan bencana. Selain itu, dalam mitigasi bencana memiliki empat tahapan pendekatakan yang dilakukan antara lain.

1. Pendekatan teknis
2. Pendekatan manusia
3. Pendekatan adminitratif
4. Pendekatan kurtural

2.2.2 Bencana

Menurut Keputusan Menteri Koordinator Kesejahteraan Rakyat Nomor 17/kep/Menko/Kesra/X/95, definisi bencana adalah peristiwa atau serangkaian peristiwa yang terjadi akibat faktor alam, manusia, atau gabungan keduanya, yang mengakibatkan timbulnya korban dan penderitaan manusia, kerugian harta benda, kerusakan lingkungan, serta kerusakan fasilitas umum, sambil mengganggu aktivitas kehidupan dan mata pencaharian manusia. Selain itu, bencana juga mengakibatkan dampak psikologis terhadap penyintas yang terdampak (BNPB,

2022). Sedangkan definisi lain menurut World Health Organization atau biasa disingkat WHO bencana adalah setiap peristiwa yang menyebabkan kerusakan, gangguan ekologi, hilangnya nyawa, memburuknya pelayanan kesehatan di skala tertentu sehingga memerlukan respon dari luar masyarakat.

2.2.3 Bencana Alam

Bencana alam merupakan bencana yang disebabkan oleh serangkaian perstiwa alam berupa gempa bumi, tsunami, gunung meletus, kekeringan, angin topan, dan tanah longsor (PMI, 2022). Faktor dari bencana alam ini akibat proses *geologis*, proses *klimatologis* dan proses *ekstra terestrial* sehingga bencana alam ini dibagi menjadi tiga jenis faktor penyebabnya, yaitu (Posponegoro & Sujudi, 2016).

1. Bencana alam *geologis* terdiri dari gempa bumi, tsunami, letusan gunung berapi, tanah longsor atau gerakan tanah, dan abrasi.
2. Bencana alam *klimatologis* terdiri atas banjir, angin puting beliung, dan kekeringan.
3. Bencana alam *ekstra tersetrial* seperti *impact* atau hantaman benda dari luar angkasa.

2.2.4 Peringatan Dini

Peringatan dini yaitu serangkaian penyeberan informasi pemberian peringatan sesegera mungkin yang dilakukan oleh BNPB maupun lembaga terkait kepada masyarakat terkait kemungkinan terjadinya bencana di daerahnya. Tujuan dari peringatan dini ini menurut (BNPB, 2022) sesuai Undang-Undang Nomor 24

Tahun 2007 yaitu untuk pengambilan tindakan secara cepat dan tepat dalam rangka mengurangi resiko bencana, serta mempersiapkan tanggap darurat. Peringatan dini ini memiliki beberapa tahapan diantaranya.

1. Pengamatan gejala.
2. Pengkajian hasil pengamatan gejala bencana.
3. Pengambilan keputusan yang dilakukan oleh pihak berwenang.
4. Melakukan penyebarluasan informasi mengenai peringatan bencana.
5. Pengambilan tindakan oleh masyarakat.

2.2.5 *Android*

Android merupakan sebuah *system* operasi perangkat *mobile* yang berbasis linux (*open source*) (Laila Safitri, 2020). *Platform i*ni *open source* bagi para *developer* untuk menciptakan aplikasi yang bisa digunakan oleh berbagai macam piranti bergerak seperti *smartphone.* Hal tersebut, tidak terlepas dari Google yang mengakusisi *Android* sebagai *developer* utama yang kemudian dibuatkan sebuah *platform*. Seiring berjalannya waktu *Android* mengeluarkan versi terbaru untuk menunjang penggunaan *smartphone*, salah satunya yaitu versi *Android* 5.0 *Lollipop*.

Android 5.0 atau versi *Lollipop* dirilis pada tanggal 15 Oktober 2014 dengan fokus pembaruan tampak dari sisi desain yang disesuaikan seiring perkembangan zaman. Hal ini dihadirkan dalam bentuk desain material dan *toolkit UI* yang telah diperluas agar mudah diintegrasikan terhadap pola desain baru

dalam aplikasi. Selain itu, fokus kinerja dari *Android 5.0* ini berjalan secara ekslusif saat waktu proses *new ART*, yang dibangun dari nol dalam mendukung panduan *AOT*, *JIT* dan kode yang diinterpretasikan. Dukungan ini hanya bisa dilakukan pada arsitektur *ARM*, *x86*, dan *MIPS* serta kompatibel sepenuhnya dengan 64 bit.

2.2.6 *Native Application*

Native application merupakan aplikasi yang berjalan secara alami tanpa bantuan *web bowser*, istilah ini biasa digunakan untuk *mobile* (Maryanto, 2014). *Native application* adalah aplikasi yang sengaja dirancang secara spesifik untuk sistem operasi tertentu. Jika aplikasi dikhususkan untuk sistem operasi *Android*, maka sudah pasti tidak dapat beroperasi pada sistem operasi lain.

Keuntungan utama dari *native application* ini dari segi performa yang tinggi dan memiliki *user experience* baik, karena pengembang membuat aplikasi ini menggunakan UI dari perangkat *native*. Dengan demikian, *native application* menawarkan *user* yang lebih baik karena mampu bekerja secara cepat, responsif dan pendistribusian melalui *App Store*.

2.2.7 Android Studio

Menurut Android Developers (2022), Android Studio merupakan Integrated Development Environment (IDE) yang secara resmi digunakan untuk mengembangkan aplikasi Android, yang mendasarinya adalah intelliJ IDEA. Android Studio berperan sebagai editor kode dan memanfaatkan fitur pengembangan yang handal dari IntellJ. Selain itu, Android Studio menyediakan

berbagai fitur untuk meningkatkan produktivitas para pengembang dalam menciptakan aplikasi berbasis Android, termasuk:

1. Sistem pembangunan berbasis Gradle yang fleksibel.
2. Emulator yang responsif dan kaya fitur.
3. Lingkungan terpadu yang memungkinkan pengembang untuk membuat aplikasi untuk berbagai perangkat Android.
4. Kemampuan untuk menerapkan perubahan kode dan sumber daya ke aplikasi yang sedang berjalan tanpa perlu memulai ulang.
5. Integrasi dengan GitHub dan template kode yang membantu pengembang dalam membuat fitur umum dan mengimpor contoh kode.
6. Alat pengujian dan kerangka kerja yang komprehensif.
7. Dukungan untuk bahasa pemrograman C++ dan NDK (Native Development Kit).
8. Alat lint yang membantu dalam memantau kinerja, penggunaan, kompatibilitas versi, dan masalah lainnya.
9. Integrasi dengan Google Cloud Platform untuk mempermudah penggunaan Google Cloud Messaging dan App Engine.*Android* studio yaitu sebuah IDE yang digunakan untuk membuat dan mengembangkan aplikasi yang bisa dijalankan di *platform Android*. *Android* studio ini berbasis IntelliJ IDEA yang mendukung Bahasa pemrograman *Java* dan *Kotlin.* Sedangkan

untuk mendukung tampilan *layoutnya* menggunakan Bahasa *XM*L. Selain itu untuk *deploy* ke perangkat *Android*, *Android* Studio telah terintegrasi degan *Android* SDK (*Software Development Kit).*

2.2.8 Perancangan

Perancangan adalah menentukan proses dan data yang dibutuhkan untuk *system* baru. Keuntungan dari tahap desain system ini yaitu memberikan deskripsi perancangan yang lengkap sebagai panduan bagi *programmer* dalam mengembangkan aplikasi. Konsisten dengan komponen *system* komputerisasi, dirancang pada tahap ini mencakup perangkat keras atau perangkat lunak, *database* dan aplikasi. Definisi lain juga di paparkan oleh (Rizky, 2011) perancangan merupakan proses mendefinisikan sesuatu menggunakan berbagai teknik, termasuk menggambarkan detail arsitektur dan komponen serta kendala yang dihadapi dalam proses pengerjaan.

2.2.8.1 *Unified Modelling Language* (UML)

Unified Modelling Language (UML) untuk pertama kali dikembangkan oleh Grady Booch, Jim Rumbaugh dan Ivars Jacobson merupakan bahasa pemodelan visual dirancang khusus untuk pengembangan dan analisis *system* berorientasi objek serta desain. UML ini sendiri dibagi menjadi 4 macam diagram untuk memodelkan aplikasi perangkat lunak berorientasi objek diantaraya :

1. *Use Case Diagram*

   Use Case Diagram adalah salah satu dari bermacam-macam bentuk dari *UML* diagram yang menceritakan antar hubungan

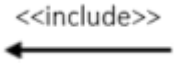
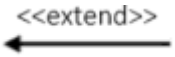
sistem dengan aktor. Tiap aktor yang dibuat akan kelihatan status dan aksesnya sesuai fungsinya dalam sebuah sistem atau aplikasi. Penamaan yang diberikan saat membuat *use case diagram* harus mudah dipahami atau dimengerti oleh *user* maupun khalayak umum. Selain itu, komponen dalam pembuatan *use case diagram* dibagi menjadi tiga dintaranya

a. Sistem
b. Aktor (*Actor*)
c. *Use Case*

Simbol yang digunakan dalam membuat *use case diagram* dapat dilihat pada Tabel 2.2.

Tabel 2.2 Simbol *Use Case*

Simbol	**Nama**	**Keterangan**
	Aktor	Mewakili peran orang, sistem yang lain atau alat ketika berkomunikasi dengan *use case*.
	*Use Case*	Abstraksi dan interaksi antar sistem dan aktor.
	*Association*	Abstraksi dari penghubung antara

		aktor dengan *use case*
	*Generalisasi*	Memperlihatkan spesialisasi aktor untuk bisa berperan serta dengan *use case.*
	*Include*	Memperlihatkan bahwa suatu *use case* seluruhnya adalah fungsionalitas dari *use case* lainnya
	*Extend*	Memperlihatkan bahwa suatu *use case* merupakan tambahan fungsional dari *use case* lainnya jika satu keadaan terpenuhi.

2. *Activity Diagram*

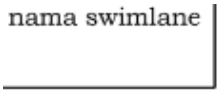
   Menurut (Henderi, 2008) *activity diagram* merupakan *diagram* yang menggambarkan sifat dinamis dari suatu sistem dalam bentuk model aliran dan kontrol dari satu aktivitas ke aktivitas lainnya. Secara khusus, *activity diagram* ini juga biasa digunakan untuk

memodelkan *diagram* alir sebuah sistem kerja atau prosedur bisnis dan operasi secara *internal* (Miller Randy, 2008).

Unsur-unsur dalam *activity diagram* ini yaitu *node* tindakan, *node* kontrol dan *node* objek. Hal ini dibagai menjadi tiga jenis kontrol *node* diantaranya awal dan akhir (final *node* memiliki dua *varietas*, aktivitas final dan final aliran), keputusan dan *merge* serta *fork* dan *join*. Oleh sebab itu, simbol yang digunakan dalam *activity diagram* dapat dilihat pada Tabel 2.3

Tabel 2.3 Simbol *Activity Diagram*

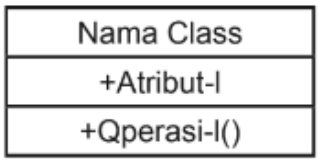
Nama Simbol	**Simbol**	**Deskripsi**
Status Awal / *Strat Point*		Sebuah diagram aktivitas memiliki status awal
Aktivitas		Aktivitas yang dilakukan dalam membuat sistem, nama aktivitas biasanya diawali dengan kata kerja.
Percabangan / *Decision*		Percabangan / *decision* digunakan jika ada pilihan aktivitas lebih dari satu.
Penggabungan		Penggabugan digunakan jika terdapat lebih dari satu aktivitas digabungkan menjadi satu.

Status Akhir / *End Point*		Status akhir dilakukan oleh sistem untuk menutup aktivitas diagram yang dibuat.
Swimlane		*Swimlane* digunakan untuk memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

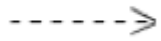
3. *Class Diagram*

   Class diagram merupakan *class-class* yang ada dari sebuah sistem dan hubungan secara logika. *Class* diagram ini menggambarkan alur statis dari sebuah sistem. Oleh sebab itu, *class diagram* yaitu tulang punggung atau kekuatan dasar dari hampir setiap metode berorientasi objek termasuk UML (Henderi *et al*, 2020). Disisi lain, menurut Jeffery menjelaskan bahwa *class diagram* merupakan gambar grafis mengenai struktur objek statis dari sistem, sehingga menunjukkan *class-class* objek dengan menyusun sebuah sistem dan juga hubungan antara *class* objek tersebut. Elemen dari *class diagram* pemodelan *UML* terbagi atas *class*, struktur *class*, sifat *class* (*class behavior*), *association*, *dependency*, relasi-relasi turunannya, keberagaman dan indikator navigasi serta *role name*. Simbol-simbol yang digunakan dalam membuat *class diagram* dapat dilihat pada Tabel 2.4

Tabel 2.4 Simbol *Class Diagram*

Nama Simbol	Simbol	Deskripsi
Class		*Class* terdiri dari tiga bagian diantaranya nama *class*, atribut dan operasi(). Atribut diletakkan dibaris kedua dan dapat digunakan untuk mendeskripsikan daftar variabel. Sedangkan operasi diletakkan dibaris ketiga dan dapat digunakan untuk mendaftar seluruh fungsi serta prosedur yang dipakai.
Association		*Association* menghubugkan antara *class* satu dengan lainnya.

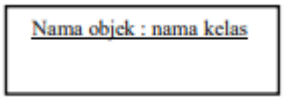
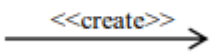
Generalization		*Generalization* merupakan proses dimana perilaku dan struktur data dari objek turunan (*descendant*) diperluas dari objek induk (*ancestor*) yang ada diatasnya.
Nary Association		*Narry Association* yaitu upaya untuk menghindari keterkaitan dengan lebih dari dua objek.
Collaboration		*Collaboration* yaitu menggambarkan rangkaian langkah-langkah yang diambil oleh suatu sistem untuk mencapai hasil tertentu yang terukur bagi seorang aktor.
Realization		*Realization* merupakan operasi yang menghasilkan tindakan

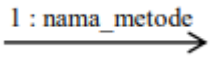
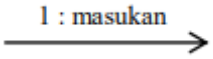
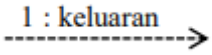
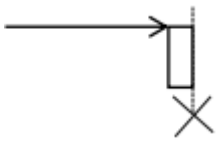
		nyata dari suatu objek.
Dependency		*Dependency* merupakan kaitan antara elemen indpenden dan elemen yang tergantung padanya, dimana jika terjadi perubahan pada elmen independen maka akan mempengaruhi elemen yang bergantung padanya dan tidak bisa berdiri sendiri.

4. *Sequence Diagram*

   Sequence diagram merupakan salah satu cara untuk menjelaskan bagaimana suatu operasi dilakukan, termasuk pesan-pesan yang dikirim antar objek dan kapan pelaksanaannya. Setiap objek termasuk actor memiliki *lifeline* yang memperesentasikan waktu aktifnya, ditunjukkan dalam bentuk kolom vertical. Pesan-pesan diwakili oleh panah menghubungkan satu *lifeline* ke *lifeline* lainnya. Garis panah ini menunjukkan aliran komunikasi atau interaksi antar objek. Pada fase desain berikutnya, pesan-pesan ini

akan dipetakan menjadi operasi atau metode dari kelas-kelas yang sesuai. Penjelasan dari simbol yang digunakan pada *sequence diagram*, dapat dilihat pada Table 2.5.

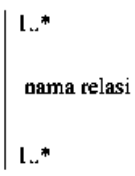
Nama Simbol	**Simbol**	**Deskripsi**
Actor		*Actor* yaitu orang atau system eksternal yang menggunakan *system*.
Lifeline / garis hidup		*Lifeline* / garis hidu menjelaskan kehidupan da suatu objek.
Objek		Objek merupakan yang terlibat dalam *system* untuk berinteraksi.
Waktu Aktif		Waktu aktif merupakan objek dalam keadaan aktif dan saling berinteraksi.
Pesan tipe *create*		Pesan tipe *create* merupakan arah panah mengarah pada objek yang dibuat.

Pesan tipe *call*		Pesan tipe *call* merupakan suatu metode yang berada pada objek lain atau dirinya sendiri.
Pesan tipe *send*		Pesan tipe *send* yaitu menyatakan sebuah abstraksi dan interaksi antara system dan *actor*.
Pesan tipe *return*		Pesan tipe *return* yaitu menunjukkan elemen lain bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemennya.
Pesan tipe destroy		Pesan tipe *destoory* menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada *create* maka ada *destroy*.

2.2.8.2 *CDM (Conceptual Data Model)*

Conceptual Data Model merupakan konsep yang terkait dengan penggunaan data yang disimpan dalam *database*. *Conceptual Data Model* ini menggunakan sistem simbol standar yang membentuk bahasa formal namun sederhana yang menyampaikan banyak informasi data yang dimodelkan. Simbol yang digunakan dalam *Conceptual Data Model* menggunakan sejumlah kontruksi pemodelan dasar yang ditemukan dalam *Entity relationship Diagram* (ERD) yang berisi entitas, atribut dan relasi. Simbol yang digunakan dalam perancangan *Conceptual Data Model* dijelaskan pada Tabel 2.7

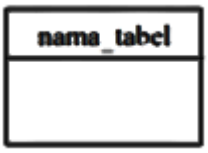
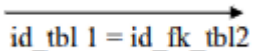
Tabel 2.7 Simbol *Conceptual Data Model*

Simbol	Keterangan
	Entitas atau tabel berfungsi untuk menyimpan data dalam basis data.
	Relasi antar tabel terdiri atas nama relasi dan *multiplicity*.

2.2.8.3 *PDM (Physical Data Model)*

Physical Data Model adalah suatu jenis model data yang menggambarkan hubungan antar *tabel* secara fisikal. Setiap tabel mempunyai sejumlah kolom dengan nama unik beserta tipe datanya masing-masing. Penerapan *Physical Data Model* membutuhkan pemahaman mengenai karakteristik dan batasan knerja dari DBMS yang dipakai. Simbol yang digunakan dalam perancangan *Physical Data Model* dapat dilihat pada Tabel 2.8.

Tabel 2.8 Simbol *Physical Data Model*

Simbol	**Keterangan**
	Tabel berfungsi untuk menyimpan data dalam basis data.
	Relasi antar tabel terdiri atas persamaan antara *primary key* tabel yang diacu dengan kunci yang menjadi referensi

	acuan pada tabel lain.

2.2.9 Firebase

Firebase merupakan *platform* yang disediakan oleh Google untuk pengembangan aplikasi berbasis *mobile* dan *website*. *Platform* ini menyediakan berbagai layanan dan alat yang memungkinkan pengembang membangun dan mengelola aplikasi dengan cepat dan efisien. Firebase juga memiliki fitur seperti *realtime database*, *authentication, cloud firestore,cloud storage, cloud functions, cloud messaging, analytics, performance monitoring, remote config*. Hal ini membuatnya menjadi pilihan popular bagi para pengembang karena kemudahan penggnaan dan manfaat yang ditwarakan dalam membangun aplikasi yang berkualitas.

2.2.9 Java

Java merupakan bahasa pemrograman yang bersifat lintas *platform*, artinya bahasa ini dapat dipakai untuk menyusun program dari berbagai system operasi seperti Linux, Windows dan UNIX. Selain itu, Java tidak menyediakan IDE khusus seperti halnya bahasa lainnya. Bahasa pemrograman Java ini sebagai *Object Oriented programming* yang artinya semua akspek yang terdapat di java adalah *object*. Menurut definisi Sun Microsystem, di dalam buku.

2.2.10 Power Designer 16

Power Designer 16 merupakan *tools* yang dikeluarkan oleh Sybase untuk membangun sebuah *system* informasi secara cepat, terstruktur dan efektif. Power Designer ini juga mendukung beberapa pemodelan seperti *requirement management, bussnies process,data modeling, XML modeling, application modeling dengan UML, information liquidity modeling dan integrated modeling*. Kelebihan dari perangkat lunak ini yaitu dapat membantu untuk memeriksa apakah model yang dibuat sudah valid atau belum dan dapat langsung di generate menjadi bentuk *Entity Relationship Diagram.*

2.2.11 Gitlab

Gitlab adalah sebuah *platform* yang digunakan untuk menggunakan Git, suatu alat yang mempermudah para pengembang. Alat ini berperan sebagai Sistem Kontrol Versi (Version Control System atau VCS) yang berfungsi untuk melacak perubahan dalam berkas kode. Bagi para pengembang, Git sangat bermanfaat karena memudahkan dalam berkolaborasi dalam menyelesaikan proyek, baik dengan pengembang lain maupun secara individu.

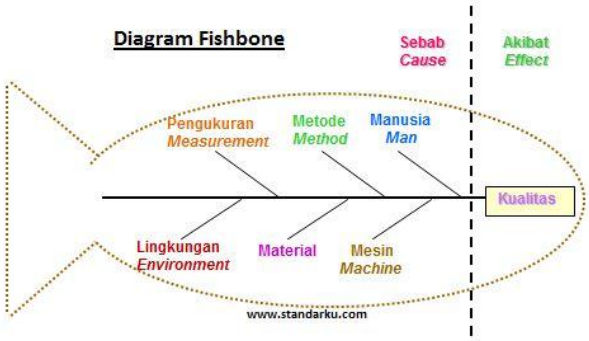
Gitlab merupakan layanan yang mengizinkan para pengembang untuk memiliki akses jarak jauh ke repositori Git. Gitlab menjadikan repositori kode pengembang sebagai tuan rumah, serta menyediakan berbagai fitur untuk mengelola siklus pengembangan perangkat lunak. Beberapa contoh fitur dari Gitlab meliputi pengelolaan pembagian kode antara anggota tim yang terlibat dalam proyek, pelacakan bug, ruang wiki, dan lain-lain.

2.2.12 Figma

Figma merupakan sebuah *platform* berbasis *website* yang digunakan untuk pengeditan grafis dan sebagai alat untuk *prototyping. Platform* ini juga memiliki fitur *offline* yang dapat diaktifkan melalui aplikasi *desktop* pada system operasi Mac OS dan Windows. Dengan kata lain, figma sebagai alat desain digital. Keunggulan Figma dibandingkan dengan AdobeXD yaitu kemampuannya yang mendukung kolaborasi dalam pekerjaan yang sama oleh beberapa individu secara bersamaan, meskipun mereka berada di lokasi yang berbeda mengizinkan kolaborasi dalam lingkungan virtual. Kemampuan ini menjadi factor utama yang menjadikan Figma favorit di kalangan banyak designer UI/UX

2.2.13 Fishbone Diagram

Menurut (Jhon Bank, 1992) *fishbone* diagram memiliki kata lain *Cause and Effect* diagram yang menyerupai ikan dan dapat menunjukkan sebab akibat dari suatu permasalahan. Faktor yang menjadi penyebab utama kualitas pada *fishbone diagram* yaitu 5M + 1 E diantarnya *machine, human, method, material, measurment* dan lingkungan. Faktor-faktor ini nantinya berguna untuk menggabungkan jenis akar permasalahan kedalam sebuah kategori. Simbol yang digunakan untuk merancang *fishbone diagrami* dapat dilihat pada Gambar 2.1.



Gambar 2.1 *Fishbone Diagram*

Pada tabel 2.9 merupakan penjelasan dari gambar *fishbone diagram* yang digunakan untuk mengidentifikasi permasalahan dalam perancangan aplikasi.

Tabel 2.7 Penjelasan Simbol *Fishbone Diagram*

Nama Simbol	Keterangan
Bagian Kepala Ikan	Pada bagian kepala dituliskan masalah atau topik yang akan dianalisa.
Bagian Tulang Ikan	Bagian tulang ikan yaitu menuliskan kategori-kategori yang dapat mempengaruhi topik tertulis pada kepala ikan. Penentuan kategori menggunakan metode 5M+1E yaitu *man, method, material, machine, measurment*, dan *environment*.