BAB II

LANDASAN TEORI

2.1 Tinjauan Pustaka

Pada penelitian ini menggunakan tinjauan Pustaka dari penelitian sebelumnya yang nantinya akan digunakan sebagai pendukung penelitian, tinjauan Pustaka yang digunakan penulis dalam penelitian ini dapat dilihat pada Gambar 2.1.

Tabel 2.1 Tinjauan Pustaka

Nomor	**Detail Jurnal**	
1	Judul	Sistem Pencatatan dan Pengelolaan Keuangan Pada Aplikasi Manajemen Keuangan E-Dompet Berbasis *Android*
	Tahun Terbit	2019
	Penulis	Ujang Juhardi, Khairullah
	Metode Penelitian	Waterfall
	Analisis Masalah	Dalam penelitian ini disebutkan bahwa pencatatan pemasukan dan pengeluaran uang pribadi pada umumnya masih dilakukan dengan cara konvensional. hal tersebut sangat tidak efisien, jumlah transaksi yang tidak sedikit di setiap harinya membuat pencatatan, perhitungan, dan pembuatan laporan membutuhkan waktu yang tidak sedikit, maka dibutuhkan aplikasi pengelolaan keuang yang dapat digunakan

		seseorang untuk mempermudah pencatatan laporan keuangan.
	Hasil	Hasil dari penelitian ini adalah sebuah Aplikasi Manajemen Keuangan E-Dompet Berbasis *Android* yang dapat melakukan pencatatan laporan keuangan secara rinci sehingga memudahkan seseorang melihat rincian laporan keuangannya.
	Kelebihan	Penyimpanan data pada plikasi ini bersifat *offline* sehingga pengguna tidak harus menggunakan internet setiap menggunakan aplikasi.
	Kekurangan	Karena penyimpanan yang digunakan *offline* atau local, jika pengguna menggunakan *device* yang berbeda tidak dapat membuka catatan yang sudah ada sebelumnya.
2	Judul	Perancangan Aplikasi Kasir *Point of Sales* Berbasis *Android* Menggunakan Metode Rapid Application Development Untuk Usaha Retail
	Tahun Terbit	2020
	Penulis	Iskandar, Umar Tsani Abdurrahman
	Metode Penelitian	Rapid Application Development
	Analisis Masalah	Dalam penelitian ini disebutkan bahwa kegiatan transaksi pada gerai pada umumnya dilakukan

		belum menggunakan sistem komputer, sehingga banyak menemukan kendala, yaitu dalam proses penghitungan sering terjadi kesalahan hitung dan prose rekap transaksi yang relatif lama karena harus menghitung ulang setelah gerai selesai berjualan sehingga membutuhkan waktu extra.
	Hasil	Sebuah aplikasi kasir *point of sales* berbasis *android* yang dapat digunakan gerai dalam melakukan pencatatan transaksi dan dapat mempermudah kegiatan transaksinya.
	Kelebihan	Aplikasi dapat terhubung dengan printer melalui bluetooth sehingga dapat Melakukan pencetakan struk penjualan
	Kekurangan	*Interface* pada aplikasi dirasa kurang menarik dan pemilihan *color palette* yang kurang selaras atau terlalu kontras.
3	Judul	Rancang Bangun Aplikasi Pengelolaan Pinjaman Koperasi Berbasis *Mobile* Pada Koperasi Pkk Sejahtera Sukabumi
	Tahun Terbit	2018
	Penulis	Ovi Sovia Maranti, Lis Saumi Ramdhani, Rusli Nugraha, Khairul Rizal
	Metode Penelitian	Waterfall

	Analisis Masalah	Dalam penelitian ini disebutkan bahwa jumlah anggota yang cukup banyak pada koperasi PKK Sejahtera Sukabumi mengakibatkan sistem pelaporan, pembukuan simpan pinjam dan pengajuan pinjaman menjadi kurang efektif, karena masih menggunakan cara konvensional, terutama dalam pengajuan pinjam, hal tersebut menyebabkan perputaran modal berjalan lambat, oleh karena itu dibutuhkanya sistem terkomputerisasi agar anggota dapat mengajukan pinjaman kapanpun dan dimana saja.
	Hasil	Hasil dari penelitian ini yaitu aplikasi pengelolaan pinjaman koperasi berbasis *mobile* yang dibuat terintegrasi dengan web server supaya memudahkan dalam pengelolaan datanya.
	Kelebihan	Penyimpanan data pinjaman yang diintegrasikandengan web server dapat memudahkan dalam pengolahan data.
	Kekurangan	Halaman admin dan anggota dibuat pada platform yang berbeda sehingga tidak efisien dan user interface yang dirasa masih kurang baik.
	Judul	Pengembangan Aplikasi Klinik Kecantikan sebagai Pengelola Transaksi berbasis *Android*

4		menggunakan Metode Prototype (Studi Kasus: Klinik Kecantikan CV Nana Beautyskin).
	Tahun Terbit	2021
	Penulis	Agung Dwi Saputra, Agi Putra Kharisma, Lutfi Fanani.
	Metode Penelitian	Prototype
	Analisis Masalah	Dalam penelitian ini disebutkan bahwa kegiatan pencatatan transaksi yang dilakukan pada CV. Nana Beauty Skin masih dilakukan dengan cara tradisional, metode pencatatan tersebut masih dirasa efektif saat kegiatan transaksi masih sedikit, namun saat kegiatan transaksi yang mulai banyak hal tersebut memiliki beberapa resiko dalam pelaksanaanya. Sehingga dibutuhkan sistem pengelolaan data transaksi yang lebih baik dari sebelumnya.
	Hasil	Aplikasi klinik kecantikan sebagai pengelola transaksi berbasis *Android*, yang dapat digunakan untuk pencatatan setiap transaksi, pencatatan bonus pegawai pada setiap pelayanan jasa, dan pembuatan struk yang lebih efektif.
	Kelebihan	Penyimpanan yang sudah secara online, pengujian kompabilitas yang berjalan dengan baik di beberapa jenis *android*

	Kekurangan	Nilai pengujian usability belum mendapatkan nilai yang tinggi.
5	Judul	Sistem Pencatatan Keuangan Toko Berbasis *Android*
	Tahun Terbit	2019
	Penulis	Willi Alham Romadony, Muhammad Ardianto, Wisnu Kartiko Arie Pangestu, Didih Rizki Chandranegara, Wildan Suharso
	Metode Penelitian	Waterfall
	Analisis Masalah	pengelolaan keuangan dengan cara konvensional dirasa kurang efektif dan juga memerlukan waktu yang lebih untuk mencatat pengeluaran yang telah dilakukan saat itu sehingga terkadang lupa untuk melakukan pencatatan pengeluaran kecil yang telah dilakukan.
	Hasil	aplikasi pencatatan keuangan berbasis *android* yang dapat digunakan dalam mengelola keuangan.
	Kelebihan	Terdapat rekap harian yang sudah langsung terekap dan dapat di cetak
	Kekurangan	Tidak disebutkan penyimpanan yang digunakan dan juga UI yang masih kurang menarik.

Kesimpulan yang dapat diambil dari tinjauan literatur diatas yaitu masalah yang muncul karena pengelolaan data transaksi yang masih menggunakan cara

konvensional dapat teratasi dengan sistem yang terkomputerisasi dengan memanfaatkan aplikasi *mobile* berbasis *android* sehingga proses transaksi yang dilakukan akan lebih efisien. Perbedaan yang terdapat dari tinjauan literatur dengan penelitian penulis yaitu pada metode pengembangan sistem yang dipakai, pada penelitian yang akan diteliti penulis menggunakan metode *extreme programming* (XP), karena fleksibilitas yang tinggi terhadap *user* dan klien dalam pengembangannya. sehingga mampu membuat aplikasi yang sesuai dengan fungsi yang dibutuhkan klien (Widodo 2008). Kemudian terdapat juga perbedaan dalam penggunaan teknologi yang digunakan penulis dalam mengembangkan aplikasi berbasis *android*, dalam penelitian ini penulis menggunakan *Framework React Native*. Pada aplikasi yang akan peneliti kembangkan juga memanfaatkan fitur *QR-code scanner* yang berguna untuk menginputkan data pelanggan yang membuat pegawai tidak harus menginputkan data pelanggan setiap transaksinya, fitur lainnya yang akan ada pada aplikasi yaitu filter data untuk memudahkan pencarian data dan aplikasi akan dapat menampilkan *history* transaksi secara *realtime* supaya pimpinan dapat mudah mengetahui dan memantau transaksi yang terjadi kapan saja dan dimana saja. Kemudian dalam pengujian sistem penulis menggunakan pengujian ISO 25010.

1.2. Pengertian Open Source

Berdasarkan artinya jika diterjemahkan kedalam Bahasa Indonesia *open source* adalah sumber yang terbuka. Ide dari dari Open source sendiri timbul ketika seorang programmer dapat memodifikasi maupun mendistribusikan kembali kode program untuk bagian dari software maupun secara keseluruhan dan orang lain dapat melakukan peningkatan, penyesuaian ataupun perbaikan bug dari software

tersebut. Kode program pada *software opensource* diberikan secara '*free'* seringkali disalah artikan, maksud sebenarnya dari istilah *free* disini iyalah bebas oleh karena itu Europe FS melakukan perubahan istilah '*free'* menjadi '*libre'* sehingga kata "*Libre software*" lebih sering digunakan (Rakhmawati, 2006).

2.3 Pengertian *Android*

Android adalah sebuah *software* yang digunakan pada perangkat *mobile* yang mencakup sistem operasi, *middleware*, dan aplikasi kunci yang dirilis oleh Google.hal tersebut membuat *android* dapat mencakup keseluruhan dari suatu aplikasi, mulai dari pengembangan aplikasi sampai sistem operasi itu sendiri. Pengembangan aplikasi yang diterapkan pada *android*, menggunakan dasar bahasa pemrograman Java. Tapi secara sempit, *Android* biasanya mengacu pada sistem operasinya saja. Sistem operasi ini bersifat open source atau terbuka, sehingga pengembang bebas mengernbangkan atau mengembangkan aplikasi dengan biaya yang sedikit, dan pengembang dapat menjual aplikasi yang diciptakan tanpa ada lisensi ke produsen atau vendor tertentu. Kemudian pengembang diperbolehkan memodifikasi atau mengubah sistem operasi *android*. Dari beberapa faktor tersebut lah yang membuat *android* menjadi salah satu sistem operasi yang popular (Tim, 2015).

Gambar 2.1 Logo *android*

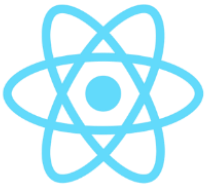
Sumber: (Dieter, 2019)

2.4 *Framework*

Framework merupakan sebuah kerangka kerja yang bertujuan untuk memudahkan *programmer* dalam pengembangan aplikasi dengan memanfaatkan library yang sudah terorganisir untuk dapat membuat suatu program lebih cepat (Solikin 2014). Dalam Bahasa Indonesia yaitu kerangka kerja dan dapat diartikan sebagai library yang bisa diturunkan, atau dapat digunakan fungsinya oleh modul modul atau fungsi yang akan dikembangkan. (Cahyati and Murti 2018). Berdasarkan definisi-definisi di atas dapat diambil kesimpulan bahwa *framework* adalah sistem yang terstruktur yang digunakan sebagai kerangka dalam mengembangkan sesuatu yang bertujuan untuk menyelesaikan suatu permasalahan atau isu – isu yang cukup kompleks lebih cepat.

2.5 *React Native*

React Native adalah salah satu *framework* yang digunakan untuk pengembangkan aplikasi *mobile* dan menggunakan Bahasa pemrograman JavaScript. Dengan menggunakan *framework React Native*, pengembang dapat merender *user interface* yang di peruntukan untuk digunakan pada platform iOS maupun *Android*. *React Native* juga merupakan *framework* yang bersifat *open source*, dan dalam penggunaannya dapat berjalan pada sistem operasi seperti Windows maupun macOS (Eisenman B. , 2016). Dalam pengembanganya juga, 75% kode program dapat digunakan jika ingin mengembangkan aplikasi ke *platform* lain tanpa harus menulis ulang kode (Hansson and Vidhall 2016). Selain itu berdasarkan data survey yang bersumber dari stack overflow, *react native* masih menjadi salah satu *framework multi-platform* yang paling popular di tahun 2022.



React Native

Gambar 2.2 Logo *React Native*

Sumber: (John3, 2019)

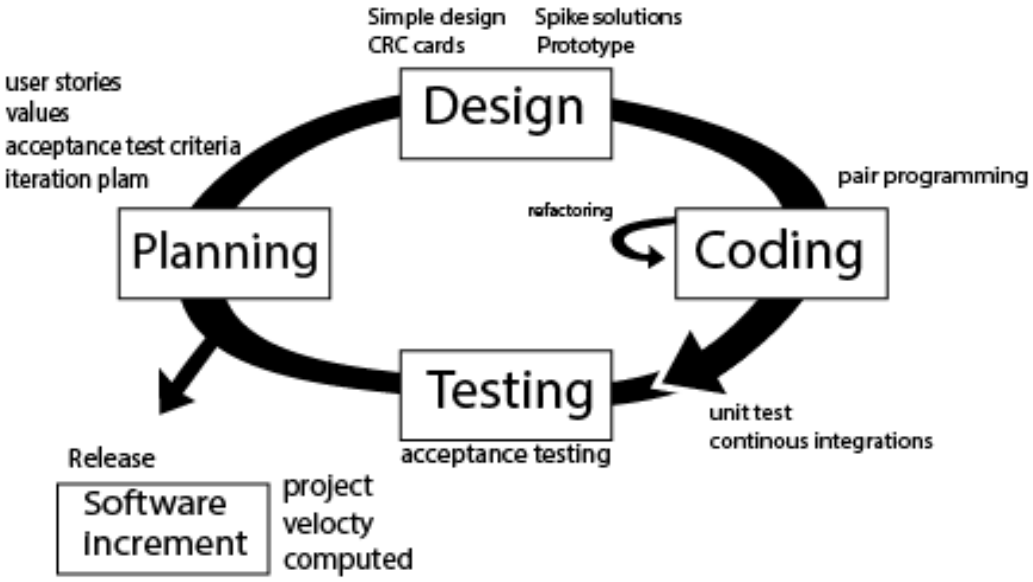
2.6 QR-*code*

QR code adalah jenis kode matriks atau kode batang dua dimensi dan dikembangkan oleh Denso Wave, salah satu divisi Denso Corporation yang merupakan perusahaan dari Jepang dan dipublikasikan pada tahun 1994. Sesuai dengan namanya, *quick response* atau respon cepat fungsi utama *QR-code* dapat dengan mudah dibaca oleh pemindai *QR-code*. tujuan adalah untuk diciptakannya *QR-code* yaitu untuk menyampaikan informasi dengan cepat dan mendapatkan respons yang cepat. Berbeda dengan kode batang, yang hanya menyimpan informasi secara *horizontal QR-code* dapat menampung informasi yang lebih banyak dari pada kode batang. (Ismail dkk., 2021).

2.7 Metode Pengembangan Sistem

Extreme Programming (XP) merupakan metodologi yang digunakan untuk pengembangan perangkat lunak yang ditujukan dalam meningkatkan kualitas perangkat lunak terhadap perubahan serta kebutuhan pelanggan. Jenis pengembangan perangkat lunak semacam ini dimaksudkan untuk meningkatkan produktivitas dan memperkenalkan pro pemeriksaan dimana persyaratan pelanggan baru dapat diadopsi. Ada beberapa tahapan yang ada pada *Extreme Programming*

yaitu terdiri dari Perencanaan (*Planning*) seperti memahami kriteria pengguna dan perencanaan pengembangan, designing seperti perancangan *prototype* dan tampilan, pengkodean juga termasuk dalam pengintegrasian, terakhir adalah testing (Ariyanti dkk., 2020). *Extreme Programming* adalah metode pengembangan perangkat lunak untuk menyederhanakan proses pengembangan sehingga lebih fleksibel, adaptif, dan dikerjakan oleh satu atau dua orang pengembang (Fatoni and Dwi 2016). Ada empat tahapan dalam pengembangan perangkat lunak menggunakan metode *extreme programming* (Pressman, 2005).



Gambar 2.3 Metode *Extreme Programming*

Sumber: (Pressman, 2005)

1. *Planning* (Perencanaan)

Pada tahapan ini merupakan tahapan yang diperlukan sebelum pengembang membuat sistem, tahapan ini penting karena ketika membuat sebuah sistem harus direncanakan atau dianalisis kebutuhan-kebutuhan yang diperlukan *user*. Dalam tahapan planning ini terdapat beberapa bagian yaitu:

a. *User Stories* yaitu Pengguna menggambarkan atau menceritakan permasalahan dan kebutuhan *user* yang diperlukan untuk sistem yang akan dibangun.

b. *Values* merupakan poin atau nilai-nilai yang dapat diambil dari *user* stories.

c. *AccepTabel test criteria* yaitu menentukan kriteria tes sebagai acuan terhadap kebutuhan sistem yang akan dibangun.

d. *Iteration plan* merupakan rencana untuk menentukan berapa kali peneliti melakukan pertemuan terhadap pengguna.

2. *Design* (Perancangan)

Setelah pada tahapan perencanaan selesai, maka tahapan selanjutnya adalah perancangan. Pada tahapan ini pengembang melakukan perancangan dengan membuat sebuah pemodelan, yang dimulai dari pemodelan sistem, kemudian pemodelan arsitektur, dan yang terakhir adalah pemodelan basis data. Dalam tahapan ini terdapat beberapa bagian yaitu:

a. *Simple design*: Pengembang mengembangkan perangkat lunak dengan desain yang sederhana.

b. *Spike solution* Jika dalam praktiknya desain yang dibuat sangatlah sulit. *Extreme programming* akan menggunakan spike solution dimana pembuatan design akan dibuat langsung ke tujuannya.

c. *CRC card* Digunakan untuk mengidentifikasikan dan mengorganisasikan object-oriented *class*es.

d. *Prototype*: Merupakan perancangan *user interface* biasanya dalam bentuk *wireframing* untuk mempermudah pengembang dan klien dalam melihat gambaran sistem

3. *Coding* (Pengkodean)

Tahapan ini merupakan tahapan untuk menerapkan pemodelan yang sudah dirancang di tahapan perancangan yang sudah dibuat kedalam bentuk *user interface* dan menggunakan bahasa pemrograman.

a. *Pair programming*: Dalam proses pengembangan terdapat dua orang *programmer*, dimana seorang *programmer* membuat *coding* dan *programmer* lainnya mengoreksi *code* yang dibuat.
b. *Refractory*: Merupakan tahapan yang dilakukan ketika terjadi ketidaksesuaian kode program kemudian dilakukan perbaikan untuk mendapatkan hasil yang diinginkan.

4. *Testing* (Pengujian)

Pada tahapan ini pengembang sistem melakukan pengujian terhadap sistem yang sudah dibuat untuk mengetahui kesalahan yang terdapat pada sistem dan untuk mengecek sistem telah dibuat sudah sesuai dengan kebutuhan pengguna atau belum.

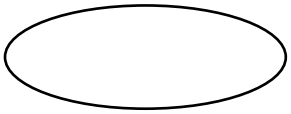
2.8 Unified Modeling Language (UML)

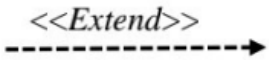
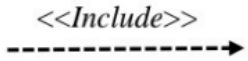
UML (*Unified Modelling Language)* adalah sebuah Bahasa berbentuk grafik atau gambar yang digunakan untuk memvisualisasikan sebuah sistem yang akan di kembangkan berbasis *Object-Oriented*. UML juga memiliki standar penulisan sebuah sistem blue print, meliputi kelas dalam Bahasa pemrograman yang spesifik, proses bisnis, skema *database* dan komponen lainya yang diperlukan sistem (Mubarak dkk., 2019). Terdapat tiga diagram UML yang memiliki fungsi masing – masing yaitu:

2.8.1 *Use Case*

Use Case adalah teknik untuk menggambarkan kebutuhan-kebutuhan fungsional dari sebuah sistem yang ingin di kembangakan atau sistem baru yang akan dibuat. Setiap *use case* memiliki satu atau lebih skenario yang menjelaskan bagaimana sebuah sistem akan berinteraksi dengan pengguna atau sistem lainya untuk tercapainya suatu sasaran bisnis tertentu. (Artina, 2006). Simbol – simbol yang terdapat pada *use case* dapat dilihat pada tabel 2.2.

Tabel 2.2 Simbol-Simbol *Use Case*

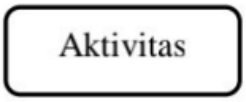
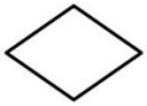
Simbol	Keterangan
	*Use Case:* Fungsionalitas yang disediakan sistem, Biasanya diawali dengan kata kerja sebelum farse nama *use case*.
	Aktor: pengguna, proses atau sistem lain yang berinteraksi dengan sistem yang akan di kembangkan, walau simbol berbentuk orang aktor belum tentu orang, nama aktor biasanya menggunakan kata benda
	Asosiasi: sebuah komunikasi antar aktor dan *use case* digunakan saat *use case* memilikiinteraksi dengan aktor

	Generalisasi: hubungan umum sampai khusus antara dua buah *use case* di mana salah satu use memiliki fungsi yang lebih umum.
	*Extend*: menunjukan *use case* tambahan dapat berdiri sendiri walau tanpa *use case* tambahan.
	*Include*: Menunjukan bahwa *use case* tambahan akan dipanggil saat *use case* tambahan dijalankan.

2.8.2 *Activity Diagram*

Activity diagram merupakan diagram yang digunakan dalam pengembangan sistem untuk menjelaskan secara prosedural alur proses sebuah sistem agar mempermudah developer dalam proses pengembangan aplikasi. Pada diagram ini memungkinkan mengevaluasi adanya lebih dari satu jalur yang terbentuk dan berjalan beriringan. pembuatan *activity diagram* dimulai dari *initial node* dan diakhiri dengan *end node*. *initial node* dalam sebuah *activity diagram* diperbolehkan lebih dari satu, hal ini berguna untuk mengakomodasi jika sistem yang dikembangkan akan memiliki input lebih dari satu (Ayu, 2017). Berikut merupakan simbol-simbol yang terdapat pada *activity diagram* yang dapat dilihat pada Tabel 2.3.

Tabel 2.3 Simbol – Simbol *Activity Diagram*

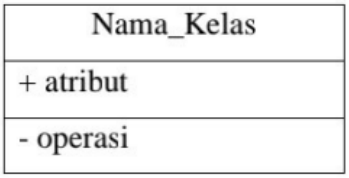
Simbol	Keterangan
	Status awal: simbol yang memiliki fungsi menunjukan status awal dari sebuah diagram
	Aktivitas: simbol yang memiliki fungsi menunjukan aktivitas yang dilakukan sistem, biasanya diawali dengan kata kerja.
	*Decision*: asosiasi percabangan, dimana jika ada pilihan aktivitas lebih dari satu
	*Joint*: asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
	Status akhir: simbol yang menunjukan aktivitas akhir diagram

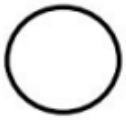
2.8.3 *Class diagram*

Class diagram adalah sebuah diagram UML yang menggambarkan hubungan antar kelas dan memiliki penjelasan detail dari sebuah sistem di dalam model desain, dan juga memperhatikan entitas perilaku sistem. Pada *class diagram* memiliki atribut-atribut dan operasi yang dimiliki dari sebuah kelas dan constraint yang berhubungan dengan objek yang dikoneksikan. Komponen-komponen yang

terdapat pada *class diagram* yaitu *class*, relasi, assosiasi, generalisasi dan agregasi, attribut, operasi, dan visibilitas, tingkat akses objek eksternal pada suatu operasi. Juga hubungan antar kelas disebut *multiplicity* dan *card*inality (Hendini 2016). Adapun simbol-simbol yang terdapat pada *class diagram* dapat dilihat pada Tabel 2.4.

Tabel 2.4 Simbol – Simbol *Class Diagram*

Simbol	Keterangan
	Menggambarkan kelas yang terdapat pada struktur
	Asosiasi berarah: relasi antar kelas yang memiliki arti kelas satu digunakan oleh kelas yang lain dan biasanya disertai dengan *multiplicity*.
	Generalisasi: relasi antar kelas yang memiliki makna umum khusus
	Ketergantungan: *dependency* merupakan relasi antar kelas dengan arti ketergantungan antar kelas.
	Agregasi: merupakan relasi antar kelas dengan arti semua bagian

	Antar muka: yaitu menunjukan interface hal ini sama dengan konsep interface pada pemrograman berorientasi objek.
	Asosiasi: merupakan relasi antar kelas dengan makna umum dan biasanya disertai multiplicity.

2.9 Class Responsibility Collaborator Card (CRC Card).

Menurut Zulhalim (2018), *Class Responsibility Collaborator Card* (*CRC Card*) adalah suatu teknik yang bertujuan untuk mengecek adanya interaksi suatu objek, mengenali, dan menentukan *class* yang diperlukan. Pada *CRC Card* terdapat *Class Name* yaitu sebagai suatu kelas yang dijabarkan, *Responsibilities* yang berfungsi sebagai tugas dan fungsi dari kelas tersebut, dan *Collaborators* berfungsi sebagai objek atau kelas yang berkaitan dengan kelas tersebut. Berikut ini merupakan gambar bagian-bagian *CRC Card*.

Tabel 2.5 Class Responsibility Collaborator Card (CRC Card)

Class Name	
Responsibilities	***Collaborators***

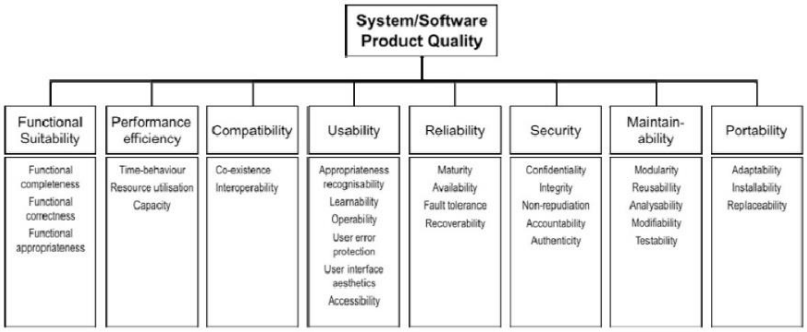
2.10 Firebase

Firebase adalah penyedia layanan *cloud* dengan *back-end* sebagai servis yang berbasis di San Fransisco, California. Firebase membuat sejumlah produk untuk pengembangan aplikasi *mobile* ataupun web. Firebase didirikan oleh Andrew Lee

dan James Tamplin pada tahun 2011 dan diluncurkan dengan *cloud database* secara *realtime* di tahun 2012 (Kumala & Winard, 2020).

2.11 Pengujian ISO 25010

Menurut (Fadli H. Wattiheluw, 2019) Pengujian ISO 25010 merupakan bagian dari *Systems and Software Quality Requirements and Evaluation* (SQuaRE) yang merupakan versi lanjutan dari ISO 91261, yang telah direvisi secara teknis dengan menambahkan beberapa struktur dan bagian dari standar model kualitas. Tujuan dari penggunaan kualitas ini adalah untuk mengukur sejauh mana produk atau sistem tersebut bisa digunakan oleh pengguna untuk memenuhi kebutuhan dalam mencapai tujuan yang diinginkan dengan efisiensi, efektivitas, kepuasan dalam konteks penggunaan yang spesifik, dan bebas dari resiko.Menurut (Harun, 2018) ISO 25010 terdiri dari delapan karakteristik yang dibagi menjadi beberapa bagian yang berhubungan dengan sifat-sifat statis perangkat lunak dan sifat dinamis dari sistem komputer, yang dapat dilihat pada Gambar 2.4



Gambar 2.4 Model ISO 25010

Sumber: (Harun, 2018)

Berdasarkan gambar diatas, dapat dijelaskan mengenai delapan karakteristik tersebut, sebagai berikut:

1. *Functional Suitability*, produk yang memberikan fungsional untuk memenuhi kebutuhan saat sistem atau produk tersebut digunakan pada keadaan tertentu.
2. *Reliability*, sistem dapat mempertahankan kinerjanya pada level tertentu ketika digunakan pada keadaan tertentu.
3. *Performance Efficiency*, sistem menyediakan performa yang baik dengan sejumlah *resource* yang akan digunakan pada sistem atau produk.
4. *Usability*, sistem atau produk mudah dimengerti, mudah dipakai, dan menarik untuk digunakan.
5. *Security*, sistem menyediakan layanan untuk melindungi akses, ataupun pengungkapan yang berbahaya.
6. *Compatibility*, merupakan kemampuan pada suatu komponen atau sistem untuk bertukar informasi.
7. *Maintainability*, merupakan tingkat suatu sistem dapat dimodifikasi, perbaikan, pengembangan untuk menyesuaikan dengan lingkungan, modifikasi pada kriteria, dan spesifikasi fungsi.
8. *Portability*, sistem dapat dipindahkan dari satu ruang ke ruang lainnya.

Berdasarkan dari kebutuhan dalam penelitian ini, dalam pengujian aplikasi *mobile* menggunakan empat karakteristik yang digunakan yaitu *functional suitability*, *compatibility*, *usability*, dan *performance efficiency*. (David, 2011)

2.12 Pengertian Figma

Figma adalah *software* yang digunakan untuk pembuatan desain berbasis cloud dengan konsep fungsionalitas sketch, dan dapat melakukan kolaborasi antar desainer lainnya. figma dapat bekerja pada berbagai sistem operasi karena dapat berjalan pada browser, fitur slack yang terdapat pada figma sebagai media

komunikasi dari satu designer ke designer lainnya dapat memudahkan suatu tim dalam berkomunikasi secara *realtime* dalam melakukan editing, keunggulan lain yaitu pengguna dapat dengan mudah mendapatkan kode dalam bentuk css sehingga memudahkan front end dalam pengimplementasian desain yang ada (Nugraha dkk., 2020)