

# Advanced Web Programming (3161611)

## Practicals

Bhagya Patel

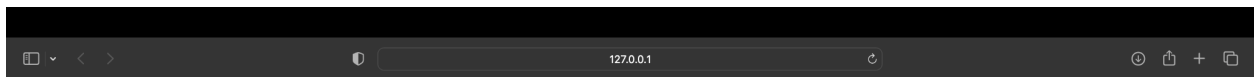
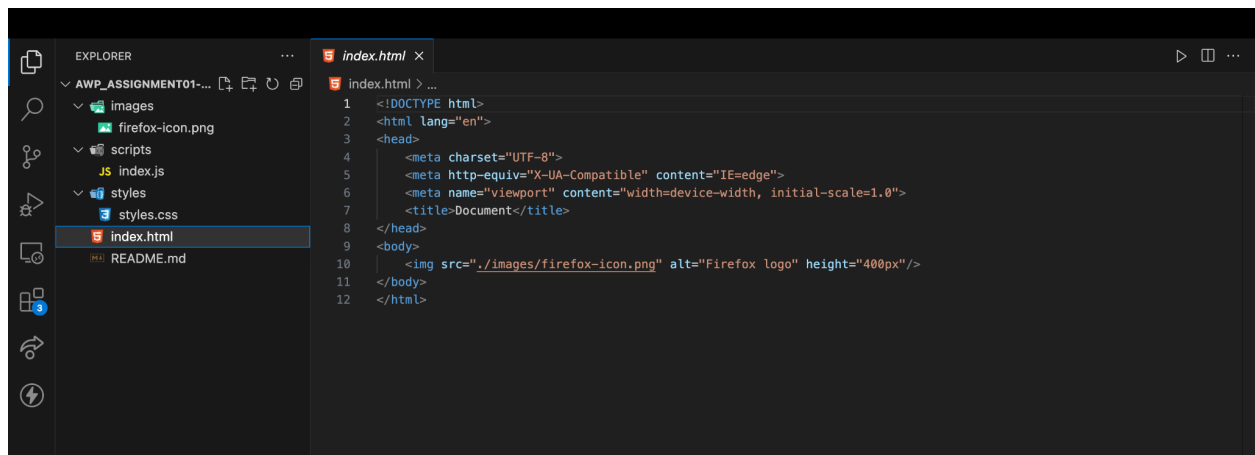
200420116002

IT 3rd



# Practical 01

- Install Visual Studio Code
- Install git
- Setup username and email in git
- Signup in github.com
- Create a blank repository in github.com
- Clone that newly created repository in your computer via Visual Studio Code
- Push Your Repository to github
- Submit the Github Repository URL



The screenshot shows the GitHub interface for the repository `Code-Knightt/AWP_assignment01`. The repository is public and has 0 stars, 1 watching, and 0 forks. The main content area displays a commit history table with 3 commits. The most recent commit, `1580772` on Feb 2, is titled `mvc Fill empty directories`. The commit details show a list of files: `images`, `scripts`, `styles`, `README.md`, and `index.html`. The `README.md` file is expanded, showing the repository name `AWP_assignment01` and a description: `Hi this is the github repository for the assignment 01 of Advanced Web Programming (3161611)`. The right sidebar contains sections for `About`, `Releases`, and `Packages`, all of which are currently empty.

File	Commit Message	Time
<code>images</code>	Create directory structure and add firefox logo	3 months ago
<code>scripts</code>	Fill empty directories	3 months ago
<code>styles</code>	Fill empty directories	3 months ago
<code>README.md</code>	Create README.md	3 months ago
<code>index.html</code>	Create directory structure and add firefox logo	3 months ago

**README.md**

## AWP\_assignment01

Hi this is the github repository for the assignment 01 of Advanced Web Programming (3161611)

[https://github.com/Code-Knightt/AWP\\_assignment01](https://github.com/Code-Knightt/AWP_assignment01)

## Practical 02

Create a simple guess the number type game. It should choose a random number between 1 and 100, then challenge the player to guess the number in 10 turns. After each turn the player should be told if they are right or wrong, and if they are wrong, whether the guess was too low or too high. It should also tell the player what numbers they previously guessed. The game will end once the player guesses correctly, or once they run out of turns. When the game ends, the player should be given an option to start playing again.

### assignment\_02.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Guessing Game</title>
  <link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link
href="https://fonts.googleapis.com/css2?family=Roboto:wght@400;500;900&display=swap"
rel="stylesheet">

<style>
  h1{
    font-family: 'Roboto';
    font-weight: 900;
  }
  h3{
    font-family: 'Roboto';
    font-weight: 500;
  }
  p, li, button{
    font-family: 'Roboto';
    font-weight: 400;
  }
```

```

    body{
        display: flex;
        flex-direction: column;
        align-items: center;
    }
</style>
</head>
<body>
    <h1>Guessing Game</h1>
    <h3>Rules</h3>
    <ul>
        <li>You have to guess a number between 1 and 100</li>
        <li>You will get 10 turns to guess the correct number</li>
        <li>The game will provide hints based on the input you provide</li>
    </ul>

    <form>
        <input type="number" id="guess_input" required>
        <button id="button" type="submit" >Guess</button>
    </form>

    <p id="previous"></p>
    <p>Guesses remaining: <span id="guesses"></span></p>
    <p id="hint"></p>
    <button id="reset" type="button">Reset</button>

    <script>
        const number = Math.floor((Math.random() * 100) + 1); console.log(number);
        const input = document.getElementById("guess_input");
        const button = document.getElementById('button');
        const reset = document.getElementById('reset');
        const hint = document.getElementById("hint");
        const guesses = document.getElementById("guesses");
        const previous = document.getElementById("previous");

        let numberOfGuesses = 10;
        guesses.textContent = `${numberOfGuesses}`;

        reset.addEventListener('click', ()=>{
            location.reload();
        })
    </script>

```

```

button.addEventListener('click', () => {
    const guess = input.value;

    if(!guess){
        return;
    }

    if(numberOfGuesses == 10){
        previous.textContent = "Previous guesses: ";
    }

    if(numberOfGuesses == 0){
        hint.textContent = `Out of guesses! The correct number was ${number}.`;
        return;
    }

    numberOfGuesses -= 1;
    guesses.textContent = `${numberOfGuesses}`;
    previous.textContent += `${guess} `;
    input.value = "";

    if(guess > number){
        hint.textContent = "Wrong! Too high";
    } else if (guess < number){
        hint.textContent = "Wrong! Too low"
    } else {
        hint.textContent="Correct! Guessed Correctly";
        input.setAttribute('disabled', '')
        button.setAttribute('disabled', '');
    }

})

</script>
</body>
</html>

```

## Guessing Game

### Rules

- You have to guess a number between 1 and 100
- You will get 10 turns to guess the correct number
- The game will provide hints based on the input you provide

Previous guesses: 34 56 78 99 90 94 95

Guesses remaining: 3

Correct! Guessed Correctly



## Practical 03

1. Create an HTML file that specifies a page that contains a heading and two paragraphs of text. Use the HTML tags `<h1>`, `</h1>`, `<p>`, and `</p>` in this exercise. As the texts in the heading and paragraphs you can use any texts you like.
2. Add an unordered list to your first web page. After you have created your unordered list, check out what happens when you convert it to an ordered list by replacing the tags `<ul>` and `</ul>` with `<ol>` and `</ol>`, respectively.
3. Add an image to your web page. In this exercise you must use the `<img>` tag. As an image, you can use any .jpg or .png file you find on the Internet.
4. Create another .html file that contains a heading and a couple of paragraphs. You could name this new file `another_page.html`, and you should place it into the same folder where your first .html is. After you have created the new .html page, add a link to the first page so that the browser will load `another_page.html` when you click the text Go to the other page. in the first page.
5. It is possible to use a picture (image) as a link. Modify your page so that the picture that is on your page will also serve as a link that leads to another page.

### assignment\_03a.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Assignment 3</title>
  <link rel="preconnect" href="https://fonts.googleapis.com">
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
  <link
href="https://fonts.googleapis.com/css2?family=Roboto:wght@400;500;900&display=swap"
rel="stylesheet">
```

```

<style>
  h1{
    font-family: 'Roboto';
    font-weight: 900;
  }
  h3{
    font-family: 'Roboto';
    font-weight: 500;
  }
  p, li, button, a{
    font-family: 'Roboto';
    font-weight: 400;
  }

  body{
    display: flex;
    flex-direction: column;
  }

  img{
    position: absolute;
    right: 10%;
    width: 40vw;
    height: auto;
  }

</style>
</head>
<body>
  <h1>The benefits of web development</h1>
  <p>Nowadays web development has grown a lot and almost every programmer should know
it</p>
  <p>Here are some reasons why</p>
  <ul>
    <li>Easy to learn</li>
    <li>Lots of resources available</li>
    <li>Money Money Money</li>
  </ul>

```

```

    <a href="assignment_03b.html">
</a><a href="assignment_03b.html">Get started</a>
</body>
</html>

```

## assignment\_03b.html

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Assignment 3</title>
    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
    <link
href="https://fonts.googleapis.com/css2?family=Roboto:wght@400;500;900&display=swap"
rel="stylesheet">

    <style>
        h1{
            font-family: 'Roboto';
            font-weight: 900;
        }
        h3{
            font-family: 'Roboto';
            font-weight: 500;
        }
        p, li, button, a{
            font-family: 'Roboto';
            font-weight: 400;
        }
        body{
            display: flex;
            flex-direction: column;
            align-items: center;
        }
        p{
            text-align: left;

```

```

        max-width: 60%;
    }
</style>
</head>
<body>
    <h1>How to get started</h1>
    <h3>Learn a language</h3>
    <p>The first big step to a career in web development is to learn the necessary
languages, libraries,
        and frameworks for that area. Getting familiar with these as well as other
tools and common
        terminology will make the next step of education much easier.
    </p>
    <h3>Create Projects</h3>
    <p>
        As you're going through your learning, it's vital that you also have projects
to show for all of your coding.
        With these projects you can start to create your web developer portfolio—a
great way of attracting attention
        among potential employers and future clients. If you're focussing more on
frontend development,
        then being able to show off your skills is even more important.
    </p>
</body>
</html>

```

## The benefits of web development

Nowadays web development has grown a lot and almost every programmer should know it

Here are some reasons why

- Easy to learn
- Lots of resources available
- Money Money Money

[Get started](#)



## How to get started

### Learn a language

The first big step to a career in web development is to learn the necessary languages, libraries, and frameworks for that area. Getting familiar with these as well as other tools and common terminology will make the next step education much easier.

### Create Projects

As you're going through your learning, it's vital that you also have projects to show for all of your coding. With these projects you can start to create your web developer portfolio—a great way of attracting attention among potential employers and future clients. If you're focussing more on frontend development, then being able to show off your skills is even more important.



# Practical 04

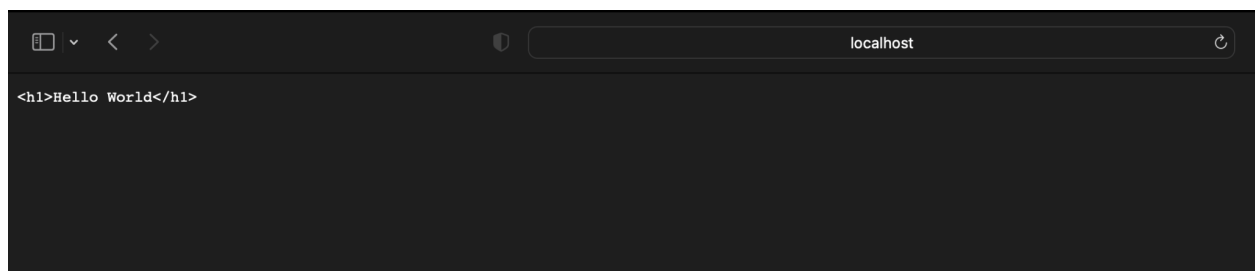
Create web application in nodejs to display "hello world" in browser

## assignment\_04.js

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

const server = http.createServer((req, res) => {
  res.write("<h1>Hello World</h1>");
  res.end();
});

server.listen(3000, () => {
  console.log("Started server on port 3000");
});
```







# Practical 05

Read file " myfile.txt" in nodejs using synchronous and asynchronous file function and find out How much time is saved by non -blocking asynchronous I/O operation?

Note: Create separate nodejs program for blocking and non-blocking I/O

```
JS assignment_05_sync.js X
JS assignment_05_sync.js > ...
1  const fs = require("fs");
2
3  console.time("Time: ");
4  const file = fs.readFileSync("myfile.txt");
5  console.log("Synchronous \n");
6  console.log(file.toString());
7  console.timeEnd("Time: ");
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS C:\Users\mvc\Desktop\NodeJS> node .\assignment_05_sync.js
Synchronous

Hi I am Bhagya
This file is for reading purposes

Time: : 4.123ms
```

```
JS assignment_05_async.js X
JS assignment_05_async.js > fs.readFile("myfile.txt") callback
1  const fs = require("fs");
2
3  console.time("Time: ");
4  fs.readFile("myfile.txt", (err, data)=>{
5      if(err){
6          console.error(err);
7      } else {
8          console.log("Asynchronous \n");
9          console.log(data.toString());
10         console.timeEnd("Time: ");
11     }
12 })
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS C:\Users\mvc\Desktop\NodeJS> node assignment_05_async.js
Asynchronous

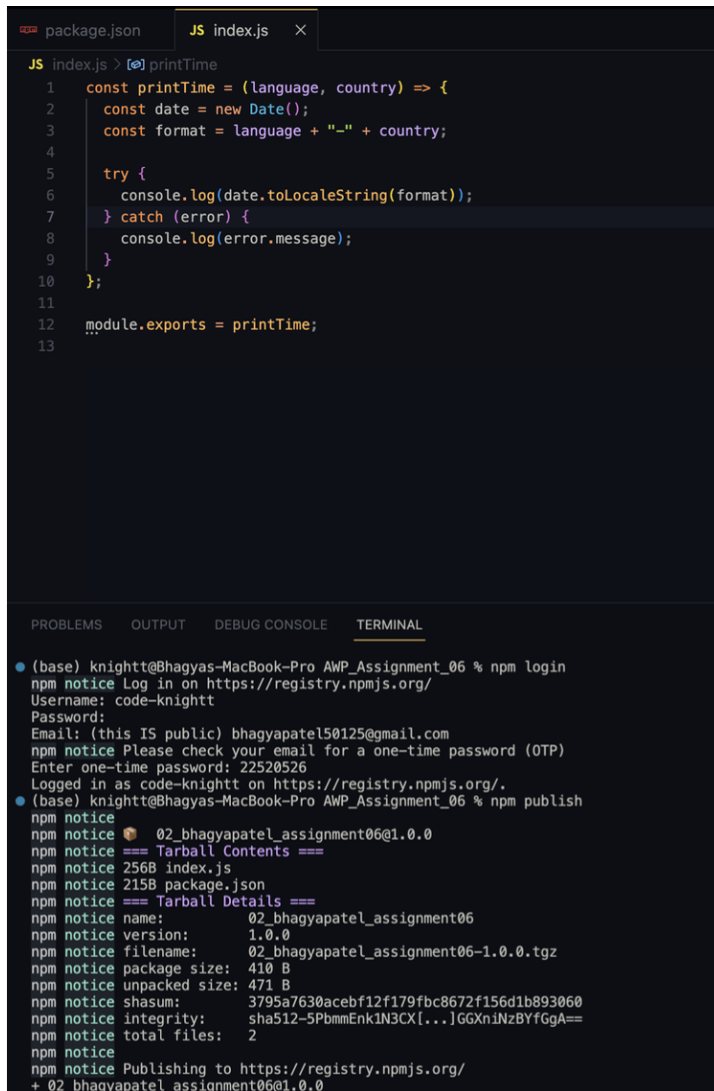
Hi I am Bhagya
This file is for reading purposes

Time: : 5.613ms
```



# Practical 06

Create and publish your own Node package to display current date and time with appropriate message



```
package.json  JS index.js  X

JS index.js > printTime
1  const printTime = (language, country) => {
2    const date = new Date();
3    const format = language + "-" + country;
4
5    try {
6      console.log(date.toLocaleString(format));
7    } catch (error) {
8      console.log(error.message);
9    }
10 };
11
12 module.exports = printTime;
13

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

• (base) knight@Bhagyas-MacBook-Pro AWP_Assignment_06 % npm login
npm notice Log in on https://registry.npmjs.org/
Username: code-knightt
Password:
Email: (this IS public) bhagyapatel50125@gmail.com
npm notice Please check your email for a one-time password (OTP)
Enter one-time password: 22520526
Logged in as code-knightt on https://registry.npmjs.org/.
• (base) knight@Bhagyas-MacBook-Pro AWP_Assignment_06 % npm publish
npm notice
npm notice 📦 02_bhagyapatel_assignment06@1.0.0
npm notice === Tarball Contents ===
npm notice 256B index.js
npm notice 215B package.json
npm notice === Tarball Details ===
npm notice name:          02_bhagyapatel_assignment06
npm notice version:       1.0.0
npm notice filename:      02_bhagyapatel_assignment06-1.0.0.tgz
npm notice package size:  410 B
npm notice unpacked size: 471 B
npm notice shasum:        3795a7630acebf12f179fbc8672f156d1b893060
npm notice integrity:      sha512-5PbmmEnk1N3CX[...]GGXniNzBYfGgA==
npm notice total files:   2
npm notice
npm notice Publishing to https://registry.npmjs.org/
+ 02_bhagyapatel_assignment06@1.0.0
```

### 02\_bhagyapatel\_assignment06

1.0.0 • Public • Published 3 minutes ago

Readme
Code Beta
0 Dependencies
0 Dependents
1 Versions
Settings

This package does not have a README. [Add a README](#) to your package so that users know how to get started.

Keywords

none

Install

> npm i 02\_bhagyapatel\_assignment06

Version	License
1.0.0	ISC
Unpacked Size	Total Files
471 B	2

```
(base) knight@Bhagyas-MacBook-Pro AWP_Assignment_06 % npm i 02_bhagyapatel_assignment06
added 1 package, and audited 2 packages in 2s
found 0 vulnerabilities
```

JS test.js

×

JS test.js > ...

```

1  const printTime = require("02_bhagyapatel_assignment06");
2
3  printTime("en", "IN");
4  printTime("ta", "IN");
5  printTime("en", "US");
6  printTime("ko", "KR");
7  printTime("e", "IN");
8

```

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

```

● (base) knight@Bhagyas-MacBook-Pro AWP_Assignment_06 % node test.js
9/3/2023, 5:46:39 pm
9/3/2023, 5:46:39 PM
3/9/2023, 5:46:39 PM
2023. 3. 9. 오후 5:46:39
Incorrect locale information provided

```

## Practical 07

Create a Hello World Web Page using AngularJS. The Web page should display the message "Hello world" by using the model value from the controller. Also make sure that the controller should be a part of a Module.

```
<html>
  <head>
    <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.3.14/angular.min.js"></script>
  </head>
  <body ng-app="assignment07">
    <div ng-controller="info">
      <table border="1">
        <tr>
          <td>Name: {{agent.name}}</td>
        </tr>
        <tr>
          <td>Age: {{agent.age}}</td>
        </tr>
        <tr>
          <td>Phone: {{agent.phone}}</td>
        </tr>
        <tr ng-repeat="movie in agent.movies">
          <td>Title: {{movie.title}}, Year:{{movie.year}}</td>
        </tr>
      </table>
    </div>
    <script>
      var module = angular.module("assignment07", []);
      module.controller("info", initialize);

      function initialize($scope) {
        $scope.agent = {
          name: "James Bond",
          age: 47,
          phone: "0070070072",
          movies: [
            { title: "Casino Royale", year: 2006 },
            { title: "Skyfall", year: 2012 },
          ]
        }
      }
    </script>
  </body>
</html>
```

```
        { title: "No Time to Die", year: 2021 },
      ],
    };
  }
</script>
</body>
</html>
```

---

Name: James Bond
Age: 47
Phone: 0070070072
Title: Casino Royale, Year:2006
Title: Skyfall, Year:2012
Title: No Time to Die, Year:2021

# Practical 08

Create a Signup Form in AngularJS. The Sign up form should ask for Name, Mobile, Email, Address, Date of Birth, Gender, Username , Password and Confirm Password. The Form should have two buttons, one for reset and one for submit. The Submit Button should be enabled only when All the input values are validated.

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.3.14/angular.min.js"></script>
  <title>Sign Up Form</title>
</head>

<body ng-app='signup'>
  <h1>Angular Form</h1>
  <div>
    <form name="myform">
      <p>*Name: <input type="text" name="name" ng-model="name" required></p>
      <p>*Phone: <input type="number" ng-model="phone" required></p>
      <p>*Email: <input type="email" ng-model="email" required></p>
      <p>Address: <textarea></textarea></p>
      <p>*DOB: <input type="date"></p>
      <p>Gender: <input type="radio" name="gen">Male</input> <input type="radio"
name="gen">Female</input></p>
      <p>*Username: <input type="text" required></p>
      <p>*Password: <input type="password" name="pass" ng-model="pass"
required></p>
      <p>*Confirm: <input type="password" name="confirm" ng-model="confirm"
required></p>
      <button type="reset">Reset</button>
      <button type="submit"
        ng-disabled="myform.$invalid || myform.pass.$viewValue !==
myform.confirm.$viewValue">Submit</button>
```

```
        </form>
    </div>
    <script>
        const module = angular.module('signup', []);
    </script>
</body>

</html>
```

## Angular Form

\*Name:

\*Phone:

\*Email:

Address:

\*DOB:

Gender: ☐ Male ☐ Female

\*Username:

\*Password:

\*Confirm:



## Practical 09

Implement Node js program to handle divide by zero exception

```
JS assignment_09.js X
JS assignment_09.js >  divide
1  function handler(err, val){
2      if(err){
3          console.log("Error: " + err.message );
4          return;
5      } else {
6          console.log("Division Result is: "+val);
7      }
8
9  }
10
11 function divide(num1, num2, next){
12     if(num2 === 0){
13         next(new Error('Cannot divide by zero!'));
14         return;
15     }
16
17     next(null, num1/num2);
18 }
19
20 divide(9,0,handler);
21 divide(9,3,handler);
```

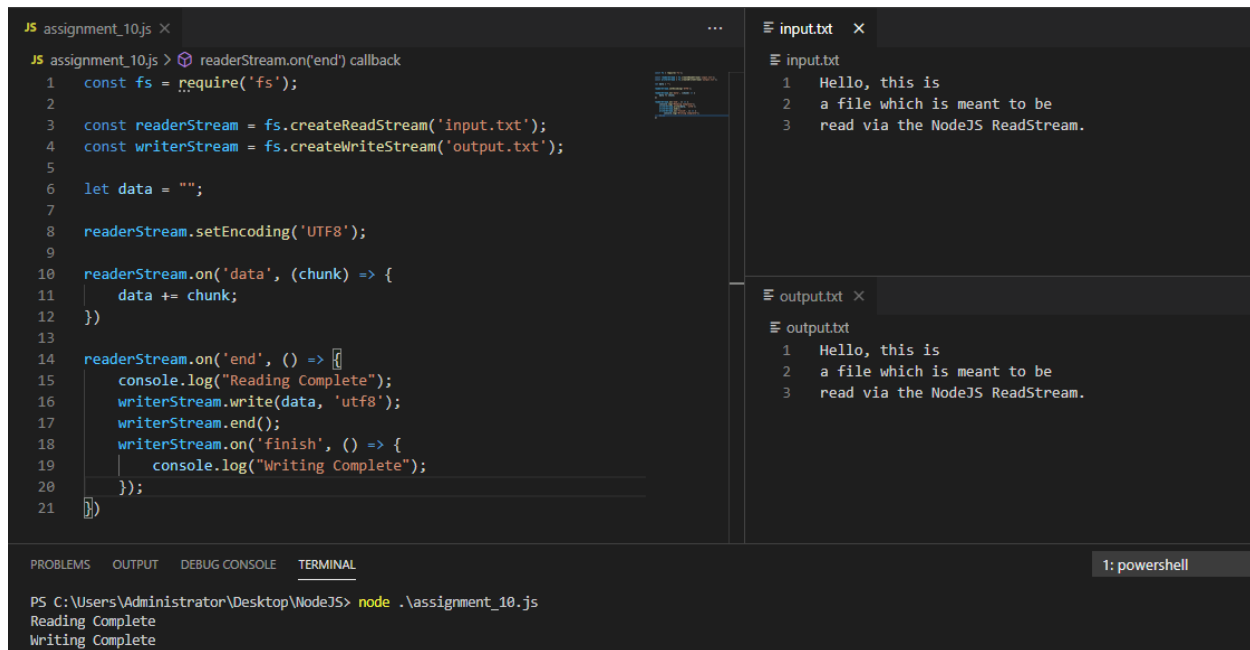
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL

```
PS C:\Users\Administrator\Desktop\NodeJS> node assignment_09
Error: Cannot divide by zero!
Division Result is: 3
PS C:\Users\Administrator\Desktop\NodeJS>
```



# Practical 10

Implement Node js program to read data from file "input.txt" and write these data to "output.txt" using concept of streams



The screenshot shows a Visual Studio Code editor with a JavaScript file named `assignment_10.js`. The code uses the `fs` module to create a `ReadStream` for `input.txt` and a `WriteStream` for `output.txt`. It sets the encoding to `UTF8` and listens for `'data'` and `'end'` events. The `'data'` event appends chunks to a `data` string, and the `'end'` event logs "Reading Complete", writes the `data` string to the output stream, and logs "Writing Complete".

```
JS assignment_10.js > readerStream.on('end') callback
1  const fs = require('fs');
2
3  const readerStream = fs.createReadStream('input.txt');
4  const writerStream = fs.createWriteStream('output.txt');
5
6  let data = "";
7
8  readerStream.setEncoding('UTF8');
9
10 readerStream.on('data', (chunk) => {
11   data += chunk;
12 })
13
14 readerStream.on('end', () => {
15   console.log("Reading Complete");
16   writerStream.write(data, 'utf8');
17   writerStream.end();
18   writerStream.on('finish', () => {
19     console.log("Writing Complete");
20   });
21 })
```

On the right, two file explorers are shown. `input.txt` contains:

```
1 Hello, this is
2 a file which is meant to be
3 read via the NodeJS ReadStream.
```

`output.txt` contains the same text:

```
1 Hello, this is
2 a file which is meant to be
3 read via the NodeJS ReadStream.
```

At the bottom, the `TERMINAL` tab shows the command `node .\assignment_10.js` and its output:

```
PS C:\Users\Administrator\Desktop\NodeJS> node .\assignment_10.js
Reading Complete
Writing Complete
```



# Practical 11

Create a Single Page Application using AngularJS.

```
<!DOCTYPE html>

<html ng-app="myApp">

<head>

    <script src=
"https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.4.7/angular.min.js">
    </script>
    <script src=
"https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.4.7/angular-route.min.js">
    </script>

    <style>
        body {
            text-align: center;
            font-family: Arial, Helvetica, sans-serif;
        }

        h1 {
            color: green;
        }
    </style>
</head>

<body>
    <h1>Jason</h1>
    <h3>Single Page Application in AngularJS</h3>

    <script type="text/ng-template" id="first.html">
        <h1>First Page</h1>
        <h2 style="color:green">
            Hello This is Jason
        </h2>
        <h3>{{message}}</h3>
    </script>
```

```

<script type="text/ng-template" id="second.html">
  <h1>Second Page</h1>
  <h2 style="color:green">
    I am an It Engineer
  </h2>
  <h3>{{message}}</h3>
</script>
<script type="text/ng-template" id="third.html">
  <h1>Third Page</h1>
  <h2 style="color:green">
    Contact me
  </h2>
  <h3>{{message}}</h3>
</script>

<a href="#/">First</a>
<a href="#/second">Second</a>
<a href="#/third">Third</a>

<div ng-view></div>

<script>
  var app = angular.module('myApp', []);
  var app = angular.module('myApp', ['ngRoute']);
  app.config(function ($routeProvider) {
    $routeProvider
      .when('/', {
        templateUrl: 'first.html',
        controller: 'FirstController'
      })

      .when('/second', {
        templateUrl: 'second.html',
        controller: 'SecondController'
      })

      .when('/third', {
        templateUrl: 'third.html',
        controller: 'ThirdController'
      })
  })

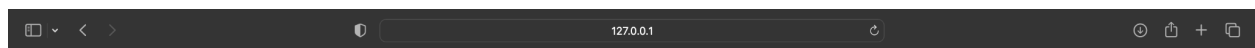
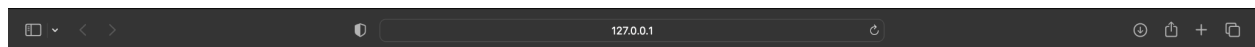
```

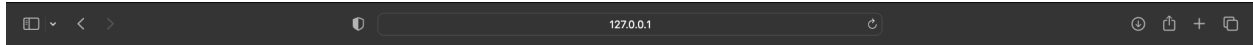
```

        .otherwise({ redirectTo: '/' });
    });

    app.controller('FirstController', function ($scope) {
        $scope.message = 'Hello from FirstController';
    });
    app.controller('SecondController', function ($scope) {
        $scope.message = 'Hello from SecondController';
    });
    app.controller('ThirdController', function ($scope) {
        $scope.message = 'Hello from ThirdController';
    });
</script>
</body>
</html>

```





**Jason**

**Single Page Application in AngularJS**

[First](#) [Second](#) [Third](#)

**Third Page**

**Contact me**

**Hello from ThirdController**