**PROGRAM CODE**

**main.py**

import streamlit as st

import add\_donor

import retrieve\_donors

import retrieve\_recipients

import find\_donors

import add\_donations

import blood\_inventory

import make\_transfusion

import add\_recipient

def main():

st.title("Blood Bank Management System")

# Sidebar menu options

menu\_options = ["Add Donor", "Add Recipient", "Find Donor", "Retrieve All Donors", "Retrieve All Recipients", "Add Donation", "Make Transfusion", "Blood Inventory"]

selected\_option = st.sidebar.selectbox("Select an option", menu\_options)

# Execute respective functions based on the selected option

if selected\_option == "Add Donor":

add\_donor.add\_donor\_page()

elif selected\_option == "Find Donor":

find\_donors.find\_donors\_page()

elif selected\_option == "Retrieve All Donors":

retrieve\_donors.retrieve\_donors\_page()

elif selected\_option == "Retrieve All Recipients":

retrieve\_recipients.retrieve\_recipients\_page()

elif selected\_option == "Add Donation":

add\_donations.add\_donation\_page()

elif selected\_option == "Blood Inventory":

blood\_inventory.inventory\_page()

elif selected\_option == "Make Transfusion":

make\_transfusion.make\_transfusion\_page()

elif selected\_option == "Add Recipient":

add\_recipient.add\_recipient\_page()

if \_\_name\_\_ == "\_\_main\_\_":

main()

**make\_transfusion.py**

import streamlit as st

import mysql.connector

def make\_transfusion\_page():

st.subheader("Make Transfusion")

# Database connection

connection = mysql.connector.connect(

host="localhost",

user="root",

password="sidd",

database="blood\_bank\_management"

)

cursor = connection.cursor()

# Transfusion form

with st.form("make\_transfusion\_form"):

st.write("Enter transfusion details:")

recipient\_id = st.text\_input("Recipient ID")

transfusion\_date = st.date\_input("Transfusion Date")

# Retrieve the blood group of the recipient based on recipient\_id

blood\_group\_query = "SELECT blood\_group FROM recipients WHERE recipient\_id = %s"

cursor.execute(blood\_group\_query, (recipient\_id,))

recipient = cursor.fetchone()

blood\_group = recipient[0] if recipient else None

submit\_button = st.form\_submit\_button("Make Transfusion")

# Handle form submission

if submit\_button and blood\_group:

# Check if the required blood is available in the inventory

check\_inventory\_query = "SELECT quantity FROM blood\_inventory WHERE blood\_group = %s"

cursor.execute(check\_inventory\_query, (blood\_group,))

inventory = cursor.fetchone()

if inventory and inventory[0] > 0:

# Insert transfusion details into the database

insert\_query = "INSERT INTO transfusions (recipient\_id, blood\_group, transfusion\_date) VALUES (%s, %s, %s)"

values = (recipient\_id, blood\_group, transfusion\_date)

cursor.execute(insert\_query, values)

connection.commit()

st.success("Transfusion made successfully!")

else:

st.warning("Required blood not available in the inventory.")

st.info("Please redirect to 'Find Donors' to search for donors with the required blood group.")

elif submit\_button and not blood\_group:

st.warning("Recipient ID not found. Please enter a valid Recipient ID.")

# Close the database connection

cursor.close()

connection.close()

**blood\_inventory.py**

import streamlit as st

import mysql.connector

def add\_donation(donor\_id, blood\_group, donation\_date):

# Database connection

connection = mysql.connector.connect(

host="localhost",

user="root",

password="sidd",

database="blood\_bank\_management"

)

cursor = connection.cursor()

try:

# Insert the donation record into the donations table

insert\_query = "INSERT INTO donations (donor\_id, blood\_group, donation\_date) VALUES (%s, %s, %s)"

values = (donor\_id, blood\_group, donation\_date)

cursor.execute(insert\_query, values)

# Update the blood inventory

update\_query = "UPDATE blood\_inventory SET quantity = quantity + 1 WHERE blood\_group = %s"

cursor.execute(update\_query, (blood\_group,))

connection.commit()

st.success("Donation recorded successfully!")

except mysql.connector.Error as error:

st.error(f"Error occurred: {error}")

finally:

# Close the database connection

cursor.close()

connection.close()

def inventory\_page(filter\_blood\_group=None):

st.subheader("Blood Inventory")

# Database connection

connection = mysql.connector.connect(

host="localhost",

user="root",

password="sidd",

database="blood\_bank\_management"

)

cursor = connection.cursor()

# Retrieve blood inventory from the database

query = "SELECT blood\_group, quantity FROM blood\_inventory"

if filter\_blood\_group:

query += " WHERE blood\_group = %s"

cursor.execute(query, (filter\_blood\_group,))

else:

cursor.execute(query)

inventory = cursor.fetchall()

# Display the blood inventory in a table

if inventory:

st.table(inventory)

else:

st.info("No inventory found!")

# Close the database connection

cursor.close()

connection.close()

**find\_donor.py**

import streamlit as st

import mysql.connector

def find\_donors\_page():

st.subheader("Find Donors")

# Database connection

connection = mysql.connector.connect(

host="localhost",

user="root",

password="sidd",

database="blood\_bank\_management"

)

cursor = connection.cursor()

# Find donors form

with st.form("find\_donors\_form"):

st.write("Enter recipient requirements:")

blood\_group = st.selectbox("Required Blood Group", ["A+", "A-", "B+", "B-", "AB+", "AB-", "O+", "O-"])

# Add more recipient requirements as needed

submit\_button = st.form\_submit\_button("Find Donors")

# Handle form submission

if submit\_button:

# Prepare the SQL query and parameter values based on the selected filters

query = "SELECT \* FROM donors WHERE blood\_group = %s"

values = (blood\_group,)

# Retrieve donors matching the recipient requirements from the database

cursor.execute(query, values)

donors = cursor.fetchall()

# Display the inventory summary

inventory\_query = "SELECT blood\_group, SUM(quantity) FROM blood\_inventory GROUP BY blood\_group"

cursor.execute(inventory\_query)

inventory\_summary = cursor.fetchall()

if inventory\_summary:

st.subheader("Inventory Summary")

for group, quantity in inventory\_summary:

st.write(f"{group}: {quantity}")

st.write("---")

# Display the retrieved donors

if donors:

st.subheader("Donors")

for donor in donors:

st.container()

col1, col2 = st.columns(2)

col1.write(f"Name: {donor[1]}")

col1.write(f"Blood Group: {donor[2]}")

col1.write(f"Age: {donor[3]}")

col1.write(f"Gender: {donor[4]}")

col2.write(f"Contact Number: {donor[5]}")

col2.write(f"Email: {donor[6]}")

col2.write(f"Address: {donor[7]}")

col2.write(f"Last Donation Date: {donor[8]}")

st.write("---")

else:

st.info("No donors found!")

# Close the database connection

cursor.close()

connection.close()

**add\_donor.py**

import streamlit as st

import mysql.connector

from datetime import date, timedelta

def add\_donor\_page():

st.subheader("Add Donor")

# Database connection

connection = mysql.connector.connect(

host="localhost",

user="root",

password="sidd",

database="blood\_bank\_management"

)

cursor = connection.cursor()

# Add donor form

with st.form("add\_donor\_form"):

st.write("Enter donor details:")

name = st.text\_input("Name")

blood\_group = st.selectbox("Blood Group", ["A+", "A-", "B+", "B-", "AB+", "AB-", "O+", "O-"])

# Define the date range for date\_of\_birth validation, default date is 18 years ago

max\_date = date.today() - timedelta(days=365 \* 18) # 18 years ago

min\_date = date.today() - timedelta(days=365 \* 60) # 60 years ago

date\_of\_birth = st.date\_input("Date of Birth", value=max\_date, min\_value=min\_date, max\_value=max\_date)

gender = st.selectbox("Gender", ["Male", "Female", "Other"])

contact\_number = st.text\_input("Contact Number")

email = st.text\_input("Email")

address = st.text\_input("Address")

submit\_button = st.form\_submit\_button("Add Donor")

# Handle form submission

if submit\_button:

# Insert donor details into the database

insert\_query = "INSERT INTO donors (name, blood\_group, date\_of\_birth, gender, contact\_number, email, address) VALUES (%s, %s, %s, %s, %s, %s, %s)"

values = (name, blood\_group, date\_of\_birth, gender, contact\_number, email, address)

cursor.execute(insert\_query, values)

connection.commit()

# Retrieve the donor id of the donor that was just added

cursor.execute("SELECT LAST\_INSERT\_ID()")

st.success("Donor added successfully! Donor ID: " + str(cursor.fetchone()[0]))

# Close the database connection

cursor.close()

connection.close()

**retrieve\_donors.py**

import streamlit as st

import mysql.connector

def retrieve\_donors\_page():

st.subheader("Retrieve All Donors")

# Database connection

connection = mysql.connector.connect(

host="localhost",

user="root",

password="sidd",

database="blood\_bank\_management"

)

cursor = connection.cursor()

# Retrieve all donors from the database

query = "SELECT \* FROM donors"

cursor.execute(query)

donors = cursor.fetchall()

# Display the retrieved donors

if donors:

st.subheader("Donors")

for donor in donors:

st.container()

col1, col2 = st.columns(2)

col1.write(f"Name: {donor[1]}")

col1.write(f"Blood Group: {donor[2]}")

col1.write(f"Age: {donor[3]}")

col1.write(f"Gender: {donor[4]}")

col2.write(f"Contact Number: {donor[5]}")

col2.write(f"Email: {donor[6]}")

col2.write(f"Address: {donor[7]}")

col2.write(f"Last Donation Date: {donor[8]}")

st.write("---")

else:

st.info("No donors found!")

# Close the database connection

cursor.close()

connection.close()

**add\_donations.py**

import streamlit as st

import mysql.connector

def add\_donation\_page():

st.subheader("Add Donation")

# Database connection

connection = mysql.connector.connect(

host="localhost",

user="root",

password="sidd",

database="blood\_bank\_management"

)

cursor = connection.cursor()

# Add donation form

with st.form("add\_donation\_form"):

st.write("Enter donation details:")

donor\_id = st.text\_input("Donor ID")

donation\_date = st.date\_input("Donation Date")

submit\_button = st.form\_submit\_button("Add Donation")

# Retrieve the blood groups of the donor and recipient

blood\_group = get\_blood\_group(cursor, "donors", "donor\_id", donor\_id)

# Handle form submission

if submit\_button:

try:

# Insert the donation record into the donations table

insert\_query = "INSERT INTO donations (donor\_id, blood\_group, donation\_date) VALUES (%s, %s, %s)"

values = (donor\_id, blood\_group, donation\_date)

cursor.execute(insert\_query, values)

connection.commit()

st.success("Donation added successfully!")

except mysql.connector.Error as error:

st.error(f"Error occurred: {error}")

# Close the database connection

cursor.close()

connection.close()

def get\_blood\_group(cursor, table, id\_column, id\_value):

# Retrieve blood group based on the ID from the specified table

query = f"SELECT blood\_group FROM {table} WHERE {id\_column} = %s"

cursor.execute(query, (id\_value,))

result = cursor.fetchone()

if result:

return result[0]

else:

return None

**retrieve\_recipients.py**

import streamlit as st

import mysql.connector

def retrieve\_recipients\_page():

st.subheader("Retrieve All Recipients")

# Database connection

connection = mysql.connector.connect(

host="localhost",

user="root",

password="sidd",

database="blood\_bank\_management"

)

cursor = connection.cursor()

# Retrieve all recipients from the database

query = "SELECT \* FROM recipients"

cursor.execute(query)

recipients = cursor.fetchall()

# Display the retrieved recipients in a table

if recipients:

st.table(recipients)

else:

st.info("No recipients found!")

# Close the database connection

cursor.close()

connection.close()

**add\_recipients.py**

import streamlit as st

import mysql.connector

from datetime import date, timedelta

def add\_recipient\_page():

st.subheader("Add Recipient")

# Database connection

connection = mysql.connector.connect(

host="localhost",

user="root",

password="sidd",

database="blood\_bank\_management"

)

cursor = connection.cursor()

# Add recipient form

with st.form("add\_recipient\_form"):

st.write("Enter recipient details:")

name = st.text\_input("Name")

blood\_group = st.selectbox("Blood Group", ["A+", "A-", "B+", "B-", "AB+", "AB-", "O+", "O-"])

# Define the date range for date\_of\_birth validation, default date is 18 years ago

max\_date = date.today() - timedelta(days=365 \* 18) # 18 years ago

min\_date = date.today() - timedelta(days=365 \* 100) # 100 years ago

date\_of\_birth = st.date\_input("Date of Birth", value=max\_date, min\_value=min\_date, max\_value=max\_date)

gender = st.selectbox("Gender", ["Male", "Female", "Other"])

contact\_number = st.text\_input("Contact Number")

email = st.text\_input("Email")

address = st.text\_input("Address")

medical\_history = st.text\_area("Medical History")

submit\_button = st.form\_submit\_button("Add Recipient")

# Handle form submission

if submit\_button:

# Insert recipient details into the database

insert\_query = "INSERT INTO recipients (name, blood\_group, date\_of\_birth, gender, contact\_number, email, address, medical\_history) VALUES (%s, %s, %s, %s, %s, %s, %s, %s)"

values = (name, blood\_group, date\_of\_birth, gender, contact\_number, email, address, medical\_history)

cursor.execute(insert\_query, values)

connection.commit()

# Retrieve the recipient id of the recipient that was just added

cursor.execute("SELECT LAST\_INSERT\_ID()")

st.success("Recipient added successfully! Recipient ID: " + str(cursor.fetchone()[0]))

# Close the database connection

cursor.close()

connection.close()

**blood\_bank\_management.sql**

DROP DATABASE blood\_bank\_management;

CREATE DATABASE blood\_bank\_management;

USE blood\_bank\_management;

CREATE TABLE donors (

donor\_id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(50) NOT NULL,

blood\_group VARCHAR(5) NOT NULL,

date\_of\_birth DATE,

gender VARCHAR(10),

contact\_number VARCHAR(15),

email VARCHAR(50),

address VARCHAR(100),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP

);

CREATE TABLE recipients (

recipient\_id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(50) NOT NULL,

blood\_group VARCHAR(5) NOT NULL,

date\_of\_birth DATE,

gender VARCHAR(10),

contact\_number VARCHAR(15),

email VARCHAR(50),

address VARCHAR(100),

medical\_history TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP

);

CREATE TABLE blood\_inventory (

blood\_id INT AUTO\_INCREMENT PRIMARY KEY,

blood\_group VARCHAR(5) NOT NULL,

quantity INT NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP

);

CREATE TABLE donations (

donation\_id INT AUTO\_INCREMENT PRIMARY KEY,

donor\_id INT,

blood\_group VARCHAR(5),

donation\_date DATE,

FOREIGN KEY (donor\_id) REFERENCES donors(donor\_id),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP

);

CREATE TABLE transfusions (

transfusion\_id INT AUTO\_INCREMENT PRIMARY KEY,

recipient\_id INT,

blood\_group VARCHAR(5),

transfusion\_date DATE,

FOREIGN KEY (recipient\_id) REFERENCES recipients(recipient\_id),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP

);

-- Trigger for updating blood inventory after a new donation

DELIMITER //

CREATE TRIGGER after\_donation\_insert

AFTER INSERT ON donations

FOR EACH ROW

BEGIN

-- Update the blood\_inventory table by increasing the quantity for the corresponding blood group

UPDATE blood\_inventory

SET quantity = quantity + 1

WHERE blood\_group = NEW.blood\_group;

END//

DELIMITER ;

-- Trigger for updating blood inventory after a new transfusion

DELIMITER //

CREATE TRIGGER after\_transfusion\_insert

AFTER INSERT ON transfusions

FOR EACH ROW

BEGIN

-- Update the blood\_inventory table by decreasing the quantity for the corresponding blood group

UPDATE blood\_inventory

SET quantity = quantity - 1

WHERE blood\_group = NEW.blood\_group;

END//

DELIMITER ;

USE blood\_bank\_management;

INSERT INTO blood\_inventory (blood\_group, quantity)

VALUES ('A+', 0),

('A-', 0),

('B+', 0),

('B-', 0),

('AB+', 0),

('AB-', 0),

('O+', 0),

('O-', 0);