Name Entity Recognition Project

NLP

GITHUB-LINK

Project Overview

- Objective:
  - Develop an AI system capable of identifying and classifying named entities in a variety of texts.
- Scope:
  - From data collection and preprocessing to model training, evaluation, and deployment on a cloud platform.

Motivation and Objectives

- Motivation:
- Industry Demand: Automated extraction of structured information from unstructured text is critical in sectors like healthcare, finance, and legal for insights and decision-making.
- Technological Advancement: Leveraging advanced NLP techniques to push the boundaries of machine understanding of human language.
- Key Objectives:
- Model Development: Create a state-of-the-art NER model using transfer learning.
- Pipeline Automation: Establish a seamless CI/CD pipeline for rapid iteration and deployment.
- Scalability: Implement the solution on a cloud platform to ensure scalability and accessibility

DATA INGESTION AND PREPROCESSING

- Data Sources:
  - Datasets: Utilized datasets like CoNLL-2003, OntoNotes, and custom datasets for specific domains.
  - Data Collection: Methods for data collection, including scraping and API usage.
- Preprocessing Steps:
  - Tokenization: Breaking down sentences into tokens using libraries like spaCy or NLTK.
  - Normalization: Converting text to lowercase, removing punctuation, and dealing with special characters.
  - Label Encoding: Converting named entities into numerical labels for model training.
  - Handling Imbalance: Techniques like SMOTE or oversampling for class imbalance.
- Tools Used:
  - Pandas & NumPy: For data manipulation and numerical operations.
  - NLTK & spaCy: For natural language processing tasks.

MODEL ARCHITECTURE

- Model Choice:
  - BERT: Pre-trained transformer model known for its bidirectional context understanding.
  - Fine-tuning: Tailoring the pre-trained BERT model for the specific NER task.
- Architecture Details:
  - Input Layer: Tokenized text sequences with corresponding attention masks.
  - Transformer Layers: Multiple layers of self-attention and feed-forward neural networks.
  - Classification Layer: Softmax layer for predicting the entity class for each token.
- Training:
  - Dataset Splitting: Train, validation, and test splits to ensure unbiased evaluation.
  - Hyperparameters: Learning rate, batch size, number of epochs, etc.
  - Optimization: Adam optimizer with a scheduler for learning rate decay.
- Libraries Used:
  - Transformers: For BERT implementation.
  - TensorFlow & PyTorch: For model training and experimentation.

EVALUATION METRICES

- Metrics Used:
- Precision: Ratio of correctly predicted positive observations to the total predicted positives.
- Recall: Ratio of correctly predicted positive observations to all observations in the actual class.
- F1-Score: Harmonic mean of precision and recall, providing a balance between the two.
- Accuracy: Overall correctness of the model.
- Results:
- Performance Visualization: Confusion matrix, precision-recall curves, and F1-score bar charts for different entity types.
- Detailed Analysis: Analysis of misclassified entities and potential reasons.

DEPLOYMENT WORKFLOW FOR THE NAME ENTITY RECOGNITION (NER) PROJECT

- PREPARATION AND SETUP
- CODE FINALIZATION: ENSURE THE FINAL VERSION OF THE CODE, INCLUDING MODEL TRAINING AND INFERENCE SCRIPTS, IS READY.
- ENVIRONMENT SETUP: CONFIGURE THE VIRTUAL ENVIRONMENT AND INSTALL NECESSARY DEPENDENCIES USING REQUIREMENTS.TXT.
- CONTAINERIZATION:
  - WRITE A DOCKERFILE TO DEFINE THE ENVIRONMENT, INCLUDING THE BASE IMAGE, DEPENDENCIES, AND ENTRY POINT.
  - USE .DOCKERIGNORE TO EXCLUDE UNNECESSARY FILES FROM THE DOCKER IMAGE.
- BUILD AND PUSH DOCKER IMAGE
- BUILD DOCKER IMAGE: USE DOCKER TO BUILD THE IMAGE LOCALLY, ENSURING IT INCLUDES ALL NECESSARY COMPONENTS (MODEL, DEPENDENCIES, SCRIPTS).
- TAG AND PUSH: TAG THE DOCKER IMAGE WITH A VERSION NUMBER AND PUSH IT TO GOOGLE ARTIFACT REGISTRY OR ANOTHER CONTAINER REGISTRY.

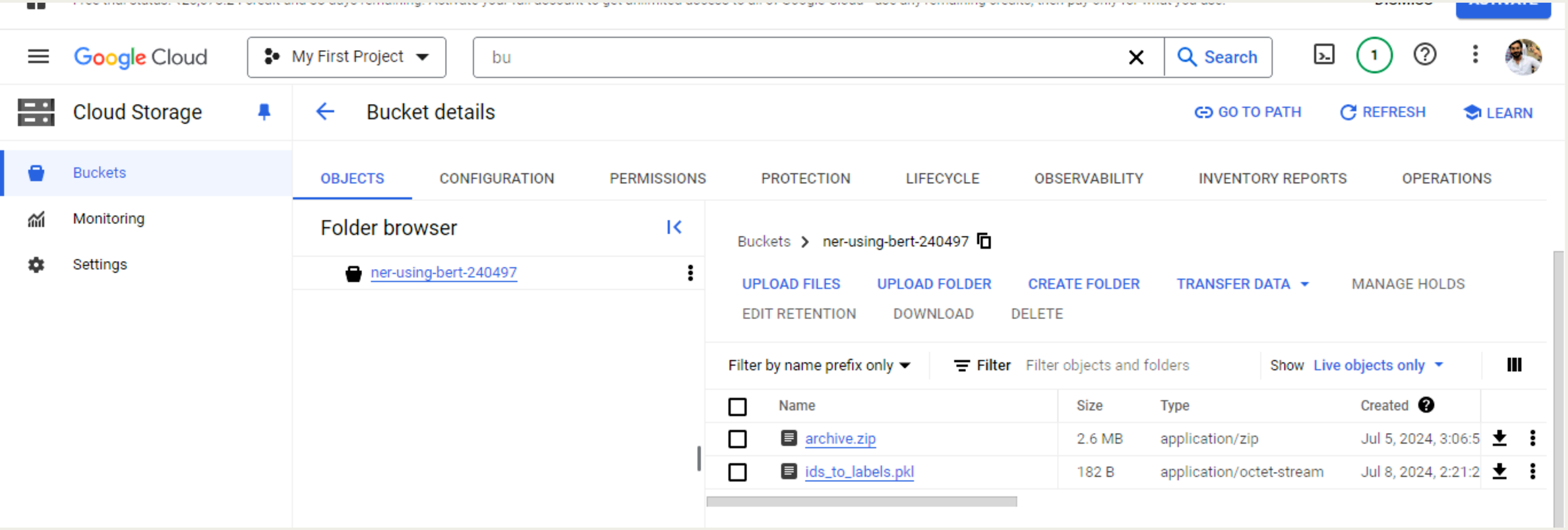
GOOGLE CLOUD PLATFORM (GCP) CONFIGURATION

- Create Artifact Registry: Set up an Artifact Registry in GCP to store Docker images.
- VM Instance Setup:
- Create a VM instance on Google Compute Engine (GCE) with the appropriate machine type and zone (e.g., asia-south1).
- Configure firewall rules and security settings for the VM instance.

DEPLOYMENT ON GCP

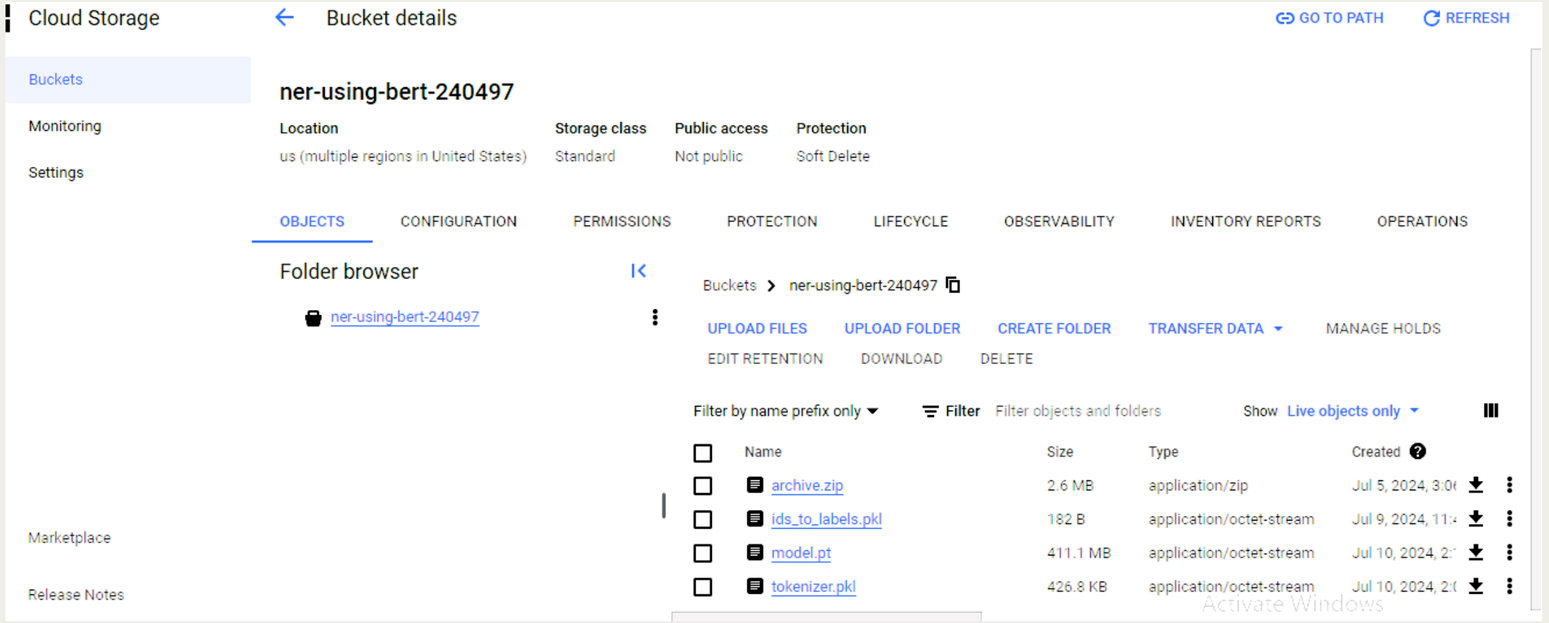
- Pull Docker Image: SSH into the VM instance and pull the Docker image from the Artifact Registry.
- Run Docker Container: Start the container with the necessary environment variables and ports exposed.
- Monitor Logs: Check container logs for successful startup and monitor for any issues.

GCP WORKING IMAGES

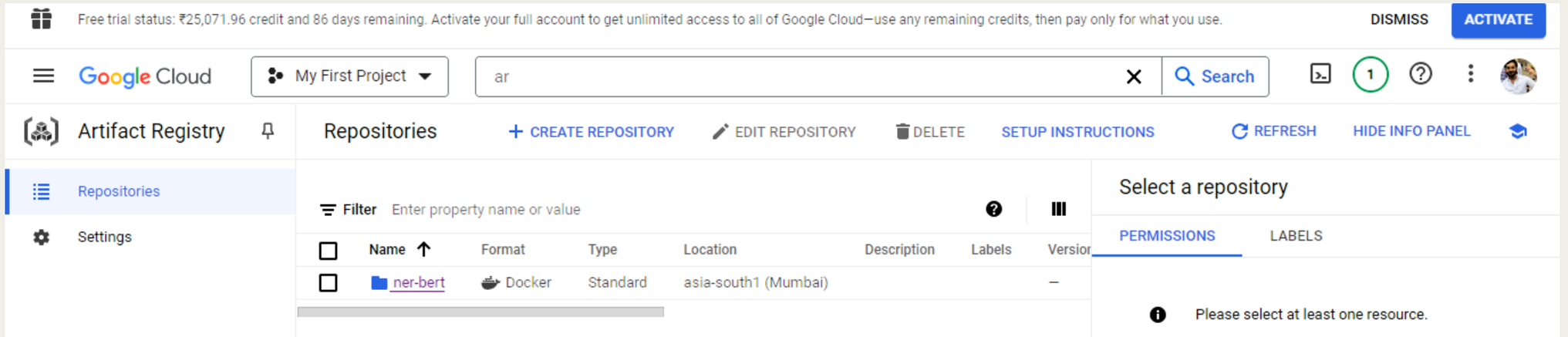


set up of data in cloud storage bucket

GCP WORKING IMAGES



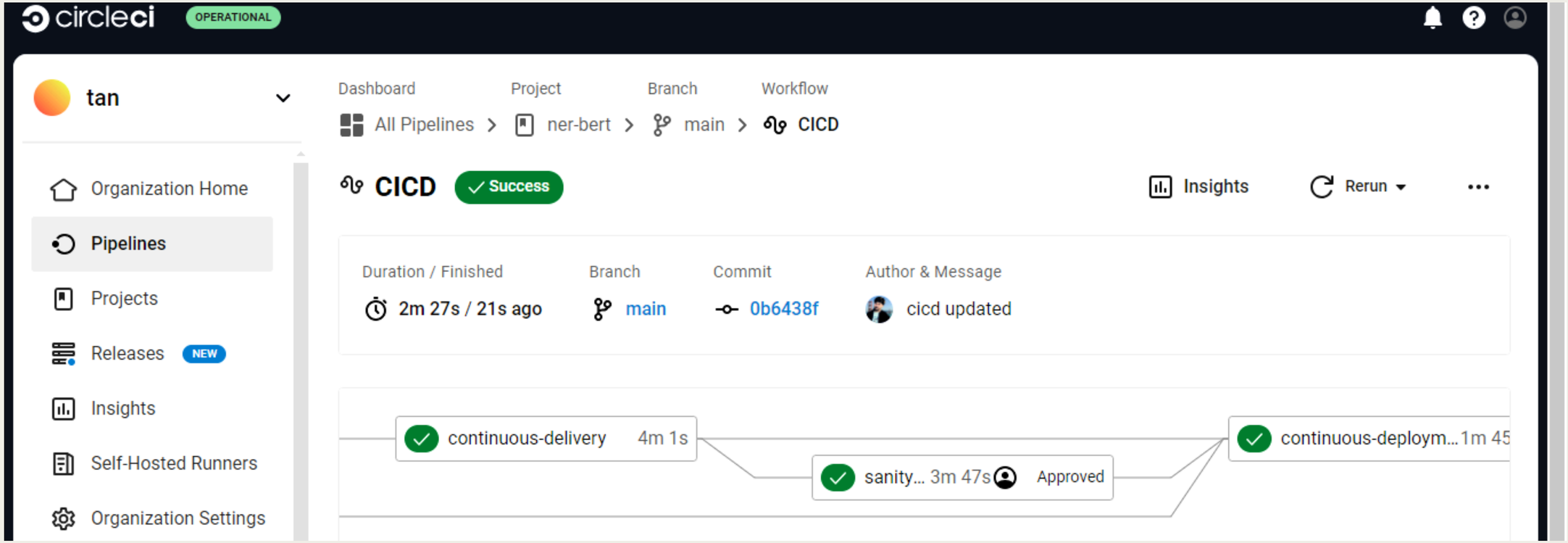
artifact registory

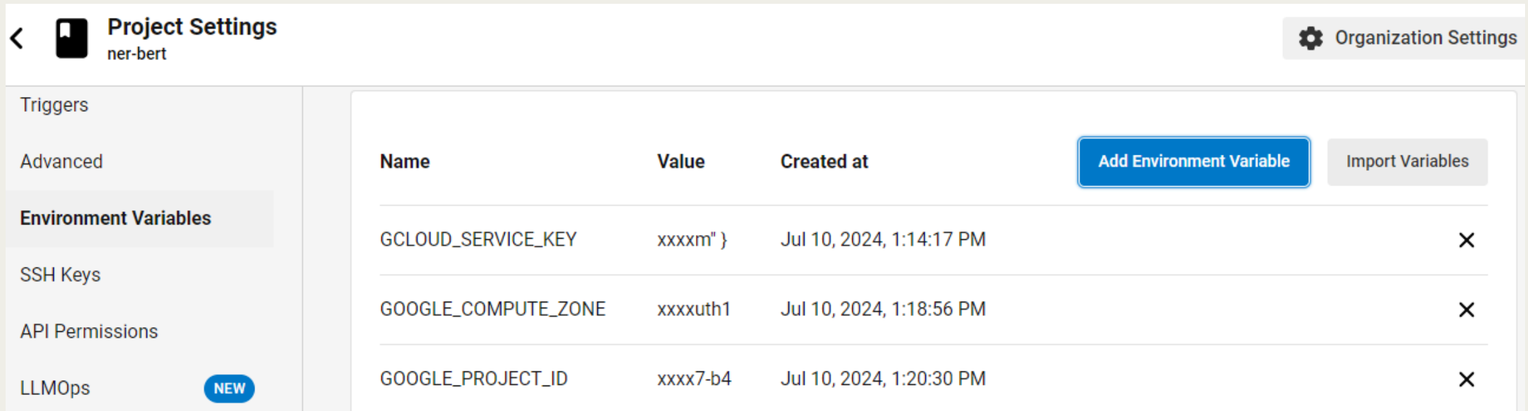


CI/CD INTEGRATION WITH CIRCLECI

- Pipeline Setup: Configure CircleCI with the repository, setting up workflows for build, test, and deployment stages.
- Environment Variables: Set essential environment variables in CircleCI, such as GCLOUD_SERVICE_KEY, GOOGLE_PROJECT_ID, GOOGLE_COMPUTE_ZONE, etc.
- Automated Testing: Implement automated tests to run on every commit to ensure code quality and functionality.
- Deployment Triggers: Set up triggers to automatically deploy to GCP on successful build and test completion.

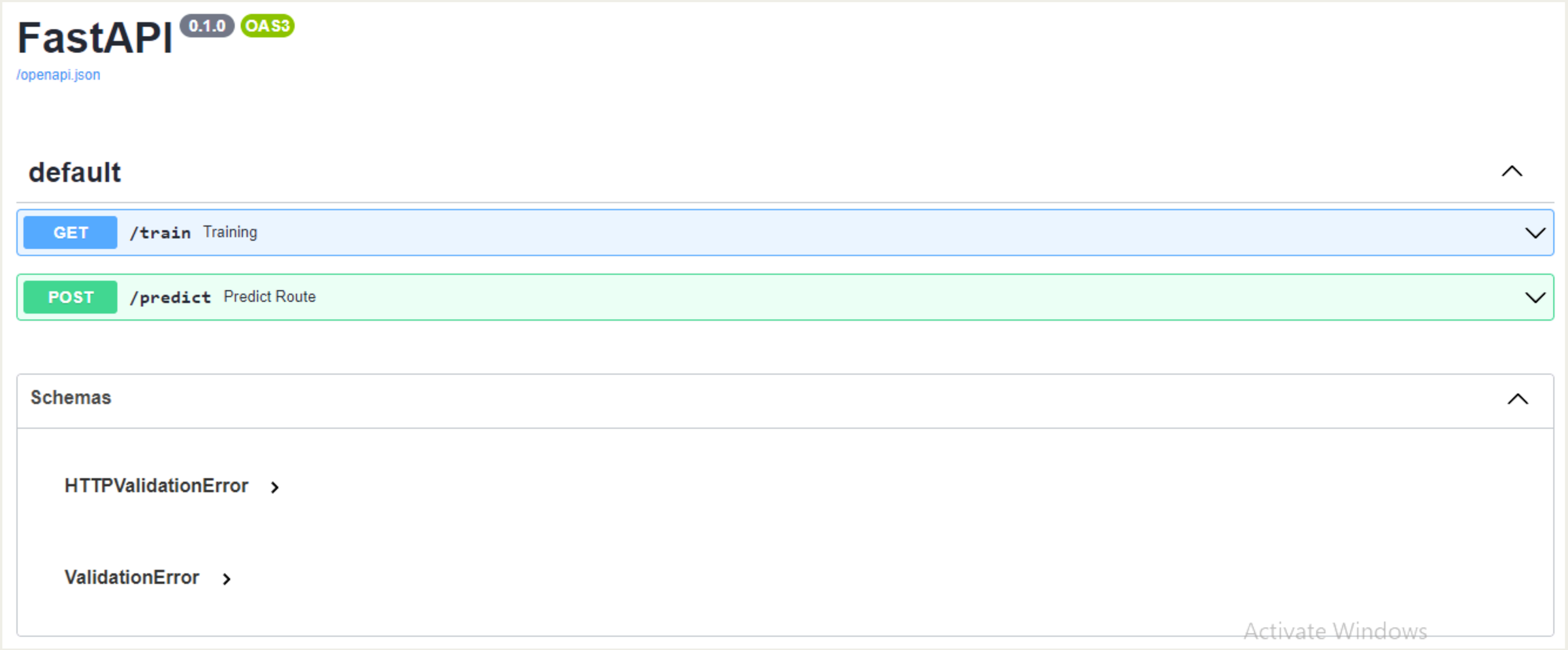
CI/CD INTEGRATION WITH CIRCLECI(IMAGES)



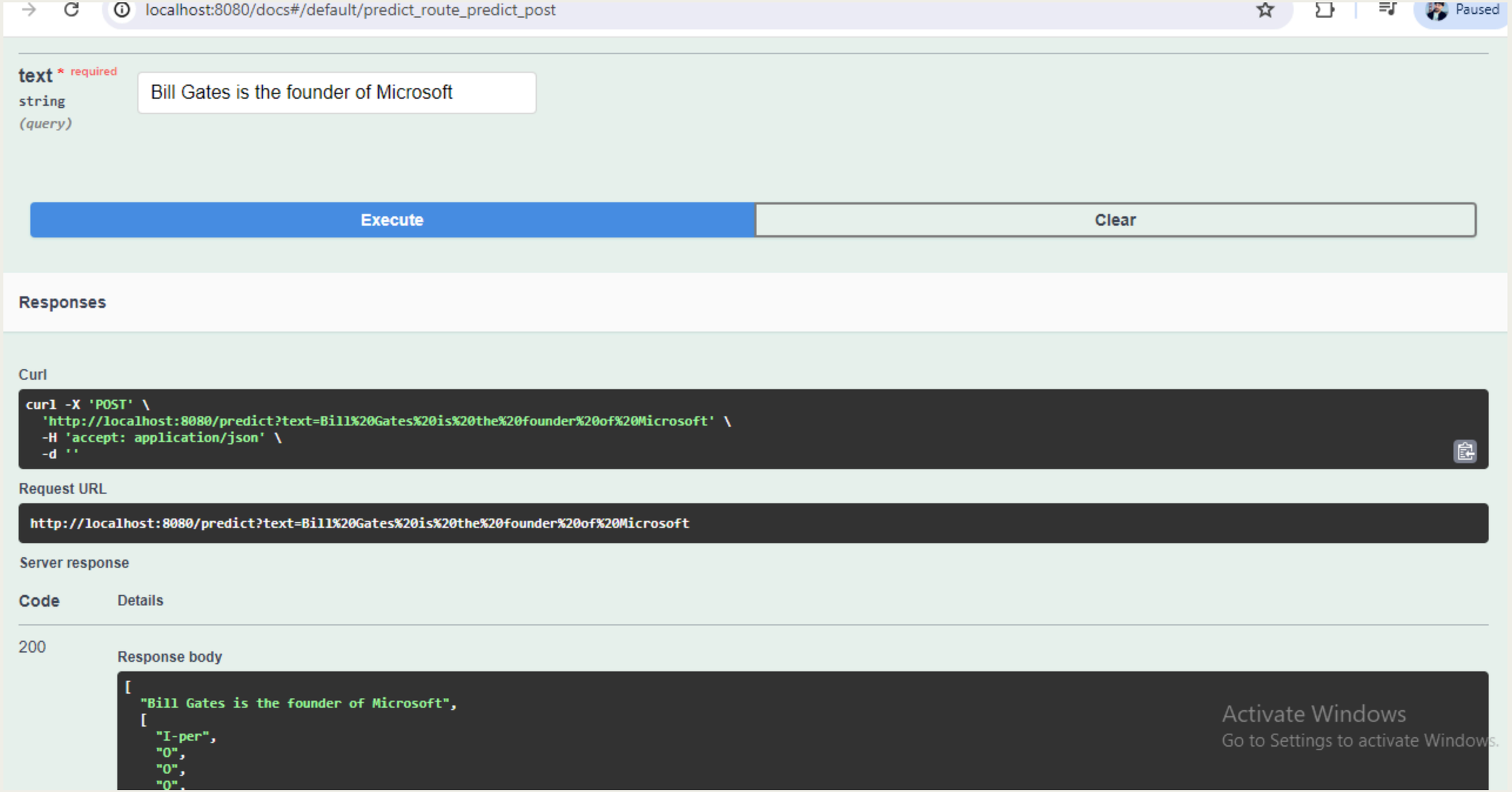


Deployement Successful !!!! performed continous delivery/sanitycheck/continous deployement

PROJECT UI USING FAST API



END RESULTS


localhost:8080/docs#/default/predict_route_predict_post
Paused
text * required
string
(query)
Bill Gates is the founder of Microsoft
Execute
Clear
Responses
Curl
curl -X 'POST' \
'http://localhost:8080/predict?text=Bill%20Gates%20is%20the%20founder%20of%20Microsoft' \
-H 'accept: application/json' \
-d ''
Request URL
http://localhost:8080/predict?text=Bill%20Gates%20is%20the%20founder%20of%20Microsoft
Server response
Code
Details
200
Response body
[
"Bill Gates is the founder of Microsoft",
[
"I-per",
"O",
"O",
"O",
Activate Windows
Go to Settings to activate Windows.

DATASET USED

```
[4]  # Reading csv data

     df = pd.read_csv('/content/drive/MyDrive/data/ner.csv')
     df.tail()
```

	text	labels
47954	Opposition leader Mir Hossein Mousavi has said...	O O O B-per I-per O O O O O O O O O O O O O ...
47955	On Thursday , Iranian state media published a ...	O B-tim O B-gpe O O O O O O O O B-org I-org O ...
47956	Following Iran 's disputed June 12 elections ,...	O B-geo O O B-tim I-tim O O O O O O O O O O ...
47957	Since then , authorities have held public tria...	O O O O O O O O O O O O O O O O O O O O O O
47958	The United Nations is praising the use of mili...	O B-org I-org O O O O O O O O O O O O O B-ti...

```
     # Creating tokenizer intsance

     tokenizer = BertTokenizerFast.from_pretrained('bert-base-cased')
```

Thank you!

Tanujkumar
email:- tanuj.mangalapally@gmail.com