BAB II
LANDASAN TEORI

2.1 Penelitian Terdahulu

Berikut adalah beberapa penelitian terdahulu :

Tabel 2 *Literature Review*

Tahun	Penulis	Judul	Metode	Hasil	Limitasi
2020	Johar Saputra Irsandi , Iskandar Fitri , Novi Dian Nathasia	Sistem In*form*asi Pemasaran dengan Penerapan *CRM (Customer Relationship Management)* Berbasis *Website* menggunakan Metode *Waterfall* dan *Agile*	*Waterfall* dan *Agile Methodology*	Dengan menggunakan metode *Customer Relationship Management* (*CRM*) untuk pelayanan transaksi secara *online* membuat hubungan antara toko usaha dan konsumen lebih dekat karena konsumen tidak lagi harus datang ke toko untuk membeli barang sehingga membuat konsumen loyal. dan dalam penyampaian in*form*asi produk terbaru akan lebih mudah dan lebih cepat, serta pelanggan dapat dengan mudah menghubungi pihak toko dalam memberi saran, kritikan, atau pertanyaan.	CRM yang berbasis web dengan menggunakan metodologi *Waterfall* dan *Agile* akan sedikit memakan waktu deployment pada aplikasi, dan CRM ditujukan untuk pengguna (pelanggan) dalam jumlah banyak.
2021	Andreyas Ariesta, Yumi Novita	Penerapan Metode *Agile* Dalam	*Agile Scrum Methodology*	Implementasi metode *Agile Scrum* ini	Metode *Agile* yang digunakan yaitu *scrum.* Penerapan

	Dewi, Findi Ayu Sariasih, Firstianty Wahyuhening Fibriany	Pengembangan *Application* Programming Interface System Pada PT XYZ		menggunakan alat bantu yaitu software Jira. Untuk memudahkan pemahaman dalam implementasinya, digunakan dua pendekatan yaitu skenario dan implementasi metode *Agile Scrum* dalam pengembangan *Application Programming Interface.* Skenario proses yang dihasilkan akan dijelaskan dalan bentuk visual pada bagian implementasi sistem. Dalam Proyek *Application Programming Interface* diperoleh kesimpulan bahwa metode *Agile Scrum* dapat diterapkan untuk memenuhi kebutuhan manajemen kerja tim di PT. XYZ untuk meningkatkan kinerja tim Developer.	ini menggunakan alat bantu s*oftware* Jira. Pendekatan skenario dalam pengembangan API membuat beberapa *Use Case* sulit dijelaskan dan diaplikasikan secara luas.
2019	Shon Hadji, M. Taufik, Sri Mulyono	Implementasi Metode Scrum Pada Pengembangan	*Scrum Methodology*	Pengembangan aplikasi delivery order berbasis website studi	Metode pengembangan sistem yang digunakan adalah

		Aplikasi Delivery Order Berbasis Website (Studi Kasus Pada Rumah Makan Lombok Idjo Semarang)		kasus di Rumah Makan Lombok Idjo telah dibangun dengan menggunakan metode *Scrum* dimana dengan metode ini dapat mengatasi perubahan requirements pada saat fase pengembangan sistem dan scrum memiliki tahapan yang bersifat perulangan dimana jika produk pada sprint pertama belum cukup memenuhi kebutuhan, maka pada sprint berikutnya dapat dikembangkan sistem yang sesuai dengan evaluasi pengguna.	*scrum.* Tahapan dari metode *scrum* terdiri dari membentuk *team scrum*, membuat *product backlog*, dan *fase sprint.* Membuat *product backlog* akan menambah waktu *deployment*.

2.2 Penelitian Pengembangan (*Development Research*)

2.2.1 Pengertian

Sesuai dengan Richey (1994:3), "Penelitian Pengembangan" didefinisikan sebagai alat yang disistematisasikan untuk merencanakan, melaksanakan, dan mengevaluasi setiap program, prosedur, atau produk yang harus memenuhi persyaratan konsistensi dan efektivitas internal. Van den Akker (1999:4) merangkum hasil studi pengembangan berdasarkan dua tujuan, yaitu pengembangan produk prototipe (termasuk memberikan umpan balik tentang

kemanjurannya) dan diskusi tentang metodologi yang tepat untuk pengujian dan evaluasinya. Dalam penelitian terkait, Richey dan Nelson (2013:1103) membandingkan dua jenis penelitian untuk bisnis, yaitu:

a. Fokus pada penelitian, pengembangan, dan/atau evaluasi terhadap produk atau program yang bersangkutan dengan tujuan untuk lebih mengetahui proses pengembangan dan memahami kondisi yang menyulitkan implementasi program.
b. Terminologi yang digunakan dalam evaluasi sebelumnya terhadap proses perancangan, pengembangan, atau evaluasi dengan tujuan untuk menetapkan model dan/atau prosedur baru untuk proses tersebut

Menurut Tim Puslitjaknov (2008:8-12), penelitian pengembangan memiliki tiga komponen utama, yaitu :

a. Model Pengembangan

   Model pengembangan merupakan alat yang digunakan untuk membantu mengembangkan produk yang akan dihasilkan. Satu-satunya model pengembangan adalah model prosedural. Model prosedural adalah model yang secara eksplisit menentukan langkah-langkah yang harus diambil untuk menghasilkan produk tertentu.

b. Prosedur Pengembangan

   Prosedur penelitian pengembangan memaparkan prosedur yang digunakan oleh penulis/pengarang saat membuat produk tertentu. Saat merakit komponen produk jadi, proses pembuatannya berbeda dari model pembuatannya. Peneliti menyebutkan sifat-sifat komponen pada kedua tahun pengembangan, menjelaskan secara analitis fungsi komponen dalam kedua tahun pengembangan produk, dan menjelaskan hubungan antar komponen dalam sistem.

c. Uji Coba Produk

Uji coba produk adalah area yang sangat penting untuk dipertimbangkan saat melakukan riset pasar. Setelah pemeriksaan akhir produk, pengadaan produk dilakukan. Uji membeli produk dengan maksud untuk menentukan apakah barang yang diproduksi layak untuk digunakan atau tidak. Uji telusuri produk juga mengkaji bagaimana produk yang dibuat dapat mencapai maksud dan tujuan.

Analisis sistematis terhadap suatu program, proses, atau produk yang menghambat tahap perancangan, pengembangan, dan evaluasi disebut penelitian pengembangan. Fokus analisis pengembangan bisa pada data produk yang menjadi objek utama kajian atau bisa juga pada data terkait metodologi yang digunakan untuk perancangan, pengembangan, dan evaluasi produk. Penelitian pengembangan dilakukan dengan menekan tiga komponen berbeda, yaitu model pengembangan, prosedur pengembangan, dan uji coba produk.

2.3 Agile Methods

Metode Pengembangan *Agile* (*Agile Methods*) merupakan salah satu metode pengembangan perangkat lunak terbaru, yang memiliki tahapan yang berbeda dengan metode pengembangan perangkat lunak lainnya. Perbedaan tersebut meliputi prinsip kerja dan langkah-langkah dalam metode tangkas (Ependi, 2012: A-1). Metode tangkas mulai berkembang pada tahun 2000-an dan merupakan metodologi baru dan sangat fleksibel. Metode *agile* dikembangkan karena metode pengembangan perangkat lunak tradisional memiliki banyak faktor yang menyebabkan proses pengembangan gagal tampil seperti yang diharapkan oleh pengguna (Widodo, 2008:1).

Widodo (2006: E-95) mencatat bahwa tangkas dapat berarti gesit, cepat, atau ringan. *Agile* adalah pendekatan yang ringan dan cepat untuk pengembangan perangkat lunak. *Agile* Manifesto (Shore, 2008: 9) secara formal menggambarkan cara berpikir *Agile* untuk pengembangan perangkat lunak. Manifesto Pengembangan Perangkat Lunak *Agile* adalah sebagai berikut (http://*agile*manifesto.org/iso/id/) : “Manifesto Pengembangan Perangkat Lunak *Agile*. Kami menemukan cara yang lebih baik untuk mengembangkan perangkat

lunak dengan melakukan dan membantu sesama untuk menggunakannya. Melalui usaha ini kami telah dapat menghargai: Individu dan interaksi **lebih dari proses dan sarana perangkat lunak,** Perangkat lunak yang bekerja **lebih dari dokumentasi yang menyeluruh,,** Kolaborasi dengan klien **lebih dari negosiasi kontrak,** Tanggap terhadap perubahan **lebih dari mengikuti rencana.** Demikian, walaupun kami menghargai hal di sisi kanan, kami lebih menghargai hal di sisi kiri."

Prinsip-prinsip metodologi pengembangan *Agile* dikenal sebagai *Agile Manifesto*. *Agile Manifesto* (Hohl et al., 2018) terdiri dari 12 prinsip utama, yaitu:

- Prioritas nomor satu adalah mempertahankan pelanggan dengan memasang perangkat lunak yang dapat diandalkan dan konsisten.
- Menyambut perubahan persyaratan, bahkan di akhir pengembangan perangkat lunak. Proses tangkas memanfaatkan perubahan untuk meningkatkan keunggulan kompetitif klien.
- Menghasilkan perangkat lunak yang berfungsi secara teratur, mulai dari minggu hingga bulan, dengan kerangka waktu yang lebih pendek lebih disukai.
- Mitra bisnis dan pengembang perangkat lunak harus bekerja sama setiap hari selama proyek berlangsung.
- Membangun proyek seputar pengembangan pribadi yang termotivasi. Berikan lingkungan dan dukungan serta kepercayaan yang diperlukan untuk menyelesaikan pekerjaan.
- Metode transfer informasi yang paling efisien dan efektif dalam tim pengembangan perangkat lunak adalah komunikasi tatap muka.
- Perangkat lunak yang tersedia adalah ukuran utama kemajuan.
- Proses tangkas mendorong pengembangan berkelanjutan. Sponsor, pengembang, dan pengguna akan dapat melanjutkan dengan kecepatan tetap.
- Fokus berkelanjutan pada keunggulan teknis dan desain yang baik meningkatkan ketangkasan.

- Kesederhanaan (seni memaksimalkan jumlah pekerjaan yang dilakukan) sangat penting.
- Arsitektur, persyaratan, dan desain perangkat lunak terbaik berasal dari tim yang mengatur diri sendiri.
- Tim pengembangan secara teratur merefleksikan bagaimana meningkatkan efisiensi, kemudian menyesuaikan dan menyesuaikan kebiasaan kerja

Agile Methods mempunyai kelebihan dibandingkan menggunakan metode-metode lainnya. Dari Ependi (2012:A-2), kelebihan *Agile Methods* diantaranya merupakan menaikkan rasio kepuasan klien. Klien bisa melakukan *review* lebih awal terhadap perangkat lunak yang dikembangkan pada ketika proses pengembangan. *Agile Methods* mengurangi resiko kegagalan implementasi *software* dari segi nonteknis. Bila terjadi kegagalan pada proses pengembangan, maka nilai kerugian (material juga imaterial) tak terlalu akbar. *Agile Methods* adalah metode pengembangan yang bersifat fleksibel dalam menanggapi perubahan- perubahan kebutuhan serta persyaratan menggunakan permanen mengedepankan komunikasi, kepuasan klien, perangkat lunak yang bekerja, serta kerja sama.

Saat ini *Agile Methods* sudah cukup banyak berkembang, di antaranya adalah (Ependi, 2012:A-2) *Extreme Programming* (XP), *Scrum Methodology*, *Crystal Family*, *Dynamic System Development Method* (DSDM), Adaptive *Software Development* (ASD), *Feature Driven Development* (FDD). *Agile Methods* paling popular yang digunakan untuk pengembangan aplikasi *Web* adalah *Extreme Programming* (XP) (Kappel, et al, 2006:211)

2.4 Scrum

Menurut schwaber & sutherland *scrum* adalah sebuah kerangka kerja yang dapat mengatasi suatu masalah komplek yang selalu berubah, dan juga dinilai dapat memberikan kualitas produk yang baik sesuai dengan keinginan pengguna secara kreatif dan produktif (K. Schwaber et al, 2013)

Berikut adalah beberapa poin kunci yang dibahas dalam "*Scrum Guide*":

Kerangka Kerja *Scrum*:

1. Panduan ini menjelaskan kerangka kerja *Scrum*, yang terdiri dari peran, acara, artefak, dan aturan tertentu. Tiga peran utama dalam *Scrum* adalah *Product Owner*, *Scrum Master*, dan Tim Pengembangan. *Product Owner* adalah pihak yang memegang tanggung jawab terhadap seluruh kegiatan dan sekaligus menjadi jembatan antara *stakeholder* dan Tim dari *Scrum*. *Scrum Master* pada aktor bertanggung jawab dalam mengawasi dan memfasilitasi Tim *Scrum* agar proyek dapat berjalan dengan baik sebagaimana mestinya. *Scrum Master* ini melakukan kegiatan *Sprint Planning* dan Mengelola *Sprint*. Sedangkan *Developer* atau Tim Pengembang memiliki tanggung jawab dalam sebagai bagian yang melaksanakan pengembangan *software* dari kegiatan yang sedang dijalankan.
2. *Product Backlog: Product Owner* bertanggung jawab untuk memelihara *Product Backlog*, yang merupakan daftar fitur, persyaratan, dan peningkatan yang diprioritaskan untuk produk. *Product Backlog* berkembang seiring perkembangan produk dan pemahaman tentangnya.
3. *Sprint: Scrum* beroperasi dalam serangkaian iterasi berdurasi tetap yang disebut "*Sprint*." *Sprint* biasanya berlangsung selama 1 hingga 4 minggu dan memiliki batasan waktu, yang berarti durasinya tidak berubah setelah dimulai.
4. *Sprint Planning*: Sebelum setiap *Sprint*, ada pertemuan *Sprint Planning* di mana Tim Pengembangan memilih item dari *Product Backlog* dan berkomitmen untuk menyelesaikannya dalam Sprint.
5. *Daily Scrum: Daily Scrum* adalah acara berdurasi 15 menit yang diadakan setiap hari selama *Sprint*. Tim Pengembangan bertemu untuk menyelaraskan aktivitas, membahas kemajuan, dan merencanakan untuk 24 jam berikutnya.
6. *Sprint Review*: Pada akhir setiap *Sprint*, ada pertemuan *Sprint Review* di mana Tim Pengembangan mendemonstrasikan pekerjaan yang diselesaikan selama *Sprint* kepada *stakeholder* dan mengumpulkan umpan balik.

7. *Sprint Retrospective: Sprint Retrospective* diadakan setelah *Sprint Review* dan memungkinkan Tim *Scrum* untuk memeriksa prosesnya dan mengidentifikasi perbaikan potensial.
8. Artefak: *Scrum* mendefinisikan tiga artefak utama: *Product Backlog*, *Sprint Backlog*, dan *Increment. Increment* adalah jumlah dari semua item *Product Backlog* yang selesai pada akhir Sprint dan harus berpotensi dapat dirilis.
9. Pengendalian Proses Empiris: *Scrum* didasarkan pada prinsip transparansi, inspeksi, dan adaptasi. Tim *Scrum* secara rutin memeriksa dan menyesuaikan proses kerjanya dan produknya untuk meningkatkan nilai yang dihasilkan.

2.5 MySQL

MySQL adalah sistem manajemen basis data relasional (*RDBMS*) sumber terbuka klien-server berdasarkan SQL (*Structured Query Language*). *RDBMS* adalah perangkat lunak yang digunakan untuk membuat dan mengelola basis data menggunakan model relasional.

2.6 Database

Dalam Secara umum *database* dapat didefinisikan sebagai kumpulan data yang saling berhubungan satu dengan yang lainnya secara sistematik. *Database* bermula dari ilmu komputer, akan tetapi seiring berkembangnya ilmu pengetahuan, makna *database* kemudian meluas. Dengan adanya *database* banyak sekali hal yang dapat diperoleh, antara lain ketepatan, kecepatan, dan kemudahan dalam pengambilan informasi, selain itu juga dapat menghemat tempat penyimpanan.

Sistem *database* merupakan sistem yang bertugas memanajemen *record* menggunakan komputer dan untuk menyimpan maupun mengambil kembali informasi yang diperlukan oleh pemakai. Selain itu sistem *database* juga bisa diartikan sebagai gabungan antara dua unsur, yaitu *database* dan sistem manajemen *database* (Arifds, 2010).

2.7 Laravel

Laravel merupakan framework yang dapat membantu *web developer* dalam memaksimalkan penggunaan *PHP* dalam proses pengembangan *website*. Seperti

diketahui, *PHP* sendiri merupakan bahasa pemrograman yang cukup dinamis. Dimana kehadiran *Laravel* kemudian membuat *PHP* menjadi lebih *powerful*, cepat, aman, dan *simple*. Terlebih lagi, *framework* ini selalu memunculkan teknologi terbarunya di antara *framework PHP* lain.

Framework ini mengikuti struktur *MVC* atau *Model View Controller*. Dimana *MVC* adalah metode aplikasi yang memisahkan data dari tampilan berdasarkan komponen-komponen aplikasi, seperti manipulasi data, *controller*, dan *User interface*. Penggunaan struktur *MVC* ini membuat *Laravel* mudah untuk dipelajari dan mampu mempercepat proses pembuatan prototipe aplikasi web. Sebagaimana ia juga menyediakan fitur bawaan seperti otentikasi, mail, perutusan, sesi, dan daftar berjalan. *Laravel* juga lebih berfokus pada *end-User*, yang artinya hanya berfokus pada kejelasan dan kesederhanaan, baik dari penulisan hingga tampilan. Ia pun bisa menghasilkan fungsional aplikasi web yang berjalan dengan semestinya.

Terdapat beberapa kelebihan dari penggunaan *Laravel* yaitu antara lain (Komputer, 2010):

1) Menghemat waktu, karena adanya struktur dan *library* yang disediakan sehingga tim pengembang hanya perlu fokus terhadap logika pemrograman.
2) *Laravel* memiliki standar yang baku sehingga dapat digunakan kembali pada proyek yang lain *(code reuse).*
3) Adanya dukungan komunitas.

Laravel memiliki kumpulan kode *best practice* yang teruji sehingga tim pengembang dapat meningkatkan kualitas dari pemrograman dari proyeknya.

2.8 Visual Studio Code

Visual Studio Code (disingkat *VSCode*) adalah perangkat lunak penyunting kode-sumber buatan Microsoft untuk Linux, macOS, dan Windows. *Visual Studio Code* menyediakan fitur seperti penyorotan sintaksis, penyelesaian kode, kutipan kode, merefaktor kode, pengawakutuan, dan Git. Microsoft merilis sumber kode *Visual Studio Code* di repositori GitHub dengan lisensi MIT (*Code - OSS*),

sedangkan biner yang dibangun oleh Microsoft tidak dirilis dengan lisensi MIT dan merupakan perangkat lunak berpemilik.

Visual Studio Code pertama kali diperkenalkan di tanggal 29 April 2015 oleh Microsoft di konferensi Build 2015. Versi pratinjau dirilis tidak lama setelah itu.

Visual Studio Code dibangun menggunakan aplikasi web Node.js dan kerangka Electron. Ini memiliki beberapa kekurangan seperti penggunaan RAM yang berlebih, mengingat bahwa *Visual Studio Code* berjalan di atas kerangka Electron yang sangat bergantung dengan peramban web (Lardinois, 2015).

2.9 HTML

HTML atau *Hypertext Markup Language* dirancang dengan tujuan untuk menghubungkan satu halaman web dengan halaman web lainnya. *HTML* pertama kali dikembangkan pada 1989 oleh Tim Berners-Lee bersamaan dengan protocol *HTTP (Hypertext Transfer Protocol)*. *HTML* merupakan bahasa pemrograman web yang memberitahukan web *browser* bagaimana menyusun dan menyajikan konten di halaman web di mana dapat diartikan bahwa *HTML* merupakan sebuah pondasi dari web. *HTML* disusun menggunakan bahasa yang sederhana, sehingga mudah dalam pengaplikasiannya. Seiring dengan perkembangannya, saat ini *HTML* dapat menampilkan objek – objek seperti teks, tabel, tautan, gambar, audio dan video pada halaman web (Solichin, 2016:10).

Standar perintah *HTML* dikembangkan oleh sebuah konsorsium W3C atau *World Wide Web Consortium*. Terdapat beberapa standar versi *HTML* yang pernah dirilis oleh W3C antara lain *HTML* 2.0, *HTML* 3.2, *HTML* 4.0 yang direvisi menjadi *HTML* 4.01, X*HTML* yang merupakan standar *HTML* 4.01 dengan terdapat beberapa penyesuaian, dan hingga saat ini standar *HTML* terakhir yang dirilis adalah *HTML*5 dengan fitur baru seperti penanganan audio, video dan penyimpanan lokal (Solichin, 2016:10).

2.10 PHP

PHP atau *Personal Home Page* (Situs Personal) adalah bahasa pemrograman *script* yang paling banyak digunakan pada saat ini di mana sering digunakan pada pemrograman situs web dinamis, walaupun tidak menutup kemungkinan

digunakan pada pemakaian yang lain. Menurut Solichin (2016:11-13) *PHP* merupakan salah satu bahasa pemrograman berbasis web yang memiliki beberapa keunggulan dibandingkan dengan bahasa pemrograman lainnya, yaitu antara lain:

1) *PHP* dapat diunduh dan digunakan dengan gratis, di mana *PHP* dikembangkan oleh komunitas *open source* dan selalu didistribusikan secara gratis dari, oleh dan untuk web *developer*.
2) *PHP* berlisensi *GNU General Publish License* (GPL) hal tersebut merupakan sebuah jaminan bahwa semua versi *PHP* akan selalu dibagikan dengan gratis. Walaupun untuk setiap program dan aplikasi yang dihasilkan dengan menggunakan *PHP* akan memiliki lisensinya sendiri dari pihak web *developer*.
3) *PHP* memiliki per*form*a yang handal dan efisien dimana dapat melayani jutaan akses di setiap harinya.
4) *PHP* mendukung hamper di semua perangkat *database*, seperti *MySQL, Oracle, PostgreSQL, Informix, Interbase, Sybase, MariaDB* hingga *SQLite* bahkan beberapa di antara *database* tersebut sudah dapat terhubung dengan *PHP* secara *default*.
5) *PHP* dirancang secara khusus untuk aplikasi berbasis web, oleh karena itu *PHP* menyediakan banyak pustaka (*library*) bawaan yang terkait dengan web agar dapat langsung digunakan.
6) *PHP* dapat dijalankan pada hampir semua sistem operasi *(cross platform).*
7) Perintah – perintah pada bahasa pemrograman *PHP* mudah dipelajari karena sebagian besar diadopsi dari bahasa pemrograman populer lainnya seperti C/C++, Java dan Perl.

2.11 CSS

CSS atau juga disebut *Cascading Style Sheets* merupakan sebuah dokumen yang berisi perintah yang bertujuan untuk memisahkan isi dengan *layout* dalam halaman – halaman web yang dirancang. *CSS* menyediakan *template* berupa *style* untuk dibuat dan mengizinkan untuk menulis kode yang lebih mudah dari halaman – halaman web yang akan dibuat. *CSS* memberikan kemudahan dalam melakukan pengaturan tampilan keseluruhan web hanya dengan menggantikan atribut – atribut atau perintah dalam *style CSS* dengan *CSS* atribut yang diinginkan. Saat ini *CSS*

merupakan *style* yang sering kali digunakan dikarenakan berbagai kemudahan dan memiliki atribut yang lengkap (Kurniawan dan Kom, 2013:1-2).

Secara umum, *CSS* digunakan untuk memformat halaman web yang dibuat dengan *HTML* dan X*HTML*. Spesifikasi *CSS* diatur oleh *W3C (World Wide Web Consortium).* Dengan menggunakan *CSS*, halaman yang tampak sama pada resolusi layar pengunjung akan berbeda – beda tanpa memerlukan tabel seperti yang ada pada html klasik. Halaman *HTML* atau X*HTML* pada web yang sama juga dapat ditampilkan secara berbeda, baik dari segi *style* tampilan atau skema warna pada *CSS* (Suryatiningsih dan Muhamad, 2009)

2.12 Unified Modeling Language (UML)

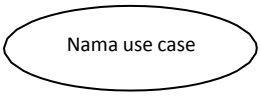
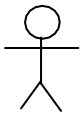
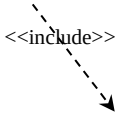
Unified Modeling Language (*UML*) merupakan bahasa spesifikasi standar yang digunakan sebagai dokumentasi, spesifikasi dan membangun sistem perangkat lunak. *Unified Modeling Language* (*UML*) adalah himpunan struktur dan teknik untuk pemodelan desain program aplikasi berorientasi objek (*OOP*). *UML* juga diartikan sebagai metodologi untuk mengembangkan sistem *OOP* dan sekelompok alat untuk mendukung pengembangan sistem tersebut. Terdapat empat jenis diagram yang paling sering digunakan untuk membangun aplikasi berorientasi objek yaitu *Use Case diagram, sequence diagram, collaboration diagram,* dan *class diagram* (Imbar dan Hartanto, 2011).

2.10.1 Use Case

Use Case merupakan cara formal yang digunakan untuk menggambarkan bagaimana sebuah sistem bisnis berinteraksi dengan lingkungannya. Sebuah *Use Case Diagram* digambarkan untuk mendeskripsikan visualisasi dari adanya interaksi antar pengguna atau aktor dengan sebuah sistem. *Use Case* juga digunakan untuk mengetahui fungsi – fungsi yang ada dalam sebuah sistem informasi dan pengguna atau aktor yang memiliki hak untuk menggunakan fungsi tersebut. *Use Case Diagram* memiliki beberapa simbol dalam penggunaannya:

Tabel 3 Simbol dalam *Use Case Diagram*

Simbol	Keterangan
Use Case	Fungsionalitas yang disediakan sistem sebagai unit – unit yang saling bertukar pesan antar unit atau aktor,

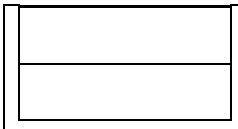
	dinyatakan dengan menggunakan kata kerja di awal frasa nama *Use Case*.
Aktor 	Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat itu sendiri.
Asosiasi	Komunikasi antar aktor dengan *Use Case* yang berpartisipasi pada *Use Case*.
Extensi 	Relasi *Use Case* tambahan ke sebuah *Use Case* di mana *Use Case* yang ditambahkan dapat berdiri sendiri walau tanpa *Use Case* tambahan itu, mirip dengan prinsip *inheritance* pada pemrograman berorientasi objek.
Generalisasi 	Hubungan generalisasi dan spesialisasi (umum – khusus) antara dua buah *Use Case* di mana fungsi salah satunya adalah fungsi umum.
Include 	Relasi *Use Case* tambahan ke sebuah *Use Case* di mana *Use Case* yang ditambahkan memerlukan *Use Case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *Use Case* ini.

2.10.2 Class Diagram

Class Diagram merupakan gambaran struktur sistem dari pendefinisian kelas – kelas yang akan dibuat untuk membangun sistem. Di mana kelas – kelas tersebut memiliki atribut dan metode atau operasi. Terdapat beberapa simbol pada *Class Diagram* yang digambarkan pada tabel di bawah ini:

Tabel 4 Simbol pada Class Diagram

Simbol	Keterangan
Kelas	Kelas pada struktur sistem.

	
Antarmuka/*interface*  nama_interface	Sama dengan konsep *interface* dalam pemrograman berorientasi objek.
Asosiasi 	Relasi antar kelas dengan makna umum, asosiasi 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/*dependency* 	Relasi antar kelas dengan makna kebergantungan antar kelas.
Agregasi 	Relasi antar kelas dengan makna semua bagian *(whole part)*.

2.10.3 Sequence Diagram

Sequence Diagram digunakan untuk menggambarkan perilaku objek pada *Use Case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. *Sequence* Diagram juga memiliki beberapa simbol yang digambarkan pada tabel berikut:

Tabel 5 Simbol dalam Sequence Diagram

Simbol	Keterangan

Aktor  Atau nama_aktor	Orang, proses atau sistem lain yang berinteraksi dengan sistem yang akan dibuat di luar sistem tersebut.
Garis Hidup/*Lifeline* 	Menyatakan kehidupan dari sebuah objek.
Objek Nama objek : nama kelas	Menyatakan objek yang berinteraksi pesan.
Waktu Aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi pesan.