# Project Development - Delivery Of Sprint-

| | |
|---|---|
| Team ID | PNT2022TMID52198 |
| Project name | Plasma donor app |
| Team member | Mohamed Ali<br>Mohamed Mahroof<br>Pradeesh<br>Revison<br>Mohamed Abdul Kader Riyaz |

FLASK MAIL IS USING HERE BECAUSE SENDGRID IS NOT WORKING*:*

```
from flask import Flask, render_template, request, redirect, url_for, session
from flask_mail import Mail, Message
app.config['MAIL_SERVER']='smtp.gmail.com'
app.config['MAIL_PORT'] = 465
app.config['MAIL_USERNAME'] = ‘example@gmail.com'
app.config['MAIL_PASSWORD'] = '**********'
app.config['MAIL_USE_TLS'] = False
app.config['MAIL_USE_SSL'] = True
mail = Mail(app)
def index(usermail,subject,content):

  msg = Message(subject, sender = 'example@gmail.com', recipients = [usermail])
  msg.body = format(content)
  mail.send(msg)
  return "Sent"
```

Dockerfile:

```
FROM python:3.9
WORKDIR /app
ADD . /app
```

# Project Development - Delivery Of Sprint-

```
COPY requirements.txt /app

RUN python3 -m pip install -r requirements.txt

EXPOSE 5000

CMD ["python","app.py"]
```

app.py:

```
from distutils.log import debug

# from sendgridmail import sendmail

from flask import Flask, render_template, request, redirect, url_for, session

from flask_mail import Mail, Message

import re

import os

import ibm_db

from dotenv import load_dotenv


load_dotenv()



app = Flask(__name__)


app.secret_key =  'a'

print("Try to connect to Db2")

conn=ibm_db.connect("DATABASE=bludb;HOSTNAME=*****;PORT=
*****;UID=*****;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;PWD=******", '','')

print("Connected Successfully")


app.config['MAIL_SERVER']='smtp.gmail.com'

app.config['MAIL_PORT'] = 465

app.config['MAIL_USERNAME'] = 'example@gmail.com'
```

# Project Development - Delivery Of Sprint-

```
app.config['MAIL_PASSWORD'] = '*****'
app.config['MAIL_USE_TLS'] = False
app.config['MAIL_USE_SSL'] = True
mail = Mail(app)

@app.route('/')
@app.route('/login')
def login():

    return render_template('login.html')



@app.route('/loginpage',methods=['GET', 'POST'])
def loginpage():
    global userid
    msg = ''

    if request.method == 'POST' :
        username = request.form['username']
        password = request.form['password']
        sql = "SELECT * FROM donors WHERE username =? AND password=?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.bind_param(stmt,2,password)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print (account)
        if account:
            session['loggedin'] = True
```

# Project Development - Delivery Of Sprint-

```
            session['id'] = account['USERNAME']
            userid=  account['USERNAME']
            session['username'] = account['USERNAME']
            msg = 'Logged in successfully !'
            index(account['EMAIL'],'Plasma donor App login','You are successfully logged in!')
            return redirect(url_for('dash'))
        else:
            msg = 'Incorrect username / password !'
    return render_template('login.html', msg = msg)


@app.route('/registration')
def home():
    return render_template('register.html')


@app.route('/register',methods=['GET', 'POST'])
def register():
    msg = ''
    if request.method == 'POST' :
        username = request.form['username']
        email = request.form['email']
        password = request.form['password']
        phone = request.form['phone']
        city = request.form['city']
        infect = request.form['infect']
        blood = request.form['blood']
        sql = "SELECT * FROM donors WHERE username =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.execute(stmt)
```

# Project Development - Delivery Of Sprint-

```
    account = ibm_db.fetch_assoc(stmt)
    print("ac",account)
    if account:
        msg = 'Account already exists !'
    elif not re.match(r'[^@]+@[^@]+\.[^@]+', email):
        msg = 'Invalid email address !'
    elif not re.match(r'[A-Za-z0-9]+', username):
        msg = 'name must contain only characters and numbers !'
    else:
        insert_sql = "INSERT INTO  donors VALUES (?, ?, ?, ?, ?, ?, ?)"
        prep_stmt = ibm_db.prepare(conn, insert_sql)
        ibm_db.bind_param(prep_stmt, 1, username)
        ibm_db.bind_param(prep_stmt, 2, password)
        ibm_db.bind_param(prep_stmt, 3, email)
        ibm_db.bind_param(prep_stmt, 4, phone)
        ibm_db.bind_param(prep_stmt, 5, city)
        ibm_db.bind_param(prep_stmt, 6, infect)
        ibm_db.bind_param(prep_stmt, 7, blood)

        ibm_db.execute(prep_stmt)
        msg = 'You have successfully registered, !'
        index(email,'Plasma donor App Registration','You are successfully Registered
{}!'.format(username))

  elif request.method == 'POST':
    msg = 'Please fill out the form !'
  return render_template('register.html', msg = msg)

@app.route('/dashboard')
```

# Project Development - Delivery Of Sprint-

```
def dash():
    if session['loggedin'] == True:
        sql = "SELECT COUNT(*), (SELECT COUNT(*) FROM DONORS WHERE blood= 'O Positive'), (SELECT COUNT(*) FROM DONORS WHERE blood='A Positive'), (SELECT COUNT(*) FROM DONORS WHERE blood='B Positive'), (SELECT COUNT(*) FROM DONORS WHERE blood='AB Positive'), (SELECT COUNT(*) FROM DONORS WHERE blood='O Negative'), (SELECT COUNT(*) FROM DONORS WHERE blood='A Negative'), (SELECT COUNT(*) FROM DONORS WHERE blood='B Negative'), (SELECT COUNT(*) FROM DONORS WHERE blood='AB Negative') FROM donors"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print(account)
        return  render_template('dashboard.html',b=account)
    else:
        msg = 'Please login!'
        return render_template('login.html', msg = msg)


@app.route('/requester')
def requester():
    if session['loggedin'] == True:
        return render_template('request.html')
    else:
        msg = 'Please login!'
        return render_template('login.html', msg = msg)


@app.route('/requested',methods=['POST'])
def requested():
    bloodgrp = request.form['bloodgrp']
    address = request.form['address']
    name=  request.form['name']
```

```
    email= request.form['email']
    phone= request.form['phone']
    insert_sql = "INSERT INTO  requested VALUES (?, ?, ?, ?, ?)"
    prep_stmt = ibm_db.prepare(conn, insert_sql)
    ibm_db.bind_param(prep_stmt, 1, bloodgrp)
    ibm_db.bind_param(prep_stmt, 2, address)
    ibm_db.bind_param(prep_stmt, 3, name)
    ibm_db.bind_param(prep_stmt, 4, email)
    ibm_db.bind_param(prep_stmt, 5, phone)
    ibm_db.execute(prep_stmt)
    index(email,'Plasma donor App plasma request','Your request for plasma is recieved.')
    return render_template('request.html', pred="Your request is sent to the concerned people.")


def index(usermail,subject,content):

  msg = Message(subject, sender = 'example@gmail.com', recipients = [usermail])
  msg.body = format(content)
  mail.send(msg)
  return "Sent"

@app.route('/logout')
def logout():
  session.pop('loggedin', None)
  session.pop('id', None)
  session.pop('username', None)
  return render_template('login.html')

if __name__== '__main__':
```

# Project Development - Delivery Of Sprint-

```
app.run(host='0.0.0.0',debug='TRUE')
```

**output :**

Mail will be send to the concerned people i.e To the donor who registered with the plasma needed by the recipient using flask mail