CLASS LIBRARY UNTUK PEMBUATAN APLIKASI *CRUD*

Wiwit Agus Triyanto
Program Studi Sistem Informasi, Fakultas Teknik, Universitas Muria Kudus
Gondangmanis, PO Box 53, Bae, Kudus 59352
Email: at.wiwit@yahoo.co.id

Abstrak

Java adalah bahasa pemrograman yang berorientasi objek (OOP) dan dapat dijalankan pada berbagai platform sistem operasi. Dalam bahasa pemrograman yang berorientasi objek (OOP) terdapat istilah Class Library. Class library merupakan suatu fungsi yang dibuat untuk mempermudah programmer dalam membuat sebuah aplikasi. Suatu aplikasi dapat dikatakan baik, jika didalamnya minimal terdapat proses CRUD (Create Read Update Delete). Dengan menggunakan konsep Object Oriented Programming (OOP) yang dibungkus dalam suatu Class Library, syntax CRUD yang kompleks dapat dijadikan menjadi lebih sederhana, sehingga pembuatan aplikasi CRUD pada project Java Swing menjadi lebih cepat dan mudah. Dalam eksperimen, Class Library CRUD dibagi menjadi 4 bagian, yaitu: Class Simpan, Class Tampil, Class Ubah, Class Hapus. Hasil dari penelitian ini, Class Library dapat mempermudah pembuatan aplikasi CRUD, tetapi masih memiliki kekurangan yaitu jumlah field yang dapat digunakan terbatas 10 field.

***Kata kunci**: Java, library, CRUD, Object Oriented Programming*

1. PENDAHULUAN

Pemrograman komputer semakin menjadi kebutuhan bagi masyarakat dalam semua bidang kehidupan. Sejak perkembangan IT yang semakin pesat, hampir semuanya telah didigitalkan dan penggunaan perangkat lunak / aplikasi merupakan aktivitas yang umum untuk semua kalangan, hal ini dikarenakan banyak terdapat perangkat lunak / aplikasi yang tersedia untuk sebuah toko kecil yang sederhana maupun untuk rumah sakit berteknologi tinggi [1].

Untuk membuat sebuah aplikasi, programmer dapat menggunakan berbagai macam bahasa pemrograman, seperti Java, PHP, C, Visual Basic, Delphi, dll. Perkembangan penggunaan bahasa Java saat ini cukup pesat, hal ini dibuktikan dengan ranking TIOBE pada bulan Juli 2014 yang menyebutkan bahwa bahasa Java sebagai bahasa pemrograman paling populer nomor dua di dunia [3].

Java adalah bahasa pemrograman yang berorientasi objek (*OOP*) dan dapat dijalankan pada berbagai platform sistem operasi. Perkembangan Java tidak hanya terfokus pada satu sistem operasi, tetapi dikembangkan untuk berbagai sistem operasi dan bersifat *open source* [4]. Berorientasi objek adalah sebuah teknik yang memusatkan desain pada *object* dan *class* berdasarkan pada skenario dunia nyata. Hal ini menegaskan keadaan (*state*), *behaviour* dan interaksi dari *object*. Selain itu juga menyediakan manfaat akan kebebasan pengembangan, meningkatkan kualitas, mempermudah pemeliharaan, mempertinggi kemampuan dalam modifikasi dan meningkatkan penggunaan kembali *software* [5].

Dalam bahasa pemrograman yang berorientasi objek (*OOP*), terdapat istilah *class library* dan *framework*. *Class library* merupakan suatu fungsi yang dibuat untuk mempermudah programmer dalam membuat sebuah aplikasi, sedangkan *Framework* adalah kumpulan dari fungsi atau *library* [2].

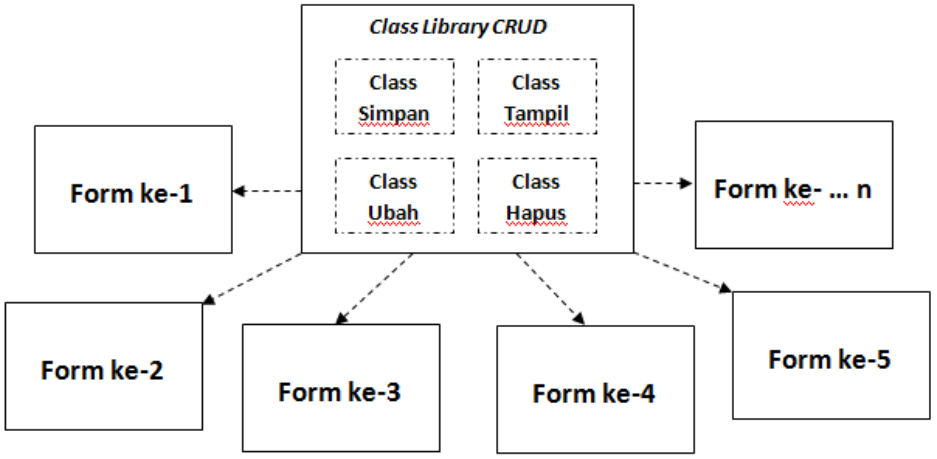
Meskipun Java menjadi bahasa pemrograman paling populer nomor dua, *syntax* Java sulit dipahami oleh para pemula, misalnya untuk membuat aplikasi *CRUD* (*Create Read Update Delete*) sederhana yang masih membutuhkan banyak baris *syntax*, padahal sebuah aplikasi dapat disebut memenuhi standar minimal jika terdapat proses *CRUD* didalamnya.

Dengan menggunakan konsep *Object Oriented Programming* (*OOP*) yang dibungkus dalam suatu *class library*, *syntax CRUD* yang kompleks dapat dijadikan menjadi lebih sederhana, sehingga pembuatan aplikasi pada *project java swing* menjadi lebih cepat dan mudah. Penelitian ini bertujuan untuk menghasilkan sebuah *class library* yang dapat mempermudah proses CRUD pada *project java swing* dengan menggunakan konsep *Object Oriented Programming*, sehingga dapat

menghemat penulisan *syntax* dan membuat proses *CRUD* menjadi lebih sederhana, serta dapat membantu programmer untuk lebih fokus membuat design aplikasi tanpa harus direpotkan dengan proses *CRUD*.

2. METODOLOGI

Dalam penelitian ini, penulis membuat desain eksperimen yang dapat dilihat pada Gambar 1:

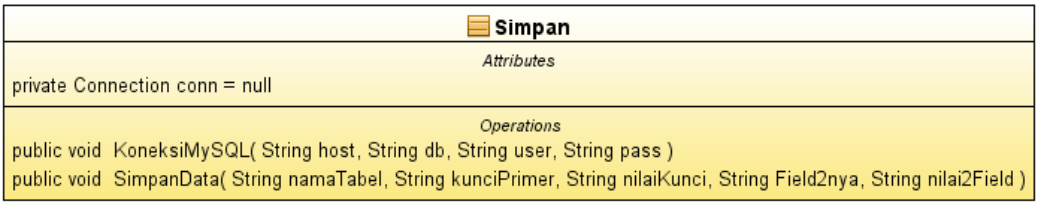


Gambar 1. Desain Eksperimen

Class Library CRUD yang dibuat memiliki 4 bagian di dalamnya, yaitu:
- *Class* Simpan untuk melakukan proses penyimpanan data ke database.
- *Class* Tampil untuk menampilkan data dari database.
- *Class* Ubah untuk melakukan proses perubahan dari dan ke database.
- *Class* Hapus untuk melakukan proses penghapusan data di database.

Class Library CRUD tersebut kemudian dipanggil oleh form yang didalamnya terdapat proses *CRUD*, pemanggilan *Class Library CRUD* dapat dilakukan tidak hanya sekali oleh sebuah *form* tetapi dapat dilakukan berulang-ulang sampai *form* ke-*n*.

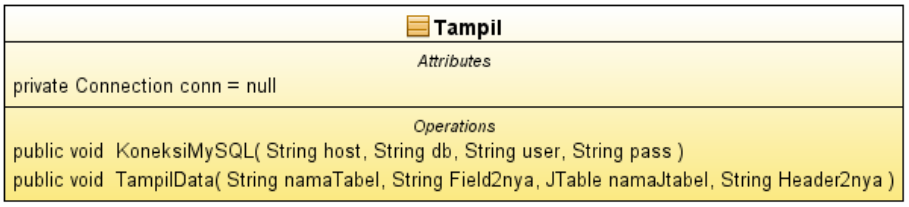
Berikut Class Diagram untuk masing-masing *Class* yang ada pada *Class Library CRUD*:
- *Class* Simpan



Gambar 2. *Class* Simpan

Pada Gambar 2, *Class* Simpan memiliki atribut conn, sedangkan *Method* / Operasi yang dimiliki adalah KoneksiMySQL dan SimpanData. *Class* Simpan digunakan untuk proses penyimpanan data ke database.

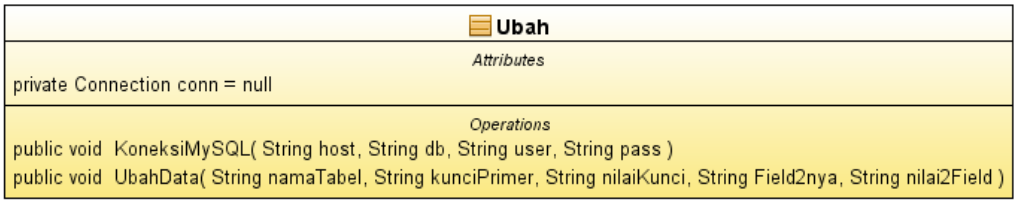
- *Class* Tampil



Gambar 3. *Class* Tampil

Pada Gambar 3, *Class* Tampil memiliki atribut conn, sedangkan *Method* / Operasi yang dimiliki adalah KoneksiMySQL dan TampilData. *Class* Tampil digunakan untuk proses menampilkan data dari database.

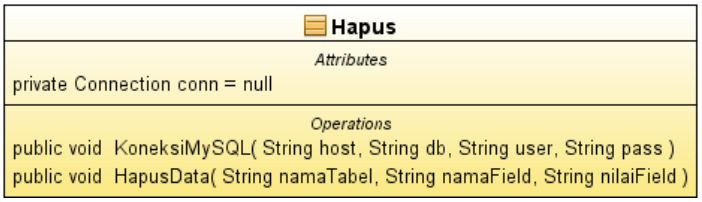
- *Class* Ubah



Gambar 4. *Class* Ubah

Pada Gambar 4, *Class* Ubah memiliki atribut conn, sedangkan *Method* / Operasi yang dimiliki adalah KoneksiMySQL dan UbahData. *Class* Ubah digunakan untuk melakukan proses perubahan dari dan ke database.

- *Class* Hapus

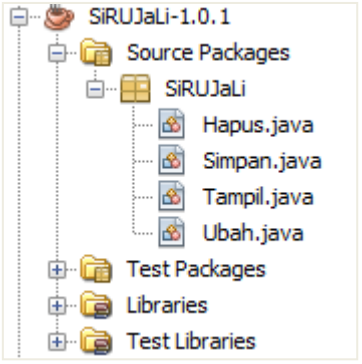


Gambar 5. *Class* Hapus

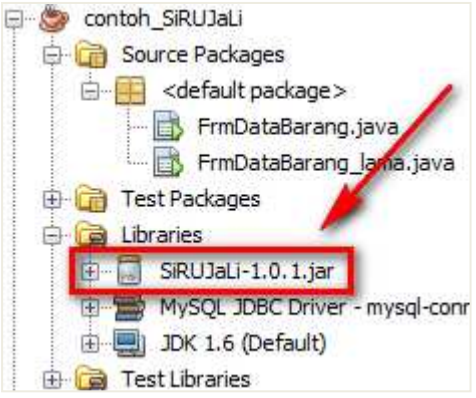
Pada Gambar 5, *Class* Hapus memiliki atribut conn, sedangkan *Method* / Operasi yang dimiliki adalah KoneksiMySQL dan HapusData. *Class* Hapus digunakan untuk melakukan proses penghapusan data di database.

3. HASIL DAN PEMBAHASAN

Masing-masing *Class* yang dibuat dijadikan satu dalam sebuah *project* dengan nama "**SiRUJaLi-1.0.1**", Project dari *Class Library CRUD* dapat dilihat pada Gambar 6. Setelah di *build*, *Class Library* tersebut menjadi "**SiRuJaLi-1.0.1.jar**" dan dapat digunakan untuk pembuatan aplikasi pada *project java swing*, penggunaan *Class Library CRUD* pada Project *Java Swing* dapat dilihat pada Gambar 7.



Gambar 6. Project *Class Library CRUD*



Gambar 7. Contoh penggunaan *Class Library CRUD* pada Project *Java Swing*

Penggunaan *Class Library CRUD* dalam proses simpan data, dapat dilihat pada Gambar 8 dan Gambar 9.

```
private void btnSaveActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        Class.forName("com.mysql.jdbc.Driver");
        conn = DriverManager.getConnection(
                "jdbc:mysql://localhost/db_toko?user=root");

        try {
            PreparedStatement pStatement = conn.prepareStatement(""
            + "INSERT INTO tb_barang(kode_barang,nama_barang,harga_satuan)"
            + "VALUES (?,?,?)");

            pStatement.setString(1, txtKodebarang.getText());
            pStatement.setString(2, txtNamaBarang.getText());
            pStatement.setString(3, txtHargaSatuan.getText());

            if (pStatement.executeUpdate()>0){
               JOptionPane.showMessageDialog(null, "Data berhasil disimpan",
                        "Informasi", JOptionPane.INFORMATION_MESSAGE);
            } else {
                JOptionPane.showMessageDialog(null, "Data gagal disimpan",
                    "Informasi", JOptionPane.INFORMATION_MESSAGE);
            }
            pStatement.close();
            conn.close();
        }
        catch (SQLException e){
            System.out.println("kesalahan: " + e.toString());
        }

    } catch (ClassNotFoundException e) {
        JOptionPane.showMessageDialog(null, "jdbc.Driver tidak ditemukan",
                "Kesalahan", JOptionPane.ERROR_MESSAGE);
        System.exit(0);
    } catch (SQLException ex){
        JOptionPane.showMessageDialog(null, "koneksi gagal: \n"
                + ex.toString(), "Kesalahan", JOptionPane.ERROR_MESSAGE);
        System.exit(0);
    }
}
```

Gambar 8. *Syntax* Simpan tanpa *Class Library CRUD*

```
private void btnSaveActionPerformed(java.awt.event.ActionEvent evt) {
    Simpan smp = new Simpan();
    smp.KoneksiMySQL("localhost", "db_toko", "root", "");
    smp.SimpanData(
            "tb_barang",
            "kode_barang",
            txtKodebarang.getText(),
            ("nama_barang,harga_satuan"),
            (txtNamaBarang.getText() + "," + txtHargaSatuan.getText())
    );
}
```

Gambar 9. *Syntax* Simpan menggunakan *Class Library CRUD*

Dari gambar 8 dan gambar 9, dapat dilihat perbandingan *syntax* simpan data ketika menggunakan *Class Library CRUD* maupun tidak memakai *Class Library CRUD*. *Syntax* simpan data yang menggunakan *Class Library CRUD* menjadi lebih sederhana dan baris penulisan *syntax* menjadi hanya beberapa baris saja.

Untuk melakukan proses simpan data menggunakan *Class Library CRUD*, cukup dengan cara memanggil *method* SimpanData() yang memiliki *parameter*: nama tabel, *primary key* tabel, nilai untuk *primary key*, nama field selain *primary key*, dan nilai untuk field selain *primary key*. Jika nilai untuk field *primary key* lebih dari satu maka digunakan tanda koma “,” sebagai pemisah.

Sebelum memanggil *method* SimpanData(), harus memanggil method KoneksiMySQL() untuk melakukan koneksi ke database MySQL yang memiliki beberapa *parameter* dengan urutan: nama *server*, nama database, nama *username*, dan *password*.

Penggunaan *Class Library CRUD* dalam proses tampil data, dapat dilihat dari gambar Gambar 10 dan Gambar 11:

```
private void btnViewActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        Class.forName("com.mysql.jdbc.Driver");
        conn = DriverManager.getConnection(
                "jdbc:mysql://localhost/db_toko?user=root");

        final String[] headers = {
            "KODE BARANG", "NAMA BARANG", "HARGA SATUAN"
        };
        DefaultTableModel DftTabModel;
        DftTabModel = new DefaultTableModel(null, headers);
        jtblBarang.setModel(DftTabModel);

        try{

            String sql = "SELECT * FROM tb_barang";

            Statement stmt=conn.createStatement();
            ResultSet rs=stmt.executeQuery(sql);

            while(rs.next()){
                String dKodeBarang=rs.getString("kode_barang");
                String dNamaBarang=rs.getString("nama_barang");
                String dHargaSatuan=rs.getString("harga_satuan");
                String[] data={dKodeBarang,dNamaBarang,dHargaSatuan};
                DftTabModel.addRow(data);
            }
            rs.close();
            conn.close();
        }catch(Exception ex){
            JOptionPane.showMessageDialog(this, "ERROR: \n" + ex.toString(),
                    "Kesalahan", JOptionPane.ERROR_MESSAGE);
        }

    } catch (ClassNotFoundException e) {
        JOptionPane.showMessageDialog(null, "jdbc.Driver tidak ditemukan",
                "Kesalahan", JOptionPane.ERROR_MESSAGE);
        System.exit(0);
    } catch (SQLException ex){
        JOptionPane.showMessageDialog(null, "koneksi gagal: \n"
                + ex.toString(), "Kesalahan", JOptionPane.ERROR_MESSAGE);
        System.exit(0);
    }
}
```

Gambar 10. *Syntax* Tampil tanpa *Class Library CRUD*

```
private void btnViewActionPerformed(java.awt.event.ActionEvent evt) {
    Tampil tmpl = new Tampil();
    tmpl.KoneksiMySQL("localhost", "db_toko", "root", "");
    tmpl.TampilData(
            "tb_barang",
            "kode_barang,nama_barang,harga_satuan",
            jtblBarang,
            "KODE BARANG,NAMA BARANG,HARGA SATUAN"
    );
}
```

Gambar 11. *Syntax* Tampil menggunakan *Class Library CRUD*

Dari gambar 10 dan gambar 11, dapat dilihat perbandingan *syntax* tampil data ketika menggunakan *Class Library CRUD* maupun tidak memakai *Class Library CRUD*. *Syntax* tampil data yang menggunakan *Class Library CRUD* menjadi lebih sederhana dan baris penulisan *syntax* menjadi hanya beberapa baris saja.

Untuk melakukan proses tampil data menggunakan *Class Library CRUD*, cukup dengan cara memanggil *method* TampilData() yang memiliki *parameter*: nama tabel, nama field yang akan ditampilkan datanya, nama *jtable* untuk menampilkan data ke *java swing*, dan judul untuk data yang ditampilkan pada jtable. Sebelum memanggil *method* tampilData(), harus memanggil method KoneksiMySQL() untuk melakukan koneksi ke database MySQL yang memiliki beberapa *parameter* dengan urutan: nama *server*, nama database, nama *username*, dan *password*.

Penggunaan *Class Library CRUD* dalam proses ubah data, dapat dilihat dari Gambar 12 dan Gambar 13:

```
private void btnEditActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        Class.forName("com.mysql.jdbc.Driver");
        conn = DriverManager.getConnection(
                "jdbc:mysql://localhost/db_toko?user=root");

        try {
            PreparedStatement pStatement = null;
            String sql ="UPDATE tb_barang " +
                "set nama_barang=?, harga_satuan=? " +
                " WHERE kode_barang=? ";
            pStatement = conn.prepareStatement(sql);

            pStatement.setString(1, txtNamaBarang.getText());
            pStatement.setString(2, txtHargaSatuan.getText());
            pStatement.setString(3, txtKodebarang.getText());

            if (pStatement.executeUpdate()>0){
               JOptionPane.showMessageDialog(null, "Data berhasil diubah",
                        "Informasi", JOptionPane.INFORMATION_MESSAGE);
            } else {
                JOptionPane.showMessageDialog(null, "Data gagal diubah",
                    "Informasi", JOptionPane.INFORMATION_MESSAGE);
            }
            pStatement.close();
            conn.close();
        }
        catch (SQLException e){
            System.out.println("kesalahan: " + e.toString());
        }

    } catch (ClassNotFoundException e) {
        JOptionPane.showMessageDialog(null, "jdbc.Driver tidak ditemukan",
                "Kesalahan", JOptionPane.ERROR_MESSAGE);
        System.exit(0);
    } catch (SQLException ex){
        JOptionPane.showMessageDialog(null, "koneksi gagal: \n"
                + ex.toString(), "Kesalahan", JOptionPane.ERROR_MESSAGE);
        System.exit(0);
    }
}
```

Gambar 12. *Syntax* Ubah tanpa *Class Library CRUD*

```
private void btnEditActionPerformed(java.awt.event.ActionEvent evt) {
    Ubah ubh = new Ubah();
    ubh.KoneksiMySQL("localhost", "db_toko", "root", "");
    ubh.UbahData(
            "tb_barang",
            "kode_barang",
            txtKodebarang.getText(),
            ("nama_barang,harga_satuan"),
            (txtNamaBarang.getText() + "," + txtHargaSatuan.getText())
    );
}
```

Gambar 13. *Syntax* Ubah menggunakan *Class Library CRUD*

Dari gambar 12 dan gambar 13, dapat dilihat perbandingan *syntax* tampil data ketika menggunakan *Class Library CRUD* maupun tidak memakai *Class Library CRUD*. *Syntax* tampil data yang menggunakan *Class Library CRUD* menjadi lebih sederhana dan baris penulisan *syntax* menjadi hanya beberapa baris saja.

Untuk melakukan proses simpan data menggunakan *Class Library CRUD*, cukup dengan cara memanggil *method* UbahData() yang memiliki *parameter*: nama tabel, *primary key* tabel, nilai untuk *primary key*, nama field selain *primary key*, nilai untuk field selain *primary key*. Jika nilai untuk field *primary key* lebih dari satu maka digunakan tanda koma "," sebagai pemisah. Sebelum memanggil *method* UbahData(), harus memanggil method KoneksiMySQL() untuk melakukan koneksi ke database MySQL yang memiliki beberapa *parameter* dengan urutan: nama *server*, nama database, nama *username*, dan *password*.

Penggunaan *Class Library CRUD* dalam proses hapus data, dapat dilihat dari Gambar 14 dan Gambar 15:

```
private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        Class.forName("com.mysql.jdbc.Driver");
        conn = DriverManager.getConnection(
                "jdbc:mysql://localhost/db_toko?user=root");

        try {
            PreparedStatement pStatement = conn.prepareStatement(""
                    + "DELETE FROM tb_barang "
                    + " WHERE kode_barang=? ");

            pStatement.setString(1, txtKodebarang.getText());

            if (pStatement.executeUpdate() > 0) {
                JOptionPane.showMessageDialog(this, "Data berhasil dihapus",
                        "Informasi", JOptionPane.INFORMATION_MESSAGE);
            } else {
                JOptionPane.showMessageDialog(this, "Data gagal dihapus",
                        "Informasi", JOptionPane.INFORMATION_MESSAGE);
            }
            pStatement.close();
            conn.close();
        } catch (SQLException e) {
            JOptionPane.showMessageDialog(this, "ERROR: \n" + e.toString(),
                    "Kesalahan", JOptionPane.WARNING_MESSAGE);
        }

    } catch (ClassNotFoundException e) {
        JOptionPane.showMessageDialog(null, "jdbc.Driver tidak ditemukan",
                "Kesalahan", JOptionPane.ERROR_MESSAGE);
        System.exit(0);
    } catch (SQLException ex){
        JOptionPane.showMessageDialog(null, "koneksi gagal: \n"
                + ex.toString(), "Kesalahan", JOptionPane.ERROR_MESSAGE);
        System.exit(0);
    }
}
```

Gambar 14. *Syntax* Hapus tanpa *Class Library CRUD*

```
private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) {
    Hapus hps = new Hapus();
    hps.KoneksiMySQL("localhost", "db_toko", "root", "");
    hps.HapusData(
            "tb_barang",
            "kode_barang",
            txtKodebarang.getText()
    );
}
```

Gambar 15. *Syntax* Hapus menggunakan *Class Library CRUD*

Dari gambar 14 dan gambar 15, dapat dilihat perbandingan *syntax* hapus data ketika menggunakan *Class Library CRUD* maupun tidak memakai *Class Library CRUD*. *Syntax* hapus data yang menggunakan *Class Library CRUD* menjadi lebih sederhana dan baris penulisan *syntax* menjadi hanya beberapa baris saja.

Untuk melakukan proses hapus data menggunakan *Class Library CRUD*, cukup dengan cara memanggil *method* HapusData() yang memiliki *parameter*: nama tabel, *primary key* tabel, dan nilai untuk *primary key*. Sebelum memanggil *method* HapusData(), harus memanggil method KoneksiMySQL() untuk melakukan koneksi ke database MySQL yang memiliki beberapa *parameter* dengan urutan: nama *server*, nama database, nama *username*, dan *password*.

4. KESIMPULAN

Dari hasil penelitian dan uji *project*, telah dihasilkan suatu *Class Library* yang dapat digunakan untuk pembuatan aplikasi *CRUD* pada *project Java Swing*. Dengan menggunakan *Class Library* ini dapat menjadikan proses pembuatan aplikasi *CRUD* menjadi lebih mudah dan cepat dikarenakan programmer tidak perlu berlama-lama dalam proses CRUD dan programmer dapat lebih fokus dalam pembuatan design aplikasi. Meskipun *Class Library* ini dapat mempermudah pembuatan aplikasi *CRUD*, tetapi masih memiliki kekurangan yaitu jumlah *field* yang dapat digunakan terbatas 10 *field*. *Class Library CRUD* ini dapat digunakan oleh siapapun, baik pemula ataupun ahli.

DAFTAR PUSTAKA

Farooq M.S., Abid A., *et al*., 2012, "A *Qualitative Framework For Introducing Programming Language At High School*", Journal of Quality and Technology Management, Volume VIII, Issue II, December 2012, Page 135–151.

Horstmann, 2010, "Big Java 4th Edition", John Wiley & Sons, INC.

Index TIOBE, Juli 2014, www.tiobe.com.

JENI, 2007, "Pengenalan *Bahasa Java*".

JENI, 2007, "*Review Konsep Dasar dalam Java*".