Building A module

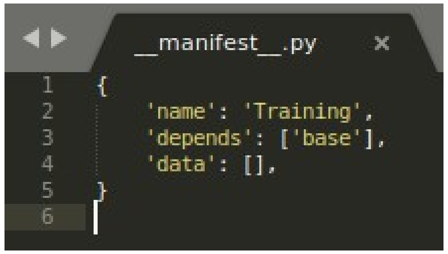
Building a module

1. Create folder on your repository:

Ex: “training”

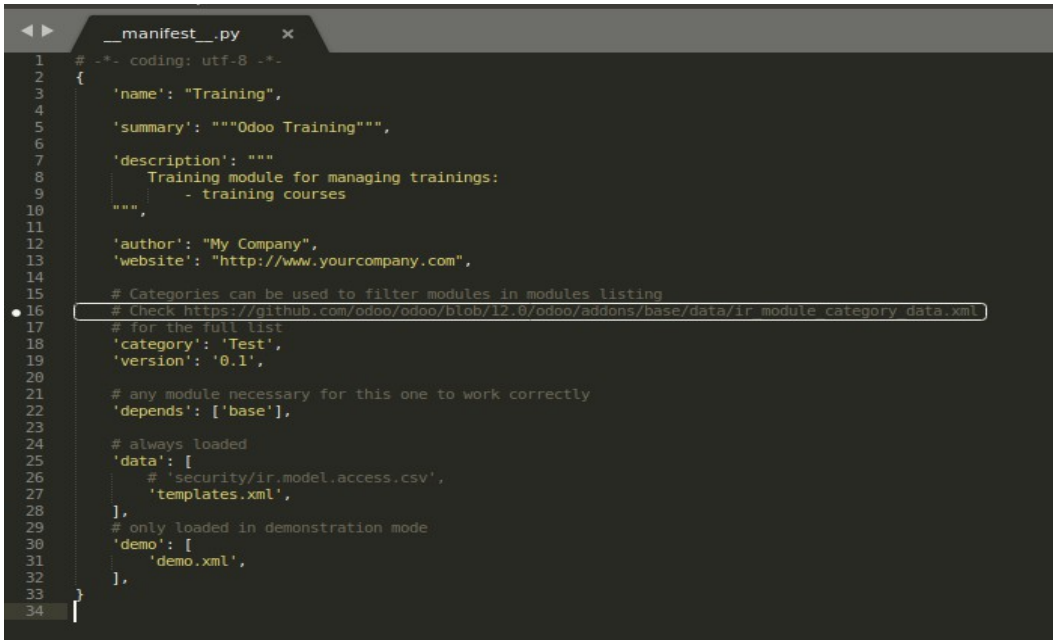
2. create __init__.py file

3. create __manifest__.py file:



__manifest__.py

```
{
    'name': 'Training',
    'depends': ['base'],
    'data': [],
}
```



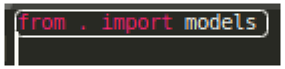
__manifest__.py

```
# -*- coding: utf-8 -*-
{
    'name': "Training",

    'summary': """Odoo Training""",

    'description': """
        Training module for managing trainings:
            - training courses
    """,

    'author': "My Company",
    'website': "http://www.yourcompany.com",

    # Categories can be used to filter modules in modules listing
    # Check https://github.com/odoo/odoo/blob/12.0/odoo/addons/base/data/ir_module_category_data.xml
    # for the full list
    'category': 'Test',
    'version': '0.1',

    # any module necessary for this one to work correctly
    'depends': ['base'],

    # always loaded
    'data': [
        # 'security/ir.model.access.csv',
        'templates.xml',
    ],
    # only loaded in demonstration mode
    'demo': [
        'demo.xml',
    ],
}
```

Building a module

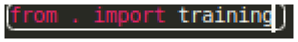
4. Create folder “models” in “training” folder

- import folder “models” in __init__.py file



```
from . import models
```

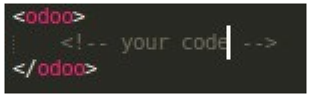
- create __init__.py file in folder models
- create python file in folder models ex: “training.py”
- import “training.py” in models/__init__.py file



```
from . import training
```

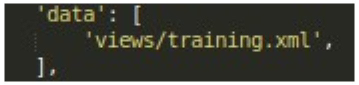
5. Create “views” folder in “training” folder

- create xml files in “views” folder ex: training.xml
- write <odoo></odoo> on “training.xml”



```
<odoo>
    <!-- your code -->
</odoo>
```

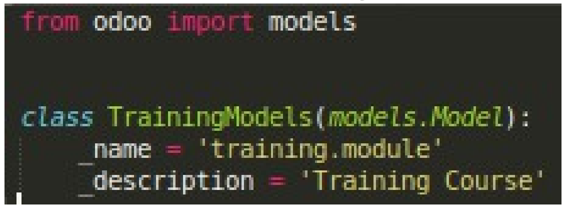
- call “training.xml” in __manifest__.py file



```
'data': [
    'views/training.xml',
],
```

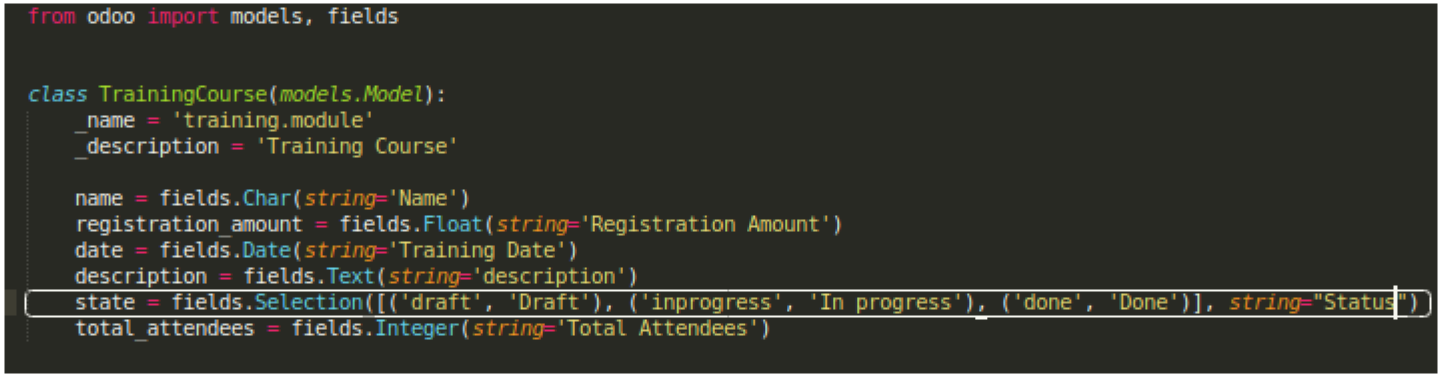
Basic Fields

1. Create new object/model



```python
from odoo import models


class TrainingModels(models.Model):
    _name = 'training.module'
    _description = 'Training Course'
```

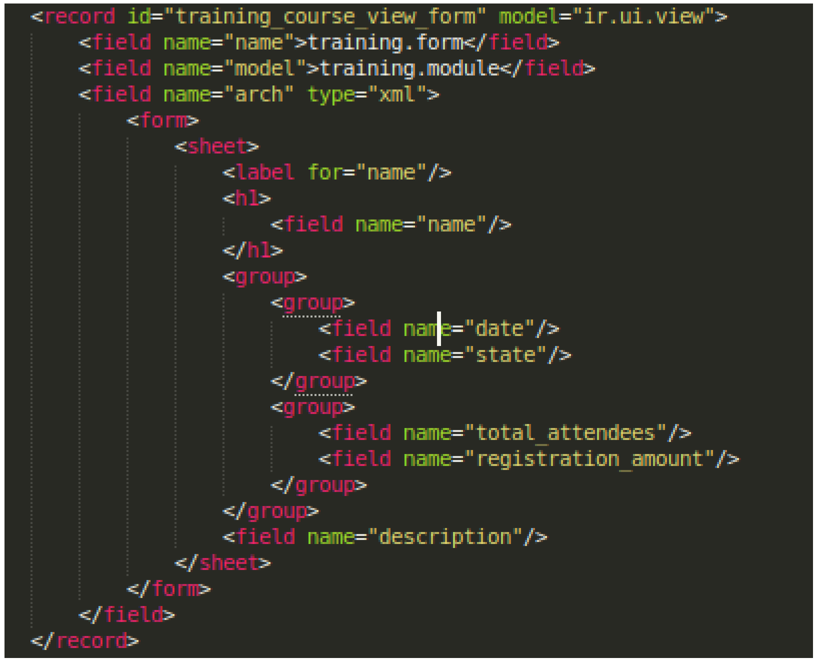
2. Create fields



```python
from odoo import models, fields


class TrainingCourse(models.Model):
    _name = 'training.module'
    _description = 'Training Course'

    name = fields.Char(string='Name')
    registration_amount = fields.Float(string='Registration Amount')
    date = fields.Date(string='Training Date')
    description = fields.Text(string='description')
    state = fields.Selection([('draft', 'Draft'), ('inprogress', 'In progress'), ('done', 'Done')], string="Status")
    total_attendees = fields.Integer(string='Total Attendees')
```

Basic Views

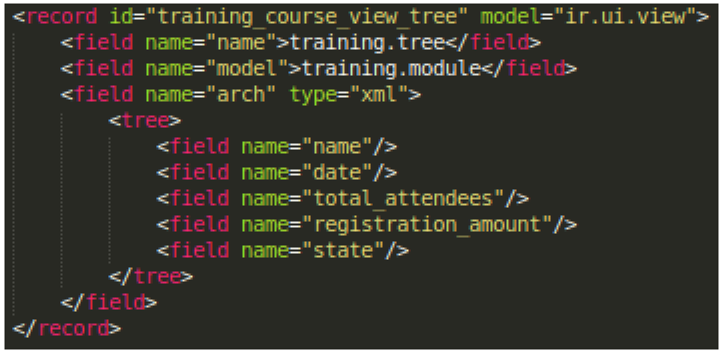
1. Form View



```
<record id="training_course_view_form" model="ir.ui.view">
    <field name="name">training.form</field>
    <field name="model">training.module</field>
    <field name="arch" type="xml">
        <form>
            <sheet>
                <label for="name"/>
                <h1>
                    <field name="name"/>
                </h1>
                <group>
                    <group>
                        <field name="date"/>
                        <field name="state"/>
                    </group>
                    <group>
                        <field name="total_attendees"/>
                        <field name="registration_amount"/>
                    </group>
                </group>
                <field name="description"/>
            </sheet>
        </form>
    </field>
</record>
```

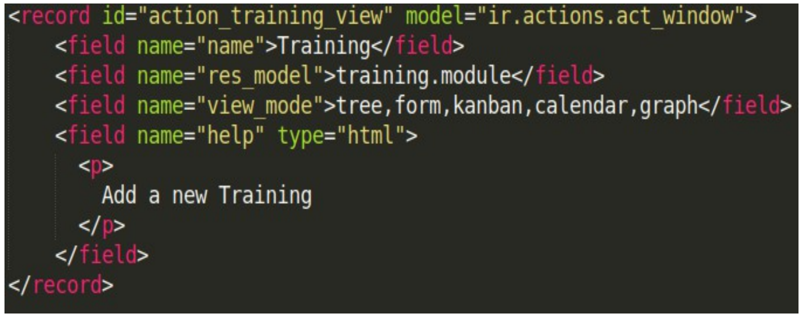
Basic Views

2. Tree View



```
<record id="training_course_view_tree" model="ir.ui.view">
    <field name="name">training.tree</field>
    <field name="model">training.module</field>
    <field name="arch" type="xml">
        <tree>
            <field name="name"/>
            <field name="date"/>
            <field name="total_attendees"/>
            <field name="registration_amount"/>
            <field name="state"/>
        </tree>
    </field>
</record>
```

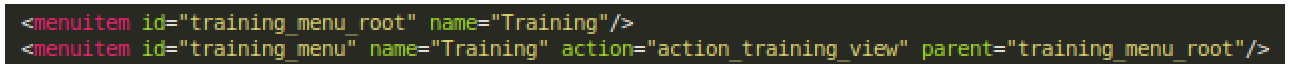
3. Action view



```
<record id="action_training_view" model="ir.actions.act_window">
    <field name="name">Training</field>
    <field name="res_model">training.module</field>
    <field name="view_mode">tree,form,kanban,calendar,graph</field>
    <field name="help" type="html">
      <p>
        Add a new Training
      </p>
    </field>
</record>
```

Basic Views

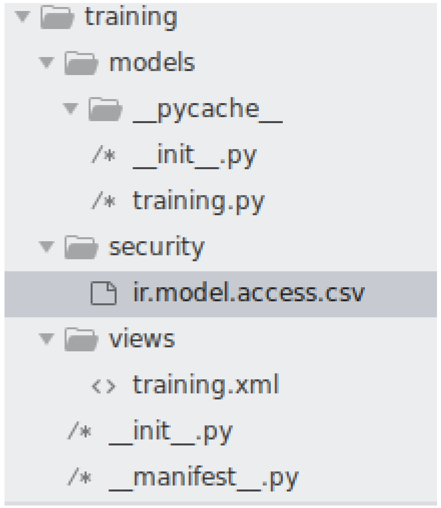
4. Menuitem

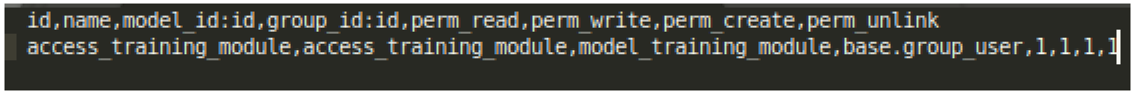


```
<menuitem id="training_menu_root" name="Training"/>
<menuitem id="training_menu" name="Training" action="action_training_view" parent="training_menu_root"/>
```

Security Access

1. **Add folder "security" in "training" folder**
2. **create csv files ex: "ir.model.access.csv" in folder "security"**

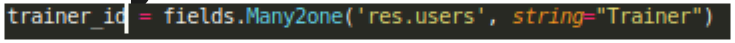




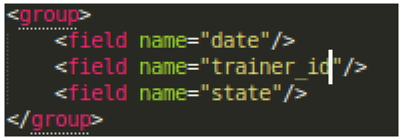
```
id,name,model_id:id,group_id:id,perm_read,perm_write,perm_create,perm_unlink
access_training_module,access_training_module,model_training_module,base.group_user,1,1,1,1
```

Relation between objects

1. Many2one Field



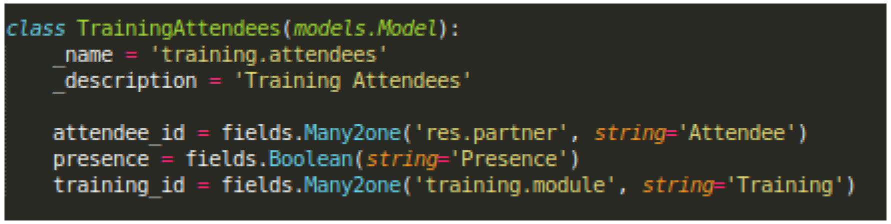
```
trainer_id = fields.Many2one('res.users', string="Trainer")
```



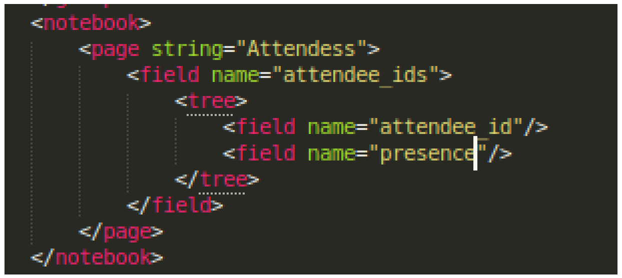
```
<group>
    <field name="date"/>
    <field name="trainer_id"/>
    <field name="state"/>
</group>
```

2. One2many Field

- create new object/models:

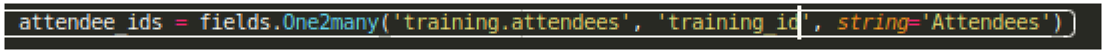


```
class TrainingAttendees(models.Model):
    _name = 'training.attendees'
    _description = 'Training Attendees'

    attendee_id = fields.Many2one('res.partner', string='Attendee')
    presence = fields.Boolean(string='Presence')
    training_id = fields.Many2one('training.module', string='Training')
```



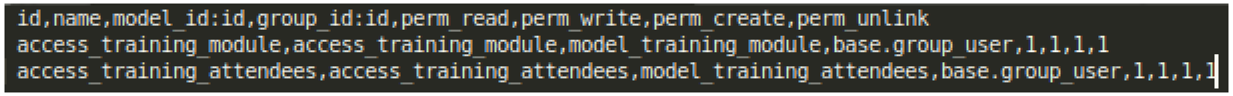
```
<notebook>
    <page string="Attendess">
        <field name="attendee_ids">
            <tree>
                <field name="attendee_id"/>
                <field name="presence"/>
            </tree>
        </field>
    </page>
</notebook>
```

- create One2many fields in “training.module” object/model



```
attendee_ids = fields.One2many('training.attendees', 'training_id', string='Attendees'))
```

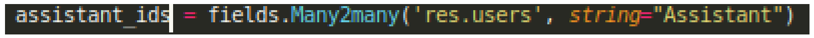
- add security for new object/model



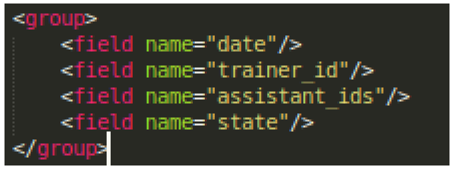
```
id,name,model_id:id,group_id:id,perm_read,perm_write,perm_create,perm_unlink
access_training_module,access_training_module,model_training_module,base.group_user,1,1,1,1
access_training_attendees,access_training_attendees,model_training_attendees,base.group_user,1,1,1,1
```

Relation between objects

3. Many2many Field

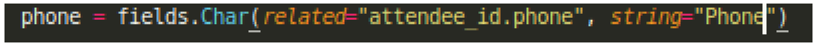


```python
assistant_ids = fields.Many2many('res.users', string="Assistant")
```

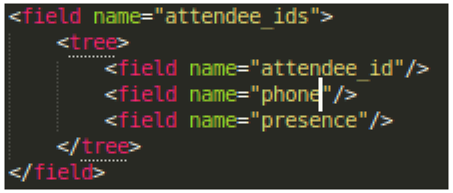


```xml
<group>
    <field name="date"/>
    <field name="trainer_id"/>
    <field name="assistant_ids"/>
    <field name="state"/>
</group>
```

4. Related Field



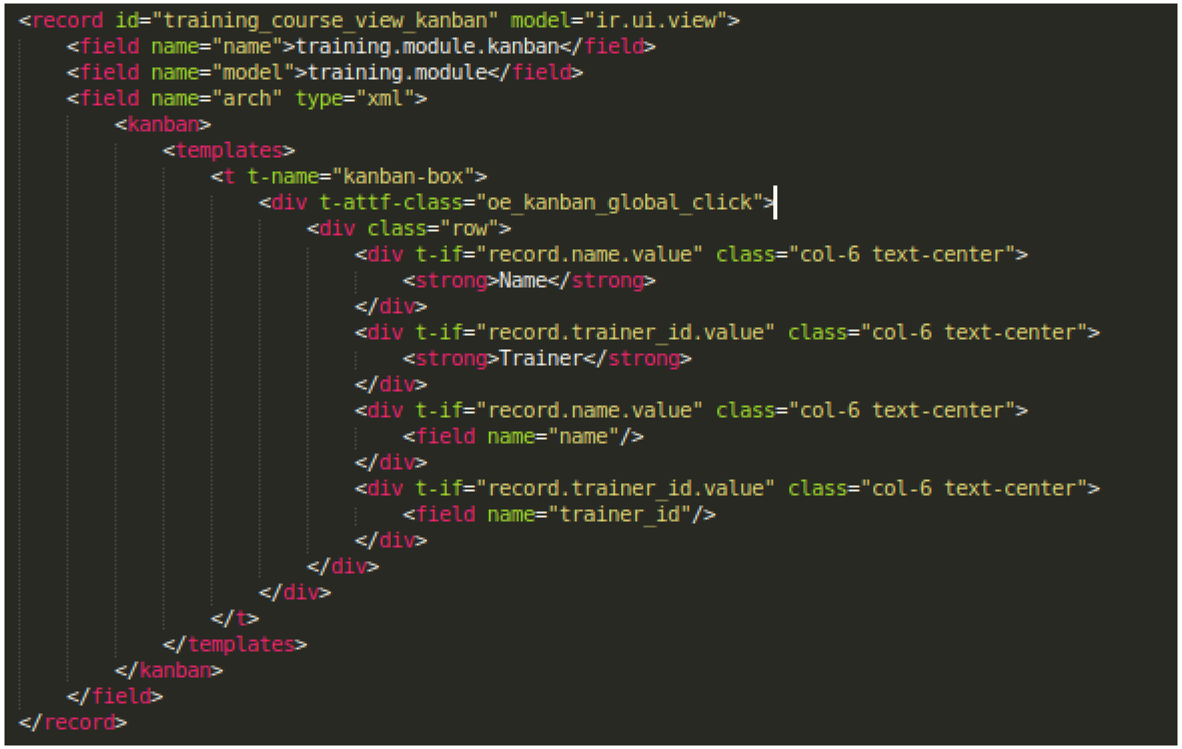
```python
phone = fields.Char(related="attendee_id.phone", string="Phone")
```



```xml
<field name="attendee_ids">
    <tree>
        <field name="attendee_id"/>
        <field name="phone"/>
        <field name="presence"/>
    </tree>
</field>
```

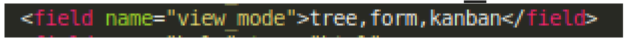
Advance Views

1. Kanban View



```xml
<record id="training_course_view_kanban" model="ir.ui.view">
    <field name="name">training.module.kanban</field>
    <field name="model">training.module</field>
    <field name="arch" type="xml">
        <kanban>
            <templates>
                <t t-name="kanban-box">
                    <div t-attf-class="oe_kanban_global_click">
                        <div class="row">
                            <div t-if="record.name.value" class="col-6 text-center">
                                <strong>Name</strong>
                            </div>
                            <div t-if="record.trainer_id.value" class="col-6 text-center">
                                <strong>Trainer</strong>
                            </div>
                            <div t-if="record.name.value" class="col-6 text-center">
                                <field name="name"/>
                            </div>
                            <div t-if="record.trainer_id.value" class="col-6 text-center">
                                <field name="trainer_id"/>
                            </div>
                        </div>
                    </div>
                </t>
            </templates>
        </kanban>
    </field>
</record>
```

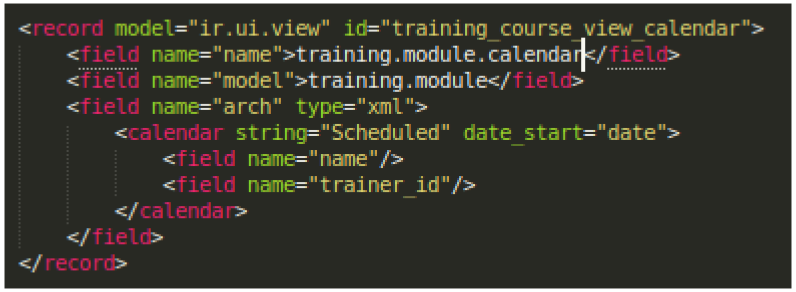
- add “kanban” in view_mode off action view.



```xml
<field name="view_mode">tree,form,kanban</field>
```

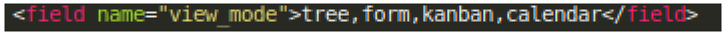
Advance Views

2. Calendar View



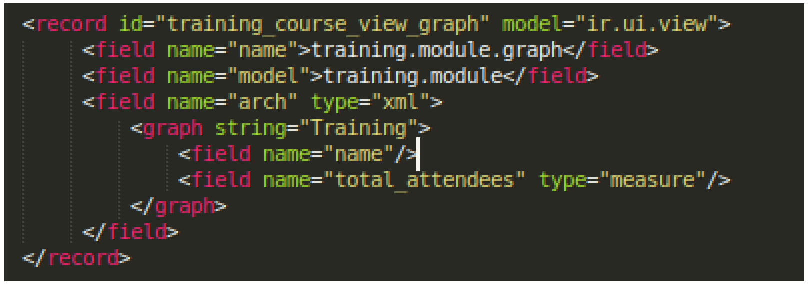
```
<record model="ir.ui.view" id="training_course_view_calendar">
    <field name="name">training.module.calendar</field>
    <field name="model">training.module</field>
    <field name="arch" type="xml">
        <calendar string="Scheduled" date_start="date">
            <field name="name"/>
            <field name="trainer_id"/>
        </calendar>
    </field>
</record>
```

- add “calendar” in view_mode off action view.



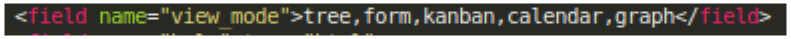
```
<field name="view_mode">tree,form,kanban,calendar</field>
```

3. Graph View



```
<record id="training_course_view_graph" model="ir.ui.view">
    <field name="name">training.module.graph</field>
    <field name="model">training.module</field>
    <field name="arch" type="xml">
        <graph string="Training">
            <field name="name"/>
            <field name="total_attendees" type="measure"/>
        </graph>
    </field>
</record>
```

- add “graph” in view_mode off action view.

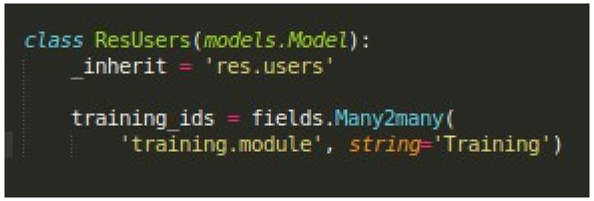


```
<field name="view_mode">tree,form,kanban,calendar,graph</field>
```

Inheritance

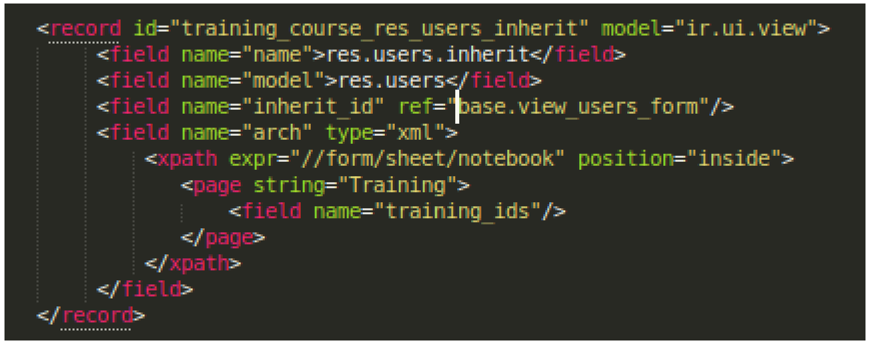
1. Inherit object

- add field ex: “training_ids” Many2many fields



```
class ResUsers(models.Model):
    _inherit = 'res.users'

    training_ids = fields.Many2many(
        'training.module', string='Training')
```

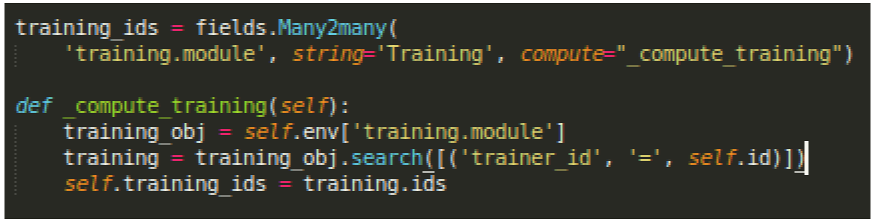
2. inherit view



```
<record id="training_course_res_users_inherit" model="ir.ui.view">
     <field name="name">res.users.inherit</field>
     <field name="model">res.users</field>
     <field name="inherit_id" ref="base.view_users_form"/>
     <field name="arch" type="xml">
         <xpath expr="//form/sheet/notebook" position="inside">
            <page string="Training">
                <field name="training_ids"/>
            </page>
         </xpath>
     </field>
</record>
```

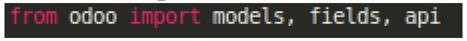
Compute Field

1. create function compute with search function

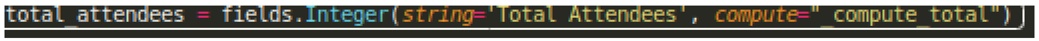


```python
training_ids = fields.Many2many(
    'training.module', string='Training', compute="_compute_training")

def _compute_training(self):
    training_obj = self.env['training.module']
    training = training_obj.search([('trainer_id', '=', self.id)])
    self.training_ids = training.ids
```

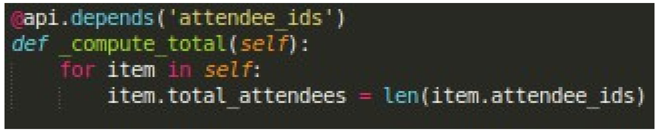
2. compute field with @api.depends



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



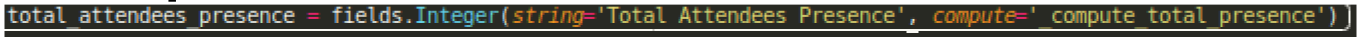
```python
total_attendees = fields.Integer(string='Total Attendees', compute="_compute_total")
```



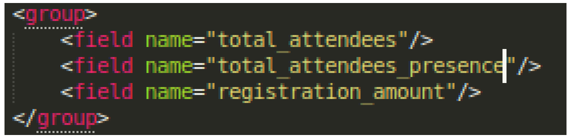
```python
@api.depends('attendee_ids')
def _compute_total(self):
    for item in self:
        item.total_attendees = len(item.attendee_ids)
```

Compute Field

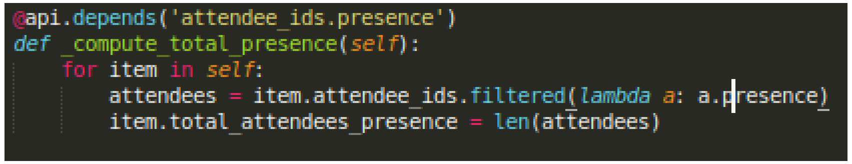
1. Compute field with filtered function



```python
total_attendees_presence = fields.Integer(string='Total Attendees Presence', compute='_compute_total_presence')
```



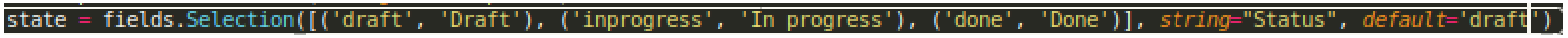
```xml
<group>
    <field name="total_attendees"/>
    <field name="total_attendees_presence"/>
    <field name="registration_amount"/>
</group>
```



```python
@api.depends('attendee_ids.presence')
def _compute_total_presence(self):
    for item in self:
        attendees = item.attendee_ids.filtered(lambda a: a.presence)
        item.total_attendees_presence = len(attendees)
```

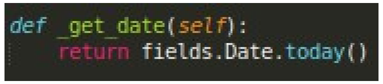
Default Value

1. add “default” attribute to field

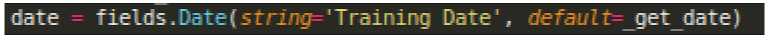


```
state = fields.Selection([('draft', 'Draft'), ('inprogress', 'In progress'), ('done', 'Done')], string="Status", default='draft')
```

2. add default value with function



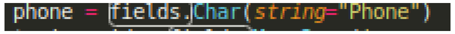
```
def _get_date(self):
    return fields.Date.today()
```



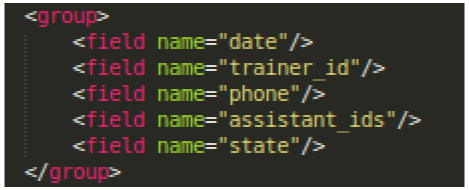
```
date = fields.Date(string='Training Date', default=_get_date)
```

Onchange

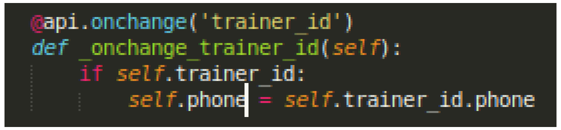
1. create fields phone in “training.module” object



```python
phone = fields.Char(string="Phone")
```



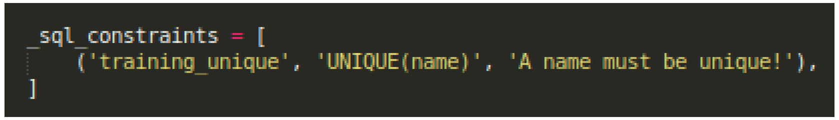
```xml
<group>
    <field name="date"/>
    <field name="trainer_id"/>
    <field name="phone"/>
    <field name="assistant_ids"/>
    <field name="state"/>
</group>
```



```python
@api.onchange('trainer_id')
def _onchange_trainer_id(self):
    if self.trainer_id:
        self.phone = self.trainer_id.phone
```

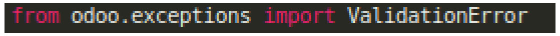
Constraint

1. _sql_constraints

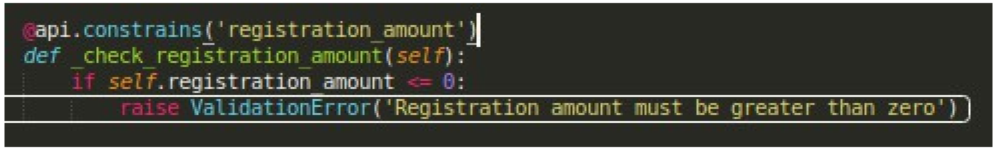


```
_sql_constraints = [
    ('training_unique', 'UNIQUE(name)', 'A name must be unique!'),
]
```

2. @api.constraints
   - import ValidationError(for dialog form)



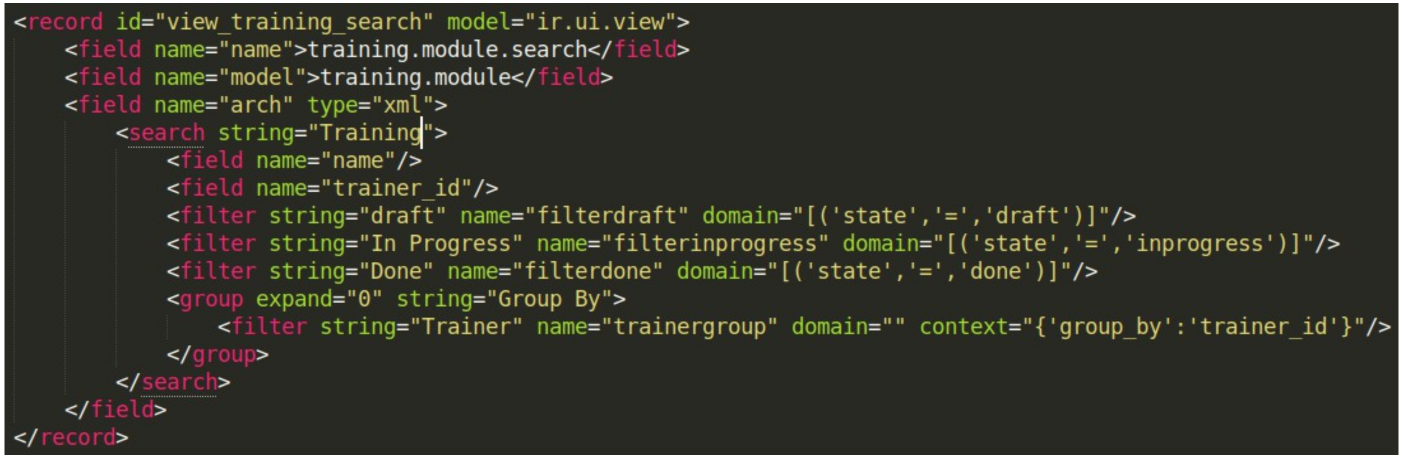
```
from odoo.exceptions import ValidationError
```



```
@api.constrains('registration_amount')
def _check_registration_amount(self):
    if self.registration_amount <= 0:
        raise ValidationError('Registration amount must be greater than zero')
```

Search View

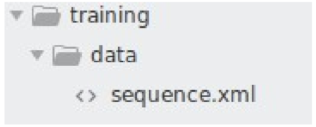
1. Group by and filter



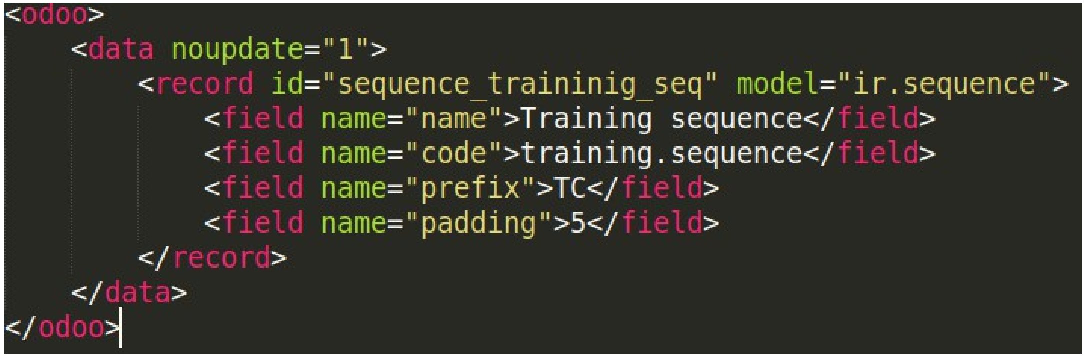
```
<record id="view_training_search" model="ir.ui.view">
    <field name="name">training.module.search</field>
    <field name="model">training.module</field>
    <field name="arch" type="xml">
        <search string="Training">
            <field name="name"/>
            <field name="trainer_id"/>
            <filter string="draft" name="filterdraft" domain="[('state','=','draft')]"/>
            <filter string="In Progress" name="filterinprogress" domain="[('state','=','inprogress')]"/>
            <filter string="Done" name="filterdone" domain="[('state','=','done')]"/>
            <group expand="0" string="Group By">
                <filter string="Trainer" name="trainergroup" domain="" context="{'group_by':'trainer_id'}"/>
            </group>
        </search>
    </field>
</record>
```

Sequence Number

1. create folder “data” in “training” folder
2. create “sequence.xml” file in “data” folder

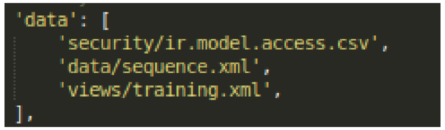


```
▾ training
  ▾ data
      <> sequence.xml
```



```
<odoo>
    <data noupdate="1">
        <record id="sequence_traininig_seq" model="ir.sequence">
            <field name="name">Training sequence</field>
            <field name="code">training.sequence</field>
            <field name="prefix">TC</field>
            <field name="padding">5</field>
        </record>
    </data>
</odoo>
```

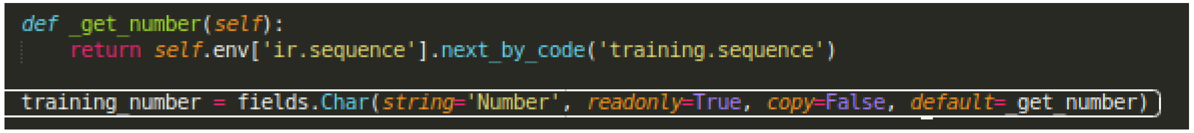
3. add “sequence.xml” in __manifest__.xml



```
'data': [
    'security/ir.model.access.csv',
    'data/sequence.xml',
    'views/training.xml',
],
```

Sequence Number

5. call sequence

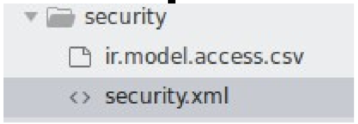


```
def _get_number(self):
    return self.env['ir.sequence'].next_by_code('training.sequence')

training_number = fields.Char(string='Number', readonly=True, copy=False, default=_get_number)
```

Security

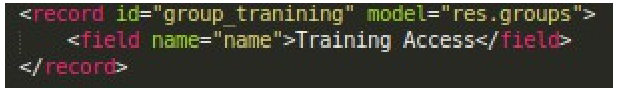
1. Groups

   - create “security.xml” file in “security” folder and put “security.xml” to __manifest__.py



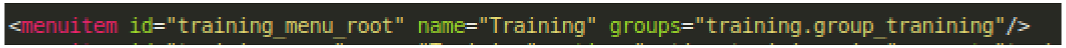
```
security
    ir.model.access.csv
    security.xml
```

   - create group



```
<record id="group_tranining" model="res.groups">
    <field name="name">Training Access</field>
</record>
```

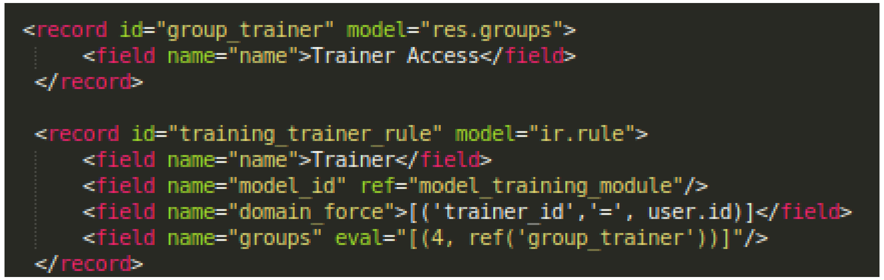
   - implement group on menu/fields.



```
<menuitem id="training_menu_root" name="Training" groups="training.group_tranining"/>
```

Security

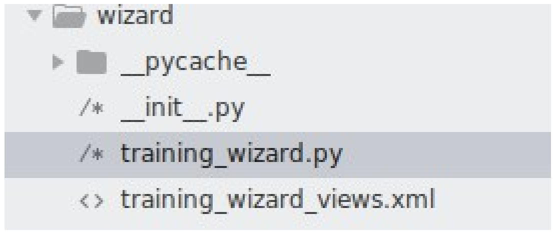
1. Record Rules
   - create new group or use existing group
   - create rule



```xml
<record id="group_trainer" model="res.groups">
     <field name="name">Trainer Access</field>
 </record>

 <record id="training_trainer_rule" model="ir.rule">
     <field name="name">Trainer</field>
     <field name="model_id" ref="model_training_module"/>
     <field name="domain_force">[('trainer_id','=', user.id)]</field>
     <field name="groups" eval="[(4, ref('group_trainer'))]"/>
 </record>
```

Wizard

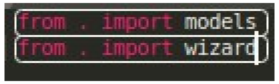
1. create wizard folder in training folder
    - create “__init__.py” file in wizard folder
    - create python file in wizard folder ex: “training_wizard.py”
    - create xml file in wizard folder ex: “training_wizard_views.xml”



```
wizard
    __pycache__
    __init__.py
    training_wizard.py
    training_wizard_views.xml
```

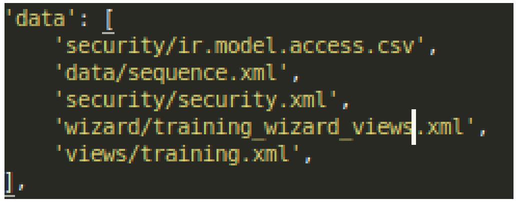
Wizard

2. call wizard
   - import wizard folder in "__init__.py"



```python
from . import models
from . import wizard
```

   - call "training_wizard_views.xml" views in "__manifest__.py"

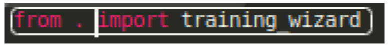


```python
'data': [
    'security/ir.model.access.csv',
    'data/sequence.xml',
    'security/security.xml',
    'wizard/training_wizard_views.xml',
    'views/training.xml',
],
```

Wizard

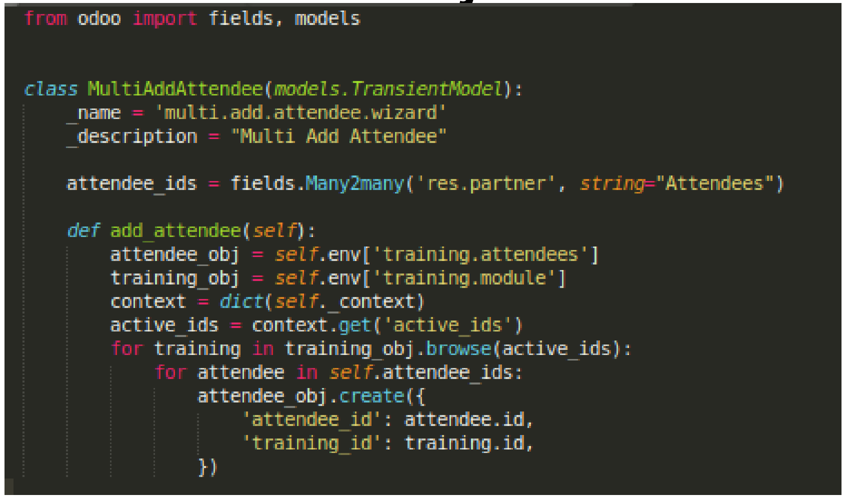
2. call wizard

   - import “training_wizard.py” in “__init__.py”



```
from . import training_wizard
```

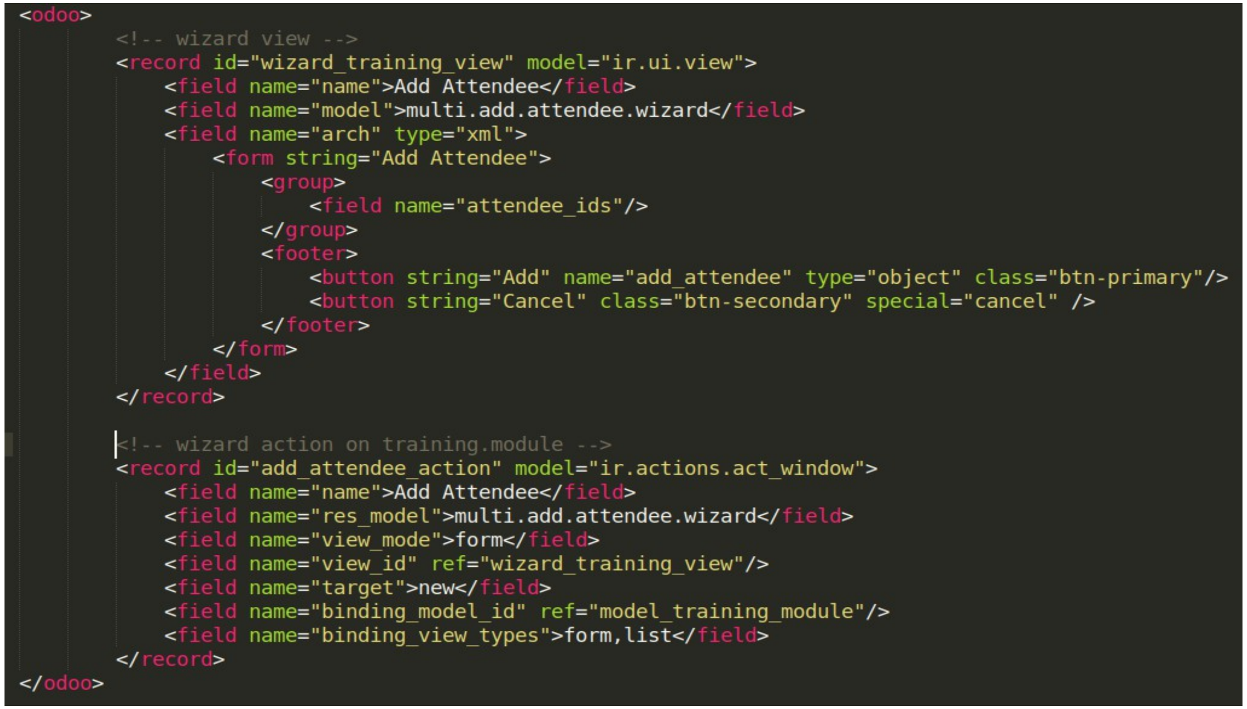
3. create new object/models, fields and function



```
from odoo import fields, models


class MultiAddAttendee(models.TransientModel):
    _name = 'multi.add.attendee.wizard'
    _description = "Multi Add Attendee"

    attendee_ids = fields.Many2many('res.partner', string="Attendees")

    def add_attendee(self):
        attendee_obj = self.env['training.attendees']
        training_obj = self.env['training.module']
        context = dict(self._context)
        active_ids = context.get('active_ids')
        for training in training_obj.browse(active_ids):
            for attendee in self.attendee_ids:
                attendee_obj.create({
                    'attendee_id': attendee.id,
                    'training_id': training.id,
                })
```

Wizard

4. create wizard view

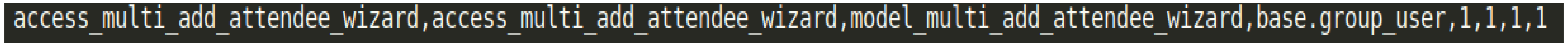


```
<odoo>
        <!-- wizard view -->
        <record id="wizard_training_view" model="ir.ui.view">
            <field name="name">Add Attendee</field>
            <field name="model">multi.add.attendee.wizard</field>
            <field name="arch" type="xml">
                <form string="Add Attendee">
                    <group>
                        <field name="attendee_ids"/>
                    </group>
                    <footer>
                        <button string="Add" name="add_attendee" type="object" class="btn-primary"/>
                        <button string="Cancel" class="btn-secondary" special="cancel" />
                    </footer>
                </form>
            </field>
        </record>

        <!-- wizard action on training.module -->
        <record id="add_attendee_action" model="ir.actions.act_window">
            <field name="name">Add Attendee</field>
            <field name="res_model">multi.add.attendee.wizard</field>
            <field name="view_mode">form</field>
            <field name="view_id" ref="wizard_training_view"/>
            <field name="target">new</field>
            <field name="binding_model_id" ref="model_training_module"/>
            <field name="binding_view_types">form,list</field>
        </record>
</odoo>
```

Wizard

5. Add Security

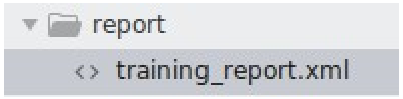
- add access for wizard on file “ir.model.access.csv”



```
access_multi_add_attendee_wizard,access_multi_add_attendee_wizard,model_multi_add_attendee_wizard,base.group_user,1,1,1,1
```

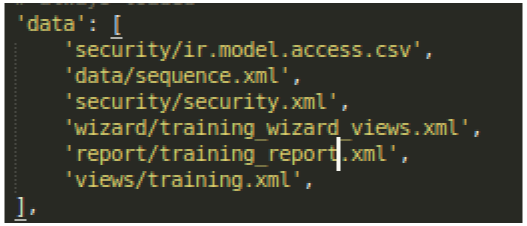
Report

1. report folder in training folder
   - create “training_report.xml” file



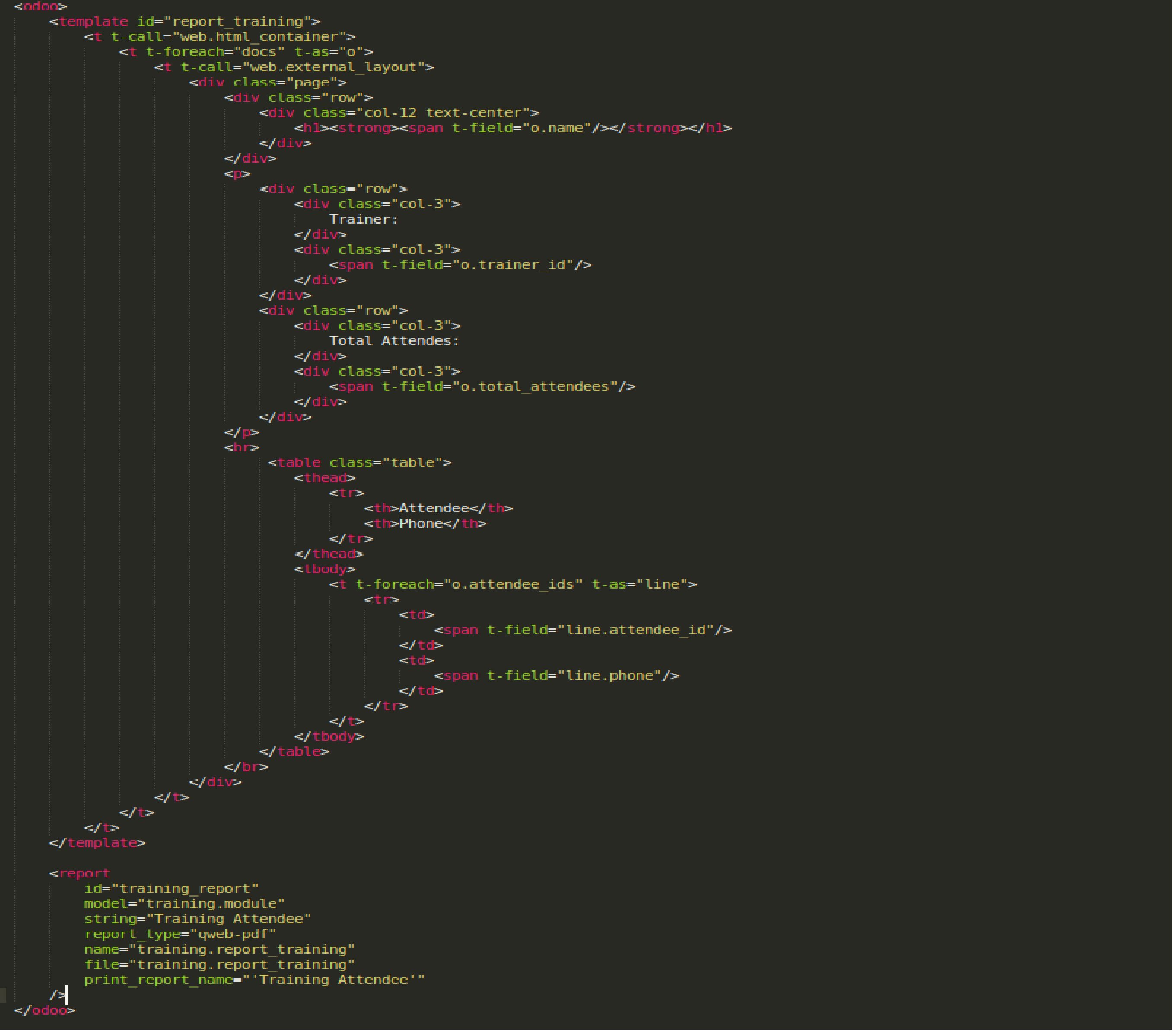
```
report
    training_report.xml
```

2. call “training_report.xml” in “__manifest__.py” file



```
'data': [
    'security/ir.model.access.csv',
    'data/sequence.xml',
    'security/security.xml',
    'wizard/training_wizard_views.xml',
    'report/training_report.xml',
    'views/training.xml',
],
```

3. create report template



```xml
<odoo>
    <template id="report_training">
        <t t-call="web.html_container">
            <t t-foreach="docs" t-as="o">
                <t t-call="web.external_layout">
                    <div class="page">
                        <div class="row">
                            <div class="col-12 text-center">
                                <h1><strong><span t-field="o.name"/></strong></h1>
                            </div>
                        </div>
                        <p>
                            <div class="row">
                                <div class="col-3">
                                    Trainer:
                                </div>
                                <div class="col-3">
                                    <span t-field="o.trainer_id"/>
                                </div>
                            </div>
                            <div class="row">
                                <div class="col-3">
                                    Total Attendes:
                                </div>
                                <div class="col-3">
                                    <span t-field="o.total_attendees"/>
                                </div>
                            </div>
                        </p>
                        <br>
                             <table class="table">
                                <thead>
                                    <tr>
                                        <th>Attendee</th>
                                        <th>Phone</th>
                                    </tr>
                                </thead>
                                <tbody>
                                    <t t-foreach="o.attendee_ids" t-as="line">
                                        <tr>
                                            <td>
                                                <span t-field="line.attendee_id"/>
                                            </td>
                                            <td>
                                                <span t-field="line.phone"/>
                                            </td>
                                        </tr>
                                    </t>
                                </tbody>
                            </table>
                        </br>
                    </div>
                </t>
            </t>
        </t>
    </template>

    <report
        id="training_report"
        model="training.module"
        string="Training Attendee"
        report_type="qweb-pdf"
        name="training.report_training"
        file="training.report_training"
        print_report_name="'Training Attendee'"
    />
</odoo>
```