

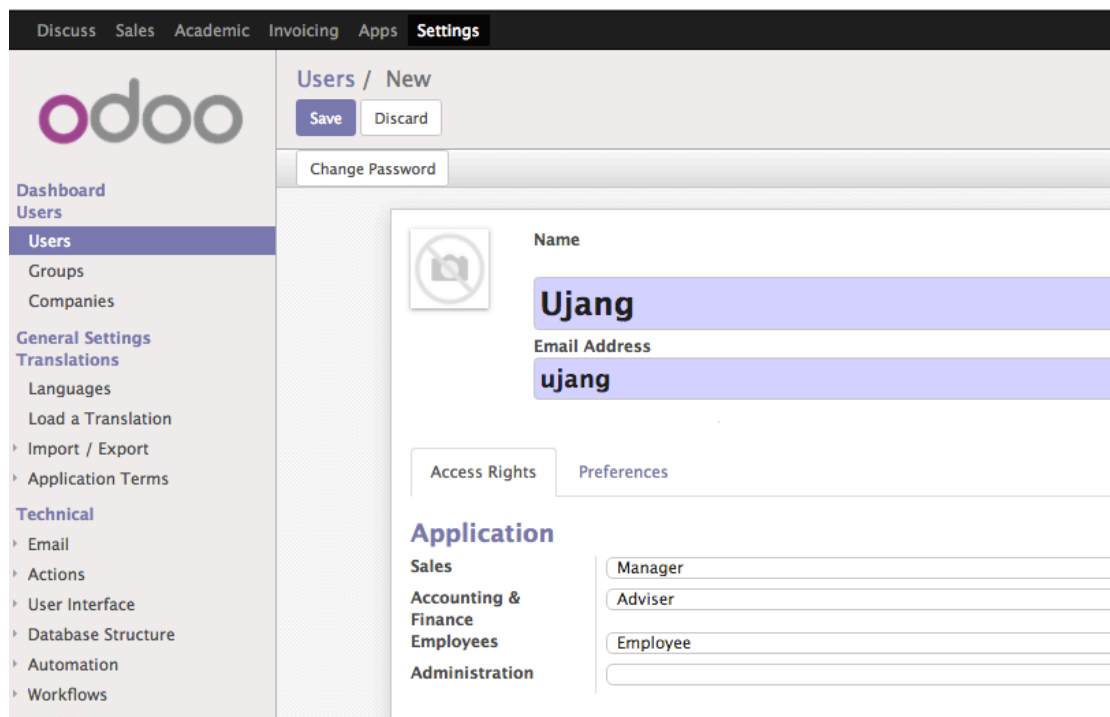
1 HARI 4: SECURITY

odoo udah nyediain sytem security yang cukup bagus, dimana kita bisa batasi akses menu terhadap group user, akses read, create, update, dan delete terhadap object, dan pembatasan akses terhadap record sesuai kriteria tertentu.

1.1 BIKIN GROUP LEWAT INTERFACE

Kita coba culu bikin kontrol akses group melalui user interface.

Melalui menu **Settings > Users > User**, create user baru “Ujang”.



Gambar 1 Create new User

Lalu create group baru yang hanya bisa read Session, Attendee, dan Course, dengan nama “Session Read” melalui menu **Settings > Users > Groups**.

Groups / New

Save Discard

Application Name Session Read

Portal ☐ Share Group ☐

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 type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	course
academic.attendee	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	attendee
academic.session	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	session

Add an Item

Gambar 2 Access Right group terhadap model

Kolom **Name** diisi dengan nama Group.


Pada tab **Access Right**, masukkan object session, attendee, dan course dengan Read Access.

Kolom name hanya informasi saja tapi harus diisi sesuai nama aksesnya misalnya session-read.

Edit user “Ujang” dan masukkan ke group “Session Read”.

Users / Ujang

Save Discard



Name

Ujang

Email Address

ujang

Access Rights

Preferences

Application

Sales

Accounting & Finance

Employees

Administration

Manager

Adviser

Employee

Other

Session Read ☒


Gambar 3 Set User ke Group

Kasi password "Ujang" melalui tombol **More > Change Password** supaya dia bisa login.

Users / Ujang

Save Discard

Change Password Send an Invitation Email



Name

Ujang

Email Address

ujang

Related Partner

Ujang

Access Rights

Preferences

Application

Sales

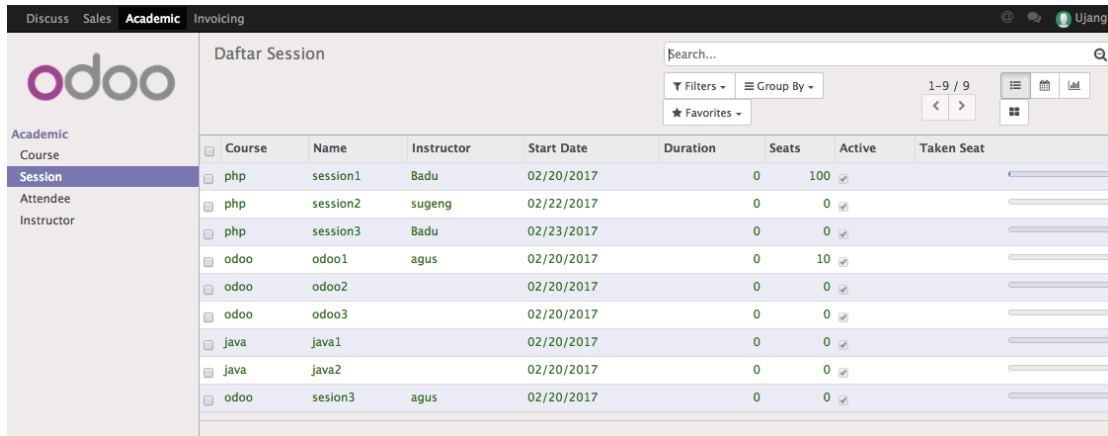
Accounting & Finance

Manager

Adviser

Gambar 4 Change User password

Oke sekarang bisa log in sebagai “Ujang” dan cek bagaimana akses ke Sesion.



Course	Name	Instructor	Start Date	Duration	Seats	Active	Taken Seat
php	session1	Badu	02/20/2017	0	100	<input checked="" type="checkbox"/>	<div></div>
php	session2	sugeng	02/22/2017	0	0	<input checked="" type="checkbox"/>	<div></div>
php	session3	Badu	02/23/2017	0	0	<input checked="" type="checkbox"/>	<div></div>
odoo	odoo1	agus	02/20/2017	0	10	<input checked="" type="checkbox"/>	<div></div>
odoo	odoo2		02/20/2017	0	0	<input checked="" type="checkbox"/>	<div></div>
odoo	odoo3		02/20/2017	0	0	<input checked="" type="checkbox"/>	<div></div>
java	java1		02/20/2017	0	0	<input checked="" type="checkbox"/>	<div></div>
java	java2		02/20/2017	0	0	<input checked="" type="checkbox"/>	<div></div>
odoo	session3	agus	02/20/2017	0	0	<input checked="" type="checkbox"/>	<div></div>

Gambar 5 Access user terhadap object sudah terbatas

Oke,... dia bisa lihat (read) Session, Course, dan Attendee... tapi nggak ada tombol **Create**, **Delete**, dan **Edit**, karena kita nggak kasi akses create, delete, dan update.

1.2 BIKIN GROUP LEWAT XML

Sekarang kita bikin definisi group melalui XML agar bisa disertakan pada module.

Group yang akan dibuat adalah “Academic / Manager” dan “Academic / User”.

Bikin folder `security` di bawah module, bikin file baru namanya `group.xml`, isinya seperti ini.

```
<openerp>
  <data noupdate="0">
    <record id="group_manager" model="res.groups">
      <field name="name">Academic / Manager</field>
    </record>
```

```

        <record id="group_user" model="res.groups">
            <field name="name">Academic / User</field>
        </record>
    </data>
</openerp>

```

Gambar 6 Bikin group via XML

Struktur file di folder addons sejauh ini... ada tambahan folder `security` dan file `group.xml`.

```

academic
|-- __init__.py
|-- __openerp__.py
|-- attendee.py
|-- attendee.xml
|-- course.py
|-- course.xml
|-- menu.xml
|-- partner.py
|-- partner.xml
|-- security
|   |-- group.xml
|-- session.py
|-- session.xml
|-- workflow.xml

```

Tambahin di `__openerp__.py`

```

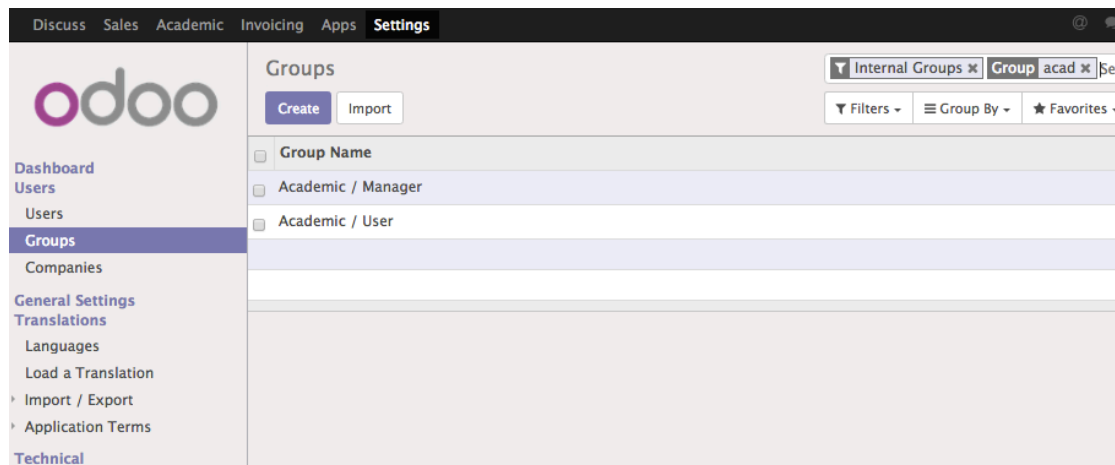
{
    "name": "Academic Information System Day 4",
    "version": "1.0",
    "depends": [
        "base",
        "account",
        "sale",
    ],
    "author": "akhmad.daniel@gmail.com",
    "category": "Education",
    'website': 'http://www.vitraining.com',
    "description": """\
Academic Information System Day 4
""",
    "data": [
        "menu.xml",
        "course.xml",
        "session.xml",
        "attendee.xml",
        "partner.xml",
        "workflow.xml",
        "security/group.xml",
    ],
    "installable": True,
    "auto_install": False,
    "application": True,
}

```

Gambar 7 Panggil security/group.xml dari __openerp__.py

Restart odoo dan update module.

Hasilnya lihat di **Settings > Users > Groups**.. search Adademic.



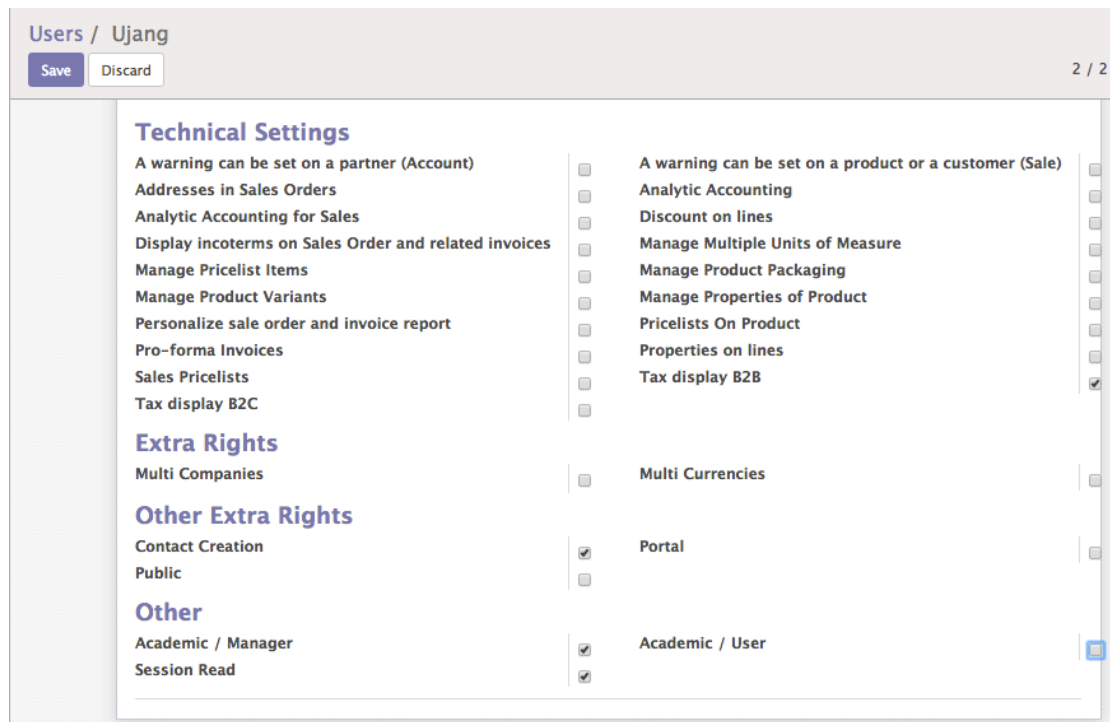
Gambar 8 Group list view

1.3 MASUKKAN USER KE GROUP

Setelah ini kita udah bisa masukkan User kedalam group Academic / Manager maupun User.

Caranya, masuk ke **Settings > Users > Users**. Pilih user atau create baru.

Klik tab **Access Rights** nya user. Disitu kita bisa tick group di bawah kategori **Other: Academic / Manager** atau **Academic / User**.



Gambar 9 Masukkan user ke group Academic/Manager

1.4 IMPORT CSV ACCESS RIGHT

Access right group yang dibuat di module biasanya kita import melalui file CSV. Pada file ini kita definisikan akses read, write, creation dan delete terhadap objek Course, Session dan Attendees untuk group Academic / Manager.

Bikin file baru CSV dengan nama `ir.model.access.csv` dibawah folder `security`.

Buka di OpenOffice supaya lebih enak editnya... isinya seperti berikut....

	A	B	C	D	E	F	G	H
1	id	name	model_id:id	group_id:id	perm_read	perm_write	perm_create	perm_unlink
2	course_manager	course_manager	model_academic_course	group_manager	1	1	1	1
3	session_manager	session_manager	model_academic_session	group_manager	1	1	1	1
4	attendee_manage	attendee_manager	model_academic_attendee	group_manager	1	1	1	1
5	course_user	course_user	model_academic_course	group_user	1	1	1	0
6	session_user	session_user	model_academic_session	group_user	1	1	1	0
7	attendee_user	attendee_user	model_academic_attendee	group_user	1	1	1	0
8	course_all	course_all	model_academic_course		1	0	0	0
9	session_all	session_all	model_academic_session		1	0	0	0
10	attendee_all	attendee_all	model_academic_attendee		1	0	0	0
11								

Gambar 10 File CSV access right

Isi file CSV secara lengkap...

```
"id","name","model_id:id","group_id:id","perm_read","perm_write","perm_create","perm_unlink"
"course_manager","course_manager","model_academic_course","group_manager",1,1,1,1
"session_manager","session_manager","model_academic_session","group_manager",1,1,1,1
"attendee_manage","attendee_manager","model_academic_attendee","group_manager",1,1,1,1
"course_user","course_user","model_academic_course","group_user",1,1,1,0
"session_user","session_user","model_academic_session","group_user",1,1,1,0
"attendee_user","attendee_user","model_academic_attendee","group_user",1,1,1,0
"course_all","course_all","model_academic_course",,1,0,0,0
"session_all","session_all","model_academic_session",,1,0,0,0
"attendee_all","attendee_all","model_academic_attendee",,1,0,0,0
```

File CSV ini terdiri dari beberapa kolom sesuai ketentuan odoo yaitu:

Kolom `id`, adalah nama reference XML ID

Kolom `name`, adalah nama XML record

Kolom `model_id:id`, nama model yang mau dikasi hak aksesnya kepada suatu group dengan format `model_<module>_<classname>`.

Kolom `group_id:id`, nama group XML ID yang udah diinsert melalui `group.xml` yaitu group yang diberi hak akses terhadap module. Kalau dikosongin berarti berlaku untuk semua group yang ada

Kolom `perm_read`, akses read boleh (1) atau nggak (0)

Kolom `perm_write`, akses update boleh (1) atau nggak (0)

Kolom `perm_create`, akses bikin record baru boleh (1) atau nggak (0)

Kolom `perm_unlink`, akses delete record boleh (1) atau nggak (0)

Lanjut, tambahi di `__openerp__.py`

```
{
  "name": "Academic Information System Day 4",
  "version": "1.0",
  "depends": [
    "base",
    "account",
    "sale",
  ],
  "author": "akhmad.daniel@gmail.com",
  "category": "Education",
  'website': 'http://www.vitraining.com',
  "description": """\
Academic Information System Day 4
""",
  "data": [
    "menu.xml",
    "course.xml",
    "session.xml",
    "attendee.xml",
    "partner.xml",
    "workflow.xml",
    "security/group.xml",
    "security/ir.model.access.csv",
  ],
  "installable": True,
  "auto_install": False,
  "application": True,
}
```

Gambar 11 Panggil security/ir.model.access.csv dari __openerp__.py

Sejauh ini struktur folder addons kita adalah...

```
academic
|-- __init__.py
|-- __openerp__.py
|-- attendee.py
|-- attendee.xml
|-- course.py
```

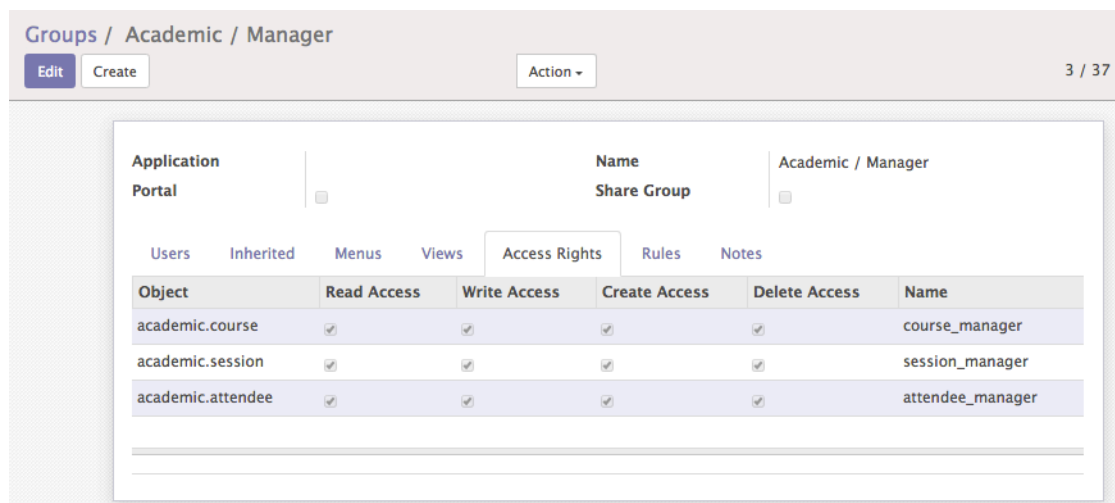
```

|-- course.xml
|-- menu.xml
|-- partner.py
|-- partner.xml
|-- security
|   |-- group.xml
|   |-- ir.model.access.csv
|-- session.py
|-- session.xml
`-- workflow.xml

```

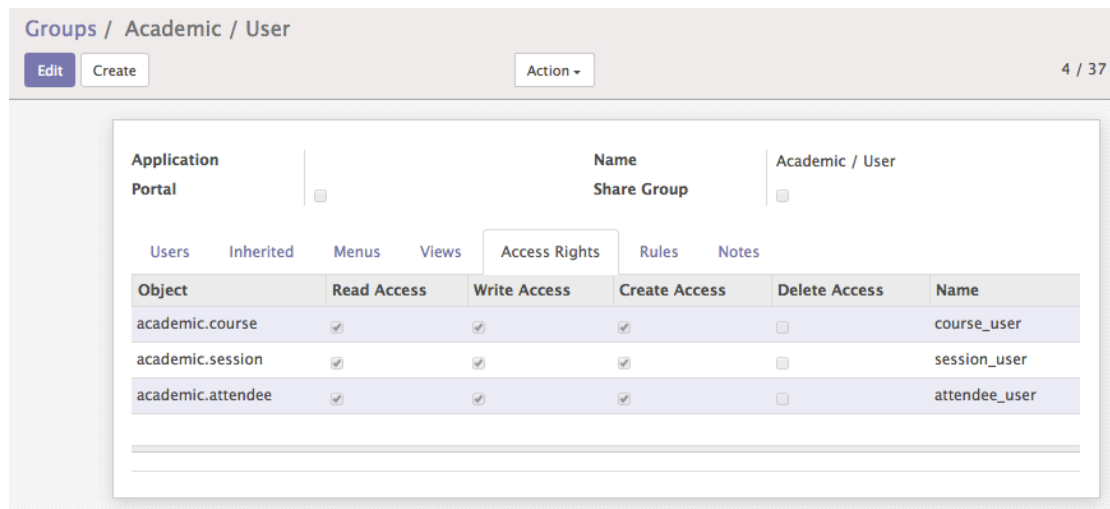
Restart odoo dan update module.

Hasilnya lihat di **Settings > Users > Groups**.. search Adademic. Klik Group Manager, klik tab **Access Rights**. Isinya harus sesuai dengan yang didefinisikan pada file CSV.



Gambar 12 Access right group Manager

Klik group User, klik tab Access Right. Isinya harus sesuai dengan yang didefinisikan pada file CSV.



Gambar 13 Access right group User

1.5 RECORD RULES

Pada awalnya diatas, semua Group Manager bisa edit dan delete Course. Kita mau hanya yang responsible aja yang bisa delete dan update. Untuk itu perlu dibuat Record Rules.

Record rule dipakai untuk memberi access rights hanya kepada beberapa records suatu model oleh suatu group. Misalnya data Customer, semua group Sales boleh akses read, create, write, dan delete. Tapi kemudian kita batasi hanya Sales User yang berada pada Area yang sama dengan Customer yang boleh read, write, dan delete. Sales User pada Area yang berbeda nggak bisa lihat Customer itu.

Record Rule merupakan record dari model `ir.rule`, dan dikaitkan ke model, beberapa groups (many2many field), permissions yang terkait, dan suatu kriteria domain yang menentukan records yang mana aja access rights diperbolehkan untuk group tersebut.

Sekarang kita tambahkan Record Rule untuk model Course pada group `OpenAcademy / Manager`, yang membatasi akses “write” and “unlink” / delete hanya kepada siapa user yang

responsible atas Course tersebut. Jika Course belum ada responsible-nya, semua user pada group itu boleh mengupdate dan delete.

Buka lagi file `security/groups.xml`, tambahi rule record seperti ini...

```
<odoo>
  <data noupdate="0">
    <record id="group_manager" model="res.groups">
      <field name="name">Academic / Manager</field>
    </record>
    <record id="group_user" model="res.groups">
      <field name="name">Academic / User</field>
    </record>

    <record id="only_responsible_can_modify" model="ir.rule">
      <field name="name">Cuman penanggung jawab yang
        bisa edit Course</field>
      <field name="model_id" ref="model_academic_course"/>
      <field name="groups" eval="[(4, ref('group_manager'))]"/>
      <field name="perm_read" eval="0"/>
      <field name="perm_write" eval="1"/>
      <field name="perm_create" eval="0"/>
      <field name="perm_unlink" eval="1"/>
      <field name="domain_force">['!', ('responsible_id','=',False),
        ('responsible_id','=',user.id)]</field>
    </record>
  </data>
</odoo>
```

Gambar 14 Bikin Record Rule dari XML

Disini kita tambahi record untuk object `ir.rule`. Field record ini adalah:

Field `name` untuk label nama record rule.

Field `model_id` adalah reference ke nama model dalam format `model_<module>_class`, contohnya `model_academic_course`.

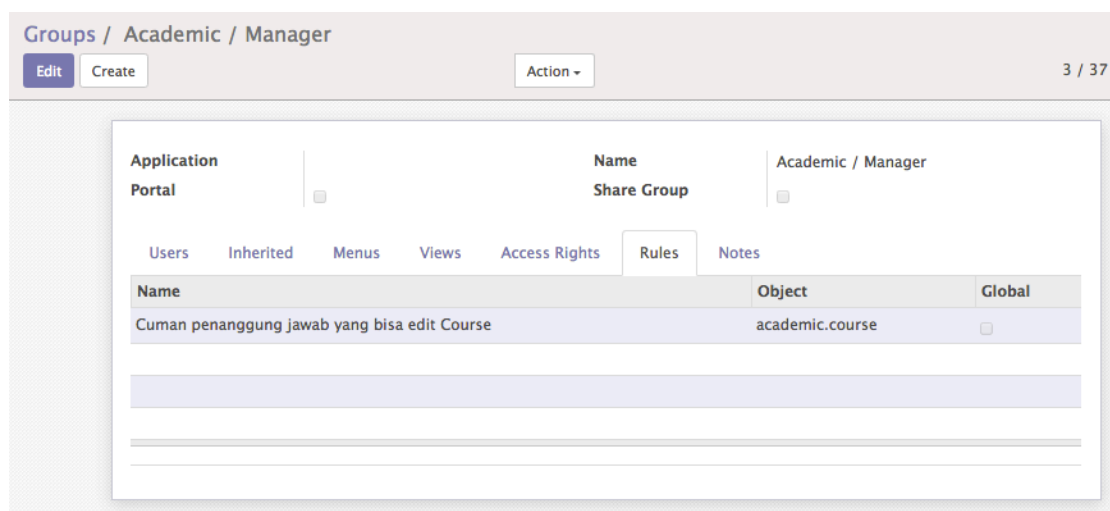
Field `groups` adalah relasi many2many ke groups, dimana penginputannya harus menggunakan atribut eval yang me-refer ke id group manager yang ditulis pada file XML `group.xml` sebelumnya, seperti ini :

```
[ (4, ref('academic.group_manager')) ]
```

Field `perm_read`, `perm_write`, `perm_create`, dan `perm_unlink` menentukan apakah boleh (1) atau tidak (0) untuk read, write (update), create, dan unlink (delete).

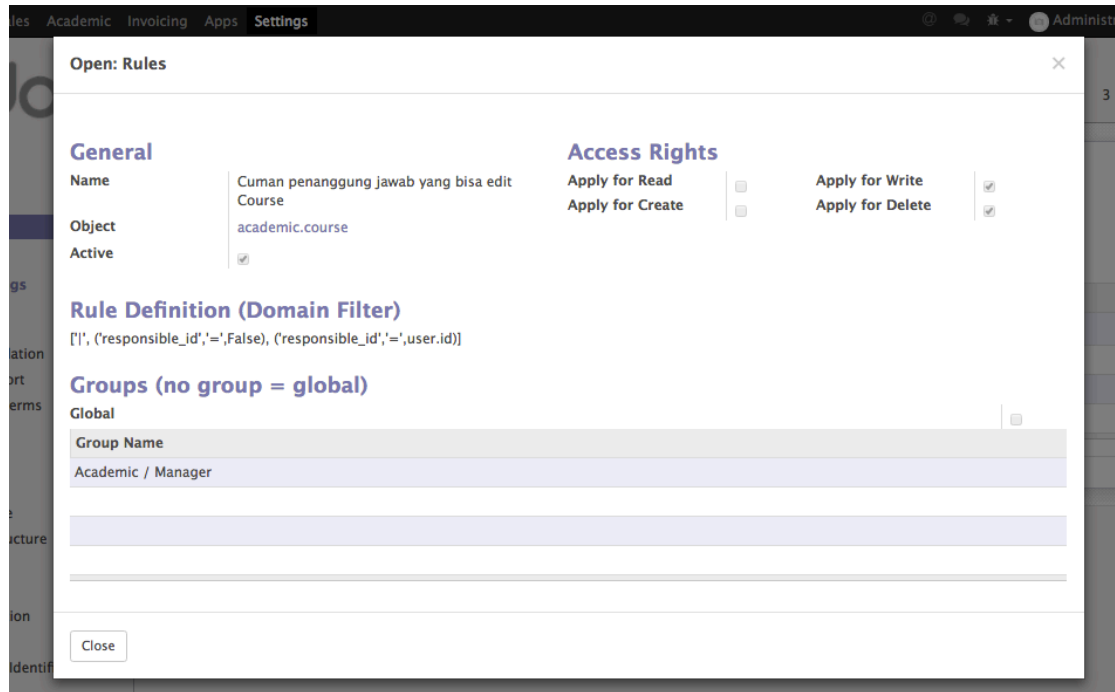
Field `domain_force` isinya filter domain yang membatasi record mana yang boleh diakses hanya oleh group yang terdaftar pada field `groups`. Jika domain ini nggak terpenuhi maka record ini nggak bisa diakses.

Restart odoo dan update module, hasilnya...



Gambar 15 Ada record rules di group Academic/Manager

Klik detail Record rule, muncul Access Rights, Groups, dan Domain Filter sesuai file XML.



Gambar 16 Isi Record rules

Sekarang, coba kita login dengan user yang bukan responsible dari suatu Course, lalu coba update dan delete...

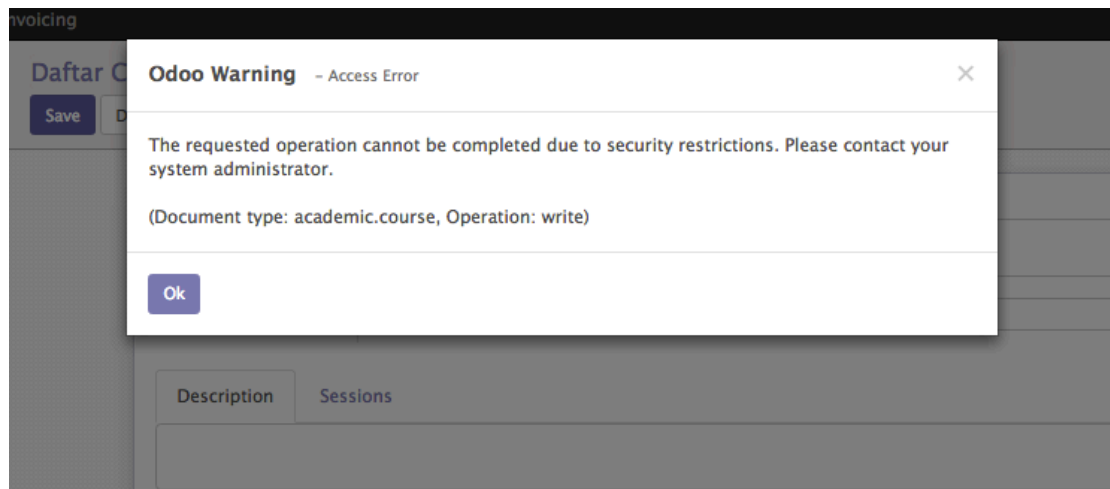
Misalkan kondisinya sekarang kayak gini..

Daftar Course			Search...
<div>Create Import</div>			<div> <div>Filters</div> <div>Group By</div> <div>Favorites</div> </div>
<input type="checkbox"/>	Name	Description	Responsible
<input type="checkbox"/>	PHP		Agus
<input type="checkbox"/>	Java		Agus
<input type="checkbox"/>	Odoo		Badu
<input type="checkbox"/>	Dot Net		Badu

Gambar 17 Responsible user masing-masing Course

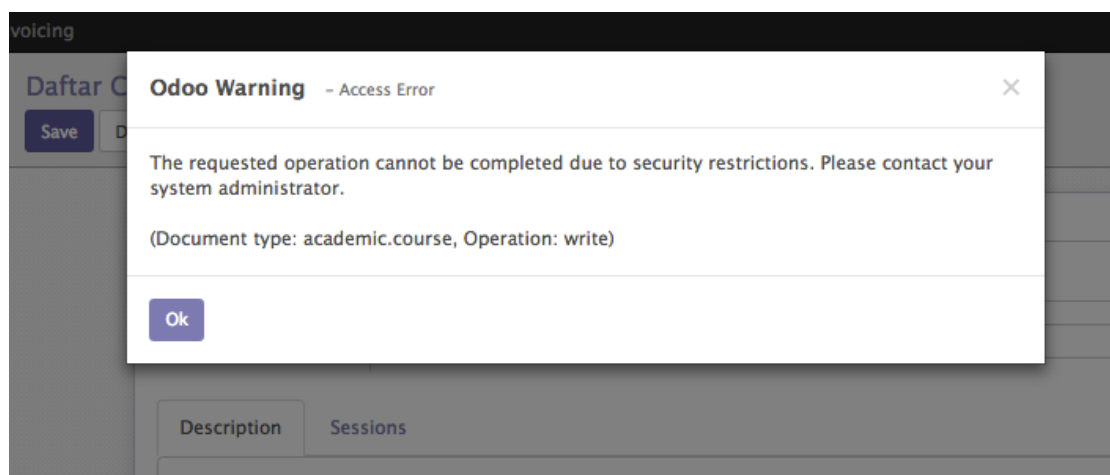
User **agus** dan **badu** udah diset sebagai group **Academic / Manager**.

Apakah **agus** bisa edit / delete Course-nya **badu**, Odoo dan Dotnet?



Gambar 18 Agus nggak bisa edit course-nya Badu

Apakah badu bisa edit / delete Course-nya agus, PHP dan Java?



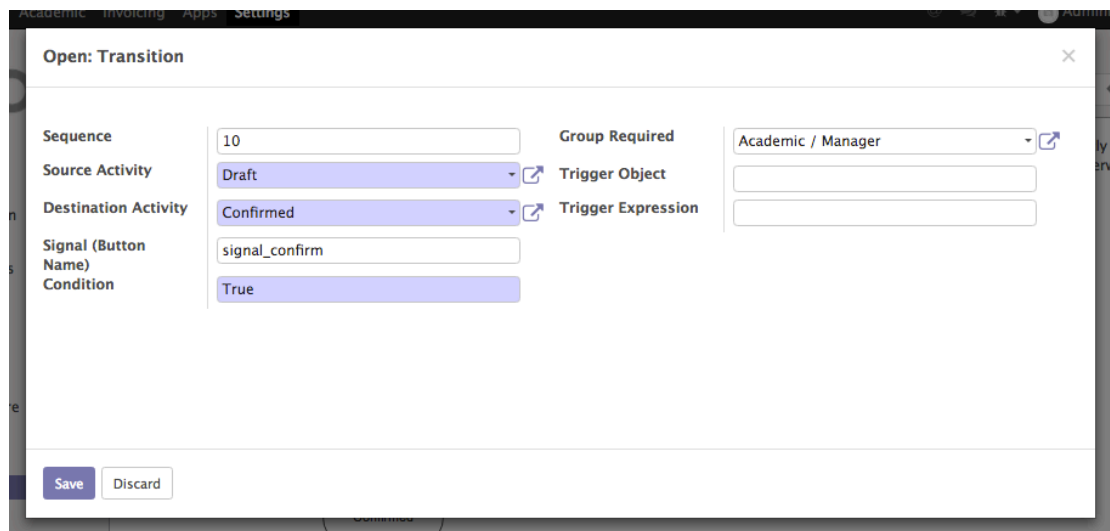
Gambar 19 Badu nggak bisa delete Course-nya Agus

1.6 HUBUNGAN GROUP WORKFLOW

Hubungan workflow dengan object adalah meng-update field `state`, jadi untuk bisa menjalankan workflow, user harus punya akses write terhadap object tersebut.

Coba kurangi access Write terhadap group User. Otomatis group User tidak bisa menjalankan workflow. Kalau dia klik tombol workflow akan terjadi error Access Denied.

Pada workflow action (panah) boleh dipilih nama group yang bisa men-trigger signal ini misalnya group Manager. Sehingga kalau tombol workflow diklik oleh group User, tidak terjadi error karena signal tidak ter trigger.



Gambar 20 Set Group yang bisa trigger signal workflow

Lalu, bagaimana meng-hide tombol sesuai group yang login?
Untuk ini kita perlu tambah atribut `groups` pada tombol supaya hanya muncul kalau user yang login tergabung pada group yang ditentukan disitu.

Edit file `session.xml`.

Tambahkan atribut `groups` pada button dengan nama-nama group yang boleh melihat :

```
<button string="Confirm" type="workflow"
        name="signal_confirm"
        groups="academic.group_manager"
        states="draft" />
<button string="Mark as Done" type="workflow"
        name="signal_done"
        groups="academic.group_manager"
        states="confirmed" />
<button string="Reset to draft" type="workflow"
        name="signal_draft"
        groups="academic.group_manager"
        states="confirmed,done" />

<field name="state" widget="statusbar" />
</header>
```


Gambar 21 Modif tombol supaya muncul sesuai group

Akibatnya, hanya user dalam group Academic/ Manager yang bisa melihat tombol-tombol workflow.

2 WIZARD

Di odoo ada yang namanya wizard, gunanya untuk memudahkan kita dalam mengisi data atau memilih parameter suatu proses berikutnya.

Misalnya untuk report kita bisa pilih terlebih dahulu parameter report misalnya periode, account, tahun, dan lain-lain. Ketika Submit maka akan diproses report sesuai dengan parameter yang dipilih pada wizard.

Wizard bisa juga untuk memudahkan kita dalam men-entry data. Misalnya disini kita buat wizard untuk menambahkan Attendee ke suatu Session tapi nggak satu-per-satu. Jadi kita bisa input banyak Attendee sekaligus kedalam suatu Session.

2.1 DEFINISIKAN CLASS WIZARD

Bikin directory baru namanya **wizard** dibawah folder addons
academic.

Bikin file baru `create_attendee.py` dibawah folder `wizard`.

Isinya ada 2 class, yaitu `CreateAttendeeWizard` dan `AttendeeWizard`.

[illegible]

Gambar 22 Bikin class wizard

Class `CreateAttendeeWizard` adalah class wizardnya itu sendiri, yang gunanya untuk membentuk window pop up wizard, dan terdiri dari kolom-kolom sesuai definisi pada class itu.

Class ini mirip banget seperti class object odoo lainnya, tapi disini kita pakai `models.TransientModel` sebagai parent class nya. Class `models.TransientModel` mirip seperti `models.Model` hanya saja datanya nggak tersimpan ke table database, tapi di memory.

Class ini punya 2 kolom yaitu `session_id` (pilihan Session yang Attendee nya mau di input), dan `attendee_ids` yaitu data calon Partner yang akan mengikuti Session yang dipilih.

Class berikutnya, `AttendeeWizard` adalah untuk menampung data Partner sementara yang dipilih pada wizard yang nantinya akan dimasukkan sebagai attendee session.

Ini diperlukan karena Partner yang dipilih disini belum masuk ke class Attendee yang di database, tapi masih ditampung dulu sementara di memory. Relasinya ke wizard class adalah many2one. Artinya satu wizard bisa punya banyak AttendeeWizard. Class ini juga punya relasi many2one ke Partner.

Lanjut, bikin file baru namanya `__init__.py` dibawah directory `wizard`:

```
import create_attendee
```

Gambar 23 Import class wizard

Struktur directory addons kita sejauh ini adalah...

```
academic
|-- __init__.py
|-- __openerp__.py
```

```

|-- attendee.py
|-- attendee.xml
|-- course.py
|-- course.xml
|-- menu.xml
|-- partner.py
|-- partner.xml
|-- security
|   |-- group.xml
|   `-- ir.model.access.csv
|-- session.py
|-- session.xml
|-- wizard
|   |-- __init__.py
|   `-- create_attendee.py
`-- workflow.xml

```

Lanjut, modif file `__init__.py` yang ada di bawah folder `academic`, tambahi `import wizard` artinya kita mengimport satu folder `wizard` beserta isinya sekaligus.

```
import wizard
```

Gambar 24 Import folder wizard

2.2 BIKIN MENU UNTUK WIZARD

Pertama-tama kita bikin menu item dan action yang diperlukan untuk membuka wizard. Buat file baru di dalam directory `wizard`, file XML dengan nama `create_attendee_view.xml` yang berisi menu item dan action window `create_attendee_wizard_action`:

```

<openerp>
  <data>
    <record model="ir.actions.act_window"
            id="create_attendee_wizard_action">
      <field name="name">Add attendee</field>
      <field name="res_model">academic.create.attendee.wizard</field>
      <field name="view_type">form</field>
      <field name="view_mode">form</field>
      <field name="target">new</field>
    </record>

    <menuitem name="Add Attendee" parent="academic_1"
              id="create_attendee_wizard_menu"
              sequence="40"
              action="create_attendee_wizard_action"/>
  </data>
</openerp>

```

```
</data>
</openerp>
```

Gambar 25 Menu dan action window wizard

Wizard dijalankan melalui record `ir.actions.act_window` alias action window, tapi ditambahi field `target` bernilai `new`. Field ini gunanya untuk membuka wizard view pada popup window. Field `res_model` isinya nama model wizard yang akan dibuka.

Action window diatas di-trigger lewat menu item yang parent nya adalah main menu `academic_1`.

Struktur file addons kita harus jadi seperti ini

```
academic/
|-- __init__.py
|-- __openerp__.py
|-- attendee.py
|-- attendee.xml
|-- course.py
|-- course.xml
|-- 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
```

Lanjut, update `__openerp__.py` yang berada di bawah folder `academic`:

```
{
  "name": "Academic Information System Day 4",
  "version": "1.0",
  "depends": [
    "base",
    "account",
    "sale",
  ],
  "author": "akhmad.daniel@gmail.com",
  "category": "Education",
  'website': 'http://www.vitraining.com',
```

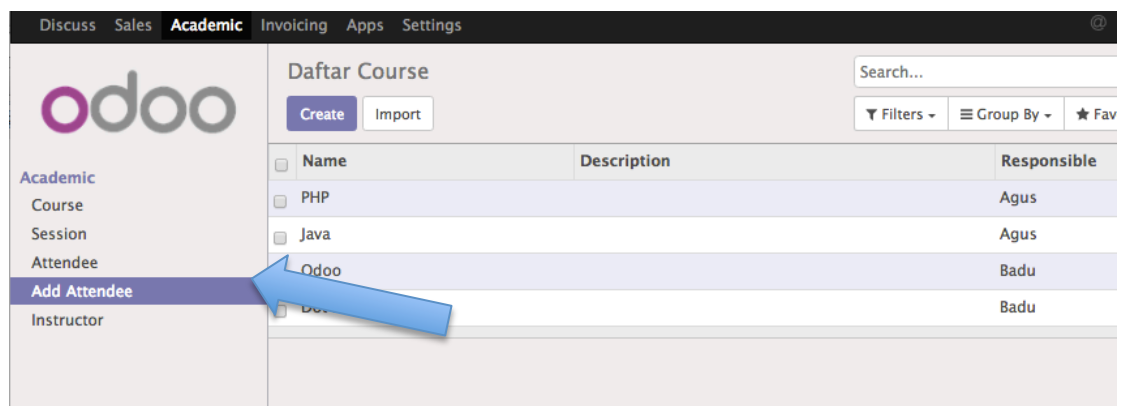
```

    "description": """\
Academic Information System Day 4
* Add security groups
* Add security ir.model.access.csv
* Add wizard
""",
    "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",
    ],
    "installable": True,
    "auto_install": False,
    "application": True,
}

```

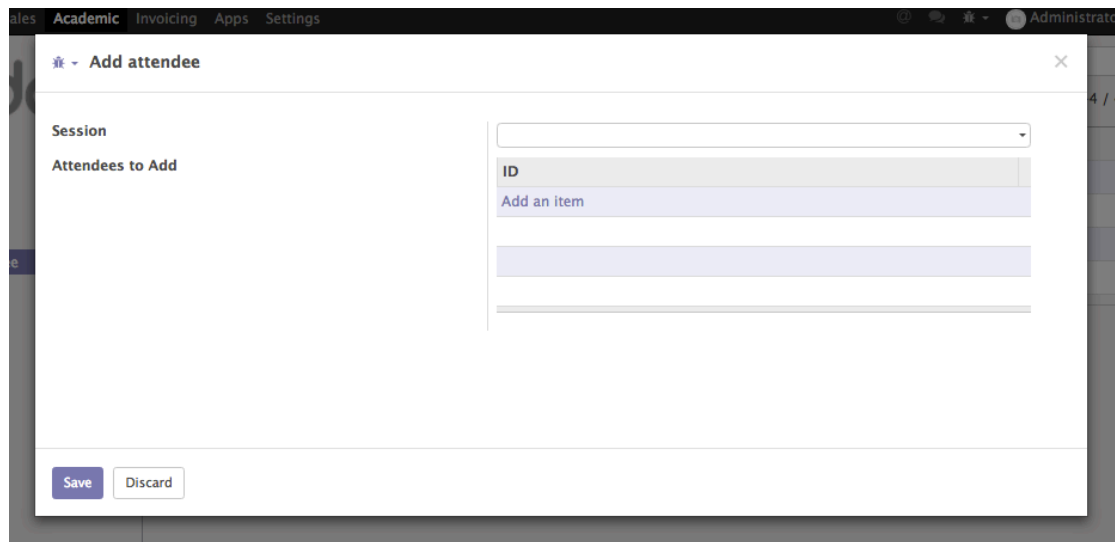
Gambar 26 Panggil wizard/create_attendee_view.xml dari __openerp__.py

Restart odoo dan update module, hasilnya...



Gambar 27 Menu Add attendee muncul

Ada menu baru, **Add Attendee**, waktu di-click menu tersebut akan muncul pop up box seperti ini...



Gambar 28 Wizard pop up muncul

Okee, menu dan wizard udah muncul waktu menu di-klik.

Tapi tampilannya masih belum betul.. kita modif abis ini...

2.3 MODIF FORM VIEW WIZARD

Edit file `create_attendee_view.xml`.

Tambahi record form view supaya nampilin semua fields class wizard. Lalu tambahi dua button pada form view yaitu untuk submit dan cancel wizard.

```
<record model="ir.ui.view" id="create_attendee_form_view">
  <field name="name">academic.create.attendee.wizard.form</field>
  <field name="model">academic.create.attendee.wizard</field>
  <field name="type">form</field>
  <field name="arch" type="xml">
    <form string="Add attendee" version="7.0">
      <group>
        <field name="session_id"/>
        <field name="attendee_ids" mode="tree">
          <tree string="Attendees"
            editable="bottom">
            <field name="partner_id"/>
          </tree>
        </field>
      </group>
      <footer>
        <button type="special"
          special="cancel"
          string="Cancel"
          icon="fa-cancel"/>

        <button type="object"
```

```

        name="action_add_attendee"
        string="Add attendees"
        icon="fa-ok"
        confirm="Are you sure you want to add those attendees?"
    />
</footer>
</form>
</field>
</record>

```

Gambar 29 Modif form view wizard dan tambah button

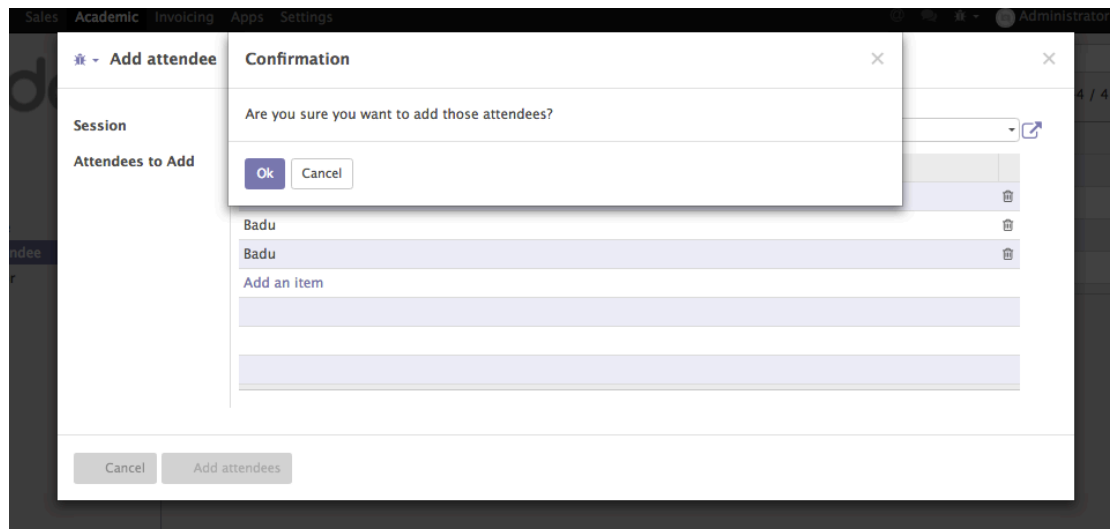
Letakkan dibawah definisi menuitem yang udah dibuat sebelumnya...

Restart odoo dan update module, hasilnya...

Gambar 30 Form view wizard udah lengkap

Semua field yang kita butuhkan udah muncul, termasuk tombol submit dan cancel.

Waktu click tombol sbmit Add Attendees...



Gambar 31 Konfirmasi submit wizard

Udah ada respon, tapi belum ada efek apa-apa karena kita belum bikin fungsi untuk nangkap action dari wizard... yaitu `action_add_attendee` sesuai dengan attribute `name` button yang ber-type object.

2.4 BIKIN METHOD UNTUK MEMPROSES DATA WIZARD

Edit file `create_attendee.py`.

Buat method baru, namanya `action_add_attendee` di dalam class, seperti ini.

```
class CreateAttendeeWizard(models.TransientModel):
    _name = 'academic.create.attendee.wizard'

    session_id = fields.Many2One(comodel_name="academic.session",
                                string="Session", required=False, )
    attendee_ids = fields.One2many(comodel_name="academic.attendee.wizard",
                                  inverse_name="wizard_id",
                                  string="Attendees to Add",
                                  required=False, )

    # create method
    @api.multi
    def action_add_attendee(self):
        self.ensure_one()
        session = self.session_id
        att_data = [{ 'partner_id': att.partner_id.id}]
```

```

        for att in self.attendee_ids]
    session.attendee_ids = [(0, 0, data) for data in att_data]

    return {'type': 'ir.actions.act_window_close'}

```

Gambar 32 Method proses add attendee

Pada method ini ada parameter bawaan odoo yaitu `self` yang berisi object record wizard. Pertama kita pastikan bahwa `self` hanya berisi 1 record dengan:

```
self.ensure_one()
```

Dari variable record wizard `self` tersebut, kita bisa ambil data **session_id** dan **attendee_ids** yang dipilih pada wizard, yaitu `self.session_id` dan **self.attendee_ids**.

Attribut **self.attendee_ids** berisi array list **partner_id**, dan inilah yang perlu kita masukkan ke database melalui object **academic.attendee**.

Lanjut, bentuk array list `att_data` melalui list comprehension

```

att_data = [{'partner_id': att.partner_id.id}
            for att in self.attendee_ids]

```

artinya, pada setiap elemen yang ada pada list `self.attendee_ids` simpan ke local variable `att`, lalu bentuk menjadi dictionary `{'partner_id': att.partner_id.id}`.

Hasil list comprehension diatas, `att` akan menjadi array list of dictionary, misalnya:

```
[ { 'partner_id' : 3 } , {'partner_id':2} , {'partner_id':20} ]
```

udah itu, kita jalankan penulisan field `attendee_ids` pada pada object session untuk mengupdate field `attendee_ids` nya.

```
session.attendee_ids = [(0, 0, data) for data in att_data]
```

Untuk mengupdate field one2many membutuhkan format data khusus yang disebut one2many commands. Dalam hal ini kita mengupdate field `attendee_ids` dalam rangka untuk nambahin record, jadi perlu one2many command insert.

Disini lagi-lagi kita gunakan list comprehension untuk membentuk array list dengan format data one2many commands:

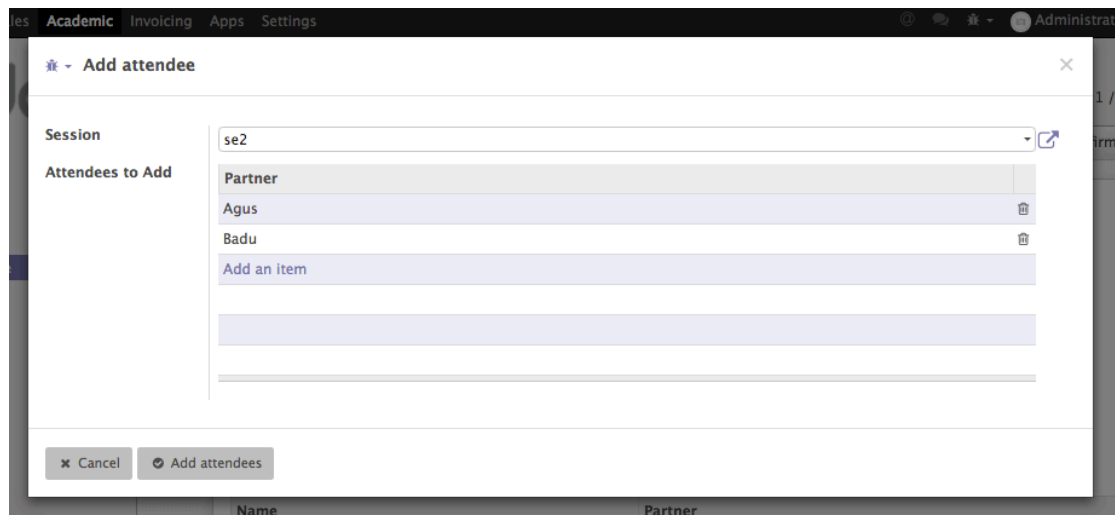
```
[
    (0,0, {'partner_id': 3} ) ,
    (0,0, {'partner_id': 2} ) ,
    (0,0, {'partner_id': 20} )
]
```

Sumber data yang kita looping pada list comprehension adalah `att_data` yang udah disiapkan di atas sebelumnya.

Setiap element `att_data` kita simpan kedalam variable local data, yang kemudian dibentuk menjadi tuple `(0,0,data)` sehingga terbentuklah array list seperti yang diinginkan di atas.

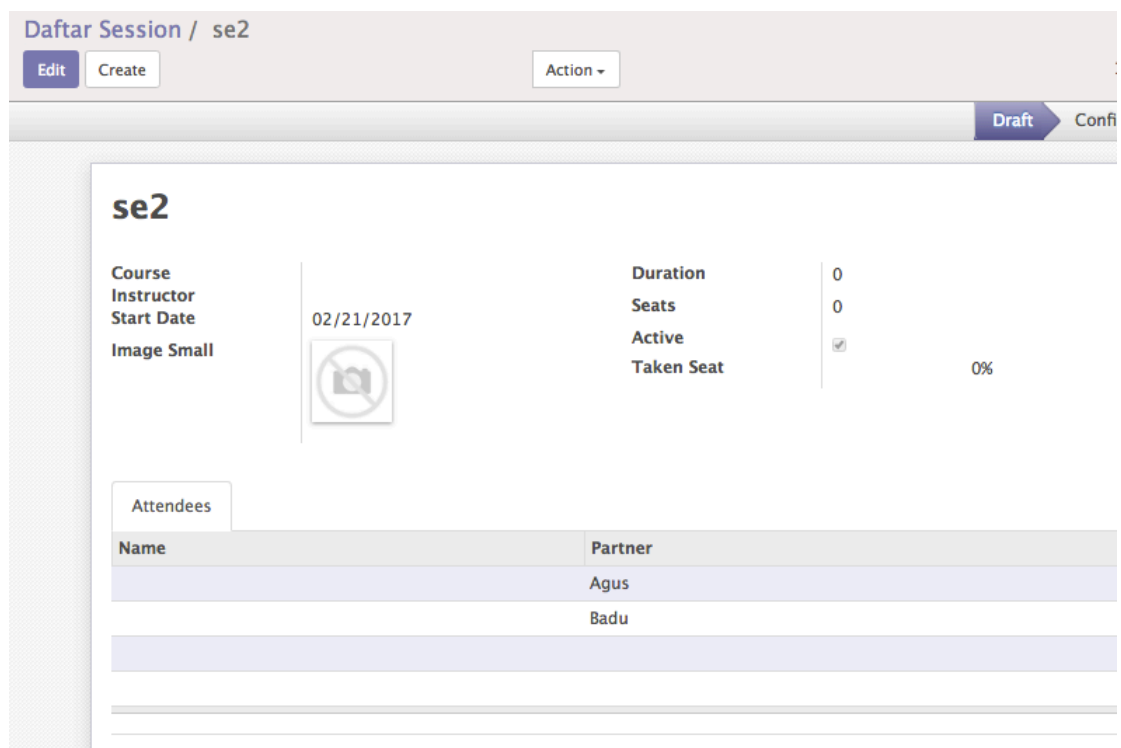
Setelah terbentuk dengan rapih maka boleh dijadikan parameter untuk nilai field one2many `attendee_ids` pada session object.

Update module dan jalankan wizard.



Gambar 33 Add attendees dari wizard

Setelah klik tombol Add attendees, dan konfirmasi maka attendees akan masuk ke session yang dipilih.



Gambar 34 Hasil add attendee melalui wizard

2.5 BIKIN MENU CONTEXT

Ada cara lain untuk membuka wizard yaitu melalui record `ir.actions.act_window` seperti diatas, tapi ditambahi field `src_model`.

Field ini menentukan pada konteks object manakah action ini tersedia. Menu Wizard akan muncul pada contextual actions modelyaitu di bawah tombol More pada form view. Karena ada hal internal yang perlu dilakukan khusus untuk ini, action context kita diklarasikan melalui XML tag `act_window`.

Disini kita akan mengkaitkan wizard dengan menu context session model. Kita akan menggunakan argumen "context" ntuk mendefinisikan session yang sedang dibuka sebagai nilai default untuk field `session_id` di wizard.

2.5.1 EDIT FILE `WIZARD/CREATE_ATTENDEE_VIEW.XML`.

Tambahi XML element yang dibutuhkan untuk membuat action context.

```
<act_window id="session_create_attendee_wizard"
  name="Add Attendees"
  src_model="academic.session"
  res_model="academic.create.attendee.wizard"
  view_mode="form"
  target="new"
  key2="client_action_multi"/>

</data>
</openerp>
```

Gambar 35 Action window untuk context menu

Field `src_model` menentukan pada object manakan action context ini akan muncul di form view nya.

Field `res_model` menentukan nama lengkap dari object tersebut.

Field `view_mode` menentukan mode view, yaitu form view.

Field `target` isinya new agar muncul pop up window.

Field `key2` menentukan bagaimana data context akan berisi apakah multi id atau single id.

2.5.2 EDIT FILE WIZARD/CREATE_ATTENDEE.PY

Pada class `CreateAttendeeWizard`, definisikan default value untuk field `session_id` yang nilainya diambil dari form view session yang sedang dibuka melalui variable context.

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

class CreateAttendeeWizard(models.TransientModel):
    _name = 'academic.create.attendee.wizard'

    def _get_active_session(self):
        context = self.env.context
        if context.get('active_model') == 'academic.session':
            return context.get('active_id', False)
        return False

    session_id = fields.Many2one(comodel_name="academic.session",
                                string="Session", required=False,
                                default=_get_active_session)
    attendee_ids = fields.One2many(comodel_name="academic.attendee.wizard",
                                   inverse_name="wizard_id",
                                   string="Attendees to Add",
                                   required=False, )
```

Gambar 36 Cara ambil active session

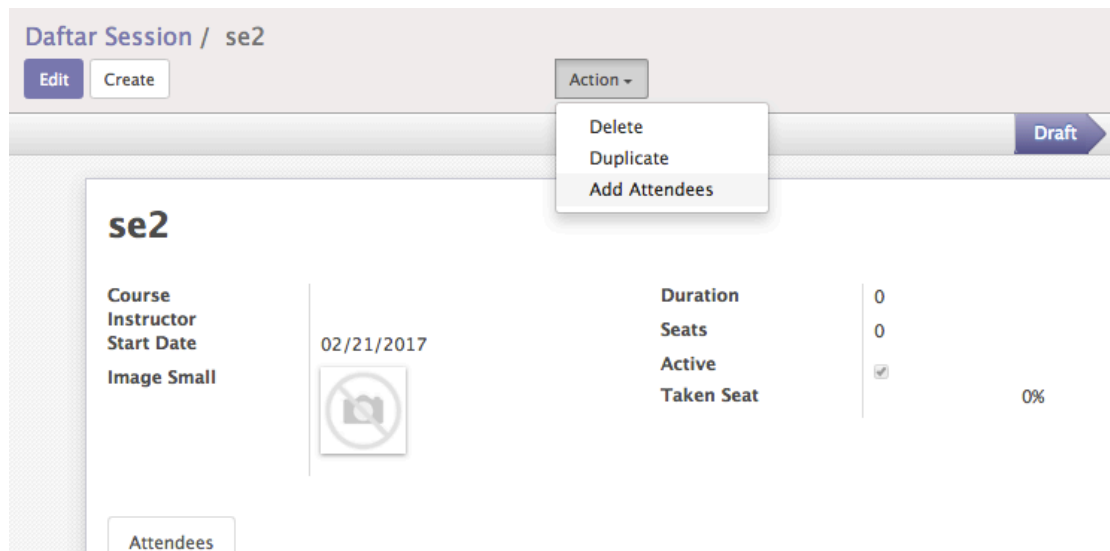
Atribut `defaults` untuk field `session_id` kita definisikan melalui method `_get_active_session()`.

Pada method ini kita membaca variable context yang datang dari action menu. Di dalam context itu tersedia key **active_model** dan **active_id** disamping key lainnya yang belum kita perlukan sekarang.

Disini kita cuma cek apakah key **active_model** adalah betul **academic.session**. jika ya, maka ambil key **active_id** dan jadikan return value method ini sehingga masuk nantinya sebagai default value untuk field `session_id`.

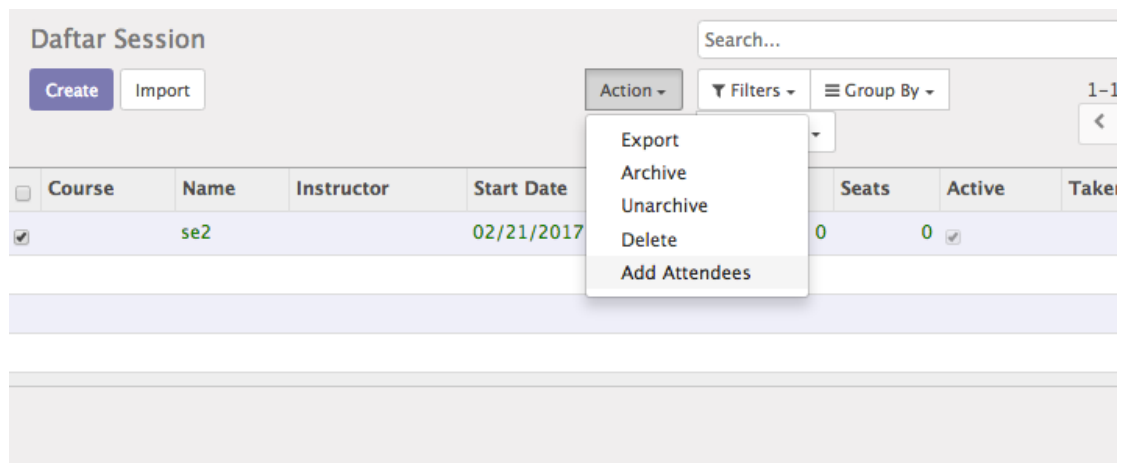
Restart odoo dan update module.

Hasilnya muncul menu Add Attendees dibawah context menu More ... di halaman form Session.



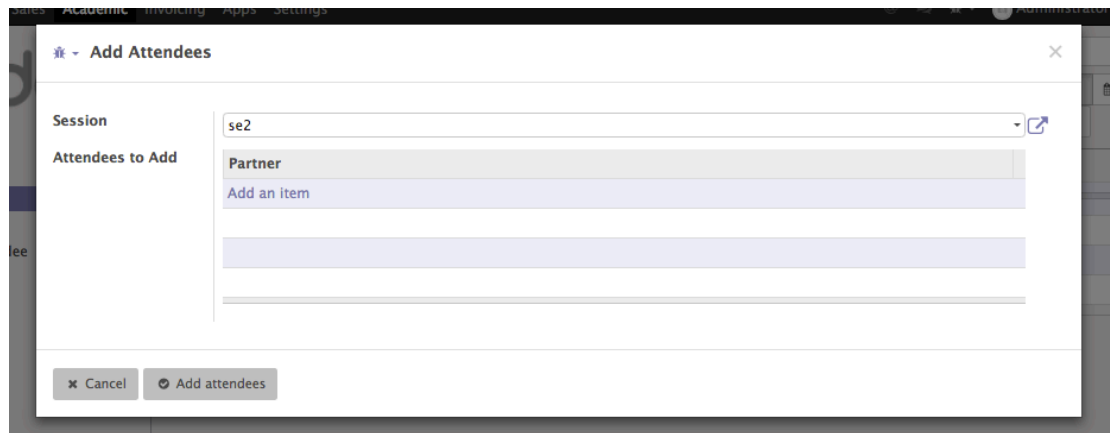
Gambar 37 Muncul context menu di form view Session

dan juga di list **Session** waktu dipilih salah satunya...



Gambar 38 Muncul context menu di list view ketika dipilih salah satu session

Waktu menu **Add Attendee** diclick, wizard otomatis muncul dan session udah terisi sesuai yang dipilih sebelumnya.



Gambar 39 Default value Session di wizard

2.6 ADD AN ONCHANGE METHOD

Quiz: tambahkan onchange method untuk menampilkan attendance yang sudah ada pada session yang dipilih pada wizard.

2.7 WIZARD UNTUK BANYAK SESSION SEKALIGUS

Gimana kalau wizardnya harus bisa nambahin banyak Attendee ke banyak Session sekaligus ?

2.7.1 EDIT FILE WIZARD/CREATE_ATTENDEE.PY

Modif model wizard, ganti field “session_id” yang tadinya many2one jadi field many2many “session_ids”.

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

class CreateAttendeeWizard(models.TransientModel):
    _name = 'academic.create.attendee.wizard'

    def _get_active_session(self):
        context = self.env.context
        if context.get('active_model') == 'academic.session':
            return context.get('active_ids', False)
        return False

    session_ids = fields.Many2many(comodel_name="academic.session",
                                  string="Sessions", )

    attendee_ids = fields.One2many(comodel_name="academic.attendee.wizard",
                                   inverse_name="wizard_id", string="Attendees to Add",
                                   required=False, )
```

Gambar 40 Field session_ids many2many

Modif juga method default valuenya supaya menyesuaikan dengan field `session_ids`...

Disini method `_get_active_sessions()` mengambil key `active_ids` dari `context` variable dan bukan `active_id` seperti sebelumnya. Hasilnya berupa list of ids dari session yang dipilih di wizard.

Nilai default `session_ids` diisikan dengan return value dari function tersebut.

Apalagi method `action_add_attendee` harus juga dimodif karena data `session_ids` yang datang dari wizard bukan berupa integer lagi, tapi udah array list dari banyak session..

```
@api.multi
def action_add_attendee(self):
    self.ensure_one()
    sessions = self.session_ids
    att_data = [{'partner_id': att.partner_id.id}
                for att in self.attendee_ids]

    for session in sessions:
        session.attendee_ids = [(0, 0, data) for data in att_data]

    return {'type': 'ir.actions.act_window_close'}
```

Gambar 41 Cara save ke session attendee_ids

Variable `sessions` akan berisi list dari id yang dipilih pada wizard.

List ini kita looping satu per satu, lalu pada setiap element list yanb berupa object session, dilakukan update terhadap field `attendee_ids` seperti cara sebelumnya.

2.7.2 EDIT FILE XML WIZARD/CREATE_ATTENDEE.XML

Ganti pemanggilan field `session_id` menjadi `session_ids`.

```
<field name="arch" type="xml">
  <form string="Add attendee" version="7.0">
    <group>
      <field name="session_ids"/>
      <field name="attendee_ids" mode="tree">
        <tree string="Attendees"
              editable="bottom">
          <field name="partner_id"/>
        </tree>
      </field>
    </group>
  </form>
</field>
```

```
</tree>
</field>
</group>
```

Restart odoo dan update module, hasilnya

The screenshot shows the 'Add attendee' window in the Odoo Academic module. The window has a title bar with 'Sales', 'Academic', 'Invoicing', 'Apps', and 'Settings'. The main content area is divided into two sections: 'Sessions' and 'Attendees to Add'.

Sessions Table:

Course	Name	Instructor	Start Date	Duration	Seats	Active	Taken Seat
se3			02/22/2017		0	<input checked="" type="checkbox"/>	<div></div>
se2			02/21/2017		0	<input checked="" type="checkbox"/>	<div></div>
Add an item							

Attendees to Add List:

- Partner**
- Badu ☐
- Budi ☐
- [Add an item](#)

At the bottom of the window, there are two buttons: 'Cancel' and 'Add attendees'.

Gambar 42 Bisa insert banyak attendee sekaligus ke banyak session

3 REKAP HARI 4

Wuiiihhh makin seru nih gan... udah hari ke-4. Mudah-mudahan bukan makin pusing ☺

Berikut rekap apa yang udah kita pelajari di hari ke 4 ini.

Security

Bikin group, access right group, masukkan user ke group, import CSV untuk access right, access menu, record rules, dan security workflow.

Wizard

Membuat wizard class, modifikasi form wizard, method untuk memproses data wizard, context menu, on change di wizard, many2many relation.