

1 HARI 5: INTERNATIONALIZATION

Setiap module bisa punya translasi bahasa yang berada di direktori `i18n`, supaya kalau odoo digunakan oleh user menggunakan bahasa tertentu, maka bahasa modul kita juga ikut berubah sesuai bahasa yang dipilih.

Direktori ini berisi file kamus bahasa bernama `LANG.po` dimana `LANG` adalah locale code bahasa yang mau disediakan. Bisa juga berupa kombinasi bahasa dan negara kalo berbeda (misalnya `pt.po` atau `pt_BR.po` untuk bahasa Portugis dan Protugis di Brazil).

Translasi akan di-load otomatis oleh odoo untuk semua bahasa yang disediakan oleh module.

Developer harus pake bahasa English waktu membuat modul, lalu export semua istilah (label, nama field, dll) yang ada pada module pake fitur export gettext POT untuk membentuk file template POT, lalu meng-copy PO files yang udah diexport ke dalam folder `i18n` module.

1.1 BIKIN DIREKTORI `i18n`

Bikin dulu direktori `i18n` dibawah direktori `addons academic`.

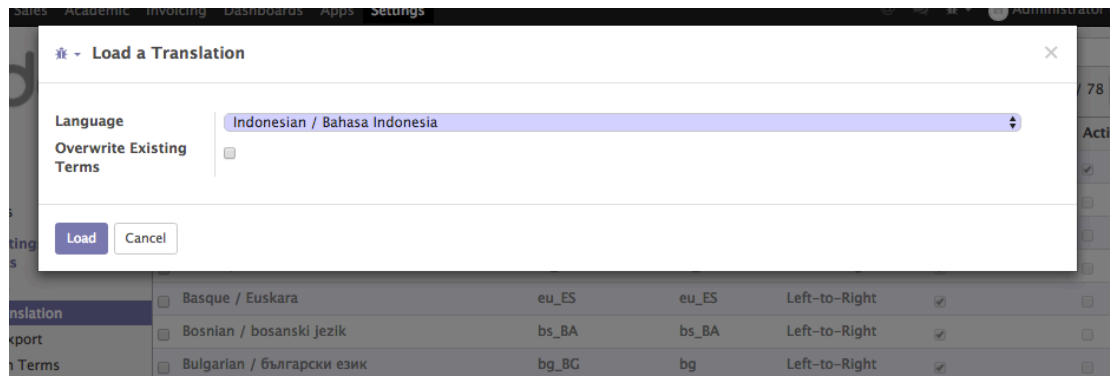
1.2 BAHASA UDAH ADA DI ODOO

1.2.1 INSTALL BAHASA TARGET

Kalo bahasa yang mau kita translate udah ada di odoo, maka instal dulu bahasa itu melalui menu odoo:

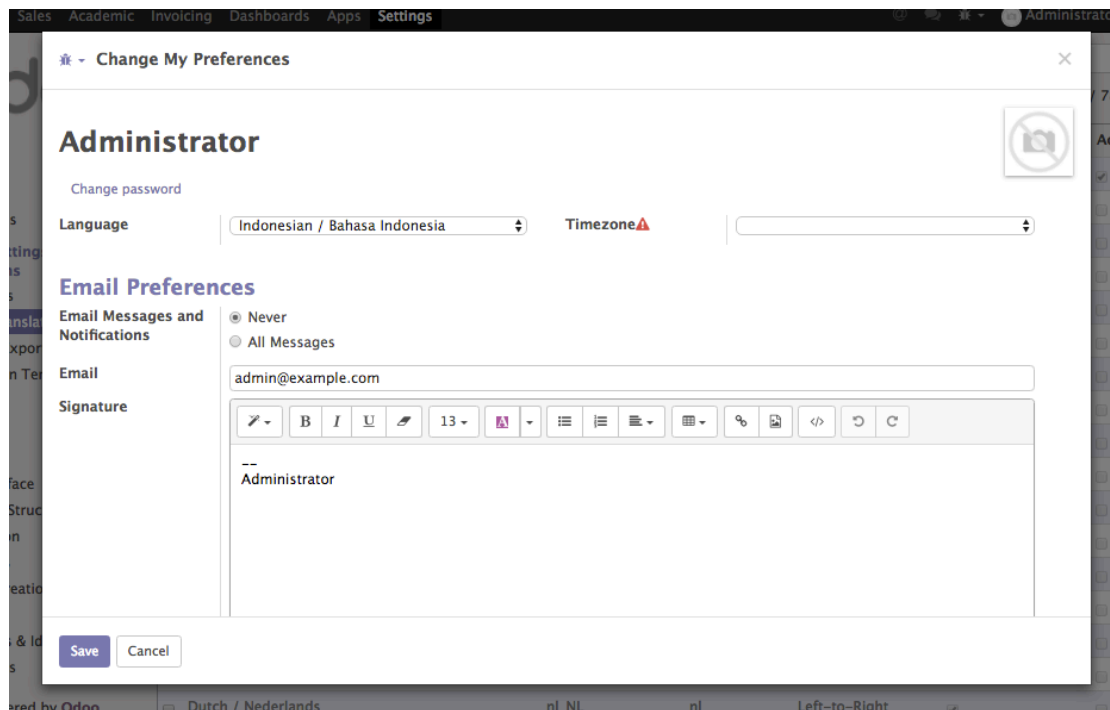
Klik menu `Settings > Translations > Load a Translation`

Pilih Bahasa Indonesia.



Gambar 1 Load Transaltion Bahasa Indonesia

Bahasa Indonesia siap digunakan oleh User odoo dan bisa dipilih melalui Preference masing-masing User.



Gambar 2 Preference user

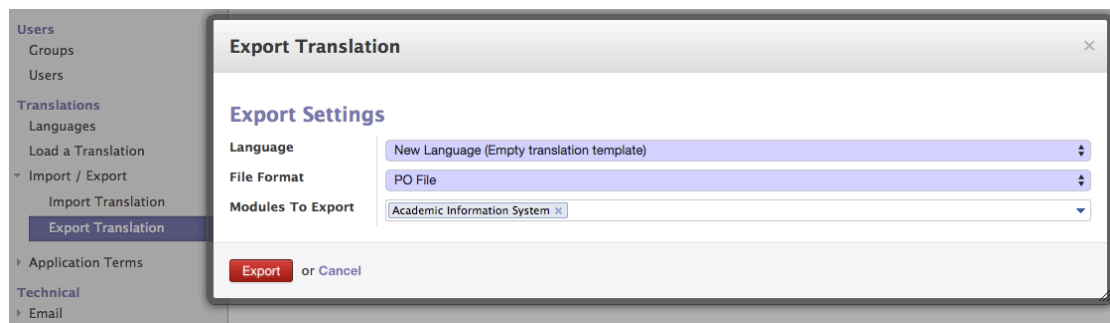
1.3 BAHASA BELUM ADA DI ODOO

1.3.1 BIKIN TEMPLATE TRANSLATE ACADEMIC.POT

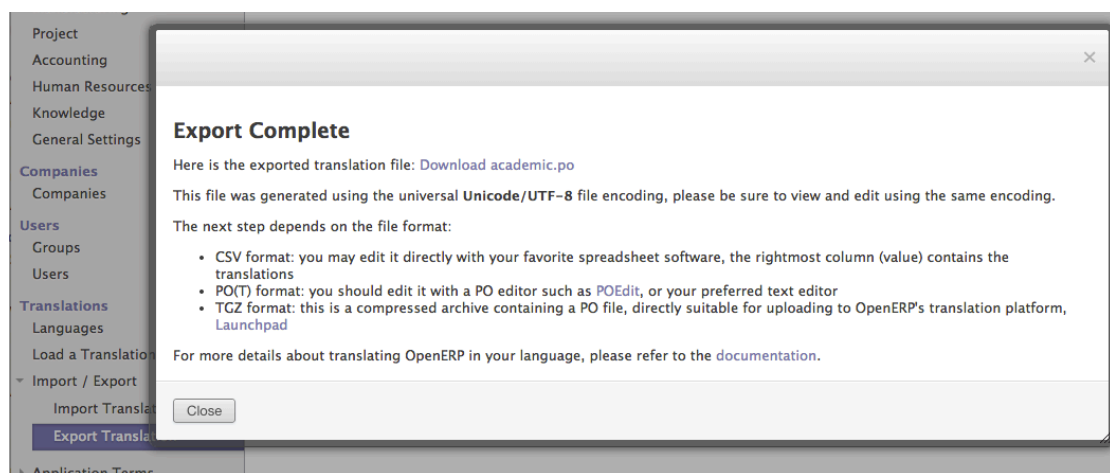
Bagian ini khusus kalau bahasa yang mau ditranslate belum ada, jadi harus bikin template-nya dulu.

Export template file translate “academic.pot” lewat menu
Settings > Translations > Import/Export >

Export Translation tanpa pilih bahasanya, terus simpan di direktori i18n.



Gambar 3 Export translation



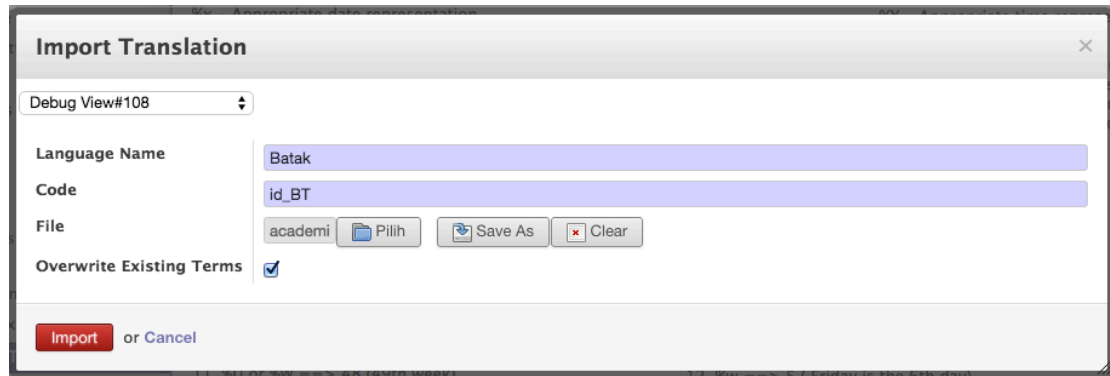
Gambar 4 Download file template academic.po

Terus rename file `academic.po` ke `academic.pot`. simpan di direktori `i18n`.

1.3.2 IMPORT KE ODOO

Kita perlu create bahasa yang baru di odoo dengan cara import file translation yang baru dibuat tadi.

Klik menu `Settings > Translations > Import/Export > Import Translation`.



Gambar 5 Import translation

Field **Language Name** adalah nama Bahasa yang akan dibuat.

Field **Code** adalah code ISO bahasa.

Field **File**, pilih file template yang barusan di download tadi, yaitu **academic.pot**.

Lanjut, bahasa yang baru dibuat muncul di daftar Languages...

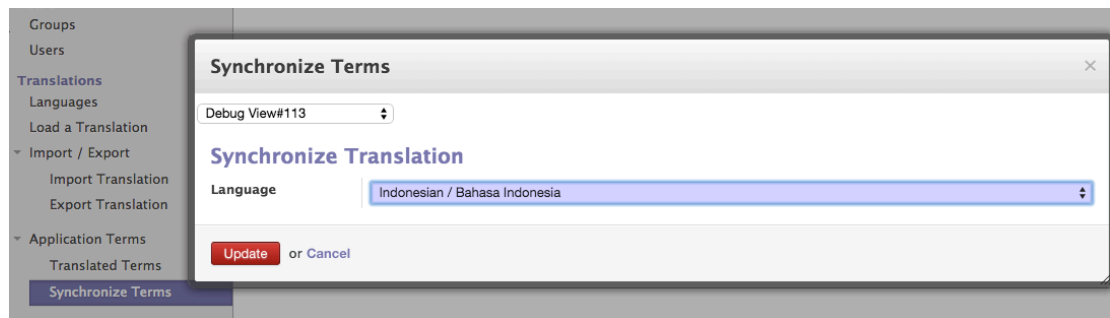
Languages						
Debug View#122						
Create or Import						
PDF or Excel 1-5 of 5						
<input type="checkbox"/>	Name	Locale Code	ISO code	Direction	Translatable	Active
<input type="checkbox"/>	English	en_US		Left-to-Right	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="checkbox"/>	English (UK)	en_GB	en_GB	Left-to-Right	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="checkbox"/>	Indonesian / Bahasa Indonesia	id_ID	id	Left-to-Right	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="checkbox"/>	Jawa	id_JW	id_JW	Left-to-Right	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="checkbox"/>	Batak	id_BT	id_BT	Left-to-Right	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Gambar 6 Daftar bahasa yang tersedia di sistem

Lanjut ke bagian dibawah untuk proses translate.

1.4 SINKRONISASI ISTILAH

Sinkronisasi dulu istilah yang hendak di translate melalui menu **Settings > Translations > Application Terms > Synchronize Translations**.



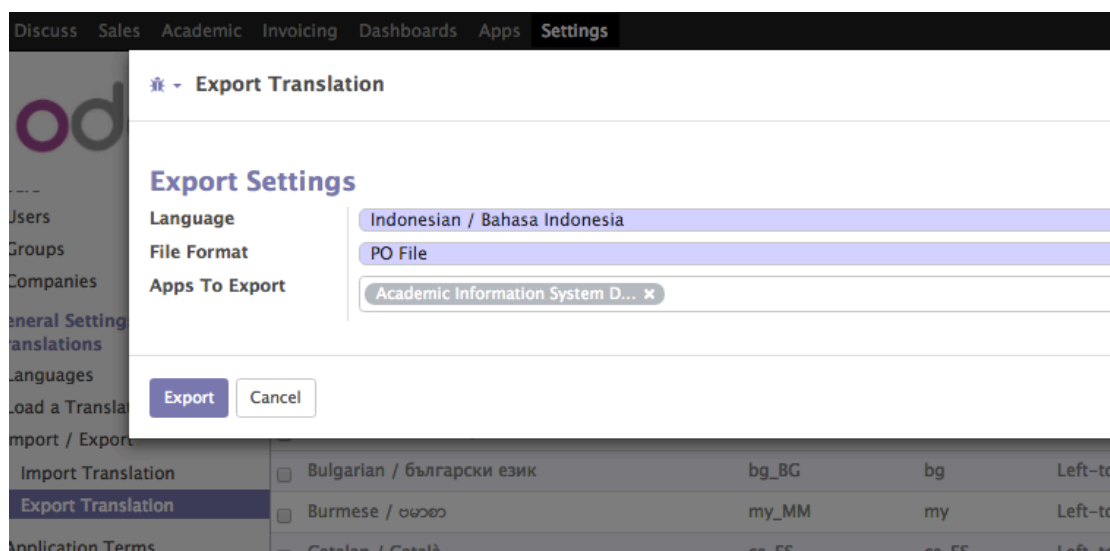
Gambar 7 Synchronize terms

1.5 BIKIN FILE TEMPLATE PER BAHASA

Bikin lagi file translate untuk Bahasa Indonesia `id.po` melalui export.

Klik menu `Settings > Translations > Import/Export > Export Translation`.

Pilih Bahasa Indonesia, simpan juga filenya di directori `i18n`.



Gambar 8 Export translation per module

Struktur file module kita sejauh ini harus seperti berikut, ada tabahan folder `i18n` dan file-file po didalamnya...

```

|-- __init__.py
|-- __openerp__.py
|-- attendee.py
|-- attendee.xml
|-- course.py
|-- course.xml
|-- i18n
|   |-- academic.po
|   `-- id.po
|-- menu.xml
|-- partner.py
|-- partner.xml
|-- security
|   |-- group.xml
|   `-- ir.model.access.csv
|-- session.py
|-- session.xml
|-- wizard
|   |-- __init__.py
|   |-- create_attendee.py
|   `-- create_attendee_view.xml
`-- workflow.xml

```

1.6 TERJEMAHIN FILE TEMPLATE BAHASA

Buka file “`id.po`” pake *poedit* (atau sembarang text editor) dan lakukan translate istilah-istilah yang ada dengan mengganti msgstr setiap istilah.

Misalnya:

```

#. module: academic
#: field:academic.session,taken_seats:0
msgid "Taken Seats"
msgstr "Tempat Sudah Terpakai"

#. module: academic
#: view:academic.create.attendee.wizard:0
msgid "Add attendees"
msgstr "Tambah Peserta"

```

```

1 # Translation of OpenERP Server.
2 # This file contains the translation of the following modules:
3 # * academic
4 #
5 msgid ""
6 msgstr ""
7 "Project-Id-Version: OpenERP Server 7.0-20130829-231103\n"
8 "Report-Msgid-Bugs-To: \n"
9 "POT-Creation-Date: 2014-09-04 09:19+0000\n"
10 "PO-Revision-Date: 2014-09-04 09:19+0000\n"
11 "Last-Translator: <>\n"
12 "Language-Team: \n"
13 "MIME-Version: 1.0\n"
14 "Content-Type: text/plain; charset=UTF-8\n"
15 "Content-Transfer-Encoding: \n"
16 "Plural-Forms: \n"
17
18 #. module: academic
19 #: field:academic.session,taken_seats:0
20 msgid "Taken Seats"
21 msgstr "Taken Seats"
22
23 #. module: academic
24 #: view:academic.create.attendee.wizard:0
25 msgid "Add attendees"
26 msgstr "Add attendees"
27
28 #. module: academic
29 #: selection:academic.session,state:0
30 msgid "Confirmed"
31 msgstr "Confirmed"

```

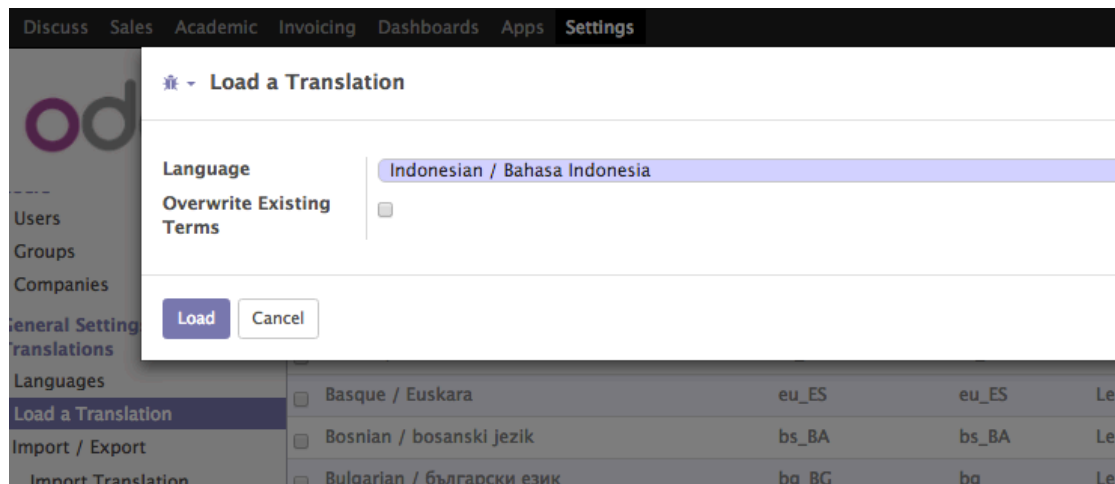
Gambar 9 File template bahasa yang harus diterjemahin

1.7 RELOAD BAHASA INDONESIA

Setelah edit file terjemahan `id.po` selesai semua istilah, data ini perlu di upload ke database supaya module bisa ditampilkan dalam Bahasa Indonesia.

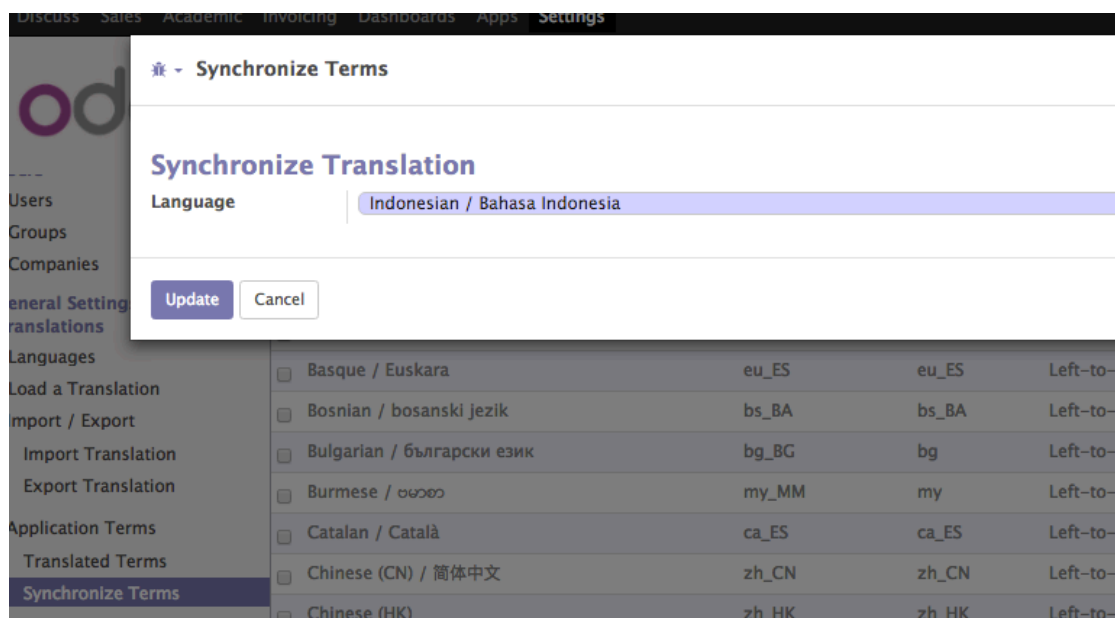
Caranya:

1. Upload module
2. Klik menu `Settings > Translations > Load Translation`, pilih Bahasa Indonesia



Gambar 10 Reload bahasa Indonesia

- Klik menu Settings > Translations > Application Terms > Synchronize Term



Gambar 11 Synchronize Terms

Selesai ! Module addons academic udah bisa dipakai jika user pilih Bahasa Indonesia.

1.8 ISTILAH TAMBAHAN

Default-nya odoo POT export hanya membaca label di dalam file XML dan pada definisi field di Python code. Tapi sebetulnya

semua string Python dapat dibaca asal udah diapit dengan `_()`,
misalnya `_('Label')`.

Modif file `session.py`, `attendee.py`, `course.py`, tambahi
import paket berikut..

```
from tools.translate import _
```

Udah itu tambah operator `"_"` di setiap tempat yang diperlukan,
lalu ulangi langkah mulai dari Synchronize Terms di atas.

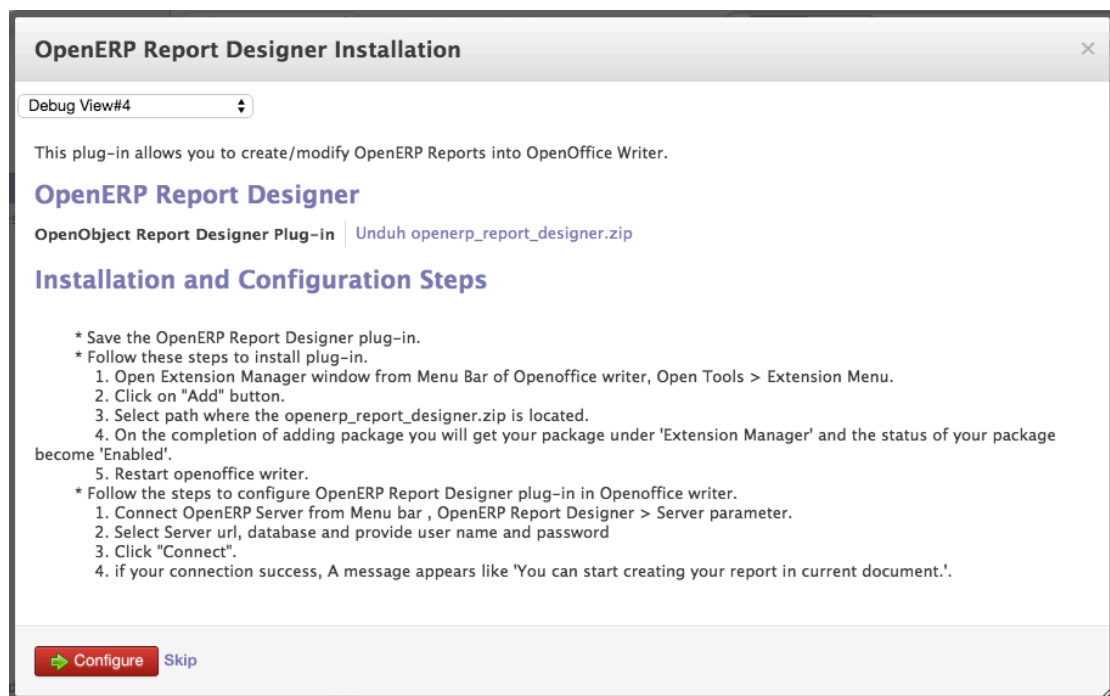
2 REPORT RML

Catatan: Sistem report RML hanya berlaku di OpenERP versi 7.0.
Untuk versi 10, report menggunakan WEB.

2.1 INSTALASI

Instal module `base_report_designer`.

Setalah instal berhasil, akan muncul popup dimana kita bisa download Report Designer Plug-in, yaitu file `odoo_report_designer.zip`.



Gambar 12 Install module `base_report_designer`

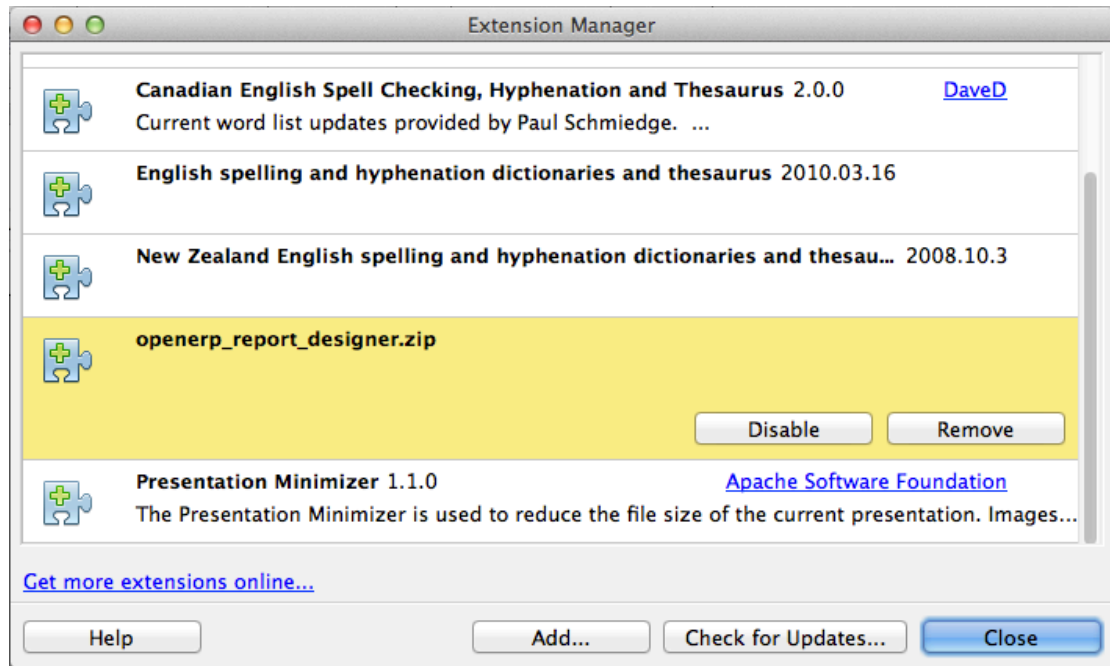
2.2 INSTALL PLUG-IN DI OPENOFFICE

Buka window Extension Manager dari Menu Bar Openoffice Writer, yaitu menu `Tools > Extension` Menu.

Klik tombol `Add`.

Pilih lokasi file `odoo_report_designer.zip` yang tadi udah didownload.

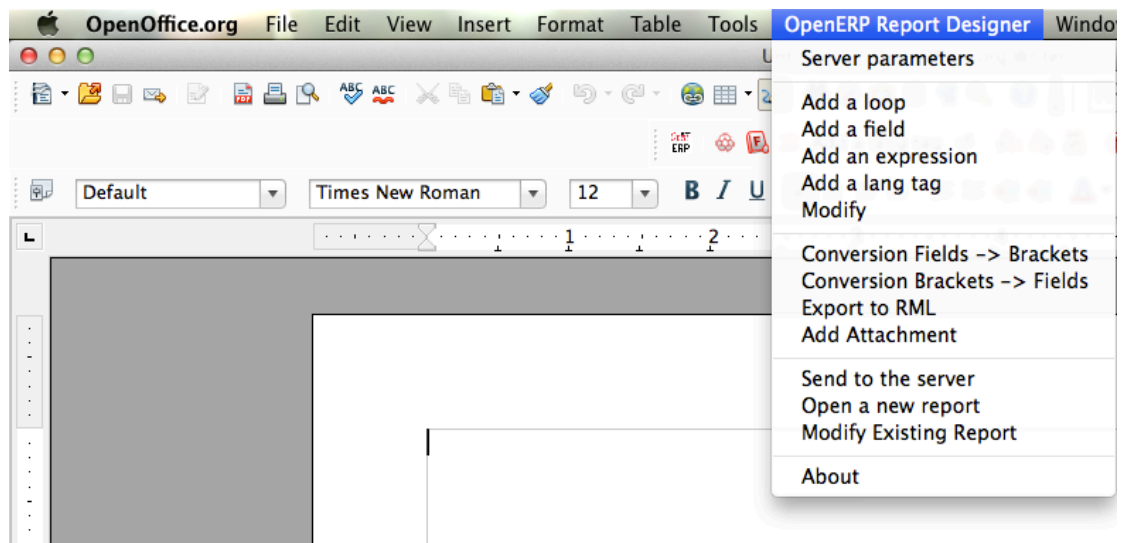
Setelah selesai nambah package kita bisa lihat di daftar 'Extension Manager' dan statusnya harus 'Enabled'.



Gambar 13 Install plug in OpenOffice

Restart OpenOffice writer.

Menu odoo Report Designer muncul di OpenOffice.



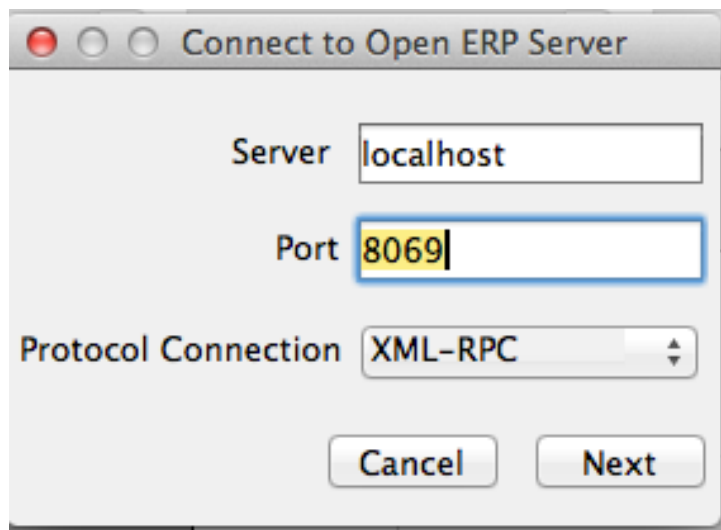
Gambar 14 Menu odoo Report Designer pada OpenOffice

2.3 KONFIGURASI

Berikut ini langkah konfigurasi plug-in odoo Report Designer pada Openoffice writer.

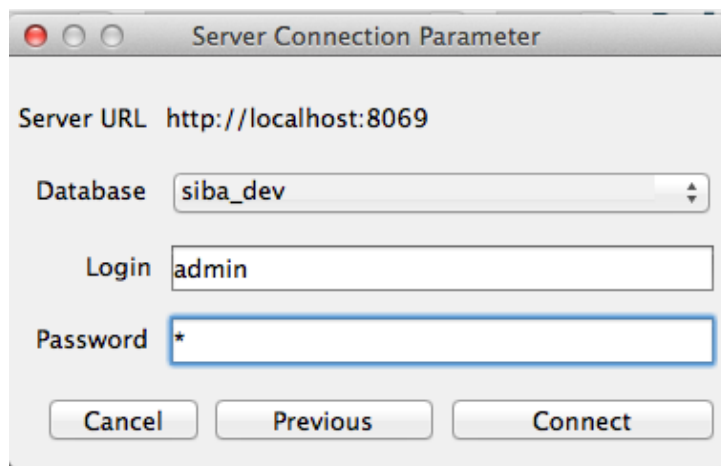
Connect ke odoo Server lewat Menu bar **odoo Report Designer > Server parameter.**

Masukkan Server URL, database, user name, dan password untuk masuk ke odoo.



Gambar 15 Koneksi ke server

Klik "Next".



Gambar 16 Login ke server

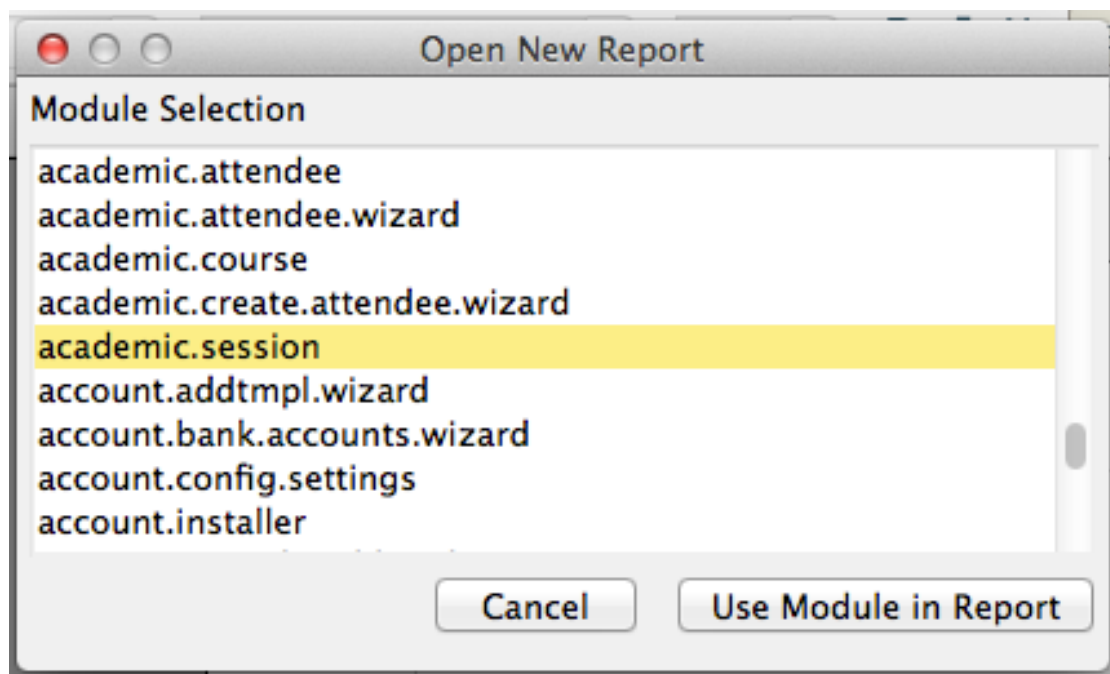
Klik "Connect". Jika berhasil akan muncul pesan 'You can start creating your report in current document.'.

2.4 BIKIN REPORT BARU

Coba kita bikin report untuk Session object, yang menampilkan data Session name, date, duration, duration, responsible name serta daftar peserta yang hadir (Attendees).

Klik menu di OpenOffice: `odoo Report Designer > Open a New Report`.

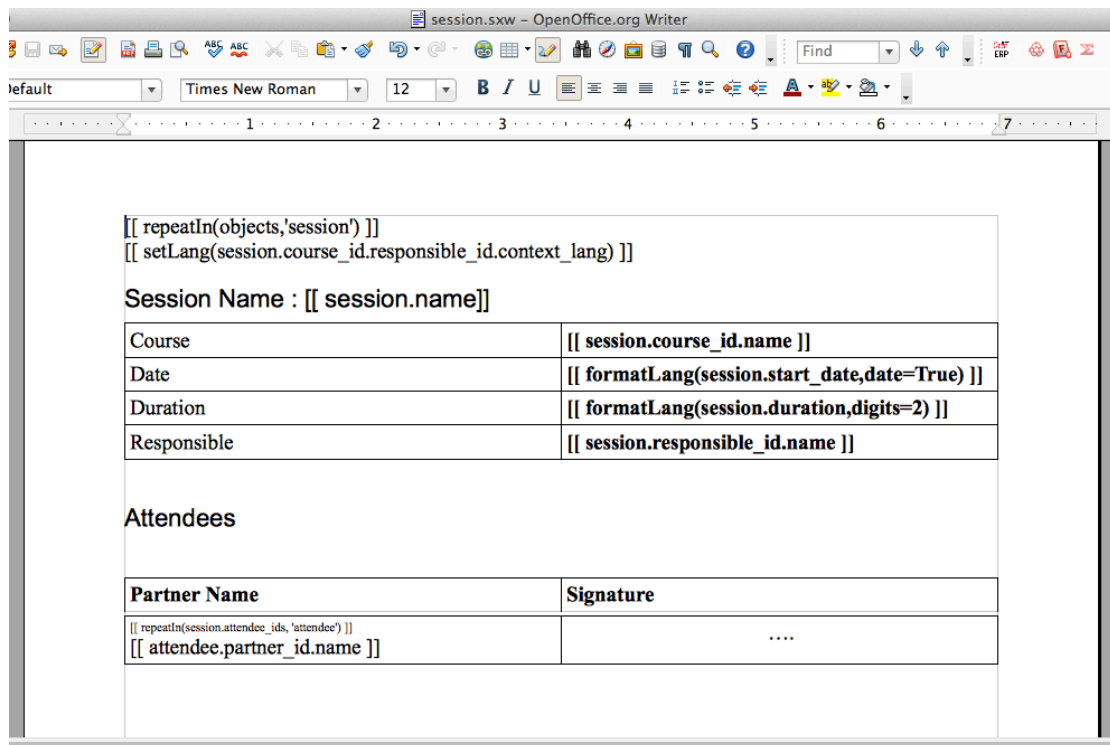
Pilih nama object yang mau dibuat reportnya, yaitu `academic.session`.



Gambar 17 Bikin report baru

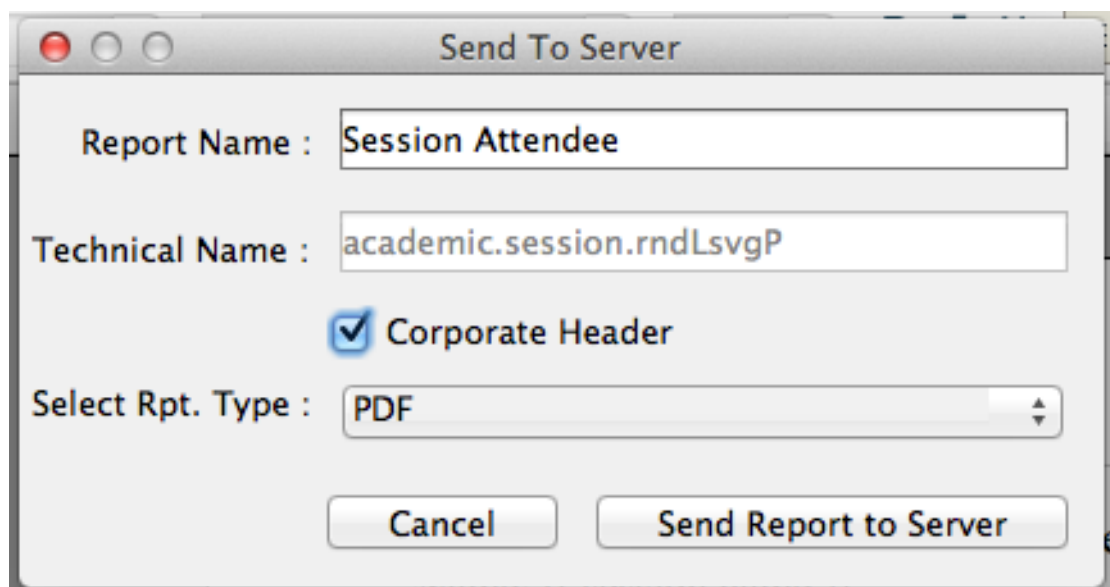
Klik `Use Module in Report`.

Bikin template seperti dibawah ini pada OpenOffice Writer.



Gambar 18 Template di OpenOffice Writer


Save file, lalu klik menu odoo Report Designer > Send Report to Server.



Gambar 19 Kirim report ke server

Otomatis akan muncul menu **Print > Session Attendee** di form view Session.

S1

Course Java
Instructor Budi
Start Date 09/05/2014
Image Small 

Duration 33
Number of Seats 5
Is Active? ☒
Taken Seats 180.00%

Attendees

Name	Partner
00167	Agus
00172	Joko

Gambar 20 Report langsung tersedia melalui tombol Print di form view Session

Ketika diklik akan terbentuk file PDF yang hasilnya sesuai dengan template yang udah dibuat di OpenOffice.

Phone: 12345678, 89900020
Mail: info@yourcompany.com

Session Name : S1

Course	Java
Date	09/05/2014
Duration	33.00
Responsible	

Attendees

Partner Name	Signature
Agus
Joko
Joko
Agus
Cabang A
Customer
Administrator
Administrator
Joko

Gambar 21 Report PDF

2.5 SYNTAX TEMPLATE

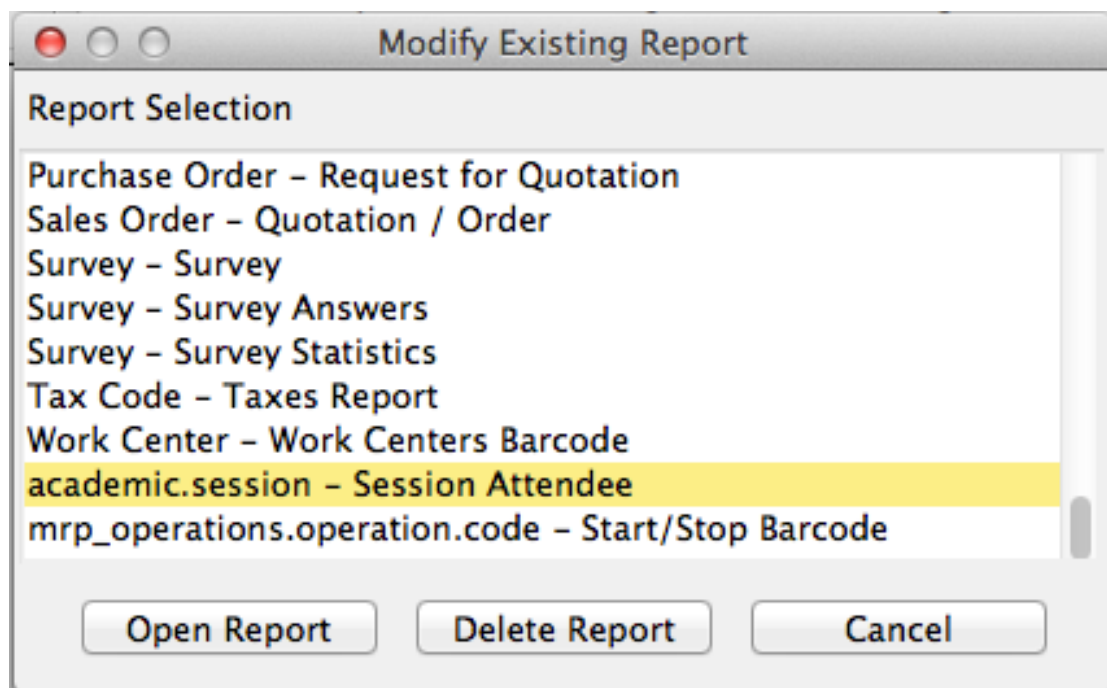
Expresi yang boleh dipakai pada report templates odoo:

<i>objects</i>	Berisi array list dari record yang mau di-print
<i>data</i>	Berasal dari wizard yang menjalankan report
<i>user</i>	User yang sedang login (as per browse())
<i>time</i>	Python time module
<i>repeatIn(list,'var','tag')</i>	Ulangi parent element dimana dia dipanggil dan bernama tag untuk setiap object yang ada pada list, dan membuat object tersebut tersedia sebagai var selama loop
<i>setTag('tag1','tag2')</i>	Ganti parent RML tag1 dengan tag2
<i>removeParentNode('tag')</i>	Hapus parent RML element bernama tag

<code>formatLang(value, digits=2, date=False, date_time=False, grouping=True, monetary=False)</code>	Untuk mem-format date, time atau nominal uang sesuai locale
<code>setLang('lang_code')</code>	Se bahasa dan locale untuk penterjemahan

2.6 MODIFY EXISTING REPORT

Dari menu `odoo Report Designer > Modify Existing Report`, pilih report yang akan dimodif



Gambar 22 Modif report

Akan terbuka lagi template asli report tersebut, lakukan edit seperti sebelumnya.

Jika sudah selesai, kirim balik ke server lewat menu `odoo Report Designer > Send Report to Server`.

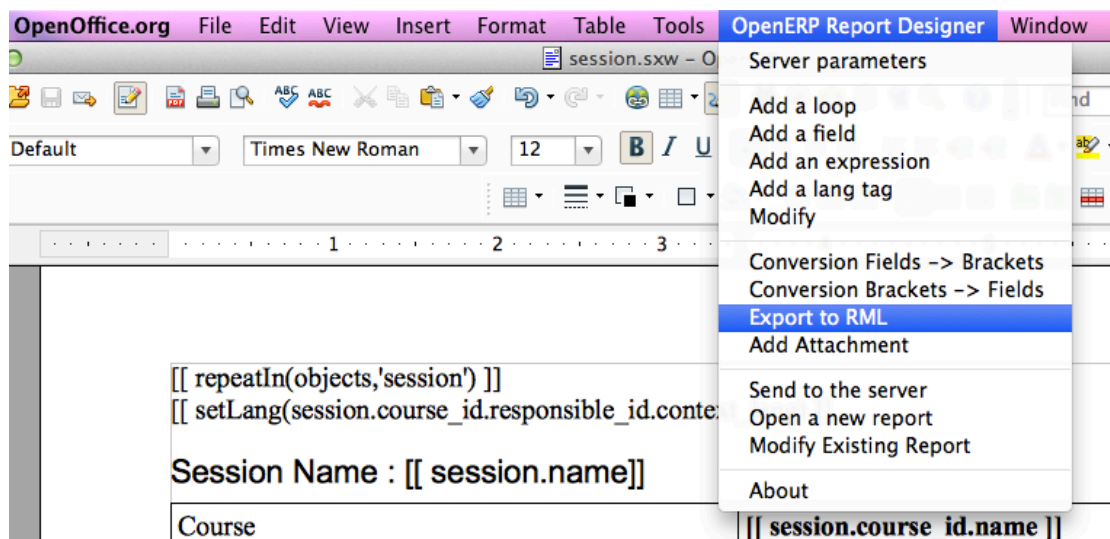
Semua report odoo yang tersedia di atas bisa diedit untuk penyesuaian sesuai kebutuhan perusahaan masing-masing.

2.7 REPORT DI ADDONS – RML

Buat folder “report” di dalam folder addons academic.

Export OpenOffice report template ke sebuah file RML dan simpan file RML di dalam report folder.

Save sebagai `session_attendee.rml`.



Gambar 23 Export ke RML

Tambah satu file XML baru untuk deklarasi report. Kasi nama misalnya `report/session_report.xml`. Isinya seperti ini.

```

1  <openerp>
2      <data>
3          <report id="session_report"
4              string="Print Session Report"
5              model="academic.session"
6              name="session.report"
7              rml="academic/report/session.rml"
8          />
9      </data>
10 </openerp>

```

Gambar 24 Bikin tag report

Edit file `__openerp__.py` dan tambahi file `report/session_report.xml`.

```

1  {
2      "name": "Academic Information System",
3      "version": "1.0",
4      "depends": ["base", "board"],
5      "author": "Author Name",
6      "category": "Education",
7      "description": """\
8  this is my academic information system module
9  """,
10     "data": ["menu.xml",
11             "course.xml",
12             "session.xml",
13             "attendee.xml",
14             "partner.xml",
15             "workflow.xml",
16             "security/group.xml",
17             "security/ir.model.access.csv",
18             "report/session_report.xml",
19     ],
20     "installable": True,
21     "auto_install": False,
22 }

```

Gambar 25 Include report/session_report.xml

Restart odoo dan update module. Maka report Session akan tersedia di Session form view.

3 REPORT WEBKIT

Catatan: Sistem report Webkit berlaku di OpenERP versi 7.0 dan 8.0. Untuk versi 9/10, semua report menggunakan QWEB.

3.1 INSTALASI

Instal `report_webkit` report module.

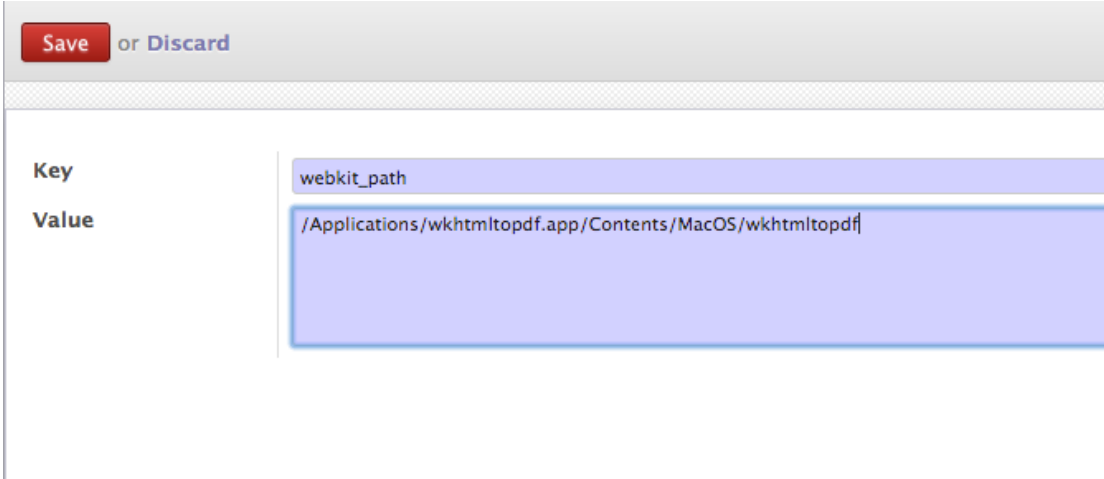
Instal program `wkhtmltopdf`.

Modul ini perlu program tambahan namanya `wkhtmltopdf` yang gunanya untuk menkonvert HTML ke PDF. Minimal version 0.9.9 download dulu dari

<http://code.google.com/p/wkhtmltopdf>. Tersedia untuk Linux, Mac OS X (i386) dan Windows (32bits).

Abis instalasi di server odoo, kita perlu menge-set PATH `wkhtmltopdf` di database yang dipakai.

Caranya lewat menu `Settings > Technical > System Parameters..`



Key	Value
webkit_path	/Applications/wkhtmltopdf.app/Contents/MacOS/wkhtmltopdf

Gambar 26 Setup system parameter `webkit_path`

Tambahkan:

Key = `webkit_path`

Value = lokasi program `wkhtmltopdf` di server

3.2 TEST BIKIN REPORT MANUAL

3.2.1 CREATE TEMPLATE

Klik menu **Settings > Technical > Action > Reports**.

Klik **Create New** .

Debug View#9 ▼ **Reports / Summary Session** ?

Save or Discard 2 / 2 ◀ ▶ ☰ □

Name	Summary Session	Action Usage	
Service Name	my_openacademy.session.s34829	Report Type	webkit
Object	my_openacademy.session	Report File	

Other Configuration Webkit Security

Base Sample ▼ ↗

Webkit Template (used if Report File is not found)

```
% for o in objects :  
<html>  
<head>  
<style type="text/css">
```

Gambar 27 Bikin action report manual

Masukkan data sebagai berikut:

Field **Name**: the report name

Field **Service Name**: <model>.<unique number>

Field **Object**: model name

Field **Report Type**: ketik webkit

Pada **Webkit** tab:

Field **Header**: select one of the available header.

Field **The Webkit Template**: ketikkan report template dalam format HTML, misalnya:

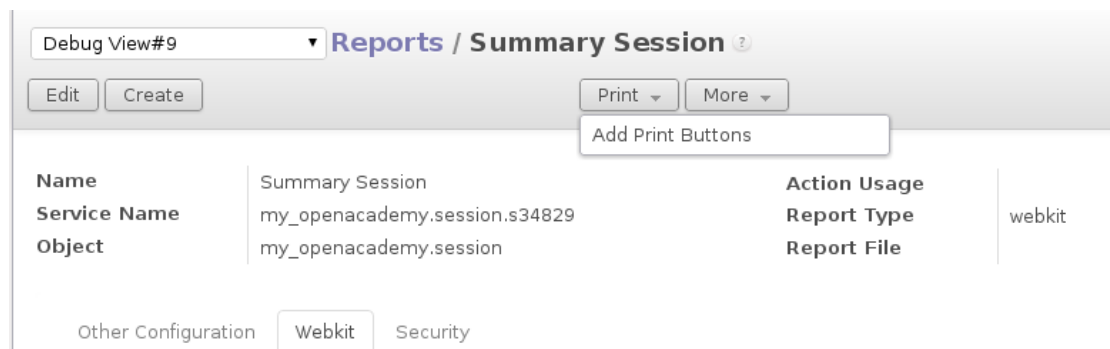
```
1 % for o in objects :
2 <html>
3 <head>
4     <style type="text/css">
5         ${css}
6         .page {page-break-after: always}
7     </style>
8 </head>
9
10 <body>
11     <div class="page">
12         <h1>${o.name}</h1>
13         <h2>Attendees</h2>
14         <table border="1" width="100%">
15             % for a in o.attendee_ids:
16                 <tr><td>${a.partner_id.name}</td></tr>
17             % endfor
18         </table>
19     </div>
20 </body>
21
22 </html>
23 % endfor
24
```

Gambar 28 Template mako

3.2.2 BIKIN ACTION BUTTON

Simpan Action Report di atas.

Klik tombol **Print > Add Print Buttons**



Gambar 29 Bikin tombol print

Ini akan membuat record Action Binding secara otomatis sehingga tombol Print akan muncul pada form dan list view model yang bersangkutan.

Debug View#322 Sessions / S3

Edit Create Print More

1 / 32

Reset to Draft Reject Summary Session Draft Confirmed Done

S3

Schedule

Start Date	03/15/2013
End Date	03/16/2013
Duration	2

Instructor

Number of Seats	100
Instructor	Agrolait2
Taken Seats	14.00%

Gambar 30 Tombol print muncul di form view Session

3.3 INSTALL REPORT DARI ADDONS

3.3.1 BUAT FILE TEMPLATE

Disini kita akan membuat report Webkit untuk Session object yang menampilkan session name, date, duration, durasi, responsible name dan daftar nama Attendees.

Di dalam folder `academic`, buat file `report/session.mako` isinya seperti ini.

```
1 <html>
2 <head>
3 <style type="text/css">${css}</style> </head>
4 <body>
5 <h1>Session Report</h1>
6 % for session in objects:
7 <h2>${session.course_id.name} - ${session.name}</h2>
8 <p>From ${formatLang(session.start_date, date=True)}</p>
9 <p>Attendees:
10 <ul>
11 % for att in session.attendee_ids:
12 <li>${att.partner_id.name}</li>
13 % endfor
14 </ul>
15 </p>
16 % endfor
17 </body>
18 </html>
```

Gambar 31 Template mako

3.3.2 BUAT XML

Buat file `report/session.xml` yang berisi deklarasi record report action:

```
1  <openerp>
2    <data>
3
4      <report id="report_webkit"
5              model="academic.session"
6              name="academic.session.report"
7              file="academic/report/session.mako"
8              string="Session Report"
9              report_type="webkit"
10             />
11    </data>
12  </openerp>
13
```

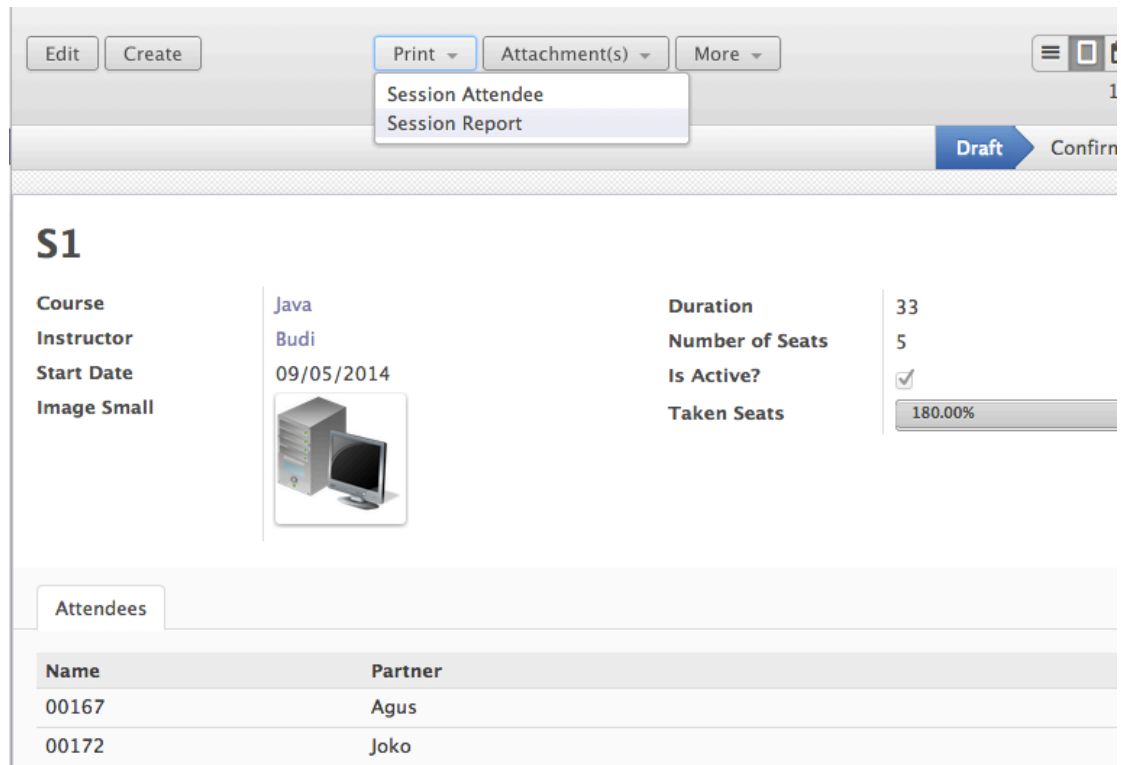
Gambar 32 Bikin action report di XML

Tambahkan file XML diatas pada `__openerp__.py`:

```
1  {
2    "name": "Academic Information System",
3    "version": "1.0",
4    "depends": ["base"],
5    "author": "Author Name",
6    "category": "Education",
7    "description": """"\
8  this is my academic information system module
9  """"
10   "data": ["menu.xml",
11            "course.xml",
12            "session.xml",
13            "attendee.xml",
14            "partner.xml",
15            "workflow.xml",
16            "security/group.xml",
17            "security/ir.model.access.csv",
18            "wizard/create_attendee_view.xml"
19            "report/session.xml"
20          ],
21   "installable": True,
22   "auto_install": False,
23 }
```

Gambar 33 Panggil dari `__openerp__.py`

Restart odoo dan update module, hasilnya:



The screenshot shows the Odoo Session Report interface. At the top, there is a header bar with buttons for 'Edit', 'Create', 'Print', 'Attachment(s)', and 'More'. The 'Print' button is highlighted, and a dropdown menu is open showing 'Session Attendee' and 'Session Report'. Below the header, there is a 'Draft' button and a 'Confirm' button. The main content area displays session details for 'S1'. The details are organized into two columns. The left column contains 'Course' (Java), 'Instructor' (Budi), 'Start Date' (09/05/2014), and 'Image Small' (a computer icon). The right column contains 'Duration' (33), 'Number of Seats' (5), 'Is Active?' (checked), and 'Taken Seats' (180.00%). Below the details, there is a tab labeled 'Attendees'. Under the 'Attendees' tab, there is a table with two columns: 'Name' and 'Partner'. The table contains two rows of data.

Name	Partner
00167	Agus
00172	Joko

Gambar 34 Muncul tombol print report webkit

Muncul menu report baru di bawah tombol Print berupa report webkit. Klik menu report akan muncul report dalam bentuk PDF.

Phone: 12345678, 89900020
Mail: info@yourcompany.com

Session Report

Java - S1

From 09/05/2014.

Attendees:

- Agus
- Joko
- Joko
- Agus
- Cabang A
- Customer
- Administrator
- Administrator
- Joko

Gambar 35 Report PDF

4 REPORT QWEB

Sejak versi 9 dan 10, semua reporting Odoo menggunakan QWEB.

Untuk membuat report QWEB terhadap semua object Odoo, lakukan langkah-langkah berikut...

4.1 BUAT FOLDER REPORT

Semua file report akan ditempatkan di dalam folder ini supaya nggak tercampur dengan file-file XML lainnya..

4.2 BUAT FILE XML REPORT UNTUK MENU REPORT

Next, buat file XML report khusus per object, contohnya untuk object session, dikasi nama session.xml.

Di dalam file ini, ada beberapa record yang perlu dicantumin, pertama...

```
<?xml version="1.0" encoding="utf-8"?>
<openerp>
  <data>

    <report
      id="report_session_menu"
      string="Session"
      model="academic.session"
      report_type="qweb-pdf"
      file="academic.session"
      name="academic.session_report"
    />
```

ini adalah definisi record untuk menu action report pada object academic.session (field model).

Label menu tersebut adalah Session (field string)...

Type report adalah qweb-pdf, dan file PDF nya nanti bernama academic.sessin (field report_type dan file)..

Terakhir, field name, nenentukan nama template QWEB yang akan dipanggil pada saat menu action dipanggil, yaitu **academic.session_report**... Disini academic adalah nama folder addons kita.

4.3 XML RECORD UNTUK TEMPLATE QWEB

Lanjut, buat record XML untuk definisi template yang dipanggil di menu action diatas, yaitu yang ID nya academic.session_report...

```
<template id="session_report">
  <t t-call="report.html_container">
    <t t-foreach="docs" t-as="doc">
      <t t-call="academic.session_report_document"/>
    </t>
  </t>
</template>
```

Ini template isinya cuma manggil template lain lagi yaitu report.html_container untuk memanggil header dan footer standard report Odoo.

Lalu dilakukan looping foreach untuk setiap variabel docs. Variable ini adalah record list dari Session yang hendak diprint, baik melalui form view maupun list view dengan cara men-select 1 atau lebih Session yang mau diprint.

Pada setiap looping variabel docs, kita simpan ke local variabel doc, lalu kita panggil lagi template lain yang namanya academic.session_report_document.

Pada template terakhir inilah layout report per satu record session kita definisikan bentuknya seperti apa.

4.4 XML RECORD UNTUK REPORT PER RECORD SESSION

Lanjut, ini definisi layout report per setiap satu record session yang hendak diprint.

```
<template id="session_report_document">
  <t t-call="report.external_layout">
    <t t-set="doc" t-value="doc.with_context(
      {'lang':doc.instructor_id.lang})" />
    <div class="page">
      <div class="oe_structure"/>
      <h2>
```

```

        SESSION: <span t-field="doc.name"/>
    </h2>

    <table class="table table-condensed">
        <tr>
            <td>Course</td>
            <td><span t-field="doc.course_id"/></td>
            <td>Instructor</td>
            <td><span t-field="doc.instructor_id"/></td>
        </tr>
        <tr>
            <td>Start Date</td>
            <td><span t-field="doc.start_date"/></td>
            <td>Duration</td>
            <td><span t-field="doc.duration"/></td>
        </tr>
        <tr>
            <td>Taken Seats</td>
            <td><span t-field="doc.taken_seats"/></td>
            <td>Active</td>
            <td><span t-field="doc.active"/></td>
        </tr>
    </table>

    <table class="table table-condensed">
        <thead>
            <tr>
                <th>No</th>
                <th>Partner</th>
                <th>Signed</th>
            </tr>
        </thead>
        <tbody>
            <tr t-foreach="doc.attendee_ids" t-as="l">
                <td>
                    <span t-field="l.name"/>
                </td>
                <td>
                    <span t-field="l.partner_id" />
                </td>
                <td>
                </td>
            </tr>
        </tbody>
    </table>

    <div class="oe_structure"/>
</div>
</t>
</template>

```

Inti dari template di atas adalah...

Bahwasanya template ini mendapat variabel doc yang isinya record session yang hendak diprint, dari object ini kita bisa panggil semua field yang ada pada session.

Pertama-tama kita panggil template external_layout bawaan Odoo untuk mengatur layout standard report Odoo..

```
<template id="session_report_document">
  <t t-call="report.external_layout">
```

Lalu kita set bahasa yang akan digunakan pada report sesuai dengan bahasanya Instruktur...

```
<t t-set="doc" t-value="doc.with_context(
    {'lang':doc.instructor_id.lang})" />
```

Kemudian kita gunakan class CSS page untuk ukuran kertas, keluarkan tag HTML H2 untuk menampilkan judul report...

```
<div class="page">
  <div class="oe_structure"/>
  <h2>
    SESSION: <span t-field="doc.name"/>
  </h2>
```

Kemudian.. kita buat table untuk menampilkan data header session...

```
<table class="table table-condensed">
  <tr>
    <td>Course</td>
    <td><span t-field="doc.course_id"/></td>
    <td>Instructor</td>
    <td><span t-field="doc.instructor_id"/></td>
  </tr>
  <tr>
    <td>Start Date</td>
    <td><span t-field="doc.start_date"/></td>
    <td>Duration</td>
    <td><span t-field="doc.duration"/></td>
  </tr>
  <tr>
    <td>Taken Seats</td>
    <td><span t-field="doc.taken_seats"/></td>
    <td>Active</td>
    <td><span t-field="doc.active"/></td>
  </tr>
</table>
```

kita lihat cara menampilkan isi field record session, yaitu dengan notasi doc.<fieldname> misalnya doc.name, doc.instructor_id, dst...

Lalu kita tampilkan detail attendee dalam bentuk tabel lagi...

```
<table class="table table-condensed">
  <thead>
    <tr>
      <th>No</th>
      <th>Partner</th>
      <th>Signed</th>
    </tr>
  </thead>
  <tbody>
    <tr t-foreach="doc.attendee_ids" t-as="l">
      <td>
        <span t-field="l.name"/>
      </td>
      <td>
        <span t-field="l.partner_id" />
      </td>
      <td>
      </td>
    </tr>
  </tbody>
</table>
```

disini dilihat kita melakukan looping for each untuk setiap attendee_ids milik session yang diprint.

Lalu pada setiap record attendee_id yang disimpan pada variabel l, kita keluarkan lagi isi fieldnya dengan cara l.name dan l.partner_id...

Restart odoo dan update module ...

Hasilnya, muncul menu action Print di list view Session ketika ada yang di -select...

Daftar Session

Create

Import

Print

Action

Filters

Group By

1-2 / 2

<

>

Session


Favorites

<input checked="" type="checkbox"/>	Course	Name	Instructor	Start Date	Duration	Seats	Active	Taken Seat
<input checked="" type="checkbox"/>	Java	sesion1	Joko	02/22/2017	20	100	<input checked="" type="checkbox"/>	
<input checked="" type="checkbox"/>	Java	Session3	Mukidi	02/22/2017	10	10	<input checked="" type="checkbox"/>	

Muncul juga menu action print di setiap form view session...

Daftar Session / sesion1		Print Action	1 / 2
Edit Create		Session	Draft Confirmed

sesion1

Course	Java	Duration	20
Instructor	Joko	Seats	100
Start Date	02/22/2017	Active	<input checked="" type="checkbox"/>
Image Small		Taken Seat	3%

[Attendees](#)

Name	Partner
01	Agus
02	Badu
03	Budi

Ketika di click, maka akan dihasilkan report PDF session sesuai dengan layout yang sudah ditentukan sebelumnya...

Session (4).pdf (page 1 of 2)

Session (4).pdf

1

2

odoo

My Company Tagline

My Company

SESSION: sesion1

Course	Java	Instructor	Joko
Start Date	02/22/2017	Duration	20
Taken Seats	3.0	Active	True

No	Partner	Signed
01	Agus	
02	Badu	
03	Budi	

Selesai ... kalau nggak error 😊

5 DASHBOARD

Disini kita akan membuat dashboard yang berisi graph view yang udah kita buat sebelumnya pada bagian Advanced View, sessions calendar view dan daftar courses (dan bisa dipindahkan ke form view).

Kita akan buat file XML baru untuk dashboard di dalam folder addons. Isinya adalah:

- the board view,
- the actions referenced in that view,
- an action to open the dashboard,
- as well as a re-definition of the main menu item “academic_menu” to add the action to open the dashboard

5.1 BIKIN SUB MENU DASHBOARD

Create file `dashboard.xml`. Isinya adalah

```
<openerp>
  <data>

    <!-- tambah menu level 1 Dashboard -->
    <menuitem id="menu_dashboard"
              name="Dashboard" sequence="0"
              parent="academic_top" />

    <!-- tambah menu sub level 2 -->
    <menuitem
              name="Session Dashboard"
              parent="menu_dashboard"
              action="open_board_session"
              sequence="1"
              id="menu_board_session"
              icon="terp-graph"/>

  </data>
</openerp>
```

Gambar 36 Menu dashboard

Update file `__openerp__.py` dan tambahkan file `dashboard.xml` yang baru dibuat tadi.

Jangan lupa tambahi depends ke modul "`board`" supaya module dashboard dikenal dari modul academic.

```
{
  "name": "Academic Information System Day 5",
  "version": "1.0",
  "depends": [
    "base",
    "account",
    "sale",
    "board",
  ],
  "author": "akhmad.daniel@gmail.com",
  "category": "Education",
  'website': 'http://www.vitraining.com',
  "description": """\n
Academic Information System Day 5
-----
* Add report QWEB
* Add dashboard

""",
  "data": [
    "menu.xml",
    "course.xml",
    "session.xml",
    "attendee.xml",
    "partner.xml",
    "workflow.xml",
    "security/group.xml",
    "security/ir.model.access.csv",
    "wizard/create_attendee.xml",
    "report/session.xml",
    "dashboard.xml",
  ],
  "installable": True,
  "auto_install": False,
  "application": True,
}
```

Gambar 37 Depends board module dan panggil dashboard.xml dari `__openerp__.py`

5.2 TAMBAHI DASHBOARD XML

Edit `dashboard.xml` tambahi board sebagai berikut:

```
<record model="ir.ui.view" id="board_session_form">
  <field name="name">Session Dashboard Form</field>
  <field name="model">board.board</field>
  <field name="type">form</field>
```

```

<field name="arch" type="xml">
  <form string="Session Dashboard" version="7.0">
    <board style="2-1">
      <column>
        <action
          string="Attendees by course"
          name="% (act_session_graph)d"
          colspan="4"
          height="150"
          width="510" />
      </column>
      <column>
        <action
          string="Sessions"
          name="% (act_session_calendar)d"
          colspan="4" />
      </column>
      <column>
        <action
          string="Courses"
          name="% (act_course_list)d"
          colspan="4" />
      </column>
    </board>
  </form>
</field>
</record>

```

Gambar 38 Definisi dashboard

Tambahi definisi action yang perlu dipanggil lewat dashboard (contoh di atas adalah `act_session_calendar`, `act_session_graph`, dan `act_course_list`).

Deklarasinya harus di atas board session form.

```

<record model="ir.actions.act_window" id="act_session_calendar">
  <field name="name">academic.session.cal</field>
  <field name="res_model">academic.session</field>
  <field name="view_type">form</field>
  <field name="view_mode">calendar</field>
  <field name="view_id" ref="session_cal"/>
</record>

<record model="ir.actions.act_window" id="act_session_graph">
  <field name="name">academic.session.graph</field>
  <field name="res_model">academic.session</field>
  <field name="view_type">form</field>
  <field name="view_mode">graph</field>
  <field name="view_id" ref="session_graph"/>
</record>

<record model="ir.actions.act_window" id="act_course_list">
  <field name="name">academic.course.list</field>
  <field name="res_model">academic.course</field>
  <field name="view_type">form</field>

```

```

    <field name="view_mode">tree,form</field>
</record>

```

Gambar 39 Action window untuk content dashboard

Tambahi action window yang dipanggil oleh menuitem
(deklarasinya harus di atas menu item)

```

<record model="ir.actions.act_window" id="open_board_session">
  <field name="name">Session Dashboard</field>
  <field name="res_model">board.board</field>
  <field name="view_type">form</field>
  <field name="view_mode">form</field>
  <field name="usage">menu</field>
  <field name="view_id" ref="board_session_form"/>
</record>

```

Gambar 40 Action window dashboard

Selengkapnya file dashboard.xml seperti ini...

```

<openerp>
  <data>

    <record model="ir.actions.act_window" id="act_session_calendar">
      <field name="name">academic.session.cal</field>
      <field name="res_model">academic.session</field>
      <field name="view_type">form</field>
      <field name="view_mode">calendar</field>
      <field name="view_id" ref="session_cal"/>
    </record>

    <record model="ir.actions.act_window" id="act_session_graph">
      <field name="name">academic.session.graph</field>
      <field name="res_model">academic.session</field>
      <field name="view_type">form</field>
      <field name="view_mode">graph</field>
      <field name="view_id" ref="session_graph"/>
    </record>

    <record model="ir.actions.act_window" id="act_course_list">
      <field name="name">academic.course.list</field>
      <field name="res_model">academic.course</field>
      <field name="view_type">form</field>
      <field name="view_mode">tree,form</field>
    </record>

    <record model="ir.ui.view" id="board_session_form">
      <field name="name">Session Dashboard Form</field>
      <field name="model">board.board</field>
      <field name="type">form</field>
      <field name="arch" type="xml">
        <form string="Session Dashboard" version="7.0">
          <board style="2-1">
            <column>
              <action
                string="Attendees by course"
                name="%(act_session_graph)d"

```

```

        colspan="4"
        height="150"
        width="510" />
        <action
            string="Sessions"
            name="% (act_session_calendar)d"
            colspan="4" />
        </column>
        <column>
        <action
            string="Courses"
            name="% (act_course_list)d"
            colspan="4" />
        </column>
    </board>
</form>
</field>
</record>

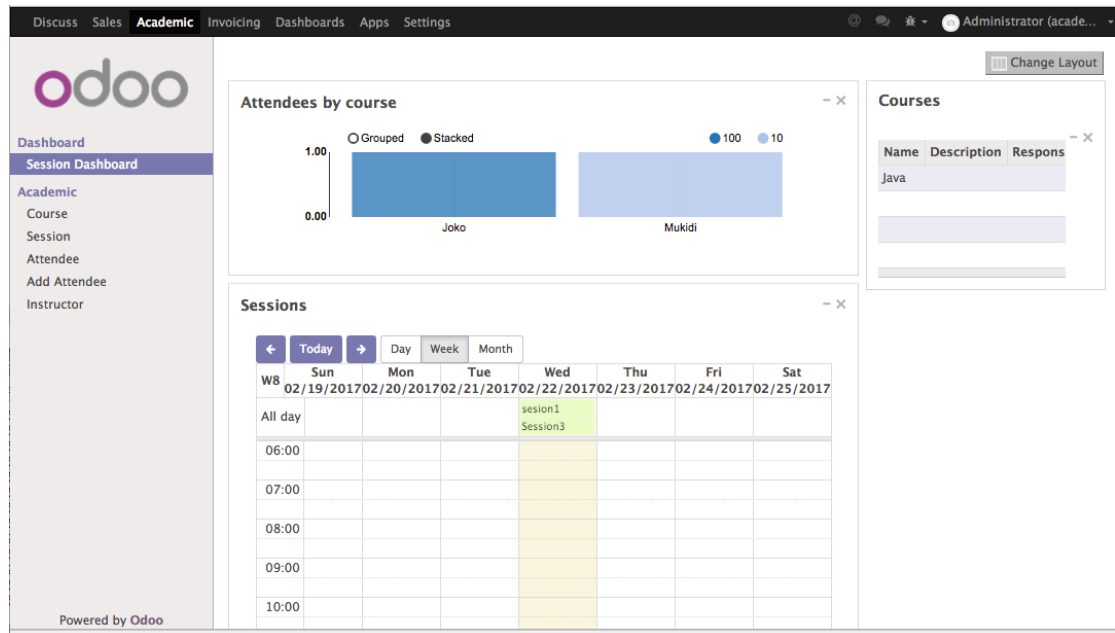
<record model="ir.actions.act_window" id="open_board_session">
    <field name="name">Session Dashboard</field>
    <field name="res_model">board.board</field>
    <field name="view_type">form</field>
    <field name="view_mode">form</field>
    <field name="usage">menu</field>
    <field name="view_id" ref="board_session_form"/>
</record>

<!-- tambah menu level 1 Dashboard -->
<menuitem id="menu_dashboard"
    name="Dashboard" sequence="0"
    parent="academic_0" />

<!-- tambah menu sub level 2 -->
<menuitem
    name="Session Dashboard"
    parent="menu_dashboard"
    action="open_board_session"
    sequence="1"
    id="menu_board_session"
    icon="terp-graph"/>
</data>
</openerp>

```

Restart odoo dan update module. Hasilnya



Gambar 41 Tampilan dashboards

6 WEB SERVICES

odoo dapat diakses melalui interface XML-RPC, dimana hampir semua bahasa pemrograman punya library-nya.

Disini kita pakai contoh kasus interfacing odoo dengan PHP.

6.1 INSTALASI XML-RPC FOR PHP

Untuk mempermudah, kita menggunakan library XML-RPC framework for PHP.

Download the XML-RPC framework for PHP dari

<http://phpxmlrpc.sourceforge.net/>

Extract file `xmlrpc-2.2.tar.gz` ke suatu folder.

6.2 AKTIFKAN PHP CURL MODULE

PHP Curl module harus aktif agar library PHP XML-RPC bisa jalan. Pastikan modulnya udah terinstal dan di-set di `php.ini`

6.3 SETUP FOLDER APLIKASI

Buat folder baru dibawah addons `academic`, kasi nama misalnya `php-xmlrpc`. Sebetulnya boleh bebas dimana aja tapi untuk memudahkan kita gabung dibawah folder addons aja.

Ambil file `xmlrpc.inc` dari directory `lib` hasil extract XML-RPC, dan tempatkan dibawah folder `php-xmlrpc`.

Lalu buat file untuk percobaan interfacing dimana kita akan menggunakan fitur-fitur XML-RPC odoo untuk login, read, create, update, dan delete data.

Buat file PHP dengan nama `test.php` dibawah folder `php-xmlrpc`.

Isinya pada awalnya sederhana aja yaitu import library `xmlrpc.lib` dan definisi class `MyodooLib` dengan beberapa atribut untuk keperluan koneksi ke odoo:

```
1 <?php
2 include("xmlrpc.inc");
3
4 class MyOpenERPLib
5 {
6     public $user      = "admin";
7     public $password  = "1";
8     public $dbname    = "devel";
9     public $server_url= "http://127.0.0.1:8069/xmlrpc/";
10    public $id         = null;
11 }
12
13
14 ?>
15
```

Gambar 42 Bikin class PHP `MyodooLib`

Variable `$user` adalah nama user yang bisa melakukan login ke odoo

Variable `$password` adalah password dari user tersebut.

Variable `$dbname` adalah nama database yang akan di connect.

Variable `$server_url` adalah alamat URL server odoo.

Variable `$id` adalah integer user id yang berhasil login.

Struktur folder addons kita sejauh ini...

```
academic
|-- __init__.py
|-- __openerp__.py
|-- attendee.py
|-- attendee.xml
|-- course.py
|-- course.xml
|-- dashboard.xml
|-- menu.xml
|-- partner.py
|-- partner.xml
|-- php-xmlrpc
|   |-- test.php
|   `-- xmlrpc.inc
|-- security
```

```

| |-- group.xml
| |-- ir.model.access.csv
|-- session.py
|-- session.xml
|-- wizard
| |-- __init__.py
| |-- create_attendee.py
|-- workflow.xml

```

6.4 LOGIN

Lanjut, kita buat method `login()` di dalam class `MyodooLib`.

Deklarasi method nya sebagai berikut.

```

11
12     function login(){
13         $conn      = new xmlrpc_client($this->server_url . 'common');
14         $msg        = new xmlrpcmsg('login');
15         $msg->addParam( new xmlrpcval($this->dbname, "string") );
16         $msg->addParam( new xmlrpcval($this->user, "string") );
17         $msg->addParam( new xmlrpcval($this->password, "string") );
18         $resp       = $conn->send( $msg );
19         $val         = $resp->value();
20         $this->id     = $val->scalarval();
21         return $this->id>0 ? $this->id : -1 ;
22     }
23

```

Gambar 43 Method `login()`

Disini kita lakukan request login melalui XML-RPC ke odoo.

Prosesnya adalah dengan membentuk dulu object `$msg` yang merupakan class XML RPC message, `xmlrpcmsg('login')`. Lalu kita tambah parameter `$msg` dengan `$this->dbname`, `$this->user`, dan `$this->password`, yang terlebih dulu dibentuk menjadi object XML RPC value, `xmlrpcval`, type-nya string.

Setelah XML RPC message jadi, kita kirimkan melalui `$conn` yang adalah object XML RPC Client, `xmlrpc_client`, yang diarahkan kepada URL `$this->server_url . 'common'`.

Respon yang didapat dari server ditangkap di variable `$resp` yang masih berbentuk object. Untuk mendapatkan nilai

scalarnya, perlu di convert menggunakan method `value()` dan `scalarval()`.

Hasil dari method login adalah integer berupa user id yang berhasil login atau integer -1 jika gagal login.

Test login dengan script ini:

```
193
194 $openerp = new MyOpenERPLib;
195 $userId = $openerp->login();
196 if ($userId == -1) {
197     die("Login error ....");
198 }
199
200
```

Gambar 44 Test method login()

Jika login berhasil maka variable `$this->id` akan terisi dengan user id yang berhasil login dan dapat digunakan untuk keperluan method selanjutnya.

Jika gagal login, maka `$this->id` isinya -1, dan langsung exit dengan pesan "Login error....".

6.5 SEARCH

Bikin lagi method `search()` pada class `MyodooLib`. Deklarasinya seperti ini...

```

24
25 function search($relation, $key){
26     $key_array = array(
27         new xmlrpcval(
28             array(
29                 new xmlrpcval("name","string"),
30                 new xmlrpcval("ilike","string"),
31                 new xmlrpcval($key, "string"),
32             ),
33             "array"
34         )
35     );
36
37     $msg = new xmlrpcmsg('execute');
38     $msg->addParam(new xmlrpcval($this->dbname, "string"));
39     $msg->addParam(new xmlrpcval($this->id, "int"));
40     $msg->addParam(new xmlrpcval($this->password, "string"));
41     $msg->addParam(new xmlrpcval($relation, "string"));
42     $msg->addParam(new xmlrpcval("search", "string"));
43     $msg->addParam(new xmlrpcval($key_array, "array"));
44     $conn = new xmlrpc_client($this->server_url . 'object');
45     $conn->return_type = 'phpvals';
46     $resp = $conn->send($msg);
47     if ($resp->faultCode())
48     {
49         var_dump($resp);
50         return -2;
51     }
52     else {
53         if($val = $resp->value()){
54             return $val;
55         }
56         else
57         {
58             return -1;
59         }
60     }
61 }

```

Gambar 45 Method search()

Disini kita lakukan request search data dengan membentuk dulu object `$msg` yang merupakan class XML RPC message, `xmlrpcmsg('execute')`. Maksudnya adalah kita akan men-execute sebuah function pada suatu object.

Lalu kita tambah parameter `$msg` dengan `$this->dbname`, `$this->user`, dan `$this->password`, yang terlebih dulu dibentuk menjadi object XML RPC value, `xmlrpcval`, type-nya string.

Parameter selanjutnya adalah nama relasi object yang akan di-search. Dalam hal ini kita jadikan input parameter `$relation`.

Parameter berikutnya adalah nama function yang akan di-execute yaitu "search".

Parameter berikutnya adalah array XML-RPC yang berisi kriteria pencarian, disini kita set pada variable \$key_array.

Pembentukan variable \$key_array dilakukan sebelumnya, yaitu merupakan array satu elemen berisi object xmlrpcval berjenis "array" dengan nilai array berisi "name", "ilike", \$key yang dicari dengan sebelumnya masing-masing dibentuk menjadi object xmlrpcval berjenis string. Berikut ini cuplikannya.

```
$key_array = array(
    new xmlrpcval(
        array(
            new xmlrpcval("name","string"),
            new xmlrpcval("ilike","string"),
            new xmlrpcval($key, "string"),
        ),
        "array"
    )
);
```

Test method search dengan menjalankan script berikut :

```
193
194 $openerp = new MyOpenERPLib;
195 $userId = $openerp->login();
196 if ($userId == -1) {
197     die("Login error ....");
198 }
199
200
201
202 echo "=====\Searching data containing java.....\n";
203 $session_ids = $openerp->search("academic.session","java");
204 var_dump($session_ids);
205
```

Gambar 46 Test method search()

Pada test di atas kita menjalankan method search() untuk mencari session yang memiliki nama ada "java"nya. Hasilnya adalah berupa array list of id dari session yang matching dengan pencarian.

6.6 READ

Lanjut, bikin method search() pada class MyodooLib.

Deklarasinya seperti ini...

```
68 function read($relation , $ids, $fields){
69
70     $id_val=array();
71     foreach ($ids as $i) {
72         $id_val[] = new xmlrpcval($i , "int");
73     }
74
75     $fields_val=array();
76     foreach ($fields as $f) {
77         $fields_val[] = new xmlrpcval($f, "string");
78     }
79
80     $msg = new xmlrpcmsg('execute');
81     $msg->addParam(new xmlrpcval($this->dbname, "string"));
82     $msg->addParam(new xmlrpcval($this->id, "int"));
83     $msg->addParam(new xmlrpcval($this->password, "string"));
84     $msg->addParam(new xmlrpcval($relation, "string"));
85     $msg->addParam(new xmlrpcval("read", "string"));
86     $msg->addParam(new xmlrpcval($id_val, "array"));
87     $msg->addParam(new xmlrpcval($fields_val, "array"));
88
89     $conn = new xmlrpc_client($this->server_url . 'object');
90     $conn->return_type = 'phpvals';
91
92     $resp = $conn->send($msg);
93
94     if ($resp->faultCode()){
95         var_dump($resp);
96         return -1;
97     }
98     else
99         return ( $resp->value() );
100
101 }
```

Gambar 47 Method read()

Disini kita lakukan request read data dengan membentuk dulu object `$msg` yang merupakan class XML RPC message, `xmlrpcmsg('execute')`. Maksudnya adalah kita akan men-execute sebuah function pada suatu object.

Lalu kita tambahi parameter `$msg` dengan `$this->dbname`, `$this->user`, dan `$this->password`, yang terlebih dulu dibentuk menjadi object XML RPC value, `xmlrpcval`, type-nya string.

Parameter selanjutnya adalah nama relasi object yang akan di-search. Dalam hal ini kita jadikan input parameter `$relation`.

Parameter berikutnya adalah nama function yang akan di-execute yaitu "read".

Parameter berikutnya adalah id record yang akan dicari dalam bentuk XML-RPC array. Disini kita udah set pada variable \$id_val.

Parameter berikutnya adalah field-field dari record yang akan dicari dalam bentuk XML-RPC array. Disini kita udah set pada variable \$fields_val.

Variabel \$id_vals harus dibentuk terlebih dahulu agar berbentuk array XML-RPC berdasarkan array PHP pada input parameter \$ids.

Berikut ini cara membentuknya.

```
$id_val=array();  
foreach ($ids as $i) {  
    $id_val[] = new xmlrpcval($i , "int");  
}
```

Disini kita hanya melakukan looping terhadap array PHP \$ids, dan pada setiap looping kita kumpulkan didalam array \$id_vals yang setiap elemennya adalah object xmlrpcval dari nilai integer setiap \$ids.

Variabel \$fields_vals juga harus dibentuk terlebih dahulu agar berbentuk array XML-RPC berdasarkan array PHP pada input parameter \$fields.

Berikut ini cara membentuknya.

```
$fields_val=array();  
foreach ($fields as $f) {  
    $fields_val[] = new xmlrpcval($f, "string");  
}
```

Disini kita juga melakukan looping terhadap array PHP \$fields, dan pada setiap looping kita kumpulkan didalam array

\$fields_vals yang setiap elemennya adalah object xmlrpcval bertipe "string" dari nilai setiap \$fields.

Test method read dengan menjalankan script lanjutan method search() seperti berikut :

```
193
194 $openerp = new MyOpenERPLib;
195 $userId = $openerp->login();
196 if ($userId == -1) {
197     die("Login error ....");
198 }
199
200 echo "=====\Searching data containing java.....\n";
201 $session_ids = $openerp->search("academic.session","java");
202 var_dump($session_ids);
203
204 echo "=====\Reading data containing java.....\n";
205 $fields      = array("name","duration");
206 $records     = $openerp->read("academic.session" , $session_ids, $fields);
207 foreach ($records as $key => $value) {
208     echo $value["name"];
209     echo "\n";
210 }
211
```

Gambar 48 Test method read()

Disini kita jalankan method read() dengan mengambil nilai array \$session_ids dari hasil pencarian sebelumnya melalui method search().

Return value dari method read() adalah array dari record yang berhasil dibaca sesuai id yang dimasukkan. Array ini memiliki key sesuai dengan nilai \$fields yang dikeluarkan pada saat read.

6.7 CREATE

Lanjut, bikin method create() pada class MyodooLib.
Deklarasinya seperti ini...


```

105
106 function create($relation, $values){
107
108     $msg = new xmlrpcmsg('execute');
109     $msg->addParam(new xmlrpcval($this->dbname, "string"));
110     $msg->addParam(new xmlrpcval($this->id, "int"));
111     $msg->addParam(new xmlrpcval($this->password, "string"));
112     $msg->addParam(new xmlrpcval($relation, "string"));
113     $msg->addParam(new xmlrpcval("create", "string"));
114     $msg->addParam(new xmlrpcval($values, "struct"));
115
116     $conn = new xmlrpc_client($this->server_url . 'object');
117     $conn->return_type = 'phpvals';
118     $resp = $conn->send($msg);
119
120     if ($resp->faultCode()){
121         var_dump($resp);
122         return -1;
123     }
124     else{
125         return $resp->value();
126     }
127 }
128

```

Gambar 49 Method create()

Disini kita lakukan request create data dengan membentuk dulu object `$msg` yang merupakan class XML RPC message, `xmlrpcmsg('execute')`. Maksudnya adalah kita akan men-execute sebuah function pada suatu object.

Lalu kita tambahi parameter `$msg` dengan `$this->dbname`, `$this->id`, dan `$this->password`, yang terlebih dulu dibentuk menjadi object XML RPC value, `xmlrpcval`, type-nya string.

Parameter selanjutnya adalah nama relasi object yang akan di-create recordnya. Dalam hal ini kita jadikan input parameter `$relation`.

Parameter berikutnya adalah nama function yang akan di-execute yaitu "create".

Parameter berikutnya adalah array XML-RPC yang berisi nilai field-field yang akan di-insert yang diambil langsung dari input parameter variable `$values`. Variabel ini harus berupa array ketika method dijalankan.

Script untuk test method create.

```
194 $openerp = new MyOpenERPLib;
195 $userId = $openerp->login();
196 if ($userId == -1) {
197     die("Login error ....");
198 }
199
200 echo "=====\\nInserting data.....\\n";
201 $values = array(
202     'name' => new xmlrpcval("Session test dari XML","string"),
203     'start_date' => new xmlrpcval("2014-09-01","string")
204 );
205 $ret = $openerp->create("academic.session", $values);
206 echo $ret ;
207 echo "\\n";
208
```

Gambar 50 Test method create()

Disini kita bentuk dulu array input \$values yang berupa array dengan elemen object-object xmlrpcval. Key dari array adalah nama field dan values nya berupa xmlrpcval dari data yang akan diinsert, ber-type sesuai dengan type field masing-masing.

Array \$values kemudian dijadikan input parameter untuk method create(). Jika berhasil maka return value nya adalah id dari record yang berhasil di-create.

6.8 DELETE

Lanjut, bikin method delete() pada class MyodooLib.

Deklarasinya seperti ini...

```

130
131 function delete($relation, $ids) {
132     $id_val=array();
133     foreach ($ids as $i) {
134         $id_val[] = new xmlrpcval($i , "int");
135     }
136
137     $msg = new xmlrpcmsg('execute');
138     $msg->addParam(new xmlrpcval($this->dbname, "string"));
139     $msg->addParam(new xmlrpcval($this->id, "int"));
140     $msg->addParam(new xmlrpcval($this->password, "string"));
141     $msg->addParam(new xmlrpcval($relation, "string"));
142     $msg->addParam(new xmlrpcval("unlink", "string"));
143     $msg->addParam(new xmlrpcval($id_val, "array"));
144
145     $conn = new xmlrpc_client($this->server_url . 'object');
146     $conn->return_type = 'phpvals';
147
148     $resp = $conn->send($msg);
149
150     if ($resp->faultCode()){
151         var_dump($resp);
152         return -1;
153     }
154     else
155         return ( $resp->value() );
156 }
157

```

Gambar 51 Method delete()

Disini kita lakukan request create data dengan membentuk dulu object `$msg` yang merupakan class XML RPC message, `xmlrpcmsg('execute')`. Maksudnya adalah kita akan men-execute sebuah function pada suatu object.

Lalu kita tambahi parameter `$msg` dengan `$this->dbname`, `$this->id`, dan `$this->password`, yang terlebih dulu dibentuk menjadi object XML RPC value, `xmlrpcval`, type-nya string.

Parameter selanjutnya adalah nama relasi object yang akan di-delete recordnya. Dalam hal ini kita jadikan input parameter `$relation`.

Parameter berikutnya adalah nama function yang akan di-execute yaitu "unlink".

Parameter berikutnya adalah id record yang akan dicari dalam bentuk XML-RPC array. Disini kita udah set pada variable `$id_val`.

Pembentukan \$id_val sama seperti pada method read().

Script untuk test method delete.

```
193
194 $openerp = new MyOpenERPLib;
195 $userId = $openerp->login();
196 if ($userId == -1) {
197     die("Login error ....");
198 }
199
200
201 echo "=====\nDeleting data id=1.....\n";
202 $ids = array(1);
203 $ret = $openerp->delete("academic.session", $ids );
204 echo $ret ;
205 echo "\n";
206
```

Gambar 52 Test method delete()

Disini kita bentuk dulu array \$ids yang berisi id yang akan di-delete, contohnya 1. Array ini dijadikan input parameter bagi method delete().

Return value method ini berupa id yang berhasil di delete.

6.9 WRITE

Lanjut, bikin method write() pada class MyodooLib. Deklarasinya seperti ini...

```

159 function write($relation, $ids, $values ){
160
161     global $conn, $dbname, $id, $password, $server_url;
162
163     $id_val=array();
164     foreach ($ids as $i) {
165         $id_val[] = new xmlrpcval($i , "int");
166     }
167
168     $msg = new xmlrpcmsg('execute');
169     $msg->addParam(new xmlrpcval($dbname, "string"));
170     $msg->addParam(new xmlrpcval($id, "int"));
171     $msg->addParam(new xmlrpcval($password, "string"));
172     $msg->addParam(new xmlrpcval($relation, "string"));
173     $msg->addParam(new xmlrpcval("write", "string"));
174     $msg->addParam(new xmlrpcval($id_val, "array"));
175     $msg->addParam(new xmlrpcval($values, "struct"));
176
177     $conn = new xmlrpc_client($server_url . 'object');
178     $conn->return_type = 'phpvals';
179
180     $resp = $conn->send($msg);
181
182     if ($resp->faultCode()){
183         var_dump($resp);
184         return -1;
185     }
186     else
187         return ( $resp->value() );
188
189 }
190

```

Gambar 53 Method write()

Disini kita lakukan request write data dengan membentuk dulu object `$msg` yang merupakan class XML RPC message, `xmlrpcmsg('execute')`. Maksudnya adalah kita akan men-execute sebuah function pada suatu object.

Lalu kita tambahi parameter `$msg` dengan `$this->dbname`, `$this->id`, dan `$this->password`, yang terlebih dulu dibentuk menjadi object XML RPC value, `xmlrpcval`, type-nya string.

Parameter selanjutnya adalah nama relasi object yang akan di-delete recordnya. Dalam hal ini kita jadikan input parameter `$relation`.

Parameter berikutnya adalah nama function yang akan di-execute yaitu "write".

Parameter berikutnya adalah id record yang akan dicari dalam bentuk XML-RPC array. Disini kita udah set pada variable \$id_val.

Pembentukan \$id_val sama seperti pada method read().

Parameter berikutnya adalah array XML-RPC yang berisi nilai field-field yang akan di-insert yang diambil langsung dari input parameter variable \$values. Variabel ini harus berupa array ketika method dijalankan.

Script test method write():

```
193
194 $openerp = new MyOpenERPLib;
195 $userId = $openerp->login();
196 if ($userId == -1) {
197     die("Login error ....");
198 }
199
200 echo "=====Updating data id=2.....\n";
201 $ids = array(2);
202 $values = array(
203     'name' => new xmlrpcval("Session update dari XML","string"),
204     'start_date' => new xmlrpcval("2014-09-01","string")
205 );
206 $ret = $openerp->write("academic.session", $ids, $values);
207 echo $ret ;
208 echo "\n";
209
```

Gambar 54 Test method write()

Disini kita bentuk dulu array \$ids yang berisi id yang akan di-delete, contohnya 2. Array ini dijadikan input parameter bagi method write().

Lalu kita bentuk array input \$values yang berupa array dengan elemen object-object xmlrpcval. Key dari array adalah nama field dan values nya berupa xmlrpcval dari data yang akan di-update, ber-type sesuai dengan type field masing-masing.

Return value method adalah id record yang berhasil di-update.

6.10 WRITE ONE2MANY FIELDS

Bagaimana cara men-update atau meng-insert data berikut relasi one2many nya sekaligus ? jadi misanya Session dan Attendee, gimana cara menginput Attendee sekaligus barengan dengan Session, tanpa harus insert Attendee satu-per-satu.

Berikut ini scriptnya.

```
200
201 echo "=====\\Create one2many data .....\\n";
202 $partners = array(
203     array("id"=>7, "name" => "nomor1"),
204     array("id"=>168, "name" => "nomor3"),
205     array("id"=>172, "name" => "joko"),
206 );
207 $att_lines = array();
208 foreach($partners as $p) {
209     $att_lines[] = new xmlrpcval(
210         array(
211             new xmlrpcval(0,'int'),
212             new xmlrpcval(0,'int'),
213             new xmlrpcval(
214                 array(
215                     'partner_id'=> new xmlrpcval($p["id"], 'string'),
216                     'name' => new xmlrpcval($p["name"], 'string'),
217                 ),
218                 "struct"
219             )
220         ),
221         "array"
222     ); // one record
223 }
224
225 $attendee_ids = new xmlrpcval(
226     $att_lines,
227     "array"
228 );
229
230 $values = array(
231     'name' => new xmlrpcval("Session Create o2m dari XML","string"),
232     'start_date' => new xmlrpcval("2014-09-02","string"),
233     'attendee_ids' => $attendee_ids
234 );
235
236 $ret = $openerp->create("academic.session", $values);
237 echo $ret ;
```

Gambar 55 Write field one2many

Pada intinya kita menggunakan method `write()` yang sama dengan sebelumnya.

Tapi disini kita update field `attendee_ids` yang merupakan field one2many pada Session object.

Untuk meng-insert atau update field jenis ini diperlukan nilai one2many commands, tapi dalam bentuk XML-RPC.

Kalau di Python kita cuman set data nya seperti ini:

```
[ (0,0,{data1}) , (0,0,{data2}) , ... ]
```

Di PHP XML-RPC data itu harus dibentuk dengan logika seperti ini:

Misalnya kita punya data partner array seperti ini ingin dimasukkan sebagai Attendee suatu Session.

```
$partners = array(
    array("id"=>7, "name" => "nomor1"),
    array("id"=>168, "name" => "nomor3"),
    array("id"=>172, "name" => "joko"),
);
```

Array partner itu harus di-looping dalam rangka membentuk array `$attendee_ids` dalam format one2many command.

Pertama kita kumpulin di variable array `$att_lines` object `xmlrpcval` berjenis array yang isinya adalah array dari 0,0, dan data. Data adalah object `xmlrpcval` berjenis "struct" yang terdiri dari array dengan key 'partner_id' dan 'name' dan value-nya adalah id partner dan nama partner dalam bentuk `xmlrpcval` berjenis string.

```
$att_lines = array();
foreach($partners as $p) {
    $att_lines[] = new xmlrpcval(
        array(
            new xmlrpcval(0,'int'),
            new xmlrpcval(0,'int'),
            new xmlrpcval(
                array(
                    'partner_id' => new
xmlrpcval($p["id"], 'string'),
                    'name' => new
xmlrpcval($p["name"], 'string'),
                ),
            ),
    );
```



```

        "struct"
    ),
    "array"
); // one record
}

```

Setelah terkumpul sesuai dengan banyaknya array partner, variable `$att_lines` dibentuk lagi menjadi variable `$attendee_ids` yang merupakan object `xmlrpcval` ber-type array.

```

$attendee_ids = new xmlrpcval(
    $att_lines,
    "array"
);

```

Baru kemudian variable `$attendee_ids` dimasukkan sebagai nilai dari field `attendee_ids` object session. Semua field dijadikan array `$values` yang siap dikirim ke method `create()` atau `update()`.

```

$values = array(
    'name' => new xmlrpcval("Session Create o2m dari XML", "string"),
    'start_date' => new xmlrpcval("2014-09-02", "string"),
    'attendee_ids' => $attendee_ids
);

$ret = $odoo->create("academic.session", $values);

```

7 WEB SERVICE DENGAN PHP RIPCORD

7.1 PERSIAPAN

Buat folder di **htdocs/odoo-client**.

Download Ricord library , simpan di **odoo-client/ripcord**.

Didalamnya ada file **ripcord.php**.

Ripcord Library URL: <https://github.com/poef/ripcord>

Dokumentasi API web service Odoo:

https://www.odoo.com/documentation/10.0/api_integration.html

7.2 INCLUDE LIBRARY RIPCORD DI SCRIPT PHP

Buat file **test.php** dan include library ripcord. Caranya:

```
<?php
include ("ripcord/ripcord.php");

$url    = "http://localhost:8069";
$username = "admin";
$password = "1";
$dbname = "academic";
```

7.3 PROSES LOGIN

Login dari PHP ke Odoo dengan user dan password Odoo user.

Sebelumnya kita harus membentuk object \$common untuk object client XMLRPC pada alamat /xmlrpc/2/common untuk proses login ke Odoo.

Lalu jalankan method authenticate untuk login user.

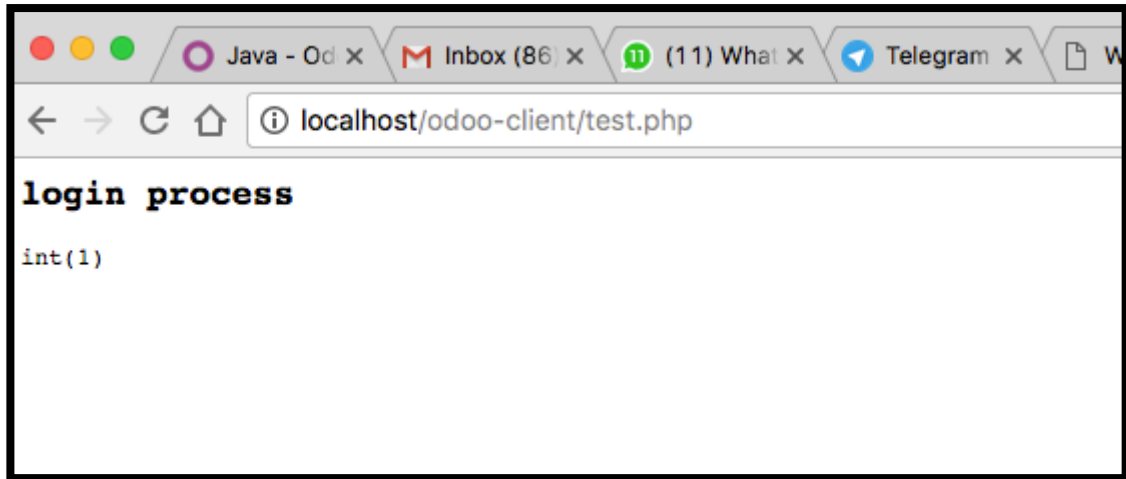
Caranya:

```
echo "<h2>login process</h2>";

$common = ripcord::client("$url/xmlrpc/2/common");
$uid = $common->authenticate($dbname, $username, $password, array());
var_dump($uid);
```

Testing dari browser: <http://localhost/odoo-client/test.php>

Variabel \$uid: integer ID user jika berhasil login atau False jika gagal login.



Gambar 56 Proses login

7.4 PROSES SEARCH DAN READ DATA

Mencari data Course berdasarkan nama. Cari dulu ID record dengan method **search** lalu jalankan method **read**.

Sebelumnya kita harus membentuk object \$model untuk object client XMLRPC pada alamat **/xmlrpc/2/object**.

```
/* create $models object*/
$models = ripcord::client("$url/xmlrpc/2/object");

echo "<h2>search and then read academic.course</h2>";

/* search ID partner */
$ids = $models->execute_kw($dbname, $uid, $password,
    'academic.course', 'search',
    array(
        array(
            array('name', '=', 'Java'),
        )
    )
);
var_dump($ids);

/* read records by ID */
$records = $models->execute_kw($dbname, $uid, $password,
```

```

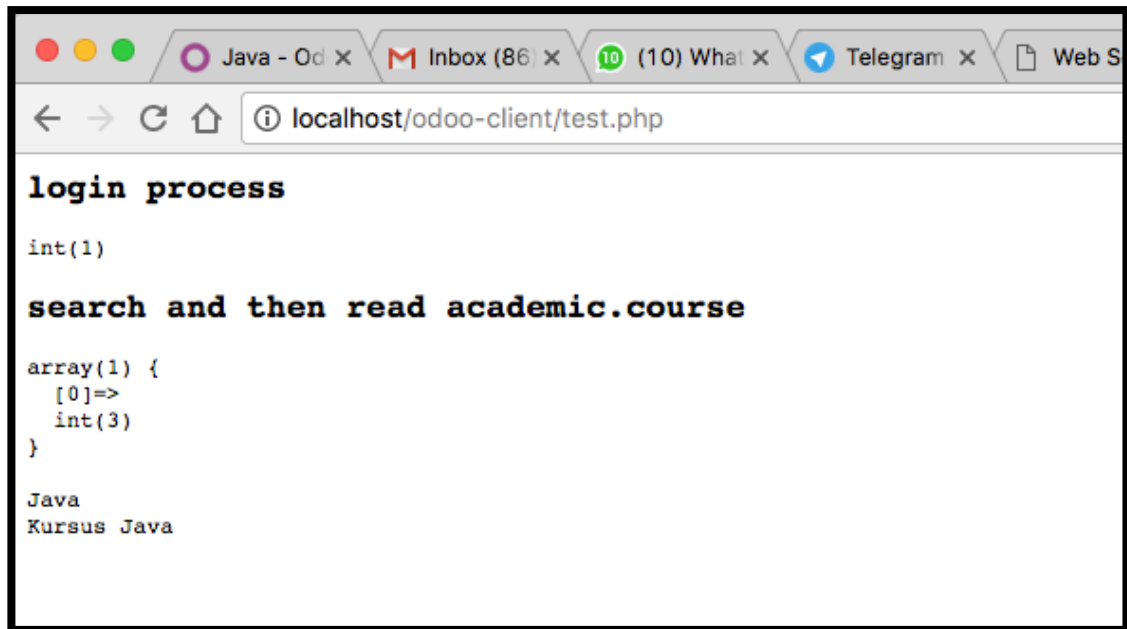
        'academic.course', 'read', array( $ids ));

echo "<br/>";

foreach ($records as $key => $value) {
    echo $value['name'] . "<br/>";
    echo $value['description'] . "<br/>";
}

```

Hasilnya:



Atau bisa menggunakan method **search_read** langsung:

```

/* search and read */

echo "<h2>search_read academic.course</h2>";

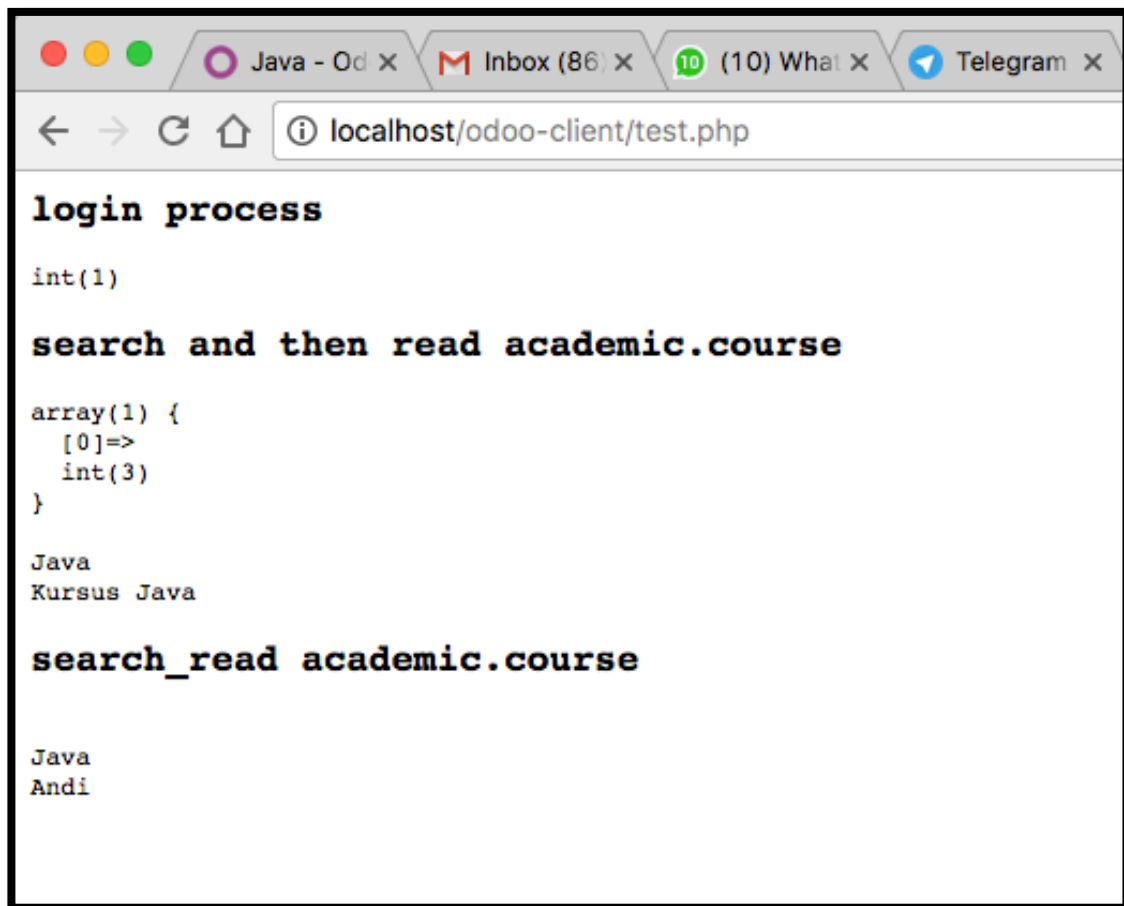
$records = $models->execute_kw($dbname, $uid, $password,
    'academic.course', 'search_read',
    array(
        array(
            array('name','ilike','java'),
        )
    )
);

echo "<br/>";

foreach ($records as $key => $value) {
    echo $value['name'] . "<br/>";
    echo $value['responsible_id'][1] . "<br/>";
}

```

Hasilnya



Disini kita menampilkan field many2one **responsible_id** dengan cara :

```
$value['responsible_id'][1]
```

karena field many2one berupa array yang terdiri dari 2 elemen yaitu ID dan name record terkait.

7.5 PROSES CREATE

Create data session pada course yang dipilih diatas. Buat file PHP baru: **create.php**.

Isinya

```

<pre>
<?php
include "ripcord/ripcord.php";

$url = "http://localhost:8069";
$username = "admin";
$password = "1";
$dbname = "academic";

/* login */
echo "<h2>login process</h2>";

$common = ripcord::client("$url/xmlrpc/2/common");
$suid = $common->authenticate($dbname, $username, $password, array());
var_dump($suid);

/* create $models object*/
$models = ripcord::client("$url/xmlrpc/2/object");

echo "<h2>create course</h2>";

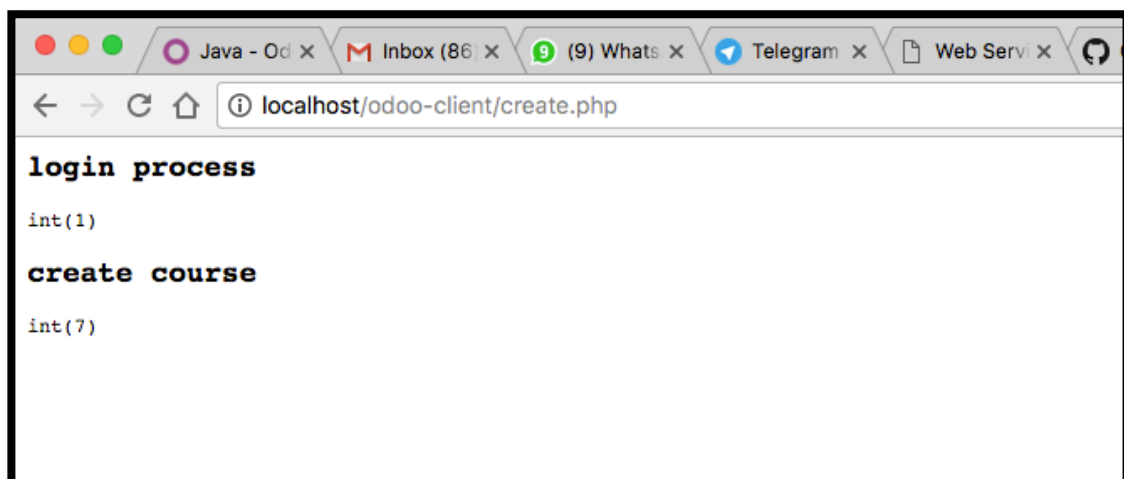
$id = $models->execute_kw($dbname, $suid, $password,
    'academic.course', 'create',
    array(
        array(
            'name'=>"PHP XMLRPC",
            'description'=>"Integrasi PHP dengan Odoo",
        ),
    ),
);
var_dump($id);

```

Pada proses ini kita jalankan method create pada object yang akan dicreate recordnya, misalnya academic.course.

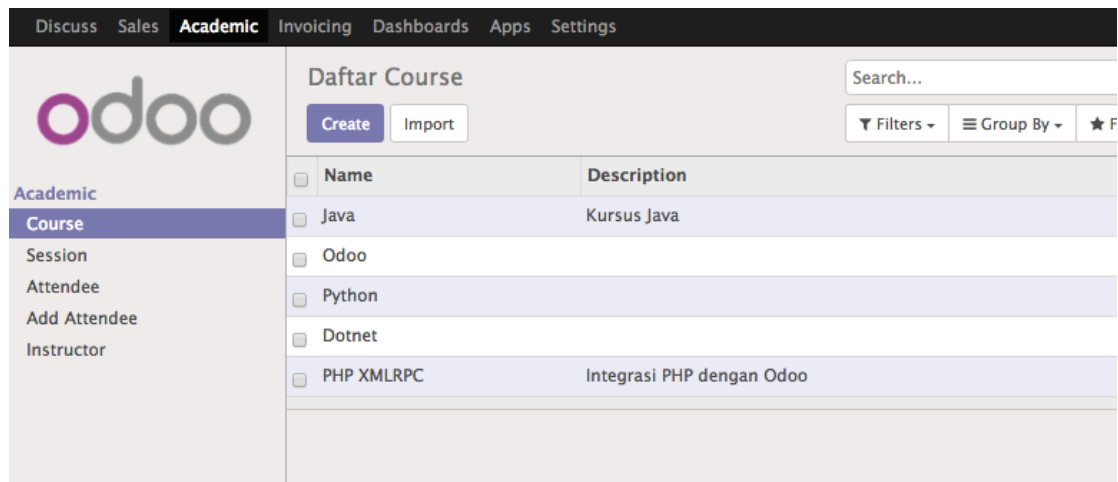
Lalu pada parameter method tersebut, kita sertakan nama field dan value-nya menggunakan array associative.

Hasilnya:



Gambar 57 Create course

pada Odoo akan terbentuk record Course yang dibuat melalui PHP.



Gambar 58 Hasil create di Odoo

7.6 PROSES WRITE

Update data session yang sudah dicreate sebelumnya. Buat file PHP baru dengan nama: **write.php**.

Isinya:

```
<pre>
<?php
include "ripcord/ripcord.php";

$url    = "http://localhost:8069";
$username = "admin";
$password = "1";
$dbname = "academic";

/* login */
echo "<h2>login process</h2>";

$common = ripcord::client("$url/xmlrpc/2/common");
$uid = $common->authenticate($dbname, $username, $password, array());
var_dump($uid);

/* create $models object*/
$models = ripcord::client("$url/xmlrpc/2/object");
```

```

echo "<h2>search and then write academic.course</h2>";

/* search ID course */
$id = $models->execute_kw($dbname, $uid, $password,
    'academic.course', 'search',
    array(
        array(
            array('name','=', 'Java'),
        )
    )
);
var_dump($id);

echo "<h2>write course with id=$id[0] </h2>";

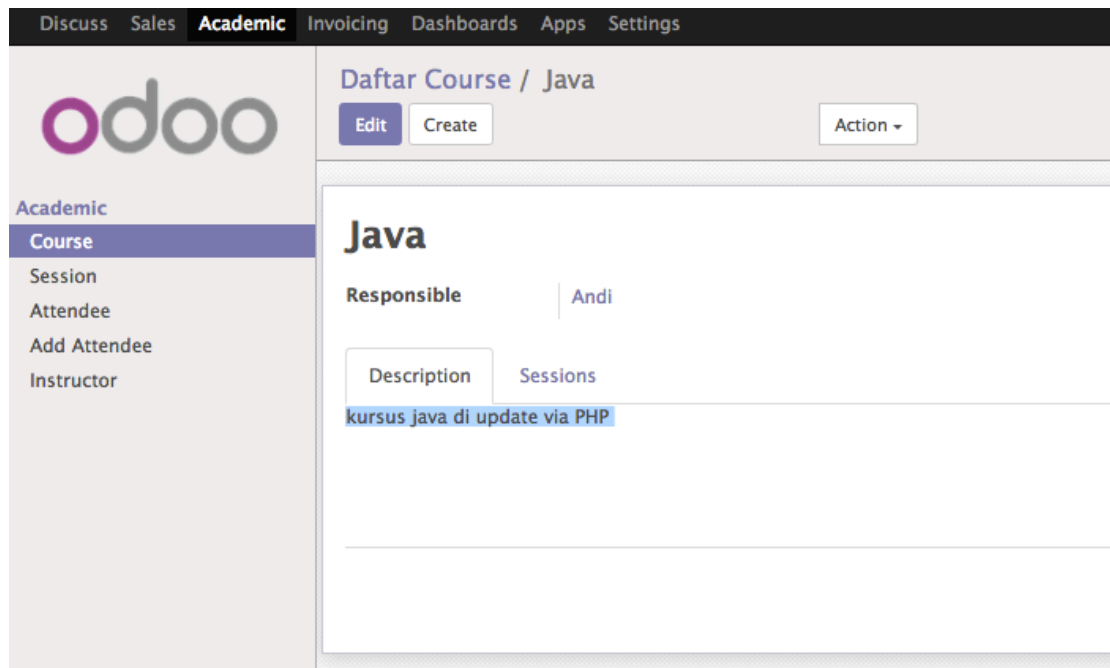
$ret = $models->execute_kw($dbname, $uid, $password,
    'academic.course', 'write',
    array(
        $id,
        array('description'=>"kursus java di update via PHP")
    )
);
var_dump($ret);

```

Hasilnya:



di Odoo, record tersebut akan terupdate:



7.7 PROSES DELETE

Delete data session berdasarkan nama. Buat file PHP baru dengan nama: **delete.php**.

Isinya:

```
<pre>
<?php
include "ripcord/ripcord.php";

$url    = "http://localhost:8069";
$username = "admin";
$password = "1";
$dbname = "academic";

/* login */
echo "<h2>login process</h2>";

$common = ripcord::client("$url/xmlrpc/2/common");
$suid = $common->authenticate($dbname, $username, $password, array());
var_dump($suid);

/* create $models object*/
$models = ripcord::client("$url/xmlrpc/2/object");

echo "<h2>search and then delete academic.course</h2>";

/* search ID course */
$ids = $models->execute_kw($dbname, $suid, $password,
    'academic.course', 'search',
```

```

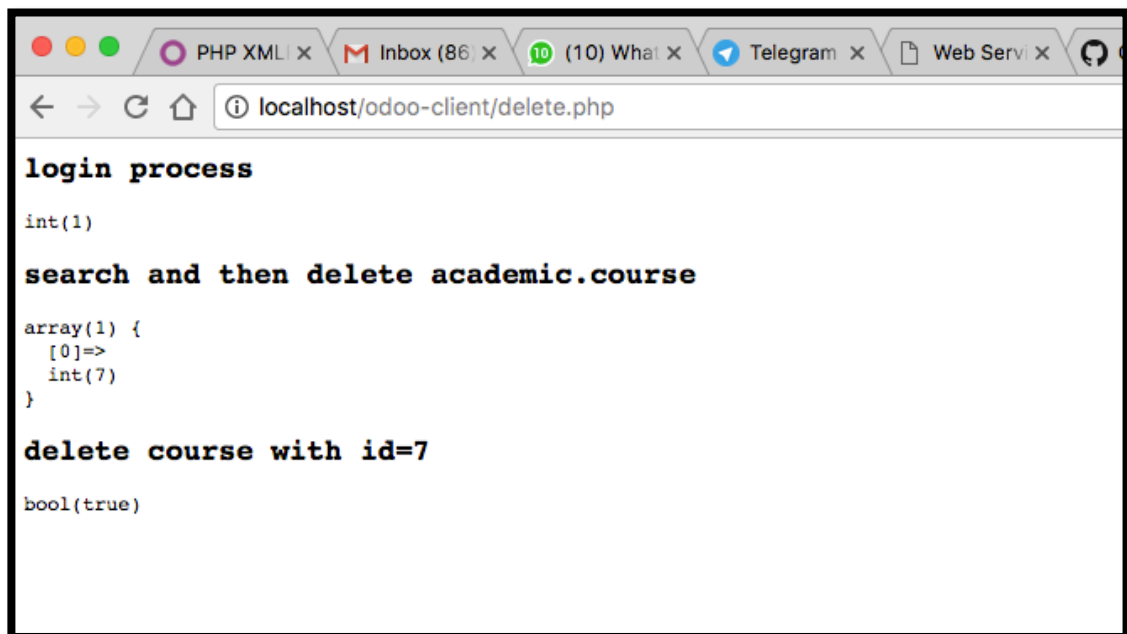
        array(
            array(
                array('name','=','PHP XMLRPC'),
            )
        );
var_dump($ids);

echo "<h2>delete course with id=$ids[0] </h2>";

$ret = $models->execute_kw($dbname, $uid, $password,
    'academic.course', 'unlink',
    array(
        $ids,
    )
);
var_dump($ret);

```

Jalankan dan lihat hasilnya:



Record tersebut akan hilang di Odoo.



7.8 PROSES CREATE ATAU WRITE FIELD ONE2MANY

Contoh mengupdate data attendee suatu session sekaligus dengan 1 command write ke session.

Buat file baru dengan nama update-attendee.php.

Isinya.

```
<pre>
<?php
include "ripcord/ripcord.php";

$url    = "http://localhost:8069";
$username = "admin";
$password = "1";
$dbname = "academic";

/* login */
echo "<h2>login process</h2>";

$common = ripcord::client("$url/xmlrpc/2/common");
$uid = $common->authenticate($dbname, $username, $password, array());
var_dump($uid);

/* create $models object*/
$models = ripcord::client("$url/xmlrpc/2/object");

/* search ID partner Agus */
$partner1_id = $models->execute_kw($dbname, $uid, $password,
    'res.partner', 'search',
    array(
        array(
            array('name', '=', 'Agus'),
```

```

    )
  )
);

/* search ID partner Budi */
$partner2_id = $models->execute_kw($dbname, $uid, $password,
    'res.partner', 'search',
    array(
        array(
            array('name','=', 'Budi'),
        )
    )
);

/* search ID session */
$ids = $models->execute_kw($dbname, $uid, $password,
    'academic.session', 'search',
    array(
        array(
            array('name','=', 'Session 1 Java'),
        )
    )
);
var_dump($ids);

echo "<h2>write attendee_ids on academic.session</h2>";

$ret = $models->execute_kw($dbname, $uid, $password,
    'academic.session', 'write',
    array(
        $ids,
        array(
            'attendee_ids' => array(
                array(0,0,array(
                    'partner_id'=> $partner1_id[0],
                    'name'=>"01")),
                array(0,0,array(
                    'partner_id'=> $partner2_id[0],
                    'name'=>"02"))
            )
        )
    )
);
var_dump($ret);

```

Untuk proses create, field attendee_ids persis sama formatnya.

Jalankan dan lihat hasilnya.

```

login process

int(1)
array(1) {
    [0]=>
        int(4)
}

write attendee_ids on academic.session

bool(true)

```

Di Odoo session tersebut akan terisi attendee-nya.

Daftar Session / Session 1 Java

Edit
Create
Print
Action

4 / 4
<
>

Draft
Confirmed
Done

Session 1 Java

Course	Java	Duration	0
Instructor		Seats	0
Start Date	02/24/2017	Active	<input checked="" type="checkbox"/>
Image Small		Taken Seat	0%

Attendees

Name	Partner
01	Agus
02	Budi

7.9 PROSES CREATE ATAU WRITE FIELD MANY2MANY

Menambah tags di instruktur. Buat file baru dengan nama **add-partner-tags.php**.

Isinya.

```
<pre>
```

```

<?php
include "ripcord/ripcord.php";

$url    = "http://localhost:8010";
$username = "admin";
$password = "1";
$dbname = "academic";

/* login */
echo "<h2>login process</h2>";

$common = ripcord::client("$url/xmlrpc/2/common");
$uid = $common->authenticate($dbname, $username, $password, array());
var_dump($uid);

/* create $models object*/
$models = ripcord::client("$url/xmlrpc/2/object");

/* create tag 1 */
$tag1_id = $models->execute_kw($dbname, $uid, $password,
    'res.partner.category', 'create',
    array(
        array(
            'name'      => "Senior",
        ),
    )
);
var_dump($tag1_id);

/* create tag 2 */
$tag2_id = $models->execute_kw($dbname, $uid, $password,
    'res.partner.category', 'create',
    array(
        array(
            'name' => 'Part Time',
        )
    )
);
var_dump($tag2_id);

/* search ID instruktur */
$ids = $models->execute_kw($dbname, $uid, $password,
    'res.partner', 'search',
    array(
        array(
            array('name', '=', 'Joko'),
            array('is_instructor', '=', True)
        )
    )
);
var_dump($ids);
if (!$ids){
    die("instructor not found!");
}

echo "<h2>write category_id on res.partner</h2>";

$ret = $models->execute_kw($dbname, $uid, $password,
    'res.partner', 'write',
    array(
        $ids,
        array(

```

```

        'category_id' =>array(
            array(4,$tag1_id),
            array(4,$tag2_id)
        )
    )
);
var_dump($ret);


```

Jalankan dan lihat hasilnya.

Instructors / Joko

Edit
Create

Print
Action



Joko

Active

\$0.00
Invoiced

Address
Website
Is Instructor
Tags

☒

Full Time
Senior
Bandung

Junior
Part Time

Job Position
Phone
Mobile
Fax
Email
Title
Language

English

Contacts & Addresses
Internal Notes
Sales & Purchases
Accounting

8 REKAP HARI 5

Report RML, install plugin OpenOffice, membuat template report baru, modif report lama, menyertakan di addons.

Report Webkit, instalasi, bikin report lewat user interface, bikin report lewat addons.

Dashboard, menampilkan graph, calendar, dan list.

Web service PHP XML-RPC, login, create, write, dan create dengan one2many fields.

9 PENUTUPAN

Download module yang udah jadi dari sini

https://www.dropbox.com/sh/2sdljmfapmlacb/AAD3_J7aEDV-6FM7jjAFYCGGa?dl=0