MIND Journal | ISSN (p): 2528-0015| ISSN (e): 2528-0902 | Vol. 4 | No. 2 | Halaman 143 - 154
DOI : https://doi.org/10.26760/mindjournal Desember 2019

Respon Alphanumerik dan Numeric *QR Code*

DEWI ROSMALA , YUSUF MIFTAHUDDIN , RUSTANDI YUSUF

Program Studi Informatika, Institut Teknologi Nasional Bandung
Email : Rosmala@yahoo.com

Received 10 Oktober 2019 **|** *Revised* 30 Oktober 2019 **|** *Accepted* 24 November 2019

ABSTRAK

QR Code atau sering disebut Quick Response Code adalah barcode dua dimensi yang dapat dibaca lebih cepat dari barcode dengan kapasitas yang sama. Sudut deteksi pola yang menjamin stabil tiga bacaan pada kecepatan tinggi dan dapat memperbaiki kesalahan atau data dapat dikembalikan bahkan jika sebagian rusak atau kotor. QR Code memiliki karakter yang terbagi dalam empat karakter yaitu numerik, alphanumerik, bit, kaji. QR Code dihasilkan dari proses encoding dengan melakukan konversi atau masukan data ke dalam bentuk biner, sehingga encoding dapat diidentifikasi kebutuhan input data yang dihasilkan dalam QR Code, menentukan versi QR Code dan tingkat Error Correction. Pada penelitian ini dilakukan cara bagaimana Melakukan proses encoding alphanumerik dan numeric QR Code dengan membandingkan respon rime dari hasil generate kedua QR Code tersebut. Dengan cara melakukan pemindaian pada QR Code alphanumerik dan numerik. Dari hasil pengujian Pemindaian QR Code alphumerik dan numerik yang dilakukan didapat bahwa QR Code numerik memiliki respon lebih cepat 0.3 Detik dari QR Code alphanumerik.

***Kata kunci**: REST API, QR Code,Android*

ABSTRACT

QR Code or often called the Quick Response Code is a two-dimensional barcode that can be read faster than a barcode with the same capacity. A pattern detection angle that guarantees stable three readings at high speed and can correct errors or data can be restored even if some are damaged or dirty. QR Code has characters that are divided into four characters, namely numeric, alphanumeric, bit, review. QR Code is generated from the encoding process by converting or inputting data into binary form, so encoding can identify the input data requirements generated in the QR Code, determine the QR Code version and the level of Error Correction. In this research, how to do the alphanumeric and numeric QR Code encoding process by comparing the rime response from the results of the two QR Codes generated. By scanning the alphanumeric and numeric QR Code. From the results of the alphanetic and numerical QR Code Scan test conducted it was found that the numeric QR Code has a response faster 0.3 seconds than the alphanumeric QR Code.

Keywords: REST API, *QR Code*,Android

1. LATAR BELAKANG

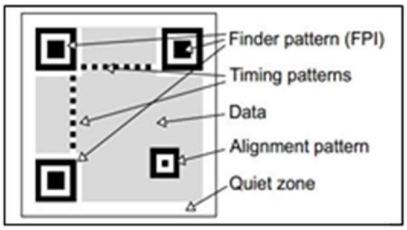
QR Code atau sering disebut *Quick Response Code* adalah barcode dengan dua dimensi. Untuk model barcode lama data yang disimpan hanya secara horizontal saja dalam penyimpanan data nama atau data yang diperkan sedangkan pada *QR Code* data yang disimpan bisa secara horizontal dan vertical dengan menampung informasi yang lebih banyak. *QR Code* mampu menampung 4.296 karakter. *QR Code* dapat dibaca oleh reader lebih cepat di bandingkan dengan barcode dalam kapasitas yang sama. QR dapat dibaca dari berbagai arah karena memiliki pola pendeteksi posisi yang terletak di 3 tempat yaitu, kiri-kanan atas dan kiri bawah *QR Code*. Maksimal 30% dari codewords yang dapat dikembalikan. Terdapat 4 versi error correction yaitu versi L, versi M, versi Q dan versi H. Setiap versi memiliki kemampuan yang berbeda untuk memperbaiki error dalam pembacaan *QR Code*.

Pada proses encoding *QR Code* dilakukan ada tahapan yang harus dilalui telebih dahulu, dikarnakan jenis dalam masukan atau data yang akan di buat menjadi *QR Code* beragam, contohnya data atau masukan tersebut dirubah dalam bentuk biner agar dapat diproses menjadi titik hitam dan putih pada *QR Code* tersebut agar memiliki informasi yang sudah di konversi, Qrcode pun harus mempunyai pola standarisasi yang dapat memudahkan dalam pembacaan atau *reading* oleh smartphone. Proses-proses secara umum yang harus dilalui dalam melakukan *generate QR Code* dari sebuah teks. Penggunaan *QR Code* menggunakan alphanumerik dan level error correction M dengan panjang bit 16 pada versi 1. Pembuatan aplikasi ini dibangun untuk melihat bagaimana perbandingan respon time dari *QR Code* alphanumerik dan numerik menggunakan proses encoding alpanumerik dan proses encoding numerik.

2. METODOLOGI PENELITIAN

Pembuatan *encoding* numerik yaitu Masukan data untuk *QR Code*, melakukan penidentifikasian *QR Code* yang dilakukan untuk mendapatkan *QR Code*, digunakan untuk menentukan sepesifikasi dalam *QR Code* dengan *version* dan *error correction* pada *QR Code*. Sistem melakukan group data masing-masing 3 digit seusai dengan jenis *QR Code*. mendapatkan nilai dengan tiga digit dan diubah dari digit tersebut menjadi bentuk biner. Jika biner sudah di dapat dan disusun sesuai dengan karakter numerik dilakukan penambahan CCL (*character count indicator*) seusai dengan *character numeric* yang dimiliki *QR Code*. Sedangkan untuk proses *encoding alphanumeric* yaitu Masukan data untuk *QR Code*, melakukan penidentifikasian *QR Code* yang dilakukan untuk mendapatkan *QR Code*, digunakan untuk menentukan sepesifikasi dalam *QR Code* dengan version dan *error correction* pada *QR Code*. Sistem melakukan konversi nilai masukan untuk QRcode, Sistem melakukan group data masing-masing 2 digit seusai dengan jenis *QR Code*. mendapatkan nilai dengan 2 digit, pada digit pertama dikalikan 45 lalu digit kedua ditambahkan pada hasil perkalian digit pertama, hasil dari penjumlahan diubah menjadi bentuk biner. Jika biner sudah di dapat dan disusun sesuai dengan karakter numeric dilakukan penambahan CCL (*character count indicator*) seusai dengan karakter numerik yang dimiliki *QR Code*.

QR Code adalah kode batang dua dimensi yang ditemukan oleh perusahaan jepang bernama denso wave pada tahun 1994. Penyimpanan data atau informasi *QR Code* secara horizontal dan vertical dengan stuktur kotak berwarna hitam dan putih. Pada kotak tersebut memiliki pola fungsi masing-masing untuk memudahkan pembaacaan data atau informasi yang tersimpan. Gambar 1 menggambarkan struktur *QR Code*.



Gambar 1. Stuktur *QR Code*

1. *Finder Pattern*

Terdiri dari 3 buah kotak yang sama terletak pada setiap pojok QRcode. Setiap pattern yang memiliki dimensi 3x3 dari warna hitam dan dikelilingi putih kemudian di kelilingi lagi hitam, pola digunakan dalam membaca atau memberikan posisi pada *QR Code*. Struktur *finder pattern* memungkinkan pendeteksian dari berbagai arah.

2. *Aligment Pattern*

Sebuah pola untuk dilakukan koreksi ditorsi dari *QR Code*. *Alignment pattern* sangat efektif untuk koreksi distorsi nonliniear. Sel hitam terisolasi iletakkan dalam aligment pattern untuk memudahkan pendeteksian koordinat sentral dari *aligment*.

3. *Timing Pattern*

Terdiri dari pola hitam dan putih dengan penyususnan bergantian dilakukan untuk identifikasi koordinat pusat dari setiap sel yang terdapat pada QRcode. *Timing pattern* dipergunakan untuk mengkoreksi koordinat pusat dari sel data saat simbol rusak atau terjadi error yang disusun baik secara vertikal maupun horizontal dan berfungsi sebagai pemisah dengan *finder paterrn*.

4. *Quite Zone*

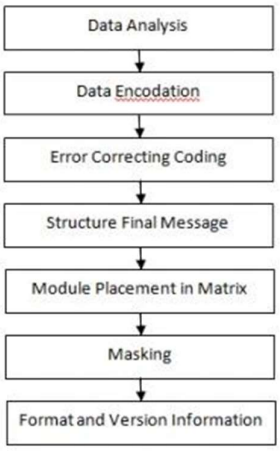
Sebuah margin yang diperlukan dalam memudahkan pembacaan QRcode.dengan zona yang terdapat disamping QRcode. Empat atau lebih dibutuhkan untuk *qiuet zone*.

5. *Data Area*

Area dalam *QR Code* yang berisi informasi yang dikodekan dalam sebuah angka biner (0 atau 1) berdasarkan aturan encoding. Angka biner dikonvesikan menjadi sel - sel hitam dan putih dan kemudian disusun sesuai aturan. Data area memilki kode fungsi error correction.

Encoding QR Code

Encoding QR Code terdiri dari beberapa proses, dilakukan berurtan seperti pada Gambar 2:



Gambar 2. Diagram Alir Encoding *QR Code*
(Sumber: Denso Wave. 2010. "*QR Code* Introduction")

Penjelasan Gambar 2 flowchat dalam membuat QRcode agar diproses menjadi titik hitam dan putih dapat dilihat proses secara umum dalam melakukan pembuatan QRcode dengan data atau informasi beragam. Dapat dilihat pada diagram alir Encoding *QR Code* yang terdapat pada Gambar 2.

a. Data Analisis

Data masukan untuk *QR Code* dianalisis, di identifikasi berbagai karakter yang berbeda. Data dikonversikan menjadi biner menurut aturan. Ada beberapa mode atau tipe data masukan yang digunakan dalam pembentukan *QR Code*, yaitu mode ECI (Extended Channel Interpretation), mode numerik, alfanumerik, byte 8-bit, dan huruf Kanji. Setiap mode encode menyimpan string yang berupa bit 1 dan 0). Setiap mode yang digunakan memiliki perbedaan metode untuk megkonversi teks menjadi bit.

b. Data *Encodation*

Sebelum proses encodation, versi dan tingkat error correction di pilih sesuai dengan jumlah karakter data masukan. Untuk menentukan versi dan tingkat error correction, dipilih versi yang menciptakan kemungkinan tali terpendek bit string untuk karakter yang digunakan dalam mode yang digunakan *QR Code*. Berikut adalah proses lengkap data encodation.

1. Masukan data untuk *QR Code*. Pda stste ini melakukan penidentifikasian *QR Code* yang dilakukan untuk mndapatkan *QR Code*.
2. Setelah langkahh ke 1 digunakan untuk menentukan sepesifikasi dalam *QR Code* dengan version dan error correction pada *QR Code*
3. Pada state ini dilakukan untuk group data masing masing 3 karakter seusai dengan version *QR Code*.
4. Setelah melakukan state ke 4 dan mendapatkan nilai dengan tiga karakter dan ubah dari karakter tersebut menjadi bentuk biner.
5. Jika biner sudah di dapat dan disusun sesuai dengan karakter numeric dilakukan penambahan CCL(*character count indicator*) seusai dengan character numeric yang dimiliki *QR Code*.

6. Pada state ini setelah melakukan CCL maka tambahkan mode indicator dengan 0001 yang dimiliki oleh karakter numeric.
7. Setelah langkah 6 pada state ini jumlahkan CCL(character count indicator),mode indicator dan data biner perbyte.
8. Menentukan jumlah bit yang digunakan dengan cara melihat versi dan error connection level dan dikurangi hasil dari langkah ke 7
9. Pada state ini melakukan penambahan terminator pada sisi kanan dan kiri jika data lebih pendek
10. Pada state ini melakukan pemisahan data biner menjadi 8 bit biner sesuai dengan LSB.
11. Jika pada langkah 10 string yang dimiliki tidak sesuai dengan hasil pada langkah 9 dilakukan state ke 11 ini dengan penambahan pad byte bit dengan jumlah 236 dan 17.
12. Pada state ini jumlah yang sudah sesuai dengan nilai yang ada pada langkah 8 maka akan terbentuk *QR Code* secara manual.

c. *Eror Correcting Coding*

Error Correction codeword memungkinkan reader *QR Code* untuk mendeteksi dan memperbaiki kesalahan dalam *QR Code*. Pada tahap ini, urutan kode bit stream yang telah dihasilkan pada tahap sebelumnya dibagi kedalam sejumlah blok sesuai dengan versinya.

d. *Structure Final Message*

Codeword data error correction telah dihasilkan. Hasil pembagian diubah menjadi 8 bit biner agar dapat dimasukan ke dalam modul-modul matriks yang terdapat pada *QR Code*. Angka bit biner dituliskan dalam satu kesatuan tanpa terputus.

e. *Module Placement In Matrix*

Data yang telah disusun ditempatkan pada sebuah matriks *QR Code*, bersama-sama dengan bagian *QR Code* lain yang sudah ada sebelumnya pada matriks QR. Pada tahap ini *QR Code* sudah hampir terbentuk seluruhnya, dan dijelaskan kapan dan bagaimana penempatan pola fungsi dan data bit.

f. *Masking*

Untuk pembacaan *QR Code* yang lebih baik, sebaiknya titik hitam dan putih disusun dalam komposisi yang baik

g. *Format dan Version Information*

Dibuat keterangan format dan versi *QR Code* yang dipakai pada *QR Code*. Format *QR Code* merupakan perpaduan dari tipe error correction dan aturan masking, sedangkan versi *QR Code* mempresentasikan ukuran data yang disimpan yang terdiri atas versi 1 hingga versi 40.

Perbandingan dalam melakukan proses generate *QR Code* manual pada alphanumerik dan numeric dengan hasil yang didapat menggunakan proses *encoding*. Pengujian dilakukan dengan melihat respon alphanumerik dan numerik Pada proses generate *QR Code* yang dilakukan pemindaian terhadap *QR Code alphanumerik* dan *numeric*. Mendapatkan hasil bahwa numeric lebih cepat dalam dilakukan pemindaian dibangdingkan dengan mengunakan numeric *QR Code*.

Proses Generate *Qr Code* Numeric

Setelah menentukan jenis data masukan yang akan dibuat menjadi QRCode, dengan contoh data 3505230721188539, maka lakukan proses data encodation, seperti pada pada Gambar 4. Dengan data masukan 3505230721188539 (Menggunakan QR versi 1 dan *Error Correction* level M)

1. Dikonversikan sesuai dengan nilai karakter numeric.
   350 523 072 118 853 9
2. Lalu dibagi kedalam grup yang masing - masing terdiri dari tiga karakter.

a. 350 c.072 e. 853
b. 534 d.118 f. 9

3. Setelah mendapatkan grup masing-masing 3 karakter di lakukan konversikan menjadi biner, dengan hasil biner yang di konversi menjadi 10 bit biner.
   350->0101011110
   534->1000010110
   072->0001001000
   118->0001110110
   853->1101010101
9 ->0000001001
4. Digabungkan data biner tersebut dalam satu kesatuan.
   0101011110 1000010110
   0001001000 0001110110
   1101010101 0000001001
5. Konversi *Character Count Indicator* kedalam biner. Jumlah karakter masukan data (3505230721188539) berjumlah 16. Dalam biner, 16 adalah 10000 dan menggunakan *QR Code* versi 1, maka dibuat menjadi 10 bit sesuai pada Tabel 6 dengan menyimpan karakter 0 di depan menjadi 0000010000.
6. *Mode Indicator* dan *Character Count Indicator* ditambahkan kedalam data biner. *Mode Indicator* untuk alfanumerik yaitu 0001 sesuai pada Tabel 1.

Tabel 1. Penambahan *Mode Indicator* dan *Character*

Mode Indicator	Character Count Indicator	Data Bit
0001	0000010000	0101011110 1000010110 0001001000 0001110110 1101010101 0000001001

7. Jumlah bit yang terdiri dari *Mode Indicator*, *Count Character Indicator*, dan data bit

dihitung seperti pada langkah 6.
0001 0000010000
0101011110 1000010110
0001001000 0001110110
1101010101 0000001001
= 74 bit

8. Mentukan jumlah bit yang diperlukan *QR Code*. Pada *QR Code* versi 1 level *error correction* memiliki 16 *codeword*, maka jumlah bit yang diperlukan adalah 16 * 8 = 128 bit.
9. Bit *string* lebih pendek dari jumlah bit yang diperlukan, tambahkan empat 0 terminator ke sisi kanan *string*. Jumlah bit yang diperlukan sebagaimana disebutkan dalam nomor 8 adalah 128 bit. Data bit *string* yang ditunjukan pada langkah 7 panjangnya adalah 74 bit. Sehingga, s*tring* yang dihasilkan masih terlalu singkat untuk mengisi kekurangan kapasitas. Kekurangan *string* tersebut adalah 104 bit – 74 bit = 30 bit. Pada Rangkaian Data Bit.

Tabel 2. Penambahan Terminator

Mode Indicator	***Character Count Indicator***	***Data Bit***	***Terminator***
0001	000001010	0101011110 1000010110 0001001000 0001110110 1101010101 0000000000	0000

10. Semua bit *string* di tuliskan dalam satu kesatuan dan kelompokan menjadi 8 bit biner. Berikut adalah penghitungan dari Mode Indicator, Character Count Indicator dan Data Bit. (Susunan string seperti langkah 7)
0001 0000010000
0101011110 1000010110
0001001000 0001110110
1101010101 0000000000
Hasilnya adalah
_____00 01000001 00000101 01111010 00010110 00010010 00000111 01101101 01010100 00000000
Dikarenakan terdapat string bit yang bukan kelipatan 8 bit, sehingga tambahkan string 0 menjadikannya sebuah bentuk 8-bit biner sesuai aturan LSB (Least Significant Bit).
00000000 01000001 00000101 01111010 00010110 00010010 00000111 01101101 01010100 00000000
Lalu susun kembali Mode Indicator, Character Count Indicator dan Data Bit dan Terminator, penambahan empat terminator 0 disisi kanan string bit. (sesuai pada langkah 9)
00000000 01000001 00000101 01111010 00010110 00010010 00000111 01101101 01010100 00000000 0000¬¬¬____
Dikarenakan terdapat string bit yang bukan kelipatan 8 bit, sehingga tambahkan string 0 menjadikannya sebuah bentuk 8-bit biner sesuai aturan LSB (Least Significant Bit).
00000000 01000001 00000101 01111010 00010110 00010010 00000111 01101101 01010100 00001001 00000000

11. String tidak sesuai dengn kapasitas maksimum maka dilakukan penambahan pad bit byte dengan nilai 236 and 17 yang di ubah ke biner dengan menginyinya dibagian akhir string secara mengulang. Pada langkah 9 seharusnya bit max 128 sedangkang kekosongan yang telah dilakukan pada langkah 10 menjadi 128-88=40 bit maka 40/8=5, oleh karena itu ditambah kan pad bit byte sebanyak 5 kali
    00000000 10000001 11001100 00011100 00000000 00001110 00110001 11000010 01100101 10000111 00001000 00000101 00000000 11101100 00010001 11101100 00010001 11101100
12. *String* sudah sesuai mengisi kapasitas maksimum yaitu 128 bit.
    00000000 10000001 11001100 00011100 00000000 00001110 00110001 11000010 01100101 10000111 00001000 00000101 00000000 11101100 00010001 11101100 00010001 11101100
    Hasil dari langkah 12 adalah data bit yang akan dibuat menjadi sebuah QRCode.

Proses Generate *Qr Code* Alphanumerik

Setelah menentukan jenis data masukan yang akan dibuat menjadi QRCode, dengan contoh data 3505230721188539, maka lakukan proses data encodation, seperti pada pada Gambar 4. Dengan data masukan 35052A0721188539 (Menggunakan QR versi 1 dan *Error Correction* level M)

1. Dikonversikan sesuai dengan nilai karakter alfanumerik.
    3=3 5=5 0=0 5=5
    2=2 A=10 0=0 5=5
    2=1 1=1 1=1 8=8
    8=8 5=5 3=3 9=9
2. Lalu dibagi kedalam grup yang masing - masing terdiri dari dua nilai.
    (3,5) , (0,5) , (2, 10) , (0,7) , (2,1) , (1,8) , (8,5) , (3,9)
3. Dikalikan karakter pertama dengan 45, hasilnya Tambahkan dengan karakter kedua, hasil penjumlahan di konversikan menjadi 11 bit biner.
    (3, 5) = (3*45) + 5 = 140 → 00010001100
    (0, 5) = (0*45) + 5 = 5 → 00000000101
    (2, A) = (2*45) + 10 = 100 → 00001011100
    (0, 7) = (0*45) + 7 = 7 → 00000000111
    (2, 1) = (2*45) + 1 = 91 → 00001011011
    (1, 8) = (1*45) + 8 = 53 → 00000110101
    (8, 5) = (8*45) + 5 = 365 → 00101101101
    (3, 9) = (3*45) + 9 = 144 → 00010010000
4. Digabungkan data biner tersebut dalam satu kesatuan.
    00010001100 00000000101 00001011100 00000000111 00001011011 00000110101 00101101101 00010010000
5. Konversi *Character Count Indicator* kedalam biner. Jumlah karakter masukan data (3505230721188539) berjumlah 16. Dalam biner, 16 adalah 10000 dan menggunakan *QR Code* versi 1, maka dibuat menjadi 9 bit sesuai pada Tabel 6 dengan menyimpan karakter 0 di depan menjadi 000010000.
6. *Mode Indicator* dan *Character Count Indicator* ditambahkan kedalam data biner. *Mode Indicator* untuk alfanumerik yaitu 0001 sesuai pada Tabel 1.

Tabel 3. Penambahan *Mode Indicator* dan *Character*

Mode Indicator	Character Count Indicator	Data Bit
0010	000010000	00010001100 00000000101 00001011100 00000000111 00001011011 00000110101 00101101101 00010010000

7. Jumlah bit yang terdiri dari *Mode Indicator*, *Count Character Indicator*, dan data bit dihitung seperti pada langkah 6.
   0010 000010000 00010001100
   00000000101 00001011101
   00000000111 00001011011
   00000110101 00101101101
   00010010000 = 101 bit
8. Mentukan jumlah bit yang diperlukan *QR Code*. Pada *QR Code* versi 1 level *error correction* M memiliki 16 *codeword*, maka jumlah bit yang diperlukan adalah 16 * 8 = 128 bit.
9. Bit *string* lebih pendek dari jumlah bit yang diperlukan, tambahkan empat 0 terminator ke sisi kanan *string*. Jumlah bit yang diperlukan sebagaimana disebutkan dalam nomor 8 adalah 128 bit. Data bit *string* yang ditunjukan pada langkah 7 panjangnya adalah 101 bit. Sehingga, s*tring* yang dihasilkan masih terlalu singkat untuk mengisi kekurangan kapasitas. Kekurangan *string* tersebut adalah 128 bit – 101 bit = 27 bit.

Tabel 4. Penambahan Terminator Pada Rangkaian Data Bit

Metode indicator	***Character Count Indicator***	***Data Bit***	***Terminator***
0010	000010000	00010001100 00000000101 00001011100 00000000111 00001011011 00000110101 00101101101 00010010000	0000

10. Semua bit *string* di tuliskan dalam satu kesatuan dan kelompokan menjadi 8 bit biner. Berikut adalah penghitungan dari *Mode Indicator, Character Count Indicator* dan Data Bit. (Susunan string seperti langkah 7)
    0010 000010000
    00010001100 00000000101 00001011100 00000000111 00001011011 00000110101 00101101101 00010010000
    Hasilnya adalah
    _____00 10000010 00000010 00110000 00000010 10000101 11010000 00001110 00010110 11000001 10101001 01101101 00010010 00000000
    Dikarenakan terdapat string bit yang bukan kelipatan 8 bit, sehingga tambahkan

string 0 menjadikannya sebuah bentuk 8-bit biner sesuai aturan LSB (Least Significant Bit).
00000000 10000010 00000010
00110000 00000010 10000101
11010000 00001110 00010110
11000001 10101001 01101101
00010010 00000000
Lalu susun kembali Mode Indicator, Character Count Indicator dan Data Bit dan Terminator, penambahan empat terminator 0 disisi kanan string bit. (sesuai pada langkah 9)
00000000 10000001 11001100
00011100 00000000 00001110
00110001 11000010 01100101
10000111 00001000 00000101
0000¬¬¬____
Dikarenakan terdapat string bit yang bukan kelipatan 8 bit, sehingga tambahkan string 0 menjadikannya sebuah bentuk 8-bit biner sesuai aturan LSB (Least Significant Bit).
00000000 10000010 00000010
00110000 00000010 10000101
11010000 00001110 00010110
11000001 10101001 01101101
00010010 00000000 00000000

11. String tidak sesuai dengn kapasitas maksimum maka dilakukan penambahan pad bit byte dengan nilai 236 and 17 yang di ubah ke biner dengan menginyinya dibagian akhir string secara mengulang. Pada langkah 9 seharusnya bit max 128 sedangkang kekosongan yang telah dilakukan pada langkah 10 menjadi 128-120=8 bit maka 8/8=1, oleh karena itu ditambah kan pad bit byte sebanyak 1 kali.
00000000 10000010 00000010 00110000 00000010 10000101 11010000 00001110 00010110 11000001 10101001 01101101 00010010 00000000 00000000 11101100
12. *String* sudah sesuai mengisi kapasitas maksimum yaitu 128 bit.
00000000 10000010 00000010 00110000 00000010 10000101 11010000 00001110 00010110 11000001 10101001 01101101 00010010 00000000 00000000 11101100

3. HASIL DAN PEMBAHASAN

Pengujian dilakukan perbandingan respon *QR Code* alphanumerik dan numerik Pengujian dilakukan dengan menlakukan pemindaian sebanyak 30 kali dengan jarak 25 cm dan pixel camera 8MP numerik dan alphanumerik, hasil rata-rata Dari semua pengujian didapat rata-rata respon didapat dalam 30 kali pemindaiain untuk setiap *QR Code* yang di generate. Berikut hasil dari pengujian Respon *QR Code* pada tabel 5 hasil pengujian respon *QR Code*.

Tabel 5. Hasil Pengujian Respon *QR Code*

No	Nama pengujian	Rata-rata respon
1	Pengujian respon *QR Code* numerik mahasiswa	6.8 detik
2	Pengujian respon *QR Code* alphanumerik *Development*	5.5 detik
3	Pengujian respon *QR Code* numerik *Development*	5.2 detik

Berdasarkan hasil kesimpulan *QR Code* numerik lebih cepat dibandingkan dengan alphanumerik yang dilakukan oleh *development* dengan waktu rata-rata 5.2 detik untuk numerik sedangkan untuk alpanumerik 5.5 detik. Sedangkan pengujian respon *QR Code* numerik oleh mahasiswa berhasil memindai dengan waktu rata-rata 6.8 detik.

5. KESIMPULAN

Berdasarkan penelitian yang telah dilakukan menggunakan aplikasi presensi dengan menguji 30 sampel data mahasiswa yang melakukan pada fungsi aplikasi seperti fungsi sign in, pemindaian *QR Code* pada waktu matakuliah, Pemindaain Ketika image QRcode ditutup. Dalam melakukan pengujian fungsi aplikasi pemindaian QRcode pada waktu perkuliahan saat mahasiswa yang hadir, yang mampu melakukan presensi 26 menggunakan aplikasi presensi dari 30 mahasiswa. Pengujian sistem Maka disimpulkan rata-rata keberhasilan dalam fungsi pemindaian *QR Code* 86%. Sedangkan pengujian yang dilakukan oleh development dengan menguji respon dari *QR Code* alphanumerik dan nemetik *QR Code* dengan menggunakan jarak 25 cm dalam melakukan pemindaian dan menggunakan camera 8MP serta menguji respon pemindaian *QR Code* numerik pada mahasiswa dengan jarak dan camera yang bermacam-macam. Dari pengujian respon yang dilakukan oleh mahasiswa Hasil dari pengujian yang dapat dilihat pada Tabel 5 mendapatkan nilai rata-rata pada respon aplikasi presensi dengan image *QR Code* 6.8 detik. Sedangkan untuk pengujian respon yang dilakukan development dengan jarak yang ditentukan dan pixel camera dengan dua jenis *QR Code* alphanumerik dan numerik dapara dilihat pada Tabel 5 mendapatkan nilai rata-rata pada respon aplikasi presensi dengan image *QR Code* alphanumerik 5.5 detik dan image *QR Code* numerik 5.2 detik. Berdasarkan hasil penelitian serta pengujian sistem seperti yang ditunjukan pada Tabel yang telah dilakukan, maka dapat disimpulkan bahwa rata-rata tingkat keberhasilan dalam pemindaian *QR Code* alphanumeric dan numeric yang telah dilakukan oleh development pemindaian QRcode numeric

DAFTAR RUJUKAN

Eby, C. (2016, desember 12). *proses encoding numeric*. Diambil kembali dari Thonky.com's *QR Code* Tutorial : Thonky.com online: http://www.thonky.com/qr-code-tutorial/numeric-mode-encoding

Farizi, M. H. (2015). Design of mobile-based applications via *QR Code*.

Mu zhang, Q. Z. (2012). The Application and Design of *QR Code* in Scenic Spot's eTicketing System -A Case Study of Shenzhen Happy Valley.

Narayanan, S. (2012). *QR Code* and security solution. *Department of Information Technology, Salalah College of Technology, 7 july 2012: Sultanate of Oman* , vol 3.

sense, D. w. (2016, desember 10). *QR Code*. Diambil kembali dari Japan corporation: http://www.*QR Code*.com

Sharma, P. (2013). Evolution of Mobile Wireless Communication Networks-1G to 5G as well as Future Prospective of Next Generation Communication Network. *International Journal Of Computer Science and Mobile Computing*, 47-53.

Tresnani, L. d. (2012). Implementation Of Employee Attendance System Using A QR cod on Android-based Smartphone. *ITB*.