

# Cloud and API Deployment

Name: SONIYA SUNNY

Batch Code: LISUM10: 30

Submission Date: July 4, 2022

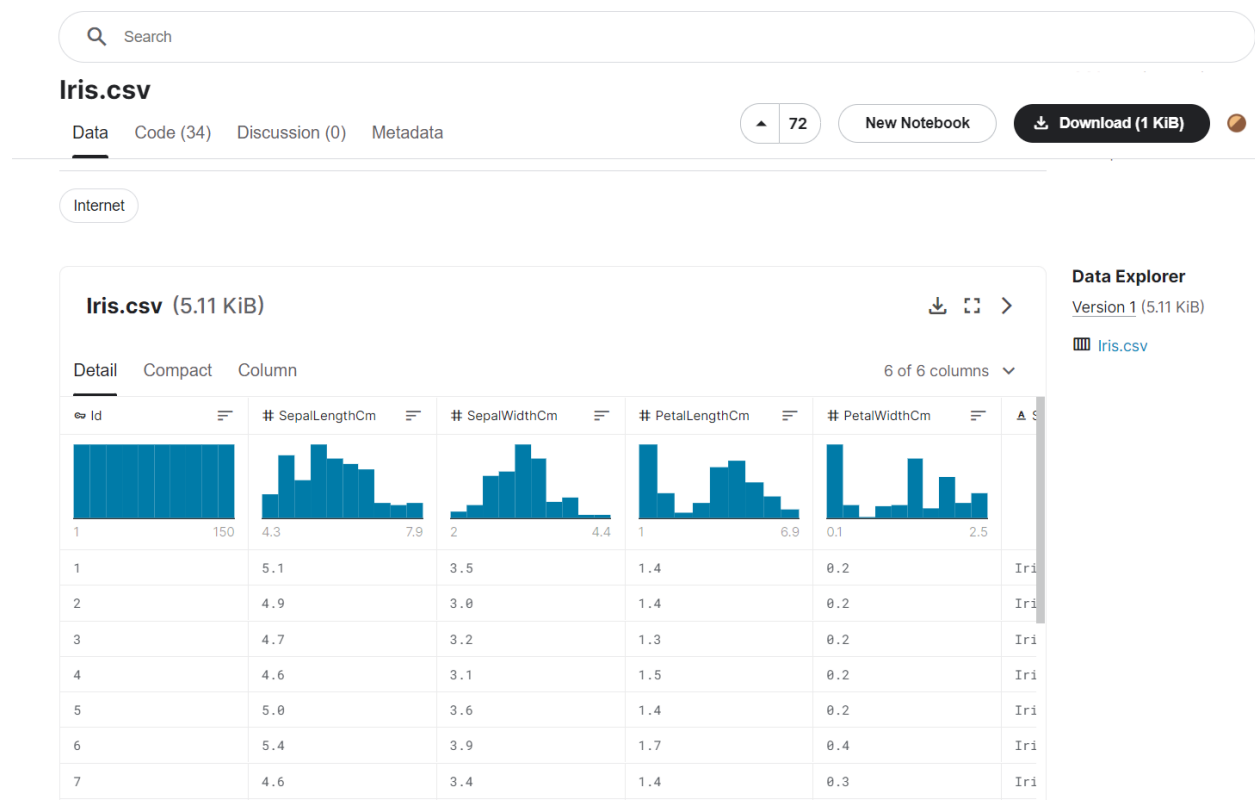
Submitted to: Data Glacier

## Steps:

### 1. ML Model Training: -

#### Data Collection

Selected the famous Iris Dataset, from Kaggle website.



## Model Building

```
# Importing necessary libraries
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
import pickle

# Reading the data
iris = pd.read_csv("Iris.csv")
print(iris.head())
iris.drop("Id", axis=1, inplace = True)
y = iris['Species']
iris.drop(columns='Species',inplace=True)
X = iris[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']]

# Training the model
x_train,x_test,y_train,y_test = train_test_split(X, y, test_size=0.3)
model = LogisticRegression()
model.fit(x_train,y_train)

#saving model to disk
pickle.dump(model, open('model.pkl','wb'))

#loading model to compare result
model = pickle.load(open('model.pkl','rb'))
print(model.predict([[5.1,3.5,1.4,0.2]]))

✓ 0.8s
```









	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

['Iris-setosa']

Built a Logistic Regression model and saved the model as a pickle file in the same directory.

## 2. Creating web app using Flask: -

Added more folders

Name	Date modified	Type	Size
 .git	2022-06-23 1:07 PM	File folder	
 static	2022-06-23 12:21 PM	File folder	
 templates	2022-06-23 12:21 PM	File folder	
 app.py	2022-06-23 1:06 PM	Python Source File	1 KB
 Iris.csv	2022-06-23 12:22 PM	Microsoft Excel Co...	5 KB
 iris_app.ipynb	2022-06-23 12:52 PM	Jupyter Source File	3 KB
 iris_model.ipynb	2022-06-23 12:24 PM	Jupyter Source File	3 KB
 model.pkl	2022-06-23 12:23 PM	PKL File	1 KB

Added static folder and templates folder.

Added index.html file in templates folder  
Added style.css file in static folder  
Created app.py file  
Created iris\_app.ipynb file for sample execution

```
# importing necessary libraries and functions
import numpy as np
from flask import Flask, request, render_template
import pickle

app = Flask(__name__) #Initialize the flask App
model = pickle.load(open('model.pkl', 'rb')) # loading the trained model

@app.route('/') # Homepage
def home():
    return render_template('index.html')

@app.route('/predict',methods=['POST'])
def predict():
    '''For rendering results on HTML GUI'''

    # retrieving values from form
    init_features = [float(x) for x in request.form.values()]
    final_features = [np.array(init_features)]
    prediction = model.predict(final_features) # making prediction

    return render_template('index.html', prediction_text='Predicted Class: {}'.format(prediction))

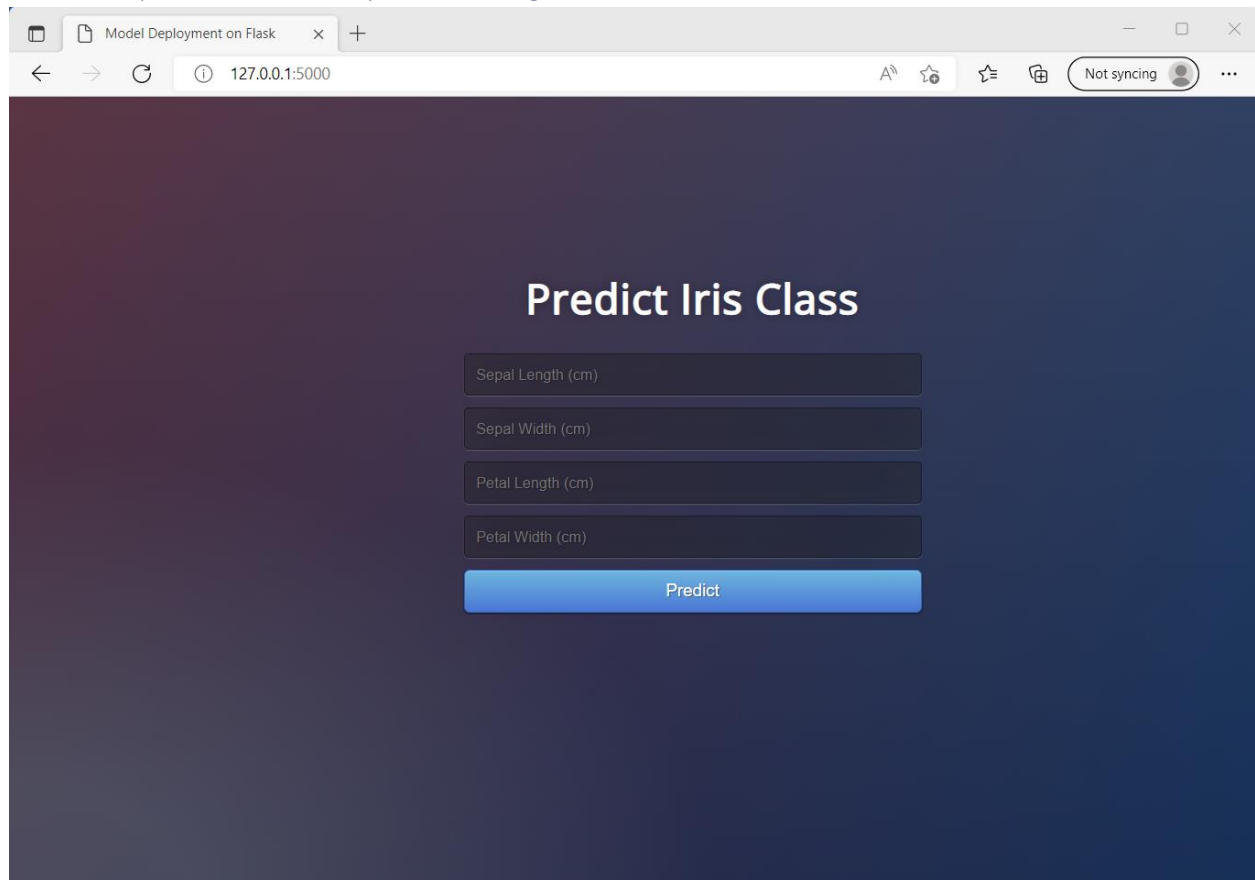
if __name__ == "__main__":
    app.run()
```

✓ 3.9s

```
* Serving Flask app "__main__" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off

* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Copied the url and opened through a browser



The screenshot shows a web browser window with a single tab titled "Model Deployment on Flask". The address bar displays the URL "127.0.0.1:5000". The browser's toolbar includes navigation buttons (back, forward, refresh), a status bar showing "Not syncing", and a user profile icon. The main content area has a dark blue gradient background and features the heading "Predict Iris Class" in white. Below the heading is a form with four input fields: "Sepal Length (cm)", "Sepal Width (cm)", "Petal Length (cm)", and "Petal Width (cm)". Each field is a dark grey rectangle with white text. At the bottom of the form is a blue "Predict" button with white text.

## Predict Iris Class

Sepal Length (cm)

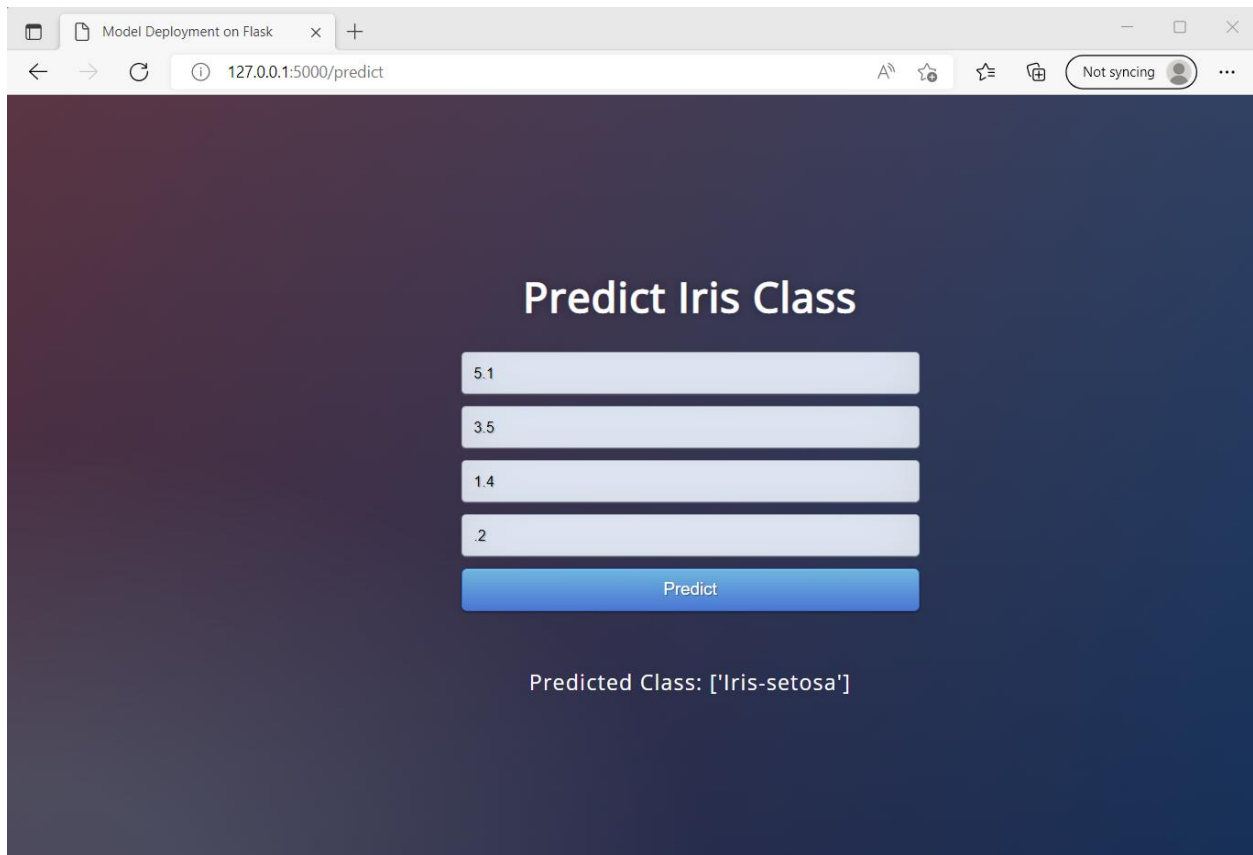
Sepal Width (cm)

Petal Length (cm)

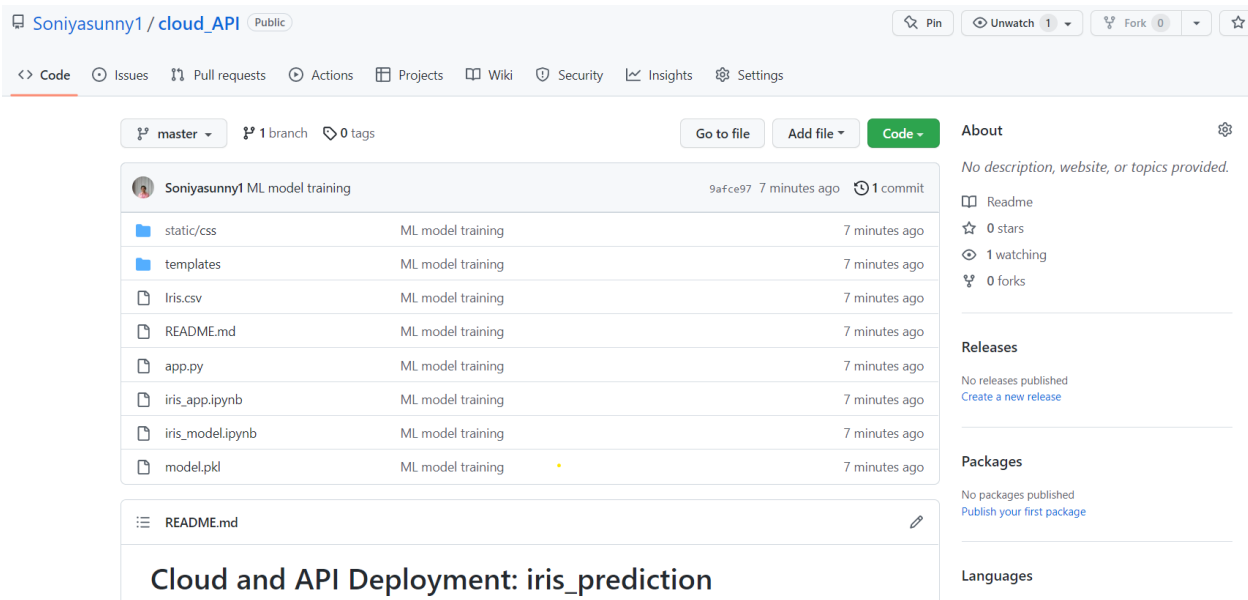
Petal Width (cm)

Predict


### Tested the model



### 3. Committing Code in online Repo: -




#### 4. Account creation in Heroku: -


 **HEROKU**

Already have an account? [Log in](#)


## Sign up for free and experience Heroku today

 **Free account**

Create apps, connect databases and add-on services, and collaborate on your apps, for free.

 **Your app platform**

A platform for apps, with app management & instant scaling, for development and production.

 **Deploy now**

Go from code to running app in minutes. Deploy,

First name \*

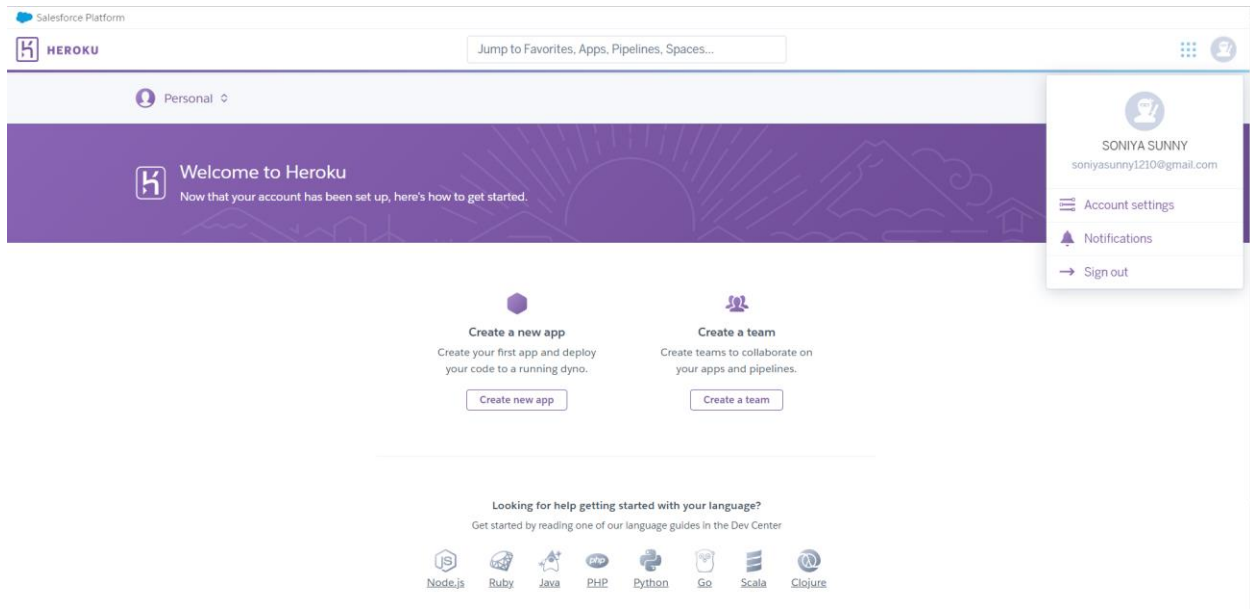
Last name \*

Email address \*

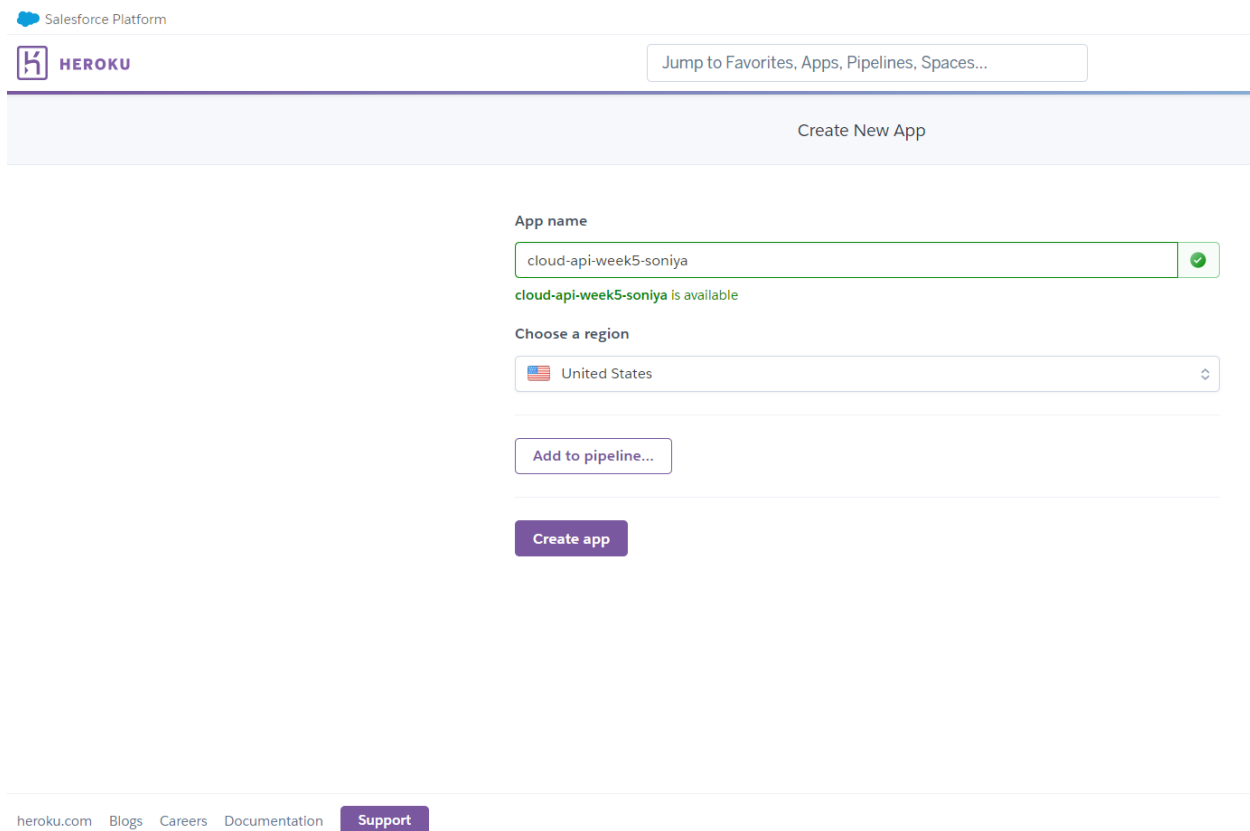
Company name

Role \*

Country/Region \*



## 5. Linking of online repo to Heroku: -



Heroku app created with name 'cloud-api-week5-soniya':



Salesforce Platform

HEROKU

Jump to Favorites, Apps, Pipelines, Spaces...

Personal

<

>

cloud-api-week5-soniya

☆

Open app

More

Overview

Resources

Deploy

Metrics

Activity

Access

Settings

Add this app to a pipeline

Create a new pipeline or choose an existing one and add this app to a stage in it.

Add this app to a stage in a pipeline to enable additional features

Pipelines let you connect multiple apps together and **promote code** between them. [Learn more.](#)

Pipelines connected to GitHub can enable **review apps**, and create apps for new pull requests. [Learn more.](#)

Choose a pipeline

Deployment method

Heroku Git

Use Heroku CLI

GitHub

Connect to GitHub

Container Registry

Use Heroku CLI

Deploy using Heroku Git

Use git in the command line or a GUI tool to deploy this app.

Install the Heroku CLI

Download and install the [Heroku CLI](#).

If you haven't already, log in to your Heroku account and follow the prompts to create a new SSH public key.

```
$ heroku login
```

## Connected to github:

Deployment method

Heroku Git

Use Heroku CLI

GitHub

Connected

Container Registry

Use Heroku CLI

App connected to GitHub

Code diffs, manual and auto deploys are available for this app.

Connected to [Soniyasunny1/cloud\\_API](#) by [Soniyasunny1](#)

Disconnect...

Releases in the [activity feed](#) link to GitHub to view commit diffs

Automatic deploys

Enables a chosen branch to be automatically deployed to this app.

You can now change your main deploy branch from "master" to "main" for both manual and automatic deploys, please follow the instructions [here](#).

Enable automatic deploys from GitHub

Every push to the branch you specify here will deploy a new version of this app. **Deploys happen automatically:** be sure that this branch is always in a deployable state and any tests have passed before you push. [Learn more.](#)

Choose a branch to deploy

master

☐ Wait for CI to pass before deploy

Soniyasunny1 / cloud\_API Public

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

## Deployments / History

Show: cloud-api-week5-soniya

cloud-api-week5-soniya at 9943916  
Deployed by Soniyasunny1 11 hours ago (Active) [View deployment](#)

© 2022 GitHub, Inc. Terms Privacy Security Status Docs Contact GitHub Pricing API Training Blog About

## 6. Deployment of ML model: -

Salesforce Platform

HEROKU

Jump to Favorites, Apps, Pipelines, Spaces...

### Manual deploy

Deploy the current state of a branch to this app.

### Deploy a GitHub branch

This will deploy the current state of the branch you specify below. [Learn more](#)

Choose a branch to deploy

master [Deploy Branch](#)

Receive code from GitHub

Build master 9943916a

```
----- No change in requirements detected, installing from cache
----- Using cached install of python-3.10.5
----- Installing pip 22.1.2, setuptools 68.10.0 and wheel 0.37.1
----- Installing SQLite3
----- Installing requirements with pip
----- Discovering process types
----- Procfile declares types -> (none)
----- Compressing...
```

☒ Autoscroll with output [View build log](#)

Release phase

Deploy to Heroku



heroku.com Blogs Careers Documentation **Support** Terms of Service Privacy Cookies © 2022

Deployed branch manually and App successfully deployed:

### Deploy a GitHub branch

This will deploy the current state of the branch you specify below. [Learn more](#).

Choose a branch to deploy

 master 

Deploy Branch

Receive code from GitHub



Build **master** 9943916a



Release phase



Deploy to Heroku











Your app was successfully deployed.

 View

## 7. Testing the web app: -

Got the output from the unique url provided by Heroku.

    https://cloud-api-week5-soniya.herokuapp.com/predict    

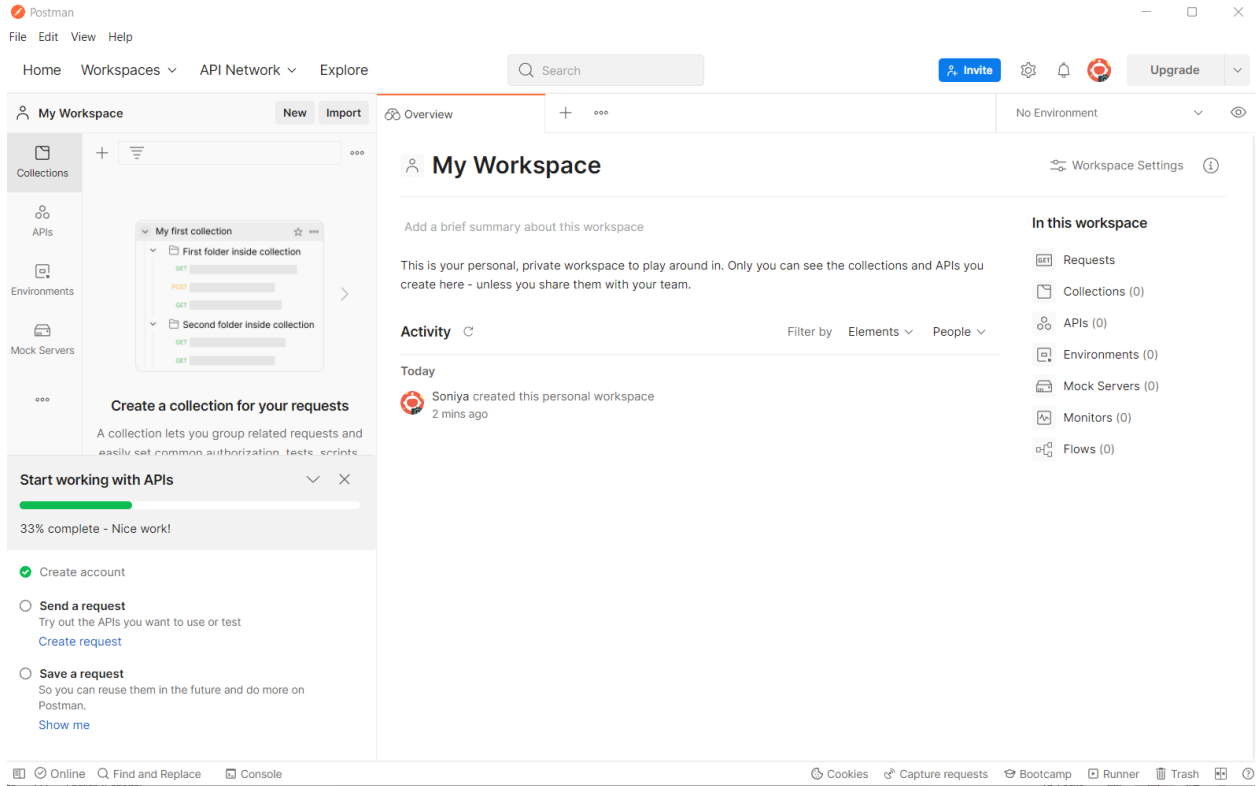
### Predict Iris Class

Predict

Predicted Class: ['Iris-setosa']

## API Deployment:

Downloaded Postman app and created workspace.



Dummy model created

```

# Importing necessary libraries
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
import pickle

# Reading the data
iris = pd.read_csv("Iris.csv")
print(iris.head())
iris.drop("Id", axis=1, inplace = True)
y = iris['Species']
iris.drop(columns='Species',inplace=True)
X = iris[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']]

# Training the model
x_train,x_test,y_train,y_test = train_test_split(X, y, test_size=0.3)
model = LogisticRegression()
model.fit(x_train,y_train)

#saving model to disk
pickle.dump(model, open('model.pkl','wb'))

#loading model to compare result
model = pickle.load(open('model.pkl','rb'))
print(model.predict([[5.1,3.5,1.4,0.2]]))

```

[4]

Created the app to predict the species based on several features

```

# importing necessary libraries and functions
import numpy as np
import pandas as pd
from flask import Flask, request, render_template, jsonify
import pickle

app = Flask(__name__) #Initialize the flask App

@app.route('/', methods= ['GET', 'POST']) # Homepage
def home():
    if(request.method == 'GET'):
        data = "hello world"
        return jsonify({'data': data})

@app.route('/predict')
def predict():
    model = pickle.load(open('model.pkl', 'rb'))
    SepalLengthCm = request.args.get('SepalLengthCm')
    SepalWidthCm = request.args.get('SepalWidthCm')
    PetalLengthCm = request.args.get('PetalLengthCm')
    PetalWidthCm = request.args.get('PetalWidthCm')
    test_df = pd.DataFrame({'SepalLengthCm':[SepalLengthCm], 'SepalWidthCm':[SepalWidthCm],
                             'PetalLengthCm':[PetalLengthCm], 'PetalWidthCm':[PetalWidthCm]})
    prediction = model.predict(test_df)
    return jsonify({'Species': str(prediction)})

# driver function
if __name__ == "__main__":
    app.run()

```

30.7s

Copy the URL

```
* Serving Flask app "__main__" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off

* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [04/Jul/2022 20:34:17] "GET / HTTP/1.1" 200 -
```

Paste the URL in the postman

The screenshot shows the Postman application interface. On the left, there's a sidebar with 'My Workspace' and a 'Collections' panel. The main area displays a GET request to 'http://127.0.0.1:5000/'. The 'Params' tab is active, showing a table with columns 'KEY', 'VALUE', and 'DESCRIPTION'. The 'Body' tab is also visible, showing a JSON response: `{\"data\": \"hello world\"}`. The status bar at the bottom indicates 'Status: 200 OK', 'Time: 14 ms', and 'Size: 168 B'.

KEY	VALUE	DESCRIPTION
Key	Value	Description

KEY	VALUE	DESCRIPTION
data	hello world	

Got the result: key as 'data' and value as "hello world"

Now, checking "/predict/" by giving input keys as Params and values:

http://127.0.0.1:5000/predict/?SepalLengthCm=5.1&SepalWidthCm=2.0&PetalLengthCm=6.1&PetalWidthCm=4.2

Save

GET

http://127.0.0.1:5000/predict/?SepalLengthCm=5.1&SepalWidthCm=2.0&PetalLengthCm=6.1&PetalWidthCm=4.2

Send

Params

Authorization

Headers (6)

Body

Pre-request Script

Tests

Settings

Cookies

Query Params

	KEY	VALUE	DESCRIPTION		Bulk Edit
<input checked="" type="checkbox"/>	SepalLengthCm	5.1			
<input checked="" type="checkbox"/>	SepalWidthCm	2.0			
<input checked="" type="checkbox"/>	PetalLengthCm	6.1			
<input checked="" type="checkbox"/>	PetalWidthCm	4.2			
	Key	Value	Description		

Got the output species:

Body

Cookies

Headers (4)

Test Results

Status: 200 OK

Time: 17 ms

Size: 178 B

Save Response

Pretty

Raw

Preview

Visualize

JSON

```
1  {
2    "Species": ["Iris-virginica"]
3  }
```