```
const express = require("express")
const path = require("path")
const fetch = require("node-fetch")
const app = express()
const AWS = require("aws-sdk");
const port = 8080

const AWS_BUCKET = "csu44000assign2useast20";
const AWS_FILENAME = "moviedata.json";

AWS.config.update({
    region: "us-east-1"
});

var dynamodb = new AWS.DynamoDB();
var docClient = new AWS.DynamoDB.DocumentClient();

let publicPath = path.resolve(__dirname, "public")
app.use(express.static(publicPath))
app.get("/", function (req, res) {res.sendFile(path.join(__dirname + "/client.html"))})

app.listen(port, function () {
    console.log("App is listening on port " +port)
})

app.post('/create', (req, res) => {
    var params = {
        TableName: "Movies",
        KeySchema: [
            { AttributeName: "year", KeyType: "HASH" },  //Partition key
            { AttributeName: "title", KeyType: "RANGE" }  //Sort key
        ],
        AttributeDefinitions: [
            { AttributeName: "year", AttributeType: "N" },
            { AttributeName: "title", AttributeType: "S" }
        ],
        ProvisionedThroughput: {
            ReadCapacityUnits: 1,
            WriteCapacityUnits: 1
        }
    };
    dynamodb.createTable(params, function (err, data) {
        if (err) {
            console.error("Error creating table :", JSON.stringify(err, null, 2));
        } else {
            console.log("Created table successfully :", JSON.stringify(data, null, 2));
        }
    });
    var s3params = {
        Bucket: AWS_BUCKET,
        Key: AWS_FILENAME
    }
    var s3 = new AWS.S3();
    s3.getObject(s3params, function (err, data) {
        if (err) {
            console.log(err, err.stack);
        } else {
            var allMovies = JSON.parse(data.Body.toString());
            allMovies.forEach(function (movie) {
                var params = {
                    TableName: "Movies",
                    Item: {
                        "year": movie.year,
                        "title": movie.title,
                        "director":  movie.info.directors,
                        "rating": movie.info.rating,
                        "rank": movie.info.rank,
                        "release": movie.info.release_date
                    }
                };

                docClient.put(params, function (err, data) {
                    if (err) {
                        console.error("Error adding movie to database :", movie.title, ". Error JSON:"
, JSON.stringify(err, null, 2));
                    } else {
                        console.log("Added movie successfully :", movie.title);
                    }
                });
            });
        }
        console.log("Database created successfully");
    })
});


app.post('/query/:title/:year', (req, res) => {
    var myArray = {
        myList :[]
    }
    var year = parseInt(req.params.year)
    var title = req.params.title
    var params = {
        TableName : "Movies",
        ProjectionExpression:"#yr, title, director, rating, #r, #re",
        KeyConditionExpression: "#yr = :yyyy and begins_with (title, :letter1)",
        ExpressionAttributeNames:{
            "#yr": "year",
            "#r":"rank",
            "#re":"release"
        },
        ExpressionAttributeValues: {
            ":yyyy": year,
            ":letter1": title
        }
    };

    docClient.query(params, function(err, data) {
        if (err) {
            console.log("Error querying database:", JSON.stringify(err, null, 2));
        } else {
            data.Items.forEach(function(item) {
                console.log(item.year +' '+ item.
title+'' + item.director+'' + item.rating);
                var yearPush = item.year
                var titlePush = item.title
                var directorPush = item.director
                var ratingPush = item.rating
                var rankPush = item.rank
                var releasePush = item.release
                myArray.myList.push(
                    {
                        Title: titlePush,
                        Year : yearPush,
                        Director: directorPush,
                        Rating: ratingPush,
                        Rank: rankPush,
                        Release: releasePush
                    }
                )
            });
            console.log("Query executed successfully.");
            res.json(myArray)
        }
    });
});



app.post('/delete', (req, res) => {
    console.log("Deleting Database");
    var params = {TableName : "Movies"};
    dynamodb.deleteTable(params, function(err, data) {
        if (err) {
            console.error("Error deleting table :", JSON.stringify(err, null, 2));
        } else {
            console.log("Deleted table successfully :", JSON.stringify(data, null, 2));
        }
    });
});
```