

1 HARI 1: PEMBUKAAN

Buku ini menceritakan bagaimana mulai membuat addons untuk odoo 10 secara lengkap. Pembahasan dimulai dari membuat folder, membuat file identifikasi addon, membuat class, membuat view XML, inherit class, inherit view, advanced view, workflow, security, wizard, dashboard dan report, bahasa, sampai dengan web services.

1.1 TARGET PENCAPAIAN 5 HARI

HARI 1:

- Intro
- Contoh Soal Aplikasi : Academic Information System
- Struktur Addons
- Desain ERD
- Bikin Addons Academic
- Bikin Menu dan ActionWindow
- Class Course
- Class Session
- Relasi Course ke Session
- Class Attendee
- Relasi Session ke Attendee

HARI 2:

- Inheritance – Instructor
- Functional Fields – Percentage Taken Seats
- Event OnChange
- Constraints
- Nilai Default – Lambda Function
- Fitur Duplicate

HARI 3:

- Advanced View
- Workflow

HARI 4:

- Security
- Wizard

HARI 5:

- Internationalization
- Report
- Dashboard
- Web Services

1.2 CONTOH SOAL APLIKASI : ACADEMIC INFORMATION SYSTEM

Contoh soal yang kita jadikan bahan praktek adalah system informasi Akademik. Terdiri dari data **Course** yang punya banyak **Session**. Setiap **Session** dihadiri oleh banyak peserta (**Attendee**).

Course ada penganggungjawabnya, yang kita link ke **User** odoo. Setiap **Session** ada instruktur yang link ke **Partner** odoo.

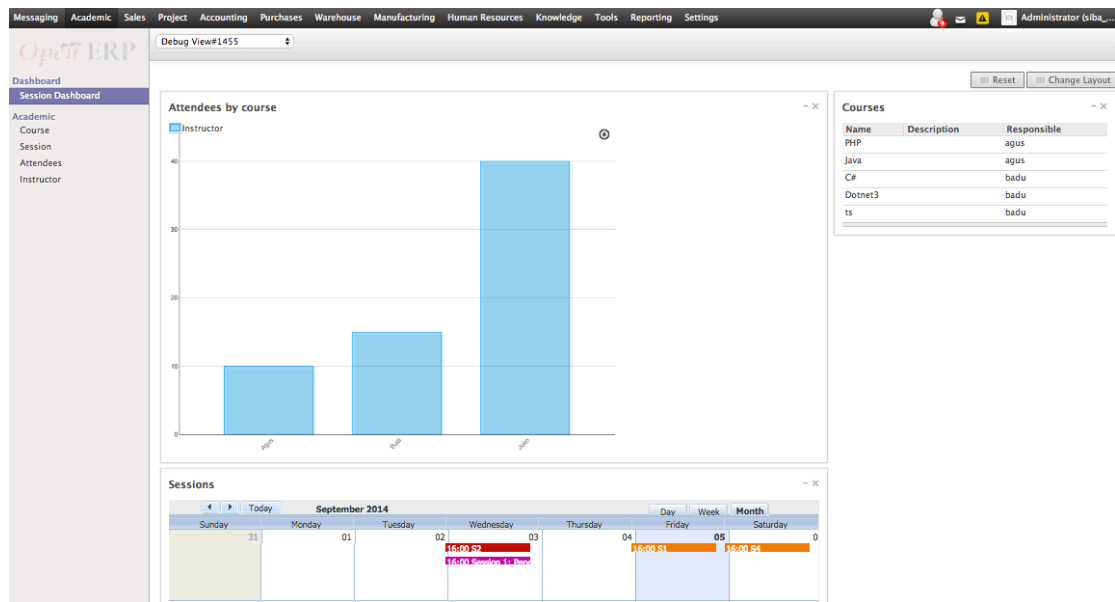
Setiap **Attendee** dihadiri oleh peserta yang juga di-link **Partner** odoo.

Partner yang udah jadi **Instruktur** pada suatu **Session** nggak boleh lagi jadi **Attendee** pada **Session** itu.

1.3 TAMPAK JADI

Berikut ini tampilan tampak jadi aplikasi Academic information system yang akan kita bangun sama-sama..

Menu utama dan tampilan awal dashboard



Gambar 1 Tampilan dashboard

Menu Daftar Course yang menampilkan daftar Course yang tersedia.

The screenshot shows the 'Daftar Course' (Add Course) menu in OpenERP. The top navigation bar includes: Messaging, Academic, Sales, Project, Accounting, Purchases, Warehouse, Manufacturing, and More. The left sidebar shows the 'Academic' menu with options: Course, Session, Attendees, Add attendee, and Instructor. The main content area features a 'Daftar Course' header with a search bar and a 'Create or Import' button. Below this is a table with columns 'Name' and 'Description'.

Name	Description
<input type="checkbox"/> PHP	
<input type="checkbox"/> Java	
<input type="checkbox"/> Dotnet	

Gambar 2 Menu daftar Course

Aplikasi ini mendukung Bahasa Indonesia.. dan bahasa apapun yang kita mau translate.

Messaging

Akademik

Sales

Project

Akuntansi

Purchases

Warehouse

Manufacturing

More

Administrator (siba_d...

OpenERP

Akademik

Kursus

Sesi

Peserta

Tambah Peserta

Instruktur

Debug View#1409

Daftar Session

Create or Import

PDF or Excel 1-5 of 5

<input type="checkbox"/>	Nama	Kursus	Instruktur	Tanggal Start	Durasi	Jumlah Kursi	Apakah Active?	Peserta	Tempat yang udah Terpakai
<input type="checkbox"/>	S1	Java	Budi	05/09/2014	33	5	<input checked="" type="checkbox"/>	(9 records)	
<input type="checkbox"/>	S2	PHP	Budi	03/09/2014	69	5	<input checked="" type="checkbox"/>	(6 records)	
<input type="checkbox"/>	S4	Java	Agus	06/09/2014	39	10	<input checked="" type="checkbox"/>	(3 records)	
<input type="checkbox"/>	S6	Java	Budi	08/09/2014	48	5	<input checked="" type="checkbox"/>	(0 records)	
<input type="checkbox"/>	Session 1: Pengenalan Dotnet	Dotnet3	Joko	03/09/2014	10	40	<input checked="" type="checkbox"/>	(1 records)	

Gambar 3 Tampilan dalam Bahasa Indonesia

Daftar Session, tampil dalam warna sesuai kondisi tertentu. Session bisa di-grouping per Course dan Tanggal. Bisa juga di-filtering sesuai kondisi tertentu.

Messaging

Academic

Sales

Project

Accounting

Purchases

Warehouse

Manufacturing

More

Administrator

OpenERP

Academic

Course

Daftar Session

Create or Import

PDF or Excel 1-4 of 4

Session

Attendees

Add attendee

Instructor

<input type="checkbox"/>	Name	Course	Instructor	Start Date	Duration	Number of Seats	Is Active?	Attendees	Taken Seats
<input type="checkbox"/>	S1	Java	Budi	09/03/2014	0	5	<input checked="" type="checkbox"/>	(2 records)	<div></div>
<input checked="" type="checkbox"/>	S2	PHP	Budi	09/03/2014	0	5	<input checked="" type="checkbox"/>	(3 records)	<div></div>
<input type="checkbox"/>	S4	Java		09/03/2014	0	10	<input checked="" type="checkbox"/>	(0 records)	<div></div>
<input type="checkbox"/>	S6	Java		09/03/2014	0	5	<input checked="" type="checkbox"/>	(0 records)	<div></div>

Gambar 4 Tampilan daftar Session

Contohnya grouping Session per Course dan Start Date...

Messaging

Academic

Sales

Project

Accounting

Purchases

Warehouse

Manufacturing

More

Administrator

OpenERP

Academic

Course

Session

Attendees

Add attendee

Instructor

Daftar Session

Create or Import

Group	<input type="checkbox"/>	Name	Course	Instructor	Start Date	Duration
▼ PHP (1)						
<input type="checkbox"/>	S2	PHP	Budi	09/03/2014		
▼ Java (3)						
<input type="checkbox"/>	S1	Java	Budi	09/03/2014		
<input type="checkbox"/>	S4	Java		09/03/2014		
<input type="checkbox"/>	S6	Java		09/03/2014	0	5

Course

Filters

Non zero duration

Group By...

Course

Start Date

Custom Filters

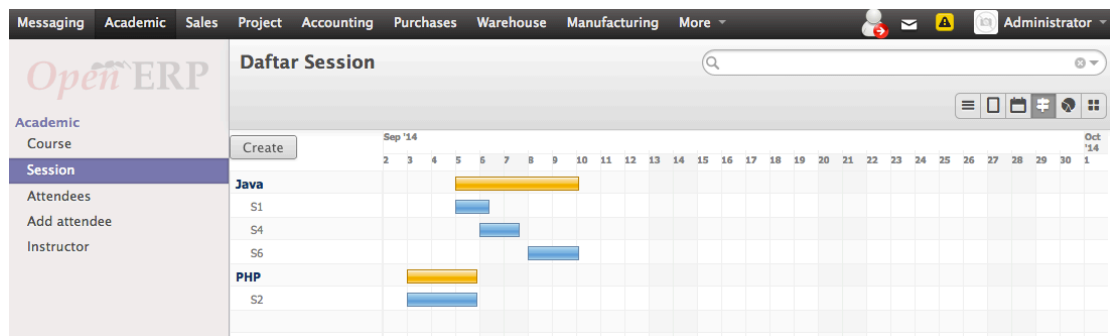
Save current filter

Advanced Search

Add to Dashboard

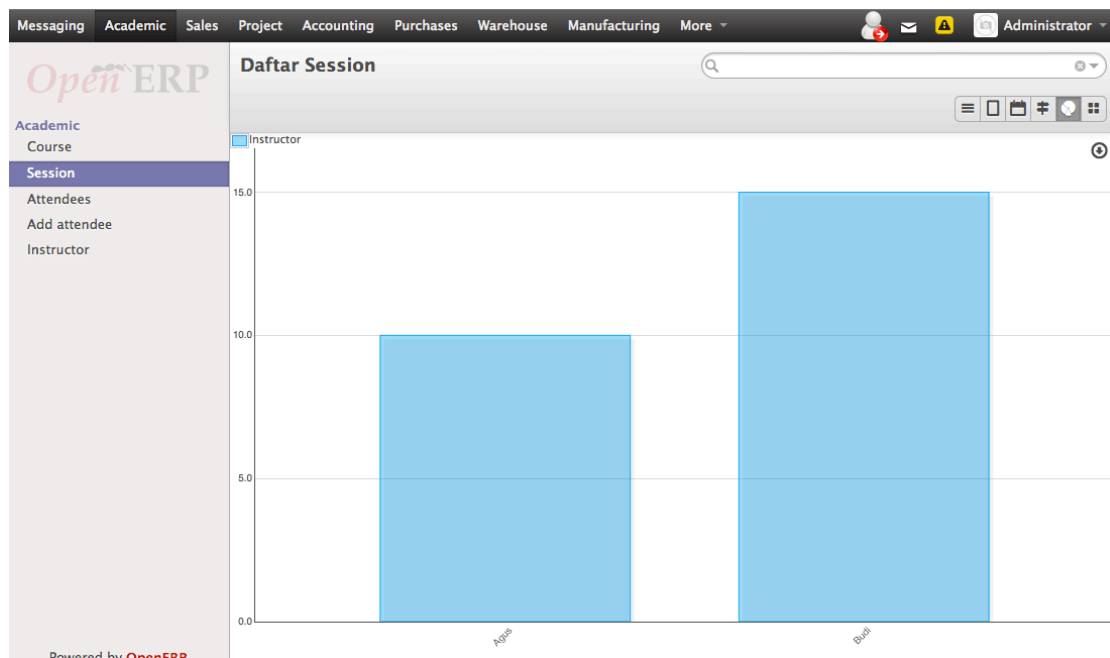
Gambar 5 Grouping Session per Course dan Start Date

Session bisa ditampilkan dalam Gantt Chart...



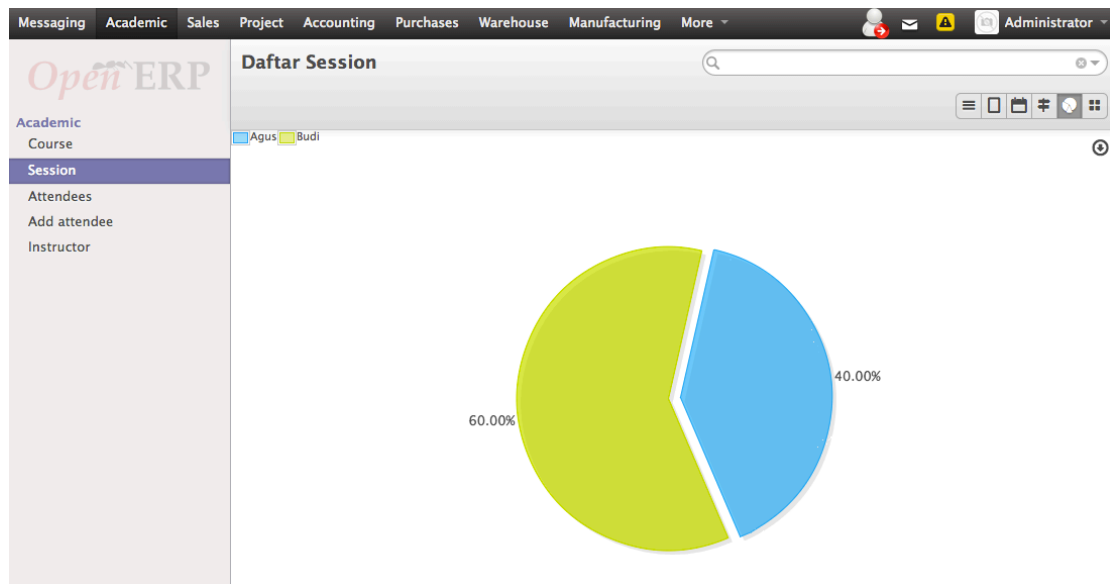
Gambar 6 Tampilan gantt chart

Session bisa disajikan dalam Graph, contohnya grafik total Durasi per Instuktur Session...



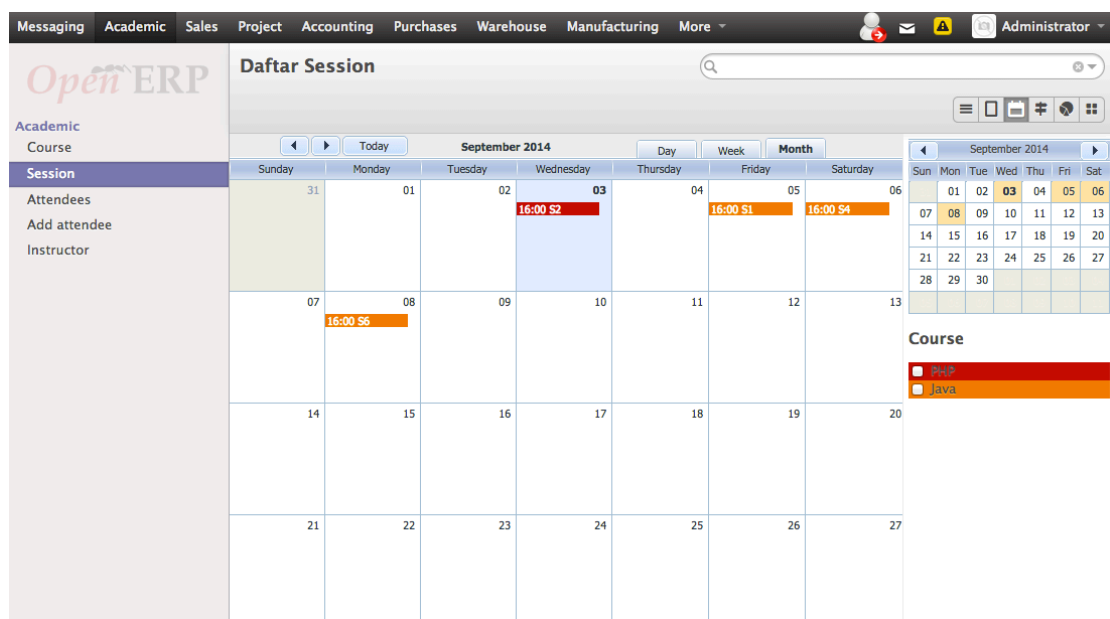
Gambar 7 Grafik Total Durasi per Instruktur Session

Grafik bisa tampil dalam Pie Chart...



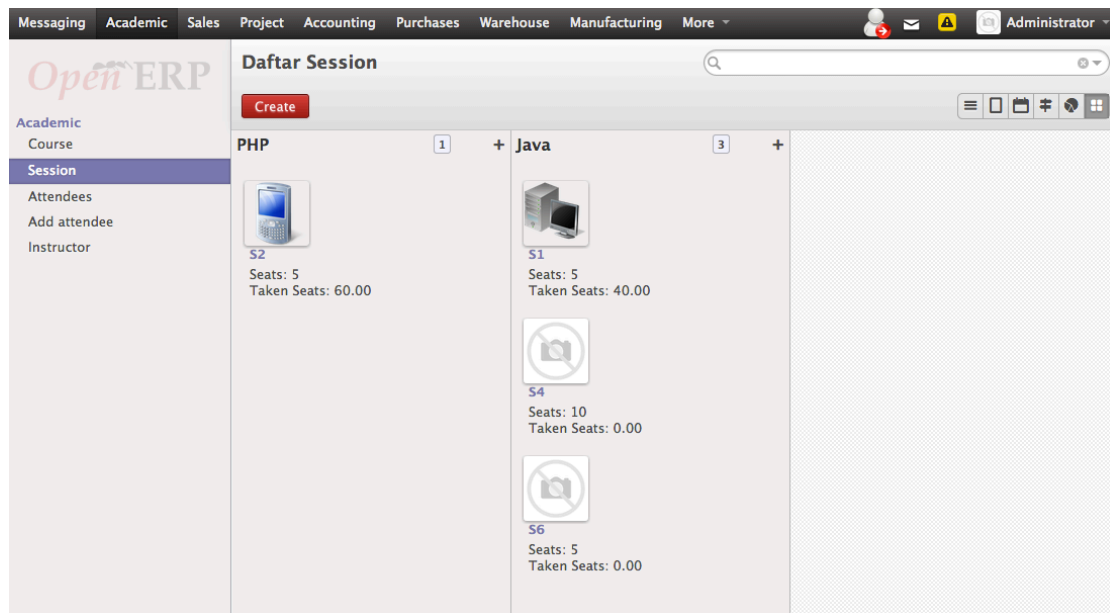
Gambar 8 Total durasi per instruktur dalam Pie hart

Session bisa ditampilkan dalam Calendar...



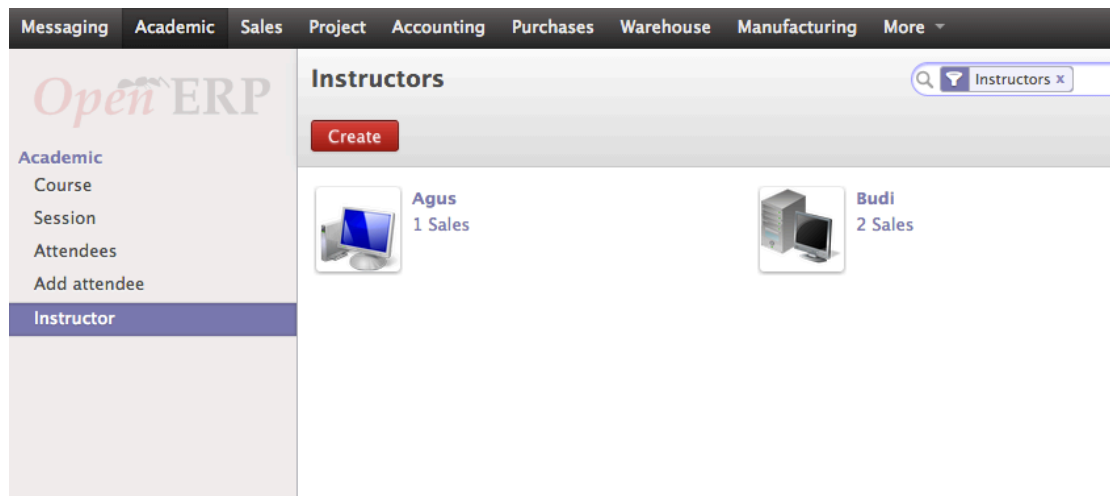
Gambar 9 Calendar view session

Session bisa ditampilkan dalam Kanban, dikelompokkan dalam Course...



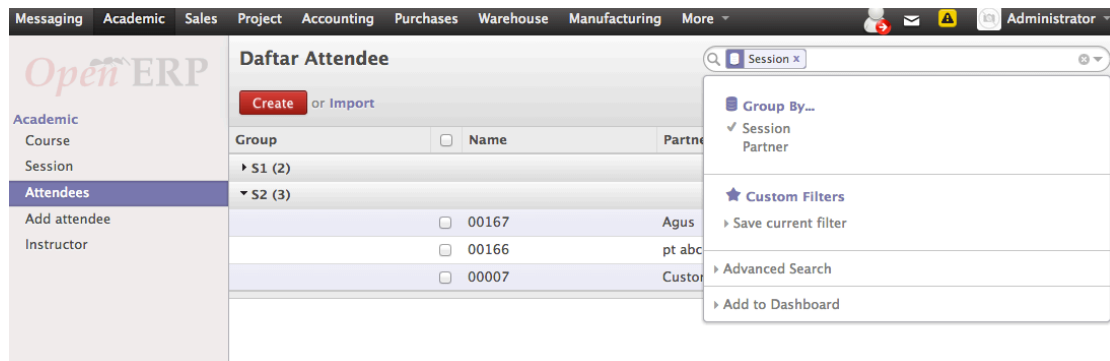
Gambar 10 Session Kanban View

Menu Daftar Instruktur, yang merupakan inherit dari Partner dengan penambahan field `is_instructor`...



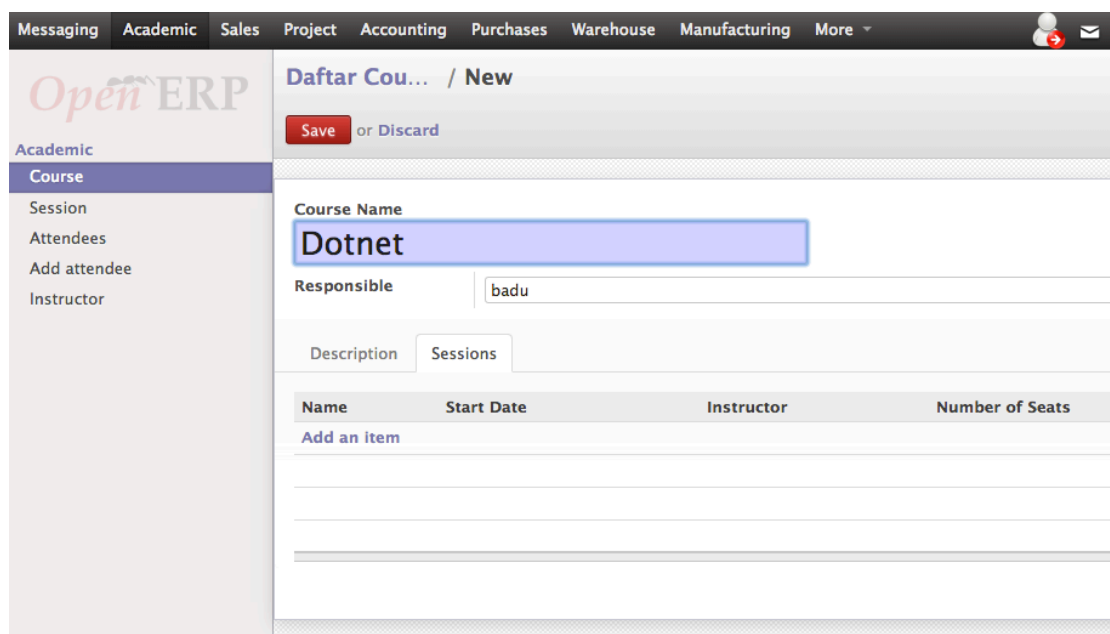
Gambar 11 Daftar instruktur

Menu Daftar Attendee, bisa dikelompokkan dalam Group atau Partner...



Gambar 12 Daftar Attendee

Membuat Course, bisa langsung bikin Session disitu...



Gambar 13 Create Course

Create Session, terdapat Constraint jika Attendee dan Instruktur sama partnernya, ada warning message jika Number of Seats nilai negative atau kurang dari jumlah Attendee yang udah ada saat itu...

Create: Sessions

Confirm **Draft** Confirmed Done

Session Name
Session 1: Pengenalan Dotnet

Course: Dotnet
Instructor: Joko
Start Date: 09/03/2014
Image Small: [Placeholder]

Duration: 10
Number of Seats: 40
Is Active?: ☒
Taken Seats: 0.00%

Attendees

Name	Partner
Add an item	

Save & Close Save & New or Discard

Gambar 14 Create Session

Create Attendee; ada event onchange di Partner yang otomatis mengisi kolom Name dengan ID partner diformat 5 digit.

Create: Attendees

Session: [Empty]
Partner: Agus
Name: 00167

Save & Close Save & New or Discard

Gambar 15 Create Attendee

Create Attendee menggunakan Wizard supaya bisa create Attendee sekaligus banyak, nggak satu per satu ...

Add attendee

Session: Session 1: Pengenalan Dotnet

Attendees:

Partner	
Joko	
Agus	
SIBA Group	
Add an item	

Buttons:

Gambar 16 Add Attendee wizard

Atau bisa juga daftarin banyak Attendee sekaligus ke banyak Session...

Add Attendees

Debug View#324

Session:

Name	Course	Instructor	Start Date	Duration	Number of Seats	Is Active?	Attendees	Taken Seats
S5	Dotnet	joko	09/04/2014	0	0	<input checked="" type="checkbox"/>	(1 records)	
S6	Dotnet	agus	09/04/2014	0	0	<input checked="" type="checkbox"/>	(3 records)	
S1	C#	badu	09/04/2014	0	0	<input checked="" type="checkbox"/>	(3 records)	
S1	MySQL	joko	09/04/2014	0	0	<input checked="" type="checkbox"/>	(0 records)	

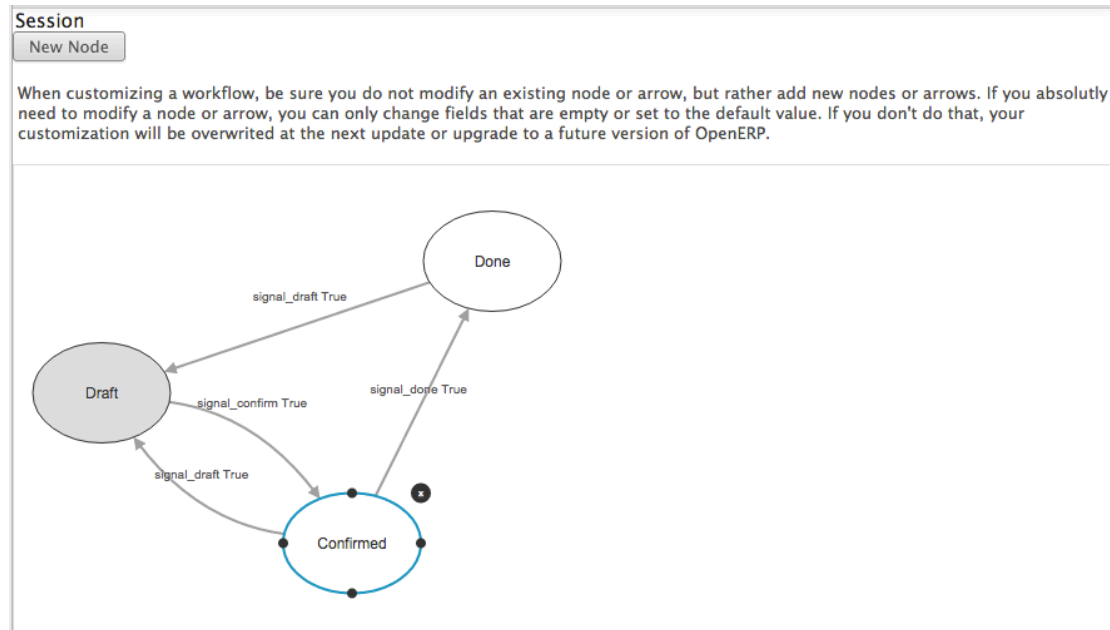
Attendees:

Partner	
Administrator	
badu	

Buttons:

Gambar 17 Add Attendees ke banyak Session

Terdapat Workflow dinamis untuk mengubah status Session mulai dari Draft, Confirmed, dan Done...



Gambar 18 Workflow diagram

Ada group access security, yaitu group Academic/Manager dan Academic/User..

Groups / Academic / Manager

Edit Create Attachment(s) More 62 / 63

Application	<input type="checkbox"/>	Name	Academic / Manager		
Share Group	<input type="checkbox"/>	Portal	<input type="checkbox"/>		
Users	Inherited	Menus	Views	Access Rights	Rules Notes
Object	Read Access	Write Access	Create Access	Delete Access	Name
academic.course	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	course_manager
academic.session	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	session_manager
academic.attendee	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	attendee_manager

Gambar 19 Group access rights

Report untuk menghasilkan laporan dalam PDF...

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

Session Report

Java - S1

From 09/05/2014.

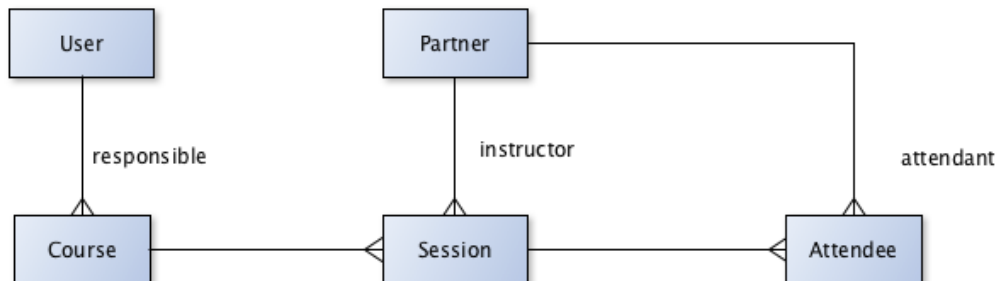
Attendees:

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

Gambar 20 Laporan PDF

2 DESAIN ERD

Berikut gambar entity relationships diagram sesuai spesifikasi yang udah dibahas diatas...



Gambar 21 ERD sistem Academic

2.1 COURSE

Berikut ini detail field table Course. Setiap Course ada penanggungjawabnya (responsible) yang link ke table User odoo.

Field	Type	Keterangan
name	Char, 100	Nama Course
description	Text	Keterangan lengkap
responsible_id	FK many2one ke table user	Siapa penanggung jawab Course ini
session_ids	One2many ke table Session	Satu Course bisa punya banyak Session

2.2 SESSION

Berikut ini detail field table Session, table yang mencatat Session yang dimiliki oleh suatu Course.

Setiap Session ada Instructor-nya yang link ke table Partner odoo.

Field	Type	Keterangan
name	Char, 100	Nama Course
course_id	Text	Keterangan lengkap

instructor_id	FK many2one ke table Partner	Siapa Instruktur Session ini
start_date	Date	Tanggal mulai Session
duration	Integer	Berapa hari durasi Session
seats	Integer	Berapa maksimal kapasitas peserta
active	Boolean	Apakah aktif atau nggak
attendee_ids	One2many ke table Attendee	Satu Session bisa punya banyak Attendee
taken_seats	Function Field	Dihitung berdasarkan seats dan jumlah peserta setiap seat
state	Enum (Draft, Confirmed, Done)	Status session terkait workflow
image_small	Binary	Bisa upload image langsung

2.3 ATTENDEE

Berikut ini detail field table Attendee, table yang mencatat siapa-siapa aja Partner OperERP yang menghadiri suatu Session.

Field	Type	Keterangan
name	Char, 100	Nama Attendee, boleh diisi dengan nomor pendaftaran
session_id	FK many2	Attendee untuk

id	one ke tabl e Ses sio n	k Sessi on yang mana
part ner_ id	FK ma ny2 one ke tabl e Par tne r	Siapa partn er yang jadi Atten dee

2.4 INSTRUCTOR PARTNER

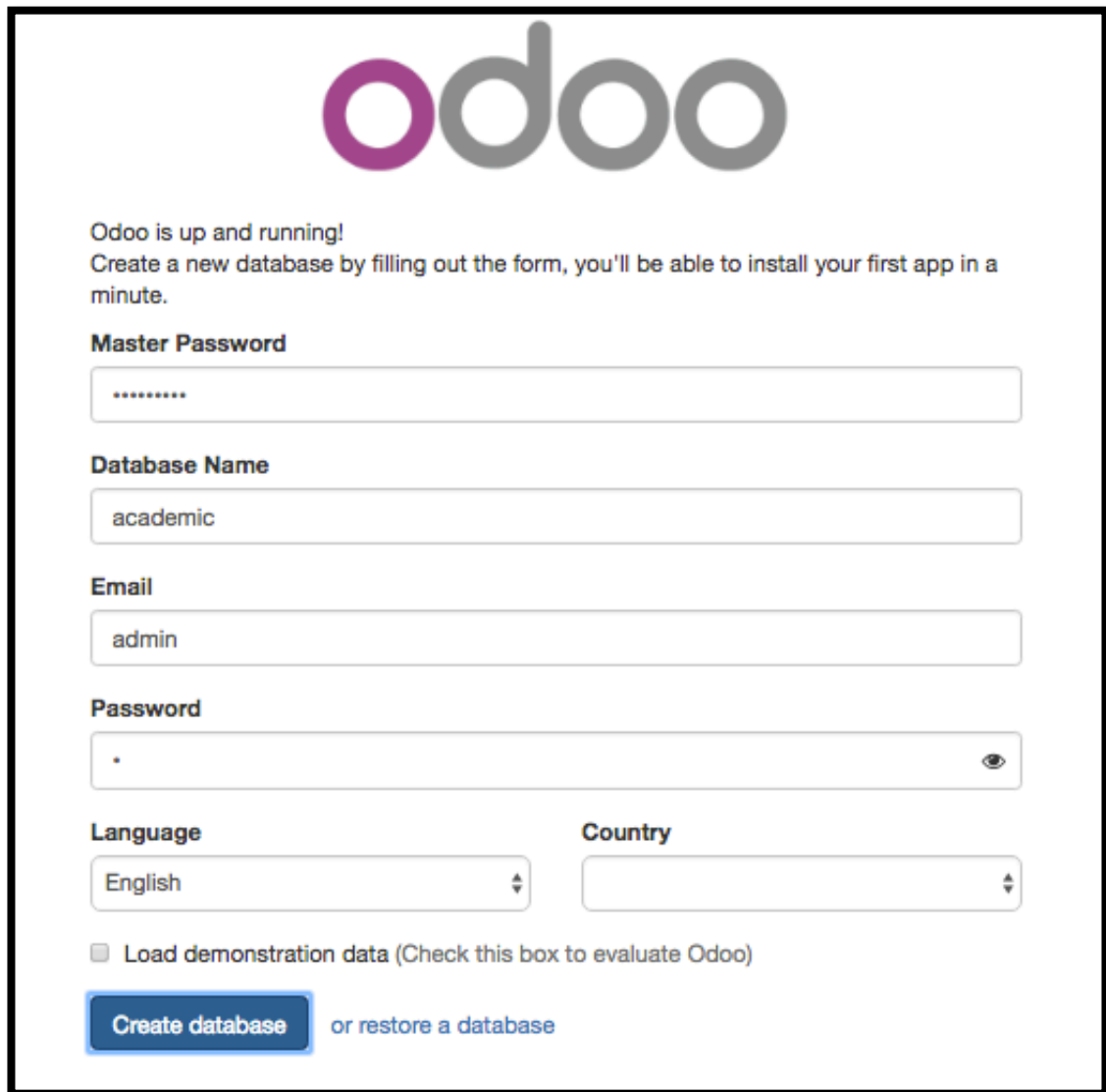
Instructor diturunkan dari table Partner, jadi memiliki semua field yang ada di Partner, dengan tambahan field berikut ini...

Field	Type	Keterangan
is_instructor	Boolean	Apakah partner ini instruktur atau bukan

3 PERSIAPAN

3.1 INSTALL DATABASE

Install database odoo baru. Dari halaman login odoo klik link [Manage Databases](#). Jika belum ada database, maka langsung masuk ke halaman Manage Database.

The image shows the Odoo database creation interface. At the top is the Odoo logo. Below it, a message states "Odoo is up and running! Create a new database by filling out the form, you'll be able to install your first app in a minute." The form contains several fields: "Master Password" (masked with dots), "Database Name" (containing "academic"), "Email" (containing "admin"), and "Password" (masked with a single dot and featuring a toggle icon). There are also dropdown menus for "Language" (set to "English") and "Country". A checkbox labeled "Load demonstration data (Check this box to evaluate Odoo)" is present. At the bottom, there is a blue "Create database" button followed by the text "or restore a database".

odoo

Odoo is up and running!
Create a new database by filling out the form, you'll be able to install your first app in a minute.

Master Password

.....

Database Name

academic

Email

admin

Password

•

Language **Country**

English

☐ Load demonstration data (Check this box to evaluate Odoo)

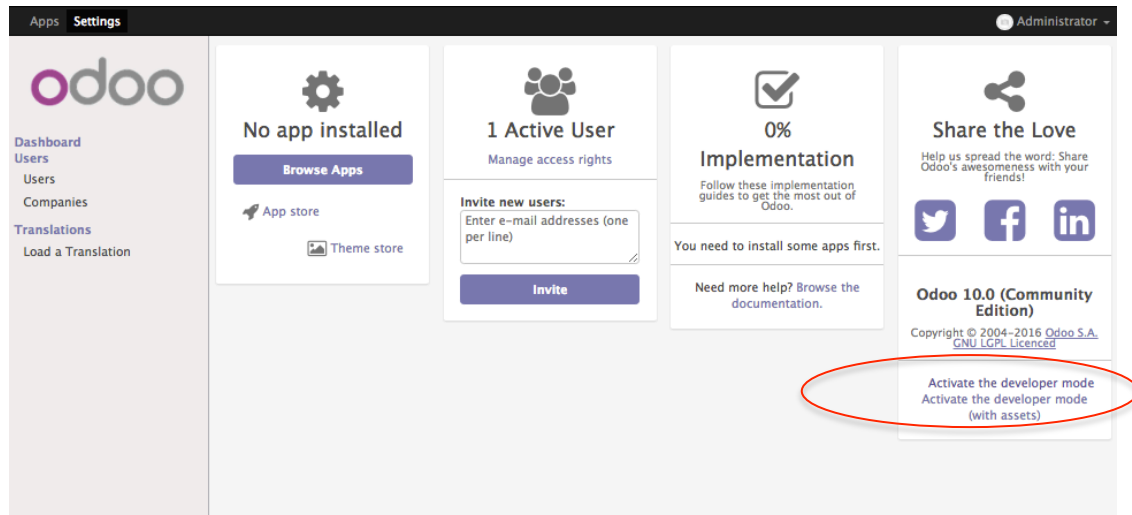
Create database or restore a database

Gambar 22 Bikin database baru

Buat database dengan nama yang dikehendaki misalnya **academic**.

3.2 ACTIVATE DEVELOPER MODE

Login ke database yang baru dibuat, activate Developer Mode melalui menu Settings agar muncul semua fitur teknis Odoo.



Gambar 23 Activate Developer Mode

3.3 START SERVER DAN UPDATE MODULE

Selama development kita akan menjalankan odoo server secara manual melalui terminal Linux atau command prompt Windows.

Disini kita perlu tentukan file konfigurasi yang akan digunakan melalui argument `-C`, supaya kita bisa atur konfigurasi waktu development lebih flexible dan cepat.

```
./odoo-server -c odoo-server.conf
```

Isi dari file konfigurasi `odoo-server.conf` yang paling penting:

```
db_host = localhost      # ip address server db postgres
db_port = 5432           # port db postgres
db_user = odoo           # username postgres
db_password = odoo       # password user postgres
addons_path = odoo/addons/,/Users/odoo/addons/  #lokasi addons
#logfile=/tmp/openrp.log  # kalo ada logfile, ditutup dulu pake #
```

Parameter yang perlu diperhatikan adalah `logfile` yang menentukan kemana informasi log server odoo akan ditampilkan. Kalau isinya berupa file seperti contoh diatas ke file `/tmp/odoo.log`, maka perlu kita disable dulu selama development biar informasi log nya muncul langsung di terminal.

Cara disablenya seperti di atas, yaitu dikasi tanda `#` yang artinya dibuat jadi comment line.

4 STRUKTUR ADDONS

4.1 STRUKTUR FILE

Addons odoo tersimpan dalam satu folder yang berada di folder addons server odoo.

Addons terdiri dari komponen-komponen :

Business Object: berupa class Python yang merupakan turunan dari `models.Model`, class bawaan odoo/`OpenObject`.

Data: bisa berupa file XML atau CSV yang berisi meta-data (view dan workflow), konfigurasi parameter module, atau data demo.

Wizard: kotak dialog yang berguna untuk memudahkan user dalam menginput data.

Report: bisa berupa QWEB template, yaitu untuk menampilkan report dari semua data untuk ditampilkan dalam HTML atau PDF.

Setiap addons modul harus punya file `__openerp__.py` dan `__init__.py`. kalo nggak, folder tersebut nggak dianggap sebagai modul oleh odoo.

4.1.1 FILE `__OPENERP__.PY`

Adalah tempat untuk kita mendefinisikan segala informasi tentang modul addons, seperti nama, keterangan, daftar modul odoo lain yang harus ada, referensi ke file XML atau CSV yang diperlukan oleh module. Data itu dituliskan dalam satu dictionary Python.

4.1.2 FILE `__INIT__.PY`

Adalah file Python module descriptor berisi semua file Python yang diperlukan dan termasuk dalam satu paket addon module.

4.2 PENEMPATAN FOLDER

By default, modul addon harus ditempatkan dibawah folder `addons` di struktur directory odoo, misalnya kalo odoo kita diinstall di folder `/opt/odoo-10`:

```
/opt/odoo-10/odoo/addons
```

Tapi sebetulnya addons yang kita buat boleh aja disimpan di sembarang folder asalkan folder itu di-set sebagai folder addons yang perlu dicari oleh odoo saat di jalankan.

Caranya adalah dengan nambahin `addons_path` pada file konfigurasi odoo, yaitu `odoo-server.conf`.

```
addons_path = odoo/addons/,/users/addons/
```

Artinya kita minta odoo untuk nyari addons di folder bawaannya yaitu `odoo/addons` dan juga folder kita yaitu `/users/addons`.

5 BIKIN ADDONS ACADEMIC

Kita asumsikan bikin folder addonsnya di lokasi addons standard odoo, yaitu `odoo/addons`.

5.1 BIKIN FOLDER

Bikin new folder dibawah situ, kasi nama `academic`.

5.2 BIKIN FILE `__OPENERP__.PY`

Bikin file dengan nama `__openerp__.py`

Isinya seperti ini:

```
{
    "name": "Academic Information System Day 1",
    "version": "1.0",
    "depends": [
        "base",
    ],
    "author": "akhmad.daniel@gmail.com",
    "category": "Education",
    'website': 'http://www.vitraining.com',
    "description": """\
Academic Information System Day 1
""",
    "data": [
    ],
    "installable": True,
    "auto_install": False,
    "application": True,
}
```

Gambar 24 Isi file `__openerp__.py` pada awalnya

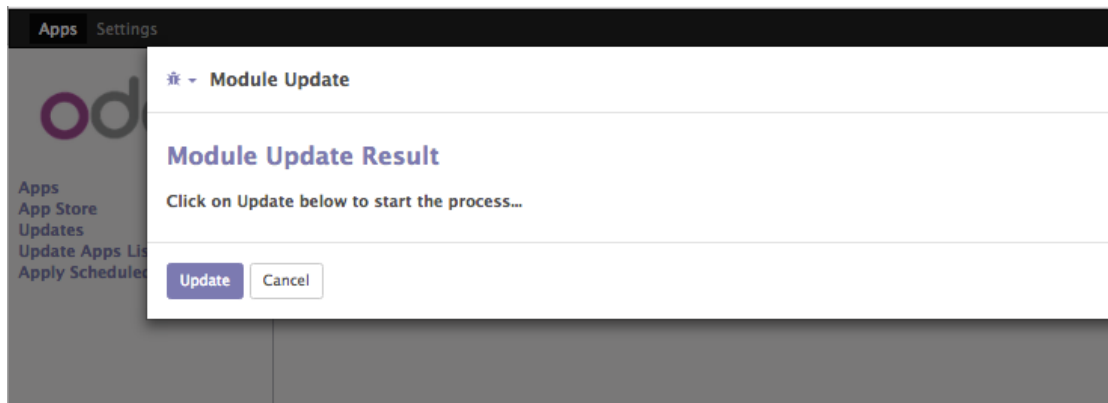
5.3 BIKIN FILE `__INIT__.PY`

Isinya sekarang masih kosong, karena kita belum punya file Python yang mau diimport, yang penting ada dulu filenya.

Struktur folder addons kita harus seperti ini.

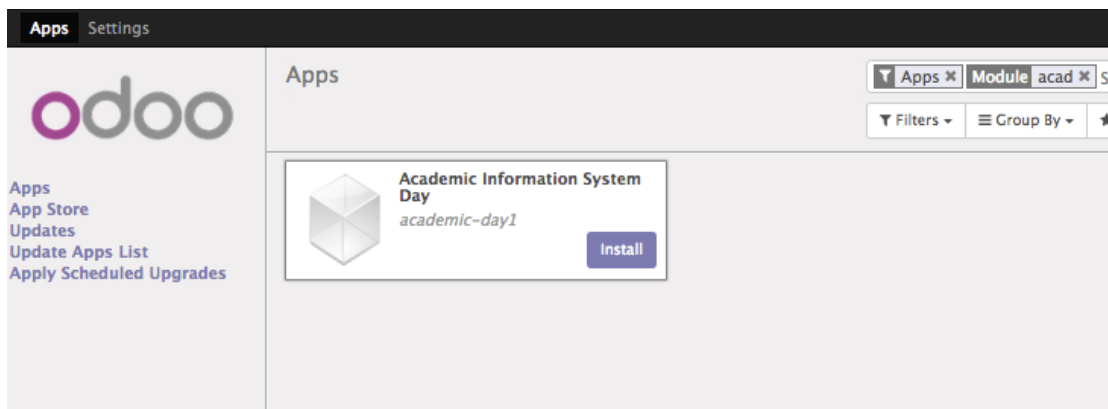
```
-- odoo
|   |-- addons
|       |-- academic
|           |-- __openerp__.py
|           \-- __init__.py
```

Supaya odoo kenal modul yang baru kita bikin diatas, harus klik menu **Settings > Apps > Update Apps List**.



Gambar 25 Update Apps List

Kalau proses pencarian modul baru udah selesai, maka modul yang baru kita buat akan terdaftar waktu kita klik menu **Settings > Apps > Apps**. Cari nama module yaitu **academic**, hilangkan filter Installed jika perlu.



Gambar 26 Modul sudah ter-detect oleh odoo

Jangan dulu klik **Install** saat ini.

5.4 RELOAD ODOO

Selama development addons kita perlu restart odoo supaya dia ngeload ulang perubahan pada modul yang baru dilakukan. Sebetulnya ada fasilitas **Upgrade** module di odoo tapi nggak bisa dilakukan sepenuhnya untuk update modul.

Jika ada perubahan di file PY maka perlu restart Odoo.

Jika hanya perubahan di file XML, cukup klik Upgrade module.

5.4.1 LINUX

Di linux gunakan command ini di dalam folder instalasi odoo untuk meng-update modul kita...

```
./odoo-server -c odoo-server.conf --update=academic
```

Pastikan juga kalo odoo udah diinstall secara service, servicenya dimatikan dulu supaya nggak bentrok.

5.4.2 WINDOWS

Development addon module lebih baik dilakukan di Linux, tapi memang bisa juga di Windows.

Untuk meng-update addon module, gunakan juga command --update melalui Command Prompt seperti di Linux...

```
cd C:\program files\odoo\server\odoo  
odoo-server.exe -c odoo-server.conf --update academic
```

Untuk di Windows, pastikan odoo di Windows Service sudah di stop, karena kalo nggak distop bisa menyebabkan dua service odoo jalan, yaitu yang di service dan yang di jalankan manual dari Command Prompt untuk development.

6 BIKIN MENU DAN ACTIONWINDOW

Langkah pertama setelah membuat folder addons `academic` dan file `__openerp__.py` dan `__init__.py` adalah membuat menu aplikasi dan halaman yang muncul ketika menu itu diklik.

6.1 BIKIN MENU

Menu aplikasi yang mau dibuat terdiri dari menu utama dan sub menu. Menu utama ditempatkan di manu bar paling atas odoo, dan kalo di klik muncul sub menu di bawahnya.

Susunan menu yang hendak kita rancang kira-kira seperti ini...

Academic (top Menu setingkat Sales, Purchase, dll)
 Academic
 Course
 Session
 Attendee
 Instructor

Bikin file `top_menu.xml` dibawah folder `academic`.

Isinya begini...

```
<odoo>
  <data>

    <menuitem id="academic_0"
              name="Academic"
              sequence="20"/>

    <menuitem id="academic_1"
              name="Academic"
              parent="academic_0"
              sequence="20"/>

    <menuitem id="menu_academic_course"
              name="Course"
              parent="academic_1"
              action="action_course_list"
              sequence="20"/>

  </data>
</odoo>
```

Gambar 27 File menu.xml

Setiap file XML di odoo berada didalam tag `<odoo>` dan `<data>`.

Lalu di setiap `<data>` boleh dimasukkan macam-macam tag yang tujuannya untuk meng-insert data ke table internal dan semua table lain yang ada di odoo.

Dalam hal ini kita meng-insert tag `<menuitem>` yang gunanya untuk menampilkan menu di odoo.

Tag `<menuitem>` punya banyak attribute, diantaranya..

Attribute `id` adalah referensi untuk setiap record menuitem yang nantinya bisa dipakai sebagai referensi buat menu yang lain.

Attribute `name` gunanya untuk nampilin label menu.

Attribute `action` gunanya untuk nentuin apa nama action window (semacam halaman) yang akan ditampilkan ketika menu ini diklik.

Attribute `sequence` gunanya untuk nentuin urutan munculnya menu ini diantara menu lain yang setingkat.

Attribute `parent` gunanya untuk nentuin induk dari menu ini. Isinya adalah attribute `id` dari menu induk tersebut. Contohnya menu Course, Session, dan Attendee induknya Academic. Sementara menu Academic sendiri induknya adalah menu Academic yang ada di top menu.

Jadi sesuai rancangan dan XML yang baru dibuat, kira-kira gini nih mapping nya...

```
Academic      (id academic_0)
  Academic    (id academic_1, parent academic_0)
    Course    (id academic_1_1, parent academic_1)
    Session   (belum dibuat)
    Attendee  (belum dibuat)
```

Instructor (belum dibuat)

6.2 BIKIN ACTION WINDOW

Next, kita perlu buat record action window untuk nangkep klik dari menu. Action window boleh kita ibaratkan kayak halaman web yang muncul waktu suatu link diklik.

Contohnya di XML tadi, waktu kita klik menu `Course`, action diarahin ke `action_course_list`. Berarti kita perlu bikin action window dengan id `action_course_list`.

Penentuan `id` di odoo boleh bebas asal representative dan nggak salah dalam me-referensikan dari elemen lain.

Saat suatu elemen XML direferensikan oleh elemen lain, syaratnya elemen yang direferensikan tersebut harus udah ada di atas yang me-referensikan.

Contohnya `action_course_list` di-refer dari menu `Course`, berarti sebelum menu item `Course` dibuat, harus udah ada dulu action window `action_course_list`. Artinya harus ditulis di bagian atas sebelum menu item.

Jadi edit lagi `menu.xml`, tambahi action window untuk masing-masing menu...

Ini action window untuk Course List...

```
<odoo>
  <data>

    <record id="action_course_list" model="ir.actions.act_window">
      <field name="name">Daftar Course</field>
      <field name="res_model">academic.course</field>
      <field name="view_mode">tree,form</field>
      <field name="help" type="html">
        <p class="oe_view_nocontent_create">
          Click to add a Course
        </p>
      </field>
    </record>
  </data>
</odoo>
```

```

        <p>klik tombol create untuk bikin Course baru</p>
    </field>
</record>

<menuitem id="academic_0"
    name="Academic"
    sequence="20"/>

<menuitem id="academic_1"
    name="Academic"
    parent="academic_0"
    sequence="20"/>

<menuitem id="menu_academic_course"
    name="Course"
    parent="academic_1"
    action="action_course_list"
    sequence="20"/>

</data>
</odoo>

```

Gambar 28 menu.xml udah ditambahi action window course

Jangan lupa, tambahan action window harus ditulis di atas deklarasi menu item yang udah kita buat sebelumnya.

Deklarasi action window menggunakan tag `<record>`. Tag ini gunanya untuk menginsert record langsung ke object (table) yang ditulis pada attribute `model`. Atribut lainnya yaitu `id` gunanya sebagai id kalau elemen ini mau di-refer oleh elemen XML lain, contohnya di-refer dari menu.

Waktu memakai elemen tag `<record>` untuk meng-insert record ke suatu object, didalamnya harus ditentukan nilai dari setiap field dari record yang akan diinsert.

Contohnya, waktu kita insert record ke object action window (nama lengkapnya `ir.actions.act_window`), field yang mandatory yang harus disertakan adalah field `name`, `res_model`, dan `view_mode`.

Field `name` adalah nama dari action window yang mau diinsert.

Field `res_model` adalah nama model (object/ table) yang dijadikan sumber data di halaman yang akan tampil. Ini adalah deklarasi model Python dan harus kita buat nanti

Field `view_mode` nentuin gimana data itu akan ditampilkan, apakah dalam list (tree) view, form, graph, calendar, dan lainnya... Field ini nantinya menentukan icon-icon apa aja yang muncul di list view Course untuk melihat data dalam list atau form view.

Field `help` sih sifatnya nggak mandatory, tapi kalo diisi gunanya untuk munculin informasi ketika belum ada record di table sumber data (`res_model`).

Oh iya, di odoo istilah object mewakili model class, dan setiap model class langsung berhubungan dengan suatu table di database. Jadi misalnya kita buat object `Course` pada module addons `academic`, maka di database otomatis ter-create table dengan nama `academic_course`. Field-fieldnya menyesuaikan dengan model class yang kita buat nanti di Python.

6.3 UPDATE `__OPENERP__`.PY

Setiap kali nambah suatu file XML, kita perlu cantumin file itu pada file `__openerp__.py` supaya waktu modul diinstall, file XML itu ikut diproses.

Taronya di key `data`, yang berupa list dari file-file XML yang mau disertakan, dipisahkan pake tanda koma kalo udah lebih dari satu file.

Seperti ini caranya...

```
{
  "name": "Academic Information System Day",
  "version": "1.0",
  "depends": [
    "base",
  ],
```

```

"author": "akhmad.daniel@gmail.com",
"category": "Education",
'website': 'http://www.vitraining.com',
"description": """\
Academic Information System Day1
""",
"data": [
    "menu.xml",
],
"installable": True,
"auto_install": False,
"application": True,
}

```

Gambar 29 Cantumin menu.xml di __openerp__.py

Pada step ini modul belum bisa di update karena kita belum buat file class model yang dipanggil sebagai sumber data di action window di atas.

Lanjut ke bagian model class dulu setelah ini...

7 CLASS COURSE

Udah di bahas sebelumnya, action window perlu informasi dari mana sumber data waktu dia dipanggil oleh menu. Ini ditentukan di field `res_model`. Isinya adalah nama model class Python.

Format penamaan class ini kita seragamkan saja dengan `nama_addons.nama_class`, walaupun sebetulnya boleh bebas tapi untuk menghindari kebingungan ke depannya.

Nama addons adalah sama dengan nama folder addon module yang udah kita buat sebelumnya, yaitu `academic`.

7.1 BIKIN CLASS COURSE

Pertama-tama buat file baru dibawah folder addon `academic`. Kasi nama yang mewakili object yang ada di dalamnya, yaitu `course.py`.

Isi file ini adalah deklarasi class `course` dan semua atribut dan method yang ada pada class tersebut.

Sesuai ERD dan spesifikasi table, berikut ini syntax deklarasi class `course` pada file `course.py`.

```
from odoo import api, fields, models, _

class Course(models.Model):
    _name = 'academic.course'
    _rec_name = 'name'

    name = fields.Char("Name")
    description = fields.Text(string="Description", required=False, )
    responsible_id = fields.Many2one(comodel_name="res.users",
                                    string="Responsible")
```

Gambar 30 Bikin class Course

Disini, class `course` buatan kita merupakan turunan dari class `models.Model` (bawaan odoo/OpenObject). Otomatis dia bakalan udah punya sifat (method dan properties) sama seperti induknya. Contohnya langsung connect ke table `academic_course`, bisa cari data, insert, update, delete, dan sebagainya.

Setiap class harus diset property `_name` dan `_columns` nya.

Property `_name` gunanya sebagai referensi bagi semua modul addons yang ada di odoo. Jadi kalo ada class atau XML lain yang perlu akses ke class ini, dia harus panggil dengan panggilan `academic.course`.

Seperti yang udah dilakukan oleh action view `action_course_list` sebelumnya dimana dia perlu field `res_model`, dan disana kita isikan dengan nama class course, yaitu `academic.course`.

Property `_columns` gunanya untuk definisiin kolom class ini. Property ini bentuknya Python dictionary, dimana ada key (nama kolom) dan value (jenis dan atribut kolom). Disini kita definisiin kolom `name`, `description`, dan `responsible_id`.

Untuk mudahnya, ibaratkan aja class Python ini langsung sebagai table database, karena nantinya setiap class pasti ada korelasinya dengan sebuah table di database, yang akan dibuat otomatis oleh odoo waktu kita install module addon ini.

Sesuai ERD, kolom `name` bertype `char` dengan panjang maximal 100, nggak boleh kosong alias `required=True`, dengan label di semua tampilan adalah `Name`.

Lalu ada kolom `description`, bertype `text` (lebih dari satu baris) dengan label di tampilan adalah `Description`.

Lalu ada kolom `responsible_id`. Ini typenya khusus yaitu `many2one` ke object lainnya yaitu `res.users`, label di tampilan adalah `Responsible`. Artinya, kolom ini punya relasi ke table lain, yaitu `res.users` (yaitu user yang bisa login di odoo) dan sifatnya `many2one`, artinya banyak course bisa dimiliki oleh satu user, alias satu user bisa punya banyak course. Ini sesuai dengan spesifikasi dan ERD aplikasi kita.

Konvensi: semua kolom berjenis relasi `many2one` kita buat dengan akhiran `_id`, supaya memudahkan kita dalam mengidentifikasi type kolom ini nantinya.

Kalo udah siap, panggil file `course.py` dari file `__init__.py`, supaya class kita diimport oleh odoo waktu dijalankan. Caranya gini...

```
import course
```

Gambar 31 Import class Course

Susunan file pada addons academic kita sejauh ini adalah seperti ini... ada tambahan `menu.xml` dan `course.xml`.

```
addons/academic
|-- __init__.py
|-- __openerp__.py
|-- course.py
`-- menu.xml
```

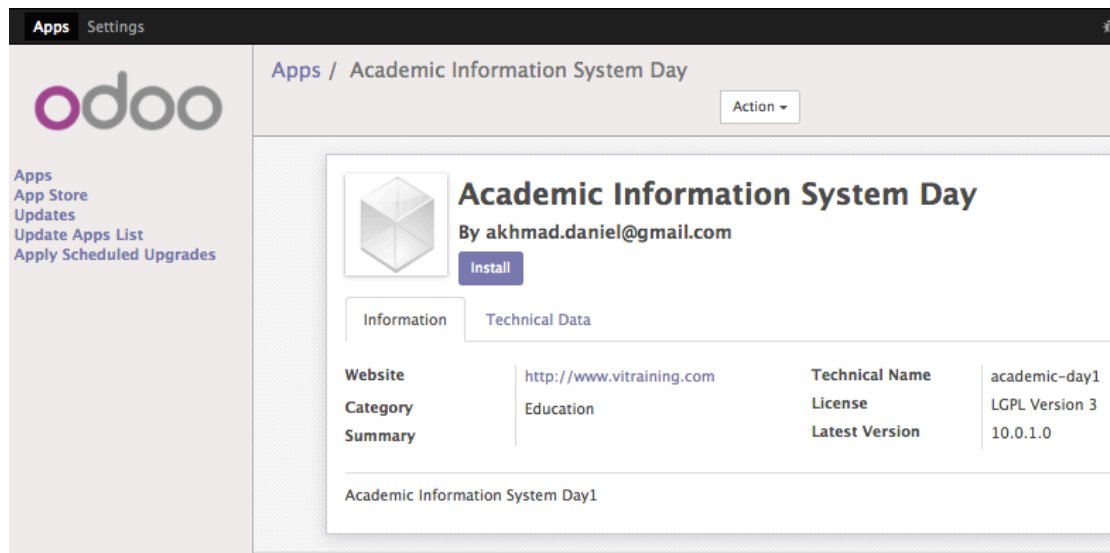
Mari test tampilan aplikasi kita...

Restart `odoo-server` dan update modul addon `academic`.

```
./odoo-server -c odoo-server.conf --u academic
```

Install module addons `academic`.

Cari modul dari menu **Apps > Apps**.



Gambar 32 Install module academic

Klik **Install**.

Lihat hasilnya di web browser... klik menu **Academic** di top menu.

Muncul list view dari course dimana belum ada data Course dan muncul tulisan help.



Gambar 33 Menu course udah muncul

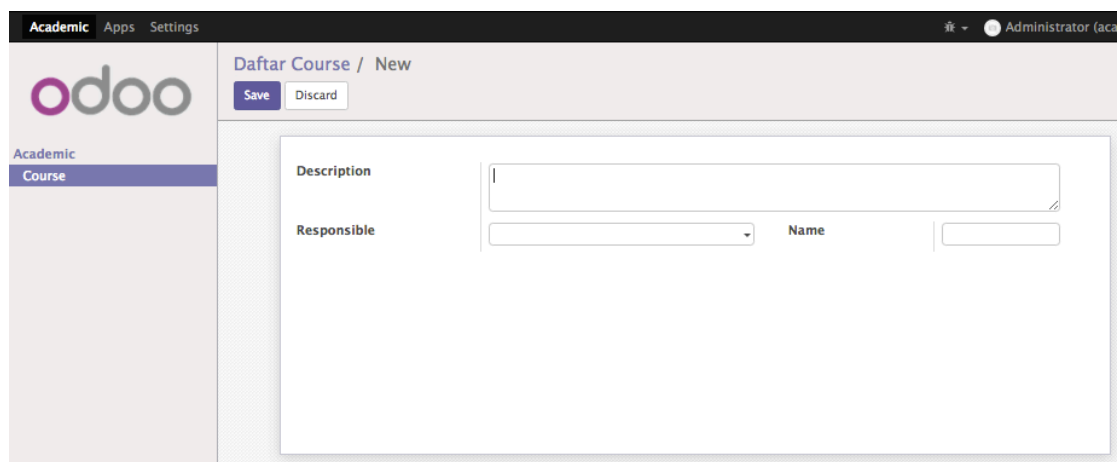
Klik tombol **Create...**

Muncul form view course dimana kita bisa isikan data Course. Kolom-kolomnya udah sesuai dengan yang kita definisikan di class course.

Kolom **Name** berupa text box dengan panjang maximal 100 dan nggak boleh kosong. Coba dikosongin terus klik tombol Save, nanti ada validasi bahwa dia nggak boleh kosong.

Kolom **Description** udah berupa text box beberapa baris dan boleh kosong.

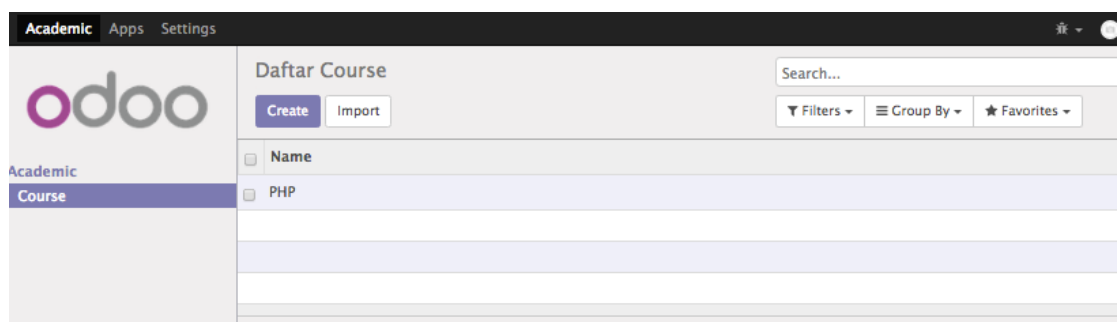
Kolom **Responsible** udah berupa pilihan drop down list karena kita set relasi many2one ke object `res.users`.

The screenshot shows the 'Daftar Course / New' form in Odoo. The form has a 'Description' text area and a 'Responsible' dropdown menu. There are 'Save' and 'Discard' buttons at the top. The left sidebar shows the 'Academic' app and 'Course' model. The top bar shows 'Academic', 'Apps', 'Settings', and the user 'Administrator (aca)'.

Gambar 34 Udah bisa create Course

Klik tombol **Save**. Akan terlihat tampilan list view data Course yang udah masuk. Dari list view boleh di klik untuk masuk lagi ke form view.

Boleh juga di centang untuk proses selanjutnya seperti **Delete**.

The screenshot shows the 'Daftar Course' list view in Odoo. It has a search bar, 'Create' and 'Import' buttons, and filters. The table has columns for 'Name' and 'PHP'. The first row is 'PHP'. The left sidebar shows the 'Academic' app and 'Course' model. The top bar shows 'Academic', 'Apps', 'Settings', and the user 'Administrator (aca)'.

Gambar 35 Udah bisa lihat list Course yang udah tersimpan dan bisa diedit dan delete

By default, odoo udah nampilin object yang kita mau di list view dan form view dengan susunan standard, yaitu hanya beberapa field yang muncul di list view dan susunan field di form view masih apa adanya.

7.2 BIKIN TREE VIEW COURSE

Tampilan list view Course boleh kita modif supaya muncul semua field yang ada di class **Course**.

Caranya, bikin file baru dibawah addons `academic`, kasi nama `course.xml`. Disini nanti kita akan simpan semua definisi yang berkaitan dengan view Course. Salah satunya tampilan list view yang mau kita modif.

Isinya adalah...

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

    <record id="view_academic_course_tree" model="ir.ui.view">
      <field name="name">academic.course.tree</field>
      <field name="model">academic.course</field>
      <field name="type">tree</field>
      <field name="priority" eval="8"/>
      <field name="arch" type="xml">
        <tree string="course">
          <field name="name"/>
          <field name="description"/>
          <field name="responsible_id"/>
        </tree>
      </field>
    </record>

  </data>
</openerp>
```

Gambar 36 Definisi tree view Course

Untuk memodif tampilan list view suatu object, di odoo kita perlu buat definisi tree view di XML terhadap object tersebut.

Contoh di atas kita bikin definisi tree view untuk object `academic.course`, yaitu dengan cara membuat record baru pada object `ir.ui.view` yang terdefinisi pada atribut `model`.

Atribut `id` sama seperti elemen record XML yang sebelumnya, merupakan referensi bagi elemen XML lainnya.

Record ini secara internal berkaitan dengan tampilan suatu object tergantung dari definisi arsitekturnya nanti.

Record ini perlu field berikut.

Field `name` adalah nama internal tree view ini.

Field `model` adalah nama object model yang mau didefinisikan tampilan list tree view nya.

Field `arch` menentukan bagaimana object ini ditampilkan, apakah secara tree, form, calendar, dan lain sebagainya.

Disini kita buat definisi arsitektur dalam tree view, caranya dengan membuat definisi tag `tree` di dalam tag field `arch`.

Di dalam tag `tree`, kita tinggal panggil nama-nama field yang mau dimunculin di list tree view. Otomatis karena kita ngomongnya object course, berarti semua field yang ada di object tersebut bisa dimunculkan disini.

Contoh disini kita munculin field `name`, `description`, dan `responsible_id`.

Lanjut...

Tambahkan file `course.xml` pada file `__openerp__.py...`

```
{  
  "name": "Academic Information System Day",  
  "version": "1.0",  
  "depends": [  
    "
```

```

        "base",
    ],
    "author": "akhmad.daniel@gmail.com",
    "category": "Education",
    'website': 'http://www.vitraining.com',
    "description": """\
Academic Information System Day1
""",
    "data": [
        "menu.xml",
        "course.xml"
    ],
    "installable": True,
    "auto_install": False,
    "application": True,
}

```

Gambar 37 Include course.xml di __openerp__.py

Susunan file pada addons `academic` kita sejauh ini adalah seperti ini... ada tambahan file `course.xml`.

```

addons/academic
|-- __init__.py
|-- __openerp__.py
|-- course.py
|-- course.xml
`-- menu.xml

```

Lanjut...

Upgrade module, coba lihat tampilan list view Course yang baru...

Daftar Course

Search...

Filters

Group By

Favorites

1-

Create

Import

<input type="checkbox"/>	Name	Description	Responsible
<input type="checkbox"/>	PHP	kursus PHP	Administrator

Gambar 38 Course list udah muncul

okee... udah muncul field nya sesuai kemauan kita...

yang menarik disini, kolom `Responsible` udah langsung muncul nama user yang direlasikan ke Course.

7.3 BIKIN FORM VIEW COURSE

Lanjut, kita mau modif juga tampilan form view Course supaya lebih bagus, nggak standard kayak yang sekarang ada.

Edit lagi file `course.xml`.

Tambahi deklarasi form view untuk class Course seperti ini...

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

    <record id="view_academic_course_tree" model="ir.ui.view">
      <field name="name">academic.course.tree</field>
      <field name="model">academic.course</field>
      <field name="type">tree</field>
      <field name="priority" eval="8"/>
      <field name="arch" type="xml">
        <tree string="course">
          <field name="name"/>
          <field name="description"/>
          <field name="responsible_id"/>
        </tree>
      </field>
    </record>

    <record id="view_academic_course_form" model="ir.ui.view">
      <field name="name">academic.course.form</field>
      <field name="model">academic.course</field>
      <field name="type">form</field>
      <field name="priority" eval="8"/>
      <field name="arch" type="xml">
        <form string="Course Form">

          <sheet>
            <div class="oe_title">
              <h1>
                <field name="name"/>
              </h1>
            </div>
            <group>
              <field name="responsible_id"/>
            </group>

            <notebook>
              <page string="Description">
                <field name="description"/>
              </page>
            </notebook>

          </sheet>
        </form>
      </field>
    </record>

  </data>
</openerp>
```

Sama seperti tree view, definisi form view juga memakai record object model `ir.ui.view`, karena terkait dengan tampilan.

Bedanya disini adalah definisi field `arch` nya.

Disini dipake tag `form` yang ditaro didalam field `arch`, yang gunanya untuk mendefinisikan ulang tampilan form dari object yang sedang diomongin yaitu `academic.course`.

Pertama-tama atribut `string` tag ini yang gunanya untuk label judul form.

Lalu atribut `version` yang nentuin jenis versi tag form apakah versi 10 atau bukan. Versi 10 punya lebih banyak fitur misalnya boleh pasang tag HTML dan CSS didalamnya.

Didalam tag form ada tag `sheet` tempat kita naro semua elemen field yang ada pada object model di atas form.

Pertama kita taro field `name` yang udah kita format pake tag HTML `div` dan `h1...` supaya tampilan font nya muncul lebih besar.

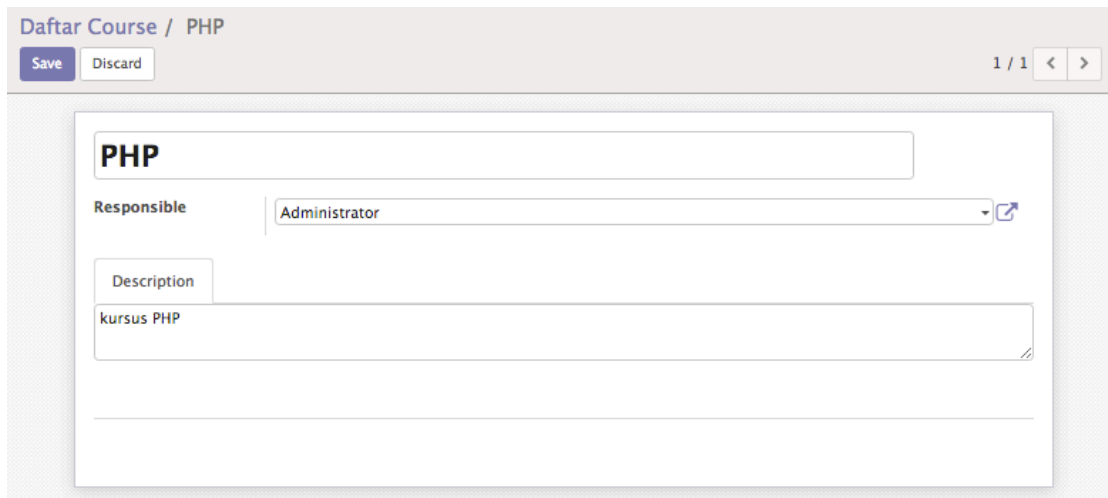
Lalu ada tag `group` yang gunanya untuk ngelompokin field-field dalam satu kelompok dan memunculkan label. Field yang di luar tag `group` nggak muncul labelnya.

Lanjut, ada tag `notebook` didalam tag `form`. Ini untuk membuat tempat untuk tab-tab yang bisa berisi field-field lain supaya tampilan lebih rapih dan terstruktur...

Didalam tag `notebook` dicantumin tag `page`. Setiap `page` akan jadi satu tab nantinya. Disini kita baru bikin satu tag `page` yang isinya field `description`.

Lanjut... restart server dan update module.

Lihat hasilnya...



Gambar 40 Form view Course

okee,... tampilan form view udah berubah, ada tab

Description dan nama field Course jadi besar font-nya.

Tapi tunggu... waktu di edit gimana caranya supaya label field name muncul? Hanya waktu edit aja, kalau view sih nggak apa...

Edit lagi file `course.xml`, tambahi ini...

```
<record id="view_academic_course_form" model="ir.ui.view">
  <field name="name">academic.course.form</field>
  <field name="model">academic.course</field>
  <field name="type">form</field>
  <field name="priority" eval="8"/>
  <field name="arch" type="xml">
    <form string="Course Form">

      <sheet>
        <div class="oe_title">
          <label for="name" class="oe_edit_only"
            string="Course Name"/>

          <h1>
            <field name="name"/>
          </h1>
        </div>
```

Gambar 41 Label Course muncul hanya pada saat edit

Disini kita tambahi tag HTML `label` untuk dan sebelum field `name` tapi dikasi class CSS khusus `oe_edit_only` yang membuat label itu hanya muncul waktu form edit.

Restart lagi server dan update module...

Harusnya label edit udah muncul.

Daftar Course / PHP

Save Discard 1 / 1

Course Name
PHP

Responsible
Administrator

Description
kursus PHP

Beres sementara untuk urusan Course, kita lanjut ke urusan object Session...

8 CLASS SESSION

8.1 MENU DAN ACTION WINDOW SESSION

Agar supaya bisa mengakses object Session kita perlu bikin dulu:

- menu **Session List** yang memanggil action window
- action window yang memanggil class Session
- class Session

Pertama kita buat dulu menu untuk Session. Edit file `menu.xml`.

Tambahi tag `menuitem` dibawah deklarasi menu Course.

```
<menuitem id="menu_academic_course"
  name="Course"
  parent="academic_1"
  action="action_course_list"
  sequence="20"/>

<menuitem id="menu_academic_session"
  name="Session"
  parent="academic_1"
  action="action_session_list"
  sequence="30"/>

</data>
</odoo>
```

Gambar 42 Menu Session

Intinya kita nambahi tag `menuitem` dengan attribut `parent` adalah menu Academic (id `academic_1`) ...

dan `action` adalah action window dengan id `action_session_list`.

Udah itu, buat action window yang dimaksud untuk Session List...

```
<odoo>
<data>

<record id="action_course_list" model="ir.actions.act_window">
  <field name="name">Daftar Course</field>
  <field name="res_model">academic.course</field>
  <field name="view_mode">tree,form</field>
  <field name="help" type="html">
```

```

        <p class="oe_view_nocontent_create">
            Click to add a Course
        </p>
        <p>klik tombol create untuk bikin Course baru</p>
    </field>
</record>

<record id="action_session_list" model="ir.actions.act_window">
    <field name="name">Daftar Session</field>
    <field name="res_model">academic.session</field>
    <field name="view_mode">tree,form</field>
    <field name="help" type="html">
        <p class="oe_view_nocontent_create">
            Click to add a Session
        </p>
        <p>klik tombol create untuk bikin Session baru</p>
    </field>
</record>

<menuitem id="academic_0"
    name="Academic"
    sequence="20"/>

```

Gambar 43 Action window Session

Baris ini harus diatas deklarasi `menuItem` Session List...

Ini sama aja seperti waktu kita mendeklarasikan action window untuk Course List, hanya saja disini kita kasi atribut `id = action_session_list` sesuai dengan yang di-refer pada menu Session List ...

dan link ke object `academic.session` sebagai sumber datanya yang didefinikan pada field `res_model`.

Susunan menu kita sejauh ini adalah...

```

Academic      (id academic_0)
  Academic    (id academic_1, parent academic_0)
    Course    (id academic_1_1, parent academic_1)
    Session   (id academic_1_2, parent academic_1)
    Attendee  (belum dibuat)
    Instructor (belum dibuat)

```

8.2 BIKIN CLASS SESSION

Action window session list perlu class object sebagai sumber datanya yaitu dengan nama `academic.session`.

Kita buat class itu sekarang.

Buat file baru dibawah folder addon `academic`. Kasi nama yang mewakili object yang ada di dalamnya, yaitu `session.py`.

Isi file ini adalah deklarasi class `session` dan semua atribut dan method yang ada pada class tersebut.

Sesuai ERD dan spesifikasi table, berikut ini syntax deklarasi class `session` pada file `session.py`.

```
from odoo import api, fields, models, _

class session(models.Model):
    _name = 'academic.session'

    name = fields.Char("Name", required=True)

    course_id = fields.Many2one(comodel_name="academic.course",
                                string="Course", required=False, )
    instructor_id = fields.Many2one(comodel_name="res.partner",
                                    string="Instructor", required=False, )
    start_date = fields.Date(string="Start Date", required=False, )
    duration = fields.Integer(string="Duration", required=False, )
    seats = fields.Integer(string="Seats", required=False, )
    active = fields.Boolean(string="Active", )
```

Gambar 44 Session Class

Sama seperti class Course sebelumnya, class `session` buatan kita merupakan turunan dari class `models.Model` (bawaan odoo/OpenObject). Otomatis dia bakalan udah punya sifat (method dan properties) sama seperti induknya. Contohnya langsung connect ke table `academic_session`, bisa cari data, insert, update, delete, dan sebagainya.

Setiap class harus diset property `_name` dan `_columns` nya.

Property `_name` gunanya sebagai referensi bagi semua modul addons yang ada di odoo. Jadi kalo ada class atau XML lain yang perlu akses ke class ini, dia harus panggil dengan panggilan `academic.session`.

Property `_columns` gunanya untuk definisiin kolom class, yaitu kolom `name`, `course_id`, dan `instructor_id`, `start_date`, `duration`, `seats`, dan `active` sesuai dengan definisi ERD.

Kalo udah siap, panggil file `session.py` dari file `__init__.py`, supaya class kita diimport oleh odoo waktu dijalankan. Caranya gini...

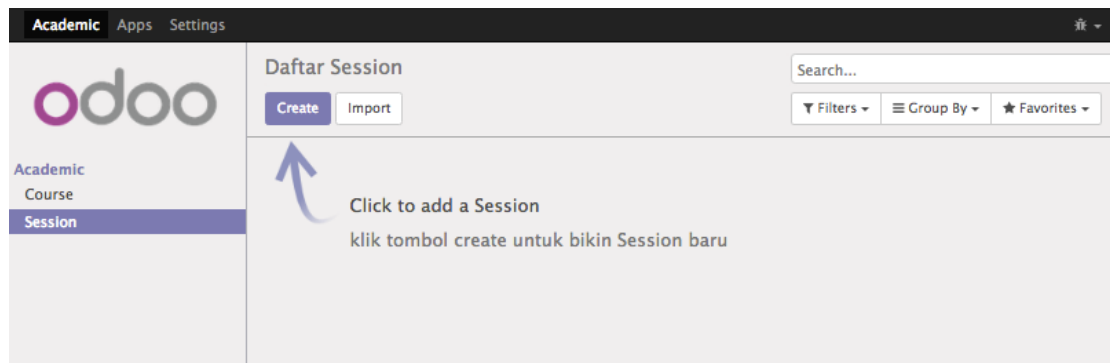
```
import session
import course
```

Gambar 45 Import Session

Susunan file pada addons `academic` kita sejauh ini adalah seperti ini... ada tambahan `session.py`.

```
addons/academic
|-- __init__.py
|-- __openerp__.py
|-- session.py
|-- course.py
`-- menu.xml
```

Yuuk,... restart odoo dan update module.. mari test tampilan aplikasi kita sekarang, jangan sampe ada error 😊 ...



Gambar 46 Menu Session muncul

okee sekarang udah nongol menu **Session** dan udah bisa klik tombol **Create...**

The screenshot shows the 'Daftar Session / New' form. At the top, there are 'Save' and 'Discard' buttons. The form fields are arranged in two columns. The left column contains 'Name' (text input), 'Duration' (text input with '0'), 'Course' (dropdown), and 'Start Date' (dropdown). The right column contains 'Instructir' (dropdown), 'Seats' (text input with '0'), and 'Active' (checkbox). The 'Name' field is currently selected.

Gambar 47 Bisa add session

klik tombol Save...

Daftar Session / Session 1

[Edit](#) [Create](#) [Action ▼](#) 1 / 1 <

Name	Session 1	Instructir	
Duration	0	Seats	0
Course	PHP	Active	<input type="checkbox"/>
Start Date			

Gambar 48 Bisa save dan view Session

klik List View icon atau menu **Session...**

Daftar Session

[Create](#) [Import](#) [Active is false ✕](#) Search...

[Filters ▼](#) [Group By ▼](#) [Favorites ▼](#)

✓ **Active is false**

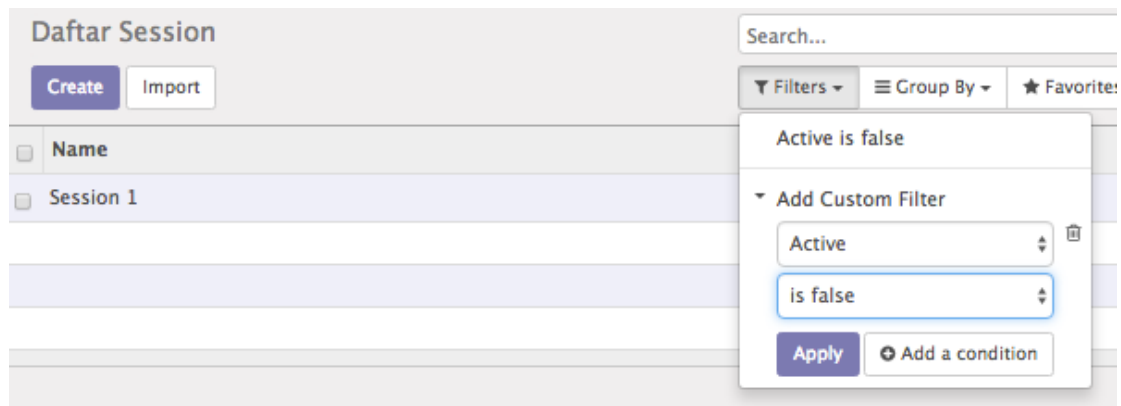
▸ Add Custom Filter

<input type="checkbox"/> Name
<input type="checkbox"/> Session 1

Gambar 49 Tree view Session

dan **Edit**, **Delete** Session... semua udah berfungsi.

Penting: Kolom **active** punya arti khusus di Odoo, yaitu jika isinya false, maka record nggak muncul di list dan form view, seolah-olah nggak ada data tersebut. Untuk membuka record yang nggak active, klik **Add Custom Filter**, pilih kolom **Active** is False lalu klik **Apply**.



8.3 MODIF LIST VIEW SESSION

Modif tampilan list view Session supaya muncul field-field yang kita perlukan.

Caranya, bikin file baru dibawah addons `academic`, kasi nama `session.xml`. Disini nanti kita akan simpan semua definisi yang berkaitan dengan view Session. Salah satunya tampilan list view yang mau kita modif.

Isinya adalah...

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

    <record id="view_academic_session_tree" model="ir.ui.view">
      <field name="name">academic.session.tree</field>
      <field name="model">academic.session</field>
      <field name="type">tree</field>
      <field name="priority" eval="8"/>
      <field name="arch" type="xml">
        <tree string="Session">
          <field name="name"/>
          <field name="instructor_id" />
          <field name="start_date" />
          <field name="duration" />
          <field name="seats" />
          <field name="active" />
        </tree>
      </field>
    </record>

  </data>
</openerp>
```

Gambar 50 Modif list view Session

Sama seperti waktu Course List, untuk memodif tampilan list view suatu object, di odoo kita perlu buat definisi tree view di XML terhadap object tersebut.

Disini kita bikin definisi tree view untuk object `academic.session`. Caranya dengan membuat record baru pada object `ir.ui.view` yang atribut `model` nya adalah object yang mau didefinisikan tree view nya.

Atribut `id` sama seperti elemen record XML yang sebelumnya, merupakan referensi bagi elemen XML lainnya.

Record ini perlu field berikut.

Field `name` adalah nama internal tree view ini.

Field `model` adalah nama object model yang mau didefinisikan tampilan list tree view nya.

Field `arch` menentukan bagaimana object ini ditampilkan, apakah secara tree, form, calendar, dan lain sebagainya.

Disini kita buat definisi arsitektur dalam tree view, caranya dengan membuat definisi tag `tree` di dalam tag field `arch`.

Di dalam tag `tree`, kita panggil nama-nama field yang mau dimunculin di list tree view. Otomatis karena kita sekarang ngomongnya object session, berarti semua field yang ada di object tersebut bisa dimunculkan disini.

Contoh disini kita munculin semua field yang ada pada object Session.

Lanjut...

Tambahkan file `session.xml` pada file `__openerp__.py...`

```
{
  "name": "Academic Information System Day",
  "version": "1.0",
  "depends": [
    "base",
  ],
  "author": "akhmad.daniel@gmail.com",
  "category": "Education",
  'website': 'http://www.vitraining.com',
  "description": """\n
Academic Information System Day1
""",
  "data": [
    "menu.xml",
    "course.xml",
    "session.xml",
  ],
  "installable": True,
  "auto_install": False,
  "application": True,
}
```

Gambar 51 Panggil session.xml dari __openerp__.py

Susunan file pada addons `academic` kita sejauh ini adalah seperti ini... ada tambahan file `session.xml`.

```
addons/academic
|-- __init__.py
|-- __openerp__.py
|-- course.py
|-- course.xml
|-- session.xml
`-- menu.xml
```

Lanjut...

Restart server, coba lihat tampilan list view Session yang baru...

Daftar Session

Create

Import

Search...

▼ Filters

≡ Group By

★ Favorites

1-1 / 1

<

>

<input type="checkbox"/>	Name	Instructir	Start Date	Duration	Seats	Active
<input type="checkbox"/>	Session 1	Administrator	02/19/2017	21	122	<input checked="" type="checkbox"/>

Gambar 52 List view Session udah lengkap

okee... udah muncul field nya sesuai kemauan kita...

Perhatikan kolom **Instructor** udah langsung muncul nama **Partner** yang direlasikan ke Session.

Kolom **Is Active** juga sudah muncul checkbox sesuai definisi di model class.

8.4 MODIF FORM VIEW

Sekarang kita modif form view Session supaya tampilan lebih terstruktur.

Edit lagi file `session.xml`.

Tambahi deklarasi form view untuk class Session seperti ini...

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

    <record id="view_academic_session_tree" model="ir.ui.view">
      <field name="name">academic.session.tree</field>
      <field name="model">academic.session</field>
      <field name="type">tree</field>
      <field name="priority" eval="8"/>
      <field name="arch" type="xml">
        <tree string="Session">
          <field name="name"/>
          <field name="course_id" />
          <field name="instructor_id" />
          <field name="start_date" />
          <field name="duration" />
          <field name="seats" />
          <field name="active" />
        </tree>
      </field>
    </record>

    <record id="view_academic_session_form" model="ir.ui.view">
      <field name="name">academic.session.form</field>
      <field name="model">academic.session</field>
      <field name="type">form</field>
      <field name="priority" eval="8"/>
      <field name="arch" type="xml">
        <form string="Session">
          <sheet>
            <div class="oe_title">
              <label for="name" class="oe_edit_only"
                string="Session Name"/>
              <h1><field name="name"/></h1>
            </div>
            <group>
              <group>
                <field name="course_id" />

```

```

        <field name="instructor_id" />
        <field name="start_date" />
    </group>
    <group>
        <field name="duration" />
        <field name="seats" />
        <field name="active" />
    </group>
</group>

</sheet>

</form>
</field>
</record>

</data>
</openerp>

```

Gambar 53 Modif form view Session

Sama seperti form view Course sebelumnya, definisi form view memakai record object model `ir.ui.view` dengan tag `form` yang ditaro didalam field `arch`, yang gunanya untuk mendefinisikan ulang tampilan form dari object yang sedang diomongin yaitu `academic.session`.

Pertama-tama atribut `string` tag ini yang gunanya untuk label judul form.

Lalu atribut `version` yang nentuin jenis versi tag form apakah versi 10 atau bukan. Versi 10 punya lebih banyak fitur misalnya boleh pasang tag HTML dan CSS didalamnya.

Didalam tag form ada tag `sheet` tempat kita naro semua elemen field yang ada pada object model di atas form.

Pertama kita taro field `name` yang udah kita format pake tag HTML `div` dan `h1`... supaya tampilan font nya muncul lebih besar. Lalu ada tag `label` HTML untuk menampilkan label Name hanya pada seat edit aja.

Lalu ada tag `group` yang gunanya untuk ngelompokin field-field dalam satu kelompok dan memunculkan label. Field yang di luar tag `group` nggak muncul labelnya.

Tapi disini group-nya dua level, ada group level satu yang terdiri dari 2 group yang masing-masing berisi field object session. Ini untuk membuat tampilan field terbagi menjadi 2 kolom terpisah.

Lanjut... restart server dan update module.

Lihat hasilnya...

Gambar 54 Form view Session

okee,... tampilan form view udah berubah, field udah terkelompok jadi 2 kolom dan nama field Session jadi besar font-nya.

8.5 BIKIN RELASI COURSE HAS MANY SESSION

Sesuai ERD sebelumnya, satu **Course** bisa punya banyak **Session**. Jadi kita perlu edit definisi class model **Course** supaya mendukung relasi ini.

Edit file `course.py`.

Tambahi field baru namanya `session_ids`.

```
from odoo import api, fields, models, _

class Course(models.Model):
    _name = 'academic.course'
    _rec_name = 'name'

    name = fields.Char("Name")
    description = fields.Text(string="Description", required=False, )
```

```

responsible_id = fields.Many2one(comodel_name="res.users",
                                string="Responsible")

session_ids = fields.One2many(comodel_name="academic.session",
                              inverse_name="course_id",
                              string="Sessions", required=False,
                              ondelete="cascade")

```

Gambar 55 Tambah field session_ids di Session class

Disini kita definisikan kolom baru dengan nama `session_ids`. Penamaan ini hanya konvesi aja, akhiran `_id` artinya dia merupakan kolom relasi dan penambahan huruf “s” sehingga menjadi `_ids` artinya relasi ke banyak record pada table lawan.

Kolom ini punya type khusus, `one2many`. Type kolom ini minta parameter sebagai berikut:

Parameter pertama comodel_name, adalah object model lawannya, yaitu yang boleh dimiliki lebih dari satu oleh object ini, disini `academic.session`. Artinya satu record di object `course` boleh punya banyak pasangan record di object `session`.

Parameter kedua inverse_name, adalah nama field pada object lawan, disini `course_id`. Lihat definisi kolom di object `session`, disana ada kolom namanya `course_id` yang menandakan bahwa dia dimiliki oleh suatu record di object `course`.

Parameter ketiga adalah label waktu ditampilkan di view.

Parameter keempat dan seterusnya sifatnya optional, disini kita set `ondelete="cascade"` artinya kalo record pada object `course` dihapus maka record-record terkait pada object `session` juga ikut terhapus.

Setelah dideklarasikan seperti diatas, maka object `COURSE` udah punya satu kolom tambahan dengan type `one2many` dan udah

bisa ditampilkan di view seperti halnya field-field regular lainnya.

8.6 MODIF VIEW FORM COURSE – TAMBAH SESSION

Untuk nampilin kolom tambahan pada object course, kita modif file `course.xml` yang merupakan kumpulan view untuk object ini.

Tambahkan baris ini pada form view Course...

```
<sheet>
  <div class="oe_title">
    <label for="name" class="oe_edit_only" string="Course Name"/>
    <h1>
      <field name="name"/>
    </h1>
  </div>
  <group>
    <field name="responsible_id"/>
  </group>

  <notebook>
    <page string="Description">
      <field name="description"/>
    </page>
    <page string="Sessions">
      <field name="session_ids">
        <tree string="Sessions">
          <field name="name"/>
          <field name="instructor_id" />
          <field name="start_date" />
          <field name="duration" />
          <field name="seats" />
          <field name="active" />
        </tree>
      </field>
    </page>
  </notebook>
</sheet>
```

Gambar 56 Tambah tab Session di Course form view

Disini kita modif form view Course, yaitu nambahin satu tag `page` baru di dalam tag `notebook` yang udah ada sebelumnya.

Didalam tag `page` kita masukkan tag `field` untuk kolom `session_ids` milik object `course`.

Didalam tag `field` kita pasang tag `tree`.

Disini tag `tree` langsung berkorelasi dengan object yang merupakan relasi dari kolom `session_ids` yaitu `session`.

Artinya kita bisa munculkan kolom-kolom yang ada pada object `session` di dalam tag `tree`. Disini kita munculkan semua kolom yang ada di `session`.

Okee... restart server dan update module.

Lihat hasilnya...

Daftar Course / PHP

Save Discard 1 / 1 < >

Course Name
PHP

Responsible
Administrator

Description Sessions

Name	Instructor	Start Date	Duration	Seats	Active	
Session 1	Administrator	02/19/2017		21	122	<input checked="" type="checkbox"/>

Add an item

Gambar 57 Tab Session muncul di Course form view

Sekarang form Course udah ada tambahan tab baru yang berisi Session yang terkait dengan Course tersebut.

Dan kalau di edit, maka kita bisa tambahkan Session baru langsung dari form Course, dengan klik link `Add new item`.

Gambar 58 Create Session dari Course form view

Form Session yang muncul disini sama persis dengan ketika kita add Session melalui menu Session.

Jangan lupa tick field **Is Active** disini agar record Session tampil.

Tapi Course boleh dikosongkan disini dan otomatis link dengan Course yang sedang di-edit sekarang.

9 CLASS ATTENDEE

9.1 MENU DAN ACTION WINDOW ATTENDEE

Agar supaya bisa mengakses object **Attendee** kita perlu bikin dulu:

- menu **Attendee List** yang memanggil action window
- action window yang memanggil class **Attendee**
- class **Attendee**

Pertama kita buat dulu menu untuk **Attendee**.

Edit file `menu.xml`.

Tambahi tag `menuitem` dibawah deklarasi menu **Course**.

```
<menuitem id="menu_academic_session"
  name="Session"
  parent="academic_1"
  action="action_session_list"
  sequence="30"/>

<menuitem id="menu_academic_attendee"
  name="Attendee"
  parent="academic_1"
  action="action_attendee_list"
  sequence="40"/>

</data>
</odoo>
```

Gambar 59 Tambah menu Attendee

Intinya kita nambahi tag `menuitem` dengan attribut `parent` adalah menu Academic (id `academic_1`) ...

dan `action` adalah action window dengan id `action_attendee_list`.

Udah itu, buat action window yang dimaksud untuk Attendee List...

```
<record id="action_session_list" model="ir.actions.act_window">
  <field name="name">Daftar Session</field>
  <field name="res_model">academic.session</field>
  <field name="view_mode">tree,form</field>
  <field name="help" type="html">
    <p class="oe_view_nocontent_create">
      Click to add a Session
    </p>
    <p>klik tombol create untuk bikin Session baru</p>
  </field>
</record>

<record id="action_attendee_list" model="ir.actions.act_window">
  <field name="name">Daftar Attendee</field>
  <field name="res_model">academic.attendee</field>
  <field name="view_mode">tree,form</field>
  <field name="help" type="html">
    <p class="oe_view_nocontent_create">
      Click to add a Attendee
    </p>
    <p>klik tombol create untuk bikin Attendee baru</p>
  </field>
</record>

<menuitem id="academic_0"
  name="Academic"
  sequence="20"/>
```

Gambar 60 Action window attendee

Baris ini harus diatas deklarasi `menuItem` Attendee List...

Ini sama aja seperti waktu kita mendeklarasikan action window untuk Session List, hanya saja disini kita kasi atribut `id = action_attendee_list` sesuai dengan yang di-refer pada menu Attendee List ...

dan link ke object `academic.attendee` sebagai sumber datanya yang didefinisikan pada field `res_model`.

Susunan menu kita sejauh ini adalah...

```
Academic      (id academic_0)
  Academic    (id academic_1, parent academic_0)
    Course    (id academic_1_1, parent academic_1)
    Session   (id academic_1_2, parent academic_1)
    Attendee  (id academic_1_3, parent academic_1)
    Instructor (belum dibuat)
```

9.2 BIKIN CLASS ATTENDEE

Action window session list perlu class object sebagai sumber datanya yaitu dengan nama `academic.attendee`.

Kita buat class itu sekarang.

Buat file baru dibawah folder addon `academic`. Kasi nama yang mewakili object yang ada di dalamnya, yaitu `attendee.py`.

Isi file ini adalah deklarasi class `attendee` dan semua atribut dan method yang ada pada class tersebut.

Sesuai ERD dan spesifikasi table, berikut ini syntax deklarasi class `attendee` pada file `attendee.py`.

```
from odoo import api, fields, models, _

class Attendee(models.Model):
    _name = 'academic.attendee'
    _rec_name = 'name'

    name = fields.Char("Name")
    session_id = fields.Many2one(comodel_name="academic.session",
                                string="Session", required=False, )
    partner_id = fields.Many2one(comodel_name="res.partner",
                                string="Partner", required=False, )
```

Gambar 61 Attendee Class

Sama seperti class Session sebelumnya, class `attendee` buatan kita merupakan turunan dari class `models.Model` (bawaan odoo/OpenObject). Otomatis dia juga bakal udah punya sifat

(method dan properties) sama seperti induknya. Contohnya langsung connect ke table `academic_attendee`, bisa cari data, insert, update, delete, dan sebagainya.

Setiap class harus diset property `_name` dan `_columns` nya.

Property `_name` gunanya sebagai referensi bagi semua modul addons yang ada di odoo. Jadi kalo ada class atau XML lain yang perlu akses ke class ini, dia harus panggil dengan panggilan `academic.attendee`.

Property `_columns` gunanya untuk definisiin kolom class, yaitu kolom `name`, `session_id`, dan `partner_id` sesuai dengan definisi ERD.

Kolom `name` bertype char panjang maximum 200 karakter.

Kolom `session_id` bertype `many2one` yaitu relasi ke object `academic.session`.

Kolom `partner_id` bertype `many2one` yaitu relasi ke object `res.partner` bawaan odoo.

Kalo udah siap, panggil file `attendee.py` dari file `__init__.py`, supaya class kita diimport oleh odoo waktu dijalankan. Caranya gini...

```
import session
import course
import attendee
```

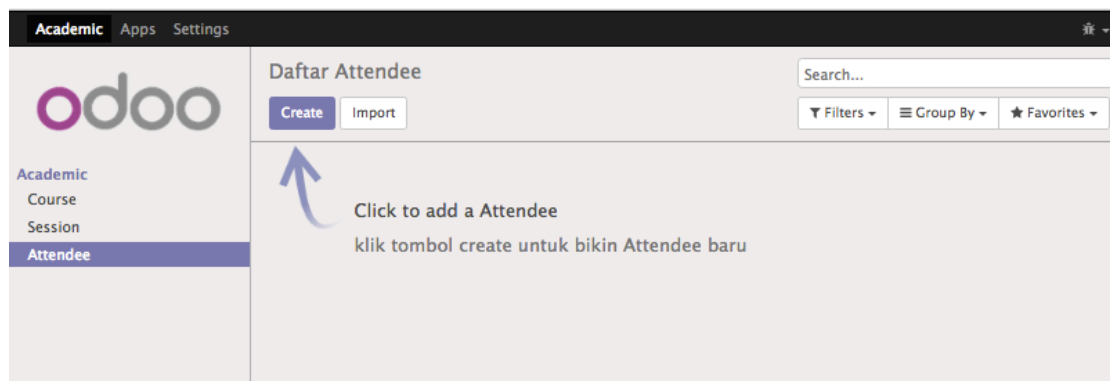
Gambar 62 Import Attendee

Susunan file pada addons `academic` kita sejauh ini adalah seperti ini... ada tambahan `attendee.py`.

```
addons/academic
|-- __init__.py
```

```
|-- __openerp___.py
|-- attendee.py
|-- session.py
|-- course.py
`-- menu.xml
```

Lanjut,... restart odoo dan update module.. mari test tampilan aplikasi kita sekarang, jangan sampe ada error 😊 ...



Gambar 63 Muncul Attendee menu

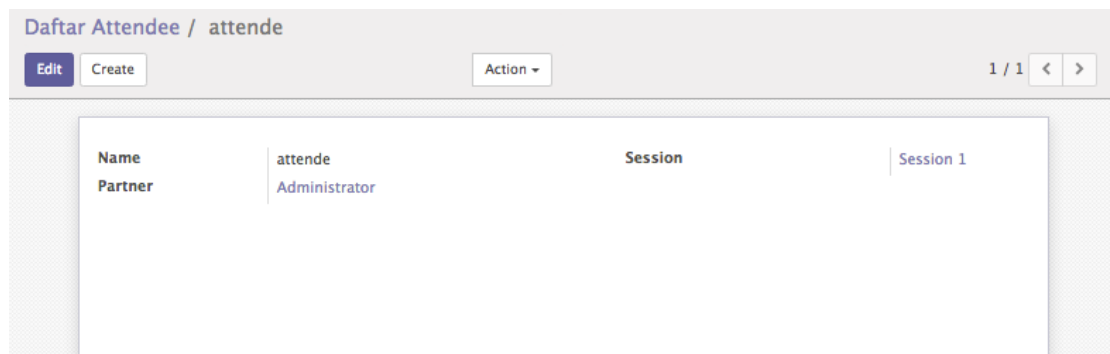
Okee , menu Attendee udah muncul, bisa klik tombol **Create**...

The screenshot shows the 'Daftar Attendee / New' form in the Odoo Academic interface. At the top, there are 'Save' and 'Discard' buttons. The form has three main fields: 'Name' with the value 'attende', 'Partner' which is a dropdown menu currently showing 'Administrator', and 'Session' which is a dropdown menu currently showing 'Session 1'. There are also small icons for adding new records next to the Partner and Session fields.

Gambar 64 Bisa Add Attendee

Input field-field sesuai definisi model, terutama field Partner dan Session yang berupa **many2one**.

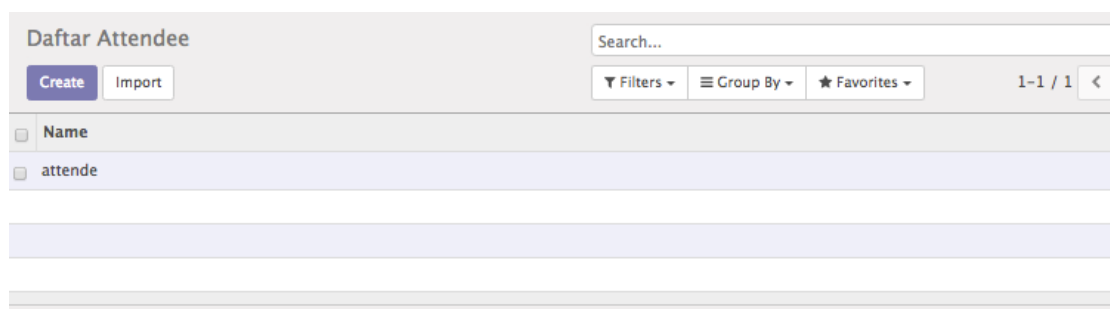
Bisa klik tombol **Save...**



Gambar 65 View Attendee

Ada link yang bisa diklik ke object yang terkait untuk semua field dengan type `many2one`.

Muncul daftar Attendee ketika di klik dari menu atau icon list view.



Gambar 66 List view Attendee

9.3 MODIF LIST VIEW ATTENDEE

Modif tampilan list view Attendee supaya muncul field-field yang kita perlukan.

Caranya, sama seperti yang lain, bikin file baru dibawah addons `academic`, kasi nama `attendee.xml`. Disini nanti kita akan simpan semua definisi yang berkaitan dengan view Attendee. Salah satunya tampilan list view yang mau kita modif.

Isinya adalah...

```

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

    <record id="view_academic_attendee_tree" model="ir.ui.view">
      <field name="name">academic.attendee.tree</field>
      <field name="model">academic.attendee</field>
      <field name="type">tree</field>
      <field name="priority" eval="8"/>
      <field name="arch" type="xml">
        <tree string="Attendee">
          <field name="name"/>
          <field name="session_id" />
          <field name="partner_id" />
        </tree>
      </field>
    </record>

  </data>
</openerp>

```

Gambar 67 Modif list view

Sama seperti waktu Session List, untuk memodif tampilan list view suatu object, di odoo kita perlu buat definisi tree view di XML terhadap object tersebut.

Disini kita bikin definisi tree view untuk object `academic.attendee`. Caranya dengan membuat record baru pada object `ir.ui.view` yang atribut `model` nya adalah object yang mau didefinisikan tree view nya.

Atribut `id` sama seperti elemen record XML yang sebelumnya, merupakan referensi bagi elemen XML lainnya.

Record ini perlu field berikut.

Field `name` adalah nama internal tree view ini.

Field `model` adalah nama object model yang mau didefinisikan tampilan list tree view nya.

Field `arch` menentukan bagaimana object ini ditampilkan, apakah secara tree, form, calendar, dan lain sebagainya.

Disini kita buat definisi arsitektur dalam tree view, caranya dengan membuat definisi tag `tree` di dalam tag field `arch`.

Di dalam tag `tree`, kita panggil nama-nama field yang mau dimunculin di list tree view. Otomatis karena kita sekarang ngomongnya object `attendee`, berarti semua field yang ada di object tersebut bisa dimunculkan disini.

Contoh disini kita munculin semua field yang ada pada object Attendee.

Lanjut...

Tambahkan file `attendee.xml` pada file `__openerp__.py`...

```
{
  "name": "Academic Information System Day",
  "version": "1.0",
  "depends": [
    "base",
  ],
  "author": "akhmad.daniel@gmail.com",
  "category": "Education",
  'website': 'http://www.vitraining.com',
  "description": """\
Academic Information System Day1
""",
  "data": [
    "menu.xml",
    "course.xml",
    "session.xml",
    "attendee.xml",
  ],
  "installable": True,
  "auto_install": False,
  "application": True,
}
```

Gambar 68 Panggil attendee.xml dari __openerp__.py

Susunan file pada addons `academic` kita sejauh ini adalah seperti ini... ada tambahan file `attendee.xml`.

`academic`

```

|-- __init__.py
|-- __openerp__.py
|-- attendee.py
|-- attendee.xml
|-- course.py
|-- course.xml
|-- menu.xml
|-- session.py
`-- session.xml

```

Lanjut...

Restart server, coba lihat tampilan list view Attendee yang baru...

Name	Session	Partner
attende	Session 1	Administrator

Gambar 69 List view Attendee udah lengkap

okee... udah muncul field nya sesuai kemauan kita...

perhatikan kolom **Partner** dan **Session** udah langsung muncul nama Partner dan Session yang direlasikan.

9.4 BIKIN RELASI SESSION HAS MANY ATTENDEE

Sesuai ERD sebelumnya, satu **Session** bisa punya banyak **Attendee**. Jadi kita perlu edit definisi class model **Session** supaya mendukung relasi ini.

Edit file `session.py`.

Tambahi field baru namanya `attendee_ids`.

```

from odoo import api, fields, models, _

class session(models.Model):
    _name = 'academic.session'

```

```

name = fields.Char("Name", required=True)

course_id = fields.Many2one(comodel_name="academic.course",
                           string="Course", required=False, )
instructor_id = fields.Many2one(comodel_name="res.partner",
                                string="Instructor", required=False, )
start_date = fields.Date(string="Start Date", required=False, )
duration = fields.Integer(string="Duration", required=False, )
seats = fields.Integer(string="Seats", required=False, )
active = fields.Boolean(string="Active", )

attendee_ids = fields.One2many(comodel_name="academic.attendee",
                               inverse_name="session_id",
                               string="Attendees",
                               required=False, )

```

Gambar 70 Tambah field attendee_ids di Session clas

Disini kita definisikan kolom baru dengan nama `attendee_ids`. Ingat konvensi sebelumnya, akhiran `_id` artinya dia merupakan kolom relasi dan penambahan huruf "s" sehingga menjadi `_ids` artinya relasi ke banyak record pada table lawan.

Sama seperti Course ke Session, kolom ini punya type `one2many` yang minta parameter sebagai berikut:

Parameter pertama adalah object model lawannya, yaitu yang boleh dimiliki lebih dari satu oleh object ini, disini `academic.attendee`. Artinya satu record di object `session` boleh punya banyak pasangan record di object `attendee`.

Parameter kedua adalah nama field pada object lawan, disini `session_id`. Lihat definisi kolom di object `attendee`, disana ada kolom namanya `session_id` yang menandakan bahwa dia dimiliki oleh suatu record di object `session`.

Parameter ketiga adalah label waktu ditampilkan di view.

Parameter keempat dan seterusnya sifatnya optional, disini kita set `ondelete="cascade"` artinya kalo record pada object `session` dihapus maka record-record terkait pada object `attendee` juga ikut terhapus.

Setelah dideklarasikan seperti diatas, maka object `attendee` udah punya satu kolom tambahan dengan type `one2many` dan udah bisa ditampilkan di view seperti halnya field-field regular lainnya.

9.5 MODIF FORM VIEW SESSION – TAMBAH ATTENDEE

Untuk nampilin kolom tambahan pada object session, kita modif file `session.xml` yang merupakan kumpulan view untuk object ini.

Tambahkan baris ini pada form view Session...

```
<group>
  <field name="duration" />
  <field name="seats" />
  <field name="active" />
</group>
</group>

<notebook>
  <page string="Attendees">
    <field name="attendee_ids">
      <tree string="Attendees">
        <field name="name" />
        <field name="partner_id" />
      </tree>
    </field>
  </page>
</notebook>

</sheet>
</form>
</field>
```

Gambar 71 Tambah tab Attendee di Session form view

Disini kita modif form view Session, yaitu nambahin tag `notebook` baru dan satu tag `page` di dalamnya.

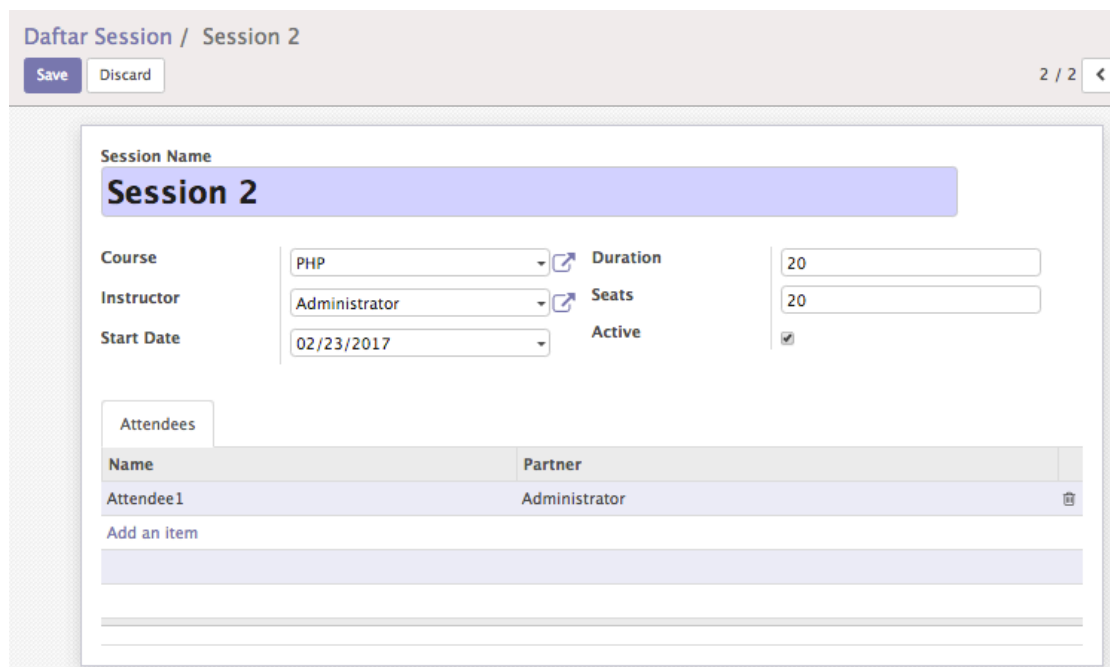
Dalam tag `page` kita keluarkan field `attendee_ids` milik object session.

Didalam tag field kita pasang tag tree, dimana disini tag `tree` langsung berkorelasi dengan object yang merupakan relasi dari kolom `attendee_ids` yaitu `attendee`.

Artinya kita bisa munculkan kolom-kolom yang ada pada object `attendee` di dalam tag `tree`. Disini kita munculkan semua kolom yang ada di object `attendee`.

Okee... restart server dan update module.

Lihat hasilnya...



Daftar Session / Session 2

Save Discard 2 / 2

Session Name
Session 2

Course: PHP Duration: 20
Instructor: Administrator Seats: 20
Start Date: 02/23/2017 Active: ☒

Attendees

Name	Partner
Attendee1	Administrator

Add an item

Gambar 72 Tab Attendee muncul di form view Session

Sekarang form Session udah ada tambahan tab baru yang berisi Attendee yang hadir dalam Session tersebut.

Dan kalau di edit, maka kita bisa tambahkan Attendee baru langsung dari form Session, dengan klik link `Add new item`.

The screenshot shows a web application interface with a modal window titled "Create: Attendees". The modal has a close button (X) in the top right corner. Inside the modal, there are three input fields: "Name" with the value "attendee2", "Partner" with a dropdown menu showing "Administrator", and "Session" which is empty. Below these fields are three buttons: "Save & Close", "Save & New", and "Discard". The background shows a sidebar with a logo and a top navigation bar with "Apps" and "Settings" tabs. The user is logged in as "Administrator (aca)".

Gambar 73 Add Attendee dari Session form view

Form Attendee yang muncul disini sama persis dengan ketika kita add Attendee melalui menu Attendee.

Tapi Session boleh dikosongkan disini dan otomatis link dengan Session yang sedang di-edit sekarang.

10 REKAP HARI 1

Wuih..... Udah banyak juga yang kita pelajari di hari pertama ini...

Berikut rekapnya

- mengetahui spesifikasi requirent aplikasi Academic Information System
- desain ERD
- mengetahui struktur Addons odoo
- mulai bikin addons Academic
- bikin Menu dan ActionWindow untuk Course, Session, dan Attendee
- bikin Class Course, Session, dan Attendee
- bikin relasi Course ke Session
- bikin relasi Session ke Attendee
- bikin tree view, form view
- bikin tab di form view