

PROJECT DOCUMENT

Date	5 November 2022
Team ID	PNT2022TMID32270
Project Name	Flight Delay Prediction Using Machine Learning

DEVELOPMENT PHASE:

Creating IBM cloud account & Required Resources

Deploy our model in IBM Watson

Creating Dashboard using HTML/CSS

Create web app and Hosting in falk

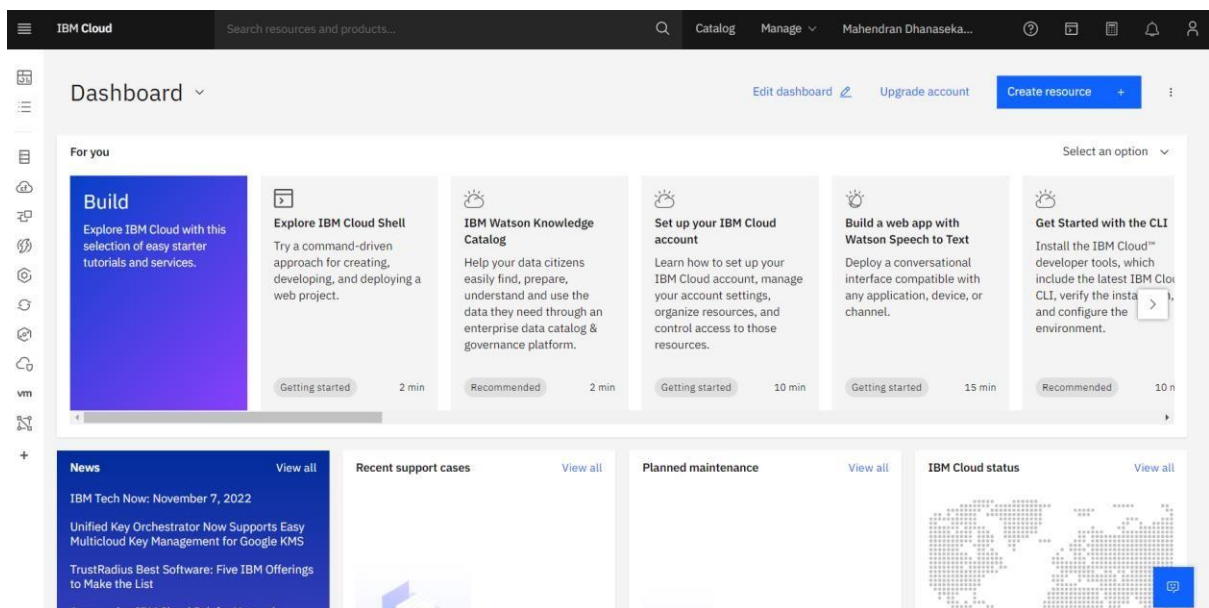
Testing web app

Creating IBM cloud account & Required Resources:

Creating IBM cloud account:

Frist, need to create IBM Cloud account by using SI Mail Id and SI Password which is provided by IBM in profile.

Below dashboard of an account after created,



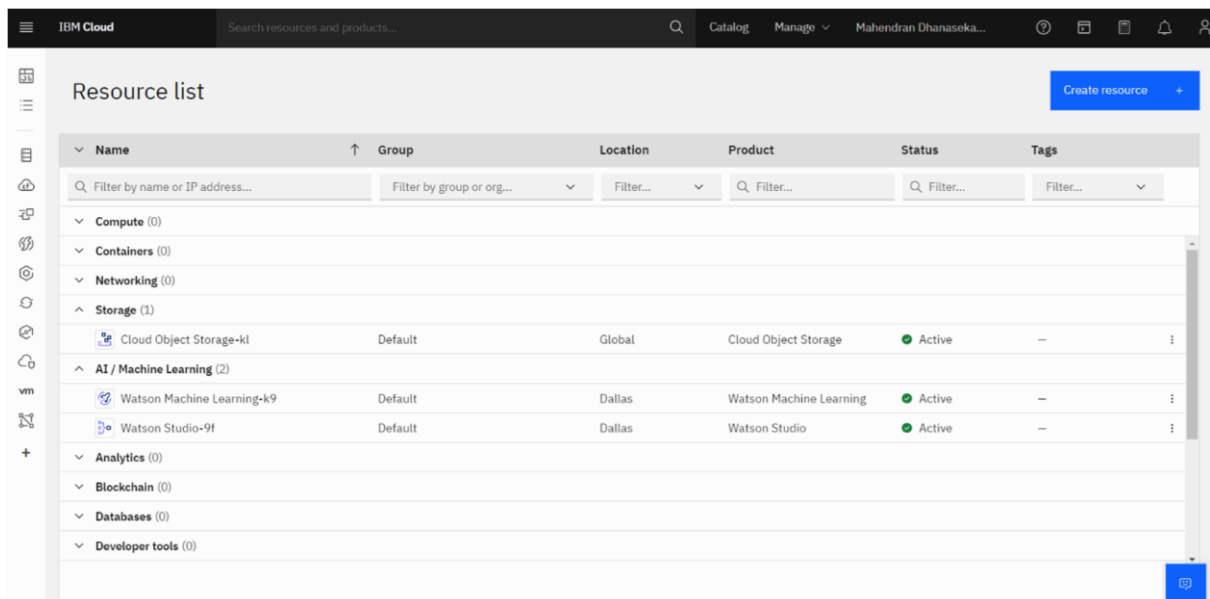
Creating IBM Cloud Required Resources:

After creating IBM cloud account, to deploy ML model, need to create following resources such as,
Cloud Object Storage

Watson Machine Learning

Watson Studio

After created above resources Resource List of an account is displayed as follow,



Name	Group	Location	Product	Status	Tags
Filter by name or IP address... Filter by group or org... Filter... Filter... Filter... Filter...					
Compute (0)					
Containers (0)					
Networking (0)					
Storage (1)					
Cloud Object Storage-kl	Default	Global	Cloud Object Storage	Active	—
AI / Machine Learning (2)					
Watson Machine Learning-k9	Default	Dallas	Watson Machine Learning	Active	—
Watson Studio-9f	Default	Dallas	Watson Studio	Active	—
Analytics (0)					
Blockchain (0)					
Databases (0)					
Developer tools (0)					

All the resource are in active state.

All the required cloud resources are created successfully.

Deploy our model in IBM Watson:

To deploy ML model in IBM cloud, need to create project in IBM Watson. After successful creation of project import .ipynb file of sprint-1 which ML models are build in Jupyter notebook.

Upload required datasets and import it.

Deploy model using following code,

```
!pip install -U ibm-watson-machine-learning from
ibm_watson_machine_learning import APIClient
import json import numpy as np wml_cred={
    "apikey":"okbr7ARnOQjyplTOyvNFC2QVkJCF6q7afpci065Hucby8",
    "url":"https://us-south.ml.cloud.ibm.com"
}
wml_clients=APIClient(wml_cred) wml_clients.spaces.list()
```

```
space_id="6d7c1218-3aca-4256-be3d-d610732530b1" wml_clients.set.default_space(space_id)
wml_clients.software_specifications.list(500)
MODEL_NAME="randomforest"
DEPLOYMENT_NAME="rf_deployment"
DEMO_MODEL=rf
soft_sepc_id=wml_clients.software_specifications.get_id_by_name("runtime-22.1-py3.9")
```

In [115]:

```
model_props={ wml_clients.repository.ModelMetaNames.NAME:MODEL_NAME,
wml_clients.repository.ModelMetaNames.TYPE:"scikit-learn_1.0",
wml_clients.repository.ModelMetaNames.SOFTWARE_SPEC_UID: soft_sepc_id

}
```

In [116]:

```
model_details=wml_clients.repository.store_model(model=DEMO_MODEL,meta_props=model_props,training_data=x_train,
training_target=y_train.values.ravel())
```

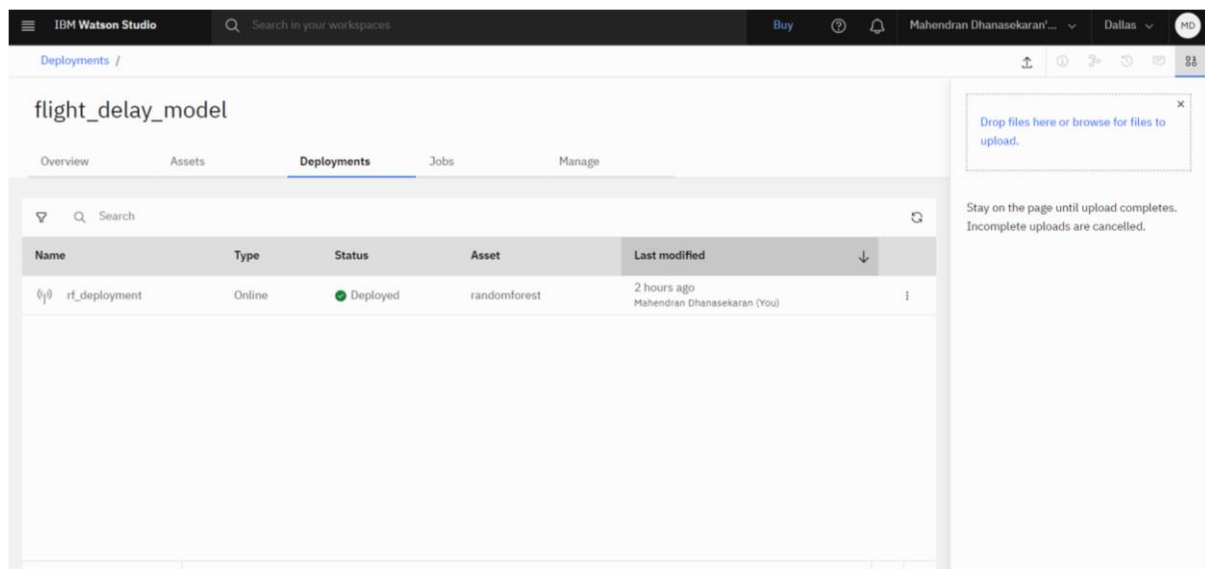
In [117]:

```
model_details
model_id=wml_clients.repository.get_model_id(model_details) dep_props={
wml_clients.deployments.ConfigurationMetaNames.NAME:DEPLOYMENT_NAME,
wml_clients.deployments.ConfigurationMetaNames.ONLINE:{}
```

In [125]: deployment=wml_clients.deployments.create(artifact_uid=model_id,meta_props=dep_props)

NOTE: APIKey must need to create to deploy and connect API

After successful of deployment, deployed is appeared in Deployment section as follow,



Testing of deployed model as follow, by giving values of all the features and it gives prediction.

IBM Watson Studio

Deployments / flight_delay_model / randomforest /

rf_deployment Deployed Online

API reference **Test**

Enter input data

Text input JSON input

Enter data manually or use a CSV file to populate the spreadsheet. Max file size is 50 MB.

[Download CSV template](#) [Browse local files](#) [Search in space](#) [Clear all](#)

	QUARTER (int64)	MONTH (int64)	DAY_OF_MONTH (int64)	DAY_OF_WEEK (int64)	FL_NUM (int64)	ORIGIN (int64)	DEST (int64)	CRS_DEP_TIME.1 (int64)	CRS_ARR_TIME.1 (int64)
1	Start typing or drag and drop a CSV file...								
2									
3									
4									
5									
6									

0 rows, 12 columns

Predict

After these, need to copy API requesting codes on required language (python).

Creating Dashboard using HTML/CSS:

Frontend Dashboard is created using HTML/CSS,

Result Web page like,

Flight Delay Prediction

File | C:/Users/ELCOT/Pictures/Screenshots/Flight/SPRINT-3/templates/index.html

FLIGHT DELAY PREDICTION

ENTER THE DETAILS

Enter name:

- ☐ Air Asia
- ☐ Air India
- ☐ IndiGo
- ☐ Blue Dart
- ☐ Star Air

Enter month:

Enter dayofmonth:

Enter dayofweek:

Enter origin:

Enter destination:

Enter dept:

Enter arrtime:

Enter actdept:

Predict

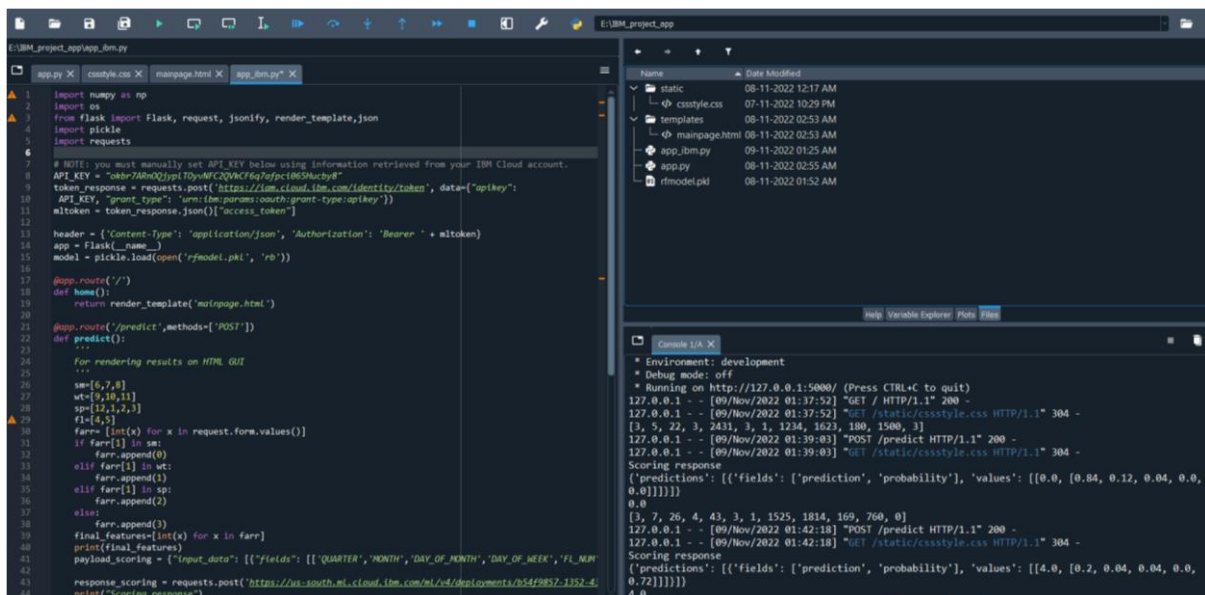
Create web app and Hosting in flask:

First thing, need to create directory as follow,

Name	Date Modified
static	08-11-2022 12:17 AM
└─ cssstyle.css	07-11-2022 10:29 PM
templates	08-11-2022 02:53 AM
└─ mainpage.html	08-11-2022 02:53 AM
app_ibm.py	09-11-2022 01:25 AM
app.py	08-11-2022 02:55 AM
rfmodel.pkl	08-11-2022 01:52 AM

Then, code the required logic in app.py file with API connection , request and response code.

Spyder IDE looks like,



```
1 import numpy as np
2 import os
3 from flask import Flask, request, jsonify, render_template, json
4 import pickle
5
6 # NOTE: you must manually set API KEY below using information retrieved from your IBM Cloud account.
7 API_KEY = "akbr7AbnQ2jyl7DyNFC2QVhCF6q7efpctkSShucy8"
8 token_response = requests.post("https://iam.cloud.ibm.com/identity/token", data={"apikey":
9 API_KEY, "grant_type": "urn:ibm:params:oauth:grant-type:apikey"})
10 mitoken = token_response.json()["access_token"]
11
12 header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mitoken}
13 app = Flask(__name__)
14 model = pickle.load(open('rfmodel.pkl', 'rb'))
15
16 @app.route('/')
17 def home():
18     return render_template('mainpage.html')
19
20 @app.route('/predict', methods=['POST'])
21 def predict():
22     """
23     For rendering results on HTML GUI
24     """
25     sm=[5,7,8]
26     wt=[9,10,11]
27     sp=[12,1,2,3]
28     fl=[4,5]
29     farr=[int(x) for x in request.form.values()]
30     if farr[1] in sm:
31         farr.append(0)
32     elif farr[1] in wt:
33         farr.append(1)
34     elif farr[1] in sp:
35         farr.append(2)
36     else:
37         farr.append(3)
38     final_features=[int(x) for x in farr]
39     print(final_features)
40     payload_scoring = {"input_data": [{"fields": [{"QUARTER", "MONTH", "DAY_OF_MONTH", "DAY_OF_WEEK", "FL_NUM"}]}]}
41     response_scoring = requests.post("https://us-south.ml.cloud.ibm.com/ml/v4/deployments/954f9857-3352-4f1e-b011-000000000000", data=json.dumps(payload_scoring), headers=header)
42     print("Scoring response")
```

Run the app.py file.

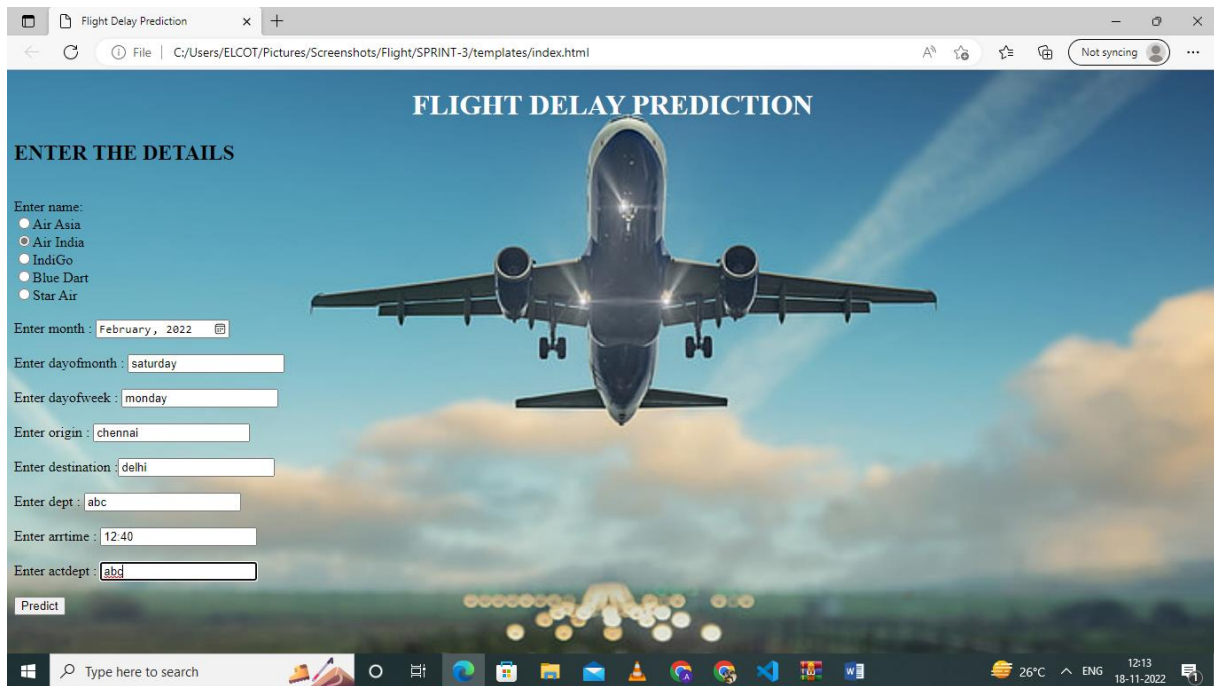
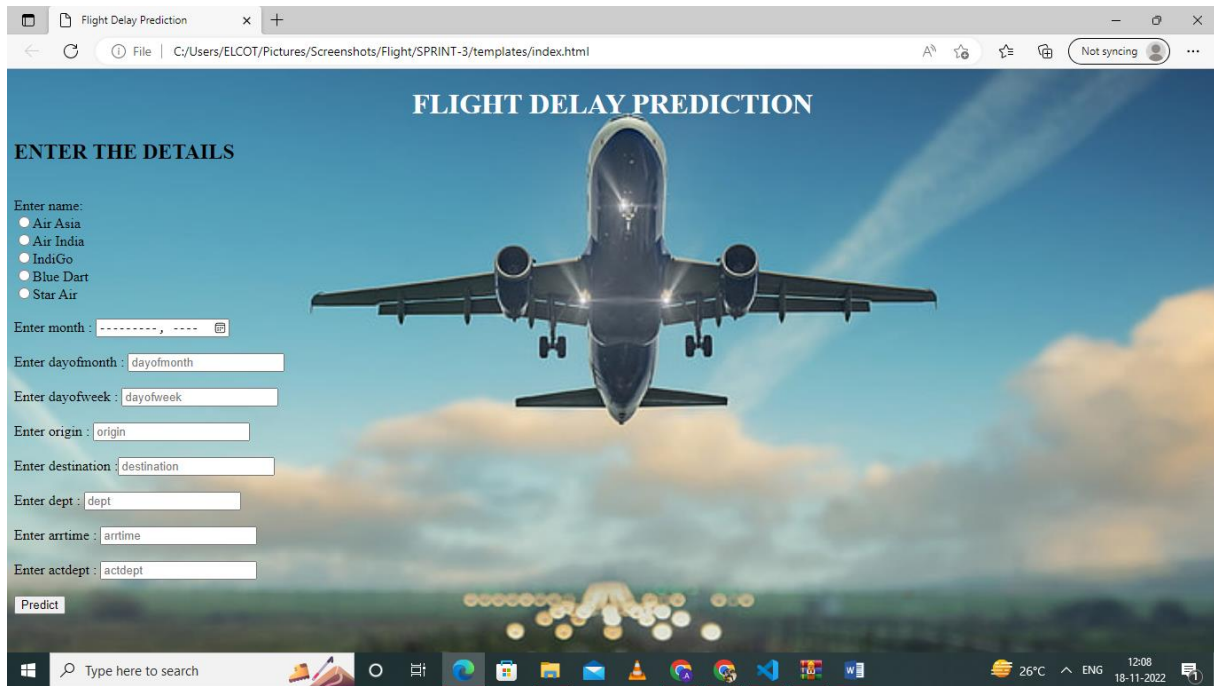
Localhost url is displayed in console, copy and paste in browser then search it , frond end HTML?CSS page is displayed. Successfully created and hosted web app in flask.

If any error caused as flask in production mode, then

Set FLASK_ENV=Development,

Then run the app

Testing web app:



Output is predicted by ML model successfully.