ANALISA DAN OPTIMALISASI RANCANGAN BASIS DATA APLIKASI PENCATAT KEUANGAN DIGITAL

M. Imam Budi Laksamana[1], Anisya Nursyah Gusman[2], Muhammad Lathifuddin Arif[3], Muhammad Fadli[4], Muhammad Syaiful Anam[5], Ema Utami[6]
[1,2,3,4,5]Magister Teknik Informatika, Universitas Amikom Yogyakarta
e-mail: [1]imambudi.1284@students.amikom.ac.id, [2]anisya.1282@students.amikom.ac.id,
[3]muhammad.1287@students.amikom.ac.id,[4]muhammad.1289@students.amikom.ac.id,
[5]anam@amikom.ac.id,[5]ema.u@amikom.ac.id

ABSTRAK

Kemajuan teknologi informasi diharapkan dapat menjadi alat yang dapat membantu proses pengelolaan data. Pengelolaan dan penggunaan data yang tersistematis dapat menghasilkan informasi yang rinci. Pengelolaan data yang baik dapat meningkatkan kualitas informasi yang diolah. Pada perancangan database, terdapat hal-hal penting yang harus diperhatikan, diantaranya adalah constraint, type data dan relasi yang dibutuhkan untuk masing-masing table yang ada pada database. Constraint, type data, dan relasi yang tepat dapat meningkatkan performasi dari database yang dimiliki. Fokus pembahasan pada paper kali ini adalah optimisasi table dari database yang digunakan pada aplikasi pencatatan keuangan digital. Optimasi yang akan dilakukan pada penelitian ini mencangkup optimasi penggunaan constraint yang tepat untuk setiap kolom untuk masing-masing table yang ada pada sistem pencatatan keuangan digital yang ada, selain constraint, hal yang dilakukan adalah optimasi untuk type data yang sesuai untuk masing-masing kolom yang ada pada setiap tabel database. Pada tahap percobaan nantinya database yang ada akan di ekspor ke dalam database MySQL menggunakan aplikasi XAMPP. Data yang akan digunakan untuk pengujian sebesar 1.585 data catatan keuangan yang dilakukan user melalui aplikasi.

Kata Kunci: database; optimisasi; catatan digital; MySQL

1. PENDAHULUAN

Database merupakan salah satu aspek yang sangat penting dalam sistem informasi dikarena fungsi dari database adalah sebagai gudang penyimpanan data yang nantinya akan diolah lebih lanjut. Database memiliki peran penting karena dengan adanya database, maka dapat menghindari duplikasi data, menghindari hubungan antar data yang tidak jelas dan terdapatnya update yang rumit. Dalam membuat rancangan pada database, terdapat beberapa hal penting yang harus diperhatikan. Diantaranya adalah constraint, type data dan relasi antar tabel. Dengan memiliki constraint, type data, dan relasi yang tepat, maka dapat meningkatkan performasi dari database yang dimiliki. Optimalisasi perlu dilakukan dalam merancang database agar meningkatkan efisiensi data.

Optimalisasi merupakan suatu langkah dalam mengoptimalkan waktu agar menjadi lebih efisien. Jika sebuah query diberikan pada sebuah basis data, maka optimasi sangat penting dilakukan untuk memilih strategi yang efisien dalam mengevaluasi relasi yang ditentukan[1]. Kasku adalah pencatatan uang digital dimana aplikasi ini mencatat pemasukkan dan pengeluaran dari user yang menggunakannya. Selain itu, aplikasi ini dapat melihat pencatatan keuangan berdasarkan hari, minggu bahkan bulan. Dalam aplikasi pencatatan uang digital ini, sering mengalami keterlambatan request data dan tidak terdapatkan keterbatasan input data sehingga data yang dimasukkan oleh satu user bisa lebih banyak.

Sehingga dalam penelitian ini perlu dilakukannya optimalisasi agar data yang ditampung pada database tidak terlalu berat. Maka dari itu, optimasi yang akan dilakukan mencangkup optimasi penggunaan constraint yang tepat untuk setiap kolom pada masing-

masing table yang ada dalam sistem pencatatan keuangan digital, kemudian optimasi yang dilakukan pada type data yang sesuai untuk masing-masing kolom yang ada pada setiap tabel database.

2. TINJAUAN PUSTAKA

Ada beberapa definisi tentang optimalisasi dari beberapa referensi terdahulu, diantaranya yaitu Sucipto dkk (2017) dalam penelitiannya menyatakan bahwa ada beberapa cara untuk melakukan Optimalisasi database diantaranya melalui DDL dan DML. DDL (Data Definition Language) merupakan perintah SQL yang digunakan untuk mendefinisikan atau mendeklarasikan objek database, menciptakan objek database atau bahkan menghapus objek database. Objek database dapat berupa tabel atau database itu sendiri. DDL yang umum digunakan adalah CREATE,DROP, ALTER. DDL juga dapat digunakan untuk membuat relasi antar tabel database beserta batasannya dengan menentukan indeks sebagai kuncinya. DML (Data Manipulation Language) merupakan query yang digunakan untuk memanipulasi data, seperti untuk menampilkan data, mengubah data, atau mengisi data[1]

Dalam menganalisis Optimalisasi query, salah satu teknik yang dapat digunakan yaitu teknik heuristic optimization. Teknik ini digunakan untuk mengurangi kompleksitas optimisasi, memecahkan masalah dengan sedikit mengabaikan apakah solusinya dapat dibuktikan dengan benar, namun biasanya menghasilkan solusi yang mendekati optimal. Tujuan dari optimasi adalah untuk mengurangi sebanyak mungkin baris data yang tidak dibutuhkan dalam prosesnya. Setelah menganalisis optimasi dari query yang digunakan diharapkan dalam proses mengolah data rekomendasi produk di dalam database perusahaan dapat lebih cepat dilakukan. Data yang akan digunakan dalam penelitian ini menggunakan data simulasi yang mendekati dengan data sebenarnya, karena data yang sesungguhnya bersifat privasi[2]

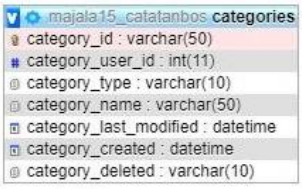
3. METODE PENELITIAN

Metodologi yang akan digunakan dalam penelitian ini adalah dengan melakukan tinjauan teori query pemrograman basis data dengan membandingkan kinerja basis data pada aplikasi yang sudah ada dengan kinerja basis data hasil normalisasi yang ada. Pada penelitian ini dilakukan testing dengan query yang dijalankan menggunakan database catatan kas yang sudah ada pada database yang ada, adapun penambahan kolom dan penambahan tabel pada database yang sudah ada bertujuan untuk meningkatkan performa dan ketepatan data yang tersimpan pada database. berikut adalah bentuk beberapa query yang digunakan pada penelitian yang ada:

a. Penggunaan Aritmatika Database
b. Penggunaan Group
c. Penggunaan Sorting
d. Penggunaan Select *

4. HASIL DAN PEMBAHASAN

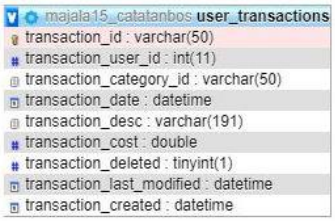
Penelitian yang dilakukan adalah mengoptimalisasikan perancangan database pada objek penelitian. pada penelitian ini terdapat beberapa tabel yang diubah, perubahan tersebut mencakup perubahan type data dan constraint data, berikut adalah table-table yang digunakan pada aplikasi KASKU sebelum dilakukan Normalisasi.



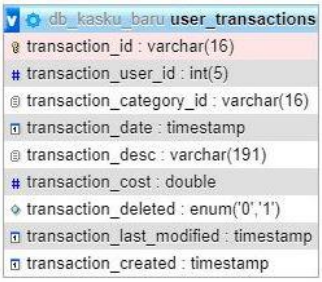
Gambar 1. table categories sebelum normalisasi

Gambar 2. table categories setelah normalisasi

Gambar 1. merupakan gambar table categories yang terdapat pada aplikasi KASKU sebelum dilakukan normalisasi, table categories digunakan untuk menyimpan data kategori dari catatan transaksi yang dilakukan user, pada table ini terdapat beberapa kolom dengan type data dan constraint yang dapat kita normalisasi untuk meminimalisir kesalahan data dan menambah kecepatan query pada database. kolom yang dapat kita rumah adalah panjang karater data pada kolom category_id yang awalnya varchar(50) menjadi varchar(16), kolom category_user_id yang awalnya int(11) menjadi int(5), dan kolom yang dapat kita rubah type datanya adalah kolom category_type yang awalnya varchar(10) menjadi enum('IN','OUT') dengan tambahan constraint check yang hanya memperbolehkan nilai 'IN' dan 'OUT', kolom category_last_modified yang awalnya datetime menjadi timestamp, kolom category_created yang awalnya datetime menjadi timestap, kolom category_deleted yang awalnya varchar(10) menjadi enum('0','1') dengan constraint check yang hanya memperbolehkan nilai '0' dan '1', untuk hasil normalisasi table categories dapat dilihat pada Gambar 2.

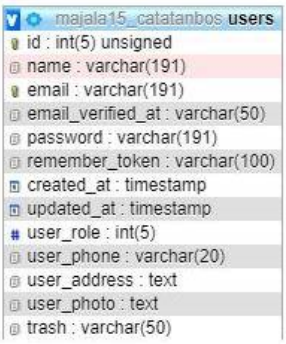


Gambar 3 table user_transactions sebelum normalisasi.

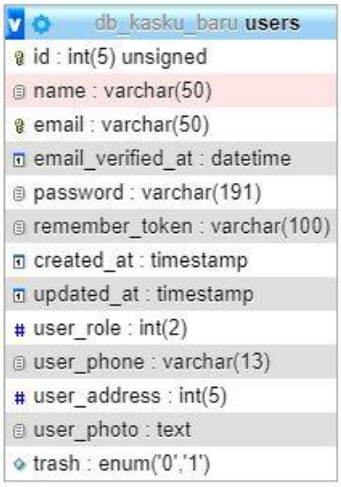


Gambar 4 table user_transactions setelah normalisasi.

Gambar 3 merupakan table yang digunakan pada aplikasi kasku, pada table ini tersimpan data transaksi yang dilakukan oleh user melalui aplikasi. pada table ini terdapat beberapa type data, panjang karakter data dan constrain yang dapat kita tambahkan untuk menambah kinerja database, untuk perubahan type data dan panjang karakter kita dapat merubah kolom transaction_id yang awalnya varchar(50) menjadi varchar(16), kolom transaction_user_id yang awalnya int(11) menjadi int(5), kolom transaction_category_id yang awalnya varchar(50) menjadi varchar(16), kolom transaction_last_modified, transaction_created yang awalnya datetime menjadi timestamp, kolom transaction_deleted yang awalnya tinyint(1) menjadi enum('0','1') dengan constraint check yang hanya memperbolehkan nilai '0' dan '1', untuk hasil normalisasi table user_transactions dapat dilihat pada Gambar 4.

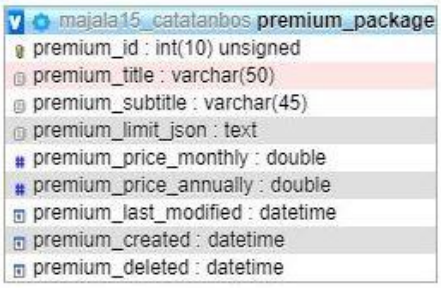


Gambar 5. Table users sebelum normalisasi.

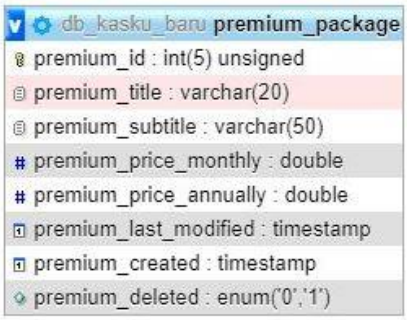


Gambar 6. Table user setelah normalisasi.

Gambar 5 adalah table users yang digunakan pada aplikasi KASKU table ini digunakan untuk menampung data user yang melakukan registrasi pada aplikasi, pada table ini terdapat beberapa point yang dapat kita tambahkan untuk meningkatkan performa table, kita dapat merubah type data dan panjang karakter pada kolom nama dan email yang awalnya varchar(191) menjadi varchar(50), kolom email verified yang awalnya varchar(50) menjadi datetime, kolom user_role yang awalnya int(5) menjadi int(2), kolom user_phone yang awalnya varchar(20) menjadi varchar(13), kolom user_address yang awalnya text menjadi int(5), kolom trash yang awalnya varchar(50) menjadi enum(‘0’,’1’) dengan constrain check yang hanya memperbolehkan nilai ‘1’ dan ‘0’. untuk hasil normalisasi table user dapat dilihat pada Gambar 6.

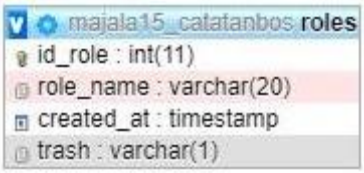


Gambar 7. table premium_package sebelum normalisasi.

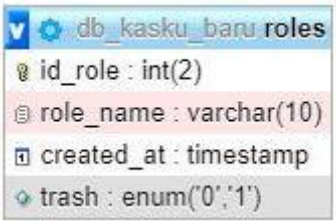


Gambar 8. table premium_package sesudah normalisasi

Gambar 7 adalah gambar table yang digunakan pada aplikasi KASKU sebelum dilakukannya normalisasi, table premium_package adalah table yang digunakan untuk menyimpan data pelanggan yang berlangganan paket premium aplikasi KASKU. pada table ini terdapat beberapa kolom yang dapat kita normalisasikan, diantaranya kita dapat merubah type data dan panjang karakter pada kolom premium_id yang awalnya int(10) menajdi int(5),kolom premium_title yang awalnya varchar(50) menjadi varchar(20), kolom premium_last_modified, premium created yang awalnya datetime menjadi timestamp, kolom premium_deleted yang awalnya date time menajadi enum(‘0,’1’) dengan tambahan constrain check yang hanya memperbolehkan nilai ‘0’ dan ‘1’. untuk hasil normalisasi table dapat dilihat pada Gambar 7.

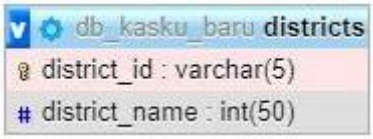


Gambar 9 table role sebelum normalisasi



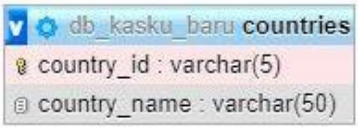
Gambar 10 table role setelah normalisasi

Gambar 9 adalah table role sebelum normaslisasi table ini digunakan untuk menyimpan data peran user yang terdapat pada aplikasi, pada table role terdat beberapa kolom yang dapat kita normalisasikan untuk meningkatkan performa database diantaranya, kita dapat merubah type data, panjang kolom dan penambahan constraint pada kolom id_role yang awalnya int(11) menjadi int(2), kolom role_name yang awalnya varchar(20) menjadi varchar(10), kolom trash yang awalnya varchar(1) menjadi enum(‘0’,’1’) dengan tambahan constrain check yang hanya memperbolehkan nilai ‘0’ dan ‘1’. untuk hasil normalisasi dapat dilihat pada Gambar 10.
setelah melakukan beberapa perubahan pada kolom pada penelitian ini juga menambahkan beberapa table tambahan yang digunakan untuk menampung data alamat agar tidak terjadi kesalahan data dan konsistensi format alamat pada kolom alamat yang terdapat pada table users, berikit adalah beberapa table yang ditambahkan.



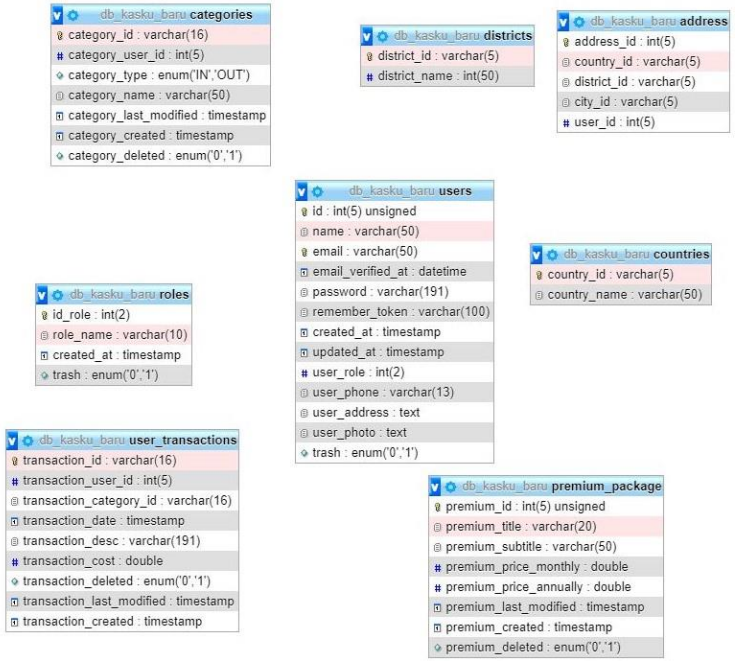
Gambar 11. table districs

Gambar 12. table address



Gambar 13. table countries

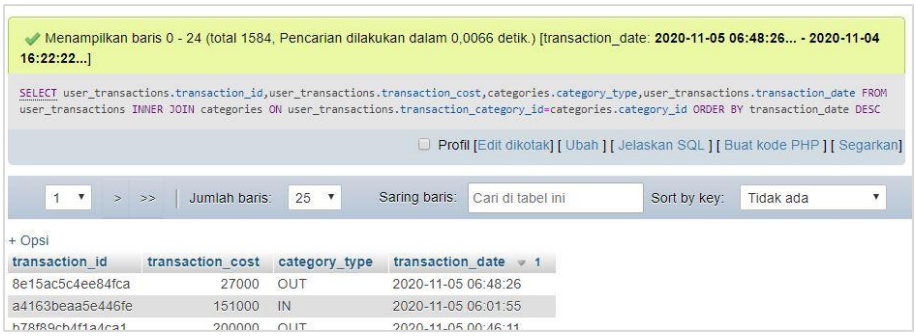
Gambar 11 adalah table districs yang digunakan untuk menampung data kabupatan, Gambar 12 adalah table yang digunakan untuk menampung data country_id, distric_id,city_id dan user_id table ini nantinya akan sangat berguna untuk menampung data alamat dengan format yang sama dan Gambar 13 adalah table countries yang digunakan untuk menampung nama kota pada aplikasi.



Gambar 14. database aplikasi pencatatan kas digital yang sudah di normalisasikan.

Selanjutnya adalah melakukan testing menggunakan query. pada bagian testing penelitian ini menggunakan data transaksi user dengan 1.585 record, untuk pengetesan pertama digunakan query select sebagai berikut

SELECT
user_transactions.transaction_id,user_transactions.transaction_cost,categories.category_type, user_transactions.transaction_date FROM user_transactions
INNER JOIN categories ON
user_transactions.transaction_category_id=categories.category_id ORDER BY transaction_date DESC
hasil testing pertama

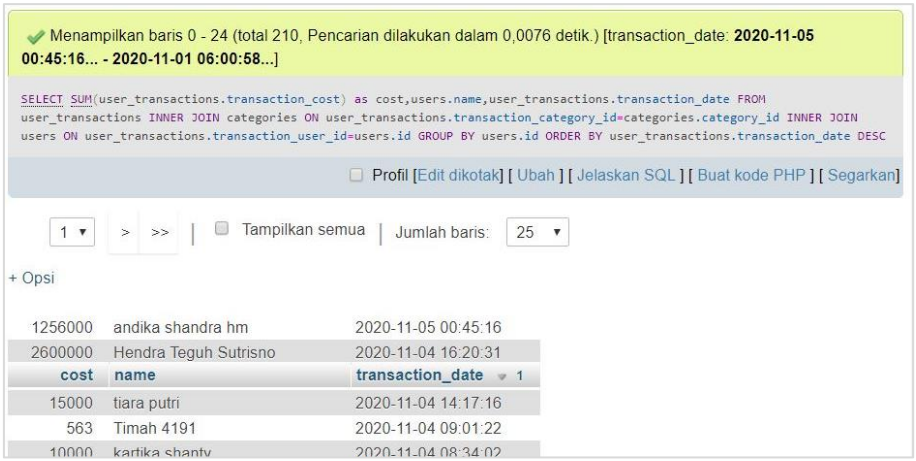


Gambar 15. hasil testing database sebelum normalisasi

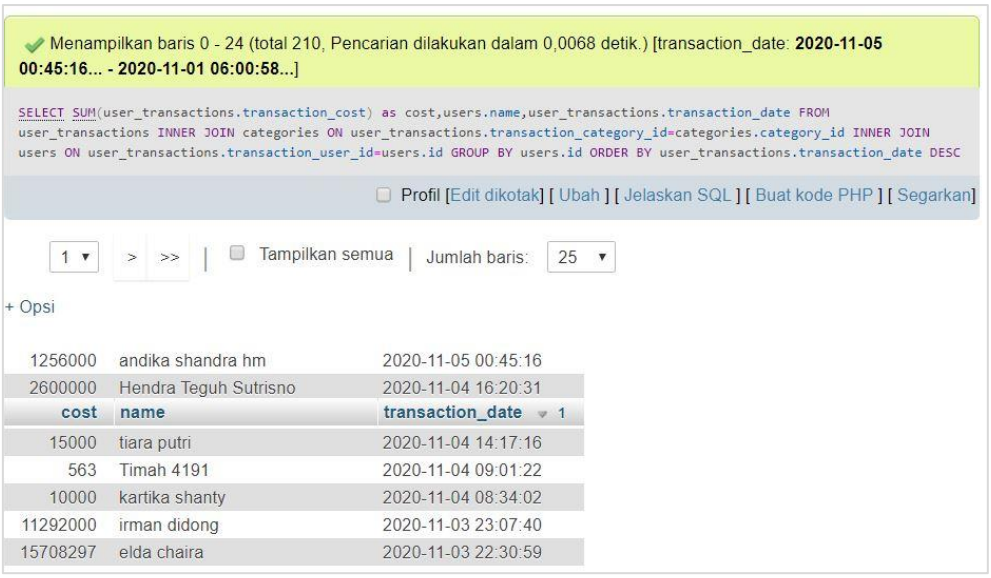
Gambar 16. hasil testing database setelah melakukan normalisasi database.

Selanjutnya mari kita lanjutkan pengetesan kedua dengan menerapkan operasi aritmatika pada query, berikut adalah query yang digunakan.

```
SELECT SUM(user_transactions.transaction_cost) as
cost,users.name,user_transactions.transaction_date
FROM user_transactions
INNER JOIN categories ON
user_transactions.transaction_category_id=categories.category_id
INNER JOIN users ON user_transactions.transaction_user_id=users.id
GROUP BY users.id ORDER BY user_transactions.transaction_date DESC
```



Gambar 17. hasil testing query pada table sebelum normalisasi



Gambar 18. hasil testing query pada table setelah normalisasi

Dapat dismpulkan waktu eksekusi query pada table setelah normalisasi membutuhkan waktu yang lebih sedikit dibandingkan waktu eksekusi database sebelum normalisasi database.

5. KESIMPULAN

Berdasarkan hasil uji coba dengan menggunakan 1.585 record data transaksi dengan menjalan query pada database sebelum dilakukannya normalisasi dan database setelah dilakukannya query dapat disimpulkan, normalisasi database sangatlah penting dalam proses perancangan database, normalisasi database dapat menambah performa dari database yang dimiliki, sebagai contoh setelah melakukan percobaan eksekusi query pada database sebelum dan sesudah normalisasi, waktu eksekusi query yang dibutuhkan oleh database yang sudah ternormalisai lebih sedikit dibandingkan waktu eksekusi query sebelum normalisasi database.

DAFTAR PUSTAKA

[1] Sucipto, R. Indriati, and F. B. Hariawaan, "Desain Database Untuk Optimalisasi Sistem Prediksi Transaksi Penjualan," vol. 02, no. 02, pp. 88–93, 2017.

[2] G. T. R. I. Mardiani and H. Irmayanti, "Analisis Optimasi Query SQL Menggunakan Teknik Heuristic Pada Kasus Data Transaksi Pelanggan Yang Layak Mendapatkan Rekomendasi Produj," vol. 16, no. 2, pp. 133–144, 2018.

[3] Warman, I., & Ramdaniansyah, R. (2018). Analisis Perbandingan Kinerja Query Database Management System (Dbms) Antara Mysql 5.7.16 Dan Mariadb 10.1. *Jurnal Teknoif*, *6*(1), 32–41.

[4] Maanari, J. I., Sengkey, R., Wowor, I. H. F., Kom, M., & Rindengan, Y. D. Y. (2013). Perancangan Basis Data Perusahaan Distribusi Dengan Menggunakan Oracle. *Jurnal Teknik Elektro Dan Komputer*, *2*(2). https://doi.org/10.35793/jtek.2.2.2013.1719