from fastapi import FastAPI

from fastapi.middleware.cors import CORSMiddleware

from pydantic import BaseModel

import pickle

import json

app = FastAPI()

origins = ["\*"]

app.add\_middleware(

CORSMiddleware,

allow\_origins=origins,

allow\_credentials=True,

allow\_methods=["\*"],

allow\_headers=["\*"],

)

class model\_input(BaseModel):

Gender : int

Age : int

A1C : float

FBS : int

GTT : int

Insulin : int

BMI : int

BPS : int

BPD : float

Skin : float

DPF : float

Smoking : float

Alcohol : float

Physical\_Activities : float

Pregnencies : int

Sleep : float

# Loading the saved model

diabetes\_model = pickle.load(open('diabetes\_model.sav', 'rb'))

@app.post('/diabetes\_prediction')

def diabetes\_pred(input\_parameters : model\_input):

input\_data = input\_parameters.json()

input\_dictionary = json.loads(input\_data)

gender = input\_dictionary['Gender']

age = input\_dictionary['Age']

a1c\_test = input\_dictionary['A1C']

fast\_blood\_sugar = input\_dictionary['FBS']

glucose\_tolarance\_test = input\_dictionary['GTT']

insulin = input\_dictionary['Insulin']

bmi = input\_dictionary['BMI']

blood\_pressure\_systolic = input\_dictionary['BPS']

blood\_pressure\_diastolic = input\_dictionary['BPD']

skin\_thickness = input\_dictionary['Skin']

diabetes\_pedigree\_function = input\_dictionary['DPF']

smoking = input\_dictionary['Smoking']

alcohol = input\_dictionary['Alcohol']

physical\_activities = input\_dictionary['Physical\_Activities']

pregencies = input\_dictionary['Pregnencies']

sleep\_hours = input\_dictionary['Sleep']

input\_list = [gender, age, a1c\_test, fast\_blood\_sugar, glucose\_tolarance\_test, insulin, bmi, blood\_pressure\_systolic, blood\_pressure\_diastolic, skin\_thickness, diabetes\_pedigree\_function, smoking, alcohol, physical\_activities, pregencies, sleep\_hours]

prediction = diabetes\_model.predict([input\_list])

if prediction[0] == 0:

return 'The Person is Normal'

elif prediction[0] == 1:

return 'The Person is Prediabetic'

elif prediction[0] == 2:

return 'The Person is Type 2 Diabetic'

elif prediction[0] == 3:

return 'The Person is Gestational Diabetic'

elif prediction[0] == 4:

return 'The Person is Type 1 Diabetic'

else:

return 'The Person is Type 1 Gestational Diabetic'