**Module 3 Activity 1: Database Access**

Sai Kishore Reddy Konatham

IFT 544: Middleware Prog & Database Sec

Dinesh Sthapit

17 Sep 2023

**GITHUB Link:** <https://github.com/saikishore632/554_M3A1>

**Part A: installing required libraries:**

Command: npm install express dotenv mssql morgan cors --save

**A screenshot of a computer

Description automatically generated**

**Installing sql server extension:**

**A screenshot of a computer

Description automatically generated**

**Database connection:**

• Server: sqlservercentralpublic.database.windows.net

• Database: AdventureWorks

• User: sqlfamily

• Password: sqlf@m1ly

**A screenshot of a computer

Description automatically generated**

**Table view:** views section is empty.

**SQL SELECT Statements:**

1. **Customer:** SELECT \* FROM SalesLT.Customer

**A screenshot of a computer

Description automatically generated**

1. **Product:** SELECT \* FROM SalesLT.Product

**A screenshot of a computer

Description automatically generated**

1. **SalesOrderDetails:** SELECT \* FROM SalesLT. SalesOrderDetails

**A screenshot of a computer

Description automatically generated**

1. **CustomerAddress:** SELECT \* FROM SalesLT. CustomerAddress

**A screenshot of a computer

Description automatically generated**

**Join Table:**

Query to get Customer address details: Using customer, customerAddress, Address tables.

SELECT c.CustomerID,

CONCAT(c.FirstName,' ',c.MiddleName,' ',c.LastName) as name,

c.EmailAddress as email,

CONCAT(a.AddressLine1,' ', a.AddressLine2,' ',a.City,' ',a.StateProvince,' ',a.CountryRegion,' ',a.PostalCode) as address

FROM SalesLT.Customer as c

INNER JOIN SalesLT.CustomerAddress as ca

ON c.CustomerID = ca.CustomerID

INNER JOIN SalesLT.Address as a

ON a.AddressID = ca.AddressID

**A screenshot of a computer

Description automatically generated**

**Implementation Node.js CRUD with SQL Server:**

**DB Manager changes:**

const sql = require('mssql');

const dbConnection = require('./db.config');

console.log(dbConnection);

async function getCustomers(){

console.log(' Connecting to SQL....... Cloud Server');

let dbContext = await sql.connect(dbConnection);

console.log('The Databse connection was Successful');

console.log('Getting data');

let results = await dbContext.request()

.query(`

SELECT \* From

salesLT.Customer

`);

console.log(`Returned SQL results`);

return results;

}

async function saveCustomer(customerData) {

try {

console.log('Connecting to SQL....... Cloud Server');

let dbContext = await sql.connect(dbConnection);

console.log('The Database connection was Successful');

console.log('Inserting customer data');

let request = dbContext.request()

// Assuming customerData is an object with properties like FirstName, LastName, etc.

request.input('Title', sql.NVarChar, customerData.Title);

request.input('FirstName', sql.NVarChar, customerData.FirstName);

request.input('MiddleName', sql.NVarChar, customerData.MiddleName);

request.input('LastName', sql.NVarChar, customerData.LastName);

// Add more input parameters as needed.

let query = `

INSERT INTO salesLT.Customer (Title, FirstName, MiddleName,LastName)

VALUES (@Title,@FirstName,@MiddleName,@LastName);

`;

await request.query(query);

console.log('Customer data saved successfully');

return request; // Or you can return any specific success indicator you prefer.

} catch (error) {

console.error('Error saving customer:', error);

throw error; // Rethrow the error for handling further up the call stack.

}

}

//Export

module.exports = {getCustomers : getCustomers,saveCustomer:saveCustomer};

**Customer controller:**

const db = require('../config/db.manager');

exports.getAllCustomers = function (req, res) {

const customers = db.getCustomers().then(results=>{

console.log(results);

res.status(200).json({

status: 'successfull',

data: results.recordsets[0]

}); // send all the data

});

}

exports.getCustomerByID = function( req, res){

const {id} = req.params; // get id

res.status(200).json({

status: 'no implemented'

});

}

exports.createNewCustomer = async function( req, res){ // must select the body to beraw->JSON in Postman

try{

const saved = await db.saveCustomer(req.body);

if (saved) {

res.status(200).json({

status: 'successful',

message: 'Customer data saved successfully',});

} else {

res.status(500).json({

status: 'error',

message: 'Failed to save customer data',

});

}

}

catch(error){

res.status(500).json({

status: 'error',

message: error.message,

});

}

}

exports.patchCustomerById = function( req, res){ // must select the body to beraw->JSON in Postman

const {body} = req;// req.body (attribute)

const {id} = req.params;// get id (attribute)

res.status(200).json({

status: 'no implemented'

});

}

exports.deleteCustomerByID = function( req, res){ // must select the body to be raw->JSON in Postman

const {body} = req;// req.body (attribute)

const {id} = req.params;// get id

res.status(200).json({

status: 'no implemented'

});

}

**Customer Router:**

const express = require('express');

const router = express.Router();

const customerController = require('../controller/customerController');

router

.route('/')

.get(customerController.getAllCustomers)

.post(customerController.createNewCustomer);

router

.route('/:id')

.get(customerController.getCustomerByID)

.patch(customerController.patchCustomerById)

.delete(customerController.deleteCustomerByID);

module.exports = router;

**OUTPUT:**

**GET Method:**

GET:http://localhost:3000/api/v1/customer

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**