**BAT-KL**

**Experimentation Capabilities – Documentation**

Table of Contents

[CICD – Pipeline Architectural Overview 3](#_Toc92043109)

[Flask Application Models 4](#_Toc92043110)

[User Acceptance Testing Document 4](#_Toc92043111)

[1. Downloading the docker image 4](#_Toc92043112)

[2. Testing the models 5](#_Toc92043113)

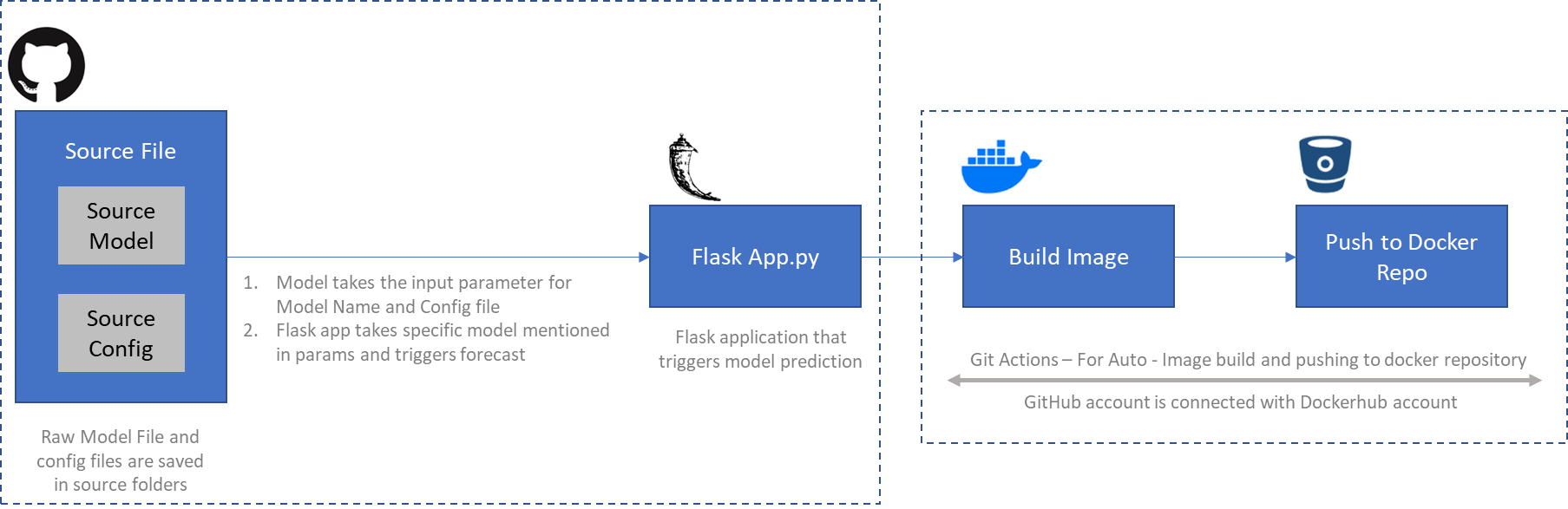
[a. Car Sales price prediction regression 5](#_Toc92043114)

[b. Bank Note Fraud Classification 6](#_Toc92043115)

[c. Text Sentiment Classification 7](#_Toc92043116)

[d. Image Colorizer Model 8](#_Toc92043117)

# CICD – Pipeline Architectural Overview



Components of Pipeline

1. GitHub Repository – Source Folders: These are the Source Model and Source Config folder which contains the model dependencies i.e. model pkl file and model configurations. Whenever user wants to push new model – they can push pkl file and config file in the Source Model and Source Config folders respectively.
2. Flask Application – Flask application that consumes the pkl file from Source Model and configurations from Source Config folder and triggers the predict function.
3. Build Image - Git Action: This is the auto build docker imager trigger for Git code. The Git Action functionality is present in the github/workflows. This workflow detects the change in the branch, and whenever any change is detected in the folder the docker image is built with a new tag.

*Detailed document on Git Workflow* - <https://docs.github.com/en/actions/managing-workflow-runs/manually-running-a-workflow>

1. Dockerhub Repository – The Dockerhub and github accounts are connected through Github secrets and Dockerhub token. Post creation of docker image it is pushed to Dockerhub account.

*Detailed document on Github and Dockerhub connection* - <https://davelms.medium.com/build-your-docker-images-automatically-when-pushing-new-code-to-github-394f4c1679cc>

\*Whenever the models are added/updated in source folders there are two docker images that are pushed in Dockerhub repository – with a ‘run number’ tag and with a ‘latest’ tag

# Flask Application Models

|  |  |  |  |
| --- | --- | --- | --- |
| Input Format | Model Name | Config File | Vars / Input |
| value | model\_car\_sale\_price\_prediction\_regression.pkl | config\_car\_sale\_price\_prediction\_regression.csv | *Vars as mentioned in config file* |
| model\_iris\_classification.pkl | config\_iris\_classification.csv | *Vars as mentioned in config file* |
| file | model\_banknote\_auth.pkl | NA | *file : csv file input* |
| text | model\_sentiment\_nlp\_model.pkl | NA | *text : string input* |
| image | colorization\_release\_v2.caffemodel  colorization\_deploy\_v2.prototxt  pts\_in\_hull.npy | NA | *image : jpg file input* |

# User Acceptance Testing Document

## Downloading the docker image

Download the docker image of the required version from Dockerhub website. The docker images can be found in ***seah101/model\_v1*** (private Repository) <https://hub.docker.com/repository/docker/seah101/model_v1/tags?page=1&ordering=last_updated>

Docker image download command:

docker pull seah101/model\_v1:19

Docker image run command:

docker run -it -p 5000:5000 seah101/model\_v1:19

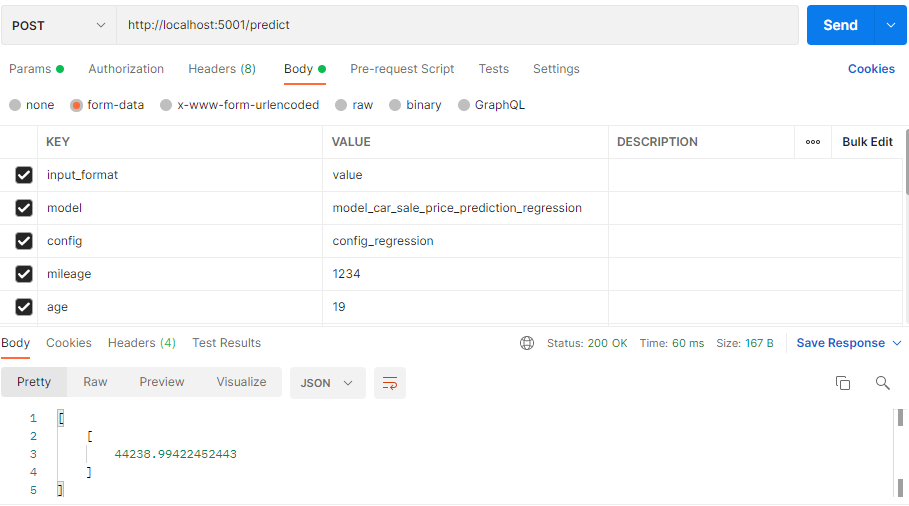
## Testing the models

### Car Sales price prediction regression

**Testing through postman:**

Input through form-data:

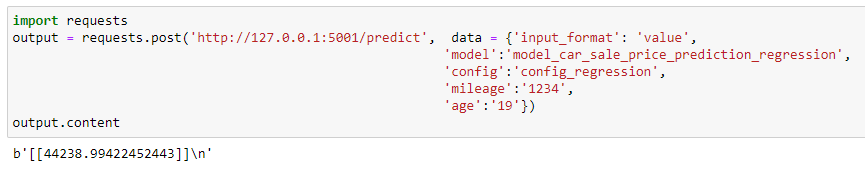
|  |  |
| --- | --- |
| **Key** | **Value** |
| input\_format | value |
| model | model\_car\_sale\_price\_prediction\_regression |
| config | config\_regression |
| mileage | 1234 |
| age | 19 |

Output:

**Testing through curl command:**

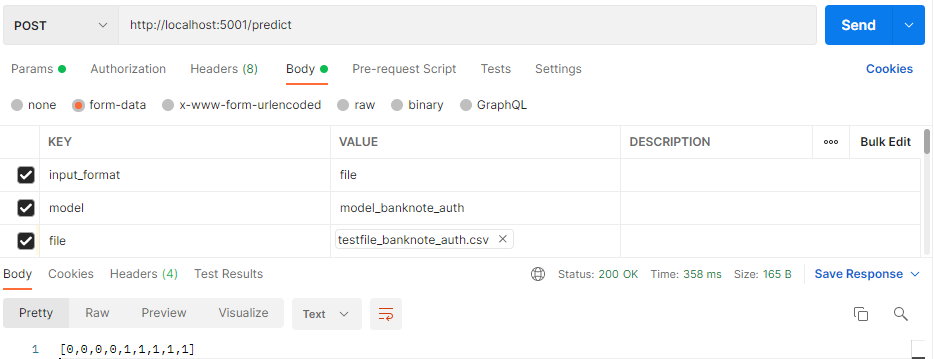
curl <http://127.0.0.1:5001/predict> -d "model=model\_car\_sale\_price\_prediction\_regression&config=config\_regression&input\_format=value&mileage=1234&age=11"



**Testing through python:**

### Bank Note Fraud Classification

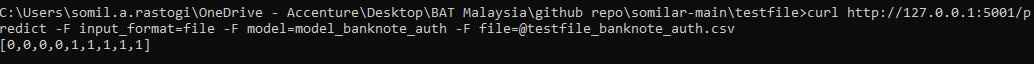
**Testing through postman:**

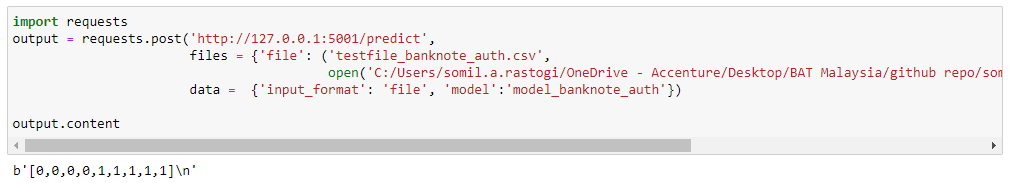
Input through form-data:

|  |  |
| --- | --- |
| **Key** | **Value** |
| input\_format | file |
| model | model\_banknote\_auth |
| file | <*testfile\_banknote\_auth*> |

**Testing through curl command:**

curl <http://127.0.0.1:5001/predict> -F input\_format=file -F model=model\_banknote\_auth -F file=@testfile\_banknote\_auth.csv



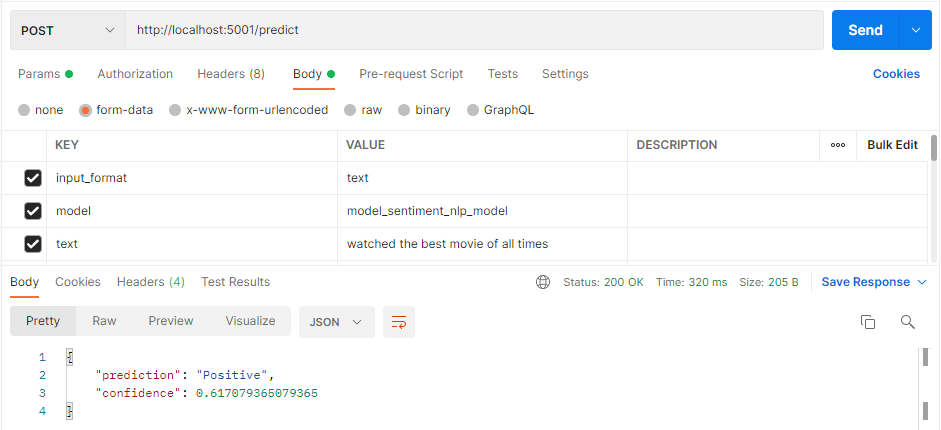
**Testing through python:**

### Text Sentiment Classification

**Testing through postman:**

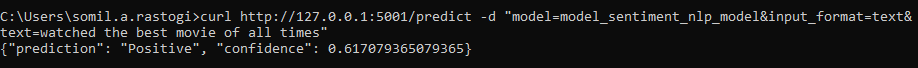
Input through form-data:

|  |  |
| --- | --- |
| **Key** | **Value** |
| input\_format | text |
| model | model\_sentiment\_nlp\_model |
| text | <*text*> |

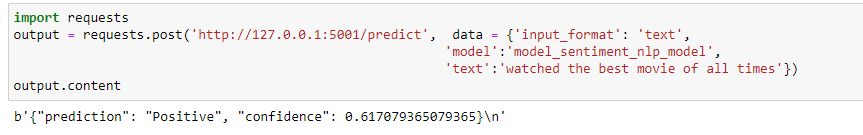


**Testing through curl command:**

curl <http://127.0.0.1:5001/predict> -d "model=model\_sentiment\_nlp\_model&input\_format=text&text=watched the best movie of all times"



**Testing through python:**



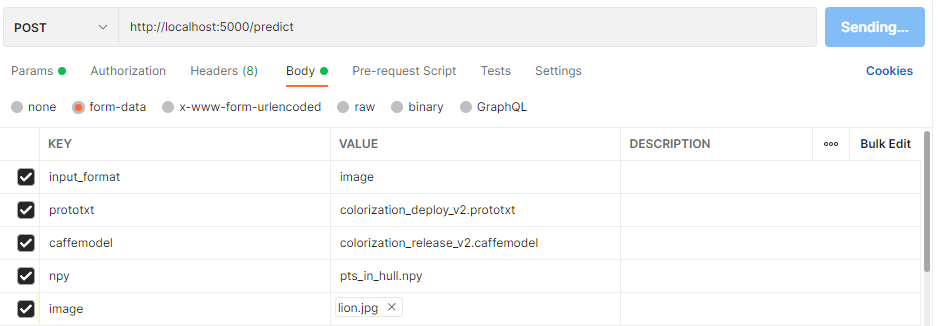
### Image Colorizer Model

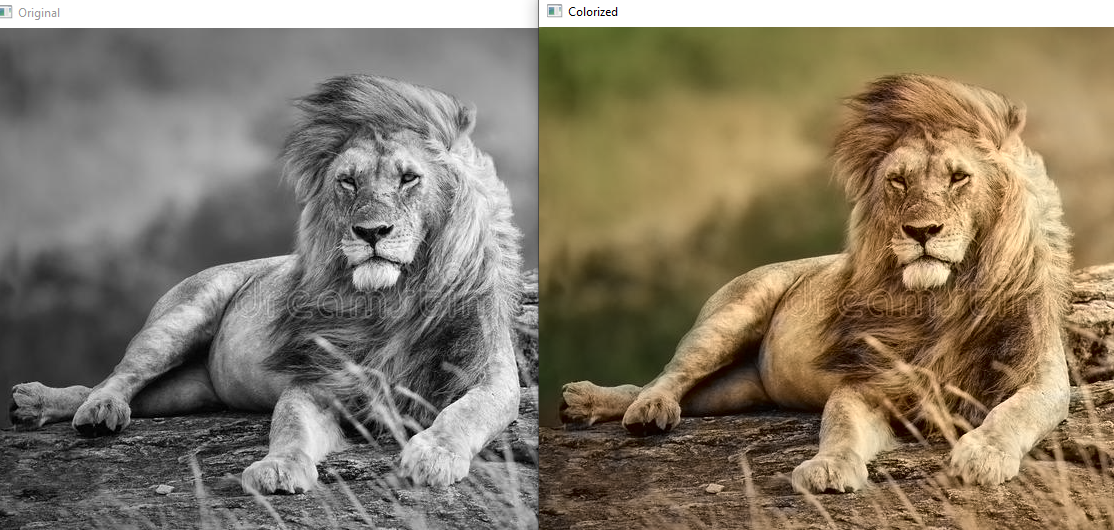
**Testing through postman:**

Input through form-data:

|  |  |
| --- | --- |
| **Key** | **Value** |
| input\_format | image |
| prototxt | colorization\_deploy\_v2.prototxt |
| caffemodel | colorization\_release\_v2.caffemodel |
| npy | pts\_in\_hull.npy |
| Image | <*image*> |

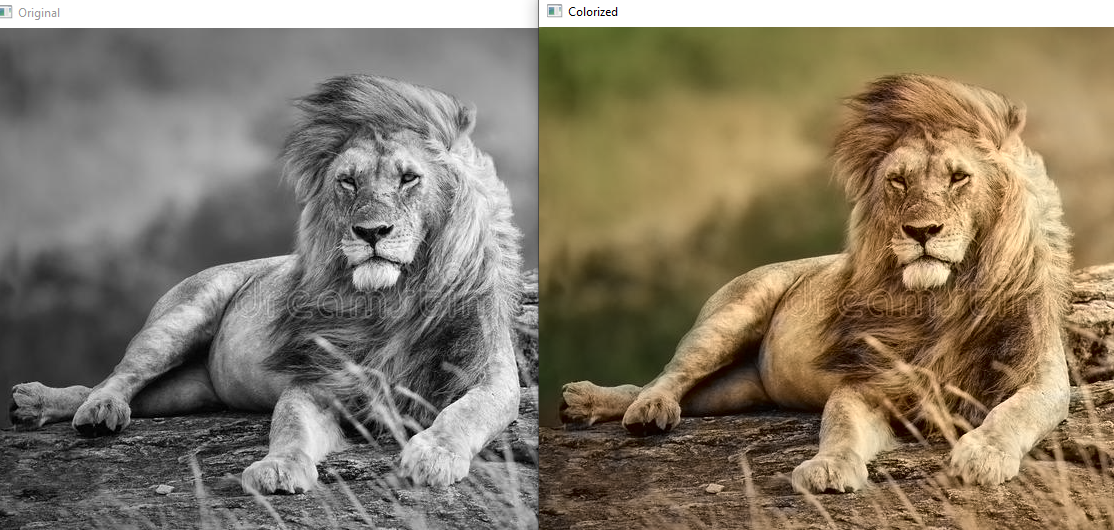
\*The model files are uploaded in shared folder (due to file size limitation in github) - https://bat.sharepoint.com/:f:/r/sites/DatascienceEngineering/Shared%20Documents/General/Pre\_Trained\_Models/Docker%20Experiment/source\_model?csf=1&web=1&e=ZPHRbO





**Testing through curl command:**

curl <http://127.0.0.1:5000/predict> -F input\_format=image -F prototxt=colorization\_deploy\_v2.prototxt -F caffemodel=colorization\_release\_v2.caffemodel -F npy=pts\_in\_hull.npy -F [image=@lion.jpg](mailto:image=@lion.jpg)



**Testing through python:**

