BAB V

IMPLEMENTASI DAN PENGUJIAN SISTEM

5.1. Implementasi

Impelementasi merupakan akhir dari sebuah pengembangan sistem informasi. Implementasi sistem tentunya mengacu pada rancangan sistem yang telah dibuat. Pada umumnya untuk implementasi sebuah sistem informasi diperlukan konfigurasi pada perangkat lunak serta perangkat keras sesuai kebutuhan sistem supaya sistem informasi bisa berjalan dengan semestinya.

5.1.1. Implementasi Perangkat Lunak

Pada tahap implementasi perangkat lunak ini, didefinisikan perangkat lunak lainnya yang dijadikan pendukung bagi *Tracking of Using Software Licensed System* dan *Helpdesk System*. Perangkat lunak tersebut diantaranya:

1. Untuk sistem operasi disarankan menggunakan *Microsoft Windows* minimum *Windows XP Service Pack 2*. Bisa juga menggunakan sistem operasi lain.
2. Minimum memerlukan *Java Runtime Environment* (JRE) versi 5. Ini dibutuhkan untuk menjalankan *class java* yang ada dalam sistem.
3. Aplikasi basis data yang dipakai dalam sistem ini adalah *MySQL*.

4. Aplikasi *web server* yang dipakai dalam sistem ini adalah *Apache Tomcat 6.0*. Ini diperlukan untuk menjalankan program berbasis web dengan bahasa pemrograman *Java Enterprise Edition (J2EE)*. Sebenarnya bisa saja menggunakan aplikasi *web server* yang lain seperti *Glassfish,* atau *JBoss.* Namun *Apache Tomcat 6.0* dirasa sudah memenuhi kebutuhan sistem karena di dalam sistem tidak sampai ada pembuatan modul *Enterprise Java Bean* (EJB).
5. Untuk *web browser* disarankan minimum menggunakan *Mozilla Firefox 3.5*.

5.1.2. Implementasi Perangkat Keras

Mengenai implementasi perangkat keras, dalam tahap ini didefinisikan sekumpulan perangkat keras untuk mendukung *Tracking of Using Software Licensed System* dan *Helpdesk System* supaya berjalan dengan semestinya. Pada dasarnya tidak ada perangkat keras khusus yang dibutuhkan sistem. Konfigurasi dari perangkat keras tersebut, diantaranya:

1. *Server-side*

   Kebutuhan perangkat keras dari sisi komputer *server* sendiri dengan jumlah *client sekitar 500 client*, yaitu:

   a. *Processor* Intel Itanium2
   b. RAM jenis ECC *Registered*/FBDIMM dengan kapasitas minimum 8GB
   c. *Hard Disk* jenis SCSI atau SAS dengan kapasitas minimum 1TB.

2. *Client-side*

   Sedangkan kebutuhan perangkat keras untuk komputer *client* tergolong standar, yaitu:

   a. *Processor* dengan frekuensi minimum 1.66Ghz, dan *cache* 512KB.

   b. RAM jenis DDR dengan kapasitas minimum 512MB

   c. *Hard Disk* jenis SATA atau PATA dengan kapasitas minimum 120GB.

5.1.3. Implementasi Basis Data (Sintaks SQL)

Pada bagian implementasi basis data akan diuraikan seperti apa sintaksnya hasil dari rancangan basis data yang terlihat pada Gambar 4.9. Nama basis data yang dibuat untuk sistem ini adalah **"skripsi"**.

1. Sintaks pembuatan basis data

```
CREATE DATABASE skripsi;
```

2. Sintaks untuk tabel "*products*"

```
CREATE TABLE `products` (
  `part_num` varchar(25) NOT NULL,
  `serial_num` varchar(30) NOT NULL,
  `prod_desc` varchar(200) NOT NULL,
  `version` varchar(20) NOT NULL,
  PRIMARY KEY  (`part_num`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

3. Sintaks untuk tabel "*vendor*"

```
CREATE TABLE `vendor` (
 `id_vendor` varchar(10) NOT NULL,
 `company_name` varchar(200) NOT NULL,
 `phone` varchar(17) NOT NULL,
 `contact_person` varchar(30) NOT NULL,
 PRIMARY KEY (`id_vendor`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

4. Sintaks untuk tabel "*master_product*"

```
CREATE TABLE `master_product` (
 `author_num` varchar(30) NOT NULL,
 `license_num` varchar(30) NOT NULL,
 `order_num` varchar(30) NOT NULL,
 `license_date` date NOT NULL,
 `reorder` date NOT NULL,
 `agreement` varchar(25) NOT NULL,
 `id_vendor` varchar(10) NOT NULL,
 PRIMARY KEY (`author_num`),
 KEY `id_vendor` (`id_vendor`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
ALTER TABLE `master_product`
 ADD CONSTRAINT `master_product_ibfk_1` FOREIGN KEY (`id_vendor`) REFERENCES
`vendor` (`id_vendor`) ON UPDATE CASCADE;
```

5. Sintaks untuk tabel “unit_kerja”

```
CREATE TABLE `unit_kerja` (
 `id_unit` varchar(10) NOT NULL,
 `unit_name` varchar(30) NOT NULL,
 `biro` varchar(50) NOT NULL,
 `direktorat` varchar(30) NOT NULL,
 `kompartemen` varchar(30) NOT NULL,
 PRIMARY KEY (`id_unit`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

6. Sintaks untuk tabel “karyawan”

```
CREATE TABLE `karyawan` (
 `no_badge` varchar(10) NOT NULL,
 `nama` varchar(50) default NULL,
 `id_unit` varchar(10) default NULL,
 PRIMARY KEY (`no_badge`),
 KEY `id_unit` (`id_unit`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
ALTER TABLE `karyawan`
 ADD CONSTRAINT `karyawan_ibfk_1` FOREIGN KEY (`id_unit`) REFERENCES
`unit_kerja` (`id_unit`) ON UPDATE CASCADE;
```

7. Sintaks untuk tabel "*categorization*"

```
CREATE TABLE `categorization` (
 `id_category` varchar(5) NOT NULL,
 `category` varchar(50) NOT NULL,
 PRIMARY KEY (`id_category`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

8. Sintaks untuk tabel "*posting*"

```
CREATE TABLE `posting` (
 `category` varchar(5) NOT NULL,
 `id_post` varchar(5) NOT NULL,
 `title` varchar(200) NOT NULL,
 `content` text,
 `author` varchar(16) NOT NULL,
 `tanggal` datetime NOT NULL,
 PRIMARY KEY (`id_post`),
 KEY `category` (`category`),
 KEY `author` (`author`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
ALTER TABLE `posting`
 ADD CONSTRAINT `posting_ibfk_2` FOREIGN KEY (`category`) REFERENCES
`categorization` (`id_category`) ON UPDATE CASCADE,
 ADD CONSTRAINT `posting_ibfk_3` FOREIGN KEY (`author`) REFERENCES
`user_account` (`username`) ON UPDATE CASCADE;
```

9. Sintaks untuk tabel “komentar”

```
CREATE TABLE `komentar` (
 `id_komen` varchar(5) NOT NULL,
 `id_post` varchar(5) NOT NULL,
 `komentar` text,
 `author` varchar(16) NOT NULL,
 `tanggal` datetime default NULL,
 PRIMARY KEY (`id_komen`),
 KEY `id_post` (`id_post`),
 KEY `author` (`author`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
ALTER TABLE `komentar`
 ADD CONSTRAINT `komentar_ibfk_1` FOREIGN KEY (`id_post`) REFERENCES `posting`
(`id_post`) ON UPDATE CASCADE,
 ADD CONSTRAINT `komentar_ibfk_2` FOREIGN KEY (`author`) REFERENCES
`user_account` (`username`) ON UPDATE CASCADE;
```

10. Sintaks untuk tabel “*inbox*”

```
CREATE TABLE `inbox` (
 `id_ticket` varchar(5) NOT NULL,
 `title` varchar(200) NOT NULL,
 `content` text,
 `tanggal` datetime NOT NULL,
 `sender` varchar(16) NOT NULL,
```

```
 `ket` varchar(6) NOT NULL,
 PRIMARY KEY (`id_ticket`),
 KEY `sender` (`sender`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
ALTER TABLE `inbox`
 ADD CONSTRAINT `inbox_ibfk_1` FOREIGN KEY (`sender`) REFERENCES
`user_account` (`username`) ON UPDATE CASCADE;
```

11. Sintaks untuk tabel "*outbox*"

```
CREATE TABLE `outbox` (
 `id_ticket` varchar(5) NOT NULL,
 `reply` text NOT NULL,
 `tanggal` datetime NOT NULL,
 `receiver` varchar(16) NOT NULL,
 `ket` varchar(6) NOT NULL,
 KEY `id_ticket` (`id_ticket`),
 KEY `receiver` (`receiver`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
ALTER TABLE `outbox`
 ADD CONSTRAINT `outbox_ibfk_2` FOREIGN KEY (`id_ticket`) REFERENCES `inbox`
(`id_ticket`) ON UPDATE CASCADE,
 ADD CONSTRAINT `outbox_ibfk_3` FOREIGN KEY (`receiver`) REFERENCES
`user_account` (`username`) ON UPDATE CASCADE;
```

12. Sintaks untuk tabel "*user_account*"

```
CREATE TABLE `user_account` (
 `username` varchar(16) NOT NULL,
 `pwd` varchar(16) NOT NULL,
 `no_badge` varchar(10) NOT NULL,
 `ket` varchar(6) NOT NULL,
 PRIMARY KEY (`username`),
 KEY `no_badge` (`no_badge`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
ALTER TABLE `user_account`
 ADD CONSTRAINT `user_account_ibfk_1` FOREIGN KEY (`no_badge`) REFERENCES
`karyawan` (`no_badge`) ON UPDATE CASCADE;
```

13. Sintaks untuk tabel "*user_product*"

```
CREATE TABLE `user_product` (
 `computer_name` varchar(25) NOT NULL,
 `ip_address` varchar(15) NOT NULL,
 `workgroup` varchar(15) default NULL,
 `id_unit` varchar(10) default NULL,
 PRIMARY KEY (`computer_name`),
 UNIQUE KEY `ip_address` (`ip_address`),
 KEY `id_unit` (`id_unit`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
ALTER TABLE `user_product`
```

```
 ADD CONSTRAINT `user_product_ibfk_1` FOREIGN KEY (`id_unit`) REFERENCES
`unit_kerja` (`id_unit`) ON UPDATE CASCADE;
```

14. Sintaks untuk tabel "*usage_mapped*"

```
CREATE TABLE `usage_mapped` (
 `computer_name` varchar(25) NOT NULL,
 `part_num` varchar(25) NOT NULL,
 `ket` varchar(10) NOT NULL,
 KEY `computer_name` (`computer_name`),
 KEY `part_num` (`part_num`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
ALTER TABLE `usage_mapped`
 ADD CONSTRAINT `usage_mapped_ibfk_1` FOREIGN KEY (`computer_name`)
REFERENCES `user_product` (`computer_name`) ON UPDATE CASCADE,
 ADD CONSTRAINT `usage_mapped_ibfk_2` FOREIGN KEY (`part_num`) REFERENCES
`products` (`part_num`) ON UPDATE CASCADE;
```

15. Sintaks untuk tabel "*olp_transaction*"

```
CREATE TABLE `olp_transaction` (
 `id_OLP` varchar(7) NOT NULL,
 `downgrade` varchar(7) default NULL,
 `computer_name` varchar(25) NOT NULL,
 `part_num` varchar(25) NOT NULL,
 PRIMARY KEY (`id_OLP`),
```

```
  KEY `computer_name` (`computer_name`),
  KEY `part_num` (`part_num`),
  KEY `downgrade` (`downgrade`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
ALTER TABLE `olp_transaction`
  ADD CONSTRAINT `olp_transaction_ibfk_3` FOREIGN KEY (`part_num`) REFERENCES
`products` (`part_num`) ON UPDATE CASCADE,
  ADD CONSTRAINT `olp_transaction_ibfk_1` FOREIGN KEY (`downgrade`) REFERENCES
`downgrade` (`id_downgrade`) ON UPDATE CASCADE,
  ADD CONSTRAINT `olp_transaction_ibfk_2` FOREIGN KEY (`computer_name`)
REFERENCES `user_product` (`computer_name`) ON UPDATE CASCADE;
```

16. Sintaks untuk tabel "*oem_transaction*"

```
CREATE TABLE `oem_transaction` (
  `id_OEM` varchar(7) NOT NULL,
  `productid` varchar(30) NOT NULL,
  `prod_desc` varchar(200) NOT NULL,
  `computer_name` varchar(25) NOT NULL,
  PRIMARY KEY (`id_OEM`),
  UNIQUE KEY `productid` (`productid`),
  KEY `computer_name` (`computer_name`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
ALTER TABLE `oem_transaction`
```

```
 ADD CONSTRAINT `oem_transaction_ibfk_1` FOREIGN KEY (`computer_name`)
REFERENCES `user_product` (`computer_name`) ON UPDATE CASCADE;
ADD CONSTRAINT `oem_transaction_ibfk_2` FOREIGN KEY (`productid`) REFERENCES
`products` (`part_num`) ON UPDATE CASCADE;
```

17. Sintakas untuk tabel "*downgrade*"

```
CREATE TABLE `downgrade` (
 `id_downgrade` varchar(7) NOT NULL,
 `computer_name` varchar(25) NOT NULL,
 `part_num` varchar(25) NOT NULL,
 PRIMARY KEY (`id_downgrade`),
 KEY `computer_name` (`computer_name`),
 KEY `part_num` (`part_num`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
ALTER TABLE `downgrade`
 ADD CONSTRAINT `downgrade_ibfk_1` FOREIGN KEY (`computer_name`) REFERENCES
`user_product` (`computer_name`) ON UPDATE CASCADE,
 ADD CONSTRAINT `downgrade_ibfk_2` FOREIGN KEY (`part_num`) REFERENCES
`products` (`part_num`) ON UPDATE CASCADE;
```

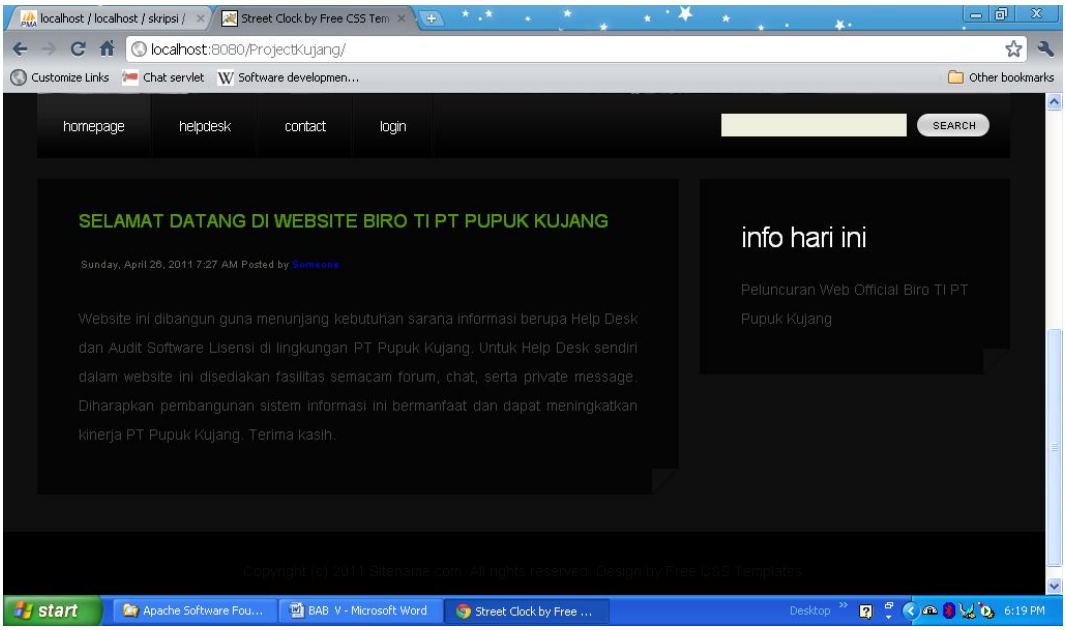
18. Sintaks untuk tabel "bajakan"

```
CREATE TABLE `bajakan` (
 `id_bajakan` varchar(7) NOT NULL,
 `computer_name` varchar(25) NOT NULL,
```

```
  `productid` varchar(25) NOT NULL,
  `prod_desc` varchar(200) NOT NULL,
  PRIMARY KEY  (`id_bajakan`),
  KEY `computer_name` (`computer_name`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
ALTER TABLE `bajakan`
  ADD CONSTRAINT `bajakan_ibfk_1` FOREIGN KEY (`computer_name`) REFERENCES
`user_product` (`computer_name`) ON UPDATE CASCADE;
```

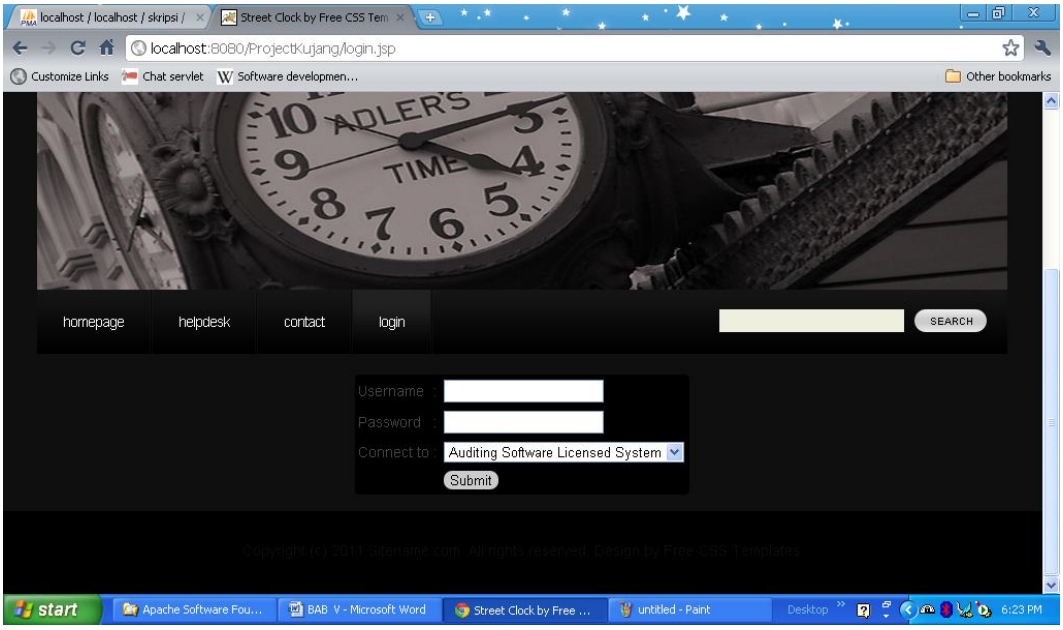
5.1.4. Implementasi Antar Muka

Pada tahap ini akan digambarkan model antar muka untuk pengoperasian sistem yang dapat dilihat langsung oleh *user*. Rincian dari antar muka ini adalah sebagai berikut.



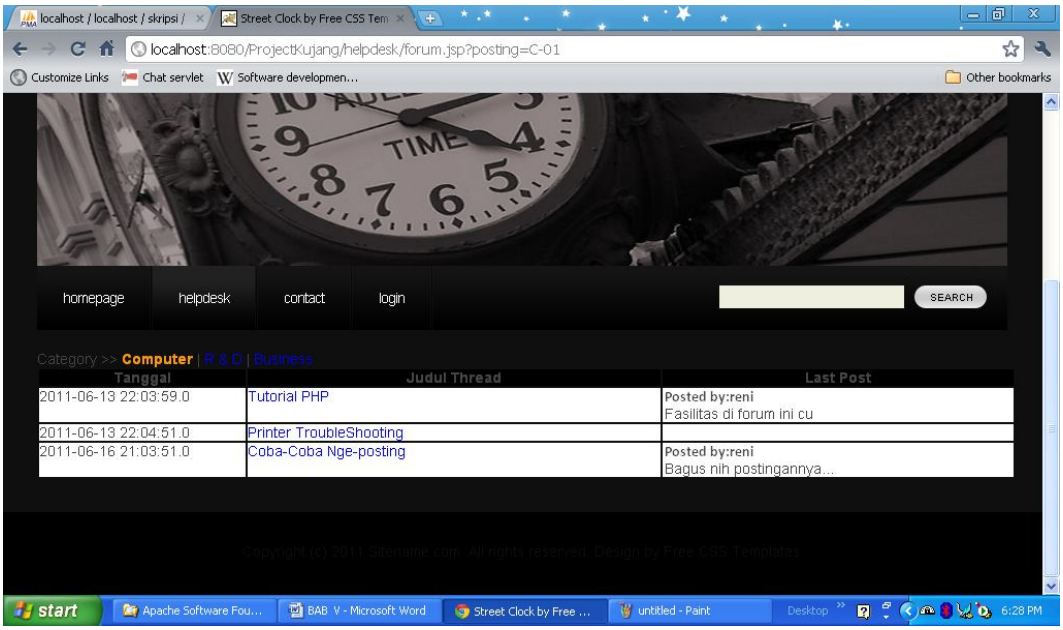
Gambar 5.1 Halaman index.html

Gambar 5.1 menunjukan halaman awal dari *Tracking of Using Software Licensed System* dan *Helpdesk System* yang berbasis web.



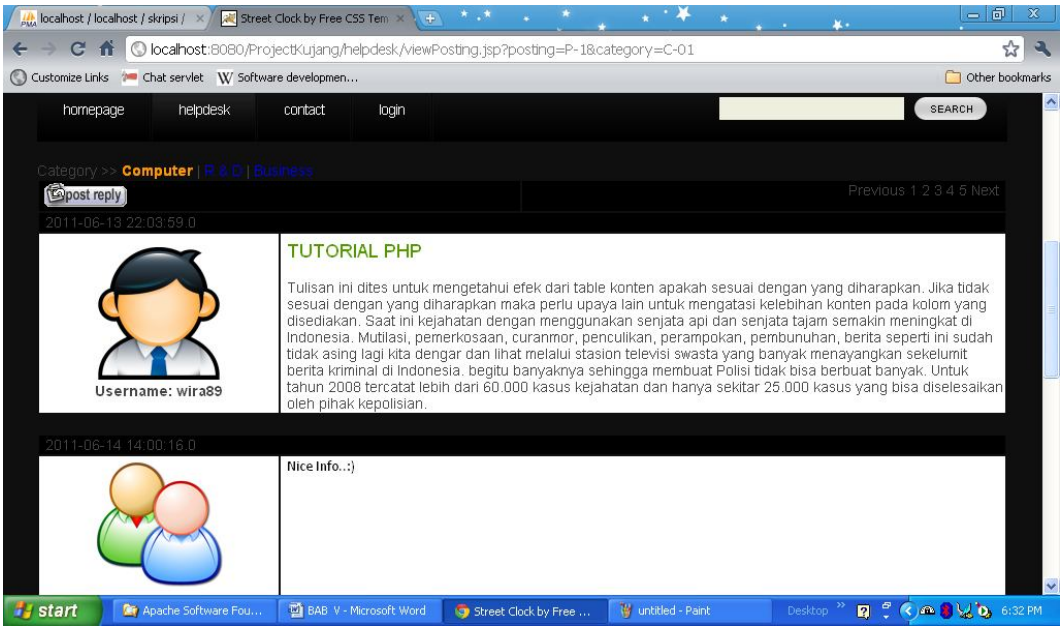
Gambar 5.2 Halaman login.jsp

Gambar 5.2 menunjukan form login untuk bisa memasuki sistem. Ada dua pilihan pada form login ini, yaitu apakah mau masuk ke *Tracking of Using Software Licensed System* atau masuk ke *Helpdesk System*. Yang bisa masuk ke *Tracking of Using Software Licensed System* hanyalah *administrator*.



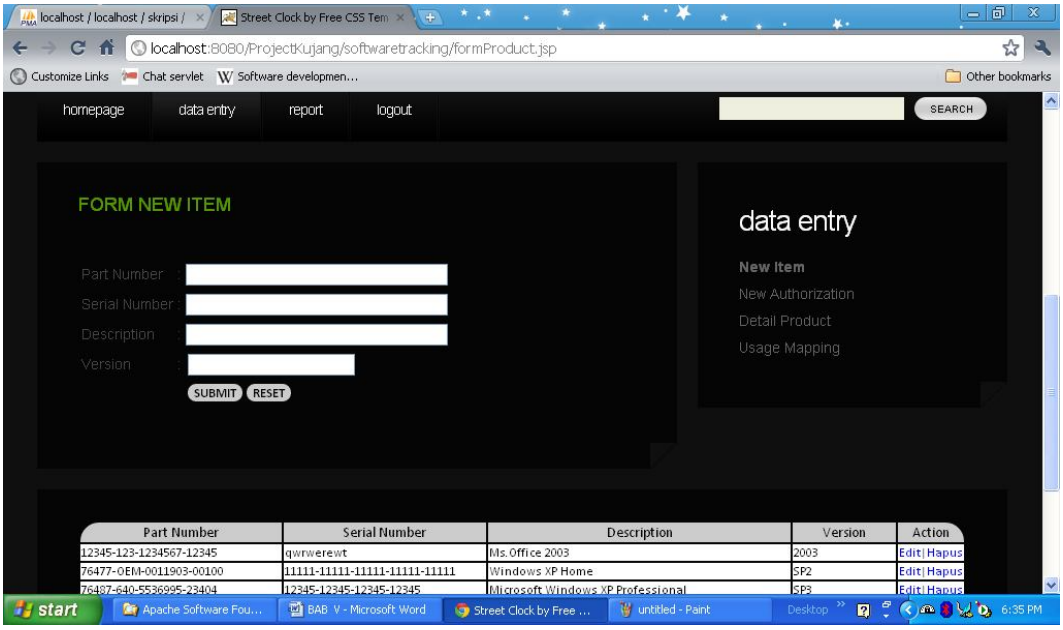
Gambar 5.3 Halaman forum.jsp

Gambar 5.3 merupakan tampilan yang berisi informasi daftar *knowledge* yang tersedia.



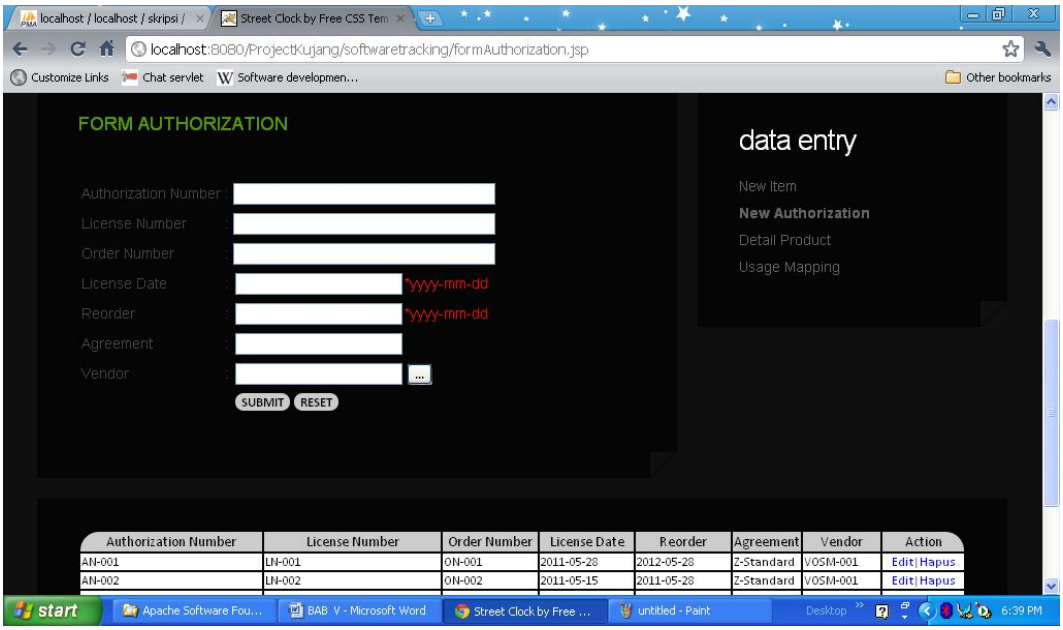
Gambar 5.4 Halaman viewPosting.jsp

Gambar 5.4 merupakan uraian atau isi dari *knowledge* yang dipilih pada daftar *knowledge* yang ada di halaman **forum.jsp**.



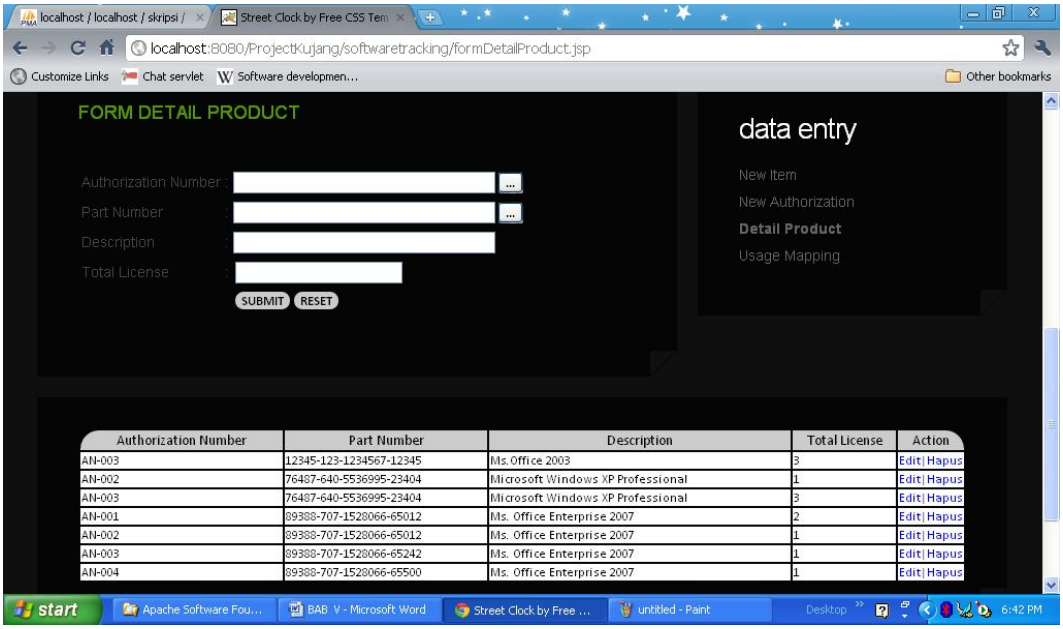
Gambar 5.5 Halaman formProduct.jsp

Gambar 5.5 merupakan model form masukan data untuk data produk *software* berlisensi dan di bawahnya terdapat daftar produk *software* berlisensi yang tercatat di dalam sistem.



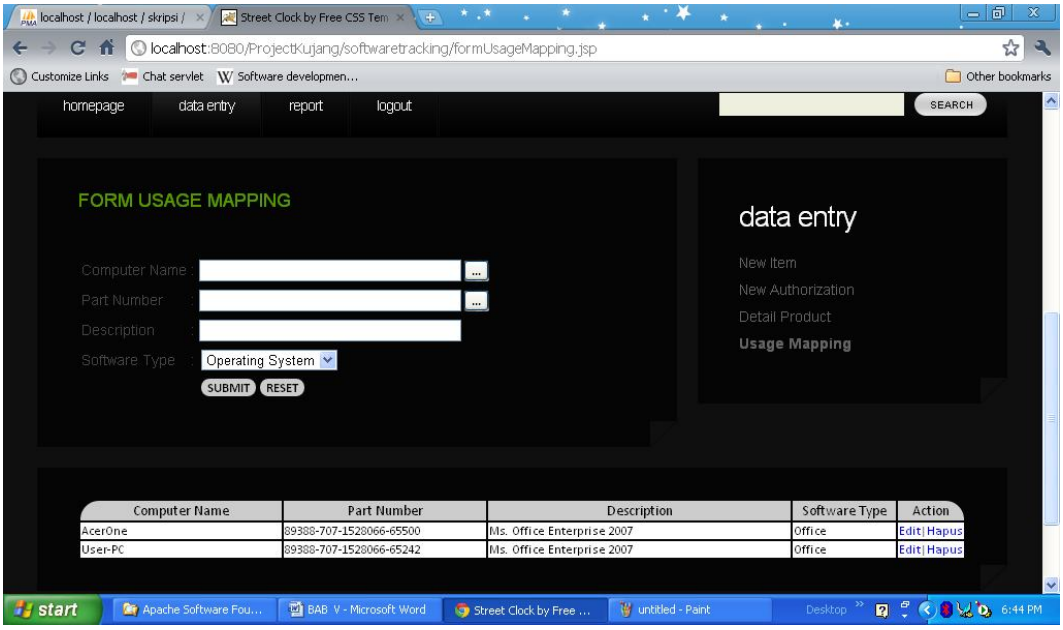
Gambar 5.6 Halaman formAuthorization.jsp

Gambar 5.6 merupakan form masukan data untuk detail otorisasi dari *software* berlisensi.



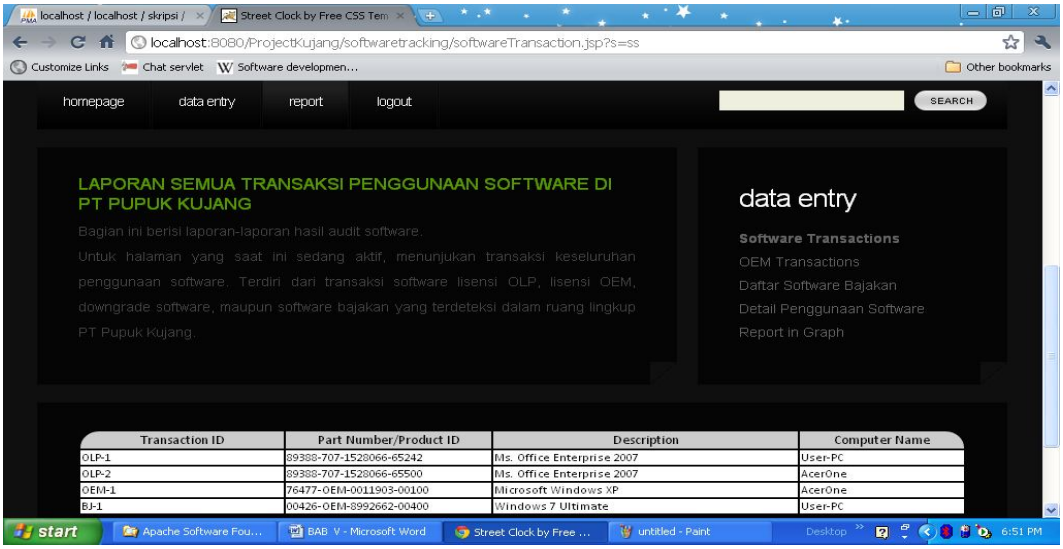
Gambar 5.7 Halaman formDetailProduct.jsp

Gambar 5.7 menggambarkan rincian dari setiap produk *software* berlisensi.



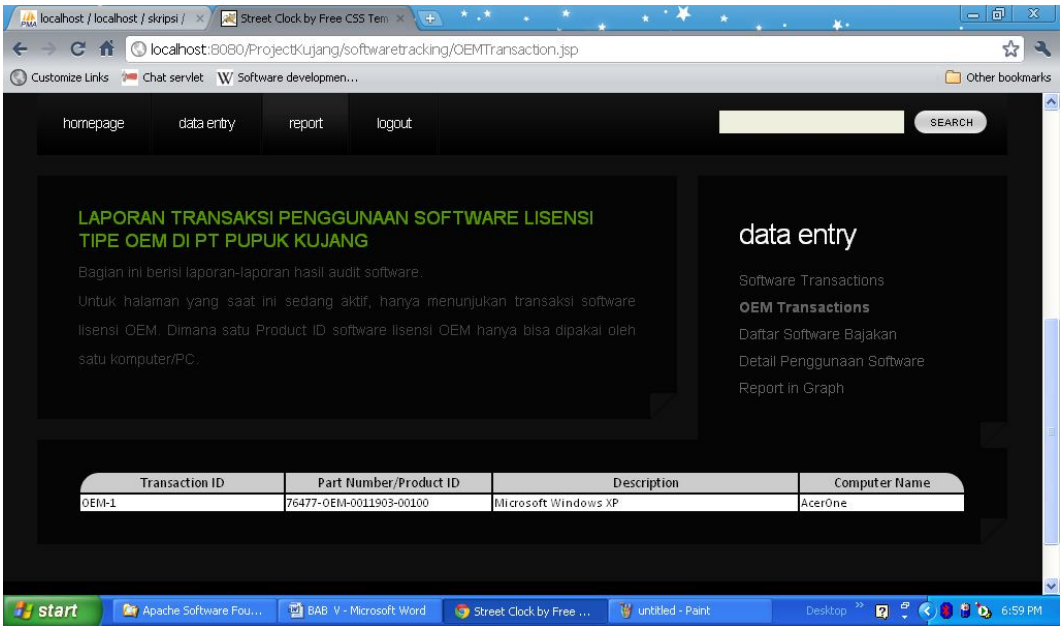
Gambar 5.8 Halaman formUsageMapping.jsp

Gambar 5.8 adalah model masukan data untuk pemetaan penggunaan *software* berlisensi jenis OLP.



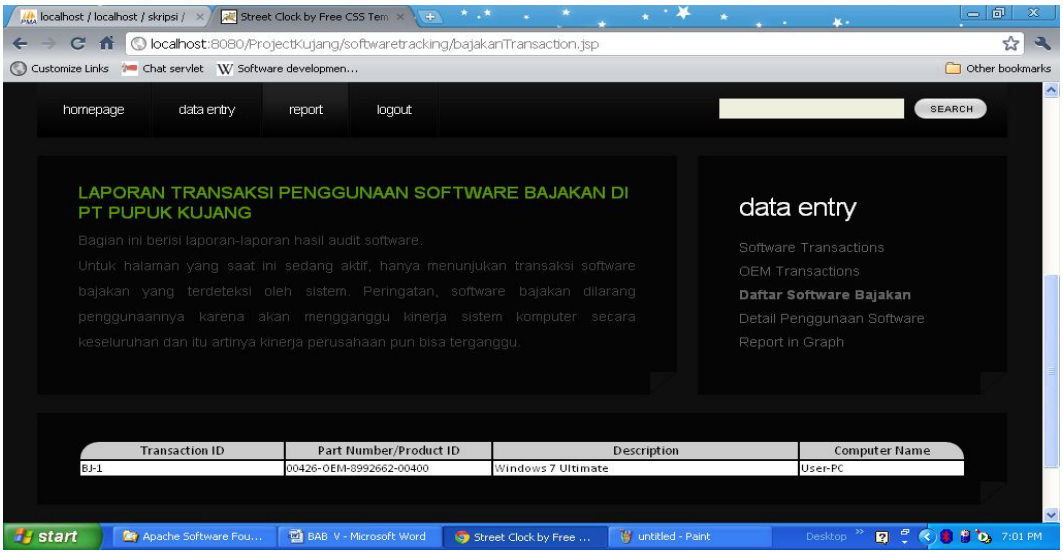
Gambar 5.9 Halaman softwareTransaction.jsp

Gambar 5.9 menampilkan informasi hasil audit *software* berlisensi secara kesuluruhan, yaitu menampilkan informasi penggunaan *software* jenis OLP, OEM, informasi kasus *downgrade*, informasi terdapatnya *software* bajakan.



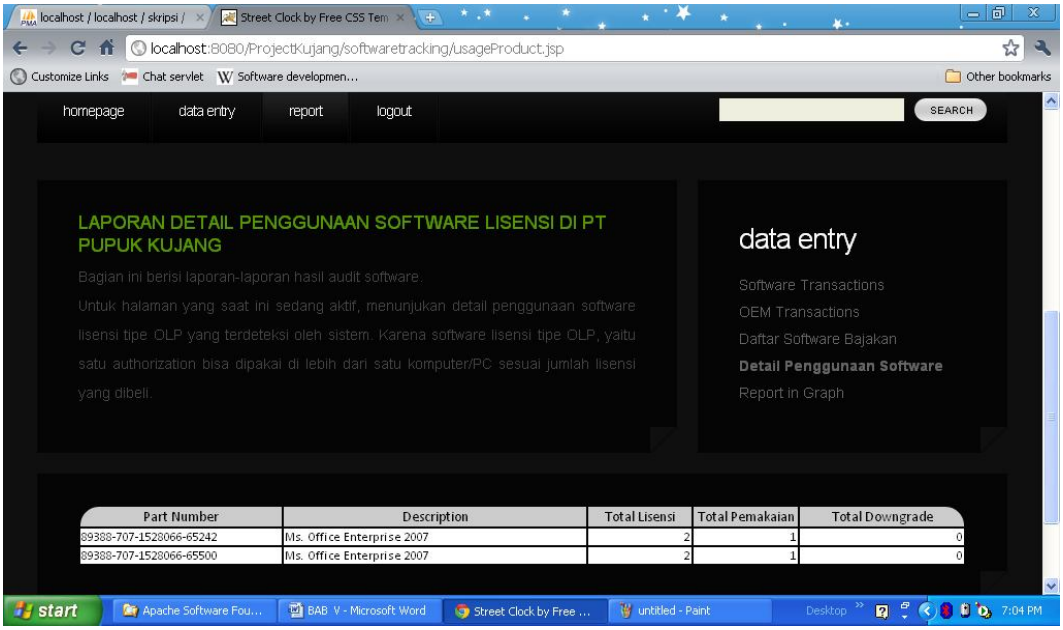
Gambar 5.10 Halaman OEMTransaction.jsp

Gambar 5.10 hanya menampilkan informasi penggunaan *software* jenis OEM.



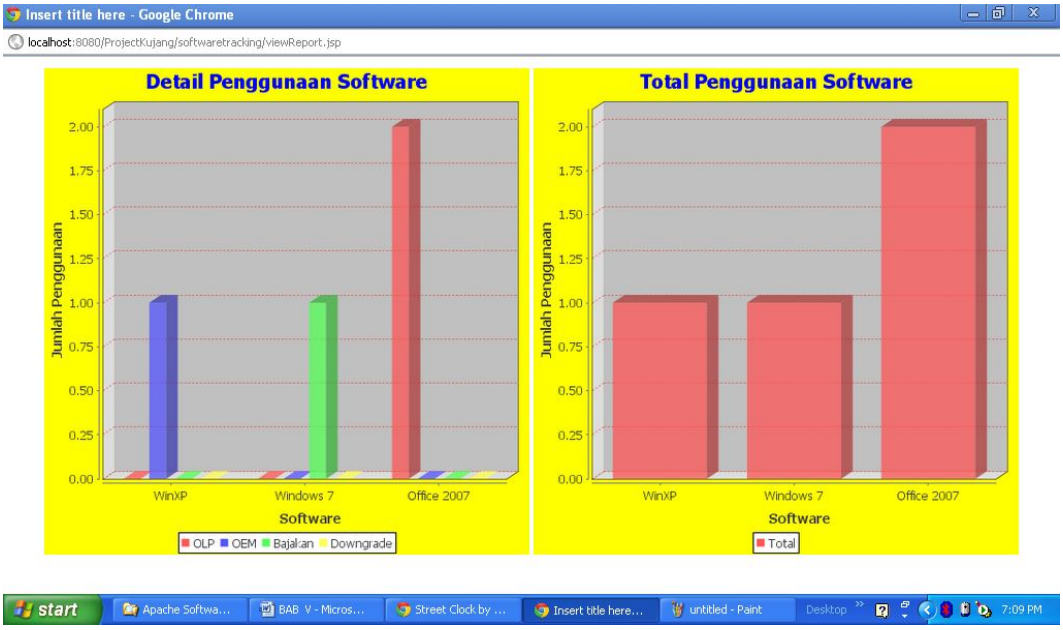
Gambar 5.11 Halaman bajakanTransaction.jsp

Gambar 5.11 hanya menampilkan informasi jika terdeteksi penggunaan *software* bajakan.



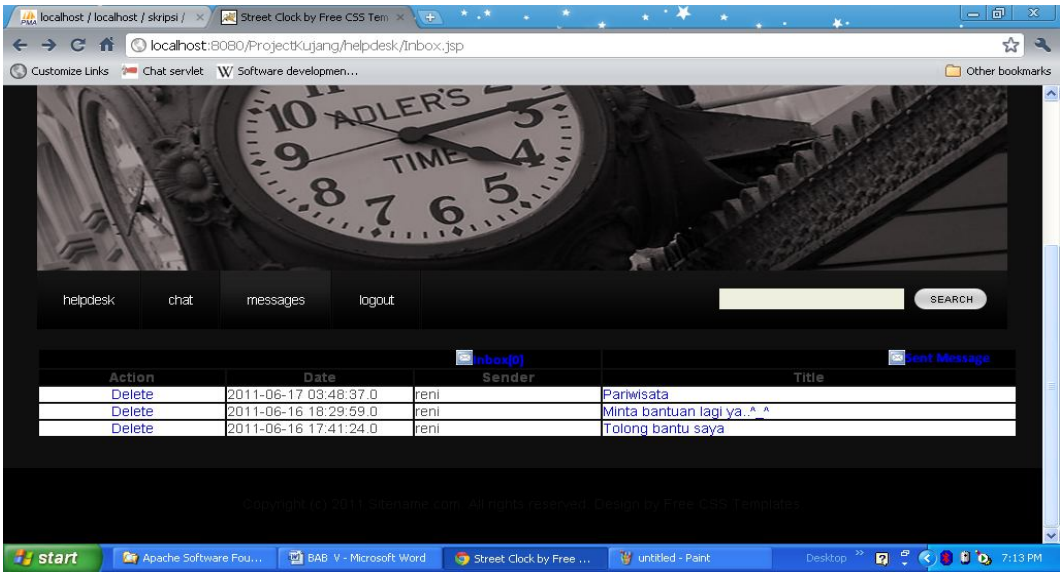
Gambar 5.12 Halaman usageProduct.jsp

Gambar 5.12 menggambarkan detail informasi penggunaan *software* jenis OLP setelah dilakukan audit.



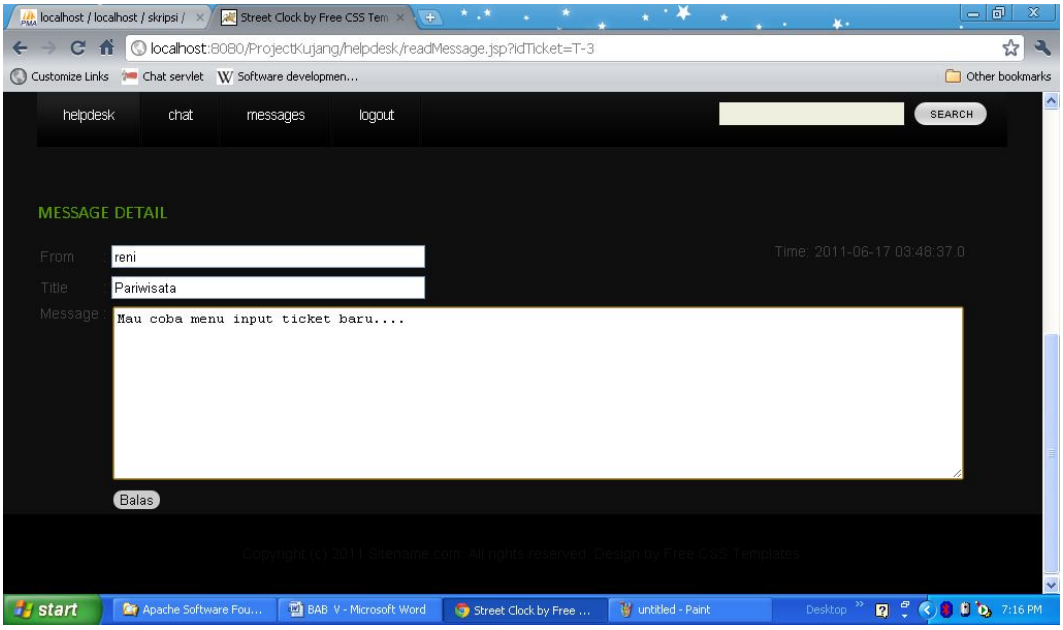
Gambar 5.13 Halaman viewReport.jsp

Gambar 5.13 menampilkan informasi hasil audit *software* berlisensi dalam bentuk grafik bar.



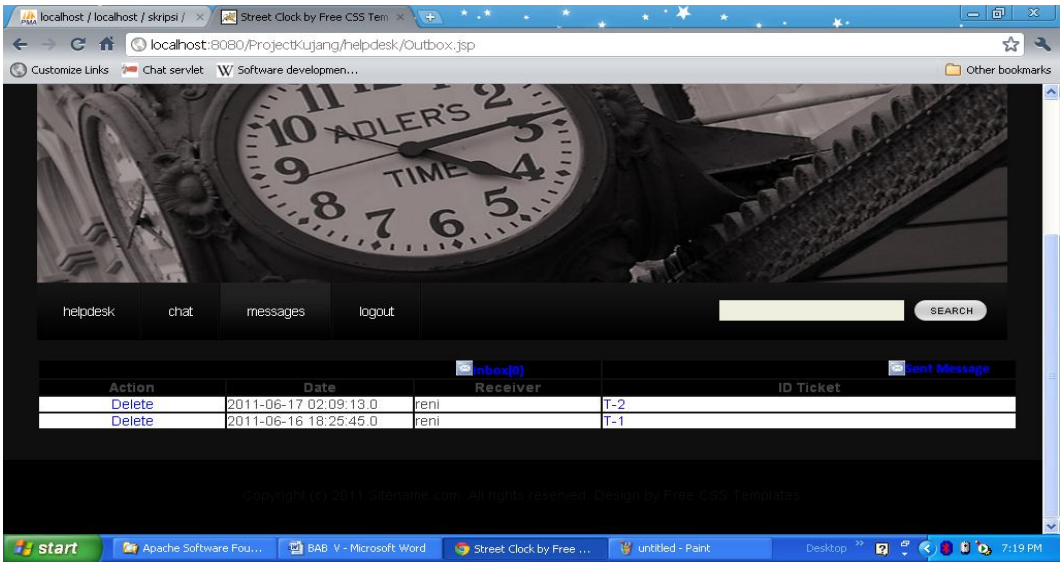
Gambar 5.14 Halaman Inbox.jsp

Gambar 5.14 menampilkan daftar *support ticket* yang masuk ke admin. Jika judul pesan tercetak tebal artinya pesan tersebut belum dibaca oleh admin.



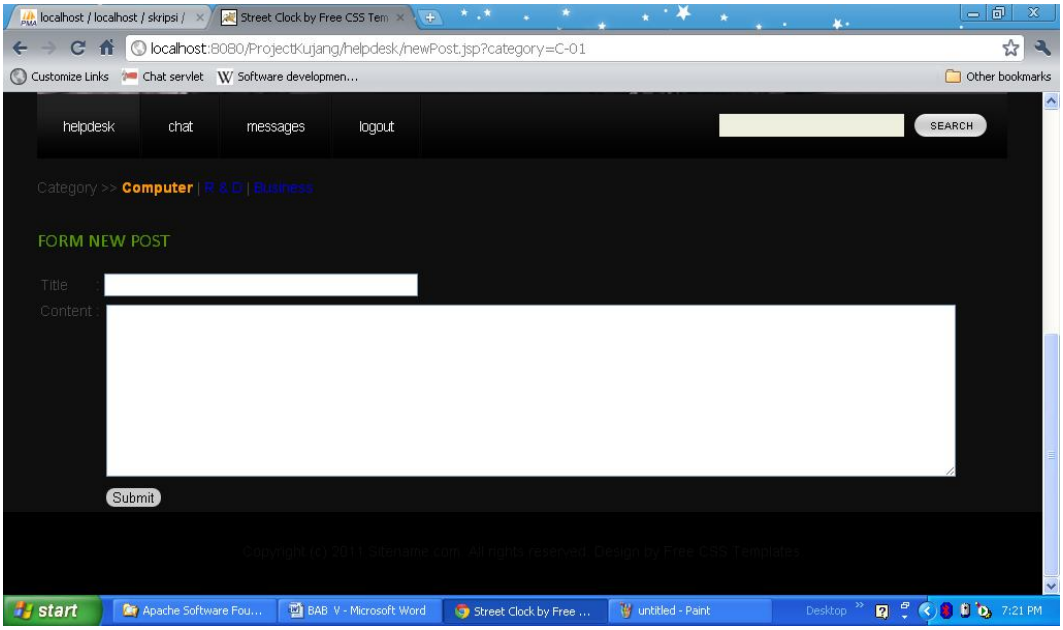
Gambar 5.15 Halaman readMessage.jsp

Gambar 5.15 menampilkan isi dari *support ticket* tersebut.



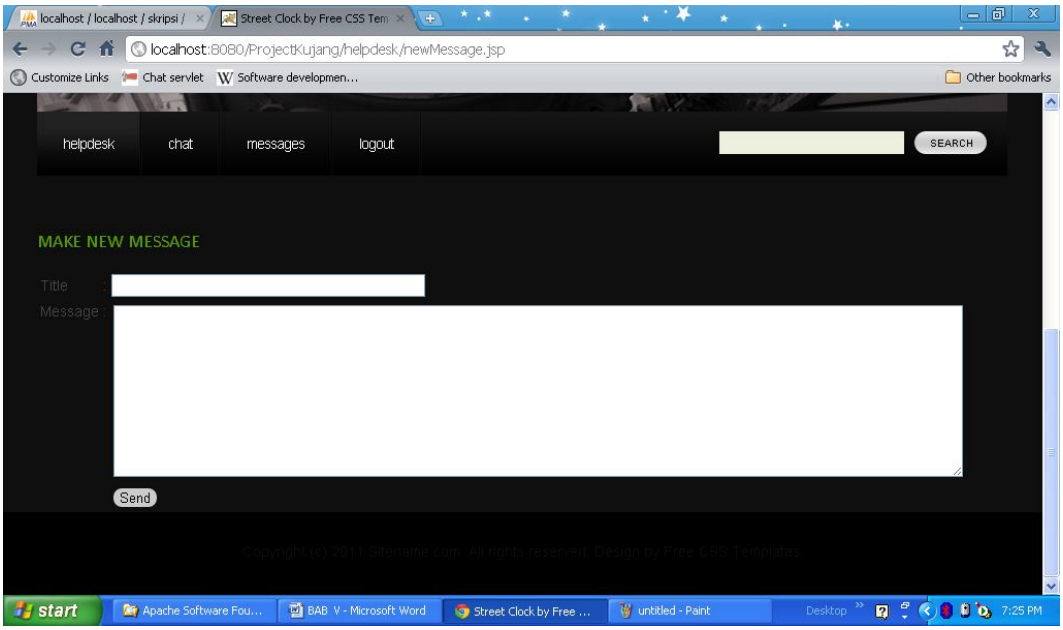
Gambar 5.16 Halaman Outbox.jsp

Gambar 5.16 menampilkan daftar *support ticket* yang terkirim atau telah direspon oleh admin.



Gambar 5.17 Halaman newPost.jsp

Gambar 5.17 merupakan form untuk memasukan *knowledge* baru. Sesuai kategori yang terpilih.



Gambar 5.18 Halaman newMessage.jsp

Gambar 5.18 merupakan model form untuk membuat pesan/*support ticket*.

5.1.5. Implementasi Instalasi Program

Program terdiri dari dua modul utama, yaitu modul berbasis web dan modul berbasis *service*. Di dalam modul berbasis *service* itu sendiri dibagi menjadi dua aplikasi diantaranya *MyRegistryServer* dan *MyRegistryClient*. Program yang akan ditanam pada komputer *server* adalah *MyRegistryServer* dan semua modul yang berbasis web. Untuk *MyRegistryServer* bisa disimpan di lokasi mana saja dalam *hard disk* komputer *server* karena sifatnya *executable*.

Sedangkan untuk modul berbasis web disimpan di *root path* instalasi *Apache Tomcat 6.0* pada komputer *server*, tepatnya di *folder*

ApacheSoftwareFoundation/Tomcat6.0/webapps. Program yang akan ditanam di masing-masing komputer *client* adalah *MyRegistryClient.* Aplikasi ini berfungsi mengirim data-data yang dibutuhkan dari setiap komputer *client* ke komputer *server.*

5.1.6. Implementasi Penggunaan Program

Implementasi penggunaan program pada sistem ini, yaitu sebagai berikut:

1. Untuk *Tracking of Using Software Licensed System*, program yang harus dijalankan dalam proses audit pertama kali adalah *MyRegistryServer.* Program ini berfungsi sebagai *listener* yang selanjutnya akan melakukan proses audit ketika ada data dari komputer *client* yang masuk. Setelah *MyRegistryServer* dijalankan, tahap selanjutnya adalah menjalankan program *MyRegistryClient* pada masing-masing komputer *client.* Supaya pengiriman berjalan otomatis, *MyRegistryClient* ini disimpan di folder *Startup* di masing-masing komputer *client*. Namun, sebelum dilakukannya audit perlu memasukan data-data utama semacam data produk *software* berlisensi, data otorisasinya, data rincian produk termasuk jumlah lisensinya untuk masing-masing *software* berlisensi, serta memetakan sementara penggunaan *software* berlisensi untuk masing-masing komputer *client*. Semua masukan data ini dilakukan melalui modul program berbasis web. Laporan hasil audit *software* bisa didapatkan pada modul

program berbasis web di dalam sistem. Sebagai catatan bahwa yang bisa masuk ke dalam *Tracking of Using Software Licensed System* hanyalah *user* dengan hak akses sebagai *administrator*.

2. Untuk *Helpdesk System*, secara umum semua bisa melihat *knowledgebase* yang tersedia di *link menu Helpdesk* pada modul program berbasis web. Namun, jika ingin memberikan komentar atau mengirim *support ticket* haruslah *login* terlebih dahulu. Jika masuk sebagai *client*, maka fitur yang didapatkan adalah fitur memberikan komentar terhadap *knowledge* yang terpublikasi, mengirim *support ticket*, melihat status *support ticket*. *Client* tidak diperkenankan membalas pesan dari ID *support ticket* yang sudah ada. Artinya jika masih perlu bantuan secara *private*, harus mengirim *support ticket* yang baru. Jika masuk sebagai *administrator*, maka fitur yang didapatkan adalah membuat *knowledge* baru, memberikan komentar pada postingan *knowledge*, mengirim balasan *support ticket*.

Penggunaan program aplikasi untuk *Tracking Software Licensed System* dan *Helpdesk System* dipisahkan ke dalam dua bagian, yaitu untuk penggunaan program pada seminar dan penggunaan program di lokasi penelitian (PT Pupuk Kujang). Untuk implementasi penggunaan program pada seminar, penulis menggunakan MySQL sebagai aplikasi basisdata dikarenakan keterbatasan *resource*. Sedangkan implementasi penggunaan program di lokasi penelitian, tetap menggunakan Oracle sebagai aplikasi basisdata.

5.2. Pengujian

Pengujian sistem perlu dilakukan untuk memastikan sistem berjalan dengan semestinya. Ada dua teknik pengujian sistem perangkat lunak, yaitu *white-box testing* dan *black-box testing*. Secara garis besar *white-box testing* digunakan untuk menguji algoritma dari suatu sistem atau dengan kata lain melihat alur proses pada *source code* yang dimiliki suatu sistem. Sedangkan *black-box testing* digunakan untuk menguji fungsionalitas sistem dengan mengabaikan algoritma pada *source code*. Teknik yang akan digunakan untuk menguji *Tracking of Using Software Licensed System* dan *Helpdesk System* adalah dengan *black-box testing*, lebih rincinya lagi dengan metode *test-case*.

5.2.1. Rencana Pengujian

Pada rencana pengujian ini, tahap-tahap yang akan dilakukan adalah:

1. Menguji validasi *form login*
2. Menguji fungsionalitas *Tracking of Using Software Licensed System.* Yang akan diujikan disini antara lain validasi form masukan data utama, seperti *form* masukan data produk, *form* masukan data otorisasi lisensi, *form* rincian jumlah lisensi tiap produk, dan *form* pemetaan penggunaan *software* berlisensi. Setelah itu akan diuji proses otomatis auditnya untuk data transaksi yang sama.

3. Menguji fungsionalitas *Helpdesk System.* Yang akan diujikan disini adalah otorisasi memberikan komentar, membuat *knowledge,* mengirim pesan (*support ticket*), dan membalas pesan (*support ticket*).

5.2.2. Kasus dan Hasil Pengujian

Pada bagian ini, kasus yang pertama kali akan diuji adalah *form* validasi *login*.

Tabel 5.1 ***Test Case Form Login***

Test Case : Validasi *Form Login*

Description : Untuk menguji apakah *user* akan diarahkan ke halaman atau menu tertentu sesuai hak akses yang dimilikinya.

ID	Aksi	Kondisi	Hasil yang diharapkan	Hasil Sesungguhnya	Keterangan
1.	User login	Username= null; Password = null	Muncul pesan error	Muncul pesan error	Sukses
		Username=true; Password = null	Muncul pesan error	Muncul pesan error	Sukses
		Username=true; Password=false	Halaman index.html muncul	Halaman index.html muncul	Sukses
		Username=true; Password=true; Hak akses=admin;	Muncul halaman utama *Tracking*	Muncul halaman utama *Tracking Software System*	Sukses

		Connect to=Tracking Software	*Software System*		
		Username=true; Password=true; Hak akses=admin; Connect to=Helpdesk	Muncul halaman forumComputerAdmin.jsp	Muncul halaman forumComputer Admin.jsp	Sukses
		Username=true; Password=true; Hak akses=client; Connect to=Helpdesk	Muncul halaman forumComputerClient.jsp	Muncul halaman forumComputer Client.jsp	Sukses

Tabel 5.2 *Test Case* Fungsionalitas *Tracking Software System*

Test Case : Fungsionalitas *Tracking Software System*

Description : Untuk menguji apakah Sistem Pelacakan Software berjalan dengan semestinya.

ID	Aksi	Kondisi	Hasil yang diharapkan	Hasil Sesungguhnya	Keterangan
1.	Input Data Produk	Part Number=true; Serial Number=true; Description=null; Version=true;	Muncul pesan error	Muncul pesan error	Sukses
		Part Number=false; Textbox yang lainnya = true;	Muncul pesan error query gagal	Muncul pesan error query gagal	Sukses
		Semua textbox=true	Muncul pesan query berhasil	Muncul pesan query berhasil	Sukses
2.	Input Otorisasi Lisensi Software	Authorization Number=false; Textbox yang lainnya= not null	Muncul pesan error query gagal	Muncul pesan error query gagal	Sukses
		Tombol pop-up disamping	Muncul window	Muncul window pop-up yang	Sukses

		textbox vendor diklik	pop-up yang menampilkan daftar vendor	menampilkan daftar vendor	
3.	Input Detail Produk	Tombol pop-up disamping textbox part number diklik	Muncul window pop-up yang menampilkan daftar software jenis OLP	Muncul window pop-up yang menampilkan daftar software jenis OLP	Sukses
4.	Input usage mapping	Tombol pop-up disamping textbox part number diklik	Muncul window pop-up yang menampilkan daftar software jenis OLP yang jumlah lisensinya > 0	Muncul window pop-up yang menampilkan daftar software jenis OLP yang jumlah lisensinya > 0	Sukses
	Input otomatis transaksi software	Part Number dan Computer Name yang sama telah tersimpan di dalam database	Data batal dimasukan ke dalam database	Data batal dimasukan ke dalam database	Sukses

Tabel 5.3 ***Test Case*** **Fungsionalitas** ***Helpdesk System***

Test Case : Fungsionalitas *Helpdesk System*

Description : Untuk menguji apakah *Helpdesk System* telah berjalan sebagaimana mestinya.

ID	Aksi	Kondisi	Hasil yang diharapkan	Hasil Sesungguhnya	Keterangan
1.	Mengirim atau membalas *support ticket*	Title=not null; Message=not null	Muncul pesan query berhasil	Muncul pesan query berhasil	Sukses
		Title=null; Message = null	Muncul pesan error	Muncul pesan query berhasil	Gagal
2.	Melihat indikator pesan masuk	Status pesan = unread	Judul / Isi pesan tersebut tercetak tebal di daftar inbox	Judul / Isi pesan tersebut tercetak tebal di daftar inbox	Sukses
		Ketika link judul/isi pesan diklik untuk melihat rinciannya	Status pesan=read; Status pesan tidak tercetak tebal	Status pesan=read; Status pesan tidak tercetak tebal	Sukses

5.2.3. Kesimpulan Hasil Pengujian

Dari hasil pengujian yang dilakukan dengan *test-case*, 99% sistem telah berjalan dengan semestinya. Ada sedikit kesalahan pada validasi pembuatan pesan yang mengijinkan nilai *null* untuk diproses.