```
import pandas as pd
import streamlit as st
from streamlit_option_menu import option_menu
import easyocr
import mysql.connector as sql
from PIL import Image
import cv2
import os
import matplotlib.pyplot as plt
import re

# SETTING PAGE CONFIGURATIONS
icon = Image.open("icon.png")
st.set_page_config(page_title="BizCardX: Extracting Business Card Data with OCR | By
Arshad Ayub Ahmed",
                page_icon=icon,
                layout="wide",
                initial_sidebar_state="expanded",
                menu_items={'About': """# This OCR app is created by *Arshad Ayub
Ahmed*!"""})
st.markdown("<h1 style='text-align: center; color: white;'>BizCardX: Extracting Business
Card Data with OCR</h1>", unsafe_allow_html=True)

# SETTING-UP BACKGROUND IMAGE
def setting_bg():
   st.markdown(f"""
   <style>
      .stApp {{
         background: linear-gradient(to right, #2b5876, #4e4376);
         background-size: cover;
         transition: background 0.5s ease;
      }}
      h1,h2,h3,h4,h5,h6 {{
         color: #f3f3f3;
         font-family: 'Roboto', sans-serif;
      }}
      .stButton>button {{
         color: #4e4376;
         background-color: #f3f3f3;
         transition: all 0.3s ease-in-out;
      }}
      .stButton>button:hover {{
         color: #f3f3f3;
         background-color: #2b5876;
      }}
      .stTextInput>div>div>input {{
         color: #4e4376;
```

```
            background-color: #f3f3f3;
        }}
    </style>
    """,unsafe_allow_html=True)
setting_bg()

# CREATING OPTION MENU
selected = option_menu(None, ["Home","Upload & Extract","Modify"],
                        icons=["home","cloud-upload-alt","edit"],
                        default_index=0,
                        orientation="horizontal",
                        styles={"nav-link": {"font-size": "25px", "text-align": "centre", "margin": "0px",
"--hover-color": "#AB63FA", "transition": "color 0.3s ease, background-color 0.3s ease"},
                                "icon": {"font-size": "25px"},
                                "container" : {"max-width": "6000px", "padding": "10px", "border-radius":
"5px"},
                                "nav-link-selected": {"background-color": "#AB63FA", "color": "white"}})



# INITIALIZING THE EasyOCR READER
reader = easyocr.Reader(['en'])

# CONNECTING WITH MYSQL DATABASE
mydb = sql.connect(host="localhost",
                    user="root",
                    password="Ibro@4321",
                    database= "bizcardx_db"
                   )
mycursor = mydb.cursor(buffered=True)
#mycursor.execute("create database bizcardx_db")

# TABLE CREATION
mycursor.execute('''CREATE TABLE IF NOT EXISTS card_data
                    (id INTEGER PRIMARY KEY AUTO_INCREMENT,
                     company_name TEXT,
                     card_holder TEXT,
                     designation TEXT,
                     mobile_number VARCHAR(50),
                     email TEXT,
                     website TEXT,
                     area TEXT,
                     city TEXT,
                     state TEXT,
                     pin_code VARCHAR(10),
                     image LONGBLOB
                     )''')
```

```
# HOME MENU
if selected == "Home":
    col1,col2 = st.columns(2)
    with col1:
        st.markdown("## :green[**Technologies Used :**] Python,easy OCR, Streamlit, SQL,
Pandas")
        st.markdown("## :green[**Overview :**] In this streamlit web app you can upload an
image of a business card and extract relevant information from it using easyOCR. You can
view, modify or delete the extracted data in this app. This app would also allow users to save
the extracted information into a database along with the uploaded business card image. The
database would be able to store multiple entries, each with its own business card image and
extracted information.")
    with col2:
        st.image("home.png")


# UPLOAD AND EXTRACT MENU
if selected == "Upload & Extract":
    st.markdown("### Upload a Business Card")
    uploaded_card = st.file_uploader("upload
here",label_visibility="collapsed",type=["png","jpeg","jpg"])

    if uploaded_card is not None:

        def save_card(uploaded_card):
            with open(os.path.join("uploaded_cards",uploaded_card.name), "wb") as f:
                f.write(uploaded_card.getbuffer())
        save_card(uploaded_card)

        def image_preview(image,res):
            for (bbox, text, prob) in res:
              # unpack the bounding box
                (tl, tr, br, bl) = bbox
                tl = (int(tl[0]), int(tl[1]))
                tr = (int(tr[0]), int(tr[1]))
                br = (int(br[0]), int(br[1]))
                bl = (int(bl[0]), int(bl[1]))
                cv2.rectangle(image, tl, br, (0, 255, 0), 2)
                cv2.putText(image, text, (tl[0], tl[1] - 10),
                cv2.FONT_HERSHEY_SIMPLEX, 0.7, (255, 0, 0), 2)
            plt.rcParams['figure.figsize'] = (15,15)
            plt.axis('off')
            plt.imshow(image)

        # DISPLAYING THE UPLOADED CARD
        col1,col2 = st.columns(2,gap="large")
        with col1:
            st.markdown("#     ")
            st.markdown("#     ")
            st.markdown("### You have uploaded the card")
            st.image(uploaded_card)
        # DISPLAYING THE CARD WITH HIGHLIGHTS
        with col2:
            st.markdown("#     ")
            st.markdown("#     ")
            with st.spinner("Please wait processing image..."):
                st.set_option('deprecation.showPyplotGlobalUse', False)
                saved_img = os.getcwd()+ "\\" + "uploaded_cards"+ "\\"+ uploaded_card.name
                image = cv2.imread(saved_img)
                res = reader.readtext(saved_img)
                st.markdown("### Image Processed and Data Extracted")
                st.pyplot(image_preview(image,res))


        #easy OCR
        saved_img = os.getcwd()+ "\\" + "uploaded_cards"+ "\\"+ uploaded_card.name
        result = reader.readtext(saved_img,detail = 0,paragraph=False)

        # CONVERTING IMAGE TO BINARY TO UPLOAD TO SQL DATABASE
        def img_to_binary(file):
            # Convert image data to binary format
            with open(file, 'rb') as file:
                binaryData = file.read()
            return binaryData

        data = {"company_name" : [],
                "card_holder" : [],
                "designation" : [],
                "mobile_number" :[],
                "email" : [],
                "website" : [],
                "area" : [],
                "city" : [],
                "state" : [],
                "pin_code" : [],
                "image" : img_to_binary(saved_img)
               }

        def get_data(res):
            for ind,i in enumerate(res):

                # To get WEBSITE_URL
                if "www " in i.lower() or "www." in i.lower():
                    data["website"].append(i)
                elif "WWW" in i:
                    data["website"] = res[4] +"." + res[5]
```

```
# To get EMAIL ID
elif "@" in i:
    data["email"].append(i)

# To get MOBILE NUMBER
elif "-" in i:
    data["mobile_number"].append(i)
    if len(data["mobile_number"]) ==2:
        data["mobile_number"] = " & ".join(data["mobile_number"])

# To get COMPANY NAME
elif ind == len(res)-1:
    data["company_name"].append(i)

# To get CARD HOLDER NAME
elif ind == 0:
    data["card_holder"].append(i)

# To get DESIGNATION
elif ind == 1:
    data["designation"].append(i)

# To get AREA
if re.findall('^[0-9].+, [a-zA-Z]+',i):
    data["area"].append(i.split(',')[0])
elif re.findall('[0-9] [a-zA-Z]+',i):
    data["area"].append(i)

# To get CITY NAME
match1 = re.findall('.+St , ([a-zA-Z]+).+', i)
match2 = re.findall('.+St,, ([a-zA-Z]+).+', i)
match3 = re.findall('^[E].*',i)
if match1:
    data["city"].append(match1[0])
elif match2:
    data["city"].append(match2[0])
elif match3:
    data["city"].append(match3[0])

# To get STATE
state_match = re.findall('[a-zA-Z]{9} +[0-9]',i)
if state_match:
     data["state"].append(i[:9])
elif re.findall('^[0-9].+, ([a-zA-Z]+);',i):
    data["state"].append(i.split()[-1])
if len(data["state"])== 2:
    data["state"].pop(0)
```

```
            # To get PINCODE
            if len(i)>=6 and i.isdigit():
                data["pin_code"].append(i)
            elif re.findall('[a-zA-Z]{9} +[0-9]',i):
                data["pin_code"].append(i[10:])
        get_data(result)

        #FUNCTION TO CREATE DATAFRAME
        def create_df(data):
            df = pd.DataFrame(data)
            return df
        df = create_df(data)
        st.success("### Data Extracted!")
        st.write(df)

        if st.button("Upload to Database"):
            for i,row in df.iterrows():
                #here %S means string values
                sql = """INSERT INTO
card_data(company_name,card_holder,designation,mobile_number,email,website,area,city,s
tate,pin_code,image)
                         VALUES (%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)"""
                mycursor.execute(sql, tuple(row))
                # the connection is not auto committed by default, so we must commit to save our
changes
                mydb.commit()
            st.success("#### Uploaded to database successfully!")

# MODIFY MENU
if selected == "Modify":
    col1,col2,col3 = st.columns([3,3,2])
    col2.markdown("## Alter or Delete the data here")
    column1,column2 = st.columns(2,gap="large")
    try:
        with column1:
            mycursor.execute("SELECT card_holder FROM card_data")
            result = mycursor.fetchall()
            business_cards = {}
            for row in result:
                business_cards[row[0]] = row[0]
            selected_card = st.selectbox("Select a card holder name to update",
list(business_cards.keys()))
            st.markdown("#### Update or modify any data below")
            mycursor.execute("select
company_name,card_holder,designation,mobile_number,email,website,area,city,state,pin_c
ode from card_data WHERE card_holder=%s",
                          (selected_card,))
            result = mycursor.fetchone()

            # DISPLAYING ALL THE INFORMATIONS
            company_name = st.text_input("Company_Name", result[0])
            card_holder = st.text_input("Card_Holder", result[1])
            designation = st.text_input("Designation", result[2])
            mobile_number = st.text_input("Mobile_Number", result[3])
            email = st.text_input("Email", result[4])
            website = st.text_input("Website", result[5])
            area = st.text_input("Area", result[6])
            city = st.text_input("City", result[7])
            state = st.text_input("State", result[8])
            pin_code = st.text_input("Pin_Code", result[9])

            if st.button("Commit changes to DB"):
              # Update the information for the selected business card in the database
              mycursor.execute("""UPDATE card_data SET
company_name=%s,card_holder=%s,designation=%s,mobile_number=%s,email=%s,websit
e=%s,area=%s,city=%s,state=%s,pin_code=%s
                            WHERE card_holder=%s""",
(company_name,card_holder,designation,mobile_number,email,website,area,city,state,pin_c
ode,selected_card))
              mydb.commit()
              st.success("Information updated in database successfully.")

        with column2:
            mycursor.execute("SELECT card_holder FROM card_data")
            result = mycursor.fetchall()
            business_cards = {}
            for row in result:
              business_cards[row[0]] = row[0]
            selected_card = st.selectbox("Select a card holder name to Delete",
list(business_cards.keys()))
            st.write(f"### You have selected :green[**{selected_card}'s**] card to delete")
            st.write("#### Proceed to delete this card?")

            if st.button("Yes Delete Business Card"):
              mycursor.execute(f"DELETE FROM card_data WHERE
card_holder='{selected_card}'")
              mydb.commit()
              st.success("Business card information deleted from database.")
    except:
        st.warning("There is no data available in the database")

    if st.button("View updated data"):
        mycursor.execute("select
company_name,card_holder,designation,mobile_number,email,website,area,city,state,pin_c
ode from card_data")
```

```
        updated_df =
pd.DataFrame(mycursor.fetchall(),columns=["Company_Name","Card_Holder","Designation"
,"Mobile_Number","Email","Website","Area","City","State","Pin_Code"])
        st.write(updated_df)
```