

UCS1601-INTERNET PROGRAMMING

ASSIGNMENT:8

NAME: V SREEDHAR

DATE :27.04.2021

REGNO: 185001161

Exercise 8: Programs using Node.js

a. Write a Node.js program that reads all the greetings from the file greetings.txt, asks the user "What is your name?", then prints a random greeting followed by the given name. Make sure to check for the case where the file doesn't exist! For example, if the greeting is "Hey", then the program will print "Hey, Joe" to the console, then pick some other greeting and do the same until finished. Use Non-blocking I/O.

Program:

(Node file greet_random.js)

```
var fs=require('fs');
const readline = require("readline");
const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout
});
rl.question("What is your Name ? ", function(name) {
  fs.readFile('./greet.txt', function(err,data){
    if(err) return console.log("File not Found");
    data+='';
    var x=data.split(/\r?\n/)
    var greet = x[Math.floor(Math.random()*x.length)]
    console.log(greet,name);
    rl.close();
  });
});
```

OUTPUT DEBUG CONSOLE TERMINAL

1: powershell

```
PS G:\Academics\SSN\6th Sem\IP Lab\Node\ip> node greet_random.js
What is your Name ? Sree
Welcome Sree
PS G:\Academics\SSN\6th Sem\IP Lab\Node\ip> node greet_random.js
What is your Name ? Sree
Hey there Sree
PS G:\Academics\SSN\6th Sem\IP Lab\Node\ip> node greet_random.js
What is your Name ? Sree
Nice to have you Sree
PS G:\Academics\SSN\6th Sem\IP Lab\Node\ip>
```

b. Write a Node.js program that reads all the greetings as before. When all the greetings are loaded, it creates a server listening on port number 8080. On request, it checks for whether there is a name value in the query string. If there isn't, the value of query.name will be undefined.

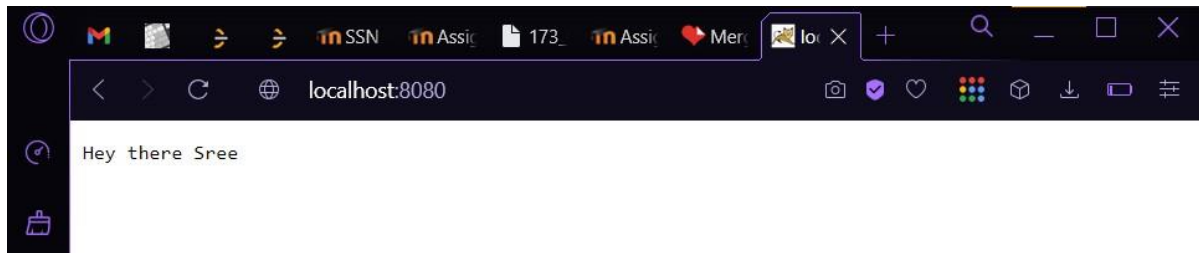
In other words, if you access <http://localhost:8080/?name=Mike>, then your browser should just display something like "Hello, Mike" when the page loads.

Program:

(Node file greet_random_server.js)

```
var fs=require('fs');
var http = require("http");
var url = require('url');
http.createServer(function (request, response) {
  const queryObject = url.parse(request.url,true).query;
  response.writeHead(200, {'Content-Type': 'text/plain'});
  fs.readFile('../input.txt', function (err, data) {
    if (err) return console.log("File Not Found");
    data+='';
    var x=data.split(/\r?\n/)
    var greet = x[Math.floor(Math.random()*x.length)]
    response.write(greet);
    response.write(" ");
    response.end(queryObject['name']);
  });
}).listen(8080);
```

Ouput Screenshot:



c. Create a web server using node.js which listens for clients request. Once the client request the server, the server returns a web page which contains a list of books and its details in table format.

(index.html)

```
<!DOCTYPE html>
<html>
  <head>
    <title>Simple Node Server</title>
  </head>
  <style>
    table:hover {
      transform: scale(1.2);
    }
    th {
      padding: 15px;
      text-align: centre;
      background-color: black;
      border: 2px solid black;
      color: white;
    }
    h1 {
      color: rgb(89, 228, 238);
    }
    tr:hover {background-color: #edffec;}
    tr:nth-child(even) {background-color: white;}
    td {
      padding: 15px;
      text-align: centre;
      border: 1px solid black;
    }
    table {
      border: 1px solid black;
      width : 50%;
      border-collapse: collapse;
      border: 1px solid black;
      margin-left: auto;
      margin-right: auto;
    }
  </style>
  <body>
    <h1>Simple Node Server</h1>
    <table>
      <tr>
        <th>Book Name</th>
        <th>Author</th>
        <th>Year</th>
        <th>Genre</th>
      </tr>
      <tr>
        <td>The Great Gatsby</td>
        <td>F. Scott Fitzgerald</td>
        <td>1925</td>
        <td>Fiction</td>
      </tr>
      <tr>
        <td>1984</td>
        <td>George Orwell</td>
        <td>1949</td>
        <td>Dystopian</td>
      </tr>
      <tr>
        <td>Pride and Prejudice</td>
        <td>Jane Austen</td>
        <td>1813</td>
        <td>Romance</td>
      </tr>
      <tr>
        <td>The Catcher in the Rye</td>
        <td>J.D. Salinger</td>
        <td>1951</td>
        <td>Fiction</td>
      </tr>
    </table>
  </body>
</html>
```

```

        transition: transform 0.3s;
    }
</style>
<body>
    <table border="1px solid black" >
        <thead>
            <th>
                <td><strong>Name</strong></td>
                <td><strong>Author</strong></td>
                <td><strong>Rating</strong></td>
            </th>
        </thead>
        <tbody>
            <tr>
                <td>A Secular Agenda</td>
                <td>Arun Shourie</td>
                <td>5</td>
            </tr>
            <tr>
                <td>A Suitable Boy</td>
                <td>Vikram Seth</td>
                <td>4</td>
            </tr>
            <tr>
                <td>Akbarnama</td>
                <td>Abul Fazal</td>
                <td>4</td>
            </tr>
            <tr>
                <td>An Autobiography</td>
                <td>Jawaharlal Nehru</td>
                <td>2</td>
            </tr>
        </tbody>
    </table>
</body>
</html>

```

(Node.js)

```

var http = require('http');
var fs = require('fs');
var url = require('url');
http.createServer( function (request, response) {
    var pathname = url.parse(request.url).pathname;
    console.log("Request for " + pathname + " received.");
    fs.readFile('.'+pathname, function (err, data) {

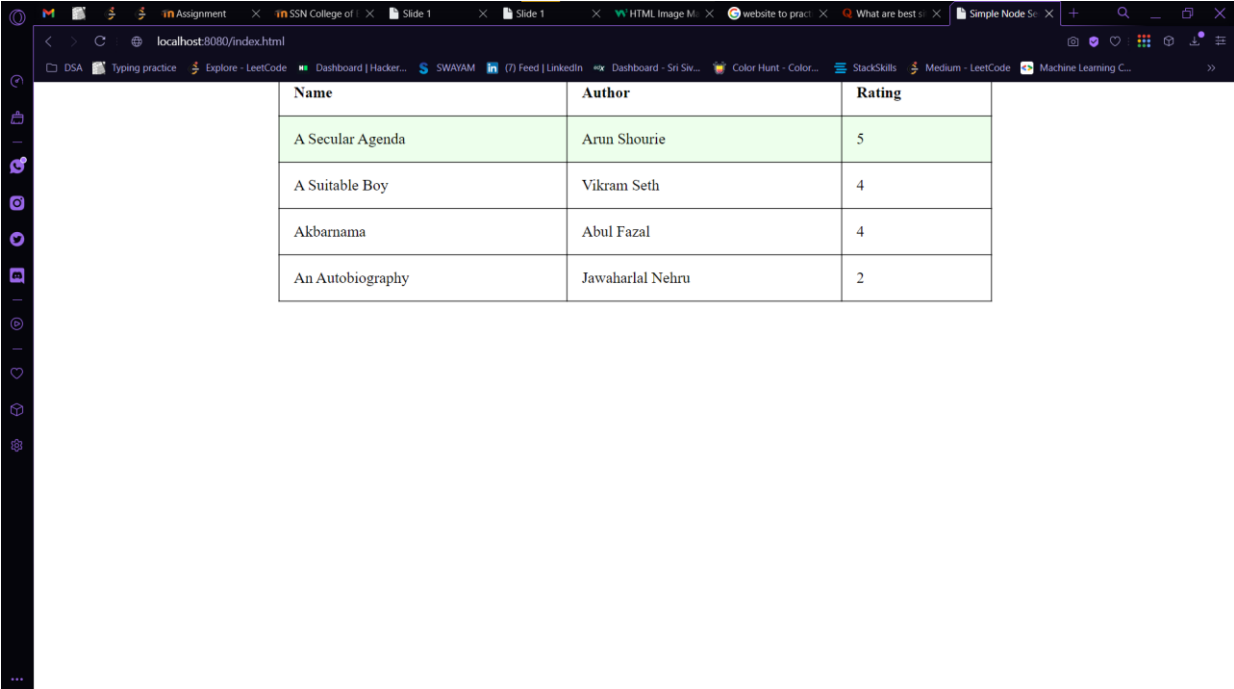
```

```

    if (err) {
      console.log(err);
      response.writeHead(404, {'Content-Type': 'text/html'});
    }
    else {
      response.writeHead(200, {'Content-Type': 'text/html'});
      response.write(data.toString());
    }
    response.end();
  });
}).listen(8080);
console.log('Server running at http://127.0.0.1:8080/');

```

Ouput Screenshot:



Name	Author	Rating
A Secular Agenda	Arun Shourie	5
A Suitable Boy	Vikram Seth	4
Akbarnama	Abul Fazal	4
An Autobiography	Jawaharlal Nehru	2

d. Create a DB with the following details using Mongoddb:

Database Name: Patient_Details

Table Schema: Name, age, ID, gender, address, marital status, Date of Visit

Wrtie a node.js program to do the following operations:

Add, Delete, Update, Search.

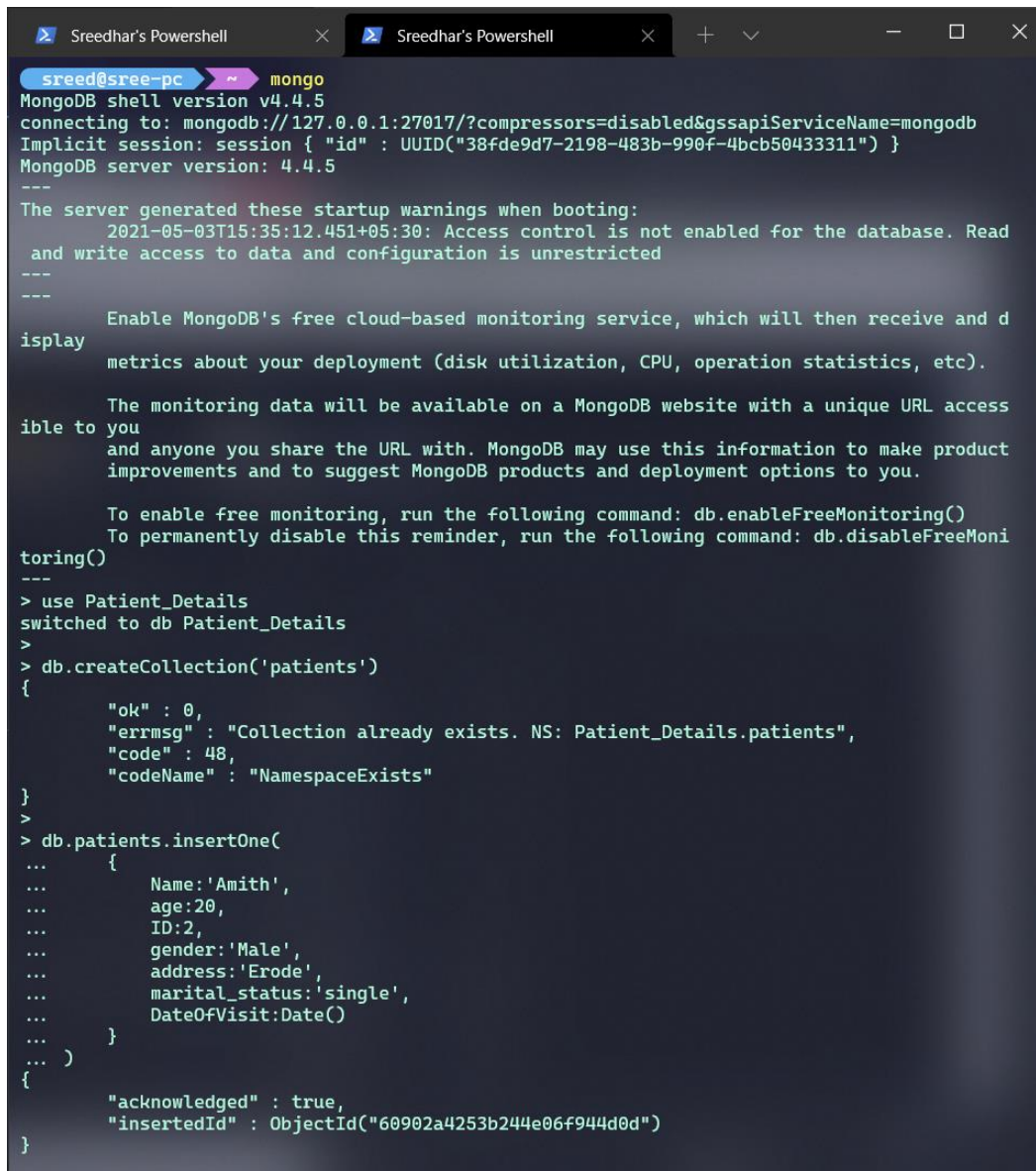
(Node.js)

```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/Patient_Details";

MongoClient.connect(url, {useUnifiedTopology: true }, function(err, db) {
  if (err) throw err;
  console.log("Database Connected");
  var dbObject = db.db("Patient_Details");
  var myobj = {
    Name:'Sree',
    age:20,
    ID:1,
    gender:'Male',
    address:'Erode',
    marital_status:'single',
    DateOfVisit:Date()
  };
  dbObject.collection("patients").insertOne(myobj, function(err, res) {
    if (err) throw err;
    console.log("inserted");
    dbObject.collection('patients').find().toArray(function(err,res){
      if(err) throw err;
      console.log(res);
      dbObject.collection('patients').deleteOne({Name:'Raj Kumar'},
function(err,res){
      if(err) throw err
      console.log('Deleted!');
      dbObject.collection('patients').find().toArray(function(e
rr,res){
        if(err) throw err;
        console.log(res);
        var upd_url = { Name:"Amith Kumar" };
        var upd_values = { $set: {Name: "Vikraman", address:
"Dubai" } };
        dbObject.collection("patients").updateOne(upd_url, up
d_values, function(err, res) {
          if (err) throw err;
          console.log("updated|");
          dbObject.collection('patients').find().toArray(fu
nction(err,res){
            if(err) throw err;
            console.log(res);
            db.close();
          });
        });
      });
    });
  });
});
```

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

Ouput Screenshot:



```
Sreedhar's Powershell x Sreedhar's Powershell x + - □ x  
sreed@sree-pc ~ mongo  
MongoDB shell version v4.4.5  
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb  
Implicit session: session { "id" : UUID("38fde9d7-2198-483b-990f-4bcb50433311") }  
MongoDB server version: 4.4.5  
---  
The server generated these startup warnings when booting:  
  2021-05-03T15:35:12.451+05:30: Access control is not enabled for the database. Read  
  and write access to data and configuration is unrestricted  
---  
---  
  Enable MongoDB's free cloud-based monitoring service, which will then receive and d  
isplay metrics about your deployment (disk utilization, CPU, operation statistics, etc).  
  
  The monitoring data will be available on a MongoDB website with a unique URL access  
ible 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.disableFreeMoni  
toring()  
---  
> use Patient_Details  
switched to db Patient_Details  
>  
> db.createCollection('patients')  
{  
  "ok" : 0,  
  "errmsg" : "Collection already exists. NS: Patient_Details.patients",  
  "code" : 48,  
  "codeName" : "NamespaceExists"  
}  
>  
> db.patients.insertOne(  
...   {  
...     Name:'Amith',  
...     age:20,  
...     ID:2,  
...     gender:'Male',  
...     address:'Erode',  
...     marital_status:'single',  
...     DateOfVisit:Date()  
...   }  
... )  
{  
  "acknowledged" : true,  
  "insertedId" : ObjectId("60902a4253b244e06f944d0d")  
}
```

```
Sreedhar's Powershell  Sreedhar's Powershell  + - X
sreed@sree-pc G:\Academics\SSN\6th Sem\IP Lab\Node\ip master # 0 +12 ~6 -5 node main.js
Database Connected
inserted
[
  {
    _id: 60902a4253b244e06f944d0d,
    Name: 'Amith',
    age: 20,
    ID: 2,
    gender: 'Male',
    address: 'Erode',
    marital_status: 'single',
    DateOfVisit: 'Mon May 03 2021 22:22:18 GMT+0530 (India Standard Time)'
  },
  {
    _id: 60902cd1666d250a88a265cc,
    Name: 'Sree',
    age: 20,
    ID: 1,
    gender: 'Male',
    address: 'Erode',
    marital_status: 'single',
    DateOfVisit: 'Mon May 03 2021 22:33:13 GMT+0530 (India Standard Time)'
  },
  {
    _id: 60902d35d107db2bbc4193de,
    Name: 'Sree',
    age: 20,
    ID: 1,
    gender: 'Male',
    address: 'Erode',
    marital_status: 'single',
    DateOfVisit: 'Mon May 03 2021 22:34:53 GMT+0530 (India Standard Time)'
  }
]
Deleted!
[
  {
    _id: 60902a4253b244e06f944d0d,
    Name: 'Amith',
    age: 20,
    ID: 2,
    gender: 'Male',
    address: 'Erode',
    marital_status: 'single',
    DateOfVisit: 'Mon May 03 2021 22:22:18 GMT+0530 (India Standard Time)'
  },
  {
    _id: 60902cd1666d250a88a265cc,
```



```
Sreedhar's Powershell  Sreedhar's Powershell  +  -  □  ×

gender: 'Male',
address: 'Erode',
marital_status: 'single',
DateOfVisit: 'Mon May 03 2021 22:33:13 GMT+0530 (India Standard Time)'
},
{
  _id: 60902d35d107db2bbc4193de,
  Name: 'Sree',
  age: 20,
  ID: 1,
  gender: 'Male',
  address: 'Erode',
  marital_status: 'single',
  DateOfVisit: 'Mon May 03 2021 22:34:53 GMT+0530 (India Standard Time)'
}
]
updated|
[
  {
    _id: 60902a4253b244e06f944d0d,
    Name: 'Amith',
    age: 20,
    ID: 2,
    gender: 'Male',
    address: 'Erode',
    marital_status: 'single',
    DateOfVisit: 'Mon May 03 2021 22:22:18 GMT+0530 (India Standard Time)'
  },
  {
    _id: 60902cd1666d250a88a265cc,
    Name: 'Sree',
    age: 20,
    ID: 1,
    gender: 'Male',
    address: 'Erode',
    marital_status: 'single',
    DateOfVisit: 'Mon May 03 2021 22:33:13 GMT+0530 (India Standard Time)'
  },
  {
    _id: 60902d35d107db2bbc4193de,
    Name: 'Sree',
    age: 20,
    ID: 1,
    gender: 'Male',
    address: 'Erode',
    marital_status: 'single',
    DateOfVisit: 'Mon May 03 2021 22:34:53 GMT+0530 (India Standard Time)'
  }
]
```