

```

from flask import Blueprint, jsonify, request

from backend import conn, config

import bcrypt

import jwt

import ibm_db

auth = Blueprint("auth", __name__)

LOGIN_FIELDS = ('email', 'password')

SIGNUP_FIELDS = ('name', 'email', 'phone_number', 'password')

@auth.route("/login", methods=['POST'])

def login_user():

    # Check if all the required feild are present

    for feild in LOGIN_FIELDS:

        if not (feild in request.json):

            return jsonify({"error": f"All feilds are required!"}), 409

    email = request.json['email']

    password = request.json['password']

    sql = f"select * from users where email='{email}'"

    stmt = ibm_db.prepare(conn, sql)

    ibm_db.execute(stmt)

    user = ibm_db.fetch_assoc(stmt)

    if not user:

        return jsonify({"error": "Invalid credentials!"}), 401

    if bcrypt.checkpw(password.encode('utf-8'),

        user["PASSWORD"].encode('utf-8')):

        token = jwt.encode(

            {"email": email},

            config["APP_SECRET"],

            algorithm="HS256"

```

```

    )

    return jsonify({"name": user["NAME"], "email": email, "phone_number":
user["PHONE_NUMBER"], "token": token}), 200

else:

    return jsonify({"error": "Invalid credentials!"}), 401

@auth.route("/signup", methods=['POST'])

def register_user():

    # Check if all the required feild are present

    for feild in SIGNUP_FEILDS:

        if not (feild in request.json):

            return jsonify({"error": f"All feilds are required!"}), 409

    email = request.json['email']

    phone_number = request.json['phone_number']

    name = request.json['name']

    password = request.json['password']

    # Sql stmt to check if email/number is already in use

    sql = f"select * from users where email='{email}' or phone_number='{phone_number}'"

    stmt = ibm_db.prepare(conn, sql)

    ibm_db.execute(stmt)

    user = ibm_db.fetch_assoc(stmt)

    if user:

        return jsonify({"error": f"Email/Phone number is alread in use!"}), 409

    # If user does not exist, then create account

    hashed_password = bcrypt.hashpw(

        password.encode('utf-8'), bcrypt.gensalt())

    sql = f"insert into users(name,email,phone_number,password)
values('{name}','{email}','{phone_number}',?)"

    stmt = ibm_db.prepare(conn, sql)

    ibm_db.bind_param(stmt, 1, hashed_password)

```

```
ibm_db.execute(stmt)

token = jwt.encode(
    {"email": email},
    config["APP_SECRET"],
    algorithm="HS256"
)

return jsonify({"name": name, "email": email, "phone_number": phone_number, "token": token}),
200
```