Lesson 03 Demo 02

Using Operators and Expressions

Objective: To demonstrate practical usage of JavaScript operators and expressions for enhanced understanding and application in real-world scenarios

Tools required: Visual Studio Code and Node.js

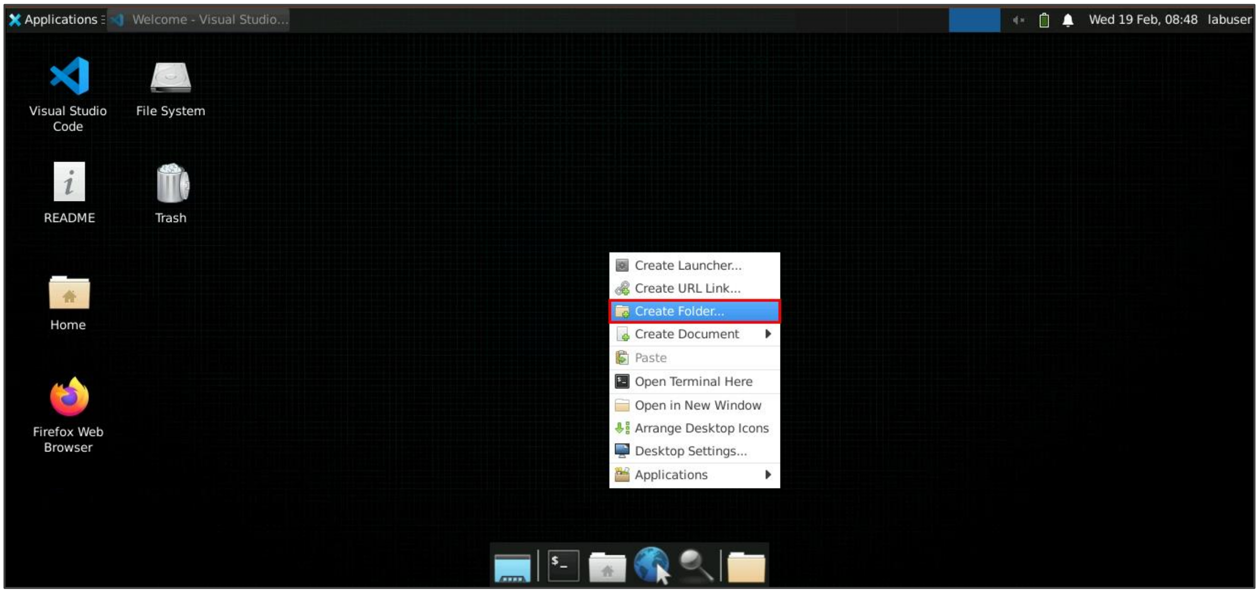
Prerequisites: A basic understanding of operators and expressions in JavaScript

Steps to be followed:

1. Create an **OperatorsAndExpressions.js** folder
2. Execute the JavaScript file

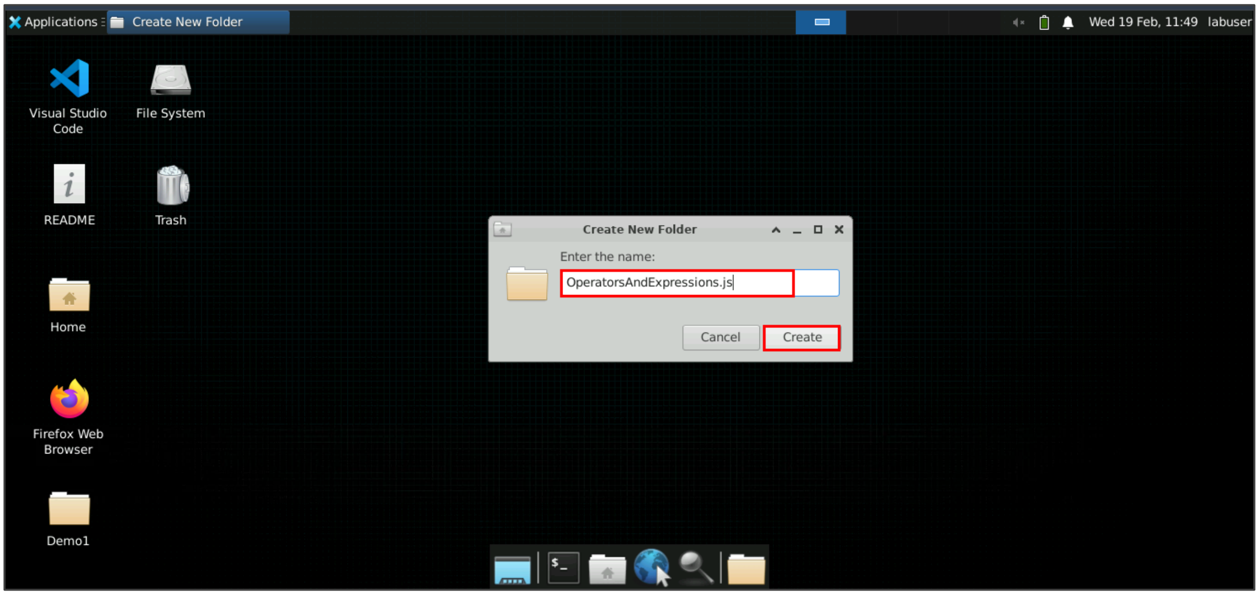
Step 1: Create an OperatorsAndExpressions.js folder

1.1 Right-click on the desktop and click on **Create Folder…**

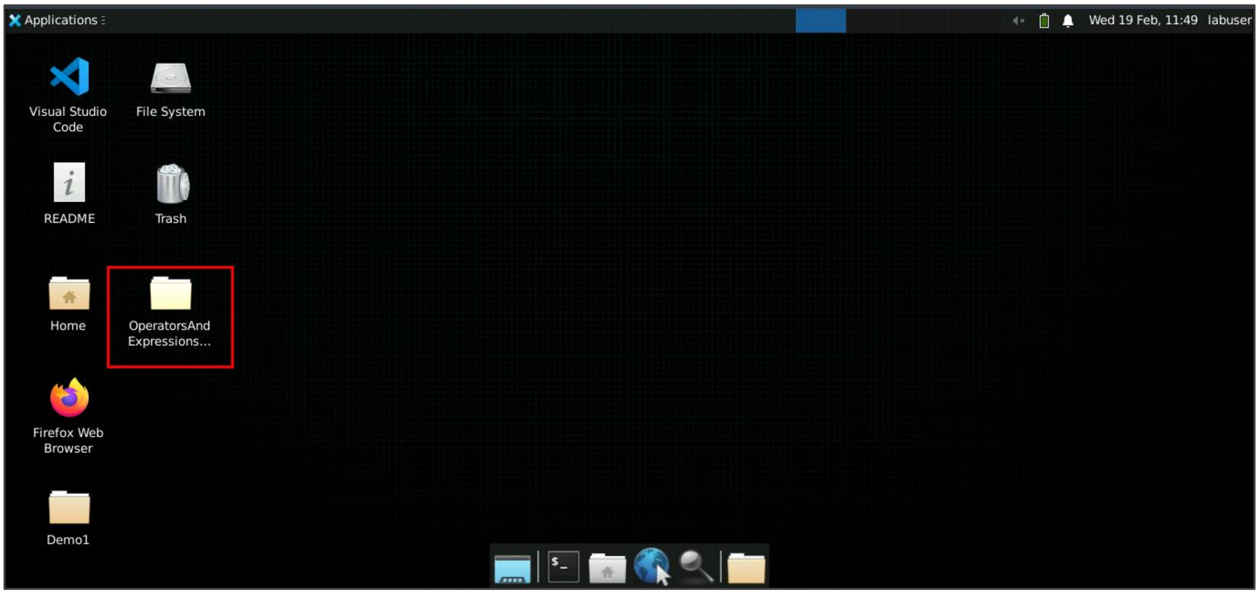


.

1.2 Enter the folder name as **OperatorsAndExpressions.js** and click on **Create**

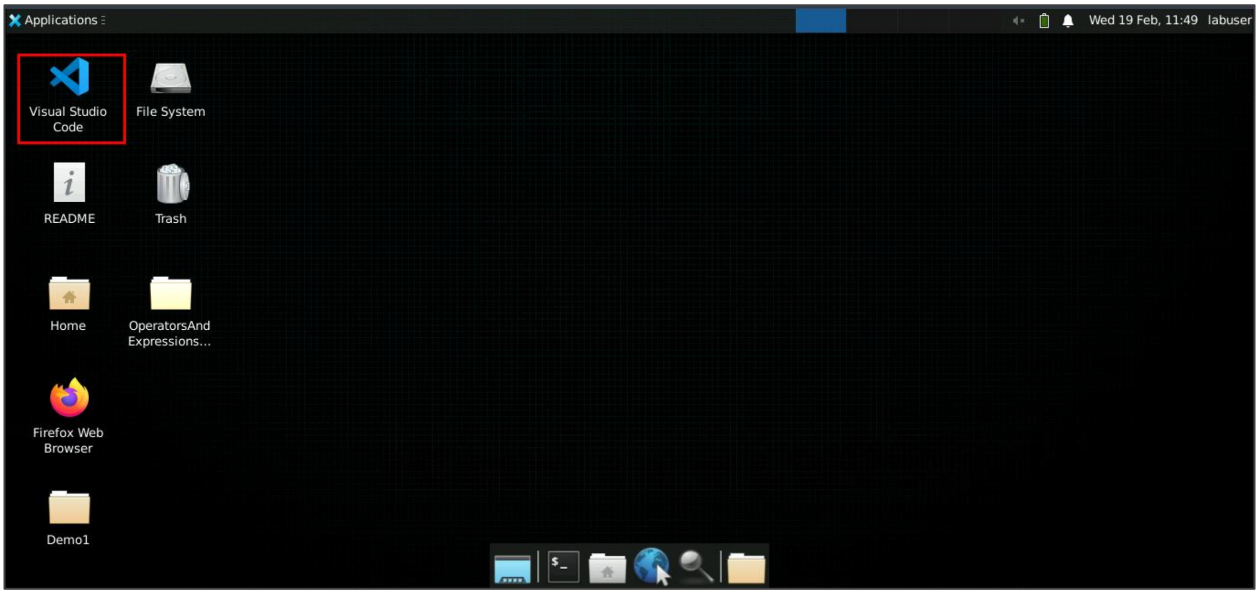


The **OperatorsAndExpressions.js** folder gets created as shown below:

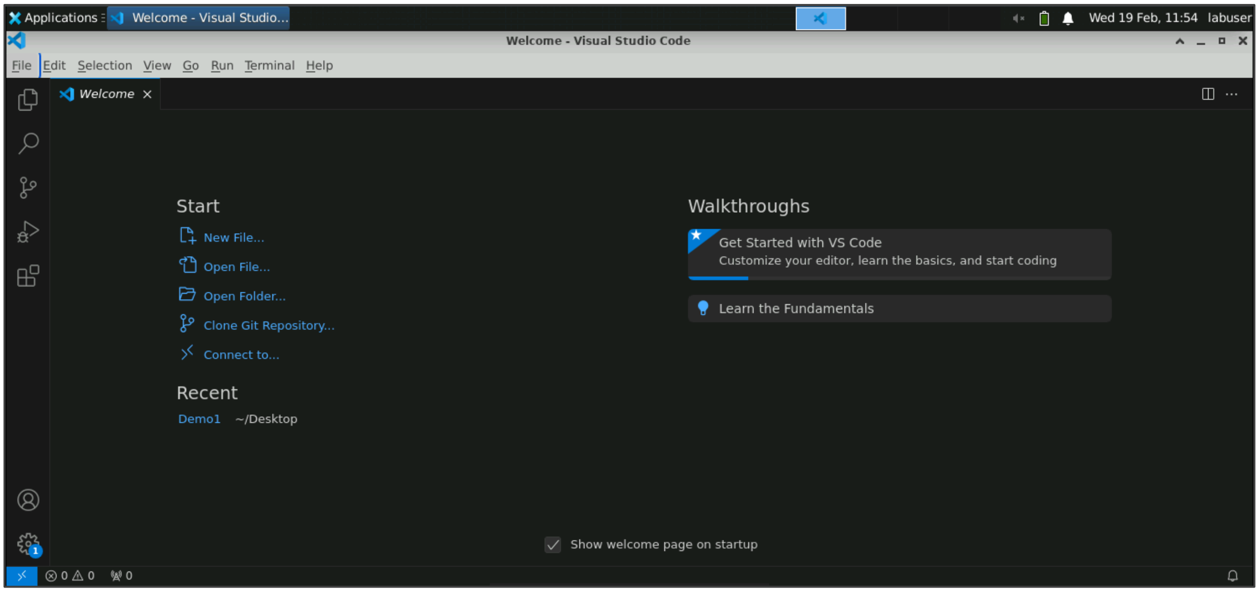


Step 2: Execute the JavaScript file

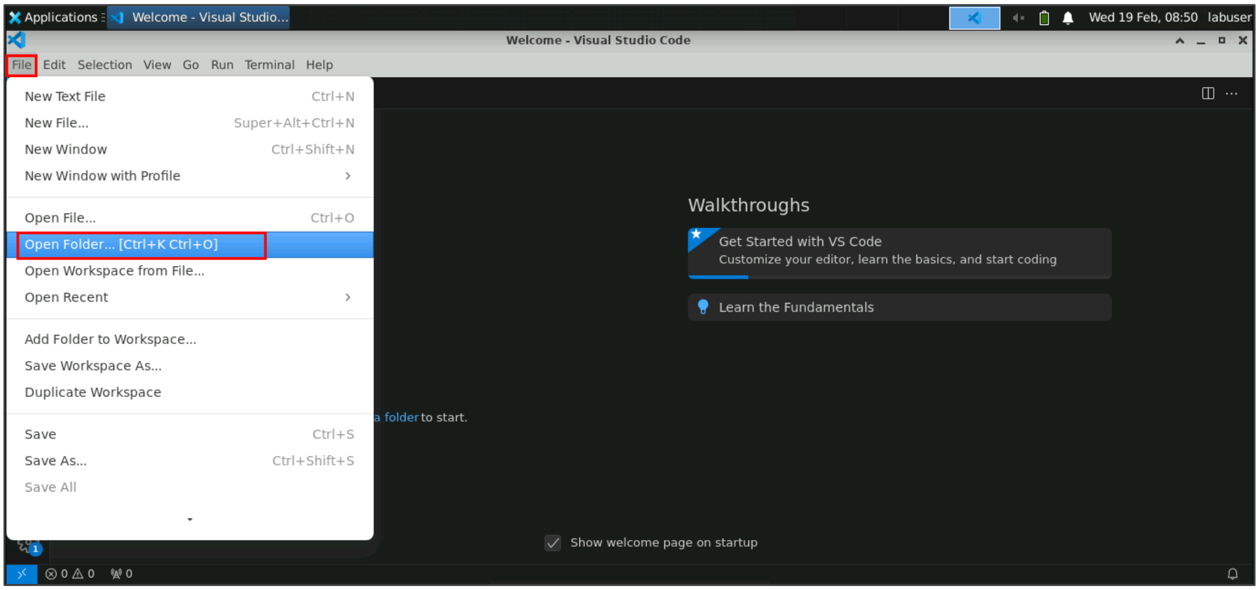
2.1 Double-click on the **Visual Studio Code** icon to open it



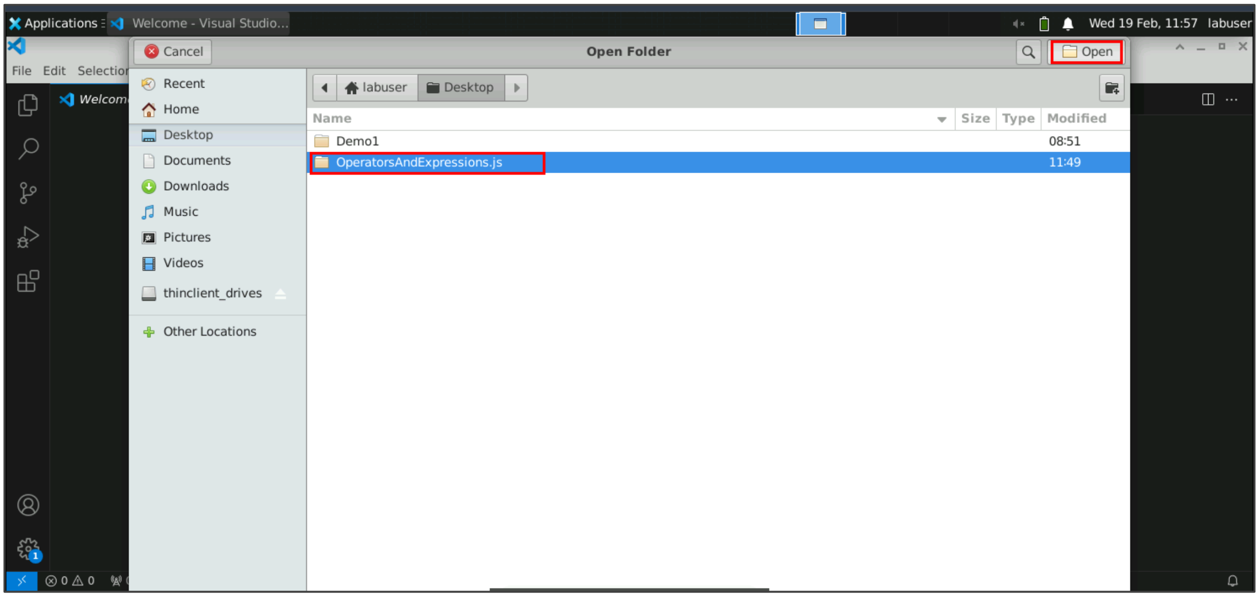
The **Visual Studio Code** opens as shown below:


Applications
Welcome - Visual Studio...
Wed 19 Feb, 11:54 labuser
Welcome - Visual Studio Code
File Edit Selection View Go Run Terminal Help
Welcome ×
Start
New File...
Open File...
Open Folder...
Clone Git Repository...
Connect to...
Recent
Demo1 ~/Desktop
Walkthroughs
Get Started with VS Code
Customize your editor, learn the basics, and start coding
Learn the Fundamentals
Show welcome page on startup
0 0 0

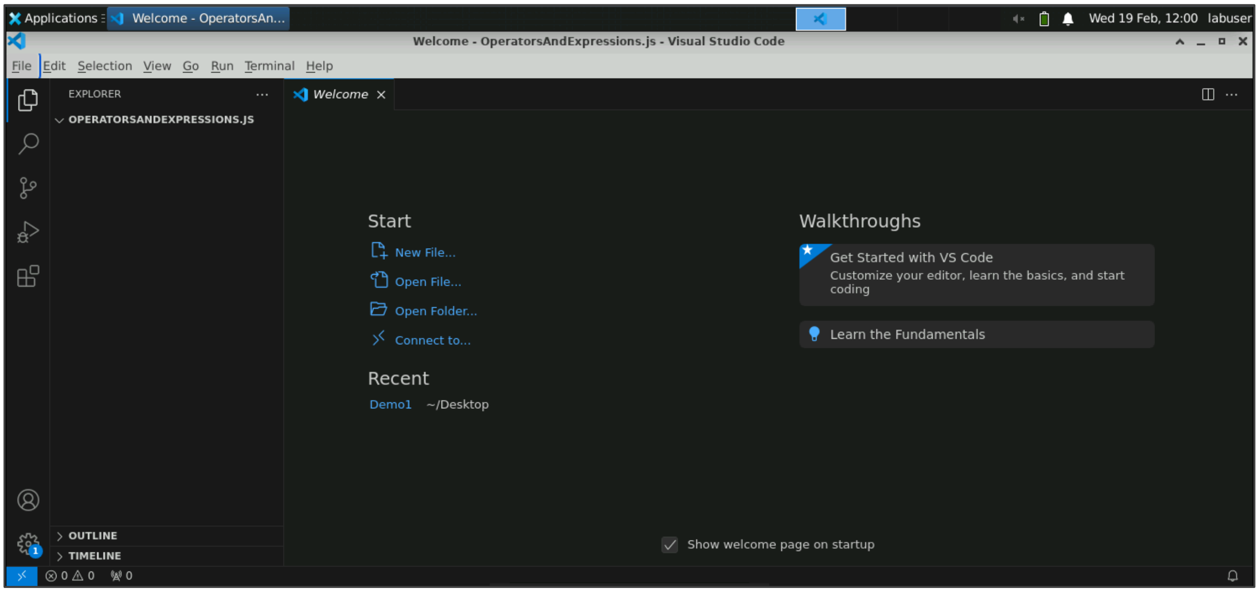
2.2 Click on **File**, then click on **Open Folder…**


Applications
Welcome - Visual Studio...
Wed 19 Feb, 08:50 labuser
Welcome - Visual Studio Code
File Edit Selection View Go Run Terminal Help
New Text File Ctrl+N
New File... Super+Alt+Ctrl+N
New Window Ctrl+Shift+N
New Window with Profile
Open File... Ctrl+O
Open Folder... [Ctrl+K Ctrl+O]
Open Workspace from File...
Open Recent
Add Folder to Workspace...
Save Workspace As...
Duplicate Workspace
Save Ctrl+S
Save As... Ctrl+Shift+S
Save All
Walkthroughs
Get Started with VS Code
Customize your editor, learn the basics, and start coding
Learn the Fundamentals
a folder to start.
Show welcome page on startup
0 0 0

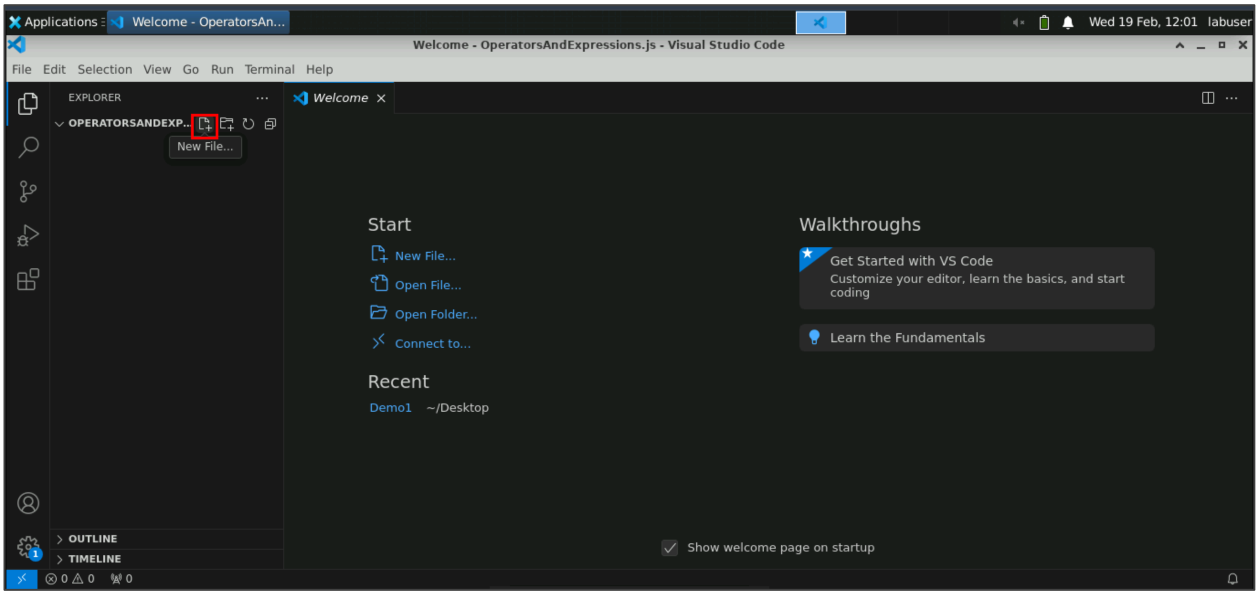
2.3 Select the **OperatorsAndExpressions.js** folder and click on the **Open** icon to open the folder in VS Code



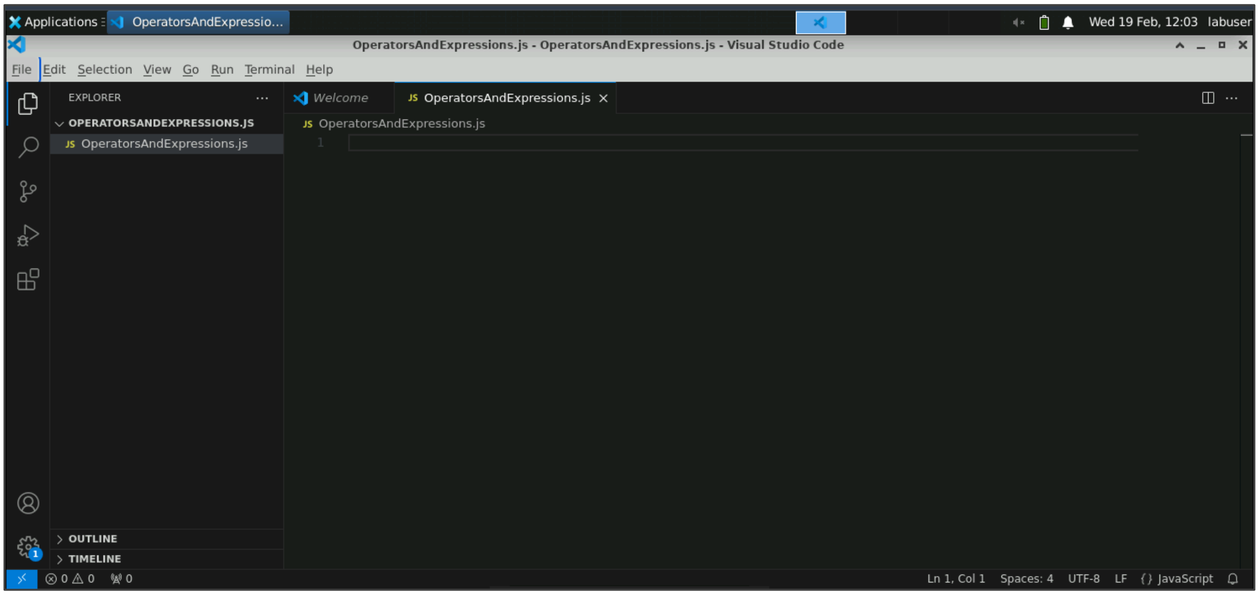
The folder opens in VS Code as shown below:


Applications
Welcome - OperatorsAn...
Wed 19 Feb, 12:00 labuser
Welcome - OperatorsAndExpressions.js - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
OPERATORSANDEXPRESSIONS.JS
Welcome
Start
New File...
Open File...
Open Folder...
Connect to...
Recent
Demo1 ~/Desktop
Walkthroughs
Get Started with VS Code
Customize your editor, learn the basics, and start coding
Learn the Fundamentals
OUTLINE
TIMELINE
Show welcome page on startup
0 0 0

2.4 Click on the **New File...** icon to create a new file named **OperatorsAndExpressions.js**



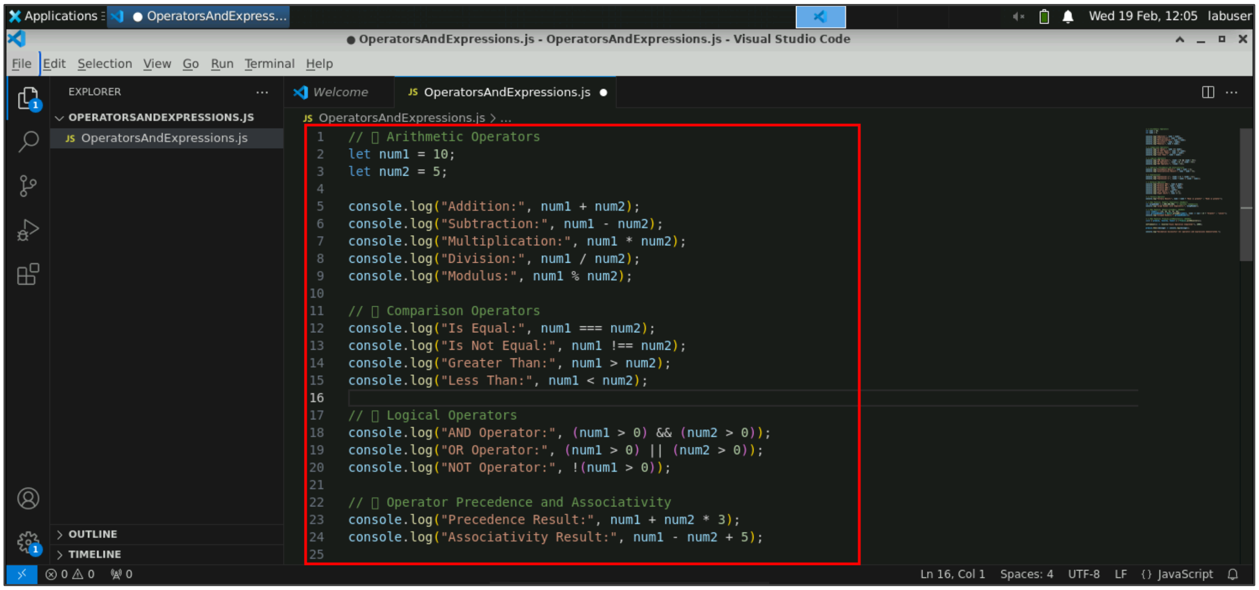
The file gets created as shown below:


Applications
OperatorsAndExpressio...
Wed 19 Feb, 12:03 labuser
OperatorsAndExpressions.js - OperatorsAndExpressions.js - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
OPERATORSANDEXPRESSIONS.JS
OperatorsAndExpressions.js
Welcome
OperatorsAndExpressions.js
OperatorsAndExpressions.js
OUTLINE
TIMELINE
0 0 0
Ln 1, Col 1 Spaces: 4 UTF-8 LF {} JavaScript

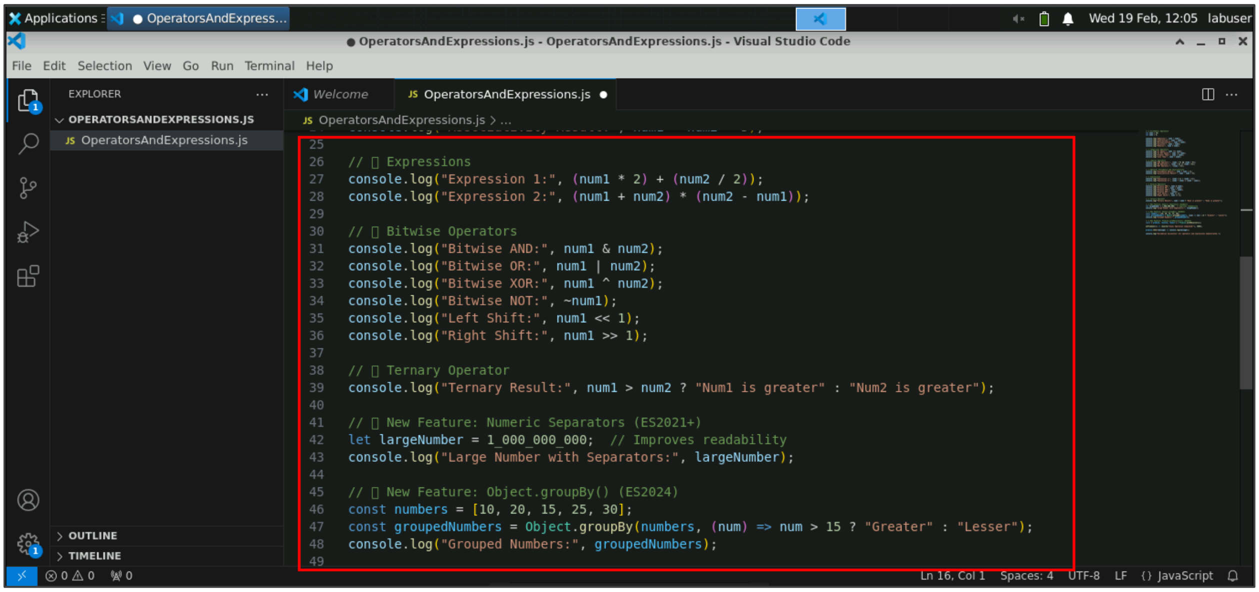
2.5 Enter the below code and save the file:

```
//Arithmetic Operators
let num1 = 10;
let num2 = 5;

console.log("Addition:", num1 + num2);
console.log("Subtraction:", num1 - num2);
console.log("Multiplication:", num1 * num2);
console.log("Division:", num1 / num2);
console.log("Modulus:", num1 % num2);

//Comparison Operators
console.log("Is Equal:", num1 === num2);
console.log("Is Not Equal:", num1 !== num2);
console.log("Greater Than:", num1 > num2);
console.log("Less Than:", num1 < num2);

//Logical Operators
console.log("AND Operator:", (num1 > 0) && (num2 > 0));
console.log("OR Operator:", (num1 > 0) || (num2 > 0));
console.log("NOT Operator:", !(num1 > 0));

//Operator Precedence and Associativity
console.log("Precedence Result:", num1 + num2 * 3);
console.log("Associativity Result:", num1 - num2 + 5);
```

```
//Expressions
console.log("Expression 1:", (num1 * 2) + (num2 / 2));
console.log("Expression 2:", (num1 + num2) * (num2 - num1));

//Bitwise Operators
console.log("Bitwise AND:", num1 & num2);
console.log("Bitwise OR:", num1 | num2);
console.log("Bitwise XOR:", num1 ^ num2);
console.log("Bitwise NOT:", ~num1);
console.log("Left Shift:", num1 << 1);
console.log("Right Shift:", num1 >> 1);

//Ternary Operator
console.log("Ternary Result:", num1 > num2 ? "Num1 is greater" : "Num2 is greater");

//New Feature: Numeric Separators (ES2021+)
let largeNumber = 1_000_000_000;  // Improves readability
console.log("Large Number with Separators:", largeNumber);

//New Feature: Object.groupBy() (ES2024)
const numbers = [10, 20, 15, 25, 30];
const groupedNumbers = Object.groupBy(numbers, (num) => num > 15 ? "Greater" :
"Lesser");
console.log("Grouped Numbers:", groupedNumbers);

//New Feature: Promise.withResolvers() (ES2024)
const { promise, resolve, reject } = Promise.withResolvers();

setTimeout(() => resolve("Async Operation Completed!"), 2000);

promise.then((message) => console.log(message));

console.log("Validation Successful! All operators and expressions demonstrated.");
```

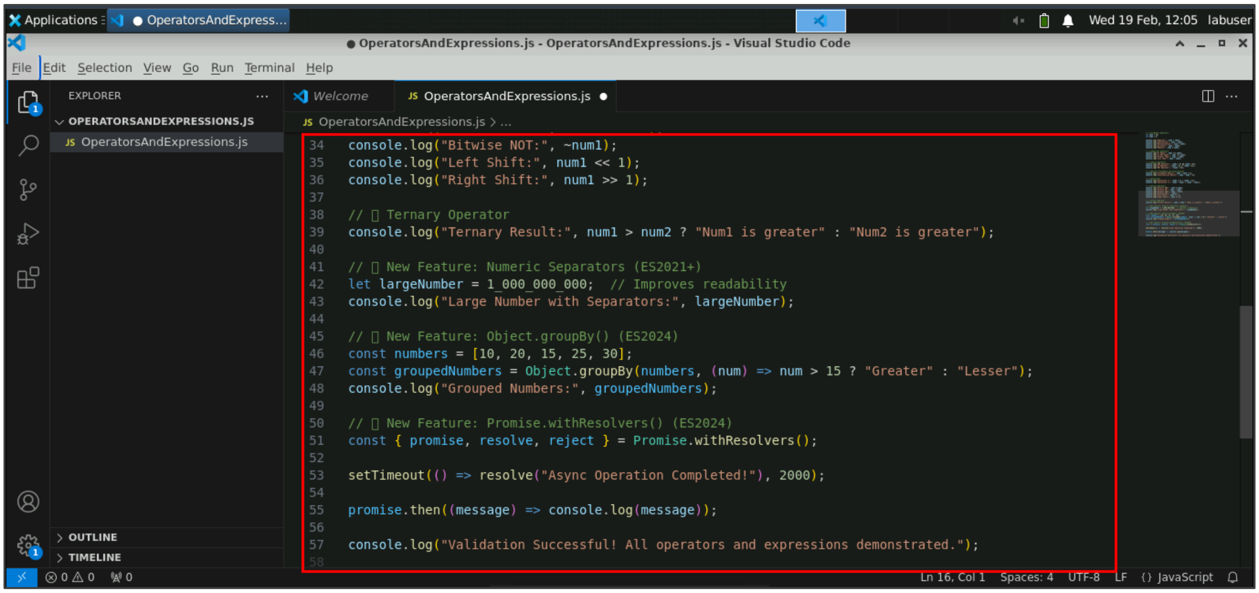

Applications
OperatorsAndExpress...
Wed 19 Feb, 12:05 labuser
● OperatorsAndExpressions.js - OperatorsAndExpressions.js - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
OPERATORSANDEXPRESSIONS.JS
OperatorsAndExpressions.js
Welcome
OperatorsAndExpressions.js ●
OperatorsAndExpressions.js > ...
OUTLINE
TIMELINE
Ln 16, Col 1 Spaces: 4 UTF-8 LF {} JavaScript

```
// □ Arithmetic Operators
let num1 = 10;
let num2 = 5;

console.log("Addition:", num1 + num2);
console.log("Subtraction:", num1 - num2);
console.log("Multiplication:", num1 * num2);
console.log("Division:", num1 / num2);
console.log("Modulus:", num1 % num2);

// □ Comparison Operators
console.log("Is Equal:", num1 === num2);
console.log("Is Not Equal:", num1 !== num2);
console.log("Greater Than:", num1 > num2);
console.log("Less Than:", num1 < num2);

// □ Logical Operators
console.log("AND Operator:", (num1 > 0) && (num2 > 0));
console.log("OR Operator:", (num1 > 0) || (num2 > 0));
console.log("NOT Operator:", !(num1 > 0));

// □ Operator Precedence and Associativity
console.log("Precedence Result:", num1 + num2 * 3);
console.log("Associativity Result:", num1 - num2 + 5);

```


Applications
OperatorsAndExpress...
Wed 19 Feb, 12:05 labuser
● OperatorsAndExpressions.js - OperatorsAndExpressions.js - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
OPERATORSANDEXPRESSIONS.JS
OperatorsAndExpressions.js
Welcome
OperatorsAndExpressions.js ●
OperatorsAndExpressions.js > ...
OUTLINE
TIMELINE
Ln 16, Col 1 Spaces: 4 UTF-8 LF {} JavaScript

```

// □ Expressions
console.log("Expression 1:", (num1 * 2) + (num2 / 2));
console.log("Expression 2:", (num1 + num2) * (num2 - num1));

// □ Bitwise Operators
console.log("Bitwise AND:", num1 & num2);
console.log("Bitwise OR:", num1 | num2);
console.log("Bitwise XOR:", num1 ^ num2);
console.log("Bitwise NOT:", ~num1);
console.log("Left Shift:", num1 << 1);
console.log("Right Shift:", num1 >> 1);

// □ Ternary Operator
console.log("Ternary Result:", num1 > num2 ? "Num1 is greater" : "Num2 is greater");

// □ New Feature: Numeric Separators (ES2021+)
let largeNumber = 1_000_000_000;  // Improves readability
console.log("Large Number with Separators:", largeNumber);

// □ New Feature: Object.groupBy() (ES2024)
const numbers = [10, 20, 15, 25, 30];
const groupedNumbers = Object.groupBy(numbers, (num) => num > 15 ? "Greater" : "Lesser");
console.log("Grouped Numbers:", groupedNumbers);

```

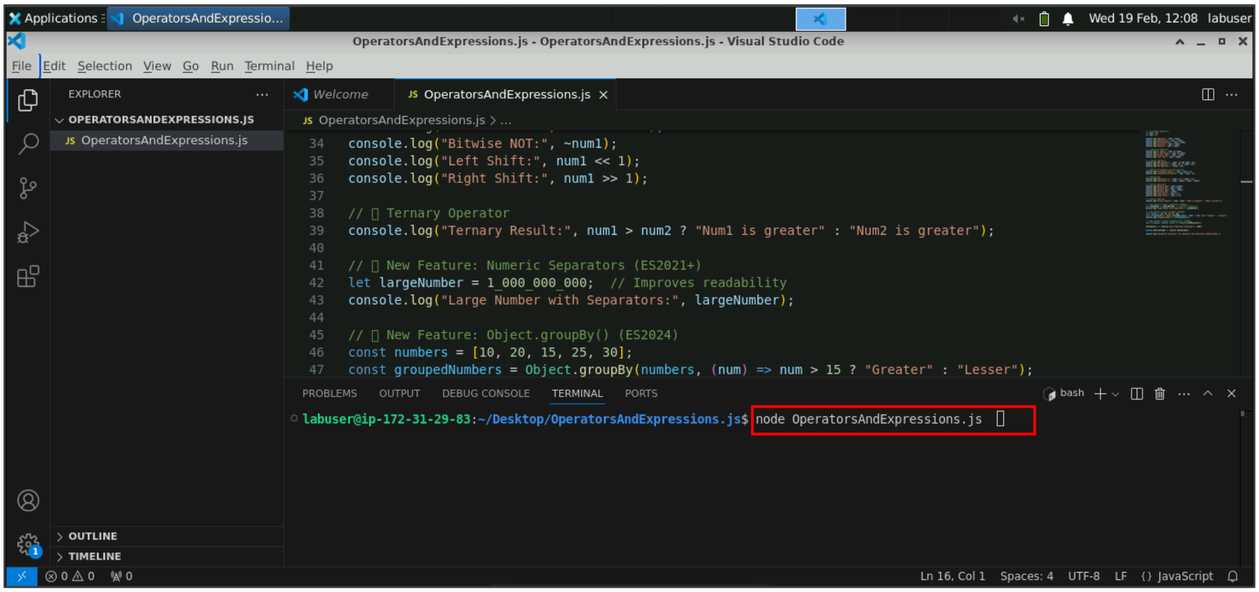
.


Applications
OperatorsAndExpress...
Wed 19 Feb, 12:05 labuser
● OperatorsAndExpressions.js - OperatorsAndExpressions.js - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
OPERATORSANDEXPRESSIONS.JS
OperatorsAndExpressions.js
Welcome
OperatorsAndExpressions.js
OperatorsAndExpressions.js > ...
OUTLINE
TIMELINE
Ln 16, Col 1 Spaces: 4 UTF-8 LF {} JavaScript

```javascript
console.log("Bitwise NOT:", ~num1);
console.log("Left Shift:", num1 << 1);
console.log("Right Shift:", num1 >> 1);

// □ Ternary Operator
console.log("Ternary Result:", num1 > num2 ? "Num1 is greater" : "Num2 is greater");

// □ New Feature: Numeric Separators (ES2021+)
let largeNumber = 1_000_000_000;  // Improves readability
console.log("Large Number with Separators:", largeNumber);

// □ New Feature: Object.groupBy() (ES2024)
const numbers = [10, 20, 15, 25, 30];
const groupedNumbers = Object.groupBy(numbers, (num) => num > 15 ? "Greater" : "Lesser");
console.log("Grouped Numbers:", groupedNumbers);

// □ New Feature: Promise.withResolvers() (ES2024)
const { promise, resolve, reject } = Promise.withResolvers();

setTimeout(() => resolve("Async Operation Completed!"), 2000);

promise.then((message) => console.log(message));

console.log("Validation Successful! All operators and expressions demonstrated.");
```

2.6 Open the terminal and run the command below:

node OperatorsAndExpressions.js

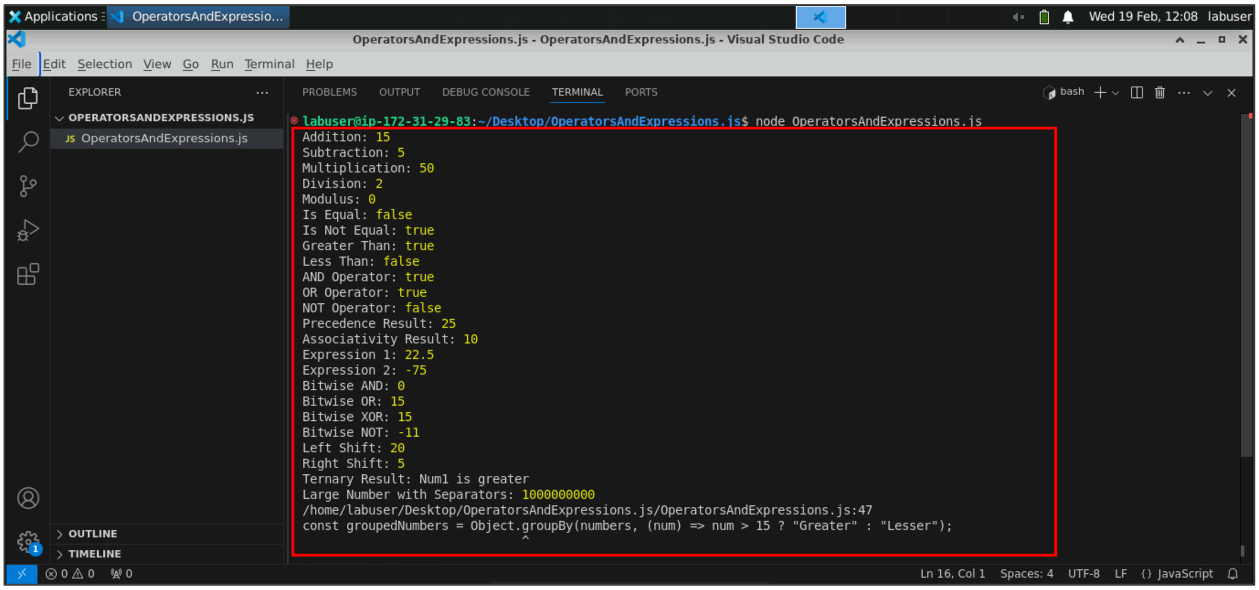

Applications
OperatorsAndExpressio...
Wed 19 Feb, 12:08 labuser
OperatorsAndExpressions.js - OperatorsAndExpressions.js - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
OPERATORSANDEXPRESSIONS.JS
OperatorsAndExpressions.js
Welcome
OperatorsAndExpressions.js
OperatorsAndExpressions.js > ...
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
bash
OUTLINE
TIMELINE
Ln 16, Col 1 Spaces: 4 UTF-8 LF {} JavaScript

```javascript
console.log("Bitwise NOT:", ~num1);
console.log("Left Shift:", num1 << 1);
console.log("Right Shift:", num1 >> 1);

// □ Ternary Operator
console.log("Ternary Result:", num1 > num2 ? "Num1 is greater" : "Num2 is greater");

// □ New Feature: Numeric Separators (ES2021+)
let largeNumber = 1_000_000_000;  // Improves readability
console.log("Large Number with Separators:", largeNumber);

// □ New Feature: Object.groupBy() (ES2024)
const numbers = [10, 20, 15, 25, 30];
const groupedNumbers = Object.groupBy(numbers, (num) => num > 15 ? "Greater" : "Lesser");
```

```
labuser@ip-172-31-29-83:~/Desktop/OperatorsAndExpressions.js$ node OperatorsAndExpressions.js
```

.

The output will appear as shown below:



```
labuser@ip-172-31-29-83:~/Desktop/OperatorsAndExpressions.js$ node OperatorsAndExpressions.js
Addition: 15
Subtraction: 5
Multiplication: 50
Division: 2
Modulus: 0
Is Equal: false
Is Not Equal: true
Greater Than: true
Less Than: false
AND Operator: true
OR Operator: true
NOT Operator: false
Precedence Result: 25
Associativity Result: 10
Expression 1: 22.5
Expression 2: -75
Bitwise AND: 0
Bitwise OR: 15
Bitwise XOR: 15
Bitwise NOT: -11
Left Shift: 20
Right Shift: 5
Ternary Result: Num1 is greater
Large Number with Separators: 1000000000
/home/labuser/Desktop/OperatorsAndExpressions.js/OperatorsAndExpressions.js:47
const groupedNumbers = Object.groupBy(numbers, (num) => num > 15 ? "Greater" : "Lesser");
                              ^
```

Applications
OperatorsAndExpressio...
Wed 19 Feb, 12:08 labuser
OperatorsAndExpressions.js - OperatorsAndExpressions.js - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
OPERATORSANDEXPRESSIONS.JS
OperatorsAndExpressions.js
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
bash
OUTLINE
TIMELINE
Ln 16, Col 1 Spaces: 4 UTF-8 LF {} JavaScript

The above code demonstrates various operators, including arithmetic, comparison, logical, and bitwise operations. It also covers operator precedence, associativity, expressions, and the ternary operator in JavaScript. Finally, it validates the execution and correctness of these operators through systematic evaluation.

By following the above steps, you have successfully demonstrated the usage of operators and expressions in JavaScript, ensuring an accurate representation and validation of their functionality.