

```

1 // CSU44000 Internet Applications Assignment 2
2 // Simple Movie DB App
3 // Pulkit Sharma
4
5
6 const express = require("express")
7 const path = require("path")
8 const app = express()
9 const PORT = 8008
10 const AWS = require("aws-sdk");
11 const AWS_ACCESS_KEY=process.env.AWS_ACCESS_KEY
12 const AWS_SECRET_KEY=process.env.AWS_SECRET_KEY
13
14 //Get the AWS credentials
15 AWS.config.update({
16     region: 'us-east-1',
17     accessKeyId: AWS_ACCESS_KEY,
18     secretAccessKey: AWS_SECRET_KEY
19 });
20
21 //Check if we got the credentials
22 AWS.config.getCredentials(function(err) {
23     if (err) console.log(err.stack);
24
25     else {
26         console.log("Access key:", AWS.config.credentials.accessKeyId);
27         console.log("Secret access key:", AWS.config.credentials.secretAccessKey);
28         console.log("Region: ", AWS.config.region);
29     }
30 });
31
32 //load the s3 movie bucket
33 var s3params = {
34     Bucket: 'csu44000assign2useast20',
35     Key: 'moviedata.json'
36 };
37
38 var dynamodb = new AWS.DynamoDB();
39 var docClient = new AWS.DynamoDB.DocumentClient();
40 var s3 = new AWS.S3();
41
42 //Load the HTML page
43 app.get("/", function (req, res) {
44     res.sendFile(path.join(__dirname + "/index.html"))
45 });
46
47 app.listen(PORT, function () {
48     console.log("AWS Movie DB running on Port: " + PORT )
49 });
50
51 //Create Database and table
52 app.post('/create', (req, res) => {
53     console.log("Creating Movie Database")
54     var params = {
55         TableName: "Movies",
56         KeySchema: [
57             { AttributeName: "year", KeyType: "HASH" },
58             { AttributeName: "title", KeyType: "RANGE" }
59         ],
60         AttributeDefinitions: [

```

```

61         { AttributeName: "year", AttributeType: "N" },
62         { AttributeName: "title", AttributeType: "S" }
63     ],
64 };
65
66     dynamodb.createTable(params, function (err, data) {
67         if (err) {
68             console.error("Unable to create table. Error JSON:", JSON.stringify(err,
69 null, 2));
70         }
71         else {
72             console.log("Created table success. Table description JSON:",
73 JSON.stringify(data, null, 2));
74         }
75     });
76
77     //Parse the movie info
78     s3.getObject(s3params, function (err, data) {
79         if (err) {
80             console.log(err, err.stack);
81         } else {
82             var allMovies = JSON.parse(data.Body.toString());
83             allMovies.forEach(function (movie) {
84                 var params = {
85                     TableName: "Movies",
86                     Item: {
87                         "year": movie.year,
88                         "title": movie.title,
89                         "rating": movie.info.rating,
90                         "rank": movie.info.rank,
91                     }
92                 };
93
94                 docClient.put(params, function (err, data) {
95                     if (err) {
96                         console.error("Unable to add movie", movie.title, ". Error
97 JSON:", JSON.stringify(err, null, 2));
98                     }
99                     else {
100                         console.log("Adding movie :", movie.title);
101                     }
102                 });
103             });
104             console.log("Database created and populated");
105         }
106     });
107
108     //Query the Database
109     app.post('/query/:title/:year', (req, res) => {
110         console.log("Querying ..")
111         var myArray = {
112             dataEntry :[]
113         }
114         var year = parseInt(req.params.year)
115         var title = req.params.title
116         var params = {
117             TableName : "Movies",

```

```

118 ProjectionExpression: "#yr, title, rating, #r, #re",
119 KeyConditionExpression: "#yr = :yyyy and begins_with (title, :letter1)",
120 ExpressionAttributeNames: {
121     "#yr": "year",
122     "#r": "rank",
123     "#re": "release"
124 },
125 ExpressionAttributeValues: {
126     ":yyyy": year,
127     ":letter1": title
128 }
129 };
130
131 //Check if query is successfull
132 docClient.query(params, function(err, data) {
133     if (err) {
134         console.log("Unable to query. Error:", JSON.stringify(err, null, 2));
135     }
136
137     else {
138         console.log("Query succeeded.");
139         data.Items.forEach(function(item) {
140
141             console.log(item.title + ' : ' + item.year + ' : ' + item.rating);
142             var movieTitle = item.title
143             var movieYear = item.year
144             var movieRating = item.rating
145             var movieRank = item.rank
146
147             myArray.dataEntry.push(
148                 {
149                     Title: movieTitle,
150                     Year : movieYear,
151                     Rating: movieRating,
152                     Rank: movieRank,
153                 }
154             )
155         });
156         res.json(myArray)
157     }
158 });
159 });
160
161 //Delete the Database
162 app.post('/destroy', (req, res) => {
163     console.log("Destroying the Database");
164     var params = { TableName : "Movies", };
165
166     dynamodb.deleteTable(params, function(err, data) {
167         if (err) {
168             console.error("Unable to delete table. Error JSON:", JSON.stringify(err,
169 null, 2));
170         }
171
172         else {
173             console.log("Deleted table. Table description JSON:",
174 JSON.stringify(data, null, 2));
175         }
176     });
177 });

```