***router.js***

const express = require("express");

const router = express.Router();

const { signup, login, logout } = require("../controllers/userControllers");

const {

  validateUserData,

  isAuthenticated,

} = require("../middlewares/userMiddlewares");

router.post("/signup", validateUserData, signup);

router.post("/login", isAuthenticated, login);

router.post("/logout", isAuthenticated, logout);

module.exports = router;

***middleware.js***

const pool = require("../config/db.js");

const validateUserData = (req, res, next) => {

  const {

    first\_name,

    last\_name,

    mobile\_number,

    college\_email,

    is\_alumini,

    email,

    password,

  } = req.body;

  if (

    !first\_name ||

    !last\_name ||

    !mobile\_number ||

    !college\_email ||

    !email ||

    !password

  ) {

    return res

      .status(400)

      .json({ error: true, message: "All fields are required!" });

  }

  if (

    validateEmail(email) &&

    validateMobileNumber(mobile\_number) &&

    validatePassword(password)

  ) {

    next();

  } else {

    res.status(400).json({ error: true, message: "Invalid Data!" });

  }

};

const isAuthenticated = async (req, res, next) => {

  try {

    const authHeader = req.headers.authorization;

    if (!authHeader) {

      return res.status(401).json({

        error: true,

        message: "You must be logged in to perform this action.",

      });

    }

    const token = authHeader.replace("Bearer ", "");

    const tokenQueryParams = [token];

    const tokenQuery = `SELECT \* FROM user\_token WHERE token = $1`;

    const tokenQueryData = await pool.query(tokenQuery, tokenQueryParams);

    if (tokenQueryData.rowCount < 1) {

      return res.status(401).json({ error: true, message: "Invalid Token!" });

    }

    const userId = tokenQueryData.rows[0].fk\_user;

    const userQuery = `SELECT id FROM users WHERE id = $1`;

    const userQueryParams = [userId];

    const userQueryData = await pool.query(userQuery, userQueryParams);

    req.user = userQueryData.rows[0];

    req.token = token;

    next();

  } catch (err) {

    console.log(err);

    res.status(500).json({ error: true, message: "Internal Server Error!" });

  }

};

module.exports = {

  validateUserData,

  isAuthenticated,

};

***controller.js***

require("dotenv").config();

const bcrypt = require("bcrypt");

const pool = require("../config/db.js");

const generateUserToken = require("../utils/generateUserToken");

const signup = async (req, res) => {

  var {

    first\_name,

    last\_name,

    mobile\_number,

    college\_email,

    is\_alumini,

    email,

    password,

  } = req.body;

  try {

    const timestamp = new Date();

    const hashedPassword = await bcrypt.hash(

      password,

      Number(process.env.SALT)

    );

    if (!is\_alumini) {

      is\_alumini = false;

    }

    const userQuery = `INSERT INTO users (first\_name, last\_name, mobile\_number, college\_email, is\_alumini, email, password, created\_at) VALUES ($1,$2,$3,$4,$5,$6,$7,$8) RETURNING \*`;

    const userQueryParams = [

      first\_name,

      last\_name,

      mobile\_number,

      college\_email,

      is\_alumini,

      email,

      hashedPassword,

      timestamp,

    ];

    const userQueryData = await pool.query(userQuery, userQueryParams);

    const token = await generateUserToken(userQueryData.rows[0].id);

    delete userQueryData.password;

    res.status(201).json({

      error: false,

      message: "Signup Successful. Welcome aboard!",

      data: {

        token,

        user: userQueryData.rows[0],

      },

    });

  } catch (err) {

    if (err.code === "23505") {

      res.status(400).json({

        error: true,

        message:

          "User with this details already exists! Please use a different email or phone number to create your account.",

      });

    } else {

      console.log(err);

      res.status(500).json({ error: true, message: "Internal Server Error!" });

    }

  }

};