```
/*-------------------- JAVASCIPT CODE FOR SERVER-----------------------------------------------*/

/*-------------------------CONSTANTS AND SETUP/CONFIGURATION---------------------------------*/
//requirements
const express = require('express');
const AWS = require("aws-sdk");
//aws credential constants-must be deleted before submission and upload to github
const ACCESS_KEY= ''
const SECRET_KEY = ''
//location of bucket movie data
const PORT = 3000
const BUCKET_NAME = "csu44000assignment220"
const FILE_NAME="moviedata.json"


/*---------------------code copied from aws tutorial ------------------------------------
//(https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/GettingStarted.NodeJs.03.html)
*/
//set up parameters of table and bucket
const TABLE_PARAMS={
    TableName :"Movies",
    KeySchema:[
        {AttributeName: "year", KeyType: "HASH"},
        {AttributeName: "title", KeyType: "RANGE"}
    ],
    AttributeDefinitions:[
        {AttributeName: "year", AttributeType: "N"},
        {AttributeName: "title", AttributeType: "S"}
    ],
    ProvisionedThroughput:{
        ReadCapacityUnits: 1,
        WriteCapacityUnits: 1
    }
};
const BUCKET_PARAMS={
    Bucket: BUCKET_NAME,
    Key: FILE_NAME,
}
//database configuration using credentials
AWS.config.update({
    region: "us-east-1",
    accessKeyId: ACCESS_KEY,
    secretAccessKey: SECRET_KEY
})
//create instances of dynamodb and s3 objects
let dynamoDB = new AWS.DynamoDB(); //dynamoDB
let s3 = new AWS.S3();


const app = express(); //build server using express
app.use(express.static('public')); //serve client html page from public folder

/*----------Functions to handle button clicks ------------*/

//function to create table - function creates table using dynamoDB and populates it using subset of data from bucket
app.get('/api/createTable', async function (req,res){

    s3.getObject(BUCKET_PARAMS, function (err, data){  //get data from bucket
        if(err){return res.status(400).json(err);} //return error
        let movieObjects =JSON.parse(data.Body) //parse data

        dynamoDB.createTable(TABLE_PARAMS, async function (err, data){//create table and populate with data from bucket
            if(err){return res.status(400).json(err)}
            else{
                await pause(5000); //allowing time for initial load of DB
                var documentClient = new AWS.DynamoDB.DocumentClient();
                movieObjects.forEach(function (movie){ //add eahc movie object to
```

```
                    new table
68                      var params={
69                          TableName: "Movies",
70                          Item: {
71                              "year": movie.year,
72                              "title": movie.title,
73                              "release_date": movie.info.release_date,
74                              "rank": movie.info.rank
75                          }
76                      };
77                      documentClient.put(params, function(err, data){ //put movie
                        object into table
78                          if(err)console.error("Error with adding movie object:",
                            movie.title, JSON.stringify(err, null, 2));
79                          else{console.log("Movie object added: ", movie.title);}
80                      });
81                  });
82                  console.log("Finished populating table")
83                  return res.status(200).send("Created table and populated")
84              }
85          });
86      })
87  })
88
89  //function to delete table
90  app.get('/api/deleteTable', async function (req, res){
91      var params = {
92          TableName: "Movies"
93      };
94      await dynamoDB.deleteTable(params, function(err, data){ //delete table
95          if(err){return res.status(400).json(err)}
96          else{return res.status(200).send('Table deleted');}
97      });
98  })
99
100 //funtion to query table - take user input and search table for matching data
101 app.get('/api/getMovies', async function (req, res){
102     const {title, year} = req.query
103     if(!title||!year){res.status(400).send('Please provide title and year');} //user
        didn't provide title and year
104     if(!dynamoDB){res.status(400).send("Table doesn't exist");} //havent created
        table yet - nothing to query
105
106     var documentClient = new AWS.DynamoDB.DocumentClient();
107     var params={
108         TableName: "Movies",
109         KeyConditionExpression: "#yr = :yyyy and begins_with(title, :t)", //sourced
            from aws tutorial
110         ExpressionAttributeNames:{
111             "#yr":"year",
112         },
113         ExpressionAttributeValues:{
114             ":yyyy": parseInt(year),
115             ":t": title
116         }
117     };
118     //perform query on table
119     documentClient.query(params, function(err, data){
120         if(err){
121             console.log(err)
122             return res.status(400).json(err);
123         }else{
124             console.log("query succesful");
125             var results = []
126             data.Items.forEach(function(item){ //add return interms to results array
127                 console.log(item)
128                 results.push({
129                     "title": item.title,
130                     "year": item.year,
131                     "release_date": item.release_date,
132                     "rank": item.rank,
133                 })
134             });
```

```
            return res.status(200).json(results);
        }
    });
})

//function to pause program for 5000ms to allow inital load of db
function pause(ms){
    return new Promise((resolve) => {
        setTimeout(resolve,ms);
    });
}

//run server on port 3000
app.listen(PORT, function (){
    console.log(`Server is running on port ${PORT}`);
});
```