PERANCANGAN APLIKASI *QR CODE GENERATOR* DAN *QR CODE READER* UNTUK ANDROID

TUGAS AKHIR
Diajukan untuk Memenuhi Salah Satu Syarat Kelulusan
Program Pendidikan Sarjana

Oleh :
Evi Hariyanti
2011130021

JURUSAN TEKNIK INFORMATIKA
SEKOLAH TINGGI MANAJEMEN INFORMATIKA & KOMPUTER LIKMI
BANDUNG
2016

PERANCANGAN APLIKASI *QR CODE GENERATOR* DAN *QR CODE READER* UNTUK ANDROID

Oleh:
Evi Hariyanti
2011130021

Bandung, 4 Mei 2016
Menyetujui,

Dr. Hery Heryanto S.Kom., M.Kom.	Dhanny Setiawan, S.T., M.T.
Pembimbing	Ketua Jurusan

JURUSAN TEKNIK INFORMATIKA
SEKOLAH TINGGI MANAJEMEN INFORMATIKA & KOMPUTER - LIKMI
BANDUNG
2016

ABSTRAK

Salah satu teknologi yang memiliki pengaruh yang sangat besar dan sudah dikenal cukup luas adalah *barcode* satu dimensi. Para pengembang mengambil keuntungan teknologi moderen untuk mengembangkan *barcode* menjadi dua dimensi. Keunggulan *barcode* dua dimensi dibandingkan dengan *barcode* satu dimensi adalah *barcode* dua dimensi dapat menyimpan data yang jauh lebih besar daripada *barcode* satu dimensi. Pada umumnya QR Code yang digunakan saat ini masih berwarna hitam dan putih. Penulis membangun *QR Code Generator* yang menghasilkan *QR Code* yang memiliki warna gradasi dan *QR Code Reader*.

Dalam *QR Code Generator* dijelaskan tentang *Reed-Solomon Codes* untuk menghitung *Error Correction*, *data masking* untuk mengatur pola menggunakan hitungan, dan pengolahan citra untuk gradasi warna. Proses pengembangan yang digunakan adalah metode *prototype* dengan analisis yaitu *Unified Modeling Language.* Pembuatan aplikasi dilakukan dengan menggunakan perangkat lunak Eclipse yang berbasis bahasa Java Android.

Aplikasi yang dibangun memiliki fitur atau spesifikasi yaitu dapat meng-*generate* teks menjadi QR Code sesuai dengan level *Error Correction* yang telah dipilih, hasil QR Code yang telah di-*generate* dapat disimpan menjadi sebuah data gambar, warna pada QR Code dapat dirubah menjadi gradasi, dapat membaca QR Code dari sebuah data gambar, dapat membaca QR Code melalui pemindaian menggunakan kamera, dan hasil pembacaan QR Code dapat ditampilkan. Analisis dari *QR Code Generator* dan *QR Code Reader* dijelaskan menggunakan *use case diagram*, *activity diagram*, *class diagram* dan *sequence diagram*. Pada perancangan antar muka terdiri dari tampilan menu utama, menu *Generate QR Code*, dan menu *Read QR Code*.

Pengujian perangkat lunak menggunakan *black box testing* dan pengujian antar muka. Aplikasi ini berhasil dibuat dengan menggunakan *library* ZXING. Aplikasi ini berhasil meng-*generate* QR Code dari hasil *input* angka atau huruf atau gabungan angka dan huruf, dan hasil QR Code tersebut pun dapat dibaca dengan menggunakan *QR Code Reader*. QR Code yang dihasilkan berhasil diubah warnanya menjadi warna gradasi serta aplikasi ini pun digunakan pada telepon genggam bersistem operasi Android.

KATA PENGANTAR

Puji dan syukur penulis panjatkan kepada Tuhan Yesus Kristus karena atas berkat, kasih serta anugerah yang diberikan-Nya kepada penulis sehingga dapat diselesaikannya penyusunan Tugas Akhir ini tepat pada waktunya.

Penulis telah memberikan kemampuan yang terbaik dalam penyusunan Tugas Akhir ini, namun penulis menyadari bahwa dalam penyusunan Tugas Akhir ini masih terdapat banyak kekurangan. Unuk itu penulis memohon maaf atas segala kekurangan yang ada dalam Tugas Akhir ini. Kiranya di balik semua kekurangan yang ada, Tugas Akhir ini bisa memberikan manfaat kepada seluruh pembaca Tugas akhir ini.

Penulis mengucapkan banyak terimakasih kepada berbagai pihak yang telah memberikan bimbingan, saran, kritik, dan dukungan sehingga penulis dapat menyelesaikan Tugas Akhir ini. Penulis mengucapkan banyak terima kasih kepada:

1. Bapak Dr. Hery Heryanto, S.Kom., M.Kom. yang telah meluangkan waktu untuk membimbing, mengoreksi, memberi saran, serta meluangkan waktu dan tenaga untuk membantu penulis menyelesaikan Tugas Akhir ini.
2. Bapak Dhanny Setiawan, S.T., M.T. sebagai ketua jurusan Teknik Informatika yang telah banyak membantu proses akademis dan memberi banyak ilmu kepada penulis selama menempuh kuliah di STMIK LIKMI.
3. Dosen-dosen STMIK LIKMI yang telah memberikan ilmu dan pengajaran kepada penulis selama menempuh pendidikan di STMIK LIKMI.
4. Staff-staff kampus yang telah memberi banyak bantuan selama penulis berkuliah di STMIK LIKMI.
5. Keluarga penulis atas doa dan segala dukungannya yang membuat penulis bisa menyelesaikan penyusunan tugas akhir ini.

6. Sahabat-sahabat terbaik Sandy Cahya, S.T., Holong Marisi Simalango, S.T., Maria Oktaviana, S.T., Sianly Sugiani, S.T., Michael Alvin, S.T., dan Asbian Wangsadijaya, S.T. yang selalu menyemangati penulis dan memberikan dukungan tanpa henti.
7. Teman-teman jurusan Teknik Informatika angkatan 2011 yang selama kurang lebih empat tahun ini berjuang bersama-sama dalam mencapai kelulusan.
8. Teman-teman mahasiswa/i STMIK LIKMI baik yang berbeda jurusan maupun angkatan yang selalu memberikan dukungan dan penghiburan.
9. Pihak lain yang membantu yang tidak dapat disebutkan namanya satu persatu.

Penulis menerima setiap saran maupun kritik membangun yang ditujukan baik kepada penulis maupun isi dari Tugas Akhir ini. Sehingga di lain kesempatan penulis dapat melakukan yang lebih baik lagi.

Bandung, 4 Mei 2016

Penulis

DAFTAR ISI

ABSTRAK i

KATA PENGANTAR ii

DAFTAR ISI iv

DAFTAR GAMBAR vi

DAFTAR TABEL vii

DAFTAR SIMBOL viii

BAB I PENDAHULUAN 1

1.1 Latar Belakang Masalah 1

1.2 Rumusan Masalah 2

1.3 Tujuan 2

1.4 Batasan Masalah 3

1.5 Kegunaan Hasil 3

1.6 Metodologi Penelitian 3

1.7 Sistematika Penulisan 4

BAB II LANDASAN TEORI 6

2.1 Quick Response Code (QR Code) 6

2.1.1 Fungsi QR Code 7

2.1.2 Kelebihan QR Code 8

2.1.3 Struktur QR Code 10

2.1.4 Bentuk QR Code 11

2.1.5 *Symbol Version* 12

2.2 Reed-Solomon Codes 12

2.3 Data Masking 13

2.4 Pengolahan Citra 15

2.5 Rekayasa Perangkat Lunak 18
2.6 Unified Modeling Language (UML) 22
2.7 Object Oriented Programming 23
2.8 Eclipse 25
2.9 Android 25
2.10 Java 27
BAB III ANALISIS DAN PERANCANGAN 29
3.1 Gambaran Umum Aplikasi 29
3.2 Spesifikasi Kebutuhan Perangkat Lunak 30
3.3 Pemodelan Perangkat Lunak 31
3.3.1 *Use Case Diagram* 31
3.3.2 *Use Case Descriptions* 31
3.3.3 *Activity Diagram* 38
3.3.4 *Class Diagram* 40
3.3.5 *Sequence Diagram* 41
3.4 Rancangan Antar Muka 43
BAB IV IMPLEMENTASI DAN PENGUJIAN PERANGKAT LUNAK 46
4.1 Implementasi 46
4.1.1 Lingkungan Pengembangan 46
4.1.2 Tampilan Antar Muka 47
4.2 Pengujian 53
BAB V KESIMPULAN DAN SARAN 60
5.1 Kesimpulan 60
5.2 Saran 60
DAFTAR PUSTAKA 62
LAMPIRAN LISTING PROGRAM 64

DAFTAR GAMBAR

Gambar 2.1 Contoh *Barcode* Satu Dimensi 6

Gambar 2.2 Contoh QR Code 7

Gambar 2.3 Struktur QR Code 10

Gambar 2.4 Bentuk QR Code 11

Gambar 2.5 *Simbol Version* QR Code 12

Gambar 2.6 *Mask Pattern* 13

Gambar 2.7 Matriks Citra Digital 15

Gambar 2.8 Model Pengembangan *Prototype* 21

Gambar 3.1 *Block Diagram* 29

Gambar 3.2 *Use Case Diagram* 31

Gambar 3.3 Activity Diagram Generate QR Code 38

Gambar 3.4 *Activity Diagram Read* QR Code 39

Gambar 3.5 *Class Diagram* 40

Gambar 3.6 *Sequence Diagram* Menu *Read* QR Code 41

Gambar 3.7 *Sequence Diagram* Menu *Generate* QR Code 42

Gambar 3.8 Rancangan Menu Utama 43

Gambar 3.9 Rancangan Menu *Generate* QR Code 43

Gambar 3.10 Rancangan Menu *Read* QR Code 44

Gambar 4.1 Tampilan Menu Utama 47

Gambar 4.2 Tampilan Menu *Generate* QR Code 48

Gambar 4.3 Tampilan Menu *Generate* QR Code setelah meng-*generate* QR Code 49

Gambar 4.4 Tampilan Menu *Generate* QR Code dengan palet warna ditampilkan 49

Gambar 4.5 Tampilan Menu *Generate* QR Code setelah warna *background* dipilih 50

Gambar 4.6 Tampilan Menu *Generate* QR Code setelah warna *foreground* dipilih 51

Gambar 4.7 Tampilan Menu *Generate* QR Code setelah warna *foreground* 1 dipilih 51

Gambar 4.8 Tampilan Menu *Read* QR Code 52

Gambar 4.9 Tampilan Menu *Read* setelah pembacaan dilakukan 52

Gambar 4.10 Tampilan Menu *Read* setelah pemindaian dilakukan 53

DAFTAR TABEL

Tabel 3.1 Skenario normal : *Generate* QR Code 31

Tabel 3.2 Skenario alternatif I : *Generate* QR Code 32

Tabel 3.3 Skenario alternatif II : *Generate* QR Code 32

Tabel 3.4 Skenario alternatif III : *Generate* QR Code 32

Tabel 3.5 Skenario alternatif IV : *Generate* QR Code 33

Tabel 3.6 Skenario alternatif V : *Generate* QR Code 33

Tabel 3.7 Skenario alternatif VI : *Generate* QR Code 33

Tabel 3.8 Skenario alternatif VII : *Generate* QR Code 34

Tabel 3.9 Skenario alternatif VIII: *Generate* QR Code 34

Tabel 3.10 Skenario alternatif IX : *Generate* QR Code 35

Tabel 3.11 Skenario alternatif X : *Generate* QR Code 35

Tabel 3.12 Skenario alternatif XI: *Generate* QR Code 36

Tabel 3.13 Skenario normal : *Read* QR Code 36

Tabel 3.14 Skenario alternatif I : *Read* QR Code 36

Tabel 3.15 Skenario alternatif II : Read QR Code 37

Tabel 3.16 Skenario alternatif III : *Read* QR Code 37

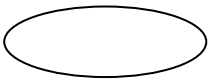
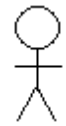
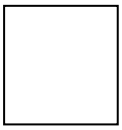
Tabel 4.1 Tabel skenario pengujian menu utama 53

Tabel 4.2 Tabel skenario pengujian *generate* QR Code 54

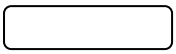
Tabel 4.3 Tabel skenario pengujian *read* QR Code 58

DAFTAR SIMBOL

Use case diagram

Nama Simbol	Simbol	Keterangan
Use Case		Sebuah kebiasaan atau fungsi utama (*key behaviour*) yang dilakukan oleh sistem perangkat lunak
Asosiasi		Hubungan antara aktor dengan *use case* yang menunjukan adanya interaksi aktor dengan sistem atau campur tangan aktor dalam suatu *use case*
Actor		Entitas eksternal yang berhubungan dengan sistem
System boundary		Menspesifikasikan paket yang menampilkan sistem secara terbatas.

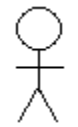
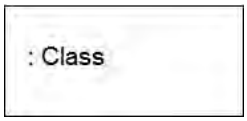
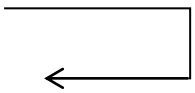
Activity diagram

Nama Simbol	Simbol	Keterangan
Kondisi awal		Menunjukkan kondisi awal kegiatan sistem
Kondisi akhir		Menunjukkan kondisi berakhirnya kegiatan sistem
Action State		Kegiatan yang sistem lakukan
Desicion		Percabangan dimana sistem harus melakukan salah satu cabang keputusan
Transition		Menunjukkan arah sebuah *action state* ke *action state* selanjutnya

Class diagram

Nama Simbol	Simbol	Keterangan
Class		Menunjukan nama kelas yang terdapat pada perangkat lunak beserta dengan attribute dan method yang ada di dalamnya.
Directed Association		Relasi antar *class* dengan makna *class* yang satu digunakan oleh *class* yang lain.

Sequence diagram

Nama Simbol	Simbol	Keterangan
Actor		Entitas eksternal yang berhubungan dengan sistem
Object		Menunjukan objek yang berkaitan dengan alur kerja sistem.
Stimulus		Menunjukkan komunikasi antara dua buah objek berupa pengiriman pesan (*message*).
Self stimulus		Menunjukkan komunikasi berupa pengiriman pesan (*message*) kepada objek itu sendiri.

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Dewasa ini kehidupan manusia mendapatkan pengaruh yang sangat besar dari perkembangan teknologi informasi. Pengaruh tersebut memasuki hampir seluruh bidang kehidupan. Salah satu teknologi yang sudah dikenal cukup luas adalah *barcode* satu dimensi. Penggunaan *barcode* satu dimensi pada bidang industri memudahkan pelaku industri dalam mengelola barang yang mereka miliki, karena *barcode* satu dimensi menyimpan data yang spesifik seperti kode produksi, nomor identitas, dan lain-lain, sehingga sistem komputer dapat mengidentifikasi informasi yang dikodekan dalam *barcode* satu dimensi dengan mudah.

Seiring dengan perkembangan teknologi yang begitu pesat, penggunaan *barcode* kini mulai dikembangkan menjadi *barcode* dua dimensi. Keunggulan *barcode* dua dimensi dibandingkan dengan *barcode* satu dimensi adalah *barcode* dua dimensi dapat menyimpan data yang jauh lebih besar daripada *barcode* satu dimensi. *Barcode* satu dimensi hanya dapat menampung maksimal dua puluh digit data sedangkan *barcode* dua dimensi dapat menyimpan hingga ratusan bahkan ribuan digit data. Hal ini dikarenakan *barcode* dua dimensi mampu menampung data secara horisontal dan vertikal. Selain itu *barcode* dua dimensi dapat dibaca dari segala arah atau sudut (360 derajat) sehingga kemungkinan gagal dalam membaca *barcode* dua dimensi sangat kecil.

Diantara semua jenis *barcode* dua dimensi, *Quick Response Code* (QR Code) merupakan *barcode* dua dimensi yang paling dikenal oleh masyarakat. Saat ini QR Code sudah digunakan oleh banyak negara terutama Jepang sebagai negara pelopor. Sebagian besar pengusaha di Jepang memilih menggunakan QR *Code* sebagai alat tambahan dalam mempromosikan produknya, baik yang bergerak dalam perdagangan maupun dalam bidang jasa. Saat ini hampir semua jenis telepon genggam di Jepang sudah dapat membaca QR Code.

Sementara di Indonesia, QR Code mulai populer dengan diterapkannya pada beberapa perusahaan. Salah satu contohnya terdapat pada bidang kuliner, Majelis

Ulama Indonesia (MUI) menghimbau agar setiap restoran menerapkan QR Code yang berisikan data sertifikasi kehalalan restoran tersebut. Dengan memanfaatkan penggunaan telepon genggam yang kini hampir dimiliki oleh seluruh masyarakat, MUI berharap masyarakat dapat lebih mudah dan cepat dalam mengakses informasi restoran tersebut.

Sekarang sudah mulai banyak aplikasi ataupun halaman *website* yang dapat digunakan untuk membuat QR Code yang lebih dikenal dengan sebutan *QR Code Generator*. Pada umumnya QR Code yang banyak dihasilkan saat ini masih berwarna *hitam* dan putih. Mencermati hal tersebut maka penulis ingin membuat *QR Code Generator* yang dapat menghasilkan QR Code yang berwarna sehingga QR Code tersebut terlihat lebih menarik. Oleh karena itu, penulis memilih untuk menjadikannya sebagai topik tugas akhir dengan judul "Perancangan Aplikasi *QR Code Generator* dan *QR Code Reader* untuk Android".

1.2 Rumusan Masalah

Adapun masalah – masalah yang dihadapi untuk membangun aplikasi ini adalah sebagai berikut :

1. Bagaimana cara pembuatan aplikasi *QR Code Generator* menggunakan bahasa pemrograman Java untuk perangkat Android?
2. Bagaimana membangun aplikasi untuk menghasilkan QR Code berwarna gradasi menggunakan bahasa pemrograman Java?
3. Bagaimana cara pembuatan aplikasi *QR Code Reader* dengan menggunakan bahasa pemrograman Java untuk perangkat Android?

1.3 Tujuan

Hasil karya tulis ini pun dibuat dengan tujuan agar penulis dapat menyelesaikan jenjang kuliah pendidikan sarjana (S1) Teknik Informatika di Sekolah Tinggi Manajemen Informatika dan Komputer LIKMI, dimana karya tulis ini menjadi salah satu syarat untuk kelulusan, serta untuk melengkapi kegiatan belajar mengajar agar kualitas mahasiswa

dapat meningkat dalam bidang ilmu pengetahuan terutama mengenai tentang *barcode* dan QR Code.

1.4 Batasan Masalah

Batasan masalah pada pembuatan aplikasi ini adalah :

1. Aplikasi yang dibuat dalam penulisan tugas akhir ini bersifat *offline.*
2. Warna yang dapat digunakan pada QR Code sesuai dengan yang terdapat pada palet warna yang tersedia dan agar dapat dibaca warna *background* harus lebih muda dari warna QR Code.
3. Arah warna gradasi pada QR Code linear dari atas ke bawah.
4. Pemindaian melalui kamera membutuhkan aplikasi tambahan yaitu *barcode scanner*.

1.5 Kegunaan Hasil

Hasil tugas akhir ini diharapkan dapat berguna bagi orang banyak, diantaranya dapat digunakan untuk merancang sebuah QR Code sesuai dengan yang diinginkan, serta hasil karya tulis ini pun diharapkan dapat menjadi referensi pembuatan tugas akhir diwaktu selanjutnya.

1.6 Metodologi Penelitian

Langkah – langkah penelitian yang digunakan untuk membuat aplikasi ini adalah :

1. Studi Pustaka

   Penelitian dilakukan dengan cara membaca buku-buku, *website*, ataupun artikel sumber yang berkaitan dengan aplikasi yang dibuat dan membantu penelitian, serta, untuk mengumpulkan dokumen dan data yang dibutuhkan.

2. Analisis

   Analisis dilakukan setelah teori yang dikumpulkan dinilai cukup dan dilanjutkan dengan menganalisa objek dan komponen apa saja yang diperlukan dalam pembuatan aplikasi ini.

3. Perancangan

   Perancangan mulai dilakukan dengan merancang desain awal aplikasi ini agar sesuai dengan hasil analisis yang dilakukan sebelumnya

4. Implementasi

   Melakukan pembangungan aplikasi sesuai dengan hasil perancangan dengan menggunakan bahasa pemrograman.

5. Pengujian

   Setelah aplikasi selesai dibuat dilakukan pengujian untuk mencari tahu keberadaan *error* ataupun *bug*.

1.7 Sistematika Penulisan

Agar tugas akhir ini dapat dipahami lebih jelas, maka, pembahasan dalam penyusunan tugas akhir ini dikelompokan menjadi beberapa bab. Untuk itu penulis akan memberikan penjelasan secara garis besar isi dari masing-masing bab, yaitu sebagai berikut :

BAB I PENDAHULUAN

Bab ini menjelaskan tentang latar belakang masalah yang menjadi alasan dalam pemilihan topik, rumusan masalah, tujuan penulisan, batasan masalah, kegunaan hasil dan sistematika penulisan.

BAB II LANDASAN TEORI

Menjelaskan tentang landasan teori yang digunakan oleh penulis dalam penyelesaian tugas akhir ini, Serta penjelasan mengenai metodologi penelitian yang digunakan, model pengembangan yang dilakukan, meliputi pembahasan tentang sistem *barcode,* QR Code, serta perangkat lunak yang digunakan.

BAB III ANALISIS DAN PERANCANGAN PERANGKAT LUNAK

Bab ini berisi tentang analisis dan perancangan perangkat lunak yang meliputi gambaran umum perangkat lunak yang akan dibuat, spesifikasi kebutuhan perangkat lunak, diagram - diagram perancangan perangkat lunak sesuai dengan metodologi penelitian, dan rancangan antara muka aplikasi.

BAB IV IMPLEMENTASI DAN PENGUJIAN PERANGKAT LUNAK

Berisikan tentang implementasi sistem dan pengujian perangkat lunak yang meliputi spesifikasi kebutuhan perangkat keras yang dibutuhkan, pengujian antar muka, serta pengujian fungsi dari aplikasi ini.

BAB V KESIMPULAN DAN SARAN

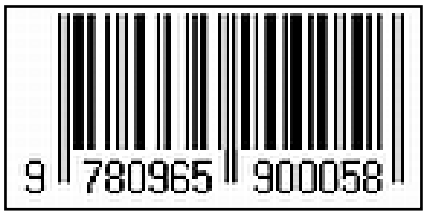
Bab ini merupakan bab penutup dari tugas akhir ini. Pada bab ini akan menerangkan kesimpulan-kesimpulan dan saran-saran yang didapat penulis selama proses pembuatan dan penelitian yang berguna untuk masa yang akan datang.

BAB II

LANDASAN TEORI

2.1 *Quick Response Code* (QR Code)

Barcode atau kode batang secara umum dapat dikategorikan menjadi dua macam yaitu *barcode* satu dimensi dan *barcode* dua dimensi. Keberadaan *barcode* dua dimensi saat ini mulai diperhatikan karena dapat menutupi kekurangan *barcode* satu dimensi. Salah satu kekurangan dari *barcode* satu dimensi adalah informasi yang hanya dapat ditampung secara horisontal. Ketinggian *barcode* satu dimensi tidak mempengaruhi informasi yang terdapat di dalamnya. Semakin banyak informasi yang ada maka semakin lebar *barcode* satu dimensi tersebut. Karena keterbatasan lebar jarak pindai dari alat atau aplikasi pemindai maka *barcode* satu dimensi tidak bisa terlalu lebar. Sehingga hanya sedikit informasi yang dapat ditampung. Karena kekurangan itulah *barcode* dua dimensi diciptakan.



Gambar 2.1
Contoh *Barcode* Satu Dimensi
(Sumber : Winter, 2010:17)

Saat ini *barcode* dua dimensi mulai mengambil alih fungsi penggunaan *barcode* satu dimensi. Diantara semua jenis *barcode* dua dimensi *Quick Response Code* merupakan yang paling banyak digunakan.

Quick Response Code yang biasa disingkat sebagai QR Code dikembangkan oleh Denso Wave, yaitu sebuah divisi di Denso Corporation. Denso Corporation yang merupakan sebuah perusahaan berasal dari Jepang. QR Code mulai dipublikasikan pada tahun 1994. Sesuai dengan namanya *Quick Response* yang jika diterjemahkan ke dalam bahasa Indonesia memiliki arti respon cepat, itulah yang menjadi tujuan awal

diciptakannya QR Code. QR Code diciptakan untuk menyampaikan informasi dengan cepat dan mendapatkan respons yang cepat pula. Menurut beberapa ahli QR Code memiliki pengertian sebagai berikut :

1. *"QR (short of "Quick Response") code is a matrix barcode (a two-dimensional code) that is readable by dedacated QR barcode readers and camera phones" (Price 2013:1).*
2. *"A QR code is two-dimensional, meaning that the information changes vertically as well as horizontally"* (Winter, 2010:18).

Dari pengertian tersebut dapat disimpulkan bahwa QR Code merupakan *barcode* dua dimensi yang dapat dibaca menggunakan telepon genggam. Karena QR Code memiliki dua dimensi dimana QR Code mampu menampung data secara horisontal dan vertikal sehingga informasi yang ditampung lebih banyak daripada *barcode* satu dimensi.

Gambar 2.2
Contoh QR Code
(Sumber : Winter, 2010:17)

2.1.1 Fungsi QR Code

Saat ini jumlah pengguna telepon genggam yang memiliki ketertarikan pada aplikasi QR Code bertambah. Pengguna dapat membuat QR Code dengan menggunakan *QR Code Generator*, lalu pengguna pun dapat menukarnya dengan pengguna lainnya melalui internet, menyimpannya di telepon genggam atau mencetaknya.

Media yang paling sering menampilkan QR Code adalah surat kabar atau majalah dan tipe data yang paling sering baca oleh pemindai adalah *Uniform Resource Locator* (URL) dan alamat *e-mail*. QR Code tidak hanya dapat digunakan saat pengguna mengakses internet saja, tetapi juga saat tidak adanya akses internet pun QR Code dapat digunakan. Seperti saat pemindai membaca QR Code yang terdapat pada kartu nama, pengguna dapat langsung mencari atau menyimpan informasi data tersebut pada kontak yang terdapat pada telepon.

Pengaplikasian QR Code pada kehidupan sehari-hari semakin banyak. Salah satu contohnya adalah QR Code dapat diaplikasikan pada kemasan sebuah produk. QR Code tersebut dapat berisikan informasi detil tentang produk tersebut seperti kontak *supplier*, tanggal kadarluawsa, data gizi, komposisi bahan dan sebagainya. QR Code dapat bekerja menjadi sebuah perantara antara produsen produk dan konsumen.

2.1.2 Kelebihan QR Code

QR Code memiliki beberapa kelebihan dibandingkan dengan *barcode* satu dimensi yaitu (Kato:2010,55) :

1. *Flexibility in symbol format*

Simbol QR Code memungkinkan untuk dapat dicetak dengan warna terang berada diatas warna gelap atau warna gelap berada di atas warna terang. Hal ini merupakan sebuah kelebihan yang sangat berharga karena QR Code dapat diaplikasikan tidak hanya pada media yang memiliki warna terang tetapi juga berwarna gelap. Dan QR Code dapat dibaca oleh pemindai dari berbagai sudut sehingga saat QR Code diaplikasikan pada media transparan seperti kaca, QR Code dapat dibaca oleh pemindai dari depan ataupun belakang.

2. *Masking*

QR Code Masking merupakan sebuah teknik yang dapat digunakan untuk mempermudah dan mempercepat pembacaan QR Code oleh pemindai. Teknik ini dapat mengalokasikan titik berwarna hitam dan putih secara rata dan membantu mencegah adanya pola yang sama. *Masking* dapat digunakan untuk mengkonsentrasikan pada satu

warna dan menghindari *module* yang seperti *position detection pattern* yang terdapat pada *data area* Sehingga performa pemindai QR Code dapat meningkat.

3. *Error Correction*

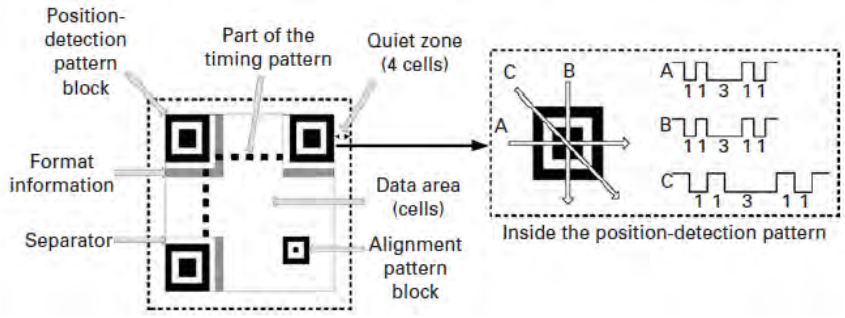
QR Code memiliki kemampuan untuk mengembalikan data asli jika *barcode* dua dimensi ini kotor atau rusak. Maksimum data yang dapat diperbaiki adalah sekitar 30%. QR Code memiliki empat level berbeda untuk *error correction* yaitu L, M, Q, dan H. Untuk level L perbaikan data dapat dilakukan sampai 7%, level M sampai 15%, untuk level Q adalah 25%, dan level H adalah 30%. Semakin tinggi level *error correction* yang digunakan maka meningkat juga kemampuan untuk memperbaiki data itu tetapi menambah juga data yang harus di-*generate*. Sehingga menghasilkan ukuran simbol yang lebih besar.

Ukuran dari simbol QR Code berhubungan dengan ukuran dari *module*. Jika ukuran *module* bertambah maka QR Code semakin stabil dan dapat lebih mudah dibaca oleh pemindai. Begitupun sebaliknya, saat ukuran *module* mengecil maka semakin lama waktu yang dibutuhkan oleh pemindai untuk membaca QR Code tersebut.

4. *Encodable Character Set*

QR Code mampu meng-*generate* hampir seluruh tipe data seperti angka, huruf, huruf bahasa Jepang (Kanji, Kana dan Hiragana), dan biner. Data maksimum yang dapat ditampung oleh QR Code adalah 7089 karakter untuk *numeric*, 4296 karakter untuk *alphanumeric*, 1817 karakter untuk huruf bahasa Jepang, dan 2953 *bytes* untuk *binary* (8 bit). Untuk setiap *numeric* dibutuhkan 3,3 *module*, untuk *alphanumeric* dibutuhkan 5,5 *module* untuk setiap karakter, 13 *module* untuk setiap karakter huruf bahasa Jepang, dan 8 *module* untuk setiap *binary* (8 bit).

2.1.3 Struktur QR Code



Gambar 2.3
Struktur QR Code
(Sumber : Kato, 2010:52)

Dalam buku "*Barcode for Mobile Devices*" dijelaskan bahwa QR Code memiliki tujuh buah elemen yaitu :

1. *Position Detection Pattern Block*

   Position detection pattern block adalah tiga buah simbol yang berada di sudut yang menjadi ciri khas dari QR Code. Bagian ini merupakan bagian pertama yang dideteksi oleh pemindai. Fungsi dari bagian ini adalah agar QR Code dapat dibaca dengan hasil yang sama dari berbagai sudut. *Position Detection Pattern Block* memiliki rasio warna yang sama yaitu 1 : 1 : 3 : 1 : 1 dimana hitam : putih : hitam : putih : hitam dari tiga sudut yang berbeda.

2. *Timming Pattern*

   Timming pattern berada di antara *position detection pattern block*. Berfungsi untuk mengidentifikasi koordinat pusat QR Code dengan pola hitam dan putih yang disusun secara bergatian.

3. *Alignment Pattern*

   Alignment pattern berfungsi untuk memperbaiki penyimpangan atau kerusakan yang terjadi pada QR Code.

4. *Format Information*

   Di sini berisikan data berupa *mask pattern* dan tentang *error corection*. Area ini merupakan tempat pertama yang melalui proses *decoding* saat dibaca oleh pemindai.

5. *Data Area*

   Merupakan tempat untuk menyimpan data yang telah dikodekan.

6. *Quiet Zone*

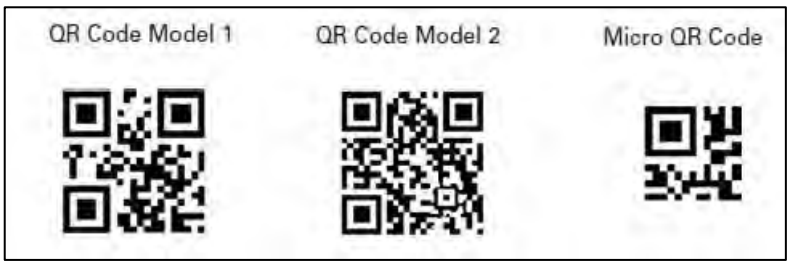
   Quit Zone adalah daerah kosong yang berada di bagian terluar dari QR *Code*. Pada area ini simbol QR Code diperbolehkan untuk diabaikan dari latar, sehingga QR Code dapat dibaca lebih cepat.

7. *Separator*

   Separator merupakan daerah kosong yang menjadi pembatas antara *position detection pattern block* dan *data area*.

2.1.4 Bentuk QR Code

Secara umum QR Code memiliki tiga buah bentuk yaitu model pertama, model kedua dan *Micro QR Code*. Model pertama merupakan bentuk asli dari QR Code, yang lainnya merupakan variasi bentuk lainnya dari model pertama. Terdapat empat belas versi untuk model pertama. Dengan tujuan untuk mengurangi kerusakan, *alignment pattern* ditambahkan pada QR Code model pertama dan hasilnya adalah QR Code model kedua. QR Code model kedua sendiri memiliki empat puluh versi. Saat ini QR Code model kedualah yang lebih sering digunakan. Sedangkan *Micro QR Code* merupakan QR Code yang berukuran kecil. *Micro QR Code* digunakan pada saat minimnya ketersediaan tempat dan data yang ada. Sebagai perbandingan ukuran minimum untuk *Micro QR Code* adalah 11 x 11 *modules*, sedangkan QR Code memiliki ukuran minimum hampir dua kali lipatnya yaitu 21 x 21 *modules*.

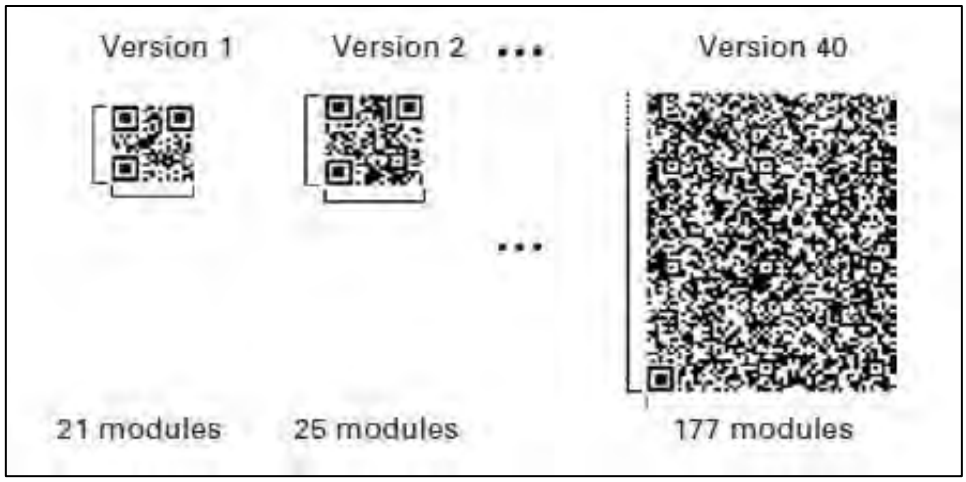


Gambar 2.4
Bentuk QR Code

(Sumber : Kato, 2010:55)

2.1.5 *Symbol Version*

QR Code terdiri dari kotak-kotak yang berwarna hitam dan putih yang disebut *module*. QR Code memiliki empat puluh versi mulai dari versi satu yang memiliki 21 x 21 *modules* sampai versi empat puluh yang memiliki 177 x 177 *modules*. Setiap versi memiliki jumlah *module* yang berbeda. Saat versi bertambah satu maka empat buah *module* ditambahkan pada setiap sisi dari QR Code tersebut.



Gambar 2.5
Simbol Version QR Code
(Sumber : Kato, 2010:226)

2.2 *Reed-Solomon Codes*

Diantara semua teknik *error detection* dan *error correction,* yang paling sering digunakan untuk *barcode* dua dimensi adalah teknik *Reed-Solomon Codes* (RS Codes). RS Codes adalah salah satu teknik yang berfungsi untuk memperbaiki *codes*. RS Codes digunakan pada CD-ROM, dan DVD, untuk sistem penyiaran DVB (*Digital Video Brodcasing*) dan lain-lain. Teori dasar RS Codes adalah Bose-Chaudhuri-Hochquenghem (BCH) codes yaitu Galois Field (GF). RS Codes diciptakan oleh Irving S. Reed dan Gustave Solomon pada tahun 1960.

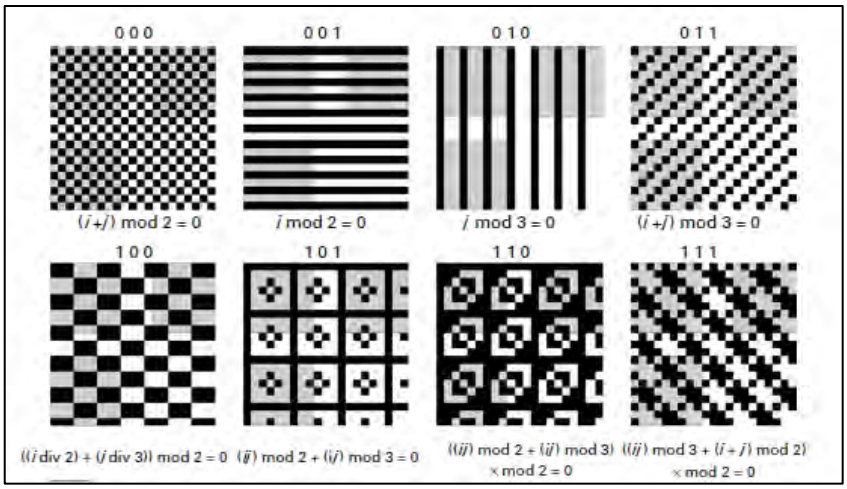
RS Codes didefinisikan sebagai RS(N,K) dengan sejumlah R bit. Dimana K merupakan panjang informasi yang akan dikodekan. Sedangkan N adalah panjang blok yang didapat dari 2^{R-1} dengan R adalah jumlah bit per simbol. Dengan menggunakan

rumus 2T = (N – K), maka dapat diketahui berapa banyak data yang dapat dikoreksi saat *error* terjadi yang dilambangkan dengan T.

Sebuah elemen GF terdiri dari sekumpulan elemen primitif yang dinotasikan oleh α, dan bernilai 0, α^0, α^1, α^2, ... , α^{N-1}. Untuk membuat sebuah sebuah set dari elemen 2^R, dimana N = 2^{R-1}. Elemen ini dikenal sebagai GF(2^R). Nilai yang digunakan untuk α biasanya adalah dua. Dari nilai α yang telah disebutkan, maka dapat dipresentasikan dalam bentuk polinomial $\propto^{R-1} x^{R-1} + \cdots + \propto^1 x + \propto^0$. Koefisien untuk $\propto^{R-1}$ sampai $\propto^0$ mengambil nilai satu atau nol. RS Codes (N,K) dapat diperoleh dengan membentuk *polynomial generator code* (g(x)). Bentuk umum dari persamaan *polynomial generator code* adalah $g(x) = (x + \propto^b)(x + \propto^{b+1}) \ldots (x + \propto^{b+2T-1})$.

2.3 *Data Masking*

Data masking merupakan teknik yang digunakan pada QR Code untuk mengurangi kerusakan yang terjadi pada saat pembacaan *barcode symbol* pada *module* putih maupun hitam. Saat *module* hitam terkonsentrasi pada bagian tertentu pada QR Code dan hanya memiliki sedikit *module* putih, hal ini dapat menyebabkan QR Code gagal untuk dibaca. Sebuah area data yang memiliki pola yang sama dengan *position detection pattern* pun dapat memperlambat proses pemindaian. *Masking* dapat digunakan untuk meminimalisir hal-hal tersebut sehingga proses dalam pemindaian QR Code dapat dipercepat.



Gambar 2.6
Mask Pattern

(Sumber : Kato, 2010:142)

Data masking memiliki delapan *mask pattern* berbeda yang telah ditentukan. Semua pola tersebut diterapkan terlebih dahulu pada simbol QR Code lalu dievaluasi untuk menentukan pola terbaik yang akan digunakan pada simbol QR Code tersebut. Terhadap setiap pola tersebut dilakukan perhitungan sesuai dengan empat peraturan yang telah ditentukan. Pola terbaik yang akan digunakan merupakan pola yang memiliki jumlah perhitungan terendah. Keempat peraturan tersebut adalah (Kato, 2010:143) :

1. Untuk setiap lima module yang memiliki warna yang sama pada setiap baris dan kolom diberikan nilai tiga poin.
2. Untuk setiap 2 x 2 area yang memiliki warna yang sama pada simbol QR Code diberikan nilai tiga poin.
3. Untuk setiap baris dan kolom yang memiliki rasio warna yang sama dengan *position detection pattern* yaitu 1 : 1 : 3 : 1 : 1 untuk warna hitam : putih : hitam : putih : hitam dan diikuti empat module putih pada salah satu sisinya diberikan nilai sepuluh poin.
4. Peraturan keempat memiliki beberapa tahapan perhitungan yaitu :
    a. Hitung total *module* yang terdapat pada simbol.
    b. Hitung berapa jumlah *module* berwarna hitam.
    c. Hitung berapa nilai persennya dengan cara jumlah *module* berwarna hitam dibagi dengan total *module* yang ada lalu dikalikan dengan seratus.
    d. Tentukan angka sesudah dan sebelum kelipatan lima dari hasil persentase dari tahapan sebelumnya.
    e. Kurangi dengan lima puluh untuk setiap angka kelipatan lima tersebut. Semua hasil dibuat menjadi nilai positif.
    f. Setiap hasil pengurangan tersebut dibagi dengan angka lima.
    g. Dari dua hasil tersebut pilih angka terkecil dan kalikan dengan sepuluh.

Tahapan yang perlu dilakukan untuk melakukan *data masking* adalah (Kato,2010:143):

1. Simbol QR Code diterapkan *mask pattern* menggunakan fungsi yang telah ditentukan untuk pola tersebut. Dengan tahapan ini *module* berwarna hitam yang

tertutupi dengan *module* berwarna hitam yang berasal dari pola akan berubah menjadi berwarna putih, berlaku juga untuk *module* berwarna putih. Untuk *module* berwarna putih yang tertutupi oleh *module* berwarna hitam atau akan berubah menjadi *module* berwarna hitam.

2. Setiap *mask pattern* dievaluasi dengan melakukan perhitungan sesuai dengan empat peraturan yang ada.
3. *Mask pattern* yang memiliki nilai terendah akan digunakan.

2.4 Pengolahan Citra

Secara umum pengolahan citra digital merupakan pemrosesan gambar ataupun data dua dimensi dengan menggunakan komputer agar gambar yang dihasilkan memiliki kualitas yang lebih baik dari kualitas sebelumnya. Pengolahan citra dilakukan untuk memperbaiki kesalahan data sinyal yang terjadi akibat transmisi data, serta untuk meningkatkan kualitas gambar agar lebih mudah diinterpretasikan oleh sistem penglihatan manusia. Suatu citra dapat didefinisikan sebagai fungsi f(x,y) berukuran M baris dan N kolom, dengan x dan y adalah koordinat. Citra digital dapat ditulis dalam bentuk matriks seperti yang diperlihatkan pada gambar 2.7.

$$f(x,y)=\begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{bmatrix}.$$

Gambar 2.7
Matriks Citra Digital
(Sumber : Putra, 2010:20)

Pixel merupakan istilah yang paling sering digunakan dalam pengolahan citra dan merupakan unsur gambar terkecil dari sebuah gambar atau citra. Sedangkan resolusi *pixel* merupakan perhitungan jumlah *pixel* dalam sebuah citra digital. Dengan tinggi N *pixel* dan M *pixel* berarti gambar tersebut memiliki resolusi *pixel* sebesar M x N. Resolusi *pixel* akan mewakili jumlah *pixel* lebar dan jumlah *pixel* dari tinggi citra tersebut.

Suatu *pixel* memiliki nilai dalam rentang tertentu bergantung pada jenis warnanya. Secara umum nilai jangkauannya adalah 0 – 255. Dalam buku "*Pengolahan Citra Digital"* Dharma Putra dijelaskan bahwa citra dapat dikelompokan berdasarkan nilai *pixel*-nya yaitu (Putra : 2010,40) :

1. Citra Biner

   Citra biner atau citra monokrom merupakan citra digital yang hanya memiliki dua nilai *pixel* yaitu hitam dan putih. Warna hitam bernilai satu dan putih berwarna putih. Citra biner dapat dilakukan dengan mengkonversi citra *grayscale* ke citra biner dengan menggunakan operasi pengambangan (*thresholding*).

2. Citra *Grayscale*

   Citra grayscale hanya memiliki warna hitam, keabuan, dan putih. Tingkatan keabuan yang dimiliki merupakan warna abu dengan berbagai tingkatan dari hitam mendekati putih.

3. Citra Warna 8 bit

   Setiap *pixel* dari citra warna 8 bit hanya diwakili oleh 8 bit dengan jumlah warna maksimum yang dapat digunakan adalah 256 warna.

4. Citra Warna 16 bit

   Citra warna 16 bit atau disebut juga sebagai citra *highcolor*, setiap *pixel*-nya diwakili oleh 2 byte memory (16 bit). Warna 16 bit memiliki 65.536 warna.

5. Citra Warna 24 bit

   Setiap *pixel* dari citra warna 24 bit diwakili dengan 24 bit. Total variasi warna dari 24 bit adalah 16.777.216 warna.

Operasi pengolahan citra yang dilakukan saat membentuk atau membaca QR Code adalah :

1. *Edge Detection*

Deteksi tepi atau *Edge Detection* merupakan suatu proses pengolahan citra yang menghasilkan tepi-tepi dari obyek gambar atau citra. *Edge Detection* dapat dilakukan dengan cara membandingkan suatu *pixel* dengan *pixel* yang bersinggungan dengan *pixel*

tersebut. Suatu *pixel* dinyatakan merupakan *edge* atau tepi adalah saat *pixel* tersebut memiliki perbedaan nilai dengan *pixel* yang bersinggungan dengan *pixel* tersebut.

2. *Gradient Color*

Warna merupakan sprektrum tertentu yang terdapat di dalam suatu cahaya. Nilai warna ditentukan oleh tingkat kecerahan maupun kesuraman warna. Diantara semua kombinasi warna yang memiliki rentang warna paling lebar adalah *Red Green Blue* (RGB). Ketiga warna tersebut merupakan warna pokok. Warna-warna lain dapat diperoleh dengan mancampurkan ketiga warna tersebut dengan perbandingan tertentu. RGB memiliki rentang nilai maksimum 255 (8 bit) dengan nilai minimum nol. Warna gradasi dapat dihasilkan dengan meningkatkan nilai RGB dari setiap *pixel* secara bertahap sampai nilai RGB tersebut sama dengan nilai RGB lainnya

3. *Pattern Recognition*

Pattern Recognition merupakan proses pengolahan citra lainnya yang digunakan untuk mendeteksi pola yang sudah pernah dilakukan berdasarkan kesamaan atau kemiripan. Proses *pattern recognition* dapat dilakukan dengan mengambil gambar dari kamera, lalu pengenalan pola akan dilakukan dengan memperoleh data dari sensor kamera. Setelah pengenalan pola dilakukan maka proses binarisasi citra dilakukan untuk mempermudah sistem mengenali pola dari gambar tersebut. Setelah mendeteksi pola dilakukan kemudian sistem akan menganalisis citra dan mengenali sudut-sudut tepi yang menjadi titik - titik utama untuk dijadikan pengenalan pola. Titik-titik tersebut akan disesuaikan dengan pola yang ada pada sistem.

4. *Grayscale Transformation*

Operasi pengolahan citra yang dilakukan untuk mengubah warna citra dari citra *true color* menjadi citra *grayscale* bernama *grayscale transformation*. *Grayscale transformation* dilakukan dengan menjumlahkan nilai *Red*, *Green*, *Blue* dari *pixel* tersebut lalu dibagi dengan angka tiga atau dengan menentukan nilai *Luminance*. *Luminance* merupakan nilai warna abu-abu dari RGB yang didapat dari rumus wr R + wg G + wb B dimana wr merupakan bobot nilai dari warna R, wg bobot nilai warna G, dan wb untuk

nilai warna B. Berdasarkan *National Television System Committee* (NTSC) wr bernilai 0,299, wg bernilai 0,587, dan wb bernilai 0,144.

2.5 Rekayasa Perangkat Lunak

Istilah *software engineering* mulai digunakan pada akhir tahun 1950-an dan awal tahun 1960-an. Dan istilah ini pun digunakan dalam *software engineering conference* yang diselenggarakan oleh komite sains *North Atlantic Treaty Organization* (NATO) pada tahun 1968. Dalam sejarah *software engineering* tercatat pernah terjadi krisis perangkat lunak pada tahun 1960-an sampai 1980-an. Terdapat banyak masalah ditemukan oleh proses pengembangan perangkat lunak. Salah satu kasus yang terkenal adalah meledaknya roket Ariane akibat kegagalan perangkat lunak.

Para peneliti melakukan penelitian selama bertahun-tahun untuk menemukan cara untuk memecahkan masalah krisis perangkat lunak tersebut. Sampai akhirnya ditemukanlah pemrograman terstruktur, pemrograman berorientasi objek, perangkat pembantu pengembangan perangkat lunak hingga *Unified Modeling Language* (UML). Tahun 1990 sampai tahun 1999 mulailah berkembangnya internet yang berdasarkan teknologi yang disponsori dan direncanakan oleh pemerintah untuk mendorong seluruh pertumbuhan cepat dalam *World Wide Web* (WWW). Pertumbuhan pada bahasa *HyperText Markup Language* (HTML) tumbuh dengan cepat sehingga cara menampilkan dan perolehan kembali informasi dari internet menjadi lebih terorganisasi. Koneksi jaringan yang luas mendorong semakin tumbuhnya virus komputer termasuk pencegahannya.

Saat memasuki tahun milenium semakin banyak permintaan untuk membuat perangkat lunak pada organisasi atau toko-toko kecil sehingga kebutuhan perangkat lunak yang murah pun memicu munculnya metodologi yang lebih cepat, yaitu pengembangan untuk menjalankan perangkat lunak, dari kebutuhan ke penyebaran yang lebih cepat dan mudah. Walaupun untuk organisasi atau perusahaan-perusahaan yang membutuhkan perangkat lunak yang sangat besar masih menggunakan metodologi yang memiliki dokumentasi yang berat karena banyaknya bahan yang didokumentasikan.

Menurut beberapa ahli RPL memiliki pengertian sebagai berikut :

1. *Suatu disiplin ilmu yang membahas semua aspek produksi perangkat lunak, mulai dari tahap awal yaitu analisa kebutuhan pengguna, menentukan spesifikasi dari awal kebutuhan pengguna, design, pengkodean, pengujian sampai pemeliharaan sistem setelah digunakan* (Mulyanto, 2008:2).
2. *"Satu bidang profesi yang mendalami cara-cara pengembagan perangkat lunak termasuk pembuatan, pemeliharaan, manajemen organisasi pengembangan perangkat lunak, dan sebagainya*. "(Simarmata, 2010:9).
3. *"The creative activity of understanding the business problem, coming up with an idea for solution, and designing the "blueprints" of solution."* (Marsic,2012:2).
4. *"Software engineering is an engineering discipline that is concerned with all aspects of software production from the early stages of system specification to maintaning the system after it has gone into use*." (Sommerville,2011:7).

Dari pengertian-pengertian diatas maka dapat disimpulkan bahwa RPL merupakan suatu ilmu disiplin yang mempelajari tahapan-tahapan dalam pembuatan sebuah perangkat lunak dari awal sampai perangkat lunak itu selesai dibuat lalu diimplementasikan dan dilakukan pemeliharaan.

RPL memiliki pendekatan sistematis yang biasa disebut sebagai proses perancangan perangkat lunak. Proses perancangan perangkat lunak tersebut memiliki empat susunan aktivitas yang biasanya terdapat dalam semua tahapan perancangan perangkat lunak. Empat aktivitas tersebut adalah spesifikasi perangkat lunak, pembangunan perangkat lunak, validasi perangkat lunak, evolusi perangkat lunak. Yang dimaksud dengan spesifikasi perangkat lunak adalah *customer* dan *programmer* menspesifikasikan fungsi-fungsi yang terdapat pada perangkat lunak yang dibuat. Lalu pengembangan perangkat lunak adalah saat perangkat lunak dibuat. Validasi perangkat lunak merupakan saat perangkat lunak diperiksa sesuai atau tidaknya dengan kebutuhan *customer.* Yang terakhir evolusi perangkat lunak adalah disaat perangkat lunak dimodifikasi untuk memenuhi kebutuhan *customer.*

Di dalam buku "*Software Engineering: A Practitioner's Approach Seventh Edition"* Roger Pressman menjelaskan tahapan-tahapan yang dimiliki dalam merancang sebuah perangkat lunak yaitu (Pressman,2010:15) :

1. Komunikasi

   Komunikasi merupakan tahapan pertama yang penting dan tidak boleh dilewatkan. Komunikasi dilakukan antara *customer* dan developer untuk menentukan fitur dan fungsi apa yang akan ada di dalam perangkat lunak yang dibuat.

2. Perencanaan

   Software project plan merupakan sebuah penuntun dalam membuat perangkat lunak yang berisi deskripsi tugas yang akan dilakukan, kemungkinan resiko yang dapat terjadi, sumber dayayang akan diperlukan, perangkat lunak yang akan dihasilkan, produk kerjayang dihasilkan, dan jadwal pekerjaan.

3. *Modeling* (Pemodelan)

   Dalam tahap *modelingprogrammer* membuat sebua sketsa agar lebih mengerti yang dibutuhkan *customer* dalam perangkat lunak yag dibuat, dan lebih mamahami masalah yang ada dan bagaimana cara mengatasinya.

4. Pembangunan

   Perangkat lunak mulai dibuat dan dilakukan *testing* untuk mengetahui *error* yang terdapat pada *code*.

5. *Deployment*

   Perangkat lunak yang telah dibuat diberikan kepada *customer* agar *customer* dapat mengevaluasi perangkat lunak tersebut dan memberikan *feedback* agar perangkat lunak tersebut sesuai dengan kebutuhan *customer*.

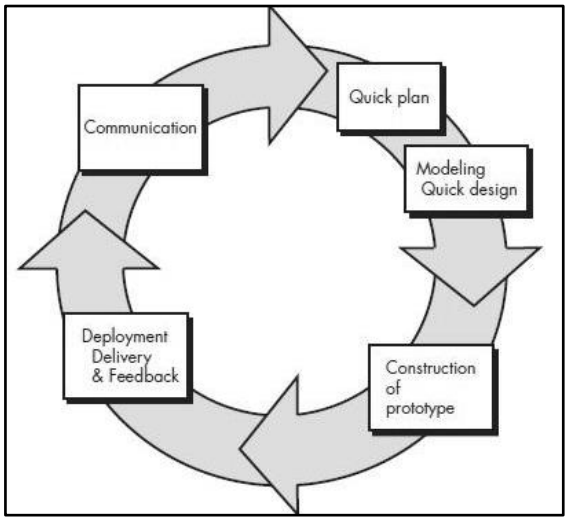
Dengan dipenuhinya tahapan-tahapan RPL diharapkan dapat menghasilkan perangkat lunak yang memiliki kinerja tinggi, biaya produksi rendah, biaya perawatan rendah, dan dapat digunakan pada berbagai jenis platform.

2.5.1 Model Pengembangan Perangkat Lunak

Model pengembangan perangkat lunak yang digunakan untuk membangun aplikasi *QR Code Generator* ini adalah *prototype*. Di dalam buku "*Software Engineering: A Practitioner's Approach Seventh Edition"* model *protoype* didefinisikan sebagai :

The prototype is deployed and evaluated by stakeholders, who provide feedback that is used to further refine requirements. Iteration occurs as the prototype is tuned to satisfy the needs of various stakeholders, while at the same time enabling you to better understand what needs to be done. (Pressman, 2010:44).

Dari kutipan tersebut dapat diketahui bahwa model pengembangan *prototype* ini memerlukan komunikasi antara developer dengan *customer* untuk menyempurnakan model perangkat lunak dasar (*prototype*) yang telah dibuat sesuai dengan kebutuhan awal *customer*. *Prototype* membantu developer dan *customer* agar lebih mengerti apa fungsi yang akan terdapat dalam perangkat lunak yang dibuat saat kebutuhan yang diinginkan oleh *customer* terasa kabur.



Gambar 2.8
Model Pengembangan *Prototype*
(Sumber : Pressman, 2010:43)

Dari gambar 2.8 dapat dijelaskan bahwa model *prototype* ini memiliki langkah-langkah pengembangan sebagai berikut (Simarmata, 2010:62) :

1. Developer mengumpulkan kebutuhan awal *customer*.
2. Developer membuat perancangan secara cepat.
3. Membangun sebuah *prototype* sesuai dengan kebutuhan awal *customer.*
4. *Prototype* diuji coba oleh *customer*, kemudian *customer* memberikan *feedback*.
5. Revisi terhadap *prototype* sesuai dengan *feedback* yang diberikan.
6. Apabila *customer* tidak puas maka dilakukan revisi kembali.
7. Jika *customer* merasa puas terhadap *prototype* yang dibuat maka pengembangan skala besar dapat dilakukan.

2.6 *Unified Modeling Language* (UML)

UML menjadi bahasa standar yang digunakan untuk berkomunikasi antara *programmer* dengan developer ataupun antar developer. Melalui UML hasil analisa dan design divisualisasikan agar lebih mudah dimengerti oleh *customer* tentang perangkat lunak yang akan dibuat. Bahasa UML dikembangkan oleh Grady Booch, Jim Rumbaugh, dan Ivar Jacobson pada tahun 1990-an.

UML memiliki tiga belas tipe diagram yang dibagi menjadi dua kelompok yaitu *structure diagram* dan *behaviour diagram* dimana dalam *behaviour diagram* terdapat kelompok *interaction diagram*. Diagram yang termasuk ke dalam kelompok *sturcture diagram* adalah *class diagram, composite diagram, component diagram, deployment diagram, object diagram, dan package diagram*.

Selanjutnya kelompok *intercation diagram* adalah *sequence diagram, communication diagram, interaction overview diagram, timing diagram*. Sedangkan untuk kelompok *behavior diagram* adalah *activity diagram, use case diagram, state machine diagram* dan diagram-diagram yang terdapat dalam kelompok *interaction diagram*. Ke tiga belas diagram tersebut dalam spesifikasi UML tidak dikatakan harus digunakan untuk metodologi tertentu. Beberapa tipe diagram yang sering dipakai adalah :

1. *Use Case Diagram*

Use case merupakan deskripsi dari perangkat lunak yang digunakan oleh *customer,* atau pengguna perangkat lunak, atau dari perangkat lunak yang lain. *Use case diagram* menunjukkan bagaimana antar *use case* berhubungan dan bagaimana pengguna dapat menggunakannya.

2. *Activity Diagram*

Activity Diagram digunaka untuk menjelaskan alur aktivitas dari awal alur aktivitas tersebut berasal, kemungkinan-kemungkinan pilihan yang terjadi, dan akhir dari alur tersebut. *Activity diagram* juga menggambarkan beberapa proses yang dapat terjadi secara bersamaan.

3. *Sequence Diagram*

Sequence diagram merupakan pedoman dalam membuat sebuah perangkat lunak, hal ini dikarenakan dalam *sequence diagram* diperlihatkan interaksi antara *actor* dengan objek dan antar objek itu sendiri, kegiatan yang dilakukan, urutan kegiatan, dan hal-hal yang dibutuhkan.

4. *Class Diagram*

Class Diagram digunakan pada saat membuat sebuah perangkat lunak berorientasi objek untuk memperlihatkan *class-class* yang terdapat dalam perangkat lunak dan hubungan antar *class* tersebut.

2.7 *Object Oriented Programming*

Object Oriented Programming (OOP) atau disebut juga pemrograman berorientasi objek merupakan sebuah konsep pemrograman yang sedang banyak digunakan saat ini. Konsep OOP ini membagi program menjadi objek-objek dimana setiap objek dapat menerima pesan, memproses data, dan mengirim pesan ke objek lainnya. Penggambaran OOP merupakan penggambaran dari dunia nyata yang diubah dalam bentuk *code*. Konsep ini banyak digunakan dalam pembuatan *software* yang rumit dan kompleks. Menurut Grady Booch, dkk dalam buku berjudul "*Object-Oriented Analysis and Design with Applications"* OOP diartikan sebagai metode impelentasi dimana program diatur sebagai koleksi dari objek, yang masing-masing merupakan contoh dari *class*, dan semua anggota dari *class* tersebut memiliki warisan dari super *class*-nya melalui hubungan *inheritance* (Booch,2007:41).

Komponen utama dalam membangun sebuah sistem perangkat lunak adalah *class* dan objek. Dimana objek memiliki pengertian sebagai berikut :

1. *An object is a thing, an entity, a noun, something you can pick up or kick, anything you can imagine that has its own identity. Some objects are living, some aren't. Examples from the real world include a car, a person, a house, a table, a dog, a pot plant, a check book or a raincoat* (O'Docherty, 2005:13).

2. "*Object something that is or is capable of being seen, touched, otherwise sensed and about which user store data and associate bahaviour" (Whitten dan Bentley, 2007:371).*

Dari pengertian – pengertian tersebut dapat disimpulkan objek merupakan sebuah entitas ataupun benda yang memiliki identitasnya sendiri. Beberapa objek merupakan makhluk hidup, dan yang lainnya, bukan mahkluk hidup yang dapat dilihat, disentuh, yang memiliki *behaviour* dan *attributes.*

Menurut Mike O'Docherty sebuah objek memiliki (O'Docherty, 2005:13-14) :

1. *Attributes*, menggambarkan nilai-nilai yang terdapat dari sebuah objek, dan membedakannya objek yang satu dengan yang lainnya.
2. *Behavior,* menggambarkan hal-hal yang dapat dilakukan oleh objek dari suatu *class*.

Sedangkan *class* menurut buku "S*ystem Analisis and Design Methods"* diartikan sebagai kumpulan dari objek yang memiliki *attributes dan behaviour* yang sama. OOP memiliki tiga konsep utama seperti yang dijabarkan dalam buku "*Object – Oriented Analysis and Design with Applications 3rd Edition"* yaitu (Booch, 2007:98):

1. *Encapsulation*

   Menyembunyikan sebuah variabel dan *method* dari suatu objek yang tidak perlu diketahui oleh objek lainnya dan menyediakan *interface* untuk mengakses variabel tersebut.

2. *Inheritance*

   Inheritance merupakan pewarisan atau penurunan dari sebuah *class* ke *class* lainnya. *Class*-*class* yang memiliki persamaan yang sama dapat digabungkan dan menjadi sebuah *class* induk (*superclass*). Sedangkan *class* yang menjadi turunan dari *superclass* disebut *subclass*. *Subclass* memuat hal-hal yang lebih spesifik dari *super class*.

3. *Polymorphism*

   Polymorphism memungkinkan beberapa *class* memiliki metode yang sama. Suatu objek dapat memiliki berbegai bentuk seperti menjadi objek dari *class* itu sendiri atau sebagai objek dari super *class*nya.

2.8 Eclipse

Eclipse merupakan salah satu IDE (*Integrated Development Environment*) yang paling banyak digunakan oleh para *develoiper* untuk mengembangkan perangkat lunak. Eclipse dikembangkan oleh IBM (*International Business Machines*) untuk menggantikan perangkat lunak IBM Visual Age for Java, yang menjadi *open source* pada November 2001. Saat ini Eclipse dikelola oleh sebuah organisasi *non-profit* yaitu Eclipse Foundation.

Walaupun Eclipse dikembangkan dengan menggunakan bahasa pemrograman Java tetapi Eclipse dapat digunakan untuk melakukan pengembangan aplikasi dengan menggunakan berbagai bahasa pemrograman lain seperti C++, Cobol, PHP, Java, dan lain-lain. Karena *platform* yang dimiliki oleh Eclipse membolehkan *plug-ins* di-*install* secara terpisah, sehingga para *developer* dapat membuat perangkat lunak sesuai dengan bahasa pemrograman apa yang dikuasai.

Setiap sistem di dalam *platform* yang dimiliki oleh Eclipse merupakan kumpulan dari beberapa *plug-ins*. Untuk membuat perangkat lunak untuk telepon genggam yang memiliki sistem operasi *android*, maka *plug-ins* yag dibutuhkan adalah Android Development Tools *plug-ins* dan Java Developemnt Tools. Eclipse pun dapat dijalankan pada hampir semua sistem operasi komputer.

2.9 Android

Android merupakan sistem operasi yang bersifat *open source* sehingga membebaskan para *developer* untuk dapat menciptakan aplikasi mereka sendiri. Sistem operasi Android dikembangkan oleh Andoird Inc., pendatang baru yang membuat aplikasi untuk telepon genggam. Pada tahun 2005 Google membeli Android Inc., dan dibentuklah Open Handset Alleance yang merupakan organisasi yang terdiri dari perusahaan-perusahaan teknologi, produsen perangkat seluler, operator nirkabel, serta produsen *chipset* yang bertujuan untuk mengembangkan standar terbuka bagi perangkat seluler. Setelah Android Inc. dimiliki oleh Google para pendiri perusahaan ini yaitu Andy Rubin, Rich Miner, Nick Sears, dan Chris White bekerja pada Google.

Perilisan perdana Android dilakukan pada 5 November 2007 yang dimanakan Android versi 1.0 atau disebut Android versi beta. Telepon genggam pertama yang menggunakan sistem operasi Android adalah HTC Dream yang dirilis pada 22 Oktober 2008. Sampai saat ini pembaharuan versi Android terus berlanjut dari awal peluncurannya. Berikut adalah perkembangan Android dari awal yang dijabarkan pada buku yang berjudul "*Sekali Baca Langsung Inget Mengupas Lengkap All About ANDROID*" (Master.com, 2012:7) :

1. Android 1.0, dirilis 5 Nevember 2007
2. Android 1.1, dirilis pada 9 Maret 2009. Android versi ini memiliki pembaharuan pada aplikasi, jam alarm, pencarian suara, pemberitahuan *e-mail*, dan pengiriman pesan dengan *G-mail*.
3. Android 1.5, Google menamakan Cupcake untuk Android versi ini. Dirilis pada pertengahan Mei 2009, Google melakukan pembaharuan serta menambahkan beberapa fitur yaitu kemampuan merekam dan menonton video dengan modus kamera, meng-*upload* video ke Youtube dan gambar ke Picasa langsung dari telepon genggam, kemampuan terhubung secara otomatis ke *headset bluetooth*, animasi layar, dan keyboard pada layar yang dapat disesuaikan dengan sistem.
4. Android 1.6, atau yang disebut Donut dirilis pada September 2009 yang memiliki proses pencarian yang lebih baik dari versi sebelumnya, penggunaan baterai indikator dan kontrol *applet* VPN. Pengguna pun dapat memilih foto yang akan dihapus dari galeri.
5. Android 2.0 – 2.1, Eclair dirilis pada 3 Desember 2009. Google kembali melakukan pembaharuan pada pengoptimalan *hardware*, pengingkatan Google Maps, dan lain-lain.
6. Android 2.2, dirilis pada 20 Mei 2010 dan diberi nama Frozen Yoghurt (Froyo). Salah satu perubahan yang ada adalah kecepatan kinerja dan aplikasi dua hingga lima kali lebih cepat.
7. Android 2.3, Gingerbread dirilis pada 6 Desember 2010. Beberapa perubahan yang terjadi adalah peningkatan kemampuan permainan, peningkatan fungsi *copy*-

paste, *user interface* didesain ulang, dan dukungan jumlah kamera yang lebih dari satu.

8. Android 3.1 – 3.1, Android Honeycomb memiliki *user interface* yang berbeda karena didesain khusus untuk tablet. Honeycomb mendukung ukuran layar yang lebih besar, dan multi prosesor.
9. Android 4.0, diberi nama ICS (*Ice Cream Sandwich*) oleh Google yang dirilis pada 19 Oktober 2011. ICS memiliki fitur baru yaitu membuka kunci dengan pengenalan wajah, perbaikan *input* teks dan suara, serta tombol virtual yang dapat menggantikan tombol fisik.
10. Android 4.1 – 4.3, dirilis pada 9 Juli 2012 yang dinamakan Jelly Bean. Terdapat beberapa fitur baru yang ditambahkan seperti *lock screen widget*, dan *bluetooth smart ready*.

Penerus dari Android versi Jelly Bean adalah Android 4.4, yang diberi nama Kitkat yang dirilis pada 31 Oktober 2013. Pembaharuan yang ada adalah perubahan terhadap *user interface*, peningkatan fitur keamanan dan performa. Google pun kembali melakukan pembaharuan sehingga diluncurkanlah Android 5.0. Untuk versi ini Google memberi nama Lollipop yang memiliki beberapa perubahan dari pendahulunya salah satunya adalah *user interface* yang lebih responsif.

Terdapat beberapa kelebihan yang dimiliki oleh sistem operasi Android ini yaitu (Master.com, 2012:13) :

1. Bersifat *open source* sehingga bisa dikembangkan oleh siapa saja
2. Mudah dalam hal notifikasi. Sistem operasi Android dapat memberitahu pengguna tentang adanya *e-mail*, SMS, atau media sosial lainnya jika ada notifikasi baru.
3. Sistem operasi Android mendukung semua layanan Google.

2.10 Java

Java merupakan bahasa pemrograman berorientasi objek yang dikembangkan oleh Sun Microsystems dan diperkenalkan kepada masyarakat pada tahun 1995. Dasar bahasa yang digunakan dalam mengembangkan Java adalah C dan C++. Bahasa C dan C++ memiliki beberapa kelemahan sehingga James Gosling dan Patrick Naughton dari

Sun Microsystems memutuskan untuk membuat bahasa pemrograman baru yaitu Java. Walaupun dikembangkan dari bahasa C dan C++, Java tidak memiliki semua fitur yang terdapat pada C dan C++.

James Gosling dan Patrick Naughton ingin mendesain sebuah bahasa pemrograman yang tidak hanya dapat diandalkan dan portabel tetapi juga sederhana. Java memiliki beberapa fitur diantaranya adalah (Balagurusamy, 2010:11):

1. *Platform Independepent* dan *Portable*

   Program yang menggunakan bahasa Java dapat dipindahkan dengan mudah kesatu komputer ke komputer lainnya. Perubahan pada sistem operasi dan prosesor tidak akan berpengaruh banyak pada program.

2. *Distributed*

   Java didesain agar mudah didistribusikan melalui internet. Aplikasi Java dapat dibuka dan diakses dari internet sama mudahnya dengan diakses melalui sistem lokal. Hal ini memungkinkan beberapa *developer* yang menggunakan komputer berbeda dapat bekerja sama dalam mengerjakan satu buah projek.

3. *Multithreaded*

   Java dapat melakukan *multitasking* sehingga *developer* tidak perlu menunggu aplikasi untuk menyelesaikan satu tugas sebelum memulai tugas yang lainnya.

4. *Secure*

   Keamanan merupakan hal yang penting untuk bahasa pemrograman yang digunakan untuk melakukan pemrograman di internet. Sistem Java memverifikasi semua akses terhadap memori.

5. *Object Oriented*

   Hampir semua yang terdapat di Java adalah objek. Semua kode program dan data adalah objek atau *class*.

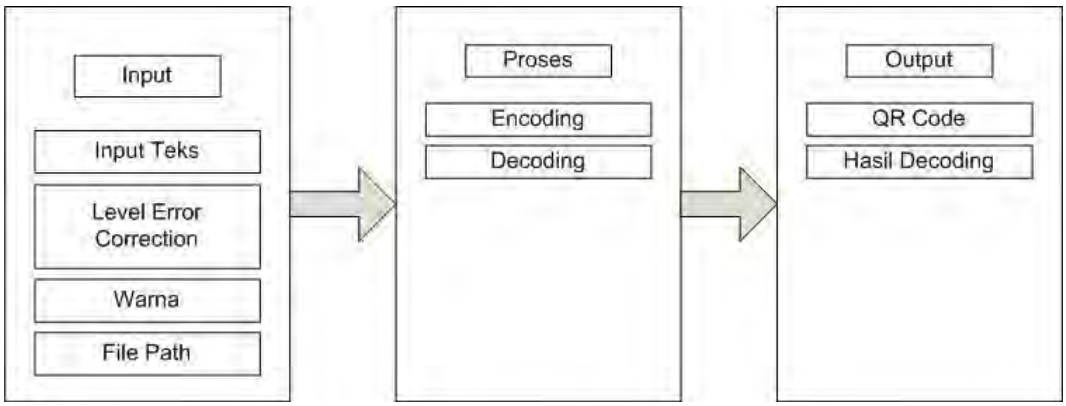
Zebra Crossing (ZXING) merupakan *open source library* yang dapat digunakan untuk melakukan pengolahan citra pada QR Code. ZXING mendukung format *barcode* satu dimensi ataupun dua dimensi yang dapat digunakan untuk pemrograman bahasa Java dan bahasa lainnya.

BAB III

ANALISIS DAN PERANCANGAN

3.1 Gambaran Umum Aplikasi

Aplikasi yang dirancang dalam Tugas Akhir ini merupakan aplikasi yang dikhususkan untuk telepon genggam yang memiliki sistem operasi *android.* Dengan menggunakan aplikasi ini pengguna dapat membuat serta membaca *Quick Responses Code* (QR Code) dengan mudah.



Gambar 3.1
Block Diagram

Gambar 3.1 memperlihatkan *input* yang perlu dimasukkan oleh pengguna pada aplikasi ini adalah teks, level *error correction*, warna dan *file path*. Untuk proses yang dilakukan oleh aplikasi ini adalah *encoding,* den *decoding*. Aplikasi ini menghasilkan gambar QR Code untuk proses *encoding* dan setelah dilakukan *decoding*, maka hasil *decoding* akan diperlihatkan.

Pada aplikasi ini pengguna hanya perlu memasukkan teks apa yang ingin dijadikan QR Code. Setelah teks dimasukkan maka aplikasi akan meng-*generate* data sehingga dihasilkanlah QR Code dan menampilkan gambar QR Code tersebut. Pengguna pun dapat merubah level *Error Correction* untuk QR Code tersebut dari *default* yaitu level L ke level lainnya. Dengan level *Error Correction* L merupakan level terendah dan level H merupakan level tertinggi.

Pengguna dapat melakukan perubahan warna pada QR Code tersebut sesuai dengan keinginan. Warna QR Code dapat diganti dari warna hitam dan putih, yang merupakan warna *default* menjadi warna-warna lain sesuai yang tersedia pada palet warna. Aplikasi ini juga dapat dimanfaatkan pengguna untuk mengganti warna QR Code tersebut menjadi warna gradasi sesuai dengan yang diinginkan. Agar warna QR Code menjadi gradasi maka pengguna hanya perlu memilih warna *foreground* atau *foreground* satu saja. *Foreground* digunakan untuk merubah warna QR Code bagian atas. Sedangkan untuk *foreground* satu digunakan untuk merubah warna QR Code bagian bawah. Arah gradasi ditentukan yaitu linear dari arah atas ke bawah. Setelah pengguna selesai melakukan pengeditan maka hasil QR Code tersebut dapat disimpan dalam bentuk *file* gambar.

Aplikasi ini juga dapat digunakan untuk membaca QR Code yang dibuat sebelumnya. Terdapat dua pilihan untuk membaca QR Code yaitu pengguna memilih data gambar QR Code yang ingin dibaca atau pengguna dapat melakukan pemindaian menggunakan kamera telepon genggam. Untuk melakukan pemindaian dibutuhkan aplikasi tambahan yaitu *barcode scanner*. Jika pengguna tidak memiliki aplikasi *barcode scanner* maka pengguna diminta untuk mengunduh aplikasi *barcode scanner* terlebih dahulu. Hasil pembacaan atau pemindaian QR Code akan ditampilkan dalam bentuk teks.

3.2 Spesifikasi Kebutuhan Perangkat Lunak

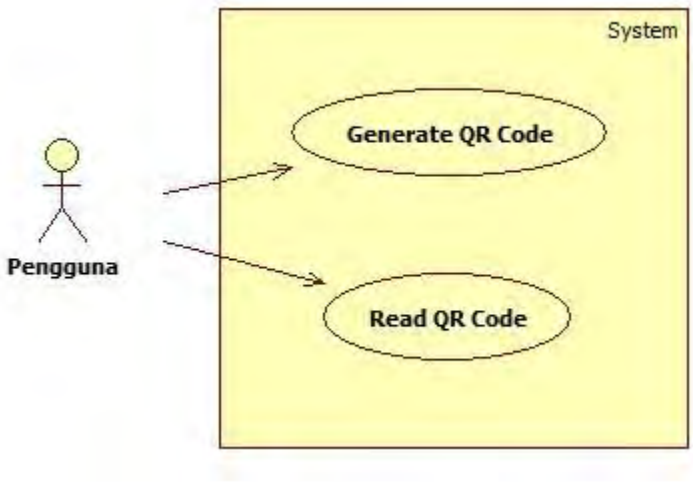
Aplikasi yang dirancang harus memiliki fitur atau spesifikasi sebagai berikut :

1. Dapat meng-*generate* teks menjadi QR Code sesuai dengan level *Error Correction* yang telah dipilih.
2. Hasil QR Code yang telah di-*generate* dapat disimpan menjadi sebuah data gambar.
3. Warna pada QR Code dapat dirubah menjadi gradasi.
4. Dapat membaca QR Code dari sebuah data gambar.
5. Dapat membaca QR Code melalui pemindaian menggunakan kamera.
6. Hasil pembacaan QR Code dapat ditampilkan.

3.3 Pemodelan Perangkat Lunak

3.3.1 *Use Case Diagram*

Dari use case diagram pada gambar 3.2 dapat diketahui bahwa aplikasi ini memiliki satu aktor, yaitu pengguna. Melalui aplikasi ini pengguna dapat meng-*generate* QR Code dan pengguna pun dapat membaca QR Code.



Gambar 3.2
Use Case Diagram

3.3.2 *Use Case Descriptions*

Terdapat beberapa *use case descripstion* dari *use case diagram* yang telah digambarkan pada gambar 3.2 yaitu :

1. Nama *use case* : *Generate* QR Code

   Aktor : Pengguna

   Pre-condition : *Menu Generate* QR Code ditampilkan

   Post-condition : Aplikasi ditutup.

Tabel 3.1
Skenario normal : *Generate* QR Code

Aksi Aktor	Reaksi Aplikasi
1. Memasukkan data yang akan dijadikan QR Code	
2. Pilih *Error Correction* “L”	
3. Klik “*Generate*”	
	4. Validasi teks
	5. Meng-*generate* QR *Code* dengan *Error Correction* level “L”

Aksi Aktor	Reaksi Aplikasi
	6. *Display* QR Code
7. Klik "*Save*"	
	8. Menyimpan QR Code

Alternatif 1 : *Error Correction* level M

Tabel 3.2
Skenario alternatif I : *Generate* QR Code

Aksi Aktor	Reaksi Aplikasi
1. Memasukkan data yang akan dijadikan QR Code	
2. Pilih *Error Correction* "M"	
3. Klik "*Generate*"	
	4. Validasi teks
	5. Meng-*generate* QR *Code* dengan *Error Correction* level "M"
	6. *Display* QR Code
7. Klik "*Save*"	
	8. Menyimpan QR Code

Alternatif 2 : *Error Correction* level Q

Tabel 3.3
Skenario alternatif II : *Generate* QR Code

Aksi Aktor	Reaksi Aplikasi
1. Memasukkan data yang akan dijadikan QR Code	
2. Pilih *Error Correction* "Q"	
3. Klik "*Generate*"	
	4. Validasi teks
	5. Meng-*generate* QR *Code* dengan *Error Correction* level "Q"
	6. *Display* QR Code
7. Klik "*Save*"	
	8. Menyimpan QR Code

Alternatif 3 : *Error Correction* level H

Tabel 3.4
Skenario alternatif III : *Generate* QR Code

Aksi Aktor	Reaksi Aplikasi
1. Memasukkan data yang akan dijadikan QR Code	
2. Pilih *Error Correction* "H"	
3. Klik "*Generate*"	

Aksi Aktor	Reaksi Aplikasi
	4. Validasi teks
	5. Meng-*generate* QR *Code* dengan *Error Correction* level "H"
	6. *Display* QR Code
7. Klik "*Save*"	
	8. Menyimpan QR Code

Alternatif 4 : mengganti warna *background*

Tabel 3.5
Skenario alternatif IV : *Generate* QR Code

Aksi Aktor	Reaksi Aplikasi
1 – 6 sama dengan skenario normal	
7. Klik "*Background*"	
	8. *Display* palet warna
9. Memilih warna	
	10. Merubah warna *background* dari QR Code
11. Klik "*Save*"	
	12. Menyimpan QR Code

Alternatif 5 : mengganti warna *foreground*

Tabel 3.6
Skenario alternatif V : *Generate* QR Code

Aksi Aktor	Reaksi Aplikasi
1 – 6 sama dengan skenario normal	
7. Klik "*Foreground*"	
	8. *Display* palet warna
9. Memilih warna	
	10. Merubah warna dari QR Code
11. Klik "*Save*"	
	12. Menyimpan QR Code

Alternatif 6 : mengganti warna *foreground 1*

Tabel 3.7
Skenario alternatif VI : *Generate* QR Code

Aksi Aktor	Reaksi Aplikasi
1 – 6 sama dengan skenario normal	
7. Klik "*Foreground 1*"	

Aksi Aktor	Reaksi Aplikasi
	8. *Display* palet warna
9. Memilih warna	
	10.Merubah warna dari QR Code
11.Klik "*Save*"	
	12.Menyimpan QR Code

Alternatif 7: mengganti warna *background* dan *foreground*

Tabel 3.8
Skenario alternatif VII : *Generate* QR Code

Aksi Aktor	Reaksi Aplikasi
1 – 6 sama dengan skenario normal	
7. Klik "*Background*"	
	8. *Display* palet warna
9. Memilih warna	
	10.Merubah warna *background* dari QR Code
11.Klik "*Foreground*"	
	12.*Display* palet warna
13.Memilih warna	
	14.Merubah warna QR Code
15.Klik "*Save*"	
	16.Menyimpan QR Code

Alternatif 8 : mengganti warna *background* dan *foreground 1*

Tabel 3.9
Skenario alternatif VIII: *Generate* QR Code

Aksi Aktor	Reaksi Aplikasi
1 – 6 sama dengan skenario normal	
7. Klik "*Background*"	
	8. *Display* palet warna
9. Memilih warna	
	10.Merubah warna *background* dari QR Code
11.Klik "*Foreground 1*"	
	12.*Display* palet warna
13.Memilih warna	
	14.Merubah warna QR Code
15.Klik "*Save*"	
	16.Menyimpan QR Code

Alternatif 9 : mengganti warna *foreground* dan *foreground 1*

Tabel 3.10
Skenario alternatif IX : *Generate* QR Code

Aksi Aktor	Reaksi Aplikasi
1 – 6 sama dengan skenario normal	
7. Klik "*Foreground*"	
	8. *Display* palet warna
9. Memilih warna	
	10.Merubah warna dari QR Code
11.Klik "*Foreground 1*"	
	12.*Display* palet warna
13.Memilih warna	
	14.Merubah warna QR Code
15.Klik "*Save*"	
	16.Menyimpan QR Code

Alternatif 10 : mengganti warna *background*, *foreground dan foreground 1*

Tabel 3.11
Skenario alternatif X : *Generate* QR Code

Aksi Aktor	Reaksi Aplikasi
1 – 6 sama dengan skenario normal	
7. Klik "*Background*"	
	8. *Display* palet warna
9. Memilih warna	
	10.Merubah warna *background* dari QR Code
11.Klik "*Foreground*"	
	12.*Display* palet warna
13.Memilih warna	
	14.Merubah warna QR Code
15.Klik "*Foreground* 1"	
	16.*Display* palet warna
17.Memilih warna	
	18.Merubah warna QR Code
19.Klik "*Save*"	
	20.Menyimpan QR Code

Alternatif 11 : teks tidak terisi

Tabel 3.12
Skenario alternatif XI: *Generate* QR Code

Aksi Aktor	Reaksi Aplikasi
1. Klik "*Generate*"	
	2. Validasi teks
	3. Menampilkan peringatan
	4. Menampilkan Menu *Generate* QR Code

2. Nama *use case* : *Read* QR Code

Akor : Pengguna

Pre-condition : *Menu Read* QR Code ditampilkan

Post-condition : Hasil pemindaian ditampilkan

Tabel 3.13
Skenario normal : *Read* QR Code

Aksi Aktor	Reaksi Aplikasi
1. Klik "*Browse*"	
2. Memilih *file* gambar QR Code yang akan dibaca	
	3. Membaca QR Code
	4. *Display* hasil pembacaan *QR Code*

Alternatif 1 : membaca QR Code menggunakan kamera, *barcode scanner* sudah ter-*install*

Tabel 3.14
Skenario alternatif I : *Read* QR Code

Aksi Aktor	Reaksi Aplikasi
1. Klik "*Scan*"	
	2. Memeriksa *barcode scanner*
	3. *Barcode scanner* diaktifkan
4. Mengarahkan kamera ke gambar simbol QR Code	
	5. Memindai QR Code
	6. Display hasil pembacaan *QR Code*

Alternatif 2 : membaca QR Code menggunakan kamera, *barcode scanner* belum ter-*install*, *download barcode scanner*.

Tabel 3.15
Skenario alternatif II : Read QR Code

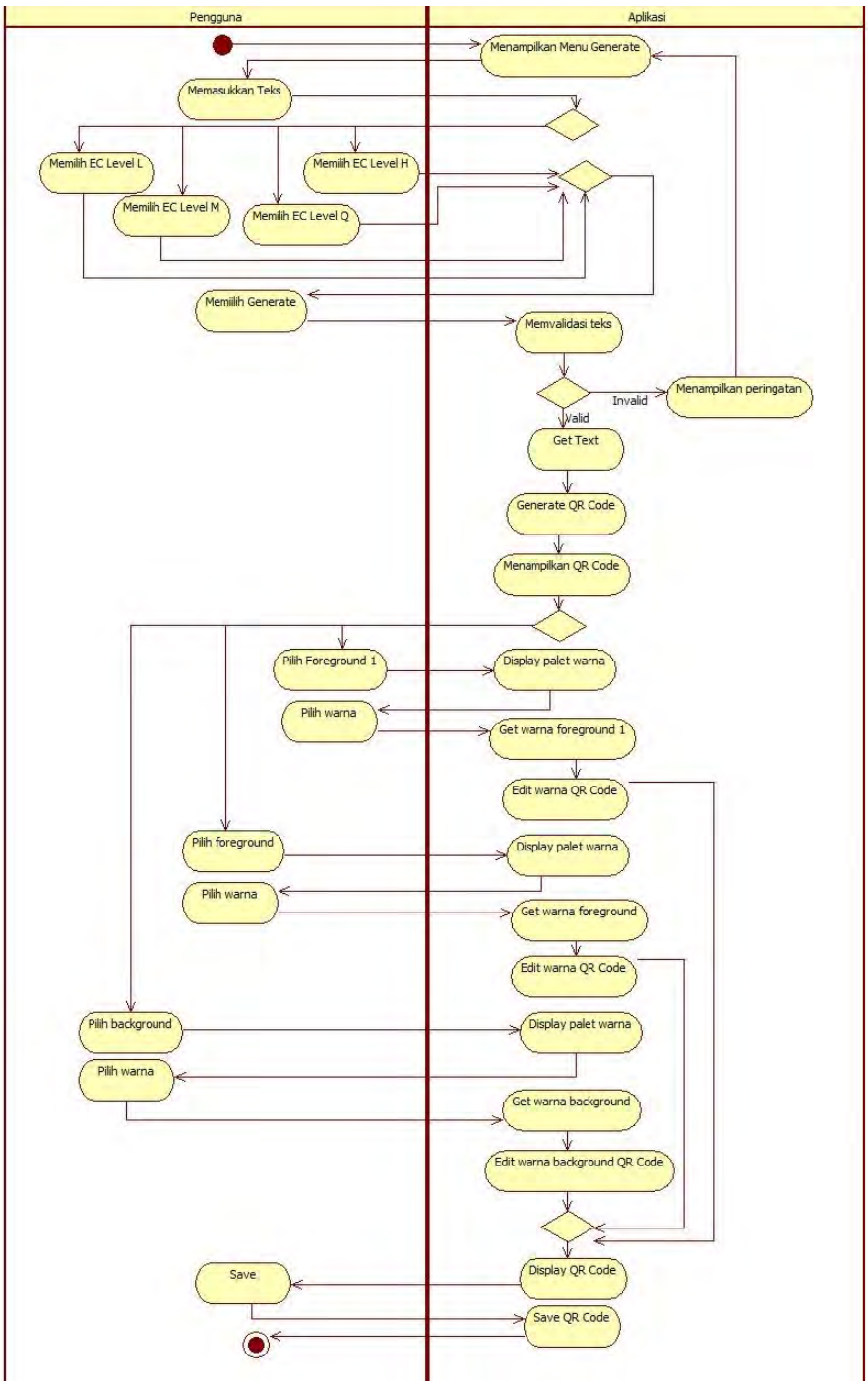
Aksi Aktor	Reaksi Aplikasi
1. Klik "*Scan*"	
	2. Memeriksa *barcode scanner*
	3. *Display download dialog*
4. Klik "Yes"	
	5. Mengarahkan ke *playstore*
6. *Download* dan *install barcode scanner*	
	7. *Barcode scanner* diaktifkan
8. Mengarahkan kamera ke gambar simbol QR Code	
	9. Memindai QR Code
	10.Display hasil pembacaan *QR Code*

Alternatif 3 : membaca QR Code menggunakan kamera, *barcode scanner* belum ter-*install,* tidak men-*download barcode scanner*

Tabel 3.16
Skenario alternatif III : *Read* QR Code

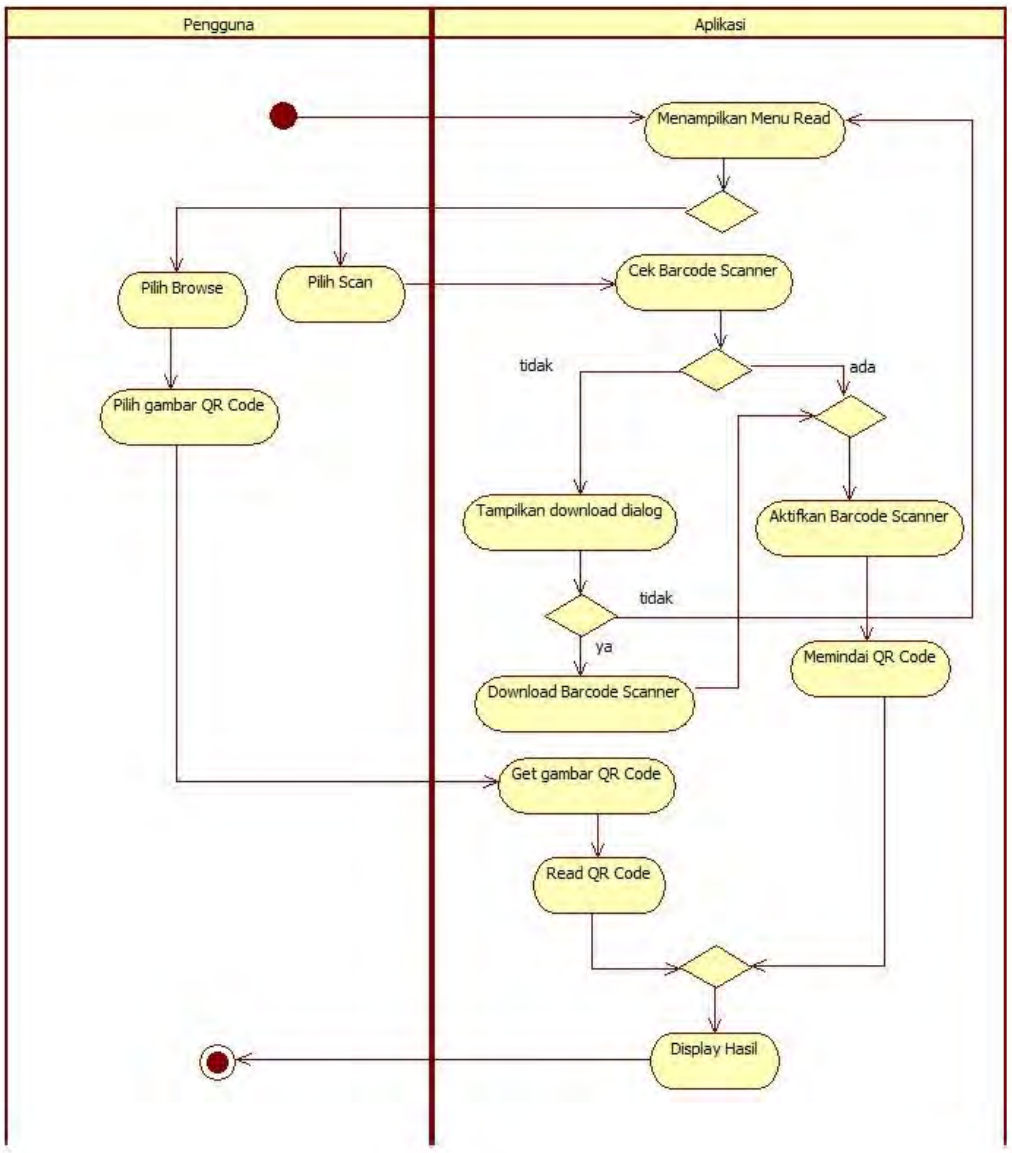
Aksi Aktor	Reaksi Aplikasi
1. Klik "*Scan*"	
	2. Memeriksa *barcode scanner*
	3. *Display download dialog*
4. Klik "*No*"	
	5. *Display menu Read* QR Code

3.3.3 *Activity Diagram*



Gambar 3.3
Activity Diagram Generate QR Code

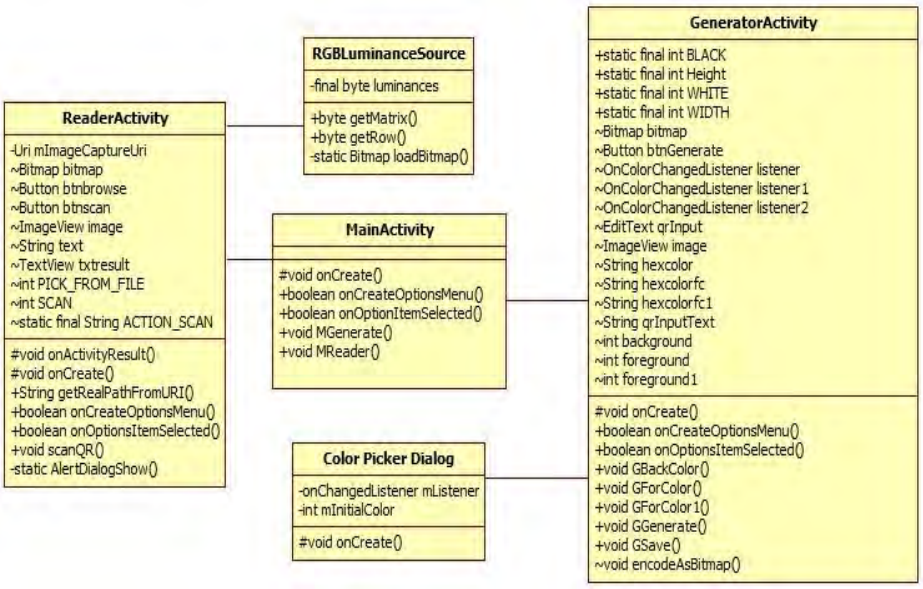
Dari gambar 3.3 diketahui bahwa saat aplikasi QR Code Generator dijalankan maka pengguna dapat memilih *error correction* yang akan digunakan. Setelah pengguna memilih tombol *generate* maka aplikasi akan memeriksa apakah teks *null* atau tidak. Jika *null,* maka aplikasi akan menampilkan pesan. Jika teks tidak *null* maka aplikasi akan meng-*generate* QR Code dan hasil tersebut akan ditampilkan. Setelah QR Code ditampilkan pengguna dapat memilih untuk mengganti warna dari QR Code tersebut atau langsung menyimpan QR Code tersebut. Pemilihan warna menggunakan palet warna yang sudah disiapkan.



Gambar 3.4
Activity Diagram Read QR Code

Sesuai dengan gambar 3.4 saat menu Read QR Code dijalankan, pengguna dapat melakukan pembacaan QR Code melalui gambar atau melakukan pemindaian dengan menggunakan kamera dari *handphone.* Untuk melakukan pemindaian dengan menggunakan kamera diperlukan aplikasi tambahan. Saat pengguna memilih pemindaian menggunakan kamera, maka aplikasi akan memeriksa ketersediaan aplikasi *barcode scanner*. Jika *smartphone* sudah memiliki aplikasi tersebut maka kamera dan aplikasi akan diaktifkan dan pemindaian akan langsung dilakukan. Jika *smartphone* belum memiliki aplikasi tersebut maka, aplikasi akan menampilkan dialog untuk download. Jika pengguna tidak ingin men-*download* aplikasi tersebut maka aplikasi akan kembali ke menu utama. Jika pengguna ingin men-*download* maka aplikasi akan mengalihkan ke *google playstore*.

3.3.4 *Class Diagram*

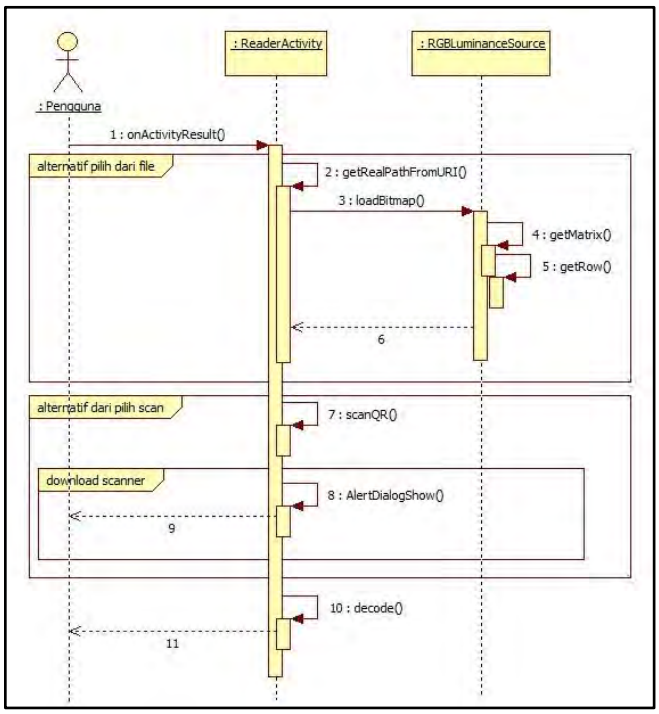


Gambar 3.5
Class Diagram

Gambar 3.5 menjelaskan bahwa aplikasi ini memiliki lima buah *class* yaitu MainActivity, GeneratorActivity, ReaderActivity, Color Picker Dialog, dan RGBLuminanceSource. MainActivity memiliki hubungan dengan GeneratorActivity dan

ReaderActivity. Sedangkan GeneratorActivity memiliki hubungan dengan Color Picker Dialog dan ReaderActivity memiliki hubungan dengan RGBLuminanceSource.

3.3.5 *Sequence Diagram*

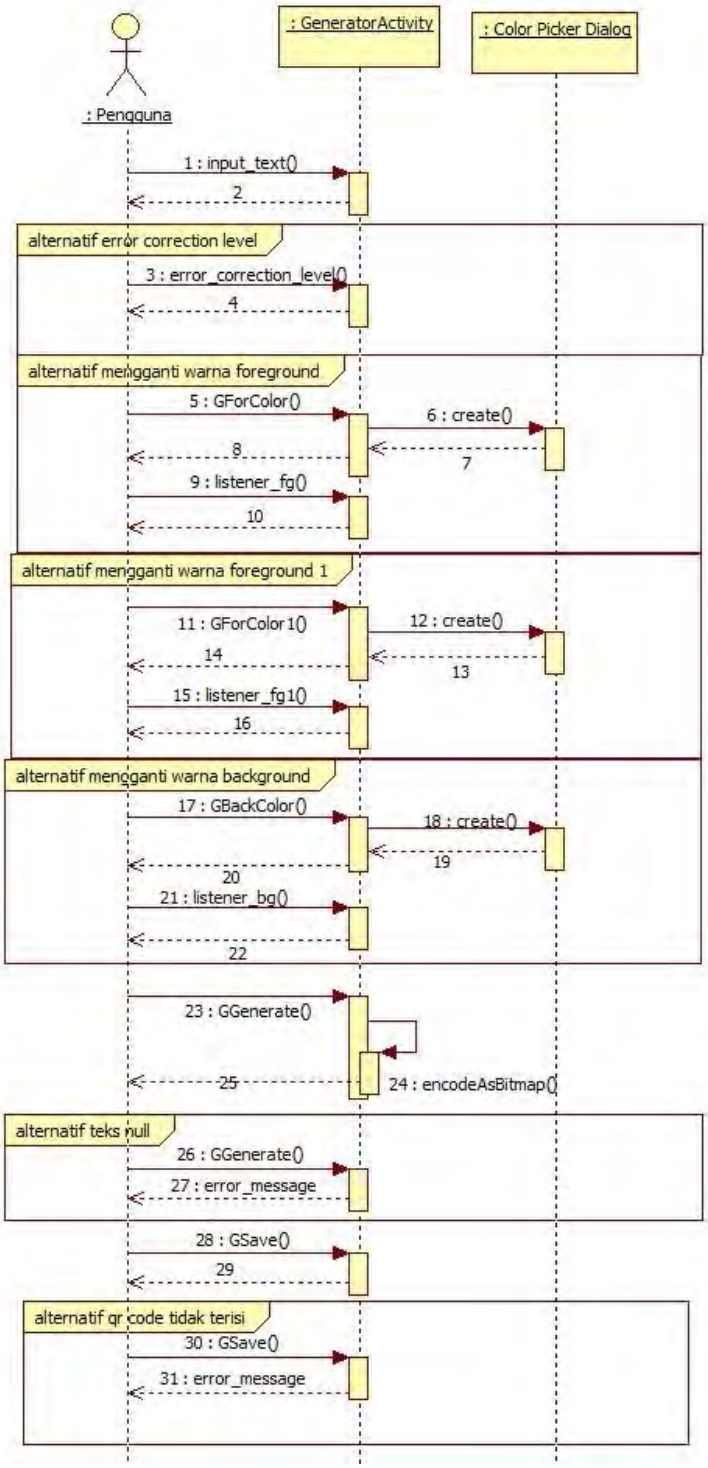


Gambar 3.6
Sequence Diagram Menu *Read* QR Code

Gambar 3.6 menjelaskan saat pengguna melakukan pembacaan QR Code dari gambar, maka aplikasi akan menggambil alamat dari gambar tersebut dan *class* RGBLunimanceSource akan melakukan grayscale warna sehingga dapat mengetahui pola dari QR Code tersebut. Setelah pola didapatkan maka hasil akan dialihkan ke *class* ReaderActivity dan akan dilakukan *decode* dan hasil akan ditampilkan.

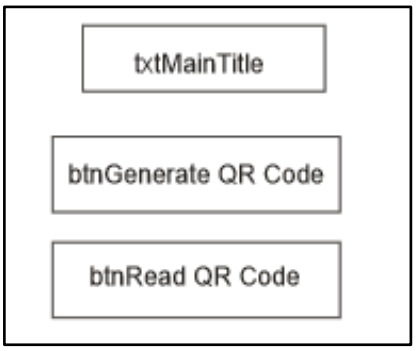
Pada gambar 3.7 diketahui setelah pengguna memasukkan teks, memilih level *error correction* dan hasil dari *generate* dari QR Code tersebut ditampilkan, pengguna dapat mengganti warna dari QR Code tersebut. Saat pengguna ingin mengganti warna dari QR Code tersebut maka *class* Color Picker Dialog akan menampilkan palet warna dan pengguna dapat memilih warna yang dinginkan dari palet warna tersebut. Setelah

warna dipilih maka *class* GeneratorActivity akan merubah warna dari QR Code sesuai dengan warna yang sudah dipilih.



Gambar 3.7
Sequence Diagram Menu *Generate* QR Code

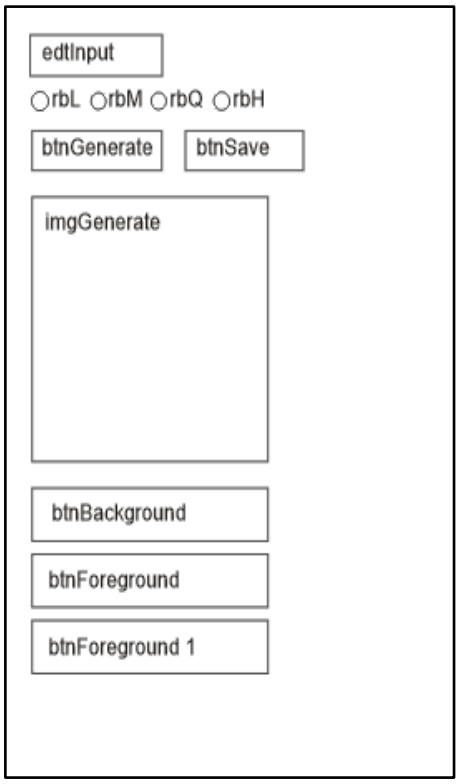
3.4 Rancangan Antar Muka



Gambar 3.8
Rancangan Menu Utama

Penjelasan mengenai komponen – komponen pada Gambar 3.8 adalah :

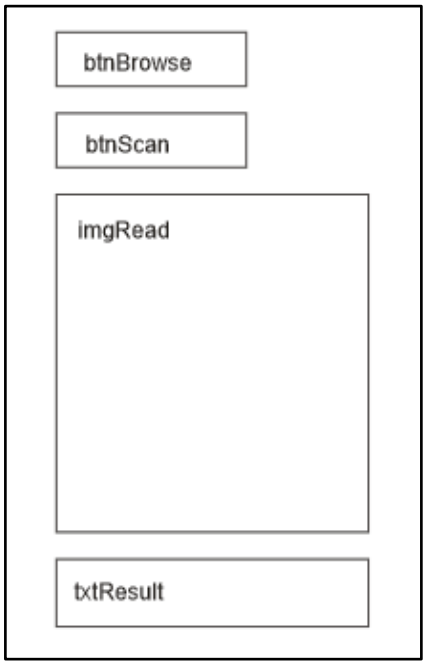
1. txtMainTitle : berfungsi untuk menampilkan tulisan nama aplikasi.
2. btnGenerate : berfungsi untuk membuka menu *generate* QR Code.
3. btnRead : berfungsi untuk membuka menu *read* QR Code.



Gambar 3.9
Rancangan Menu *Generate* QR Code

Penjelasan mengenai komponen – komponen pada Gambar 3.9 adalah :

1. edtInput : berfungsi untuk mengisikan teks yang akan dijadikan QR Code.
2. rbL : radio group untuk *Error Correction* level L.
3. rbM : radio group untuk *Error Correction* level M.
4. rbQ : radio group untuk *Error Correction* level Q.
5. rbH : radio group untuk *Error Correction* level H.
6. btnGenerateQRCode : berfungsi untuk meng-*generate* QR Code.
7. imgGenerate : berfungsi untuk menampilkan QR Code.
8. btnSave : berfungsi untuk menyimpan hasil *generate* QR Code.
9. btnBackground : berfungsi untuk membuka palet warna Background.
10. btnForeground : berfungsi untuk membuka palet warna Foreground.
11. btnForeground1 : berfungsi untuk membuka palet warna Foreground1.



Gambar 3.10
Rancangan Menu *Read* QR Code

Penjelasan mengenai komponen – komponen pada Gambar 3.10 adalah :

1. btnBrowse : berfungsi untuk memilih gambar QR Code yang akan dibaca.

2. btnScan : berfungsi untuk mengaktifkan *barcode scanner* untuk memindai QR Code.
3. imgRead : berfungsi untuk menampilkan *file* gambar QR Code yang dipilih.
4. txtResult : berfungsi untuk menampilkan hasil pembacaan QR Code.

BAB IV

IMPLEMENTASI DAN PENGUJIAN PERANGKAT LUNAK

4.1 Implementasi

Pada bab ini akan dijelaskan bagaimana seluruh perancangan dan analisa diimplementasikan untuk menghasilkan aplikasi dapat berjalan pada telepon genggam yang memiliki sistem operasi Android.

4.1.1 Lingkungan Pengembangan

Pada sub bab ini akan dijelaskan spesifikasi perangkat keras yang digunakan untuk membangun aplikasi ini. Pengembangan dilakukan di sebuah laptop dengan spesifikasi sebagai berikut:

1. Model : Dell Inspiron 14-3458
2. Processor : Intel(R) Core(TM) I3-4005U CPU
3. RAM : DDR3 4GB
4. Layar : 14" dengan resolusi 1366x768
5. VGA : Intel® HD Graphics 4400
6. HDD : 500 GB

Selain perangkat keras yang telah disebutkan, digunakan pula perangkat lunak yang mendukung dalam pengembangan game ini. Perangkat lunak tersebut yaitu :

1. Windows 8.1

   Sistem operasi yang digunakan untuk mengembangkan aplikasi ini adalah Windows 8.1. Sistem operasi ini dipilih karena umum dipakai dan dapat mendukung perangkat lunak lain yang digunakan.

2. Eclipse Mars 3.1

   Eclipse Mars 3.1 merupakan versi terbaru dari Eclipse IDE. Eclipse Mars 3.1 merupakan *software open source* sehingga dapat digunakan tanpa perlu membeli lisensi profesional.

3. ADT 23.0.6

   ADT merupakan *plug-in* untuk Eclipse IDE yang digunakan untuk membuat aplikasi telepon genggam yang memiliki sistem operasi Android.

4. Android SDK

   Android SDK digunakan sebagai *emulator* Android untuk menggantikan telepon genggam sehingga *developer* dapat langsung mencoba aplikasi yang dibuat seolah-olah aplikasi dijalankan langsung pada telepon genggam.

5. Java Development Kit 8.0

   JDK 8.0 digunakan untuk menjalankan *source code* Java pada Eclipse. Tanpa JDK, Eclipse tidak dapat menjalankan source code Java yang telah dibuat.

4.1.2 Tampilan Antar Muka

Hasil *screenshot* dari kemungkinan-kemungkinan tampilan aplikasi yang terjadi adalah sebagai berikut :

1. Menu Utama

   Saat dijalankan aplikasi memiliki tampilan seperti pada gambar 4.1

Gambar 4.1
Tampilan Menu Utama

Pada menu utama terdapat dua buah tombol yaitu :

a. Tombol *Generate* QR Code

   Tombol ini berfungsi untuk membuka menu *Generate* QR Code yang digunakan pada saat pengguna ingin membuat sebuah QR Code.

b. Tombol *Read* QR Code

Tombol ini berfungsi untuk membuka menu *Read* QR Code. Saat pengguna ingin memindai atau membaca sebuah QR Code pengguna menggunakan tombol ini.

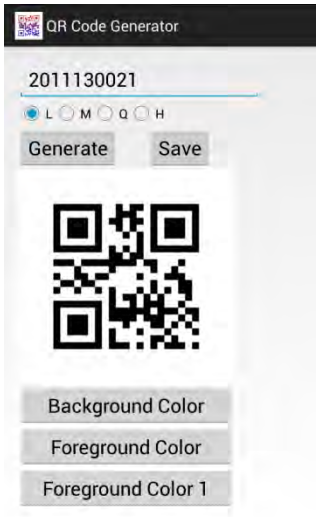
2. Menu Generate QR Code

Gambar 4.2
Tampilan Menu *Generate* QR Code

Menu *Generate* QR Code ditampilkan saat tombol *Generate* QR Code ditekan pada menu utama. Seperti pada gambar 4.2 menu ini memiliki sebuah *edit text*, lima buah tombol, empat buah *radio button* dan satu buah *image view*.

a. Tombol *Generate*

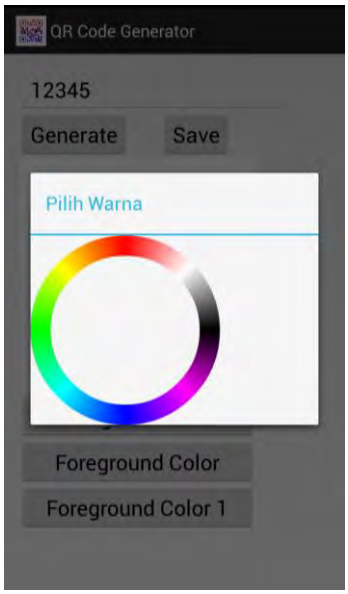
Setelah pengguna mengetikan teks yang akan dijadikan QR Code dan tombol ini ditekan maka gambar simbol QR Code



Gambar 4.3
Tampilan Menu *Generate* QR Code setelah meng-*generate* QR Code

b. Palet Warna

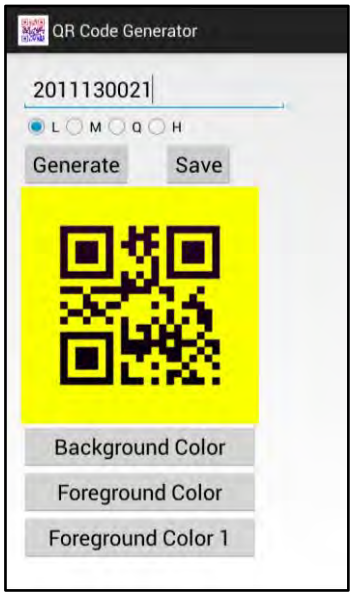
Tombol *Background Color*, *Foreground Color*, *Foreground Color* 1 memiliki palet warna seperti berikut :



Gambar 4.4
Tampilan Menu *Generate* QR Code dengan palet warna ditampilkan

c. Tombol *Background Color*

Setelah warna *background* QR Code dipilih maka hasil QR Code akan seperti pada gambar 4.5.



Gambar 4.5
Tampilan Menu *Generate* QR Code setelah warna *background* dipilih

d. Tombol *Foreground Color*

Gambar 4.6 memperlihatkan perubahan warna pada QR Code setelah warna *foreground* dipilih. Warna yang ditampilkan sesuai dengan yang dipilih oleh pengguna.

Gambar 4.6
Tampilan Menu *Generate* QR Code setelah warna *foreground* dipilih

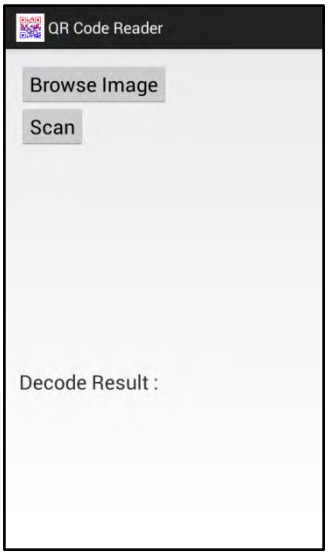
e. Tombol *Foreground Color* 1

Saat warna *foreground* 1 sudah dipilih maka QR Code kan berubah warna seperti pada gambar 4.7.

Gambar 4.7
Tampilan Menu *Generate* QR Code setelah warna *foreground* 1 dipilih

3. Menu *Read* QR Code

Menu *Read* QR Code memiliki dua buah tombol, satu buah *image view*, dan satu buah *text view*.



Gambar 4.8
Tampilan Menu *Read* QR Code

a. Tombol *Browse Image*

Gambar 4.9 memperlihatkan tampilan pada saat pembacaan dilakukan melalui data gambar.

Gambar 4.9
Tampilan Menu *Read* setelah pembacaan dilakukan

b. Tombol *Scan*

Tombol *Scan* digunakan untuk melakukan pemindaian menggunakan kamera. Setelah pemindaian dilakukan tampilan menu read akan menjadi seperti gambar 4.10

Gambar 4.10
Tampilan Menu *Read* setelah pemindaian dilakukan

4.2 Pengujian

Skenario pengujian terbagi menjadi tiga yaitu skenario pengujian untuk menu utama, skenario pengujian *generate* QR Code dan skenario pengujian *Read* QR Code.

Tabel 4.1
Tabel skenario pengujian menu utama

No	Skenario Pengujian	Ekspektasi	Output	Ket
1	Tombol *generate* QR Code ditekan	*Menu generate* QR Code ditampilkan	*Menu generate* QR Code ditampilkan	OK
2	Tombol *read* QR Code ditekan	*Menu read* QR Code ditampilkan	*Menu read* QR Code ditampilkan	OK

Tabel 4.2
Tabel skenario pengujian *generate* QR Code

No	Skenario Pengujian	Ekspektasi	Output	Ket
1	*Input* = *null* Tombol *generate* ditekan dengan *Error Correction* level L	Notifikasi ditampilkan	Notifikasi ditampilkan	OK
2	*Input* = 12345 (angka saja) Tombol *generate* ditekan dengan *Error Correction* level L	QR Code berhasil di-*generate* dan ditampilkan		OK
3	*Input* = LIKMI (huruf saja) Tombol *generate* ditekan dengan *Error Correction* level L	QR Code berhasil di-*generate* dan ditampilkan		OK
4	*Input* = likmi (huruf saja) Tombol *generate* ditekan dengan *Error Correction* level L	QR Code berhasil di-*generate* dan ditampilkan		OK
5	*Input* = 2011130021 Evi Hariyanti (huruf dan angka) Tombol *generate* ditekan dengan *Error Correction* level L	QR Code berhasil di-*generate* dan ditampilkan		OK

No	Skenario Pengujian	Ekspektasi	Output	Ket
6	*Input* =漢語 (*Hànyǔ*) Tombol *generate* ditekan dengan *Error Correction* level L	QR Code berhasil di-*generate* dan ditampilkan		OK
7	*Input* =ひらがな (*Hiragana*) Tombol *generate* ditekan dengan *Error Correction* level L	QR Code berhasil di-*generate* dan ditampilkan		OK
8	QR code tidak berhasil di-*generate* dan tombol *save* ditekan	Notifikasi ditampilkan	Notifikasi ditampilkan	OK
9	QR code berhasil di-*generate* dan tombol *save* ditekan	Notifikasi QR Code berhasil disimpan ditampilkan	Notifikasi QR Code berhasil disimpan ditampilkan	OK
10	QR code tidak berhasil di-*generate* dan tombol *background* ditekan	Notifikasi ditampilkan	Notifikasi ditampilkan	OK
11	QR code berhasil di-*generate* dan tombol *background* ditekan	Palet warna ditampilkan	Palet warna ditampilkan	OK
12	QR code berhasil di-*generate* Warna *background* telah dipilih dengan *Error Correction* level L	Warna *background* QR Code berubah		OK

No	Skenario Pengujian	Ekspektasi	Output	Ket
13	QR code tidak berhasil di-*generate* dan tombol *foreground* ditekan	Notifikasi ditampilkan	Notifikasi ditampilkan	OK
14	QR code berhasil di-*generate* dan tombol *foreground* ditekan	Palet warna ditampilkan	Palet warna ditampilkan	OK
15	QR code berhasil di-*generate* Warna *foreground* telah dipilih dengan *Error Correction* level L	Warna QR Code bagian atas berubah		OK
16	QR code tidak berhasil di-*generate* dan tombol *foreground* 1 ditekan	Notifikasi ditampilkan	Notifikasi ditampilkan	OK
17	QR code berhasil di-*generate* dan tombol *foreground* 1 ditekan	Palet warna ditampilkan	Palet warna ditampilkan	OK
18	QR code berhasil di-*generate* Warna *foreground* 1 telah dipilih dengan *Error Correction* level L	Warna QR Code bagian bawah berubah		OK
19	QR code berhasil di-*generate* Warna *background* dan *foreground* telah dipilih dengan *Error Correction* level L	Warna *background* dan QR Code bagian atas berubah		OK

No	Skenario Pengujian	Ekspektasi	Output	Ket
20	QR code berhasil di-*generate* Warna *background* dan *foreground* 1 telah dipilih dengan *Error Correction* level L	Warna *background* dan QR Code bagian bawah berubah		OK
21	QR code berhasil di-*generate* Warna *foreground* dan *foreground* 1 telah dipilih dengan *Error Correction* level L	Warna QR Code berubah		OK
22	QR code berhasil di-*generate* Warna *background*, *foreground* dan *foreground* 1 telah dipilih dengan *Error Correction* level L	Warna *background* dan QR Code berubah		OK
23	*Input* =abcde dengan *Error Correction* level L	QR Code berhasil di-*generate* dengan *Error Correction* level L		OK
24	*Input* =abcde dengan *Error Correction* level M	QR Code berhasil di-*generate* dengan *Error Correction* level M		OK

No	Skenario Pengujian	Ekspektasi	Output	Ket
25	*Input* =abcde dengan *Error Correction* level Q	QR Code berhasil di-*generate* dengan *Error Correction* level Q		OK
26	*Input* =abcde dengan *Error Correction* level H	QR Code berhasil di-*generate* dengan *Error Correction* level H		OK

Tabel 4.3
Tabel skenario pengujian *read* QR Code

No	Skenario Pengujian	Ekspektasi	Output	Ket
1	Tombol *browse* ditekan	*Pop-up file manager*	*Pop-up file manager*	OK
2	*File* gambar QR Code telah dipilih (*alphanumeric*)	*File* gambar ditampilkan, hasil pembacaan ditampilkan (*Decode Result* : abcde)	*File* gambar ditampilkan, hasil pembacaan ditampilkan (*Decode Result* : abcde)	OK

No	Skenario Pengujian	Ekspektasi	Output	Ket
3	*File* gambar QR Code telah dipilih (漢語(*Hànyǔ*))	File gambar ditampilkan, hasil pembacaan ditampilkan (*Decode Result* : 漢語)	*File* gambar ditampilkan, hasil pembacaan ditampilkan (*Decode Result* : (漢語)	OK
4	*File* gambar QR Code telah dipilih (ひらがな(*Hiragana*))	*File* gambar ditampilkan, hasil pembacaan ditampilkan (*Decode Result* : ひらがな)	*File* gambar ditampilkan, hasil pembacaan ditampilkan (*Decode Result* : (ひらがな)	OK
5	Tombol *scan* ditekan, Tidak ada *barcode scanner*	*Download dialog* ditampilkan	*Download dialog* ditampilkan	OK
6	*Download dialog* tekan “Yes”	Masuk *playstore*	Masuk *playstore*	OK
7	*Download dialog* tekan “No”	Kembali ke menu *read* QR Code	Kembali ke menu *read* QR Code	OK
8	Tombol *scan* ditekan, Ada *barcode scanner*	Pemindaian dilakukan	Pemindaian dilakukan	OK
9	Pemindaian berhasil dilakukan	Hasil pemindaian ditampilkan	Hasil pemindaian ditampilkan	OK

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Kesimpulan yang didapatkan oleh penulis setelah melalui semua tahap dalam mengembangkan aplikasi ini adalah :

1. Aplikasi *QR Code Generator* pada aplikasi android ini dibuat dengan menggunakan bahasa Java dengan memanfaatkan *library* ZXING. Proses *generate* QR Code dilakukan dengan menganalisa teks, lalu meng-*encode* teks tersebut menjadi biner. Lalu pengkodean *error correction* dilakukan sehingga dihasilkan struktur final dari dari teks tersebut. Struktur tersebut disimpan dalam matriks dan dilakukanlah *data masking*, dan ditambahkan informasi dari format dan versi QR Code tersebut.
2. Warna gradasi pada aplikasi *QR Code Generator* dibuat dengan memanfaatkan operasi pengolahan citra yaitu *gradient*. Proses gradasi warna dilakukan dengan cara meningkatkan nilai RGB dari setiap *pixel* secara bertahap sampai nilai RGB tersebut sama dengan nilai RGB lainnya.
3. Aplikasi *QR Code Reader* pada aplikasi android ini dibuat dengan menggunakan bahasa Java dengan memanfaatkan *library* ZXING. Proses pembacaan QR Code dilakukan dengan membaca dari data gambar dari *file manager* android atau dari kamera. Lalu *bitmap* gambar akan dibaca dengan memilah antara warna hitam dengan warna putih. Setelah itu informasi format dan versi QR Code akan dibaca, *data masking* akan dilepaskan dan pendeteksian *error* akan dilakukan. Jika terdapat *error* maka *error* akan dikoreksi lalu teks akan di-*decode*.

5.2 Saran

Aplikasi yang dihasilkan dalam Tugas Akhir ini dapat dikembangkan lagi karena masih banyak hal dari segi teknis pemrograman yang belum lengkap. Adapun saran yang diberikan untuk pengembangan Tugas Akhir ini diantaranya :

1. Membuat aplikasi QR Code dapat menerima input warna lebih banyak, dengan menyediakan palet warna yang lebih kompleks, atau dengan memasukkan kode RGB.
2. Mengembangkan fitur arah gradasi pada aplikasi *QR Code Generator* agar arah dan bentuk gradasi dapat lebih bervariasi dan dapat dirubah sesuai dengan keingginan pengguna.
3. Mengembangkan fitur *scan* pada aplikasi *QR Code Reader* agar saat akan dilakukan pemindaian, pengguna tidak perlu meng-*install* aplikasi tambahan.

DAFTAR PUSTAKA

Balagurusamy, Elappa, 2010, "*Programming with Java : A Primer Fourth Edition*", New Dehli: Tata-McGraw-Hill

Booch, Robert, dkk, 2007, "Object Oriented Analysis and Design with Applications Third Edition", Boston: Pearson Education, Inc.

Bovik, Al, 2009, "*The Essential Guide to Image Processing*", London: Elsevier Inc.

Burnette, Ed, 2005, "*Eclipse IDE Pocket Guide",* Sebastopol: O'Reilly Media,Inc.

Cinar, Onur, 2012, "*Android Apps with Eclipse",* New York: Apress

Gergenta, Marko, Nakamura, Masumi, 2014, "*Learning Android Second Edition",* Sebastopol: O'Reilly Media,Inc.

Hamilton, Kim, Miles, Russell, 2006, "*Learning UML 2.0"*, Sebastopol: O'Reilly Media,Inc.

Kato, Tan, dkk, 2010, "*Barcode for Mobile Devices*", New York: Cambridge University Press

Marsic, Ivan, 2012, "*Software Engineering*", New Jersey: Rutgers University.

Master.com, 2012,"Sekali Baca Langsung Inget Mengupas Lengkap All About ANDROID", Jakarta: Kuncikom

Meng, Lee Wei, 2013, "Android Application Development Cookbook : 93 Recipes for Building Winning Apps", Indianapolis: John Wiley & Sons Ltd.

Mulyanto, Aunur Rofic, 2008, "*Rekayasa Pearangkat Lunak Jilid 1"*, Direktorat Jenderal Manajemen Pendidikan Dasar dan Menengah Departemen Pendidikan Nasional.

O'Docherty, Mike, 2005, "Object-Oriented Analysis and Design Understanding System Development with UML 2.0", Indianapolis: John Wiley & Sons Ltd.

Pakhira, Malay Kumar, 2011, "*Digital Image Processing and Patern Recognition",* New Dehli: PHI Learning Private Limited

Pressman, Roger, 2010, "Software Engineering: A Practitioner's Approach Seventh Edition", New York: McGraw-Hill Companies, Inc.

Price, Kella, "*QR Codes for Trainers",* Infoline Volume 30 Issue 1301 January 2013, United States of America

Putra, Dharma, 2010, *"Pengolahan Citra Digital",* Yogyakarta: ANDI

Sengupta, Probal, Chudhuri, Bidyut Baran, 2004, "*Object-Oriented Programming: Fundamentals And Applications",* India: Asoke K. Ghosh.

Simarmata, Janer, 2010, *"Rekayasa Perangkat Lunak 2",* Yogyakarta: ANDI

Sommerville, Ian, 2011, "*Software Engineering Ninth Edition*", Boston: Pearson Education, Inc.

Wahyono, Teguh, 2010, "*Membuat Sendiri Aplikasi Dengan Memanfaatkan Barcode",* Jakarta: Elex Media Komputindo

Whitten, Jeffrey L, Bentley, Lonnie D, 2007, "*System Analysis and Design Methods*", New York: The McGraw-Hill Companies, Inc.

Winter, Mick, 2010, "*Scan Me: Everybody's Guide to the Magical World of QR Codes***",** Napa: Westong Publishing

http://mui.or.id/mui/homepage/berita/berita-singkat/qr-code-permudah-konsumen-tahu-kehalalan-produk.html diakes tanggal 13 Agustus 2015 jam 19:33 WIB

http://www.qrcode.com/en/history/ diakes tanggal 16 Agustus 2015 jam 19:40 WIB

http://www.onbarcode.com/qr_code/ diakes tanggal 20 September 2015 jam 23:08 WIB

https://github.com/zxing/zxing diakses tanggal 28 November 2015 jam 11:47 WIB

LAMPIRAN
LISTING PROGRAM

1. MainActivity.xml

```
<
    RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:tools="http://schemas.android.com/tools"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:paddingBottom="@dimen/activity_vertical_margin"
        android:paddingLeft="@dimen/activity_horizontal_margin"
        android:paddingRight="@dimen/activity_horizontal_margin"
        android:paddingTop="@dimen/activity_vertical_margin"
        tools:context="com.qrcode.ta.MainActivity"
>

<
    TextView
        android:id="@+id/txtMTitle"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="34dp"
        android:text="@string/txtMTitle"
        android:textSize="35sp"
        android:layout_gravity="center"
/>

 <
    Button
        android:id="@+id/btnMRead"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/btnMGenerate"
        android:layout_alignRight="@+id/btnMGenerate"
        android:layout_below="@+id/txtMTitle"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="138dp"
        android:onClick="MReader"
        android:text="@string/BtnMRead"
        android:layout_gravity="center"
        android:textSize="25sp"
/>

<
    Button
         android:id="@+id/btnMGenerate"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:layout_alignBottom="@+id/btnMRead"
         android:layout_centerHorizontal="true"
         android:layout_marginBottom="53dp"
         android:onClick="MGenerate"
         android:text="@string/BtnMGenerate"
         android:textSize="25sp"
/>

</RelativeLayout>
```

2. activity_generator.xml

```
<
    RelativeLayout xmlns:tools="http://schemas.android.com/tools"
          xmlns:android="http://schemas.android.com/apk/res/android"
          android:layout_width="match_parent"
          android:layout_height="match_parent"
          android:paddingBottom="@dimen/activity_vertical_margin"
          android:paddingLeft="@dimen/activity_horizontal_margin"
          android:paddingRight="@dimen/activity_horizontal_margin"
```

```
        android:paddingTop="@dimen/activity_vertical_margin"
        android:scrollbars="horizontal|vertical"
        tools:context="com.qrcode.ta.MainActivity"
>

    <
        ScrollView
            android:id="@+id/scrollView1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
    >

        <
            RelativeLayout
                android:layout_width="match_parent"
                android:layout_height="match_parent"
            >

            <
                HorizontalScrollView
                    android:id="@+id/horizontalScrollView1"
                    android:layout_width="match_parent"
                    android:layout_height="match_parent"
                >

                <
                    RelativeLayout
                        android:layout_width="match_parent"
                        android:layout_height="match_parent"
                    >

                    <
                        EditText
                            android:id="@+id/edtGInput"
                            android:layout_width="280dp"
                            android:layout_height="wrap_content"
                            android:hint="@string/edtGInput"
                            android:textSize="25sp"
                        >
                        <requestFocus />
                        </EditText>

                        <
                            ImageView
                                android:id="@+id/imgG"
                                android:layout_width="250dp"
                                android:layout_height="250dp"
                                android:layout_below="@+id/btnGGenerate"
                        />

                        <
                            Button
                                android:id="@+id/btnGBackColor"
                                android:layout_width="250dp"
                                android:layout_height="wrap_content"
                                android:layout_below="@+id/imgG"
                                android:onClick="GBackColor"
                                android:text="@string/btnGBackColor"
                                android:textSize="25sp"
                        />

                        <
                            Button
                                android:id="@+id/btnGForColor"
                                android:layout_width="250dp"
                                android:layout_height="wrap_content"
                                android:layout_below="@+id/btnGBackColor"
                                android:onClick="GForColor"
```

```
        android:text="@string/btnGForColor"
        android:textSize="25sp"
/>

<
    Button
        android:id="@+id/btnGForColor1"
        android:layout_width="250dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/btnGForColor"
        android:onClick="GForColor1"
        android:text="@string/btnGForColor1"
        android:textSize="25sp"
/>

<
    Button
        android:id="@+id/btnGSave"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/rbgEC"
        android:layout_marginLeft="150dp"
        android:onClick="GSave"
        android:text="@string/btnGSave"
        android:textSize="25sp"
/>

<
    Button
        android:id="@+id/btnGGenerate"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/rbgEC"
        android:onClick="GGenerate"
        android:text="@string/btnGGenerate"
        android:textSize="25sp"
/>

    <
        RadioGroup
            android:id="@+id/rbgEC"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:orientation="horizontal"
            android:layout_below="@+id/edtGInput"
     >

        <
            RadioButton
                android:id="@+id/rbL"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:checked="true"
                android:text="L"
        />

        <
            RadioButton
                android:id="@+id/rbM"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="M"
        />

        <
            RadioButton
                android:id="@+id/rbQ"
```

```
                                android:layout_width="wrap_content"
                                android:layout_height="wrap_content"
                                android:text="Q"
                            />

                            <
                                RadioButton
                                android:id="@+id/rbH"
                                android:layout_width="wrap_content"
                                android:layout_height="wrap_content"
                                android:text="H
                             />
                        </RadioGroup>
                    </RelativeLayout>
                </HorizontalScrollView>
            </RelativeLayout>
        </ScrollView>
    </RelativeLayout>
```

3. activity_read.xml

```
<
    RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:tools="http://schemas.android.com/tools"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:paddingBottom="@dimen/activity_vertical_margin"
        android:paddingLeft="@dimen/activity_horizontal_margin"
        android:paddingRight="@dimen/activity_horizontal_margin"
        android:paddingTop="@dimen/activity_vertical_margin"
        android:scrollbars="horizontal|vertical"
        tools:context="com.qrcode.ta.MainActivity"
>

    <
        ScrollView
            android:id="@+id/scrollView1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
        >

        <
            RelativeLayout
                android:layout_width="match_parent"
                android:layout_height="match_parent"
            >

            <
                HorizontalScrollView
                    android:id="@+id/horizontalScrollView1"
                    android:layout_width="match_parent"
                    android:layout_height="match_parent"
                >

                <
                    RelativeLayout
                        android:layout_width="match_parent"
                        android:layout_height="match_parent"
                    >

                    <
                        Button
                            android:id="@+id/btnRBrowse"
                            android:layout_width="wrap_content"
                            android:layout_height="wrap_content"
                            android:text="@string/BtnRBrowse"
                            android:textSize="25sp"
                        />
```

```
                    <
                        ImageView
                            android:id="@+id/imgR"
                            android:layout_width="250dp"
                            android:layout_height="250dp"
                            android:layout_below="@+id/btnRScan"
                    />

                    <
                        TextView
                            android:id="@+id/txtRResult"
                            android:layout_width="280dp"
                            android:layout_height="wrap_content"
                            android:text="@string/txtRResult"
                            android:layout_below="@+id/imgR"
                            android:textSize="25sp"
                    />

                    <
                        Button
                            android:id="@+id/btnRScan"
                            android:layout_width="wrap_content"
                            android:layout_height="wrap_content"
                            android:layout_below="@+id/btnRBrowse"
                            android:layout_alignParentLeft="true"
                            android:text="@string/btnRScan"
                            android:textSize="25sp"
                            android:onClick="scanQR"
                    />

                </RelativeLayout>
            </HorizontalScrollView>
        </RelativeLayout>
    </ScrollView>
</RelativeLayout>
```

4. string.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="app_name">
        QR Code
    </string>

    <string name="action_settings">
        Settings
    </string>

    <string name="txtRTitle">
        QR Code Reader
    </string>

    <string name="txtGTitle">
        QR Code Generator
    </string>

    <string name="txtMTitle">
        QR Code
    </string>

    <string name="BtnMGenerate">
        Generate QR Code
    </string>

    <string name="BtnMRead">
        Read QR Code
```

```
    </string>

    <string name="BtnRBrowse">
        Browse Image
    </string>

    <string name="txtRResult">
        Decode Result :
    </string>

    <string name="edtGInput">
        Masukkan Text
    </string>

    <string name="btnGGenerate">
        Generate
    </string>

    <string name="btnGSave">
        Save
    </string>

    <string name="btnRScan">
        Scan
    </string>

    <string name="btnGBackColor">
        Background Color
    </string>

    <string name="btnGForColor">
        Foreground Color
    </string>

    <string name="btnGForColor1">
        Foreground Color 1
    </string>

</resources>
```

5. MainActivity.java

```
package com.qrcode.ta;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;

public class MainActivity extends Activity
{

    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public void MGenerate(View v)
    {
        Intent intent = new Intent(getApplicationContext(), GeneratorActivity.class);
        startActivity(intent);
    }

    public void MReader(View v)
    {
        Intent intent = new Intent(getApplicationContext(), ReaderActivity.class);
```

```
                startActivity(intent);
            }

        }
```

6. ColorPickerDialog.java

```
        package com.qrcode.ta;

        import android.app.Dialog;
        import android.content.Context;
        import android.graphics.Canvas;
        import android.graphics.Color;
        import android.graphics.ColorMatrix;
        import android.graphics.Paint;
        import android.graphics.RectF;
        import android.graphics.Shader;
        import android.graphics.SweepGradient;
        import android.os.Bundle;
        import android.view.MotionEvent;
        import android.view.View;

        public class ColorPickerDialog extends Dialog
        {
            public interface OnColorChangedListener
            {
                void colorChanged(int color);
            }

            private OnColorChangedListener mListener;
            private int mInitialColor;

            private static class ColorPickerView extends View
            {
                private Paint mPaint;
                private Paint mCenterPaint;
                private final int[] mColors;
                private OnColorChangedListener mListener;

                ColorPickerView(Context c, OnColorChangedListener l, int color)
                {
                    super(c);
                    mListener = l;

                    mColors = new int[]
                    {
                        0xFF000000, 0xFFFF00FF, 0xFF0000FF, 0xFF00FFFF, 0xFF00FF00,  0xFFFFFF00, 0xFFFF0000,
                        0xFFFFFFFF, 0xFF000000
                    };

                    Shader s = new SweepGradient(0, 0, mColors, null);
                    mPaint = new Paint(Paint.ANTI_ALIAS_FLAG);
                    mPaint.setShader(s);
                    mPaint.setStyle(Paint.Style.STROKE);
                    mPaint.setStrokeWidth(32);
                    mCenterPaint = new Paint(Paint.ANTI_ALIAS_FLAG);
                    mCenterPaint.setColor(color);
                    mCenterPaint.setStrokeWidth(5);
                }

                private boolean mTrackingCenter;
                private boolean mHighlightCenter;

                @Override
                protected void onDraw(Canvas canvas)
                {
                    float r = CENTER_X - mPaint.getStrokeWidth()*0.5f;
                    canvas.translate(CENTER_X, CENTER_X);
```

```
        canvas.drawOval(new RectF(-r, -r, r, r), mPaint);
        canvas.drawCircle(0, 0, CENTER_RADIUS, mCenterPaint);
        if (mTrackingCenter)
        {
            int c = mCenterPaint.getColor();
            mCenterPaint.setStyle(Paint.Style.STROKE);
            if (mHighlightCenter)
                {
                    mCenterPaint.setAlpha(0xFF);
                }
            else
                {
                    mCenterPaint.setAlpha(0x80);
                }
            canvas.drawCircle(0, 0,  CENTER_RADIUS + mCenterPaint.getStrokeWidth(),  mCenterPaint);
            mCenterPaint.setStyle(Paint.Style.FILL);
            mCenterPaint.setColor(c);
        }
    }

    @Override
    protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec)
    {
        setMeasuredDimension(CENTER_X*2, CENTER_Y*2);
    }

    private static final int CENTER_X = 150;
    private static final int CENTER_Y = 150;
    private static final int CENTER_RADIUS = 75;

    private int floatToByte(float x)
    {
        int n = java.lang.Math.round(x);
        return n;
    }

    private int pinToByte(int n)
    {
        if (n < 0)
        {
            n = 0;
        }
        else if (n > 255)
        {
            n = 255;
        }
        return n;
    }

    private int ave(int s, int d, float p)
    {
        return s + java.lang.Math.round(p * (d - s));
    }

    private int interpColor(int colors[], float unit)
    {
        if (unit <= 0)
        {
            return colors[0];
        }

        if (unit >= 1)
        {
            return colors[colors.length - 1];
        }

        float p = unit * (colors.length - 1);
        int i = (int)p;
```

```
        p -= i;

        int c0 = colors[i];
        int c1 = colors[i+1];
        int a = ave(Color.alpha(c0), Color.alpha(c1), p);
        int r = ave(Color.red(c0), Color.red(c1), p);
        int g = ave(Color.green(c0), Color.green(c1), p);
        int b = ave(Color.blue(c0), Color.blue(c1), p);
        return Color.argb(a, r, g, b);
    }

    private int rotateColor(int color, float rad)
    {
        float deg = rad * 180 / 3.1415927f;
        int r = Color.red(color);
        int g = Color.green(color);
        int b = Color.blue(color);
        ColorMatrix cm = new ColorMatrix();
        ColorMatrix tmp = new ColorMatrix();
        cm.setRGB2YUV();
        tmp.setRotate(0, deg);
        cm.postConcat(tmp);
        tmp.setYUV2RGB();
        cm.postConcat(tmp);
        final float[] a = cm.getArray();
        int ir = floatToByte(a[0] * r + a[1] * g + a[2] * b);
        int ig = floatToByte(a[5] * r + a[6] * g + a[7] * b);
        int ib = floatToByte(a[10] * r + a[11] * g + a[12] * b);
        return Color.argb(Color.alpha(color), pinToByte(ir),  pinToByte(ig), pinToByte(ib));
    }

    private static final float PI = 3.1415926f;

    @Override
    public boolean onTouchEvent(MotionEvent event)
    {
        float x = event.getX() - CENTER_X;
        float y = event.getY() - CENTER_Y;
        boolean inCenter = java.lang.Math.sqrt(x*x + y*y) <= CENTER_RADIUS;

        switch (event.getAction())
        {
            case MotionEvent.ACTION_DOWN:
            mTrackingCenter = inCenter;
            if (inCenter)
            {
                mHighlightCenter = true;
                invalidate();
                break;
            }

            case MotionEvent.ACTION_MOVE:
            if (mTrackingCenter)
            {
                if (mHighlightCenter != inCenter)
                {
                    mHighlightCenter = inCenter;
                    invalidate();
                }
            }
            else
            {
                float angle = (float)java.lang.Math.atan2(y, x);
                float unit = angle/(2*PI);
                if (unit < 0)
                {
                    unit += 1;
                }
```

```
                    mCenterPaint.setColor(interpColor(mColors, unit));
                    invalidate();
                }
                break;

                case MotionEvent.ACTION_UP:
                if (mTrackingCenter)
                {
                    if (inCenter)
                    {
                        mListener.colorChanged(mCenterPaint.getColor());
                    }
                    mTrackingCenter = false;
                    invalidate();
                }
                break;
            }
        return true;
        }
    }

    public ColorPickerDialog(Context context,  OnColorChangedListener listener,  int initialColor)
        {
            super(context);
            mListener = listener;
            mInitialColor = initialColor;
        }

    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        OnColorChangedListener l = new OnColorChangedListener()
        {
            public void colorChanged(int color)
            {
                mListener.colorChanged(color);
                dismiss();
            }
        };
        setContentView(new ColorPickerView(getContext(), l, mInitialColor));
        setTitle("Pilih Warna");
    }
}
```

7. GeneratorActivity.java

```
package com.qrcode.ta;

import java.io.File;
import java.io.FileOutputStream;
import java.io.OutputStream;
import java.io.UnsupportedEncodingException;
import java.util.Date;
import java.util.HashMap;
import java.util.Map;

import com.google.zxing.BarcodeFormat;
import com.google.zxing.EncodeHintType;
import com.google.zxing.MultiFormatWriter;
import com.google.zxing.WriterException;
import com.google.zxing.common.BitMatrix;
import com.google.zxing.qrcode.decoder.ErrorCorrectionLevel;
import com.qrcode.ta.ColorPickerDialog.OnColorChangedListener;

import android.app.Activity;
import android.graphics.Bitmap;
import android.graphics.Color;
```

```
import android.os.Bundle;
import android.os.Environment;
import android.text.format.DateFormat;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.RadioButton;
import android.widget.Toast;

public class GeneratorActivity extends Activity implements OnColorChangedListener
{
    public final static int WHITE = 0xFFFFFFFF;
    public final static int BLACK = 0xFF000000;
    public final static int WIDTH = 250;
    public final static int HEIGHT = 250;

    ImageView image;
    EditText qrInput;
    Button btnGenerate;
    String qrInputText;
    Bitmap bitmap;
    String hexcolor;
    String hexcolorfc;
    String hexcolorfc1;
    int background;
    int foreground;
    int foreground1;

    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_generator);

        background=WHITE;
        foreground=BLACK;
        foreground1=BLACK;
        image = (ImageView) findViewById(R.id.imgG);
        qrInput = (EditText) findViewById(R.id.edtGInput);
        btnGenerate = (Button) findViewById(R.id.btnGGenerate);
        qrInputText = qrInput.getText().toString();
    }

    public void GGenerate(View v)
    {
        qrInputText = qrInput.getText().toString();
        if(qrInputText.equals(""))
        {
            Toast.makeText(getApplicationContext(), "Masukkan teks", Toast.LENGTH_SHORT).show();return;
        }
        else try
        {
            bitmap = encodeAsBitmap(qrInputText);
            image.setImageBitmap(bitmap);
        }
        catch (WriterException e)
        {
            e.printStackTrace();
        }
    }

    @SuppressWarnings("unchecked")
    Bitmap encodeAsBitmap(String qrInputText) throws WriterException
    {
        String charset = "SHIFT_JIS";
        Map hintMap = new HashMap();
        RadioButton rl = (RadioButton) findViewById(R.id.rbL);
```

```
RadioButton rm = (RadioButton) findViewById(R.id.rbM);
RadioButton rq = (RadioButton) findViewById(R.id.rbQ);
RadioButton rh = (RadioButton) findViewById(R.id.rbH);

if(rl.isChecked())
{
      hintMap.put(EncodeHintType.ERROR_CORRECTION, ErrorCorrectionLevel.L);
}
else if(rm.isChecked())
{
      hintMap.put(EncodeHintType.ERROR_CORRECTION, ErrorCorrectionLevel.M);
}
else if(rq.isChecked())
{
      hintMap.put(EncodeHintType.ERROR_CORRECTION, ErrorCorrectionLevel.Q);
}
else if(rh.isChecked())
{
      HintMap.put(EncodeHintType.ERROR_CORRECTION, ErrorCorrectionLevel.H);
}

BitMatrix result=null;

try
{
      try
      {
            result = new MultiFormatWriter().encode(new String(qrInputText.getBytes(charset), charset),
                        BarcodeFormat.QR_CODE, WIDTH, HEIGHT, hintMap);
      }
      catch (UnsupportedEncodingException e)
            {
                  // TODO Auto-generated catch blocke.printStackTrace();
            }
}
      catch (IllegalArgumentException iae)
      {
            return null;
       }

      int w = result.getWidth();
      int h = result.getHeight();
      int[] pixels = new int[w * h];
      double r1=Color.red(foreground1);
       double r0=Color.red(foreground);
      double g1=Color.green(foreground1);
       double g0=Color.green(foreground);
       double b1=Color.blue(foreground1);d
       ouble b0=Color.blue(foreground);
      double adjust_r =(r1-r0)/h;
      double adjust_g = (g1-g0)/h;
      double adjust_b = (b1-b0)/h;
      double r=Color.red(foreground);
      double g=Color.green(foreground);
      double b=Color.blue(foreground);

      for (int y = 0; y < h; y++)
{
       int offset = y * w;
       for (int x = 0; x < w; x++)
       {
            pixels[offset + x] = (Integer) (result.get(x, y) ? Color.rgb((int)r,(int)g,(int)b) : background);
       }

       r+=adjust_r;
       g+=adjust_g;
       b+=adjust_b;
}
```

```
        bitmap = Bitmap.createBitmap(w, h, Bitmap.Config.ARGB_4444);
        bitmap.setPixels(pixels, 0, w, 0, 0, w, h);
        return bitmap;
    }

    public void GSave(View v)
    {
        OutputStream output;
        qrInputText = qrInput.getText().toString();

        if(qrInputText.equals(""))
        {
            Toast.makeText(getApplicationContext(), "Masukkan teks", Toast.LENGTH_SHORT).show();return;
        }
        else if(bitmap==null)
        {
            Toast.makeText(getApplicationContext(), "Tekan generate terlebih dahulu",
            Toast.LENGTH_SHORT).show();return;
        }
        else try
        {
            File filepath = Environment.getExternalStorageDirectory();
            File dir = new File(filepath.getAbsolutePath() + "/QRCode");
            dir.mkdirs();

            Toast.makeText(GeneratorActivity.this, "QR Code sudah di save", Toast.LENGTH_SHORT).show();
            Date d = new Date();
            CharSequence s  = DateFormat.format("ddMMyy kk-mm-ss", d.getTime());
            File file = new File(dir, s.toString() +".png");
            output = new FileOutputStream(file);
            bitmap.compress(Bitmap.CompressFormat.PNG, 100 , output);
            output.flush();
            output.close();
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
    }

    public void GBackColor(View v)
    {
        qrInputText = qrInput.getText().toString();
        if(qrInputText.equals(""))
        {
            Toast.makeText(getApplicationContext(), "Masukkan teks", Toast.LENGTH_SHORT).show();return;
        }
        else if(bitmap==null)
        {
            Toast.makeText(getApplicationContext(), "Tekan generate terlebih dahulu",
            Toast.LENGTH_LONG).show();return;
                                                }
            ColorPickerDialog cpd=new ColorPickerDialog(GeneratorActivity.this, listener, 0);
            cpd.show();
}

OnColorChangedListener listener=new OnColorChangedListener()
{
      public void colorChanged(int color)
      {
            // TODO Auto-generated method stub
            hexcolor = Integer.toHexString(color);
            background=color;
            qrInputText = qrInput.getText().toString();
            try
            {
                bitmap = encodeAsBitmap(qrInputText);
```

```
            }
            catch (WriterException e)
            {
                // TODO Auto-generated catch blocke.printStackTrace();
            }
            image.setImageBitmap(bitmap);
        };
};

public void GForColor(View v)
{
    qrInputText = qrInput.getText().toString();
    if(qrInputText.equals(""))
    {
        Toast.makeText(getApplicationContext(), "Masukkan teks", Toast.LENGTH_SHORT).show();return;
    }
    else if(bitmap==null)
    {
        Toast.makeText(getApplicationContext(), "Tekan generate terlebih dahulu",
        Toast.LENGTH_SHORT).show();return;
    }
    ColorPickerDialog cpd=new ColorPickerDialog(GeneratorActivity.this, listener1, 0);
    cpd.show();
}

OnColorChangedListener listener1=new OnColorChangedListener()
{
    public void colorChanged(int colorfc)
    {
        // TODO Auto-generated method stub
        hexcolorfc = Integer.toHexString(colorfc);
        foreground=colorfc;
        qrInputText = qrInput.getText().toString();

        try
        {
            bitmap = encodeAsBitmap(qrInputText);
        }
        catch (WriterException e)
        {
            // TODO Auto-generated catch blocke.printStackTrace();
        }
        image.setImageBitmap(bitmap);
    };
};

public void GForColor1(View v)
{
    qrInputText = qrInput.getText().toString();
    if(qrInputText.equals(""))
    {
        Toast.makeText(getApplicationContext(), "Masukkan teks", Toast.LENGTH_SHORT).show();return;
    }
    else if(bitmap==null)
    {
        Toast.makeText(getApplicationContext(), "Tekan generate terlebih dahulu",
        Toast.LENGTH_SHORT).show();return;
    }
    ColorPickerDialog cpd=new ColorPickerDialog(GeneratorActivity.this, listener2, 0);
    cpd.show();
}

OnColorChangedListener listener2=new OnColorChangedListener()
{
    public void colorChanged(int colorfc1)
    {
        // TODO Auto-generated method stub
```

```
                hexcolorfc1 = Integer.toHexString(colorfc1);
                foreground1=colorfc1;
                qrInputText = qrInput.getText().toString();

                try
                {
                    bitmap = encodeAsBitmap(qrInputText);
                }
                catch (WriterException e)
                {
                    // TODO Auto-generated catch blocke.printStackTrace();
                }
                image.setImageBitmap(bitmap);
                };
            };

        @Override
        public void colorChanged(int color)
        {
            // TODO Auto-generated method stub
        }
    }
```

8. RGBLuminanceSource.java

```
    package com.qrcode.ta;

    import com.google.zxing.LuminanceSource;
    import android.graphics.Bitmap;
    import android.graphics.BitmapFactory;
    import java.io.FileNotFoundException;

    public final class RGBLuminanceSource extends LuminanceSource
    {
        private final byte[] luminances;

        public RGBLuminanceSource(String path) throws FileNotFoundException
        {
            this(loadBitmap(path));
        }

        public RGBLuminanceSource(Bitmap bitmap)
        {
            super(bitmap.getWidth(), bitmap.getHeight());

            int width = bitmap.getWidth();
            int height = bitmap.getHeight();
            int[] pixels = new int[width * height];
            bitmap.getPixels(pixels, 0, width, 0, 0, width, height);
            luminances = new byte[width * height];

            for (int y = 0; y < height; y++)
            {
                int offset = y * width;
                for (int x = 0; x < width; x++)
                {
                    int pixel = pixels[offset + x];
                    int r = (pixel >> 16) & 0xff;
                    int g = (pixel >> 8) & 0xff;
                    int b = pixel & 0xff;
                    if (r == g && g == b)
                    {
                        luminances[offset + x] = (byte) r;
                    }
                    else
                    {
                        luminances[offset + x] = (byte) ((r + g + g + b) >> 2);
                    }
```

```
                }
            }
        }

        public byte[] getRow(int y, byte[] row)
        {
            if (y < 0 || y >= getHeight())
            {
                throw new IllegalArgumentException("Image di luar batas " + y);
            }

            int width = getWidth();
            if (row == null || row.length < width)
            {
                row = new byte[width];
            }

            System.arraycopy(luminances, y * width, row, 0, width);
            return row;
        }

        public byte[] getMatrix()
        {
            return luminances;
        }

        private static Bitmap loadBitmap(String path) throws FileNotFoundException
        {
            Bitmap bitmap = BitmapFactory.decodeFile(path);
            if (bitmap == null)
            {
                throw new FileNotFoundException("Image tidak bisa dibuka "+ path);
            }
        return bitmap;
        }
    }
```

9. ReaderActivity.java

```
package com.qrcode.ta;

import com.google.zxing.BinaryBitmap;
import com.google.zxing.ChecksumException;
import com.google.zxing.FormatException;
import com.google.zxing.LuminanceSource;
import com.google.zxing.MultiFormatReader;
import com.google.zxing.NotFoundException;
import com.google.zxing.Reader;
import com.google.zxing.common.HybridBinarizer;

import android.app.Activity;
import android.app.AlertDialog;
import android.content.ActivityNotFoundException;
import android.content.DialogInterface;
import android.content.Intent;
import android.database.Cursor;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.net.Uri;
import android.os.Bundle;
import android.provider.MediaStore;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;

public class ReaderActivity  extends Activity
```

```
{
    private Uri uri;
    int PICK_FROM_FILE=0,SCAN=1;
    static final String ACTION_SCAN = "com.google.zxing.client.android.SCAN";

    TextView txtresult;
    String text;
    Button btnbrowse, btnscan;
    Bitmap bitmap;
    ImageView image;

    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_read);

        txtresult =(TextView)findViewById(R.id.txtRResult);
        image = (ImageView)findViewById(R.id.imgR);
        btnbrowse = (Button)findViewById(R.id.btnRBrowse);
        btnscan = (Button)findViewById(R.id.btnRScan);

        ((Button) findViewById(R.id.btnRBrowse)).setOnClickListener(new View.OnClickListener()
        {
            @Override
            public void onClick(View v)
            {
                Intent intent = new Intent();
                intent.setType("image/*");
                intent.setAction(Intent.ACTION_GET_CONTENT);

                startActivityForResult(Intent.createChooser(intent,""), PICK_FROM_FILE);
            }
        });
    }

    public void scanQR(View v)
    {
        try
        {
            Intent intent = new Intent(ACTION_SCAN);
            intent.putExtra("SCAN_MODE", "QR_CODE_MODE");
            startActivityForResult(intent, SCAN);
        }
        catch (ActivityNotFoundException anfe)
        {
            showDialog(ReaderActivity.this, "Tidak ada barcode scanner", "Download barcode scanner?", "Yes",
            "No").show();
        }
    }

    private static AlertDialog showDialog(final Activity act, CharSequence title, CharSequence message,
    CharSequence buttonYes, CharSequence buttonNo)
    {
        AlertDialog.Builder downloadDialog = new AlertDialog.Builder(act);
        downloadDialog.setTitle(title);
        downloadDialog.setMessage(message);
        downloadDialog.setPositiveButton(buttonYes, new DialogInterface.OnClickListener()
        {
            public void onClick(DialogInterface dialogInterface, int i)
            {
                Uri uri = Uri.parse("market://search?q=pname:" + "com.google.zxing.client.android");
                Intent intent = new Intent(Intent.ACTION_VIEW, uri);
                try
                {
                    act.startActivity(intent);
                }
            catch (ActivityNotFoundException anfe)
```

```
                {
                }
            }
        });
        downloadDialog.setNegativeButton(buttonNo, new DialogInterface.OnClickListener()
        {
            public void onClick(DialogInterface dialogInterface, int i)
            {
            }
        });
    return downloadDialog.show();
    }

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data)
{
        if (resultCode != RESULT_OK) return;

        txtresult.setText("Decode Result : ");
        bitmap   = null;
        String path    = "";

        if(requestCode == SCAN)
        {
            String contents = data.getStringExtra("SCAN_RESULT");
            txtresult.setText("Decode Result : "+contents);
        }
        Else
        {
            if (requestCode == PICK_FROM_FILE)
                {
                    uri = data.getData();
                    path = getRealPathFromURI(uri);

                    if (path == null)
                    path = uri.getPath();

                    if (path != null)
                    bitmap  = BitmapFactory.decodeFile(path);
                }
            else
            {
                path    = uri.getPath();
                bitmap  = BitmapFactory.decodeFile(path);
            }
            image.setImageBitmap(bitmap);
            try
            {
                Log.i("images path", path);

                bitmap = BitmapFactory.decodeFile(path);
                image.setImageBitmap(bitmap);

                LuminanceSource source = new com.qrcode.ta.RGBLuminanceSource(bitmap);
                BinaryBitmap bitmap = new BinaryBitmap(new HybridBinarizer(source));

                Reader reader = new MultiFormatReader();
                com.google.zxing.Result result;

                result = reader.decode(bitmap);
                text = result.getText();
                txtresult.setText("Decode Result : "+text);

            }
            catch (NotFoundException e)
            {
                e.printStackTrace();
```

```
                }
                catch (ChecksumException e)
                {
                    e.printStackTrace();
                }
                catch (FormatException e)
                {
                    e.printStackTrace();
                }
            }
        }

        public String getRealPathFromURI(Uri contentUri)
        {
            String [] proj     = {MediaStore.Images.Media.DATA};
            Cursor cursor      = managedQuery( contentUri, proj, null, null,null);

            if (cursor == null) return null;

            int column_index    = cursor.getColumnIndexOrThrow(MediaStore.Images.Media.DATA);
            cursor.moveToFirst();

            return cursor.getString(column_index);
        }
    }
```

10. AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.qrcode.ta"
    android:versionCode="1"
    android:versionName="1.0"
>

<uses-sdk
    android:minSdkVersion="15"
    android:targetSdkVersion="16"
/>

<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />

<application
    android:allowBackup="true"
    android:icon="@drawable/ic_launcherr"
    android:label="@string/app_name"
    android:theme="@style/AppTheme" >

    <activity
        android:name=".MainActivity"
        android:label="@string/app_name" >

        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>

    <activity
        android:name=".ColorPickerDialog">
    </activity>

    <activity
        android:name=".RGBLuminanceSource">
    </activity>

    <activity
```

```
            android:name=".GeneratorActivity"
            android:label="@string/txtGTitle" >
        </activity>

        <activity
            android:name=".ReaderActivity"
            android:label="@string/txtRTitle" >
        </activity>

        <activity
            android:name=".SaveFileActivity">
        </activity>

    </application>
    </manifest>
```