Summer Fellowship Report

On

DRUPAL (Web Development)

Submitted by

PRERNA

Under the guidance of

Prof . P. SUNTHAR and Prof. KANNAN M MOUDGALYA

Chemical Engineering Department

IIT Bombay

April 21st- June 16th ,2020

Acknowledgment

This report has been prepared for the internship that has been done virtually through remote mode under fossee (iit Bombay) in order to study the practical aspects of the Drupal its implementation in the real field with the purpose of learning and exploring in the mentioned field and for professional development.

The aim of this internship is to be familiar with the implication of Drupal and uses of concerned theoretical knowledge and clarifying the career goals, so I have successfully completed the internship and compiled this report as the summary.

I respect and thank Dr. P. Sunthar and Dr.Kannan M Moudgalya for providing me an opportunity to do the project work in Drupal. I owe my deep gratitude to our Project Mentors: Mr.Tejas Vaidya and Miss Ruchi Kumari, who took keen interest on our project work and guided us all along, till the completion of our project work .

Prerna
Place: Delhi
Date: 16th June 2020

Contents

CHAPTER-1

INTRODUCTION TO DRUPAL

Drupal is a free and open-source content management framework written in PHP and distributed under the General Public License (GNU). Drupal provides a back-end framework for at least 2.3% of all websites worldwide ranging from personal blogs to corporate, political, and government sites.

The standard release of Drupal, known as Drupal core, contains basic features common to content-management systems. These include user account registration and maintenance, menu management, RSS feeds, taxonomy, page layout customization, and system administration. The Drupal core installation can serve as a simple website, an Internet forum, or a community website providing for user-generated content.

Although Drupal offers a sophisticated API for developers, basic Web-site installation and administration of the framework requires no programming skills because of which the user with minimal knowledge of coding can build a website with Drupal. Drupal runs on any computing platform that supports both a web server capable of running PHP and a database to store content and configuration.

Drupal is a content management software. It's used to make many of the websites and applications you use every day. But what sets it apart is its flexibility; modularity is one of its core principles.

Its tools help you build the versatile, structured content that dynamic web experiences need. For any special functionality, one needs in his website can be attained by modules provided by Drupal.
Drupal makes it easy for the user to build a website. Drupal is one of the CMS because:

1. Drupal comes with the option of choosing from the variety of modules and third-party integrates that can be used for developing a website adhering to your preference.

2. Drupal comes with clean mark-up code out of the box that makes it easy for developers to manage content publishing.

3. The competency of Drupal lies in its versatility. Since it is open-source, any web developer can work on it and provide the user with numerous choices

CHAPTER-2

INTRODUCTION TO MIGRATION

We use the word "migration" as a term for any process that seeks to take data from some source to the current Drupal 8 site and use it to automatically create nodes, users, configuration, and any other component of our site. In short, automating what might otherwise be a tedious job of copying and pasting.
Drupal 8 brings a new migration system into Drupal core, with the goal of making it **easier to import data** from a variety of sources. The migrate system is both a framework designed to facilitate writing custom migrations, and an implementation of that framework aimed at Drupal-to-Drupal migrations.
The migration system makes it possible to pull content into Drupal from just about anywhere. The core API supports extraction from any SQL data source, including previous versions of Drupal. Contributed Modules extend this system to support other data types like CSV or JSON, as well as other platforms like WordPress.
Some of the existing data sources include:

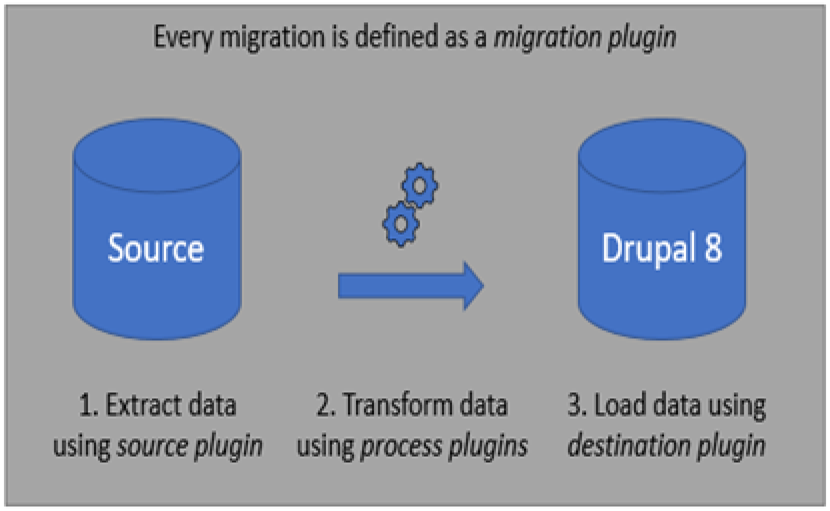
- MySQL
- Previous versions of Drupal
- CSV
- JSON,

- XML
- Etc.

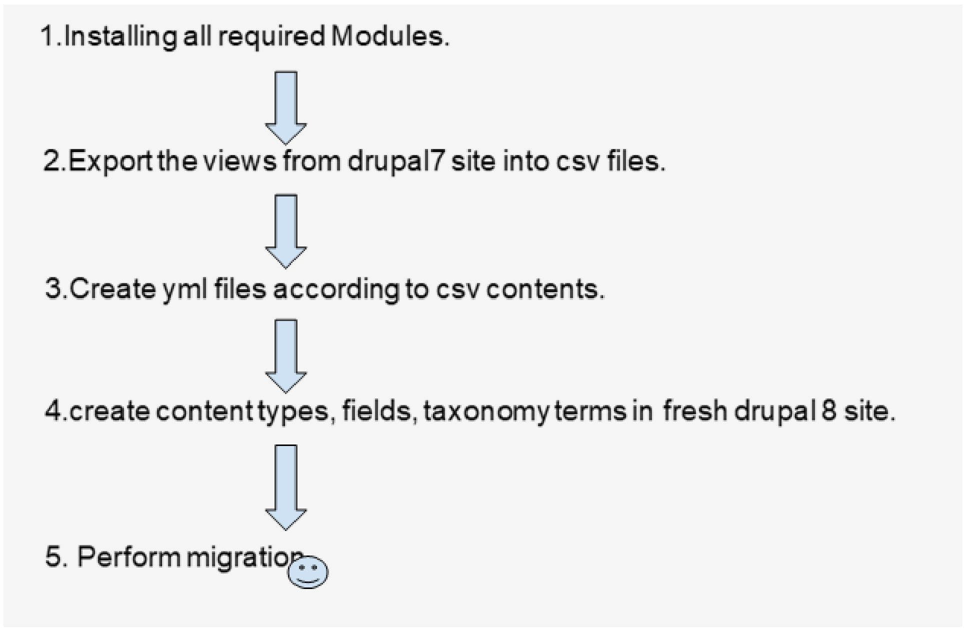
Some of the things we can create with a migration include:

- Content (nodes, taxonomy, any generic entity) including any attached files and images
- Content types
- User accounts
- Roles and permissions
- Simple configuration like the site name
- Complex configuration like image styles

The E-T-L process(Extract-Transform-Load)


Every migration is defined as a *migration plugin*
Source
Drupal 8
1. Extract data using *source plugin*
2. Transform data using *process plugins*
3. Load data using *destination plugin*

STEPS TO PERFORM THE MIGRATION.


1.Installing all required Modules.

2.Export the views from drupal7 site into csv files.

3.Create yml files according to csv contents.

4.create content types, fields, taxonomy terms in fresh drupal 8 site.

5. Perform migration

CHAPTER-3

Migration Modules

A **module is a set of PHP, JavaScript, and/or CSS files that extends site features and adds functionality**. We can turn the features and functionality on by *installing* the module, and we can turn it off by *uninstalling* the module; before uninstalling, you may need to remove data and configuration related to the feature or functionality. Each module that is installed adds to the time needed to generate pages on your site, so it is a good idea to uninstall modules that are not needed.

Modules needed for Migration

- Drupal 8 core ***Migrate*** module implements the general-purpose framework.
- Drupal 8 core *Migrate Drupal* module builds on that foundation to provide an upgrade path from Drupal 6 and Drupal 7 to Drupal 8.
- The contributed ***Migrate Source CSV*** module provides capabilities for migrating data to Drupal 8 from a simple CSV file.
- The ***Migrate_plus*** project provides extensions to core migration framework functionality, as well as examples.
- The ***Migrate Tools*** module provides tools for running and managing Drupal 8 migrations.

How to choose the version of the module to download

To download these modules, we can simply go to Drupal.org. To find the correct version of the module to download simply search the module on drupal.org and don't take the link of the yellow colour box version because that indicates in development may have same errors and may cause some problem in our site. So, choose the green colour box version as the green colour box means that it is developed and would not cause any additional errors to our site.

How to download through Drupal UI

The core module that we need for migration is migrate module, We do not need install the module as it is already installed but we just need to enable it by going into extend and checking on migrate and then enabling it and there are other modules that we need to install to do our migration. These modules can be downloaded from the Drupal UI by going into Extend then click on the install new modules here we can paste the link of the tar/zip file of the module we want to download.

We can install all the modules by the UI but we need to make sure we install the migrate Source CSV by composer as so it has all the dependencies it requires to download by composer we just need to write a simple command in the terminal “composer require ‘migrate_source_csv:3.x’”.
After installing the modules make sure to enable them by going into extend and checking the modules and then enabling them.

How to install modules using Command line

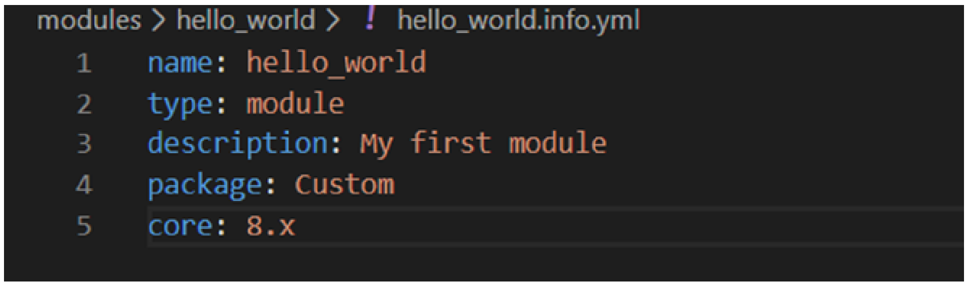
For installing modules in Drupal using command line, we simply have to perform the following command :

```
composer require drupal/<modulename>
```

While using Drush we have to make sure we are using correct version of Drush that is suitable for our Drupal version.
After installing the modules make sure to enable them by going into extend and checking the modules and then enabling them.

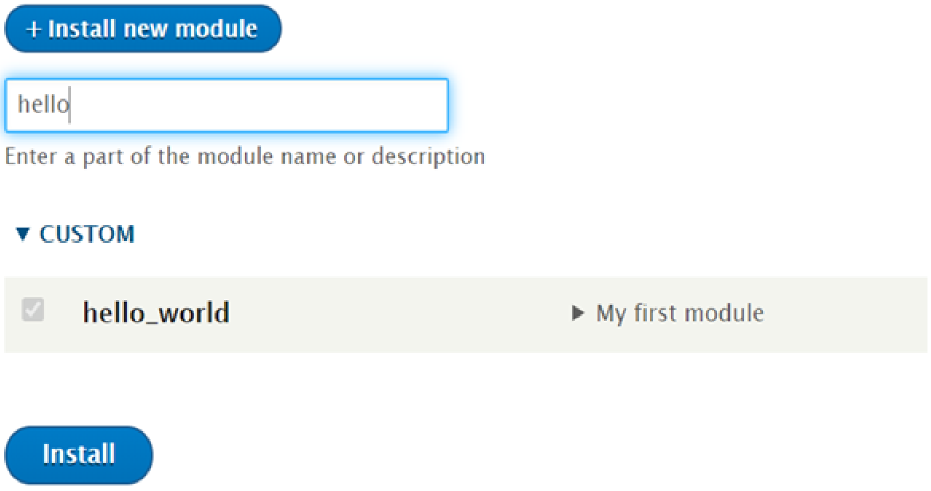
Custom module

≥create a new folder [module name]here, hello_world in drupal/modules and inside that create src/plugin/migrate/process/hello_world.info.yml


modules > hello_world > ! hello_world.info.yml

```yaml
name: hello_world
type: module
description: My first module
package: Custom
core: 8.x
```

after creating it go to drupal UI->extend->install new module->hello_world.Check the box shown.

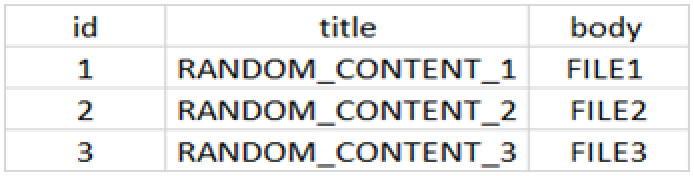


custom modules are required when we make custom source/process plugins(in our project we had made ExplodeArray plugin for use).

CHAPTER-4

INTRODUCTION to CSV

A **CSV File is a Comma Separated File** used to store data in a tabular format. It is an extension of a file, which may initially be in txt format or even xlsx format or any other format. The first line of a CSV File represents the header fields of the data we want to migrate. Below mentioned is a sample of a CSV File:



id	title	body
1	RANDOM_CONTENT_1	FILE1
2	RANDOM_CONTENT_2	FILE2
3	RANDOM_CONTENT_3	FILE3

Here, the header fields are ‘id’, ‘title’ and ‘body’. This CSV File contains 3 sets of data. For data migration, we have to paste the absolute path of this CSV File in the yml .

INTRODUCTION to YML

A YML file is a specific type of file, which is an extension to the YAML file, but the Drupal adds some extra information to it (by using some more key-value pairs) to make it more elaborative.
Below mentioned is a brief description of all the key-value pairs used in this yml:

● ‘id’: This key maps to the migration id. Please note that migration id for all migrations (which are present in our project) have to be all distinct. As otherwise, the system will throw an error. For each unique migration id, Drupal assigns a unique uuid to each migration.

● ‘label’: A label is simply assigned to a migration, so that the user can identify this migration from UI. This may be the same for different migration ids.

●‘migration_group’: By default, there is a migration group named ‘default’, which will contain all of your migrations. But we can make your custom migration group and use it for specific migration. In such cases, we have to write the exact machine name for this key-value pair.

The later section of the yml is further divided into 3 categories:

1. Source
2. Process
3. Destination

Here is a brief about all the 3 subsections and the associated terms:

● **Source**

This subsection contains information about the source of the data from
where the data has to be migrated.
1. ‘plugin’ : In this case this key is mapped to ‘csv’, as we will be extracting the data from a CSV file. 2. ‘path’ : This contains the absolute path of the CSV file.
3. ‘delimiter’ : This contains a specific character (generally ‘,’) , which is used to help distinguish Drupal between 2 adjacent fields.
4.‘enclosure’ : An enclosure is set to ‘”’ (double quotes), so that it by default encloses all the values mentioned in the CSV with double quotes. It can’t be set to single quotes (Drupal throws an error).
5. ‘header_offset’ : This key is mostly set to 0, as this describes the gap between the 1st line and the line containing the header fields in the CSV file.

6. ‘ids’: The above-mentioned syntax for this key has to be followed consistently, as it makes an array of all the data sets mentioned in the CSV and processes them one-by-one.

● **Process**

This subsection maps the data of the header fields to the corresponding destinations(fields) in your project. Thus, the name of each header field of the CSV file is mapped to the corresponding machine name of the field where the data is supposed to be injected.

- **Destination**

This part of the YAML does not consist much but generally a ‘plugin: entity’ which is set to ‘node’ for a content type. This is so, because all the migrations aimed at importing data for a content type are inserted in new ‘nodes’. This key can also be set to ‘user’ for user migration.

CHAPTER-5

Introduction

Plugins are much like PHP native interfaces with a little extra: the plugin system can discover every implementation of an interface (the default is magic namespacing), deals with metadata (by default this is provided by annotations)and provides a factory for the plugin classes.

Some plugin are that I used are:

Sub-process Plugin:

This plugin is very essential when we want to import multiple data in a specific multi-value field in a content type. Enabling this plugin iterates over all the data of a specific field and migrates them one-by-one.

Explode Plugin:

This plugin is again important for importing data in a multi-value field. This plugin creates an array of strings by splitting the source parameter on boundaries formed by the delimiter.

<u>Concat Plugin:</u>

The function of this plugin is to concatenate strings which previously have been separated (for example, used to obtain a path of source of file by forming a connected string of directories and subdirectories).

<u>Migration Lookup Plugin:</u>

This plugin helps us to look into the previous migrations and even extract some fields and properties into the new migration. The migration process maintains the relationships between source and destination identifiers in map tables, and this information is used because of the migration_lookup process plugin.

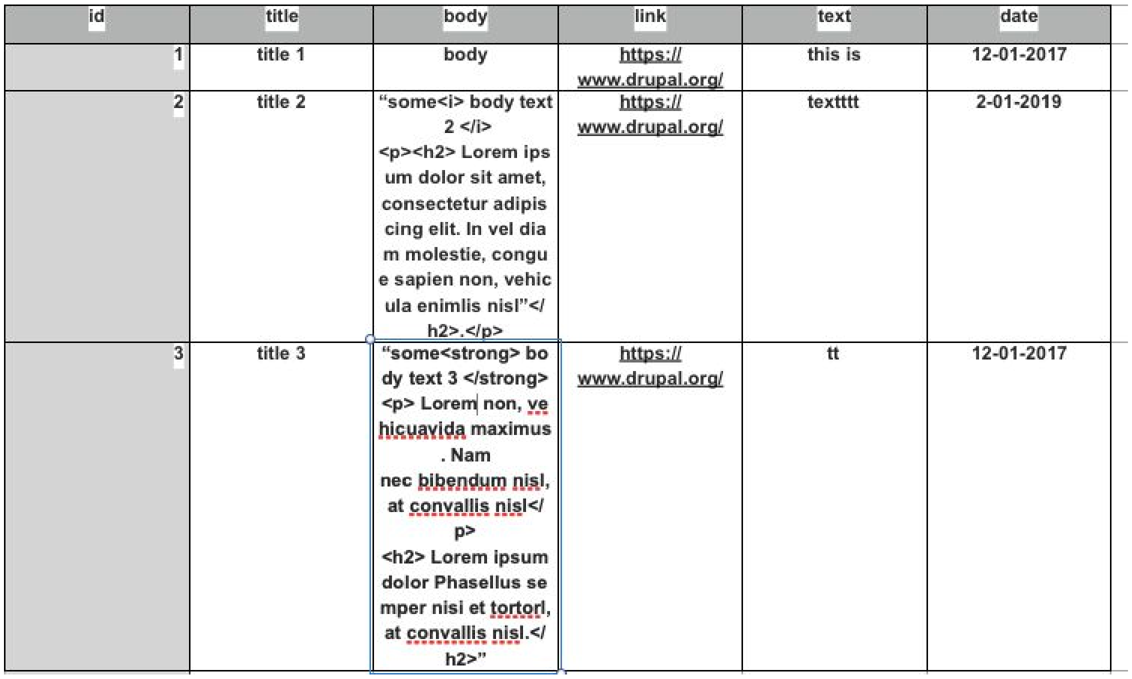
CHAPTER-6

MIGRATION USING DRUPAL UI

Migration in Drupal can be done from any method be it drush or Drupal UI. We just need to learn Drupal UI management for migration for using Drupal UI for migration. As we know migration can be done from any data source. So here is a demonstration of how to do migration from a CSV file.

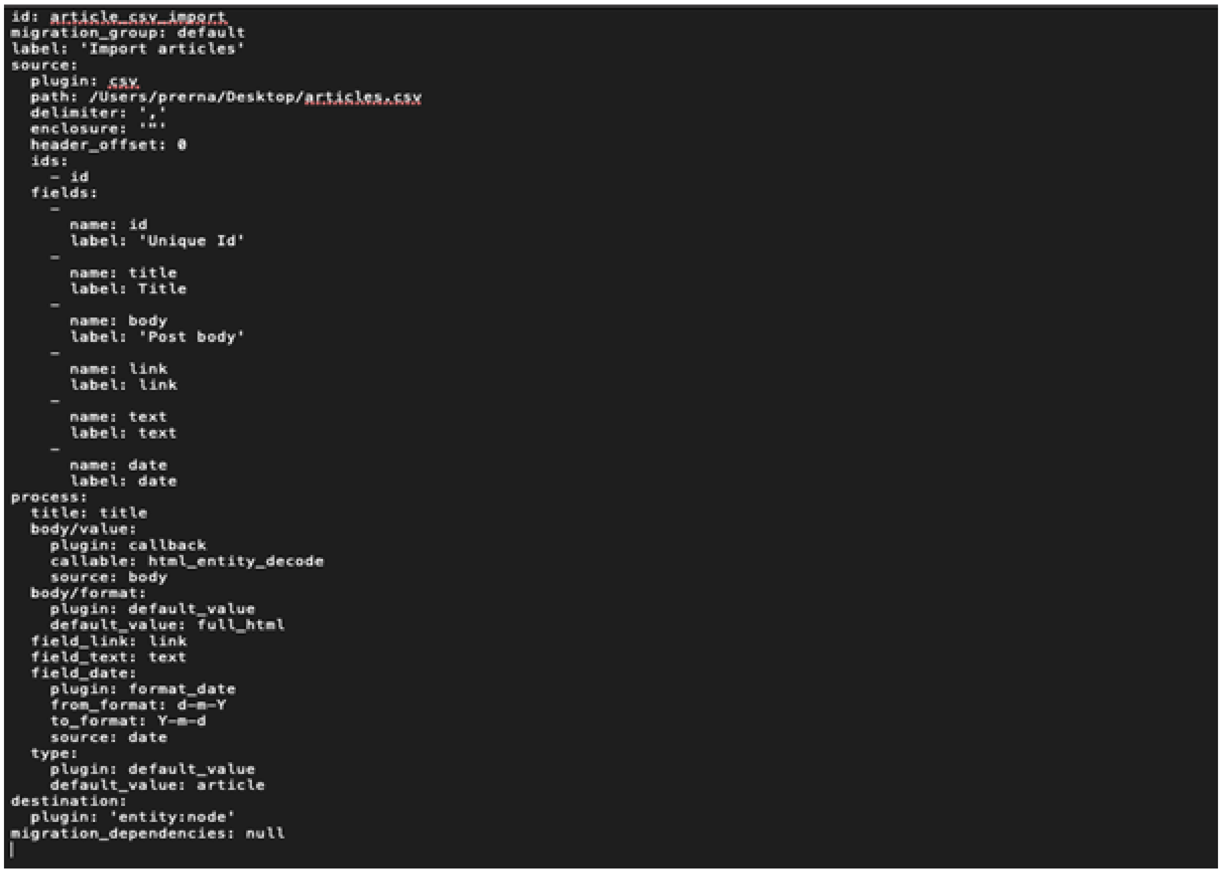
Migration from CSV data source

1. We can start with our pre-existing CSV file or create a new one For example:



id	title	body	link	text	date
1	title 1	body	https://www.drupal.org/	this is	12-01-2017
2	title 2	“some<i> body text 2 </i> <p><h2> Lorem ipsum dolor sit amet, consectetur adipiscing elit. In vel diam molestie, congue sapien non, vehicula enimlis nisl”</h2>.</p>	https://www.drupal.org/	texttt	2-01-2019
3	title 3	“some<strong> body text 3 </strong> <p> Lorem non, vehicuavida maximus . Nam nec bibendum nisl, at convallis nisl</p> <h2> Lorem ipsum dolor Phasellus semper nisi et tortorl, at convallis nisl.</h2>”	https://www.drupal.org/	tt	12-01-2017

2. Then we create the .yml for the csv and make sure we point the correct machine-name to the correct location in the yml and use proper indentation.The example of .yml file is as follows:



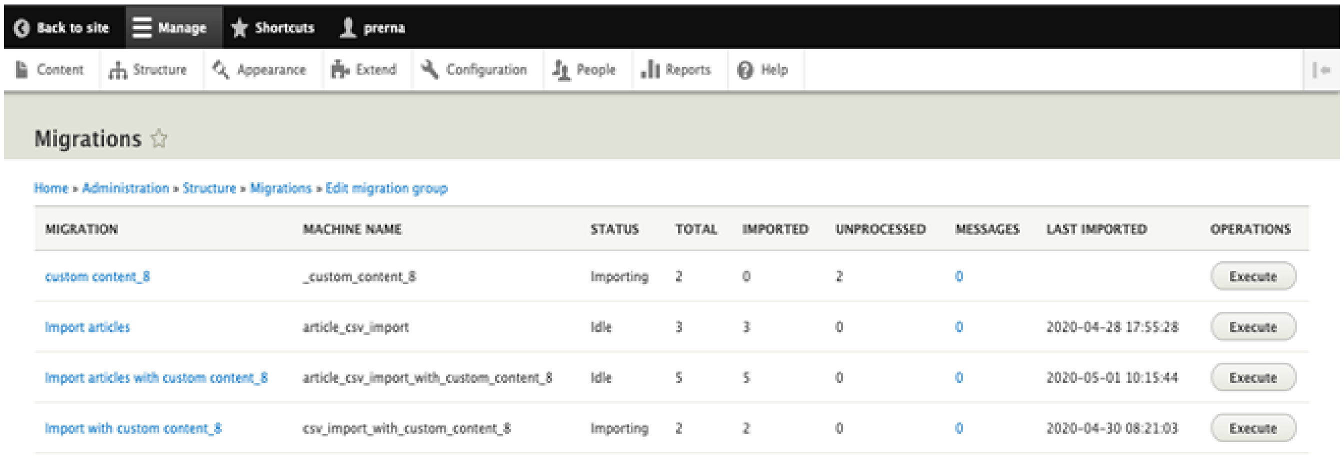
```
id: article_csv_import
migration_group: default
label: 'Import articles'
source:
  plugin: csv
  path: /Users/prerna/Desktop/articles.csv
  delimiter: ','
  enclosure: '"'
  header_offset: 0
  ids:
    - id
  fields:
    -
      name: id
      label: 'Unique Id'
    -
      name: title
      label: Title
    -
      name: body
      label: 'Post body'
    -
      name: link
      label: link
    -
      name: text
      label: text
    -
      name: date
      label: date
process:
  title: title
  body/value:
    plugin: callback
    callable: html_entity_decode
    source: body
  body/format:
    plugin: default_value
    default_value: full_html
  field_link: link
  field_text: text
  field_date:
    plugin: format_date
    from_format: d-m-Y
    to_format: Y-m-d
    source: date
  type:
    plugin: default_value
    default_value: article
destination:
  plugin: 'entity:node'
migration_dependencies: null
```

Now put the name of the migration group in which you want the imported .yml to go in migration_group .**Delimiter** is the symbol like here" **,**" from which we differentiate between two fields of CSV. **Id** of each migration needs to be unique and after importing unique uuid is generated for each migration.

Go to configuration/configuration -> synchronization/import -> single import -> paste our .yml -> import it.

3. Now go to structure -> choose migration, -> migration group which you mentioned in the migration_group. There, you will be able to see our imported file.

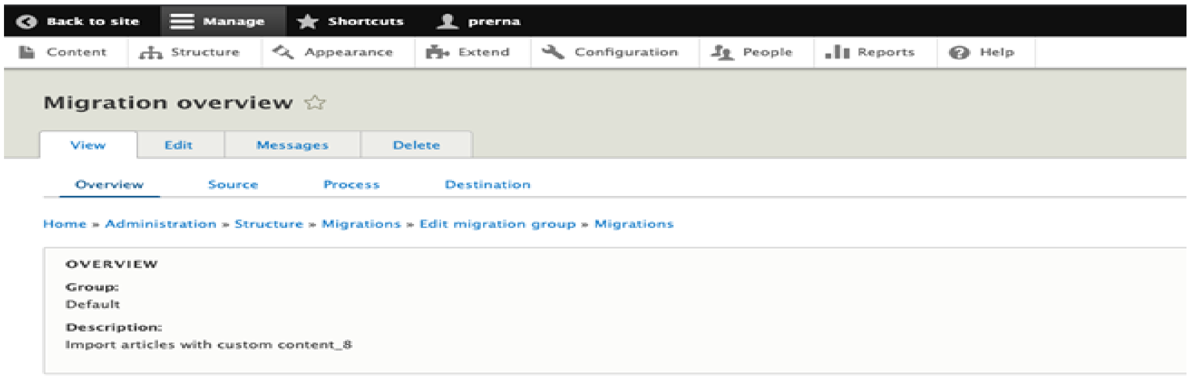
Now below we can see all the imported .yml files and we can choose to execute the file we want. In the messages below we can see the errors that happen while executing. If we want to delete an imported migration we should **first do rollback** then click on the messages then there we can get an option to **delete** the migration.



Migrations

Home » Administration » Structure » Migrations » Edit migration group

MIGRATION	MACHINE NAME	STATUS	TOTAL	IMPORTED	UNPROCESSED	MESSAGES	LAST IMPORTED	OPERATIONS
custom content_8	_custom_content_8	Importing	2	0	2	0		Execute
Import articles	article_csv_import	Idle	3	3	0	0	2020-04-28 17:55:28	Execute
Import articles with custom content_8	article_csv_import_with_custom_content_8	Idle	5	5	0	0	2020-05-01 10:15:44	Execute
Import with custom content_8	csv_import_with_custom_content_8	Importing	2	2	0	0	2020-04-30 08:21:03	Execute



Migration overview

Home » Administration » Structure » Migrations » Edit migration group » Migrations

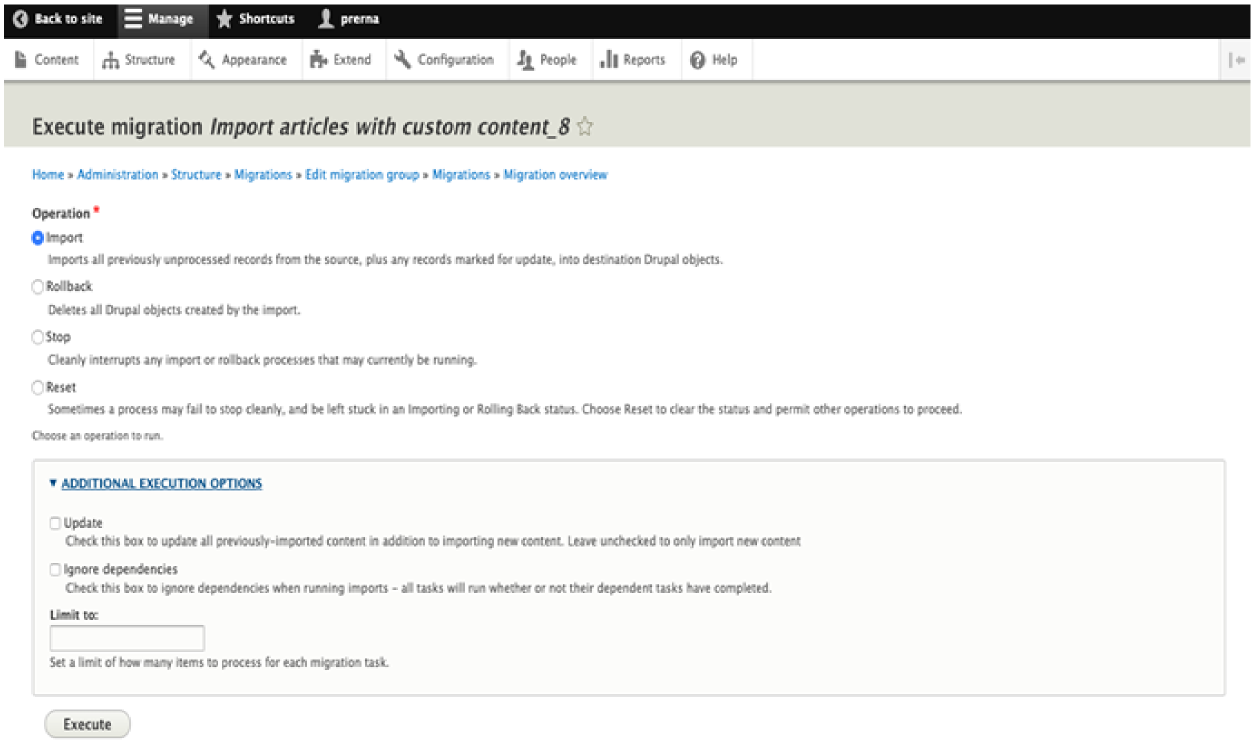
OVERVIEW

Group:
Default

Description:
Import articles with custom content_8

By clicking on the migration name, we can see we can have the migration overview,see its details or delete the migration

4. Now to import the migration choose execute and then choose import and execute. Now your migration is complete.



Execute migration *Import articles with custom content_8*

Home » Administration » Structure » Migrations » Edit migration group » Migrations » Migration overview

Operation *

[✓ Import]
Imports all previously unprocessed records from the source, plus any records marked for update, into destination Drupal objects.

Rollback
Deletes all Drupal objects created by the import.

Stop
Cleanly interrupts any import or rollback processes that may currently be running.

Reset
Sometimes a process may fail to stop cleanly, and be left stuck in an Importing or Rolling Back status. Choose Reset to clear the status and permit other operations to proceed.

Choose an operation to run.

▾ ADDITIONAL EXECUTION OPTIONS

Update
Check this box to update all previously-imported content in addition to importing new content. Leave unchecked to only import new content

Ignore dependencies
Check this box to ignore dependencies when running imports – all tasks will run whether or not their dependent tasks have completed.

Limit to:

Set a limit of how many items to process for each migration task.

Execute

Now here we can see we have the option to do import, reset, stop,rollback, update our previous import.

CHAPTER-7

Introduction to Drush

Drush is a **command line shell and scripting interface** for Drupal. It is an alternative to the migration performed using the UI of Drupal. Drush provides a list of commands that can be executed from the CLI itself.

Drush installation

For installing Drush, the most convenient way is to use "Composer" for performing this task! Composer is a **PHP package management tool** to help manage your project or libraries' dependencies.
For downloading Drush, we simply can perform either of the 2 tasks:

1. Run 'composer require drush/drush'. This will enable you to work with Drush for a specific project.
2. Run ' composer global require drush/drush'. This will enable you to work with Drush for all projects on your machine.

Also, made sure the compatibility of the versions of Drush compatible with the Drupal version of their project. I now made a CSV data source as mentioned earlier. Then as per the earlier mentioned steps, I have to paste your yml(or YAML) file in the path stated. Now we have to execute the following commands step-by-step in the CLI inside the directory of your local Drupal project:

1. `drush migrate-status` (This command displays the Migration Status of several migration ids).

2. `drush migrate-import <migration-id>` (Mention the migration id as mentioned in the yml file without quotes).

3. After we have executed the migration, verify the result on your Drupal site. If we find out that we need to modify the migration a bit, we can roll back the migration with the 'migrate-rollback' Drush command.

`drush migrate-rollback <migration-id>.`

4. In the worst case, if the site crashes due to migration, then we can even perform **`drush cdel migrate_plus.migration.<migration-id>`** to completely remove the imported configuration. But this must follow the rollback step and should only be executed when the site stops to respond and function.

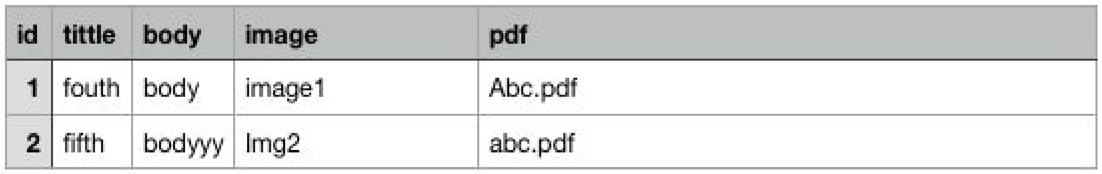
5. If we opted for deleting the configuration, then you also will have to rebuild the cache of your site. For that run `drush cr`.

In this way, we can successfully perform migration using drush for a Drupal project.

Image and file migration

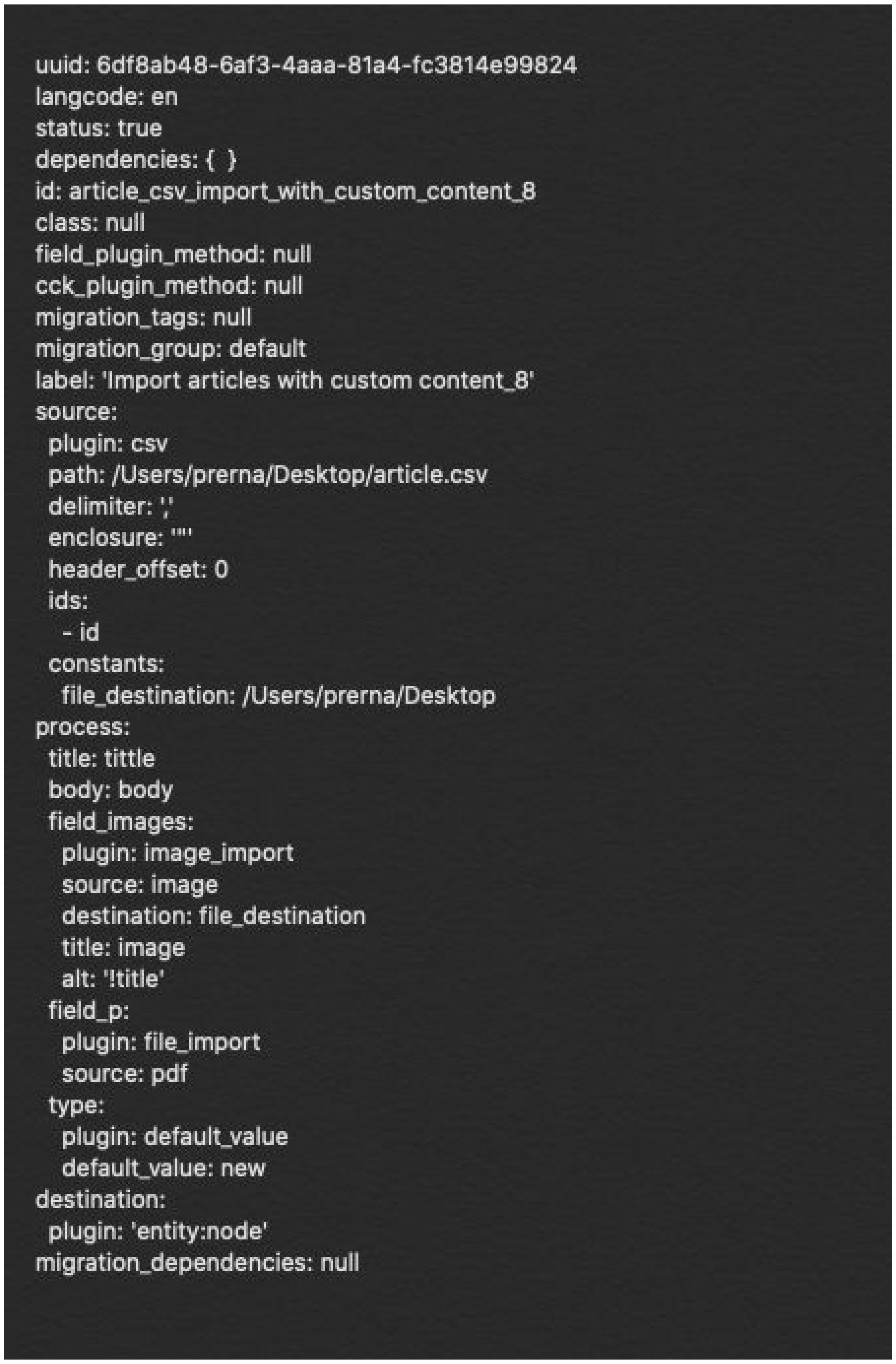
For image and file importing, we have to install a separate module '**Migrate Files (extended)'**. We can install this module by UI or by Drush

The CSV for image and file import is



id	tittle	body	image	pdf
1	fouth	body	image1	Abc.pdf
2	fifth	bodyyy	Img2	abc.pdf

The yml for image and file import is



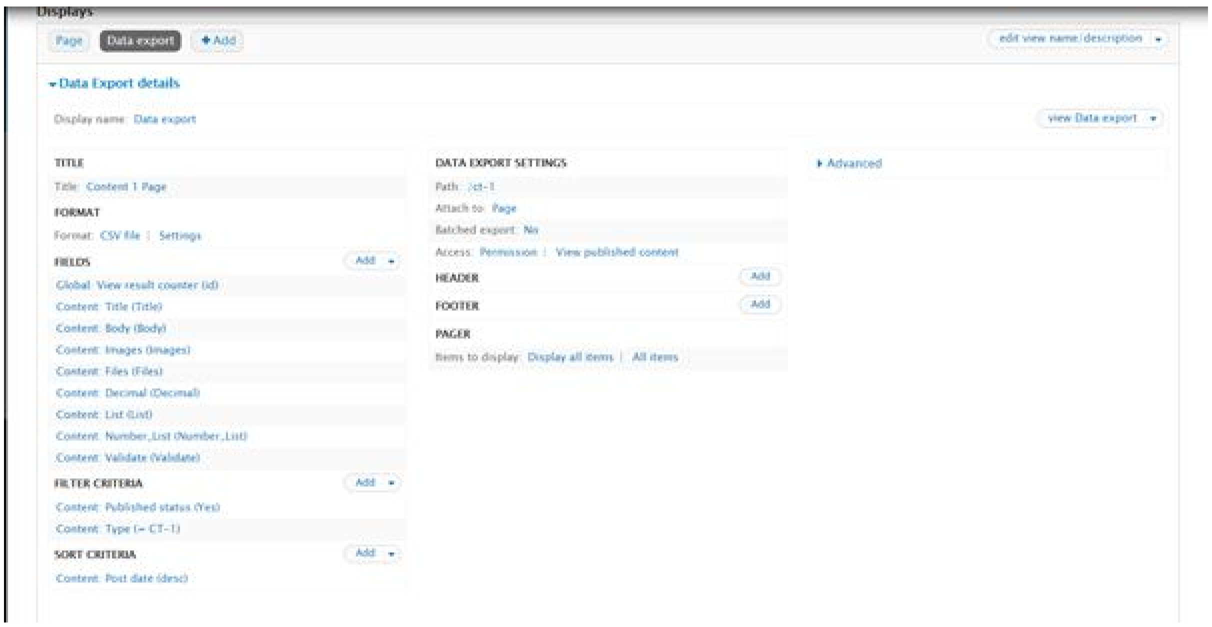
```
uuid: 6df8ab48-6af3-4aaa-81a4-fc3814e99824
langcode: en
status: true
dependencies: { }
id: article_csv_import_with_custom_content_8
class: null
field_plugin_method: null
cck_plugin_method: null
migration_tags: null
migration_group: default
label: 'Import articles with custom content_8'
source:
  plugin: csv
  path: /Users/prerna/Desktop/article.csv
  delimiter: ','
  enclosure: '"'
  header_offset: 0
  ids:
    - id
  constants:
    file_destination: /Users/prerna/Desktop
process:
  title: tittle
  body: body
  field_images:
    plugin: image_import
    source: image
    destination: file_destination
    title: image
    alt: '!title'
  field_p:
    plugin: file_import
    source: pdf
  type:
    plugin: default_value
    default_value: new
destination:
  plugin: 'entity:node'
migration_dependencies: null
```

CHAPTER -8

MIGRATION FROM DRUPAL 7 TO 8

VIEWS DATA EXPORT MODULE

After learning about how migration is performed using different plugins and yml files, how can we export data from one Drupal site to another Drupal site? For that there is a module named '**Views Data Export**', which converts data of different content types and places them in a csv file format (data can be fetched in other formats like .txt,.json,etc.) . After we download this CSV file from the Drupal site, we can follow the same procedure (as written earlier) for importing data through a csv file and using a suitable yml file. But before importing the data, similar content-types should be created in the Drupal site (in which we want to import the data) like the content-types in the older site, otherwise some data would not be imported. Different version releases of this module and its specifications can be found on https://www.drupal.org/project/views_data_export. For using this module in **Drupal 7**, we have to use the **views module** in our site. Since Drupal 7 does not have views as one of the core modules, we have to install the 'views' module, which also has a pre-requisite of the 'CTools' module. After installing these modules and enabling them, we can add **data export** option from the views of a specific content-type.


Displays

Page | Data export | + Add | edit view name/description

Data Export details

Display name: Data export | view Data export

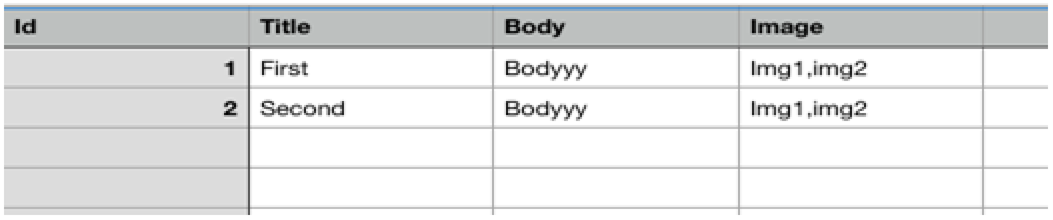
TITLE
Title: Content 1 Page
FORMAT
Format: CSV file | Settings
FIELDS
Global: View result counter (id)
Content: Title (Title)
Content: Body (Body)
Content: Images (Images)
Content: Files (Files)
Content: Decimal (Decimal)
Content: List (List)
Content: Number_List (Number_List)
Content: Validate (Validate)
FILTER CRITERIA
Content: Published status (Yes)
Content: Type (= CT-1)
SORT CRITERIA
Content: Post date (desc)

DATA EXPORT SETTINGS
Path: /ct-1
Attach to: Page
Batched export: No
Access: Permission | View published content
HEADER
FOOTER
PAGER
Items to display: Display all items | All items

Advanced

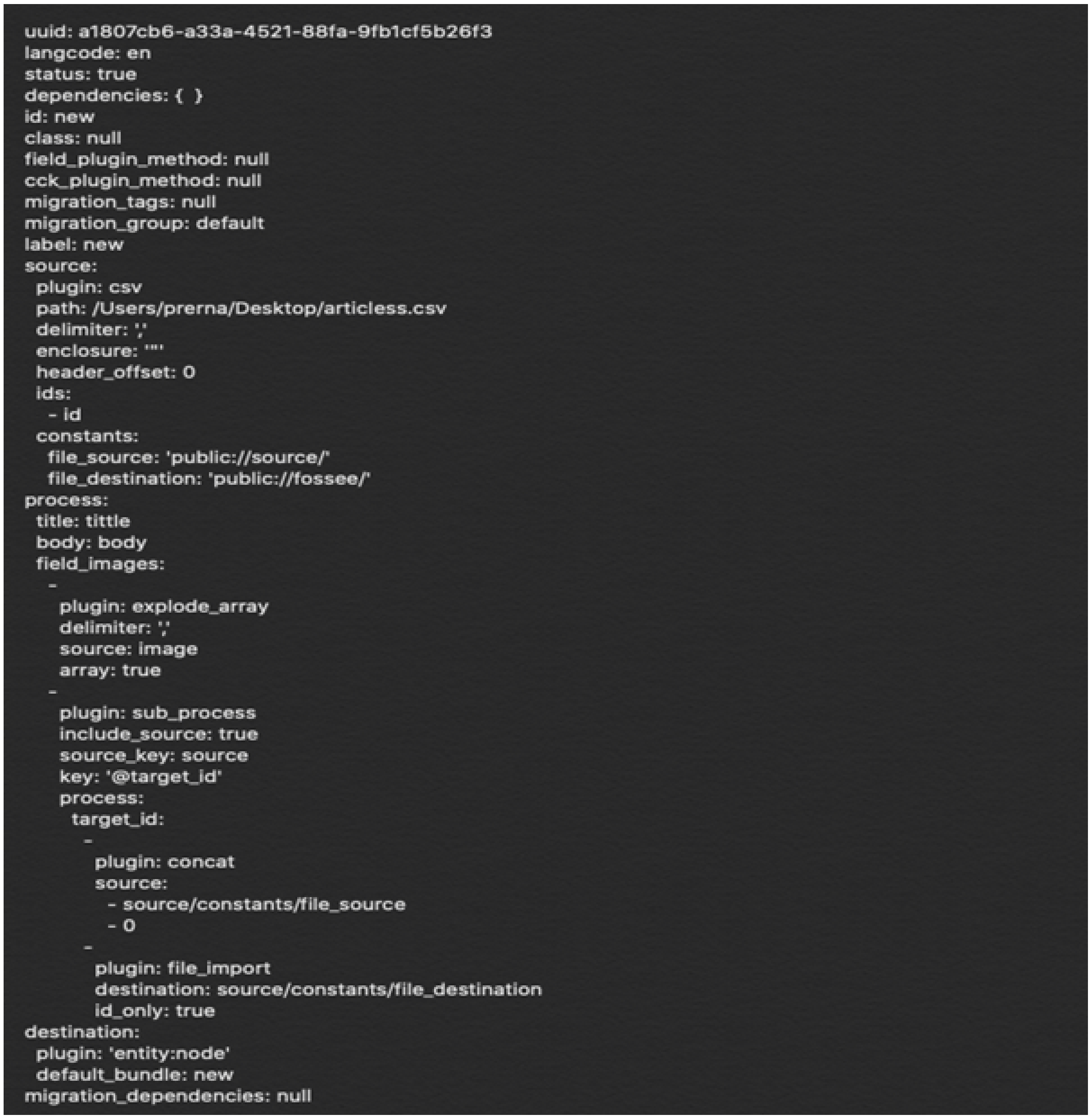
Here we can define the path from where we can download the CSV file for data export.Similarly, data of all content types can be exported using this module, and also can be imported using yml files and CSVs.

For importing multiple image/file fields: we use the **Subprocess plugin** but the subprocess plugin requires the input in the form of array so we have to create the array of files/image we want to import. For that we used the ExplodeArray **process plugin** which was made by the help of our mentors. The namespace of the plugin should be the location of the process plugin,like here we are putting our custom plugin in a custom module so namespace will be drupal\hello_world\plugin\migrate\process and then the code.
the csv for multiple image import is



Id	Title	Body	Image	
1	First	Bodyyy	Img1,img2	
2	Second	Bodyyy	Img1,img2	

The yml used for migration of multiple images is



```
uuid: a1807cb6-a33a-4521-88fa-9fb1cf5b26f3
langcode: en
status: true
dependencies: { }
id: new
class: null
field_plugin_method: null
cck_plugin_method: null
migration_tags: null
migration_group: default
label: new
source:
  plugin: csv
  path: /Users/prerna/Desktop/articless.csv
  delimiter: ','
  enclosure: '"'
  header_offset: 0
  ids:
    - id
  constants:
    file_source: 'public://source/'
    file_destination: 'public://fossee/'
process:
  title: tittle
  body: body
  field_images:
    -
      plugin: explode_array
      delimiter: ','
      source: image
      array: true
    -
      plugin: sub_process
      include_source: true
      source_key: source
      key: '@target_id'
      process:
        target_id:
          -
            plugin: concat
            source:
              - source/constants/file_source
              - 0
          -
            plugin: file_import
            destination: source/constants/file_destination
            id_only: true
destination:
  plugin: 'entity:node'
  default_bundle: new
migration_dependencies: null
```

Here in the sub process plugin we have used two more plugin **concat** to concatenate the destination and **file import** to import the file/image field.
Here we are migrating into custom content type namely new so we have that in destination
After importing this yml we can simply go to the migration UI and execute it and our migration will be done

REFERENCE

- Drupal
- Drupal stack exchange
- OS training