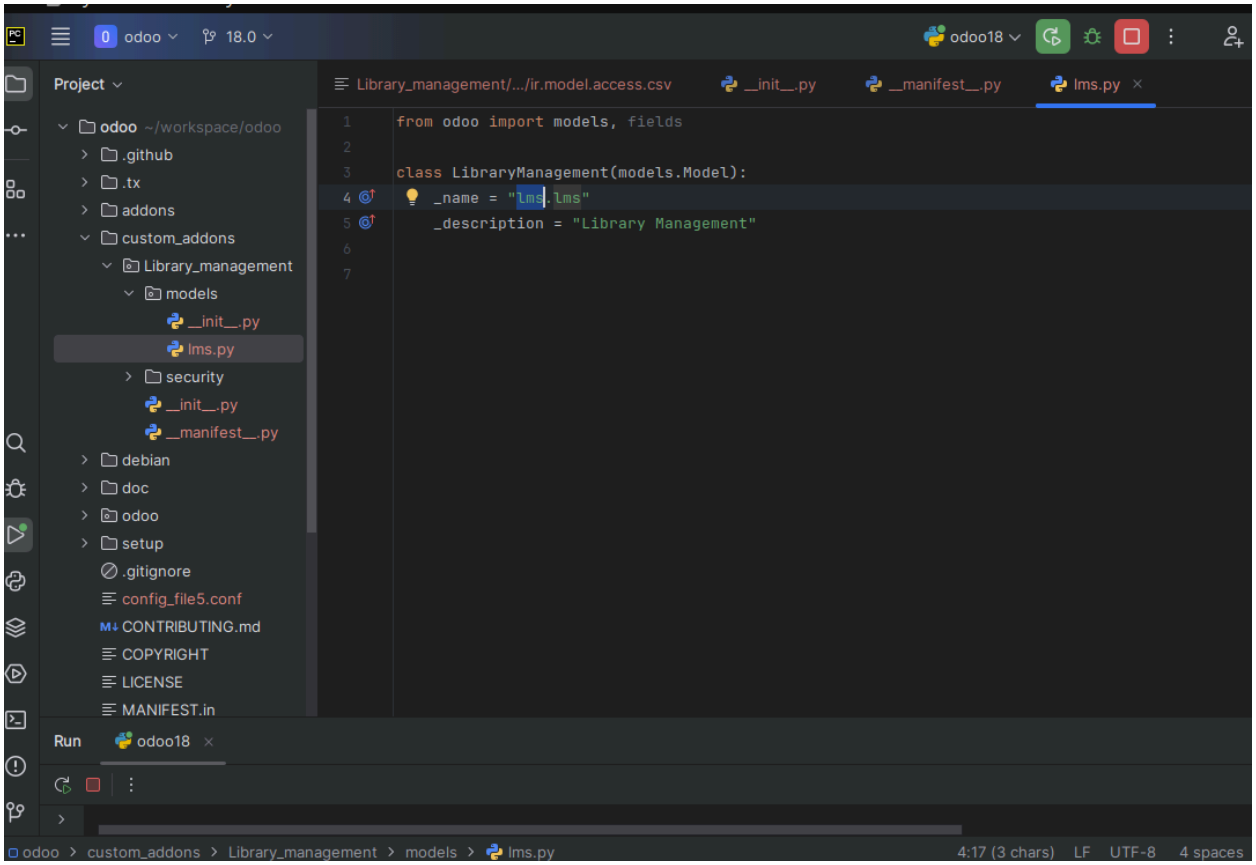
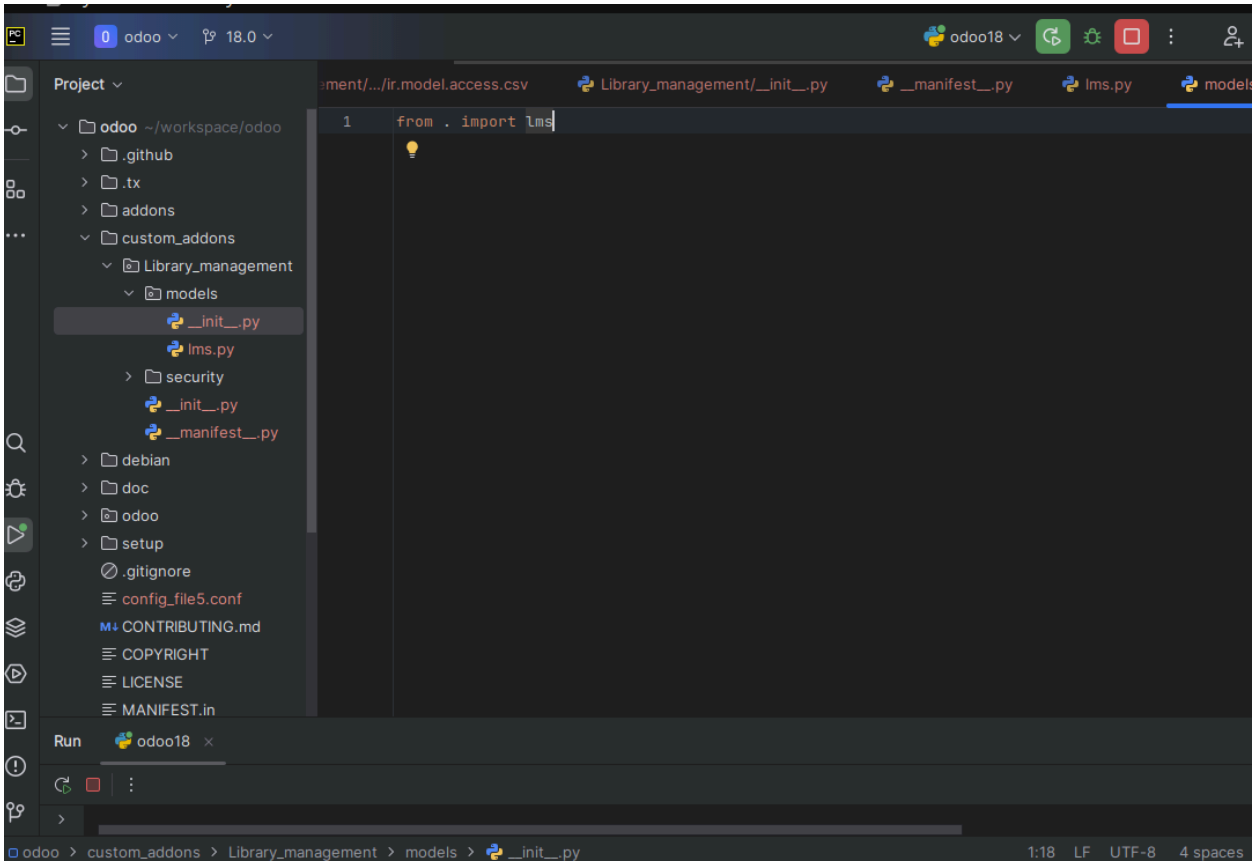
QUE 2. Create a new module and explore the manifest file in detail.

ANS: -

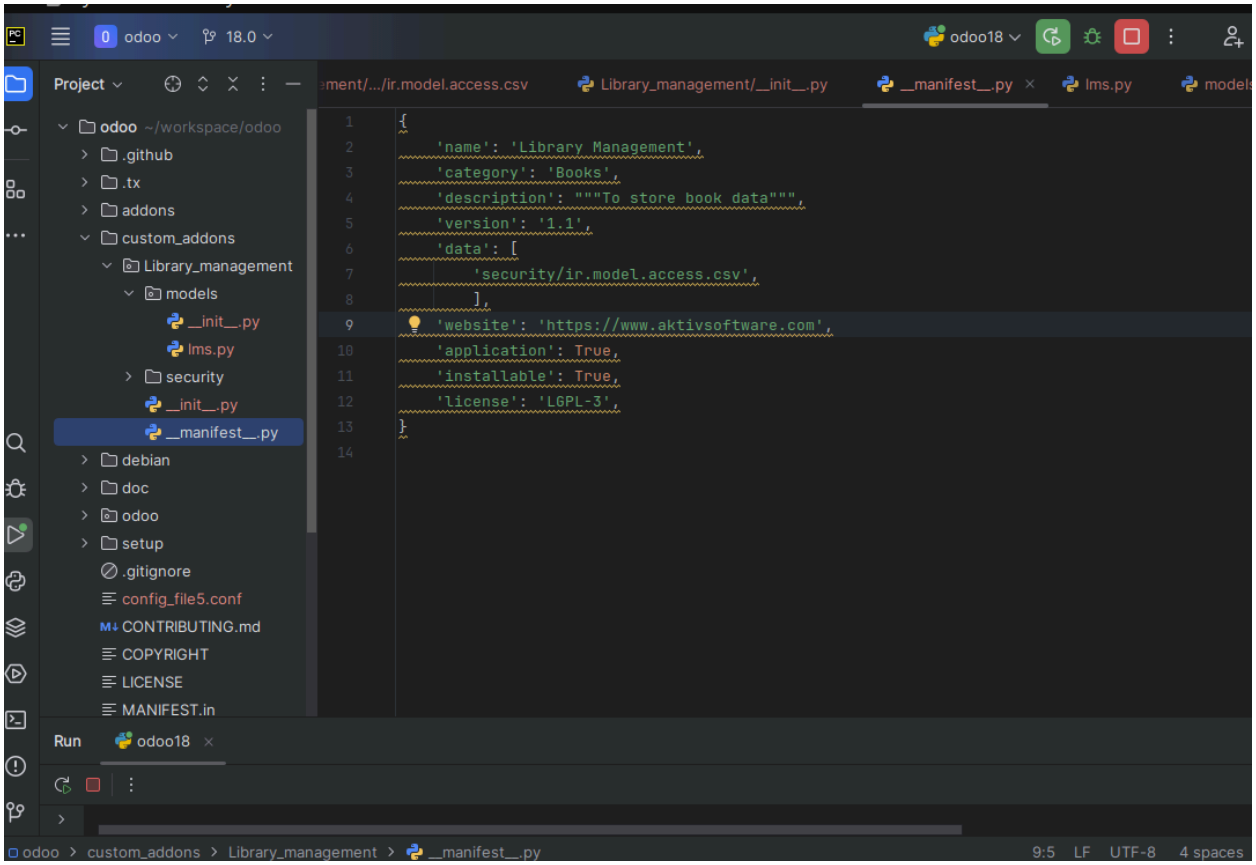
-> Create one custom_addons -> Create one package : Library_management -> Create one package : models -> lms.py



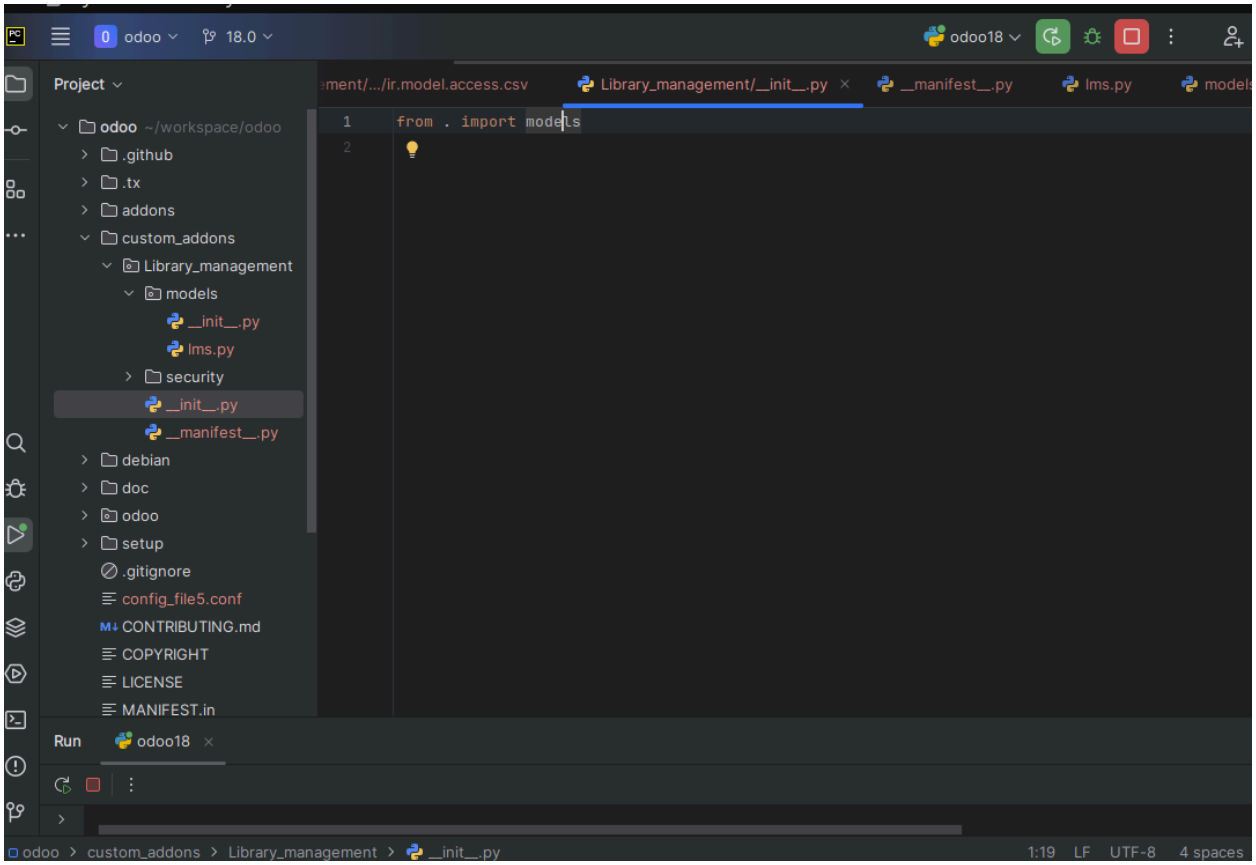
-> Create one custom_addons -> Create one package : Library_management -> Create one package : models -> __init__.py



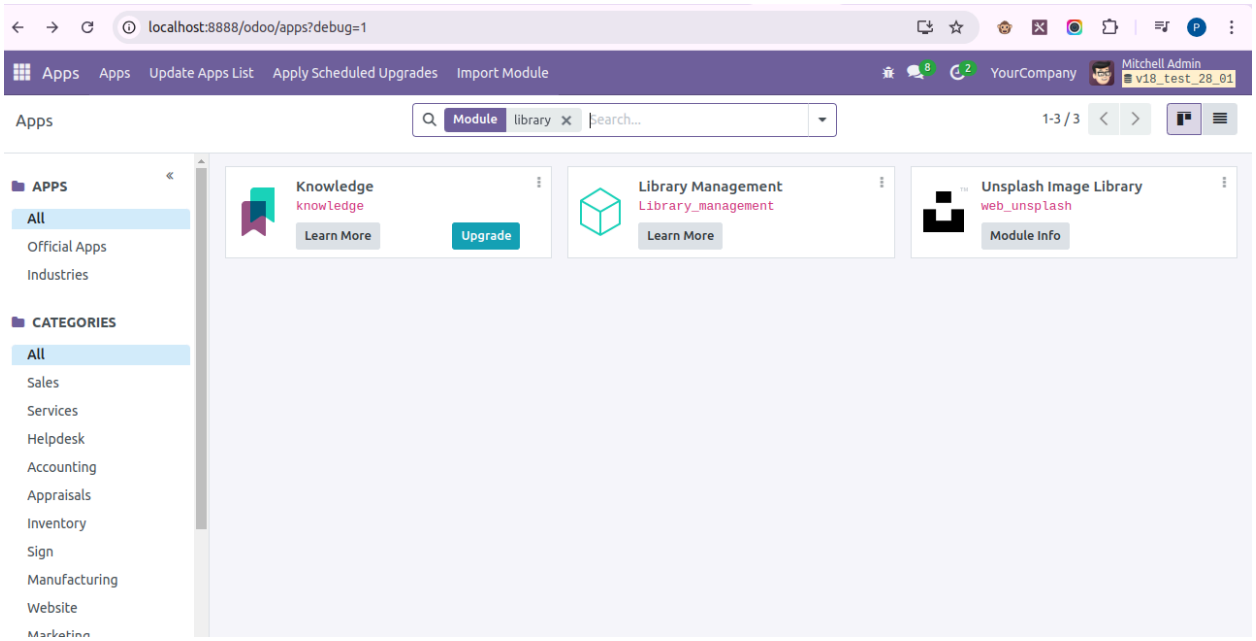
-> Create one custom_addons -> Create one package : Library_management -> Create one __manifest__.py file



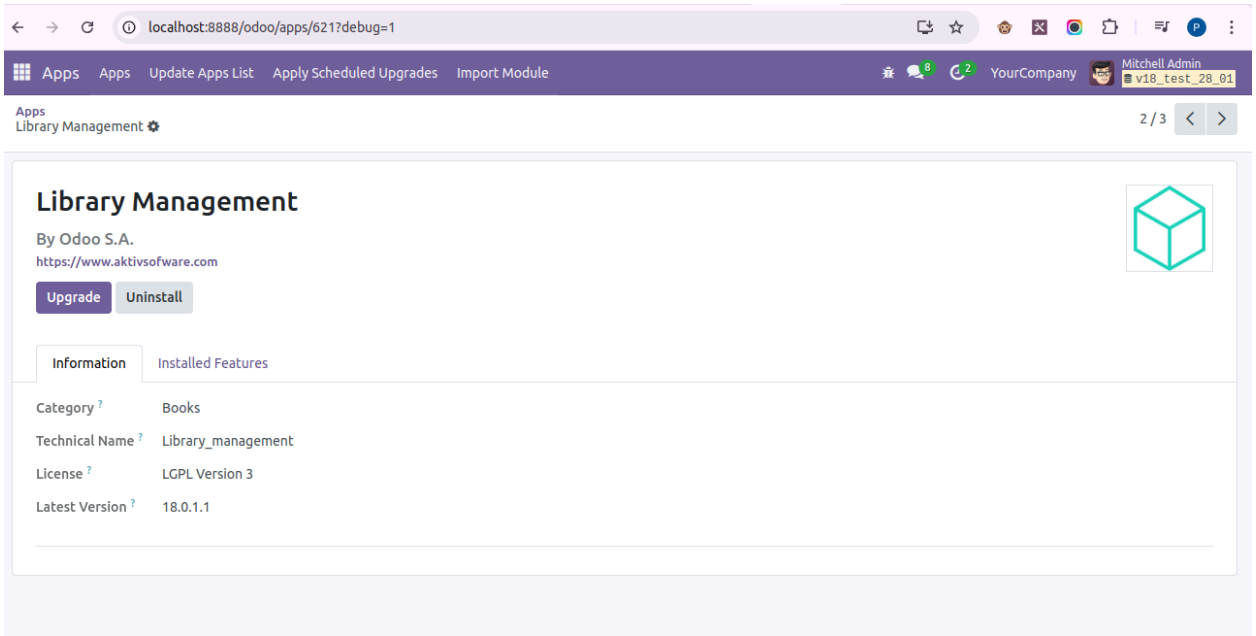
-> Create one custom_addons -> Create one package : Library_management -> __init__.py



-> Activate Library management



-> Module info of Library Management



-> Containing models only.

```
from odoo import models, fields

class LibraryManagement(models.Model):
    _name = "lms.lms"
    _description = "Library Management"
```

Manifest file

-> name :-

It can shows the name of the module in the form also it gives the detail information of the module or the human-readable name of the module.

-> category :-

It can show the category of module in the form or classification category within Odoo, rough business domain for the module.

-> description :-
It gives the description about module or extended description for the module, in reStructuredText.

-> version :-
we can specify the latest version accordingly.

-> data :-
List of data files which must always be installed or updated with the module. A list of paths from the module root directory.

-> depends :-
To specify the modules that are required for a module to work.

-> website :-
website url of module author.

-> author :-
name of the module author.

-> `application` (`bool`, default: `False`) :-
Whether the module should be considered as a fully-fledged application (`True`) or is just a technical module (`False`) that provides some extra functionality to an existing application module

-> demo :-
List of data files which are only installed or updated in *demonstration mode*.

-> `installable` (`bool` default: `True`) :-
Whether a user should be able to install the module from the Web UI or not.

What happens if we don't create __init__.py?

- Odoo won't recognize module's Python files.
- Import errors may occur, causing module installation to fail.
- custom models, controllers, or wizards won't be registered in Odoo.

Why Do We Create __init__.py in models?

- **Marks models/ as a Python Package**
  - Just like the __init__.py in the main module folder, having an __init__.py in the models/ folder tells Python that /models is a package.
  - Without it, Python won't recognize /models as a valid module, and Odoo won't load the model files.
- **Imports Model Files in the Correct Order**
  - Suppose you have multiple model files inside models/ (e.g., book.py, book_tags.py, library_member.py).
  - The __init__.py file in models/ ensures they are loaded correctly.

Why don't we create __init__.py files in other folders like views?

- **We don't need to create an __init__.py file in the views folder because the XML files in that folder don't require any initialization code. The initialization process for the Odoo module happens in Python files (like models, controllers, etc.) where Python code needs to be executed.**