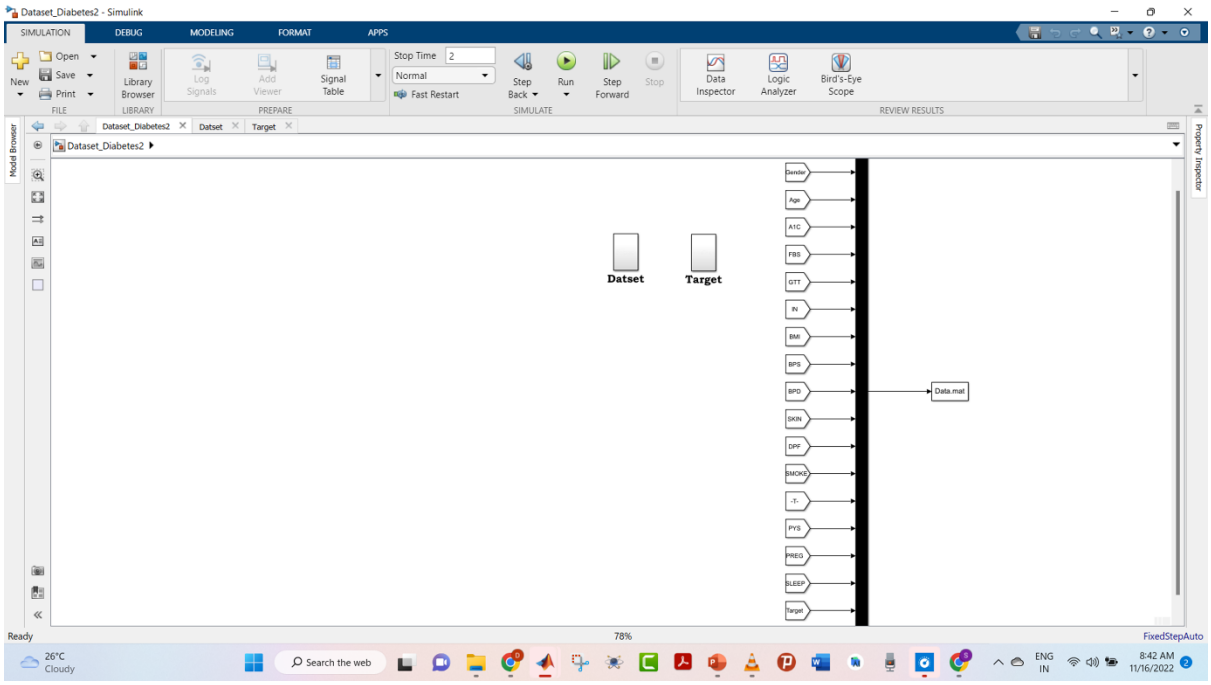
IDEATION

DATE	18 NOVEMBER 2022
TEAM ID	PNT2022TMID19223
PROJECT TITTLE	A GETURE BASED TOOL FOR STERILE BROWSING FOR RADIOLOGY IMAGES
TOTAL MARKS	4



IDEATION :

```
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'
```