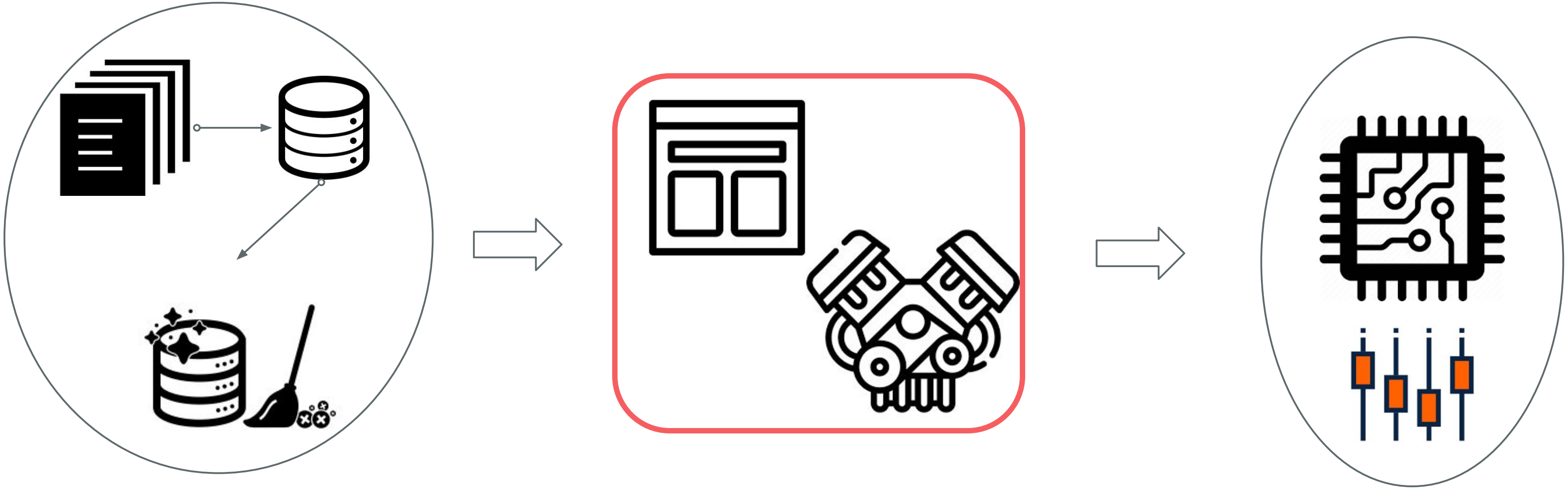
Wiki Article Generation

Templating and XML Generation

Overviews

Pipeline



Data

- Collection
- Structuring
- Cleaning

Template

- Manually
- Engine (jinja2)
- Render

Article Creation

- Tweaking
- XML Creation
- Import in wiki

Templating

1. Manually Writing

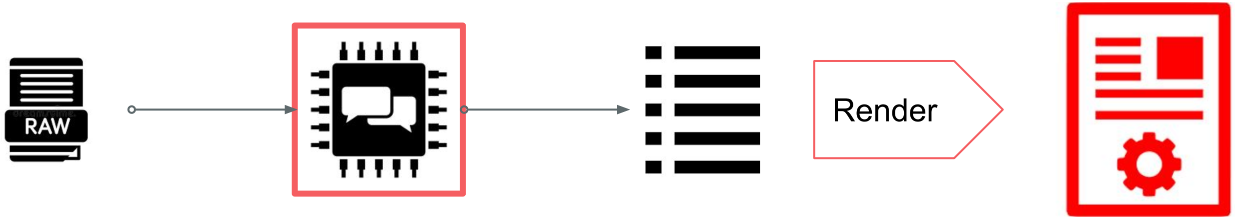
   First, we decide what all columns/attributes are important
   Then we write down a template manually. Add variations !

2. Template Engine

   Writing the same template using a templating engine, like Jinja2

3. Text Processing and Template Rendering

   Providing **data** to the template and generating text



Text Processing

- Corner Cases

If not done already, we have thoroughly go through the data, understand all the corner cases and handle them accordingly in the template. This can be automated and made easy using the sweetviz library !

- Translation and Transliteration

Now, we can transform the data we have to an indian language. For this, we can use any library available. The ones that were smart in handling all cases and worked for me are anuvaad and deeptranslit.

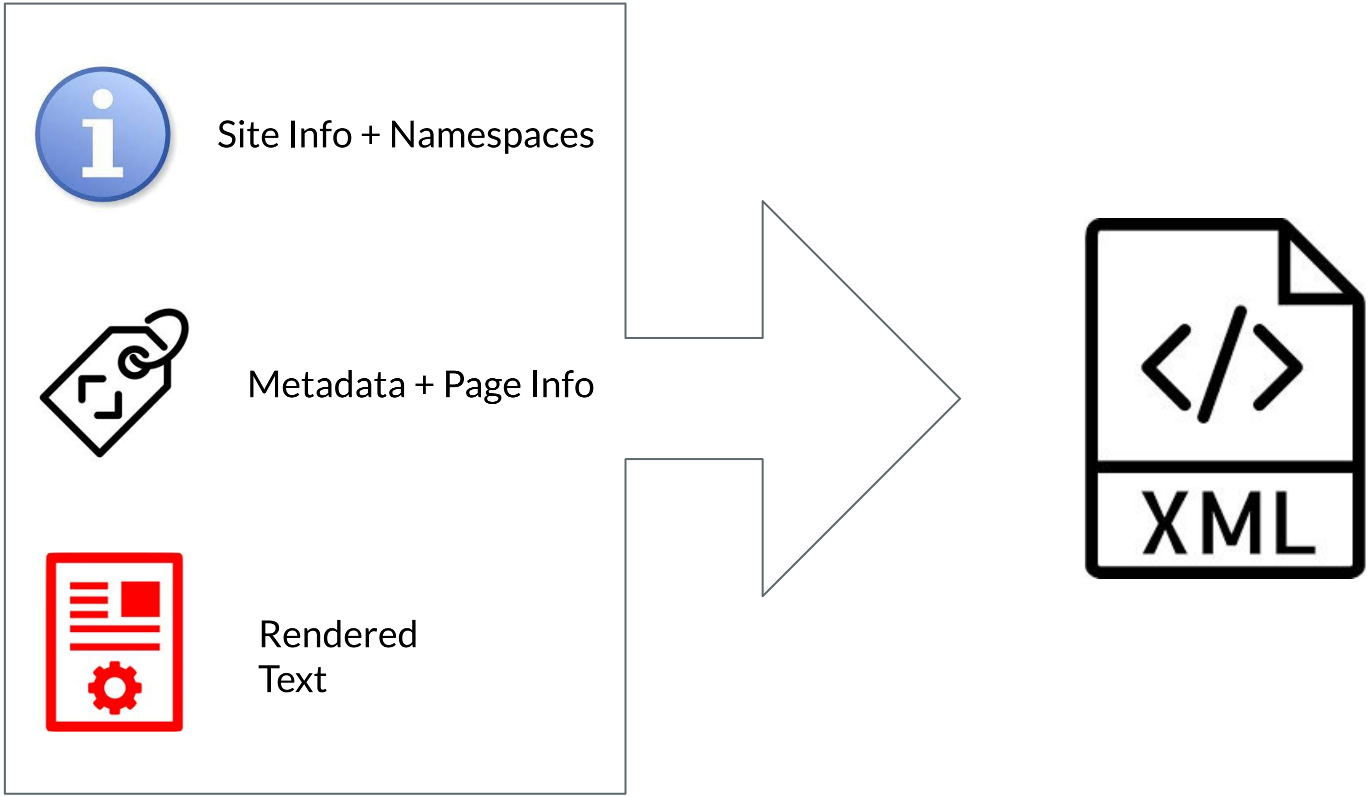
★ Numbers

○ We have to handle numbers separately if we want a text representation of numerical data

★ Exception Words

○ There could be few high frequency words, which are translated or transliterated incorrectly by the libraries. We could make a dictionary and of such words and their respective translations/transliterations and handle them separately.

XML Generation



Site Info + Namespaces

Metadata + Page Info

Rendered
Text

Details

Virtual Environment Setup

1. Python 3.7

   Highly recommend python3.7 to avoid any and all issues with anuvaad and deeptranslit.

2. Venv

   ```
   $ pip install virtualenv
   
   $ sudo apt install python3.7
   
   $ virtualenv -p python3.7 env_name
   ```

3. Requirements.txt

   ```
   $ source env_name/bin/activate (ubuntu)/ source env_name\Scripts\activate (windows)
   
   $ pip install -r requirements.txt
   ```

Sweetviz

Exploratory Data Analysis

```
import sweetviz as sv


report = sv.analyze([dataframe])

report.show_html()
```

Jinja

Template Syntax

{# A Comment #}

{{ variable }}

{% conditionals/tags %}

{%- escaping white spaces -%}

{% macro name(parameters) %}

{% endmacro %}

{% set variable=value %}

{{‘{{ escape braces }}’}

Anything literally

Jinja

Rendering

```
from jinja2 import Environment,
FileSystemLoader


file_loader = FileSystemLoader(‘dir_path’)

env = Environment(loader=file_loader)

template = env.get_template(‘name.j2’)


text = template.render(data)
```

Anuvaad

Translation

```
from anuvaad import Anuvaad

telugu = Anuvaad('english-telugu')


telugu.anuvaad(‘Cars’)
```

DeepTranslit

Transliteration

```
from deeptranslit import DeepTranslit

translit= DeepTranslit('telugu').transliterate



result = translit(‘Cars’)[0]

print( result[‘pred’], result[‘prob’] )
```

Helpful Libraries/Website

- Dataframe
- Sweetviz

- లేఖిని , IndiaTyping
- Jinja

- Anuvaad
- Deeptranslit

ToyProject

Please find the code for the ToyProject here:

- https://github.com/indicwiki-iiit/schools/tree/main/ToyProject

Thank You

Hope it helped !