**Practical 20**

**AIM:**

**Create CRUD Operation page using ExpressJs, MongoDB and**

**CODE:**

**Address.js**

const mongoose = require("mongoose");

const addressSchema = new mongoose.Schema({

  street: String,

  city: String,

  zipCode: String,

});

const Address = mongoose.model("Address", addressSchema);

module.exports = Address;

**Profile.js**

const mongoose = require("mongoose");

const profileSchema = new mongoose.Schema({

  fullName: String,

  age: Number,

});

const Profile = mongoose.model("Profile", profileSchema);

module.exports = Profile;

**User.js**

const mongoose = require("mongoose");

const userSchema = new mongoose.Schema({

  username: String,

});

const User = mongoose.model("User", userSchema);

module.exports = User;

**dbConfig.js**

require("dotenv").config();

const mongoose = require("mongoose");

const connectDB = async () => {

  try {

    await mongoose.connect(process.env.MONGO\_URL, {

      useNewUrlParser: true,

      useUnifiedTopology: true,

    });

    console.log("Connected successfully to MongoDB");

  } catch (error) {

    console.error("Error connecting to MongoDB:", error);

  }

};

module.exports = connectDB;

**Server.js**

require("dotenv").config();

const express = require("express");

const connectDB = require("./dbConfig");

const User = require("./model/User");

const Profile = require("./model/Profile");

const Address = require("./model/Address");

const app = express();

const PORT = process.env.PORT || 3000;

connectDB(); // Connect to MongoDB

app.use(express.json());

// Create a new user with profile and address

app.post("/users", async (req, res) => {

  try {

    const { username, profile, address } = req.body;

    const newUser = new User({ username });

    const newProfile = new Profile(profile);

    const newAddress = new Address(address);

    newUser.profile = newProfile;

    newUser.address = newAddress;

    await newUser.save();

    res.json(newUser);

  } catch (error) {

    console.error(error);

    res.status(500).json({ error: "Unable to create user." });

  }

});

// Get all users with their profiles and addresses

app.get("/users", async (req, res) => {

  try {

    const users = await User.find().populate("profile").populate("address");

    res.json(users);

  } catch (error) {

    console.error(error);

    res.status(500).json({ error: "Unable to fetch users." });

  }

});

// Get a specific user by ID with their profile and address

app.get("/users/:id", async (req, res) => {

  try {

    const userId = req.params.id;

    const user = await User.findById(userId)

      .populate("profile")

      .populate("address");

    if (!user) {

      return res.status(404).json({ error: "User not found." });

    }

    res.json(user);

  } catch (error) {

    console.error(error);

    res.status(500).json({ error: "Unable to fetch user." });

  }

});

// Update user information by ID

app.put("/users/:id", async (req, res) => {

  try {

    const userId = req.params.id;

    const { username, profile, address } = req.body;

    const user = await User.findById(userId);

    if (!user) {

      return res.status(404).json({ error: "User not found." });

    }

    // Update user fields

    user.username = username;

    await user.save();

    // Update associated profile and address

    if (user.profile) {

      user.profile.fullName = profile.fullName;

      user.profile.age = profile.age;

      await user.profile.save();

    }

    if (user.address) {

      user.address.street = address.street;

      user.address.city = address.city;

      user.address.zipCode = address.zipCode;

      await user.address.save();

    }

    res.json(user);

  } catch (error) {

    console.error(error);

    res.status(500).json({ error: "Unable to update user." });

  }

});

app.listen(PORT, () => {

  console.log(`Server is running on port ${PORT}`);

});

**OUTPUT:**

