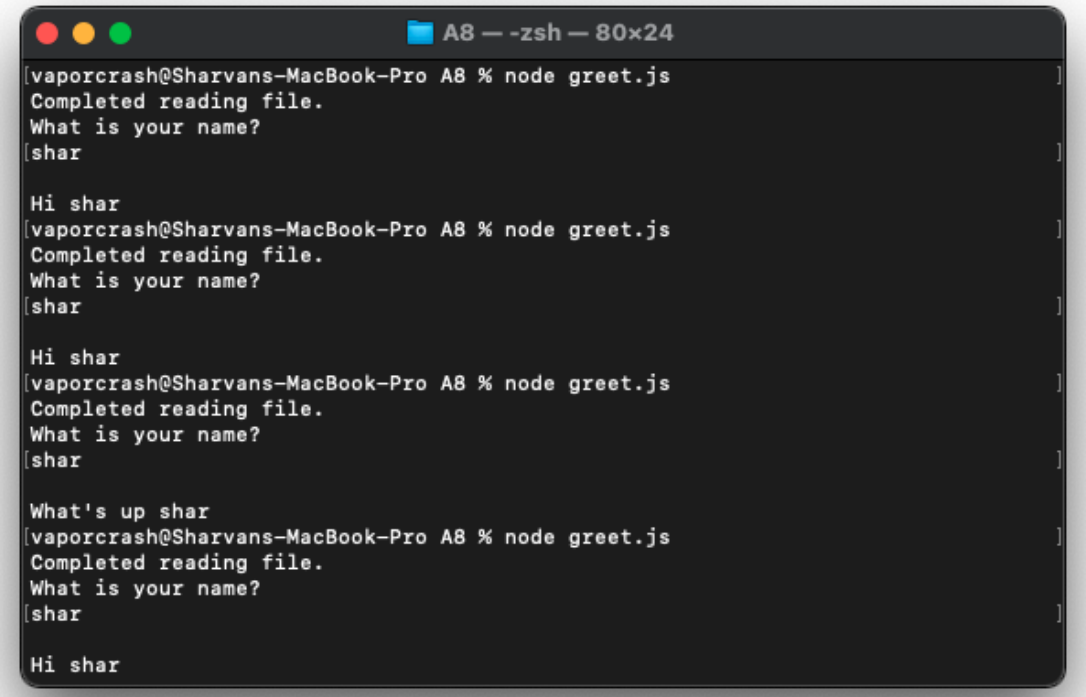
Sharvan Ram Kumaran
CSE-C
185001143
Assignment 8

<u>Aim-</u> Use node.js to print randomly generated greetings

i)<u>Code:</u>

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

var greetings = []

fs.readFile("greetings.txt", function(err, info){
   if(err){
      console.log("ERROR:File not found!");
      return 1;
   }
   greetings = info.toString().split("\n");
});
console.log("Completed reading file.");

const read = require('readline').createInterface({
  input: process.stdin,
  output: process.stdout
})

read.question("What is your name?\n", (name) => {
  console.log(`\n${greetings[Math.floor(Math.random() * greetings.length)]} ${name}`);
  read.close();
})
```

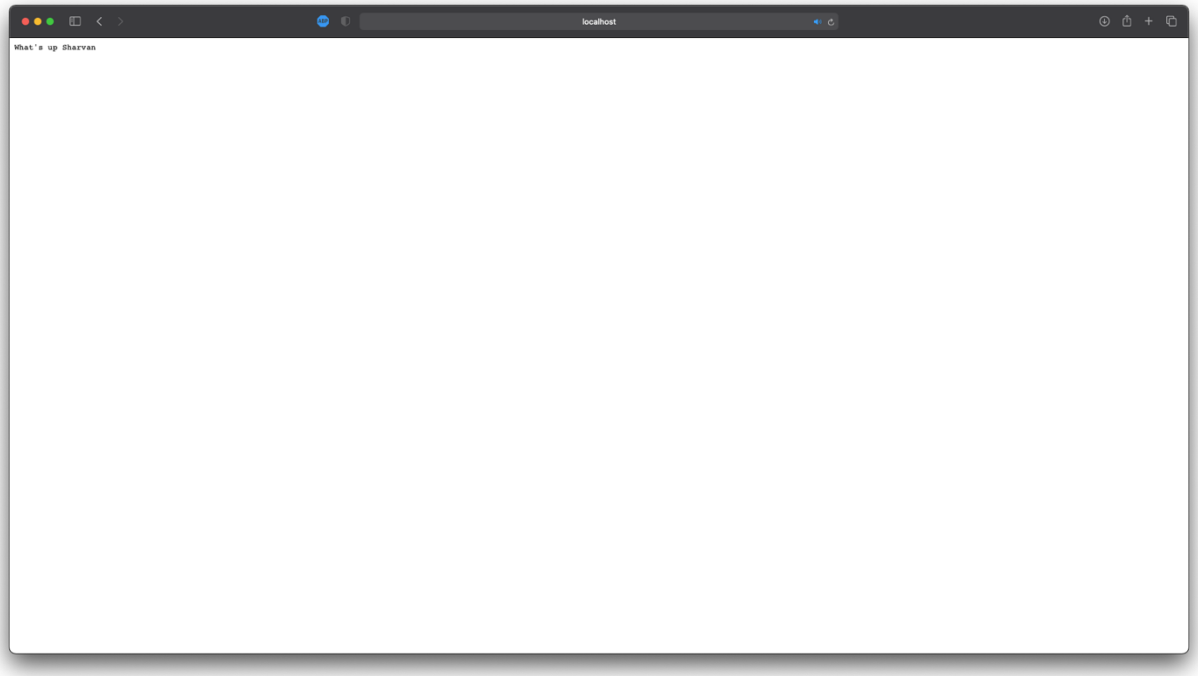
Output:

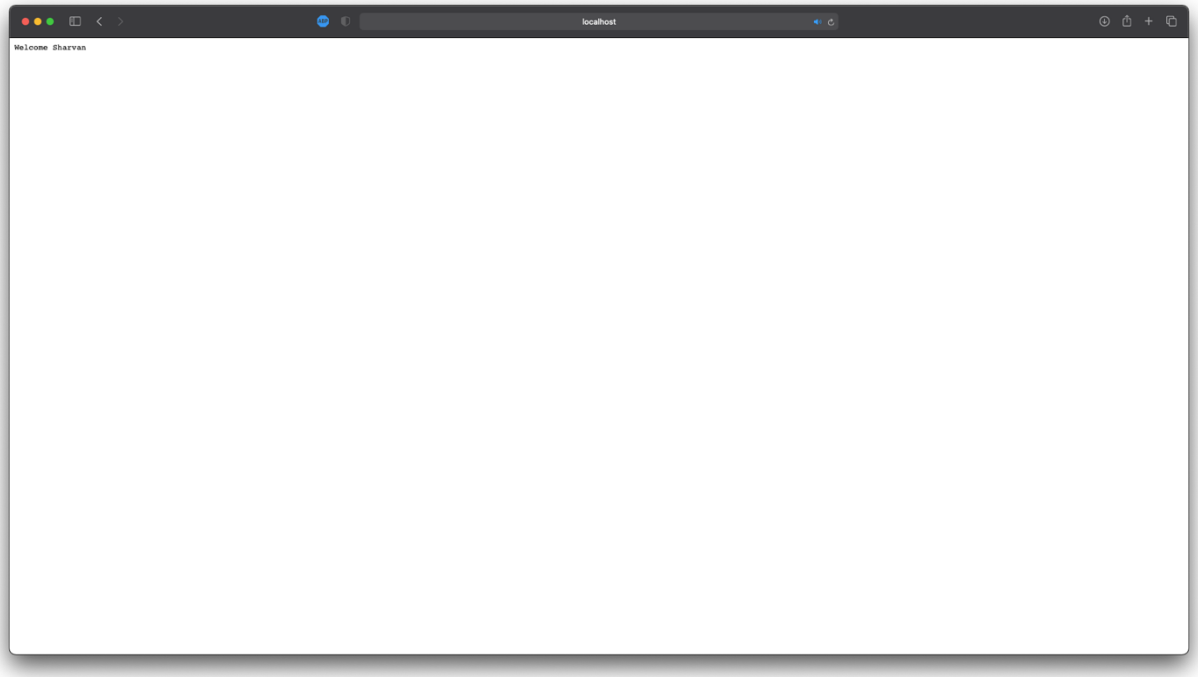


ii)Code:

```
const fs = require("fs");
var http = require("http");
const url = require("url");

http.createServer(function(req,res){
  const query = url.parse(req.url,true).query;
  res.writeHead(200,{'Content-Type':'text/plain'});
  name = query.name;
  fs.readFile('greetings.txt',function(err,data){
    var read = data.toString().split("\n");
    if(err){
      res.end("404");
    }
    len= read.length;
    res.end(read[Math.floor(Math.random()*len+1)-1] + " " + name);
  });
}).listen(8080);
```

Output:





Greetings.txt

Hello
Hey
Hi
What's up
Welcome

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.

Books.js

```
const fs = require("fs");
const http = require("http");
const url = require("url");

http.createServer(function(req,res){
  res.writeHead(200,{'Content-type':'text/html'});
  fs.readFile('index.html',function(err,data){
    if(err){
      res.writeHead(404);
      res.write("File Not Found!");
    } else{
      res.write(data);
    }
    res.end();
  });
}).listen(8080);
```

Index.html

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Books</title>
    <style media="screen">
      table,tb,td{
        cellspacing: 25px;
        border: 1px solid black;
        border-collapse: collapse;
        padding: 10px;
      }
    </style>
  </head>
```

```
  <body>
    <table>
      <thead>
        <tr>
          <td>Title</td>
          <td>Author</td>
          <td>Price (Rs)</td>
        </tr>
      </thead>
      <tbody>
        <tr>
          <td>Harry Potter</td>
          <td>J.K Rowling</td>
          <td>499.99</td>
        </tr>
        <tr>
          <td>Percy Jackson</td>
          <td>Rick Riordan</td>
          <td>399.99</td>
        </tr>
        <tr>
          <td>Artemis Fowl</td>
          <td>Eoin Colfer</td>
          <td>599.99</td>
        </tr>
        <tr>
          <td>Calvin & Hobbes</td>
          <td>Bill Watterson</td>
          <td>499.99</td>
        </tr>
        <tr>
          <td>Murder on the Orient Express</td>
          <td>Agatha Christie</td>
          <td>299.99</td>
        </tr>
      </tbody>
    </table>
  </body>
</html>
```

Output:

Title	Author	Price (Rs)
Harry Potter	J.K Rowling	499.99
Percy Jackson	Rick Riordan	399.99
Artemis Fowl	Eoin Colfer	599.99
Calvin & Hobbes	Bill Watterson	499.99
Murder on the Orient Express	Agatha Christie	299.99

d)Implement given database using mongoDB and Node.js

patients.js

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

var pobj = [{
  name: "John",
  age: 21,
  id: 01,
  gender: "Male",
  address: "ECR",
  marital: "Single",
  dov: "27/06/21",
},
```

```
{
  name: "Kate",
  age: 21,
  id: 02,
  gender: "Female",
  address: "OMR",
  marital: "Single",
  dov: "29/06/21",
},
{
  name: "Leo",
  age: 25,
  id: 03,
  gender: "Male",
  address: "Adyar",
  marital: "Married",
  dov: "30/06/21",
}];

MongoClient.connect(url,async function(err,db){
  if(err) throw err;
  var patient = db.db("Patient_Details");

   await patient.collection("patients").insertMany(pobj,function(err,res){
    if(err) throw err;
    console.log("Inserted " + res.insertedCount + " documents");

  });

  await patient.collection("patients").find({}).toArray(function(err,res){
    if(err) throw err;
    console.log(res);
  });

  await patient.collection("patients").deleteOne({name:"Leo"},function(err,res){
    if(err) throw err;
    console.log("1 document deleted");

  });

  await patient.collection("patients").find({}).toArray(function(err,res){
    if(err) throw err;
    console.log(res);
  });
```

```
  await patient.collection("patients").updateOne({name: "Kate"},{$set:{age:
23}},function(err,res){
    if (err) throw err;
    console.log("1 doc updated");
  });

  await patient.collection("patients").find({name:"Kate"}).toArray(function(err,res){
   if(err) throw err;
   console.log(res);

 });
  db.close();
});
```

Output:

```
A8 — -zsh — 80×42
Inserted 3 documents
[
  {
    _id: 608eecb33f14270ef8917ccc,
    name: 'John',
    age: 21,
    id: 1,
    gender: 'Male',
    address: 'ECR',
    marital: 'Single',
    dov: '27/06/21'
  },
  {
    _id: 608eecb33f14270ef8917ccd,
    name: 'Kate',
    age: 21,
    id: 2,
    gender: 'Female',
    address: 'OMR',
    marital: 'Single',
    dov: '29/06/21'
  },
  {
    _id: 608eecb33f14270ef8917cce,
    name: 'Leo',
    age: 25,
    id: 3,
    gender: 'Male',
    address: 'Adyar',
    marital: 'Married',
    dov: '30/06/21'
  }
]
1 document deleted
[
  {
    _id: 608eecb33f14270ef8917ccc,
    name: 'John',
    age: 21,
    id: 1,
    gender: 'Male',
    address: 'ECR',
```

```
A8 — -zsh — 80×42
    address: 'Adyar',
    marital: 'Married',
    dov: '30/06/21'
  }
]
1 document deleted
[
  {
    _id: 608eecb33f14270ef8917ccc,
    name: 'John',
    age: 21,
    id: 1,
    gender: 'Male',
    address: 'ECR',
    marital: 'Single',
    dov: '27/06/21'
  },
  {
    _id: 608eecb33f14270ef8917ccd,
    name: 'Kate',
    age: 21,
    id: 2,
    gender: 'Female',
    address: 'OMR',
    marital: 'Single',
    dov: '29/06/21'
  }
]
1 doc updated
[
  {
    _id: 608eecb33f14270ef8917ccd,
    name: 'Kate',
    age: 23,
    id: 2,
    gender: 'Female',
    address: 'OMR',
    marital: 'Single',
    dov: '29/06/21'
  }
]
vaporcrash@Sharvans-MacBook-Pro A8 %
```

Learning Objective:

- Given server/client programs were done using Node.js
- Given database programs were done using MongoDB and Node.js