Simple REST App using Express

Step 1: Install Express, Body-parser and Cors in the Project using following command

Npm install - -save-dev express body-parser cors

Step 1: Create a file of name restapi.js and add the following code in it

```
class RESTApi {

    constructor() {

        this.Employees = [{

            EmpNo: 1,

            EmpName: 'A'

        }, {

            EmpNo: 2,

            EmpName: 'B'

        }];

       console.log('In ctor ' + this.Employees.length);

    }

    get(req, res) {

        console.log('In Get' + res + '   ' + this);

        console.log(this.Employees.length);

        console.log(JSON.stringify(this.Employees));

        res.send(JSON.stringify(this.Employees));
```

```
    }

    post(req, res) {

        console.log('In Post');

        let emp = {

            EmpNo: req.body.EmpNo,

            EmpName: req.body.EmpName

        };

        this.Employees.push(emp);

        res.send(JSON.stringify(this.Employees));

    }

}

module.exports = RESTApi;
```

Step 2: In the project add a new file of name server.js and add the following code in it

```
import express from "express";

import bodyParser from "body-parser";

let api = require('./restapi');

let instance = express();

let apiObj = new api();

instance.use(bodyParser.json());
```

```
instance.use(bodyParser.urlencoded({
    extended: false
}));
instance.get('/api/employees', (req, resp) => {
    apiObj.get(req, resp);
});
instance.post('/api/employees', (req, resp) => {
    apiObj.post(req, resp);
});
instance.listen(3002, () => {
    console.log('Server started on port 3002');
});
console.log('service started...');
```

Run the application and browse the REST API using following Url

http://localhost:3002/api/employees

Please test GET and POST requests

Working with MySql using Node.js

Step 1: Create database in MySql of name company. Install following packages in the code

Sequelize, sequelize-auto, sequelize-cli, mysql, mysql2

Step 2: In the project add a new file of name connectdb.js and add the following code in it

```
let sequelize = require('sequelize');

// the object accepts, database name, username and password

let db = new sequelize("company", "root", "P@ssw0rd_", {

    host: 'localhost',

    dialect: 'mysql', // this is for mysql

    pool: {

        max: 5,

        min: 0,

        idle: 10000

    }

});

// define model

var Person = db.define('person', {

    firstName: {

        type: sequelize.STRING,

        field: 'firstName'

    },
```

```
    lastName: {
        type: sequelize.STRING,
        field: 'firstName'
    },
    age: {
        type: sequelize.INTEGER,
        field: 'age'
    }
}, {
    freezeTableName: true // Model Table name will be same
});


Person.sync({
    force: true
}).then(() => {
    return Person.create({
        firstName: 'Mahesh',
        lastName: 'Sabnis',
        age: 43
    });
});
```

Run the node application this will create person table in it

Exercise 3: Creating Models from the existing database

Step 1: In the database add following tables

```
CREATE TABLE `department` (
  `DeptId` int(11) NOT NULL,
  `DeptNo` varchar(20) NOT NULL,
  `DeptName` varchar(45) NOT NULL,
  `Location` varchar(45) NOT NULL,
  `Capacity` int(11) NOT NULL,
  PRIMARY KEY (`DeptId`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `employee` (
  `EmpId` int(11) NOT NULL,
  `EmpNo` varchar(45) NOT NULL,
  `EmpName` varchar(45) NOT NULL,
  `Salary` int(11) NOT NULL,
  `DeptId` int(11) NOT NULL,
  PRIMARY KEY (`EmpId`),
  KEY `DeptId_idx` (`DeptId`),
```

CONSTRAINT `DeptId` FOREIGN KEY (`DeptId`) REFERENCES `department` (`DeptId`) ON DELETE CASCADE ON UPDATE CASCADE

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

Step 2: Run the following command from the command prompt

```
sequelize-auto -h localhost -d company -u root -x P@ssw0rd_ --dialect
```

```
mysql -o models -t department,employee
```

This will create department.js and employee.js in the models folder.

Step 3: In the project add the new file of name connectpredefinedtables.js and add the following code in it

```
let OrmMySql = require('sequelize');

// mysql://username:password@host:port/database

const dbConnect = new OrmMySql("mysql://root:P@ssw0rd_@localhost/company",

{

    define: {

        timestamps: false // true by default. false because bydefault

sequelize adds createdAt, modifiedAt columns with timestamps.if you want

those columns make ths true.

    }

});

var employees = dbConnect.import('./../../models/employee.js');
```

```
/*make sure you use false here. otherwise the total data
  from the imported models will get deleted and new tables will be
created*/
dbConnect.sync({
    force: false
})
    .then(() => {
        employees.findAll().then(employees => {
            console.log(`data is ${JSON.stringify(employees)}`);
        });
        employees.create({
            EmpId: 1008,
            EmpNo: 'Emp-1008',
            EmpName: 'Anil',
            Salary: 30000,
            DeptId: 30
        }).then((d) => {
            console.log('Created Successfully');
            employees.findAll().then(employees => {
                console.log(`data is ${JSON.stringify(employees)}`);
            });
```

```
        });
        console.log('sync is complete');
    });
```

Run this file, you will see the records will be displayed in the console.

Performing Update and Delete operations

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

```
let orm = require('sequelize');
let ormInstance = new orm("mysql://root:P@ssw0rd_@localhost/company", {
    define: {
        timestamps: false // true by default. false because bydefault
sequelize adds createdAt, modifiedAt columns with timestamps.if you want
those columns make ths true.
    }
});

var employees = ormInstance.import('./../../models/employee.js');
// find records based on condition
ormInstance.sync({
    force: false
```

```
}).then(() => {
    employees.findAll({
        where: {
            DeptId: 10
        }
    }).then(emps => {
        console.log(`Employees ${JSON.stringify(emps)}`);
    });
});
// find records based on condition and update
// ormInstance.sync({
//     force: false
// }).then(() => {
//     employees.update({
//         Salary: 200000
//     }, {
//             where: {
//                 DeptId: 10
//             }
//         }).then(emps => {
//             console.log(`Updated Employees ${JSON.stringify(emps)}`);
```

```
//         });
// });


// find records based on condition and delete
ormInstance.sync({
    force: false
}).then(() => {
    employees.destroy({
        where: {
            DeptId: 30
        }
    }).then((emps) => {
        console.log(` Employees ${JSON.stringify(emps)}`);
    });
});


// using row queries for update
ormInstance.query("Update employee SET Salary=Salary*0.5 WHERE DeptId=10")
    .spread((result, metadata) => {
        console.log(`Result ${result}`);
        console.log(`Metadata foe number of records effected ${metadata}`);
```

```
});
```

The above code will update an delete record from tables