from keras.models import load\_model from tkinter import \* import tkinter as tk import win32gui from PIL import ImageGrab, Image import numpy as np model = load\_model('mnist.h5') def predict\_digit(img): #resize image to 28x28 pixels img = img.resize((28,28)) #convert rgb to grayscale img = img.convert('L') img = np.array(img)

#reshaping to support our model input and normalizing img = img.reshape(1,28,28,1) img = img/255.0 #predicting the class res = model.predict([img])[0] return np.argmax(res), max(res) class App(tk.Tk): def \_\_init\_\_(self): tk.Tk.\_\_init\_\_(self) self.x = self.y = 0

# Creating elements self.canvas = tk.Canvas(self, width=300, height=300, bg = "white", cursor="cross") self.label = tk.Label(self, text="Thinking..", font=("Helvetica", 48))

self.classify\_btn = tk.Button(self, text = "Recognise", command = self.classify\_handwriting) self.button\_clear = tk.Button(self, text = "Clear", command = self.clear\_all)

# Grid structure

self.canvas.grid(row=0, column=0, pady=2, sticky=W, ) self.label.grid(row=0, column=1,pady=2, padx=2) self.classify\_btn.grid(row=1, column=1, pady=2, padx=2) self.button\_clear.grid(row=1, column=0, pady=2) #self.canvas.bind("<Motion>", self.start\_pos) self.canvas.bind("<B1-Motion>", self.draw\_lines) def clear\_all(self):

self.canvas.delete("all") def classify\_handwriting(self):

HWND = self.canvas.winfo\_id() # get the handle of the canvas rect = win32gui.GetWindowRect(HWND) # get the coordinate of the canvas im = ImageGrab.grab(rect) digit, acc = predict\_digit(im) self.label.configure(text= str(digit)+', '+ str(int(acc\*100))+'%') def draw\_lines(self, event):

self.x = event.x self.y = event.y

r=8 self.canvas.create\_oval(self.x-r, self.y-r, self.x + r, self.y + r, fill='black')

app = App() mainloop()

Output

