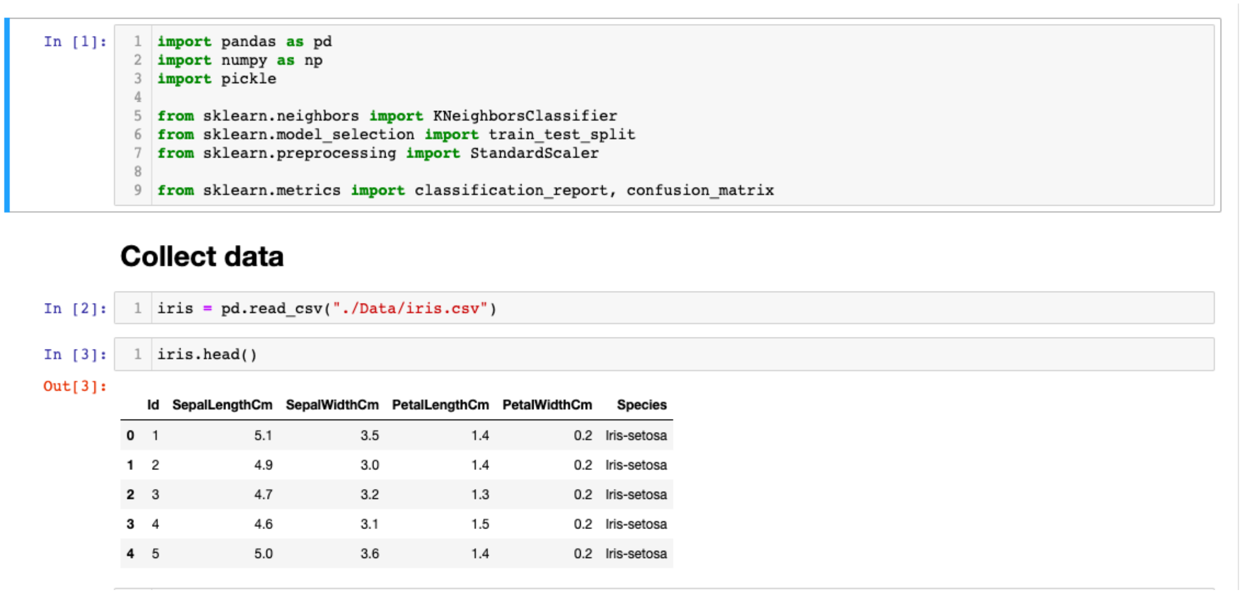
Name: Kierra Dangerfield
Batch Code: LISUM23: 30 June - 30 Sep. 2023
Submission Date: August 6, 2023
Submitted to:

Deploying ML Model with Flask

1. Coding Model

iris-modeling.ipynb

I created my model in a jupyter notebook. I start off by collecting my data that I got from Kaggle. I used the Iris dataset.



```
In [1]:
import pandas as pd
import numpy as np
import pickle

from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

from sklearn.metrics import classification_report, confusion_matrix
```

Collect data

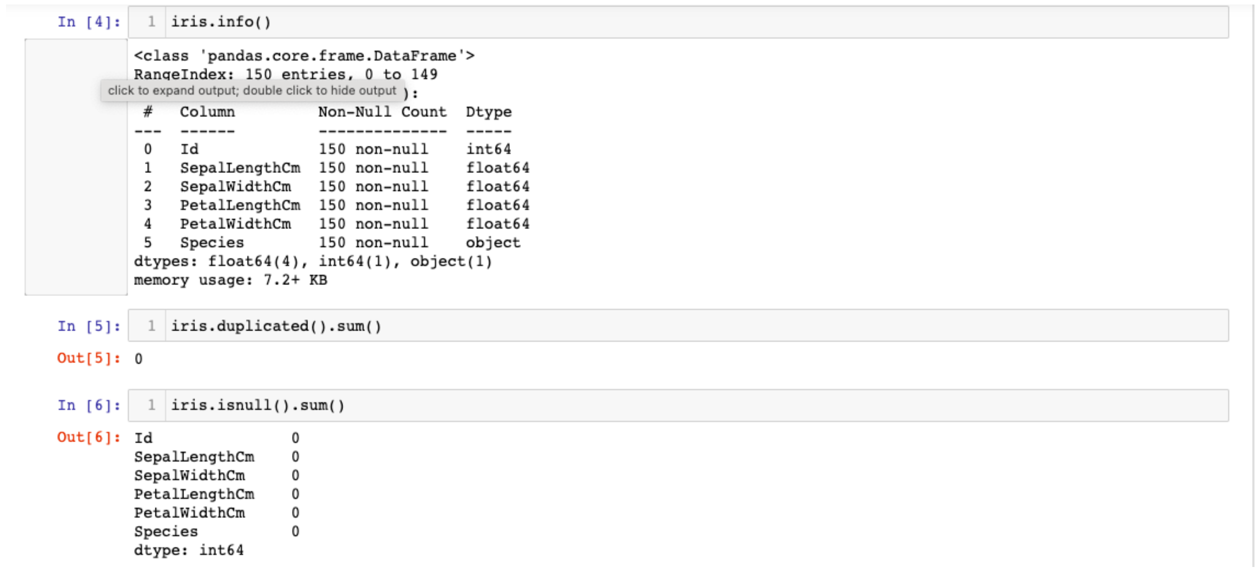
```
In [2]:
iris = pd.read_csv("./Data/iris.csv")
```

```
In [3]:
iris.head()
```

Out[3]:

	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

I then did a quick check for null and duplicate values



```
In [4]: iris.info()
```

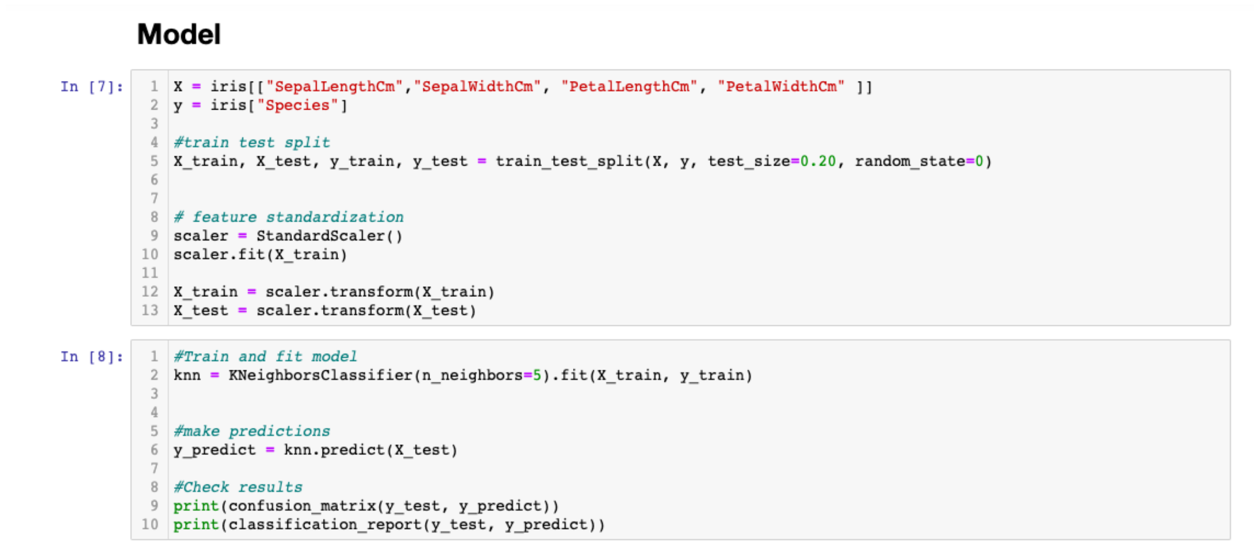
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
click to expand output; double click to hide output ):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             150 non-null    int64
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    object
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

```
In [5]: iris.duplicated().sum()
Out[5]: 0
```

```
In [6]: iris.isnull().sum()
Out[6]: Id               0
        SepalLengthCm    0
        SepalWidthCm     0
        PetalLengthCm    0
        PetalWidthCm     0
        Species          0
        dtype: int64
```

Model

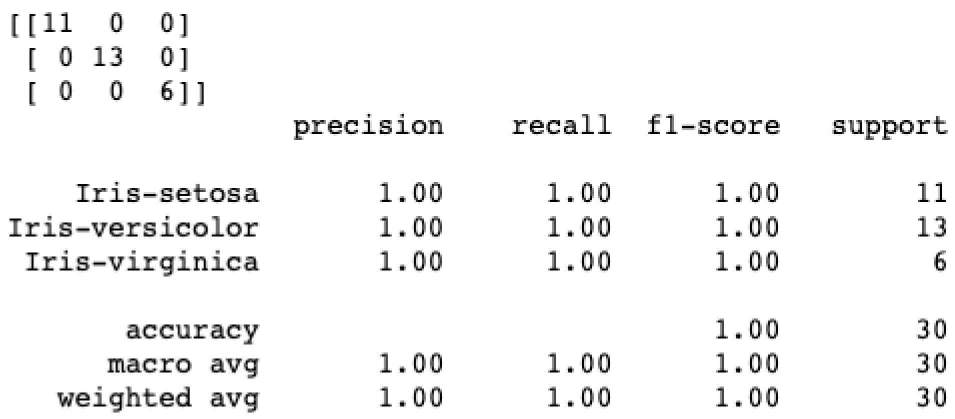
1. Store the input features that I want in X and the output feature (Species) in y.
2. train/test split
3. Feature Standardization with X_train and X_test
4. Tain and fit model using a KNN classifier
5. Make predictions on X_test
6. Check results



Model

```
In [7]:
X = iris[["SepalLengthCm","SepalWidthCm", "PetalLengthCm", "PetalWidthCm" ]]
y = iris["Species"]

#train test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=0)


# feature standardization
scaler = StandardScaler()
scaler.fit(X_train)

X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

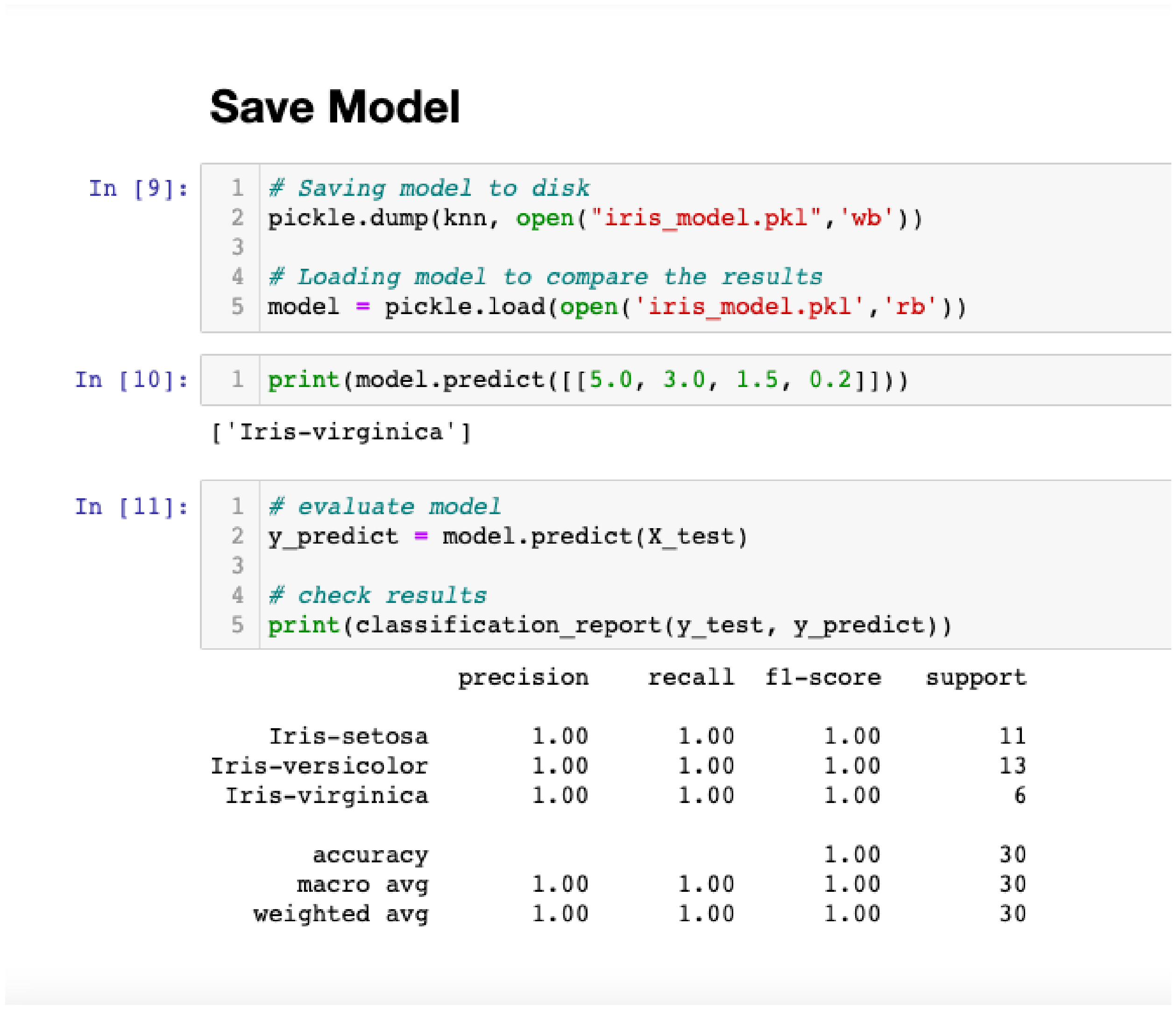
```
In [8]:
#Train and fit model
knn = KNeighborsClassifier(n_neighbors=5).fit(X_train, y_train)


#make predictions
y_predict = knn.predict(X_test)

#Check results
print(confusion_matrix(y_test, y_predict))
print(classification_report(y_test, y_predict))
```



```
[[11  0  0]
 [ 0 13  0]
 [ 0  0  6]]
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        11
Iris-versicolor       1.00      1.00      1.00        13
 Iris-virginica       1.00      1.00      1.00         6

       accuracy                           1.00        30
      macro avg       1.00      1.00      1.00        30
   weighted avg       1.00      1.00      1.00        30
```

Save Model

1. Save model to disk using pickle called “iris_model.pkl”
2. Load model to compare the results
3. Evaluate model



Save Model

In [9]:
```
# Saving model to disk
pickle.dump(knn, open("iris_model.pkl",'wb'))

# Loading model to compare the results
model = pickle.load(open('iris_model.pkl','rb'))
```

In [10]:
```
print(model.predict([[5.0, 3.0, 1.5, 0.2]]))
```

```
['Iris-virginica']
```

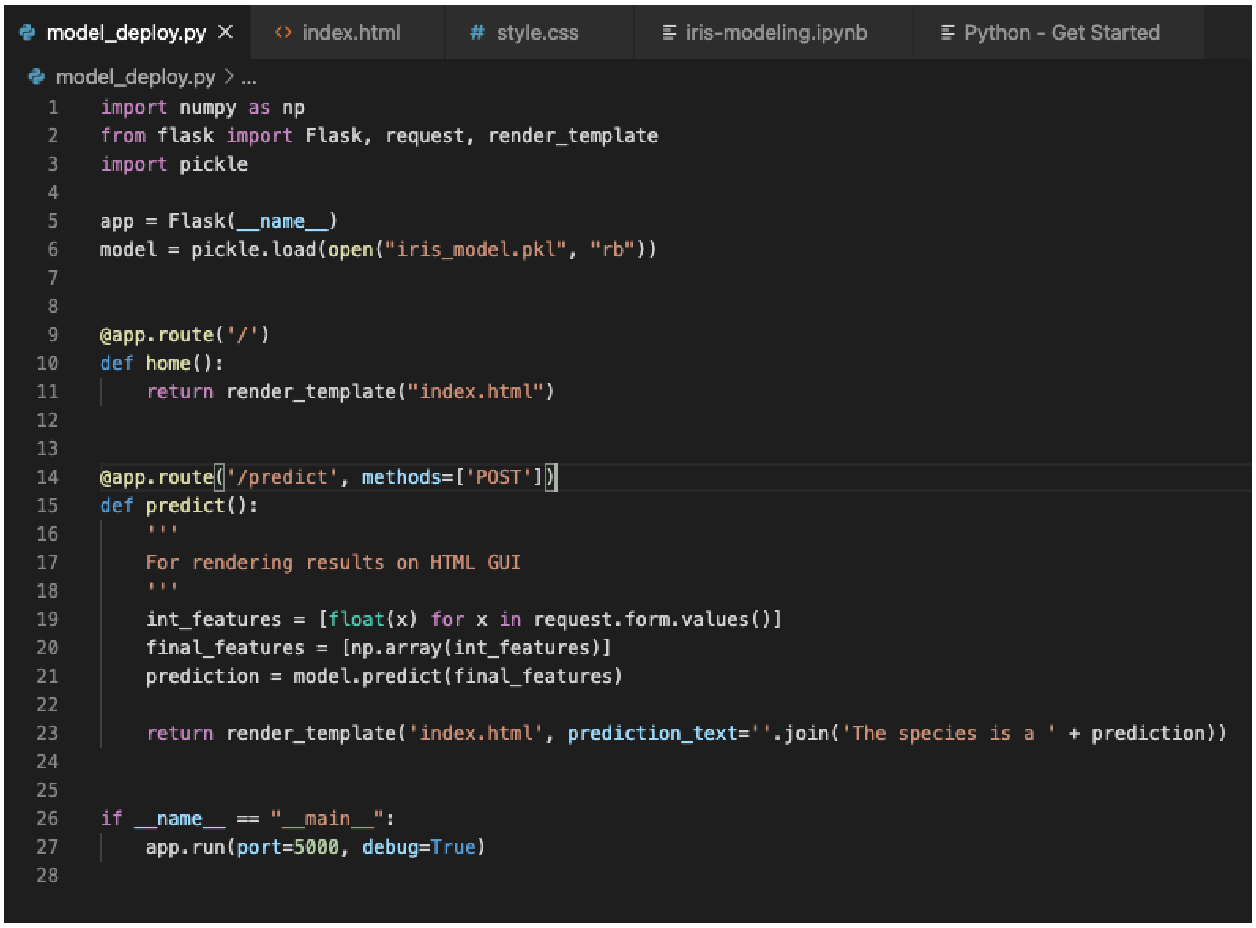
In [11]:
```
# evaluate model
y_predict = model.predict(X_test)

# check results
print(classification_report(y_test, y_predict))
```

```
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        11
Iris-versicolor       1.00      1.00      1.00        13
 Iris-virginica       1.00      1.00      1.00         6

       accuracy                           1.00        30
      macro avg       1.00      1.00      1.00        30
   weighted avg       1.00      1.00      1.00        30
```

2. Create flask application

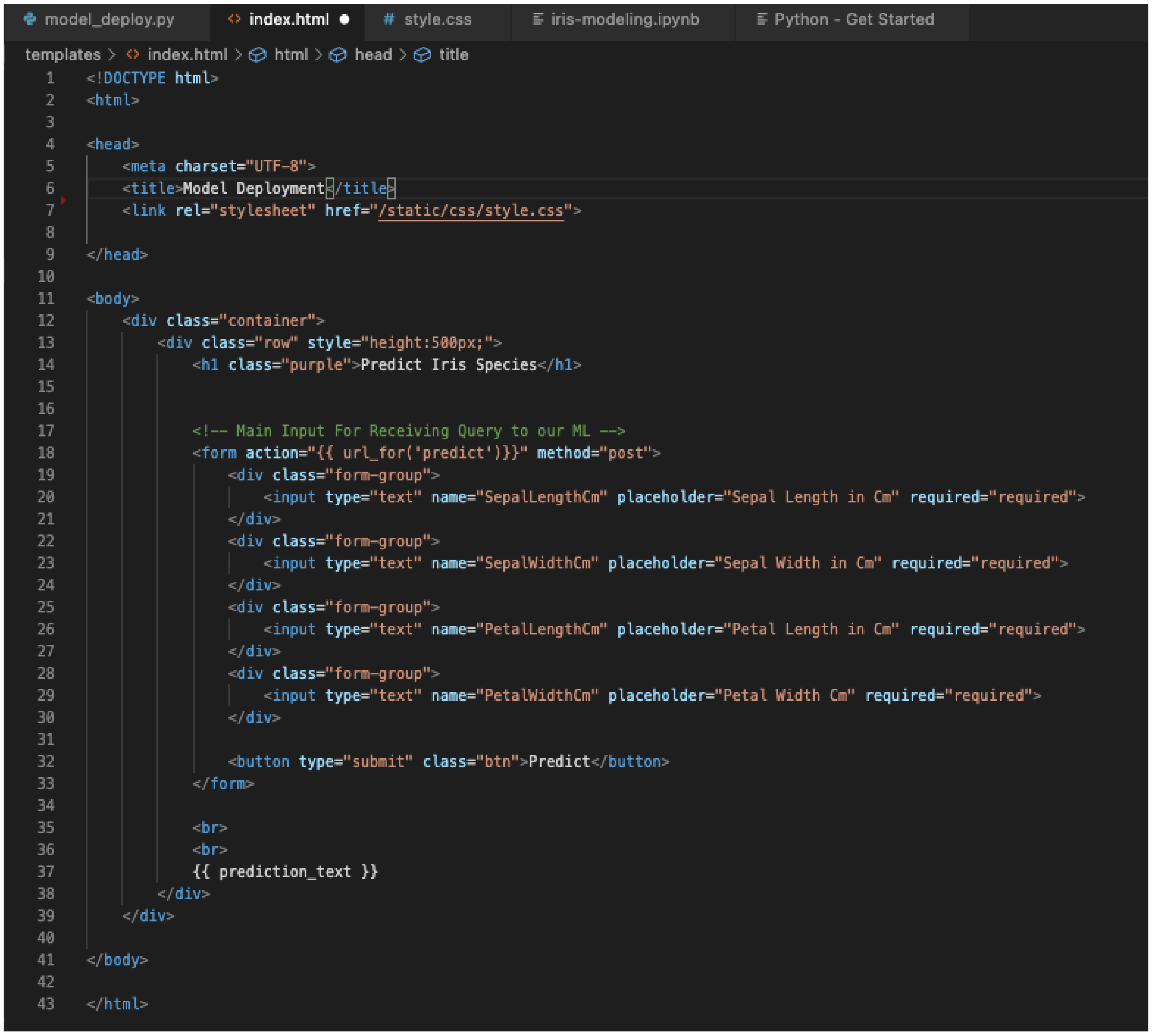
model_deploy.py

I created the Flask application in visual studio code.



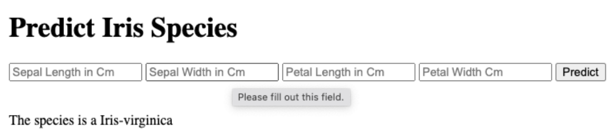
```python
import numpy as np
from flask import Flask, request, render_template
import pickle

app = Flask(__name__)
model = pickle.load(open("iris_model.pkl", "rb"))


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


@app.route('/predict', methods=['POST'])
def predict():
    '''
    For rendering results on HTML GUI
    '''
    int_features = [float(x) for x in request.form.values()]
    final_features = [np.array(int_features)]
    prediction = model.predict(final_features)

    return render_template('index.html', prediction_text=''.join('The species is a ' + prediction))


if __name__ == "__main__":
    app.run(port=5000, debug=True)

```

I then created the **index.html** file for output.



```html
<!DOCTYPE html>
<html>

<head>
    <meta charset="UTF-8">
    <title>Model Deployment</title>
    <link rel="stylesheet" href="/static/css/style.css">

</head>

<body>
    <div class="container">
        <div class="row" style="height:500px;">
            <h1 class="purple">Predict Iris Species</h1>


            <!-- Main Input For Receiving Query to our ML -->
            <form action="{{ url_for('predict')}}" method="post">
                <div class="form-group">
                    <input type="text" name="SepalLengthCm" placeholder="Sepal Length in Cm" required="required">
                </div>
                <div class="form-group">
                    <input type="text" name="SepalWidthCm" placeholder="Sepal Width in Cm" required="required">
                </div>
                <div class="form-group">
                    <input type="text" name="PetalLengthCm" placeholder="Petal Length in Cm" required="required">
                </div>
                <div class="form-group">
                    <input type="text" name="PetalWidthCm" placeholder="Petal Width Cm" required="required">
                </div>

                <button type="submit" class="btn">Predict</button>
            </form>

            <br>
            <br>
            {{ prediction_text }}
        </div>
    </div>

</body>

</html>
```

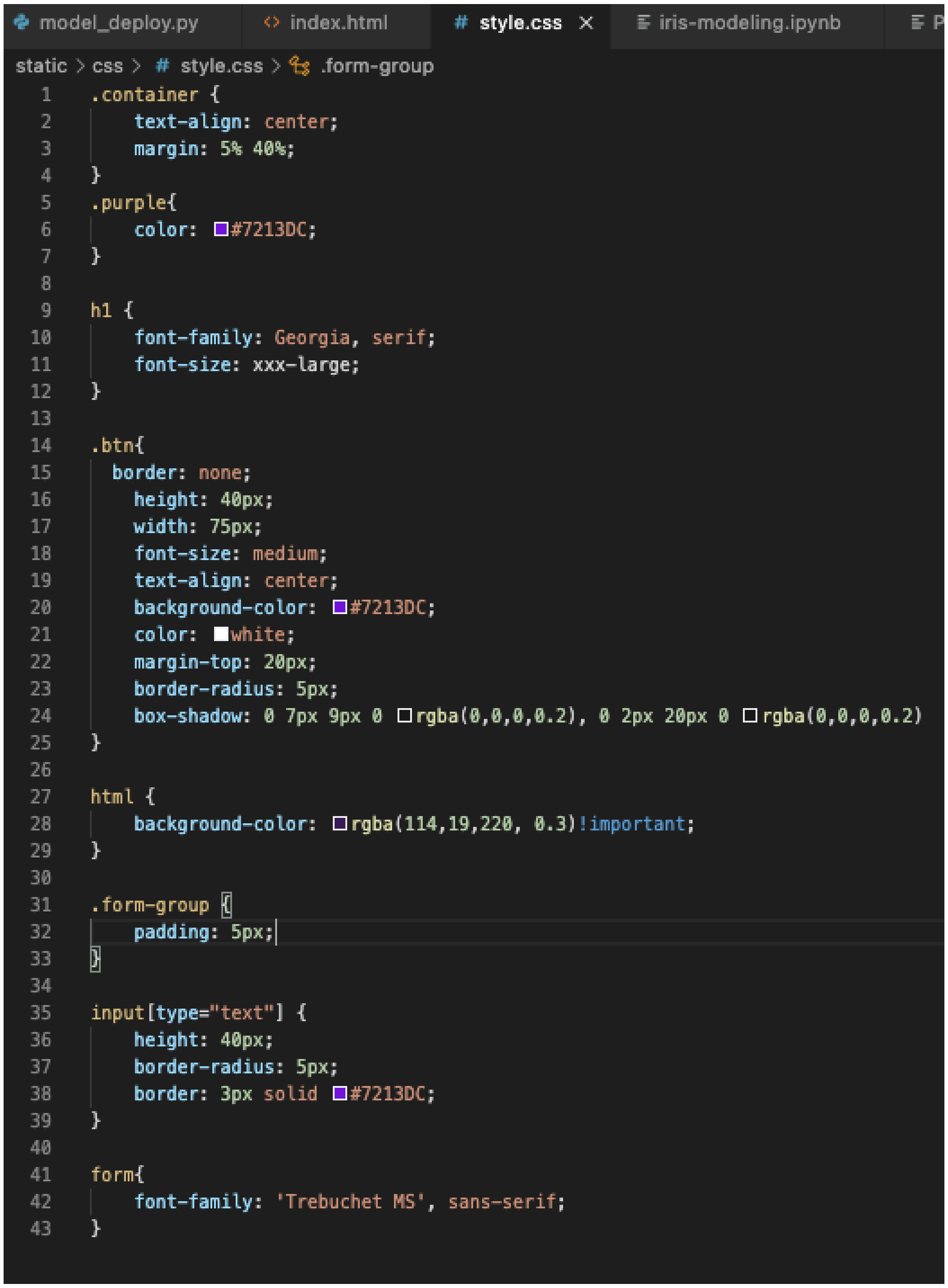
This is the output without CSS styling. I used the command **python model_deploy.py** to view the output.


Predict Iris Species

The species is a Iris-virginica

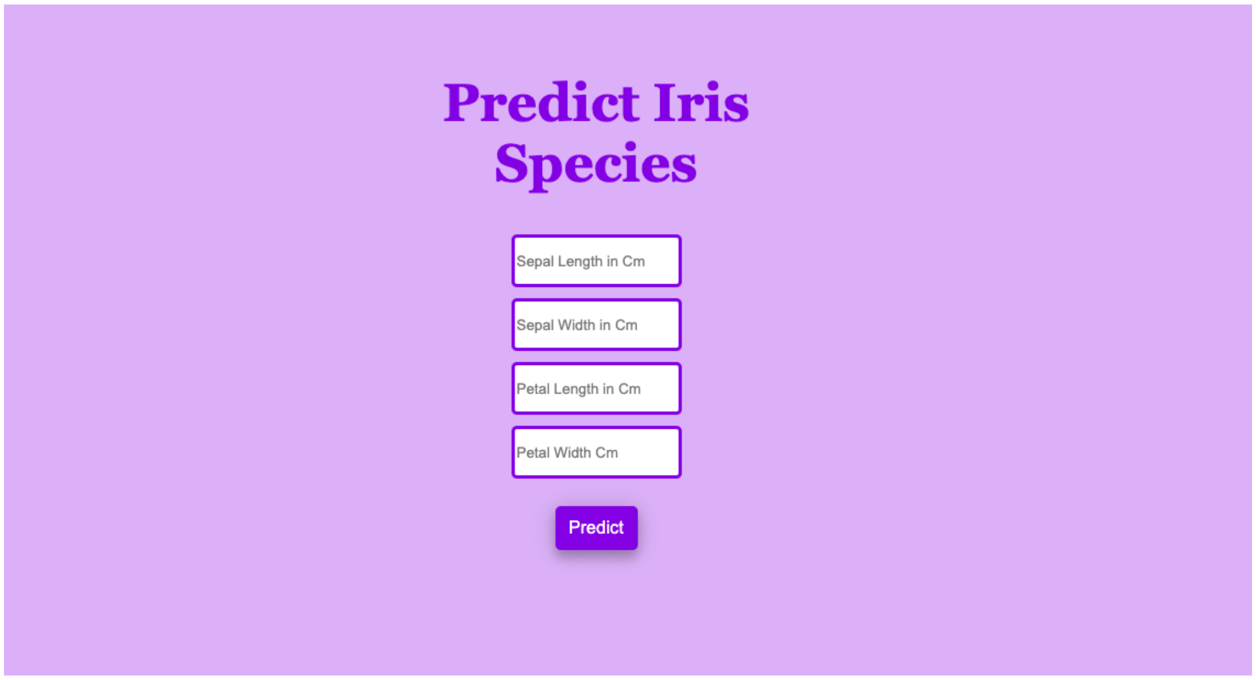
3. CSS Styling

Added some CSS for styling.



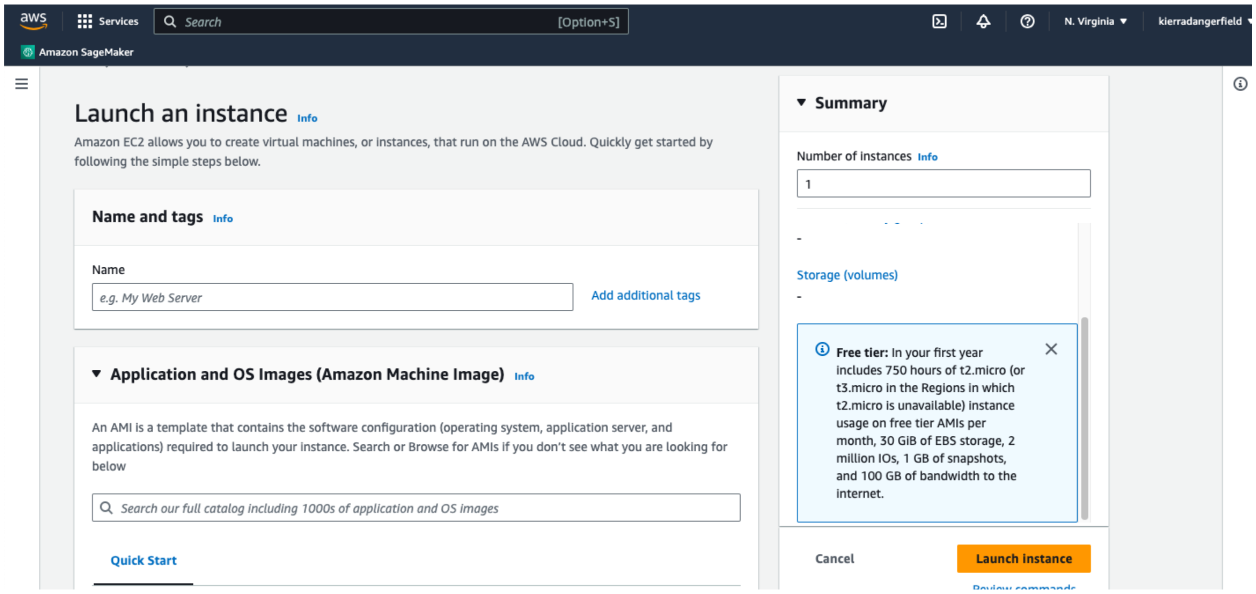
```css
.container {
    text-align: center;
    margin: 5% 40%;
}
.purple{
    color: #7213DC;
}

h1 {
    font-family: Georgia, serif;
    font-size: xxx-large;
}

.btn{
  border: none;
    height: 40px;
    width: 75px;
    font-size: medium;
    text-align: center;
    background-color: #7213DC;
    color: white;
    margin-top: 20px;
    border-radius: 5px;
    box-shadow: 0 7px 9px 0 rgba(0,0,0,0.2), 0 2px 20px 0 rgba(0,0,0,0.2)
}

html {
    background-color: rgba(114,19,220, 0.3)!important;
}

.form-group {
    padding: 5px;
}

input[type="text"] {
    height: 40px;
    border-radius: 5px;
    border: 3px solid #7213DC;
}

form{
    font-family: 'Trebuchet MS', sans-serif;
}
```

Final output


Predict Iris Species
Sepal Length in Cm
Sepal Width in Cm
Petal Length in Cm
Petal Width Cm
Predict

4. Cloud

I created an AWS account. I created an instance.



Launch an instance Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags Info

Name: e.g. My Web Server — Add additional tags

Application and OS Images (Amazon Machine Image) Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Quick Start

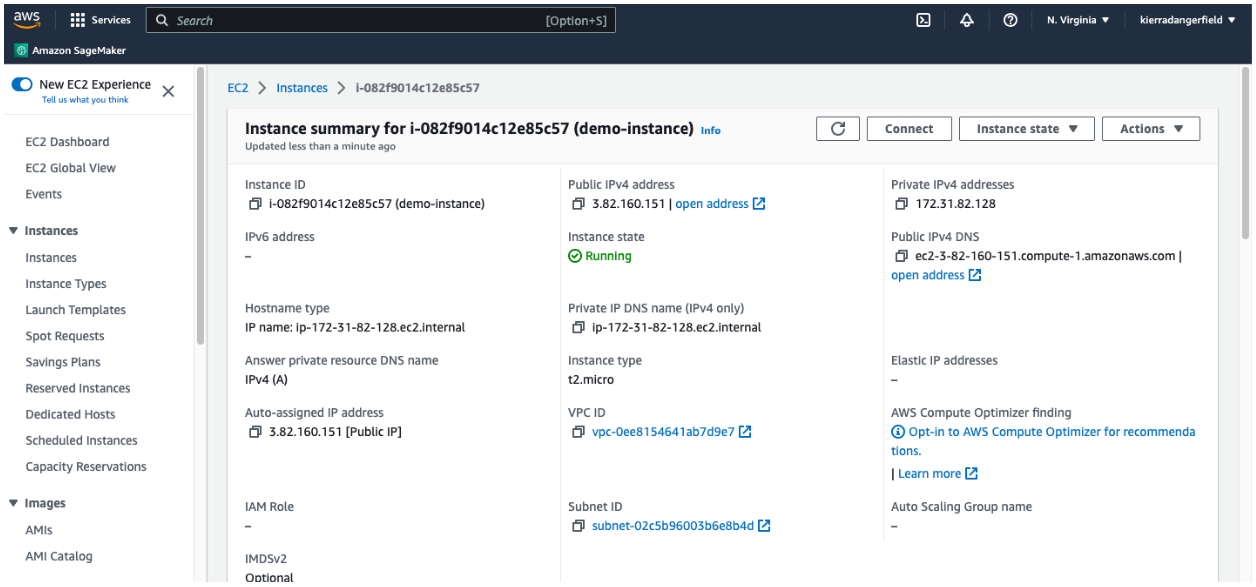
Summary

Number of instances Info: 1

Storage (volumes)

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Cancel — Launch instance

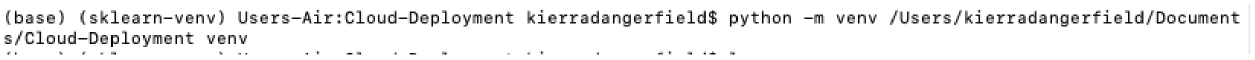


Instance summary for i-082f9014c12e85c57 (demo-instance) Info

Updated less than a minute ago

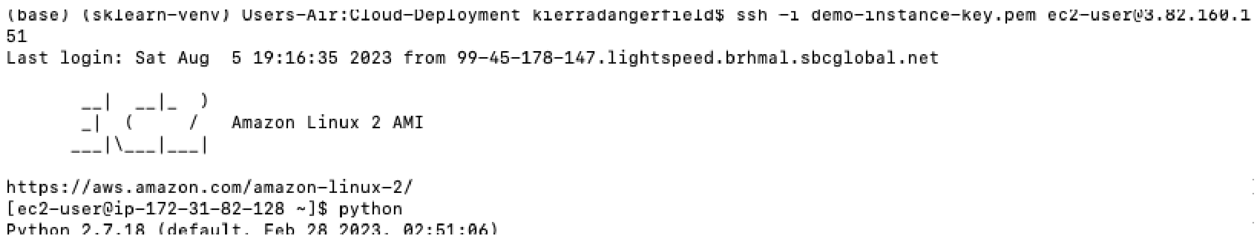
Instance ID i-082f9014c12e85c57 (demo-instance)	Public IPv4 address 3.82.160.151 | open address	Private IPv4 addresses 172.31.82.128
IPv6 address –	Instance state Running	Public IPv4 DNS ec2-3-82-160-151.compute-1.amazonaws.com | open address
Hostname type IP name: ip-172-31-82-128.ec2.internal	Private IP DNS name (IPv4 only) ip-172-31-82-128.ec2.internal	
Answer private resource DNS name IPv4 (A)	Instance type t2.micro	Elastic IP addresses –
Auto-assigned IP address 3.82.160.151 [Public IP]	VPC ID vpc-0ee8154641ab7d9e7	AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations. | Learn more
IAM Role –	Subnet ID subnet-02c5b96003b6e8b4d	Auto Scaling Group name –
IMDSv2 Optional		

Created a virtual environment



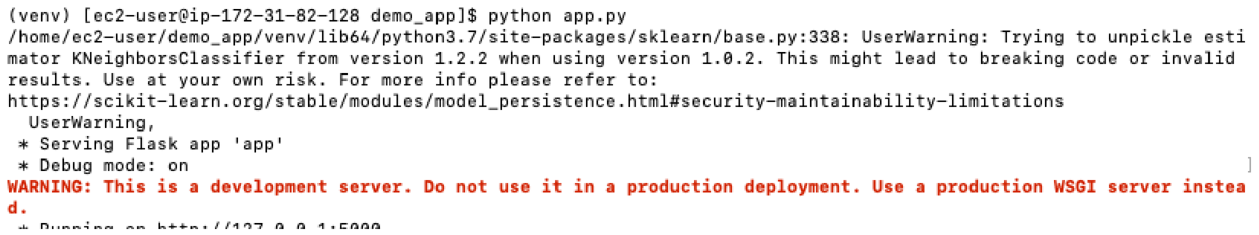
```
(base) (sklearn-venv) Users-Air:Cloud-Deployment kierradangerfield$ python -m venv /Users/kierradangerfield/Document
s/Cloud-Deployment venv
```

Connected the instance to the virtual environment



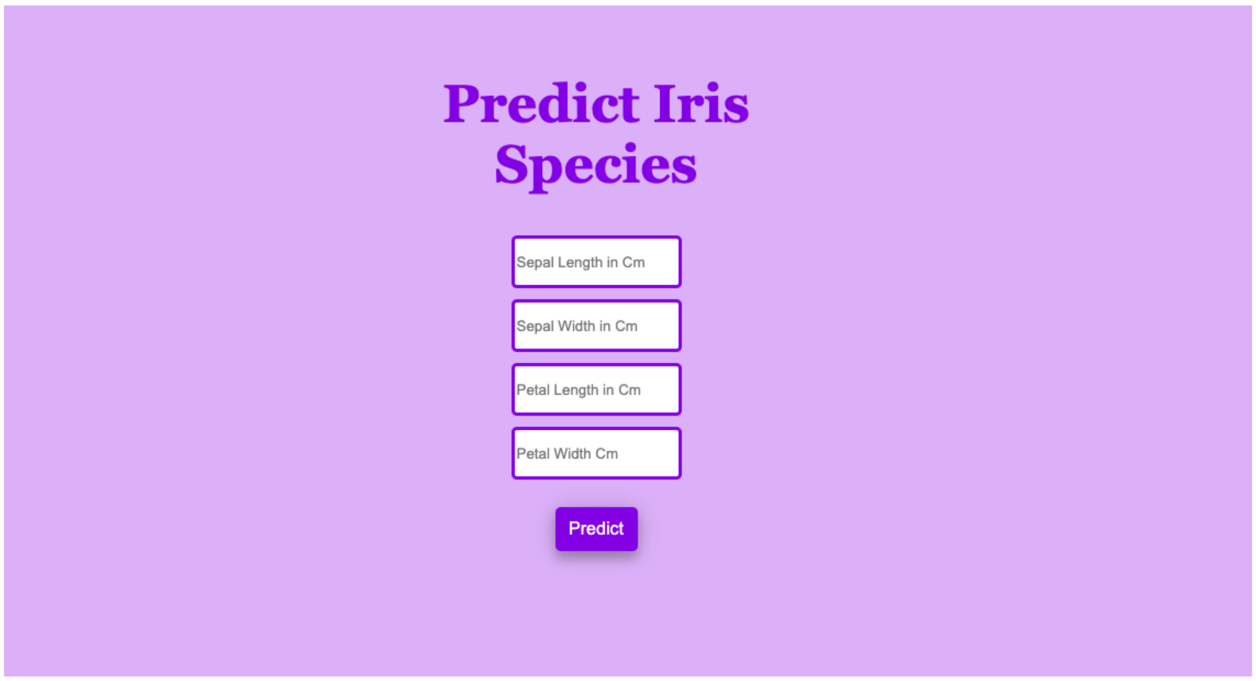
```
(base) (sklearn-venv) Users-Air:Cloud-Deployment kierradangerfield$ ssh -i demo-instance-key.pem ec2-user@3.82.160.1
51
Last login: Sat Aug  5 19:16:35 2023 from 99-45-178-147.lightspeed.brhmal.sbcglobal.net

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-82-128 ~]$ python
Python 2.7.18 (default, Feb 28 2023, 02:51:06)
```

Ran app



```
(venv) [ec2-user@ip-172-31-82-128 demo_app]$ python app.py
/home/ec2-user/demo_app/venv/lib64/python3.7/site-packages/sklearn/base.py:338: UserWarning: Trying to unpickle esti
mator KNeighborsClassifier from version 1.2.2 when using version 1.0.2. This might lead to breaking code or invalid
results. Use at your own risk. For more info please refer to:
https://scikit-learn.org/stable/modules/model_persistence.html#security-maintainability-limitations
  UserWarning,
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instea
d.
 * Running on http://127.0.0.1:5000
```

App


Predict Iris Species
Sepal Length in Cm
Sepal Width in Cm
Petal Length in Cm
Petal Width Cm
Predict