Access Local instance od DynamoDB using the Node.js app

For the please install aws-sdk using node package manager

1. Please installa the local DynamoDB instance

You can connect to local instance of DynamoDb using the following command

```
2. java -Djava.library.path=./DynamoDBLocal_lib -jar DynamoDBLocal.jar -
   sharedDb
```

Configure the DynamoDB using the following command

```
 aws configure
AWS Access Key ID [****************eyId]:
AWS Secret Access Key [****************sKey]:
Default region name [None]: local
Default output format [None]: json
```

Here you need to set access Key and Id

To create a table you need to use the git shell and run the following command

```
aws dynamodb create-table  --table-name Music  --attribute-definitions
AttributeName=CategoryName,AttributeType=S
AttributeName=ProductId,AttributeType=S  --key-schema
AttributeName=CategoryName,KeyType=HASH AttributeName=ProductId,KeyType=RANGE
--provisioned-throughput ReadCapacityUnits=1,WriteCapacityUnits=1
```

Query can be fired using the following command

```
aws dynamodb put-item \
--table-name Product  \
--item \
    '{"CategoryName": {"S": "Electronics"}, "ProductId": {"S": "Prd0001"},
"ProductName": {"S": "Laptop"}, "Price":{"N":120000},"Manufacturer":{"S":"MS-
UT"}}' \
--return-consumed-capacity TOTAL
```

Query to read data from tables

```
aws dynamodb scan --table-name Product


aws dynamodb query --table-name Product --key-condition-expression
"CategoryName = :name"  --expression-attribute-values
'{\":name\":{\"S\":\"Electronics\"}}'
```

In the project add a new file, name it as 'callaws.js' and add the following code in it. This code will be used to connect to DynamoDB instance locally and create a table in it

```
import aws from 'aws-sdk';



aws.config.update({

    region: 'local',

    endpoint: 'http://localhost:8000'

});



let dynamoDb = new aws.DynamoDB();



var params = {

    TableName: "Product",

    KeySchema: [

        {

            AttributeName: "CategoryName",

            KeyType: "HASH", //Partition key
```

```
        },
        {
            AttributeName: "ProductId",
            KeyType: "RANGE" //Sort key
        }
    ],
    AttributeDefinitions: [
        {
            AttributeName: "CategoryName",
            AttributeType: "S"
        },
        {
            AttributeName: "ProductId",
            AttributeType: "S"
        }
    ],
    ProvisionedThroughput: {       // Only specified if using provisioned
mode
        ReadCapacityUnits: 5,
        WriteCapacityUnits: 5
    }
```

```
};



dynamoDb.createTable(params, function (err, data) {
    if (err) {
        console.error("Unable to create table. Error JSON:",
JSON.stringify(err, null, 2));
    } else {
        console.log("Created table. Table description JSON:",
JSON.stringify(data, null, 2));
    }
});
```

Run the application, this will show table create.

You can use the following command to list table

```
aws dynamodb list-tables --endpoint-url http://localhost:8000
```

Add a new file of name putitem.js and add the following code in it

```
import aws from 'aws-sdk';

import fs from 'fs';

aws.config.update({
    region: 'local',
    endpoint: 'http://localhost:8000'
});
```

```
// 1. doc client

let documentClient = new aws.DynamoDB.DocumentClient();

console.log('Please wait, I am importing Product Data from JSOn file ');

let productsData = [

    {

        "CategoryName": "Electrical",

        "Manufacturer": "MS-Power",

        "Price": 2000,

        "ProductId": "Prd0001",

        "ProductName": "Iron"

    },

    {

        "CategoryName": "Electrical",

        "Manufacturer": "TS-Power",

        "Price": 2400,

        "ProductId": "Prd0002",

        "ProductName": "Mixer"

    },

    {

        "CategoryName": "Electronics",
```

```
        "Manufacturer": "MSIT",
        "Price": 98000,
        "ProductId": "Prd0003",
        "ProductName": "Laptop"
    },
    {
        "CategoryName": "Electronics",
        "Manufacturer": "TSIT",
        "Price": 20000,
        "ProductId": "Prd0004",
        "ProductName": "Desktop"
    },
    {
        "CategoryName": "Food",
        "Manufacturer": "MS-Food",
        "Price": 20,
        "ProductId": "Prd0005",
        "ProductName": "Lays"
    },
    {
        "CategoryName": "Food",
```

```
        "Manufacturer": "TS-Food",
        "Price": 300,
        "ProductId": "Prd0006",
        "ProductName": "Chips"
    }
];
//let products = JSON.parse(fs.readFileSync(__dirname +
'../../data/data.json', 'utf-8'));
//let products = JSON.parse(productsData);

productsData.forEach((prd) => {
    console.log(JSON.stringify(prd));

    let params = {
        TableName: "Product",
        Item: {
            "ProductId": prd.ProductId,
            "ProductName": prd.ProductName,
            "Price": prd.Price,
            "CategoryName": prd.CategoryName,
            "Manufacturer": prd.Manufacturer
```

```
        }

    };

    documentClient.put(params, (err, data) => {

        if (err) {

            console.error("Unable to add Product", prd.ProductId, ". Error
JSON:", JSON.stringify(err, null, 2));

        } else {

            console.log("PutItem succeeded:", prd.ProductId);

        }

    });

});
```

The above file will add bulk records in the collection.

In the application create a new folder of name awsrest. In this folder add a new file, name it as dal.js and add the following code in it

```
import aws from 'aws-sdk';
aws.config.update({
    region: 'local',
    endpoint: 'http://localhost:8000'
});

class AWSRest {
    constructor() {
        this.docClient = new aws.DynamoDB.DocumentClient();
    }

    getData(req, resp) {
        let params = {
            TableName: "Product",
```

```
            ProjectionExpression: "#ProductId, #ProductName, #Price,
#CategoryName, #Manufacturer",
            ExpressionAttributeNames: {
                "#ProductId": "ProductId",
                "#ProductName": "ProductName",
                "#Price": "Price",
                "#CategoryName": "CategoryName",
                "#Manufacturer": "Manufacturer"
            }
        };

        console.log('Scanning data from Product Table');
        this.docClient.scan(params, (err, data) => {
            if (err) {
                console.error("Unable to scan the table. Error JSON:",
JSON.stringify(err, null, 2));
            } else {
                resp.send(data);
                console.log("Scan succeeded.");
                data.Items.forEach(function (prd) {
                    console.log(`${prd.ProductId} ${prd.ProductName}
${prd.Price} ${prd.CategoryName} ${prd.Manufacturer}`);
                });
            }
        });
    }

    getSingleData(req, resp) {
        console.log(`Received Product Id ${req.params.id}`);
        let params = {
            TableName: "Product",
            ProjectionExpression: "#ProductId, #ProductName, #Price,
#CategoryName, #Manufacturer",
            FilterExpression: "#ProductId = :id",
            ExpressionAttributeNames: {
                "#ProductId": "ProductId",
                "#ProductName": "ProductName",
                "#Price": "Price",
                "#CategoryName": "CategoryName",
                "#Manufacturer": "Manufacturer"
            },
            ExpressionAttributeValues: {
                ":id": req.params.id
            }
        };

        console.log('Scanning data from Product Table');
        this.docClient.scan(params, (err, data) => {
            if (err) {
                console.error("Unable to scan the table. Error JSON:",
JSON.stringify(err, null, 2));
            } else {
                resp.send(data);
                console.log("Scan succeeded.");
                data.Items.forEach(function (prd) {
                    console.log(`${prd.ProductId} ${prd.ProductName}
${prd.Price} ${prd.CategoryName} ${prd.Manufacturer}`);
```

```
                });
            }
        });
    }

    postData(req, resp) {
        let prd = {
            ProductId: req.body.ProductId,
            ProductName: req.body.ProductName,
            Price: req.body.Price,
            CategoryName: req.body.CategoryName,
            Manufacturer: req.body.Manufacturer
        };
        let params = {
            TableName: "Product",
            Item: {
                "ProductId": prd.ProductId,
                "ProductName": prd.ProductName,
                "Price": prd.Price,
                "CategoryName": prd.CategoryName,
                "Manufacturer": prd.Manufacturer
            }
        };
        this.docClient.put(params, (err, data) => {
            if (err) {
                console.error("Unable to add Product", prd.ProductId, ".
Error JSON:", JSON.stringify(err, null, 2));
            } else {
                resp.send({ status: 200, message: `Item Added
${prd.ProductId}` });
                console.log("PutItem succeeded:", prd.ProductId);
            }
        });
    }

    putData(req, resp) { }

    deleteData(req, resp) { }
}

module.exports = AWSRest;
```

The above file contains code to connect to DynamoDB and perform Read and Write operation.

In the folder add a new file, name it as api.js and add the following code in it

```
import express from 'express';
import bodyParser from 'body-parser';
import cors from 'cors';
let dal = require('./dal');

let objDal = new dal();

let instance = express();
```

```
instance.use(bodyParser.json());
instance.use(bodyParser.urlencoded({ extended: false }));
instance.use(cors());

instance.get('/api/products', (req, resp) => {
    objDal.getData(req, resp);
});

instance.get('/api/products/:id', (req, resp) => {
    objDal.getSingleData(req, resp);
});

instance.post('/api/products', (req, resp) => {
    objDal.postData(req, resp);
});

instance.put('/api/products', (req, resp) => {
    objDal.putData(req, resp);
});

instance.delete('/api/products', (req, resp) => {
    objDal.deleteData(req, resp);
});

instance.listen(8090, () => {
    console.log('Started Reading on port 8090');
});
```

This is a REST API to access DAL and hence DynamoDB.

Run the application and test API.