

```
1  const dotenv = require("dotenv");
2  const express = require('express');
3  const AWS = require("aws-sdk");
4
5  // configure .env secrets
6  dotenv.config();
7
8
9  // constants
10 const AWS_ACCESS_KEY=process.env.AWS_ACCESS_KEY
11 const AWS_SECRET_KEY=process.env.AWS_SECRET_KEY
12 const PORT = process.env.PORT
13 const BUCKET="csu44000assignment220"
14 const FILE_LOCATION = "moviedata.json"
15 const TABLE_PARAMS = {
16     TableName : "Movies",
17     KeySchema: [
18         { AttributeName: "year", KeyType: "HASH"}, //Partition key
19         { AttributeName: "title", KeyType: "RANGE" } //Sort key
20     ],
21     AttributeDefinitions: [
22         { AttributeName: "year", AttributeType: "N" },
23         { AttributeName: "title", AttributeType: "S" }
24     ],
25     ProvisionedThroughput: {
26         ReadCapacityUnits: 1,
27         WriteCapacityUnits: 1
28     }
29 };
30
31 const BUCKET_PARAMS = {
32     Bucket: BUCKET,
33     Key: FILE_LOCATION,
34 }
35
36 const app = express();
37 app.use(express.static('public')); // serves html page
38
39
40 // database config
41 AWS.config.update({
42     region: "us-east-1",
43     accessKeyId: AWS_ACCESS_KEY,
44     secretAccessKey: AWS_SECRET_KEY
45 })
46
47
48 let dynamodb = new AWS.DynamoDB();
49 let s3 = new AWS.S3();
50
51
52 // routes
53 app.get('/api/create-database', async function (req, res) {
54
55     s3.getObject(BUCKET_PARAMS, function(err,data){
56         if(err){
57             console.log('error:',err)
58             return res.status(400).json(err);
59         }
60     })
61 }
```

```

61     let allMovies = JSON.parse(data.Body)
62
63     dynamodb.createTable(TABLE_PARAMS, async function(err, data) {
64         if (err) {
65             console.error("Unable to create table. Error JSON:",
JSON.stringify(err, null, 2));
66             return res.status(400).json(err)
67         } else {
68             console.log("Created table. Table description JSON:",
JSON.stringify(data, null, 2));
69             await sleep(5000); // sleep so table has time to create
70             var docClient = new AWS.DynamoDB.DocumentClient();
71             allMovies.forEach(function (movie) {
72
73                 var params = {
74                     TableName: "Movies",
75                     Item: {
76                         "year": movie.year,
77                         "title": movie.title,
78                         "release_date": movie.info.release_date,
79                         "rank": movie.info.rank
80                     }
81                 };
82
83                 docClient.put(params, function(err, data) {
84                     if (err) {
85                         console.error("Unable to add movie", movie.title,
". Error JSON:", JSON.stringify(err, null, 2));
86                     } else {
87                         console.log("PutItem succeeded:", movie.title);
88                     }
89                 });
90             });
91             return res.status(200).send("Created table and populated.")
92         }
93     });
94 })
95
96 })
97
98 app.get('/api/get-movies', async function (req, res) {
99     const {title, year} = req.query
100
101     if(!title || !year){
102         res.status(400).send('Please provide title AND year');
103     }
104
105     if(!dynamodb){
106         res.status(400).send('Unable to query as table does not exist');
107     }
108
109     var docClient = new AWS.DynamoDB.DocumentClient();
110
111     var params = {
112         TableName : "Movies",
113         KeyConditionExpression: "#yr = :yyyy and begins_with(title, :t)",
114         ExpressionAttributeNames:{
115             "#yr": "year",
116         },
117         ExpressionAttributeValues: {

```

```
118         ":yyyy": parseInt(year),
119         ":t": title
120     }
121 };
122
123 docClient.query(params, function(err, data) {
124     if (err) {
125         console.log(err)
126         return res.status(400).json(err);
127     } else {
128         console.log("Query succeeded.");
129         var results = []
130         data.Items.forEach(function(item) {
131             console.log(item)
132             results.push({
133                 "title": item.title,
134                 "year": item.year,
135                 "release_date": item.release_date,
136                 "rank": item.rank,
137             })
138         });
139         return res.status(200).json(results);
140     }
141 });
142 })
143
144 app.get('/api/delete-database', async function (req, res) {
145     var params = {
146         TableName : "Movies"
147     };
148
149     await dynamodb.deleteTable(params, function(err, data) {
150         if (err) {
151             console.error("Unable to delete table. Error JSON:",
152 JSON.stringify(err, null, 2));
153             return res.status(400).json(err)
154         } else {
155             return res.status(200).send('Table Deleted');
156         }
157     });
158 })
159
160 // helpers
161 function sleep(ms) {
162     return new Promise((resolve) => {
163         setTimeout(resolve, ms);
164     });
165 }
166
167
168 // launch server
169 app.listen(PORT, function () {
170     console.log(`Server is running on ${PORT} port`);
171 });
```