

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.

## Login

Username:

Password:

Remember me

**submit**

[Register here](#)



---

Not valid credential 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.

```
14     min:dbConfig.pool.min,
15     max:dbConfig.pool.max,
16     acquire:dbConfig.pool.acquire,
17     idle:dbConfig.pool.idle
18
19   }
20 });
21
22 let EmployeeTable = sequelize.define('Employees',{
23   emp_id:{
24     primaryKey:true,
25     type:Sequelize.INTEGER
26   },
27   name:Sequelize.STRING,
28   dept:Sequelize.STRING,
29   designation:Sequelize.STRING
30
31 },{
32   timestamps:false,
33   freezeTableName:true
34 });
```

The screenshot shows a code editor interface with a dark theme. At the top, there are tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The TERMINAL tab is active, showing the command `node - postgres`. Below the tabs, the code for the `EmployeeTable` definition is displayed. In the bottom right corner of the terminal area, there is a small icon with a checkmark and the text "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' }
```

```

12
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);
65         res.status(400).send(err);
66     })
67 })

```

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' }
]

```

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

$$\text{total\_salary} = \text{Basic} + \text{HRA} + \text{DA} - (\text{IT} + \text{PF})$$

and displays the total\_salary as the result of this function.

```
1 const express = require("express");
2 const app = express();
3
4 app.use(express.json());
5
6 app.post('/salary',(req,res)=>{
7     var bas = req (property) Request<{}, any, any, QueryString.ParsedQ
8     var HRA = req Record<string, any>.body: any
9     var DA = req.body.DA;
10    var IT = req.body.IT;
11    var PF = req.body.PF;
12    var total = bas + HRA +DA -(IT+PF);
13    console.log("total salary is :" +total);
14    res.send("your salary is :" +total);
15 });
16 app.listen(8000, ()=>{
17     console.log("server is listening at 8000");
18 })
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL node - postgreS

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.

```
20      },
21
22  let EmployeeTable = sequelize.define('Employees',{
23      emp_id:{
24          primaryKey:true,
25          type:Sequelize.INTEGER
26      },
27      name:Sequelize.STRING,
28      dept:Sequelize.STRING,
29      designation:Sequelize.STRING
30
31 },{
32     timestamps:false,
33     freezeTableName:true
34 });
35
36
37
38 app.get('/',(req,res)=>{
39     console.log("at get of localhost:8000");
40     res.send("hello.....")
41 })
42
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL node - postgreSQL

(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);
```

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

node - postgreS

```
[{"emp_id": 101, "name": "Leanne", "dept": "HR", "designation": "Manager"}]
```

Working locally in Scratch Pad. [Switch to a Workspace](#)

Import Overview GET http://localhost:80... ● GET https://fakestorea... ● + ⚙ No Environment

http://localhost:8001/getAllEmployees [Save](#)

GET [Raw](#) http://localhost:8001/getAllEmployees

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL **JSON** ▾

```
1 {
2   "Insurance_ID": 12,
3   "policyName": "charan",
4   "policyAmt": 123123,
5   "M_amt": 123,
6   "nominee": "nrk"
7 }
```

Body Cookies Headers (8) Test Results Status: 200 OK Time: 165 ms Size: 720 B

Pretty Raw Preview Visualize JSON [Copy](#)

```
1 [
2   {
3     "emp_id": 101,
4     "name": "navnath",
5     "dept": "it",
6     "designation": "Full stack"
7   },
8   {
9     "emp_id": 102,
10    "name": "vazun",
11    "dept": "cs",
12    "designation": "Full stack"
13  },
14  {
15    "emp_id": 103,
16    "name": "sanket",
17    "dept": "it",
18    "designation": "ASE"
19  }
].
```

(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(name)
73 .then(data=>{
74         console.log(data);
75         res.status(200).send(data);
76     }).catch(err=>{
77         console.error("error is :" + err);
78     })
79 })
80 })

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

node - PostgreSQL

```

{ 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 JavaScript

Working locally in Scratch Pad. [Switch to a Workspace](#)

Import Overview GET http://localhost:80... ● GET https://fakestorea... ● + ⚙ No Environment

http://localhost:8001/getAllEmployeeById/101 [Save](#)

GET http://localhost:8001/getAllEmployeeById/101

Params Authorization Headers (8) Body [JSON](#) Pre-request Script Tests Settings

Body (8) Test Results Status: 200 OK Time: 57 ms Size: 337 B

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON [Copy](#)

```
1 {  
2   "Insurance_ID": 12,  
3   "policyName": "charan",  
4   "policyAmt": 123123,  
5   "M_amt": 123,  
6   "nominee": "nrk"  
7 }
```

```
1 {  
2   "emp_id": 101,  
3   "name": "navnath",  
4   "dept": "it",  
5   "designation": "Full stack"  
6 }
```

(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 - PostgreSQL

```
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 JavaScript

Working locally in Scratch Pad. [Switch to a Workspace](#)

Import Overview GET http://localhost:80... ● GET https://fakestorea... ● + ⚙ No Environment

http://localhost:8001/getAllEmployeeByName/navnath [Save](#)

GET [http://localhost:8001/getAllEmployeeByName/navnath](#)

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

none  form-data  x-www-form-urlencoded  raw  binary  GraphQL [JSON](#) ▾

```

1 {
2   "Insurance_ID": 12,
3   "policyName": "charan",
4   "policyAmt": 123123,
5   "M_amt": 123,
6   "nominee": "nrk"
7 }
```

Body Cookies Headers (8) Test Results Status: 200 OK Time: 63 ms Size: 337 B

Pretty Raw Preview Visualize [JSON](#) ▾

```

1 {
2   "emp_id": 101,
3   "name": "navnath",
4   "dept": "it",
5   "designation": "Full stack"
6 }
```

(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:dept});
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;
105
106     var updateObj = EmployeeTable.build({name:name, dept:dept, designation:designation});
107     updateObj.where("emp_id", emp_id).update();
108
109     res.status(200).send("Record updated successfully");
110 })
111
112 //delete the records by id
113 app.delete("/deleteData", (req, res) =>{
114     var emp_id = req.query.emp_id;
115
116     var deleteObj = EmployeeTable.build({emp_id:emp_id});
117     deleteObj.where("emp_id", emp_id).delete();
118
119     res.status(200).send("Record deleted successfully");
120 })
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL node - PostgreSQL

```
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" = '101'
{ emp_id: 101, name: 'navnath', dept: 'it', designation: 'Full stack' }
```

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

Working locally in Scratch Pad. [Switch to a Workspace](#)

Import Overview POST http://localhost:8... ● GET https://fakestorea... ● + ⚙ No Environment

http://localhost:8001/insertData [Save](#)

POST http://localhost:8001/insertData

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL **JSON** ▾

```
1 {"emp_id": 107,
2 ..... "name": "charan",
3 ..... "dept": "it",
4 ..... "designation": "Full stack"
5 }
```

Body Cookies Headers (8) Test Results Status: 201 Created Time: 65 ms Size: 284 B

Pretty Raw Preview Visualize HTML [Copy](#)

```
1 record is inserted
```

(e) put - updateEmployeeData() -- Takes an object and updates 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 JavaScript

Working locally in Scratch Pad. [Switch to a Workspace](#)

Import Overview PUT http://localhost:80... ● GET https://fakestorea... ● + ⚙ No Environment

http://localhost:8001/updateData [Save](#)

PUT http://localhost:8001/updateData

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL **JSON**

```
1 {"emp_id": 107,
2 ..... "name": "navnath",
3 ..... "dept": "it",
4 ..... "designation": "F-stack"
5 }
```

Body Cookies Headers (8) Test Results Status: 201 Created Time: 47 ms Size: 283 B

Pretty Raw Preview Visualize HTML

```
1 data updated.....
```

(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;
141     var UPD = req.body.UPD;
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL node - PostgreSQL
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 JavaScript
```

Working locally in Scratch Pad. [Switch to a Workspace](#)

Import Overview No Environment

http://localhost:8001/deleteData/107 [Save](#)

DELETE http://localhost:8001/deleteData/107

Params Authorization Headers (8) Body [JSON](#) Pre-request Script Tests Settings

none  form-data  x-www-form-urlencoded  raw  binary  GraphQL

```
1 {"emp_id": 107,
2 ..... "name": "navnath",
3 ..... "dept": "it",
4 ..... "designation": "F-stack"
5 }
```

Body Cookies Headers (8) Test Results Status: 200 OK Time: 48 ms Size: 283 B

Pretty Raw Preview Visualize HTML [Copy](#)

```
1 record is deleted now..
```

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

```

25 // sequelize.authenticate().then(()=>{
26 //     console.log("connected to the databases successfully");
27 // }).catch(err=>{
28 //     console.error("unable to connect"+err);
29 // }).finally(()=>{
30 //     sequelize.close();
31 // })
32 let InsuranceTable = sequelize.define('Insurance',{
33     Insurance_ID:{
34         primaryKey:true,
35         type:Sequelize.INTEGER
36     },
37     policyName:Sequelize.STRING,
38     policyAmt:Sequelize.INTEGER,
39     M_amt:Sequelize.INTEGER,
40     nominee:Sequelize.STRING
41 },{
42     timestamps:false,
43     freezeTableName:true
44 });
45
46 // InsuranceTable.sync().then(()=>{

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

[ node ]

```

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

(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      }).catch(err=>{
76          console.error("error is :"+err);
77          res.status(400).send(err);
78      })
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

policyName: 'q2eqw',  
policyAmt: 123123,  
M\_amt: 1212,  
nominee: 'asqas'  
}

The screenshot shows the Postman application interface. At the top, there are tabs for 'Import', 'Overview', and two other requests: 'GET http://localhost:80...' and 'GET https://fakestorea...'. A 'No Env' button is also present. Below the tabs, the current request is displayed: 'http://localhost:8000/insurance' with a 'GET' method. The 'Body' tab is selected, showing the following JSON payload:

```
1  {"uid": "11101", "pass": "passwords"}
```

Below the body, there are tabs for 'Params', 'Authorization', 'Headers (8)', 'Body', 'Pre-request Script', 'Tests', and 'Settings'. The 'Body' tab is currently active, showing the selected 'JSON' type. The 'Headers' tab shows 8 items. At the bottom of the request section, there are buttons for 'Body', 'Cookies', 'Headers (8)', and 'Test Results'. To the right, status information is displayed: 'Status: 200 OK', 'Time: 195 ms', and 'Size: 45'. The response body is shown in a 'Pretty' format:

```
1
2  {
3      "Insurance_ID": 11,
4      "policyName": "qweq",
5      "policyAmt": 123123,
6      "M_amt": 123,
7      "nominee": "aeqwe"
8  },
9  {
10     "Insurance_ID": 111,
11     "policyName": "q2eqw",
12     "policyAmt": 123123,
13     "M_amt": 1212,
14     "nominee": "asqas"
15 }
16
```

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

```
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
89      }).catch(err=>{
90          console.log(err);
91          res.status(200).send(err);
92      })
93  })
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Insurance_ID: 11,
policyName: 'qweq',
policyAmt: 123123,
M_amt: 123,
nominee: 'aeqwe'
}
```

The screenshot shows the Postman application interface. At the top, there are two tabs: "Import" and "Overview". Below the tabs, a header bar displays two requests: "GET http://localhost:80..." and "GET https://fakestorea...". A "+" button and a "ooo" link are also in the header. On the right, it says "No Envi".

The main area shows a request for "http://localhost:8000/insuranceByID/11" with a "GET" method. The "Body" tab is selected, showing the JSON payload:

```
1  {"uid": "11101", "pass": "passwords"}
```

Below the body, the status is shown as "Status: 400 Bad Request Time: 52 ms Size: 36". The "Pretty" tab is selected in the preview section, displaying the JSON response:

```
1  {
2      "Insurance_ID": 11,
3      "policyName": "qweq",
4      "policyAmt": 123123,
5      "M_amt": 123,
6      "nominee": "aeqwe"
7  }
```

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



The screenshot shows the Postman application interface. At the top, there are tabs for 'Import', 'Overview', and several other requests like 'POST http://localhost:8...', 'GET https://fakestorea...', and 'ooo'. A 'No Env' button is also present. Below the tabs, the URL 'http://localhost:8000/insertData' is selected. The main area shows a 'POST' method and the URL 'http://localhost:8000/insertData'. Below this, there are tabs for 'Params', 'Authorization', 'Headers (8)', 'Body' (which is currently selected), 'Pre-request Script', 'Tests', and 'Settings'. Under 'Body', there are options for 'none', 'form-data', 'x-www-form-urlencoded', 'raw' (which is selected), 'binary', 'GraphQL', and 'JSON' (with a dropdown arrow). The 'raw' section contains the following JSON payload:

```
1 {
2     "Insurance_ID": 12,
3     "policyName": "navnath",
4     "policyAmt": 123123,
5     "M_amnt": 123,
6     "nominee": "nrk"
7 }
```

Below the body, there are tabs for 'Body', 'Cookies', 'Headers (8)', and 'Test Results'. The 'Test Results' tab is selected, showing the response: 'Status: 201 Created Time: 48 ms Size: 28'. Under 'Test Results', there are buttons for 'Pretty', 'Raw', 'Preview', 'Visualize', and 'HTML' (which is selected). The 'Pretty' view shows the response message: '1 record is inserted'.

(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

The screenshot shows the Postman application interface. At the top, there are tabs for 'Import', 'Overview', and several other requests like 'PUT http://localhost:80...', 'GET https://fakestorea...', and 'ooo'. A 'No Env' button is also present. Below the tabs, a search bar contains the URL 'http://localhost:8000/updateData'. The main area shows a 'PUT' request with the URL 'http://localhost:8000/updateData'. The 'Body' tab is selected, showing a JSON payload:

```
1 {
2   "Insurance_ID": 12,
3   "policyName": "charan",
4   "policyAmt": 123123,
5   "M_amt": 123,
6   "nominee": "nrk"
7 }
```

Below the body, there are tabs for 'Body', 'Cookies', 'Headers (8)', and 'Test Results'. The 'Test Results' tab is selected, displaying the response: 'data updated.....'. At the bottom right, status information is shown: Status: 201 Created, Time: 55 ms, Size: 28.

(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... + ooo No Env

ooo [http://localhost:8000/deleteData/12](#) [Save](#)

**DELETE** v [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** v

```
1 {
2   "Insurance_ID": 12,
3   "policyName": "charan",
4   "policyAmt": 123123,
5   "M_amt": 123,
6   "nominee": "nrk"
7 }
```

Body Cookies Headers (8) Test Results Status: 200 OK Time: 56 ms Size: 28

Pretty Raw Preview Visualize HTML -

```
1 record is deleted now..
```

Working locally in scratch, add context to a Workspace

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 ▼ CSV

```
1 valid users
```

```
src/postgreAssignment/ass.js:1:1: app.get('/login', callback) > strMsg
9     var uid =req.body.uid;
0     var pass = req.body.pass;
1
2     fl = false;
3     const Op = Sequelize.Op;
4
5     usersTable.findAll({where:{[Op.and]:[{uid:uid},{pass:pass}]}},
6         raw:true,
7     ).then((data)=>{
8         console.log(data);
9         console.log(typeof data);
0         if(data){
1             fl =true;
2         }
3     });
4     if(fl){
5         var strMsg="valid user";
6         res.status(201).send(strMsg);
7     }else{
8         var strMsg ="invalid users enter correct details";
9         res.status(401).send(strMsg);
0     }
-
LEMS OUTPUT DEBUG CONSOLE TERMINAL
[mon] restarting due to changes...
[mon] starting `node ass.js`
er 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

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

Pretty Raw Preview Visualize HTML

record is inserted

Status: 201 Created Time: 168 ms Size: 251 B

The screenshot shows the Postman application interface. At the top, there's a header bar with the text 'Working locally in Scratch Pad. [Switch to a Workspace](#)' and some environment information. Below the header is a navigation bar with tabs for 'Overview', 'POST http://localhost:8...', 'GET https://fakestorea...', and other options. The main area shows a request for 'http://localhost:8000/register' with a 'POST' method selected. The 'Body' tab is active, showing a JSON payload: { "uid": "11101", "pass": "passwords" }. Below the body, there are tabs for 'Pretty', 'Raw', 'Preview', 'Visualize', and 'HTML'. The 'HTML' tab is currently selected. At the bottom of the request panel, it says 'record is inserted'. On the right side of the interface, there's a status summary: 'Status: 201 Created' with a timestamp of 'Time: 168 ms' and a size of 'Size: 251 B'. The overall layout is clean and modern, typical of a developer tool like Postman.

```
172  })
173  💡
174  app.post("/register", (req, res) => {
175    var uid = req.body.uid;
176    var pass = req.body.pass;
177
178    var userObj = usersTable.build({uid:uid,pass:pass});
179    userObj.save().then(data =>{
180      var stri = "record is inserted";
181      res.status(201).send(stri);
182    }).catch(err =>{
183      console.error("error is :" +err);
184      res.status(400).send(err);
185    })
186  })
187
188  app.listen(8000, () =>{
189    console.log("server is listening at 8000");
190  })
191
```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

[node - postgres]

```
[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";
[]
```