Report for Diabetics Prediction System

21st of November, 2020

Prepared By: Kashifa Khursheed

IBM Project

### Proposed System:

In this, we need to diagnostically predict whether or not a patient has diabetes, based on certain diagnostic measurements included in the dataset. Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage.

The datasets consist of several medical predictor variables and one target variable, Diabetes. Predictor variables include the number of pregnancies the patient has had, their BMI, insulin level, age, and so on.

### Skills Required:

* Flask Integration
* IBM Nodered
* IBM Watson Studio
* IBM Machine Learning
* IBM Cloud Object Storage

### Software Requirements:

1. Operating system: Windows 10 9/8/19

2. Coding Language: HTML, PYTHON

3. Services used:

* IBM Watson Studio-ky
* Machine Learning-te
* node-red-hzblo-2020--cloudant
* Continuous Delivery

4. Cloud Foundry apps: Node RED HZBLO

5. Storage: Cloud Object Storage-ol

### Outcome:

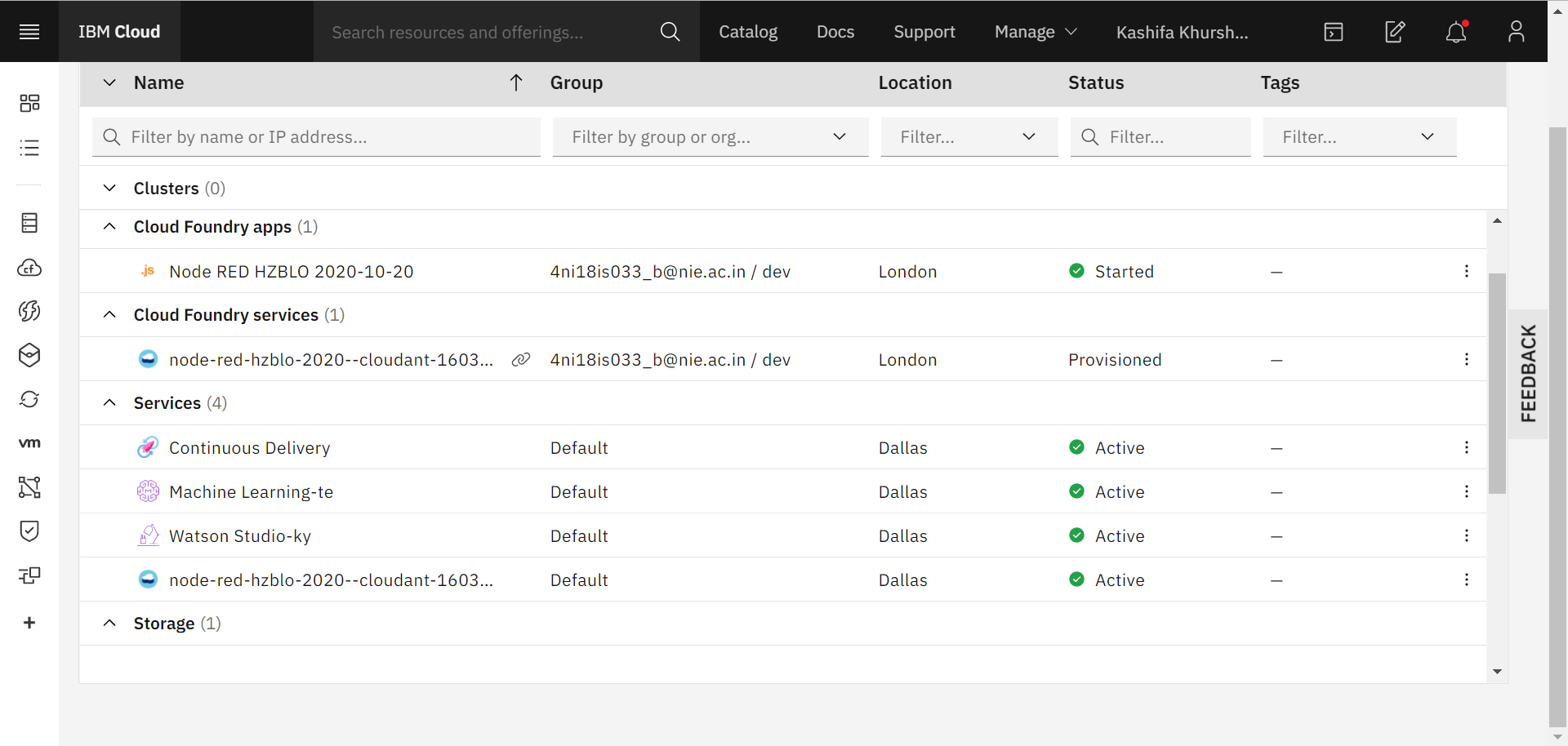
Develop an end-to-end web application that predicts the probability of females having diabetes. The application has to be built with Python-Flask or Django framework with the machine learning model trained & deployed on IBM Watson Studio.

### Tasks Completed:

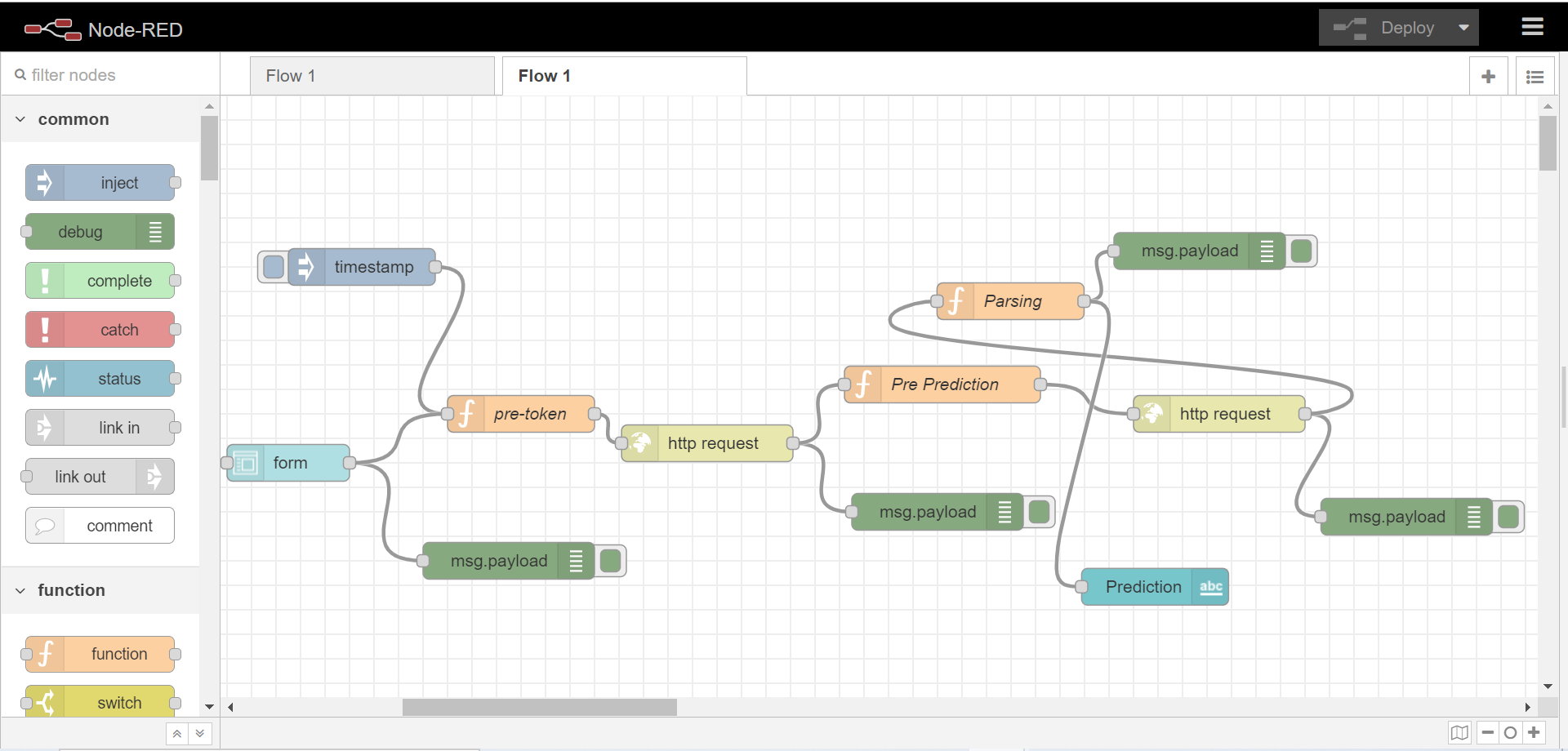
* Collecting Dataset
* Create IBM Academic Initiative Account
* Login to IBM Cloud
* Create Cloud Object Storage Service
* Download Watson Studio Desktop
* Create Watson Studio Platform
* Create Machine Learning Service
* Create A Project in Watson Studio
* Upload the Dataset
* Train a model in Watson Studio
* Deploying the Model
* Create UI using Node Red service
* Build Flask Application

### Steps followed and Snapshots:

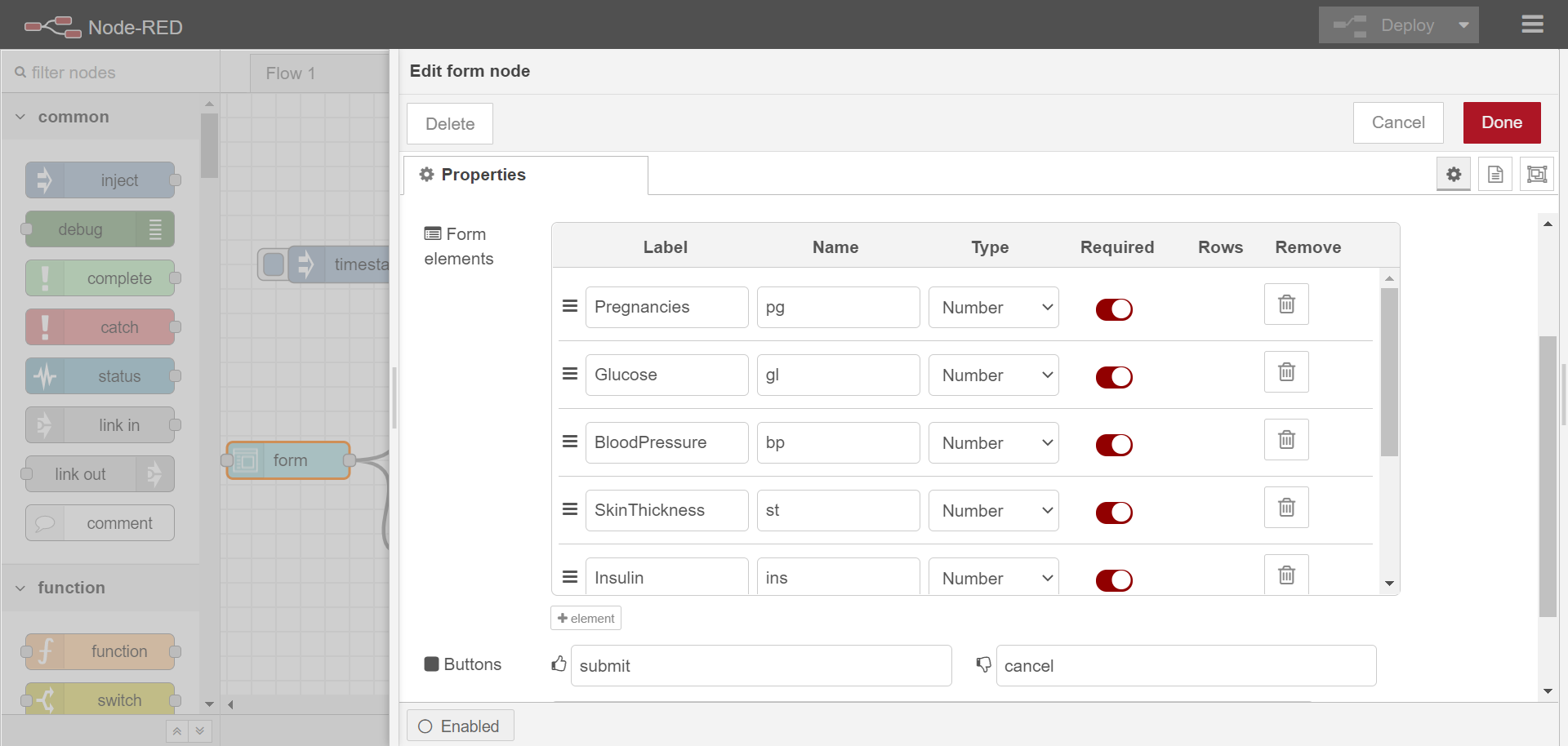
* In the below snapshot, the services, storage space and cloud foundry app used can be seen.



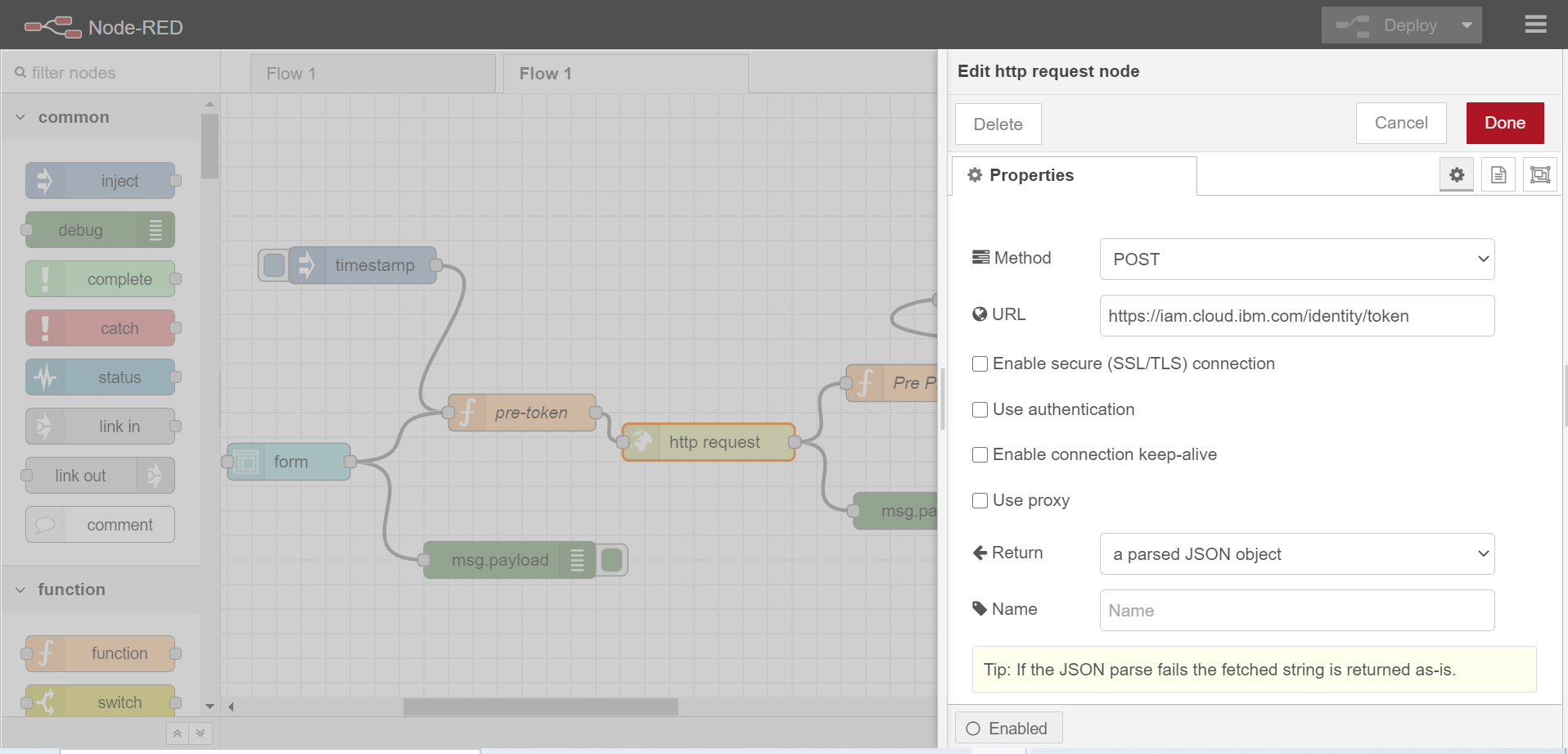
* The below is the complete snapshot of the node-red flow used:



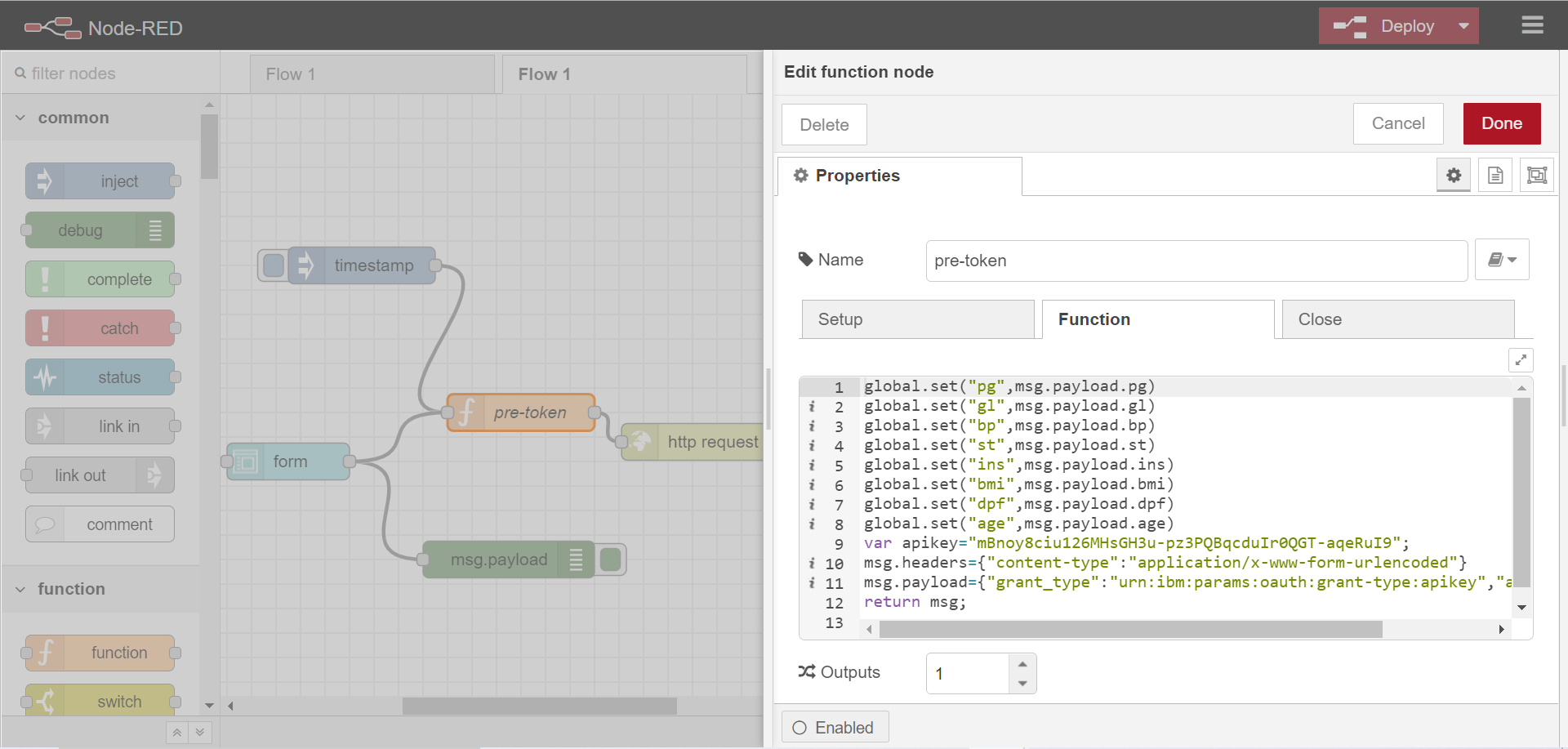
* The below snapshot shows form node details:



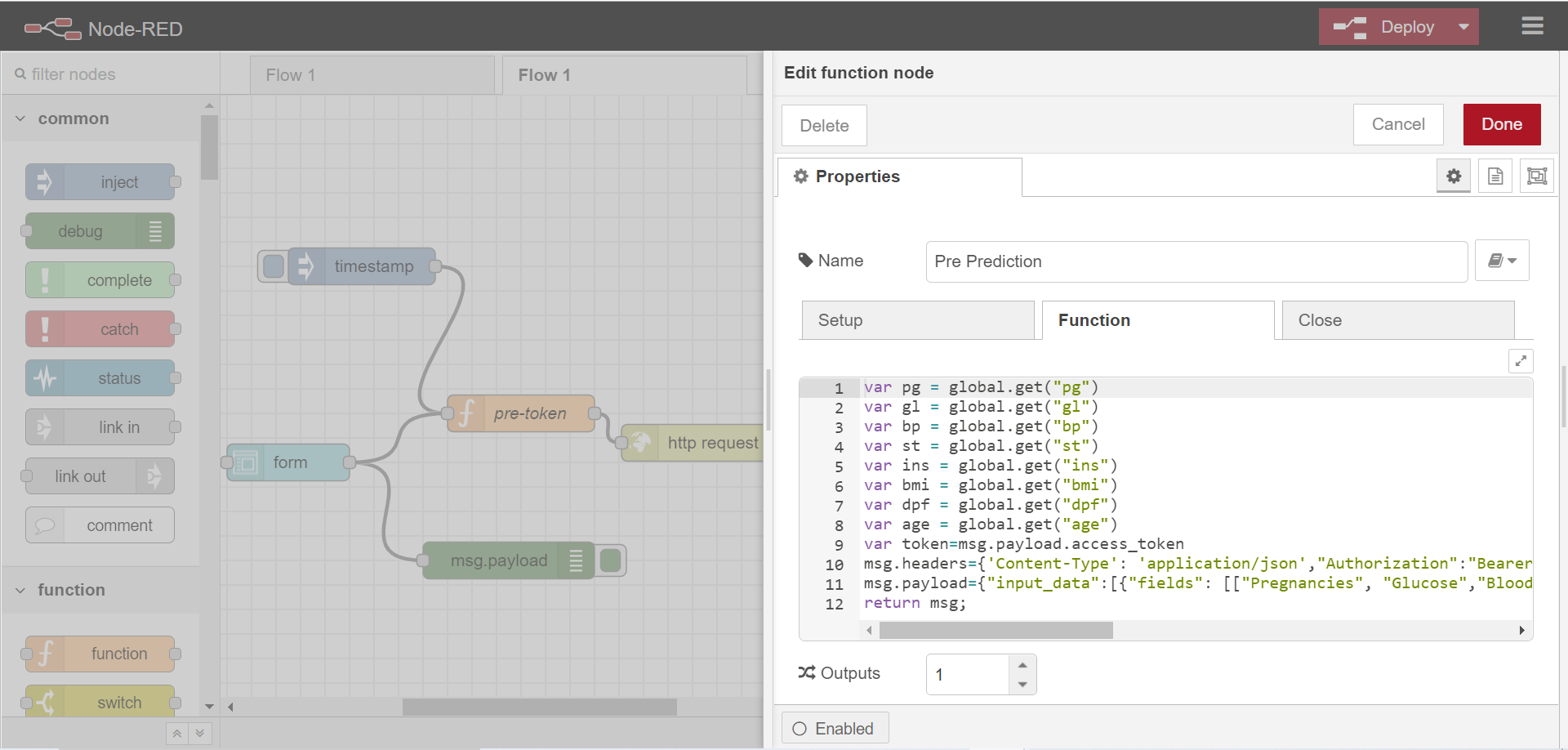
* The below snapshot shows http request node details:



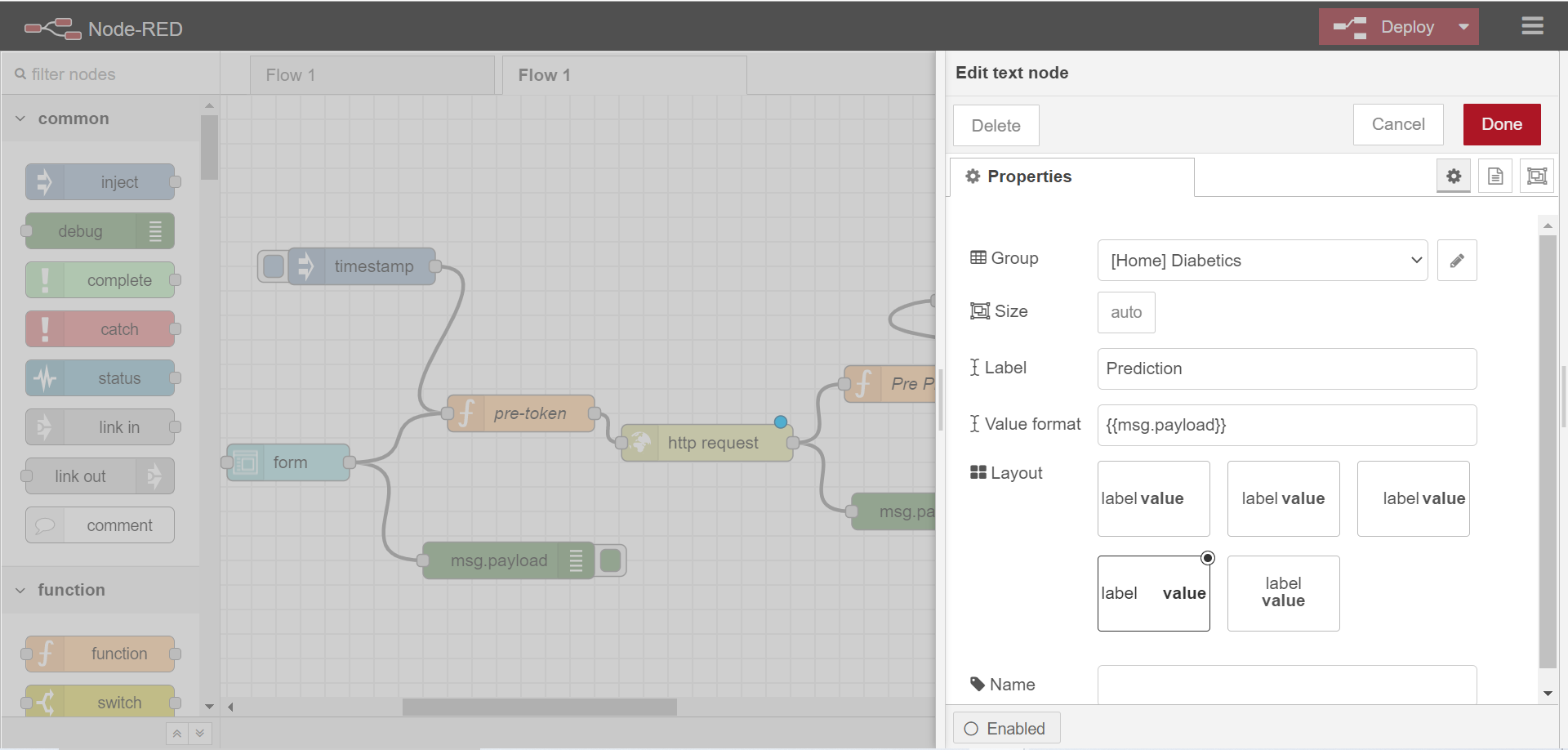
* The below snapshot shows pre-token node details:



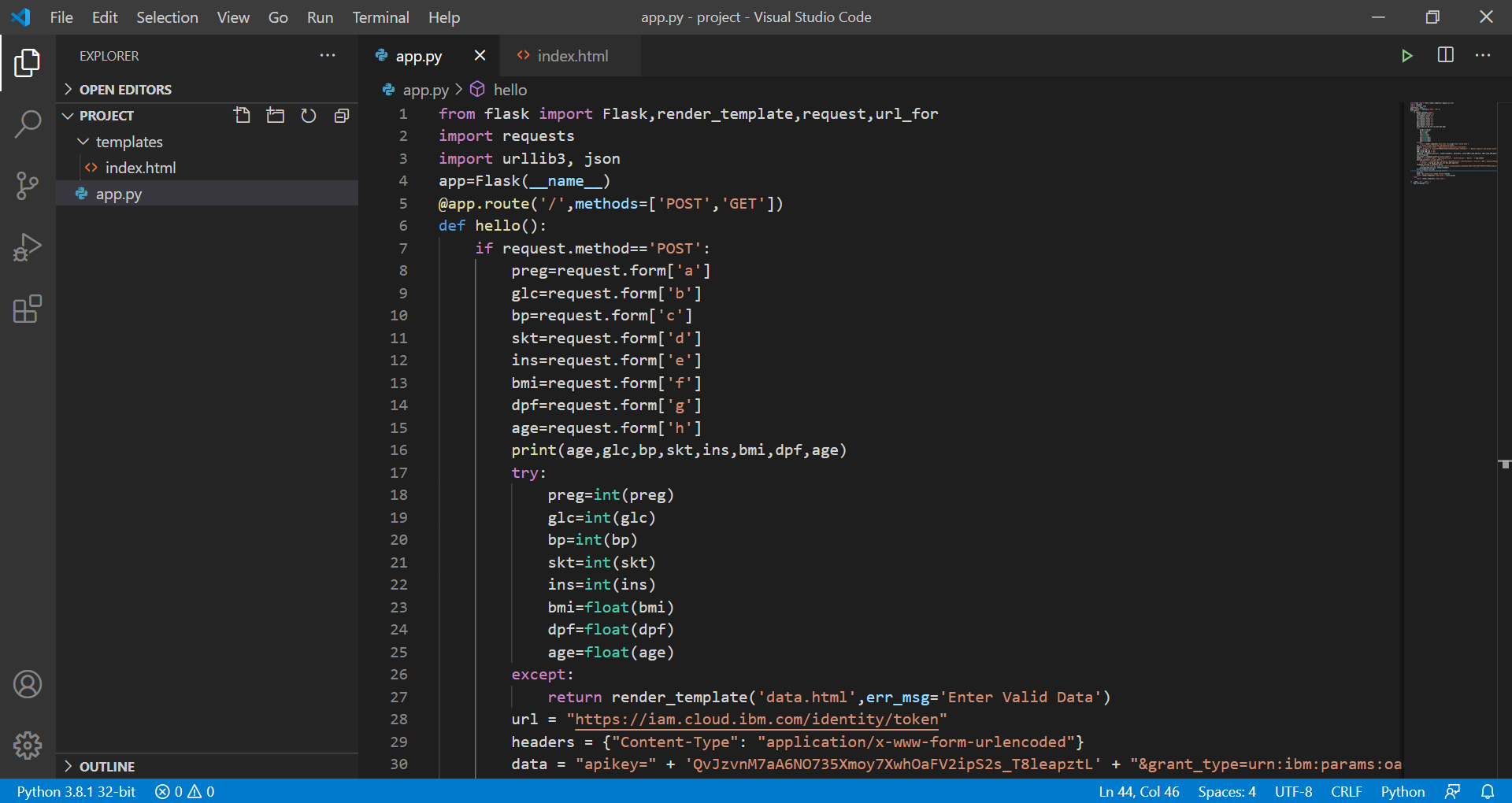
* The below snapshot shows pre-prediction node details:



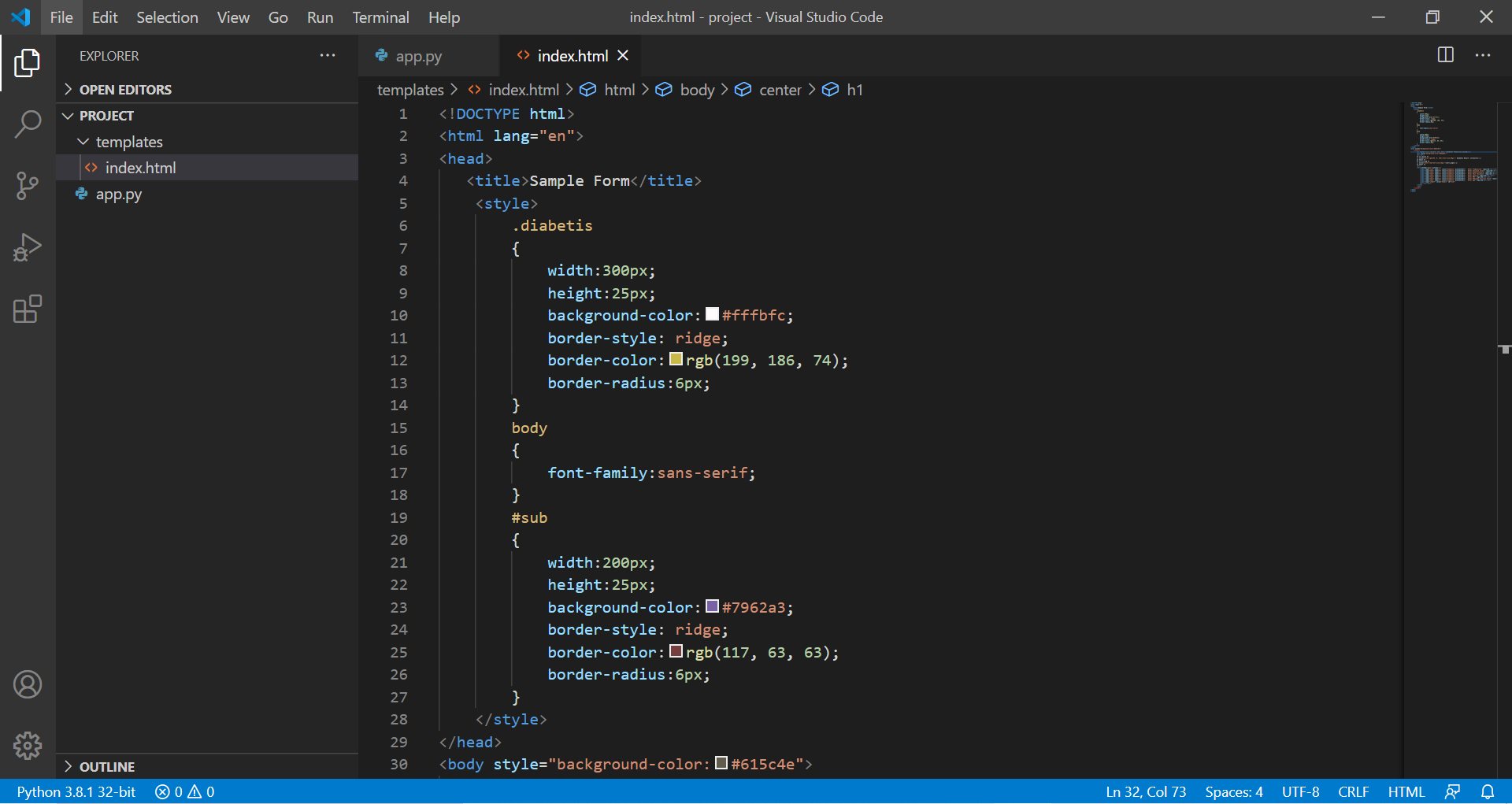
* The below snapshot shows prediction node details:



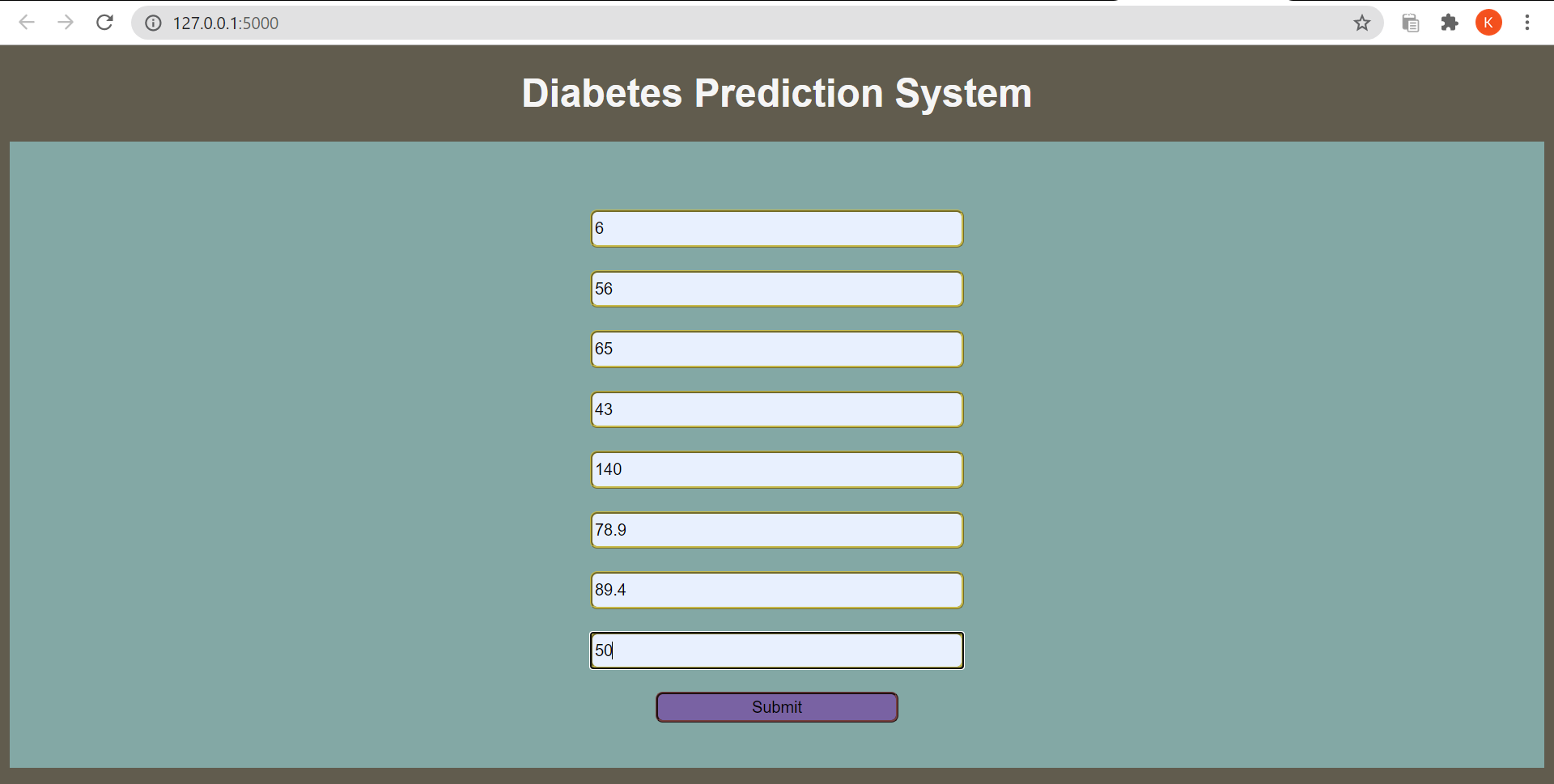
* The below snapshot shows the app.py written in VS code:



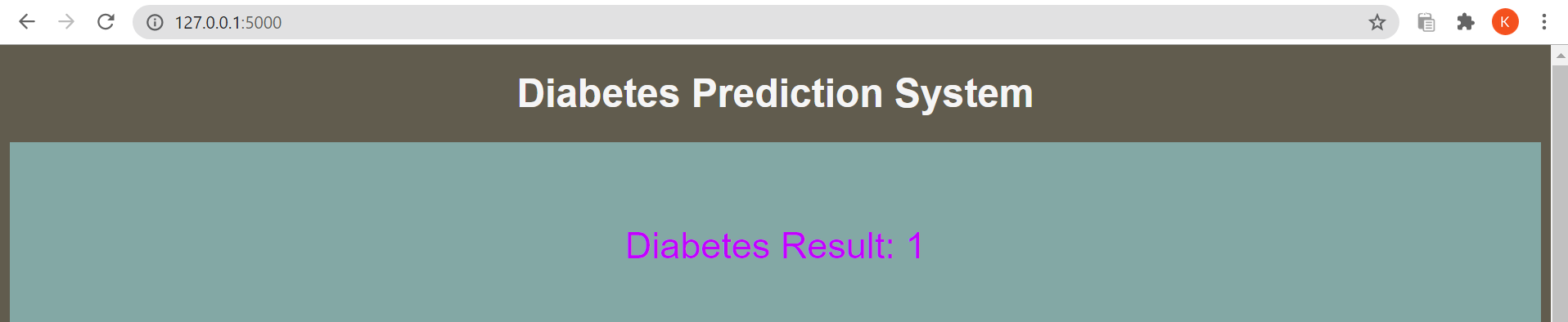
* The below snapshot shows index.html written in VS code:



* The below snapshot shows the UI where the user enters the specified details and gets back the result:



* The result that the user enters, he gets a predicted result right away on the screen: (1-positive/yes and 0-negative/no)



### Conclusion:

The model has been succesfully deployed and the UI has been succesfully created. The model used here is Binary Classification with an accuracy of upto 80%.