

## Sprint - 4 objectives:

- Integrate Flask with the deployed model
- Test the Working Model for output

## Screenshots:

### Connecting using API endpoint:

The screenshot shows the IBM Cloud API reference page for the `ml/v4/deployments` endpoint. The page is titled "API reference" and "Test". Under the "Direct link" section, the endpoint URL is displayed: `https://us-south.ml.cloud.ibm.com/ml/v4/deployments/7d6c3b49-ec70-4cfe-ab88-4f6dbe6a7997/predictions?version=2019-07-01`. To the right, there is a "Bearer <token>" field with a dropdown menu showing "IAM". Below the "Code snippets" section, there are tabs for different languages: cURL, Java, JavaScript, Python (selected), and Scala. The Python code snippet is as follows:

```
import requests

# NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account.
API_KEY = "<your API key>"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey":
API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}

# NOTE: manually define and pass the array(s) of values to be scored in the next line
payload_scoring = {"input_data": [{"fields": [array_of_input_fields], "values": [array_of_values_to_be_scored, another_array_of_val

response_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/7d6c3b49-ec70-4cfe-ab88-4f6dbe6a7997/predicti
headers={'Authorization': 'Bearer ' + mltoken})
```

Flask integration achieved successfully.

The screenshot shows a code editor with the following Python code:

```
1 import os
2 from pymongo import MongoClient
3 from flask import Flask, request, render_template
4 import requests
5 client = MongoClient('localhost', 27017)
6 db = client.login
7
8 login = db.users
9 # NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud a
10 API_KEY = "FDu8w9acEuLpZiojHLEoW5Rc2uHT9889GjnPT5QZ0-LN"
11 token_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey":
12 API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
13 mltoken = token_response.json()["access_token"]
14
15 header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}
16 app=Flask(__name__)
```

## Deployed model testing in ibm:

The screenshot displays the IBM Watson Studio interface for testing a deployed model. The top navigation bar shows the user is logged in as Manishankar Karthikeyan. The main content area is titled "rf\_deployment" and is in the "Test" tab. Below the title, there are tabs for "Text input" and "JSON input". The "Text input" tab is active, showing a table with 12 columns: QUARTER (int64), MONTH (int64), DAY\_OF\_MONTH (int64), DAY\_OF\_WEEK (int64), FL\_NUM (int64), ORIGIN (int64), DEST (int64), CRS\_DEP\_TIME (int64), and CRS\_ARR\_TIME (int64). The first row of data is visible, with values: 1, 2, 6, 25, 2, 2560, 5, 4, 1894, and 1600. A "Predict" button is located at the bottom right of the input area.

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 (int64)	CRS_ARR_TIME (int64)
1	2	6	25	2	2560	5	4	1894	1600
2									
3									
4									
5									
6									

1 row, 12 columns

Predict

Prediction results

Prediction type: Multiclass classification

Prediction percentage: 1 Record

Confidence level distribution: 1

Table view JSON view

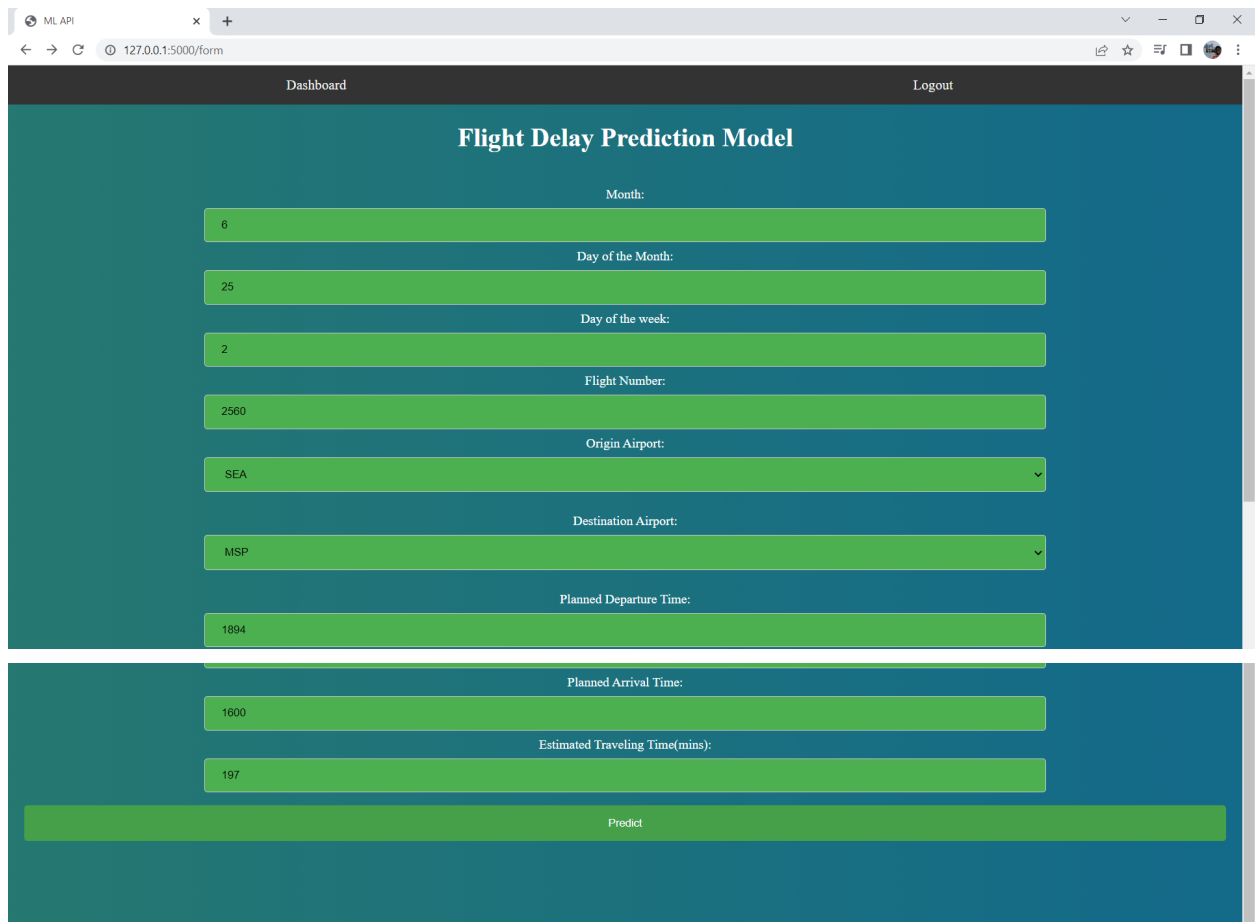
	Prediction	Confidence
1	2	38%
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		

Download

Output of 2 means it predicts delay in both arrival and departure.

Testing for the same input in the application:

Inputs:

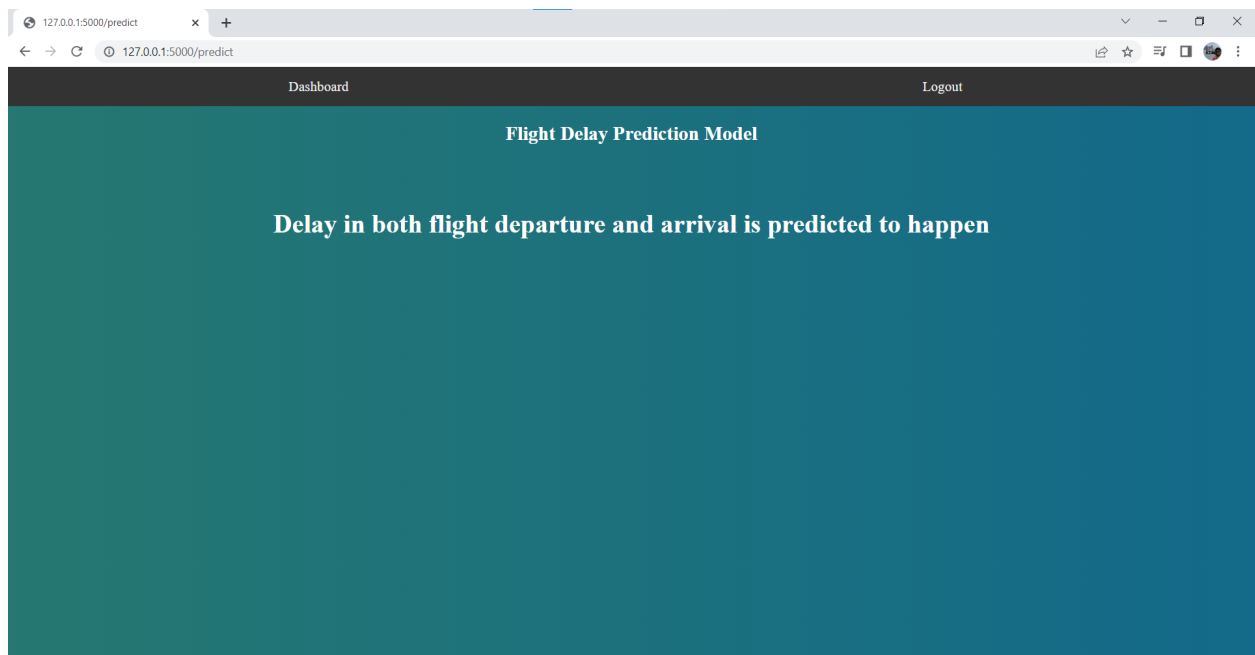


The screenshot shows a web browser window with the URL `127.0.0.1:5000/form`. The page has a dark header with "Dashboard" and "Logout" links. The main content area is titled "Flight Delay Prediction Model" and contains a form with the following inputs:

- Month: 6
- Day of the Month: 25
- Day of the week: 2
- Flight Number: 2560
- Origin Airport: SEA
- Destination Airport: MSP
- Planned Departure Time: 1894
- Planned Arrival Time: 1600
- Estimated Traveling Time(mins): 197

A green "Predict" button is located at the bottom of the form.

Result:



The screenshot shows a web browser window with the URL `127.0.0.1:5000/predict`. The page has a dark header with "Dashboard" and "Logout" links. The main content area is titled "Flight Delay Prediction Model" and displays the following result:

**Delay in both flight departure and arrival is predicted to happen**