### Physical Activity Fitness Prediction Using IBM Watson

**A project on Physical Activity Fitness Prediction Using IBM Watson with a small web application**

By HEMANTH DURGA RAJESH PAYYAVULA

Physical Activity Fitness Prediction Using IBM Watson

**Project Idea**:

Sedentary lifestyle is defined by the absence of physical activity practices throughout the day and causes a decrease in caloric expenditure. This behavior is explained by the inappropriate lifestyle, for example, too much time sitting or lying down and still eating unhealthy foods during this time of immobilization. Currently, a third of the adult world population is physically inactive and this generates 5 million deaths per year (The LAncet, 2012). In addition to contributing to several chronic diseases, physical inactivity also influences mood, sleep quality and body weight

The objective of the project is answering a simple question, “does exercise/working-out improve a person’s activeness?”. For the scope of this project a person’s activeness was the measure of their daily step-count (the number of steps they take in a day). We are going to build a Machine Learning model which predicts the activeness or inactiveness of a person based on the Mood and number of steps taken in a day. Mood was measured in either "Happy", "Neutral" or "Sad" which were given numeric values of 300, 200 and 100 respectively. Feeling of activeness was measured in either "Active" or "Inactive" which were given numeric values of 500 and 0 respectively.

**By the end of this project:**

* You’ll be able to understand the problem to classify if it is a regression or a classification kind of problem.
* You will be able to know how to pre-process / clean the data using different data pre-processing techniques.
* You will be able to analysis or get insights of data through visualization.
* Applying different algorithms according to dataset and based on visualization.
* You will be able to know how to find the accuracy of the model.
* You will be able to know how to build a web application using the Flask framework.

**Prerequisites :**

To complete this project, you must require following software’s, concepts and packages

1. In order to develop this project, we need to install Anaconda and PyChram
2. **Anaconda Navigator**:

Anaconda Navigator is a free and open-source distribution of the Python and R programming languages for data science and machine learning related applications. It can be installed on Windows, Linux, and macOS.Conda is an open-source, cross-platform, package management system. Anaconda comes with so very nice tools like Jupyter Lab, Jupyter Notebook,

For this project, we will be using Jupyter notebook and Spyder

1. To install Anaconda navigator and to know how to use Jupyter Notebook & Spyder using Anaconda watch the video with this

<https://youtu.be/5mDYijMfSzs>

**Follow below steps to download required packages :**

* Open the anaconda prompt.
* Type “pip install pandas” and click enter.
* Type “pip install matplotlib” and click enter.
* Type “pip install numpy” and click enter.
* Type “pip install scikit-image” and click enter.
* Type “pip install scikit-learn” and click enter.
* Type “pip install Flask” and click enter.

For Prior Knowledge

Watch video By

<https://youtu.be/kE5QZ8G_78c>

<https://youtu.be/6za9_mh3uTE>

<https://youtu.be/lj4I_CvBnt0>

Tasks:

1. Data Collection.

a.Collect the dataset or Create the dataset

1. Data Pre- processing.
   1. Import the Libraries.
   2. Importing the dataset.
   3. Exploratory Data Analysis
   4. Checking for Null Values.
   5. Data Visualization.
   6. Splitting Data into Train and Test.
2. Model Building
   1. Training and testing the model
   2. Evaluation of Model
3. Application Building
   1. Create an HTML file
   2. Build a Python Code.

**Project Folder**

We have three folders dataset, Flask and Model Building

A python file called app.py for server side scripting.

We need the model which is saved and the saved model in this content is (**fitness.pkl**).

Templates folder which contains home.html, result.html and resultnew.html files.

Static folder which contains css styles.css folder which contains

### The above image indicates the Project Structure

### First

### Dataset Collection

### 1.Collect The Dataset Or Create The Dataset

### 2.collect the dataset using this

### <https://www.kaggle.com/aroojanwarkhan/fitness-data-trends#25.csv>

3.Here we are using a data set which you can find in the above link and you can download it from the reference.

### Second

### Data Pre-Processing

### 

### Import libraries

1.Pandas

2.Numpy

3.Sklearn

4.Joblib

5.Pickle

6.Flask

7 Matplotlib and Seaborn

8. Train test split

9. Counter

### Reading The Dataset

Read the Dataset using command

data=pd.read\_csv(r” F:\Project\Physical activity Fitness prediction using IBM Watson\dataset\25.csv”)

**Or**

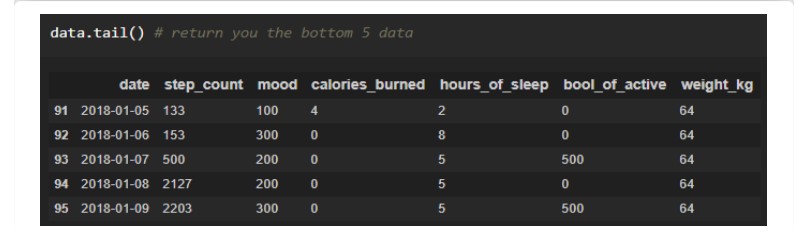
Data=pd.read\_csv(“25.csv”)

### Exploratory Data Analysis

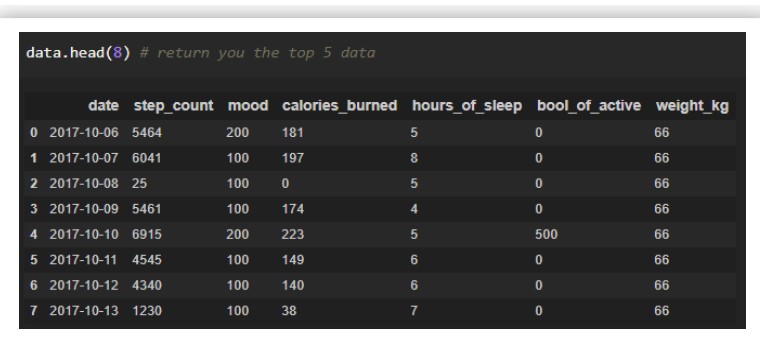
* To check the first five rows of the dataset, we have a function called head( ).



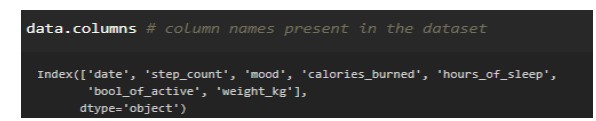
* To check the last five rows of the dataset, we have a function called tail().



* However, you can always specify how many rows to show such as dataset.head(8) to show 8 rows.

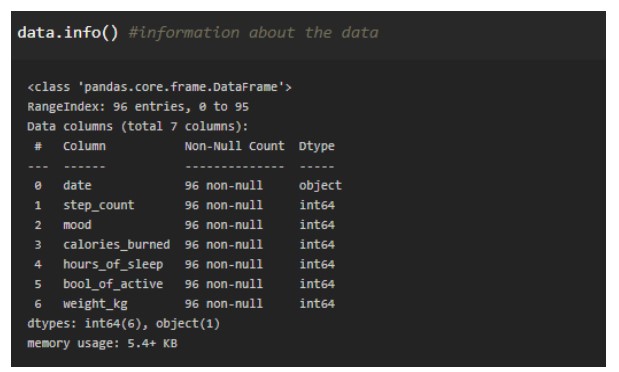


* For finding the names of the columns present in the dataset we make use of columns



**Understanding Data Type and Summary of features**

see how our dataset is, by using the info()

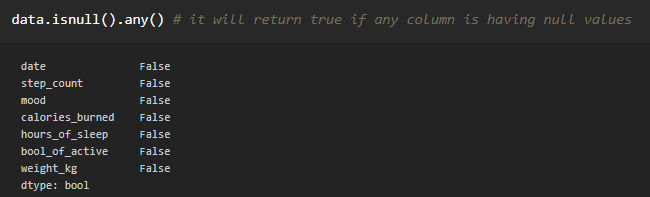


**Selecting the Categorical features**  
 Creating a function to fetch the number of unique elements present in each column. The columns which are having a minimum number of unique elements or category present will be considered as the categorical columns and the remaining columns will be a numerical column.

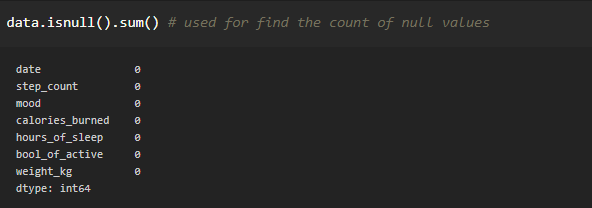


### Checking For Null Values

* We will be using **isnull().any()** method to see which column has missing values.
* We get this as result

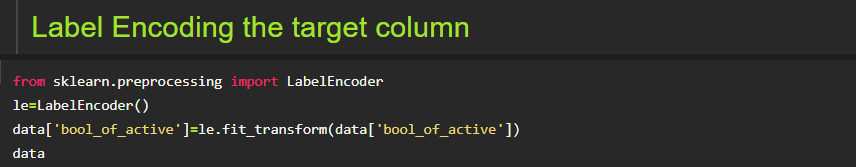


* Word “True” that the particular column has missing values, we can also see the count of null values in each column by using **isnull().sum()**.



* As our data is not having any null values so we don’t have to handle the null values but by using the seaborn heatmap libraries we can observe the null values graphically.

### Label Encoding

**Label Encoding** is a popular encoding technique for handling categorical variables. In this technique, each label is assigned a unique integer based on alphabetical ordering.

### Data Visualization

* For

step\_count

calories\_burned

weights\_kg

hours\_of\_sleep

**Finding correlation between the independent Columns using :**

* finding the correlation between variables we have corr() available.

corr=data.corr()

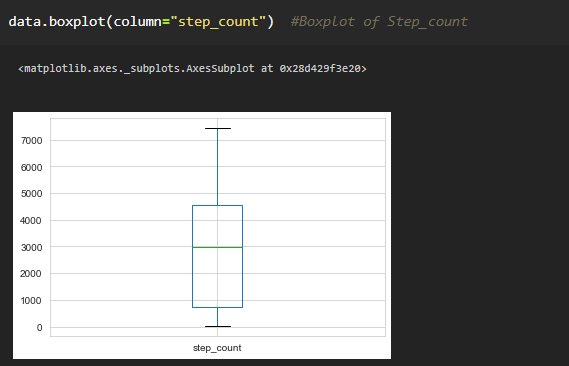
sns.heatmap(corr,xticklabels=corr.columns,yticklabels=corr.columns)

* Plotting a pair plot to showcase the relationship among all the different columns

sns.set\_style("whitegrid")

sns.pairplot(data)

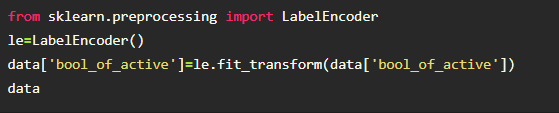
**Bivariate Analysis**

* step count vs mood
* Step\_count vs Activeness
* Mood vs Calories\_burned
* Calories\_burned vs Activeness
* Sleeping\_hours vs mood
* Sleeping\_hours vs Activeness
* Mood vs Activeness
* **Outliers Detection using boxplot**
* Outliers are observations in a dataset that don't fit in some way or you can say those values are of no use and also may effect our results.
* So with the help of boxplot we can visualize and check whether the data contains any outliers or not

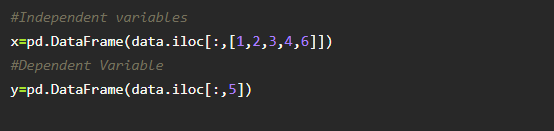


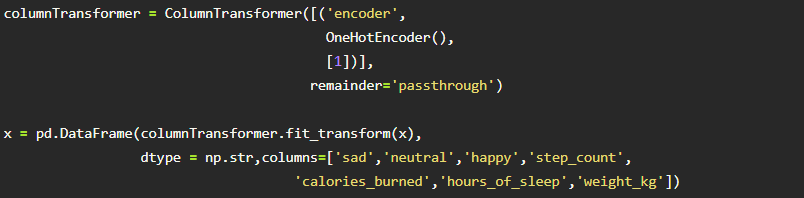
Label Encoding of target data

As in our target data we have two classes i,e. 0(Inactive) , 500 (active) we will be using a label encoding technique to encode it to 0 and 1 for better understanding of the classes



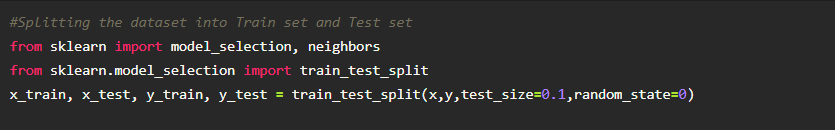
### Splitting The Dataset Into Dependent And Independent Variables

Let’s create out independent and dependent variables: 

[Column Transformer](https://scikit-learn.org/stable/modules/generated/sklearn.compose.ColumnTransformer.html) is a new approach where we apply both labels and one hot encoding in a single step.

### Split The Dataset Into Train Set And Test Set

The train-test split is a technique for evaluating the performance of a machine learning algorithm.

* **Train Dataset**: Used to fit the machine learning model.
* **Test Dataset**: Used to evaluate the fit machine learning model.

### Train And Test The Model Using Decision Tree

There are several Machine learning algorithms to be used depending on the data you are going to process such as images, sound, text, and numerical values. The algorithms that you can choose according to the objective that you might have may be Classification algorithms are Regression algorithms.

Example: 1. Linear Regression.

2. Logistic Regression.

3. Random Forest Regression / Classification.

4. Decision Tree Regression / Classification.

**Now we apply the Decision Tree algorithm on our dataset**.

A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.

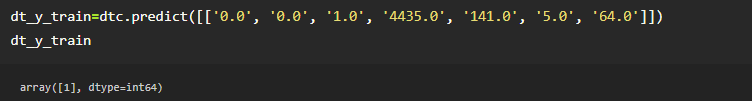
**Build the model with the DecisionTreeClassifier**.

We’re going to use x\_train and y\_train obtained above in train\_test\_split section to train our decision tree classifier model. We’re using the fit method and passing the parameters as shown below.

**Predict the values**

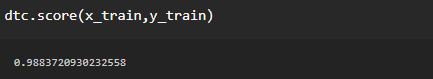
Once the model is trained, it’s ready to make predictions. We can use the **predict** method on the model and pass **x\_test** as a parameter to get the output as **dt\_y\_train**.

Notice that the prediction output is an array of real numbers corresponding to the input array.



### Model Evaluation

Finally, we need to check to see how well our model is performing on the test data. There are many evaluation techniques. For this, we evaluate scores produced by the model.



**Save the Model**

Pickle is used for serializing and de-serializing Python object structures, also called marshalling or flattening. Serialization refers to the process of converting an object in memory to a byte stream that can be stored on disk or sent over a network. Later on, this character stream can then be retrieved and de-serializedback to a Python object.Save our model by importing pickle file



Here, **dtc** is our decision tree classifier class with saving as **fitness.pkl** file. **Wb** is the write binary in bytes.

### Application Building

### Create An HTML File

* We use HTML to create the front end part of the web page.
* Here, we created 2 html pages- index.html, web.html.
* index.html displays the home page.
* web.html accepts the values from the input and displays the prediction.
* For more information regarding [HTML](https://www.w3schools.com/bootstrap/bootstrap_forms_inputs.asp)

We also use [JavaScript-main.js](https://www.w3schools.com/js/DEFAULT.asp) and [CSS-main.css](https://www.w3schools.com/css/) to enhance our functionality and view of HTML pages



Above is the home.html page

Like this create indexnew.html(for Prediction)

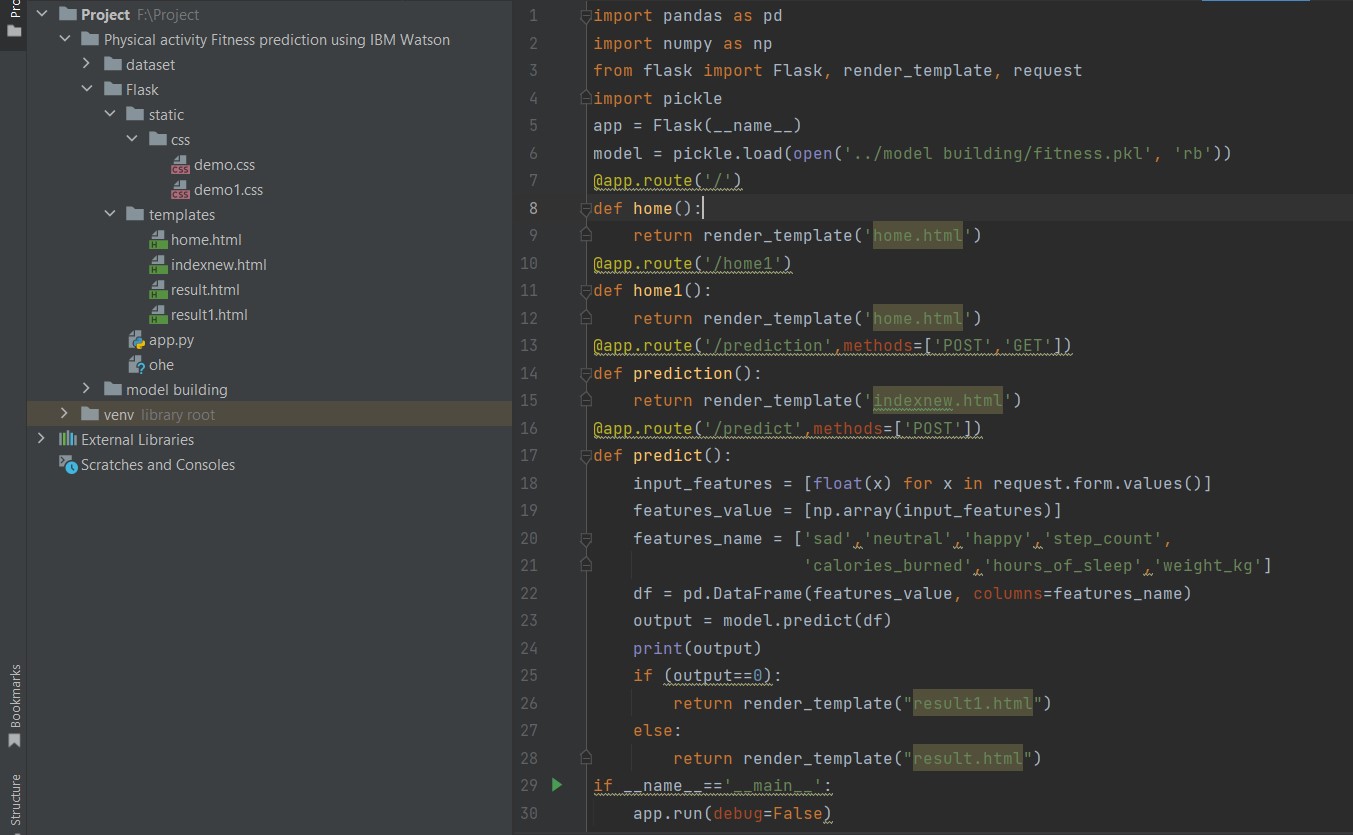
result.html(you are Active….)

result.html(you are Lazzy…..)

Next create python Flask file(app.py) for web framework

### Build Python Code

* Let us build a flask file ‘app.py’ which is a web framework written in python for server-side scripting. Let’s see the step by step procedure for building the backend application.
* App starts running when the “\_\_name\_\_” constructor is called in main.
* render\_template is used to return an html file.
* “GET” method is used to take input from the user.
* “POST” method is used to display the output to the user.



**Here we are running it on localhost:5000**

<http://127.0.0.1:5000>

### Run The App

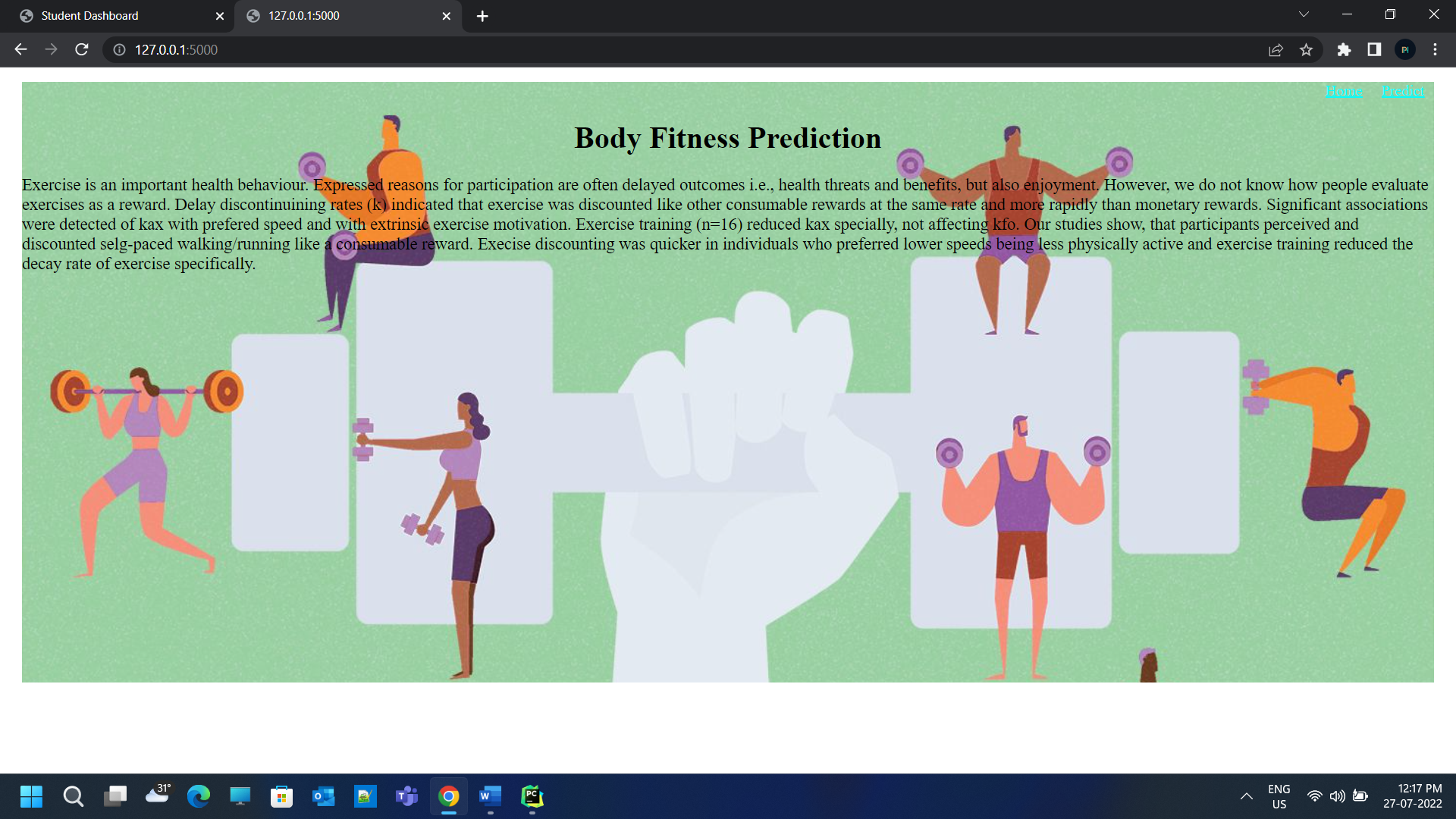
· Open anaconda prompt from the start menu

· Navigate to the folder where your python script is.

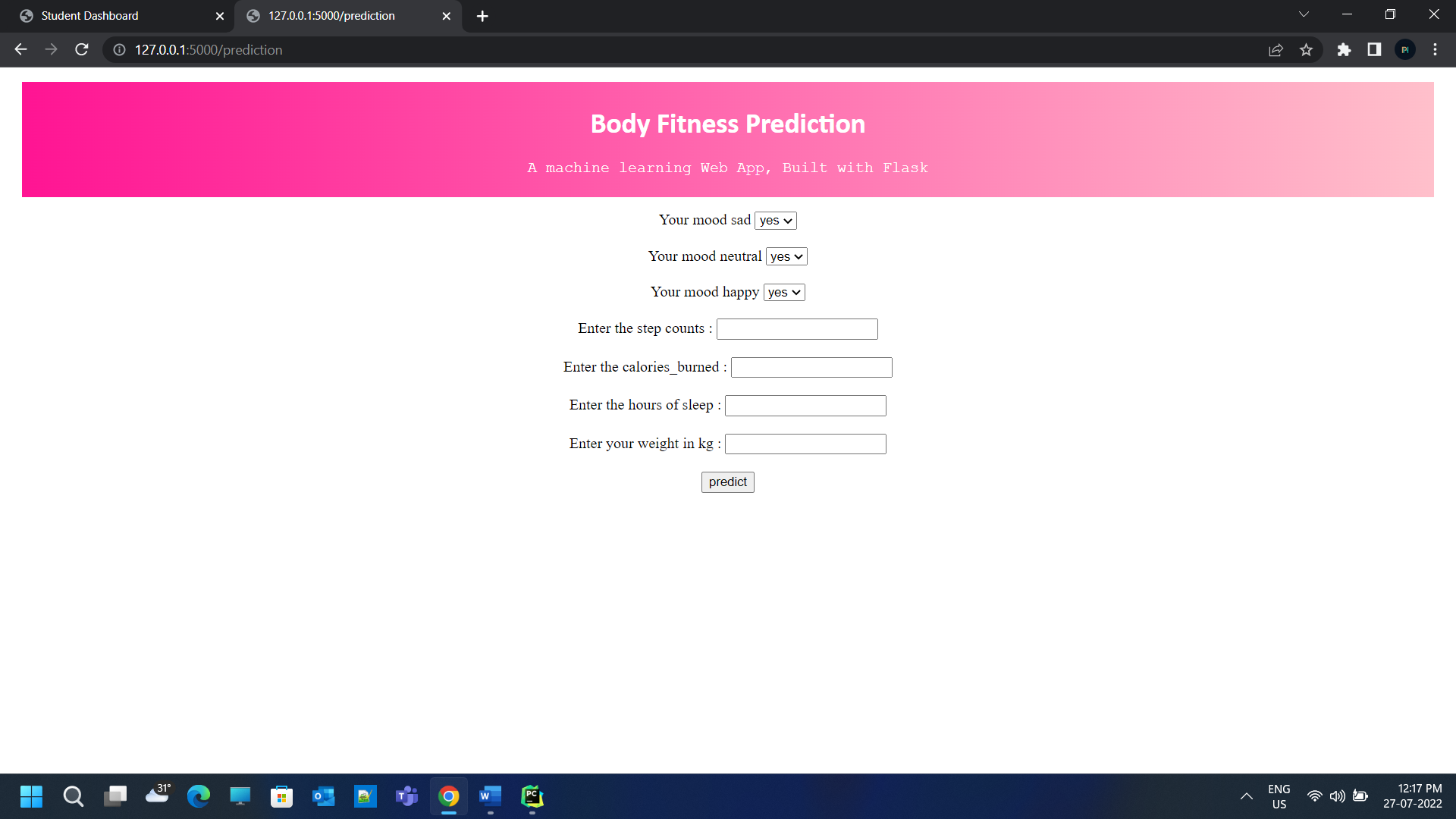
· Now type “python app.py” command

· Navigate to the localhost where you can view your web page

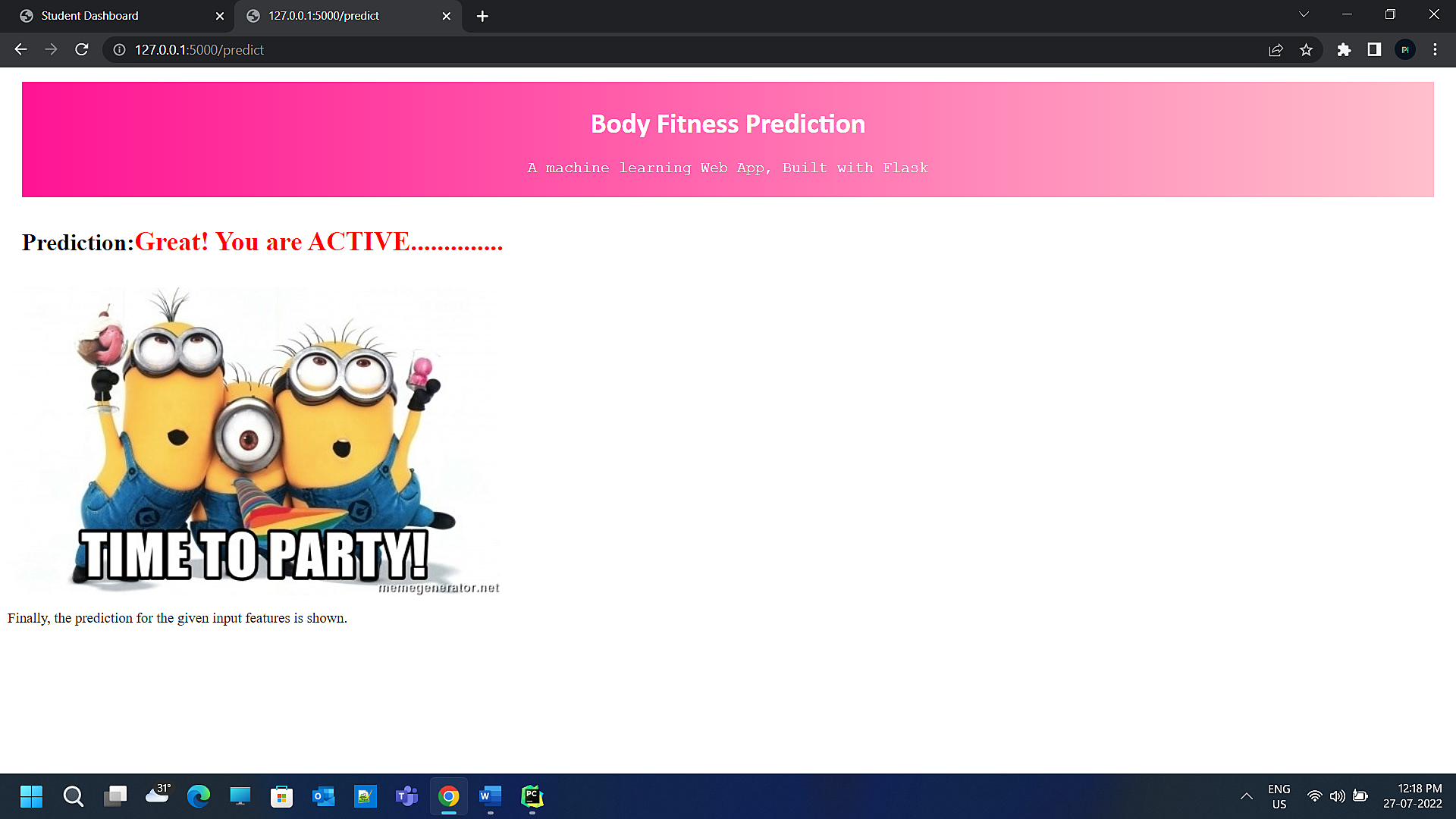
* Showcasing The UI

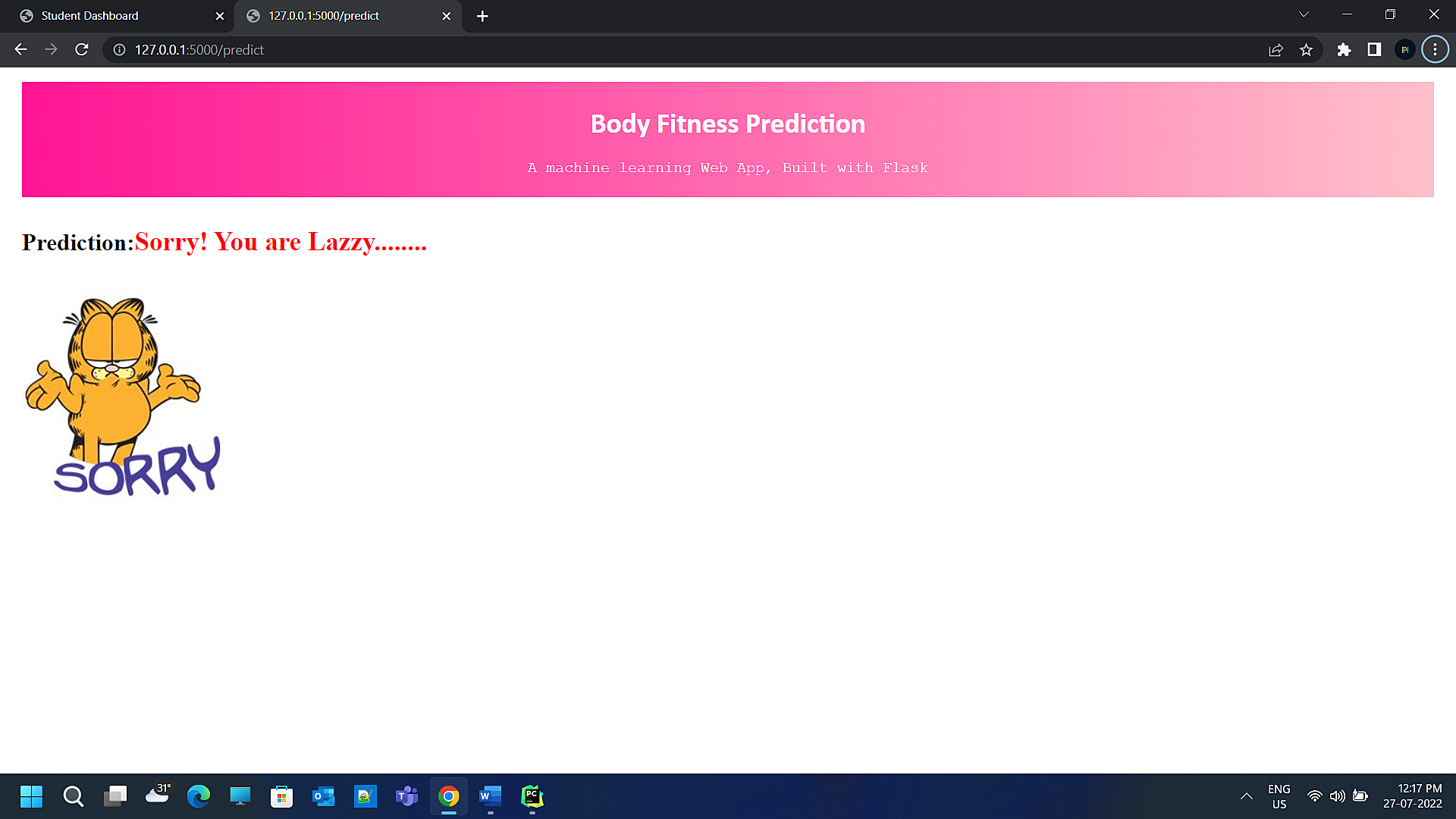


This is the prediction page where we get to choose the input from our local system and predict the output.



Finally, the prediction for the given input features is shown.





**Finally**

### Train The Model On IBM

In this milestone, you will learn how to build a Machine Learning Model and deploy it on the IBM Cloud.

### Train The ML Model On IBM

Train ML Model with IBM Notebook

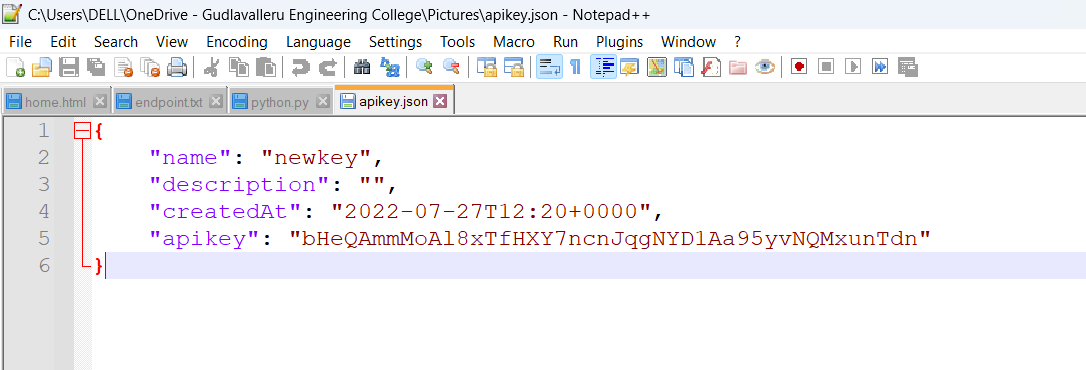
while Train ML Model with IBM Notebook we should use the (.ipynb) file in the in IBM Jpyter Notebook and upload the our own dataset

And we should install IBM watson machine Learning packing our notebook and it with our model to create IBM model.

**APIKEY**

APIkey is key use to link your IBM cloud account to your ibmapp.py to use the cloud dataset

APIkey is in the JSON format and to look like this



**IBM MODEL FILE**

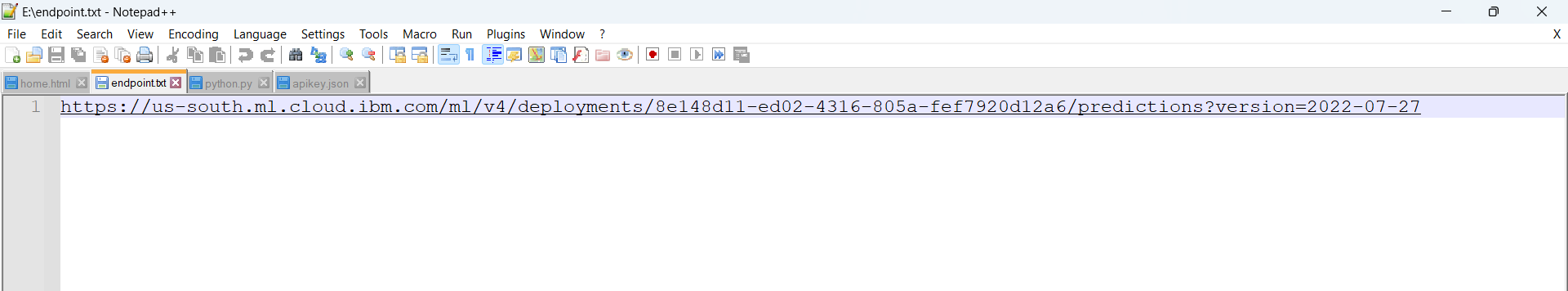
After

Using the this APIkey and IBM Machine Learning should creaate the IBM Model which is used to store in the IBM cloud and depolyed

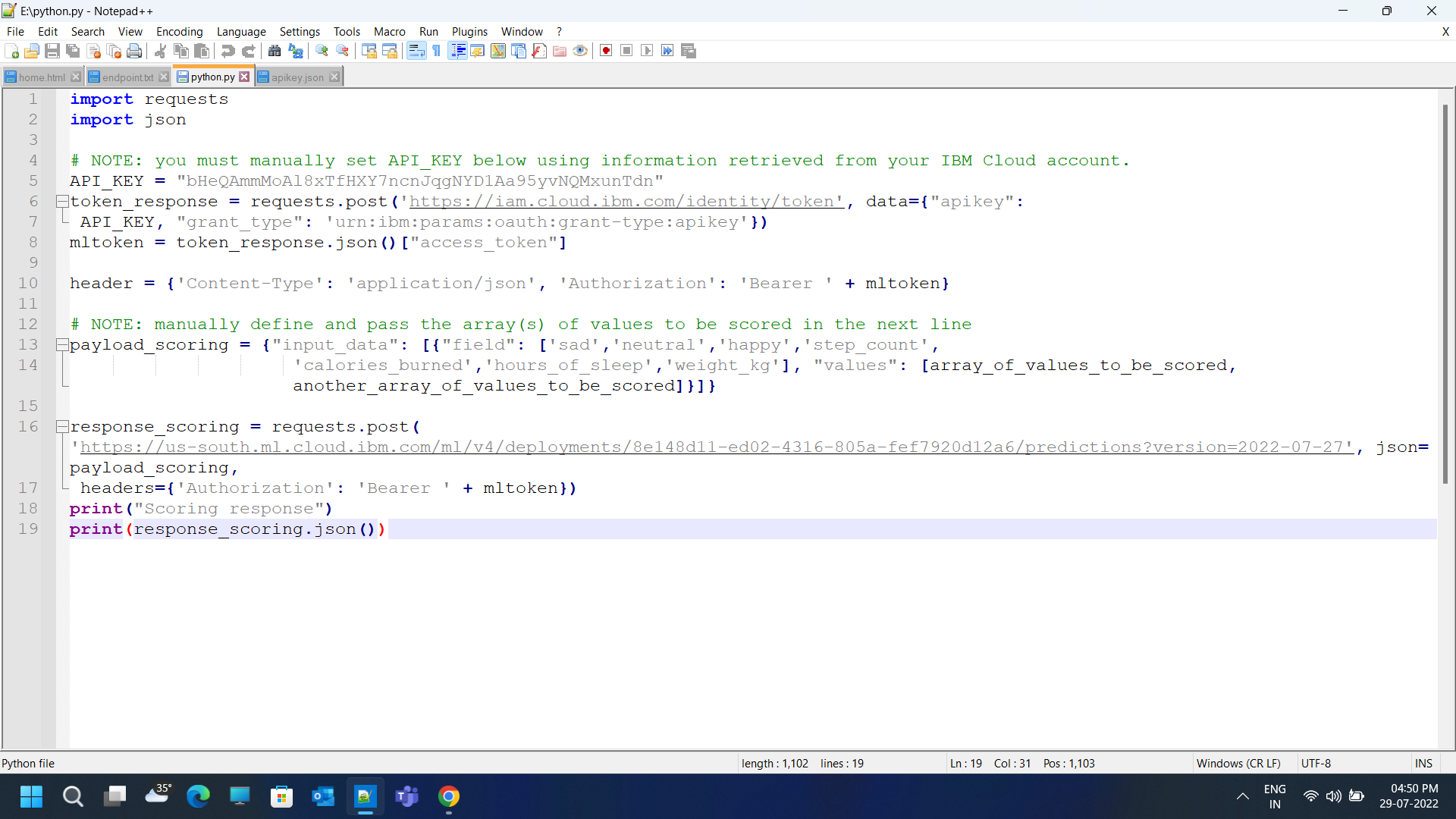
**DEPOLYMENT**

Depolyment is use in model for cloud storge of the IBM .Depolyment is done in the online method to get endpoint and python socring code

**ENDPOINT**



**PYTHON CODE OF IBM**



**AFTER DEPOLYMENT**

we should integrate this code with our app.py code to acess the IBM Model file

For the Reference

I used this videos

FOR DEPOLYMENT

[https://youtu.be/TysuP3KgSzc](https://youtb.be/TysuP3KgSzc)

FOR INTEGRATE

<https://youtu.be/ST1ZYLmYw2U>

**By this we successfully complete our project**

**Thank you**