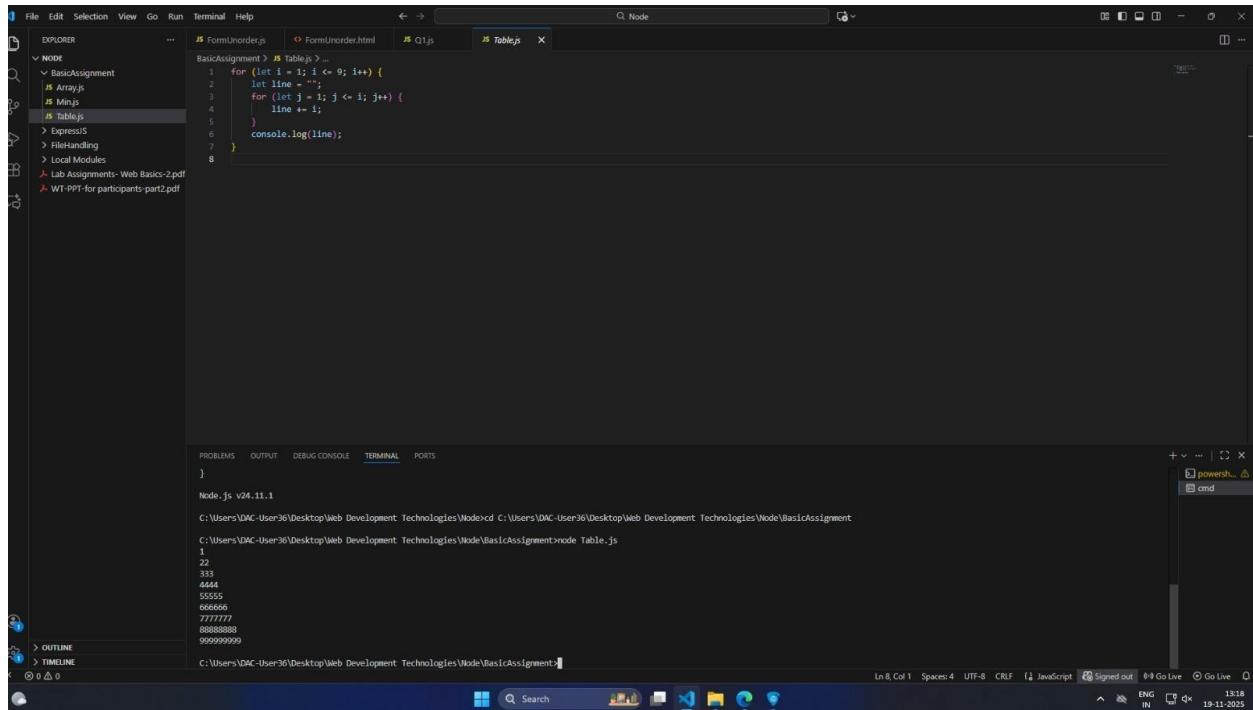


## WPT in Assignment 2

Q1.

```
for (let i = 1; i <= 9; i++) {  
    let line = "";    for (let j = 1;  
        j <= i; j++) {        line += i;  
    }  
  
    console.log(line);  
}
```



The screenshot shows the Visual Studio Code interface. The left sidebar displays a file tree under the 'NODE' category, including files like 'BasicAssignment', 'Array.js', 'Minjs.js', 'Table.js', 'Express.js', 'FileHandling', 'Local Modules', 'Lab Assignments- Web Basics-2.pdf', and 'WT-PPT for participants-part2.pdf'. The main editor area has three tabs open: 'FormUnorder.js', 'FormUnorder.html', and 'Table.js'. The 'Table.js' tab contains the provided code for Q1. Below the editor is a terminal window showing the command 'node Table.js' being run and the resulting output:  
Node.js v24.11.1  
C:\Users\DAC-User36\Desktop\Web Development Technologies\Node>cd C:\Users\DAC-User36\Desktop\Web Development Technologies\Node\BasicAssignment  
C:\Users\DAC-User36\Desktop\Web Development Technologies\Node\BasicAssignment>node Table.js  
1  
22  
333  
4444  
55555  
666666  
777777  
8888888  
99999999

Q2.

```
for (let i = 1; i <= 9; i++) {  
    let line = "";    for (let j =  
        1; j <= i; j++) {        line +=  
    i;
```

```

    }
    console.log(line);
}

}

```

The screenshot shows the Visual Studio Code interface. In the top right, there's a terminal window with the command `node Table.js` run, displaying the output of a script that prints a 9x9 multiplication table. Below the terminal is an output pane showing the same table. The bottom right corner of the screen shows system status icons.

```

BasicAssignment > JS Table.js ->
1   for (let i = 1; i <= 9; i++) {
2     let line = "";
3     for (let j = 1; j <= i; j++) {
4       line += i;
5     }
6     console.log(line);
7   }
8

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
C:\Users\DAC-User36\Desktop\Web Development Technologies\Node\BasicAssignment>node Table.js
1
22
333
4444
55555
666666
777777
8888888
99999999

C:\Users\DAC-User36\Desktop\Web Development Technologies\Node\BasicAssignment>node Array.js
length :
Meenal
Anne
C:\Users\DAC-User36\Desktop\Web Development Technologies\Node\BasicAssignment>

```

Q3.

//Write a function min that takes a array and return minimum number in the set of argument

```

function min(arr) { let smallest =
arr[0]; for (let i = 1; i < arr.length;
i++) { if (arr[i] < smallest)
{ smallest = arr[i];
}
}
return smallest;
}

```

```
console.log(min([5, 2, 9, 1, 7]));
```

The screenshot shows the Visual Studio Code interface. In the top right, there's a terminal window with the command 'node Min.js' and its output:

```
22
333
4444
55555
666666
777777
8888888
99999999
C:\Users\DAC-User36\Desktop\Web Development Technologies\Node\BasicAssignment>node Array.js
length :
Meena
Arun
C:\Users\DAC-User36\Desktop\Web Development Technologies\Node\BasicAssignment>node Min.js
1
```

The bottom right corner displays the date and time: 19-11-2025 13:22.

## Local Modules :

Q1.

```
// greet.js function
```

```
greet() {
```

```
    const hour = new Date().getHours();
```

```
    if (hour < 12) {      return
```

```
        "Good Morning";  } else if
```

```
    (hour < 17) {      return
```

```
        "Good Afternoon"; } else {
```

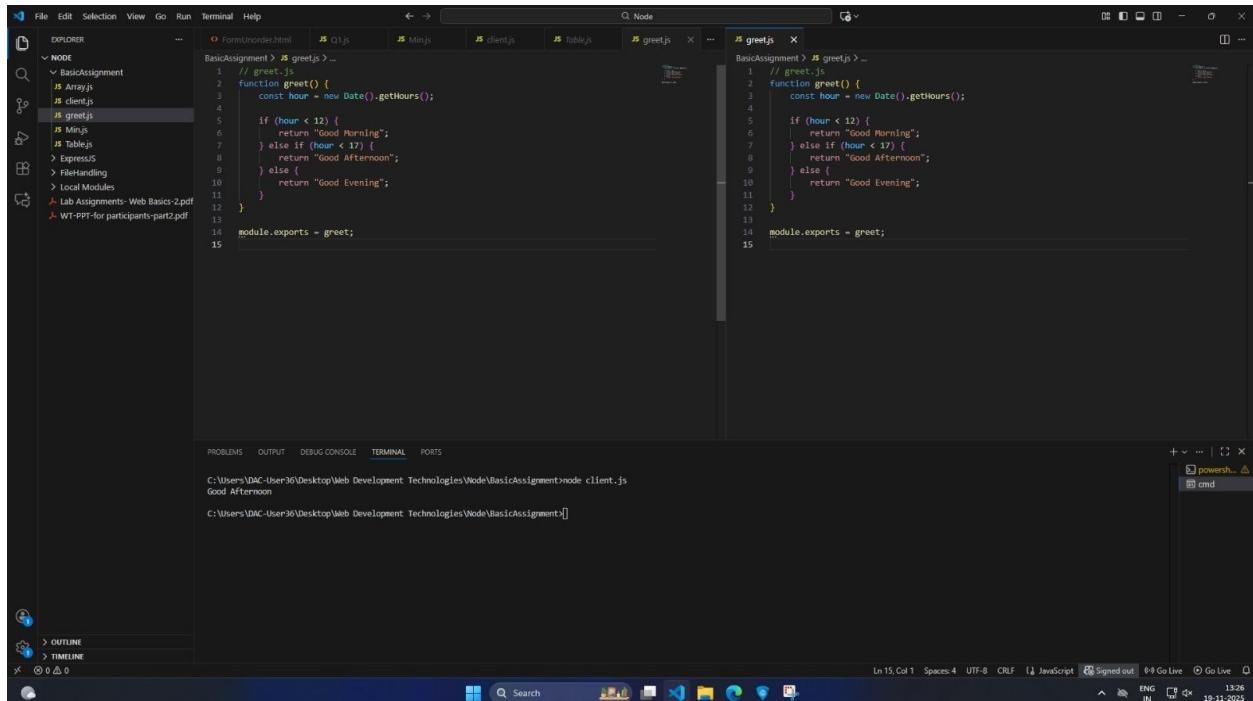
```
        return "Good Evening";
```

```
}
```

```
}
```

```
module.exports = greet;  
  
// client.js const greet =  
  
require('./greet');
```

```
console.log(greet());
```



```
// greet.js  
function greet() {  
    const hour = new Date().getHours();  
  
    if (hour < 12) {  
        return "Good Morning";  
    } else if (hour < 17) {  
        return "Good Afternoon";  
    } else {  
        return "Good Evening";  
    }  
}  
  
module.exports = greet;
```

```
// client.js  
const greet = require('./greet');  
  
greet();
```

Q2.

```
// calc.js module.exports =
```

```
{
```

```
    add: (a, b) => a + b,
```

```
    subtract: (a, b) => a - b,
```

```
    multiply: (a, b) => a * b,
```

```
    divide: (a, b) => {
```

```
if (b === 0) return "Cannot divide by zero";

return a / b;
},

square: (a) => a * a,

min: (a, b, c) => {
    let smallest
    = a; if (b < smallest) smallest =
    b; if (c < smallest) smallest = c;
    return smallest;
},

max: (a, b, c) => {
    let
    largest = a; if (b > largest)
    largest = b; if (c > largest)
    largest = c; return largest;
}

};

// calcClient.js

const calc = require('./calc');

console.log("Add:", calc.add(5, 3)); console.log("Subtract:",
calc.subtract(10, 4)); console.log("Multiply:", calc.multiply(6,
2)); console.log("Divide:", calc.divide(20, 5));

console.log("Square:", calc.square(7)); console.log("Min:",
calc.min(10, 5, 8)); console.log("Max:", calc.max(10, 5, 8));
```

The screenshot shows the Visual Studio Code interface with several tabs open at the top: Q1.js, Min.js, client.js, Table.js, greet.js, calcClient.js, green.js, client.js, and calc.js. The calcClient.js tab is selected. In the Explorer sidebar, there's a 'NODE' folder containing files like BasicAssignment, Array.js, calc.js, calcClient.js, client.js, greet.js, Min.js, Table.js, ExpressJS, FileHandling, Local Modules, Lab Assignments- Web Basics-2.pdf, and WT-PPT-for participants-part2.pdf. The main editor area contains code for a basic assignment. The terminal below shows the output of running client.js, which includes console.log statements for various operations like Add, Subtract, Multiply, Divide, Square, Min, and Max.

```

BasicAssignment > node calcClient.js
1 // calcClient.js
2 const calc = require('./calc');
3
4 console.log("Add:", calc.add(5, 3));
5 console.log("Subtract:", calc.subtract(10, 4));
6 console.log("Multiply:", calc.multiply(6, 2));
7 console.log("Divide:", calc.divide(20, 5));
8 console.log("Square:", calc.square(7));
9 console.log("Min:", calc.min(10, 5, 8));
10 console.log("Max:", calc.max(10, 5, 8));
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30

```

```

C:\Users\DAC-User36\Desktop\Web Development Technologies\Node\BasicAssignment>node client.js
Good Afternoon
Add: 8
Subtract: 5
Multiply: 12
Divide: 4
Square: 49
Min: 5
Max: 10

```

Q3.

```

// circle.js module.exports = { calcArea: (radius) => Math.PI * radius * radius,
calcCircumference: (radius) => 2 * Math.PI * radius,
calcDiameter: (radius) => 2 * radius

}; // rectangle.js module.exports = { calcArea: (length, breadth) => length * breadth,
calcPerimeter: (length, breadth) => 2 * (length + breadth)

}; // triangle.js module.exports = { isEquilateral: (a, b, c) => a === b && b === c,
calcPerimeter: (a, b, c) => a + b + c

}; // shapesClient.js const circle = require('./circle'); const rect = require('./rectangle'); const tri =
require('./triangle');

```

```

console.log("Circle Area:", circle.calcArea(5)); console.log("Circle Circumference:",
circle.calcCircumference(5)); console.log("Circle Diameter:", circle.calcDiameter(5));

console.log("Rectangle Area:", rect.calcArea(10, 5)); console.log("Rectangle Perimeter:",
rect.calcPerimeter(10, 5));

console.log("Triangle is Equilateral:", tri.isEquilateral(5, 5, 5)); console.log("Triangle Perimeter:",
tri.calcPerimeter(5, 6, 7));

```

The screenshot shows a Node.js development environment with the following details:

- Explorer Panel:** Shows files in the `NODE` folder, including `BasicAssignment`, `Array.js`, `calc.js`, `calcClient.js`, `circle.js`, `client.js`, `greet.js`, `Min.js`, `rectangle.js`, and `shapesClient.js`.
- Editor:** Displays the contents of `shapesClient.js` with code for calculating areas and perimeters of shapes.
- Terminal:** Shows the command `node shapesClient.js` being run, followed by the output of the program's calculations.
- Bottom Status Bar:** Includes information like file path, line number, column number, spaces, encoding, and date.

```

BasicAssignment > JS shapesClient.js > ...
1 // shapesClient.js
2 const circle = require('./circle');
3 const rect = require('./rectangle');
4 const tri = require('./triangle');
5
6 console.log("Circle Area:", circle.calcArea(5));
7 console.log("Circle Circumference:", circle.calcCircumference(5));
8 console.log("Circle Diameter:", circle.calcDiameter(5));
9
10 console.log("Rectangle Area:", rect.calcArea(10, 5));
11 console.log("Rectangle Perimeter:", rect.calcPerimeter(10, 5));
12
13 console.log("Triangle is Equilateral:", tri.isEquilateral(5, 5, 5));
14 console.log("Triangle Perimeter:", tri.calcPerimeter(5, 6, 7));
15

```

```

C:\Users\DAC-User\36\Desktop\Web Development Technologies\Node\BasicAssignment>node calcClient.js
Add: 6
Subtract: 6
Multiply: 12
Divide: 4
Square: 49
Min: 1
Max: 18

C:\Users\DAC-User\36\Desktop\Web Development Technologies\Node\BasicAssignment>node shapesClient.js
Circle Area: 78.53981633974483
Circle Circumference: 31.41592653589793
Circle Diameter: 18
Rectangle Area: 50
Rectangle Perimeter: 30
Triangle is Equilateral: true
Triangle Perimeter: 18

```

## File Handling :

Q1.

```
// names.js
const fs =
require('fs');
```

```
let arr = ["aaa", "bbb", "ccc"];
```

```
// Convert array → string with pipe ()
```

```
let data = arr.join("|");
```

```

// Write to file names.txt fs.writeFile("names.txt",
data, (err) => {
  if (err) {
    console.log("Error writing file:", err);
  } else {
    console.log("names.txt written successfully!");
  }
});

```

The screenshot shows the Visual Studio Code interface with the following details:

- Explorer View:** Shows files in the 'BasicAssignment' folder, including Q1.js, Min.js, client.js, Table.js, and names.js.
- Code Editor:** Displays the 'names.js' file content. The code uses the fs module to write an array to a file.
- Terminal:** Shows the command line output of running the script. It shows the file being created and the contents "aa|bbb|ccc".
- Status Bar:** Provides information like file path (C:\Users\DAC-User30\Desktop\Web Development Technologies\Node\BasicAssignment), line and column numbers (Ln 15, Col 6), and encoding (UTF-8).

Q2.

```
const fs = require('fs');
```

```

fs.readFile('emp.txt', 'utf8', (err, data) =>
{
  if (err) {
    console.error("File read error:", err);
  }
  return;
}

```

```
}

let total = 0;

const lines = data.split('\n');

lines.forEach(line =>
{
    if (line.trim().length > 0)
    {
        const parts =
line.split(':');
        const
salary = parseInt(parts[3]);
total += salary;
    }
});

console.log("Total Salary:", total);
});
```

The screenshot shows the Visual Studio Code interface. The Explorer sidebar on the left displays a project structure under 'WEB HTML DAC'. The 'File Handling' folder contains several files: emp.txt, filewrite.js, node\_modules, app.js, package-lock.json, package.json, Append.js, ChainedPromise.js, CheckAllfileExist.js, copy.txt, CopyFile.js, CreateFolder.js, Deletefile.js, DeleteFolder.js, First.js, module1.js, module2.js, newname.txt, Promise.js, ReadAllfiles.js, ReadFile.js, Rename.js, usemodule1.js, useModule2.js, Assignment-5.html, Assignment-6.html, Assignment-7.html, and Assignment-7.2.html. The 'Terminal' tab at the bottom shows command-line output from running 'node Emp.js', which reads data from 'emp.txt', processes it, and calculates a total salary of 78000.

```

const fs = require('fs');

fs.readFile('emp.txt', 'utf8', (err, data) => {
  if (err) {
    console.error("File read error:", err);
    return;
  }

  let total = 0;

  const lines = data.split('\n');

  lines.forEach(line => {
    if (line.trim().length > 0) {
      const parts = line.split(':');
      const salary = parseInt(parts[3]);
      total += salary;
    }
  });
});

```

Q3.

```
const fs = require('fs');
```

```
let employees = [
```

```
  { id: 1001, name: "Harry", dept: "Sales", salary: 23000 },
  { id: 1002, name: "Sarita", dept: "Accounts", salary: 20000 },
  { id: 1003, name: "Monika", dept: "TechSupport", salary: 35000 },
  { id: 1004, name: "Rohit", dept: "HR", salary: 25000 }
```

```
];
```

```
let dataToWrite = "";
```

```
employees.forEach(emp => {
  dataToWrite += `${emp.id}:${emp.name}:${
  emp.dept}: ${emp.salary}\n`;
});
```

```

fs.writeFile("employee_output.txt", dataToWrite, (err) => {
  if (err) console.log("Error writing file:", err); else
  console.log("Employee data saved successfully.");
});

```

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows a folder structure for a project named "myfirstnodeproject". Inside the "FileHandling" folder, there are files: ChainedPromise.js, Emp.js, Write.js (which is the active file), and Write.txt.
- Code Editor:** Displays the content of the "Write.js" file. The code uses the fs module to write employee data to a file. It defines an array of employees and a variable "dataToWrite". It then loops through the array, concatenating each employee's ID and name separated by a colon. Finally, it writes this data to the "employee\_output.txt" file.
- Terminal:** Shows the command "node Write.js" being run, followed by the output "Employee data saved successfully."

```

const fs = require('fs');

let employees = [
  { id: 1001, name: "Harry", dept: "Sales" },
  { id: 1002, name: "Sarita", dept: "Accounts" },
  { id: 1003, name: "Monika", dept: "TechSupport" },
  { id: 1004, name: "Rohit", dept: "HR", sa
];

let dataToWrite = "";
employees.forEach(emp => {
  dataToWrite += `${emp.id}:${emp.name}:`;
});

fs.writeFile("employee_output.txt", dataToWrite, (err) => {
  if (err) console.log("Error writing file:");
  else console.log("Employee data saved successfully.");
});

```

Q4.

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

```

fs.readFile("customer.json", "utf-8", (err, data) => {    if
(err) {
  console.log("Error reading file:", err);
  return;
}

let customers = JSON.parse(data);

```

```

customers.forEach(c      =>
{
    console.log(Name: ${c.custname});

    console.log(Phone:      ${c.phno});

    console.log(Address:     ${c.address});

    console.log("-----");

    });
});

```

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows the project structure under "myfirstnodeproject" with files like customer.json, Employee\_output, Read.js, Write.js, and various module files.
- Terminal:** Displays the code for `Read.js` and its execution output.
- Output:** Shows the JSON data from `customer.json` and the results of the `forEach` loop.

```

const fs = require("fs");
fs.readFile("customer.json", "utf-8", (err, data) => {
  if (err) {
    console.log("Error reading file:", err);
    return;
  }
  let customers = JSON.parse(data);
  customers.forEach(c => {
    console.log(`Name: ${c.custname}`);
    console.log(`Phone: ${c.phno}`);
    console.log(`Address: ${c.address}`);
    console.log("-----");
  });
});

```

```

PS C:\Users\vipul\OneDrive\Desktop\Web Html DAC\Nodejs\myfirstnodeproject\FileHandling> node Read.js
Name: Anil Patil
Phone: 8877656988
Address: A123
-----
Name: Anita Kulkarni
Phone: 99675456
Address: A102, Highstreet
-----
Name: Kavita Menon
Address: A102, Highstreet
-----
Address: A102, Highstreet
Address: A102, Highstreet
Address: A102, Highstreet
-----
Name: Kavita Menon
Phone: 123456789
Address: B8203, Pune

```

Q5.

```

const fs = require("fs");

fs.readFile("books.json", "utf-8", (err, data) => {    if
(err) {

    console.log("Error reading json:", err);

    return;
}

```

```
 }
```

```
let books = JSON.parse(data);
```

```
let output = "Bookid | bookname | author | Sellingprice | finalprice\n";
```

```
books.forEach(b => {      let finalPrice = b.sellingprice -  
(b.sellingprice * 0.10);      output += `${b.bookid}|$  
{b.bookname}|${b.author}|${b.sellingprice}|${finalPrice}  
\n`;  
});
```

```
fs.writeFile("book.txt", output, (err) => {      if  
(err) console.log("Error writing file:", err);  
else console.log("File saved as book.txt");  
});  
});
```

```

File Edit Selection View Go Run Terminal Help ← → Q Web HTML DAC
File Edit Selection View Go Run Terminal Help ← → Q Web HTML DAC
book_write.js | book.json | book_write.js | books.json | book_write.js |
Node.js > mynodeproject > Handling > book_write.js > fs.readFile('books.json', 'utf-8') callback
Node.js > mynodeproject > Handling > books.json
1 const fs = require('fs');
2
3 fs.readFile("books.json", "utf-8", (err, data) => {
4     if (err) {
5         console.log("Error reading json:", err);
6         return;
7     }
8
9     let books = JSON.parse(data);
10
11     let output = "BookId | Bookname | Author | SellingPrice | FinalPrice\n";
12
13     books.forEach(b => {
14         let finalPrice = b.sellingprice - (b.sellingprice * 0.10);
15         output += `${b.bookId}|${b.bookname}|${b.author}|${b.sellingprice}|${finalPrice}\n`;
16     });
17
18     fs.writeFile("book.txt", output, (err) => {
19         if (err) console.log("Error writing file:", err);
20         else console.log("File saved as book.txt");
21     });
22 });
23

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\saiprakash\Desktop\Node.js\myNodeProject\Handling> node book\_write.js  
File saved as book.txt  
PS C:\Users\saiprakash\Desktop\Node.js\myNodeProject\Handling>

Q6.

`const fs = require('fs');`

`fs.readFile('info.txt', 'utf8', (err, data) =>`

```
{  if (err) {

    console.log("Error reading file:", err);

    return;
}
```

`const lines = data.split('\n');`

```
lines.forEach((line, index) => {      console.log(`$ ${index + 1}. ${line}`);
});
```

```

const fs = require('fs');
if (err) {
    console.log("Error reading file:", err);
    return;
}
const lines = data.split('\n');
lines.forEach((line, index) => {
    console.log(`${index + 1}. ${line}`);
});

```

## Node HTTP & the web Q1.

```
const http = require('http'); const
```

```
greet = require('./Greet');
```

```
const server = http.createServer((req, res) => {
```

```
    const message = greet.getGreeting();
```

```
        res.writeHead(200, { "Content-Type": "text/plain" });    res.end(message);
```

```
});
```

```
server.listen(3000, () => {    console.log("Server running on
```

```
http://localhost:3000");
```

```
});
```

The screenshot shows a Microsoft Edge browser window displaying the text "Good Evening!" at the top. Below it is the VS Code interface. The Explorer sidebar shows a project structure with files like Read.js, book\_write.js, read\_lines.js, Greet.js, and server.js. The code editor contains a Node.js script (server.js) with the following code:

```

1 const http = require('http');
2 const greet = require('./Greet');
3
4 const server = http.createServer((req, res) =>
5   const message = greet.getGreeting();
6
7   res.writeHead(200, { "Content-Type": "text/plain" });
8   res.end(message);
9 );
10
11 server.listen(3000, () => {
12   console.log("Server running on http://localhost:3000");
13 });
14

```

The Terminal tab shows the command "PS C:\Users\vipul\OneDrive\Desktop\Web HTML DAC\Nodejs\myfirstnodeproject\FileHandling> node server.js" and the output "Server running on http://localhost:3000".

## EXPRESSJS :

Q1.      const      express      =

require("express");    const    app    =

express();      const      path      =

require("path");    app.get("/",    (req,

res)                  =>

{    res.sendFile(path.join(\_\_dirname  
, "form.html"));

});

app.get("/display", (req, res) =>

{    const { s1, s2, s3 } = req.query;

res.send(`

<h2>The parameters are</h2>

```

<ul>
  <li>${s1}</li>
  <li>${s2}</li>
  <li>${s3}</li>
</ul>
`);

});

};


```

```

app.listen(3000, () => {  console.log("Server running at
http://localhost:3000");
});
```

The screenshot shows a Node.js development environment with two main windows:

- Code Editor (server.js):**

```

1 const express = require("express");
2 const app = express();
3 const path = require("path");
4
5 app.get("/", (req, res) => {
6   res.sendFile(path.join(__dirname, "Form.html"));
7 });
8
9 app.get("/display", (req, res) => {
10   const { s1, s2, s3 } = req.query;
11
12   res.send(`<h2>The parameters are</h2>
13   <ul>
14     <li>${s1}</li>
15     <li>${s2}</li>
16     <li>${s3}</li>
17   </ul>
18 `);
19 });
20
21
22 app.listen(3000, () => {
23   console.log("Server running at http://localhost:3000");
24 });
25 
```
- Browser Output:**

The browser window shows the rendered HTML output: "The parameters are" followed by an ul list containing three li items: Vipul, Sunil, and Talele.

Q2.      const      express      =  
 require("express");    const    path    =

```
require("path");    const    app    =
express();

app.get("/", (req, res) => {    res.sendFile(path.join(__dirname,
"simpleInt.html"));

});

app.get("/calc", (req, res) => {    const
p = parseFloat(req.query.p);    const n
= parseFloat(req.query.n);    const r =
parseFloat(req.query.r);

const si = (p * n * r) / 100;

res.send(`

<h2>Simple Interest Result</h2>

<p>Principal: ${p}</p>
<p>Years: ${n}</p>
<p>Rate: ${r}</p>
<h3>Simple Interest = ${si}</h3>

`);

});

app.listen(3000, () => {    console.log("Server running at
http://localhost:3000");
});
```

The screenshot shows a development environment with three main windows:

- Code Editor:** Displays a file named `simpleint_server.js` containing Express.js code for calculating simple interest.
- Terminal:** Shows the command `node simpleint_server.js` being run, with the output "Server running at http://localhost:3000".
- Browser:** Shows the result of the calculation `localhost:3000/calc?p=50000&n=2&r=7`, displaying "Simple Interest Result" with the output "Simple Interest = 7000".

```

const express = require("express");
const app = express();

// Middleware to read form data
app.use(express.urlencoded({ extended: true }));

// Hardcoded users array let
users = [
  { uname: "shrilata", pass: "secret" },
  { uname: "admin", pass: "admin123" },
  { uname: "user1", pass: "pass1" }
];

```

Q3.

```

const express = require("express");
const app = express();

```

```

// Middleware to read form data
app.use(express.urlencoded({ extended: true }));

```

```

// Hardcoded users array let
users = [
  { uname: "shrilata", pass: "secret" },
  { uname: "admin", pass: "admin123" },
  { uname: "user1", pass: "pass1" }
];

```

```
// Serve login.html as homepage
app.get("/", (req, res) =>
{ res.sendFile(__dirname + "/login.html");
});

app.post("/login", (req, res) =>
{ let uname = req.body.uname;
let pass = req.body.pass;

let found = users.find(u => u.uname === uname && u.pass === pass);

if (found) {
  res.sendFile(__dirname + "/success.html");
} else {
  res.sendFile(__dirname + "/failure.html");
}
});

app.listen(3000, () => { console.log("Server running on
http://localhost:3000");
});
```

The screenshot shows a Node.js project structure in the Explorer panel. The project includes files like app.js, simple.html, simple\_server.js, login.html, and server.js. The server.js file contains code for an Express application. The browser tab shows the result of running the application at localhost:3000/login, displaying the message "Login Successful!".

```

const express = require("express");
const bodyParser = require("body-parser");
const path = require("path");

const app = express();
app.use(bodyParser.urlencoded({ extended: true }));
app.use(express.static("public"));

app.get("/", (req, res) => {
  res.sendFile(__dirname + "/login.html");
});

app.post("/login", (req, res) => {
  let uname = req.body.uname;
  let pass = req.body.pass;

  let found = users.find(u => u.uname === uname && u.pass === pass);

  if (found) {
    res.sendFile(__dirname + "/success.html");
  } else {
    res.sendFile(__dirname + "/failure.html");
  }
});

app.listen(3000, () => {
  console.log("Server running on http://localhost:3000");
});

```

Q4. const express = require("express"); const

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

path = require("path");

const app = express();

const PORT = 3000;

let users = [];

app.use(bodyParser.urlencoded({ extended: true })); app.use(express.static("public"));

app.post("/signup", (req, res) => { const

{ username, password } = req.body;

```

const userObj = { username, password }; users.push(userObj);

console.log("Current Users Array:", users);

res.send("<h2>Signup Successful!</h2><p>User stored in array.</p>");
});

app.listen(PORT, () => { console.log(`Server running at
http://localhost:${PORT}`);
});

```



Q5.     const     express     =  
 require("express");    const    app    =  
 express();

```

let studs = [  

  {sid:101, sname:"Savita", course:"DloT"},  

  {sid:102, sname:"Kavita", course:"DAC"},  

  {sid:103, sname:"Anita", course:"DESD"},  

  {sid:104, sname:"Sunita", course:"DloT"},  

  {sid:105, sname:"Babita", course:"DMC"}  

];
```

```

// RESTful URL: localhost:3000/students/Savita
app.get("/students/:name", (req, res) => {
    let
name = req.params.name;

// search in array (case insensitive)    let student = studs.find(s =>
s.sname.toLowerCase() === name.toLowerCase());

if (!student) {
    return res.send(`<h2>Student not found</h2>`);

}

res.send(`

<h3>Student id : ${student.sid}</h3>
<h3>Student name : ${student.sname}</h3>
<h3>Student course : ${student.course}</h3>
<br><a href="/students">Go Back to List</a>
`);

});

// list of all students app.get("/students", (req, res) => {
let html =
"<h2>Student List</h2><ul>";
studs.forEach(s => {
    html += `<li><a href="/students/${s.sname}">${s.sname}</a></li>`;
});
html += "</ul>";

res.send(html);

});

```

```
app.listen(3000, () => {   console.log("Server running at
http://localhost:3000");
});
```

The screenshot shows a development environment with two main windows:

- VS Code (Left):** The Explorer sidebar shows a project structure under "WEB HTML DAC". The "Nodes" folder contains files like "emp.js", "EmployeeOutput.js", "index.html", "read\_lines.js", "Read.js", "server.js", and "Write.js". The "server.js" file is open in the editor, displaying code for an Express.js application that handles student data.
- Microsoft Edge (Right):** A browser window is open at "localhost:3000/students/Savita". The page displays the following information:
  - Student id :** 101
  - Student name :** Savita
  - Student course :** DlOT
 Below this, there is a link: [Go Back to List](#).

```
Q6. const express = require("express"); const
students = require("./data/students"); const
app = express();
```

```
app.use(express.static("public"));
```

```
// API: All students
app.get("/students", (req, res) => {  res.json(students);
});
```

```
// API: Search students app.get("/search",
  (req, res) => { const name =
    req.query.name?.toLowerCase(); const
    result = students.filter(s =>
      s.name.toLowerCase().includes(name));
    res.json(result);
  });
}
```

```
app.listen(3000, () => { console.log("Server running on
http://localhost:3000");
});
}
```

```
{
  "name": "studentapp",
  "version": "1.0.0",
  "main": "server1.js",
  "scripts": {
    "start": "node server1.js"
  },
  "dependencies": {
    "express": "^4.18.2"
  }
}
```

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
    <title>All Students</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>

<h1>All Students</h1>

<div id="studentList"></div>

<script>
fetch("/students")
    .then(res => res.json())
    .then(data =>
{
    document.getElementById("studentList").innerHTML =
        data.map(s => `
            <div class="card">
                <h3>${s.name}</h3>
                <p>Age: ${s.age}</p>
                <p>Course: ${s.course}</p>
            </div>
        `).join("");
});
</script>

</body>
```

```
</html>

<!DOCTYPE html>
<html>
<head>
    <title>Search Student</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>

<h1>Search Student</h1>

<input type="text" id="searchBox" placeholder="Enter student name">
<button onclick="searchStudent()">Search</button>

<div id="result"></div>

<script> function searchStudent() {           let name =
document.getElementById("searchBox").value;

fetch(`/search?name=${name}`)
    .then(res => res.json())
    .then(data => {      if (!
data.length) {
        document.getElementById("result").innerHTML = "<p>No student found</p>";
    }
    return;
}
</script>
```

```
}

document.getElementById("result").innerHTML =
data.map(s => `

<div class="card">

<h3>${s.name}</h3>

<p>Age: ${s.age}</p>

<p>Course: ${s.course}</p>

</div>

`).join("");

});

</script>

</body>
</html>

module.exports = [
  { id: 1, name: "Rahul", age: 20, course: "BCA" },
  { id: 2, name: "Sneha", age: 22, course: "BBA" },
  { id: 3, name: "Aman", age: 21, course: "BSc IT" },
  { id: 4, name: "Priya", age: 23, course: "MBA" }
];
```

```

const express = require("express");
const students = require("./data/students");
const app = express();
app.use(express.static("public"));
// API: All students
app.get("/students", (req, res) => {
  res.json(students);
});
// API: Search students
app.get("/search", (req, res) => {
  const name = req.query.name.toLowerCase();
  const result = students.filter(s => s.name.toLowerCase() === name);
  res.json(result);
});
app.listen(3000, () => {
  console.log("Server running on http://localhost:3000");
});

```

**All Students**

- Rahul**  
Age: 20  
Course: BCA
- Sneha**  
Age: 22  
Course: BBA
- Aman**  
Age: 21  
Course: BSc IT
- Priya**  
Age: 23  
Course: MBA

## NEXT GEN JAVASCRIPT ASS :

Q1. function transformToObjects(numberArray)

```
{   return numberArray.map(num => ({ val: num }));}
```

```
console.log(transformToObjects([1, 2, 3]));
```

The screenshot shows a code editor interface with the following details:

- Explorer View:** Shows the project structure under "WEB HTML DAC". Key files include `Nodejs`, `myfirstnodeproject`, `FileHandling`, `express_assignment`, `book_write.js`, `book.txt`, `books.json`, `customer.json`, `Emp.js`, `empty.txt`, `employee_output...`, `filewrite.js`, `Greet.js`, `info.txt`, `read_lines.js`, `Read.js`, `server.js`, `Write.js`, `node_modules`, `app.js`, `package-lock.json`, `package.json`, `node_modules`, `StudentApp`, `data`, `students.js`, `transformToObj...`, `public`, `search.html`, `students.html`, `style.css`, `package.json`, `server1.js`, `Append.js`, `ChainedPromise.js`, `CheckAllFileExist.js`, `copy.txt`, `Copyfile.js`, `CreateFolder.js`, `DeleteFile.js`, `DeleteFolder.js`, `First.js`, `module1.js`, `module2.js`, `newname.txt`, `package-lock.json`, `package.json`, `Promise.js`.
- Code Editor:** Displays a file named `transformToObjects.js` with the following content:

```

1 function transformToObjects(numberArray) {
2   return numberArray.map(num => ({ val: num }));
3 }
4
5 // Test:
6 console.log(transformToObjects([1, 2, 3]));
7

```
- Terminal:** Shows the output of running the script:

```

PS C:\Users\vipul\OneDrive\Desktop\Web Html DAC\Nodejs\StudentApp\data> node transformToObjects.js
>>
[ { val: 1 }, { val: 2 }, { val: 3 } ]
PS C:\Users\vipul\OneDrive\Desktop\Web Html DAC\Nodejs\StudentApp\data>

```
- Status Bar:** Shows the current file is `transformToObjects.js`, and the terminal is set to `powershell`. Other tabs in the status bar include `PROBLEMS`, `OUTPUT`, `DEBUG CONSOLE`, `TERMINAL`, `PORTS`, and `JavaScript`.

Q2.

```
<!DOCTYPE html>

<html>

<head>

<title>Product Table</title>

</head>

<body>
```

`<h2>Product List</h2>`

`<table border="1" cellpadding="6">`

```

<thead>
  <tr>
    <th>Product ID</th>
    <th>Name</th>
    <th>Price</th>
    <th>Category</th>
  </tr>
</thead>
<tbody id="productBody"></tbody>
</table>

<script>  let
  productArr = [
    { pid: 1001, prodName: "Gaming Headset", price: 3000, category: "Headset" },
    { pid: 1002, prodName: "Lego Car", price: 1500, category: "toys" },
    { pid: 1003, prodName: "Football", price: 800, category: "sports" },
    { pid: 1004, prodName: "Toy Train", price: 1200, category: "toys" }
  ];
let tableBody = document.getElementById("productBody");
productArr.forEach(product =>
{
  tableBody.innerHTML += `
<tr>
<td>${product.pid}</td>
<td>${product.prodName}</td>

```

```

        <td>${product.price}</td>
        <td>${product.category}</td>
    </tr>
};

});

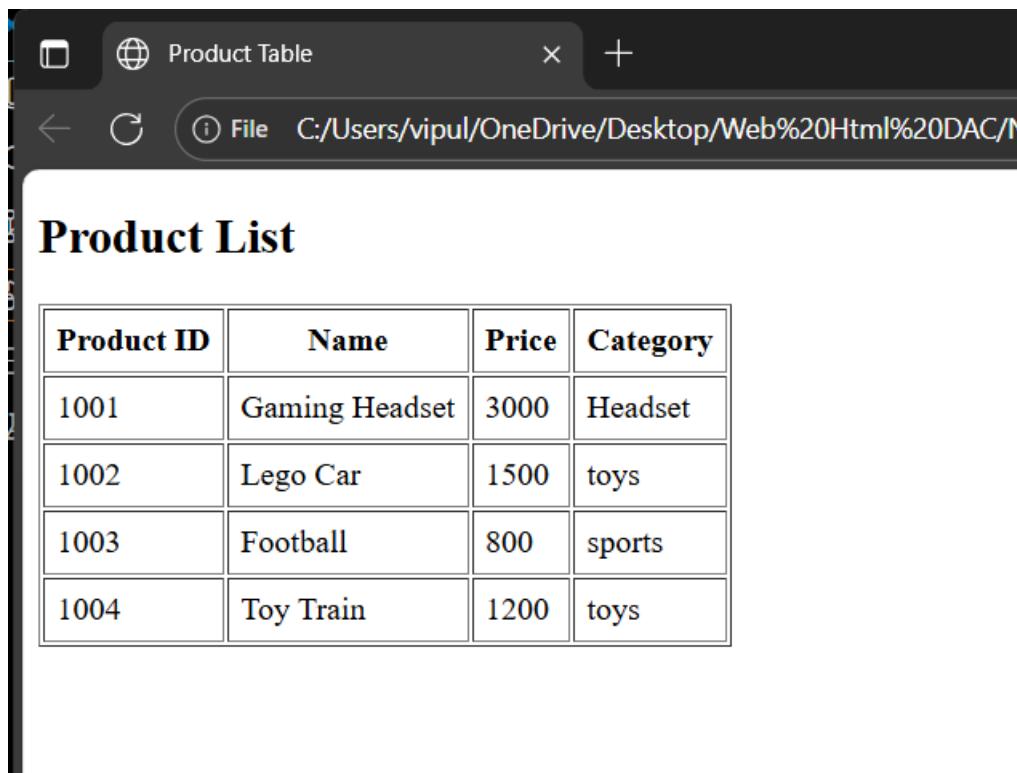
</script>

```

```

</body>
</html>

```



The screenshot shows a web browser window titled "Product Table". The address bar indicates the file is located at "C:/Users/vipul/OneDrive/Desktop/Web%20Html%20DAC/N". The main content area displays a table titled "Product List" with four columns: Product ID, Name, Price, and Category. The table contains four rows of data.

Product ID	Name	Price	Category
1001	Gaming Headset	3000	Headset
1002	Lego Car	1500	toys
1003	Football	800	sports
1004	Toy Train	1200	toys

Q3.

```
<!DOCTYPE html>
```

```
<html>
<head>
    <title>Toy Products</title>
</head>
<body>

<h2>Toy Products (Category: toys)</h2>

<table border="1" cellpadding="6">
    <thead>
        <tr>
            <th>Product ID</th>
            <th>Name</th>
            <th>Price</th>
            <th>Category</th>
        </tr>
    </thead>
    <tbody id="toyBody"></tbody>
</table>

<script>

let productArr = [
    { pid: 1001, prodName: "Gaming Headset", price: 3000, category: "Headset" },
    { pid: 1002, prodName: "Lego Car", price: 1500, category: "toys" },
    { pid: 1003, prodName: "Football", price: 800, category: "sports" },
]
```

```
{ pid: 1004, prodName: "Toy Train", price: 1200, category: "toys" }  
];  
  
let ToyArray = productArr.filter(product => product.category === "toys");  
  
let toyBody = document.getElementById("toyBody");  
  
ToyArray.forEach(product =>  
{  
    toyBody.innerHTML += `  
        <tr>  
            <td>${product.pid}</td>  
            <td>${product.prodName}</td>  
            <td>${product.price}</td>  
            <td>${product.category}</td>  
        </tr>  
    `;  
};  
});  
</script>  
  
</body>  
</html>
```

Toy Products

← ⌂ ⌂ File C:/Users/vipul/OneDrive/Desktop/Web%20Html%20DAC/Nodejs/Third.html

## Toy Products (Category: toys)

Product ID	Name	Price	Category
1002	Lego Car	1500	toys
1004	Toy Train	1200	toys