

# **COVID -19 Predictor using End-to-End AIOps**

Final Report Submitted  
in Fulfillment of the Requirements  
for the Course of  
**Major Project - II**  
In  
Fourth Year – VIII Semester of  
**Bachelor of Technology**  
specialization  
In  
**DevOps, CCVT and GG**

Under

**Dr. Hitesh Kumar Sharma**

By

<b>500060720</b>	<b>R171217041</b>	<b>Nishkarsh Raj</b>
<b>500060911</b>	<b>R110217071</b>	<b>Karan Nautiyal</b>
<b>500059999</b>	<b>R110217083</b>	<b>Megha Rawat</b>
<b>500060227</b>	<b>R142217022</b>	<b>Rishabh Negi</b>



**UNIVERSITY WITH A PURPOSE**

DEPARTMENT OF CYBERNETICS AND VIRTUALIZATION  
SCHOOL OF COMPUTER SCIENCE  
UNIVERSITY OF PETROLEUM AND ENERGY STUDIES,  
BIDHOLI, DEHRADUN, UTTARAKHAND, INDIA  
**May 2021**

## **Abstract**

COVID-19 situation was declared as a pandemic by WHO as it was spreaded across the world. To understand the impact of COVID-19 it is imperative we stay up to date with the numbers of this pandemic. With India on the second rise of covid cases the burden on frontline workers has been greater than ever.

This project intends to offload the testing load from the medical workers in big cities where around 44,000 Covid tests are being conducted daily.

This project uses a model trained using the database made public by the Israeli Ministry of Health containing data of all individuals who were tested for SARS-CoV-2 via RT-PCR assay of a nasopharyngeal swab.

**Keywords:** *covid-19, RT-PCR, Covid Testing*

## **Introduction**

The Coronavirus flare-up became exposed on 31 December 2019, when China reported the World Health Organization regarding numerous cases of pneumonia in Wuhan City. The illness spread to more places in China, and the world. The World Health Organization has now proclaimed it a pandemic. The infection has been named SARS-CoV-2 and the sickness is currently called COVID-19<sup>[1]</sup>.

COVID-19 is a sickness brought about by a newfound CoronaVirus. The majority of people infected with the COVID-19 infection will encounter respiratory disease and recover without requiring unique treatment. This disease mostly affects senior citizens or the person with pre-existing medical conditions like cancer, diabetes, cardiovascular problem, etc.

There have been millions of Coronavirus cases to date and more than a million deaths across the world. The pandemic is in its midst and medical professionals across the globe are working on creating a coronavirus vaccine, there are hundreds of teams working to find the covid vaccine as the vaccine could save millions of lives globally. However, the distribution of the vaccine is still a very big hurdle and might take a lot of time<sup>[2]</sup>. The vaccine is only a way to control the spread and would not be able to eradicate it as it is a form of seasonal flu that can easily mutate. Covid-19 infection would last for the coming years as informed by WHO.

The project aim is to create an AIOPs pipeline for deploying a containerized application that helps detect covid from medical scans of an individual, since the virus is constantly mutating we might need to constantly modify and update the model and deploy the new model this agile model will help us achieve this with minimum time delays. This will ensure the project can be utilized to the maximum capacity.

## **Motivation**

With India reporting around 3 lakh cases ( as of May, 2021) there is immense burden on the medical system of the nation. The time spent testing can be better utilized for attending to the patients. Testing also puts individuals who suspect they might have covid but are healthy. Therefore, it is better to have an automated screening process before the testing to ensure only highly susceptible individuals arrive at physical testing locations.

## **Problem Statement**

As COVID-19 pandemic is at a dangerous peak we need to ensure we can test and quarantine as efficiently and safely as possible. Crowding and test centers may create a risk in individuals that suspect they are infected but maybe healthy. Can we have a method to pre-screen individuals so only individuals who are highly likely to be infected have to be physically present at the test centre and healthy individuals can avoid exposure risks?

## **Objective**

The aim is to engineer an AIOPs pipeline to deploy COVID-19 predictor application that can work as a pre-screener before needing to visit the physical location for the RT-PCR test. The intent is to get a more accurate estimate of people who might be suffering from the disease to get a better understanding of the extent of the disease and serve as a guideline for precautionary measures that should be taken to reduce further transmission of the virus, so that we can be sure when a strict lockdown might be necessary.

## **Literature Review**

This project uses the database made public by the Israeli Ministry of Health containing data of all individuals who were tested for SARS-CoV-2 via RT-PCR assay of a nasopharyngeal swab

Medical Explanations of coronavirus and facts are validated from THE NEW ENGLAND JOURNAL OF MEDICINE which states the strategies and evidences as Coronaviruses are RNA viruses that are divided into four general; alpha coronaviruses and beta coronaviruses are known to infect humans.

SARS-CoV-2 is related to bat coronaviruses and to SARS-CoV, the virus that causes SARS. Similar to SARS-CoV, SARS-CoV-2 enters human cells through the angiotensin-converting-enzyme 2 (ACE2) receptor. SARS-CoV-2 has

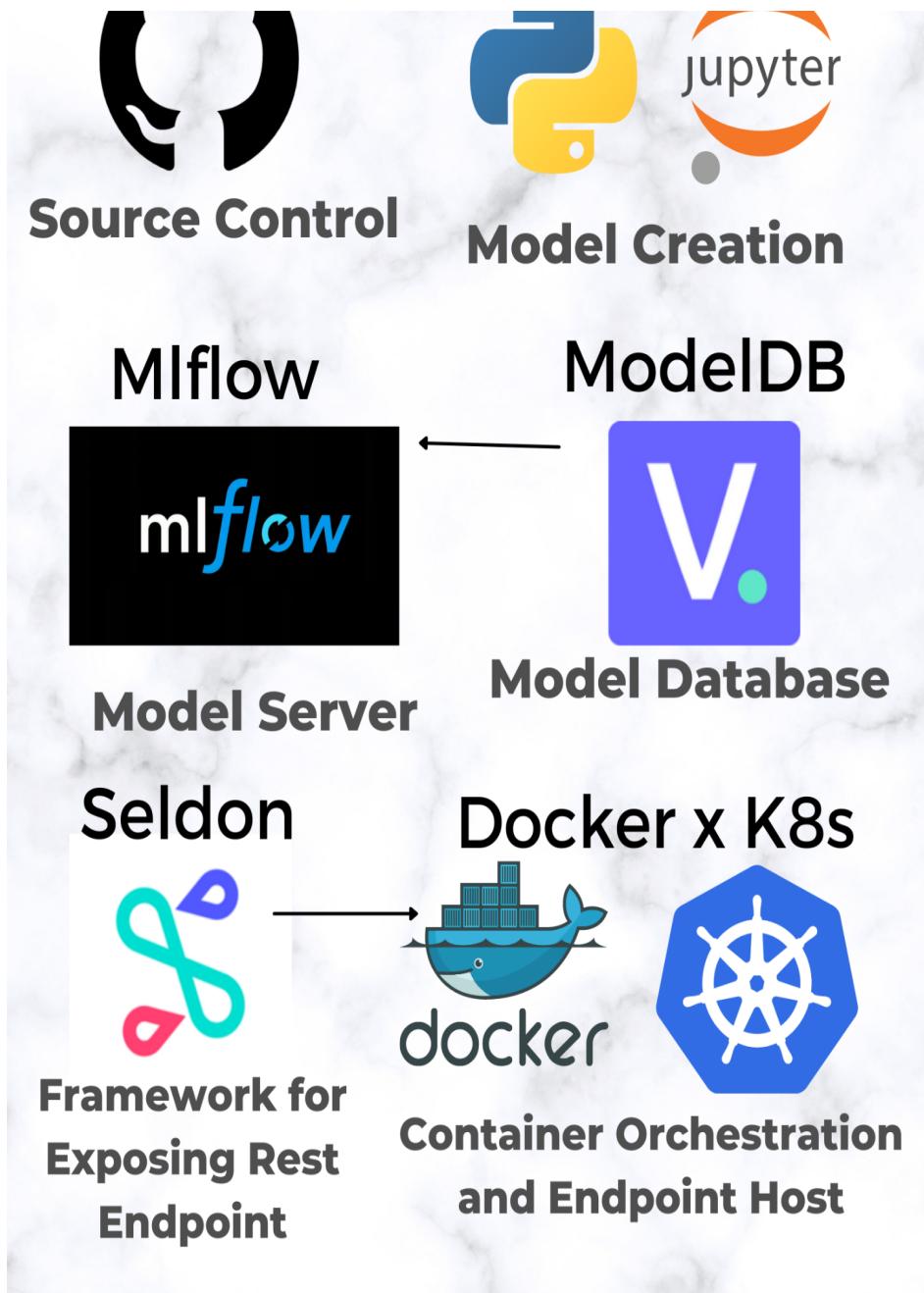
RNA-dependent RNA polymerase and proteases, which are targets of drugs under investigation.<sup>[3]</sup>

The model takes into consideration how the virus differently affects males and females.<sup>[17]</sup>

## **Proposed Method**

1. This project uses public data set populated by Israel's health ministry for training
2. A pytorch based machine learning pipeline is created to build a two-classification model (Covid and Non-Covid)
3. The machine learning model is then serialized with cloud pickle.
4. A predictor function is wrapped around the model to take images of CT scan as an input and return which class the image belongs to.
5. For hosting the model's predictor function publicly, we expose REST and gRPC endpoints by migrating the predictor function to the Seldon framework.
6. Containerized application wrapper is created with Docker and hosted publicly on managed Kubernetes.
7. Model storage is handled with an instance of the Verta AI's ModelDB platform.
8. For logging and registration of the model for endpoints in gRPC, Mlflow functionality support is added.

## Methodology



## **System Requirements:**

### **1. Software:**

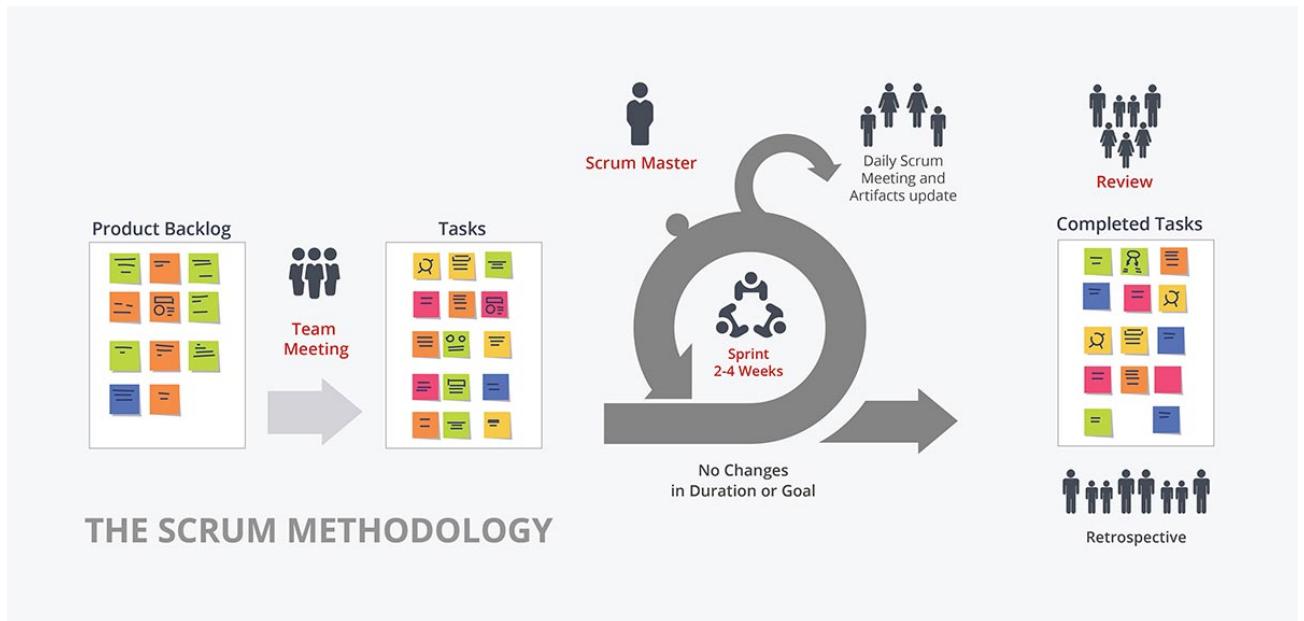
- Python 3.8.5 or above
- Jupyter Notebook
- VSCode/ IntelliJ/ Eclipse IDE
- Seldon, Mlflow and ModelDB dependencies of Python
- Docker Engine
- Azure CLI
- Git Bash

### **2. Hardware:**

- 4 GB RAM or above
- Intel Pentium 4 processor or later that's SSE2 capable

## SDLC Model: SCRUM Model

The entire project is divided into individual modules, based on their functionalities. Which will be developed in isolation by the entire team, based on the priority of implementation. Using the Version Control tool Git.

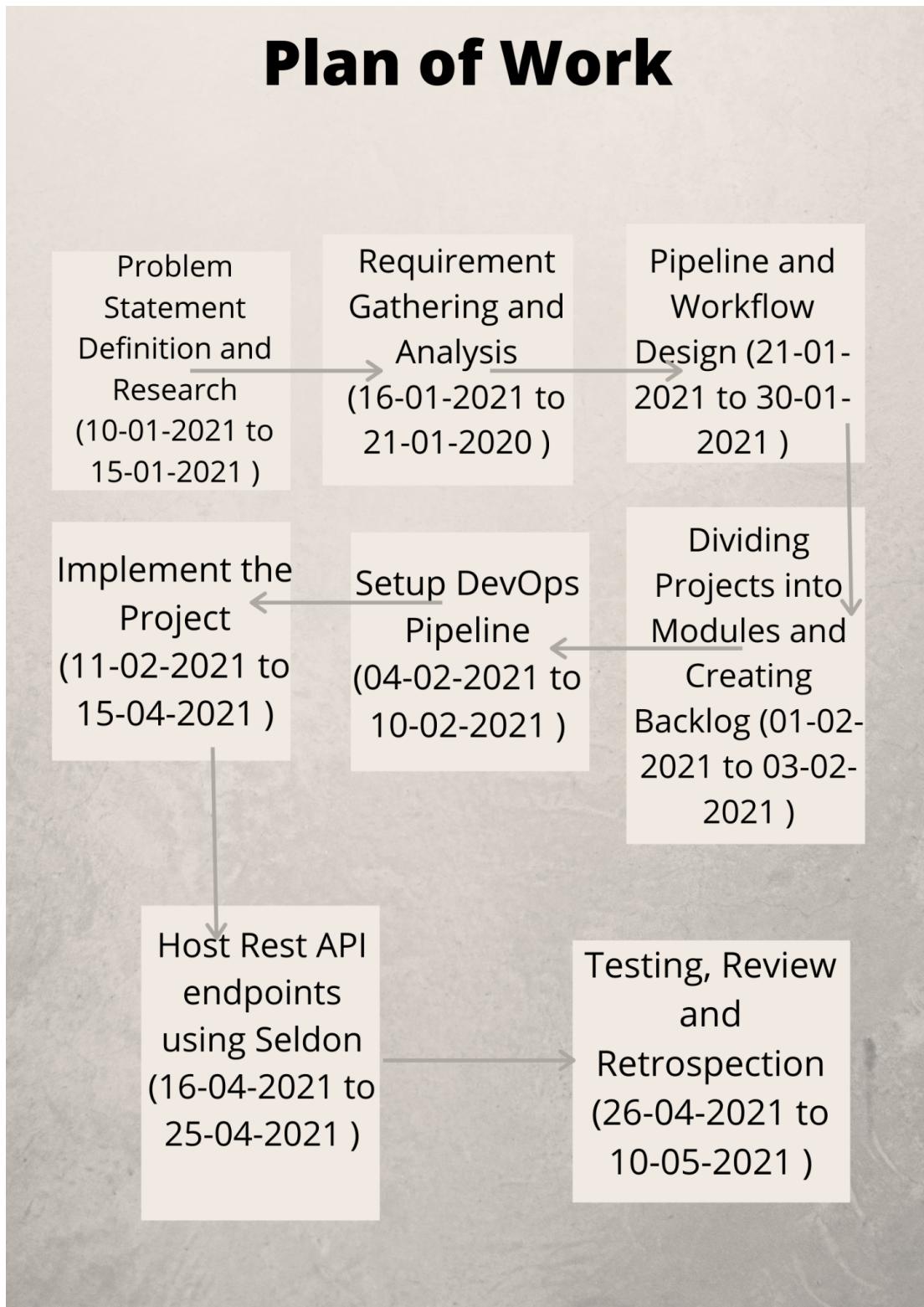


Scrum Model

## Efficiency Parameters

- Sensitivity:** The test should be sensitive enough to not miss many positive cases as that defeats the purpose.
- Specificity:** The model requires moderate specificity as someone with a false positive result can always go to a physical screening location.
- Frequency:** The frequency of updates should be as much as possible to ensure the model is relevant.

## Plan of Work



## **Project progress**

We are using the SCRUM model so we have divided the project into different modules. All these modules are listed in the product log

### **Product Log :**

This is the most crucial document. It lists all the requirements of the project and can be considered as a to-do list. All these modules provide some functionality to the business process. Our product log is as follows

1. Research about a reliable and up-to-date data source for Covid-19
2. Create a GitHub repository to collaborate in a distributed environment
3. Create a .gitignore file to ignore python executables and dependencies
4. Create a .dockerignore file to optimize the production level Docker build
5. Create a pipeline to generate a machine learning model using pytorch which is serialized with pickle.
6. Create a metric evaluation code to find accuracy, specificity and Area under ROC of the model.
7. Create a predictor function that takes URL/local image as input and generates prediction of Covid19
8. Migrate the predictor function into Seldon framework
9. Create an optimized docker image of the Seldon workflow of the model
10. Expose REST Endpoints of the model with Docker and Kubernetes
11. Host a Model storage server with Verta AI's ModelDB
12. Log and Register Models in MLflow
13. Serve gRPC endpoints in MLflow

## Increment of Sprint I (Sprint Review)

1. Host the project on GitHub -

<https://github.com/stormsinbrewing/Covid-19-Predictor>

The screenshot shows the GitHub repository page for 'Covid-19-Predictor'. The repository is private and has 22 commits. The main branch is 'main'. The repository description is: 'A Covid-19 Predictor using PyTorch library that intakes a CT Scan image of chest of a person and detects if they have COVID or not.' It includes links to 'Readme', 'MIT License', and 'Create a new release'. The code section shows a list of files and their last commit times.

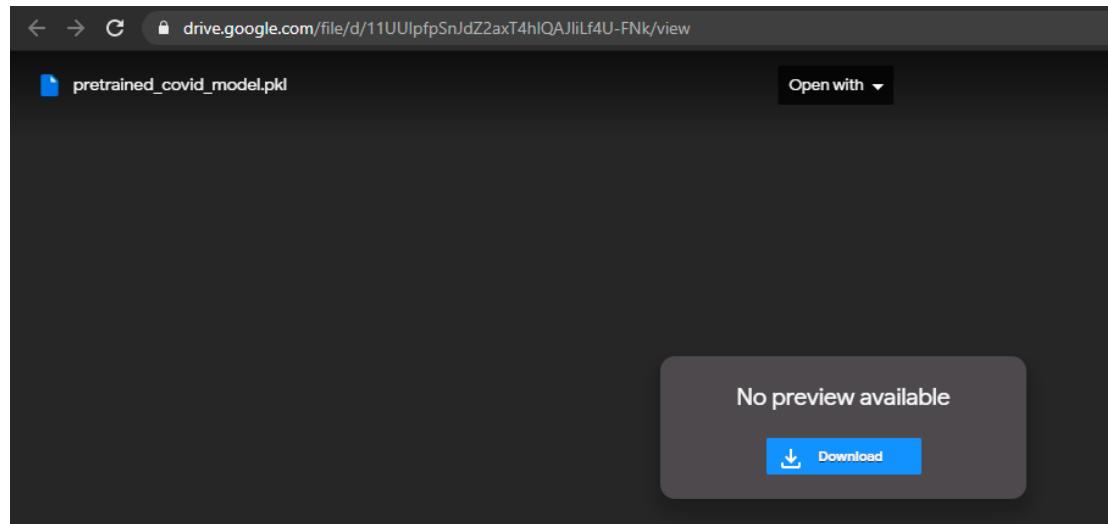
File	Description	Last Commit
1_ModelCreator	Model Creator   Model Metrics   Predictor   Readme	6 days ago
2_ModelMetrics	Update Readme.md	6 days ago
3_Predictor	Model Creator   Model Metrics   Predictor   Readme	6 days ago
4_Seldon	Seldon Image x DockerHub x Kubernetes	6 days ago
5_AIOps	AIOps Pipeline   V1 launch	5 days ago
docs	Model Creator   Model Metrics   Predictor   Readme	6 days ago
img	AIOps Pipeline   V1 launch	5 days ago
.gitignore	Model Attributes and Metrics Implementation	6 days ago
LICENSE	MIT Licensing for open source	6 days ago

2. Create Pipeline in Jupyter Notebook to create a machine learning model with Pytorch library and pickle serialization

The screenshot shows a Jupyter Notebook interface. The left sidebar displays the file structure of the 'COVID-19-PREDICTOR' repository, including files like 'Notebook.ipynb', 'Metric.py', and 'Predict.py'. The main notebook cell contains Python code for a COVID-19 chest CT classifier using PyTorch. The code imports various libraries such as torch, torchvision, and sklearn, and performs tasks like data loading, model training, and evaluation.

```
import torch
import torch.nn as nn
from torch.utils.data import Dataset, DataLoader
from torchvision import transforms as transforms
from skimage.util import montage
import os
import cv2 as cv2
import random
import matplotlib.pyplot as plt
import torch.optim as optim
from PIL import Image
from sklearn.metrics import classification_report, roc_auc_score, roc_curve, confusion_matrix
from torch.utils.tensorboard import SummaryWriter
import glob
import shutil
import numpy as np
from torchvision.models import vgg19_bn
import numpy as np
import seaborn as sns
```

Since the model size is large and takes a lot of time to build, we store it safely on Google Drive for backup and sharing.



### 3. Download the dataset from <https://github.com/UCSD-AI4H/COVID-CT>

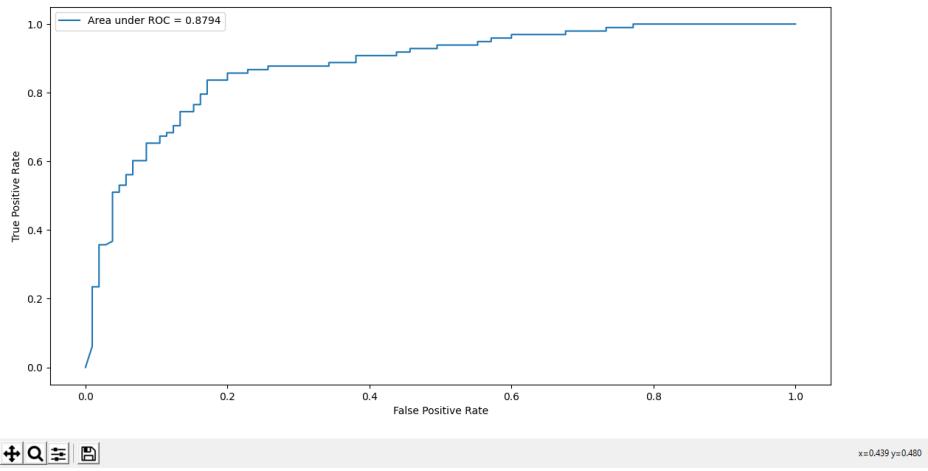
A screenshot of a GitHub repository page. The URL is github.com/UCSD-AI4H/COVID-CT. The page header includes a search bar, navigation links for Pull requests, Issues, Marketplace, and Explore, and user statistics (Watch 45, Star 844, Fork 361). The main content area shows the 'Code' tab selected. It displays the 'master' branch with 1 commit and 0 tags. The commit details show a merge pull request from fanweixiao/master. The commit was made by ikoo on Jan 26, 2021, with 210 commits. Below the commit list are file changes: Data-split (Update testCT\_COVID.txt), Images-processed (Delete .DS\_Store), baseline methods (Update README.md), COVID-CT-MetaInfo.xlsx (Add files via upload), NonCOVID-CT-MetaInfo.csv (csv format), and README.md (Update README.md). On the right side, there is an 'About' section with a description: 'COVID-CT-Dataset: A CT Scan Dataset about COVID-19', and a list of tags: computer-vision, deep-learning, dataset, ct, computed-tomography, and covid-19. There is also a 'Readme' link and a 'Releases' section indicating 'No releases published'.

4. Write a model evaluation function in python to derive accuracy, specificity and AREA under ROC

The screenshot shows a Jupyter Notebook interface with three tabs at the top: 'Notebook.ipynb' (selected), 'Metric.py', and 'Predict.py'. Below the tabs, the current file is '2\_ModelMetrics > Metric.py > ...'. The notebook contains Python code for importing libraries (torch, torch.nn, Dataset, DataLoader, transforms, skimage.util, os) and handling warnings. It then prints 'Test Performance' followed by a table of metrics: Accuracy (0.828), Sensitivity (0.837), Specificity (0.819), and Area Under ROC (0.879). The code cell is preceded by a multi-line comment explaining how to revert changes if the source code is modified.

```
## Import Libraries
import torch
import torch.nn as nn
from torch.utils.data import Dataset, DataLoader
from torchvision import transforms as transforms
from skimage.util import montage
import os

# you can retrieve the original source code by accessing the object's source attribute tool to revert the changes.
warnings.warn(msg, SourceChangeWarning)
C:\Python37\lib\site-packages\torch\serialization.py:656: SourceChangeWarning:
d. you can retrieve the original source code by accessing the object's source attribute tool to revert the changes.
warnings.warn(msg, SourceChangeWarning)
C:\Python37\lib\site-packages\torch\serialization.py:656: SourceChangeWarning:
s changed. you can retrieve the original source code by accessing the object's source attribute tool to revert the changes.
warnings.warn(msg, SourceChangeWarning)
C:\Python37\lib\site-packages\torch\serialization.py:656: SourceChangeWarning:
ou can retrieve the original source code by accessing the object's source attribute tool to revert the changes.
warnings.warn(msg, SourceChangeWarning)
C:\Python37\lib\site-packages\torch\serialization.py:656: SourceChangeWarning:
you can retrieve the original source code by accessing the object's source attribute tool to revert the changes.
warnings.warn(msg, SourceChangeWarning)
----- Test Performance -----
Accuracy      0.828
Sensitivity   0.837
Specificity   0.819
Area Under ROC 0.879
-----
```



5. Write a Predictor function that takes an image of CT scan of the chest and predicts whether you are Covid positive or not based on the model created.

The screenshot shows a Jupyter Notebook interface with the following details:

- File Bar:** Shows "Notebook.ipynb", "Metric.py", "Predict.py", and a close button.
- Cell Bar:** Shows the current cell path: "3\_Predictor > Predictor.py > CovidPredictor".
- Code Cell:**

```

1 import torch
2 from PIL import Image
3 from torchvision import transforms
4 import cv2
5 import numpy as np
6
7 def CovidPredictor(imgPath,label):
8     model = torch.load("model.pkl",map_location=torch.device('cpu'))
9     img = Image.open(imgPath)
10    img = cv2.cvtColor(np.float32(img), cv2.COLOR_BGRA2BGR)
11    img = transforms.ToTensor()(img).unsqueeze_(0)
12    output = model(img)
13    pred = output.argmax(dim=1, keepdim=True)
14    if pred.item() == 1 and label == 1:
15        return "True Positive"

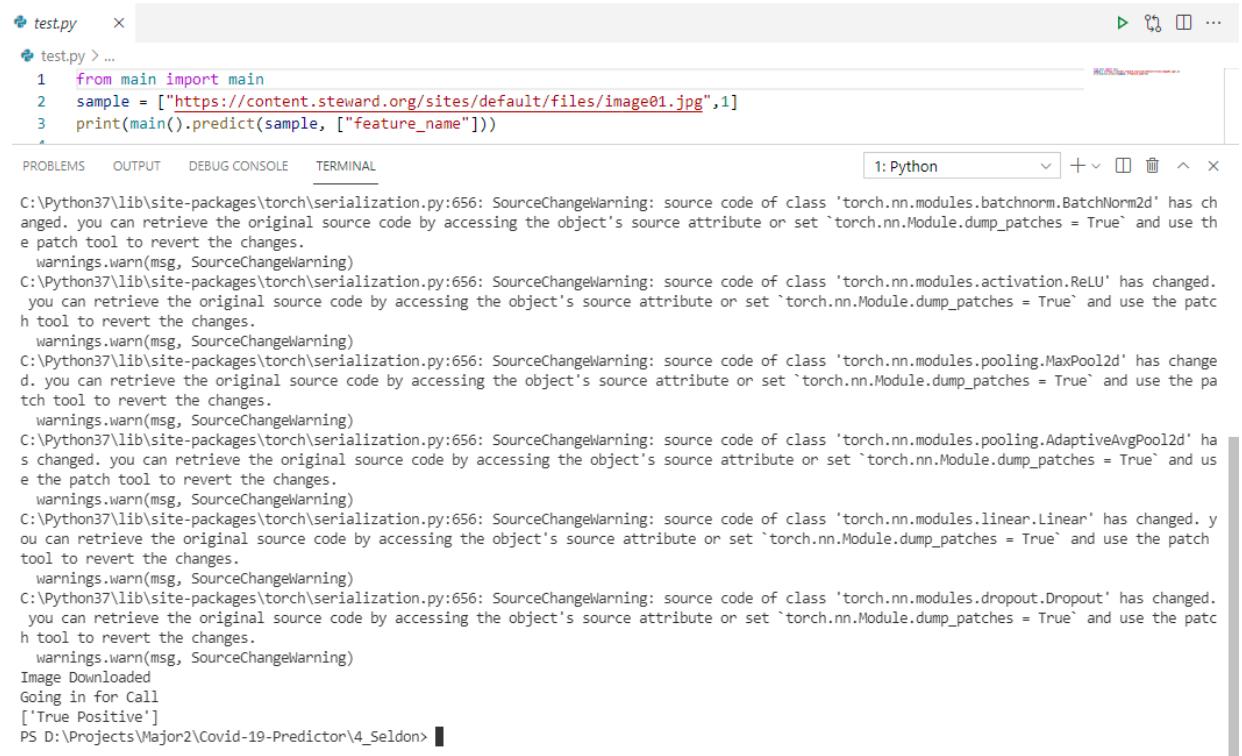
```
- Toolbar:** Shows buttons for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL.
- Output Cell:**

```

tool to revert the changes.
warnings.warn(msg, SourceChangeWarning)
C:\Python37\lib\site-packages\torch\serialization.py:656: SourceChangeWarning: sour
you can retrieve the original source code by accessing the object's source attribu
h tool to revert the changes.
warnings.warn(msg, SourceChangeWarning)
True Positive
PS D:\Projects\Major2\Covid-19-Predictor\3_Predictor>

```

## 6. Migrate the predictor code to Seldon Framework to support REST and gRPC calls.

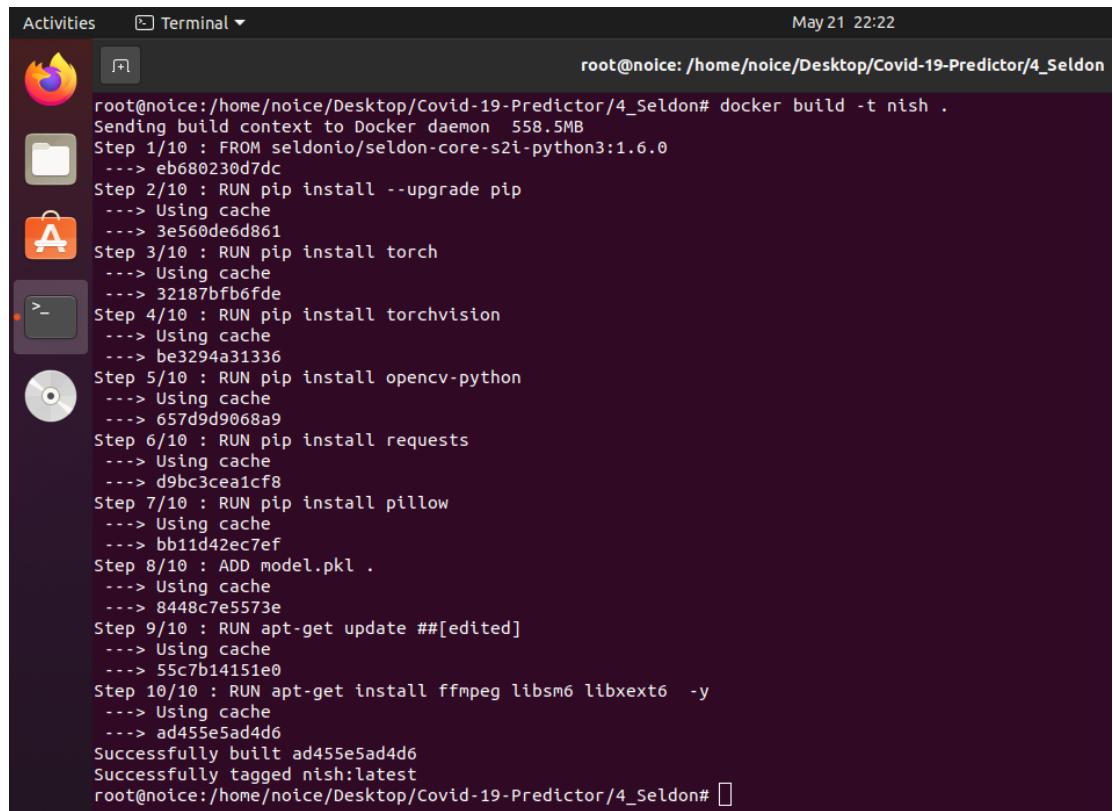


```
test.py > ...
1  from main import main
2  sample = ["https://content.steward.org/sites/default/files/image01.jpg",1]
3  print(main().predict(sample, ["feature_name"]))
4

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: Python + - ×
```

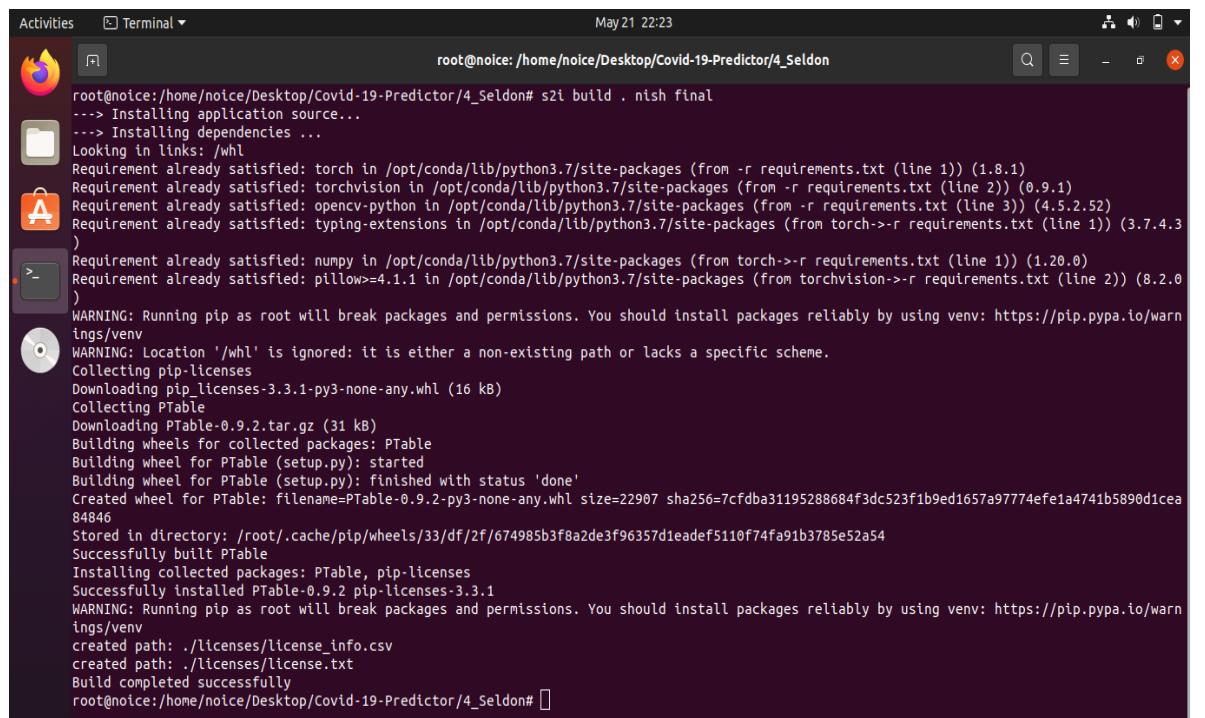
C:\Python37\lib\site-packages\torch\serialization.py:656: SourceChangeWarning: source code of class 'torch.nn.modules.batchnorm.BatchNorm2d' has changed. you can retrieve the original source code by accessing the object's source attribute or set `torch.nn.Module.dump\_patches = True` and use the patch tool to revert the changes.  
warnings.warn(msg, SourceChangeWarning)  
C:\Python37\lib\site-packages\torch\serialization.py:656: SourceChangeWarning: source code of class 'torch.nn.modules.activation.ReLU' has changed. you can retrieve the original source code by accessing the object's source attribute or set `torch.nn.Module.dump\_patches = True` and use the patch tool to revert the changes.  
warnings.warn(msg, SourceChangeWarning)  
C:\Python37\lib\site-packages\torch\serialization.py:656: SourceChangeWarning: source code of class 'torch.nn.modules.pooling.MaxPool2d' has changed. you can retrieve the original source code by accessing the object's source attribute or set `torch.nn.Module.dump\_patches = True` and use the patch tool to revert the changes.  
warnings.warn(msg, SourceChangeWarning)  
C:\Python37\lib\site-packages\torch\serialization.py:656: SourceChangeWarning: source code of class 'torch.nn.modules.pooling.AdaptiveAvgPool2d' has changed. you can retrieve the original source code by accessing the object's source attribute or set `torch.nn.Module.dump\_patches = True` and use the patch tool to revert the changes.  
warnings.warn(msg, SourceChangeWarning)  
C:\Python37\lib\site-packages\torch\serialization.py:656: SourceChangeWarning: source code of class 'torch.nn.modules.linear.Linear' has changed. you can retrieve the original source code by accessing the object's source attribute or set `torch.nn.Module.dump\_patches = True` and use the patch tool to revert the changes.  
warnings.warn(msg, SourceChangeWarning)  
Image Downloaded  
Going in for Call  
['True Positive']  
PS D:\Projects\Major2\Covid-19-Predictor\4\_Seldon>

## 7. Build a base docker image for Seldon functionality



```
root@noice:/home/noice/Desktop/Covid-19-Predictor/4_Seldon# docker build -t nish .
Sending build context to Docker daemon 558.5MB
Step 1/10 : FROM seldonio/seldon-core-s2i-python3:1.6.0
--> eb680230d7dc
Step 2/10 : RUN pip install --upgrade pip
--> Using cache
--> 3e500deed861
Step 3/10 : RUN pip install torch
--> Using cache
--> 32187bfb6fde
Step 4/10 : RUN pip install torchvision
--> Using cache
--> be3294a31336
Step 5/10 : RUN pip install opencv-python
--> Using cache
--> 657d9d9068a9
Step 6/10 : RUN pip install requests
--> Using cache
--> d9bc3cea1cf8
Step 7/10 : RUN pip install pillow
--> Using cache
--> bb11d42ec7ef
Step 8/10 : ADD model.pkl .
--> Using cache
--> 8448c7e5573e
Step 9/10 : RUN apt-get update ##[edited]
--> Using cache
--> 55c7b14151e0
Step 10/10 : RUN apt-get install ffmpeg libsm6 libxext6 -y
--> Using cache
--> ad455e5ad4d6
Successfully built ad455e5ad4d6
Successfully tagged nish:latest
root@noice:/home/noice/Desktop/Covid-19-Predictor/4_Seldon#
```

## 8. Create a seldon image using S2I functionality

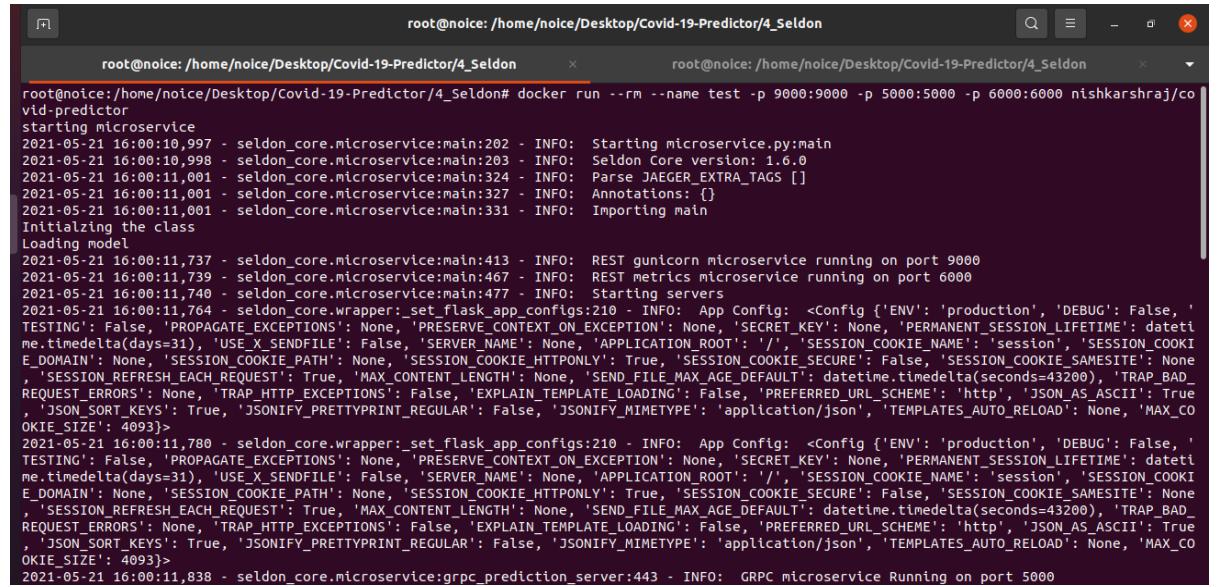


```
root@noice:/home/noice/Desktop/Covid-19-Predictor/4_Seldon# s2i build . nish final
--> Installing application source...
--> Installing dependencies ...
Looking in links: /whl
Requirement already satisfied: torch in /opt/conda/lib/python3.7/site-packages (from -r requirements.txt (line 1)) (1.8.1)
Requirement already satisfied: torchvision in /opt/conda/lib/python3.7/site-packages (from -r requirements.txt (line 2)) (0.9.1)
Requirement already satisfied: opencv-python in /opt/conda/lib/python3.7/site-packages (from -r requirements.txt (line 3)) (4.5.2.52)
Requirement already satisfied: typing-extenstions in /opt/conda/lib/python3.7/site-packages (from torch->-r requirements.txt (line 1)) (3.7.4.3)
Requirement already satisfied: numpy in /opt/conda/lib/python3.7/site-packages (from torch->-r requirements.txt (line 1)) (1.20.0)
Requirement already satisfied: pillow=4.1.1 in /opt/conda/lib/python3.7/site-packages (from torchvision->-r requirements.txt (line 2)) (8.2.0)
WARNING: Running pip as root will break packages and permissions. You should install packages reliably by using venv: https://pip.pypa.io/warnings/venv
WARNING: Location '/whl' is ignored: it is either a non-existing path or lacks a specific scheme.
Collecting pip-licenses
  Downloading pip_licenses-3.3.1-py3-none-any.whl (16 kB)
Collecting PTable
  Downloading PTable-0.9.2.tar.gz (31 kB)
Building wheels for collected packages: PTable
  Building wheel for PTable (setup.py): started
  Building wheel for PTable (setup.py): finished with status 'done'
  Created wheel for PTable: filename=PTable-0.9.2-py3-none-any.whl size=22907 sha256=7cfdba31195288684f3dc523f1b9ed1657a97774efe1a4741b5890d1cea84846
  Stored in directory: /root/.cache/pip/wheels/33/df/2f/674985b3f8a2de3f96357d1eadef5110f74fa91b3785e52a54
Successfully built PTable
Installing collected packages: PTable, pip-licenses
Successfully installed PTable-0.9.2 pip-licenses-3.3.1
WARNING: Running pip as root will break packages and permissions. You should install packages reliably by using venv: https://pip.pypa.io/warnings/venv
created path: ./licenses/license_info.csv
created path: ./licenses/license.txt
Build completed successfully
root@noice:/home/noice/Desktop/Covid-19-Predictor/4_Seldon#
```

## 9. Expose REST Endpoints using Docker on local server with Seldon

The REST Calls are routed to port 9000

The gRPC Calls are routed to port 5000



root@noice:/home/noice/Desktop/Covid-19-Predictor/4\_Seldon# docker run --rm -name test -p 9000:9000 -p 5000:5000 -p 6000:6000 nishkarshraj/coolvid-predictor

starting microservice

2021-05-21 16:00:10,997 - seldon\_core.microservice:main:202 - INFO: Starting microservice.py:main

2021-05-21 16:00:10,998 - seldon\_core.microservice:main:203 - INFO: Seldon Core version: 1.6.0

2021-05-21 16:00:11,001 - seldon\_core.microservice:main:324 - INFO: Parse JAEGER\_EXTRA\_TAGS []

2021-05-21 16:00:11,001 - seldon\_core.microservice:main:327 - INFO: Annotations: {}

2021-05-21 16:00:11,001 - seldon\_core.microservice:main:331 - INFO: Importing main

Initialzing the class

Loading model

2021-05-21 16:00:11,737 - seldon\_core.microservice:main:413 - INFO: REST unicorn microservice running on port 9000

2021-05-21 16:00:11,739 - seldon\_core.microservice:main:467 - INFO: REST metrics microservice running on port 6000

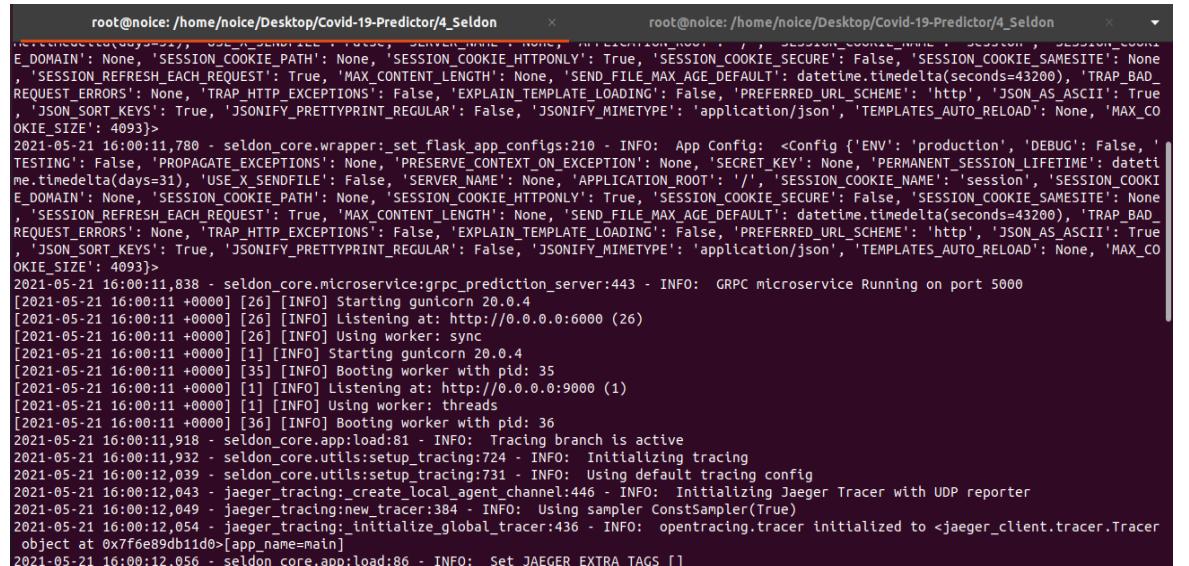
2021-05-21 16:00:11,740 - seldon\_core.microservice:main:477 - INFO: Starting servers

2021-05-21 16:00:11,764 - seldon\_core.wrapper:\_set\_flask\_app\_configs:210 - INFO: App Config: <Config {'ENV': 'production', 'DEBUG': False, 'TESTING': False, 'PROPAGATE\_EXCEPTIONS': None, 'PRESERVE\_CONTEXT\_ON\_EXCEPTION': None, 'SECRET\_KEY': None, 'PERMANENT\_SESSION\_LIFETIME': datetime.timedelta(days=31), 'USE\_X\_SENDFILE': False, 'SERVER\_NAME': None, 'APPLICATION\_ROOT': '/', 'SESSION\_COOKIE\_NAME': 'session', 'SESSION\_COOKIE\_DOMAIN': None, 'SESSION\_COOKIE\_PATH': None, 'SESSION\_COOKIE\_HTTPONLY': True, 'SESSION\_COOKIE\_SECURE': False, 'SESSION\_COOKIE\_SAMESITE': None, 'SESSION\_REFRESH\_EACH\_REQUEST': True, 'MAX\_CONTENT\_LENGTH': None, 'SEND\_FILE\_MAX\_AGE\_DEFAULT': datetime.timedelta(seconds=43200), 'TRAP\_BAD\_REQUEST\_ERRORS': None, 'TRAP\_HTTP\_EXCEPTIONS': False, 'EXPLAIN\_TEMPLATE\_LOADING': False, 'PREFERRED\_URL\_SCHEME': 'http', 'JSON\_AS\_ASCII': True, 'JSON\_SORT\_KEYS': True, 'JSONIFY\_PRETTYPRINT\_REGULAR': False, 'JSONIFY\_MIMETYPE': 'application/json', 'TEMPLATES\_AUTO\_RELOAD': None, 'MAX\_COOKIE\_SIZE': 4093}>

2021-05-21 16:00:11,780 - seldon\_core.wrapper:\_set\_flask\_app\_configs:210 - INFO: App Config: <Config {'ENV': 'production', 'DEBUG': False, 'TESTING': False, 'PROPAGATE\_EXCEPTIONS': None, 'PRESERVE\_CONTEXT\_ON\_EXCEPTION': None, 'SECRET\_KEY': None, 'PERMANENT\_SESSION\_LIFETIME': datetime.timedelta(days=31), 'USE\_X\_SENDFILE': False, 'SERVER\_NAME': None, 'APPLICATION\_ROOT': '/', 'SESSION\_COOKIE\_NAME': 'session', 'SESSION\_COOKIE\_DOMAIN': None, 'SESSION\_COOKIE\_PATH': None, 'SESSION\_COOKIE\_HTTPONLY': True, 'SESSION\_COOKIE\_SECURE': False, 'SESSION\_COOKIE\_SAMESITE': None, 'SESSION\_REFRESH\_EACH\_REQUEST': True, 'MAX\_CONTENT\_LENGTH': None, 'SEND\_FILE\_MAX\_AGE\_DEFAULT': datetime.timedelta(seconds=43200), 'TRAP\_BAD\_REQUEST\_ERRORS': None, 'TRAP\_HTTP\_EXCEPTIONS': False, 'EXPLAIN\_TEMPLATE\_LOADING': False, 'PREFERRED\_URL\_SCHEME': 'http', 'JSON\_AS\_ASCII': True, 'JSON\_SORT\_KEYS': True, 'JSONIFY\_PRETTYPRINT\_REGULAR': False, 'JSONIFY\_MIMETYPE': 'application/json', 'TEMPLATES\_AUTO\_RELOAD': None, 'MAX\_COOKIE\_SIZE': 4093}>

2021-05-21 16:00:11,838 - seldon\_core.microservice:grpc\_prediction\_server:443 - INFO: GRPC microservice Running on port 5000

Set Jaeger Tags message defines the successful hosting of Seldon application



root@noice:/home/noice/Desktop/Covid-19-Predictor/4\_Seldon# docker run --rm -name test -p 9000:9000 -p 5000:5000 -p 6000:6000 nishkarshraj/coolvid-predictor

2021-05-21 16:00:11,780 - seldon\_core.wrapper:\_set\_flask\_app\_configs:210 - INFO: App Config: <Config {'ENV': 'production', 'DEBUG': False, 'TESTING': False, 'PROPAGATE\_EXCEPTIONS': None, 'PRESERVE\_CONTEXT\_ON\_EXCEPTION': None, 'SECRET\_KEY': None, 'PERMANENT\_SESSION\_LIFETIME': datetime.timedelta(days=31), 'USE\_X\_SENDFILE': False, 'SERVER\_NAME': None, 'APPLICATION\_ROOT': '/', 'SESSION\_COOKIE\_NAME': 'session', 'SESSION\_COOKIE\_DOMAIN': None, 'SESSION\_COOKIE\_PATH': None, 'SESSION\_COOKIE\_HTTPONLY': True, 'SESSION\_COOKIE\_SECURE': False, 'SESSION\_COOKIE\_SAMESITE': None, 'SESSION\_REFRESH\_EACH\_REQUEST': True, 'MAX\_CONTENT\_LENGTH': None, 'SEND\_FILE\_MAX\_AGE\_DEFAULT': datetime.timedelta(seconds=43200), 'TRAP\_BAD\_REQUEST\_ERRORS': None, 'TRAP\_HTTP\_EXCEPTIONS': False, 'EXPLAIN\_TEMPLATE\_LOADING': False, 'PREFERRED\_URL\_SCHEME': 'http', 'JSON\_AS\_ASCII': True, 'JSON\_SORT\_KEYS': True, 'JSONIFY\_PRETTYPRINT\_REGULAR': False, 'JSONIFY\_MIMETYPE': 'application/json', 'TEMPLATES\_AUTO\_RELOAD': None, 'MAX\_COOKIE\_SIZE': 4093}>

2021-05-21 16:00:11,780 - seldon\_core.wrapper:\_set\_flask\_app\_configs:210 - INFO: App Config: <Config {'ENV': 'production', 'DEBUG': False, 'TESTING': False, 'PROPAGATE\_EXCEPTIONS': None, 'PRESERVE\_CONTEXT\_ON\_EXCEPTION': None, 'SECRET\_KEY': None, 'PERMANENT\_SESSION\_LIFETIME': datetime.timedelta(days=31), 'USE\_X\_SENDFILE': False, 'SERVER\_NAME': None, 'APPLICATION\_ROOT': '/', 'SESSION\_COOKIE\_NAME': 'session', 'SESSION\_COOKIE\_DOMAIN': None, 'SESSION\_COOKIE\_PATH': None, 'SESSION\_COOKIE\_HTTPONLY': True, 'SESSION\_COOKIE\_SECURE': False, 'SESSION\_COOKIE\_SAMESITE': None, 'SESSION\_REFRESH\_EACH\_REQUEST': True, 'MAX\_CONTENT\_LENGTH': None, 'SEND\_FILE\_MAX\_AGE\_DEFAULT': datetime.timedelta(seconds=43200), 'TRAP\_BAD\_REQUEST\_ERRORS': None, 'TRAP\_HTTP\_EXCEPTIONS': False, 'EXPLAIN\_TEMPLATE\_LOADING': False, 'PREFERRED\_URL\_SCHEME': 'http', 'JSON\_AS\_ASCII': True, 'JSON\_SORT\_KEYS': True, 'JSONIFY\_PRETTYPRINT\_REGULAR': False, 'JSONIFY\_MIMETYPE': 'application/json', 'TEMPLATES\_AUTO\_RELOAD': None, 'MAX\_COOKIE\_SIZE': 4093}>

2021-05-21 16:00:11,838 - seldon\_core.microservice:grpc\_prediction\_server:443 - INFO: GRPC microservice Running on port 5000

[2021-05-21 16:00:11 +0000] [26] [INFO] Starting unicorn 20.0.4

[2021-05-21 16:00:11 +0000] [26] [INFO] Listening at: http://0.0.0.0:6000 (26)

[2021-05-21 16:00:11 +0000] [26] [INFO] Using worker: sync

[2021-05-21 16:00:11 +0000] [1] [INFO] Starting unicorn 20.0.4

[2021-05-21 16:00:11 +0000] [35] [INFO] Booting worker with pid: 35

[2021-05-21 16:00:11 +0000] [1] [INFO] Listening at: http://0.0.0.0:9000 (1)

[2021-05-21 16:00:11 +0000] [1] [INFO] Using worker: threads

[2021-05-21 16:00:11 +0000] [36] [INFO] Booting worker with pid: 36

2021-05-21 16:00:11,918 - seldon.core.app:load:81 - INFO: Tracing branch is active

2021-05-21 16:00:11,932 - seldon.core.util:setup\_tracing:724 - INFO: Initializing tracing

2021-05-21 16:00:12,039 - seldon.core.util:setup\_tracing:731 - INFO: Using default tracing config

2021-05-21 16:00:12,043 - jaeger\_tracing:create\_local\_agent\_channel:446 - INFO: Initializing Jaeger Tracer with UDP reporter

2021-05-21 16:00:12,049 - jaeger\_tracing:new\_tracer:384 - INFO: Using sampler ConstSampler(True)

2021-05-21 16:00:12,054 - jaeger\_tracing:\_initialize\_global\_tracer:436 - INFO: opentracing.tracer initialized to <jaeger\_client.Tracer object at 0x7f6e89db11d0>[app\_name=main]

2021-05-21 16:00:12,056 - seldon.core.app:load:86 - INFO: Set JAEGER\_EXTRA\_TAGS []

```
root@noice:/home/noice/Desktop/Covid-19-Predictor/4_Seldon
/opt/conda/lib/python3.7/site-packages/torch/serialization.py:656: SourceChangeWarning: source code of class 'torch.nn.modules.conv.Conv2d' has changed. you can retrieve the original source code by accessing the object's source attribute or set 'torch.nn.Module.dump_patches = True' and use the patch tool to revert the changes.
    warnings.warn(msg, SourceChangeWarning)
/opt/conda/lib/python3.7/site-packages/torch/serialization.py:656: SourceChangeWarning: source code of class 'torch.nn.modules.batchnorm.BatchNorm2d' has changed. you can retrieve the original source code by accessing the object's source attribute or set 'torch.nn.Module.dump_patches = True' and use the patch tool to revert the changes.
    warnings.warn(msg, SourceChangeWarning)
/opt/conda/lib/python3.7/site-packages/torch/serialization.py:656: SourceChangeWarning: source code of class 'torch.nn.modules.activation.ReLU' has changed. you can retrieve the original source code by accessing the object's source attribute or set 'torch.nn.Module.dump_patches = True' and use the patch tool to revert the changes.
    warnings.warn(msg, SourceChangeWarning)
/opt/conda/lib/python3.7/site-packages/torch/serialization.py:656: SourceChangeWarning: source code of class 'torch.nn.modules.pooling.MaxPool2d' has changed. you can retrieve the original source code by accessing the object's source attribute or set 'torch.nn.Module.dump_patches = True' and use the patch tool to revert the changes.
    warnings.warn(msg, SourceChangeWarning)
/opt/conda/lib/python3.7/site-packages/torch/serialization.py:656: SourceChangeWarning: source code of class 'torch.nn.modules.pooling.AdaptiveAvgPool2d' has changed. you can retrieve the original source code by accessing the object's source attribute or set 'torch.nn.Module.dump_patches = True' and use the patch tool to revert the changes.
    warnings.warn(msg, SourceChangeWarning)
/opt/conda/lib/python3.7/site-packages/torch/serialization.py:656: SourceChangeWarning: source code of class 'torch.nn.modules.linear.Linear' has changed. you can retrieve the original source code by accessing the object's source attribute or set 'torch.nn.Module.dump_patches = True' and use the patch tool to revert the changes.
    warnings.warn(msg, SourceChangeWarning)
/opt/conda/lib/python3.7/site-packages/torch/serialization.py:656: SourceChangeWarning: source code of class 'torch.nn.modules.dropout.Dropout' has changed. you can retrieve the original source code by accessing the object's source attribute or set 'torch.nn.Module.dump_patches = True' and use the patch tool to revert the changes.
    warnings.warn(msg, SourceChangeWarning)
2021-05-21 16:00:44,049 - jaeger_tracing:report_span:73 - INFO: Reporting span 4e0677fbca01179d:f2a1bb1398b494c6:0:1 main.Predict
```

Whenever the endpoint is hit, the host shows a log of the invocation of the predict function.

Hit the endpoint with a cURL POST call with bash or POSTMAN client.

```
root@noice:/home/noice/Desktop/Covid-19-Predictor/4_Seldon
root@noice:/home/noice/Desktop/Covid-19-Predictor/4_Seldon
root@noice:/home/noice/Desktop/Covid-19-Predictor/4_Seldon# curl -X POST -H 'Content-Type: application/json' -d '{"data":{"ndarray":["https://content.steward.org/sites/default/files/image01.jpg",1]}}' http://localhost:9000/api/v1.0/predictions
{"data":{"names":[],"ndarray":["True Positive"]}, "meta":{}}
root@noice:/home/noice/Desktop/Covid-19-Predictor/4_Seldon#
```

10. Host ModelDB on a local server for Model storage.

Docker-compose is written to launch the entire stack of Model DB including frontend, backend, postgres and proxy server.

```
root@noice:/home/noice/Desktop/Covid-19-Predictor/5_AIOPs          root@noice:/home/noice
root@noice:/home/noice/Desktop/Covid-19-Predictor/5_AIOPs x           root@noice:/home/noice x
root@noice:/home/noice/Desktop/Covid-19-Predictor/5_AIOPs# docker-compose up
Creating network "5_ailops_modeldb_network" with the default driver
Creating modeldb-postgres ... done
Creating modeldb-backend ... done
Creating modeldb-proxy ... done
Creating modeldb-graphql ... done
Creating modeldb-frontend ... done
Attaching to modeldb-postgres, modeldb-backend, modeldb-graphql, modeldb-proxy, modeldb-frontend
modeldb-graphql  | 2021/05/21 15:34:27 SERVER_HTTP_PORT : 4000
modeldb-postgres | The files belonging to this database system will be owned by user "postgres".
modeldb-postgres | This user must also own the server process.
modeldb-postgres | 
modeldb-postgres | The database cluster will be initialized with locale "en_US.utf8".
modeldb-postgres | The default database encoding has accordingly been set to "UTF8".
modeldb-postgres | The default text search configuration will be set to "english".
modeldb-postgres | 
modeldb-postgres | Data page checksums are disabled.
modeldb-postgres | 
modeldb-postgres | 2021/05/21 15:34:27 MDB_ADDRESS : modeldb-backend:8085
modeldb-postgres | fixing permissions on existing directory /var/lib/postgresql/data/pgdata ... ok
modeldb-proxy    | 2021/05/21 15:34:27 SERVER_HTTP_PORT : 8080
modeldb-proxy    | 2021/05/21 15:34:27 Starting verta-backend proxy on port : 8080
modeldb-postgres| creating subdirectories ... ok
modeldb-postgres| selecting dynamic shared memory implementation ... posix
modeldb-postgres| selecting default max_connections ... 100
modeldb-postgres| selecting default shared_buffers ... 128MB
modeldb-postgres| selecting default time zone ... Etc/UTC
modeldb-postgres| creating configuration files ... ok
modeldb-postgres| running bootstrap script ... ok
modeldb-postgres| performing post-bootstrap initialization ... ok
```

```
root@noice:/home/noice/Desktop/Covid-19-Predictor/5_AIOps
root@noice:/home/noice/Desktop/Covid-19-Predictor/5_AIOps
modeldb-postgres | /usr/local/bin/docker-entrypoint.sh: ignoring /docker-entrypoint-initdb.d/*
modeldb-postgres |
modeldb-postgres | 2021-05-21 15:34:33.206 UTC [47] LOG: received fast shutdown request
modeldb-postgres | waiting for server to shut down...2021-05-21 15:34:33.221 UTC [47] LOG: aborting any active transactions
modeldb-postgres | 2021-05-21 15:34:33.278 UTC [47] LOG: background worker "logical replication launcher" (PID 54) exited with exit code 1
modeldb-postgres | 2021-05-21 15:34:33.291 UTC [49] LOG: shutting down
modeldb-postgres | 2021-05-21 15:34:33.361 UTC [47] LOG: database system is shut down
modeldb-postgres |   done
modeldb-postgres |   server stopped
modeldb-postgres |
modeldb-postgres | PostgreSQL init process complete; ready for start up.
modeldb-postgres |
modeldb-postgres | 2021-05-21 15:34:33.632 UTC [1] LOG: starting PostgreSQL 13.3 (Debian 13.3-1.pgdg100+1) on x86_64-pc-linux-gnu, compiled by gcc (Debian 8.3.0-6) 8.3.0, 64-bit
modeldb-postgres | 2021-05-21 15:34:33.640 UTC [1] LOG: listening on IPv4 address "0.0.0.0", port 5432
modeldb-postgres | 2021-05-21 15:34:33.640 UTC [1] LOG: listening on IPv6 address ":::", port 5432
modeldb-postgres | 2021-05-21 15:34:33.695 UTC [1] LOG: listening on Unix socket "/var/run/postgresql/.s.PGSQL.5432"
modeldb-postgres | 2021-05-21 15:34:33.731 UTC [66] LOG: database system was shut down at 2021-05-21 15:34:33 UTC
modeldb-postgres | 2021-05-21 15:34:33.804 UTC [1] LOG: database system is ready to accept connections
modeldb-frontend | [nodemon] 1.19.4
modeldb-frontend | [nodemon] to restart at any time, enter `rs`
modeldb-frontend | [nodemon] watching dir(s): ***!
modeldb-frontend | [nodemon] watching extensions: js,mjs,json
modeldb-frontend | [nodemon] starting `node server.js`
modeldb-frontend | [HPM] Proxy created: / -> http://modeldb-proxy:8080
modeldb-frontend | [HPM] Proxy rewrite rule created: "^\api/v1/modeldb" -> "/v1"
modeldb-frontend | [HPM] Proxy created: / -> http://modeldb-backend:8086
modeldb-frontend | [HPM] Proxy created: / -> http://modeldb-graphql:4000
modeldb-frontend | [HPM] Proxy rewrite rule created: "^\api/v1/graphql/" -> "/"
```



```

root@noice: /home/noice          root@noice: /home/noice/Desktop/Covid-19-Predictor/5_AIOps

modeldb-backend | {"thread": "main", "level": "INFO", "loggerName": "ai.verta.modeldb.cron.jobs.CronJobUtils", "message": "Delete entities cron job scheduled successfully", "endOfBatch": false, "loggerFqn": "org.apache.logging.log4j.spi.AbstractLogger", "instant": {"epochSecond": 1621611463, "nanoOfSecond": 43000000}, "threadId": 1, "threadPriority": 5, "hostName": "59382ff8810e", "kubernetes.podIP": ""}
modeldb-backend | {"thread": "main", "level": "INFO", "loggerName": "ai.verta.modeldb.cron.jobs.CronJobUtils", "message": "Exit from CronJobUtils : initializeBasedOnConfig()", "endOfBatch": false, "loggerFqn": "org.apache.logging.log4j.spi.AbstractLogger", "instant": {"epochSecond": 1621611463, "nanoOfSecond": 44000000}, "threadId": 1, "threadPriority": 5, "hostName": "59382ff8810e", "kubernetes.podIP": ""}
modeldb-backend | {"thread": "main", "level": "INFO", "loggerName": "ai.verta.modeldb.App", "message": "Backend server started listening on 8085", "endOfBatch": false, "loggerFqn": "org.apache.logging.log4j.spi.AbstractLogger", "instant": {"epochSecond": 1621611463, "nanoOfSecond": 418000000}, "threadId": 1, "threadPriority": 5, "hostName": "59382ff8810e", "kubernetes.podIP": ""}
modeldb-frontend | /personal/projects
modeldb-frontend | /static/css/2.722a1fc9.chunk.css
modeldb-frontend | /static/css/main.3a7bd0dc.chunk.css
modeldb-frontend | /static/js/2.2beaca20.chunk.js
modeldb-frontend | /static/js/main.a839b627.chunk.js
modeldb-frontend | /static/media/verta logo.b2461278.svg
modeldb-frontend | /api/v1/modeldb/hydratedData/findHydratedProjects
modeldb-frontend | Requesting /api/v1/modeldb/hydratedData/findHydratedProjects
modeldb-backend | {"thread": "main", "level": "INFO", "loggerName": "ai.verta.modeldb.ModelDBAuthInterceptor", "message": "met hodName: ai.verta.modeldb.HydratedService/findHydratedProjects", "endOfBatch": false, "loggerFqn": "org.apache.logging.log4j.spi.AbstractLogger", "instant": {"epochSecond": 1621611655, "nanoOfSecond": 235000000}, "threadId": 44, "threadPriority": 5, "hostName": "59382ff8810e", "kubernetes.podIP": ""}
modeldb-backend | {"thread": "grpc-default-executor-0", "level": "INFO", "loggerName": "ai.verta.modeldb.ProjectDAOImpl", "message": "Projects final query : select pr from ProjectEntity as pr where pr.deleted=:param0 order by pr.date_updated desc", "endOfBatch": false, "loggerFqn": "org.apache.logging.log4j.spi.AbstractLogger", "instant": {"epochSecond": 1621611656, "nanoOfSecond": 491000000}, "threadId": 44, "threadPriority": 5, "hostName": "59382ff8810e", "kubernetes.podIP": ""}
modeldb-backend | {"thread": "grpc-default-executor-0", "level": "DEBUG", "loggerName": "ai.verta.modeldb.advancedService.AdvancedServiceImpl", "message": "ProjectPaginationDTO record count : 0", "endOfBatch": false, "loggerFqn": "org.apache.logging.log4j.spi.AbstractLogger", "instant": {"epochSecond": 1621611656, "nanoOfSecond": 664000000}, "threadId": 44, "threadPriority": 5, "hostName": "59382ff8810e", "kubernetes.podIP": ""}
modeldb-frontend | Returning 200 OK; 2b sent

```

When everything is up and running, the frontend returns HTTP 200 status code on the logging server.

The screenshot shows the Verta.ai web application interface. At the top, there's a navigation bar with the Verta.ai logo, a search bar, and a 'Create' button. Below the navigation, there are three main categories: 'Projects', 'Datasets', and 'Repositories'. The 'Projects' section is currently active and displays a message 'No projects'. On the left side, there are links for 'Docs' and the 'Verta.ai' logo. The overall layout is clean and modern, typical of a cloud-based data management platform.

## 11. Log Model in ModelDB with Python SDK

```

Activities Terminal May 21 21:22
root@noice: /home/noice          root@noice: /home/noice/Desktop/Covid-19-Predictor/5_AIOps
root@noice: /home/noice# python3 ModelDB.py
connection successfully established
got existing Project: MajorII
got existing Experiment: CovidPredictor
got existing ExperimentRun: Version 1
upload complete (model.pkl)
Username: name: MajorII
url: http://localhost:3000/personal/projects/876a096f-6e3a-41d6-bf66-89b1b9eb6e65/summary
Experiment: CovidPredictor
Experiment Run: Version 1
Serialization: pickle
Library: pytorch
root@noice:/home/noice/Desktop/Covid-19-Predictor/5_AIOps#

```

The screenshot shows the Verfa.ai interface for a project named 'Majorll'. The top navigation bar includes a back arrow, forward arrow, refresh button, and a search bar with a magnifying glass icon. The URL 'localhost:3000/personal/projects' is visible. On the left, a sidebar menu has 'Projects' selected, along with 'Datasets' and 'Repositories'. A 'Create' button is in the top right. The main content area displays the project details: 'Majorll', 'Description', 'Tags' (with a trash bin icon), and creation information ('Created: 21/5/2021' and 'Updated: 21/5/2021'). Below the main content is a navigation bar with icons for 'Docs' and 'Verta.ai'.

This screenshot shows the 'Summary' tab for the 'Majorll' project. The top navigation bar is identical to the previous screenshot. The main content area shows the 'DESCRIPTION' section with a 'Description' field containing placeholder text. Below it is an 'ID' field with the value '876a096f-6e3a-41d6-bf66-89b1b9eb6e65' and a 'Copy' button. At the bottom is a file preview for 'README.md'.

This screenshot shows the 'Experiment Runs' tab for the 'Majorll' project. The top navigation bar is identical. The main content area includes a 'Compare' button, filter options ('Filter Runs' with 'Drag and drop to filter' and '+ Add Filter'), and a table for experiment runs. The table columns are 'Actions', 'Run Summary', 'Metrics', 'Hyperparameters', and 'Artifacts'. One run is listed: 'Version 1' with 'Experiment Name: CovidPredictor', 'Timestamp: 21/05/2021, 9:17:17 pm', and an artifact link 'model.pkl'.

X

## Artifact

Key model.pkl

Type MODEL

Path dfe10ee2709150e3eb4e5172f0b2c92909f995c8257fa  
67ab72218534241f0d4/model.pkl

[Download Artifact](#)

[Delete Artifact](#)

```
root@noice:/home/noice/Desktop/mlflow-learning/sklearn-logistic-regression
root@noice:/home/noice/Desktop/Covid-19-Predictor/4_Seldon      x  root@noice:/home/noice/Desktop/mlflow-learning/sklearn-logistic-regression
mlflow$ mlflow ui --backend-store-uri sqlite:///mlruns.db
[2021-05-21 21:28:26 +0530] [7626] [INFO] Starting gunicorn 20.1.0
[2021-05-21 21:28:26 +0530] [7626] [INFO] Listening at: http://127.0.0.1:5000 (7626)
[2021-05-21 21:28:26 +0530] [7626] [INFO] Using worker: sync
[2021-05-21 21:28:26 +0530] [7628] [INFO] Booting worker with pid: 7628
```

**mlflow** Experiments Models GitHub Docs

Experiments + <

**Default**

Search Experiments

**Default** Edit Delete

Track machine learning training runs in an experiment. [Learn more](#)

X

Experiment ID: 0 Artifact Location: ./mlruns/0

▼ Notes Edit

None

Search Runs:  Filter Search Clear

Showing 1 matching run Compare Delete Download CSV

Columns

	Start Time	Run Name	User	Source	Version	Models
<input type="checkbox"/>	2021-05-21 21:27:42	-	root	final-log.py	bcf968	sklearn - LogisticRe.../1

**mlflow** Experiments Models GitHub Docs

Default > Run 523a69bdd6d54b66914c9e36132c984d ▾

Date: 2021-05-21 21:27:42 Source: final-log.py Git Commit: bcf96872ae982533b607a8f3c939133d13af4dd5

User: root Duration: 10.2s Status: FINISHED

▼ Notes Edit

None

▼ Parameters

Name	Value
------	-------

▼ Metrics

Name	Value
------	-------

## References

1. Dong, E., Du, H. & Gardner, L. An interactive web-based dashboard to track COVID-19 in real time. *Lancet Infect. Dis.* [https://doi.org/10.1016/S1473-3099\(20\)30120-1](https://doi.org/10.1016/S1473-3099(20)30120-1) (2020).
2. Gozes, O. et al. Rapid AI development cycle for the coronavirus (COVID-19) pandemic: initial results for automated detection & patient monitoring using deep learning CT image analysis. *arXiv e-prints* 2003, arXiv:2003.05037 (2020).
3. Song, Y. et al. Deep learning Enables Accurate Diagnosis of Novel Coronavirus (COVID-19) with CT images. *medRxiv* <https://doi.org/10.1101/2020.02.23.20026930> (2020).
4. Wang, S. et al. A deep learning algorithm using CT images to screen for CoronaVirus Disease (COVID-19). *medRxiv*, <https://doi.org/10.1101/2020.02.14.20023028> (2020).
5. Jin, C. et al. Development and evaluation of an AI system for COVID-19 diagnosis. *medRxiv*, <https://doi.org/10.1101/2020.03.20.20039834> (2020).
6. Punn, N. S. & Agarwal, S. Automated diagnosis of COVID-19 with limited posteroanterior chest X-ray images using fine-tuned deep neural networks. *arXiv:2004.11676 [cs, eess]* (2020).
7. Tostmann, A. et al. Strong associations and moderate predictive value of early symptoms for SARS-CoV-2 test positivity among healthcare workers, the Netherlands, March 2020. *Eurosurveillance* 25, 2000508 (2020).
8. Feng, C. et al. A novel triage tool of artificial intelligence assisted diagnosis aid system for suspected COVID-19 pneumonia in fever clinics. *medRxiv*, <https://doi.org/10.1101/2020.03.19.20039099> (2020).
9. Punn, N. S., Sonbhadra, S. K. & Agarwal, S. COVID-19 Epidemic Analysis using Machine Learning and Deep Learning Algorithms. *medRxiv*, <https://doi.org/10.1101/2020.04.08.20057679> (2020).
10. Mei, X. et al. Artificial intelligence–enabled rapid diagnosis of patients with COVID-19. *Nat. Med.* 26, 1224–1228 (2020).
11. COVID-19-Government Data. <https://data.gov.il/dataset/covid-19> (2020).

12. The Novel Coronavirus Israel Ministry of Health.  
<https://govextra.gov.il/ministry-of-health/corona/corona-virus-en/> (2020).
13. COVID-19-Government Data Information.  
<https://data.gov.il/dataset/covid-19/resource/3f5c975e-7196-454b-8c5b-ef8581f78db/download/-readme.pdf> (2020).
14. Struyf, T. et al. Signs and symptoms to determine if a patient presenting in primary care or hospital outpatient settings has COVID-19 disease. Cochrane Database Syst. Rev., <https://doi.org/10.1002/14651858.CD013665> (2020).
15. Liu, Y., Gayle, A. A., Wilder-Smith, A. & Rocklöv, J. The reproductive number of COVID-19 is higher compared to SARS coronavirus. J. Travel Med. 27 (2020).
16. Jin, J.-M. et al. Gender Differences in Patients With COVID-19: Focus on Severity and Mortality. Front. Public Health 8 (2020).
17. BMJ GH Blogs. Sex, gender and COVID-19: Disaggregated data and health disparities. BMJ Global Health blog  
<https://blogs.bmj.com/bmjgh/2020/03/24/sex-gender-and-covid-19-disaggregated-data-and-health-disparities/> (2020).
18. Whittington, A. M. et al. Coronavirus: rolling out community testing for COVID-19 in the NHS. BMJ Opinion  
<https://blogs.bmj.com/bmj/2020/02/17/coronavirus-rolling-out-community-testing-for-covid-19-in-the-nhs/> (2020).
19. Menni, C. et al. Real-time tracking of self-reported symptoms to predict potential COVID-19. Nat. Med. 26, 1037–1040 (2020).
20. Hastie, T., Tibshirani, R. & Friedman, J. In The Elements of Statistical Learning: Data Mining, Inference, and Prediction (eds. Hastie, T., Tibshirani, R. & Friedman, J.) 337–387 (Springer, 2009).
21. Fernández-Delgado, M., Cernadas, E., Barro, S. & Amorim, D. Do we need hundreds of classifiers to solve real world classification problems? J. Mach. Learn. Res. 15, 3133–3181 (2014).
22. Omar, K. B. A. XGBoost and LGBM for Porto Seguro's Kaggle challenge: A comparison Semester Project (ETH Zurich, 2018).

23. Josse, J., Prost, N., Scornet, E. & Varoquaux, G. On the consistency of supervised learning with missing values. arXiv:1902.06931 [cs, math, stat] (2019).
24. Chen, T. & Guestrin, C. XGBoost: A scalable tree boosting system. in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining 785–794 (Association for Computing Machinery, 2016).
25. Ke, G. et al. In Advances in Neural Information Processing Systems 30 (eds. Guyon, I. et al.) 3146–3154 (Curran Associates, Inc., 2017).
26. Raskutti, G., Wainwright, M. J. & Yu, B. Early stopping for nonparametric regression: An optimal data-dependent stopping rule. in 2011 49th Annual Allerton Conference on Communication, Control, and Computing (Allerton) 1318–1325 (2011).
27. Lundberg, S. & Lee, S.-I. A Unified Approach to Interpreting Model Predictions. arXiv:1705.07874 [cs, stat] (2017).
28. Lundberg, S. M. et al. Explainable machine-learning predictions for the prevention of hypoxaemia during surgery. Nat. Biomed. Eng. 2, 749–760 (2018).
29. Efron, B. & Tibshirani, R. J. An Introduction to the Bootstrap. (CRC press, 1994).

---

Signature of HoD

---

Signature of Mentor