SAVITRIBAI PHULE PUNE UNIVERSITY

S. Y. B. B. A. (C.A.) SEMESTER IV
(CBCS 2019 PATTERN)

PRACTICAL SLIP

NAME : LALIT DEVIDAS PATIL
COLLEGE NAME: SINHGAD COLLEGE OF ARTS & COMMERCE WARJE PUNE-58
ROLL NO : 106 DIVISION:B SEAT NO: | | | | | |

ACADEMIC YEAR : 2023-24

Certificate

This is to certify that
Mr. PATIL LALIT DEVIDAS

Seat Number_____of S.Y.BBA(CA) Sem- IV has Successfully completed Laboratory course (NODE JS) in the Year . He has scored mark out of 10 (For Lab Book).

Subject Teacher **H.O.D./Coordinator**

Internal Examiner **External Examiner**

Slip 11

A) - Create A Simple Web Server Using Node.Js That Shows The College Information.

```
const http = require('http');
const CollegeInfo = {
   name: "SCOAC",
   location: "Warje Pune",
   website:"http://www.sinhgad.edu/sinhgad-institutes-
ACS/college-pages/SCOAC_Sr/Principals_Desk.html",
   description: "SCOAC College is a prestigious institution known for
its academic excellence and vibrant campus life."
};
const server = http.createServer((req, res) => {
   res.writeHead(200, { 'Content-Type': 'text/html' });
   res.write(`<h1>${CollegeInfo.name}</h1>`);
   res.write(`<p><strong>Location:</strong>
${CollegeInfo.location}</p>`);
   res.write(`<p><strong>Website:</strong> <a
href="${CollegeInfo.website}">${CollegeInfo.website}</a></p>`)
;
   res.write(`<p><strong>Description:</strong>
${CollegeInfo.description}</p>`);
   res.end();
});
const PORT = process.env.PORT || 8000;
server.listen(PORT, () => {
   console.log(`Server running on port ${PORT}`);
});
```

```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_11_A.JS
Server running on port 8000
```

localhost:8000

localhost:8000

SCOAC

Location: Warje Pune

Website: http://www.sinhgad.edu/sinhgad-institutes-ACS/college-pages/SCOAC_Sr/Principals_Desk.html

Description: SCOAC College is a prestigious institution known for its academic excellence and vibrant campus life.

B) Using Node.js Create a Computer Science Department Portal.

```
const express = require('express');
const app = express();
const port = 1111;
const courses = [
    { id: 1, name: 'Node JS', credits: 3 },
    { id: 2, name: 'Object Oriented Concept Through CPPS', credits: 3},
    { id: 3, name: 'Operating System', credits: 3 }];
const faculty = [
    { id: 101, name: 'Mrs. Dipashri Mokashi', specialization: 'Node JS' },
    { id: 102, name: 'Mr. Pradeep Shitole', specialization: 'Object Oriented Concept Through CPPS' },
    { id: 103, name: 'Mrs. Kanchan Pavate', specialization: 'Operating System' }];
app.get('/', (req, res) => {
    res.send('Welcome to the Computer Science Department Portal');});
app.get('/courses', (req, res) => {
    res.json(courses);});
app.get('/faculty', (req, res) => {
    res.json(faculty);});
app.listen(port, () => {
    console.log(`Server is running at http://localhost:${port}`);});
```

```
PROBLEMS   DEBUG CONSOLE   OUTPUT   TERMINAL   PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_11_B.JS
Server is running at http://localhost:1111
```

localhost:1111

Welcome to the Computer Science Department Portal

localhost:1111/courses

Pretty print


```
[
  {
    "id": 1,
    "name": "Node JS",
    "credits": 3
  },
  {
    "id": 2,
    "name": "Object Oriented Concept Through CPPS",
    "credits": 3
  },
  {
    "id": 3,
    "name": "Operating System",
    "credits": 3
  }
]
```


localhost:1111/faculty

Pretty print


```
[
  {
    "id": 101,
    "name": "Mrs. Dipashri Mokashi",
    "specialization": "Node JS"
  },
  {
    "id": 102,
    "name": "Mr. Pradeep Shitole",
    "specialization": "Object Oriented Concept Through CPPS"
  },
  {
    "id": 103,
    "name": "Mrs. Kanchan Pavate",
    "specialization": "Operating System"
  }
]
```

Slip 12

A) Create Node.js Application That Binds Two Listeners to Single Events

```
const EventEmitter=require('events');
const eventEmitter= new EventEmitter();
eventEmitter.on('myEvent',()=>{
   console.log('First listener executed');
});
eventEmitter.on('myEvent', () => {
   console.log('Second listener exeecuted');
});
eventEmitter.emit('myEvent');
```

```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS    powershell +

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_12_A
First listener executed
Second listener exeecuted
PS C:\Node JS\Practical Slips>
```

B) Using Node.Js Create A User Login System

```
const mysql = require('mysql');
const express = require("express");
const bodyParser = require('body-parser');
const encoder = bodyParser.urlencoded;
const app = express();
const con = mysql.createConnection({
   host: "localhost",
   user: "LALIT PATIL",
   password: "l_patil__",
   port: "3306",
});
con.connect(function (error) {
   if (error) throw error;
   else console.log("connected");
});
app.get("/", function (req, res) {
   res.sendFile(__dirname + "/index.html");
});
app.post("/", encoder, function (req, res) {
   var username = req.body.username;
   var password = req.body.password;
   con.query("select * from loginuser where user_name=? and
user_pass=?", [username, password], function (error, results, fields)
{
      if (results.length > 0) {
         res.redirect("/Login");
      } else {
         res.redirect("/");
      }
      res.end();
   });
});
app.get("/Login", function (req, res) {
   res.sendFile(__dirname + "/Login.html");
});
app.listen(1106);
```

Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Login Page</title>
</head>
<body>
      <h2>Login</h2>
      <form action="#" method="post">
         <label for="username">Username:</label><br>
         <input type="text" id="username" name="username" required><br><br>
         <label for="password">Password:</label><br>
         <input type="password" id="password" name="password" required><br><br>
         <input type="submit" value="Login">
      </form>
   </div>
</body>
</html>
```

Login.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Document</title>
</head>
<body>
   <h1> WELCOME SUCCESSFULLY LOGIN</h1>
</body>
</html>
```

```
PROBLEMS    DEBUG CONSOLE    TERMINAL    ...    node

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_12_B.JS
connected
```

Login Page

localhost:1106

Login

Username:
lalitpatil2117@gmail.com

Password:
••••••••

Login

localhost:1106/Login

localhost:1106/Login

WELCOME SUCCESSFULLY LOGIN

Slip 13

A) Create Node.js Application Using User Defined Rectangle Module To find Area Of Rectangle and Display The Details On Console.

NODE_SLIP_13_A.JS

```
exports.calculateArea = function(length, width) {
   return length * width;
};
exports.calculatePerimeter = function(length, width) {
   return 2 * (length + width);
};
```

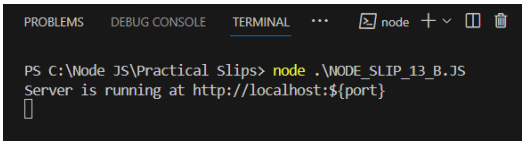
NODE_SLIP_13_A_APP.JS

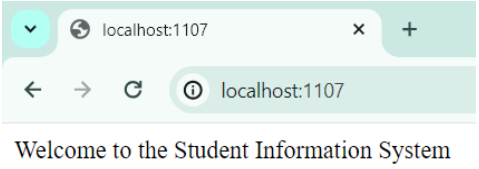
```
const rectangle = require('./NODE_SLIP_13_A.JS');
const length =11;
const width = 7;
const area = rectangle.calculateArea(length, width);
const perimeter = rectangle.calculatePerimeter(length, width);
console.log(`Rectangle with length ${length} and width ${width}:`);
console.log(`Area: ${area}`);
console.log(`Perimeter: ${perimeter}`);
```

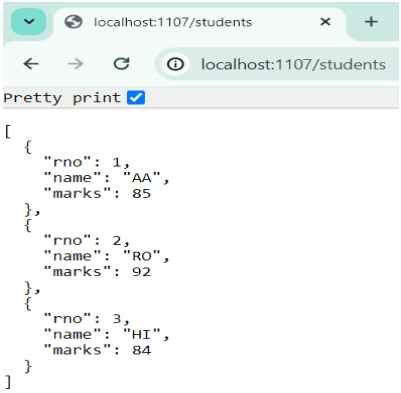
```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node ./NODE_SLIP_13_A_APP.JS
Rectangle with length 11 and width 7:
Area: 77
Perimeter: 36
PS C:\Node JS\Practical Slips>
```

B) Create A Node.Js Application That Update Marks Of Given Student Rno In "Student" Table And Display The Result.

```
const express = require('express');
const bodyParser = require('body-parser');
const app = express();
const port = 1107;
app.use(bodyParser.json());
let students = [
    { rno: 1, name: 'AA', marks: 85 },
    { rno: 2, name: 'RO', marks: 92 },
    { rno: 3, name: 'HI', marks: 84 },];
app.get('/', (req, res) => {
    res.send('Welcome to the Student Information System');});
app.get('/students', (req, res) => {
    res.json(students);});
app.put('/students/:rno', (req, res) => {
    const { rno } = req.params;
    const { marks } = req.body;
    const student = students.find(s => s.rno == rno);
    if (student) {
        student.marks = marks;
        res.json({message:'Marks updated successfully', updatedStudent:
student });
    } else {
        res.status(404).json({ error: 'Student not found' });
    }});
app.listen(port, () => {
    console.log('Server is running at http://localhost:${port}');
});
```







Slip 14

A) Create A Node.Js Application To Search A Particular Word In A File And Display Result On Console.

```
const fs = require('fs');
const readline = require('readline');
function searchWordInFile(fileName, word) {
    const rl = readline.createInterface({
        input: fs.createReadStream(fileName),
        output: process.stdout,
        terminal: false
    });
    rl.on('line', (line) => {
        if (line.includes(word)) {
            console.log(line);
        }
    });
    rl.on('close', () => {
        console.log('Search Complete.');
    });
}
const fileName = 'NODE_SLIP_14_A.TXT';
const wordToSearch = 'Sinhgad';
searchWordInFile(fileName, wordToSearch);
```

NODE_SLIP_14_A.TXT

Sinhgad College of Arts and Commerce popularly known as SCOAC.

```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_14_A.JS
Sinhgad College of Arts and Commerce popularly known as SCOAC.
Search Complete.
PS C:\Node JS\Practical Slips>
```

Slip 14
B) Using Node.js create an Electricity Bill Calculation System.

```
function calculateElectricityBill(units) {
    let totalBill = 0;
    if (units <= 50) {
        totalBill = units * 0.50;
    } else if (units <= 150) {
        totalBill = 50 * 0.50 + (units - 50) * 0.75;
    } else if (units <= 250) {
        totalBill = 50 * 0.50 + 100 * 0.75 + (units - 150) * 1.20;
    } else {
        totalBill = 50 * 0.50 + 100 * 0.75 + 100 * 1.20 + (units - 250) *
1.50;
    }
    return totalBill;
}
const unitsConsumed = 1000;
const billAmount = calculateElectricityBill(unitsConsumed);
console.log(`Electricity Bill Amount: ${billAmount.toFixed(2)}`);
```

```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_14_B.JS
Electricity Bill Amount: 1345.00
PS C:\Node JS\Practical Slips>
```

Slip 15

A) Create A Node.Js Application To Count Occurrence Of Given Word In A File And Display The Count On Console.

```
var fs = require("fs");
function countOcc(string, word) {
   return string.split(word).length - 1;
}
var text = fs.readFileSync('NODE_SLIP_15_A.TXT', 'utf8');
var count = countOcc(text, 'a'); // Use lowercase 'l' here
console.log(`The word "a" occurs ${count} times in the file.`);
```

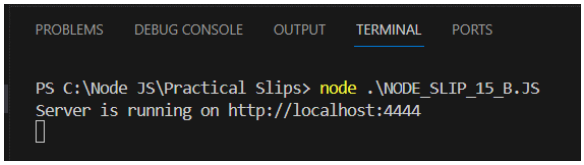
NODE_SLIP_15_A.TXT

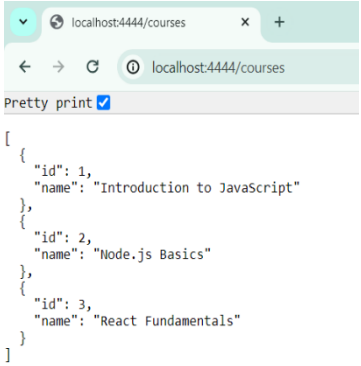
Sinhgad College of Arts and Commerce popularly known as SCOAC, has always strived for excellence in academics and complementing co-curricular learning that meets student expectations.
All our efforts are driven to make student life on college campus an enriching experience.
Our journey for the last decade has been extremely successful and satisfying in terms of accomplishments and accolades in scholastic, co-scholastic and infrastructural development areas..

```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_15_A.JS
The word "a" occurs 34 times in the file.
PS C:\Node JS\Practical Slips>
```

B) Using Node.js create a eLearning System.

```
const express = require('express');
const bodyParser = require('body-parser');
const app = express();
const PORT = 4444;
let courses = [
    { id: 1, name: 'Introduction to JavaScript' },
    { id: 2, name: 'Node.js Basics' },
    { id: 3, name: 'React Fundamentals' }];
app.use(bodyParser.json());
app.get('/courses', (req, res) => {
    res.json(courses);});
app.get('/courses/:id', (req, res) => {
    const id = parseInt(req.params.id);
    const course = courses.find(course => course.id === id);
    if (!course) {
        res.status(404).send('Course not found');
    } else {
        res.json(course); }});
app.post('/courses', (req, res) => {
    const { name } = req.body;
    if (!name) {
        res.status(400).send('Name is required');
  } else {
        const newCourse = {
            id: courses.length + 1,
            name: name   };
        courses.push(newCourse);
        res.status(201).json(newCourse);}});
app.listen(PORT, () => {
    console.log(`Server is running on http://localhost:${PORT}`);
});
```





Slip 16

A) Create A Node.Js File Named Main.Js For Event-Driven Application. There Should Be A Main Loop That Listens For Events, And Then Triggers A Callback Function When One Of Those Events Is Detected.

```
const EventEmitter = require('events');
const eventEmitter = new EventEmitter();
const handleMessage = (message) => {
   console.log('Message Received:', message);};
const handleData = (data) => {
   console.log('Data Received:', data);};
eventEmitter.on('messageReceived', handleMessage);
eventEmitter.on('dataReceived', handleData);
let messageCount = 0;
let dataCount = 0;
const maxMessages = 3;
const maxData = 2;
const intervalMessage = setInterval(() => {
   eventEmitter.emit('messageReceived', 'Hello World!');
   messageCount++;
   if (messageCount >= maxMessages) {
      clearInterval(intervalMessage);    }}, 2000);
const intervalData = setInterval(() => {
   eventEmitter.emit('dataReceived', { temperature: 25, humidity: 60
});
   dataCount++;
   if (dataCount >= maxData) {
      clearInterval(intervalData);
      console.log('Program stopped after emitting required number of
events.');
   }
}, 3000);
console.log('Listening for events...');
```

```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_16_A.JS
Listening for events...
Message Received: Hello World!
Data Received: { temperature: 25, humidity: 60 }
Message Received: Hello World!
Data Received: { temperature: 25, humidity: 60 }
Program stopped after emitting required number of events.
Message Received: Hello World!
PS C:\Node JS\Practical Slips>
```

B) Create A Node.Js File That Select All Records From The "Employee" Table, And Update The Salary Of The Given Eno…

```
const mysql = require('mysql');
const connection = mysql.createConnection({
  host: 'localhost',
  user: 'LALIT PATIL',
  password: 'l_patil__',
  database: 'employee'
});
connection.connect((err) => {
  if (err) {
    console.error('Error connecting to MySQL database: ' + err.stack);
    return;
  }
  console.log('Connected to MySQL database as id ' +
connection.threadId);
});
const updateSalary = (eno, newSalary) => {
  const sql = 'UPDATE Employee SET Esal = ? WHERE ENO = ?';
  connection.query(sql, [newSalary, eno], (err, result) => {
    if (err) {
      console.error('Error updating salary: ' + err.message);
      return;
    }
    console.log('Salary updated successfully.');
  });
};
const selectAllEmployees = () => {
  const sql = 'SELECT * FROM Employee';
  connection.query(sql, (err, rows) => {
    if (err) {
      console.error('Error selecting records: ' + err.message);
      return;
    }
    console.log('Employee records:');
    console.table(rows);
  });
};
```

```
const enoToUpdate = 1;
const newSalary = 50000;
updateSalary(enoToUpdate, newSalary);
selectAllEmployees();
connection.end((err) => {
  if (err) {
    console.error('Error closing MySQL connection: ' + err.message);
    return;
  }
  console.log('MySQL connection closed.');
});
```

```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_16_B.JS
Connected to MySQL database as id 200
Salary updated successfully.
Employee records:
```

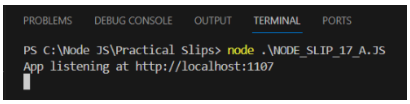
(index)	ENO	EName	Esal	EDept
0	1	'AAROHI PATIL'	50000	'HR'
1	2	'LALIT PATIL'	7800	'SE'

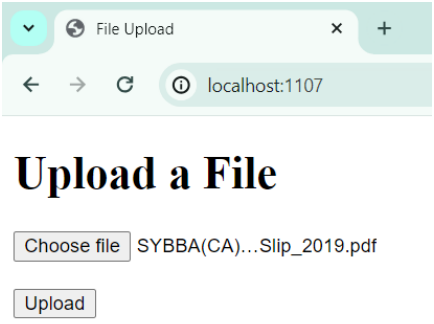
```
MySQL connection closed.
PS C:\Node JS\Practical Slips>
```

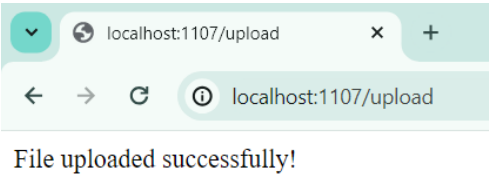
Slip 17

A) Write Node.Js Application That Transfer A File As An Attachment On Web And Enables Browser To Prompt The User To Download File Using Express Js..

```
const express = require('express');
const multer = require('multer');
const path = require('path');
const fs = require('fs');
const app = express();
const port = 1107;
const storage = multer.diskStorage({
  destination: function (req, file, cb) {
    cb(null, 'uploads/')  },
  filename: function (req, file, cb) {
    cb(null, file.originalname)  }});
const upload = multer({ storage: storage });
app.get('/', (req, res) => {
  res.sendFile(path.join(__dirname, 'NODE_SLIP_17_A.HTML'));});
app.post('/upload', upload.single('file'), (req, res) => {
  res.send('File uploaded successfully!');});
app.get('/download', (req, res) => {
  const file = path.join(__dirname, 'uploads', req.query.filename);
  fs.access(file, fs.constants.F_OK, (err) => {
    if (err) {
      res.status(404).send('File not found!');
      return;    }
    res.download(file, (err) => {
      if (err) {
        res.status(500).send('Error downloading file!');    }    });  });});});
app.listen(port, () => {
  console.log(`App listening at http://localhost:${port}`);
});
```







B) Using Node.js Display The Employee Details Order by Salary In Table Format.

```
const Table = require('cli-table');
const employees = [
    { name: 'LALIT PATIL', salary: 800000 },
    { name: 'SAGAR PATIL', salary: 810000 },
];
employees.sort((a, b) => a.salary - b.salary);
const table = new Table({
    head: ['Name',  'Salary'],
    colWidths: [20, 20]
});
employees.forEach(employee => {
    table.push([employee.name,
employee.salary]);
});
console.log(table.toString());
```

```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_17_B.JS

| Name        | Salary |
|---|---|
| LALIT PATIL | 800000 |
| SAGAR PATIL | 810000 |

PS C:\Node JS\Practical Slips>
```

Slip 18

A) Create Node.js Application To Bind Custom 'Receive_Data' With 'Data_Receive_Handler Function'..

```
const EventEmitter = require('events');
class CustomEmitter extends EventEmitter {}
const customEmitter=new CustomEmitter();
const Data_Receive_Handler = data => {
   console.log('Data Received:',data);
};
customEmitter.on ('Receive_Data',Data_Receive_Handler);
customEmitter.emit('Receive_Data',{ message: 'LALIT PATIL'});
```

```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_18_A.JS
Data Received: { message: 'LALIT PATIL' }
PS C:\Node JS\Practical Slips>
```

B) Using Node.Js Create Application That Contains Voters Details And Check Proper Validation For (Name, Age, Nationlity), As Name Should Be In Upper Case Letter Only, Age Should Not Be Less Than 18 yrs And Nationality Should Be Indian And Store Be Indian And Store The Data In Database…

```
const express = require('express');
const mysql = require('mysql');
const bodyParser = require('body-parser');
const app = express();
const db = mysql.createConnection({
    host: 'localhost',
    user: 'LALIT PATIL',
    password: 'l_patil__',
    database: 'voter_details',
    port: 3306
});
db.connect((err) => {
    if (err) {
        throw err;
    }
    console.log('MySQL Connected');
});
app.use(bodyParser.urlencoded({ extended: false }));
app.get('/', (req, res) => {
    res.sendFile(__dirname + '/NODE_SLIP_18_B.HTML');
});
app.post('/add-voter', (req, res) => {
    const { id, name, age, nationality } = req.body;
    if (!id || !name || !age || !nationality) {
        return res.status(400).send('All fields are required');
    }
    if (name !== name.toUpperCase()) {
        return res.status(400).send('Name should be in uppercase
letters');
    }
    if (age < 18) {
        return res.status(400).send('Age should not be less than 18
years');
```

```
    }
    if (nationality.toLowerCase() !== 'indian') {
        return res.status(400).send('Nationality should be Indian');
    }
    const sql = 'INSERT INTO voter (id, name, age, nationality)
VALUES (?, ?, ?, ?)';
    db.query(sql, [id, name, age, nationality], (err, result) => {
        if (err) {
            console.error('MySQL Error:', err);
            return res.status(500).send('Database error');
        }
        return res.status(201).send('Voter added successfully');
    });
});
const PORT = process.env.PORT || 1107;
app.listen(PORT, () => {
    console.log(`Server is running on port ${PORT}`);
});
```

NODE_SLIP_18_B.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Voter Details</title>
</head>
<body>
    <h2>Enter Voter Details</h2>
    <form action="/add-voter" method="POST">
        <label for="id">ID:</label><br>
        <input type="text" id="id" name="id"><br>
        <label for="name">Name:</label><br>
        <input type="text" id="name" name="name"><br>
        <label for="age">Age:</label><br>
        <input type="number" id="age" name="age"><br>
        <label for="nationality">Nationality:</label><br>
```

```
        <input type="text" id="nationality" name="nationality"><br>
        <input type="submit" value="Submit">
    </form>
</body>
</html>
```

```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_18_B.JS
Server is running on port 1107
MySQL Connected
```

Voter Details

localhost:1107

Enter Voter Details

ID:
106
Name:
LALIT D PATIL
Age:
19
Nationality:
INDIAN
Submit

localhost:1107/add-voter

localhost:1107/add-voter

Voter added successfully

voter_details - Schema voter - Table voter

Limit to 1000 rows

```
1 • SELECT * FROM voter_details.voter;
```

Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content:

id	name	age	nationality
106	LALIT D PATIL	19	INDIAN
NULL	NULL	NULL	NULL

Slip 19

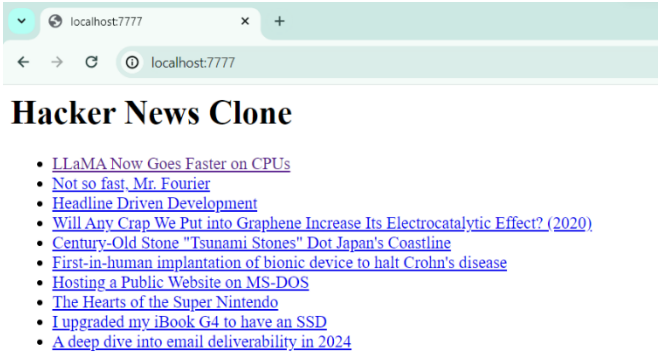
A) Write Node.Js Application That Shows Some Information When Your Node.Js Application Gets Started, Warning When It Tries To Open An Existing File And Error Message While That Specified File Is Not Found…

```
const fs = require('fs');
function displayStartupInfo() {
  console.log("Node.js application started.");
}
function handleExistingFile(filePath) {
  console.warn(`Warning: ${filePath} already exists. Opening the
file...`);
}
function handleFileNotFoundError(filePath) {
  console.error(`Error: ${filePath} not found.`);
}
function main() {
  const filePath = 'NODE_SLIP_15_A.TXT';
  displayStartupInfo();
  fs.access(filePath, fs.constants.F_OK, (err) => {
    if (err) {
      handleFileNotFoundError(filePath);
    } else {
      handleExistingFile(filePath);
    }
  });
}
main();
```

```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_19_A.JS
Node.js application started.
Warning: NODE_SLIP_15_A.TXT already exists. Opening the file...
PS C:\Node JS\Practical Slips>
```

B) Using Node.Js Clone The "Hacker News" Website…

```
const express = require('express');
const axios = require('axios');
const app = express();
async function fetchHackerNews() {
   try {
      const response = await axios.get('https://hacker-
news.firebaseio.com/v0/topstories.json');
      const topStoryIds = response.data.slice(0, 10); // Get top 10
stories
      const stories = await Promise.all(topStoryIds.map(id =>
         axios.get(`https://hacker-
news.firebaseio.com/v0/item/${id}.json`) ));
      return stories.map(story => story.data);
   } catch (error) {
      console.error('Error fetching Hacker News data:', error);
      return []; }}
app.get('/', async (req, res) => {
   const stories = await fetchHackerNews();
   res.send(`
      <h1>Hacker News Clone</h1>
      <ul>
         ${stories.map(story => `<li><a
href="${story.url}">${story.title}</a></li>`).join('')}
      </ul>
   `);});
const PORT = process.env.PORT || 7777;
app.listen(PORT, () => {
   console.log(`Server is running on port ${PORT}`);
});
```



Slip 20

A) Write Node.Js Application Containing An Event Handler To Open And Read The Contents Of A File…

```
const fs = require('fs');
const filePath = 'NODE_SLIP_20_A.TXT';
const readStream = fs.createReadStream(filePath, { encoding: 'utf8'
});
readStream.on('data', (chunk) => {
   console.log('Chunk received:');
   console.log(chunk);
});
readStream.on('end', () => {
   console.log('End of file reached');
});
readStream.on('error', (err) => {
   console.error('Error occurred:', err);
});
```

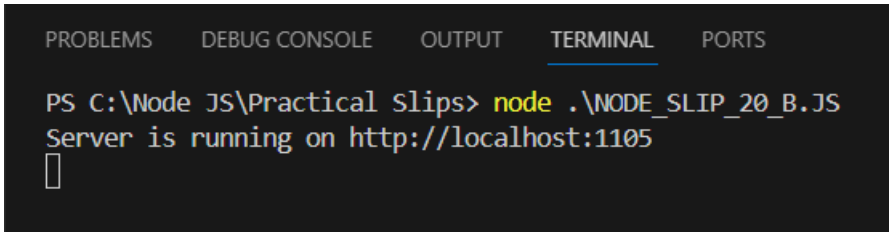
NODE_SLIP_20_A.TXT

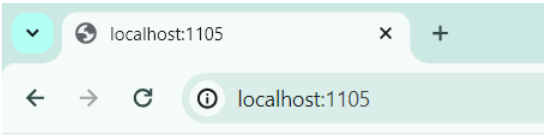
Sinhgad College of Arts and Commerce popularly known as SCOAC, has always strived for excellence in academics and complementing co-curricular learning that meets student expectations.

```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_20_A.JS
Chunk received:
Sinhgad College of Arts and Commerce popularly known as SCOAC,
has always strived for excellence in academics and complementing
co-curricular learning that meets student expectations.

End of file reached
PS C:\Node JS\Practical Slips>
```

B) Using Node.Js Create An Application That Shows SY BBA(CA) Course Structure.

```
const express = require('express');
const app = express();
const port = 1105;
const courseStructure = [
   { semester: 'Semester 3', subjects: ['301-Digital Marketing', '302-Data Structure', '303-Software Engineering ', '304-Angular JS', '305-Big Data'] },
   { semester: 'Semester 4', subjects: ['401-Networking',  '402-Object Oriented Concepts Through CPP ','403-Opeating System','405-Project']},
];
app.get('/', (req, res) => {
   res.send('Welcome to SY BBA(CA) Course Structure!');
});
app.get('/course-structure', (req, res) => {
   res.json(courseStructure);
});
app.listen(port, () => {
   console.log(`Server is running on http://localhost:${port}`);
});
```





Welcome to SY BBA(CA) Course Structure!

Slip 21

A) Write Node.Js Application To Create An Eventemitter Which Will Emit An Event That Contains Information About The Application's Uptime, Every Second…

```
const EventEmitter = require('events');
class UptimeEmitter extends EventEmitter {}
const uptimeEmitter = new UptimeEmitter();
setInterval(() => {
   const uptime = process.uptime();
   uptimeEmitter.emit('uptime', { uptime });
}, 1000);
uptimeEmitter.on('uptime', ({ uptime }) => {
   console.log(`Application uptime: ${uptime} seconds`);
});
uptimeEmitter.on('uptime', ({ uptime }) => {
   if (uptime % 10 === 0) {
      console.log(`Checkpoint: Application uptime reached ${uptime}
seconds`);
   }
});
```

```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_21_A.JS
Application uptime: 1.0417303 seconds
Application uptime: 2.0569201 seconds
Application uptime: 3.0671683 seconds
Application uptime: 4.0777441 seconds
Application uptime: 5.0910371 seconds
Application uptime: 6.1027368 seconds
Application uptime: 7.1178714 seconds
PS C:\Node JS\Practical Slips>
```

B) Using Node.Js Create An Application That Shows The Mini Statement Of Particular Account. (Customer Table)…

```
const express = require('express');
const mysql = require('mysql');
const app = express();
const port = 1105;
const connection = mysql.createConnection({
  host: 'localhost',
  user: 'LALIT PATIL',
  password: 'l_patil__',
  database: 'Bank'
});
connection.connect(err => {
  if (err) {
    console.error('Error connecting to database: ' + err.stack);
    return;
  }
  console.log('Connected to database as ID ' + connection.threadId);
});
app.get('/', (req, res) => {
  res.send('Welcome to the Mini Statement App');
});
app.get('/mini-statement/:accountNumber', (req, res) => {
  const accountNumber = req.params.accountNumber;
  const query = `SELECT * FROM customer WHERE
account_number = ${accountNumber}`;
  connection.query(query, (err, results) => {
    if (err) {
      console.error('Error retrieving mini statement: ' + err.stack);
      res.status(500).send('Error retrieving mini statement');
      return;
    }
    if (results.length === 0) {
      res.status(404).send('Account not found');
      return;
    }
    res.json(results);
  });
```

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

```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_21_B.JS
Server is listening at http://localhost:1105
Connected to database as ID 90
```

localhost:1105

localhost:1105

Welcome to the Mini Statement App

localhost:1105/mini-statement/

localhost:1105/mini-statement/0512

Pretty print ☑


```
[
  {
    "id": 101,
    "account_number": "0512",
    "transaction_type": "Withdrawal",
    "amount": 5000,
    "transaction_date": "2024-04-01T18:30:00.000Z"
  },
  {
    "id": 103,
    "account_number": "0512",
    "transaction_type": "Deposit",
    "amount": 17000,
    "transaction_date": "2024-03-24T18:30:00.000Z"
  },
  {
    "id": 105,
    "account_number": "0512",
    "transaction_type": "Withdrawal",
    "amount": 25000,
    "transaction_date": "2024-03-02T18:30:00.000Z"
  }
]
```

Slip 22

A) Create A Node.Js Application To Count Number Of Lines In A File And Display The Count On Console.

```
const fs = require('fs');
function countLines(filePath) {
   return new Promise((resolve, reject) => {
      let lineCount = 0;
      const stream = fs.createReadStream(filePath, { encoding: 'utf8'
});
      stream.on('data', (chunk) => {
         lineCount += chunk.split('\n').length - 1;
      });
      stream.on('end', () => {
         resolve(lineCount);
      });
      stream.on('error', (err) => {
         reject(err);
      });
   });
}
const filePath = 'NODE_SLIP_22_A.TXT';
countLines(filePath)
   .then((count) => {
      console.log(`Number of lines in the file: ${count}`);
   })
   .catch((err) => {
      console.error('Error:', err);
   });
```

NODE_SLIP_22_A.TXT

Sinhgad College of Arts and Commerce popularly known as SCOAC, has always strived for excellence in academics and complementing co-curricular learning that meets student expectations.All our efforts are driven to make student life on college campus an enriching experience.Our journey for the last decade has been extremely successful and satisfying in terms of accomplishments and accolades in scholastic,co-scholastic and infrastructural development areas..In our endeavor, we draw upon reserves of goodwill among the diasporas and whole hearted commitment of our well trained,qualified, and experienced faculty and staff who contribute to holistic learning that grooms' young minds,readies them to respond confidently to the challenges and newer demands of a knowledgeable society.We have set ourselves for adopting newer pedagogy that blends technology to facilitate easy interface for exchange of information..We believe that Education does not only encourage personal development, it also offers the general growth of an entire community providing a place for people to interact, socialize, and unify their societies.

```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_22_A.JS
Number of lines in the file: 12
PS C:\Node JS\Practical Slips>
```

B) Using Node.Js Create An Application To Perform The Following Operations On Customer Account(Minimum Balance Should Be Maintained 1000 And Use Customer Table)

1. Withdraw The Amount

2. Balance Enquiry

```
const mysql = require('mysql');
const readline = require('readline');
const connection = mysql.createConnection({
  host: 'localhost',
  user: 'LALIT PATIL',
  password: 'l_patil__',
  database: 'Customer'
});
const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout
});
function withdrawAmount(accountId, amount) {
  connection.query('SELECT balance FROM Customers WHERE id
= ?', [accountId], (error, results) => {
    if (error) {
      console.error('Error occurred:', error);
      return;
    }
    const currentBalance = results[0].balance;
    if (currentBalance - amount < 1000) {
      console.log('Error: Insufficient balance');
    } else {
      const newBalance = currentBalance - amount;
      connection.query('UPDATE Customers SET balance = ?
WHERE id = ?', [newBalance, accountId], (updateError,
updateResults) => {
        if (updateError) {
          console.error('Error occurred:', updateError);
        } else {
          console.log('Amount withdrawn successfully');
          console.log('Updated balance:', newBalance);
        }}); }});}
```

```
function checkBalance(accountId) {
  connection.query('SELECT balance FROM Customers WHERE id
= ?', [accountId], (error, results) => {
    if (error) {
      console.error('Error occurred:', error);
      return; }
    const currentBalance = results[0].balance;
    console.log('Current balance:', currentBalance);
  });}
function main() {
  rl.question('Enter customer ID: ', (id) => {
    rl.question('Choose operation (1. Withdraw | 2. Balance Enquiry):
', (operation) => {
      if (operation === '1') {
        rl.question('Enter amount to withdraw: ', (amount) => {
          withdrawAmount(id, parseFloat(amount));
          rl.close();
        });
      } else if (operation === '2') {
        checkBalance(id);
        rl.close();
      } else {
        console.log('Invalid operation');
        rl.close(); } });  });}
connection.connect((err) => {
  if (err) {
    console.error('Error connecting to MySQL:', err);
    return;
  }
  console.log('Connected to MySQL');
  main();});
```

```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_22_B.JS
Connected to MySQL
Enter customer ID: 105
Choose operation (1. Withdraw | 2. Balance Enquiry): 2
Current balance: 60000
PS C:\Node JS\Practical Slips> node .\NODE_SLIP_22_B.JS
Connected to MySQL
Enter customer ID: 105
Choose operation (1. Withdraw | 2. Balance Enquiry): 1
Enter amount to withdraw: 45000
Amount withdrawn successfully
Updated balance: 15000
```

Slip 23

A) Create A Node.Js Application To Search A Particular Word In A File And Replace All Occurrences Of That Word With Another Word And Bold All.

```
const fs = require('fs');
function replaceWord(filePath, searchWord, replaceWord) {
   fs.readFile(filePath, 'utf8', (err, data) => {
      if (err) {
         console.error("Error reading file:", err);
         return;
      }
      const replacedData = data.replace(new RegExp(searchWord, 'gi'), replaceWord);
      fs.writeFile(filePath, replacedData, 'utf8', (err) => {
         if (err) {
            console.error("Error writing to file:", err);
            return;
         }
         console.log(`Successfully replaced '${searchWord}' with '${replaceWord}' in file: ${filePath}`);
      });
   });
}
const filePath = 'NODE_SLIP_23_A.TXT';
const searchWord = 'Sinhgad';
const replacementWord = 'SCOAC';
replaceWord(filePath, searchWord, replacementWord);
```

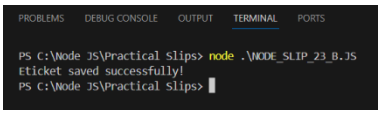
NODE_SLIP_23_A.TXT

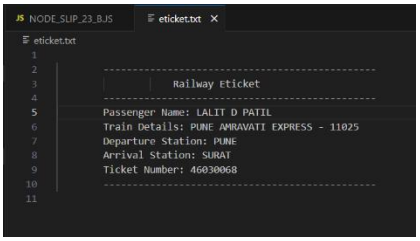
SCOAC College of Arts and Commerce popularly known as SCOAC, has always strived for excellence in academics and complementing co-curricular learning that meets student expectations.

```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_23_A.JS
Successfully replaced 'Sinhgad' with 'SCOAC' in file: NODE_SLIP_23_A.TXT
PS C:\Node JS\Practical Slips>
```

B) Using Node.Js Create An Application That Generates The "Etickect" Of Railway.

```
const fs = require('fs');
const path = require('path');
functiongenerateEticket(passengerName,trainDetails,
departureStation, arrivalStation, ticketNumber) {
   const eticket = `
      ---------------------------------------------
            Railway Eticket
      ---------------------------------------------
      Passenger Name: ${passengerName}
      Train Details: ${trainDetails}
      Departure Station: ${departureStation}
      Arrival Station: ${arrivalStation}
      Ticket Number: ${ticketNumber}
      ---------------------------------------------
   `; return eticket;}function saveEticketToFile(eticket) {
   const filePath = path.join(__dirname, 'eticket.txt');
   fs.writeFile(filePath, eticket, (err) => {
      if (err) {
         console.error('Error saving Eticket:', err);
      } else {
         console.log('Eticket saved successfully!'); }   });}
const passengerName = 'LALIT D PATIL';
const trainDetails = 'PUNE AMRAVATI EXPRESS - 11025';
const departureStation = 'PUNE';
const arrivalStation = 'SURAT';
const ticketNumber = '46030068';
const eticket = generateEticket(passengerName, trainDetails,
departureStation, arrivalStation, ticketNumber);
saveEticketToFile(eticket);
```





Slip 24

A) Create A Node.Js Application That Raise And Bind An Event By Returning Eventemitter Object From A Function.

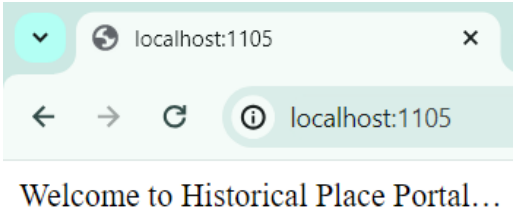
```
const EventEmitter = require('events');
function createEventEmitter() {
  const emitter = new EventEmitter();
  function raiseEvent(eventName, delay) {
    setTimeout(() => {
      emitter.emit(eventName, 'Event raised after ' + delay + '
milliseconds');
    }, delay);
  }
  return {
    emitter,
    raiseEvent
  };
}
const eventHandler = createEventEmitter();
eventHandler.emitter.on('customEvent', (data) => {
  console.log('Event Received:', data);
});
eventHandler.raiseEvent('customEvent', 2000);
```

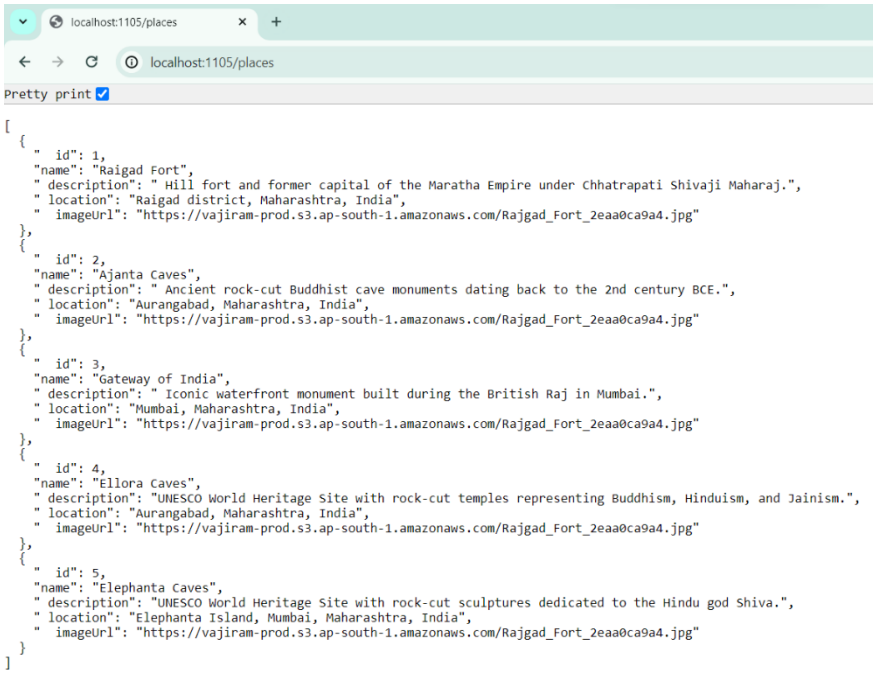
```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_24_A.JS
Event Received: Event raised after 2000 milliseconds
PS C:\Node JS\Practical Slips>
```

B) Using Node.Js Create A Historical Place Portal…

```
const express = require('express');
const bodyParser = require('body-parser');
const mysql = require('mysql');
const app = express();
const port = 1105;
const connection = mysql.createConnection({
   host: 'localhost',
   user: 'LALIT PATIL',
   password: 'l_patil__',
   database: 'historical_places'
});
connection.connect((err) => {
   if (err) {
      console.error('Error connecting to MySQL: ' + err.stack);
      return;
   }
   console.log('Connected to MySQL as id ' + connection.threadId);
});
app.use(bodyParser.urlencoded({ extended: true }));
app.use(bodyParser.json());
app.get('/', (req, res) => {
   res.send('Welcome to  Historical Place Portal…');
  });
app.get('/places', (req, res) => {
   connection.query('SELECT * FROM places', (err, results) => {
      if (err) {
         console.error('Error fetching places: ' + err.stack);
         res.status(500).json({ message: err.message });
         return;
      }
      res.json(results);
   });
});
app.post('/places', (req, res) => {
   const { name, description, location, imageUrl } = req.body;
   connection.query('INSERT INTO places (name, description,
location, imageUrl) VALUES (?, ?, ?, ?)',
```

```
        [name, description, location, imageUrl],
        (err, result) => {
            if (err) {
                console.error('Error creating place: ' + err.stack);
                res.status(400).json({ message: err.message });
                return;
            }
            res.status(201).json({ id: result.insertId, name, description,
location, imageUrl });
        });
});
app.listen(port, () => {
    console.log(`Server is running on port ${port}`);
});
```





Slip 25

A) Create A Node.Js Application That Wait Until The Promise Returns The Result Using Wait Function…

```
function fetchData() {
   return new Promise((resolve, reject) => {
     setTimeout(() => {
       resolve("Data fetched successfully");
     }, 2000);
   });
 }
 async function waitUntilPromise() {
   console.log("Waiting for promise to resolve...");
   const result = await fetchData();
   console.log("Promise resolved:", result);
 }
 waitUntilPromise()
   .then(() => {
     console.log("Application finished executing");
   })
   .catch((error) => {
     console.error("Error occurred:", error);
   });
```

```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_25_A.JS
Waiting for promise to resolve...
Promise resolved: Data fetched successfully
Application finished executing
PS C:\Node JS\Practical Slips>
```

B) Using Node.Js Create An Application That Shows The Events Of The Day.

```
const moment = require('moment');
const events = [
    { title: "Lecture time", date: "2024-04-02T09:00:00" },
    { title: "Lunch Break", date: "2024-04-02T12:00:00" },
    { title: "Going Home", date: "2024-04-02T14:00:00" }
];
function getEventsOfTheDay(events) {
    const today = moment().startOf('day');
    const eventsOfTheDay = events.filter(event => {
        const eventDate = moment(event.date);
        return eventDate.isSame(today, 'day');
    });
    return eventsOfTheDay;
}
function displayEvents(events) {
    if (events.length === 0) {
        console.log("No events for today.");
    } else {
        console.log("Events for today:");
        events.forEach(event => {
            console.log(`-${event.title}
(${moment(event.date).format('h:mm A')})`);
        });
    }
}
const eventsOfTheDay = getEventsOfTheDay(events);
displayEvents(eventsOfTheDay);
```

```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_25_B.JS
Events for today:
- Lecture time (9:00 AM)
- Lunch Break (12:00 PM)
- Going Home (2:00 PM)
PS C:\Node JS\Practical Slips>
```

Slip 26

A) Create A Node.Js Application That Raise And Bind A Event Using Extending The Event Emitter Class…

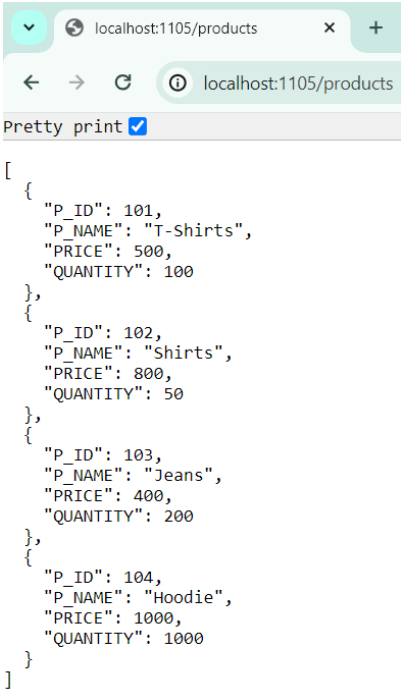
```
const EventEmitter = require('events');
class MyEmitter extends EventEmitter {}
const myEmitter = new MyEmitter();
myEmitter.on('event', () => {
  console.log('Event occurred!');
});
myEmitter.on('anotherEvent', (data) => {
  console.log('Another event occurred with data:', data);
});
myEmitter.emit('event');
myEmitter.emit('anotherEvent', { message: 'Hello, World!' });
```

```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_26_A.JS
Event occurred!
Another event occurred with data: { message: 'Hello, World!' }
PS C:\Node JS\Practical Slips>
```

B) Using Node.Js Create A Department Store Portal…

```
const express = require('express');
const bodyParser = require('body-parser');
const mysql = require('mysql');
const app = express();
app.use(bodyParser.json());
const connection = mysql.createConnection({
  host: 'localhost',
  user: 'LALIT PATIL',
  password: 'l_patil__',
  database: 'department_store'
});
connection.connect((err) => {
  if (err) {
    console.error('Error connecting to MySQL database: ', err);
    return;
  }
  console.log('Connected to MySQL database');
});
app.get('/', (req, res) => {
    res.send('Welcome To The Department Store Portal!');
  });
app.get('/products', (req, res) => {
  const sql = 'SELECT * FROM products';
  connection.query(sql, (err, results) => {
    if (err) {
      console.error('Error executing MySQL query: ', err);
      res.status(500).json({ message: 'Internal server error' });
      return;
    }
    res.json(results);
  });
});
app.post('/products', (req, res) => {
  const { name, price, description } = req.body;
  const sql = 'INSERT INTO products (name, price, description)
VALUES (?, ?, ?)';
  connection.query(sql, [name, price, description], (err, result) => {
```

```
    if (err) {
      console.error('Error executing MySQL query: ', err);
      res.status(400).json({ message: 'Failed to add product' });
      return;
    }
    res.status(201).json({ message: 'Product added successfully' });
  });
});
const PORT = process.env.PORT || 1105;
app.listen(PORT, () => {
  console.log(`Server is listening on port ${PORT}`);
});
```



Slip 27

A) Create A Node.Js Application Create A Directory And The Contents Of The Directory.

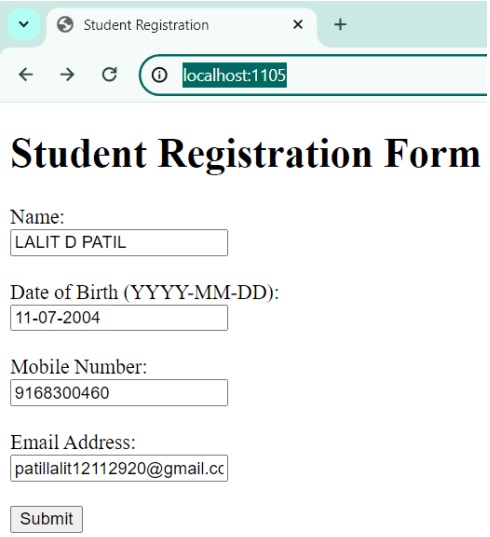
```
const fs = require('fs');
const path = require('path');
function createDirectory(directoryPath, contents) {
    fs.mkdir(directoryPath, (err) => {
        if (err) {
            console.error('Error creating directory:', err);
            return;
        }
        console.log(`Directory created successfully: ${directoryPath}`);
        contents.forEach((file) => {
            const filePath = path.join(directoryPath, file.name);
            fs.writeFile(filePath, file.content, (err) => {
                if (err) {
                    console.error(`Error creating file ${file.name}:`, err);
                    return;
                }
                console.log(`File created successfully: ${filePath}`);
            });
        });
    });
}
const directoryPath = 'Student_directory';
const directoryContents = [
    { name: 'R .txt', content: ' file 1.' },
    { name: 'H .txt', content: ' file 2.' },
    { name: 'T .txt', content: ' file 3.' },
    { name: 'D .txt', content: ' file 4.' },
    { name: 'M .txt', content: ' file 5.' },
];
createDirectory(directoryPath, directoryContents);
```

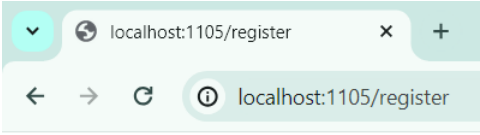
```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_27_A.JS
Directory created successfully: Student_directory
File created successfully: Student_directory\H .txt
File created successfully: Student_directory\R .txt
File created successfully: Student_directory\T .txt
File created successfully: Student_directory\D .txt
File created successfully: Student_directory\M .txt
PS C:\Node JS\Practical Slips>
```

B) Create An Node.Js Application That Contain The Student Registration Details And Write A Javascript To Validate DOB, Mobile Number, And Email Address…

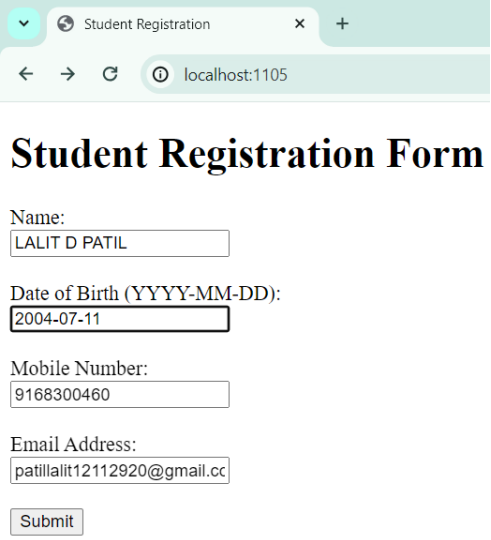
```
const express = require('express');
const bodyParser = require('body-parser');
const app = express();
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: true }));
app.use(express.static('public'));
app.get('/', (req, res) => {
  res.sendFile(__dirname + '/NODE_SLIP_27_B.HTML');
});
app.post('/register', (req, res) => {
  const { name, dob, mobile, email } = req.body;
  const dobRegex = /^\d{4}-\d{2}-\d{2}$/;
  const mobileRegex = /^\d{10}$/;
  const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
  if (!dobRegex.test(dob)) {
    return res.status(400).send('Invalid date of birth');
  }
  if (!mobileRegex.test(mobile)) {
    return res.status(400).send('Invalid mobile number');
  }
  if (!emailRegex.test(email)) {
    return res.status(400).send('Invalid email address');
  }
  res.send('Registration successful!');
});
const port = process.env.PORT || 1105;
app.listen(port, () => {
  console.log(`Server is running on port ${port}`);
});
```

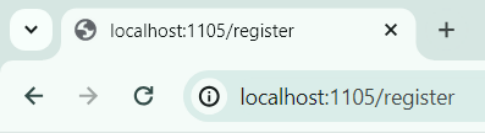
```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_27_B.JS
Server is running on port 1105
```





Invalid date of birth





Registration successful!

Slip 28

A) Create A Node.Js Application To Check Whether Given Name Is File Of Directory, If It File, Truncate The Content After 10 Bytes.

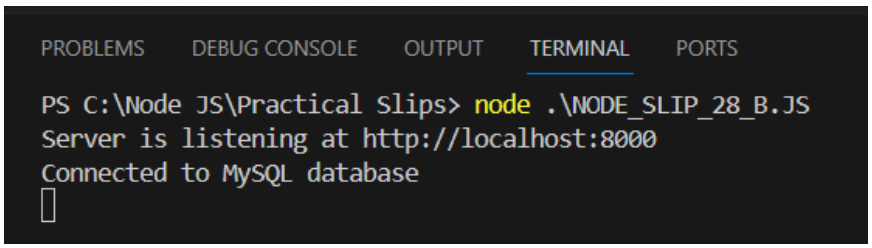
```
const fs = require('fs');
const path = require('path');
function isFile(filePath) {
   try {
      return fs.statSync(filePath).isFile();
   } catch (err) {
      return false;}}
function truncateFileContent(filePath) {
   fs.readFile(filePath, 'utf8', (err, data) => {
      if (err) {
         console.error('Error reading file:', err);
         return;   }
      const truncatedContent = data.slice(0, 10);
      fs.writeFile(filePath, truncatedContent, 'utf8', (err) => {
         if (err) {
            console.error('Error truncating file content:', err);
            return;    }
         console.log('File content truncated successfully.');});});}
function checkName(name) {
   const filePath = path.resolve(name);

   if (isFile(filePath)) {
      console.log(`${name} is a file.`);
      truncateFileContent(filePath);
   } else {
      console.log(`${name} is not a file.`);
   }
}
const fileName = 'NODE_SLIP_28_A.TXT';
```

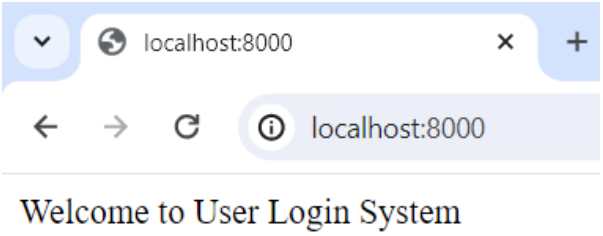
```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_28_A.JS
NODE_SLIP_28_A.TXT is a file.
File content truncated successfully.
PS C:\Node JS\Practical Slips>
```

B) Using Node.Js Create A User Login System With Forgot Password Option Which Can Set New Password…

```
const express = require('express');
const bodyParser = require('body-parser');
const mysql = require('mysql');
const app = express();
const port = 8000;
app.use(bodyParser.urlencoded({ extended: true }));
app.use(bodyParser.json());
const db = mysql.createConnection({
  host: 'localhost',
  user: 'LALIT PATIL',
  password: 'l_patil__',
  database: 'user_login_system'
});
db.connect(err => {
  if (err) throw err;
  console.log('Connected to MySQL database');
});
app.get('/', (req, res) => {
  res.send('Welcome to User Login System');
});
app.post('/login', (req, res) => {
  const { username, password } = req.body;
  db.query(
    'SELECT * FROM users WHERE username = ? AND password
= ?',
    [username, password],
    (err, results) => {
      if (err) throw err;
      if (results.length > 0) {
        res.send('Login successful');
      } else {
        res.status(401).send('Invalid username or password');
      }
    }
  );
});
```

```
app.post('/forgot-password', (req, res) => {
  const { email } = req.body;
  const newPassword = 'new_password';
  db.query(
    'UPDATE users SET password = ? WHERE email = ?',
    [newPassword, email],
    (err, results) => {
      if (err) throw err;
      res.send('Password updated successfully');
    }
  );
});
app.listen(port, () => {
  console.log(`Server is listening at http://localhost:${port}`);
});
```





Slip 29

A) Create Node Js Application Using User Defined Rectangle Module To Find Area Of Rectangle And Display The Details On Console.

```
exports.calculateArea = (length, width) => {
   return length * width;
};
exports.displayDetails = (length, width) => {
   console.log("Rectangle Details:");
   console.log("Length:", length);
   console.log("Width:", width);
   console.log("Area:", this.calculateArea(length, width));
};
```

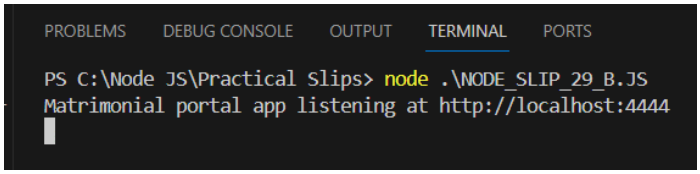
NODE_SLIP_29_AA.JS

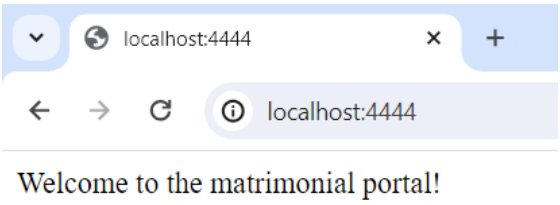
```
const rectangle = require('./NODE_SLIP_29_A.JS');
const length = 11;
const width = 5;

const area = rectangle.calculateArea(length, width);
rectangle.displayDetails(length, width);
```

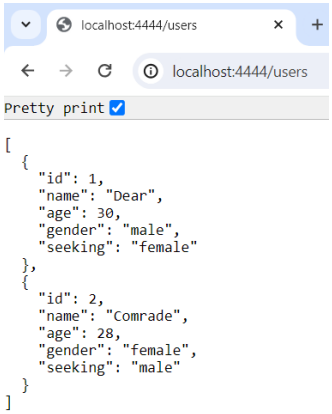
```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_29_AA.JS
Rectangle Details:
Length: 11
Width: 5
Area: 55
PS C:\Node JS\Practical Slips>
```

B) Using Node.Js Create A Matrimonial Portal.

```
const express = require('express');
const app = express();
const port = 4444;
const users = [
  { id: 1, name: 'Dear', age: 30, gender: 'male', seeking: 'female' },
  { id: 2, name: 'Comrade', age: 28, gender: 'female', seeking: 'male' },

];
app.get('/', (req, res) => {
  res.send('Welcome to the matrimonial portal!');
});
app.get('/users', (req, res) => {
  res.json(users);
});
app.get('/find-matches/:gender', (req, res) => {
  const gender = req.params.gender.toLowerCase();
  const matches = users.filter(user => user.seeking === gender);
  res.json(matches);
});
app.listen(port, () => {
  console.log(`Matrimonialportal app listening at
http://localhost:${port}`);
});
```







Slip 30

A) Create A Node.Js Application That Uses User Defined Module Circle.Js Which Exports The Functions Area. () And Circumference () And Display The Details On Console…

```
exports.area = (radius) => {
   return Math.PI * radius * radius;
};
exports.circumference = (radius) => {
   return 2 * Math.PI * radius;
};
```

NODE_SLIP_30_AA.JS

```
const circle = require('./NODE_SLIP_30_A.JS');
const radius = 11;
console.log(`Area of the circle with radius ${radius}: ${circle.area(radius)}`);
console.log(`Circumference of the circle with radius ${radius}: ${circle.circumference(radius)}`);
```

```
PROBLEMS    DEBUG CONSOLE    OUTPUT    TERMINAL    PORTS

PS C:\Node JS\Practical Slips> node .\NODE_SLIP_30_AA.JS
Area of the circle with radius 11: 380.1327110843649
Circumference of the circle with radius 11: 69.11503837897544
PS C:\Node JS\Practical Slips>
```

B) Create a Node.js file that demonstrate create database and Hospital table (hReg, hName, address, contact) in MySQL.

```
const mysql = require('mysql');
const connection = mysql.createConnection({
  host: 'localhost',
  user: 'LALIT PATIL',
  password: 'l_patil__',
  database:'Hospital',
  port:3306
});
connection.connect(function(err) {
  if (err) {
    console.error('Error connecting to MySQL: ' + err.stack);
    return;
  }
  console.log('Connected to MySQL as id ' + connection.threadId);
});
connection.query('CREATE DATABASE IF NOT EXISTS hospital_database', function(err, result) {
  if (err) throw err;
  console.log('Hospital database created');
});
connection.query(`CREATE TABLE IF NOT EXISTS Hospital (
  hReg INT PRIMARY KEY AUTO_INCREMENT,
  hName VARCHAR(255) NOT NULL,
  address VARCHAR(255) NOT NULL,
  contact VARCHAR(20) NOT NULL
)`, function(err, result) {
  if (err) throw err;
  console.log('Hospital table created');
});
connection.end();
```