# Architecture and Design - News Article Category Prediction (Week 3 - Milestones)

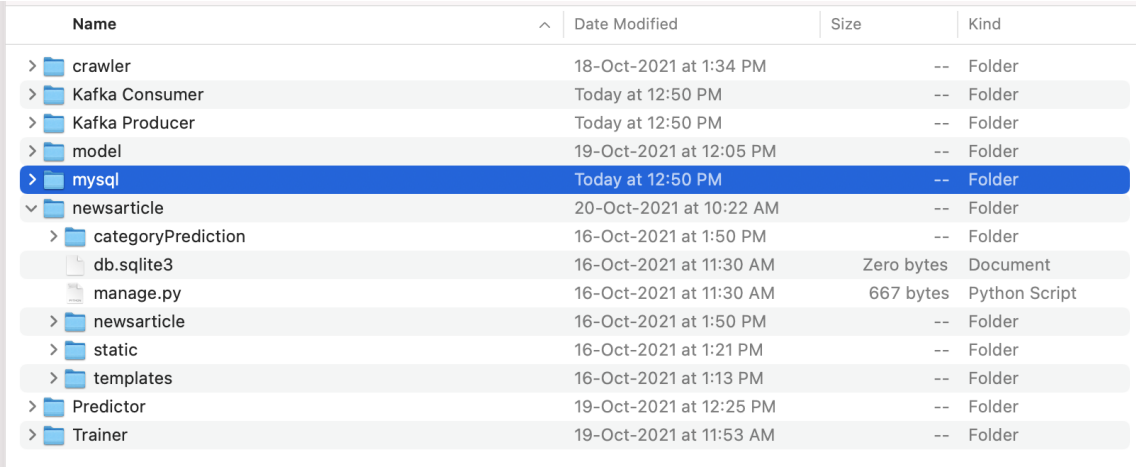
## Objective

This document is about classifying News Articles into categories - With information overload today users are inundated with news articles of all topics, even the ones which may not be relevant to users. Design a system which can classify incoming news articles and appropriately tag the corresponding category.

## Main Software Components

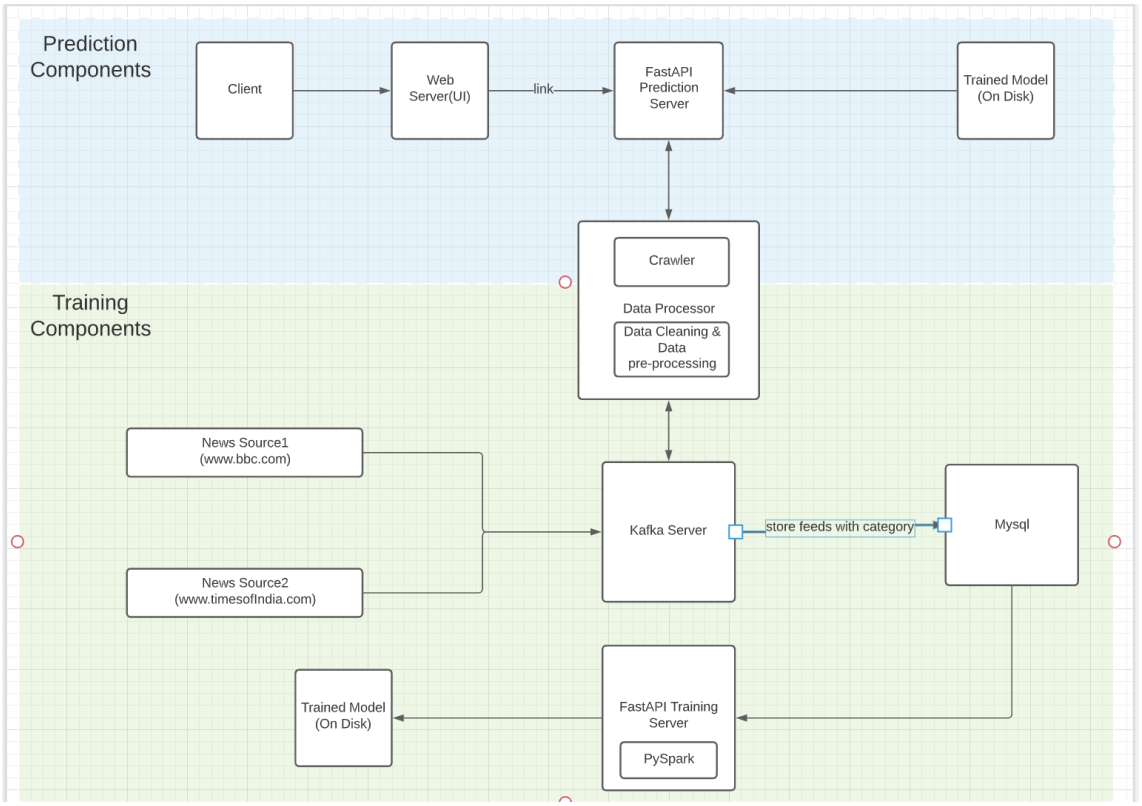
- html, css, js
- django
- python FastAPI
- Crawler(Beautiful Soup)
- pyspark
- Kafka
- hadoop
- mysql
- NLP libraries

## Project Setup




- All the packages and services are accumulated in the docker-compose(yaml) file.

## Architecture



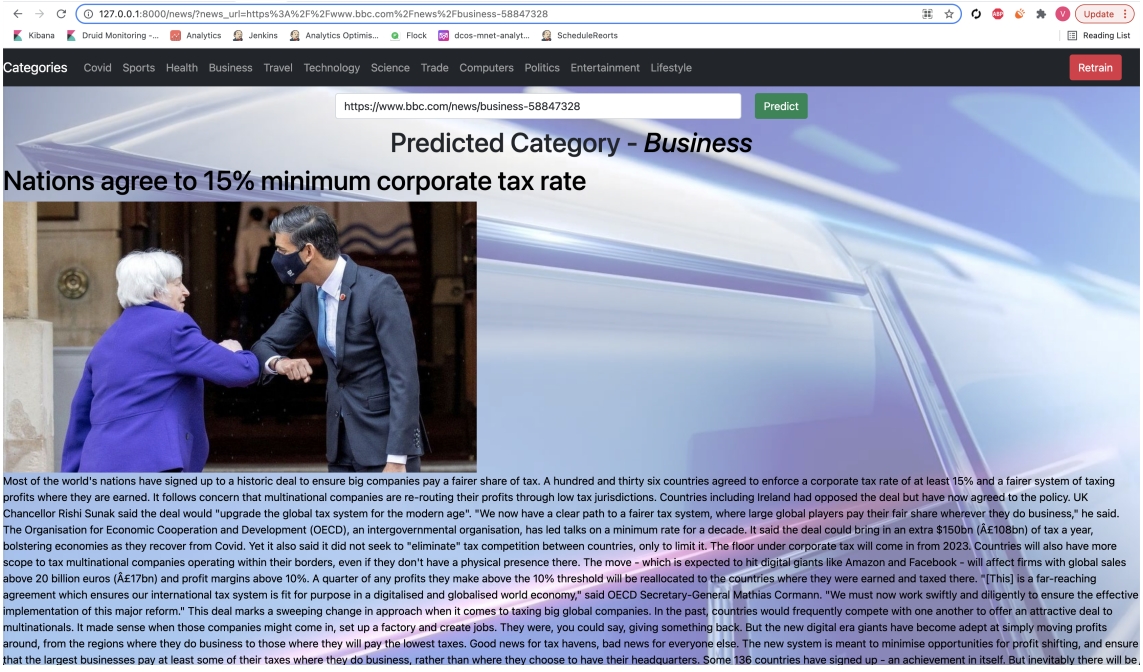

## UI Server(django)

### Technologies

- We are using django server as a frontend web server.
- HTML, CSS, bootstrap for rendering webpages.
- Javascript for input validations.

### Functionality

- User can enter url or article in the input text box.
- Then he can click on `Predict` button for category predections.
- It will get the article from the crawler and display it on the page.
- It will also the predictor api which returns the predicted category.
- `Retrain` button is also provided to retrain the model, it will call the Trainer API for the same.




## Categories Supported

```
{'business':1,'computers':2,'covid':3,'entertainment':4,'health':5,'lifestyle':6
,'politics':7,'science':8,'sport':9,'technology':10,'trade':11,'travel':12}
```

## Segregate the preprocessed data into training and test

```
X_train, X_test, y_train, y_test = train_test_split(
    news_df['article'],
    news_df['category'],
    random_state = 1
)
```

## Complete model training and deployment (and model registry)

- We are using MLflow for model training and deployment.
- It will serialize the model and store it as the artifacts.

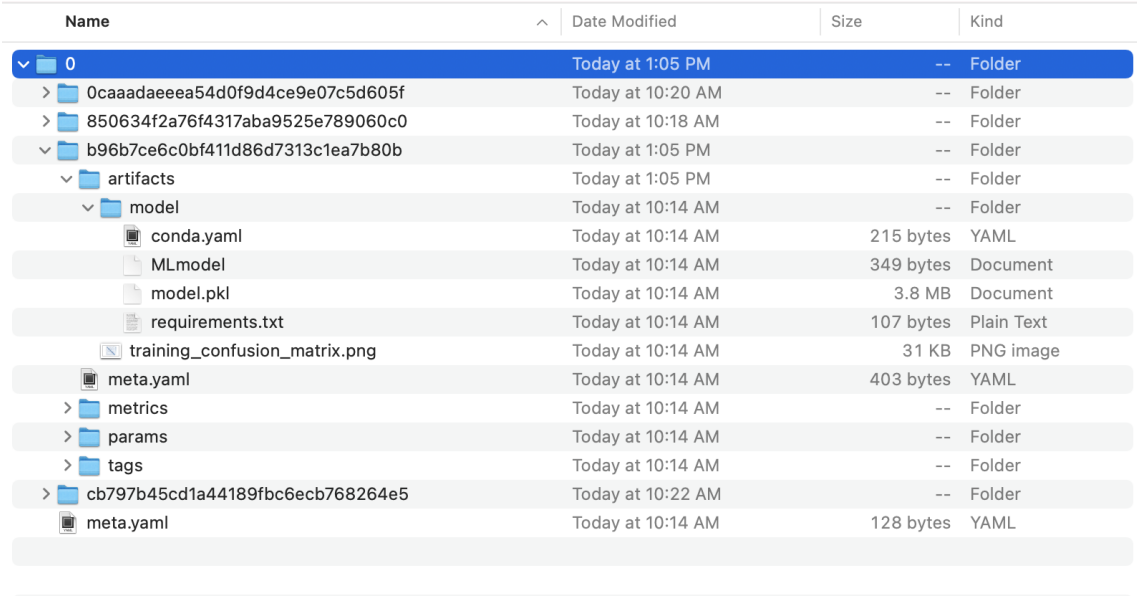
```
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.base import TransformerMixin, BaseEstimator, clone
import mlflow
import mlflow.sklearn

with mlflow.start_run(run_name='complaint_classifier_sklearn'):
```

```
  # get mlflow run Id
  run_id = mlflow.active_run().info.run_id

  # Text preprocessing, tokenizing and filtering of stopwords are all included in
CountVectorizer
  count_vect = CountVectorizer(ngram_range=(2,2), min_df=3)
  tf_transformer = TfidfTransformer(use_idf=True)
  clf = MultinomialNB()

  # define pipeline
  pipeline = Pipeline([
    ('vect', count_vect),
    ('tfidf', tf_transformer),
    ('clf', clf)
  ])

  # Train pipeline
  pipeline.fit(X_train, y_train)
  y_pred = pipeline.predict(X_test)
  accuracy = accuracy_score(y_pred, y_test)

  # Log pipeline to mlflow
  mlflow.sklearn.log_model(pipeline, "pipeline")
  mlflow.log_metric("accuracy", accuracy)

  # Log classification report
  clsf_report = pd.DataFrame(classification_report(y_test, y_pred, target_names=
[str(i) for i in range(1,13)], output_dict=True)).transpose()
  clsf_report.to_csv('classification_report.csv', index=True)
  mlflow.log_artifact('classification_report.csv')

clsf_report
```

**Serialize the re-trained model.**

- Trained models are logged as MLflow model artifacts.

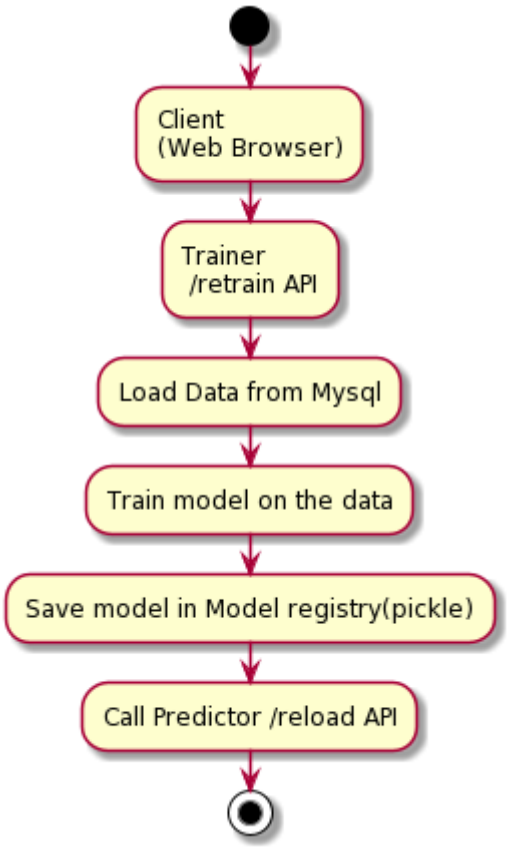
**Push this model to model-registry (MLFlow) along with hyperparameters**



| Name | Date Modified | Size | Kind |
|---|---|---|---|
| 0 | Today at 1:05 PM | -- | Folder |
| 0caaadaeeea54d0f9d4ce9e07c5d605f | Today at 10:20 AM | -- | Folder |
| 850634f2a76f4317aba9525e789060c0 | Today at 10:18 AM | -- | Folder |
| b96b7ce6c0bf411d86d7313c1ea7b80b | Today at 1:05 PM | -- | Folder |
| artifacts | Today at 1:05 PM | -- | Folder |
| model | Today at 10:14 AM | -- | Folder |
| conda.yaml | Today at 10:14 AM | 215 bytes | YAML |
| MLmodel | Today at 10:14 AM | 349 bytes | Document |
| model.pkl | Today at 10:14 AM | 3.8 MB | Document |
| requirements.txt | Today at 10:14 AM | 107 bytes | Plain Text |
| training_confusion_matrix.png | Today at 10:14 AM | 31 KB | PNG image |
| meta.yaml | Today at 10:14 AM | 403 bytes | YAML |
| metrics | Today at 10:14 AM | -- | Folder |
| params | Today at 10:14 AM | -- | Folder |
| tags | Today at 10:14 AM | -- | Folder |
| cb797b45cd1a44189fbc6ecb768264e5 | Today at 10:22 AM | -- | Folder |
| meta.yaml | Today at 10:14 AM | 128 bytes | YAML |

**Convert the project to an On-Demand service, to be able to retrain the model as needed. This could be a flask API that could trigger the entire pipeline in model-training-service (loading data into PySpark, preprocessing, model training)**

- `Retrain` API is created on the Trainer service.
- The service will get the articles from the mysql db and train the model.
- This model is then logged into the model registry by MLflow.
- Trainer will then inform the predictor to reload the model.




{.callout}

## Crawler

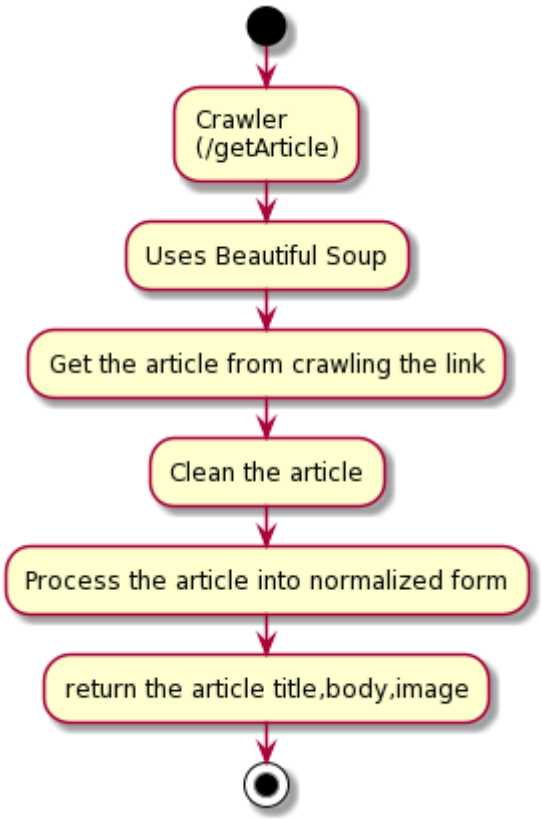
- There are mainly three components of the Crawler:

### 1. Crawler Scheduler

- Scheduler runs the cron job which will run once in a day.
- It will get the latest links from the google search web page.
- And then it pushes it to mysql db.

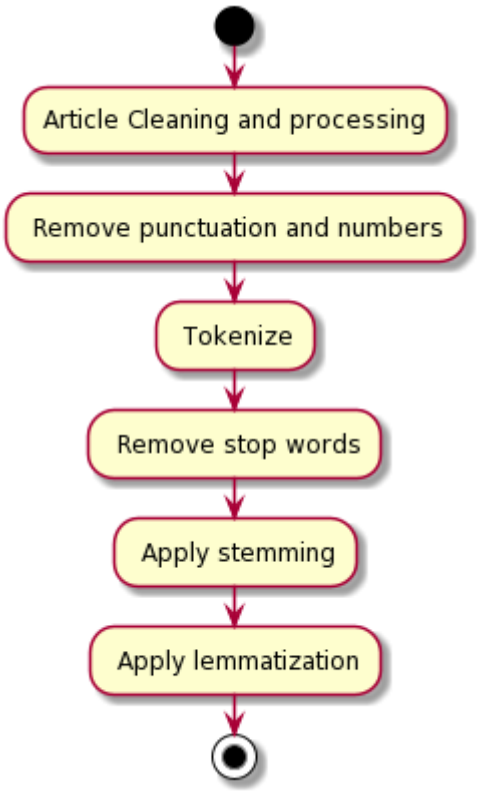

**2. Crawler getArticle API**

- This API will crawl the webpage, given the article link. and returns the article along with the image.



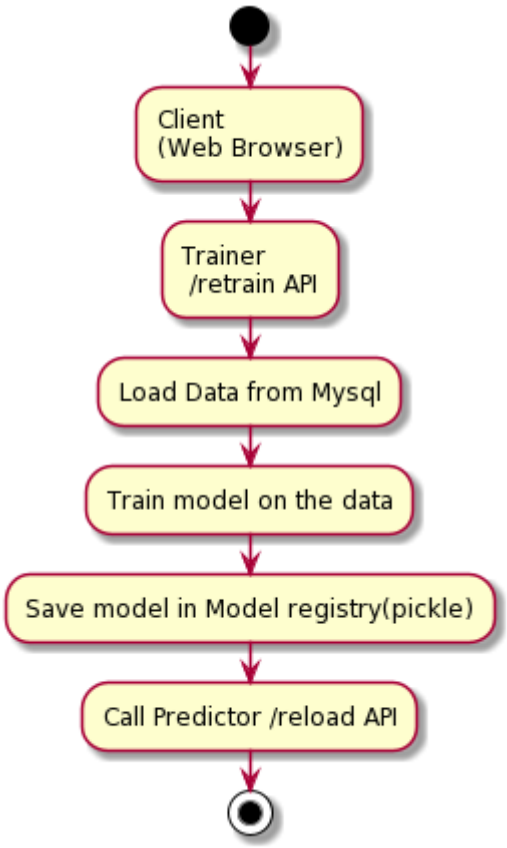

**3. Preprocessing Article**

- It involves text normalization, stop words removal, stemming, lemmatization etc.



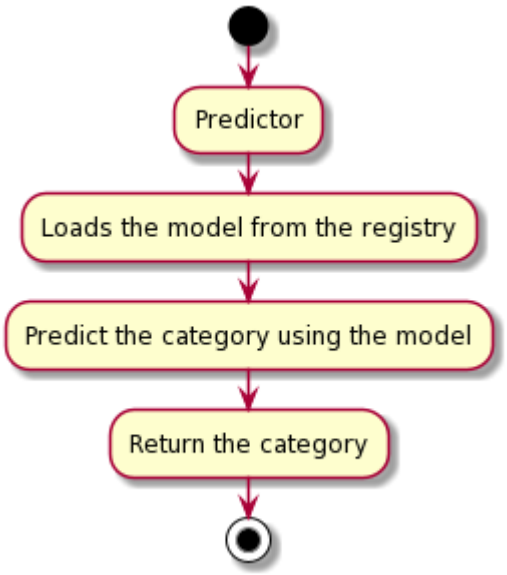

**Trainer**

- Trainer first loads the articles from mysql db.
- Then train the model using MLflow and pyspark.
- MLflow saves the model in the modele-registry as the artifacts.



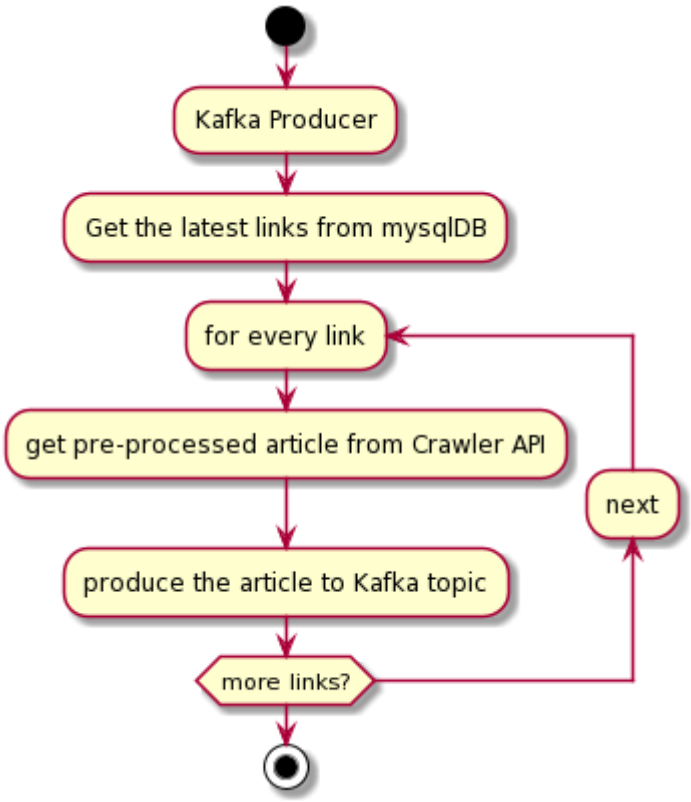

**Predictor**

- Predictor loads the model from the registry.
- Predicts the category, given the article.



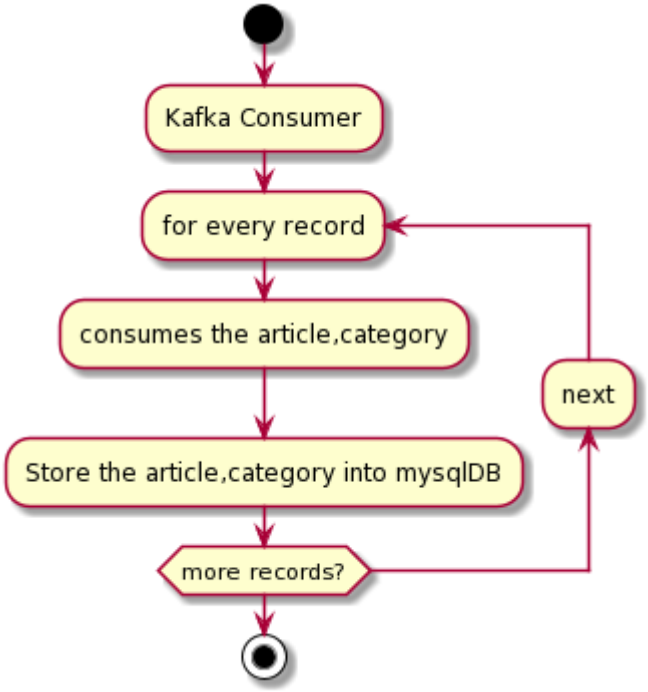
**Kafka Producer**

- Gets the latest links from the mysql db.
- For every link, gets the preprocessed article from the crawler and produce it to Kafka topic.




**Kafka Consumer**

- Kafka Consumer consumes the article and write the article into mysql.




## Group Details

- Group-Name: VK Learners
- - Member1: Vinod Adwani
- - Member2: KSRC Murthy