

**Perancangan Aplikasi *Chatting* Berbasis Web  
menggunakan *Framework* CodeIgniter, Socket.IO dan  
*Framework* Foundation  
(Studi Kasus : PT Pura Barutama)**

**Artikel Ilmiah**

**Diajukan kepada  
Fakultas Teknologi Informasi  
untuk Memperoleh Gelar Sarjana Komputer**



**Peneliti**

**Ridvandani Dwi Purnomo Aji (672012078)  
Ramos Somya, S.Kom., M.Cs.**

**Program Studi Teknik Informatika  
Fakultas Teknologi Informasi  
Universitas Kristen Satya Wacana  
Salatiga  
Juli 2016**



PERPUSTAKAAN UNIVERSITAS  
UNIVERSITAS KRISTEN SATYA WACANA  
Jl. Diponegoro 52 – 60 Salatiga 50711  
Jawa Tengah, Indonesia  
Telp. 0298 – 321212, Fax. 0298 321433  
Email: library@adm.uksw.edu ; http://library.uksw.edu

### PERNYATAAN TIDAK PLAGIAT

Saya yang bertanda tangan di bawah ini:

Nama : RIDVANDANI DWI PURNOMO AJI  
NIM : 672012078 Email : ridvandaniidwipa@gmail.com  
Fakultas : TEKNOLOGI INFORMASI Program Studi : TEKNIK INFORMATIKA  
Judul tugas akhir : PERANCANGAN APLIKASI CHATTING BERBASIS WEB MENGGUNAKAN  
FRAMEWORK CODEIGNITER, SOCKET.IO, DAN FRAMEWORK FOUNDATION  
(STUDI KAJUS : PT PURA BARUTAMA)  
Pembimbing : 1. RAMOS SOMYA, S.KOM., M.CS.  
2. \_\_\_\_\_

Dengan ini menyatakan bahwa:

1. Hasil karya yang saya serahkan ini adalah asli dan belum pernah diajukan untuk mendapatkan gelar kesarjanaan baik di Universitas Kristen Satya Wacana maupun di institusi pendidikan lainnya.
2. Hasil karya saya ini bukan saduran/terjemahan melainkan merupakan gagasan, rumusan, dan hasil pelaksanaan penelitian/implementasi saya sendiri, tanpa bantuan pihak lain, kecuali arahan pembimbing akademik dan narasumber penelitian.
3. Hasil karya saya ini merupakan hasil revisi terakhir setelah diujikan yang telah diketahui dan disetujui oleh pembimbing.
4. Dalam karya saya ini tidak terdapat karya atau pendapat yang telah ditulis atau dipublikasikan orang lain, kecuali yang digunakan sebagai acuan dalam naskah dengan menyebutkan nama pengarang dan dicantumkan dalam daftar pustaka.

Pernyataan ini saya buat dengan sesungguhnya. Apabila di kemudian hari terbukti ada penyimpangan dan ketidakbenaran dalam pernyataan ini maka saya bersedia menerima sanksi akademik berupa pencabutan gelar yang telah diperoleh karena karya saya ini, serta sanksi lain yang sesuai dengan ketentuan yang berlaku di Universitas Kristen Satya Wacana.

Salatiga, 13 September 2016



Ridvandani Dwi Purnomo Aji

F-LIB-080



PERPUSTAKAAN UNIVERSITAS  
UNIVERSITAS KRISTEN SATYA WACANA  
Jl. Diponegoro 52 – 60 Salatiga 50711  
Jawa Tengah, Indonesia  
Telp. 0298 – 321212, Fax. 0298 321433  
Email: library@adm.uksw.edu ; http://library.uksw.edu

## PERNYATAAN PERSETUJUAN AKSES

Saya yang bertanda tangan di bawah ini:

Nama : RIDVANDANI DWI PURNOMO AJI  
NIM : 612012078 Email : ridvandaniidwipa@gmail.com  
Fakultas : TEKNOLOGI INFORMASI Program Studi : TEKNIK INFORMATIKA  
Judul tugas akhir : PERANCANGAN APLIKASI CHATTING BERBASIS WEB MENGGUNAKAN  
FRAMEWORK CODEIGNITER, SOCKET.IO, DAN FRAMEWORK FOUNDATION  
(STUDI KASUS : PT PURA BARUTAMA)

Dengan ini saya menyerahkan hak *non-eksklusif*\* kepada Perpustakaan Universitas – Universitas Kristen Satya Wacana untuk menyimpan, mengatur akses serta melakukan pengelolaan terhadap karya saya ini dengan mengacu pada ketentuan akses tugas akhir elektronik sebagai berikut (beri tanda pada kotak yang sesuai):

- ☒ a. Saya mengizinkan karya tersebut diunggah ke dalam aplikasi Repositori Perpustakaan Universitas, dan/atau portal GARUDA
- ☐ b. Saya tidak mengizinkan karya tersebut diunggah ke dalam aplikasi Repositori Perpustakaan Universitas, dan/atau portal GARUDA\*\*

\* Hak yang tidak terbatasnya bagi satu pihak saja. Pengajar, peneliti, dan mahasiswa yang menyerahkan hak non-eksklusif kepada Repositori Perpustakaan Universitas saat mengumpulkan hasil karya mereka masih memiliki hak copyright atas karya tersebut.

\*\* Hanya akan menampilkan halaman judul dan abstrak. Pilihan ini harus dilampiri dengan penjelasan/ alasan tertulis dari pembimbing TA dan diketahui oleh pimpinan fakultas (dekan/kaprodi).

Demikian pernyataan ini saya buat dengan sebenarnya.

Salatiga, 13 September 2016

RIDVANDANI DWI PURNOMO AJI

Tanda tangan & nama terang mahasiswa

Mengetahui,

Ramus Somp

Tanda tangan & nama terang pembimbing



**Perancangan Aplikasi *Chatting* Berbasis *Web* Menggunakan  
Framework CodeIgniter, Socket.IO, dan Framework Foundation  
(Studi Kasus : PT Pura Barutama)**

Oleh,

**Ridvandani Dwi Purnomo Aji**  
NIM : 672012078

**ARTIKEL ILMIAH**


Diajukan Kepada Program Studi Teknik Informatika guna memenuhi sebagian  
dari persyaratan untuk mencapai gelar Sarjana Komputer

Disetujui oleh,

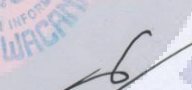


Ramos Somya, S.Kom., M.Cs.  
Pembimbing

Diketahui oleh,



Dr. Dharmaputra T. Palekahelu, M.Pd.  
Dekan



Dr. Dharmaputra T. Palekahelu, M.Pd.  
Pjs. Ketua Program Studi

**FAKULTAS TEKNOLOGI INFORMASI  
UNIVERSITAS KRISTEN SATYA WACANA  
SALATIGA  
2016**

### Lembar Pengesahan


Judul Artikel : Perancangan Aplikasi *Chatting* Berbasis *Web*  
Menggunakan *Framework* CodeIgniter, Socket.IO  
dan *Framework* Foundation (Studi Kasus : PT Pura  
Barutama)  
Nama Mahasiswa : Ridvandani Dwi Purnomo Aji  
NIM : 672012078  
Program Studi : Teknik Informatika  
Fakultas : Teknologi Informasi

Menyetujui,

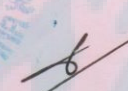


Ramos Somya, S.Kom., M.Cs.  
Pembimbing

Mengesahkan



Dr. Dharmaputra T. Palekahelu, M.Pd.  
Dekan



Dr. Dharmaputra T. Palekahelu, M.Pd.  
Pjs. Ketua Program Studi

Dinyatakan Lulus tanggal: 9 September 2016

Reviewer :

1956

- Ariya Dwika Cahyono, S.Kom., M.T.



## 1. Pendahuluan

Aplikasi *chatting* merupakan aplikasi yang penting pada sebuah perusahaan yang besar karena dapat dipastikan antara divisi satu dan divisi yang lainnya berada di tempat yang berjauhan. Divisi keuangan PT Pura Barutama selalu mengontrol dan berkomunikasi dengan seluruh *unit* yang ada di PT Pura Barutama. Penggunaan aplikasi *chatting* akan meningkatkan produktivitas dan efisiensi kerja. Misalnya sebagai sarana berkomunikasi, *attachment file* dan sebagai media pengumuman.

EDP Keuangan adalah salah satu sub divisi dari divisi keuangan di PT Pura Barutama selalu berinteraksi dengan berbagai macam divisi yang ada. Selain itu EDP Keuangan bertugas dalam pembuatan program yang terkait dengan pengolahan data keuangan, faktur di semua *unit* PT Pura Barutama. Aplikasi keuangan yang telah ada akan selalu berkembang sesuai kebutuhan *user*.

Aktivitas komunikasi dan penyebaran informasi pada PT Pura Barutama dilakukan dengan menggunakan layanan *email* maupun aplikasi *messenger*, untuk menggunakan layanan tersebut, perangkat yang digunakan harus selalu terkoneksi dengan *internet*. Apabila *internet* mati maka komunikasi tidak dapat dilakukan. Jalan satu-satunya untuk menyebarkan informasi adalah dengan menelepon satu-persatu ke tiap-tiap *unit*. Hal ini menimbulkan ketidakefisienan pekerjaan dan produktivitas menjadi terganggu.

Berdasarkan permasalahan tersebut, maka dapat diketahui rumusan masalah dalam penelitian ini adalah bagaimana membuat aplikasi *chatting* berbasis *web* menggunakan *Framework* CodeIgniter, Socket.IO dan *Framework* Foundation dengan memanfaatkan jaringan komputer lokal yang ada di PT Pura Barutama. Aplikasi *chatting* tersebut hanya membutuhkan sebuah *browser* yang digunakan untuk mengakses aplikasi *chatting* selama perangkat terkoneksi dengan jaringan lokal baik menggunakan kabel maupun *wireless* tanpa terkoneksi dengan *internet*. Selain itu, aplikasi berbasis *web* tergolong ringan, sehingga spesifikasi komputer yang digunakan oleh *user* tidak berat.

Aplikasi *chatting* akan dibangun menggunakan *Framework* CodeIgniter (CI) dan Socket.IO sebagai *back-end*. *Framework* CodeIgniter mendukung konsep *Model View Controller* (MVC) sehingga dalam pengembangan aplikasi *chatting* akan menjadi lebih terstruktur dan terorganisir. Sedangkan untuk *interface* (*front-end*) aplikasi *chatting* menggunakan *Framework* Foundation untuk membuat aplikasi *chatting* ini menjadi *responsive* dan untuk penyimpanan data menggunakan *database* Oracle karena menyesuaikan dengan *database* yang digunakan PT Pura Barutama.

Mengingat begitu luasnya ruang lingkup pembuatan aplikasi *chatting* dan terbatasnya kemampuan dan identifikasi maka diperlukan batasan agar sistem yang dibangun tidak menyimpang dari permasalahan dalam sistem aplikasi *chatting*. Adapun batasan masalah yang dilakukan adalah aplikasi hanya dapat dipergunakan untuk perorangan (*Private Chat*), dikarenakan aplikasi *chatting* ini bersifat rahasia. Selain digunakan untuk melakukan *private chat*, aplikasi ini juga dapat memberikan informasi berupa pengumuman kepada *user* maupun *unit* yang lain apabila ada pembaharuan aplikasi yang dibuat oleh EDP Keuangan PT Pura Barutama.

Diharapkan dengan dibangunnya aplikasi ini, dapat mempermudah para penggunanya untuk berkomunikasi dan bertukar informasi. Sehingga dapat meningkatkan produktivitas kerja dan mengefisienkan waktu.



## 2. Kajian Pustaka

Terdapat beberapa penelitian terdahulu yang digunakan sebagai acuan dalam penelitian ini. Penelitian pertama menghasilkan aplikasi yang dapat melakukan pembuatan suatu *room* yang dapat dilakukan oleh semua pengguna baik itu pengguna *handphone* maupun pengguna komputer. Apabila melakukan mengirim dengan ukuran *file* yang besar maka proses pengiriman akan menjadi lama [1].

Penelitian kedua menghasilkan aplikasi yang dibangun dengan menggunakan Visual Basic 6.0. Aplikasi yang dilengkapi dengan menu *send file* untuk mengirimkan *text* yang mempermudah komunikasi antar pengguna yang berada dalam sebuah jaringan. Aplikasi *chatting* yang dikembangkan tidak adanya pemisahan *form* antara *private* dengan *group*. Model pengiriman *text* melalui menu *send file* harus memasukkan alamat IP tujuan pengguna sehingga dinilai tidak efektif dan efisien. Program aplikasi hanya dapat berjalan pada sistem operasi Windows [2].

Penelitian ketiga mengimplementasi Algoritma Rijndael 128 pada aplikasi *chatting* berbasis HTML5 *webSocket*. Pada penelitian tersebut menghasilkan aplikasi *group chatting*. Pesan yang dikirim akan dienkripsi dengan menggunakan algoritma Rijndael 128, sehingga pesan menjadi aman [3].

Berdasarkan penelitian yang telah dilakukan sebelumnya yang menjadi pembeda antara penelitian ini dan penelitian terdahulu adalah aplikasi *chatting* dibuat dengan menggunakan *Framework CodeIgniter*, *Socket.IO* dan *Framework Foundation*. Penggunaan *Framework CodeIgniter* penulisan kode akan menjadi terstruktur dan terorganisir karena *Framework CodeIgniter* menerapkan konsep MVC (*Model View Controller*). *Socket.IO* memungkinkan pembuatan aplikasi komunikasi *realtime* antara *client* dan *server*. Penggunaan *Framework Foundation* membuat halaman *web* menjadi *responsive* sehingga dapat menyesuaikan di perangkat manapun saat aplikasi diakses.

*Chatting* adalah percakapan dua orang atau lebih secara *real time* melalui komputer yang terhubung dengan jaringan. Layanan untuk *chatting* di *internet* antara lain *Yahoo*, *Skype*, *mIRC*, *Windows Live Messenger*. Adanya layanan *chat* memungkinkan untuk dapat berkomunikasi melalui *internet* dengan orang-orang yang berada di seluruh dunia [4].

*Framework* adalah kumpulan perintah atau fungsi dasar yang membentuk aturan-aturan tertentu dan saling berinteraksi satu sama lain sehingga dalam pembuatan aplikasi *website*, kita harus mengikuti aturan dari *Framework* tersebut. *CodeIgniter* dilengkapi dengan berbagai pustaka siap pakai untuk berbagai kebutuhan. Misalnya saja koneksi *database*, *email*, *session* dan *cookies*, keamanan, manipulasi gambar dan banyak lagi sehingga mempermudah pekerjaan [5].

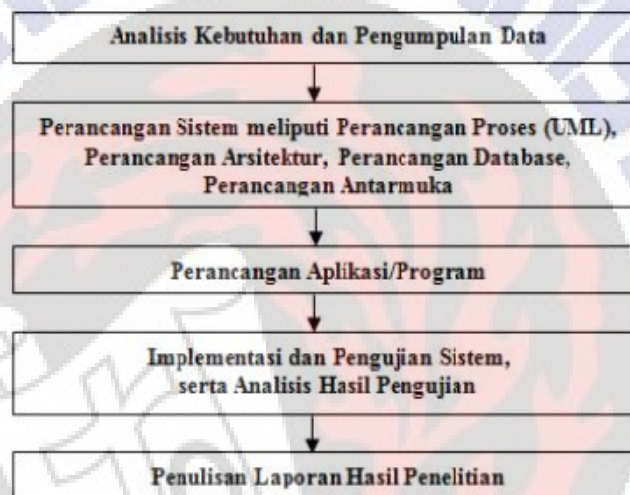
*Socket.IO* adalah lapisan komunikasi berbasis *event* untuk aplikasi *web realtime*, yang dibangun di atas *Engine.IO*. Hal ini memungkinkan pengembang untuk mengirim dan menerima data tanpa khawatir tentang kompatibilitas *browser* yang berbeda [6].

*Framework Foundation* adalah *fremework* yang dibangun dengan HTML, CSS dan Javascript, sebagai komponen utama dari *web*. *Framework Foundation* menggunakan teknologi JQuery, HTML 5 dan Normalizr [7]. *Framework Foundation* digunakan untuk membuat *web* dapat menyesuaikan resolusi layar yang digunakan.

Oracle adalah sistem basis data yang memiliki banyak fitur yang memungkinkan administrator basis data dapat mengelola data secara lebih akurat sehingga Oracle lebih sesuai digunakan sebagai sistem basis data untuk aplikasi yang berukuran besar dan kompleks [8].

### 3. Metode Perancangan Sistem

Pada penelitian ini dilakukan beberapa tahapan yang saling berkaitan dengan tahapan selanjutnya, yaitu : 1) Analisis kebutuhan dan pengumpulan data yang diperlukan. 2) Perancangan sistem. 3) Perancangan aplikasi/program. 4) Implementasi dan pengujian sistem, serta analisis hasil pengujian. 5) Penulisan laporan hasil penelitian [9]. Tahapan-tahapan yang dilakukan dalam penelitian ini dapat dilihat pada Gambar 1.



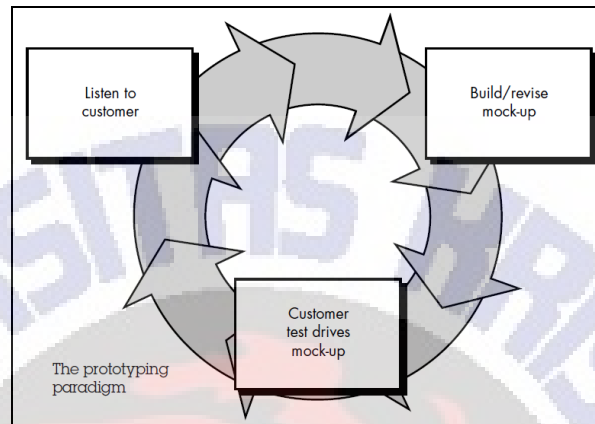
Gambar 1 Tahapan Penelitian

Berdasarkan Gambar 1 dapat dijelaskan tahapan penelitian yang dilakukan adalah sebagai berikut : 1) Tahapan pertama : analisis dan pengumpulan data, di mana peneliti melakukan wawancara dengan karyawan EDP Keuangan tentang aplikasi yang akan dibuat. Berdasarkan hasil wawancara dengan karyawan EDP Keuangan bahwa selama ini belum ada media yang menghubungkan *unit* satu ke *unit* yang lainnya dalam menyebarkan informasi atau *attachment file* tanpa menggunakan *internet*. Selama ini hanya menggunakan telepon dan *email* untuk berkomunikasi. *Email* sendiri memiliki kekurangan yaitu membutuhkan koneksi *internet*. Jika koneksi *internet* mati, maka tidak bisa terjadi komunikasi. Jika melalui telepon akan terasa tidak efisien karena karyawan EDP Keuangan harus menghubungi satu persatu *unit* yang banyak di PT Pura Barutama jika terjadi perubahan program. Sehingga dibutuhkan aplikasi yang dapat menyediakan fasilitas *chatting*, *attachment file* dan penyebaran pengumuman dengan memanfaatkan jaringan lokal. Tahap kedua, ketiga dan keempat dilakukan perancangan aplikasi *chatting* menggunakan metode *Prototype*. Sedangkan tahap kelima dilakukan penulisan artikel ilmiah atau laporan penelitian.

Metode perancangan yang dipakai dalam pembuatan aplikasi *chatting* adalah metode *prototyping*. Metode *prototyping* adalah metode dalam pengembangan rekayasa *software* yang bertahap dan berulang serta mementingkan sisi *user* sistem.



Penggunaan metode *prototyping*, pengembang dan karyawan EDP Keuangan dapat saling berinteraksi selama proses pembuatan sistem sampai aplikasi sesuai dengan kebutuhan pengguna.

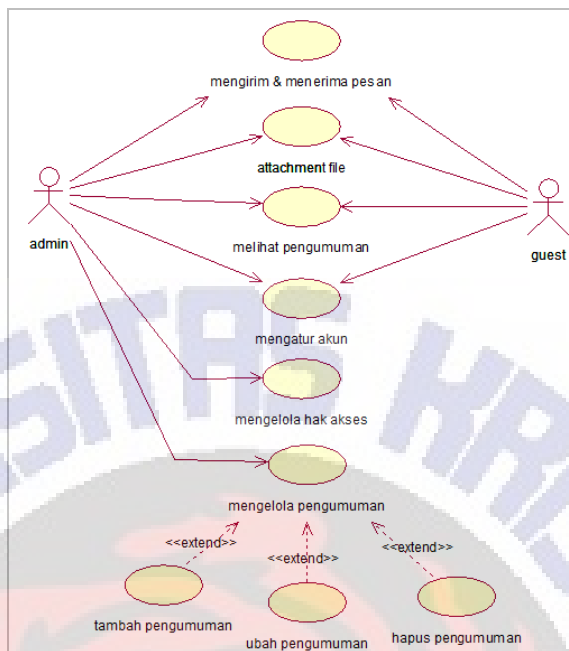


**Gambar 2 Model Prototyping [10]**

Tahap pengumpulan kebutuhan dilakukan untuk mengetahui permasalahan dan kebutuhan sistem. Pada tahap ini juga dilakukan pencarian data yang dibutuhkan oleh sistem. Agar aplikasi *chatting* yang dibangun dapat memenuhi kebutuhan pengguna. Analisis kebutuhan sistem dilakukan bersama dengan karyawan EDP Keuangan. Berdasarkan analisa kebutuhan sistem yang dilakukan bahwa selain untuk melakukan *chatting* dengan *user* lain, dibutuhkan juga fasilitas untuk *attachment file*, dan melakukan pengumuman. Pada aplikasi *chatting* yang dibangun dibagi menjadi dua hak akses yaitu administrator dan *user* biasa. Kebutuhan administrator mencakup : *chatting* dengan pengguna lain dengan fasilitas *attachment file*, *posting* pengumuman, hapus pengumuman, mengubah pengumuman jika terjadi kesalahan, mengelola *user*, mengelola *user* di sini maksudnya adalah memberikan hak akses administrator kepada *user* biasa jika *user* tersebut ini memposting pengumuman. Sedangkan kebutuhan *user* biasa hanya melakukan *chatting* dengan pengguna lain dengan fasilitas *attachment file*. Data *user* sendiri menggunakan data yang sudah dimiliki oleh PT Pura Barutama, sehingga pada aplikasi *chatting* ini tidak perlu menambahkan data *user* baru, karena diambil dari *database* yang sudah ada.

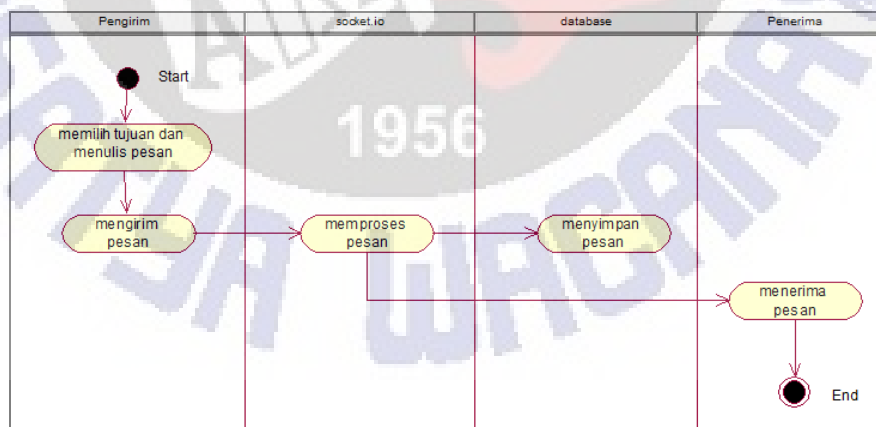
Analisis kebutuhan *hardware* dan spesifikasi *software* yang digunakan dalam membangun aplikasi *chatting* ini yaitu : analisis perangkat keras yang akan digunakan adalah Prosesor Intel Core i3, 2.3 GHz, RAM 6 GB dan Harddisk 500 GB. Sedangkan perangkat lunak yang digunakan adalah sistem Operasi Windows 7 Ultimate, Sublime sebagai *text editor*, *web server* (Apache), *web browser* (dalam penelitian ini menggunakan Google Chrome), Oracle sebagai *database*, dan Rational Rose untuk membuat UML diagram sistem.

Sebelum dilakukan pengkodean, dilakukan perancangan UML diagram untuk memvisualisasikan alur proses dan kebutuhan data. UML dibuat dalam diagram *Use Case*, diagram *Activity*, diagram *Sequence* dan diagram *Class* yang akan dijelaskan satu persatu.



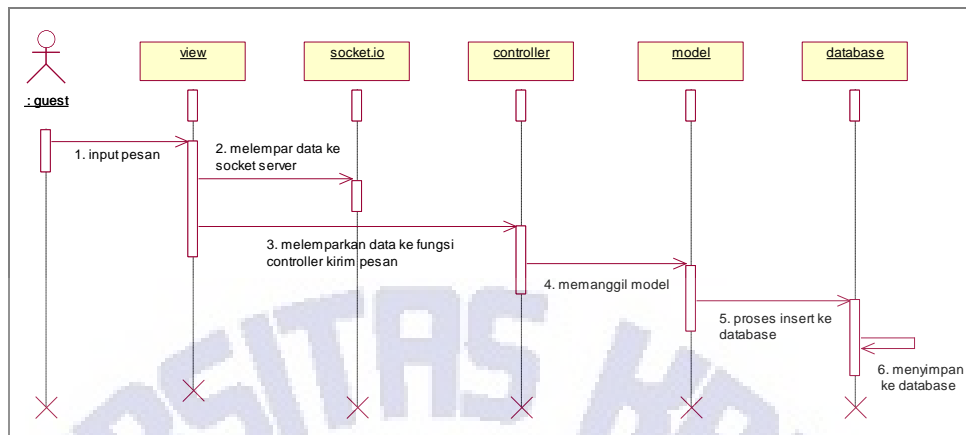
**Gambar 3** Diagram Use Case

Gambar 3 menunjukkan diagram *use case* dari aplikasi *chatting*. Aplikasi yang dibuat dibedakan menjadi dua jenis *user* yaitu administrator dan *guest* (*user* biasa). Aktor dengan hak akses admin dapat mengirimkan dan menerima pesan, *attachment file*, melihat pengumuman, mengelola data *user* dengan memberikan hak akses administrator kepada *user* biasa, *memposting* pengumuman, menghapus pengumuman, mengubah pengumuman jika terjadi kesalahan. Sedangkan aktor dengan hak akses *guest* (*user* biasa) hanya dapat mengirimkan dan menerima pesan, *attachment file*, melihat pengumuman, dan mengatur akunnya sendiri.



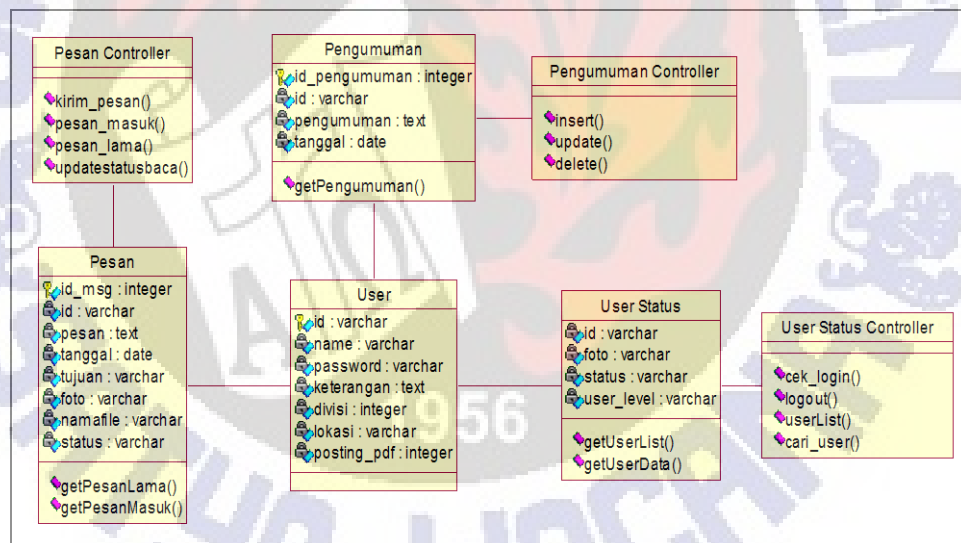
**Gambar 4** Diagram Activity Mengirim Pesan

Gambar 4 menunjukkan diagram *activity* ketika pengirim mengirimkan pesan. Hal pertama yang dilakukan adalah pengirim menentukan tujuan dan pesan yang akan dikirim. Kemudian data yang sudah diinputkan akan diproses melalui Socket kemudian ditampilkan pada panerima. Data pesan yang dikirim akan disimpan ke dalam *database*, sehingga histori percakapan masih bisa dilihat.



**Gambar 5** Diagram *Sequence* Proses Kirim Pesan

Gambar 5 merupakan diagram *sequence* dari proses pengiriman pesan. Pertama *user* menginputkan pesan pada halaman *view*. Kemudian data diproses oleh *server socket* dan data menjadi *parameter* di *controller* kirim pesan. Setelah itu *controller* akan memanggil fungsi pada *model* dengan *parameter* yang dikirimkan *controller* kemudian *model* akan berhubungan dengan *database* untuk menyimpan data.



**Gambar 6** Diagram *Class*

Gambar 6 merupakan diagram *class* dari aplikasi *chatting* yang menunjukkan *model* dan *controller*. Pada aplikasi *chatting* dibuat dalam 4 *model class* yaitu *user class*, *user status class*, *pesan class* dan *pengumuman class*. *Model class* ini yang nantinya akan berhubungan dengan *database*. Sedangkan *controller class* akan menghubungkan antara *view* dan *model class*. Pada aplikasi *chatting* dibuat dalam 3 *controller class* yaitu *user status controller*, *pesan controller*, dan *pengumuman controller*.

Perulangan dari proses pada metode *prototyping* terus berlangsung hingga semua kebutuhan terpenuhi. Proses evaluasi dilakukan sebanyak 3 kali, evaluasi pertama adalah menunjukkan kepada karyawan EDP Keuangan bagaimana aplikasi bekerja. Evaluasi kedua dilakukan penambahan fitur untuk *user* yang memiliki hak



akses sebagai administrator agar dapat melakukan *posting* pengumuman, dan setiap pesan yang masuk mendapatkan notifikasi. Evaluasi ketiga dilakukan perbaikan beberapa fungsi yang masih memiliki *bug* di dalamnya.

#### 4. Hasil Implementasi dan Pembahasan

Sebelum melakukan pengkodean. Hal pertama yang harus dilakukan adalah menginstall NodeJS. Tujuan dari instalasi NodeJS karena Socket.IO berjalan di atas NodeJS sebagai *module*. Kemudian dilakukan inisialisasi *project* NodeJS dengan perintah `npm init`. Perintah ini akan menghasilkan *file* `package.json` yang terlihat pada Kode Program 1.

**Kode Program 1** Konfigurasi NodeJS

```
1.  {
2.    "name": "puramessenger",
3.    "version": "1.0.0",
4.    "description": "puramessenger",
5.    "main": "server.js",
6.    "scripts": {
7.      "test": "echo \"Error: no test specified\" && exit 1"
8.    },
9.    "author": "ridvandani672012078",
10.   "license": "ISC",
11.   "dependencies":{
12.     "Socket.IO":"*",
13.     "express":"*"
14.   }
15. }
```

Kode Program 1 menunjukkan deskripsi mengenai aplikasi yang dibuat. Kemudian melakukan penambahan *dependencies* untuk menambahkan Socket.IO dan Express. Setelah konfigurasi `package.json`, selanjutnya penginstallan Socket.IO dan Express dengan menjalankan perintah `npm install` pada *command line*. Setelah proses instalasi, selanjutnya membuat *server* untuk aplikasi *chatting* yang ditunjukkan pada Kode Program 2.

**Kode Program 2** Konfigurasi Server

```
1.  var express  = require('express');
2.  var app      = express();
3.  var server   = require('http').createServer(app);
4.  var io       = require('Socket.IO').listen(server);
5.  var port     = process.env.PORT || 3001;
6.  var users    = {};
7.  server.listen(port, function(){
8.    console.log("Server bejalan pada port %d", port);
9.  });
```

Kode Program 2 menunjukkan inisialisasi Socket dan *port*. Pada baris 5 diinisialisasikan bahwa aplikasi *chatting* akan berjalan pada *port* 3001. Selanjutnya dilakukan konfigurasi pada *Framework* CodeIgniter agar aplikasi dapat terhubung dengan *database* yang ditunjukkan pada Kode Program 3.

### Kode Program 3 Konfigurasi Koneksi *Database Framework* CodeIgniter

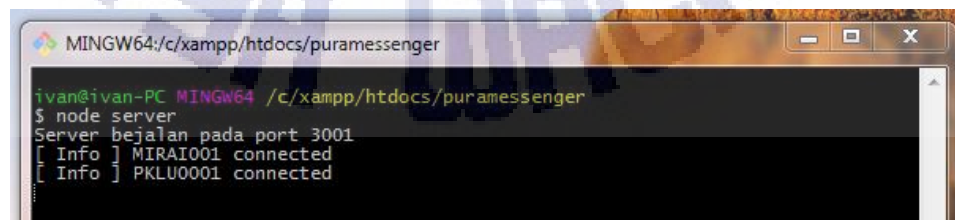
```
1. $db['default']['hostname'] = 'localhost/XE';
2. $db['default']['username'] = 'pura_m';
3. $db['default']['password'] = 'pura_m';
4. $db['default']['database'] = '';
5. $db['default']['dbdriver'] = 'oci8';
6. $db['default']['dbprefix'] = '';
7. $db['default']['pconnect'] = TRUE;
8. $db['default']['db_debug'] = TRUE;
9. $db['default']['cache_on'] = FALSE;
10. $db['default']['cachedir'] = '';
11. $db['default']['char_set'] = 'utf8';
12. $db['default']['dbcollat'] = 'utf8_general_ci';
13. $db['default']['swap_pre'] = '';
14. $db['default']['autoinit'] = TRUE;
15. $db['default']['stricton'] = FALSE;
```

Kode Program 3 menunjukkan konfigurasi *database* yang berisi *hostname*, *username*, *password*, dan *driver*. Pada penelitian ini *database* yang digunakan adalah Oracle, maka *driver* yang digunakan adalah OCI8. Sedangkan untuk pengaturan *route* terlihat pada Kode Program 4.

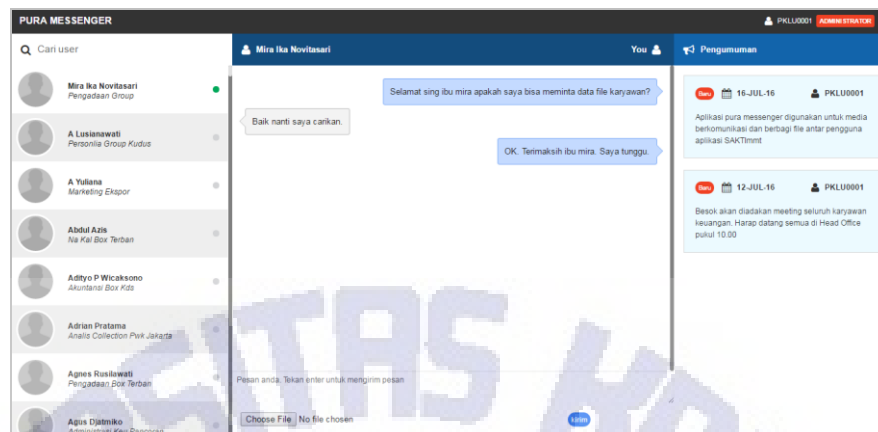
### Kode Program 4 Konfigurasi *Route Framework* CodeIgniter

```
1. $route['default_controller'] = "user_con";
2. $route['404_override'] = '';
3. $route['translate_uri_dashes'] = FALSE;
```

Kode Program 4 menunjukkan *controller* mana yang akan dipanggil pertama kali saat aplikasi dijalankan. Perubahan dilakukan pada baris 1 yaitu saat aplikasi dijalankan akan memanggil kelas *controller* *user\_con* yang mengarah pada halaman *log in*. Pada halaman *log in* terdapat dua *text field* dan satu tombol. *Text field* pertama adalah tempat *user* untuk memasukkan *user id* *user*. *Text field* kedua adalah tempat *user* memasukkan *password*, dan tombol *log in* yang digunakan untuk memproses inputan dari *user*. *Username* dan *password* kemudian diproses untuk dilakukan pengecekan ke *database*. Apabila *username* dan *password* tidak terdapat dalam *database*, akan kembali ke halaman *log in*. Jika cocok akan maka *user* berhasil *log in*. Gambar 7 adalah respon *server* ketika ada *user* yang *log in*. Gambar 8 merupakan halaman yang akan muncul ketika *user* berhasil *log in*.

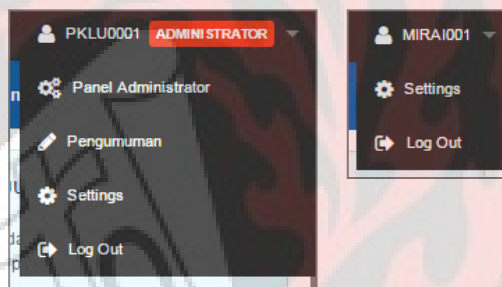


Gambar 7 Respon *Server* Ketika *User Log in*



**Gambar 8** Halaman Utama

Gambar 8 menunjukkan halaman yang ditampilkan ketika *user* berhasil *log in*. Pada bagian *header* sebelah kanan terdapat menu untuk *user*. *User* dibagi menjadi 2 hak akses dengan menu yang berbeda. Hak akses yang pertama adalah administrator dan hak akses yang kedua adalah *user* biasa. Perbedaan menu dapat dilihat pada Gambar 9.



**Gambar 9** Perbedaan Hak Akses Administrator dan *User* Biasa

Gambar 9 menunjukkan perbedaan hak akses pada tiap-tiap *user*. *User* yang mendapat hak akses sebagai administrator terdapat label “administrator” sedangkan untuk *user* yang tidak mendapat hak akses administrator tidak terdapat label “administrator”. *User* yang mendapat hak administrator terdapat menu panel administrator, pengumuman, *settings* dan *log out*. Sedangkan *user* biasa hanya terdapat menu *settings* dan *log out*. Kedua hak akses tersebut bisa melakukan *chatting* dengan sesama *user*. Pada bagian di bawah *header* dibagi lagi menjadi 3 bagian yaitu bagian sebelah kiri untuk menampilkan *user* yang *online* maupun yang sedang *offline*. Bagian tengah untuk melakukan *chatting*, dan bagian kanan untuk menampilkan pengumuman. Untuk menampilkan *user* yang sedang *online* maupun *offline* aplikasi akan mereload secara berkala halaman *userlist*.

#### **Kode Program 5** Menampilkan *User* yang *Online* atau *Offline*

```

1. function cek_userlist() {
2.     $.ajax({
3.         url: "<?php echo base_url('index.php/user_con/userList');?>",
4.         cache: false,
5.         success: function(msg) {
6.             $('#userList').load("<?php echo
7.                 base_url('index.php/user_con/userList');?>");
8.         });
9.         var waktu = setTimeout("cek_userlist()", 10000);
10.     }

```



Kode Program 5 menunjukkan fungsi untuk menampilkan *user* yang *online* maupun *offline* dengan memanfaatkan JQuery ajax, setiap 10000 *millisecond* *class controller userList* akan *direload* secara terus menerus tanpa *merefresh* keseluruhan halaman. Proses *chatting* pada penelitian ini memanfaatkan *library* Javascript Socket.IO. Penggunaan Socket.IO memungkinkan komunikasi secara *realtime* antara *client* dan *server*. Socket.IO berjalan pada *web browser* dan berjalan pada *server*. Kode Program 6 merupakan fungsi pada sisi *client* untuk mengirimkan data dari *client* ke *server*.

#### Kode Program 6 Mengirim Pesan Melalui Socket

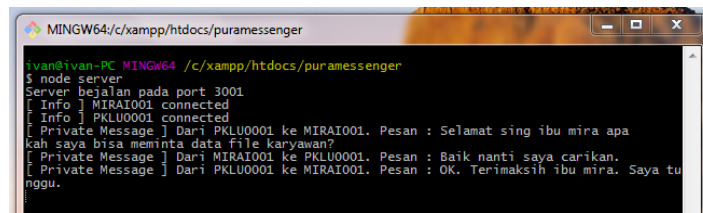
```
1. $('#form-kirim-pesan').submit(function(e) {
2.   e.preventDefault();
3.   var url = $(this).attr('action');
4.   var data = new FormData(this);
5.   var pesan = '/w ' + $inp_tujuan.val() + " " + $inp_pesan.val();
6.   var filename = $('input[type=file]').val().replace(/C:\\fakepath\\/i, '');
7.   var pData = {pesan: pesan, file: filename, gambar: $inp_gambar.val()}
8.
9.   Socket.emit('send message', pData);
10.  ...
11. });
```

Kode Program 6 menunjukkan fungsi untuk mengirimkan data dari *client* ke *server*. Data yang diinputkan akan menjadi parameter pada fungsi '*send message*' pada komputer *client* dan diterima oleh *server*.

#### Kode Program 7 Server Mengolah Data Pesan yang Dikirim

```
1. Socket.on('send message', function(data){
2.   var msg = data.pesan;
3.   if(msg.substring(0,3) === '/w '){
4.     msg = msg.substring(3);
5.     var ind = msg.indexOf(' ');
6.     if(ind !== -1){
7.       var name = msg.substring(0, ind);
8.       var msg = msg.substring(ind + 1);
9.       if(name in users){
10.        users[name].emit('whisper', {user: Socket.user, tujuan: name,
11.        msg: msg, file: data.file, gambar: data.gambar});
12.        console.log('[ Private Message ] Dari '+Socket.user
13.        +' ke '+name+'. Pesan : ' + msg);
14.      }
15.    }
16.    users[Socket.user].emit('kirim', {user: Socket.user, tujuan: name,
17.    msg: msg, file: data.file, gambar: data.gambar});
18.  }else{
19.    io.Sockets.emit('new message', {user: Socket.user, msg: msg,
20.    file: data.file, gambar: data.gambar});
21.  }
22. });
```

Kode Program 7 menunjukkan *server* menerima data yang dikirimkan *client* dengan nama fungsi "*send message*". Data akan diolah dan ditampilkan ke komputer *client*.



**Gambar 10** Respon Server Ketika User Mengirim Pesan

Gambar 10 menunjukkan aktivitas *server*. Pesan yang masuk akan ditampilkan ke dalam *console*. Server mengirim nilai pengembalian data pada *client* dalam fungsi ‘*whisper*’ dan ‘*kirim*’.

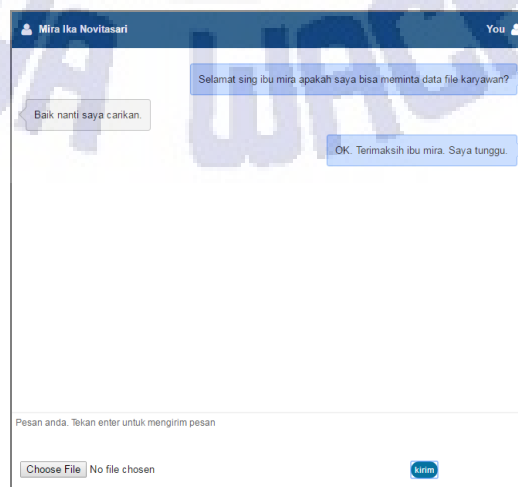
**Kode Program 8** Client Menerima Data yang Dikirim Server

```

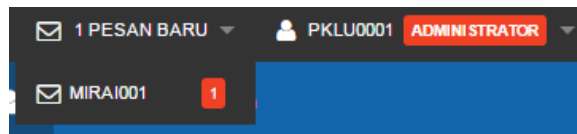
1. Socket.on('whisper', function(data) {
2.   if (data.user == $inp_tujuan.val()) {
3.     if (data.file != '') {
4.       $chat.append("<div class='bubble-receiver'>" + data.msg
5.       + " &nbsp; <a class='button round btn-5'href=\"<?php echo
6.       base_url('/assets/upload/files/' + data.file + '');?>\" >\"
7.       + data.file + "</a></div>");
8.     } else {
9.       $chat.append("<div class='bubble-receiver'>" + data.msg + "</div>");
10.    }
11.  }
12. });
13. Socket.on('kirim', function(data) {
14.   if (data.file != '') {
15.     $chat.append("<div class='bubble-sender'>" + data.msg
16.     + " &nbsp; <a class='button round btn-5'href=\"<?php echo
17.     base_url('/assets/upload/files/' + data.file + '');?>\" >\"
18.     + data.file + "</a></div>");
19.   } else {
20.     $chat.append("<div class='bubble-sender'>" + data.msg + "</div>");
21.   }
22. });

```

Kode Program 8 menunjukkan *client* menerima data percakapan yang dikirimkan oleh *server* dan ditampilkan ke dalam tag *div* pada baris 4 dan baris 9 atau baris 15 dan baris 20. Gambar 11 merupakan hasil data yang ditampilkan di *web browser*. Gambar 12 merupakan notifikasi ketika ada pesan yang belum dibaca.



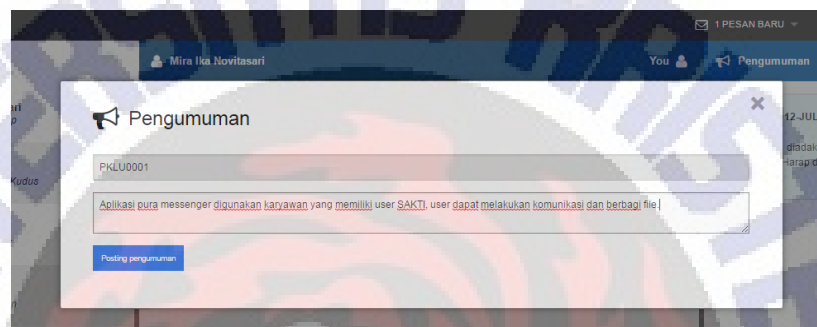
**Gambar 11** Proses Chatting



**Gambar 12** Notifikasi Pesan yang Belum Dibaca

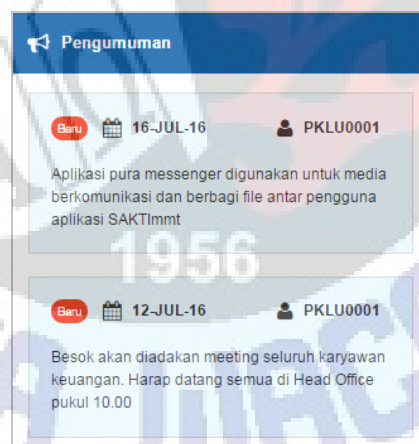
Gambar 12 menunjukkan notifikasi pesan yang belum dibaca. Ketika ada pesan yang baru saja masuk atau pesan yang belum dibaca, pada *header* akan muncul notifikasi yang berisi jumlah pesan dan siapa pengirim pesan tersebut.

*User* dengan hak akses administrator juga bisa melakukan *posting* pengumuman. *Posting* pengumuman juga dilakukan melalui Socket.



**Gambar 13** Form Posting Pengumuman

Gambar 13 menunjukkan *form* untuk menulis pengumuman. Data inputan akan dikirimkan ke *server*. Kemudian *server* mengolah data dan mengirimkan kembali ke *client* untuk ditampilkan yang terlihat pada Gambar 14.



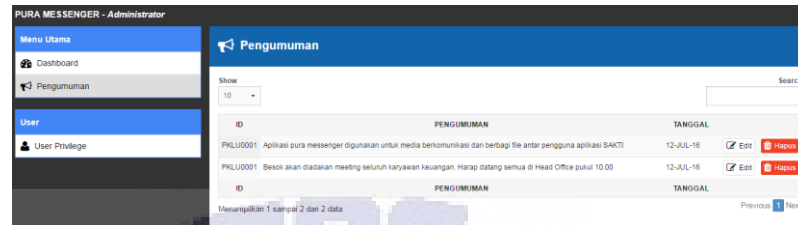
**Gambar 14** Daftar Pengumuman

Pengembangan aplikasi selanjutnya adalah penambahan menu administrator. Menu administrator hanya bisa diakses oleh *user* yang memiliki hak akses administrator. Administrator dapat menghapus atau mengedit pengumuman, dan mengubah hak akses kepada *user* biasa menjadi administrator atau sebaliknya.

Pengumuman yang *diposting* akan masuk ke dalam *database*. Hal pertama yang dilakukan adalah pemanggilan *class controller* *admin\_con* dengan fungsi *pengumuman\_page* kemudian dari *class controller* akan memanggil *class model* pengumuman *model* pada fungsi *oldAnnouncement\_m* kemudian dari *class model*

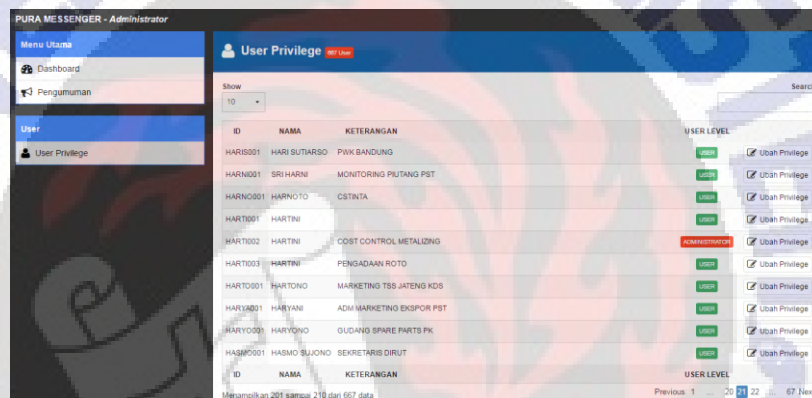


mengembalikan nilai (data) ke *class controller*, kemudian ditampilkan ke *view* *pengumuman\_page\_v*.



**Gambar 15** Daftar Pengumuman pada Halaman Administrator

Gambar 15 menunjukkan aplikasi menampilkan data pengumuman. Selain itu juga bisa mencari, menghapus, mengubah pengumuman dan disertakan *pagination* agar data yang ditampilkan menjadi lebih rapi dan tidak semua data ditampilkan seluruhnya. Administrator juga dapat mengelola data *user*.



**Gambar 16** Halaman untuk Mengelola Hak Akses pada *User*

Gambar 16 menunjukkan menu *user privilege*. Halaman ini berfungsi untuk pengelolaan hak akses yang dimiliki *user*. Adminsitrator dapat mengubah hak akses *user* menjadi administrator atau *user* biasa. Pada halaman ini ditampilkan seluruh data *user* dengan tambahan tombol untuk mengubah hak akses.

Aplikasi *chatting* yang dibangun dapat menyesuaikan dengan resolusi layar ketika aplikasi diakses dengan memanfaatkan *Framework Foundation*. Pemanfaatan *Framework Foundation* aplikasi *web* akan menjadi *responsive*.

#### Kode Program 11 Grid pada *Framework Foundation*

```

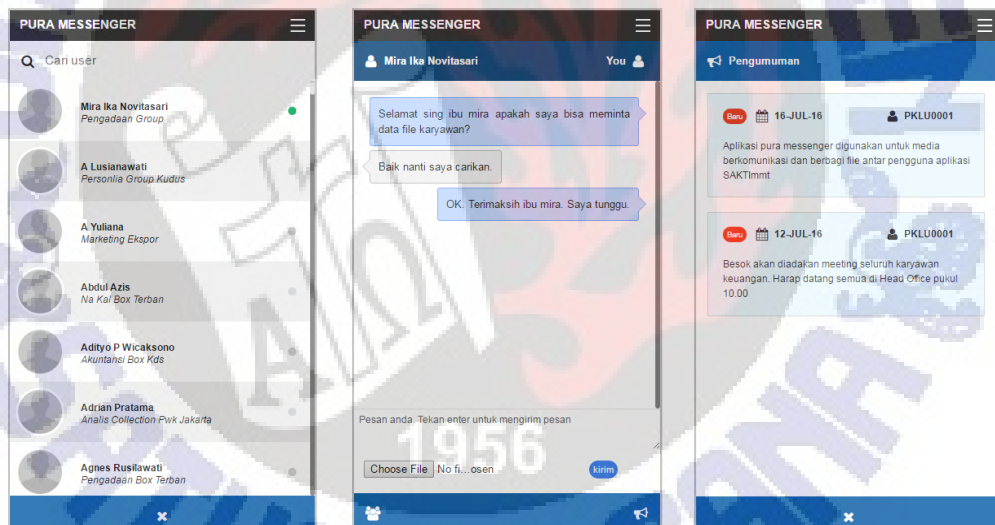
1. <div class="chatContainer medium-6 columns no-padding-left-right white"
2. style="overflow-y : hidden; height:100%;">
3.   <div class="chatWrap blue infobar">
4.     <div class="row clearfix title">
5.       <div class="small-9 medium-6 columns no-padding-left-right">
6.         ...
7.       </div>
8.       <div class="small-3 medium-6 columns no-padding-left-right text-
9.       right">
10.        ...
11.      </div>
12.    </div>
13.  </div>
14.  ...
15. </div>

```

### Kode Program 12 Media Query

```
1. @media (max-width: 700px){
2.   .userListContainer {display:none;}
3.   .announcementContainer {display: none;}
4.   .chatbox{height: 67% !important;}
5. }
6. @media (min-width: 700px) {
7.   .menuMobile{display: none;}
8.   .tab-bar {display: none;}
9.   ...
10. }
```

Kode Program 11 menunjukkan penggunaan *grid* pada *Framework Foundation*. Maksimum penggunaan *grid* pada *Framework Foundation* adalah 12 kolom. *Grid* dikelompokkan menjadi 3 properti yaitu *small*, *medium*, *large*. Properti *small* digunakan untuk menyesuaikan resolusi layar yang kecil, properti *medium* digunakan untuk menyesuaikan resolusi layar yang lebih besar dari properti *small* dan properti *large* untuk resolusi layar yang lebih besar lagi. Kode Program 12 menunjukkan pengaturan *media query* yang digunakan untuk menampilkan *layout* yang berbeda berdasarkan media yang digunakan.



Gambar 17 Tampilan Responsive

Gambar 17 menunjukkan tampilan aplikasi ketika diakses pada resolusi layar yang lebih kecil. Dilakukan penambahan menu pada sisi bawah layar untuk mengakses menu melihat daftar *user* dan melihat pengumuman.

Pengujian sistem dilakukan untuk mencari kesalahan pada aplikasi *chatting* yang dibuat. Pengujian dilakukan untuk mengetahui apakah aplikasi yang sudah dibuat berjalan dengan baik dan sesuai dengan kebutuhan *user*. Pengujian alpha menggunakan metode *blackbox* yaitu pengujian fungsi-fungsi aplikasi secara langsung tanpa memperhatikan alur eksekusi program. Pengujian ini dilakukan dengan memperhatikan apakah fungsi telah berjalan sesuai rancangan dan sesuai yang diharapkan.

**Table 1** Hasil Pengujian *BlackBox*

Fungsi yang diuji	Kondisi	Output yang diharapkan	Output yang dihasilkan sistem	Status Pengujian
Log in	Username dan password benar	Sukses log in	Sukses log in	Valid
	Username dan password salah maupun kosong	Gagal log in	Gagal log in	
Implementasi Socket.IO untuk pengiriman pesan	Server hidup. User mengirim pesan dengan <i>attachment file</i> atau tanpa <i>attachment file</i> .	Sukses mengirim pesan. Pesan diterima <i>client</i> secara <i>realtime</i>	Sukses mengirim pesan. Pesan diterima <i>client</i> secara <i>realtime</i>	Valid
Posting pengumuman	Form diisi dengan benar	Sukses posting pengumuman	Sukses posting pengumuman	Valid
Hapus pengumuman	Pilih salah satu pengumuman	Sukses hapus pengumuman	Sukses hapus pengumuman	Valid
Edit pengumuman	Form diisi dengan benar	Sukses ubah pengumuman	Sukses ubah pengumuman	Valid
Load data pengumuman	Buka halaman pengumuman	Sukses load data	Sukses load data	Valid
Ubah hak akses user	Pilih salah satu user yang ingin diubah hak aksesnya	Sukses mengubah hak akses	Sukses mengubah hak akses	Valid
Responsivitas aplikasi dengan resolusi layar yang digunakan.	Mengakses aplikasi dengan resolusi layar yang lebih kecil	Aplikasi menyesuaikan dengan resolusi layar yang digunakan	Aplikasi menyesuaikan dengan resolusi layar yang digunakan	Valid

Berdasarkan pengujian yang dilakukan pada aplikasi *chatting* dapat dilihat status pengujian dari setiap fungsi *valid*, maka disimpulkan bahwa aplikasi ini berfungsi dengan baik dan sesuai yang diharapkan.

Pengujian beta dilakukan dengan melakukan presentasi dengan EDP Keuangan. Presentasi dilakukan dengan melakukan demo program yang telah diupload ke server untuk mengetahui apakah aplikasi *chatting* berfungsi dengan baik dan sesuai kebutuhan pengguna. Berdasarkan hasil presentasi dengan karyawan EDP Keuangan, aplikasi berfungsi dengan baik dan sesuai dengan kebutuhan pengguna.

## 5. Simpulan

Berdasarkan penelitian yang telah dilakukan maka dapat diambil kesimpulan bahwa pembuatan aplikasi *chatting* dapat dibuat dengan menggunakan *Framework* CodeIgniter, Socket.IO dan *Framework* Foundation. Pemanfaatan *Framework* CodeIgniter dapat diterapkan konsep MVC (*Model View Controller*) sehingga penulisan kode menjadi lebih terstruktur dan terorganisir. Pemanfaatan Socket.IO cocok untuk pembuatan aplikasi *real time* seperti aplikasi *chatting* dan *Framework* Foundation membuat tampilan aplikasi menjadi *responsive* yang dapat menyesuaikan diberbagai resolusi layar.

Berdasarkan hasil pengujian, aplikasi *chatting* dapat membantu karyawan EDP Keuangan atau user lainnya untuk berkomunikasi, *attachment file*, dan menyebarkan informasi jika sewaktu-waktu terjadi penambahan atau perubahan program yang dikembangkan oleh karyawan EDP Keuangan. Spesifikasi *hardware* maupun *software* aplikasi *chatting* ini tergolong aplikasi yang ringan karena merupakan aplikasi berbasis *web*, user hanya membutuhkan *web browser* kemudian mengakses aplikasi *chatting* yang sudah diupload ke server dan dengan memanfaatkan jaringan lokal maka untuk mengakses aplikasi *chatting* tidak membutuhkan koneksi *internet*.



## 6. Pustaka

- [1] M. Z. Teddy, & D. W. Surya. 2009. Aplikasi Chat pada Handphone dan Komputer dengan Media Bluetooth (Bluetooth Chat). *Jurnal Teknologi Informasi-Aiti*, VI (1): pp.60-74.
- [2] S. Roni, & S. Edhy. 2009. Membangun Aplikasi Chatting Berbasis Multiuser. *Jurnal Dasi*, X(1):pp.3-22.
- [3] Sularso, E, & Raharjo. W. S, & Lukito. Y. 2014. Implementasi Algoritma RijnDeal 128 pada Aplikasi Chatting Berbasis HTML WebSocket. *Jurnal Informatika*, X(2):pp.66-79.
- [4] Priyanto, Duwi. 2009. *Mahir Komputer Tanpa Kursus. Belajar Mudah Internet*. Yogyakarta : MediaKom.
- [5] Wardana. 2010. *Menjadi Master PHP dengan Framework CodeIgniter*. Elex Media Komputindo.
- [6] Nodesource, <https://nodesource.com/blog/understanding-socketio/>. Diakses pada tanggal 6 Juli 2016
- [7] Foundation Zurb, <http://foundation.zurb.com/learn/why-foundation.html>. Diakses pada 6 Juli 2016.
- [8] Nugroho, Adi. 2010. *Mengembangkan Aplikasi Basis Data Menggunakan C# dan SQL Server*. Yogyakarta : Andi.
- [9] Hasibuan, Zainal A. 2007. *Metodologi Penelitian Pada Bidang Ilmu Komputer dan Teknologi Informasi : Konsep, Teknik, dan Aplikasi*. Jakarta : Ilmu Komputer Univesitas Indonesia.
- [10] Pressman, R.S, 2001, *Software Engineering : A Practitioner's Approach*, Amerika Serikat : R.S. Pressman and Associates.