Ex. No 4.a) NodeJS with Http Module

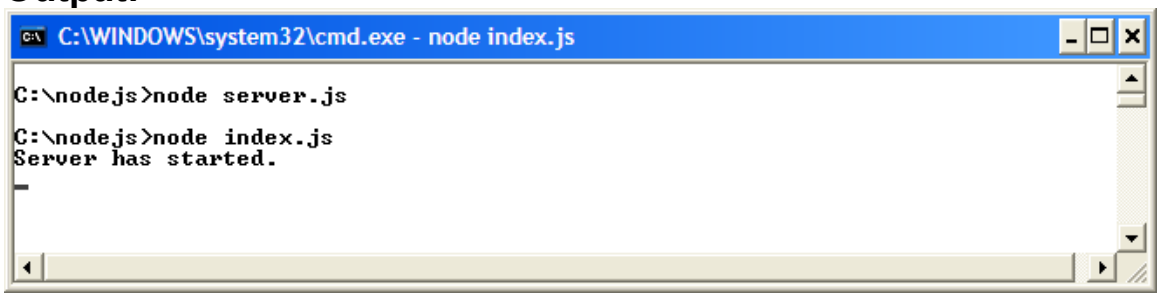
server.js

```
var http = require("http");
function start() {
  function onRequest(request, response) {
    console.log("Request received.");
    response.writeHead(200, {"Content-Type": "text/plain"});
    response.write("Hello Welcome to ALLIANCE");
    response.end();
  }
  http.createServer(onRequest).listen(8888);
  console.log("Server has started.");
}
exports.start = start;
```

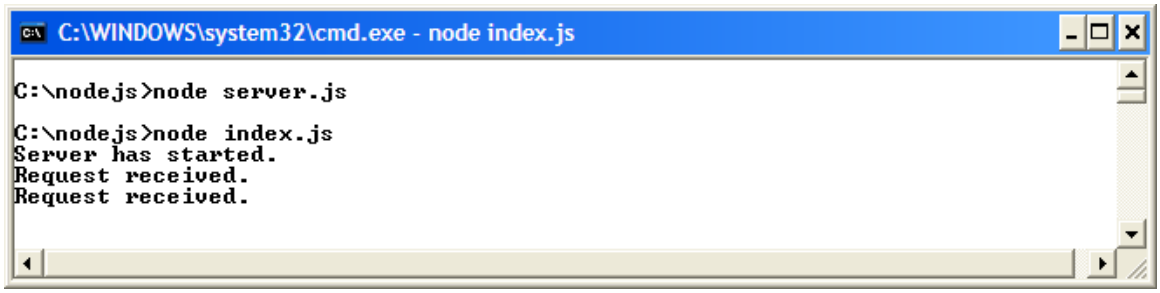
index.js

```
var server = require("./server");
server.start();
```

Output:



In the Browser



Ex. No. 4.b) NodeJS Request Handlers

requestHandlers.js

```
function start() {
console.log("Request handler 'start' was called.");
return "Hello Start";
}
function upload() {
```

```
console.log("Request handler 'upload' was called.");
return "Hello Upload";
}
Function start1()
{
Console.log(“We have only cheked the URL”);
Return “URL for local host displayed”;
}
exports.start = start;
exports.upload = upload;
exports.start1=start1;
```

router.js

```
function route(handle, pathname) {
console.log("About to route a request for " + pathname);
if (typeof handle[pathname] === 'function') {
return handle[pathname]();
} else {
console.log("No request handler found for " + pathname);
return "404 Not found";
}
}
exports.route = route;
```

server.js

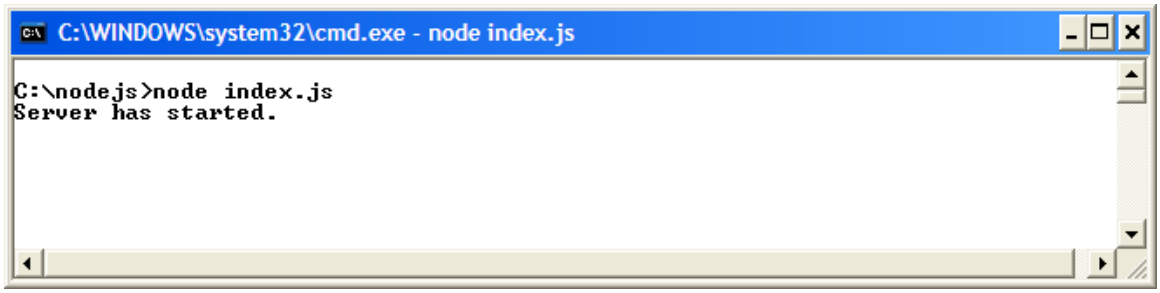
```
var http = require("http");
var url = require("url");
function start(route, handle) {
function onRequest(request, response) {
var pathname = url.parse(request.url).pathname;
console.log("Request for " + pathname + " received.");
response.writeHead(200, {"Content-Type": "text/plain"});
var content = route(handle, pathname)
response.write(content);
response.end();
}
http.createServer(onRequest).listen(8888);
console.log("Server has started.");
}
exports.start = start;
```

index.js

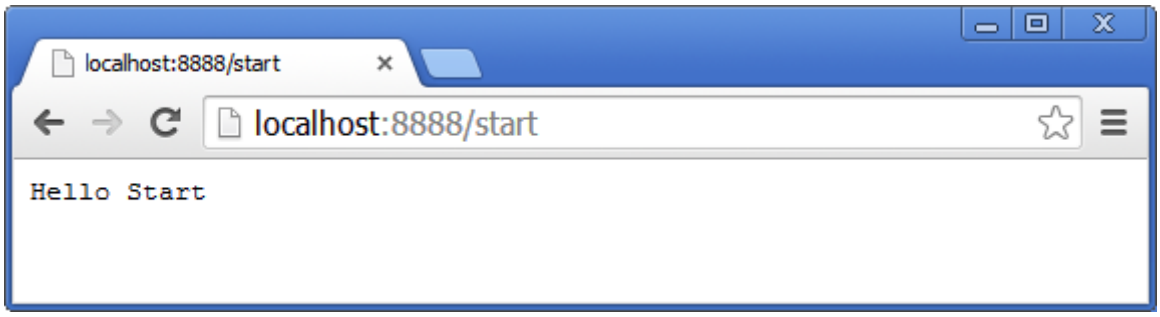
```
var server = require("./server");
var router = require("./router");
var requestHandlers = require("./requestHandlers");
var handle = {}
handle["/"] = requestHandlers.start1;
handle["/start"] = requestHandlers.start;
handle["/upload"] = requestHandlers.upload;
```

server.start(router.route, handle);

Ouput:



Browser:



Ex. No. 4.c) NodeJS FS module

Index.html

```
<html>
   <head>
      <title>Sample Page</title>
   </head>
   <body>
      Hello Welcome to Index.html!
   </body>
</html>
```

Fileservr.js

```
var http = require('http');
var fs = require('fs');
var url = require('url');
// Create a server
http.createServer( function (request, response) {
   // Parse the request containing file name
```

```
    var pathname = url.parse(request.url).pathname;
    // Print the name of the file for which request is made.
    console.log("Request for " + pathname + " received.");
    // Read the requested file content from file system
    fs.readFile(index.html, function (err, data) {
       if (err) {
          console.log(err);
          response.writeHead(404, {'Content-Type': 'text/html'});
       } else {
                    response.writeHead(200, {'Content-Type': 'text/html'});
          // Write the content of the file to response body
          response.write(data.toString());
       }
       // Send the response body
       response.end();
    });
}).listen(8081);
// Console will print the message
console.log('Server running at http://127.0.0.1:8081/');
```

Ouput:

```
C:\Program Files\nodejs>node fileserver.js
Server running at http://127.0.0.1:8081/
Request for /index.html received.
```

Ex. No. 4.d) NodeJS File Handling operations

fileempty.js

```
var fs = require('fs');
//open a file
fs.open('Alliance1.txt', 'w', function (err, file) {
  if (err) throw err;
  console.log('File Opened!');
});
//Write a file
fs.writeFile('Alliance1.txt', 'Welcome to Alliance!', function (err) {
  if (err) throw err;
```

```
  console.log('File Written!');
});
//Append a file
fs.appendFile('Alliance1.txt', 'Welcome to Department of CSE & IT!', function (err)
{
  if (err) throw err;
  console.log('File Append!');
});
//Delete a file
fs.unlink('Alliance.txt', function (err) {
  if (err) throw err;
  console.log('File deleted!');
});
//Rename a file
fs.rename('Alliance1.txt', 'Alliance200.txt', function (err) {
  if (err) throw err;
  console.log('File Renamed!');
});
```

Output:

```
C:\Program Files\nodejs>node fileempty.js
File deleted!
File Opened!
File Renamed!
File Append!
File Written!
```

Ex. No. 4.e) NodeJS Data Handling using Post method

H1.js

// Request Handlers

```
var querystring = require("querystring");
function start(response, postData) {
console.log("Request handler 'start' was called.");
var body = '<html>'+
'<head>'+
'<meta http-equiv="Content-Type" content="text/html; '+
'charset=UTF-8" />'+
'</head>'+
'<body>'+
'<form action="/upload" method="post">'+
'<textarea name="text" rows="20" cols="60"></textarea>'+
'<input type="submit" value="Submit text" />'+
```

```
'</form>'+
'</body>'+
'</html>';
response.writeHead(200, {"Content-Type": "text/html"});
response.write(body);
response.end();
}
function upload(response, postData) {
console.log("Request handler 'upload' was called.");
response.writeHead(200, {"Content-Type": "text/plain"});
response.write("You've sent: " + postData);
response.end();
}
exports.start = start;
exports.upload = upload;
```

R1.js

```
function route(handle, pathname, response, postData) {
console.log("About to route a request for " + pathname);
if (typeof handle[pathname] === 'function') {
handle[pathname](response, postData);
} else {
console.log("No request handler found for " + pathname);
response.writeHead(404, {"Content-Type": "text/plain"});
response.write("404 Not found");
response.end();
}
}
exports.route = route;
```

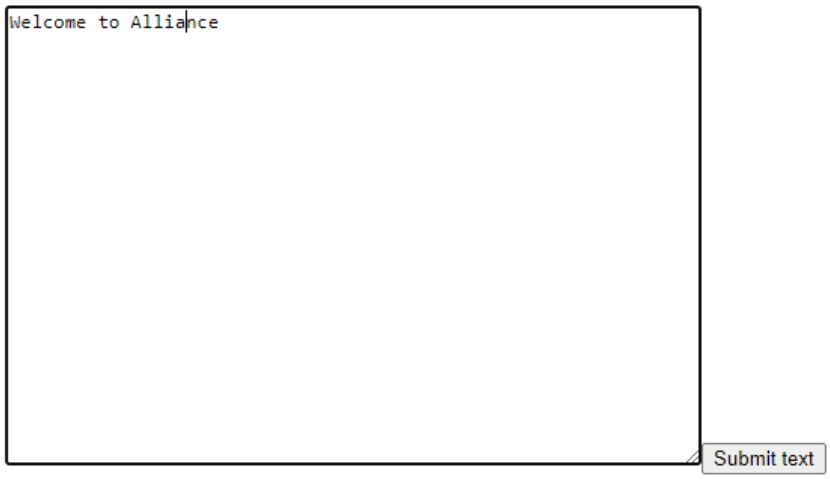
S1.js

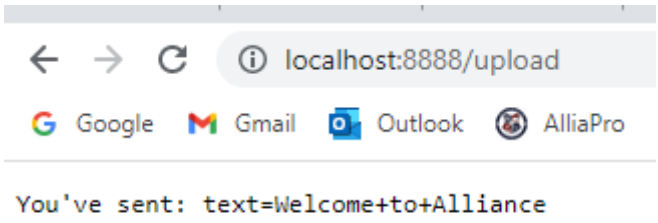
```
var http = require("http");
var url = require("url");
function start(route, handle) {
function onRequest(request, response) {
var postData = "";
var pathname = url.parse(request.url).pathname;
console.log("Request for " + pathname + " received.");
request.setEncoding("utf8");
request.addListener("data", function(postDataChunk) {
postData += postDataChunk;
console.log("Received POST data chunk '"+postDataChunk + "'.");
});
request.addListener("end", function() {
route(handle, pathname, response, postData);
});
}
http.createServer(onRequest).listen(8888);
console.log("Server has started.");
}
```

```
exports.start = start;
```

I1.js

```
var server = require("./S1");
var router = require("./R1");
var requestHandlers = require("./H1");
var handle = {}
handle["/"] = requestHandlers.start;
handle["/start"] = requestHandlers.start;
handle["/upload"] = requestHandlers.upload;
server.start(router.route, handle);
```





Ex. No. 4.f) Post Method with HTML

Web.js

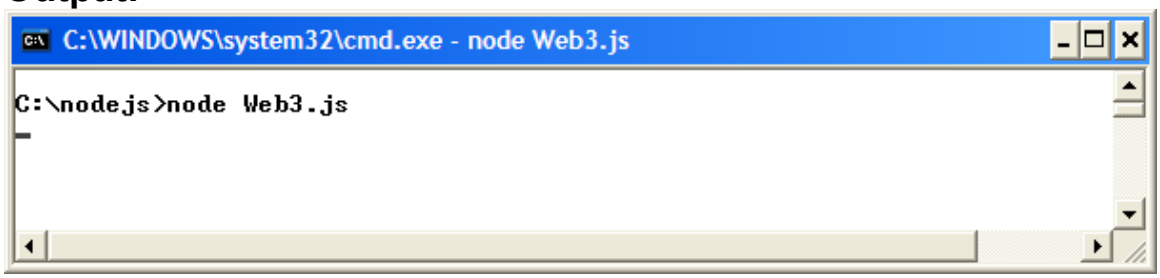
```
var http = require("http");
var qs = require("querystring");
var server = http.createServer(function(request, response) {
var bodyString = "";
request.setEncoding("utf8");
request.on("data", function(data) {
bodyString += data;
});
request.on("end", function() {
var body = qs.parse(bodyString);
for (var b in body) {
response.write(b + ' = ' + body[b] + "\n");
```

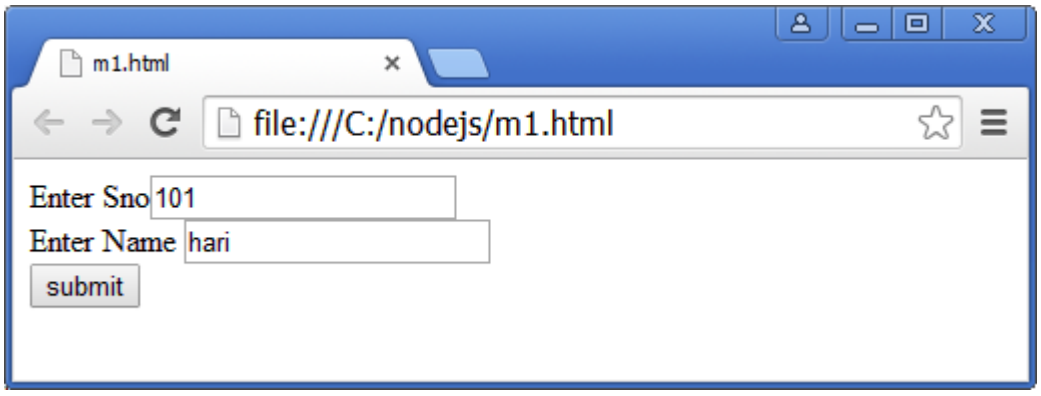
```
}
response.end();
});
});
server.listen(8000);
```

m1.html

```
<form method="post" action="http://localhost:8000">
Enter Sno<input type="text" name="sno" value="101">
<br>
Enter Name <input type="text" name="name" value="hari">
<br>
<input type="submit" value="submit">
</form>
```

Output:





Ex. No. 4.g) NodeJS with Middleware

Step:1 npm install connect

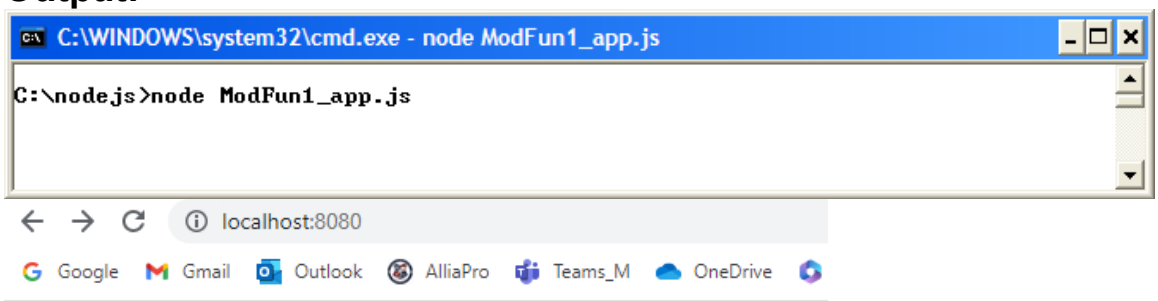
Step:2 fn16.js

```
var connect = require("connect");
function JsFun1(req, res) {
res.writeHead(500, {'Content-Type': 'text/html'});
res.end('<h1>This is a Module Example Program!</h1>\n<h2>Alliance Bangalore</h2>');
}
module.exports = JsFun1;
```

Step:3 app1.js

```
var connect = require('connect');
// import middlewares
var rem1 = require('./f16');
var app = connect();
app
   .use(rem1)
   .listen(8080);
console.log("Server is listening");
```

Output:



This is a Module Example Program!

Alliance Bangalore

Ex. No. 4.h) NodeJS with MySQL

```
npm install -g mysql
or
npm install db-mysql
or
npm install mysql
```

Create a database in MySQL and give the user name and password.
Create table Alliance2(sno int(5) primary key, name varchar(10). Sc int(2))
Insert into Alliance1 values(101,'ravi',1)
Like that enter 5 data.

Fn11.js

```
var mysql = require('mysql');
var connection = mysql.createConnection(
{"host":"localhost",
"user":"root",
"password":"root",
"database":"paul",
  debug : false,    });
connection.connect();
var queryString = "SELECT * FROM Alliance2";
```

```
connection.query(queryString,function(err, rows, fields) {
if (err) throw err;
for(var i in rows){
     console.log('Sno   : ', rows[i].sno);
     console.log('Name: ', rows[i].name);
     console.log('Sc     : ', rows[i].sc);
    }
});
connection.end();
```

```
C:\Users\mpaul\Documents>node f15.js
Sno   :  101
Name:  Rohith
Sc     :  1
Sno   :  102
Name:  Kumar
Sc     :  2
Sno   :  103
Name:  Raja
Sc     :  1
```

Ex. No. 5.a)) MongoDB & NodeJS with MongoDB

Mongodb shell or Mongodb - Create Databases and proceed with the following commands

```
>show dbs
>use school
>school.dropDatabase();
```

Create Collections:

```
>show collections
>db.createCollection('student',{});
```

Insert Values:

```
>db.student.insert({id:101,name:"Ravi",email:ravi@gmail.com})
```

Similarly you will insert records as individual or group. We can insert group records through json file format.

Ex) P1.json

```
[{    "id": 1,
    "firstName": "Steve",
    "lastName": "Jobs"
},{    "id": 2,
    "firstName": "Bill",
    "lastName": "Gates"
}]
```

```
>db.student.insertMany( ….copy and paste the entire json file….)
```

It will create collection for all documents

Find Values:

To display the values use find()

```
>db.student.find()
```

To display in a good format use pretty()

```
>db.student.find().pretty()
```

To select only appropriate files, use appropriate values like

```
>db.student.find({firstname:"bill"}).pretty()
```

If the name starts with bill it will return its record

```
>db.student.find({grade:10,gpa:{$gt:2.0}}).pretty()
```

If gpa > 2 it will return. Similarly $lt, $lte, $gte, $ne used…

```
>db.student.find({classes:{$size:3}}).pretty()
```

It returns classes array of size 3 values.

Update Values:

```
>db.student.update({firstname:"ravi"},{$set:{firstname:"saavi"}}).pretty()

>db.student.remove({grade:1})
```

Delete values:
>db.student.deleteOne({_id:Object{“ADEA324588BCA5475O97”)})

>db.student.find({grade:12})

>db.student.deleteMany({grade:12})

Functions:

$lte,$lt, $gte, $gt, $eq, $ne, $set, $and, $or…

Aggregation:
Min, max, sort, limit, group, lookup, match, merge, project, unwind, unset
With attribute used….use {$sort:{name:1}}
With function used… use db.stu.find().limit(3), db.stu.find().skip(3)

Examples:
db.collection('student').find({projection:{_id:0,firstName:1}})
db.collection('student').find({firstName:/^B/})
db.collection('student').find({name:”Bill”})
db.collection('student').find().sort({name:-1})

Creating indexes:
>db.student.ensureIndex(firstName:1)

Ex. No. 5. b) Mongodb Connected with NodeJS Programs:

```
var MongoClient = require('mongodb').MongoClient;
MongoClient.connect('mongodb://localhost:27017/', function (err, client)
{  if (err) throw err;
  var db=client.db('school');
//var multi=[{slno:103,name:'ACED'},{regno:104,name:'Mahesh'}];
//db.collection('student').insertMany(multi, function(err, result) {
//db.collection('student').updateOne(myf,newmyf,function(err,result) {
//
db.collection('student').find({name:”ACED”},{projection:{_id:0,name:1}}).toArray(function(err,res
ult) {
//db.collection('student').find({name:/^A/},{projection:{_id:0,name:1}}).toArray(function(err,result
) {
// db.collection('student').find().sort(sortorder).toArray(function(err,result) {
db.collection('student').insertOne({ id: 12,firstName:'Steve'}, function(err, result) {
        if(err)     throw err;
          console.log("Document inserted successfully");
          client.close();
          });
});
```

Ex. No. 5. c) Rest API (CRUD Application)

Install Modules

npm install -g express (express JS)
npm install express –save (express JS)
npm install body-parser –save (middleware)
npm install cookie-parser –save (requesting cookies)
npm install multer –save (multi-part / form-data)

3 files to be created: f13.html, f13.js, users.json

F13.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>REST API</title>
</head>
<body>
    <h1>Add user

    </h1>
    <form action="/adduser" method="post">
user id: <input type="text" name="userid" size="20"> <br>
Name is: <input type="text" name="name" size="20"> <br>
DOB is: <input type="text" name="dob" size="20"> <br>
Profession: <input type="text" name="profession" size="20"> <br>
 <button type="submit" name="submit">
   Add User </button>
    </form>

    <h1>Get Specific User     </h1>
    <form action="/particularuser" method="post">
User id: <input type="text" name="userid" size="20"> <br>
 <button type="submit" name="submit">
 Get User Info </button>
    </form>


    <h1>Delete User details     </h1>
    <form action="/deleteuser" method="post">
User id: <input type="text" name="userid" size="20"> <br>
 <button type="submit" name="submit">
 Delete User Info </button>
    </form>
```

```
    <h1>Show All User details     </h1>
    <form action="/showall" method="get">
 <button type="submit" name="submit">
Show All User Info </button>
    </form>
</body>
</html>
```

F13.js

```
const express=require('express');
const bodyparser= require('body-parser');
var fs= require('fs');
const app=express();
app.use(bodyparser.urlencoded({extended:true}));

app.get("/",function(req,res){
    res.sendFile(__dirname+"/f13.html")
});
app.post("/adduser",function(req,res)
{    var name= req.body.name;
    var dob=req.body.dob;
    var profession= req.body.profession;
    var obj={};
    var key= req.body.userid;
    var newuser={'name':name,'dob':dob,'profession':profession}
    obj[key]=newuser;
    fs.readFile("users.json","utf8",function(err,data)
    {
        data=JSON.parse(data);
        data[key]=obj[key];
        console.log(data);
        var updateuser=JSON.stringify(data);
        fs.writeFile("users.json",updateuser,function(err)
        {
            res.end(JSON.stringify(data));
        });
    });
});
app.post("/particularuser",function(req,res)
{
    fs.readFile("users.json","utf8",function(err,data)
    {
        var users=JSON.parse(data);
        var user=users[req.body.userid];
        console.log(user);
        res.end(JSON.stringify(user));
    });
});
app.post("/deleteuser",function(req,res)
```

```
{
   fs.readFile("users.json","utf8",function(err,data)
   {     var data=JSON.parse(data);
         delete data[req.body.userid];
        console.log(data);
            var updateuser=JSON.stringify(data);
        fs.writeFile("users.json",updateuser,function(err)
        {
            res.end(JSON.stringify(data));
        });
   });
});
app.get("/showall",function(req,res)
{
   fs.readFile("users.json","utf8",function(err,data)
   {    console.log(data);
        res.end(data);
   });
});
app.listen(3000, function(){
   console.log("Server is listening on port 3000");
});
```

Users.json

{"1":{"name":"mano","dob":"01","profession":"Teacher"},"2":{"name":"sam","dob":"02","professi
on":"Professor"},"3":{"name":"Mano","dob":"04","profession":"Manager"},"101":{"name":"Jayase
kar","dob":"23","profession":"Doctor"},"102":{"name":"Paul","dob":"08","profession":"AssociateP
rofessor"}}

Add user

user id:
Name is:
DOB is:
Profession:
Add User

Get Specific User

User id:
Get User Info

Delete User details

User id:
Delete User Info

Show All User details

Show All User Info

Ex. No. 5. d) Component-based Appln with ReactJS

Ex. No. 5. e) Class based Appln with React JS