BAB 2
TINJAUAN PUSTAKA

2.1 Dart

Dart adalah pemrograman yang dapat digunakan secara prosedural,fungsional dan pemrograman berbasis objek oriented [6].Bahasa pemrograman dart dapat dikatakan sebagai bahasa pemrograman yang dapat dengan mudah dipelajari karna sifatnya yang open source sehingga siapapun dapat mengaksesnya dan mempelajarinya.Bahasa pemrograman dart juga bersifat fleksibel karena dapat digunakan di berbagai macam *platform*.Apabila kebutuhannya untuk pengembangan mobile bahasa dart bisa digunakan dengan objek *oriented* sementara untuk web bahasa dart bisa digunakan secara prosedural.

2.2 Android Studio

Android studio merupakan sebuah *Intregeted Development Environment* (IDE) untuk platform android. *Android Studio* ini diumumkan pada tanggal Mei 2013 pada Google I/O oleh produk Manager Google , Ellie Powers .Versi pertama *android* adalah 0.1 yang dirilis pada Mei 2013 , dan memasuki tahap beta sejak versi 0.8 dirilis pada juni 2014 . Versi stabil yang pertama kali dirilis adalah versi 1.0 sejak Desember 2014 . Sedangkan versi stabil saat ini adalah versi 3.6.1 per-Maret 2020.

2.3 Flutter

Flutter adalah toolkit UI *portable* Google yang dikompilasi secara native untuk mobile,web dan desktop dari basis kode tunggal [7].Flutter sendiri dikembangkan dengan menggunakan bahasa pemrograman DART.Flutter biasanya digunakan untuk mendevelop aplikasi *hybrid* dimana aplikasi tersebut dapat berjalan di *Android* maupun IOS namun flutter juga mampu mengembangkan aplikasi untuk web dan *desktop*.Flutter SDK sudah tersedia di kode editor *visual studio code* dan *android studio* untuk pengguna sistem operasi windows serta di X code untuk sistem operasi macOS

2.4 Code Refractoring

Code Refractoring atau *Reenginering* merupakan proses untuk membentuk ulang sebuah kode yang sudah ada tanpa mengubah fungsional dari kode itu sendiri [5].Misalkan disebuah ada sebuah program perhitungan angka ganjil pada sebuah array maka output sebuah program perhitungan yang disudah direfraktor haruslah program perhitungan angka ganjil juga.Refraktor kode juga biasa dilakukan untuk mengubah sebuah sebuah program dengan bahasa pemrograman A dijadikan program menggunakan dengan bahasa pemrograman B yang mana umumnya kerap terjadi banyak untuk migrasi sebuah aplikasi desktop ke android maupun sebaliknya.Selain mengubah sebuah algoritma pada program yang ada misalkan diprogram sebelumnya algoritma yang digunakan adalah algoritma C45 dan akan dirubah ke algoritma ID3 ini merupakan bentuk kode refraktoring juga karena kedua algoritma memiliki tujuan yang sama yaitu untuk menentukan pohon keputusan.

2.5 Design Pattern

Design pattern atau pola desain adalah solusi umum yang bisa digunakan kembali untuk menyelesaikan masalah umum tertentu yang ditemui dalam sebuah desain program [1].Kita misalkan saja design pattern sebagai sebuah resep masakan.Setiap resep masakan pasti tentunya memiliki beberapa syarat tertentu seperti bahan-bahan apa yang dibutuhkan dan bagaimana langkah-langkanya.Analogikan kita akan memasak sebuah mie instan tentu ada beberapa cara untuk memasaknya,yang pertama kita bisa merebus mienya dahulu dan mencampur bumbunya dimangkuk atau kita bisa mencampurkan bumbunya langsung dengan mie yang ada di wajan.Walapun langkah yang dilakukan berbeda namun hasil dari kedua hal yang dilakukan itu adalah sebuah mie yang siap disajikan seperti itu jugalah *design pattern* kita sudah mendapatkan resep tapi langkah implementasi terserah kepada kita selama itu menghasilkan *output* yang sama.

2.6 BLOC

BLOC atau atau *Business Logic Component* adalah sebuah *design pattern* untuk flutter yang membantu memisahkan presentation dan business logic [2].*Presentation* pada bloc menghasilkan sebuah *events* atau interaksi seperti pada saat tombol diklik atau halaman diload.Kemudian BLOC akan menghasilkan output yaitu berupa sebuah *state* tergantung interaksi yang sudah dilakukan sebelumnya.Dengan memisahkan *logic* dan *presentation maintenance* kode tentunya akan lebih mudah karena kita hanya akan merubah interaksi pada satu sisi saja.Sebagai contoh jika kita ingin merubah events atau interaksi yang terjadi kita bisa mengubahnya dibagian presentation saja tanpa perlu melakukan perubahan pada bagian logicnya.

2.7 Interoperability

Interoperability atau interoperabilitas adalah cara ideal bagi computer dan perangkat lainnya untuk saling berhubungan [8].Kita misalkan saja sebuah aplikasi untuk mendengarkan lagu *online* yaitu Spotify.Adapun fitur di Spotify dimana kita bisa mengoperasikan lagu yang kita dengar dari *smartphone* untuk dikoneksikan ke laptop atau komputer.Sehingga suara yang tadinya dihasilkan oleh *smartphone* tersebut dialihkan ke laptop atau *computer* begitu pula sebaliknya.Hal ini tentunya bisa sebuah *interoperability* dimana adanya interaksi lebih dari satu device.

2.8 Maintainability

Maintainability dapat didefinisikan sebagai fasilitas yang memungkinkan sebuah sistem ataupun peralatan dapat dipelihara atau diperbaiki sehingga layanan dapat beroperasi kembali [9].Pada dunia IT *maintainability* lebih berfokus bagaimana perawatan sebuah perangkat lunak ataupun aplikasi yang dibuat dapat berjalan dengan baik untuk kedepannya.Penerapan *clean code* dan *design pattern* merupakan salah satu bagian dari *maintainability* karena dengan menerapkan kedua hal tersebut kode sumber yang dibuat tentunya akan lebih mudah dibaca ketimbang tidak menerapkannya sama sekali sehingga membuat *engineer* yang

bekerja di bagian

maintenance ataupun update aplikasi di masa yang akan datang akan lebih mudah dilakukan.

2.9 Material Design

Material design adalah desain dengan peningkatan penggunaan *layout* berbasis grid,animasi dan transisi responsif,padding serta efek kedalaman seperti pencahayaan atau bayangan [10]. *Material design* ini sendiri adalah desain yang dikembangkan oleh google yang dapat digunakan di android versi 2.1 keatas.*Material design* bukan untuk mengubah interface secara keseluruhan namun mewujudkan visualisasi agar menyerupai bentuk myata baik dalam teksture,bayangan dan pencahayaan.*Material design* biasanya menggunakan warna solid tanpa gradient,dan hanya menggunakan efek pencahayaan sederhana namun terkesan elegan.

2.10 Cupertino

Cupertino adalah design khusus untuk platform IOS.Di flutter ada *widget* khusus yang meggunakan *style* Cupertino yang dinamanakan *CupertinoWidget [11].Widget-widget* ini sendiri terdiri diri *CupertinoApp*, *CupertinoPageScaffold*, *CupertinoButton*,*CupertinoTextfield*,*CupertinoDialog* dan lain-lain.

2.11 Black Box Testing

Blackbox Testing adalah metode pengujian perangkat lunak yang memeriksa aplikasi tanpa mengintip struktur atau cara kerjanya [12].Metode pengujian ini dapat diterapkan secara virtual ke tiap tingkat pengujian perangkat lunak.Pengetahuan khusus tentang kode dan pengetahuan pemrograman secara umum tidak diperlukan dalam pengujian ini.penguji tahu apa yang dapat dilakukan oleh perangkat lunak yang sedang dia uji.Namun biasanya penguji tidak dapat mengetahui bagaimana sebuah program dapat menghasilkan output tertentu.Kasus pengujian biasanya berasal dari deskripsi eksternal perangkat lunak itu sendiri.Untuk beberapa kasus penguji membuat scenario *input valid* dan *input* tidak *valid* untuk mengetes sebuah fungsional yang ada di aplikasi.

2.12 UML(Unified Modelling Language)

Unified Modeling Language (UML) adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membanngun perangkat lunak[17]. UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem.Secara filosofi UML di ilhami oleh konsep yang sudah ada yaitu pemodelan *Object Oriented* karena konsep ini menganologikan sistem seperti kehidupan nyata yang didominasi oleh obyek dan digambarkan atau dinotasikan dalam simbol-simbol yang cukup spesifik.UML sendiri terdiri dari beberapa macam diagram diantaranya *Usecase Diagram,Class Diagram,Activity Diagram,Sequence Diagram,Package Diagram*

2.13 Maintainability Index

Maintainability Index merupakan sebuah metrik yang meengukur perangkat lunak untuk mengembangkan perangkat lunak tersebut. Maintainability Indk (MI) memudahkan pengembangan jika kode sumbernya hendak dirubah. Pada umumnya. indeks pada MI dihitung terdiri dari *Lines of Code, Cyclomatic Complexity dan Halstead volume*. Namun, pada perangkat lunak android, ada beberapa spesial fitur spesial yang tidak terdapat pada OO. Oleh karena itu, indeks matriks yang ditentukan untuk perhitungan pada MI untuk Android berbeda. Berikut ini adalah metriks tersebut:

Tabel 2.1 Metriks dan Definisinya

Metriks	Definisi
CL_FUN	Jumlah total function yang terdapat di kelas ini
IN_BASES	Jumlah kelas yang mewarisi kelas ini
CU_CDUSED	Jumlah kelas yang digunakan langsung oleh kelas ini
CL_CMOF	Jumlah baris dalam kode
CL_STAT	Jumlah statement yang dapat dieksekusi
CL_DATA	Jumlah Total atribut yang ada pada kelas ini
CL_FUN_PU	Jumlah total public function yang ada pada

B	kelas ini

CU_CDUSERS	Jumlah kelas lain yang menggunakan kelas ini secara langsung
CL_DATA_PUB	Jumlah total atribut public yang terdapat pada kelas ii
IN_NOC	Jumlah anak dari kelas ini
NOP	Jumlah permission yang digunakan pada kelas ini<uses-permission>
CL_WMC	Weight method/class

Sedangkan untuk rumus menghitung MI nya sebagai berikut :

NOP + IN_NOC + IN_BASES + 2*CL_FUN + CL_CMOF + CL_DATA + CL_STAT + CL_FUN_PUB + CU_CDUSERS + CL_DATA_PUB + 2 *CU_CDUSED + 2*CL_WM

Sementara berdasarkan jurnal penelitian[19].Beginilah Kriteria MI yang untuk menunjukkan tingkat maintainability sebuah aplikasi.

Tabel 2.2 Maintainability Index Scale

Nilai Maintainbility Index	Klasifikasi
MI > 85	*Highly Maintainable*
MI 65 ≤ 85	*Moderately Maintanable*
MI < 65	*Difficult to Maintain*