WorkShop GTFW Oracle

Tim Integrasi UGM

WorkShop Pengenalan Database Oracle dan

GTFW (GamaTechno web application development FrameWork)

www.oit.ugm.ac.id

Jadwal Kegiatan

I. Hari ke 1, Senin, 11 Oktober 2010

1. Installasi Oracle XE (eXpress Edition)
   Windows & Linux (rpm, deb)

2. Installasi Apache – PHP – Oci8
   XAMPP / LAMPP

3. Dasar Pemrograman PHP Oracle
   PHP Oci8 Extension, ADODB

II. Hari ke 1, Senin, 11 Oktober 2010

1. Installasi GTFW UGM

2. Fitur GTFW UGM
   GTFW+, (HMVC, Themes, dll)

3. Pemrograman GTFW UGM
   Menu, Modul, dll

Daftar Isi

Daftar Tabel

Daftar Gambar

Hari ke 1, Senin 11 Oktober 2010

1. SESI I

1.1. Installasi Oracle Database 10g eXpress Edition

Tujuan Pembelajaran Khusus

- Peserta WorkShop secara mandiri dapat melakukan installasi Oracle Database 10g eXpress Edition baik di sistem operasi Windows maupun Linux.

1.1.1. Mengenal Oracle Database 10g eXpress Edition

Oracle Database 10g eXpress Edition yang lebih dikenal dengan sebutan Oracle Database XE merupakan sebuah *entry-level* dan *small-footprint* produk Database dari Oracle. Oracle Database XE dikembangkan berdasar *code base* dari Oracle Database 10g Release 2. Dengan Oracle Database XE ini kita dapat men-*develop*, men-*deploy* dan medistribusikan aplikasi baik untuk kepentingan komersial maupun non-komersial.

Oracle Database XE sangat cocok untuk :

- Para Developer yang berkerja dengan PHP, Java, .NET, XML, dan aplikasi *open source* lainnya.
- Para DBA yang menginginkan produk Oracle Database yang *free* serta dapat pula digunakan untuk kebutuhan *training* maupun *deployment.*
- *Independent Software Vendors* (ISVs) dan vendor *hardware* yang ingin mendistribusikan produk yang dikembangkannya dengan Oracle Database XE.
- Institusi Pendidikan yang ingin menggunakan Oracle Database XE sebagai alat bantu dalam kurikulumnya.

Dengan Oracle Database XE, kita dapat men-*develop* dan men-*deploy* aplikasi yang *powerfull*, *proven*, *industry-leading infrastruktur*, dan jika perlu dapat di-*upgrade* ke produk Oracle Database versi di atasnya dengan mudah.

Oracle Database XE dapat di-*install* pada semua mesin komputer dengan jumlah CPU berapapun (satu database per-host komputer), namun XE membatasi kapasitas penyimpanan datanya hingga mencapai 4GB dan batas memori yang dapat digunakan adalah 1GB dari seluruh total memori yang ada, serta satu CPU per-*host* nya.

1.1.2.Paket Installasi

Oracle telah meyediakan paket installasi Oracle Database 10g eXpress Edition untuk sistem operasi Linux dan Windows. Untuk sistem operasi Linux, oracle telah menyediakan dua paket installasi Oracle Database XE dengan format rpm (untuk Linux keluarga Ret Hat) dan format deb (untuk Linux keluarga Debian). Untuk paket installasi terbaru dari Oracle Database XE ini dapat di unduh di www.oracle.com.

1.1.3. Installasi Oracle Database XE di Sistem Operasi Windows

1.1.3.1.System Requirement

Untuk dapat meng-install Oracle Database XE di sistem operasi Windows, Anda membutuhkan spesifikasi minimal sebagai berikut :

Kebutuhan	Nilai Minimal	Keterangan
Arsitektur Sistem	Intel (x86)	
Sistem Operasi	Salah satu sistem operasi Windows 32 bit yang tercantum di bawah ini : • Windows 2000 Service Pack 4 or later • Windows Server 2003 • Windows XP Service Pack 1 or later • Windows 7	
Protokol Network	TCP/IP	Jika terhubung ke jaringan
Disk space	• Server Component : minimal 1.6 GB • Client Component : minimal 75 MB	Yang dimaksud Client Component adalah Oracle Database XE Client
RAM	• Minimal 256 MB • Rekomendasi dari Oracle : 512 MB	
Microsoft Windows Intaller (MSI)	MSI version 2.0 or later	

Tabel 1: Kebutuhan Installasi Oracle Database XE di Windows

1.1.3.2.Langkah Installasi

Sebelum menginstall Oracle Database XE di Windows, pastikan bahwa Anda login dengan *user* yang mempunyai hak akses sebagai Administrator. Untuk lebih jelasnya, silahkan ikuti langkah-langkah berikut :

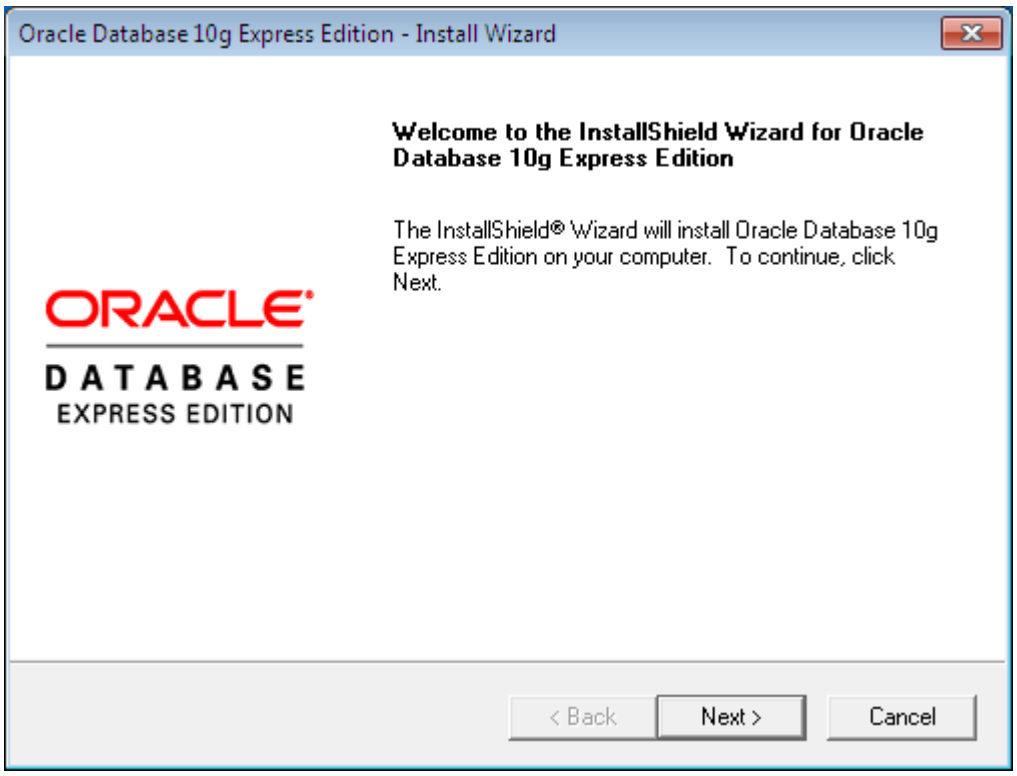
1. Download paket installasi Oracle Database XE untuk Windows dari url yang telah disediakan.

   Url Download : http://dev.oit.ugm.ac.id/

   Unduh file berikut :

   - `OracleXEUniv.exe`

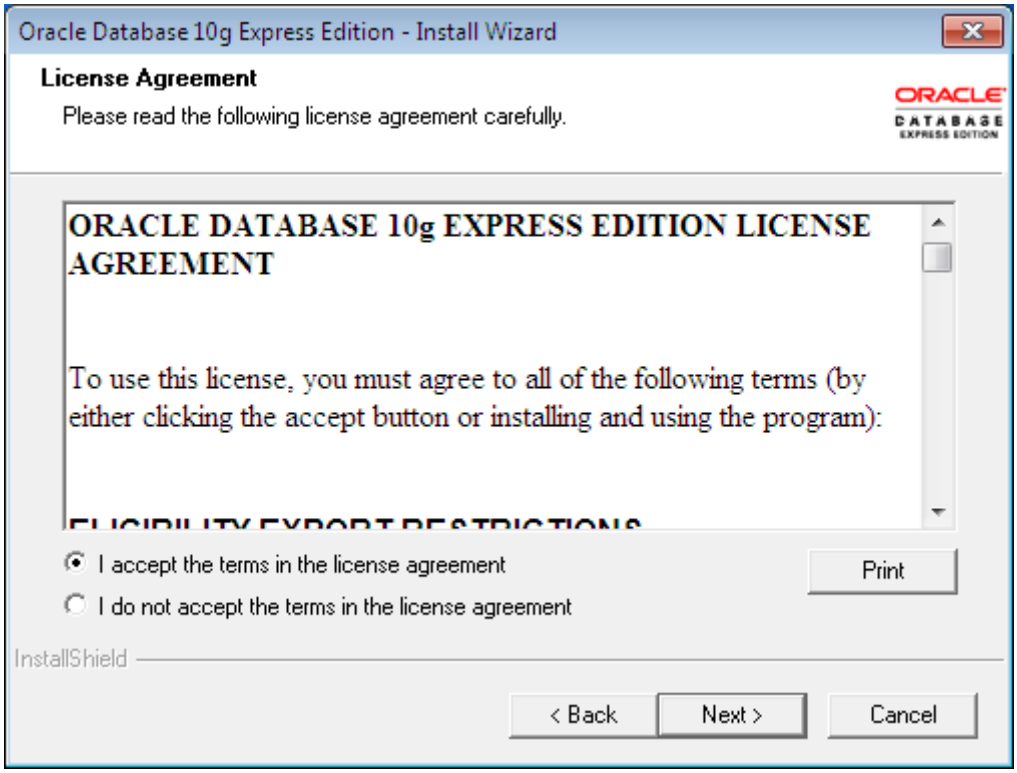
2. Jalankan file paket installasi Oracle Database XE. Jika sukses, maka akan muncul tampilan Windows seperti berikut :



Gambar 1: Capture Langkah Pertama Installasi Oracle Database XE di Windows

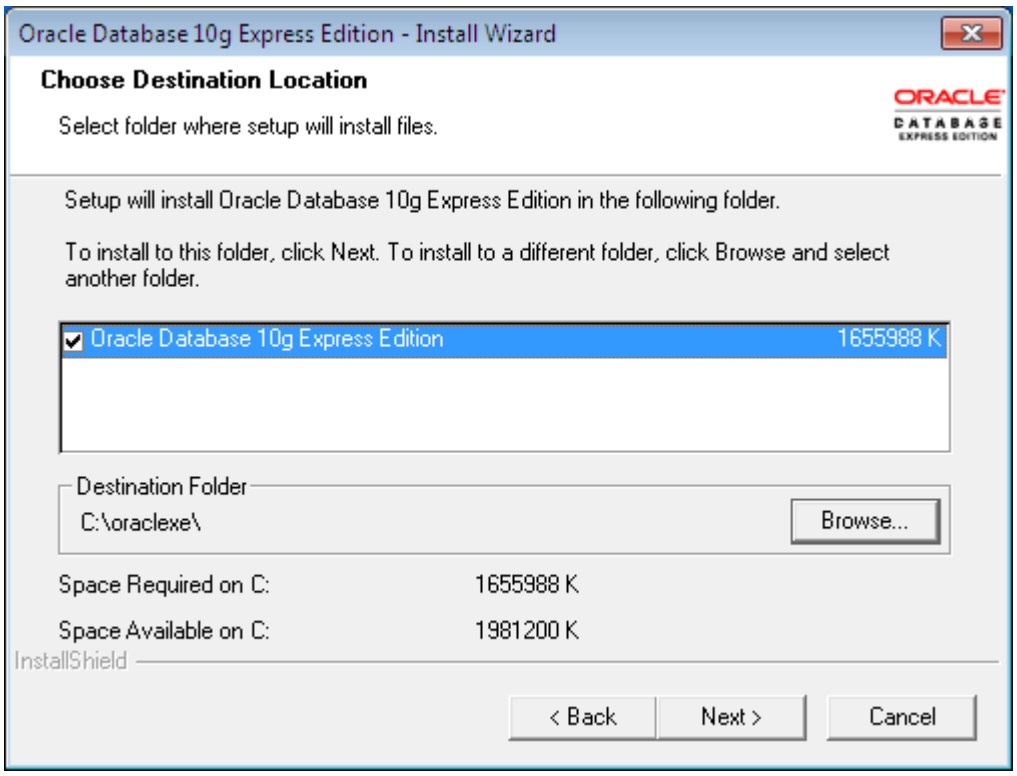
Jika dikomputer Anda telah terinstall Oracle Database XE, dan jika Anda ingin me-re-install, maka pilih tombol ***Repair*** dan diikuti dengan tombol ***Next***.

3. Pada tahapan "*Window License Agreement*", pilih ***I accept*** kemudian klik tombol ***Next***.



Gambar 2: Window License Agreement pada Installasi Oracle Database XE di Windows

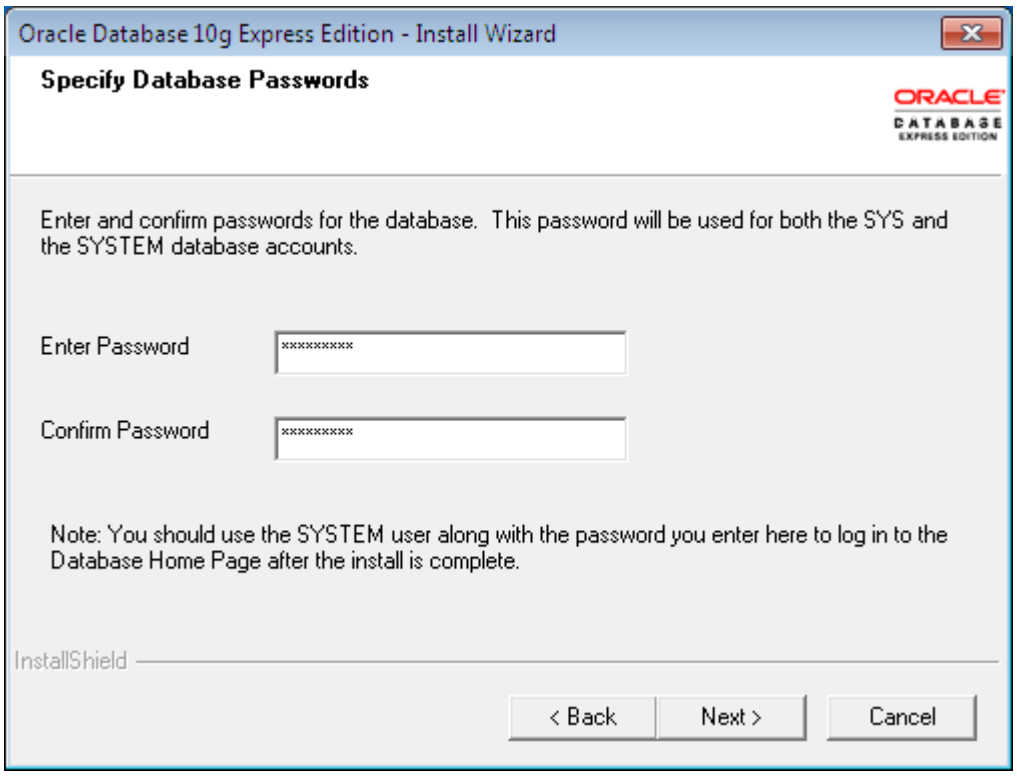
4. Pada tahapan "*Window Choose Destination*", sesuaikan letak *folder* installasi sesuai dengan kebutuhan Anda, kemudian klik tombok ***Next***.



Gambar 3: Capture Tahapan Pemilihan Folder Installasi Oracle Database XE di Windows

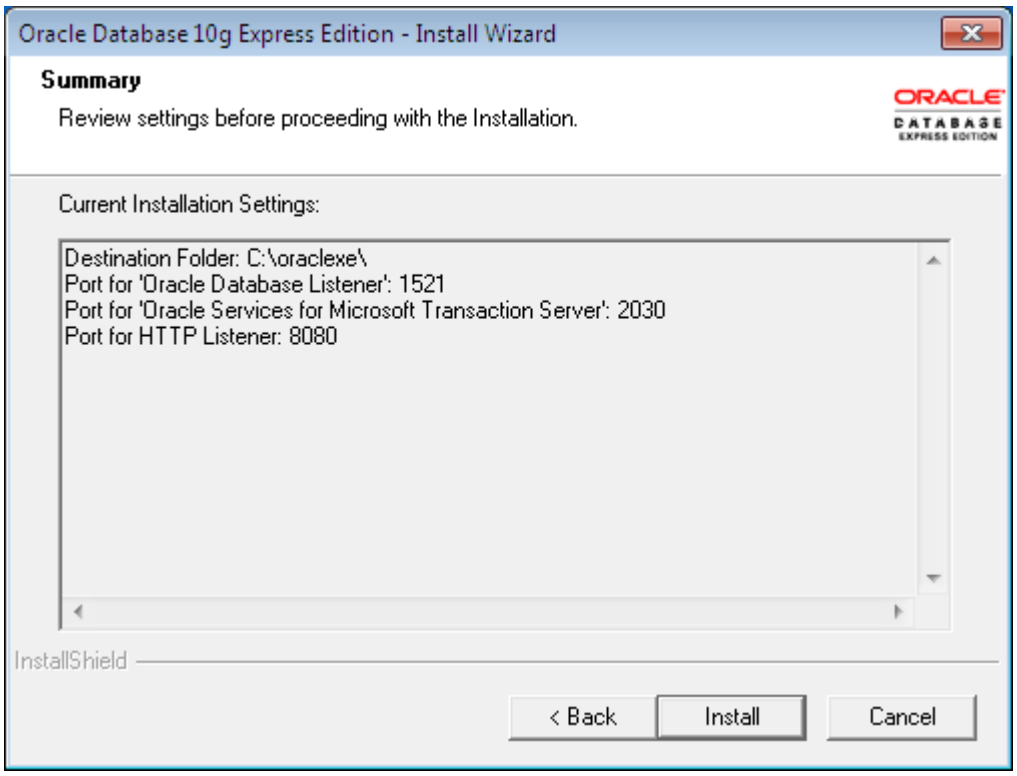
5. Jika muncul window yang berisi perintah untuk memasukkan nomor port, maka silahkan isi sesuai dengan kebutuhan Anda. Secara default Oracle Database XE akan membutuhkan port-port sebagai berikut :
   - Port 1521 : Oracle Database Listener.
   - Port 2030 : Oracle Services untuk Microsoft Transaction Server.
   - Port 8080 : GUI berbasis Web untuk manajemen Oracle Database XE (protokol HTTP).

Pada langkah berikutnya Anda akan diminta untuk memasukkan password untuk *user* SYS dan SYSTEM, kemudian klik tombol ***Next.***



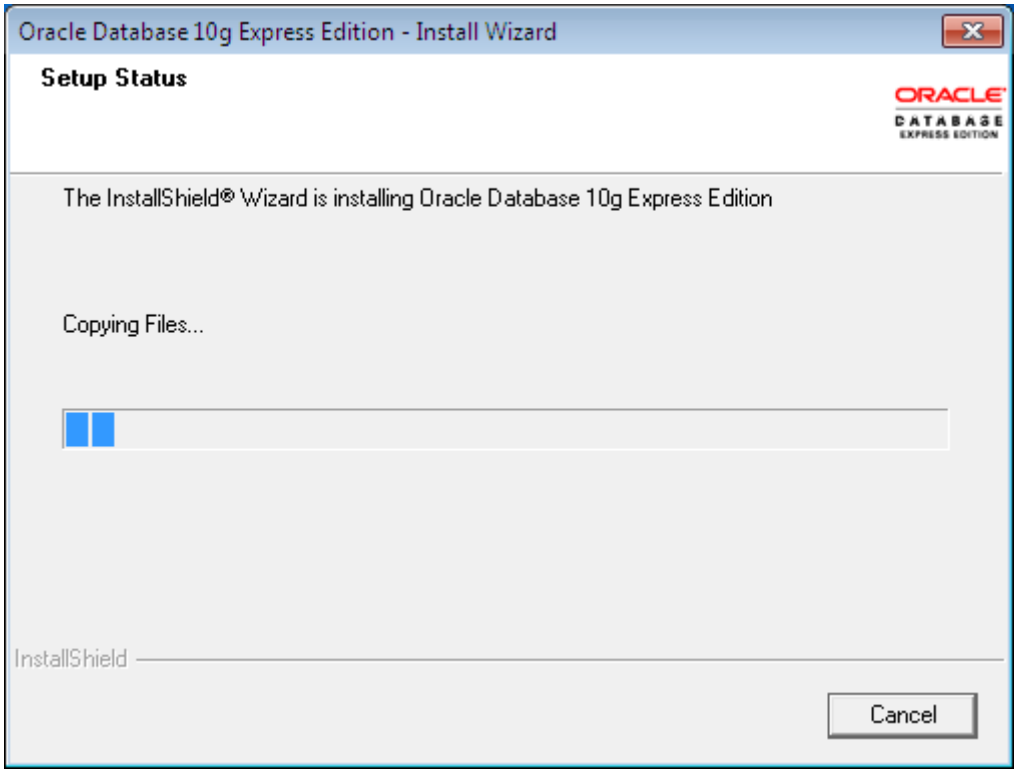
Gambar 4: Capture Tahapan Input Password Installasi Oracle Database XE di Windows

6. Pada tahapan ini Anda akan di hadapkan pada tampilan yang berisi *Review* konfigurasi yang akan digunakan dalam proses installasi Oracle Database XE. Jika Anda sudah yakin, maka klik tombol ***Install***.



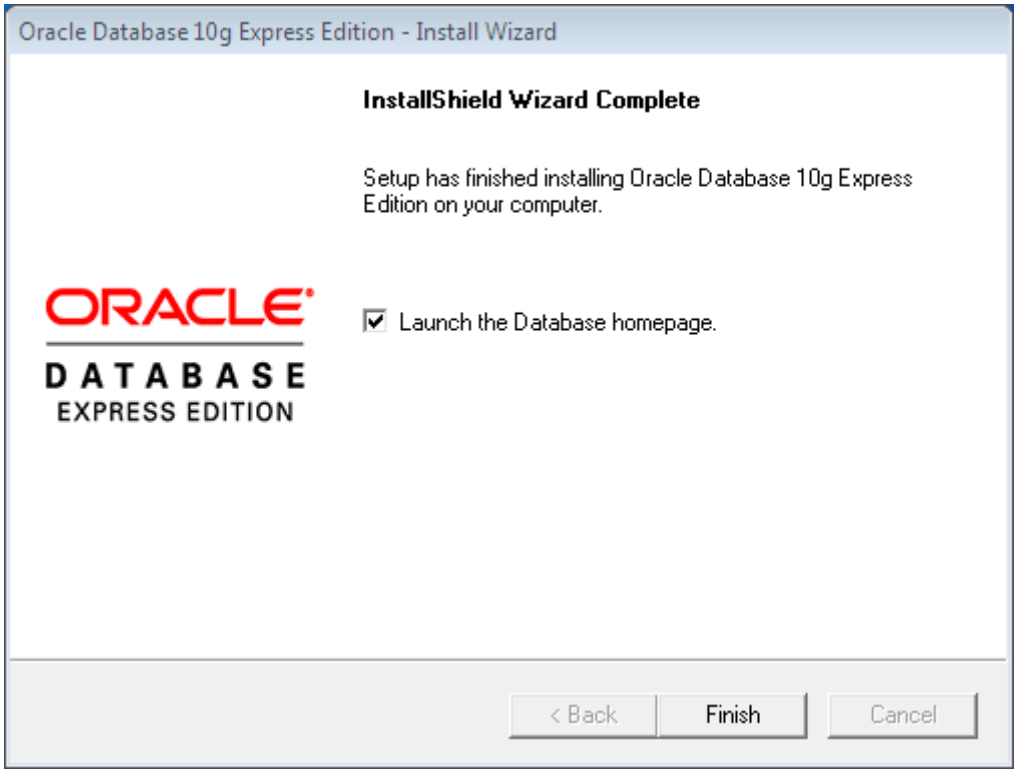
Gambar 6: Capture Tahapan Review Intallasi Oracle Database XE di Windows

7. Pada tahap ini Anda akan dihadapkan pada window proses installasi. Biarkan proses ini berjalan sampai selesai.



Gambar 7: Proses Installsi Oracle Database XE di Windows

8. Setelah proses installasi selesai, maka akan muncul jendela konfirmasi untuk membuka halaman Administrasi Database XE melalui browser Anda. Centang *Launch the Database homepage* jika Anda ingin membuka halaman tersebut.



Gambar 8: Window konfirmasi Launch the Database homepage

1.1.4. Installasi Oracle Database XE di Sistem Operasi Linux

1.1.4.1.System Requirement

Tabel berikut menjelaskan spesifikasi minimal yang dibutuhkan untuk dapat menginstall Oracle Database XE di sistem operasi Linux.

Kebutuhan	Nilai Minimal	Keterangan
Sistem Operasi	Salah satu dari sistem operasi Linux 32 bit di bawah ini : • Oracle Enterprise Linux (OEL) • Red Hat Enterprise Linux (RHEL) • Suse (SLES) • Mandriva • Novell • Debian • Ubuntu	Berlaku juga untuk keluarga / distro yang dapat menjalankan format rpm dan format deb
Network Protocol	TCP/IP	
RAM	• Server Component : Minimal 256 MB, recommended 512 MB • Client Component : 256 MB	
Disk Space	• Server Component : minimal 1.5 GB • Client Component : minimal 100 MB	
Package	Untuk Oracle Database XE Server dan Client : • glibc – 2.3.2 • libaio – 0.3.96	
SWAP Space Requirement		
RAM ukuran 0 s.d 256 MB	3 kali ukuran RAM	
RAM ukuran 256 s.d 512 MB	2 kali ukuran RAM	
RAM ukuran di atas 512 MB	1024 MB	

Tabel 2: Kebutuhan Installasi Oracle Database XE di Linux

1.1.4.2.Langkah Installasi

Pastikan Anda login ke sistem operasi linux kesayangan Anda dengan *user* yang mempunyai hak akses root. Untuk lebih jelasnya silahkan ikuti langkah-langkah berikut :

1. Download paket installasi Oracle Database XE untuk sistem operasi Linux dari url yang telah disediakan.

   Url Download : http://dev.oit.ugm.ac.id/

Unduh file berikut :

- Paket rpm : `oracle-xe-universal-10.2.0.1-1.0_i386.rpm`
- Paket deb : `oracle-xe-universal_10.2.0.1-1.0_i386.deb`

2. Jalankan file paket installasi Oracle Database XE untuk meng-install atau untuk meng-upgrade Oracle Database XE Server.
   - Untuk file installasi dalam format debian maka jalankan perintah berikut :

     ```
     $ sudo dpkg -i oracle-xe-universal_10.2.0.1-1.0_i386.deb
     ```

   - Untuk file installasi dalam format rpm jalankan perintah beikut :

     ```
     $ rpm -ivh oracle-xe-universal_10.2.0.1-1.0_i386.rpm
     ```

3. Berikutnya Anda akan diminta untuk melakukan konfigurasi. Ketik perintah berikut untuk melakukan konfigurasi :

   ```
   $ /etc/init.d/oracle-xe configure
   ```

4. Untuk selanjutnya Anda akan diminta untuk memasukkan nomor-nomor port yang dibutuhkan oleh Oracle Database XE. Berikut informasi mengenai konfigurasi port tersebut :
   - Port HTTP untuk GUI manajemen Oracle Database XE (secara default akan di set ke port 8080).
   - Port Oracle Database Listener (secara default akan di set ke port 1521).
   - Password untuk user SYS dan SYSTEM (user administrator).
   - Jika Anda menginginkan service Oracle Database XE berjalan secara otomatis saat startup, maka pilih Yes untuk dialog "*Do you want Oracle Database 10g Express Edition to be started on boot?*".

```
Oracle Database 10g Express Edition Configuration
-------------------------------------------------
This will configure on-boot properties of Oracle Database 10g Express
Edition.  The following questions will determine whether the database should
be starting upon system boot, the ports it will use, and the passwords that
will be used for database accounts.  Press <Enter> to accept the defaults.
Ctrl-C will abort.

Specify the HTTP port that will be used for Oracle Application Express [8080]:8080

Specify a port that will be used for the database listener [1521]:1521

Specify a password to be used for database accounts.  Note that the same
password will be used for SYS and SYSTEM.  Oracle recommends the use of
different passwords for each database account.  This can be done after
initial configuration: password
Confirm the password: password

Do you want Oracle Database 10g Express Edition to be started on boot (y/n) [y]: y
```

Tekan ***Enter*** setelah selesai memasukkan konfigurasi. Kemudian tunggu sampai proses konfigurasi selesai.

```
Starting Oracle Net Listener...Done
Configuring Database...Done
Starting Oracle Database 10g Express Edition Instance...Done
Installation Completed Successfully.
To access the Database Home Page go to "http://127.0.0.1:8080/apex"
```

5. Berikutnya jika Anda ingin membuka halaman administrasi manajemen Oracle Database XE, Anda dapat membukanya melalui browser Anda dengan alamat url : http://127.0.0.0.1:8080/apex.

2. SESI II

2.1. Installasi Paket Apache, PHP dan Oracle Instant Client

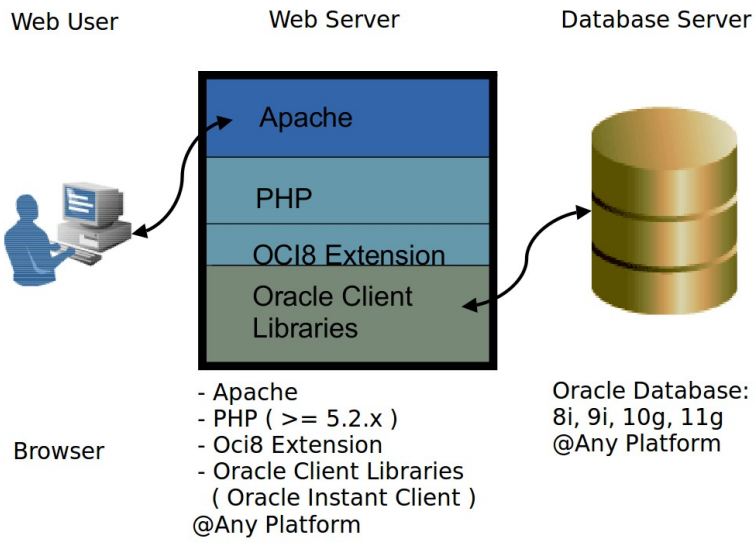
Tujuan Pembelajaran Khusus

- Peserta Workshop secara mandiri dapat melakukan installasi paket Apache, PHP dan Oracle Instant Client baik di sistem operasi Windows maupun Linux

2.2. Paket Installasi

Untuk keperluan workshop kali ini, peserta workshop akan menggunakan paket XAMPP (yaitu paket software development yang di dalamnya terdapat paket perangkat lunak Apache Web Server, MySQL, PHP dan Perl). Paket XAMPP sendiri tersedia dalam beberapa pilihan sistem operasi yaitu : Windows, Linux, Mac Os X, dan Solaris. Namun dalam workshop kali ini yang akan dibahas hanyalah installasi XAMPP untuk sistem operasi Windows dan Linux.

Agar PHP dapat berkomunikasi dengan Oracle Database, maka dibutuhkan *library* tambahan, *library* yang akan digunakan kali ini adalah Oracle Instant Client (selain itu Anda juga dapat menginstall Oracle Client sebagai pengganti Oracle Instant Client). Berikut gambaran singkat skema komunikasi antara PHP dan Oracle Database :



Gambar 9: Skema Arsitektur Komunikasi PHP dengan Database Oracle

2.3. Installasi OWAP (Oci8 Extension-Windows-Apache-PHP)

2.3.1.Installasi Oracle InstantClient di Windows

Berikut adalah langkah-langkah untuk menginstall oracle instantclient di sistem operasi Windows :

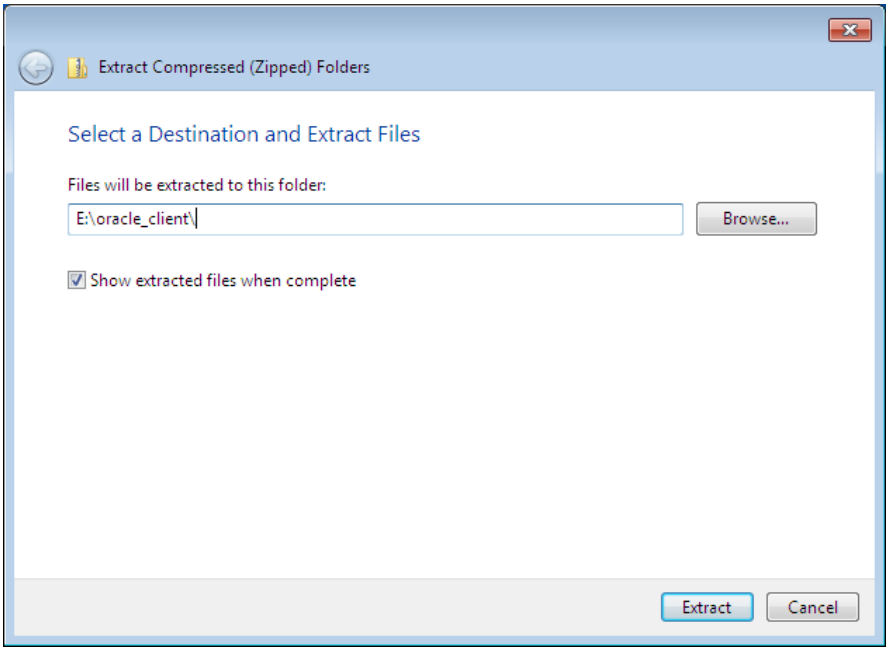
1. Download paket oracle instant client yang telah di sediakan.

   Url Download : http://dev.oit.ugm.ac.id

   Unduh file berikut :

   - `instantclient-basic-win32-10.2.0.4.zip`

2. *Extract* file `instantclient-basic-win32-10.2.0.4.zip` ke folder yang Anda inginkan. Untuk lebih jelasnya silahkan lihat gambar berikut (dalam hal ini paket tersebut di extract ke folder `e:\oracle_client\` sehingga folder letak paket oracle instantclient adalah `e:\oracle_client\instantclient_10_2\`):



Gambar 10: Menentukan folder letak paket oracle instantclient basic

3. Langkah selanjutnya adalah menambahkan direktori letak oracle instantclient pada tahap 3 ke environtment PATH. Tambahkan path oracle insantclient tersebut sebelum semua direktori milik Oracle. Hal ini dimaksudkan agar oracle client yang nantinya akan digunakan adalah oracle instantclient yang terletak di folder yang telah Anda buat pada tahap 3 (`e:\oracle_client\instantclient_10_2\`). Untuk Windows XP, Anda dapat menemukan setting environtment PATH melalui menu **`Start -> Control Panel -> System -> Advanced -> Environment Variables.`**

2.3.2.Installasi XAMPP di Windows

Dalam workshop kali ini, versi XAMPP yang akan digunakan adalah versi 1.7.0. Versi ini dipilih karena secara default XAMPP versi 1.7.0 dipersiapkan untuk Oracle Database versi 10g dengan menyediakan library oracle client versi 10.x (`php_oci8.dll`). Ikuti langkah-langkah berikut untuk menginstall XAMPP di Windows.

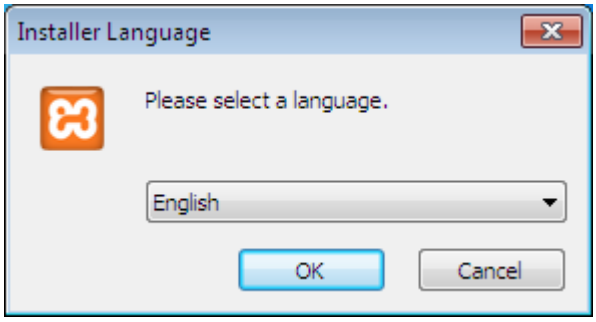
1. Download paket XAMPP versi 1.7.0 dari url yang telah disediakan.

   Url Download : http://dev.oit.ugm.ac.id

Unduh file :

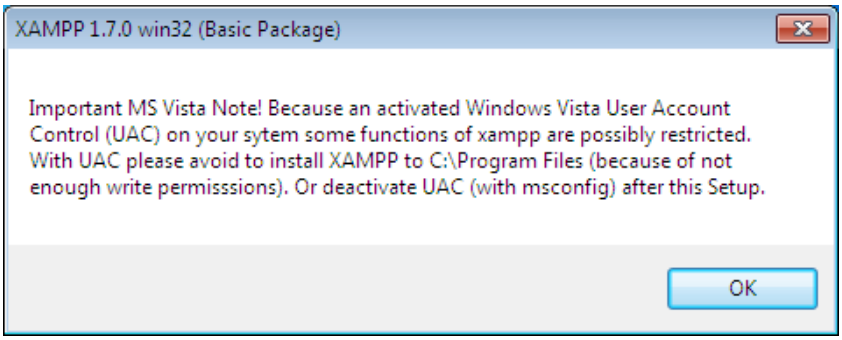
- `xampp-win32-1.7.0-installer.exe`

2. Jalankan file paket insatallasi XAMPP. Maka akan muncul tampilan sebagai berikut. Kemudian pilih bahasa yang Anda inginkan dan klik ***Ok***.



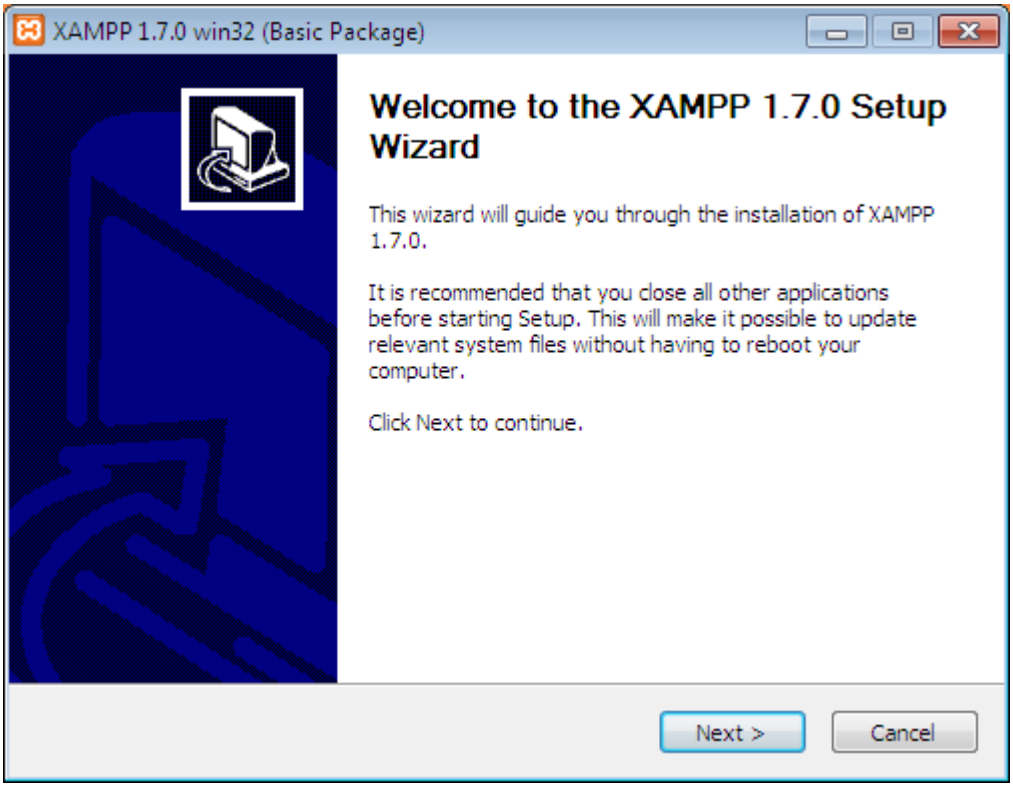
Gambar 11: Dialog memilih bahasa untuk installasi XAMPP di Windows

3. Tahap berikutnya Anda akan dihadapkan pada informasi tentang letak folder yang direkomendasikan untuk installasi paket XAMPP. Jika muncul dialog ini, silahkan klik ***Ok***.



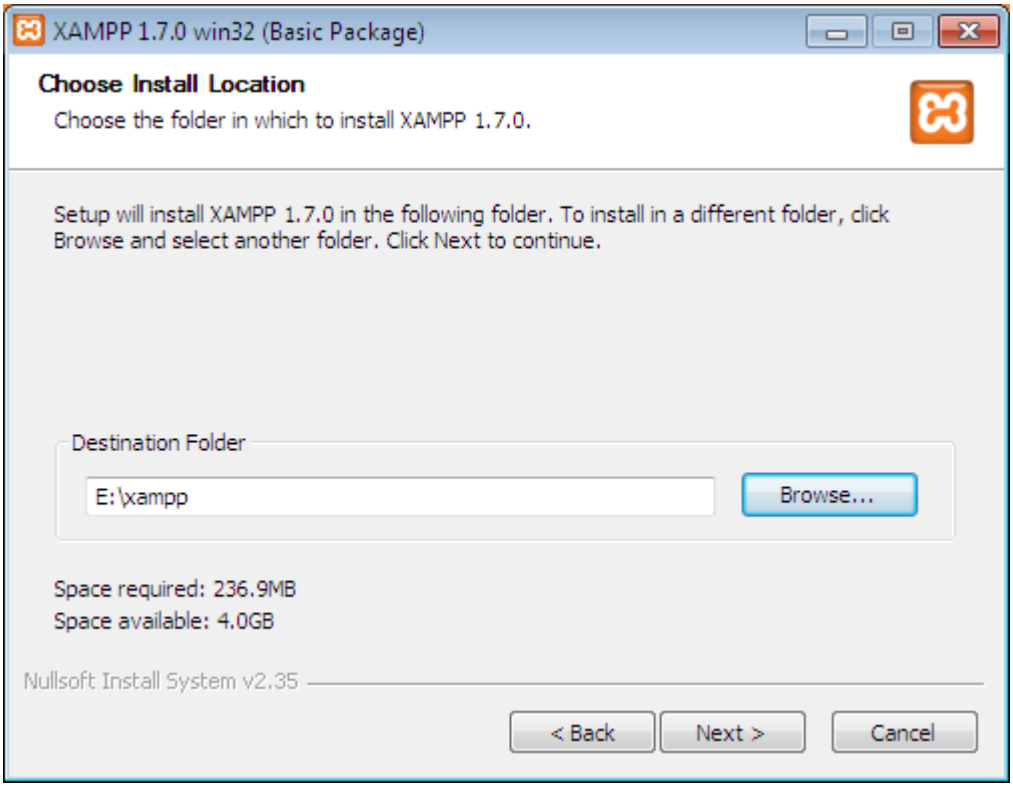
Gambar 12: Dialog informasi rekomendasi letak folder intallasi paket XAMPP

4. Pada tahap ini akan muncul “Welcome Window”, klik tombol ***next***.



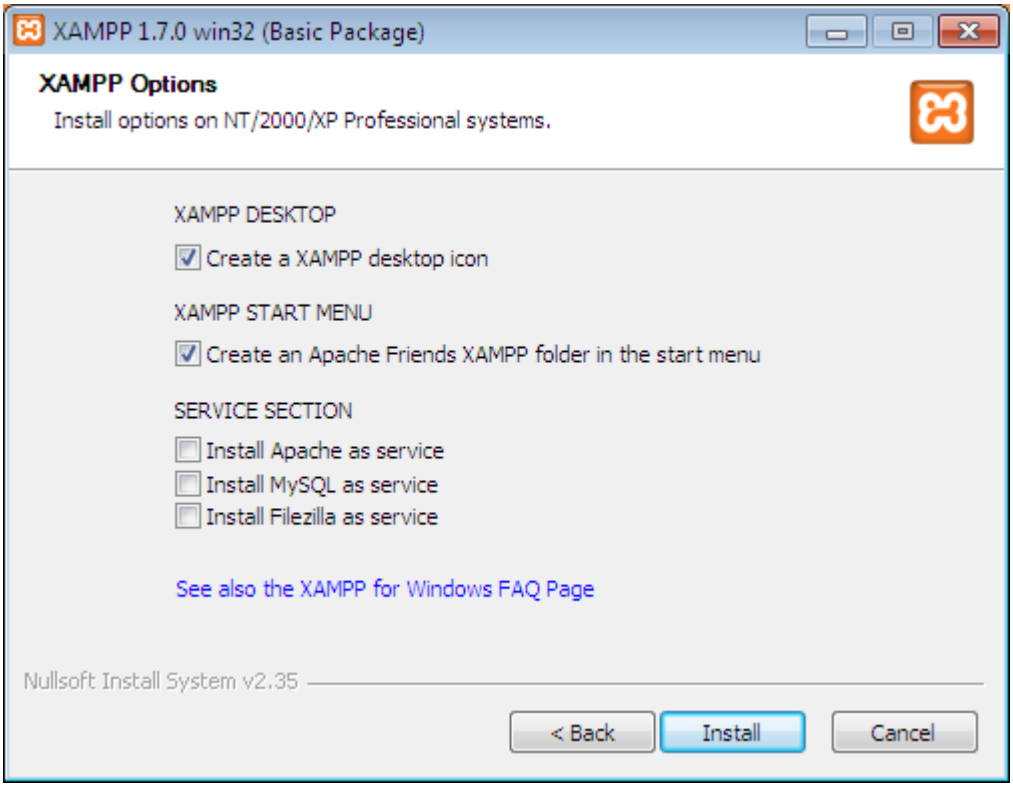
Gambar 13: "Welcome Window" pada installasi XAMPP di Windows

5. Langkah selanjutnya adalah memilih letak direktori tempat installasi paket XAMPP. Klik tombol ***next s***etelah Anda memilih letak folder installasi XAMPP!



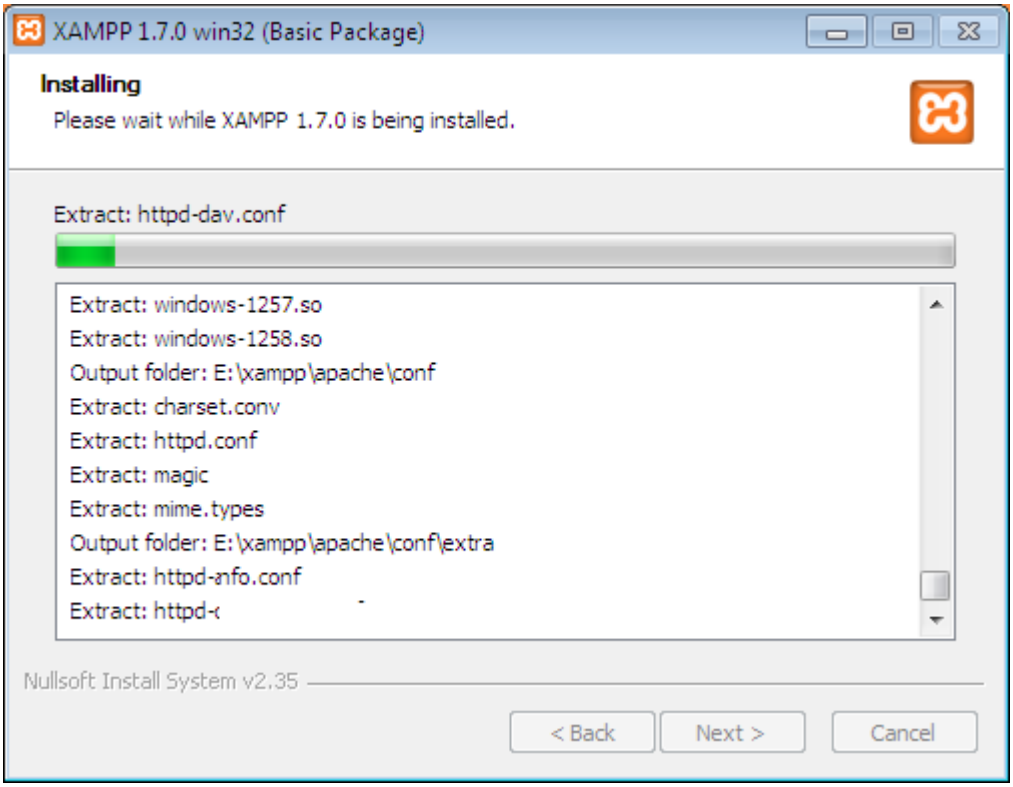
Gambar 15: Window memilih folter tujuan installasi paket XAMPP di Windows.

6. Tahap berikutnya Anda akan diminta untuk memilih pilihan installasi XAMPP. Silahkan pilih "XAMPP Options" sesuai dengan kebutuhan, kemudian klik ***Install***.



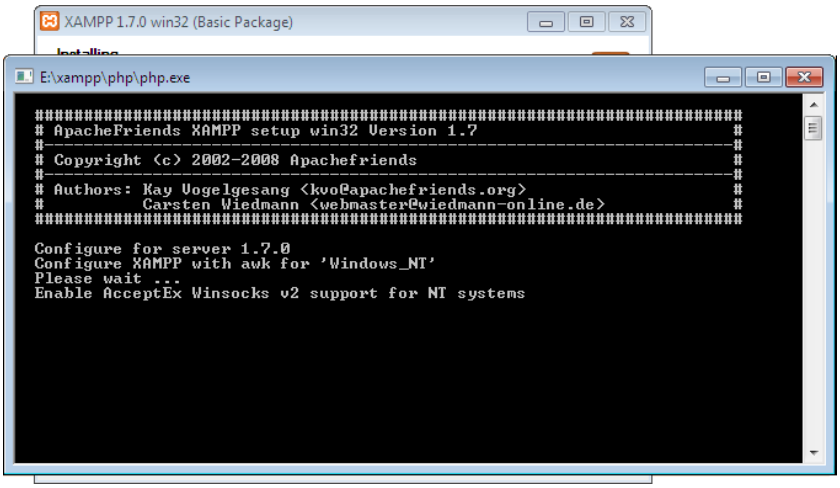
Gambar 16: Dialog pilihan "XAMPP Options" installasi paket XAMPP di Windows.

7. Selanjutnya akan muncul window proses installasi paket XAMPP. Tunggu proses installasi ini hingga selesai.



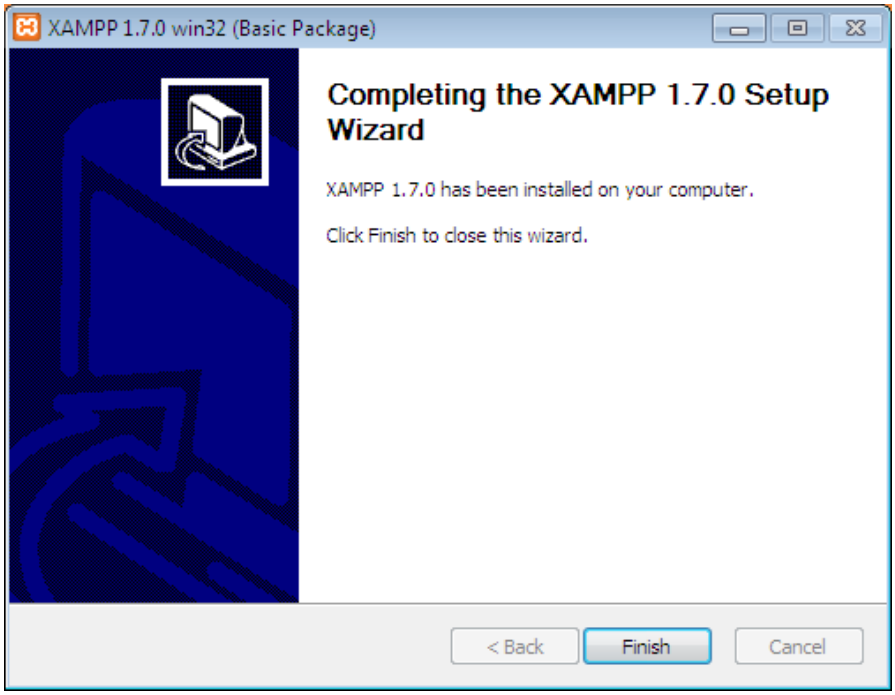
Gambar 17: Window proses installasi paket XAMPP di Windows.

8. Setelah proses installasi selesai, XAMPP akan melakukan setting konfigurasi seperti tampak pada gambar berikut.



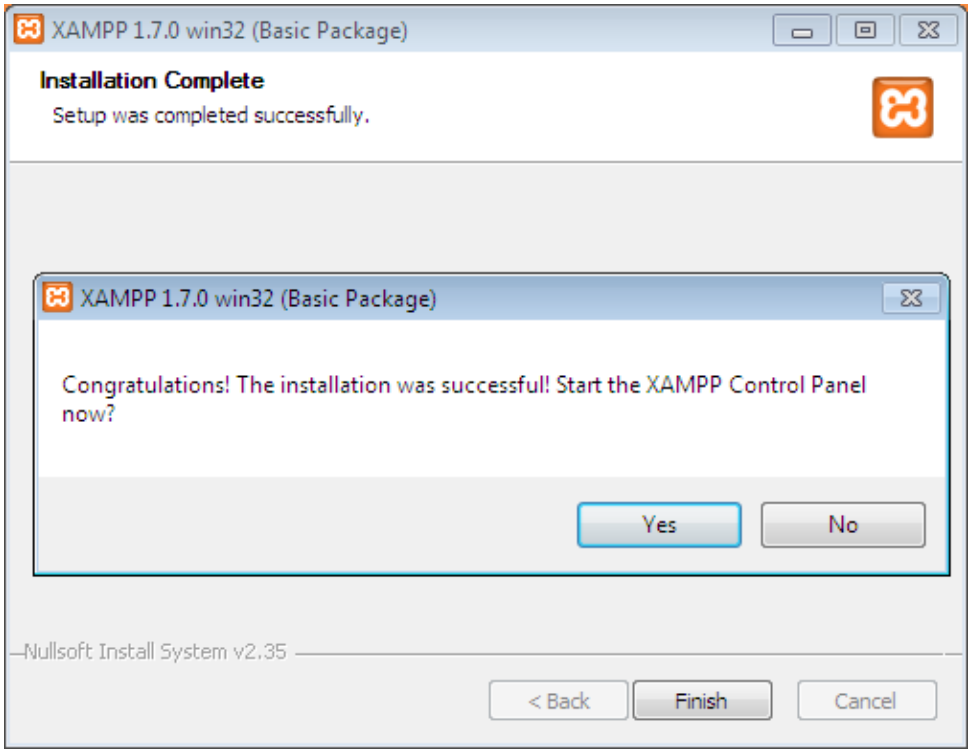
Gambar 18: Capture Window konfigurasi paket XAMPP di Windows.

9. Setelah proses konfigurasi selesai, maka akam muncul window informasi bahwa XAMPP telah berhasil diinstall. Klik ***finish*** untuk mengakhiri proses installasi XAMPP.



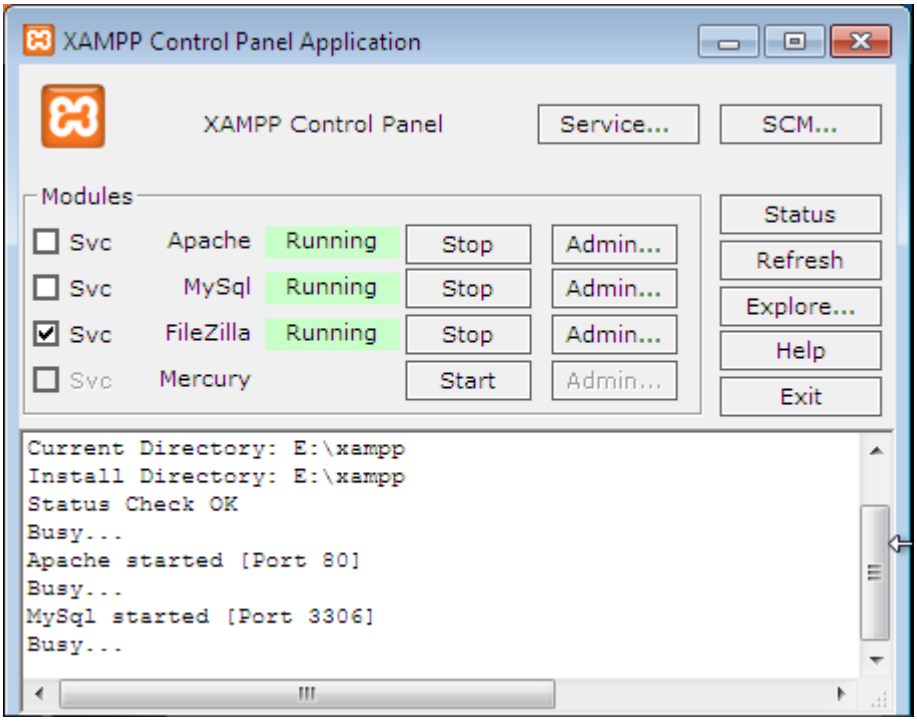
Gambar 19: Window konfirmasi XAMPP selesai diinstall di Windows

10. Tahapan berikutnya adalah menjalankan service Apache Web Server XAMPP di Windows. Setelah proses installasi selesai, Anda dapat menjalankan service Apache Web Server dengan menjalankan aplikasi "XAMPP Control Panel". Pilih ***Yes*** untuk membuka "XAMPP Control Panel".



Gambar 20: Window konfirmasi menjalankan "XAMPP Control Panel"

11. Tahapan berikutnya Anda dapat menjalankan service Apache Web Server dengan XAMPP Control Panel. Klik tombol start pada pilih “Apache” Web Server. Selain itu terdapat pilihan service yang lain seperti MySQL Server dan FileZilla Server.



Gambar 21: Menjalankan service Apache Web Server dengan "XAMPP Control Panel"

Untuk melihat bahwa service Apache Web Server telah berhasil dijalankan, Anda dapat membukanya melalui browser dengan alamat http://localhost.

2.3.3.Konfigurasi PHP Oci Extension Paket XAMPP di Windows

Agar PHP dapat berkomunikasi dengan Oracle, maka diperlukan library tambahan yaitu PHP Oci8 Extension. Hal ini dapat dilakukan dengan mengaktifkan module PHP Oci Extension (php_oci8.dll) pada file konfigurasi php.ini. Berikut adalah contoh tahapan untuk mengaktifkan PHP Oci Extension pada paket XAMPP di Windows.

1. Buka file php.ini yang dapat ditemukan di folder: `path to xampp/apache/bin/php.ini`.
2. Aktifkan modul PHP Oci Extension dengan mengubah konfigurasi dari php.ini.

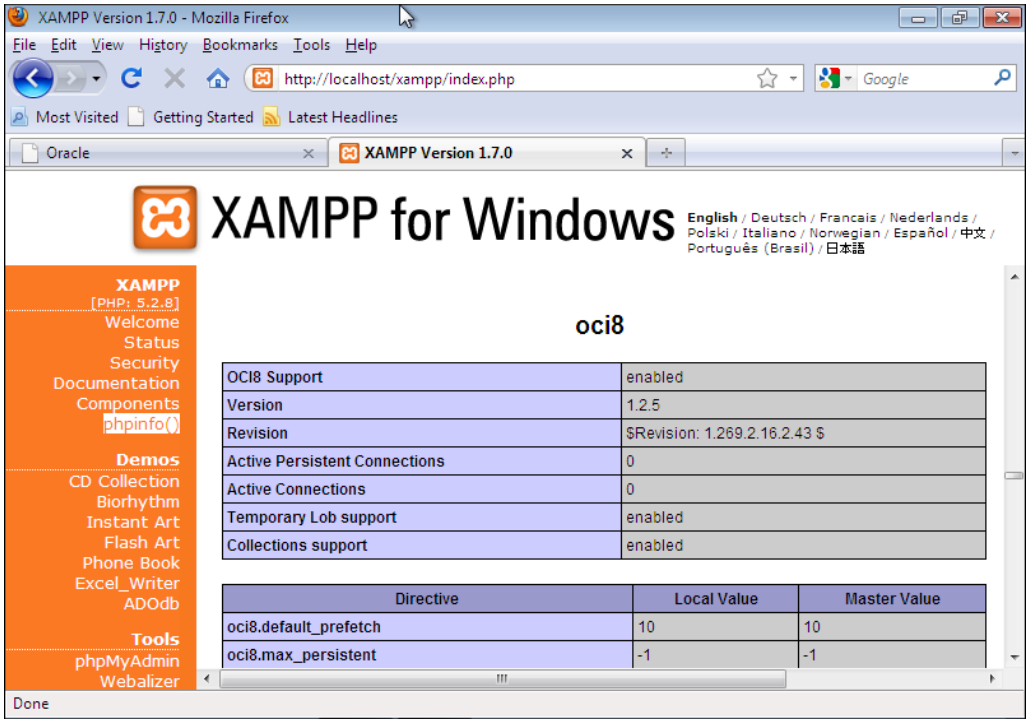
   Sebelum :

- `;extension=php_oci8.dll`
- `;extension=php_pdo_oci8.dll`

Sesudah :

- `extension=php_oci8.dll`
- `extension=php_pdo_oci8.dll`

3. Setelah Anda selesai mengubah file konfigurasi PHP, Anda harus me-restart service Apache Web Server agar perubahan tersebut dapat berjalan. Jika berhasil, maka Anda dapat mengecek dengan melihat informasi phpinfo di XAMPP.



Gambar 22: PHP Oci Extension telah berhasil di install

2.4. Installasi OLAP (Oci8 Extension-Linux-Apache-PHP)

2.4.1.Installasi Oracle InstantClient di Linux

Berbeda dengan installasi oracle instantclient di Windows yang hanya memerlukan paket "oracle instantclient basic", installasi oracle instantclient di Linux membutuhkan tambahan paket "oracle instantclient devel". Ikuti langkah-langkah berikut untuk menginstall oracle instantclient di Linux.

1. Download paket oracle instantclient melalui url yang telah disediakan.

   Url Download : http://dev.oit.ugm.ac.id

   Unduh file :

   - `oracle-instantclient-basic-10.2.0.4-1.i386.zip`
   - `oracle-instantclient-devel-10.2.0.4-1.i386.zip`

2. Install paket oracle instantclient dengan tahapan sebagai berikut.

   - Buat direktori `/opt/oracle_client/`

     ```
     $ mkdir /opt/oracle_client
     ```

   - Copy paket oracle instantclient ke direktori `/opt/oracle_client/`

     ```
     $ cp oracle-instantclient-basic-10.2.0.4-1.i386.zip /opt/oracle_client
     $ cp oracle-instantclient-devel-10.2.0.4-1.i386.zip /opt/oracle_client
     ```

   - Extract paket oracle instantclient ke direktori `/opt/oracle_client/`

     ```
     $ unzip oracle-instantclient-basic-10.2.0.4-1.i386.zip
     $ unzip oracle-instantclient-devel-10.2.0.4-1.i386.zip
     ```

- Buat symlink libclntsh.so dan libocci.so

  Masuk ke direktori oracle instantclient kemudian buat symlink.

  ```
  $ cd /opt/oracle_client/instantclient_10_2/
  $ ln -s libclntsh.so.10.1 libclntsh.so
  $ ln -s libocci.so.10.1 libocci.so
  ```

2.4.2.Installasi XAMPP di Linux

Tahapan installasi paket XAMPP di Linux sangatlah sederhana. Ikuti langkah-langkah berikut untuk menginstall paket XAMPP di Linux.

1. Download paket XAMPP melalui url yang telah disediakan.

   Url Download : http://dev.oit.ugm.ac.id

   Undul file :

   - `xampp-linux-1.7.tar.gz`

2. Extract file paket XAMPP tersebut ke direktori `/opt`.

   ```
   $ tar xvfz xampp-linux-1.7.tar.gz -C /opt
   ```

3. Menjalankan service Apache Web Server paket XAMPP (dengan user root).

   ```
   $ /opt/lampp/lampp startapache
   ```

4. Paket XAMPP telah selesai diinstall di sistem operasi Linux kesayangan Anda. Untuk mengecek, Anda dapat membukanya melalui browser dengan alamat http://localhost.

2.4.3.Konfigurasi PHP Oci Extension Paket XAMPP di Linux

Berbeda dengan konfigurasi PHP Oci Extension paket XAMPP di Windows yang membutuhkan beberapa tahapan, konfigurasi PHP Oci Extension pada paket XAMPP di Linux hanya membutuhkan satu langkah saja, yaitu dengan memasukkan letak installasi oracle instantclient pada perintah pengaktifan PHP Oci Extension (`lampp oci8`). Lakukan perintah berikut dengan user root untuk mengaktifkan PHP Oci Extension pada paket XAMPP di Linux.

```
$ /opt/lampp/lampp oci8

Please enter the path to your Oracle or Instant Client installation:

[/opt/oracle] /opt/oracle_client/instantclient_10_2

installing symlink...

patching php.ini...

OCI8 add-on activation likely successful.
```

3. SESI III

3.1. Dasar Pemrograman PHP Oracle

Tujuan Pembelajaran Khusus

- Peserta WorkShop dapat mengerti dan memahami serta mempraktikan dasar-dasar pemrograman PHP Oracle dengan menggunakan PHP Oci Extension dan library ADODB dari PHPLens

3.2. Pendahuluan

Modul ini dibuat untuk programmer PHP yang membuat aplikasi menggunakan database Oracle. Modul ini berisi dasar- dasar pemrograman PHP menggunakan Oracle baik menggunakan PHP Oci Extension maupun library ADODB. Penulis berasumsi peserta sudah menguasai dasar pemrograman PHP dan SQL.

Database Oracle terkenal karena skalabilitas, reliabilitas, dan fiturnya serta dapat digunakan pada beragam platform. Terdapat beberapa perbedaan istilah yang digunakan saat menjelaskan database Oracle dan sebuah database dari vendor lain. Berikut beberapa istilah penting dalam Oracle yang dapat membantu Anda untuk memahami istilah-istilah dalam Oracle.

Database dan Instance

Kata database dalam Oracle memiliki makna Database Management System dan sering disebut database saja. Database digunakan untuk menyimpan dan mengirimkan data. Setiap database terdiri dari satu atau banyak datafile. Instance adalah kumpulan dari oracle process dan alokasi memori yang ada di oracle disebut sebagai System Global Area (SGA) yang digunakan unuk mengakses informasi-informasi yang tersimpan pada database. User tidak dapat mengakses secara langsung informasi yang tersimpan dalam database tanpa melalui instance. Jika instance sedang drop, semua informasi yang ada pada database tidak dapat diakses melalui level operating system.

Tablespace

Tablespace adalah tempat penyimpanan objek database secara logical. Objek database tidak dapat menempati lebih dari satu tablespace. Tablespace itu sendiri dapat mempunyai lebih dari satu

datafiles (tempat penyimpanan data pada Oracle). Sebuah konfigurasi database yang paling sederhanahanya memiliki dua datafiles. Yang pertama adalah SYSTEM datafiles, yaitu untuk menyimpan data dictionary, dan yang kedua adalah USER datafiles untuk menyimpan segment-segment yang lain.

Schema dan User

Schema adalah sekumpulan dari objek database, misalnya tabel dan index. Dalam Oracle schema dimiliki oleh seorang user database yang mempunyai nama (usernamenya) sama dengan nama schema tersebut. Secara default user hanya mempunyai hak akses (permission) terhadap schema yang dimiliki saja. Meskipun begitu, user mempunyai hak untuk memberikan GRANT kepada user database lain untuk mengakses schema miliknya.

3.2.1.Menggunakan Schema HR

Secara default Oracle XE terdapat schema HR yang dapat kita gunakan untuk belajar dasar pemrograman PHP menggunakan database Oracle. Namun kita tidak bisa langsung menggunakan schema tersebut karena secara default schema ini di lock sehingga untuk menggunakan schema ini login terlebih dahulu menggunakan username sys atau system pada Oracle Database XE Home Page, atau secara default terdapat pada halaman http://127.0.0.1:8080/apex.

Gambar 23: Schema HR

Setelah berhasil masuk ke dalam database buka menu Administration > Manage Database Users lalu pilih schema HR. Pastikan tidak ada centang pada checkbox Expire Password dan Account Status pada nilai Unlocked. Ketikkan hr pada textbox password lalu tekan tombol alter user. Anda dapat mengisi textbox password sesuai dengan password yang anda inginkan. Sekarang schema HR

sudah dapat digunakan.

3.2.2.Koneksi Database

Sebelum melakukan manipulasi data di dalam database pertama kita melakukan koneksi terlebih dahulu. Ada beberapa cara yang dapat digunakan untuk melakukan koneksi baik menggunakan Oci Extension maupun Library Adodb.

Koneksi standar

Oci8	Adodb
$conn = oci_connect ($user,$pass, $connStr)	$conn->Connect($user,$pass, $connStr);

Unique Connection

Oci8	Adodb
$conn = oci_new_connect ($user,$pass, $connStr)	$conn->NConnect($user,$pass, $connStr);

Persistent Connection

Oci8	Adodb
$conn = oci_pconnect ($user,$pass, $connStr)	$conn->PConnect($user,$pass, $connStr);

3.2.3.Database Connection String

Easy Connection String

Menggunakan parameter input berupa [//]hostname[:port][/service_name]. Misalnya //localhost:1521/xe.

Oci8	Adodb
$conn = oci_connect ($user,$pass, $connStr)	$conn->Connect($user,$pass, $connStr);

Database Alias

Pada folder instalasi Oracle XE terdapat file tnsname.ora yang terdapat pada folder instalasi oracle. Anda dapat menggunakan alias untuk parameter service name yang terdapat yang terdapat pada tnsname.ora untuk melakukan koneksi. Biasanya digunakan untuk koneksi remote server.

Windows	$ORACLE_HOME\oraclexe\app\oracle\product\10.2.0\server\NETWORK\ADMIN
Linux	$ORACLE_HOME/network/admin

```
XE =
(DESCRIPTION = (ADDRESS = (PROTOCOL = TCP)(HOST = localhost)(PORT = 1521))
        (CONNECT_DATA = (SERVER = DEDICATED)
         (SERVICE_NAME = XE))
)
```

Sehingga dapat dijalankan koneksi dengan oci_connect(‘username’,’password’,’xe’). Atau

$conn->Connect(‘username’,’password’,’xe’);

Full Database Connection String

Dapat juga dijalankan koneksi dengan parameter input :

```
$dbci =
' (DESCRIPTION =
(ADDRESS = (PROTOCOL = TCP)(HOST = sample2.com)(PORT = 1521))
                (CONNECT_DATA =
                (SERVER = DEDICATED)
(SERVICE_NAME = XE)
      )
)';
```

Sehingga dapat dijalankan koneksi dengan $conn = oci_connect(‘username’, 'password', $dbci); atau bisa juga dengan $conn->Connect(‘username’,’password’,$dbci);

3.2.4. Oci8 Connection

Buatlah file **define_conn.php** yang berisi parameter yang diperlukan untuk melakukan koneksi database oracle.

```
<?php // File: define_conn.php
define('DB_USER', 'hr'); // User name
define('DB_PASS', 'hr'); // Password
define('DB_HOST', 'localhost'); // Host
define('DB_PORT', '1521'); //Port
define('DB_SID', 'xe'); //SID
?>
```

Setelah mendefinisikan parameter untuk koneksi database kita buat skrip untuk menampilkan data secara sederhana menggunakan oci extension terlebih dahulu. Buat file **oci8_conn.php** untuk membuat koneksi database.

```
//menginclude kan parameter
include "define_conn.php";

//string koneksi menggunakan easy connecting method
$connStr = "//".DB_HOST.":".DB_PORT."/".DB_SID;

//koneksi dengan connect descriptor
if(!$conn = oci_connect(DB_USER,DB_PASS,$connStr)) {
   $err =      oci_error();

   //memicu error yang menghentikan execution query
   trigger_error('Koneksi gagal : '.
           $err['message'], E_USER_ERROR);
};

//query menampilkan waktu dalam char menggunakan table dual
$sql = "SELECT TO_CHAR(SYSDATE, 'HH:MM:SS') waktu FROM DUAL";

//menyiapkan statement, sebelum di eksekusi
$stmt = oci_parse($conn,$sql);

//statement dieksekusi, return value (true/false)
if (!oci_execute($stmt)) {
   $err = oci_error($stmt);
```

```
   trigger_error('Query gagal : ' .
            $err['message'], E_USER_ERROR);
};

//Menampilkan row
oci_fetch($stmt);

//Menampilkan isi kolom
//Oracle mengembalikan nama kolom dalam UPPERCASE
$rslt = oci_result($stmt, 'WAKTU');

print "<h3>Sekarang menunjukkan pukul   ".$rslt."</h3>";
```

Skrip di atas menghasilkan keluaran pada browser seperti berikut :

Gambar 24: Oci8 Connection

Dapat juga mengganti variabel connection string menggunakan full database connection string method.

```
//string koneksi menggunakan full database connection string
$connStr =
"(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=".DB_HOST.")(PORT=".DB_PORT."))
   (CONNECT_DATA=(SERVICE_NAME=".DB_SID.")))";
```

3.2.5. AdodbConnection

ADOdb adalah kumpulan pustaka program (library) database untuk bahasa pemrograman PHP dan Python yang dikembangkan berdasarkan konsep ActiveX Data Objects (ADO) milik Microsoft. Dengan ADOdb memungkinkan pengembang software (programmer) menuliskan kode program untuk aplikasi yang dibuat menjadi lebih konsisten dalam berkomunikasi dengan sebuah database. Keuntungan utama dari menggunakan pustaka ADOdb adalah perubahan database pada sisi aplikasi

dimungkinkan dengan minimnya perubahan kode program atau perubahan kode program hanya terjadi pada baris kode untuk koneksi ke database saja

Dengan menggunakan library adodb kita juga dapat menampilkan seperti meggunakan oci extension di atas. Library adodb dapat di unduh di http://sourceforge.net/projects/adodb/files/ . Buatlah file **adodb_conn.php.** Copy library adodb yang telah diunduh ke dalam folder.

```
//mengincludekan library adodb
require_once "$PATH_ADODB_LIBRARY/adodb5/adodb.inc.php";
//mengincludekan parameten koneksi
include "define_conn.php";
//set driver oci8
$db = NewADOConnection('oci8');
```

Setelah itu buatlah objek koneksi dengan menggunakan parameter yang telah dipanggil. Koneksi yang akan dilakukan dapat dengan easy connectiong method maupun full database connection string method seperti berikut.

```
//objek koneksi dengan parameter easy connecting method
if (!$conn = $db->Connect(DB_HOST,DB_USER,DB_PASS,DB_SID)){
   die("Connection failed");
}
```

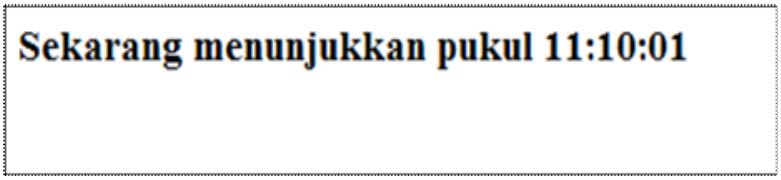
Selain menggunakan easy connecting method dapat juga menggunakan full database connection string seperti berikut.

```
//string koneksi
$connStr = "(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=".DB_HOST.")
(PORT=".DB_PORT."))
           (CONNECT_DATA=(SID=".DB_SID.")))";

//koneksi dengan parameter connect descriptor
if (!$conn = $db->Connect($connStr, DB_USER, DB_PASS)){
   die ('Koneksi gagal : '.$db->ErrorMsg());
}
```

Selanjutnya membuat SQL untuk menampilkan data waktu saat ini serta menampilkanya ke dalam skrip php seperti berikut.

```
/query
$sql = "SELECT TO_CHAR(SYSDATE, 'HH:MM:SS') waktu FROM DUAL";

//set fetchmode param ADODB_FETCH_ASSOC / ADODB_FETCH_NUM
$db->SetFetchMode(ADODB_FETCH_ASSOC);

//execute sql
if (!$rs = $db->Execute($sql)){
   die("Query gagal : ".$db->ErrorMsg());
}

//Menampilkan isi kolom
//Oracle mengembalikan nama kolom dalam UPPERCASE
$row = $rs->FetchRow();
print "<h3>Sekarang menunjukkan pukul    ".$row['WAKTU']."</h3>";
```

Skrip di atas menghasilkan keluaran pada browser seperti berikut :



Gambar 25: Adodb Connection

3.3. Menampilkan data

3.3.1. Menampilkan data menggunakan Oci Extension

Menampilkan data dengan menggunakan oci extension dapat dilakukan dengan beberapa langkah seperti di bawah ini.

1. **Parse (*)** : Menyiapkan statement sebelum dieksekusi. Pesan error muncul pada tahap eksekusi, sedangkan pada tahap ini tidak mengeluarakan pesan error.

2. **Bind (?)** : Melakukan binding data ke dalam variabel pada statement SQL, biasanya digunakan untuk keamanan.

3. **Define (?)** : Menentukan variabel yang digunakan untuk menampilkan data.

4. **Execute (*)** : Mengirimkan SQL untuk diproses Oracle dan menghasilkan suatu keluaran.

5. **Fetch (*)** : Menampilkan data dari database.

Keterangan :

(*) = dibutuhkan untuk menampilkan data

(?) = dapat memilih akan digunakan atau tidak

Sebelum memulai pertama tambahkan pada skrip **define_conn.php** dengan beberapa fungsi yang akan kita gunakan.

```
function print_header($title) {
   $header =  "<h2>".strtoupper($title)."</h2><hr>";
   echo $header;
}

function print_footer() {
   $footer =  "<hr> Tim Integrasi : Oracle & GTFW Workshop - ".date("d F Y");
   echo $footer;
}
```

Buatlah file **oci8_func.php** yang berisi fungsi oci extension yang akan kita gunakan untuk melakukan koneksi ke database.

```
//mengincludekan file parameter koneksi
require_once 'define_conn.php';

//fungsi koneksi database
function oci8_conn() {

 $connStr =
 "(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=".DB_HOST.")(PORT=".DB_PORT."))
   (CONNECT_DATA=(SERVICE_NAME=".DB_SID.")))";

 if(!$conn = oci_connect( DB_USER , DB_PASS , $connStr )) {
   $err =      oci_error();
   trigger_error( 'Koneksi gagal : '.
```

```
   $err['message'], E_USER_ERROR );
  };
  return $conn;
}
```

Buatlah file **oci8_binding.php** yang akan kita gunakan untuk menampilkan data yang kita ambil dari database.

```
//mengincludekan fungsi
include 'oci8_func.php';
//koneksi ke database
$conn = oci8_conn();

$sql  = 'SELECT * FROM DEPARTMENTS WHERE LOCATION_ID = :location';

//parameter location
$location = '1700';

$stmt = oci_parse ( $conn, $sql );
oci_bind_by_name($stmt, ':location', $location);
oci_define_by_name( $stmt , 'DEPARTMENT_ID', $deptno );
oci_define_by_name( $stmt , 'DEPARTMENT_NAME', $deptname );
oci_define_by_name( $stmt , 'MANAGER_ID', $deptmanager );
oci_define_by_name( $stmt , 'LOCATION_ID', $deptloc );
oci_execute($stmt);

if (!oci_execute($stmt, OCI_DEFAULT)) {
  $err = oci_error($stmt);
  trigger_error('Query failed: ' . $err['message'], E_USER_ERROR);
}
print_header('departemen');
print '<font face="Arial">';
print "<h5>Data Departement yang mempunyai ID lokasi : ".$row."</h5>";

print '<table border="1">';
print '<th>DEPT_ID</th>
    <th>DEPT NAME</th>
    <th>MANAGER_ID</th>
    <th>LOCATION_ID</th>';
while (oci_fetch_array($stmt, OCI_NUM+OCI_RETURN_NULLS)) {
  print '<tr>';
  print '<td>'.$deptno.'</td>';
  print '<td>'.$deptname.'</td>';
  print '<td>'.$deptmanager.'</td>';
  print '<td>'.$deptloc.'</td>';
  print '</tr>';
}
```

```
print '</table>';
print '</font>';
print_footer();
```

Fungsi **oci_bind_by_name** digunakan untuk menghubungkan variabel php yaitu `$location` dengan placeholder oracle `:location` yang telah didefinisikan dengan nilai 1700. Menggunakan variabel binding mempunyai dua kelebihan yaitu mencegah SQL injection serta untuk meningkatkan performa dalam eksekusi SQL karena hanya dijalankan satu kali saja, dengan cara merubah variabel bindingnya sebagai parameter input. Sehingga cenderung lebih cepat daripada tidak menggunakan variabel binding.

Fungsi **oci_define_by_name** digunakan untuk mendefinisikan variabel yang digunakan untuk menampilkan data pada kolom tertentu. Biasanya digunakan untuk membuat skrip menjadi menjadi mudah dibaca. Misalnya pada kolom `'DEPARTMENT_ID'` didefinisikan menjadi variabel dengan nama `$deptno`, sehingga saat akan menampilkan data penulisan pada skrip menjadi mengikuti variabel `$deptno`.

Fungsi **oci_execute** digunakan untuk mengeksekusi statement, yang akan mengembalikan nilai menjadi pesan error atau keluaran data yang dapat ditampilkan menggunakan fungsi oci_fetch_array**.**

Fungsi **oci_fetch_array** tiap baris dari statement berupa associative array, numerically indexed array, maupun keduanya targantung dari result_type yang diinputkan misalnya seperti berikut ini :

OCI_ASSOC : menghasilkan baris dengan assosiatif array sesuai dengan nama kolom pada tabel, misalnya `$data['DEPARTMENT_ID']`, `$data['DEPARTMENT_NAME']`

OCI_NUM : Menghasilkan baris dengan numerical index array, misalnya `$data[0]`, `$data[1]`

OCI_BOTH : Menghasilkan baris dengan numerical index array maupun asosiatif array, misalnya `$data['DEPARTMENT_ID']`, `$data[1]`

OCI_RETURN_NULLS : Menghasilkan elemen kosong untuk kolom dengan nilai NULL

OCI_RETURN_LOBS :Menghasilkan nilai dari kolom berupa menjadi LOB, misalnya

```
$row['LOB_FIELD']
```

3.3.2. Menampilkan data menggunakan Library Adodb

Buatlah file **adodb_binding.php** yang akan kita gunakan untuk menampilkan data yang kita ambil dari database menggunakan library adodb.

```
include "adodb_func.php";

$db = NewADOConnection("oci8");
$db->Connect('localhost', "hr", "hr");

$rs = $db->Execute("SELECT * FROM DEPARTMENTS WHERE LOCATION_ID = :location",
         array('location' => 1700));

print_header('departemen');
print '<font face="Arial">';
print "<h5>Data Departement yang mempunyai ID lokasi : ".$row."</h5>";

print '<table border="1">';
print '<th>DEPT_ID</th>
    <th>DEPT NAME</th>
    <th>MANAGER_ID</th>
    <th>LOCATION_ID</th>';
while ($arr = $rs->FetchRow()) {
 print '<tr>';
 print '<td>'.$arr[0].'</td>';
 print '<td>'.$arr['DEPARTMENT_NAME'].'</td>';
 print '<td>'.$arr['MANAGER_ID'].'</td>';
 print '<td>'.$arr['LOCATION_ID'].'</td>';
 print '</tr>';
}
print '</table>';
print '</font>';
print_footer();
```

Fungsi **Execute** pada adodb digunakan untuk mengeksekusi statement SQL. Fungsi **FetchRow** digunakan untuk menampilkan data.

3.4. CRUD (CREATE READ UPDATE DELETE)

3.4.1. CRUD menggunakan Oci Extension

Setelah berhasil menampilkan data seperti di atas kita dapat melanjutkan dengan proses manipulasi data yaitu create, update, dan delete data. Pertama tambahkanlah beberapa kode seperti di bawah ini pada file **oci8_fetch_array.**

```
include 'oci8_func.php';

$conn = oci8_conn();
$sql  = 'SELECT * FROM DEPARTMENTS WHERE LOCATION_ID = 1700';
$stmt = oci8_exec( $conn, $sql);
$row  = oci8_fetch_row( $stmt, 'LOCATION_ID');
//$data = oci8_fetch_array($stmt, OCI_ASSOC+OCI_RETURN_NULLS );
print_header('departemen');
echo "<h5>Data Departement yang mempunyai ID lokasi : ".$row."</h5>";
echo "<a href='oci8_create.php'/>Create</a>";

echo '<table border="1"><tr><td><b>Id</b></td><td><b>Nama
Departemen</b></td><td><b>Id Manager</b></td><td><b>Id
Lokasi</b></td><td><b>Aksi</b></td></tr>';
   while ($item = oci_fetch_array($stmt, OCI_ASSOC+OCI_RETURN_NULLS)) {
       echo '<tr>';
       echo '<td>'.$item['DEPARTMENT_ID'].'</td>';
       echo '<td>'.$item['DEPARTMENT_NAME'].'</td>';
       echo '<td>'.$item['MANAGER_ID'].'</td>';
       echo '<td>'.$item['LOCATION_ID'].'</td>';
       echo '<td><a href="oci8_create.php?id='.
$item['DEPARTMENT_ID'].'&mode=update" />update</a> | <a href="oci8_sql.php?
id='.$item['DEPARTMENT_ID'].'&mode=delete" />delete</a></td>';
       echo '</tr>';
   }
   echo '</table>';
print_footer();
```

Buatlah file **oci8_create.php** untuk menampilkan form input data department seperti pada berikut ini.

```
<?php
include('oci8_func.php');
```

```
if (isset($_GET[id])){
   $id = $_GET[id];
   $conn = oci8_conn();
   $sql  = "SELECT * FROM DEPARTMENTS WHERE DEPARTMENT_ID = :id";

   $stmt = oci_parse($conn, $sql);
   oci_bind_by_name($stmt,':id', $id);
   oci_execute($stmt, OCI_DEFAULT );
   $row  = oci_fetch_array($stmt, OCI_NUM);
}

print_header('DEPARTEMEN');
print '<font face="Arial">';
if (isset($_GET[id])) {echo "<h5>Update Data</h5>"; }else{echo "<h5>Create
Data</h5>";}
?>
<form action="<?php if (isset($_GET[id])) {echo "oci8_sql.php?mode=update"; }
else{echo "oci8_sql.php?mode=create";}?>" method="post" name="create">
<table width="200" border="1">
 <input type="hidden" name="id">
 <tr>
  <th scope="row">DEPT_ID</th>
  <td><input name="Dept_id" type="text" <?php if (isset($_GET[id])) {echo
"readonly"; }else{echo "";}?> value="<?php echo $row[0];?>"  id="dept_id"
maxlength="4" /></td>
 </tr>
 <tr>
  <th scope="row">DEPT_NAME</th>
  <td><input name="Dept_name" type="text" value="<?php echo $row[1];?>"
id="dept_name" maxlength="25" /></td>
 </tr>
 <tr>
  <th scope="row">MANAGER_ID</th>
  <td><input name="Manager_id" type="text" value="<?php echo $row[2];?
>"id="manager_id" maxlength="4" /></td>
 </tr>
 <tr>
  <th scope="row">LOCATION_ID</th>
  <td><input name="Location_id" type="text" value="<?php echo $row[3];?
>"id="location_id" maxlength="4" /></td>
 </tr>
  <tr>
  <th colspan="2" scope="row" style="text-align:right;"><input type="submit"
name="submit" id="submit" value="<?php if (isset($_GET[id])) {echo "Update"; }
else{echo "Create";}?>" /></th>
 </tr>
</table>
</form>
```

```
<?php
print_footer();
?>
```

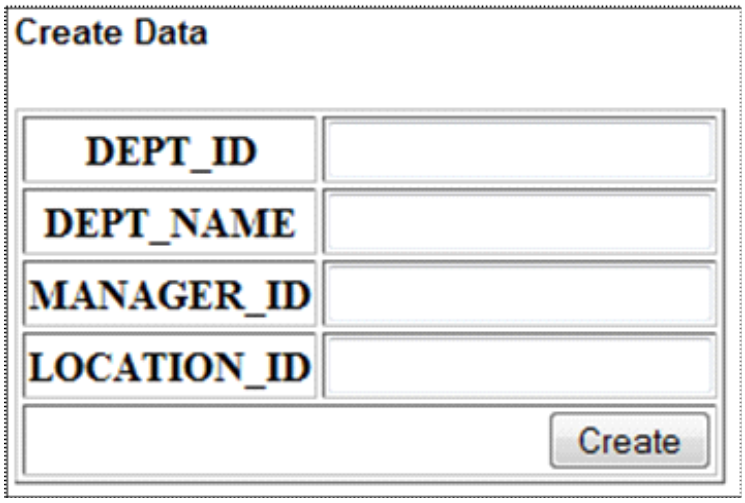
Pada tampilan di atas akan menampilkan data jika digunakan dalam proses update data. Dan jika dalam proses create maka akan menampilkan tampilan form input saja. Selanjutnya buat file dengan nama **oci8_sql.php** yang berisi query untuk melakukan manipulasi data berupa create, update, dan delete.

```
include('oci8_func.php');

$mode = $_GET['mode'];

$conn = oci8_conn();

$Dept_id = $_POST['Dept_id'];
$Dept_name = $_POST['Dept_name'];
$Manager_id = $_POST['Manager_id'];
$Location_id = $_POST['Location_id'];

switch ($mode){
   case 'create':

      if (isset($_POST['submit']))
      {
         // Insert
         $stmt = oci_parse($conn,
            "INSERT INTO
               DEPARTMENTS (DEPARTMENT_ID, DEPARTMENT_NAME, MANAGER_ID,
LOCATION_ID)
            VALUES (:Dept_id, :Dept_name, :Manager_id, :Location_id)");

         oci_bind_by_name($stmt, ':Dept_id', $Dept_id);
         oci_bind_by_name($stmt, ':Dept_name', $Dept_name);
         oci_bind_by_name($stmt, ':Manager_id', $Manager_id);
         oci_bind_by_name($stmt, ':Location_id', $Location_id);

         $result = oci_execute($stmt);

         // Display an appropriate message
         if ($result)
         {
```

```
            echo "<p>Data berhasil di buat!</p>";
            $commit = oci_commit($conn);
            if (!$commit) {
               $error = oci_error($conn);
               echo 'Commit failed. Oracle reports: ' . $error['message'];
            }
         }
         else
         {
            echo "<p>Data gagal dimasukkan!</p>";
            var_dump(oci_error($stmt));
         }

         oci_close($conn);
      }
      break;

   case 'update':
      // Insert
      $stmt = oci_parse($conn,
            "UPDATE DEPARTMENTS
               SET
                  DEPARTMENT_NAME = :Dept_name,
                  MANAGER_ID = :Manager_id,
                  LOCATION_ID = :Location_id
               WHERE
                  DEPARTMENT_ID = :Dept_id");

      oci_bind_by_name($stmt, ':Dept_id', $Dept_id);
      oci_bind_by_name($stmt, ':Dept_name', $Dept_name);
      oci_bind_by_name($stmt, ':Manager_id', $Manager_id);
      oci_bind_by_name($stmt, ':Location_id', $Location_id);

      $result = oci_execute($stmt);

      // Display an appropriate message
      if ($result)
      {
         echo "<p>Data berhasil di ubah!</p>";
         $commit = oci_commit($conn);
         if (!$commit) {
            $error = oci_error($conn);
            echo 'Commit failed. Oracle reports: ' . $error['message'];
         }
      }
      else
      {
```

```
        echo "<p>Data gagal dimasukkan!</p>";
        var_dump(oci_error($stmt));
      }

      oci_close($conn);


      break;
   case 'delete':

      $Dept_id = $_GET[id];

      // Insert
      $stmt = oci_parse($conn,
            "DELETE FROM DEPARTMENTS
             WHERE      DEPARTMENT_ID = :Dept_id");

      oci_bind_by_name($stmt, ':Dept_id', $Dept_id);
      $result = oci_execute($stmt);


      // Display an appropriate message
      if ($result)
      {
        echo "<p>Data berhasil di hapus!</p>";
        $commit = oci_commit($conn);
        if (!$commit) {
           $error = oci_error($conn);
           echo 'Commit failed. Oracle reports: ' . $error['message'];
        }
      }
      else
      {
        echo "<p>Data gagal dihapus!</p>";
        var_dump(oci_error($stmt));
      }

      oci_close($conn);
      break;
}
```

Dari skrip di atas dapat dilihat tampilan berupa :



Gambar 26: Create Data Oci8

3.4.2. Menampilkan data menggunakan Adodb Library

Ubahlah file **adodb_binding.php** dan tambahkan link untuk menampilkan form input dan link untuk mengubah dan menghapus menjadi seperti di bawah ini.

```
<?php
include "adodb_func.php";

$db = NewADOConnection("oci8");
if (!$conn = $db->Connect( $connStr, DB_USER, DB_PASS )){
     die ( 'Koneksi gagal : '.$db->ErrorMsg() );
}
$rs = $db->Execute("SELECT * FROM DEPARTMENTS WHERE LOCATION_ID = :location",
array('location' => 1700));

print_header('departemen');
print '<font face="Arial">';
print "<h5>Data Departement yang mempunyai ID lokasi : 1700 </h5>";
echo "<a href='adodb_create.php'/>Create</a>";

print '<table border="1">';
print '<th>DEPT_ID</th>
    <th>DEPT NAME</th>
    <th>MANAGER_ID</th>
    <th>LOCATION_ID</th>
    <th>Aksi</th>';
while ($arr = $rs->FetchRow()) {
   print '<tr>';
   print '<td>'.$arr['DEPARTMENT_ID'].'</td>';
```

```
   print '<td>'.$arr['DEPARTMENT_NAME'].'</td>';
   print '<td>'.$arr['MANAGER_ID'].'</td>';
   print '<td>'.$arr['LOCATION_ID'].'</td>';
   echo '<td><a href="adodb_create.php?id='.
$arr['DEPARTMENT_ID'].'&mode=update" />update</a> | <a href="adodb_sql.php?
id='.$arr['DEPARTMENT_ID'].'&mode=delete" />delete</a></td>';
   print '</tr>';
}
print '</table>';
print '</font>';
print_footer();
```

Setelah menambahkan link untuk menampilkan form input selanjutnya kita buat file dengan nama **adodb_create.php** untuk menampilkan form input dan update data seperti di bawah ini.

```
<?php
include('adodb_func.php');

$db = NewADOConnection("oci8");
if (!$conn = $db->Connect( $connStr, DB_USER, DB_PASS )){
     die ( 'Koneksi gagal : '.$db->ErrorMsg() );
}
if (isset($_GET[id])){
   $id = $_GET[id];
   $row = $db->GetRow("SELECT * FROM DEPARTMENTS WHERE DEPARTMENT_ID = :id",
array('id' => $id));

}

print_header('DEPARTEMEN');
print '<font face="Arial">';
if (isset($_GET[id])) {echo "<h5>Update Data</h5>"; }else{echo "<h5>Create
Data</h5>";}
?>
<form action="<?php if (isset($_GET[id])) {echo "adodb_sql.php?mode=update"; }
else{echo "adodb_sql.php?mode=create";}?>" method="post" name="create">
<table width="200" border="1">
 <input type="hidden" name="id">
 <tr>
  <th scope="row">DEPT_ID</th>
  <td><input name="Dept_id" type="text" <?php if (isset($_GET[id])) {echo
"readonly"; }else{echo "";}?> value="<?php echo $row[0];?>"  id="dept_id"
maxlength="4" /></td>
 </tr>
```

```
 <tr>
  <th scope="row">DEPT_NAME</th>
  <td><input name="Dept_name" type="text" value="<?php echo
$row['DEPARTMENT_NAME'];?>" id="dept_name" maxlength="25" /></td>
 </tr>
 <tr>
  <th scope="row">MANAGER_ID</th>
  <td><input name="Manager_id" type="text" value="<?php echo $row[2];?
>"id="manager_id" maxlength="4" /></td>
 </tr>
 <tr>
  <th scope="row">LOCATION_ID</th>
  <td><input name="Location_id" type="text" value="<?php echo $row[3];?
>"id="location_id" maxlength="4" /></td>
 </tr>
  <tr>
  <th colspan="2" scope="row" style="text-align:right;"><input type="submit"
name="submit" id="submit" value="<?php if (isset($_GET[id])) {echo "Update"; }
else{echo "Create";}?>" /></th>
 </tr>
</table>
</form>
<?php
print_footer();
?>
```

Setelah itu kita membuat file **adodb_sql.php** yang berisi query untuk memanipulasi data menggunakan library adodb.

```
<?php

include 'adodb_func.php';

$mode = $_GET['mode'];

$db = NewADOConnection('oci8');

if (!$conn = $db->Connect( $connStr, DB_USER, DB_PASS ) ) {
   die ('Koneksi Gagal : '. $db->ErrorMsg());
}
$Dept_id = $_POST['Dept_id'];
$Dept_name = $_POST['Dept_name'];
$Manager_id = $_POST['Manager_id'];
$Location_id = $_POST['Location_id'];

switch ($mode) {
```

```
   case 'create':
      $sql = "INSERT INTO
                DEPARTMENTS (DEPARTMENT_ID,DEPARTMENT_NAME, MANAGER_ID,
LOCATION_ID)
              VALUES (:Dept_id, :Dept_name, :Manager_id, :Location_id)";
      $result = $db->Execute($sql,
         array(
            'Dept_id'=>$Dept_id,
            'Dept_name'=>$Dept_name,
            'Manager_id'=>$Manger_id,
            'Location_id'=>$Location_id
         ));
      if ($result){
         echo 'Data berhasil ditambahkan!';
      }else{
         echo 'Data gagal ditambahkan'.$db->ErrorMsg();
      }
      break;

   case 'update':
      $sql = "UPDATE DEPARTMENTS
              SET
                 DEPARTMENT_NAME = :Dept_name,
                 MANAGER_ID = :Manager_id,
                 LOCATION_ID = :Location_id
              WHERE
                 DEPARTMENT_ID = :Dept_id";
      $result= $db->Execute($sql,
         array(
            'Dept_id'=>$Dept_id,
            'Dept_name'=>$Dept_name,
            'Manager_id'=>$Manger_id,
            'Location_id'=>$Location_id
         ));
      if ($result){
         echo 'Data berhasil diupdate!';
      }else{
         echo 'Data gagal diupdate'.$db->ErrorMsg();
      }
      break;

   case 'delete':
      $Dept_id = $_GET[id];
      $sql = "DELETE FROM DEPARTMENTS
            WHERE      DEPARTMENT_ID = :Dept_id";
      $result = $db->Execute($sql,array('Dept_id'=>$Dept_id));
      if ($result){
         echo 'Data berhasil dihapus!';
```

```
      }else{
         echo 'Data gagal dihapus'.$db->ErrorMsg();
      }
      break;
}
```

Hari ke 2, Rabu 13 Oktober 2010

4. SESI IV

4.1. Installasi GTFW UGM

Tujuan Pembelajaran Khusus

- Peserta Workshop secara mandiri dapat melakukan installasi GTFW UGM baik di simtem operasi Windows maupun Linux.

4.1.1.Pendahuluan GTFW 3 UGM

Gamatechno Web Application Development Framework 3 UGM (GTFW 3 UGM) adalah suatu kerangka kerja untuk mengembangkan aplikasi berbasis web yang dikembangkan oleh UGM dan P.T. Gamatechno Indonesia.

4.1.2.Keunggulan GTFW 3 UGM

1. Simple.
2. Modular, mudah menambah atau mengurangi fitur.
3. Scalable, dapat digunakan untuk mengembangkan aplikasi sederhana maupun kelas berat.
4. Upgradeable, dapat ditingkatkan kemampuan core framework-nya.
5. Mengadopsi konsep Model View Controller (MVC).
6. Multi database dan multi koneksi database.

4.1.3.Teknologi GTFW 3 UGM

1. PHP 5.2x – json
2. JavaScript 2.x - AJAX
3. xHtml
4. CSS 2.x
5. Mysql 5.x
6. Oracle
7. Postgresql
8. Firebird 2.x

4.1.4.Browser Support

1. Firefox 2.x, 3.x – recommended
2. Opera
3. Internet Explorer
4. Konqueror
5. Safari
6. Chrome

4.1.5. Library GTFW 3 UGM

1. Addodb
2. Fpdf
3. Jpgraph
4. pat_template

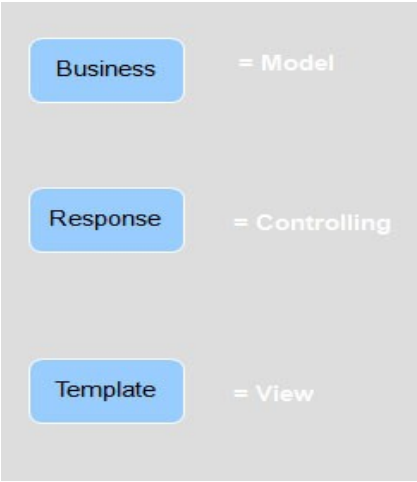
5. Writeexcel

6. AdvAjax

7. jQuery, dan lain-lain.

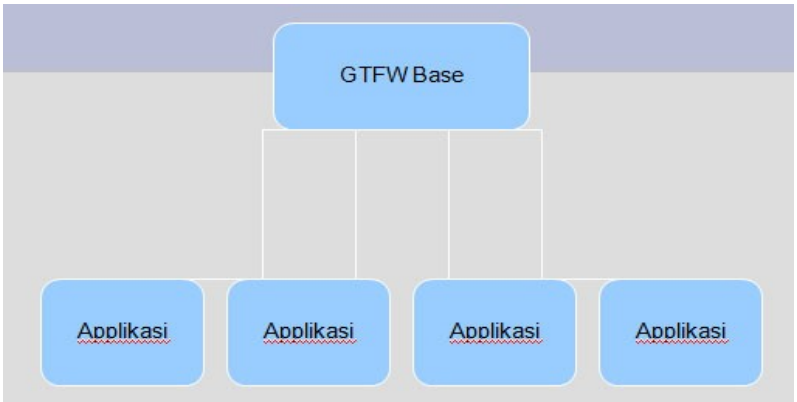
4.2. Struktur dan Hierarki GTFW 3 UGM

1. MVC di GTFW 3 UGM



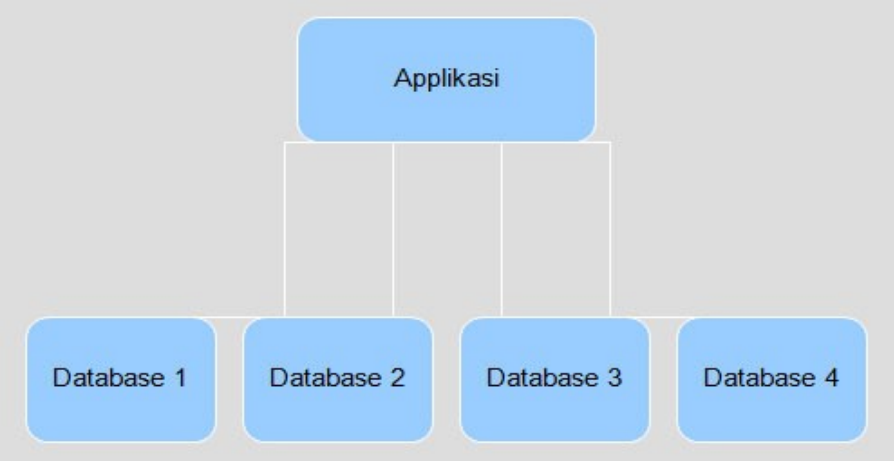
Gambar 27: Business Response Template

2. GTFW 3 UGM Base dan Aplikasi



Gambar 28: GTFW Base

3. GTFW 3 UGM Aplikasi dan Database



Gambar 29: GTFW App dan Database

4.3. Instalasi GTFW 3 UGM

4.3.1.Langkah-langkah instalasi GTFW 3 UGM :

1. Compressed file (zip/rar dll)

- Extract file.
- Copy dan paste atau upload folder dan file hasil extract ke dalam direktori document root web server.
- Export file dump database GTFW 3 UGM (default database adalah Oracle)
- Buka file gtfw-app/config/gtfw_base_dir.def dengan text editor dan kofigurasi letak folder gtfw-base. Sebagai contoh, Misal folder gtfw_base ada dalam direktori : *D:/OIT/Aplikasi/GTFW3_ORA/gtfw3-oracle/gtfw-base*
- Buka file gtfw-app/config/application.conf.php dengan text editor dan lakukan beberapa konfigurasi untuk aplikasi seperti berikut :

Konfigurasi base url :

```
//Id Aplikasi yang harus terdaftar pada tabel GTFW_APPLICATION
$application['application_id'] = 100;
//Dengan trailling slash
$application['basedir'] = '/gtfw3-ugm/gtfw-app/';
//Tanpa trailling slash
$application['baseaddress'] = 'http://localhost';
```

Konfigurasi database :

```
//database driver
$application['db_conn'][0]['db_driv'] = 'adodb';
//untuk oracle
$application['db_conn'][0]['db_type'] = 'oci8';
//hostname
$application['db_conn'][0]['db_host'] = 'localhost';
//username schema
$application['db_conn'][0]['db_user'] = 'gtfw3_devel';
//password schema
$application['db_conn'][0]['db_pass'] = 'gtfw3_devel';
//SID
$application['db_conn'][0]['db_name'] = 'xe';
```

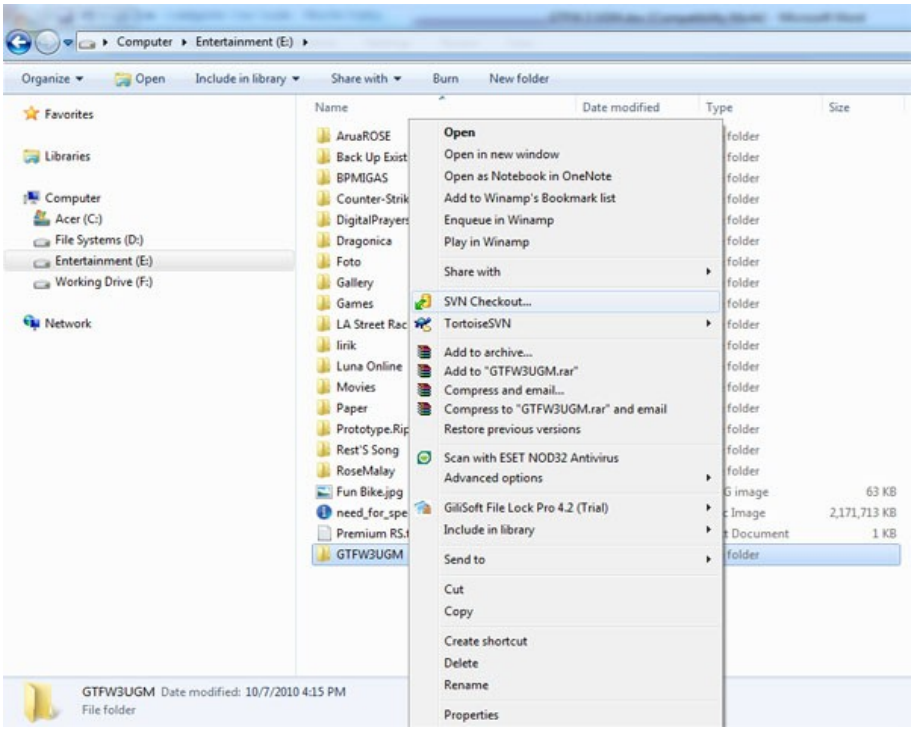
4.3.2. SVN

- Alamat SVN : http://dev.oit.ugm.ac.id/svn/gtfw/gtfw3-ugm.
- Username dan password :

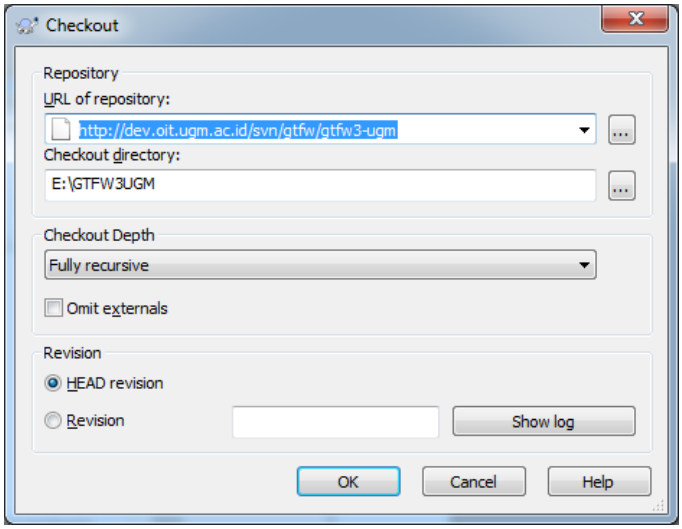
  username : integrasi

  password : integrasi
- Sebelum menggunakan SVN, pastikan sudah terinstal SVN Client, misal TortoiseSVN.
- Siapkan/buat folder untuk melakukan checkout. Setelah folder siap kemudian lakukan

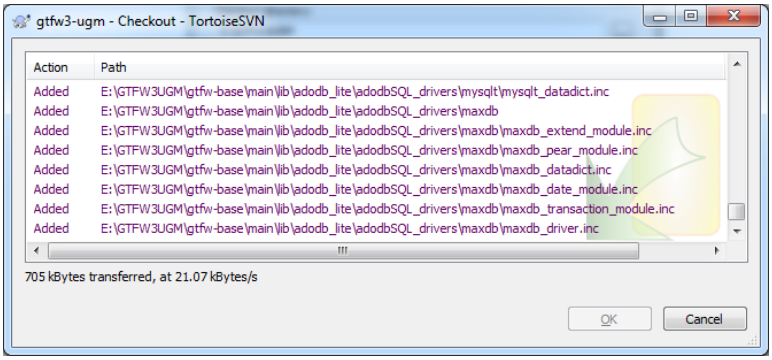
checkout ke alamat SVN http://dev.oit.ugm.ac.id/svn/gtfw/gtfw3-ugm. Seperti tampak pada ilustrasi berikut :



Gambar 30: Buat Folder Baru dan Checkout



Gambar 31: Checkout SVN



Gambar 32: Proses Checkout

- Setelah proses checkout selesai pindahkan folder dan file GTFW 3 UGM ke web server.
- Export file dump database GTFW 3 UGM (default database adalah Oracle)
- Buka file gtfw-app/config/gtfw_base_dir.def dengan text editor dan kofigurasi letak folder gtfw-base. Sebagai contoh, Misal folder gtfw_base ada dalam direktori : *D:/OIT/Aplikasi/GTFW3_ORA/gtfw3-oracle/gtfw-base*
- Buka file gtfw-app/config/application.conf.php dengan text editor dan lakukan beberapa konfigurasi untuk aplikasi seperti berikut :

Konfigurasi base url :

```
//Id Aplikasi yang harus terdaftar pada tabel GTFW_APPLICATION
$application['application_id'] = 100;
//Dengan trailling slash
$application['basedir'] = '/gtfw3-ugm/gtfw-app/';
//Tanpa trailling slash
$application['baseaddress'] = 'http://localhost';
```

Konfigurasi database :

```
//database driver
$application['db_conn'][0]['db_driv'] = 'adodb';
```

```
//untuk oracle
$application['db_conn'][0]['db_type'] = 'oci8';
//hostname
$application['db_conn'][0]['db_host'] = 'localhost';
//username schema
$application['db_conn'][0]['db_user'] = 'gtfw3_devel';
//password schema
$application['db_conn'][0]['db_pass'] = 'gtfw3_devel';
//SID
$application['db_conn'][0]['db_name'] = 'xe';
```

5. SESI V

5.1. Konvensi GTFW3 UGM

Tujuan Pembelajaran Khusus

- Peserta WorkShop diharapkan selaku memperhatikan konvensi/tata-cara penggunaan GTFW UGM dalam setiap pengembangan aplikasi berbasis GTFW

5.2. Konvensi GTFW UGM

Konvensi GTFW UGM diperlukan untuk mempermudah dalam mengembangkan aplikasi. Konvensi ini di antaranya adalah konvensi penamaan file, penamaan modul, dan penggunaan fitur GTFW UGM lainya.

Konvensi	Keterangan
business	Dalam konsep MVC business ekuivalen dengan ***model***. Merupakan folder tempat menyimpan file-file model.
response	Ekuivalen dengan ***controller***. Merupakan folder tempat file-file yang mengatur logika/bisnis proses aplikasi.
template	Ekuivalen dengan ***view***.
Penamaan modul	Menggunakan huruf kecil (lowercase) dan garis bawah (_). Contoh : ***ref_coa_jenis***.
Penamaan file	Semua penamaan file menggunakan lowercase dan jika lebih dari satu kata maka setiap kata dihubungkan dengan garis bawah (_). Contoh : ***model_ref_coa_jenis.php***.
Penamaan file business	File class business diawali dengan prefiks *model* kemudian diikuti nama modulnya dan untuk file yang menampung query diawali dengan prefiks *sql*. Contoh : Nama modul : ref_coa_jenis Nama file class business : ***model***_ref_coa_jenis.php

	Nama file query : ***sql***_ref_coa_jenis.php
Penamaan file response	a. File yang digunakan untuk memanipulasi tampilan data diawali dengan prefiks ***view***. Contoh : ***view***_ref_coa_jenis.***class***.php. b. File yang digunakan untuk memanipulasi tampilan form input diawali dengan prefiks ***view_input***. Contoh : ***view_input***_ref_coa_jenis.***class***.php. c. File yang berisi class untuk memanipulasi operasi *create*, *update* dan *delete* data diawali dengan prefiks ***mgr***. Contoh : ***mgr***_ref_coa_jenis.***class***.php. d. File yang berisi class untuk memanggil class dan fungsi operasi *create*, *update* dan *delete* data diawali dengan prefiks ***mgr*** dan ditambahkan sufiks ***html*** atau ***json***.Contoh : ***mgr***_ref_coa_jenis_***html***.***class***.php dan ***mgr***_ref_coa_jenis_***json***.***class***.php
Penamaan file template	a. File template yang digunakan untuk menampilkan data hasil manipulasi dinamai dengan prefiks ***view***. Contoh : ***view***_ref_coa_jenis.html b. File template untuk menampilkan form input diawali dengan prefiks ***view_input***. Contoh ***view_input***_ref_coa_jenis.html
Folder “asset”	Digunakan untuk menyimpan resources (javascript, css, images, dsb.) file dan library yang bersifat global.
Folder “themes”	Untuk menyimpan template layout dan resources yang dibutuhkan oleh template yang bersangkutan.
Folder “config”	Untuk menyimpan file-file konfigurasi framework.
Folder “modules”	Untuk meletakkan modul-modul.
Folder “main”	Untuk menambah library.
Pendaftaran modul	a. Module dengan menu didaftarkan menggunakan menu “***Menu***” yang ada dibawah menu “***Manajemen Sistem***”. b. Module tanpa menu didaftarkan menggunakan menu “***Module***”. c. Pengisian sub-module: - ***read*** : digunakan untuk modul yang bertujuan untuk membaca/menampilkan data. - ***create*** : digunakan untuk modul yang bertujuan untuk membuat/menambah data. - ***update*** : digunakan untuk modul yang bertujuan untuk mengubah data. - ***delete*** : digunakan untuk modul yang bertujuan untuk menghapus data. - ***read_create*** : digunakan untuk membaca/menampilkan data pada modul

	form create. - ***read_update*** : digunakan untuk membaca/menampilkan data pada modul form update.

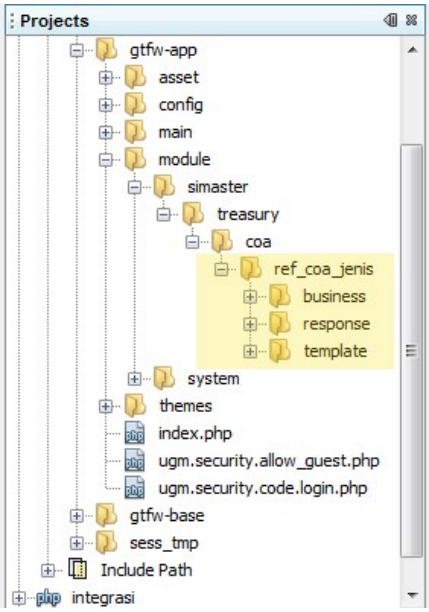
6. SESI VI

6.1. Pemrograman GTFW UGM

Tujuan Pembelajaran Khusus

- Peserta WorkShop secara mandiri dapat membangun aplikasi berbasis GTFW UGM

Buatlah direktori seperti tampak pada gambar berikut (yang diberi background warna kuning) :



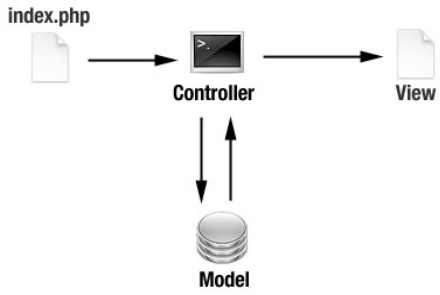
Gambar 33: Folder Modul GTFW

Dalam konsep pemrograman MVC maka *Business* ekuivalen dengan *Model*, *Response* dengan *Controller* dan *Template* dengan *View*. Secara sederhana ketiganya berfungsi sebagai berikut :

- *Model* (*business*), menghubungkan antara aplikasi dengan *database*. Di sinilah aktivitas *add, select, update* dan *delete* pada *database* dilakukan. Maka di dalam *model* inilah kita akan membuat *query-query* untuk melakukan manipulasi pada *database*. Kata kunci *model* adalah *query*.

- *View* (*template*), bertanggung jawab pada penampilan *interface* aplikasi yang akan terlihat oleh *user*. Maka di dalam *view* merupakan *code-code* dalam bentuk HTML atau JavaScript. Kata kunci *view* adalah HTML.

- *Controller* (*response*), adalah pusat kendali atas *model* dan *view*. Data yang diperoleh dari *database* akan diolah di sini, untuk kemudian dikirim ke *view* untuk ditampilkan kepada *user*. Selain mengambil data dari *database*, *controller* juga memanipulasi data untuk kemudian dikirim ke *model*. Kata kunci *controller* adalah PHP.



Gambar 34: Proses MVC

6.2. Membuat Modul Referensi Jenis COA (CRUD = Create Read Update Delete).

Read Data Jenis COA dari Database.

Buat file php dengan nama **view_ref_coa_jenis.class.php** dan simpan di folder ***response***. Isi dengan *script* berikut :

```
class view_ref_coa_jenis extends UgmHtmlResponse{

    function view_ref_coa_jenis(){

        parent::UgmHtmlResponse();

        $this->LoadBusiness('ref_coa_jenis','ref_coa_jenis',5);

    }

    function TemplateModule(){

        $this->LoadTemplate('ref_coa_jenis');

    }

    function ProcessRequest(){
    }

    function ParseTemplate($data = null){
    }
}
```

File di atas merupakan *class* **view_ref_coa_jenis** yang merupakan *extends* dari UgmHtmlResponse, penamaan *class* ini sesuai (sama) dengan penamaan file yang menampung class yang bersangkutan (*naming confession* file sudah ditentukan). File ini digunakan untuk memanipulasi pembacaan dan tampilan data. Dalam file ini terdapat 3 fungsi utama, yaitu :

- **TemplateModule** (yang diberi warna dasar), untuk mendefinisikan file template yang akan digunakan.

Dalam contoh akan menggunakan file **view_ref_coa_jenis.html** sebagai template untuk menampilkan data dalam *user interface*. Fungsi yang digunakan untuk mendefinisikan file template yang akan digunakan adalah fungsi *LoadTemplate()* dengan parameter nama file templatenya. Dalam hal ini harus diperhatikan bahwa semua nama file template diawali dengan prefiks *view_*, sedangkan dalam penggunaan fungsi *LoadTemplate()* prefiks tersebut dihilangkan. Mengacu pada contoh, menggunakan file **view_ref_coa_jenis.html,** sehingga penggunaan dalam fungsi *LoadTemplate()* adalah *LoadTemplate(‘**ref_coa_jenis’**)*.

- **ProcessRequest**, untuk pengolahan data, input dari *user* maupun data dari *database* (*business*). Perhatikan *source* program berikut :

```
    function ProcessRequest(){

        $return['param'] = $param = $_REQUEST->AsArray();



        //Membaca data dari tabel database untuk mengisi opsi form select/combobox

        $combo_periode_akun = $this->model_ref_coa_jenis->combo_periode_akun();

        $combo_periode_akun = array_merge(array(array('id'=>'','name'=>'--ALL--')),
$combo_periode_akun);

        //--



        //Rendering/menampilkan combobox pada tempalte

        Messenger::Instance()->SendToComponent('combobox', 'Combobox', 'view', 'html',
```

```
'p_periode_akun',

            array('p_periode_akun',$combo_periode_akun, $param['p_periode_akun'],'',''),
Messenger::CurrentRequest);

        //--



        //Membaca total data yang ada dalam tabel database

        $total = $this->model_ref_coa_jenis->count_data($param['p_cari'],$param['p_periode_akun']);

        //--



        //Inisialisasi untuk paging data yang ditampilkan

        $itemViewed = GTFWConfiguration::GetValue('application', 'item_viewed');

        $currPage = 1;

        $startRec = 0;

        if(isset($_GET['page'])) {

            $currPage = (string)$_GET['page']->StripHtmlTags()->SqlString()->Raw();

            $startRec =($currPage-1) * $itemViewed;

        }

        //--



        //Membaca data dari tabel

        $data = $this->model_ref_coa_jenis->read_data($param['p_cari'], $param['p_periode_akun'],
$currPage,$itemViewed);

        //--



        //Setting url untuk paging

        $url = Dispatcher::Instance()->GetUrl(Dispatcher::Instance()->mModule,
Dispatcher::Instance()->mSubModule, Dispatcher::Instance()->mAction, Dispatcher::Instance()-
```

```
>mType.'&kode='.$param['kode'].'&label='.$param['label']);

        //--

        //Rendering/menampilkan paging ke template

        Messenger::Instance()->SendToComponent('paging', 'Paging', 'view', 'html', 'paging_top',

            array($itemViewed, $total, $url, $currPage), Messenger::CurrentRequest);

        //--



        $return['data'] = $data;

        $return['start'] = $startRec+1;



        //Return value

        return $return;

        //--

        }
```

Contoh diatas merupakan contoh fungsi *ProcessRequest()* yang digunakan untuk memanipulasi data baik data dari *database* maupun data yang diinput oleh *user*. Sekarang kita kupas apa saja yang ada dalam fungsi ini.

- Menyimpan inputan *user* dalam sebuah variabel.

```
$return['param'] = $param = $_REQUEST->AsArray();
```

- Membaca data dari database.

```
//Mengambil data dari tabel database untuk mengisi opsi form select/combobox

$combo_periode_akun = $this->model_ref_coa_jenis->combo_periode_akun();

$combo_periode_akun = array_merge(array(array('id'=>'', 'name'=>'--ALL--')), $combo_periode_akun);

//--
```

Dari potongan *source* diatas, kita melakukan pembacaan data dari *database* yang kita

simpan dalam variabel *$combo_periode_akun*, untuk membaca data ini berarti kita harus membuat turunan *class database* dan membuat fungsi *combo_periode_akun()* dimana fungsi tersebut berisi *query* untuk membaca *database*.

- Rendering/menampilkan data ke dalam template.

```
//Rendering/menampilkan combobox pada tempalte
Messenger::Instance()->SendToComponent('combobox', 'Combobox', 'view',
'html', 'p_periode_akun', array('p_periode_akun', $combo_periode_akun,
$param['p_periode_akun'], '', ''), Messenger::CurrentRequest);
//--
```

Potongan code diatas merupakan instansiasi untuk merender/menampilkan komponen form yang berupa combobox ke dalam template. Perhatikan isi parameter dalam array :

```
array('p_periode_akun', $combo_periode_akun, $param['p_periode_akun'], '',
'')
```

Parameter pertama (*p_periode_akun*) adalah nama form combobox yang kita render, parameter kedua (*$combo_periode_akun*) adalah variabel yang berisi data dalam format sebuah *array* (*array(array('id'=>'id_1', 'name'=>'name_1'), array('id'=>'id_2', 'name'=>'name_2'))),* parameter ketiga digunakan untuk memberi nilai awal pada combobox, parameter keempat digunakan untuk memberi tambahan pilihan combobox yaitu opsi "PILIH" yang tidak ada nilainya hanya untuk kepentingan *labeling* saja, dan parameter yang terakhir merupakan atribut form combobox misal kita akan menambahkan atribut *onChange* dan sejenisnya bisa ditambahkan dalam parameter terakhir.

- **ParseTemplate**, untuk melakukan *parsing* data ke dalam template HTML.

```
    function ParseTemplate($data=null){
```

```
	//Render/menampilkan button (tambah, edit, hapus) sesuai dengan hak akses user yang login
	$this->ButtonRendering();
	//--


	//Inisialisasi pesan untuk aksi yang dilakukan oleh user
	if ($_GET['err']!='') {
	   if ($_GET['err']=='1') {
	      $pesan = 'Penambahan data aksi berhasil dilakukan';
	      $class = 'notebox-done';
	   } elseif ($_GET['err']=='2') {
	      $pesan = 'Penambahan data aksi gagal dilakukan';
	      $class = 'notebox-alert';
	   } elseif ($_GET['err']=='3') {
	      $pesan = 'Pengubahan data aksi berhasil dilakukan';
	      $class = 'notebox-done';
	   } elseif ($_GET['err']=='4') {
	      $pesan = 'Pengubahan data aksi gagal dilakukan';
	      $class = 'notebox-alert';
	   } elseif ($_GET['err']=='5') {
	      $pesan = 'Penghapusan data aksi berhasil dilakukan';
	      $class = 'notebox-done';
	   } elseif ($_GET['err']=='6') {
	      $pesan = 'Penghapusan data aksi gagal dilakukan';
	      $class = 'notebox-alert';
	   }
	   $this->mrTemplate->SetAttribute('warning_box', 'visibility', 'visible');
	   $this->mrTemplate->AddVar('warning_box', 'ISI_PESAN', $pesan);
	   $this->mrTemplate->AddVar('warning_box', 'CLASS_PESAN', $class);
	}
```

```
        //--


        //Inisialisasi url untuk mengisi atribut action pada form pencarian

        $datas['URL_SEARCH'] = $this->c_dispacter->GetUrl($this->v_module, $this->v_sub_module,
$this->v_action, $this->v_type);

        $datas['KEY_CARI'] = $data['param']['p_cari'];

        $this->mrTemplate->AddVars('content', $datas);

        //--


        //Inisialisasi url untuk melakukan create/menambah data

        $this->mrTemplate->AddVar('button_add', 'URL_ADD', Dispatcher::Instance()-
>GetUrl('ref_coa_jenis', 'read_create', 'view', 'html'));

        //--


        //Render data ke dalam template

        if (empty($data['data'])) $this->mrTemplate->AddVar('data', 'EMPTY', 'YES');

        else {

           $this->mrTemplate->AddVar('data', 'EMPTY', 'NO');

           for ($i=0; $i<sizeof($data['data']); $i++) {

              $data['data'][$i]['number'] = $i+$data['start'];

              if ($i % 2 == 0) $data['data'][$i]['class_name'] = 'table-common-even'; else
$data['data'][$i]['class_name'] = '';

              $id = Dispatcher::Instance()->Encrypt($data['data'][$i]['KEY_ID']);


              //Inisialisasi url untuk update data

              $url_edit = Dispatcher::Instance()->GetUrl('ref_coa_jenis', 'read_update', 'view',
'html') . '&id=' . $id;

              $this->mrTemplate->AddVar('button_update', 'URL_EDIT', $url_edit);

              //--


              //Inisialisasi url untuk delete/menghapus data
```

```
            $url_delete = Dispatcher::Instance()->GetUrl('confirm', 'confirmDelete', 'do',
'html').
          '&id='.Dispatcher::Instance()->Encrypt($id.'&message='.Dispatcher::Instance()-
>Encrypt($message).
          '&dataName='.Dispatcher::Instance()->Encrypt($data['data'][$i]['CJ_DESKRIPSI']).
          '&label='.Dispatcher::Instance()->Encrypt($label).
          '&urlDelete='.Dispatcher::Instance()->Encrypt('ref_coa_jenis|delete|do|html').
          '&urlReturn='.Dispatcher::Instance()->Encrypt('ref_coa_jenis|read|view|html'));
             $this->mrTemplate->AddVar('button_delete', 'URL_DELETE', $url_delete);
             //--

             $this->mrTemplate->AddVars('data_item', $data['data'][$i]);
             $this->mrTemplate->parseTemplate('data_item', 'a');
          }
       }
       //--

    }
```

Fungsi *ButtonRendering()* digunakan untuk melakukan *render* button dimana button tersebut mewakili hak akses yang dimiliki oleh *user* yang sedang *login* dalam aplikasi.

Fungsi konstruktor, dalam contoh yaitu *function* view_ref_coa_jenis(), merupakan fungsi opsional bisa ditiadakan. Jika fungsi konstruktor tidak ada, maka untuk mendefinisikan pemanggilan model (LoadBusiness()) bisa dilakukan dalam fungsi ProcessRequest().

Setelah membuat file response, buat file html dengan nama **view_ref_coa_jenis.html** dan simpan di folder ***template***, file ini untuk mengatur tampilan/template yang ditampilkan pada *user.*

Isi dengan *script* berikut :

```
    <!-- patTemplate:tmpl name="content" -->
```

```
    <h1>Manajemen  Jenis COA</h1>
    <br/>
    <form method="POST" action="{URL_SEARCH}" class="dataquest xhr_simple_form dest_subcontent-
element" id="filterbox">
      <table>
        <tr>
          <th colspan="2"><h2><strong>Pencarian</strong></h2></th>
        </tr>
        <tr>
          <th width="20%">Kunci</th>
          <td>
            <input type="text" name="p_cari" value="{KEY_CARI}" size="40" />
          </td>
        </tr>
          <tr>
          <th width="20%">Periode Akun</th>
          <td>
            <!-- patTemplate:gtfwrendermodule module="combobox" submodule="combobox" action="view"
name="p_periode_akun"  / -->
          </td>
        </tr>
        <tr>
          <th> </th>
          <td>
            <input type="submit" name="carimenu" value=" Tampilkan &raquo;" class="buttonSubmit"/>
          </td>
        </tr>
```

```
      </table>
    </form>
    <br>
    <!-- patTemplate:tmpl name="warning_box" visibility="hidden" -->
    <div class="{CLASS_PESAN}">
      {ISI_PESAN}
    </div>
    <br/>
    <!-- /patTemplate:tmpl -->
    <!-- patTemplate:tmpl name="button_view" visibility="hidden" -->
    <div class="pageBar">
      <!-- patTemplate:tmpl name="button_add" visibility="hidden" -->
      <div class="toolbar">
        <a class="xhr dest_subcontent-element" href="{URL_ADD}" title="Tambah Data" tabindex="2">
        <img src="asset/images/button-add.gif" alt=""> Tambah</a>
      </div>
      <!-- /patTemplate:tmpl -->
      <!-- patTemplate:gtfwrendermodule module="paging" submodule="paging" action="view"
name="paging_top"  / -->
    </div>
    <table class="table-common" width="100%">
      <tr>
        <th width="10">No</th> <th>PERIODE AKUN</th> <th>KODE</th> <th>JENIS COA</th> <th>Aksi</th>
      </tr>
      <!-- patTemplate:tmpl name="data" type="condition" conditionvar="EMPTY" -->
        <!-- patTemplate:sub condition="YES" -->
          <tr><td colspan="4" align="center"><em>-- Data tidak ditemukan --</em></td></tr>
        <!-- /patTemplate:sub -->
```

```
    <!-- patTemplate:sub condition="NO" -->
      <!-- patTemplate:tmpl name="data_item" -->
        <tr class="{CLASS_NAME}">
          <td>{NUMBER}</td>
          <td>{PERIODE_AKUN_KEY_ID}</td>
          <td>{CJ_KD}</td>
          <td>{CJ_DESKRIPSI}</td>
          <td class="links">
            <!-- patTemplate:tmpl name="button_update" visibility="hidden" -->
            <a class="xhr dest_subcontent-element" href="{URL_EDIT}" title="Ubah">
            <img src="asset/images/button-edit.gif" alt="Ubah"/></a>
            <!-- /patTemplate:tmpl -->
            <!-- patTemplate:tmpl name="button_delete" visibility="hidden" -->
            <a class="xhr dest_subcontent-element" href="{URL_DELETE}" title="Hapus">
            <img src="asset/images/button-delete.gif" alt="Delete" /></a>
            <!-- /patTemplate:tmpl -->
          </td>
        </tr>
      <!-- /patTemplate:tmpl -->
    <!-- /patTemplate:sub -->
  <!-- /patTemplate:tmpl -->
</table>
<!-- /patTemplate:tmpl -->
<!-- /patTemplate:tmpl -->
```

File template dibuka dan ditutup dengan *tag* patTemplate dengan nama *content* sebagai *main*

template. *Tag* pembuka dan penutup ini bertujuan untuk *mapping* area data, *mapping* maksudnya adalah penempatan suatu data pada area yang sudah didefinisikan untuk ditampilkan pada templatenya. Pada contoh terdapat beberapa area patTemplate.

- *Render*/menampilkan variabel yang sudah didefinisikan/diinisialkan dalam file *response*.

Contoh : `action="{URL_SEARCH}"`

{URL_SEARCH} adalah variabel yang sudah kita definisikan dalam file *response* yang dibuat.

- Render combobox

```
    <!-- patTemplate:gtfwrendermodule module = "combobox" submodule = "combobox" action = "view" name = "p_periode_akun"  / -->
```

Perlu diingat bahwa *name* di sini harus sama dengan *name* yang didefinisikan dalam file *response*.

- *Render paging* tabel data

```
    <!-- patTemplate:gtfwrendermodule module = "paging" submodule = "paging" action = "view" name = "paging_top"  / -->
```

- *Render* data

```
    <table class="table-common" width="100%">
      <tr>
        <th width="10">No</th> <th>PERIODE AKUN</th> <th>KODE</th> <th>JENIS COA</th> <th>Aksi</th>
      </tr>
      <!-- patTemplate:tmpl name="data" type="condition" conditionvar="EMPTY" -->
        <!-- patTemplate:sub condition="YES" -->
          <tr><td colspan="4" align="center"><em>-- Data tidak ditemukan --</em></td></tr>
        <!-- /patTemplate:sub -->
        <!-- patTemplate:sub condition="NO" -->
```

```
        <!-- patTemplate:tmpl name="data_item" -->
          <tr class="{CLASS_NAME}">
            <td>{NUMBER}</td>
            <td>{PERIODE_AKUN_KEY_ID}</td>
            <td>{CJ_KD}</td>
            <td>{CJ_DESKRIPSI}</td>
            <td class="links">
              <!-- patTemplate:tmpl name="button_update" visibility="hidden" -->
              <a class="xhr dest_subcontent-element" href="{URL_EDIT}" title="Ubah">
              <img src="asset/images/button-edit.gif" alt="Ubah"/></a>
              <!-- /patTemplate:tmpl -->
              <!-- patTemplate:tmpl name="button_delete" visibility="hidden" -->
              <a class="xhr dest_subcontent-element" href="{URL_DELETE}" title="Hapus">
              <img src="asset/images/button-delete.gif" alt="Delete" /></a>
              <!-- /patTemplate:tmpl -->
            </td>
          </tr>
        <!-- /patTemplate:tmpl -->
      <!-- /patTemplate:sub -->
    <!-- /patTemplate:tmpl -->
  </table>
```

Selanjutnya buat file **model_ref_coa_jenis.php** dalam folder ***business***. File ini digunakan untuk melakukan koneksi (*create*, *read*, *update*, *delete*) dengan *database*.

```
    class model_ref_coa_jenis extends UgmModel{
```

```
        function __construct($connectionNumber=0) {

            $this->LoadSQLModel('ref_coa_jenis','ref_coa_jenis');

            //$this->adodb->debug = true;

            parent::__construct($connectionNumber);

        }

        function read_data($p_cari, $p_periode_akun, $page, $page_size){

             $data = sprintf($this->mSqlQueries['read_data'], '%'.$p_periode_akun.'%', '%'.
$p_cari.'%', '%'.$p_cari.'%', $page, $page_size);

            $data = $this->Open($data, array());

            return $data;

        }

        function count_data($p_cari, $p_periode_akun){

            $data = sprintf($this->mSqlQueries['count_data'], '%'.$p_periode_akun.'%', '%'.
$p_cari.'%', '%'.$p_cari.'%');

            $data = $this->Open($data, array());

            return $data[0]['TOTAL'];

        }

        function combo_periode_akun(){

            $data = $this->DB_Open($this->mSqlQueries['combo_periode_akun']);

            return $data;

        }

    }
```

File model tersebut berisi fungsi-fungsi untuk memanipulasi *query* untuk melengkapi file ini buat file yang berisi *query*-nya. Buat file **sql_ref_coa_jenis.php** satu folder dengan file tersebut.

```
    $sql['read_data'] ="
```

```
  SELECT * FROM (
  SELECT rownum as no,a.* FROM
  (
   SELECT key_id, periode_akun_key_id, cj_kd, cj_deskripsi
   FROM tref_coa_jenis a
   WHERE a.periode_akun_key_id like '%s'
   AND (upper(a.cj_deskripsi) like upper('%s') OR upper(cj_kd) like upper('%s'))
  ) a ) WHERE no between %d AND %d";


$sql['count_data'] ="
  SELECT count(key_id) as total
  FROM tref_coa_jenis a
  WHERE a.periode_akun_key_id like '%s'
  AND (upper(a.cj_deskripsi) like upper('%s') OR upper(cj_kd) like upper('%s'))
 ";


$sql['combo_periode_akun']="
    SELECT key_id as \"id\", PAK_NAMA as \"name\"
    FROM TCFG_PERIODE_AKUNTANSI
";
```

6.3. Register Modul

Setelah semua file modul siap langkah selanjutnya adalah mendaftarkan modul yang dibuat tersebut ke dalam *database framework* GTFW 3 UGM. Dalam mendaftarkan modul harus

diperhatikan apakah modul tersebut nantinya akan mempunyai menu atau tidak. Jika modul tersebut akan diakses melalui sebuah menu maka harus didaftarkan melalui menu **Menu** yang ada didalam menu **Manajemen Sistem** dalam *framework* GTFW 3 UGM, dan jika tidak memiliki menu, didaftarkan melalui menu **Module**.

- Modul dengan sebuah menu.

Modul dengan sebuah menu diregisterkan melalui menu Menu yang ada dalam menu Manajemen Sistem GTFW 3 UGM.

1. Nama Menu : nama menu/label menu yang sesuai dengan kegunaan dan fungsi modul. Contoh : Jenis COA
2. Parent Menu : induk dari menu yang dibuat. Misal menu dibuat dibawah menu COA.
3. Show : Ya jika menu akan ditampilkan, Tidak jika menu tidak ditampilkan dalam *interface*.
4. Icon : gambar ikon untuk menu bersifat opsional.

Gambar 35: Register Menu dan Modul

5. Order : urutan menu.

6. Module : modul dari menu yang dibuat. Jika dipilih **Home** maka menu tersebut merupakan *link* ke modul **Home**. Jika dipilih **Module Baru** maka kita harus mendefinisikan modul baru untuk menu tersebut. Misal membuat menu untuk modul yang baru.

7. Path : direktori modul. Misal modul ada didalam direktori *simaster/treasury/coa*.

8. Module : modul baru yang didaftarkan. Modul yang telah kita buat adalah *ref_coa_jenis* (sesuai nama folder yang menampung file-file yang sudah dibuat).

9. Label : label/deskripsi modul bersifat opsional.

10. Sub Module : merupakan aksi dari modul.

    - *Read* : digunakan untuk modul yang bertujuan untuk membaca/menampilkan data.
    - *Create* : digunakan untuk modul yang bertujuan untuk membuat/menambah data.
    - *Update* : digunakan untuk modul yang bertujuan untuk mengubah data.
    - *Delete* : digunakan untuk modul yang bertujuan untuk menghapus data.
    - *Read_create* : digunakan untuk membaca/menampilkan data pada modul form create.
    - *Read_update* : digunakan untuk membaca/menampilkan data pada modul form update.

Sub Module/aksi yang sudah dibuat adalah modul untuk membaca jadi form ini diisi *read*.

11. Deskripsi : deskripsi bersifat opsional.

12. Hak akses : mengatur akses terhadap modul. '*All*', bisa diakses oleh semua '*user*', dan *Exclusive*, hanya bisa diakses oleh *user* yang diberi hak untuk mengakses. Modul yang sudah dibuat direncanakan hanya bisa diakses oleh user tertentu, modul diset *Exclusive*.

13. Load File : 'Ya', untuk mendefinisikan file yang akan di-*load* ketika menu atau modul

diakses. ‘Tidak’, jika diisi opsi ini maka file yang akan di-*load* ketika menu atau modul diakses adalah yang diisikan dalam form Sub Module. Karena modul yang dibuat didefinisikan dengan nama *view_ref_coa_jenis.class.php*, artinya nama file kita berbeda dengan yang diisikan pada form Sub Module maka kita mendefinisikan file baru jadi pada form ini kita isi ‘Ya’.

14. Nama File : form ini hanya diisi jika form Load File diisi ‘Ya’, file ini yang nantinya akan diakses jika menu atau modul yang bersangkutan dipanggil. Mengacu pada contoh kita isi form ini dengan ‘view_ref_coa_jenis’.

- Modul tanpa menu.

Modul tanpa menu diregisterkan melalui menu Module yang ada dalam menu Manajemen Sistem GTFW 3 UGM.

Gambar 36: Register Modul

1. Path : direktori modul. Misal modul ada didalam direktori *simaster/treasury/coa*.

2. Module : modul baru yang didaftarkan. Misalnya modul yang telah kita buat adalah *ref_coa_jenis*.

3. Label : label/deskripsi modul bersifat opsional.

4. Sub Module : merupakan aksi dari modul.

   - *Read* : digunakan untuk modul yang bertujuan untuk membaca/menampilkan data.
   - *Create* : digunakan untuk modul yang bertujuan untuk membuat/menambah data.
   - *Update* : digunakan untuk modul yang bertujuan untuk mengubah data.
   - *Delete* : digunakan untuk modul yang bertujuan untuk menghapus data.
   - *Read_create* : digunakan untuk membaca/menampilkan data pada modul form create.
   - *Read_update* : digunakan untuk membaca/menampilkan data pada modul form update.

Sub Module/aksi yang sudah dibuat misal adalah modul untuk membaca jadi form ini diisi *read*.

5. Action : modul digunakan untuk *view* atau untuk *do*.

6. Type : jenis file apakah html atau json.

7. Hak akses : mengatur akses terhadap modul. '*All*', bisa diakses oleh semua '*user*', dan *Exclusive*, hanya bisa diakses oleh *user* yang diberi hak untuk mengakses. Modul yang sudah dibuat direncanakan hanya bisa diakses oleh user tertentu, modul diset *Exclusive*.

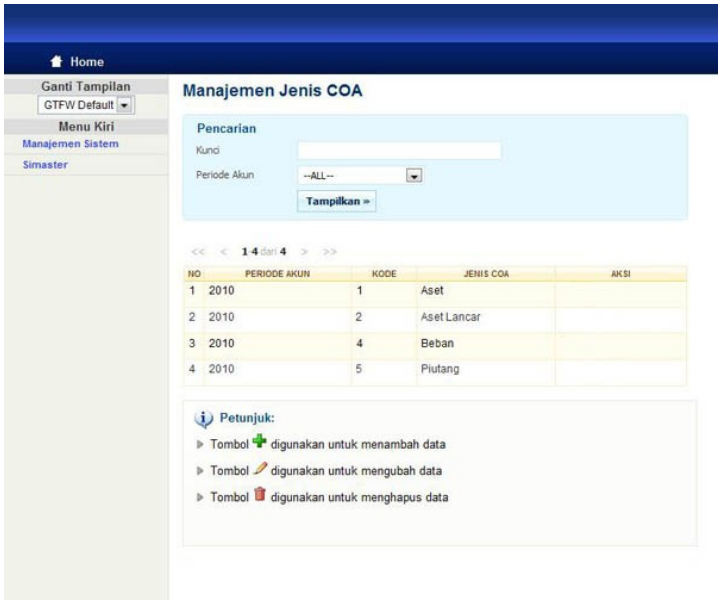
8. Menu : menu yang menjadi induk dari modul.

9. Aksi : fungsi dari modul apakah untuk lihat data (*read*), tambah data (*create*), ubah data (*update*), atau hapus data (*delete*).

10. Load File : 'Ya', untuk mendefinisikan file yang akan di-*load* ketika menu atau modul diakses. 'Tidak', jika diisi opsi ini maka file yang akan di-*load* ketika menu atau modul diakses adalah yang diisikan dalam form Sub Module. Karena modul yang dibuat didefinisikan dengan nama *view_ref_coa_jenis.class.php*, artinya nama file kita berbeda dengan yang diisikan pada form Sub Module maka kita mendefinisikan file baru jadi pada form ini kita isi 'Ya'.

11. Nama File : form ini hanya diisi jika form Load File diisi 'Ya', file ini yang nantinya akan diakses jika menu atau modul yang bersangkutan dipanggil. Mengacu pada contoh kita isi form ini dengan 'view_ref_coa_jenis'.

Jika register modul yang dilakukan benar maka akan terlihat tampilan seperti berikut :



Gambar 37: Capture Modul yang berhasil diregister

Create dan Update Data Jenis COA.

Buat file php dengan nama **view_input_ref_coa_jenis.class.php** dan simpan dalam folder ***response***. Isi dengan *script* berikut :

```
class view_input_ref_coa_jenis extends UgmHtmlResponse{


  //Konstruktor
  function view_input_ref_coa_jenis(){
    parent::UgmHtmlResponse();
    //mendefinisikan file Model yang digunakan
    $this->LoadBusiness('ref_coa_jenis', 'ref_coa_jenis',5);
    //--
  }
```

```
      //--


      //Fungsi pendefinisian template yang akan digunakan
      function TemplateModule(){
        $this->LoadTemplate('ref_coa_jenis', 'input_ref_coa_jenis');
      }
      //--


      function ProcessRequest(){
        $combo_periode_akun = $this->model_ref_coa_jenis->combo_periode_akun();
        $return['param'] = $param = $_REQUEST->AsArray();


        //Membaca database untuk mendapatkan data jenis coa berdasarkan id
        //untuk proses update
        $jenis_coa = $this->model_ref_coa_jenis->read_data_by_id($param['id']);
        //--


        if($jenis_coa[0]['PERIODE_AKUN_KEY_ID']=='') $param['p_periode_akun'] = date(Y);
        else $param['p_periode_akun'] = $jenis_coa[0]['PERIODE_AKUN_KEY_ID'];


        //Render combobox periode akun dalam template
        Messenger::Instance()->SendToComponent('combobox', 'Combobox', 'view', 'html',
'p_periode_akun', array('p_periode_akun', $combo_periode_akun, $param['p_periode_akun'], '', ''),
 Messenger::CurrentRequest);
        //--
```

```
        $return['jenis_coa'] = $jenis_coa;

        return $return;

      }


      function ParseTemplate($data=null){

        //Mengindentifikasi proses yang dilakukan apakah create atau update

        $sub = ($data['param']['id']=='') ? 'create' : 'update';

        $data['data']['judul'] = (empty($data['param']['id'])) ? 'Tambah' : 'Update';

        //--


        if($_GET['err']!='') {

          $this->mrTemplate->SetAttribute('warning_box', 'visibility', 'visible');

          $this->mrTemplate->AddVar('warning_box', 'ISI_PESAN', 'Masukkan isian bertanda *');

          $this->mrTemplate->AddVar('warning_box', 'CLASS_PESAN', 'notebox-alert');

        }


        //Inisialisasi parameter yang akan dikirim/ditampailkan ke template

        $data['data']['ID'] = (string)$data['jenis_coa'][0]['KEY_ID'];

        $data['data']['CJ_KD'] = (string)$data['jenis_coa'][0]['CJ_KD'];

        $data['data']['CJ_DESKRIPSI'] = (string)$data['jenis_coa'][0]['CJ_DESKRIPSI'];

        //--


        //Mendefinisikan URL action untuk proses selanjutnya

        $data['data']['URL_ACTION'] = Dispatcher::Instance()->GetUrl('ref_coa_jenis', $sub, 'do',
'html');

        //--
```

```
      //Mengirim parameter ke template
      $this->mrTemplate->AddVars('content', $data['data']);
      //--
    }
}
```

File ini untuk mengatur data dan tampilan pada form yang akan digunakan untuk proses *create* data sekaligus untuk proses *update*. Selanjutnya buat file template untuk menampilkan form *create* dan form *update*, simpan dengan nama **view_input_ref_coa_jenis.html** dalam folder ***template***. Isi file tersebut dengan *script* berikut :

```
<!-- patTemplate:tmpl name="content" -->
<h1>Manajemen Jenis COA</h1> <br/>

<!-- patTemplate:tmpl name="warning_box" visibility="hidden" -->
<div class="{CLASS_PESAN}"> {ISI_PESAN} </div> <br/>
<!-- /patTemplate:tmpl -->

<form method="POST" class="xhr_form std_form" action="{URL_ACTION}" id="frmInput" >
  <table class="table-edit" width="100%">
    <tr class="subhead">
      <th colspan="2">{JUDUL} Jenis COA</th>
    </tr>
    <tr>
      <th width="20%">Periode Akun</th>
```

```
        <td>
          <!-- patTemplate:gtfwrendermodule module="combobox" submodule="combobox" action="view"
name="p_periode_akun"  / -->
        </td>
      </tr>
      <tr>
        <th width="">Kode Jenis COA</th>
        <td>
          <input type="text" name="p_kode" value="{CJ_KD}" size="10"/> *
        </td>
      </tr>
      <tr>
        <th>Deskripsi</th>
        <td>
          <textarea name="p_deskripsi">{CJ_DESKRIPSI}</textarea>
        </td>
      </tr>
      <tr class="buttons">
        <th> </th>
        <td>
          <input type="hidden" name="p_key_id" value="{ID}"/>
          <input type="submit" name="btnsimpan" value=" Simpan " class="buttonSubmit"/>
                <input type="submit"  name="btnbalik" value=" Batal " class="buttonSubmit"/>
        </td>
      </tr>
    </table>
```

```
    </form>
    <div class="petunjuk-area">
      <h4>Keterangan :</h4>
      <ul>
        <li>Tanda * menunjukkan bahwa field ini harus diisi.</li>
      </ul>
    </div>
    <!-- /patTemplate:tmpl -->
```

Langkah berikutnya tambahkan fungsi *read_data_by_id()* pada file model **model_ref_coa_jenis.php** untuk membaca data jenis coa di database berdasarkan *id*.

```
    function read_data_by_id($p_key_id){
            $data = $this->Open($this->mSqlQueries['read_data_by_id'],
array($p_key_id));
            return $data;
        }
```

Dan tambahkan *query* untuk mendukung fungsi tersebut pada file **sql_ref_coa_jenis.php**, *query*-nya adalah *select* data dengan filter *id*.

```
    $sql['read_data_by_id']="
        select * from tref_coa_jenis
        where key_id = %d
    ";
```

Selanjutnya daftarkan modul yang baru agar dikenali oleh *framework* melalui menu Module, langkah-langkahnya sama dengan pendaftaran modul sebelumnya. Karena modul digunakan sebagai form *create* dan form *update* sekaligus maka didaftarkan 2 (dua) kali dalam *framework* yaitu sebagai

modul *read_create* dan *read_update*.

- Sebagai modul read_create.

Gambar 38: Register Modul Read Create

- Sebagai modul read_update.

Gambar 39: Register Modul read_update

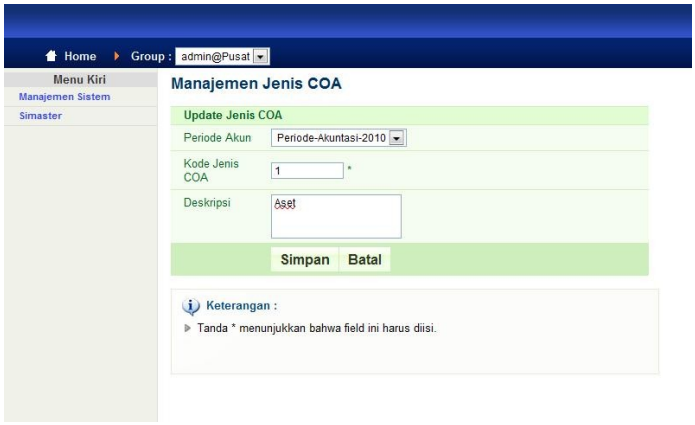
Yang perlu diperhatikan adalah pengisian *form* Sub Module dan Aksi. Jika modul yang dibuat bertujuan untuk menampilkan form *create* maka modul didaftarkan sebagai *read_create* dan aksi diisi sesuai kegunaan yaitu untuk menambah data, maka dipilih opsi "Tambah". Jika modul bertujuan untuk menampilkan form *update* maka modul didaftarkan sebagai *read_update* dan aksi

diisi pilihan “Ubah”. Hasilnya adalah sebagai berikut :

- Create.

Gambar 40: Tampilan Create

- Update.



Gambar 41: Tampilan Update

Selanjutnya membuat modul untuk melakukan penyimpanan data yang diinputkan oleh *user*. Modul ini tidak memiliki *view*, hanya merupakan fungsi untuk melakukan koneksi penyimpanan ke dalam *database*. Khusus untuk proses penyimpanan, ubah dan hapus dibuat versi html dan ajax sehingga dibuat 2 (dua) modul. Tujuannya adalah jika *web browser* yang digunakan oleh *user*

mendukung ajax maka yang dieksekusi adalah versi ajax dan jika tidak mendukung maka yang dieksekusi oleh *framework* adalah versi html biasa, sehingga aplikasi tetap berjalan jika *web browser* tidak mendukung ajax. Buat file untuk versi html dengan nama **mgr_ref_coa_jenis_html.class.php** file ini akan berisi *class* dengan fungsi *create()*, *update()* dan *delete()*.

```
class mgr_ref_coa_jenis_html extends UgmHtmlResponse{

  function mgr_ref_coa_jenis_html(){

    parent::UgmHtmlResponse();

    $this->LoadClass('mgr_ref_coa_jenis', 'ref_coa_jenis', 'response');

  }



  function ProcessRequest(){

    eval('$result = $this->'.strtolower($this->v_sub_module).'();');

    return $result;

  }



  function create(){

    $obj = $this->mgr_ref_coa_jenis;

    $urlRedirect = $obj->create();

    $this->RedirectTo($urlRedirect);

    return NULL;

  }



  function update(){

    $obj = $this->mgr_ref_coa_jenis;

    $urlRedirect = $obj->update();
```

```
    $this->RedirectTo($urlRedirect);

    return NULL;

  }



  function delete(){

    $obj = $this->mgr_ref_coa_jenis;

    $urlRedirect = $obj->delete();

    $this->RedirectTo($urlRedirect);

    return NULL;

  }

}
```

Kemudian untuk menampung versi ajax buat file **mgr_ref_coa_jenis_json.class.php** isinya secara umum sama dengan versi html.

```
class mgr_ref_coa_jenis_json extends UgmJsonResponse{

  function mgr_ref_coa_jenis_json(){

    parent::UgmJsonResponse();



    $this->LoadClass('mgr_ref_coa_jenis','ref_coa_jenis','response');

  }



  function ProcessRequest(){

    eval('$result = $this->'.strtolower($this->v_sub_module).'();');

    return $result;
```

```
      }


      function create(){
        $obj = $this->mgr_ref_coa_jenis;
        $urlRedirect = $obj->create();
        return array( 'exec' => 'GtfwAjax.replaceContentWithUrl("subcontent-element","'.
$urlRedirect.'&ascomponent=1")') ;

      }


      function update(){
        $obj = $this->mgr_ref_coa_jenis;

        $urlRedirect = $obj->update();
        return array( 'exec' => 'GtfwAjax.replaceContentWithUrl("subcontent-element","'.
$urlRedirect.'&ascomponent=1")') ;

      }


      function delete(){
        $obj = $this->mgr_ref_coa_jenis;
        $urlRedirect = $obj->delete();
        return array( 'exec' => 'GtfwAjax.replaceContentWithUrl("subcontent-element","'.
$urlRedirect.'&ascomponent=1")') ;
      }
    }
```

Selanjutnya buat file untuk menampung class proses, buat file mgr_ref_coa_jenis.class.php. Dalam file tersebut berisi *class* yang menghubungkan dengan model untuk eksekusi *database*.

```
    class mgr_ref_coa_jenis extends UgmProcess{

       function mgr_ref_coa_jenis(){

        $this->LoadBusiness('ref_coa_jenis','ref_coa_jenis',5);

       }



       function create() {

          $sub = 'read';

          if (isset($_POST['btnsimpan'])) {

             $obj = $this->model_ref_coa_jenis;

             if ($_POST['p_kode'] != '' and $_POST['p_deskripsi'] != '' and $_POST['p_periode_akun'] !
= '') {

                $result = $obj->write_data($_POST['p_kode'], $_POST['p_deskripsi'],
$_POST['p_periode_akun']);

                if ($result) $err = '&err=1'; else $err = '&err=2';

             } else {

                $err = '&err=1';

                $sub = 'create';

             }

          }



        return Dispatcher::Instance()->GetUrl('ref_coa_jenis', $sub, 'view', 'html') . $err;

         }
```

```
      function delete() {

        $obj = $this->model_ref_coa_jenis;

        $result = $obj->delete_data_by_id($_POST['idDelete']);

        if ($result) $err = '&err=5'; else $err = '&err=6';

        return Dispatcher::Instance()->GetUrl('ref_coa_jenis', 'read', 'view', 'html') . $err;

      }



      function update() {

         $sub = 'read';

        if (isset($_POST['btnsimpan'])) {

             if ($_POST['p_kode']!='' and $_POST['p_deskripsi']!='' and $_POST['p_periode_akun']!=''
and $_POST['p_key_id']!='') {

            //$obj = new ModuleAksi();

            $obj = $this->model_ref_coa_jenis;

            $result = $obj->update_data($_POST['p_kode'], $_POST['p_deskripsi'],
$_POST['p_periode_akun'], $_POST['p_key_id']);

            if ($result) $err = '&err=3'; else $err = '&err=4';

          } else {

            $err = '&err=1&id='.$_POST['p_key_id'];

            $sub = 'update';

          }

        }

        return Dispatcher::Instance()->GetUrl('ref_coa_jenis', $sub, 'view', 'html') . $err;

      }

    }
```

Kemudian kita perlu menyesuaikan file model agar modul bisa melakukan *create*, *update* dan

delete database dengan menambahkan fungsi-fungsi untuk melakukan proses tersebut dan melengkapi model dengan *query* sesuai dengan fungsinya. Tambahkan *script* berikut pada file **model_ref_coa_jenis.php** :

```
    function write_data($p_kode, $p_deskripsi, $p_periode_akun){

           return $this->Execute($this->mSqlQueries['write_data'], array($p_periode_akun, $p_kode,
$p_deskripsi));

        }



        function update_data($p_kode, $p_deskripsi, $p_periode_akun, $p_key_id){

          return $this->Execute($this->mSqlQueries['update_data'], array($p_periode_akun, $p_kode,
$p_deskripsi, $p_key_id));

        }



        function delete_data_by_id($p_key_id){

            return $this->Execute($this->mSqlQueries['delete_data_by_id'], array($p_key_id));

       }
```

Kemudian lengkapi *query* pada file **sql_ref_coa_jenis.php** dengan script berikut :

```
    $sql['write_data']="

      insert into tref_coa_jenis(periode_akun_key_id, cj_kd, cj_deskripsi)

      values(%s, %s, '%s')";



    $sql['update_data']="

      update tref_coa_jenis set periode_akun_key_id = %d, cj_kd = %d, cj_deskripsi = '%s'
```

```
      where key_id = %d";


    $sql['delete_data_by_id']="
      delete from tref_coa_jenis where key_id = %d";
```

Langkah terakhir adalah mendaftarkan modul yang telah dibuat. Caranya sama dengan mendaftarkan modul sebelumnya.

-SEKIAN DAN TERIMAKASIH-