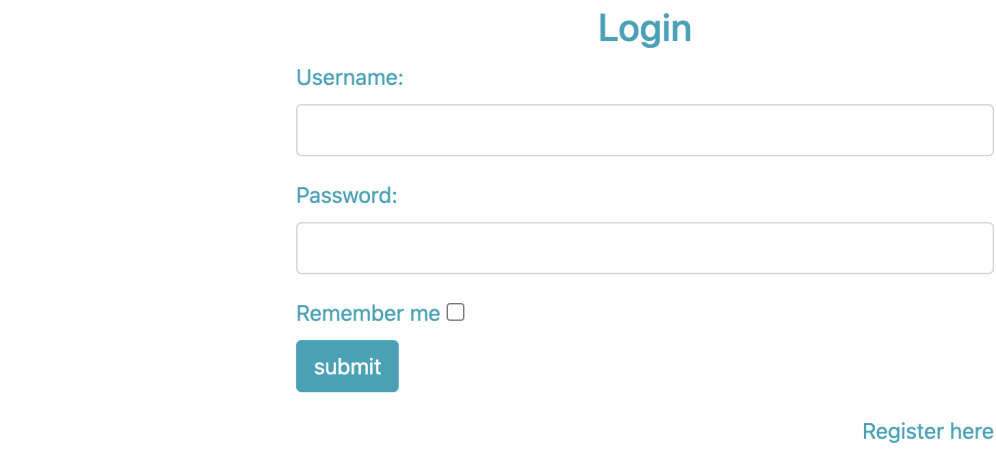
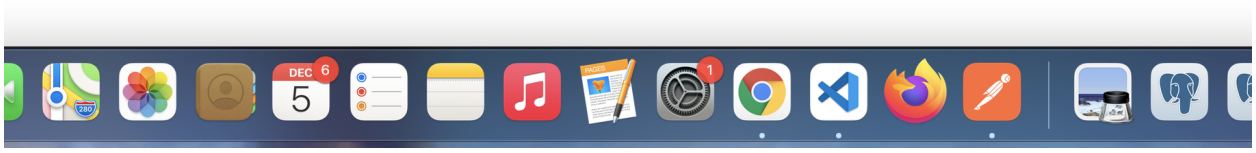
1. Write the Login.html which takes 2 inputs as user id and password. At the express side, validate it against your name for user id
and password as admin. If this criteria matches then return as valid user else return as invalid user.





Not valid crediantial please enter valid uid and password

2. Write a express script to take the URL as /getAllEmployeeData and when the browser gives the URL

http://localhost:8000/getAllEmployeeData

Array of Employee objects should be returned from express and it should be displayed on browser.

Employee object should have the fields : Id, name, dept and designation.

```
        min:dbConfig.pool.min,
        max:dbConfig.pool.max,
        acquire:dbConfig.pool.acquire,
        idle:dbConfig.pool.idle

    }
});

let EmployeeTable = sequelize.define('Employees',{
    emp_id:{
        primaryKey:true,
        type:Sequelize.INTEGER
    },
    name:Sequelize.STRING,
    dept:Sequelize.STRING,
    designation:Sequelize.STRING

},{
    timestamps:false,
    freezeTableName:true
});
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL                    node - postgre

  { emp_id: 103, name: 'sanket', dept: 'it', designation: 'ASE' },
  { emp_id: 104, name: 'suraj', dept: 'cs', designation: 'ASE' },
  { emp_id: 105, name: 'nrk', dept: 'cs', designation: 'FSASE' },
  { emp_id: 106, name: 'nrk', dept: 'cs', designation: 'FSASE' },
  { emp_id: 108, name: 'abcd', dept: 'CS', designation: 'ASE' }
]
```

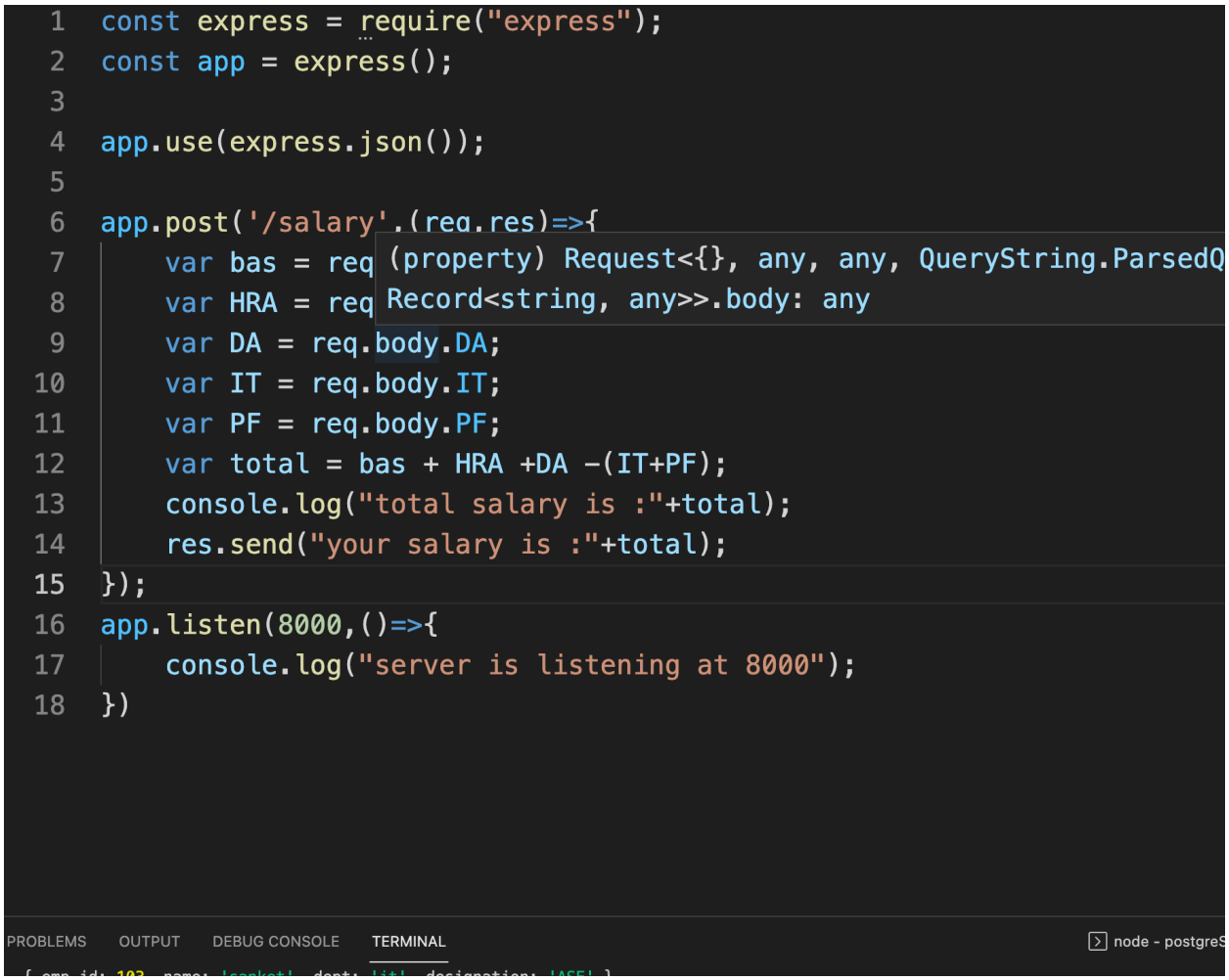
```
43  //its get the all the in the table
44  app.get('/getAllEmployees',(req,res)=>{
45      EmployeeTable.findAll({raw:true}).then(data=>{
46          console.log(data);
47          res.status(200).send(data);
48
49      }).catch(err=>{
50          console.error("error is :"+err);
51          res.status(400).send(err);
52      })
53  })
54
55  //show data by id
56  app.get('/getAllEmployeeById/:id',(req,res)=>{
57      var id = req.params.id;
58      console.log("given id is :"+id);
59
60      EmployeeTable.findByPk(id,{raw:true}).then(data=>{
61          console.log(data);
62          res.status(200).send(data);
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL                    node - postgre

  { emp_id: 103, name: 'sanket', dept: 'it', designation: 'ASE' },
  { emp_id: 104, name: 'suraj', dept: 'cs', designation: 'ASE' },
  { emp_id: 105, name: 'nrk', dept: 'cs', designation: 'FSASE' },
  { emp_id: 106, name: 'nrk', dept: 'cs', designation: 'FSASE' },
  { emp_id: 108, name: 'abcd', dept: 'CS', designation: 'ASE' }
]
```

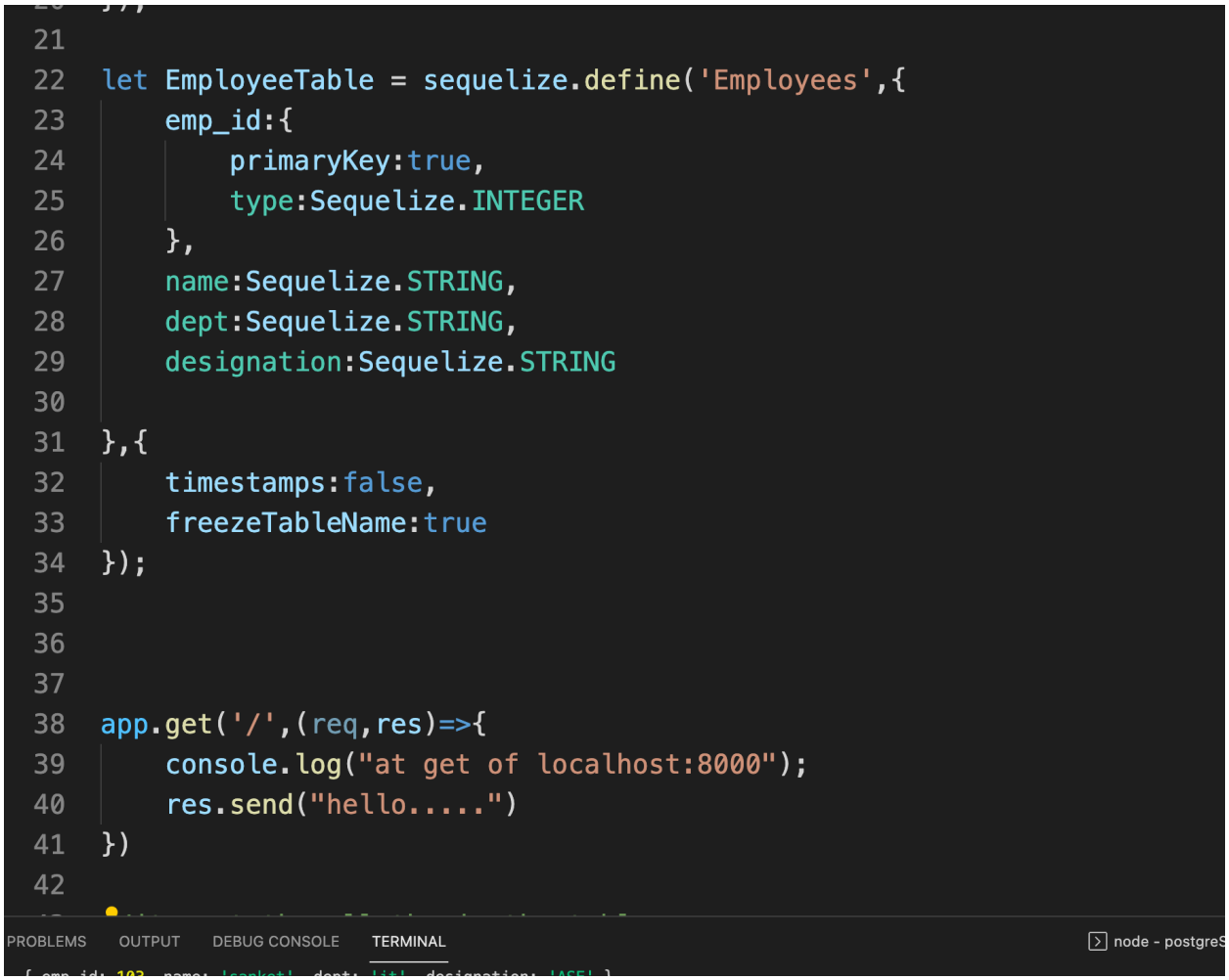
3. Create a post method in express which takes these parameters Basic, HRA,DA, IT and PF and this function calculates the total salary as

total_salary = Basic + HRA + DA - (IT + PF)

and displays the total_salary as the result of this function.

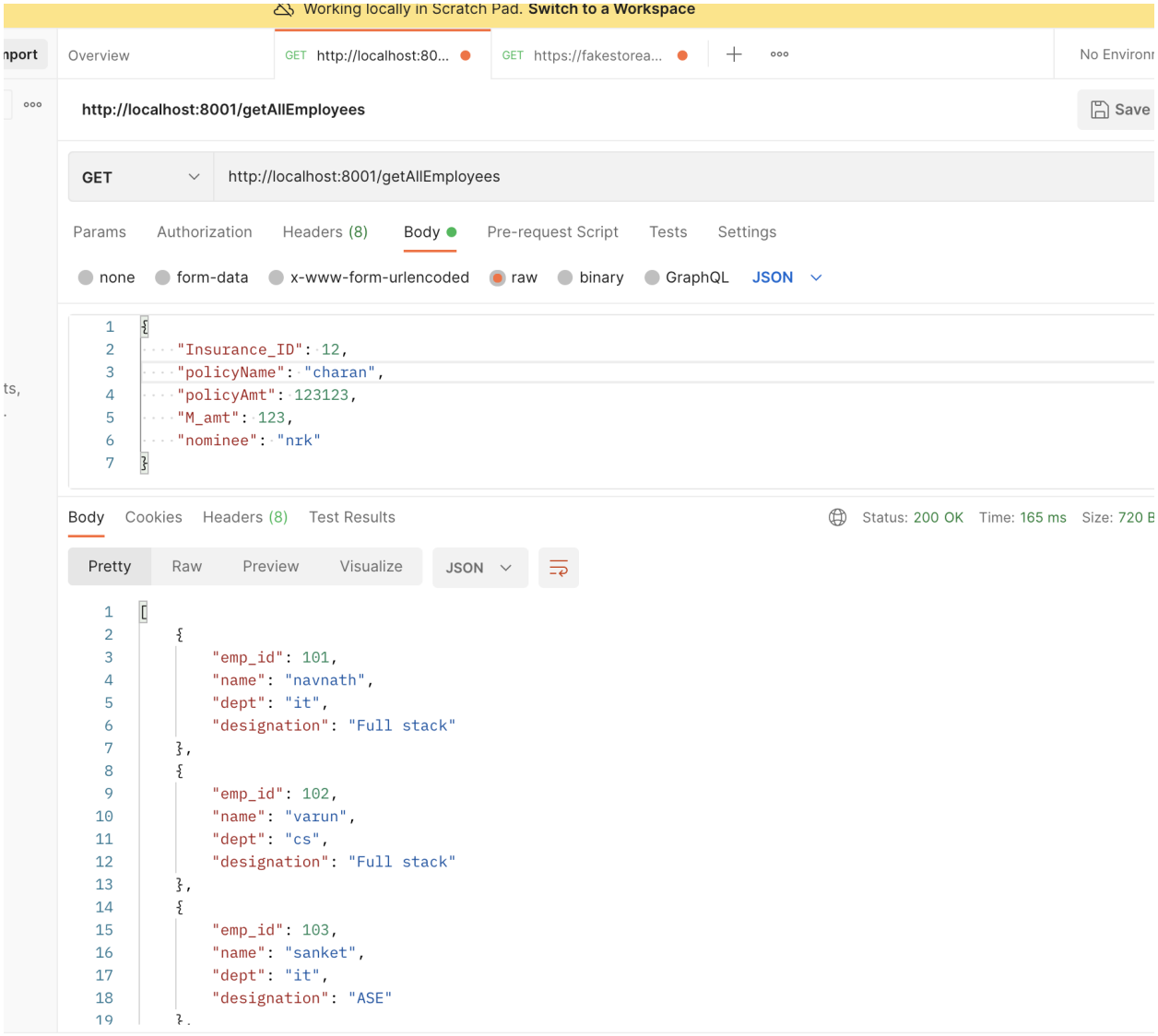


4. Create array of objects of employees with the fields id, name, dept and designation and create below given functions in express and
perform the actions as specified.



(a) get - getAllEmployees() -- Returns all the employees of the array of objects.

```
42
43  //its get the all the in the table
44  app.get('/getAllEmployees',(req,res)=>{
45      EmployeeTable.findAll({raw:true}).then(data=>{
46          console.log(data);
47          res.status(200).send(data);
48
49      }).catch(err=>{
50          console.error("error is :"+err);
51          res.status(400).send(err);
52      })
53  })
54
55  //show data by id
56  app.get('/getAllEmployeeById/:id',(req,res)=>{
57      var id = req.params.id;
58      console.log("given id is :"+id);
59
60      EmployeeTable.findByPk(id,{raw:true}).then(data=>{
61          console.log(data);
62          res.status(200).send(data);
63      }).catch(err=>{
64          console.error("error is :"+err);
```



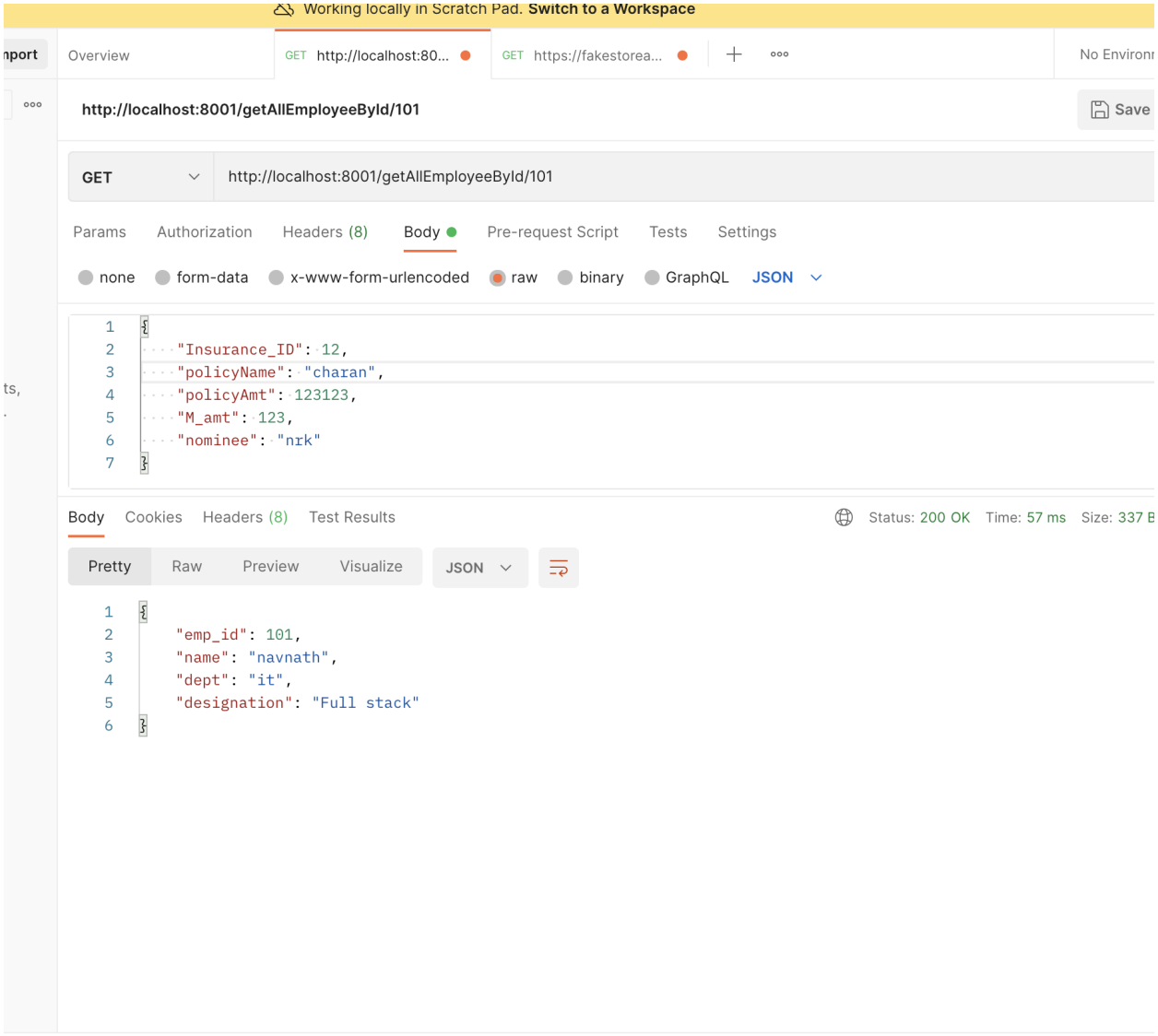
(b) get - getEmployeeById() -- Takes the empid as the path param and displays employee record based on given Id.

```
55  //show data by id
56  app.get('/getAllEmployeeById/:id',(req,res)=>{
57      var id = req.params.id;
58      console.log("given id is :"+id);
59
60      EmployeeTable.findByPk(id,{raw:true}).then(data=>{
61          console.log(data);
62          res.status(200).send(data);
63      }).catch(err=>{
64          console.error("error is :"+err);
65          res.status(400).send(err);
66      })
67  })
68  app.get('/getAllEmployeeByName/:name',(req,res)=>{
69      var name = req.params.name;
70      console.log("given id is :"+name);
71
72      EmployeeTable.find({name: 'name'},function (err, name) {res.json(n
73  .then(data=>{
74          console.log(data);
75          res.status(200).send(data);
76      }).catch(err=>{
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL node - postgreS

```
  { emp_id: 103, name: 'sanket', dept: 'it', designation: 'ASE' },
  { emp_id: 104, name: 'suraj', dept: 'cs', designation: 'ASE' },
  { emp_id: 105, name: 'nrk', dept: 'cs', designation: 'FSASE' },
  { emp_id: 106, name: 'nrk', dept: 'cs', designation: 'FSASE' },
  { emp_id: 108, name: 'abcd', dept: 'CS', designation: 'ASE' }
]
[]
```

Ln 44, Col 26 (15 selected) Spaces: 4 UTF-8 LF {} Babel JavaSc



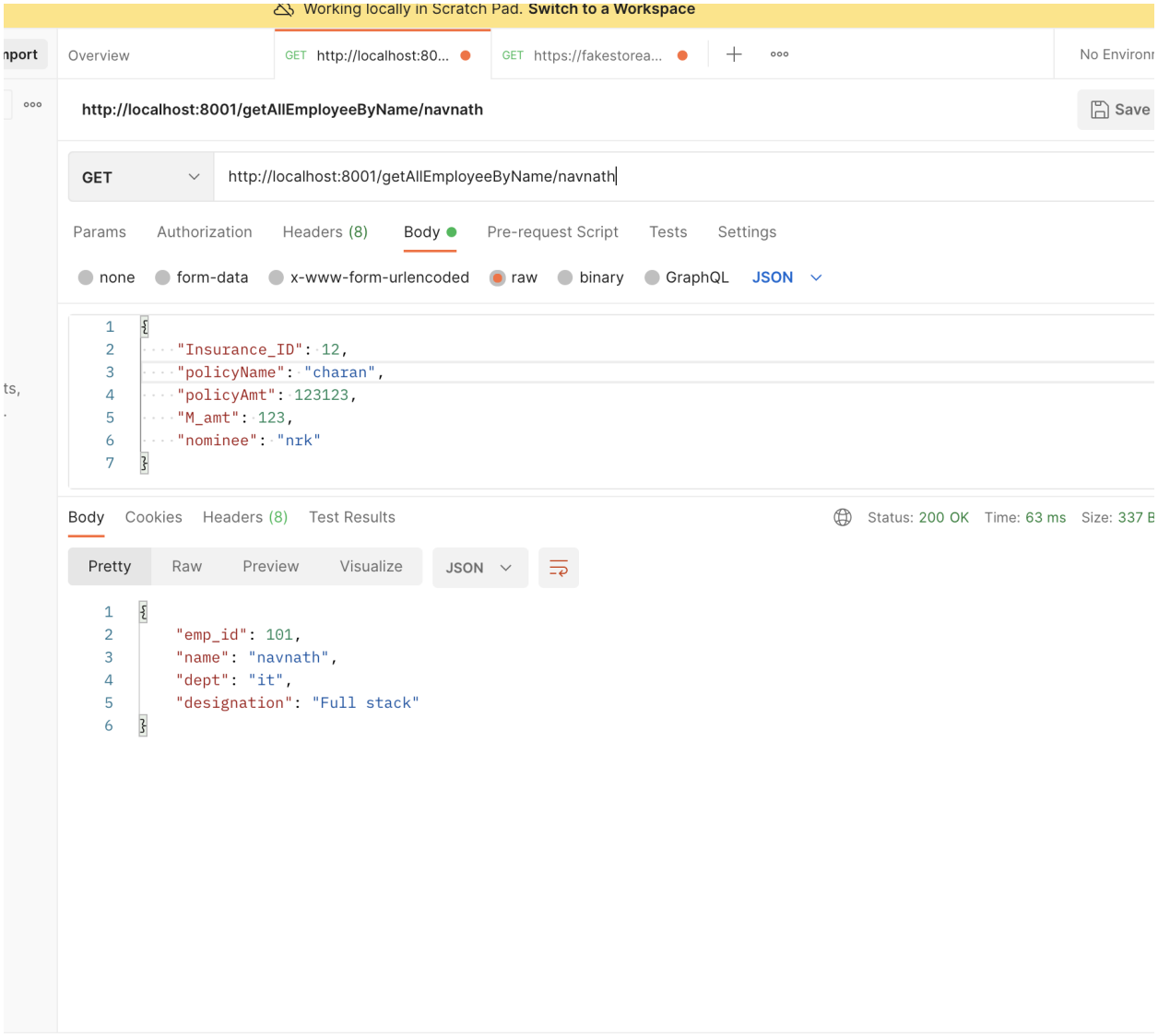
(c) get - getEmployeeByName() -- Returns all the matching records with the given name.

```
65          res.status(400).send(err);
66      })
67  })
68  app.get('/getAllEmployeeByName/:name',(req,res)=>{
69      var name = req.params.name;
70      console.log("given id is :"+name);
71
72      EmployeeTable.find({name: 'name'},function (err, name) {res.json(n
73  .then(data=>{
74          console.log(data);
75          res.status(200).send(data);
76      }).catch(err=>{
77          console.error("error is :"+err);
78          res.status(400).send(err);
79      })
80  })
81
82  //insert the data into table
83  app.use(express.json());
84  app.post("/insertData",(req,res)=>{
85      var emp_id = req.body.emp_id;
86      var name = req.body.name;
87      var dept = req.body.dept;
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL node - postgreS

```
    at param (/Users/navnathkumbhar/Desktop/FULL_STACK/Express/node_modules/express/lib/router/index.js:365:14)
    at Function.process_params (/Users/navnathkumbhar/Desktop/FULL_STACK/Express/node_modules/express/lib/router/index.js:410:3)
    at next (/Users/navnathkumbhar/Desktop/FULL_STACK/Express/node_modules/express/lib/router/index.js:275:10)
given id is :101
Executing (default): SELECT "emp_id", "name", "dept", "designation" FROM "Employees" AS "Employees" WHERE "Employees"."emp_id" = '
{ emp_id: 101, name: 'navnath', dept: 'it', designation: 'Full stack' }
```

Ln 68, Col 31 (20 selected) Spaces: 4 UTF-8 LF {} Babel JavaSc



(d) post - insertEmployeeData() -- Takes an object from input and inserts into the array of objects and returns the string

"Record inserted successfully" or "Unable to insert the record". After this operation, verify in postman that given record

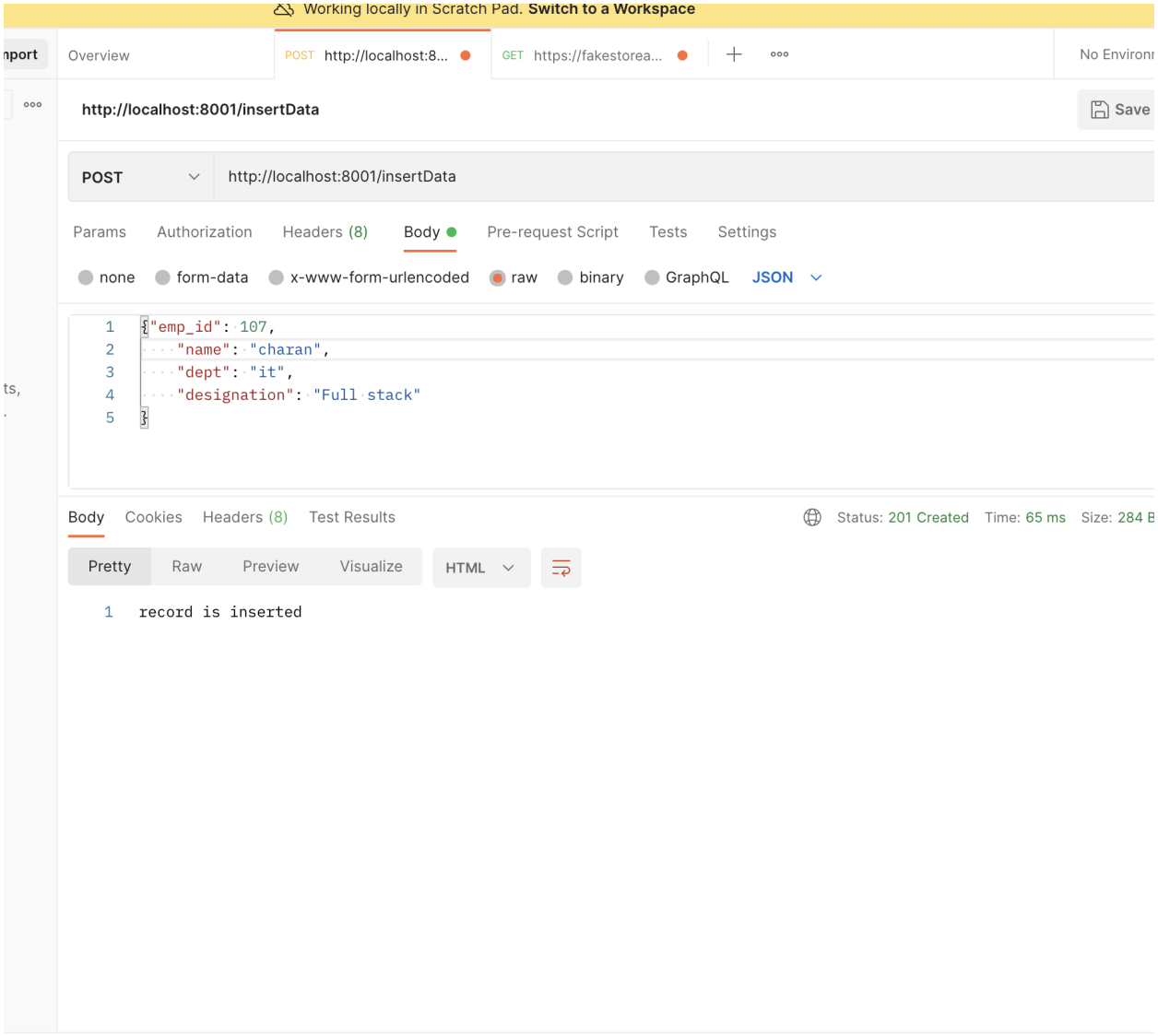
is inserted by giving http://localhost:8001/getAllEmployees

```
82  //insert the data into table
83  app.use(express.json());
84  app.post("/insertData",(req,res)=>{
85      var emp_id = req.body.emp_id;
86      var name = req.body.name;
87      var dept = req.body.dept;
88      var designation = req.body.designation;
89
90      var empObj = EmployeeTable.build({emp_id:emp_id,name:name,dept:dep
91      empObj.save().then(data=>{
92          var strMsg = 'record is inserted ';
93          res.status(201).send(strMsg);
94      }).catch(err=>{
95          console.error("error is :"+err);
96          res.status(400).send(err);
97      })
98  })
99
100 //update the records by id
101 app.put("/updateData",(req,res)=>{
102     var emp_id = req.body.emp_id;
103     var name = req.body.name;
104     var dept = req.body.dept;
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL node - postgreS

```
    at param (/Users/navnathkumbhar/Desktop/FULL_STACK/Express/node_modules/express/lib/router/index.js:365:14)
    at Function.process_params (/Users/navnathkumbhar/Desktop/FULL_STACK/Express/node_modules/express/lib/router/index.js:410:3)
    at next (/Users/navnathkumbhar/Desktop/FULL_STACK/Express/node_modules/express/lib/router/index.js:275:10)
given id is :101
Executing (default): SELECT "emp_id", "name", "dept", "designation" FROM "Employees" AS "Employees" WHERE "Employees"."emp_id" = '
{ emp_id: 101, name: 'navnath', dept: 'it', designation: 'Full stack' }
```

Ln 68, Col 31 (20 selected) Spaces: 4 UTF-8 LF {} Babel JavaSc



(e) put - updateEmployeeData() -- Takes an object and udpates that record in array of objects. After this operation,

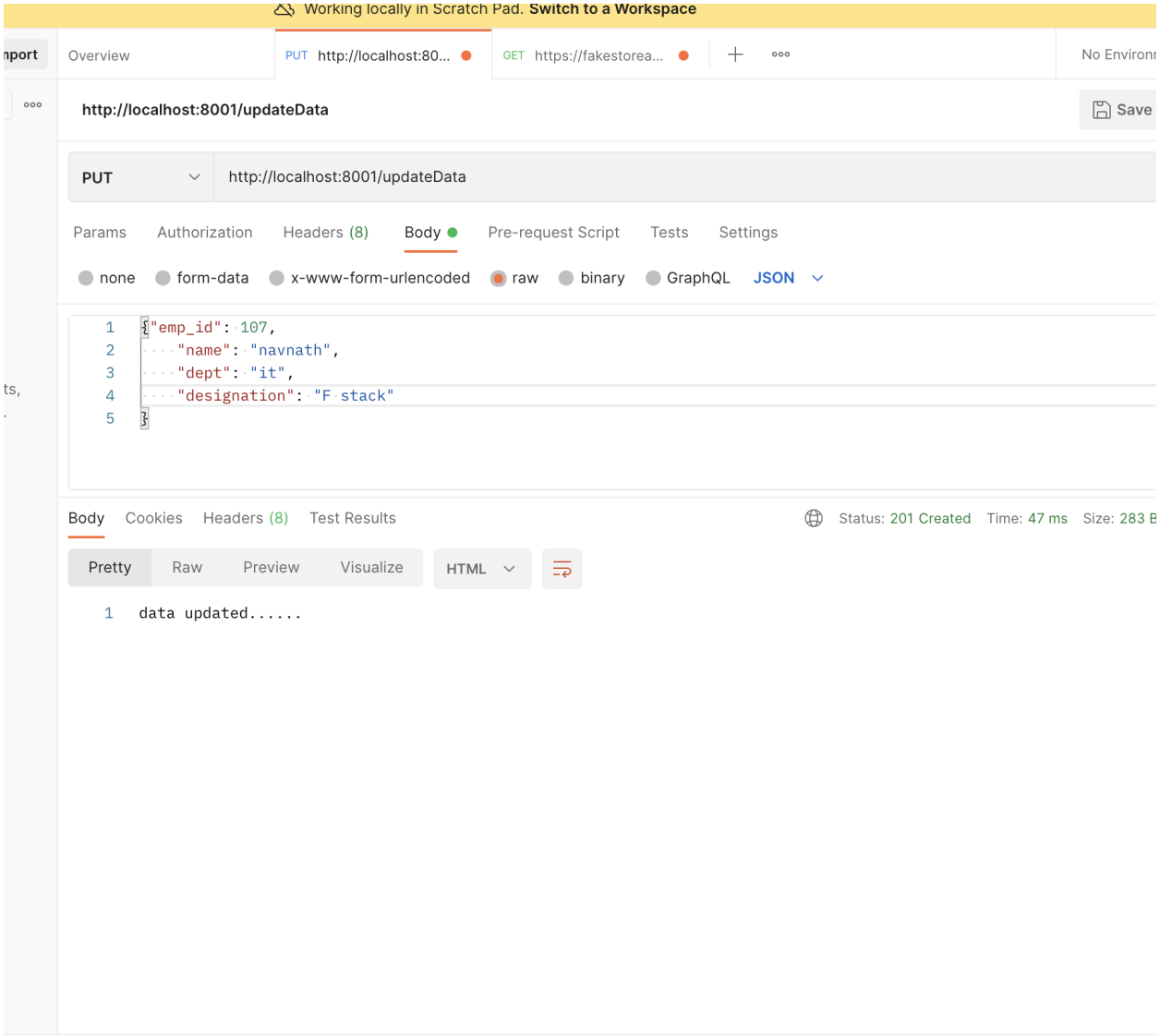
verify in postman that given record is updated by giving

http://localhost:8001/getAllEmployees

```
100  //update the records by id
101  app.put("/updateData",(req,res)=>{
102      var emp_id = req.body.emp_id;
103      var name = req.body.name;
104      var dept = req.body.dept;
105      var designation = req.body.designation;
106
107      var empObj = EmployeeTable.update(
108          {name:name,dept:dept,designation:designation},
109          {where:{emp_id:emp_id}
110      }).then(data=>{
111              console.log("data");
112              var strmsg = "data updated......";
113              res.status(201).send(strmsg);
114          }).catch(err=>{
115              console.error("there is an error :"+err);
116              res.status(400).send(err);
117          })
118
119  });
120
121  //for deleting the data from table
122
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL                                  node - postgreS

    at Function.process_params (/Users/navnathkumbhar/Desktop/FULL_STACK/Express/node_modules/express/lib/router/index.js:410:3)
    at next (/Users/navnathkumbhar/Desktop/FULL_STACK/Express/node_modules/express/lib/router/index.js:275:10)
given id is :101
Executing (default): SELECT "emp_id", "name", "dept", "designation" FROM "Employees" AS "Employees" WHERE "Employees"."emp_id" = '
{ emp_id: 101, name: 'navnath', dept: 'it', designation: 'Full stack' }
Executing (default): INSERT INTO "Employees" ("emp_id","name","dept","designation") VALUES ($1,$2,$3,$4) RETURNING "emp_id","name"
```

```
Ln 84, Col 22 (10 selected)  Spaces: 4  UTF-8  LF  {} Babel JavaSc
```



(f) delete - deleteRecord() -- Takes the id as the path parameter and removes the record on given id and returns

the "Record deleted successfully" or "Unable to delete record".After this operation, verify in postman that given record

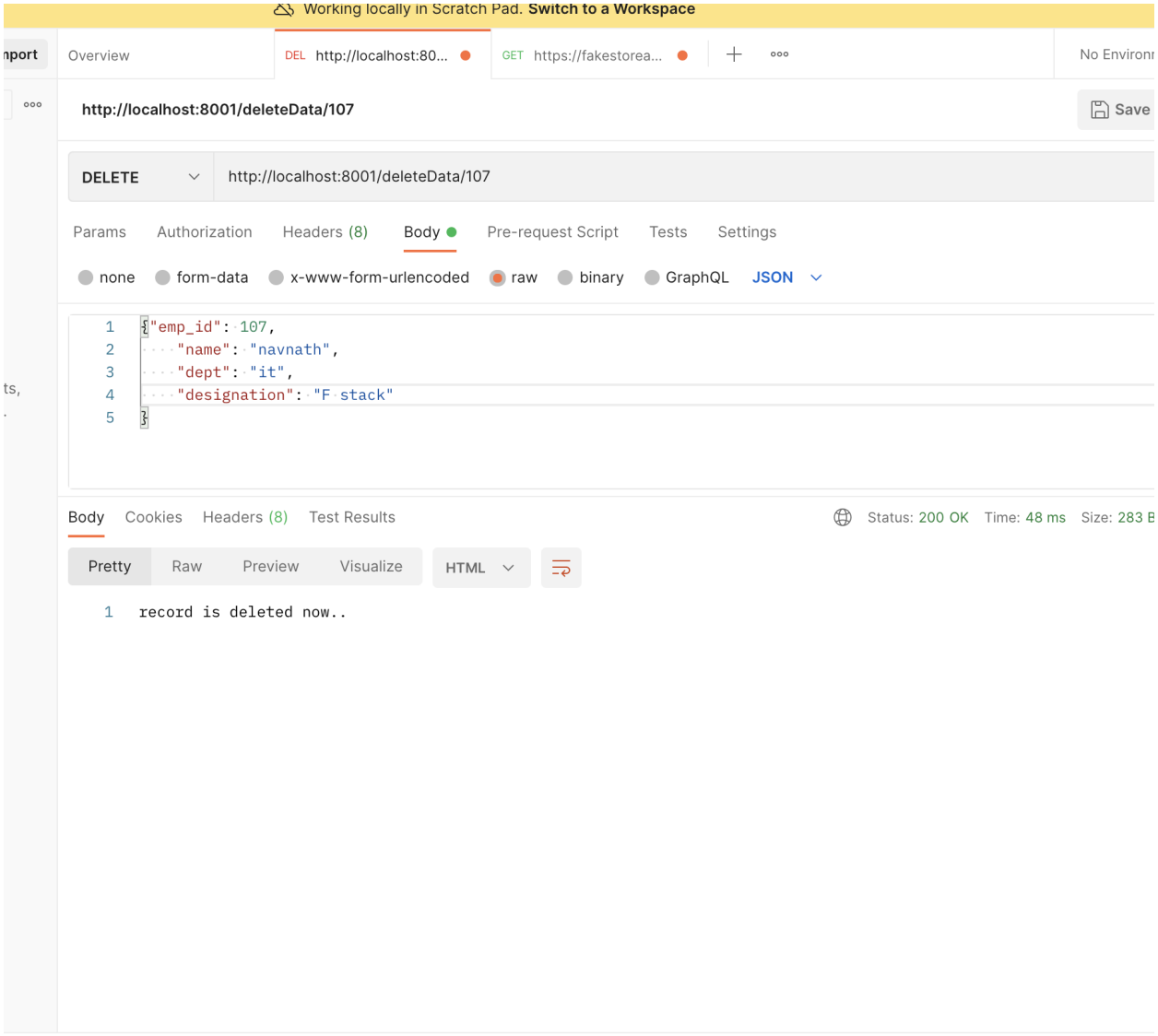
is deleted by giving http://localhost:8001/getAllEmployees

```
119 });
120
121 //for deleting the data from table
122
123 app.delete("/deleteData/:id",(req,res)=>{
124     console.log("enter deleteby the id");
125     var id =req.params.id;
126     console.log("given id is :"+id);
127
128     EmployeeTable.destroy({where:{emp_id:id}}).then(data=>{
129         console.log(data);
130         var strMsg = "record is deleted now..";
131         res.status(200).send(strMsg);
132     }).catch(err=>{
133         console.error("there is some error :"+err);
134         res.status(400).send(err);
135     })
136 })
137
138
139 app.post('/salary',(req,res)=>{
140     var bas = req.body.bas;
```

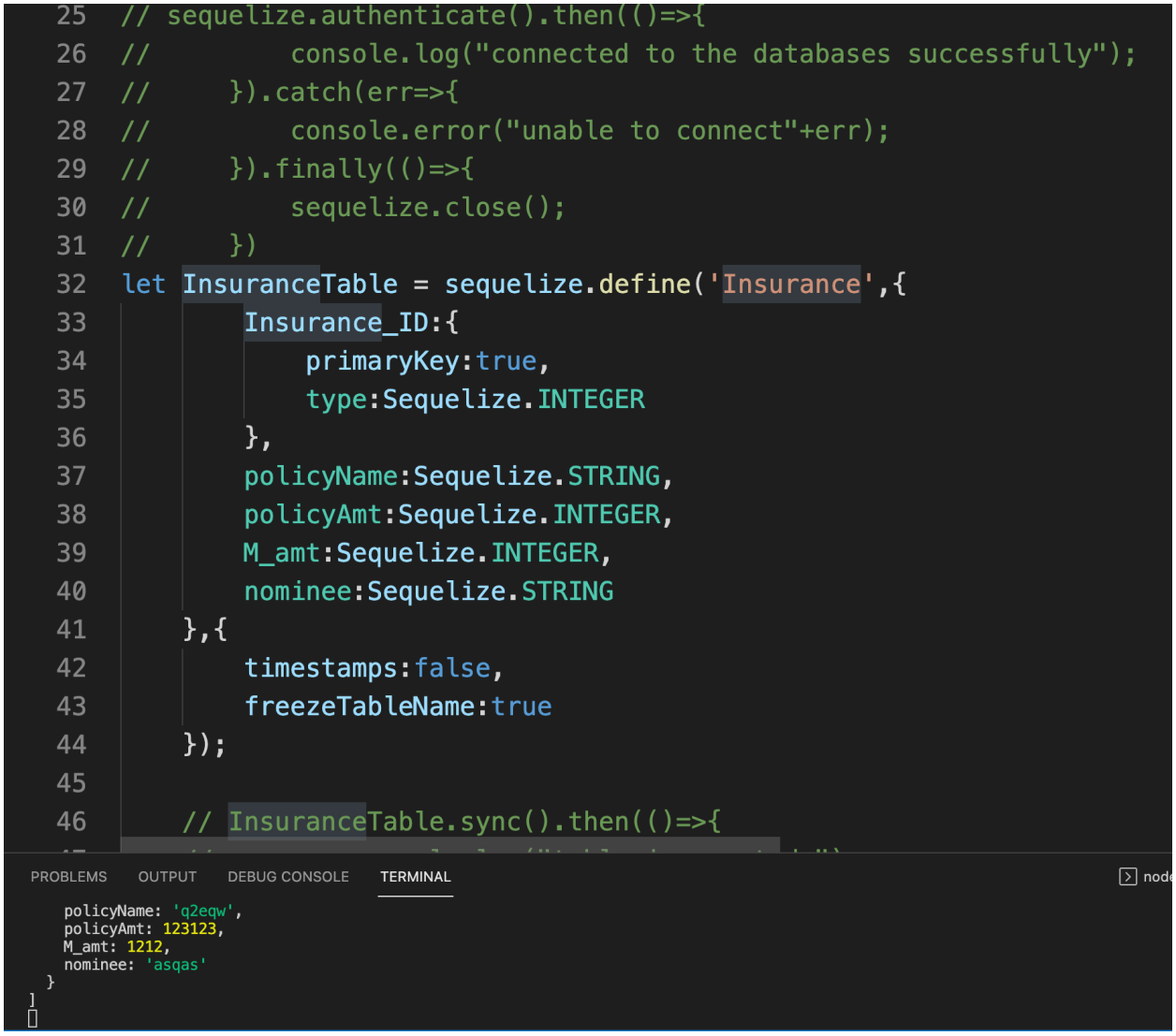
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL node - postgreS

```
Executing (default): UPDATE "Employees" SET "name"=$1,"dept"=$2,"designation"=$3 WHERE "emp_id" = $4
data
enter deleteby the id
given id is :107
Executing (default): DELETE FROM "Employees" WHERE "emp_id" = '107'
1
```

Ln 123, Col 24 (10 selected) Spaces: 4 UTF-8 LF {} Babel JavaSc



(5) Define the table named Insurance and have the fields (a) PolicyNumber (Primary Key)



(b)Policy Holders Name (c)Policy Amt (d) Maturity Amount (e) Nominee

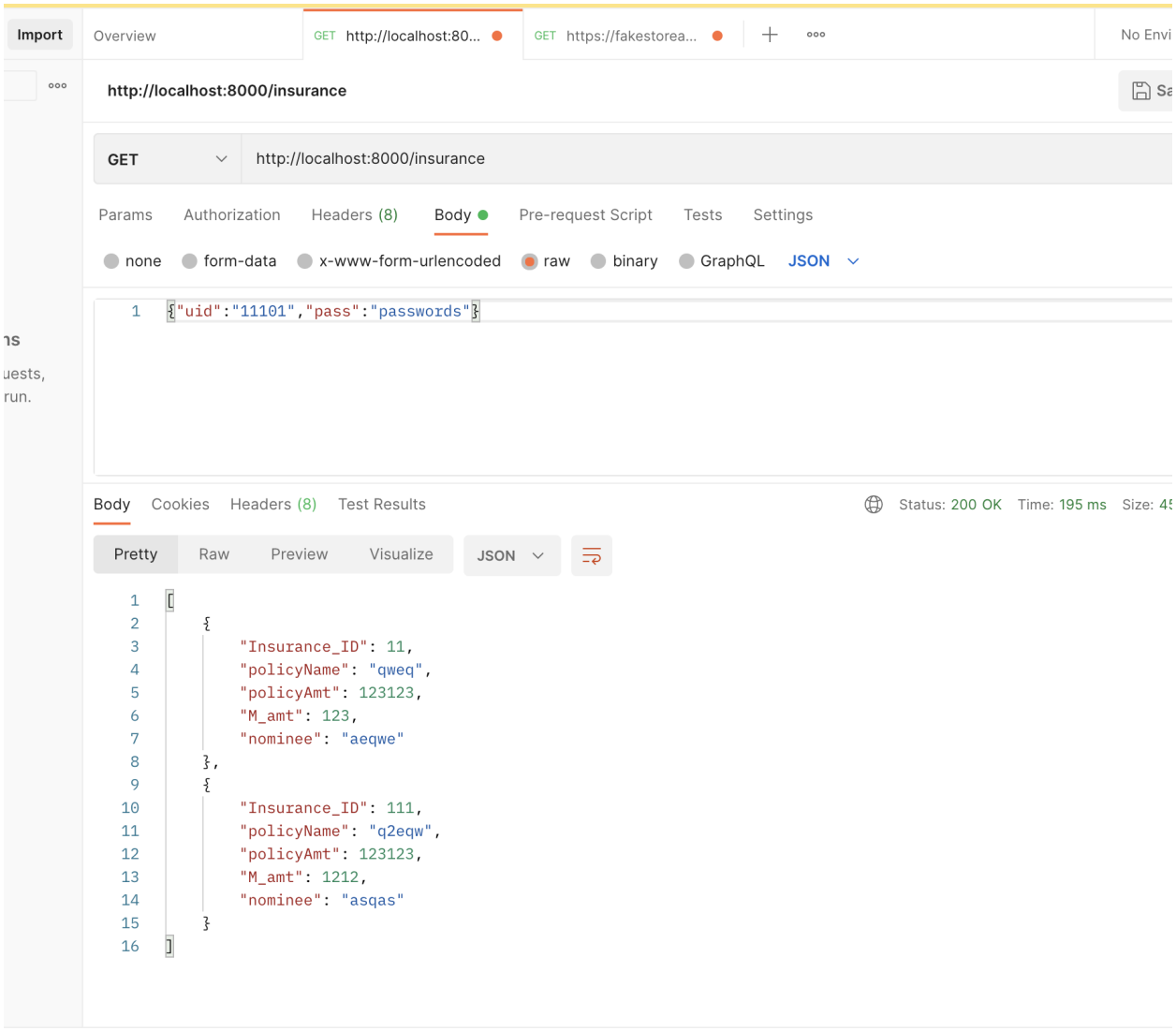
Define the following functions in express and data has to be fetched from DB using Sequelize.

(a)get - getAllPolicies - Retrieves all the policies in the table.

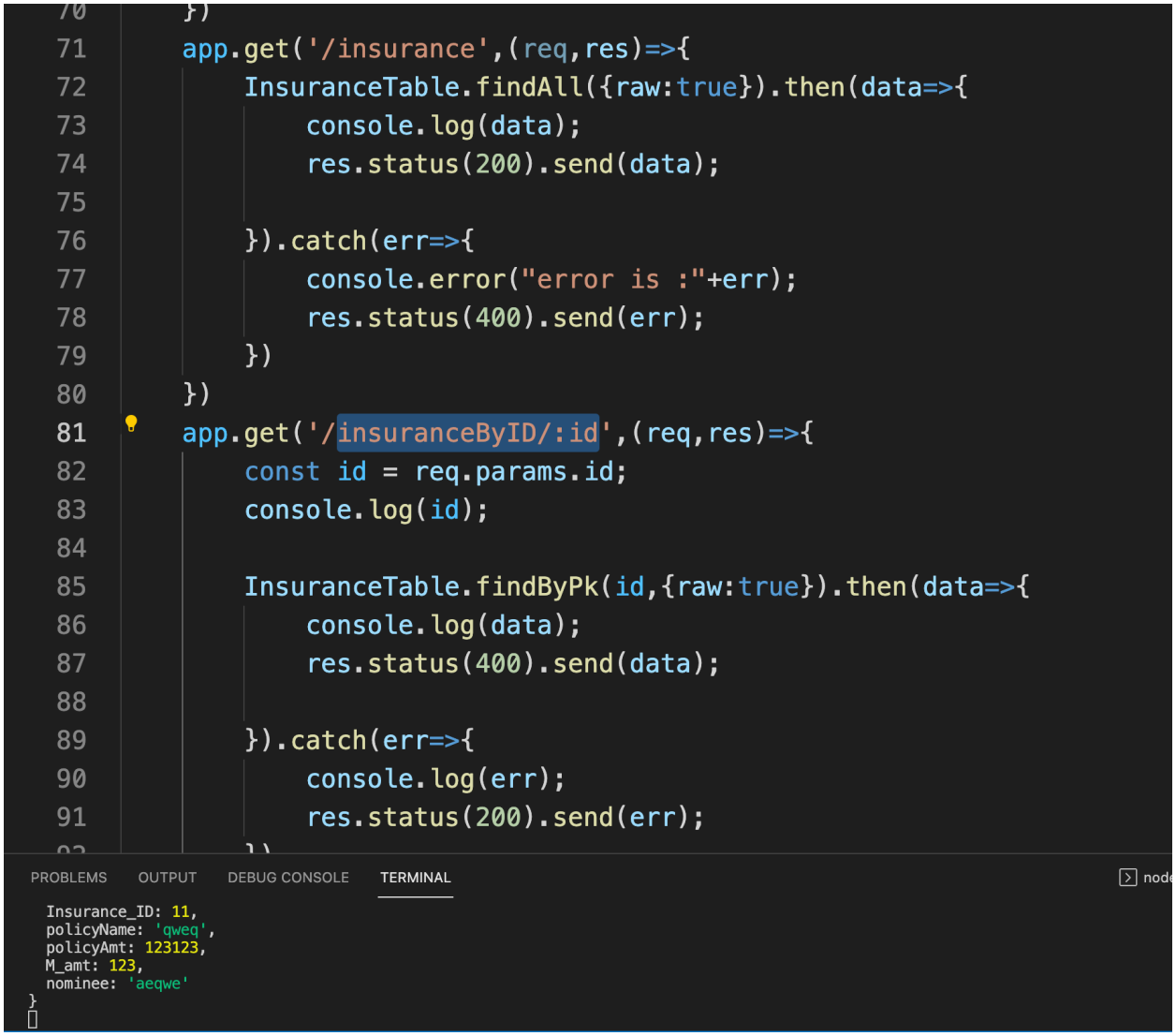
```
67      app.get('/',(req,res)=>{
68          console.log("working fine");
69          res.send("working with 8000 port")
70      })
71      app.get('/insurance',(req,res)=>{
72          InsuranceTable.findAll({raw:true}).then(data=>{
73              console.log(data);
74              res.status(200).send(data);
75
76          }).catch(err=>{
77              console.error("error is :"+err);
78              res.status(400).send(err);
79          })
80      })
81      app.get('/insuranceByID/:id',(req,res)=>{
82          const id = req.params.id;
83          console.log(id);
84
85          InsuranceTable.findByPk(id,{raw:true}).then(data=>{
86              console.log(data);
87              res.status(400).send(data);
88
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL node

```
    policyName: 'q2eqw',
    policyAmt: 123123,
    M_amt: 1212,
    nominee: 'asqas'
  }
]
```



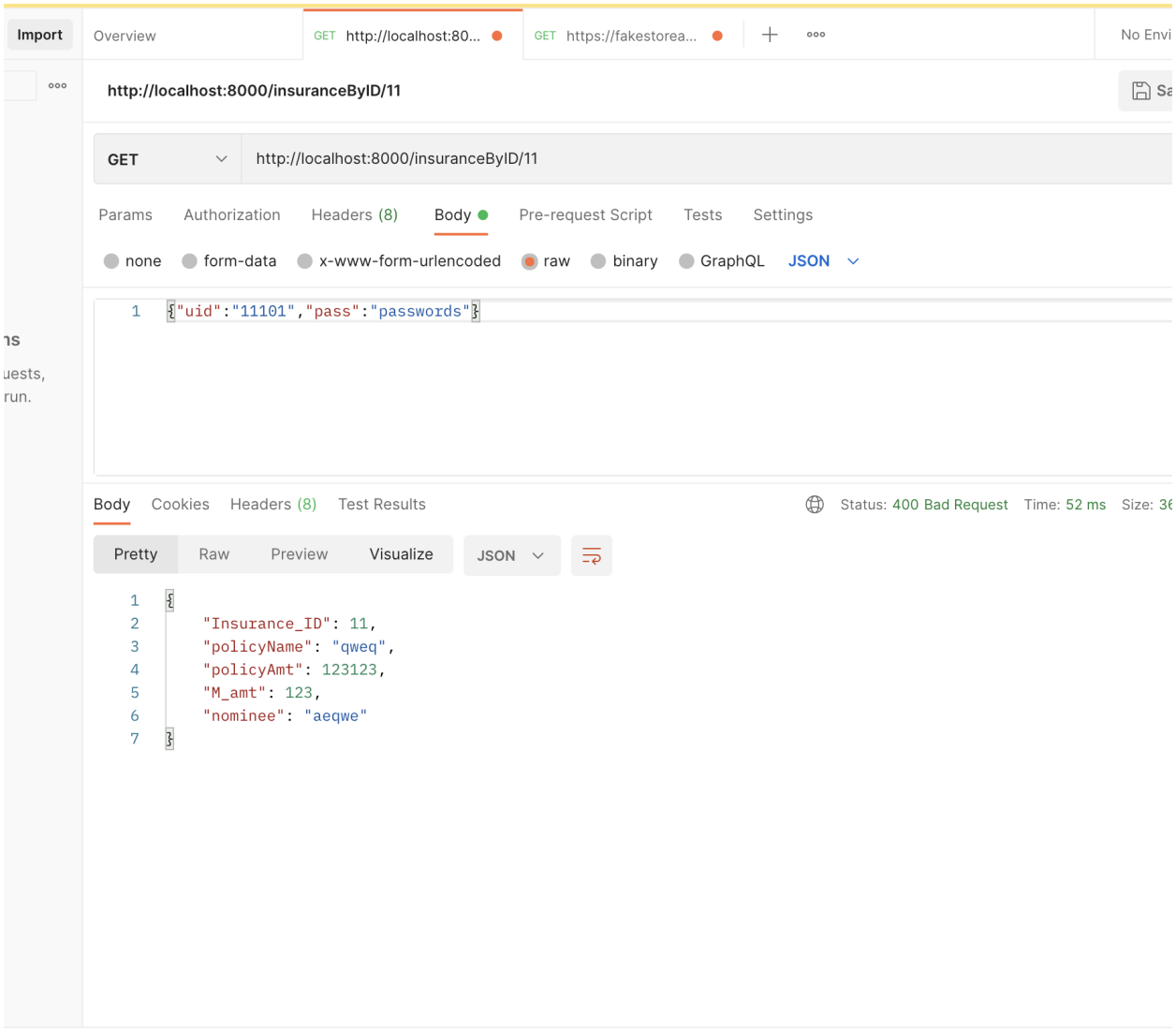
(b)get - getPolicyById - Retrieves the record based on the given policy id.



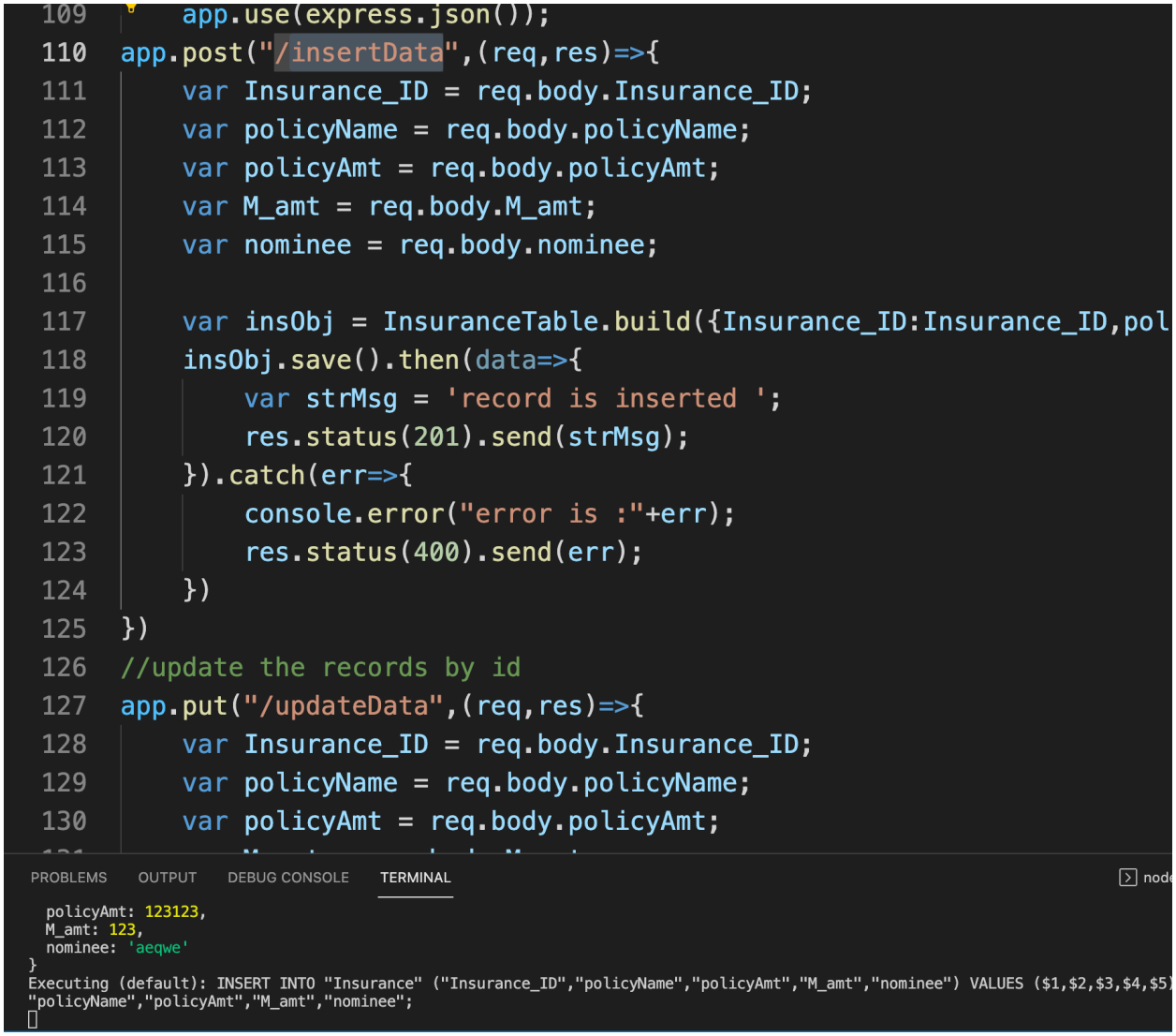
```
})
app.get('/insurance',(req,res)=>{
    InsuranceTable.findAll({raw:true}).then(data=>{
        console.log(data);
        res.status(200).send(data);

    }).catch(err=>{
        console.error("error is :"+err);
        res.status(400).send(err);
    })
})
app.get('/insuranceByID/:id',(req,res)=>{
    const id = req.params.id;
    console.log(id);

    InsuranceTable.findByPk(id,{raw:true}).then(data=>{
        console.log(data);
        res.status(400).send(data);

    }).catch(err=>{
        console.log(err);
        res.status(200).send(err);
    })
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL node

```
  Insurance_ID: 11,
  policyName: 'qweq',
  policyAmt: 123123,
  M_amt: 123,
  nominee: 'aeqwe'
}
```



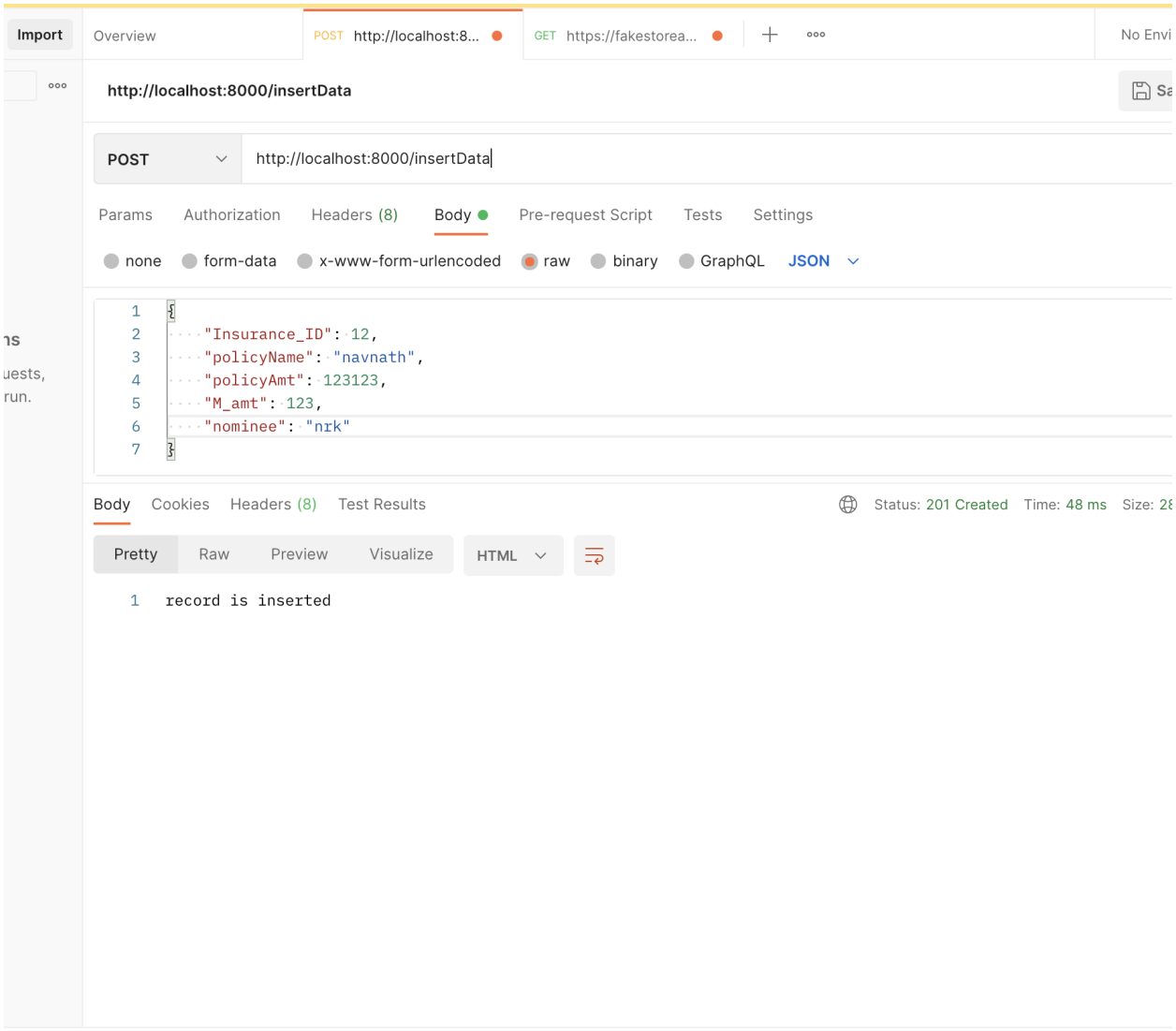
(c)post - newRecord - Inserts the data in the Insurance table.



```
109     app.use(express.json());
110 app.post("/insertData",(req,res)=>{
111     var Insurance_ID = req.body.Insurance_ID;
112     var policyName = req.body.policyName;
113     var policyAmt = req.body.policyAmt;
114     var M_amt = req.body.M_amt;
115     var nominee = req.body.nominee;
116 
117     var insObj = InsuranceTable.build({Insurance_ID:Insurance_ID,pol
118     insObj.save().then(data=>{
119         var strMsg = 'record is inserted ';
120         res.status(201).send(strMsg);
121     }).catch(err=>{
122         console.error("error is :"+err);
123         res.status(400).send(err);
124     })
125 })
126 //update the records by id
127 app.put("/updateData",(req,res)=>{
128     var Insurance_ID = req.body.Insurance_ID;
129     var policyName = req.body.policyName;
130     var policyAmt = req.body.policyAmt;
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL node

```
  policyAmt: 123123,
  M_amt: 123,
  nominee: 'aeqwe'
}
Executing (default): INSERT INTO "Insurance" ("Insurance_ID","policyName","policyAmt","M_amt","nominee") VALUES ($1,$2,$3,$4,$5)
"policyName","policyAmt","M_amt","nominee";
```

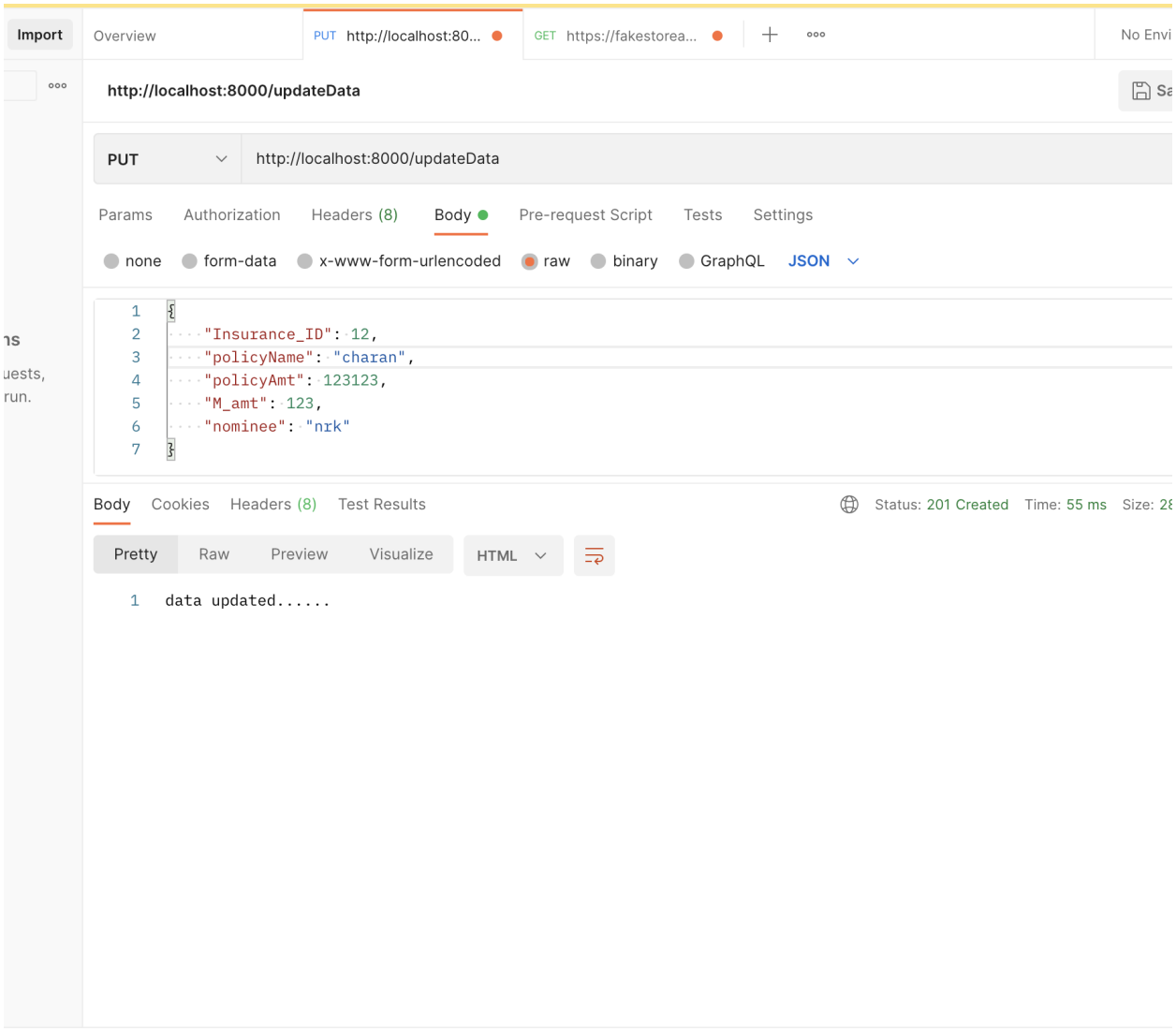


(d) put - updatePolicy - Updates the policy record based on policy number.

```
127 app.put('/updateData',(req,res)=>{
128     var Insurance_ID = req.body.Insurance_ID;
129     var policyName = req.body.policyName;
130     var policyAmt = req.body.policyAmt;
131     var M_amt = req.body.M_amt;
132     var nominee = req.body.nominee;
133
134     var insObj = InsuranceTable.update(
135         {Insurance_ID:Insurance_ID,
136             policyName:policyName,
137             policyAmt:policyAmt,
138             M_amt:M_amt,
139             nominee:nominee},
140         {where:{Insurance_ID:Insurance_ID}
141     }).then(data=>{
142             console.log("data");
143             var strmsg = "data updated......";
144             res.status(201).send(strmsg);
145         }).catch(err=>{
146             console.error("there is an error :"+err);
147             res.status(400).send(err);
148         })
149
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL node

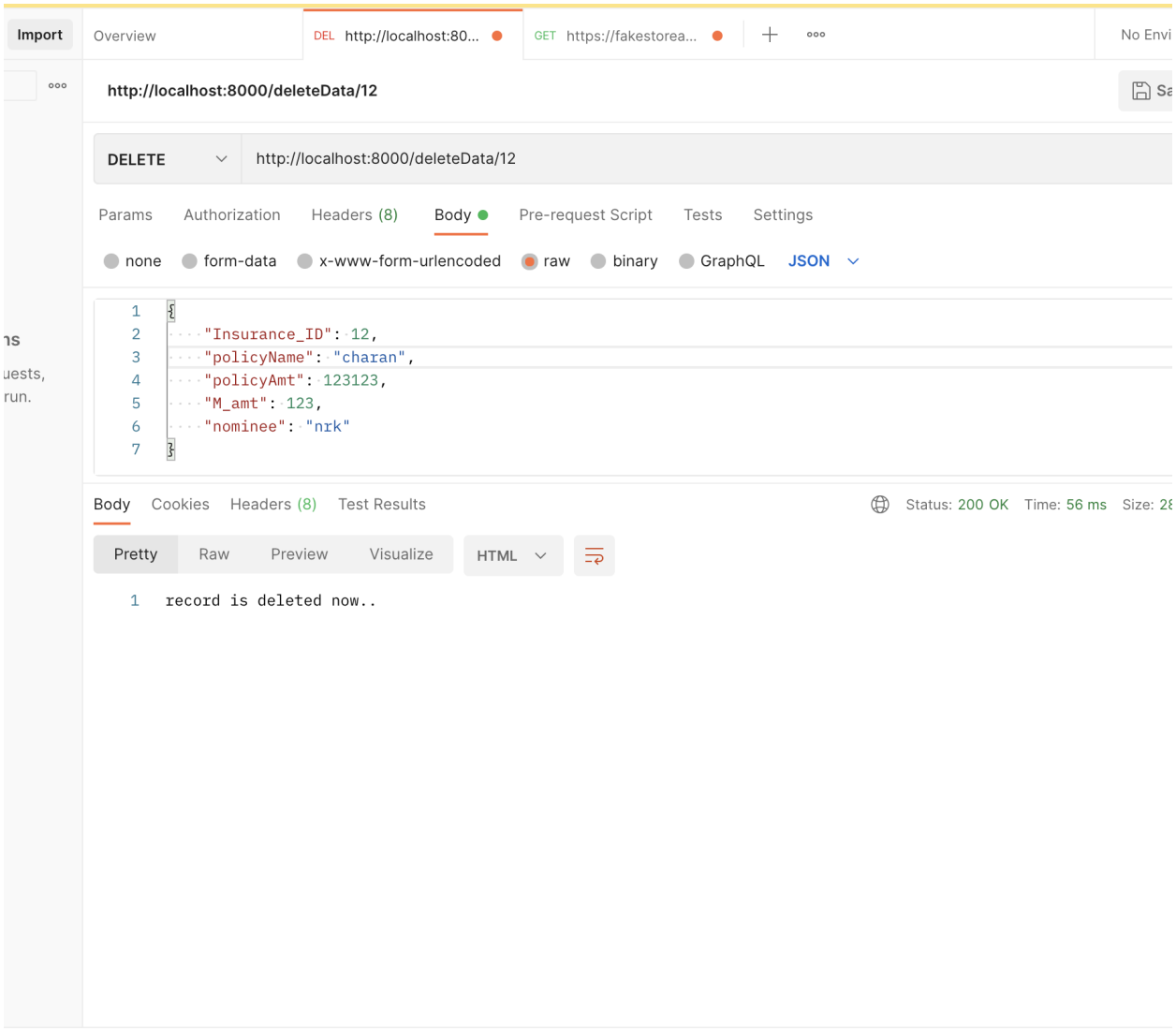
```
  nominee: 'aeqwe'
}
Executing (default): INSERT INTO "Insurance" ("Insurance_ID","policyName","policyAmt","M_amt","nominee") VALUES ($1,$2,$3,$4,$5)
"policyName","policyAmt","M_amt","nominee";
Executing (default): UPDATE "Insurance" SET "Insurance_ID"=$1,"policyName"=$2,"policyAmt"=$3,"M_amt"=$4,"nominee"=$5 WHERE "Insu
data
```

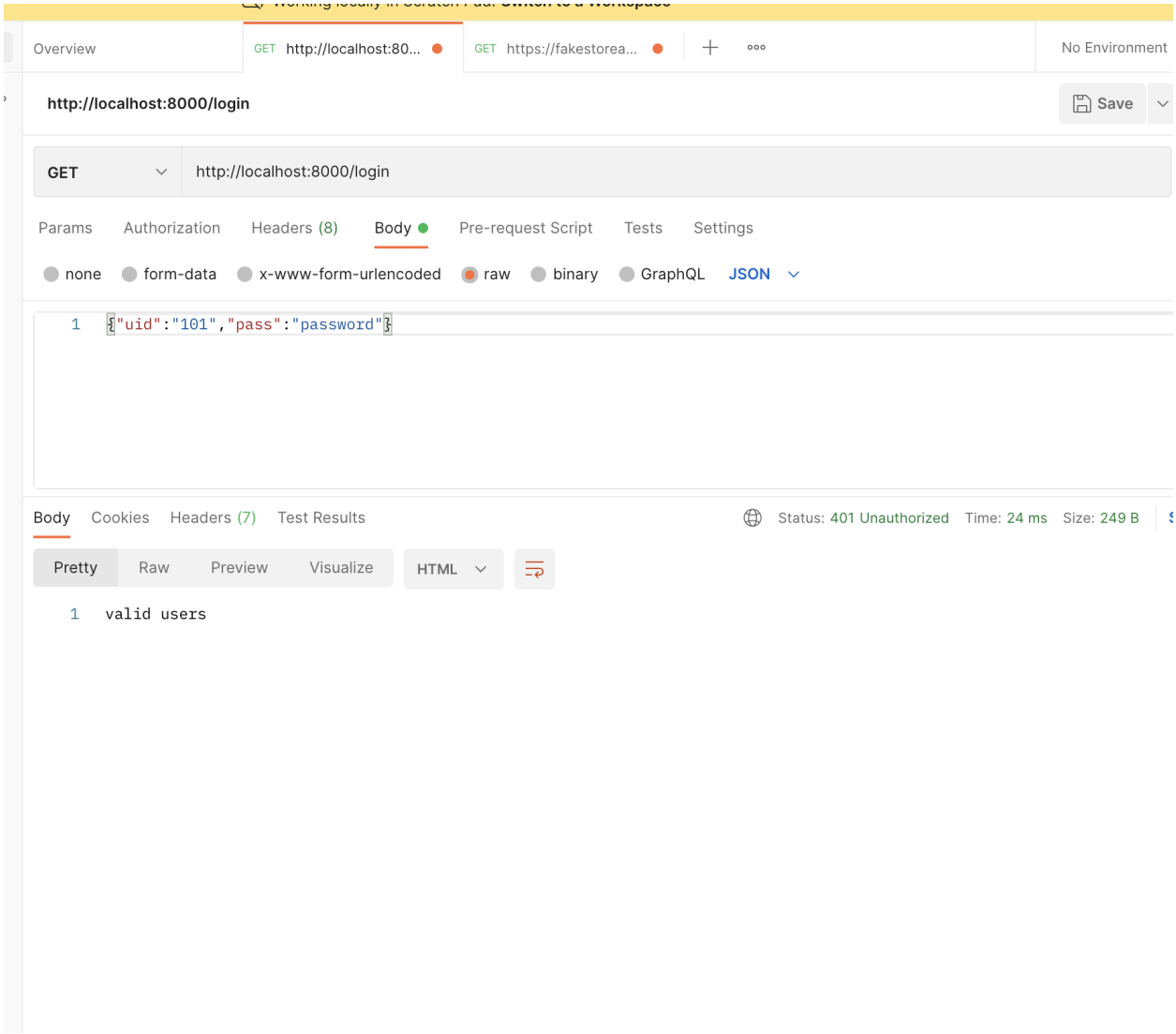


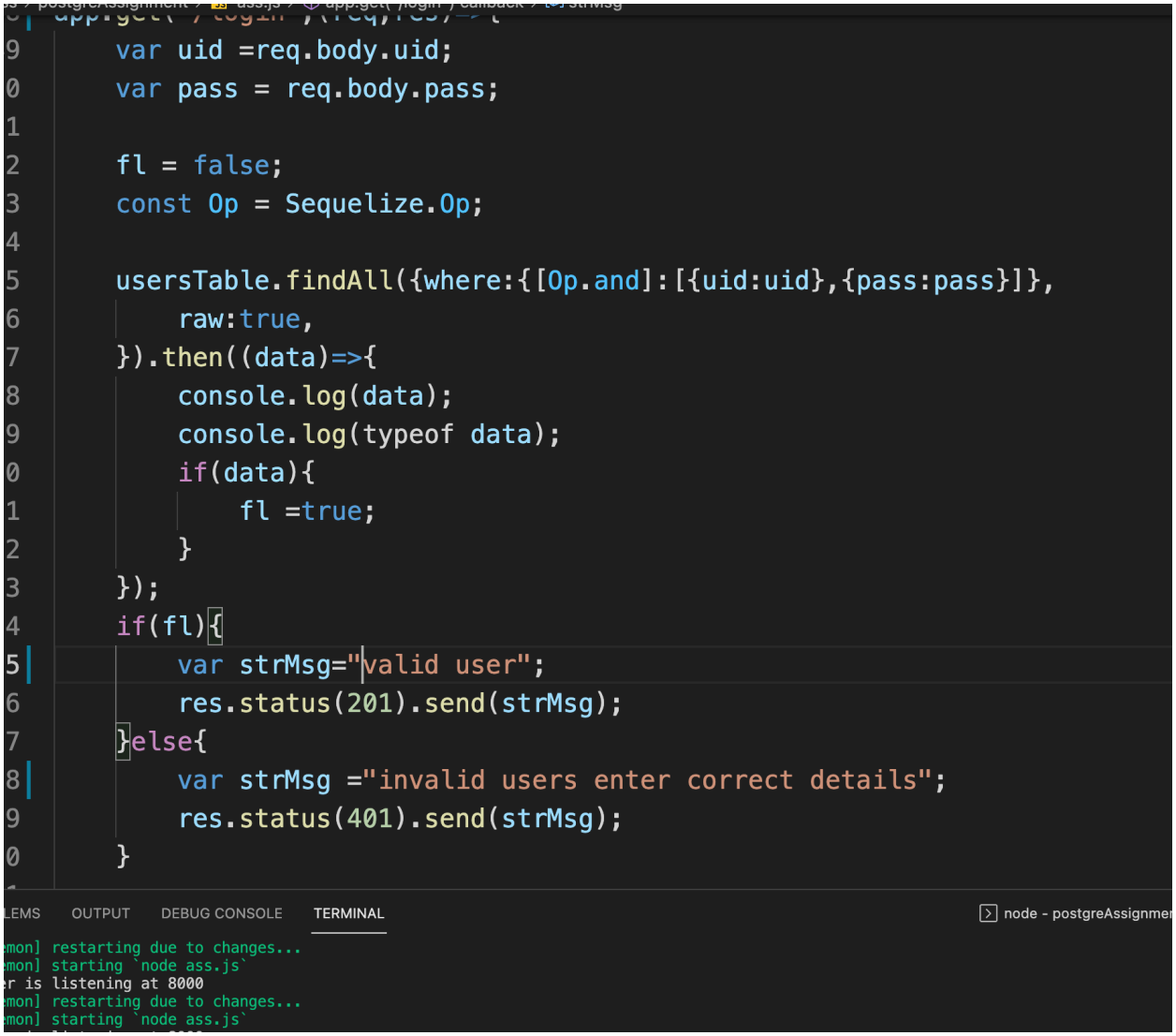
(f) delete - deletePolicy - Deletes the policy record based on policyNumber

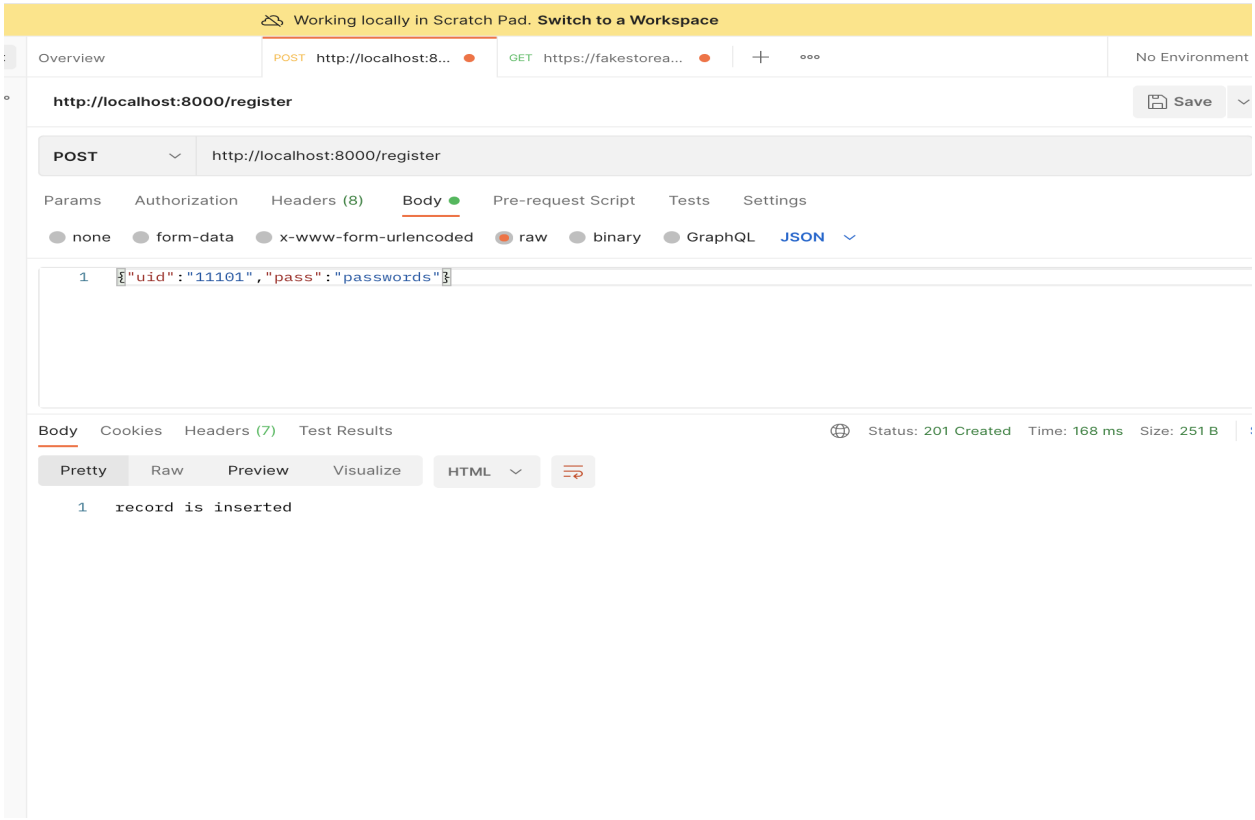
```
145         }).catch(err=>{
146             console.error("there is an error :"+err);
147             res.status(400).send(err);
148         })
149 
150 });
151 
152 app.delete("/deleteData/:id",(req,res)=>{
153     console.log("enter deleteby the id");
154     var id =req.params.id;
155     console.log("given id is :"+id);
156 
157     InsuranceTable.destroy({where:{Insurance_ID:id}}).then(data=>{
158         console.log(data);
159         var strMsg = "record is deleted now..";
160         res.status(200).send(strMsg);
161     }).catch(err=>{
162         console.error("there is some error :"+err);
163         res.status(400).send(err);
164     })
165 })
166 
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL                                  node

Executing (default): UPDATE "Insurance" SET "Insurance_ID"=$1,"policyName"=$2,"policyAmt"=$3,"M_amt"=$4,"nominee"=$5 WHERE "Insu
data
enter deleteby the id
given id is :12
Executing (default): DELETE FROM "Insurance" WHERE "Insurance_ID" = '12'
1
```


Import
Overview
DEL http://localhost:80...
GET https://fakestorea...
No Envi
http://localhost:8000/deleteData/12
DELETE
http://localhost:8000/deleteData/12
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
{
"Insurance_ID": 12,
"policyName": "charan",
"policyAmt": 123123,
"M_amt": 123,
"nominee": "nrk"
}
Body
Cookies
Headers (8)
Test Results
Status: 200 OK
Time: 56 ms
Pretty
Raw
Preview
Visualize
HTML
record is deleted now..


Overview
GET http://localhost:80...
GET https://fakestorea...
No Environment
http://localhost:8000/login
Save
GET
http://localhost:8000/login
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
1 {"uid":"101","pass":"password"}
Body
Cookies
Headers (7)
Test Results
Status: 401 Unauthorized
Time: 24 ms
Size: 249 B
Pretty
Raw
Preview
Visualize
HTML
1 valid users


var uid =req.body.uid;
var pass = req.body.pass;
fl = false;
const Op = Sequelize.Op;
usersTable.findAll({where:{[Op.and]:[{uid:uid},{pass:pass}]},
raw:true,
}).then((data)=>{
console.log(data);
console.log(typeof data);
if(data){
fl =true;
}
});
if(fl){
var strMsg="valid user";
res.status(201).send(strMsg);
}else{
var strMsg ="invalid users enter correct details";
res.status(401).send(strMsg);
}
OUTPUT
DEBUG CONSOLE
TERMINAL
node - postgreAssignmer
mon] restarting due to changes...
mon] starting `node ass.js`
r is listening at 8000
mon] restarting due to changes...
mon] starting `node ass.js`


Working locally in Scratch Pad. Switch to a Workspace
Overview
POST http://localhost:8...
GET https://fakestorea...
No Environment
http://localhost:8000/register
Save
POST
http://localhost:8000/register
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
1 {"uid":"11101","pass":"passwords"}
Body
Cookies
Headers (7)
Test Results
Status: 201 Created
Time: 168 ms
Size: 251 B
Pretty
Raw
Preview
Visualize
HTML
1 record is inserted

```
})

app.post("/register",(req,res)=>{
    var uid = req.body.uid;
    var pass = req.body.pass;

    var userObj = usersTable.build({uid:uid,pass:pass});
    userObj.save().then(data=>{
        var stri = "record is inserted";
        res.status(201).send(stri);
    }).catch(err=>{
        console.error("error is :"+err);
        res.status(400).send(err);
    })
})

app.listen(8000,()=>{
    console.log("server is listening at 8000");
})

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL node - postgre

```
[nodemon] starting `node ass.js`
server is listening at 8000
[nodemon] restarting due to changes...
[nodemon] starting `node ass.js`
server is listening at 8000
Executing (default): INSERT INTO "users" ("uid","pass") VALUES ($1,$2) RETURNING "uid","pass";
```