

## SPRINT – 2 PROJECT DOCUMENT

Date	13 November 2022
Team ID	PNT2022TMID15779
Project Name	Flight Delay Prediction Using Machine Learning

### DEVELOPMENT PHASE:

#### SPRINT-2:

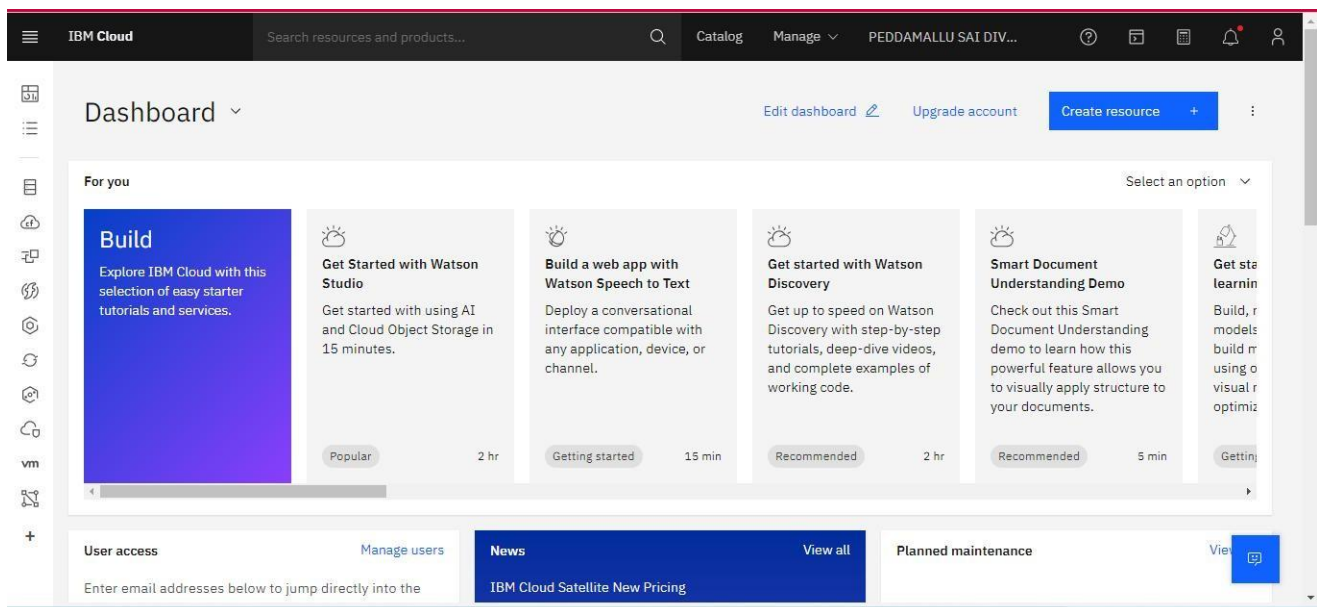
- Creating IBM cloud account & Required Resources
- Deploy our model in IBM Watson
- Creating Dashboard using HTML/CSS
- Create web app and Hosting in falk
- Testing web app

### 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,



The screenshot displays the IBM Cloud Dashboard interface. At the top, a dark navigation bar includes the IBM Cloud logo, a search bar, and links for Catalog, Manage, and the user profile (PEDDAMALLU SAI DIV...). Below this, the main dashboard area is titled 'Dashboard' and features a 'Create resource' button. The central section, labeled 'For you', contains a grid of recommended resources and tutorials, including 'Build', 'Get Started with Watson Studio', 'Build a web app with Watson Speech to Text', 'Get started with Watson Discovery', 'Smart Document Understanding Demo', and 'Get started with Watson Machine Learning'. Each card provides a brief description and estimated completion time. The bottom section includes 'User access' with a 'Manage users' link, a 'News' section featuring 'IBM Cloud Satellite New Pricing', and 'Planned maintenance'.

## Creating IBM Cloud Required Resources:

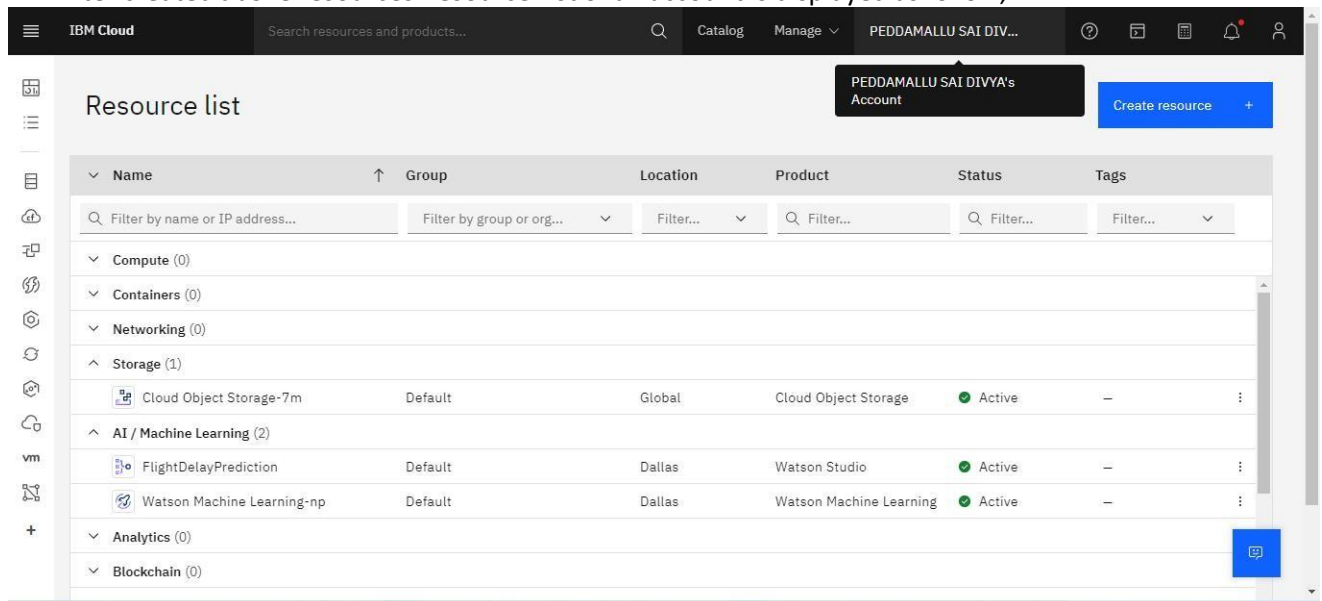
After creating IBM cloud account, to deploy ML model, need to create following resources such as,

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...					
Filter...					
Compute (0)					
Containers (0)					
Networking (0)					
Storage (1)					
Cloud Object Storage-7m	Default	Global	Cloud Object Storage	Active	-
AI / Machine Learning (2)					
FlightDelayPrediction	Default	Dallas	Watson Studio	Active	-
Watson Machine Learning-np	Default	Dallas	Watson Machine Learning	Active	-
Analytics (0)					
Blockchain (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":"okbr7ARnOQjypIT0yvNFC2QV6q7afpci065Hucby8",
    "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:{}
}

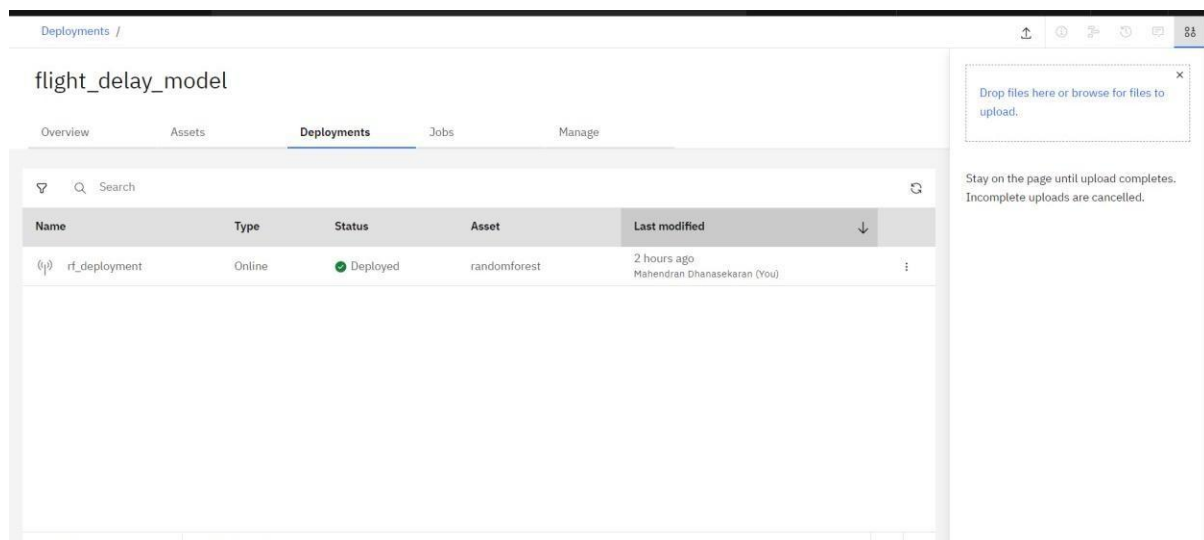

```

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.

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 as web page like,

app.component.html

File | D:/angularproject/ibmproject/src/app/app.component.html

### Prediction of flight delay

Enter flight Number

Month

Day of month

Day of week

Origin

Destination

Scheduled Departure Time

Scheduled Arrival Time

Actual Departure Time

23°C

ENG

21:48

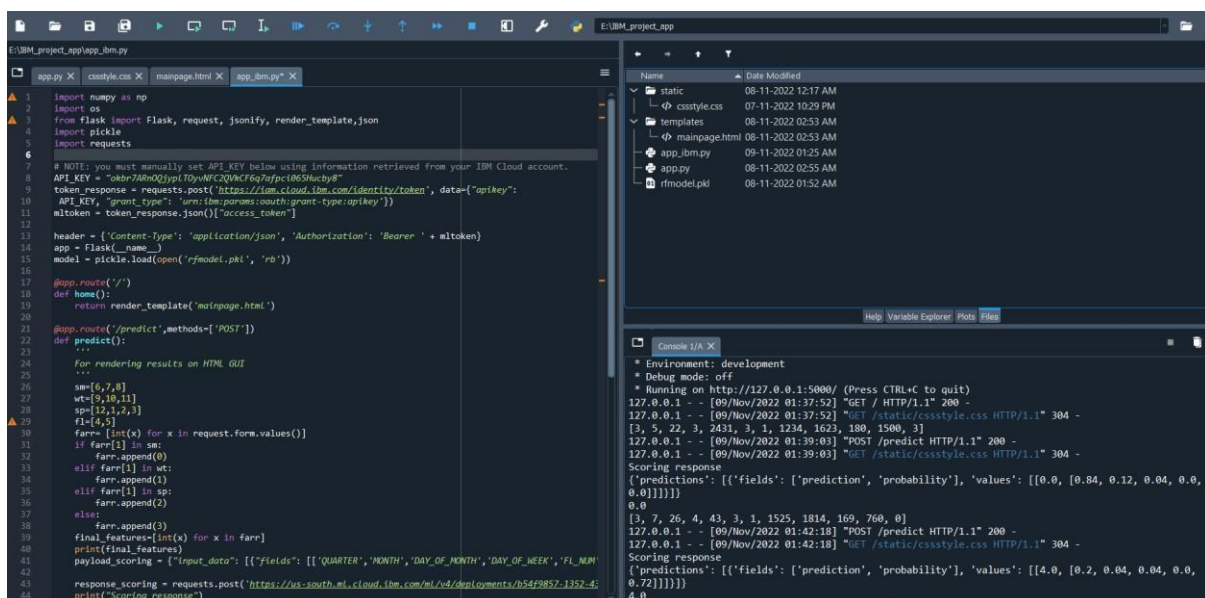
## Create web app and Hosting in flask:

First thing, need to create directory as follow,

Name	Date Modified
static	08-11-2022 12:17 AM
└─ csstyle.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 import requests
6
7 # NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account.
8 API_KEY = "akbr7ARnQ2jyn1TDyWfC2QVnCF6gJpfjK8Shuay8"
9 token_response = requests.post("https://iam.cloud.ibm.com/identity/token", data={"apikey":
10 API_KEY, "grant_type": "urn:ibm:params:oauth:grant-type:apikey"})
11 mltoken = token_response.json()["access_token"]
12
13 header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}
14 app = Flask(__name__)
15 model = pickle.load(open('rfmodel.pkl', 'rb'))
16
17 @app.route('/')
18 def home():
19     return render_template('mainpage.html')
20
21 @app.route('/predict', methods=['POST'])
22 def predict():
23     """
24     for rendering results on HTML GUI
25     """
26     sm=[6,7,8]
27     wt=[9,10,11]
28     sp=[12,1,2,3]
29     fl=[4,5]
30     farr=[int(x) for x in request.form.values()]
31     if farr[1] in sm:
32         farr.append(0)
33     elif farr[1] in wt:
34         farr.append(1)
35     elif farr[1] in sp:
36         farr.append(2)
37     else:
38         farr.append(3)
39     final_features=[int(x) for x in farr]
40     print(final_features)
41     payload_scoring = {"input_data": [{"fields": ["QUARTER", "MONTH", "DAY_OF_MONTH", "DAY_OF_WEEK", "FL_NUM"]
42     response_scoring = requests.post("https://us-south.ml.cloud.ibm.com/ml/v4/deployments/054f9857-1352-4c
43     print("Scoring response")
```

Environment: development  
\* Debug mode: off  
\* Running on https://127.0.0.1:5000/ (Press CTRL+C to quit)  
127.0.0.1 - - [09/Nov/2022 01:37:52] "GET / HTTP/1.1" 200 -  
127.0.0.1 - - [09/Nov/2022 01:37:52] "GET /static/cssstyle.css HTTP/1.1" 304 -  
[3, 5, 22, 3, 2431, 3, 1, 1234, 1623, 180, 1500, 3]  
127.0.0.1 - - [09/Nov/2022 01:39:03] "POST /predict HTTP/1.1" 200 -  
127.0.0.1 - - [09/Nov/2022 01:39:03] "GET /static/cssstyle.css HTTP/1.1" 304 -  
Scoring response  
{'predictions': [{'fields': ['prediction', 'probability'], 'values': [[0.0, [0.84, 0.12, 0.04, 0.0, 0.0]]]]]  
0.0  
[3, 7, 26, 4, 43, 3, 1, 1525, 1814, 169, 760, 0]  
127.0.0.1 - - [09/Nov/2022 01:42:18] "POST /predict HTTP/1.1" 200 -  
127.0.0.1 - - [09/Nov/2022 01:42:18] "GET /static/cssstyle.css HTTP/1.1" 304 -  
Scoring response  
{'predictions': [{'fields': ['prediction', 'probability'], 'values': [[4.0, [0.2, 0.04, 0.04, 0.0, 0.72]]]]]  
4.0

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:

Enter the data on the required fields,

The screenshot shows a web browser window with the URL `app.component.html`. The page title is "Prediction of flight delay". The form contains the following fields and values:

Field	Value
Enter flight Number	1234
Month	november
Day of month	17
Day of week	thursday
Origin	mso
Destination	mso
Scheduled Departure Time	21:48
Scheduled Arrival Time	02:55
Actual Departure Time	01:00

A green "Submit" button is located at the bottom right of the form. The background of the page is a dark, cloudy sky with a runway and two airplanes at the bottom.

The screenshot shows the same web browser window as the previous one, but with a success message displayed in a white box at the top center. The message reads: "This page says: Flight is delayed". A blue "OK" button is located to the right of the message. The form fields and values are the same as in the previous screenshot. The background of the page is the same dark, cloudy sky with a runway and two airplanes at the bottom.

Output is predicted by ML model successfully.