Name: Gaurav Kumar Singh

Registration Number: 19BCE2119

Course: Web Technologies

Digital Assignment 4

1)

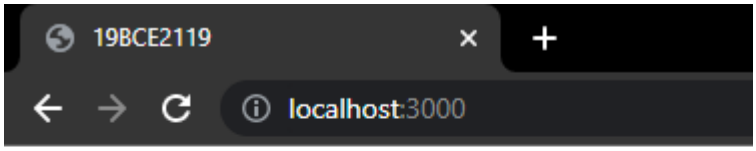
1.html

```
<!DOCTYPE html>
<html lang="en">

<head>
   <meta charset="utf-8">
   <title>19BCE2119</title>
</head>

<body>
   Name: Gaurav Kumar Singh <br>
   Reg no. 19BCE2119
   Ph no. 9664395951 <br>
   Credits Earned: 88 <br>
</body>

</html>
```

1.js

```
var http = require('http');
var fs = require('fs');
http.createServer(function (req, res) {
   fs.readFile('1.html', function (err, data) {
      res.writeHead(200, {
         'ContentType': 'text/html'
      });
```

```
        res.write(data);
        return res.end();
    });
}).listen(3000);
```

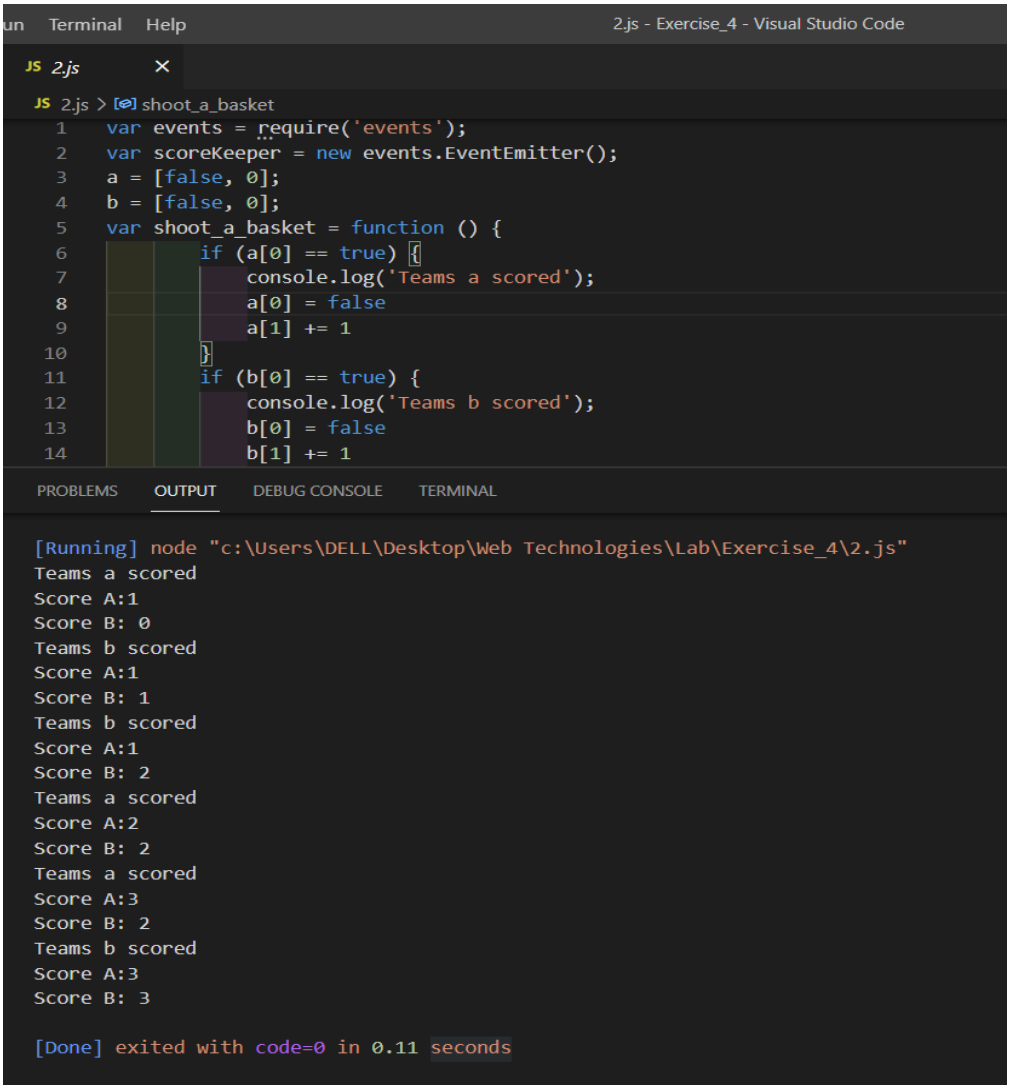


Name: Gaurav Kumar Singh
Reg no. 19BCE2119 Ph no. 9664395951
Credits Earned: 88

2)

```
var events = require('events');
var scoreKeeper = new events.EventEmitter();
a = [false, 0];
b = [false, 0];
var shoot_a_basket = function () {
    if (a[0] == true) {
        console.log('Teams a scored');
        a[0] = false
        a[1] += 1
    }
    if (b[0] == true) {
        console.log('Teams b scored');
        b[0] = false
        b[1] += 1
    }
    console.log('Score A:' + a[1]);
    console.log('Score B: '+b[1]);
}
```

```
scoreKeeper.on('make_basket',shoot_a_basket);

a[0]=true;

scoreKeeper.emit('make_basket');

b[0]=true;

scoreKeeper.emit('make_basket');

b[0]=true;

scoreKeeper.emit('make_basket');

a[0]=true;

scoreKeeper.emit('make_basket');

a[0]=true;

scoreKeeper.emit('make_basket');

b[0]=true;

scoreKeeper.emit('make_basket');
```



3)

3. use Student

```
Command Prompt - mongo
Microsoft Windows [Version 10.0.19043.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\DELL>mongo
MongoDB shell version v4.0.25
connecting to: mongodb://127.0.0.1:27017/?gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("8ba120fb-cee7-499b-b95b-d6d4880479b9") }
MongoDB server version: 4.0.25
Server has startup warnings:
2021-11-19T11:17:37.368+0530 I CONTROL  [initandlisten]
2021-11-19T11:17:37.368+0530 I CONTROL  [initandlisten] ** WARNING: Access control is not enabled for the database.
2021-11-19T11:17:37.368+0530 I CONTROL  [initandlisten] **          Read and write access to data and configuration is unrestricted.
2021-11-19T11:17:37.368+0530 I CONTROL  [initandlisten]
---
Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---

> use Student
switched to db Student
>
```

a) db.student.insertOne({name:"Dora"})

```
> db.student.insertOne({name:"Dora"})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("61a5c83e581019c773ef3ae3")
}
> db.student.find()
{ "_id" : ObjectId("61a5c83e581019c773ef3ae3"), "name" : "Dora" }
>
```

b) db.student.insertOne({_id:2,name:"Sinchan"})

```
> db.student.insertOne({_id:2,name:"Sinchan"})
{ "acknowledged" : true, "insertedId" : 2 }
> db.student.find()
{ "_id" : ObjectId("61a5c83e581019c773ef3ae3"), "name" : "Dora" }
{ "_id" : 2, "name" : "Sinchan" }
>
```

c) db.student.insertOne({name:"Angush", scores:{midtermScore:80, finalScore:100}})

```
> db.student.insertOne({name:"Angush", scores:{midtermScore:80, finalScore:100}})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("61a5ca6f581019c773ef3ae5")
}
> db.student.find()
{ "_id" : ObjectId("61a5c83e581019c773ef3ae3"), "name" : "Dora" }
{ "_id" : 2, "name" : "Sinchan" }
{ "_id" : ObjectId("61a5c971581019c773ef3ae4"), "name" : "Sinchan", "scores" : { "midtermScore" : 80, "finalScore" : 100 } }
{ "_id" : ObjectId("61a5ca6f581019c773ef3ae5"), "name" : "Angush", "scores" : { "midtermScore" : 80, "finalScore" : 100 } }
>
```

d) db.student.insertOne({name:"Gaurav Kumar Singh"})
db.student.find({name:"Gaurav Kumar Singh"})

```
> db.student.insertOne({name:"Gaurav Kumar Singh"})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("61a5cbe9581019c773ef3ae6")
}
> db.student.find({name:"Gaurav Kumar Singh"})
{ "_id" : ObjectId("61a5cbe9581019c773ef3ae6"), "name" : "Gaurav Kumar Singh" }
>
```

e) db.student.find({"scores.midtermScore":{$gt:50}})

```
> db.student.find({"scores.midtermScore":{$gt:50}})
{ "_id" : ObjectId("61a5c971581019c773ef3ae4"), "name" : "Sinchan", "scores" : { "midtermScore" : 80, "finalScore" : 100 } }
{ "_id" : ObjectId("61a5ca6f581019c773ef3ae5"), "name" : "Angush", "scores" : { "midtermScore" : 80, "finalScore" : 100 } }
>
```

f) db.student.find({"scores.midtermScore":{$gt:50, $lte:80}, "scores.finalScore": {$gt:80,$lte:100}})

```
> db.student.find({"scores.midtermScore":{$gt:50, $lte:80}, "scores.finalScore": {$gt:80,$lte:100}})
{ "_id" : ObjectId("61a5c971581019c773ef3ae4"), "name" : "Sinchan", "scores" : { "midtermScore" : 80, "finalScore" : 100 } }
{ "_id" : ObjectId("61a5ca6f581019c773ef3ae5"), "name" : "Angush", "scores" : { "midtermScore" : 80, "finalScore" : 100 } }
>
```

g) db.student.update({name:"Sinchan"}, {$set: {scores:{midtermScores: 50, finalScores:100}}})

```
> db.student.update({name:"Sinchan"}, {$set: {scores:{midtermScores: 50, finalScores:100}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.student.find({name:"Sinchan"})
{ "_id" : ObjectId("61a5c971581019c773ef3ae4"), "name" : "Sinchan", "scores" : { "midtermScores" : 50, "finalScores" : 100 } }
>
```

h) db.student.find({},{name:1,scores:1,_id:0}).sort({"scores.finalScores":-1})

```
> db.student.find({},{name:1,scores:1,_id:0}).sort({"scores.finalScores":-1})
{ "name" : "Sinchan", "scores" : { "midtermScores" : 50, "finalScores" : 100 } }
{ "name" : "Dora" }
{ "name" : "Angush", "scores" : { "midtermScore" : 80, "finalScore" : 100 } }
{ "name" : "Gaurav Kumar Singh" }
>
```

i) db.student.findOneAndDelete({name:"Angush"})

```
> db.student.findOneAndDelete({name:"Angush"})
{
        "_id" : ObjectId("61a5ca6f581019c773ef3ae5"),
        "name" : "Angush",
        "scores" : {
                "midtermScore" : 80,
                "finalScore" : 100
        }
}
>
```

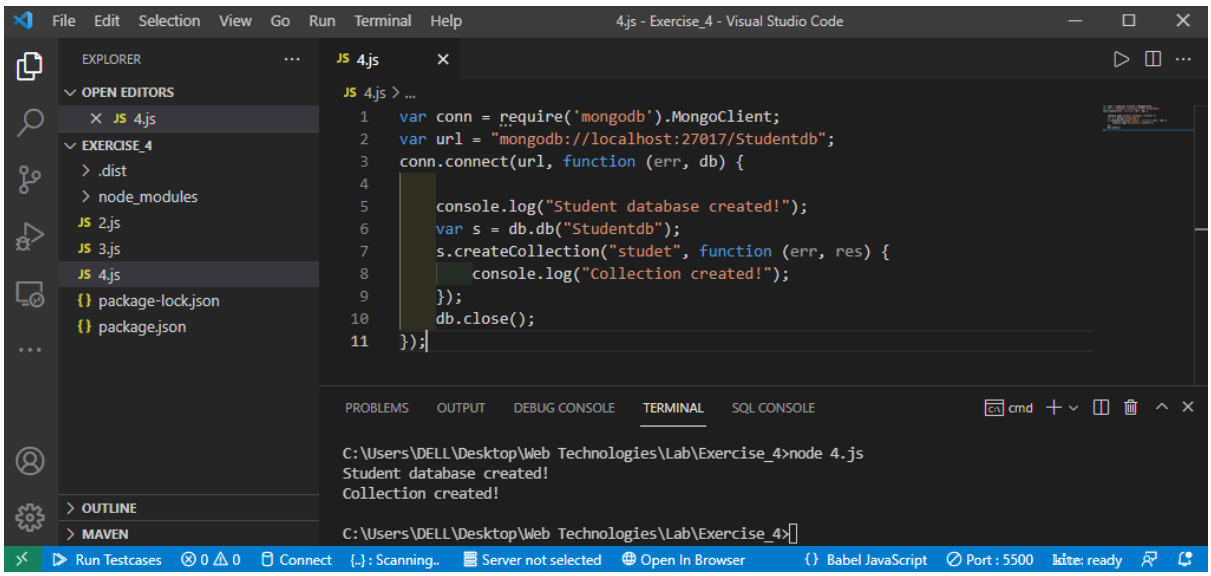
j) db.student.deleteMany({"scores.midtermScores":{$lte:80}})

```
> db.student.deleteMany({"scores.midtermScores":{$lte:80}})
{ "acknowledged" : true, "deletedCount" : 1 }
> db.student.find()
{ "_id" : ObjectId("61a5c83e581019c773ef3ae3"), "name" : "Dora" }
{ "_id" : ObjectId("61a5cbe9581019c773ef3ae6"), "name" : "Gaurav Kumar Singh" }
>
```

4)

i)

```
var conn = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/Studentdb";
conn.connect(url, function (err, db) {

  console.log("Student database created!");
  var s = db.db("Studentdb");
  s.createCollection("studet", function (err, res) {
    console.log("Collection created!");
  });
  db.close();
});
```

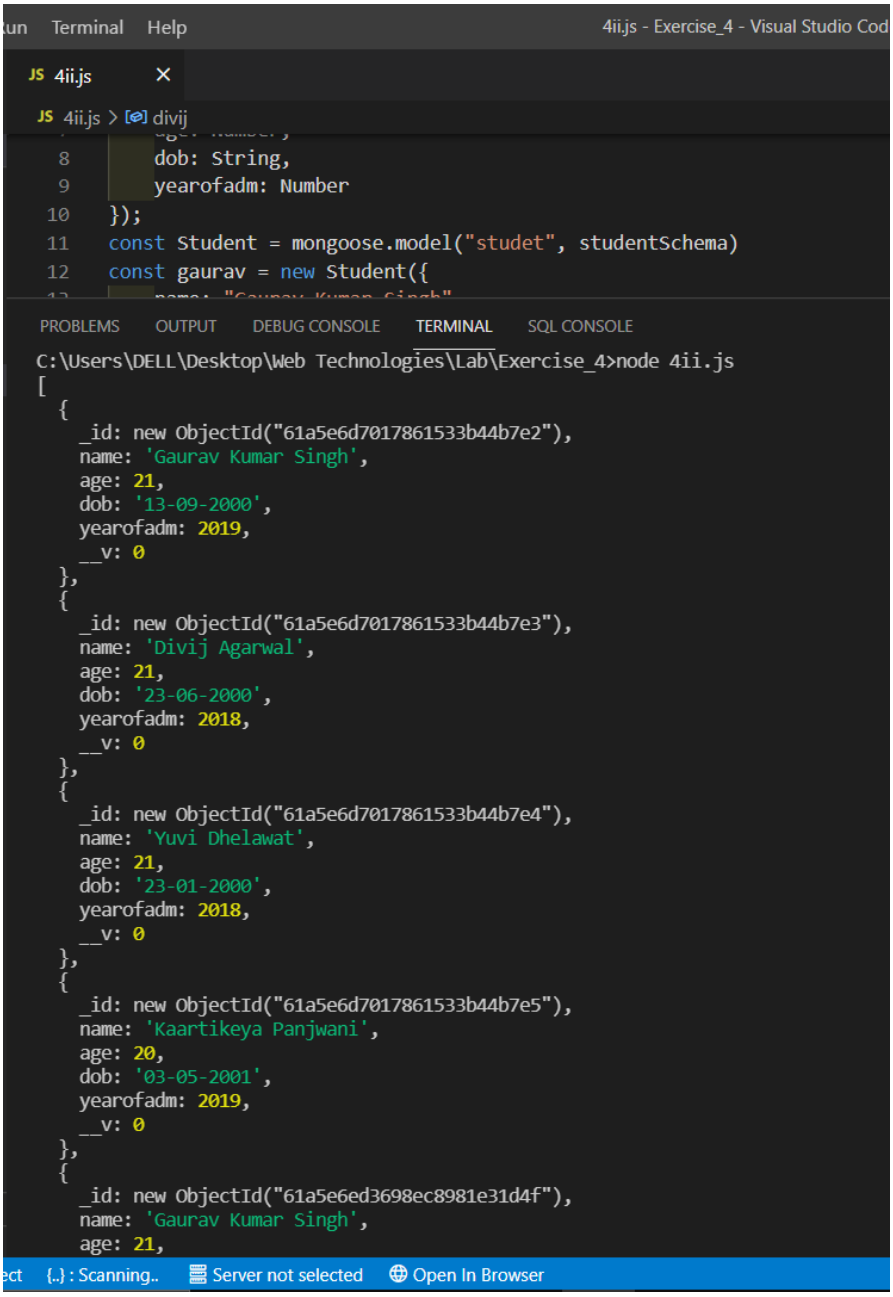


ii)

```
const mongoose = require("mongoose");
mongoose.connect("mongodb://localhost:27017/Studentdb", {
```

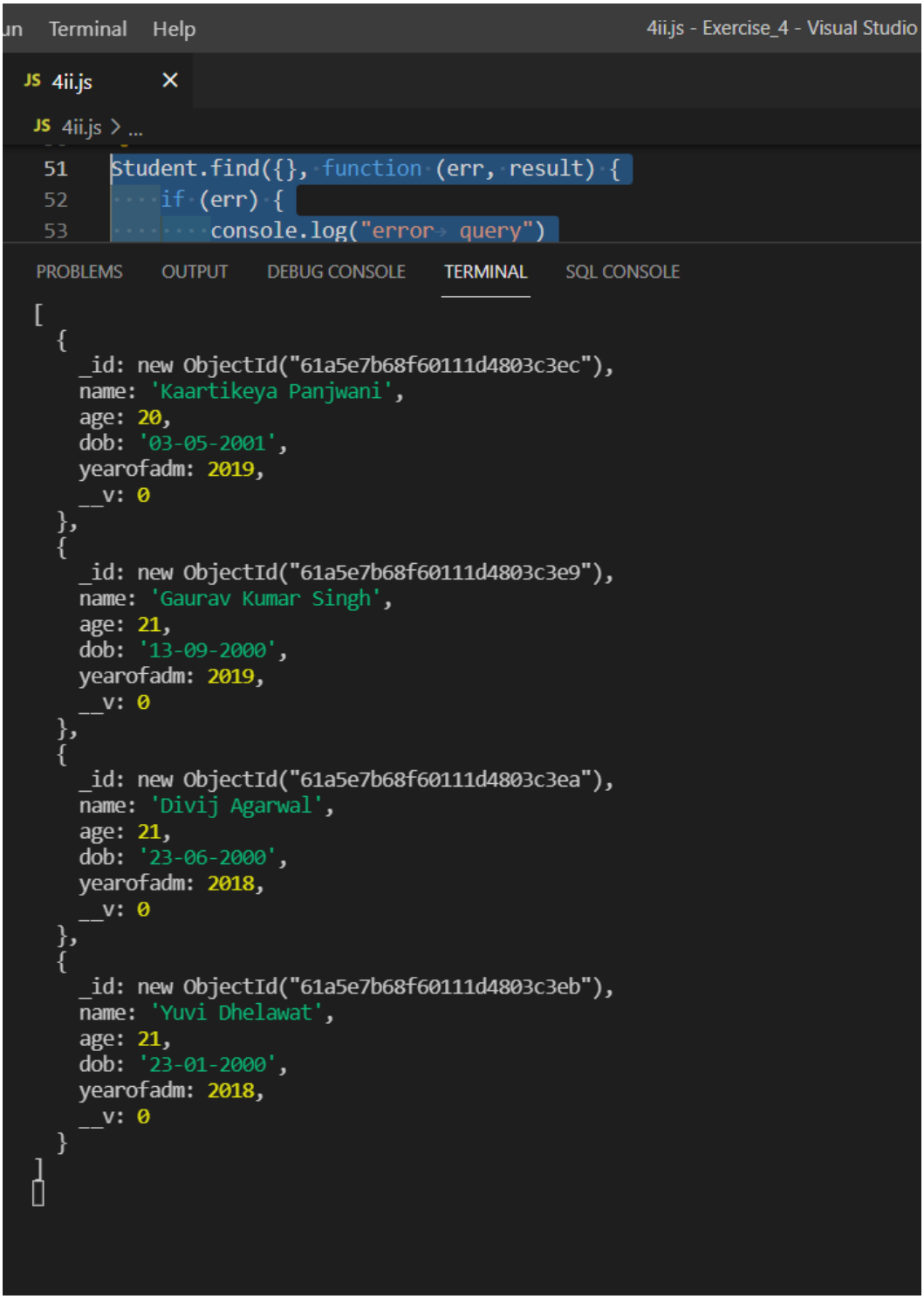
```
    useNewUrlParser: true
});
const studentSchema = new mongoose.Schema({
    name: String,
    age: Number,
    dob: String,
    yearofadm: Number
});
const Student = mongoose.model("studet", studentSchema)
const gaurav = new Student({
    name: "Gaurav Kumar Singh",
    age: 21,
    dob: "13-09-2000",
    yearofadm: 2019
});
const divij = new Student({
    name: "Divij Agarwal",
    age: 21,
    dob: "23-06-2000",
    yearofadm: 2018
});
const yuvi = new Student({
    name: "Yuvi Dhelawat",
    age: 21,
    dob: "23-01-2000",
    yearofadm: 2018
});
const kaartikeya = new Student({
    name: "Kaartikeya Panjwani",
    age: 20,
    dob: "03-05-2001",
```

```
    yearofadm: 2019
});
Student.insertMany([gaurav, divij, yuvi, kaartikeya], function (err) {
    if (err) {
        console.log(err);
    } else {
        console.log("successfully added student details to database")
    }
});
Student.find({},function (err, student) {
    if (err) {
        console.log(err)
    } else {
        console.log(student)
    }
});
```



iii)

```
Student.find({}, function (err, result) {
   if (err) {
      console.log("error  query")
   } else {
      console.log(result)
   }
}).sort({age:1})
```



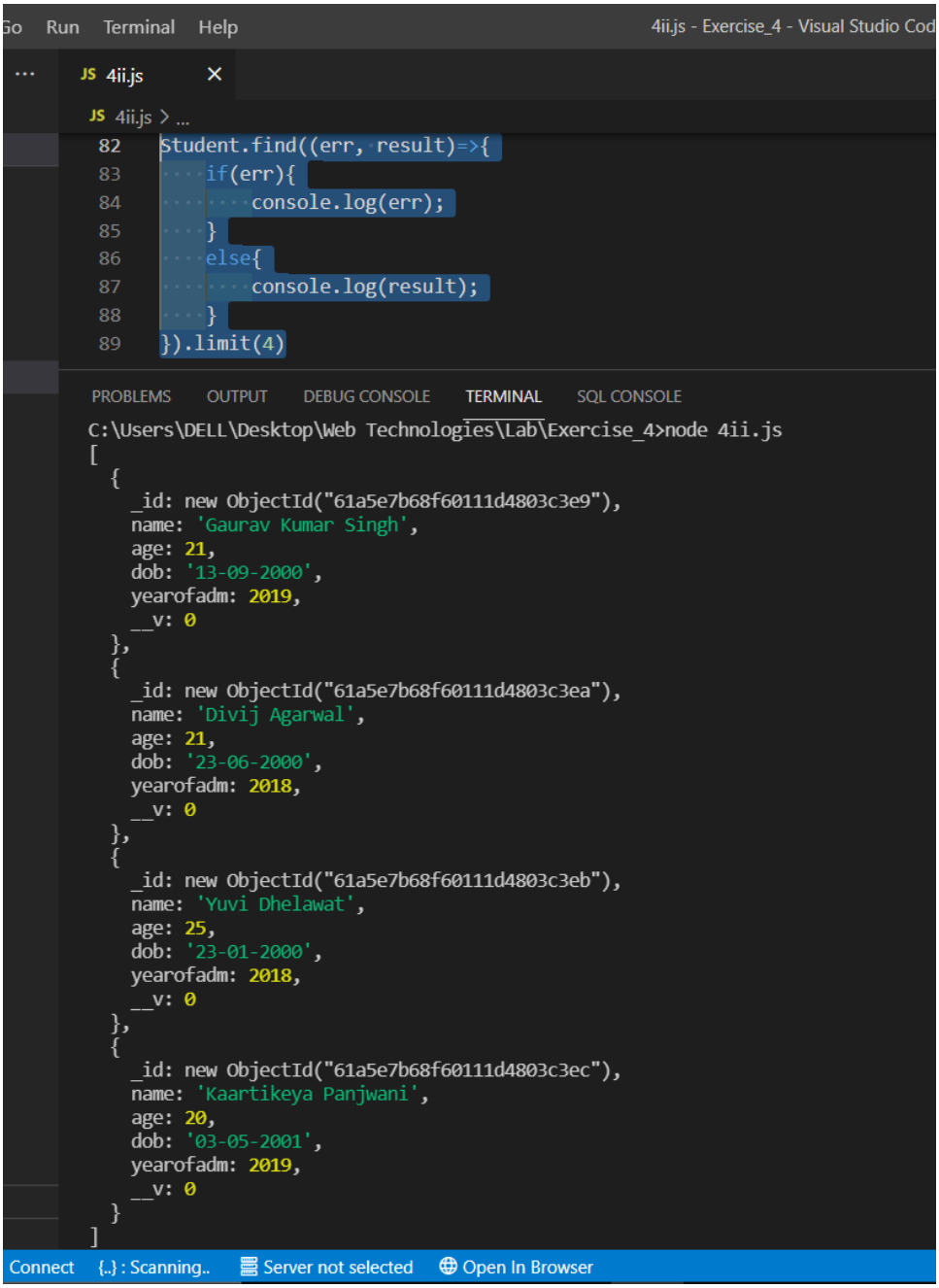
iv)

```
Student.find({name: "Yuvi Dhelawat"}, function (err, result) {
    if (err) {
        console.log("error  query")
    } else {
        console.log("BEFORE UPDATE")
```

```
        console.log(result)
    }
})
Student.updateOne({name: "Yuvi Dhelawat"},{age:25}, function(err){
    if(err){
        console.log(err)
    }
    else{
        console.log('Update Successful')
    }
})
Student.find({name: "Yuvi Dhelawat"}, function (err, result) {
    if (err) {
        console.log("error  query")
    } else {
        console.log("AFTER UPDATE")
        console.log(result)
    }
})
```

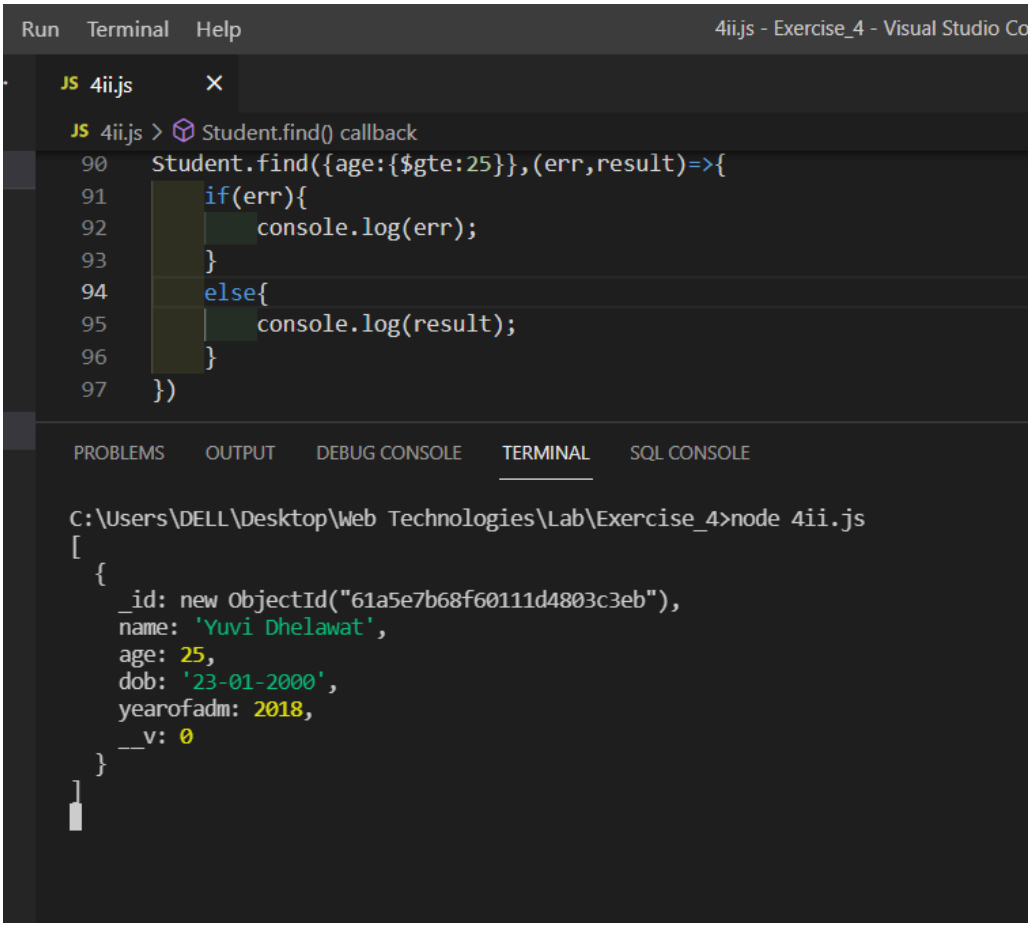
v)

```
Student.find((err, result)=>{
   if(err){
      console.log(err);
   }
   else{
      console.log(result);
   }
}).limit(4)
```



vi)

```
Student.find({age:{$gte:25}},(err,result)=>{
   if(err){
      console.log(err);
   }
   else{
      console.log(result);
   }
})
```



5)

5.js

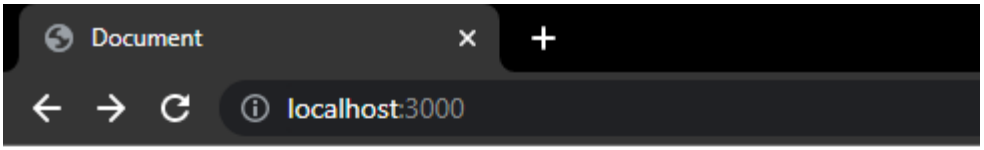
```
const express = require("express")
const app = express()
const bodyParser = require("body-parser");
app.use(bodyParser.urlencoded({
   extended: true
}));
var port=3000;
app.get('/', function (req, res) {
   res.sendFile(__dirname + '/index.html');
});
app.get('/get', function (req, res) {
   var name = req.query.username;
```

```
    res.send(name + ' Get Request ');
});
app.post('/post', function (req, res) {
    var name = req.body.username;
    res.send(name + ' Post Request ');
});
app.put('/put/:username', function (req, res) {
    var name = req.params.username;
    res.send(name + ' Put Request ');
});
app.delete('/delete/:username', function (req, res) {
    var name = req.params.username;
    res.send(name + ' Delete Request ');
});
app.listen(port, function () {
    console.log("Server Sucessfully Started");
})
```

index.html

```
<!DOCTYPE html>
<html>

<head>
    <meta charset="utf-8" />
    <title>Document</title>
</head>

<body>
    <h1>Post</h1>
    <form action="/post" method="post">
        Username: <input name="username" type="text" /> <br />
```

```
    <input type="submit" value="POST" />
  </form>
  <h1>Get</h1>

  <form action="/get" method="get">
    Username: <input name="username" type="text" /> <br />
    <input type="submit" value="GET" />
  </form>

  <p>PUT :- <a href="http://127.0.0.1:3000/put/username">http://127.0.0.1:3000/put/username</a></p>
  <p>DELETE :- <a href="http://127.0.0.1:3000/delete/username">http://127.0.0.1:3000/delete/username</a></p>
</body>
</html>
```



Post

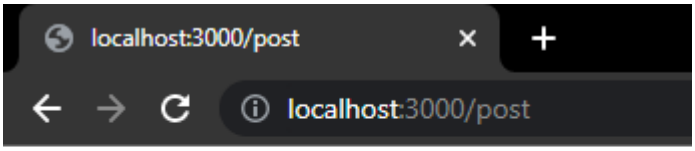
Username: 19BCE2119
POST

Get

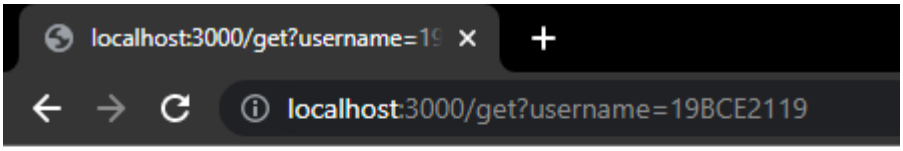
Username: 19BCE2119
GET

PUT :- http://127.0.0.1:3000/put/username

DELETE :- http://127.0.0.1:3000/delete/username



19BCE2119 Post Request



19BCE2119 Get Request

6)

6.js

```
var express = require("express");
var bodyParser = require("body-parser");
const mongoose = require('mongoose');
mongoose.connect('mongodb://localhost:27017/sessioninfo ', {
   useNewUrlParser: true
});
var app = express();
var count = 2;
app.use(bodyParser.json());
app.use(express.static('public'));
app.use(bodyParser.urlencoded({
   extended: true
}));
const user = new mongoose.Schema({
   username: String,
   password: String
```

```
})
const User = mongoose.model("User", user)
app.post("/signup", function (req, res) {
    var username = req.body.username;
    var password = req.body.password;
    var checkbox = req.body.remember;
    if (checkbox === "yes") {
        let data = new User({
            username: username,
            password: password
        });
        data.save();
        res.send("Registered successfully!!")
    }
});
app.post("/welcome", function (req, res) {
    var username = req.body.username;
    var password = req.body.password;
    if (count !== 0) {
        User.find({
            username: username,
            password: password
        }, function (err, result) {
            if (err) {
                console.log(err);
            } else {
                if (result.length == 0) {
                    count = count - 1;
                    res.send("Incorrect username or password")
                } else {
                    count = 2;
```

```
                res.send("Welcome")
            }
        }
    })
} else {
    res.send("You are blocked!")
}
})
app.get("/", function (req, res) {
    res.sendFile(__dirname + "/index.html")
})
app.get("/login", function (req, res) {
    res.sendFile(__dirname + "/login.html")
})
app.listen(3000, function () {
    console.log("Server Started");
})
```

Index.html

```
<!DOCTYPE html>
<html lang="en" dir="ltr">

<head>
    <meta charset="utf-8">
    <title>Register</title>
</head>

<body>
    <h1>Sign-Up (Enter Credentials)</h1>
    <form action="/signup" method="post">
        <p>Username:</p>
        <input type="text" name="username" value="">
```

```
        <p>Password:</p>
        <input type="password" name="password" value=""> <br>
        <input type="checkbox" name="remember" value="yes">
        Remember Me
        <input type="submit" name="" value="submit">
    </form>
    <a href="/login">Log-In</a>
</body>
</html>
```

Login.html

```
<!DOCTYPE html>
<html lang="en" dir="ltr">

<head>
    <meta charset="utf-8">
    <title>Login</title>
</head>

<body>
    <h1>Log In</h1>
    <form action="/welcome" method="post">
        <p>Username:</p>
        <input type="text" name="username" value="">
        <p>Password:</p>
        <input type="password" name="password" value=""> <br>
        <input type="submit" name="" value="submit">
    </form>
</body>
</html>
```

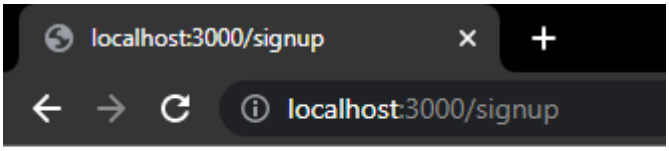
Sign-Up (Enter Credentials)

Username:

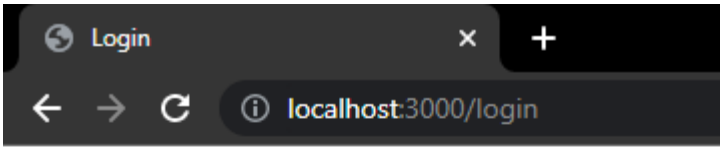
Password:

Remember Me submit

Log-In



Registered successfully!!

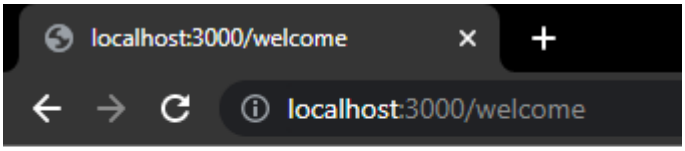


Log In

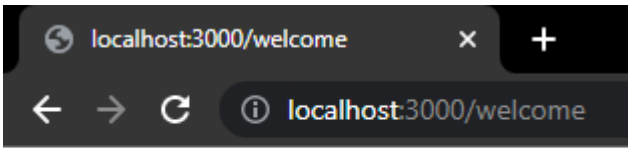
Username:

Password:

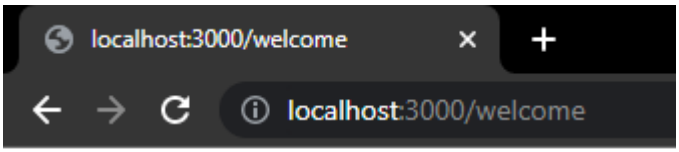
submit



Welcome



Incorrect username or password



You are blocked!

7)

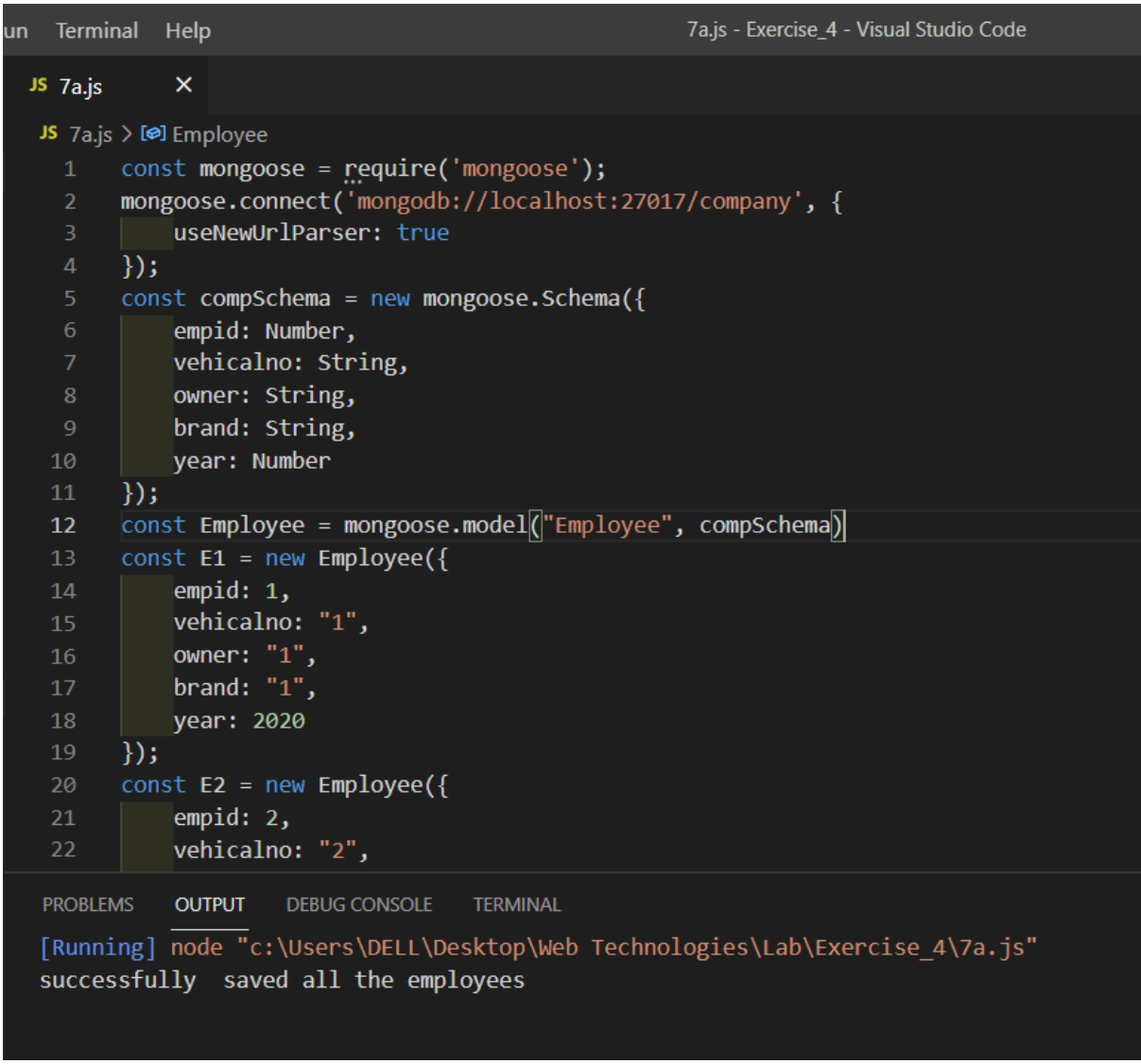
a)

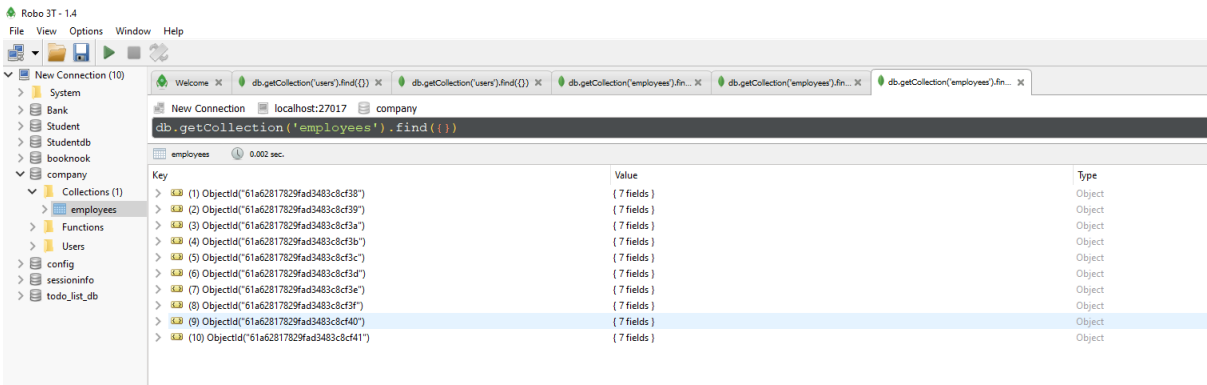
```
const mongoose = require('mongoose');
mongoose.connect('mongodb://localhost:27017/company', {
   useNewUrlParser: true
});
const compSchema = new mongoose.Schema({
   empid: Number,
   vehicalno: String,
   owner: String,
   brand: String,
   year: Number
});
const Employee = mongoose.model("Employee", compSchema)
const E1 = new Employee({
   empid: 1,
   vehicalno: "1",
   owner: "1",
```

```
    brand: "1",
    year: 2020
});
const E2 = new Employee({
    empid: 2,
    vehicalno: "2",
    owner: "2",
    brand: "2",
    year: 2020
});
const E3= new Employee({
    empid: 3,
    vehicalno: "3",
    owner: "3",
    brand: "3",
    year: 2020
});
const E4= new Employee({
    empid: 4,
    vehicalno: "4",
    owner: "4",
    brand: "4",
    year: 2020
});
const E5= new Employee({
    empid: 5,
    vehicalno: "5",
    owner: "5",
    brand: "5",
    year: 2020
});
```

```
const E6= new Employee({
    empid: 6,
    vehicalno: "6",
    owner: "6",
    brand: "6",
    year: 2020
});
const E7= new Employee({
    empid: 7,
    vehicalno: "7",
    owner: "7",
    brand: "7",
    year: 2020
});
const E8= new Employee({
    empid: 8,
    vehicalno: "8",
    owner: "8",
    brand: "8",
    year: 2020
});
const E9= new Employee({
    empid: 9,
    vehicalno: "9",
    owner: "9",
    brand: "9",
    year: 2020
});
const E10= new Employee({
    empid: 10,
    vehicalno: "10",
```

```
    owner: "10",
    brand: "10",
    year: 2020
});

Employee.insertMany([E1, E2, E3, E4, E5, E6, E7, E8, E9, E10],
    function (err) {
        if (err) {
            console.log(err);
        } else {
            console.log("successfully   saved   all the employees")
        }
    });
```





7b.js

```
var express = require("express");

var bodyParser = require("body-parser");

const mongoose = require('mongoose');

mongoose.connect('mongodb://localhost:27017/company', {
```

```
    useNewUrlParser: true
});
var app = express()
app.use(bodyParser.json());
app.use(express.static('public'));
app.use(bodyParser.urlencoded({
    extended: true
}));
const compSchema = new mongoose.Schema({
    empid: Number,
    vehicalno: String,
    owner: String,
    brand: String,
    year: Number
});
const Employee = mongoose.model("Employee", compSchema)
app.post('/confirm', function (req, res) {
    var empid = req.body.empid;
    var vehno = req.body.vehno;
    var owner = req.body.owner;
    var brand = req.body.brand;
    var year = req.body.year;
    Employee.findOneAndUpdate({
        empid: empid
    }, {
        vehicalno: vehno,
        owner: owner,
        brand: brand,
        year: year
    }, function (err) {
        if (err) {
```

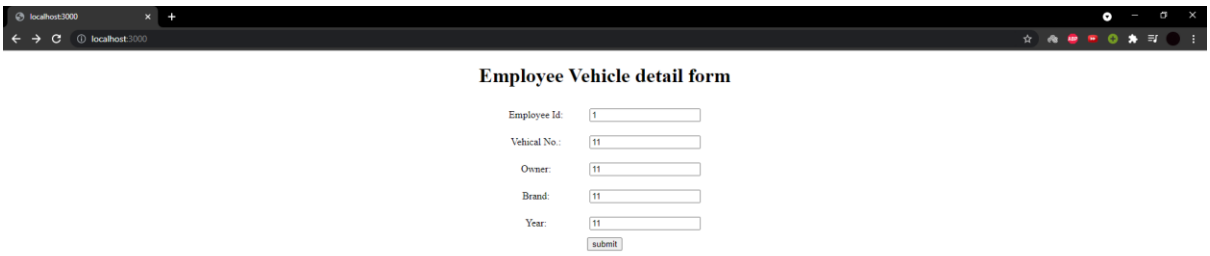
```
            console.log(err);
        } else {
            res.send("Updation  completed");
        }
    })
})
app.get("/", function (req, res) {
    res.sendFile(__dirname + "/index.html")
})
app.listen(3000, function () {
    console.log("server has started on 3000");
})
```

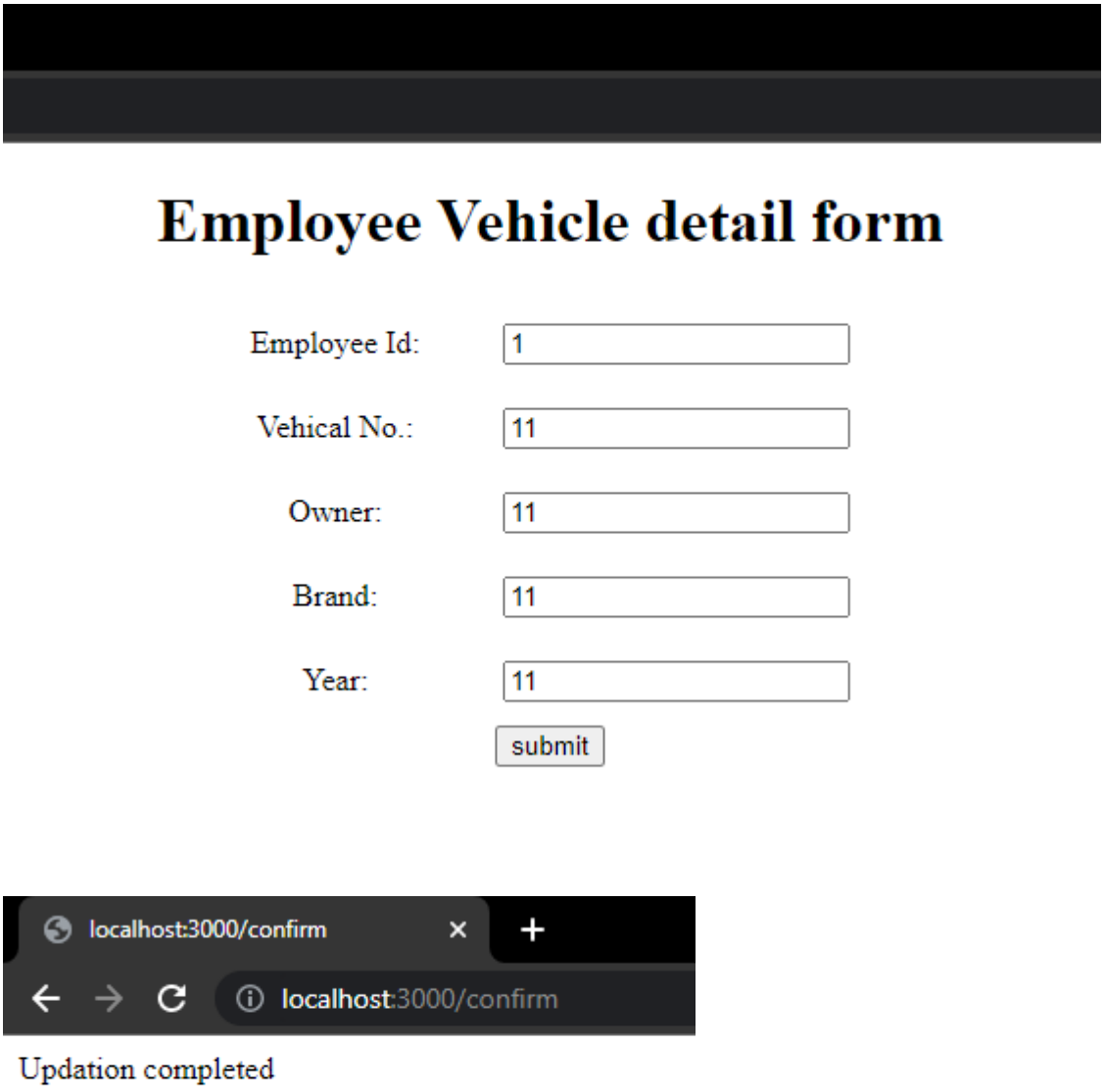
Index.html

```
<!DOCTYPE html>
<html lang="en">

<head>
    <style media="screen">
        body {
            text-align: center;
            align-items: center;
            justify-content: center;
        }

        form {
            text-align: center;
            align-items: center;
            justify-content: center;
        }

        table {
```

```
        margin: auto;
      }

      td {
        padding: 10px 20px;
      }

      #submit-button {
        padding: 5px;
        background-color: lightblue;
        margin: 20px;
      }
    </style>
</head>

<body>
    <h1>Employee Vehicle detail form</h1>
    <form action="/confirm" method="post">
      <table>
        <tr>
          <td>Employee Id:</td>
          <td><input type="number" name="empid" value=""></td>
        </tr>
        <tr>
          <td>Vehical No.:</td>
          <td><input type="text" name="vehno" value=""></td>
        </tr>
        <tr>
          <td>Owner:</td>
          <td><input type="text" name="owner" value=""></td>
        </tr>
```

```
            <tr>
              <td>Brand:</td>
              <td><input type="text" name="brand" value=""></td>
            </tr>
            <tr>
              <td>Year:</td>
              <td><input type="number" name="year" value=""></td>
            </tr>
          </table>
          <input type="submit" name="" value="submit" id="submit-bu&on">
        </form>
    </body>


    </html>
```





8)

8.js

```
var express = require("express");
var bodyParser = require("body-parser");
const mongoose = require('mongoose');

mongoose.connect('mongodb://localhost:27017/IPL', {
   useNewUrlParser: true
});
var app = express();
app.use(bodyParser.json());
```

```
app.use(express.static('public'));
app.use(bodyParser.urlencoded({
    extended: true
}));
const players = new mongoose.Schema({
    name: String,
    ipf: String,
    country: String,
    bid: Number
})
const Players = mongoose.model("players", players);
app.get("/",(req, res)=>{
    res.sendFile(__dirname + "/8.html");
})
app.post("/filter",(req,res)=>{
    var find= req.body.find;
    var find_details= req.body.find_details;
    if(find_details==""){
        Players.find({name:find},(err,result)=>{
            return res.send(JSON.stringify(result),null,3)
        })
    }
    else if(find_details=="ipl"){
        Players.find({ipf:find},(err,result)=>{
            return res.send(JSON.stringify(result),null,3)
        })
    }
    else if(find_details=="country"){
        Players.find({country:find},(err,result)=>{
            return res.send(JSON.stringify(result),null,3)
        })
```

```
    }
    else{
        Players.find({bid:{$gte:find}},(err,result)=>{
            return res.send(JSON.stringify(result),null,3)
        })
    }

})
app.listen(3000, function () {
    console.log("Server Started");
})
```

8.html

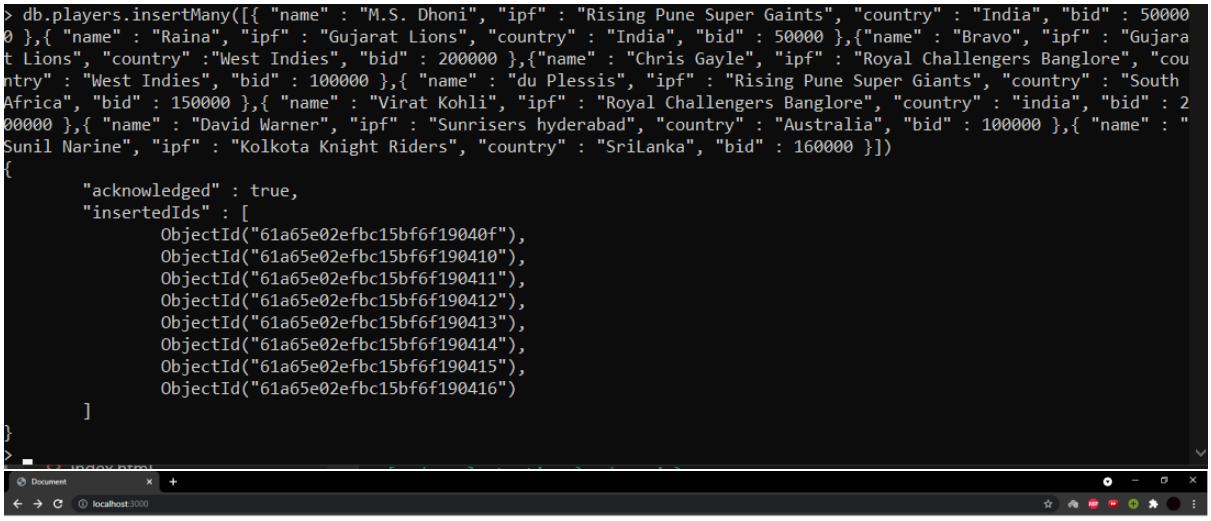
```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
rel="stylesheet"
        integrity="sha384-
1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
crossorigin="anonymous">
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"
        integrity="sha384-
ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p"
crossorigin="anonymous">
    </script>
    <script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.10.2/dist/umd/popper.min.js"
```

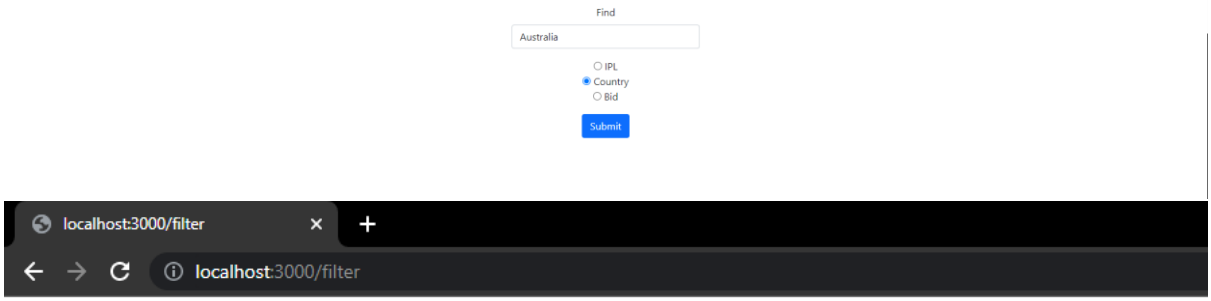
```
    integrity="sha384-
7+zCNj/IqJ95wo16oMtfsKbZ9ccEh31eOz1HGyDuCQ6wgnyJNSYdrPa03rtR1zdB"
crossorigin="anonymous">
  </script>
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.min.js"
    integrity="sha384-
QJHtvGhmr9XOIpI6YVutG+2QOK9T+ZnN4kzFN1RtK3zEFEIsxhlmWl5/YESvpZ13"
crossorigin="anonymous">
  </script>
  <style>
    body {
      text-align: center;
    }
    form {
      margin-top:100px;
      display: inline-block;
    }
  </style>
</head>

<body>

  <form action="/filter" method=post>
    <div class="mb-3">
      <label for=find class="form-label">Find</label>
      <input name=find id=find class="form-control" style="width:300px" required>
    </div>
    <div class=mb-3 form-check>
      <input type=radio name=find_details value=ipl> IPL<br>
      <input type=radio name=find_details value=country> Country<br>
      <input type=radio name=find_details value=bid> Bid<br>
    </div>
```

```
    <input class="btn btn-primary" type=submit>
  </form>
</body>


</html>
```

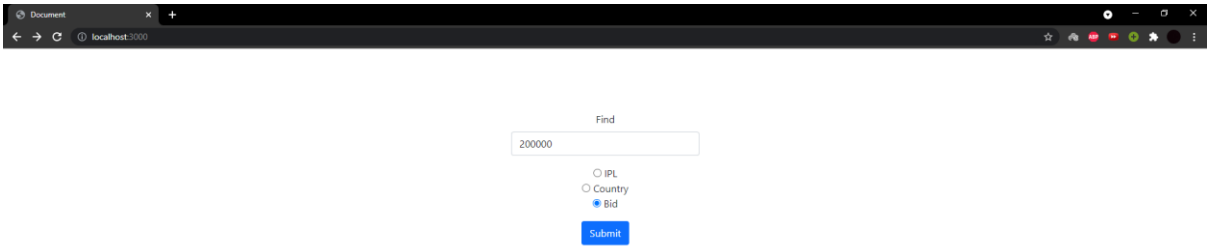
INSERT QUERY

```
db.players.insertMany([{ "name" : "M.S. Dhoni", "ipf" : "Rising Pune Super Gaints", "country" : "India", "bid" : 500000 },{ "name" : "Raina", "ipf" : "Gujarat Lions", "country" : "India", "bid" : 50000 },{"name" : "Bravo", "ipf" : "Gujarat Lions", "country" :"West Indies", "bid" : 200000 },{"name" : "Chris Gayle", "ipf" : "Royal Challengers Banglore", "country" : "West Indies", "bid" : 100000 },{ "name" : "du Plessis", "ipf" : "Rising Pune Super Giants", "country" : "South Africa", "bid" : 150000 },{ "name" : "Virat Kohli", "ipf" : "Royal Challengers Banglore", "country" : "india", "bid" : 200000 },{ "name" : "David Warner", "ipf" : "Sunrisers hyderabad", "country" : "Australia", "bid" : 100000 },{ "name" : "Sunil Narine", "ipf" : "Kolkota Knight Riders", "country" : "SriLanka", "bid" : 160000 }])
```





[{"_id":"61a65e02efbc15bf6f190415","name":"David Warner","ipf":"Sunrisers hyderabad","country":"Australia","bid":100000}]



[{"_id":"61a65e02efbc15bf6f19040f","name":"M.S. Dhoni","ipl":"Rising Pune Super Giants","country":"India","bid":500000},{"_id":"61a65e02efbc15bf6f190411","name":"Bravo","ipl":"Gujarat Lions","country":"West Indies","bid":200000},{"_id":"61a65e02efbc15bf6f190414","name":"Virat Kohli","ipl":"Royal Challengers Banglore","country":"india","bid":200000}]