AIM: To create a HTML webpage containing HTML elements like head, body, paragraph, image, video, link and table

THEORY:

HTML: Hypertext markup language is used to design The web pages link between web pages is defined by hypertext. Text document inside the tag specifies the structure of web pages using a markup language. It employees tags to specify what text processing is required.

PARAGRAPH: HTML <p> tag is used to define it. A para always starts from a new line, and browser automatically adds space after and before it.

IMAGE: <img> tag is used to embed an image on webpage. Images aren't technically inserted into a web page, they are linked to it. The tag adds a blankspace for reference image.

TABLE: <table> tags allows to arrange data in rows & columns. Table headers can be added using <th> tag. Each table is defined using <td> and <tr> tag.

VIDEO: <frame> tag can be used to play videos from youtube. <video> tag is used to embed a video in html doc. It contains one or more <source> tags.

LIST: These allow to group related items in lists. An unordered list starts with <li> tag and an ordered list by <ol> tag.

HYPERLINK: <a> tag defines hyperlink. You ca click on link and jump to another element/doc. The most important aspect of the tag is <href> attribute which indicates the link's destination.

CODE:

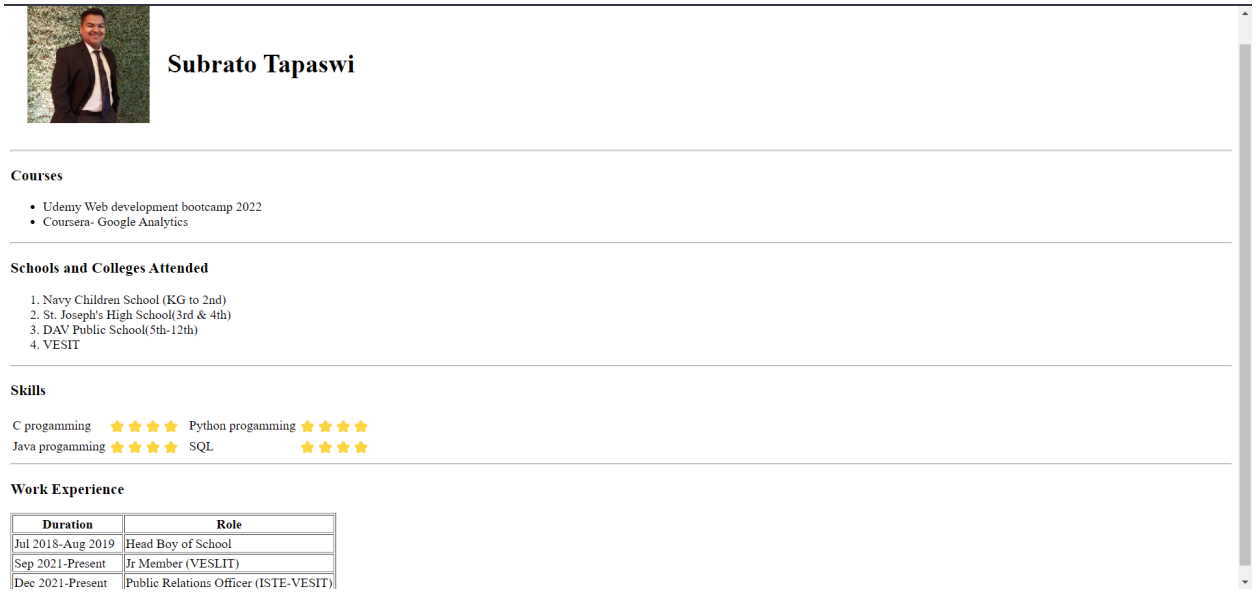
```
<!DOCTYPE html>
<html lang="en">

<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>Subrato Tapaswi</title>
```

```
    <link rel="stylesheet" href="styles/styles.css " />
    <!-- <link rel="icon" href="styles/favicon1.ico" /> -->
    <link rel="icon" href="styles/favicon.ico" />

</head>

<body>
    <div class="iintro">
        <table cellspacing="20">
            <tr>
                <td>
                    <img
src="https://pbs.twimg.com/profile_images/1538840700401332224/a01ToLgn_400x400.jpg"
                        width="150px" height="150px" alt="subrato">
                </td>
                <td>
                    <h1>Subrato Tapaswi</h1>
                </td>
            </tr>
        </table>
    </div>

    <hr>
    <h3>Courses</h3>
    <ul>
        <li>Udemy Web development bootcamp 2022</li>
        <li>Coursera- Google Analytics</li>
    </ul>
    <hr>
    <h3>Schools and Colleges Attended</h3>
    <ol>
        <li>Navy Children School (KG to 2nd)</li>
        <li>St. Joseph's High School(3rd & 4th)</li>
        <li>DAV Public School(5th-12th)</li>
        <li>VESIT</li>
    </ol>
    <hr>
    <h3>Skills</h3>
    <table>
        <tr>
```

```
            <td>C progamming</td>
            <td>⭐⭐⭐⭐  </td>

            <td>Python progamming</td>
            <td>⭐⭐⭐⭐</td>
        </tr>
        <tr>
            <td>Java progamming</td>
            <td>⭐⭐⭐⭐</td>
            <td>SQL</td>
            <td>⭐⭐⭐⭐</td>
        </tr>

    </table>
    <hr>
    <h3>Work Experience</h3>
    <table border="1" class="info">
        <thead>
            <tr>
                <th>Duration</th>
                <th>Role</th>
            </tr>
        </thead>
        <tbody>
            <tr>
                <td>Jul 2018-Aug 2019  </td>

                <td>Head Boy of School</td>
            </tr>
            <tr>
                <TD>Sep 2021-Present</TD>
                <td>Jr Member (VESLIT)</td>
            </tr>
            <tr>
                <TD>Dec 2021-Present</TD>
                <td>Public Relations Officer (ISTE-VESIT) </td>
            </tr>
        </tbody>
    </table>
</body>
```

```
</html>
```

OUTPUT:



Subrato Tapaswi

Courses

- Udemy Web development bootcamp 2022
- Coursera- Google Analytics

Schools and Colleges Attended

1. Navy Children School (KG to 2nd)
2. St. Joseph's High School(3rd & 4th)
3. DAV Public School(5th-12th)
4. VESIT

Skills

C progamming ★★★★ Python progamming ★★★★
Java progamming ★★★★ SQL ★★★★

Work Experience

Duration	Role
Jul 2018-Aug 2019	Head Boy of School
Sep 2021-Present	Jr Member (VESLIT)
Dec 2021-Present	Public Relations Officer (ISTE-VESIT)

CONCULSION:

We have successfully implemented a HTML web page with requirements like para, video, image, etc.

AIM:

A. To create a html page with css attributes using Colour, Background, Fonts, Tables,lists, CSS3 selectors, Pseudo classes, Pseudo elements.
B. To create a form /*w*eb page including css3 selectors pseudo elements, pseudo

THEORY:

CSS stands for cascading style sheets. CSS is the language we use to style an HTML document. CSS describes how HTML elements should be displayed. A CSS rule consists of a selector and a declaration block.

Fonts: Using this we add different fonts to our web page according to our requirement or basically for beautification purpose.

There are 5 generic font families:

1. Serif
2. Sans-serif
3. Monospace
4. Cursive
5. Fantasy Syntax:

   Eg . .p1{ Font-family: "serif";

Lists: It is same like that of the html lists, but here in css lists we can add style to it such as adding background colors to lists and list items, margin and padding.

Pseudo class: It is used to define a special state of an element. For example, it Can be used to: Style an element when a user mouses over it. Style visited and unvisited links differently.

For example, it can be used to:

- Style an element when a user mouses over it

  Style visited and unvisited links **differently**
- Style an element when it gets focus

Syntax:

```
selector:pseudo-class {
   property: value;
```

Pseudo element: A CSS pseudo-element is used to style specified parts of an element.(such as first line or first letter of a paragraph) Color: Using this we can set background color to page or set color to text.

A CSS pseudo-element is used to style specified parts of an element.
For example, it can be used to:
• Style the first letter, or line, of an element
• Insert content before, or after, the content of an element

SYNTAX:
selector:: pseudo-element {
property: value;

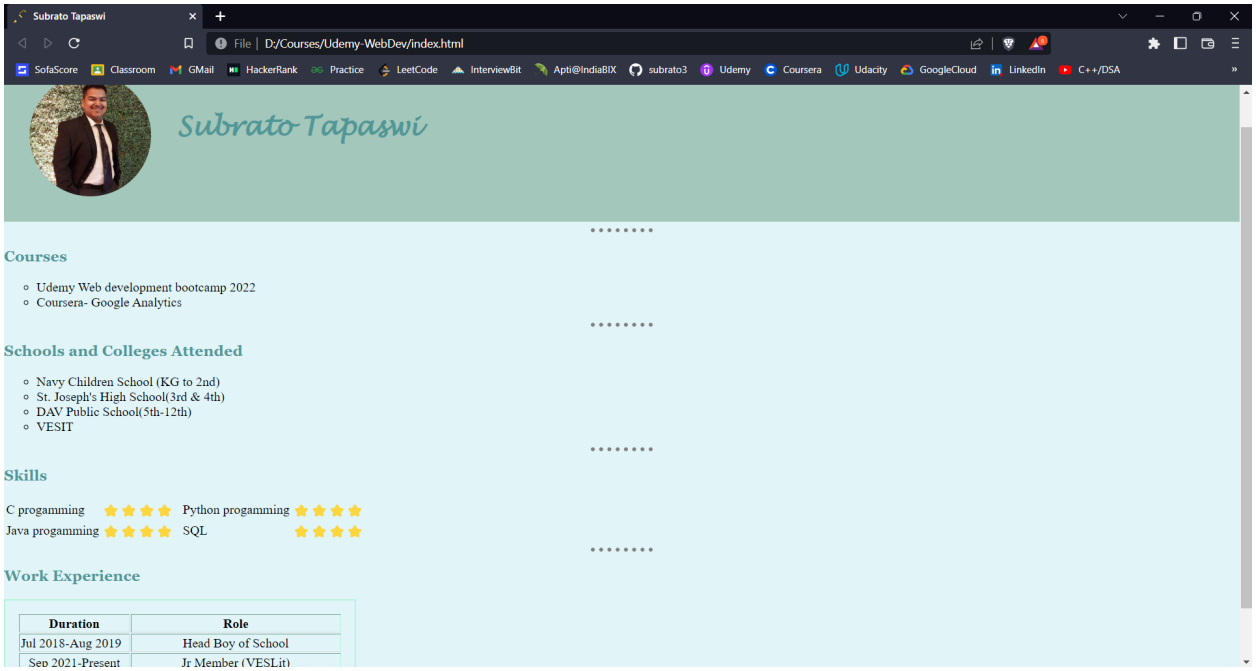
Syntax: <h1 style="background-color:blue;">This is my first page</h1>
There are different ways to color , it can be dome by simply mentioning the color, rgb color code system.

CODE: (styles.css file)

```
body {
   background-color : rgb(225, 245, 248);
   margin:auto;
   }
   hr {
   border-style: none ;
   border-top-style : dotted ;
   border-color : grey ;
   border-width : 5px ;
   width : 5% ;
   }
   img{
      border-radius: 50%;
      margin: 10px;
      float:left;
   }
   h1,h3{
      color:#509797;
   }
```

```
   h1{
      font-family: "lucida handwriting";
   }
   h3{
      font-family:"georgia";
   }

div {
background-color: rgb(164, 199, 187);
text-align:center;
}

li{
   list-style-type: circle;
}

.intro{
   align-items:center;
}

body{
   margin:0;
}
h1{
   margin-top: 0;
}
.info{
   padding: 15px;
   border: 1.5px solid rgb(181, 244, 217);
   border-width: 2px;
   text-align: center;
   border-bottom: 2px solid #ddd;
}
.info:hover{
   background-color: rgb(95, 145, 145);
}
```

OUTPUT:



CONCULSION:

We have successfully implemented a HTML web page with requirements like para, video, image, etc.

AIM:

A. To use javascript and code the following : Fibonacci series, factors and objects.
B. Use javascript to validate the form which includes name validation , email validation, phone number validation , password validation.

THEORY:

A. Javascript is used to program the behavior of web pages. JavaScript (js) is a light-weight object-oriented programming language which is used by several websites for scripting the webpages. It is an interpreted, full-fledged programming language that enables dynamic interactivity on websites when applied to an HTML document.
With JavaScript, users can build modern web applications to interact directly without reloading the page every time. The traditional website uses js to provide several forms of interactivity and simplicity.

B. It is important to validate the form submitted by the user because it can have inappropriate values. So, validation is a must to authenticate users.

JavaScript provides the facility to validate the form on the client side data processing will be faster than server-side validation. Most web developers prefer JavaScript form validation. Through JavaScript, we can validate name, password, email, date, mobile numbers and more fields.

Form validation can happen on the client side and the server side. Client side validation occurs using HTML5 attributes and client side JavaScript.

Here are some common validation cases:

- Making fields required using required
- Constraining the length of data:
- minlength, maxlength: for text data

Syntax: to create variables

Var x; // value can be changed

Let y;

Const z; // value cannot be changed after declaring

Comments: we have single line comments (//this is a comment) and

multi line comments (/*this is comment*/)

Data types: primitive data types – they are the pre defined data types in javascript.

Numbers, strings, Boolean, null,undefined, symbol, NaN

Non primitive data types – they are defined by the users

Objects , arrays

Loops: there are 3 types of loops :

- For loop

Syntax: for(initialisation; condition; incrementation)

- While loop

Syntax: while (condition)

- For of loop- loops through the values of an iterable object.

Syntax: For(variable of iterable)

A. CODE & OUTPUT:

1. Fibonacci:

```
let x=0;
let y=1;
let z;
function fib(n){
   if (n>1)
   {
      z=x+y;
      x=y;
      y=z;
      console.log(z)
      fib(n-1)
   }
}
console.log(0);
console.log(1);
console.log(fib(8));
```

```
node /tmp/AKeeuXv2LR.js
0
1
1
2
3
5
8
13
21
```

2. **Factors:**

```
function factor(n){
   for (let i=1;i<=n;i++)
   {
      if (n%i==0)
      {
         console.log(i)
      }
   }
}
factor(15)
```

```
node /tmp/AKeeuXv2LR.js
1
3
5
15
```

3. **Objects**

```
const ves={
   name:'VESIT',
   address:'Chembur',
```

```
    phone: 8355943385,
    types:['Engineering', 'Architecture'],
    greet:function()
    {
       console.log("welcome to "+ this.name);
    },
    branch:
    {
       name:'AIDS',
       strength: 69
    }
}
console.log('type of object is: '+typeof ves)
console.log(ves.greet)
console.log(ves)
```

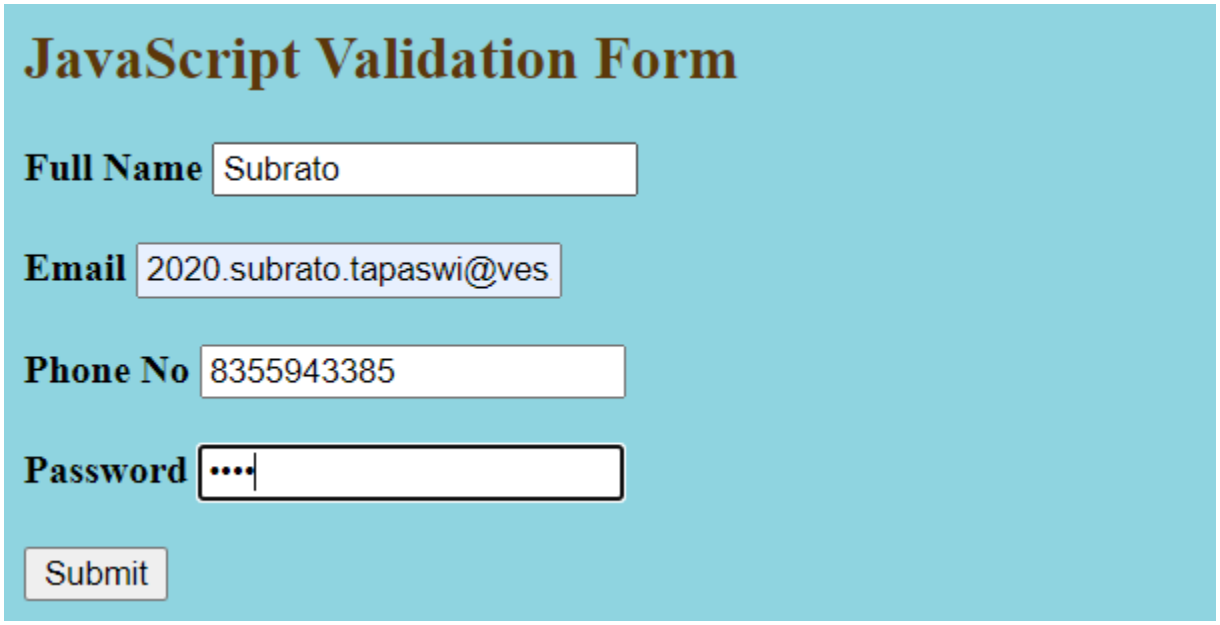
```
node /tmp/AKeeuXv2LR.js
type of object is: object
[Function: greet]
{ name: 'VESIT',
  address: 'Chembur',
  phone: 8355943385,
  types: [ 'Engineering', 'Architecture' ],
  greet: [Function: greet],
  branch: { name: 'AIDS', strength: 69 } }
```

B. VALIDATION FORM:

```
<!DOCTYPE html>
<html>

<head>
  <script>
    function validateForm(inputText) {
      let x = document.forms["myForm"]["fname"].value; if (x.length < 2) {
        alert("Enter a Valid Name");
        return false;
      }
```

```
        let y = document.forms["myForm"]["email"].value; if (y == "") {
            alert(" Enter a Valid E-mail");
            return false;
        }
        let z = document.forms["myForm"]["phone"].value; if (z.length < 10) {
            alert("Enter a valid Phone Number");
            return false;
        }
        let p = document.forms["myForm"]["pwd"].value;
        if (p.length < 6) {
            alert("enter a valid Password");
            return false;
        }
    }
  </script>
</head>

<body>
  <h2>JavaScript Validation Form</h2>
  <form name="myForm" action="/action_page.php" onsubmit="return validateForm()"
method="post" class="form"> <label
        for="name"><b>Full Name</b></label>
    <input type="text" placeholder="Enter Full Name" name="fname" id="fname"
required><br><br>
    <label for="email"><b>Email</b></label>
    <input type="text" placeholder="Enter Email" name="email" id="email"
required><br><br>
    <label for="phone no"><b>Phone No</b></label>
    <input type="number" maxlength="10" placeholder="enter contact number"
name="phone number" id="phone"
        required><br><br>
    <label for="password"><b>Password</b></label>
    <input type="password" placeholder="Enter the password" pwd="password"
id="pwd" required><br><br>
    <input type="submit" value="Submit">
  </form>
```

```
    <body style="background-color:#8fd4e0;"></body>
    <style>
        h2 {
            color: #5c3709;
        }

        b {
            color: rgb(0, 0, 0)
        }
    </style>
</body>

</html>
```



CONCLUSION: Hence we have studied the basics of javascript such as data types, loops, var, let and form validation and implemented the above questions.

Aim:

A. To install and configure React and use basic props, components and JSX
B. To understand Basic State and Events in React
C. To develop a single page application in react using Router.
D. To create Controlled and Uncontrolled forms using react js.
E. To create a simple interactive UI using Material UI

Theory:

React Js – ReactJS is a declarative, efficient, and flexible JavaScript library for building reusable UI components. It is an open-source, componentbased front-end library which is responsible only for the view layer of the application. It was initially developed and maintained by Facebook and later used in its products like WhatsApp & Instagram.

The main objective of ReactJS is to develop User Interfaces (UI) that improves the speed of the apps. It uses virtual DOM (JavaScript object), which improves the performance of the app. The JavaScript virtual DOM is faster than the regular DOM. We can use ReactJS on the client and server-side as well as with other frameworks. It uses component and data patterns that improve readability and helps to maintain larger apps

Components –
A Component is one of the core building blocks of React. In other words, we can say that every application you will develop in React will be made up of pieces called components. Components make the task of building UIs much easier. You can see a UI broken down into multiple individual pieces called components and work on them independently and merge them all in a parent component which will be your final UI.

Props –
Props are a type of object where the value of attributes of a tag is stored. The word "props" implies "properties", and its working functionality is quite similar to HTML attributes. Basically, these props components are read-only components. In ReactJS, the data can be passed from one component to another component using these props, similar to how the arguments are passed in a function. Inside the component, we can add the attributes called props; however, we cannot change or modify props inside the component as they are immutable

Component State –
React components has a built-in state object. The state object is where you store property values that belongs to the component. When the state object changes, the component re-renders.

Difference between props and state is props get passed to the component (similar to

function parameters) whereas state is managed within the component (similar to variables declared within a function).

Events –
Just like HTML DOM events, react can perform actions based on user events. React has the same events as HTML: click, change, mouseover etc. React events are written in camelCase and their event handlers are written inside curly braces.

Router –
React Router is a standard library for routing in React. It enables the navigation among views of various components in a React Application, allows changing the browser URL, and keeps the UI in sync with the URL.

1) Installing React Router: React Router can be installed via npm in your React application. To install the React Router use the following command in ur project directory: npm install – -save react-router-dom or npm i react-router-dom
2) Components of React Router: The Components of React Router Are as follows: i. : It is used for handling the dynamic URL. ii. : It is used for handling the static request.

Forms In React –

Forms are an integral part of any modern web application. It allows the users to interact with the application as well as gather information from the users. React offers a stateful, reactive approach to build a form. The component rather than the DOM usually handles the React form. In React, the form is usually implemented by using controlled components.

There are mainly two types of form input in React.

1. Uncontrolled Forms: It is similar to the traditional HTML form inputs. Here, the form data is handled by the DOM itself. It maintains their own state and will be updated when the input value changes.
2. Controlled Forms: A controlled component is bound to a value, and its changes will be handled in code by using event-based callbacks. Here, the input form element is handled by the react itself rather than the DOM. Controlled components have functions that govern the data passing into them on every onChange event occurs. This data is then saved to state and updated with setState() method. It makes component have better control over the form elements and data.

Material UI –

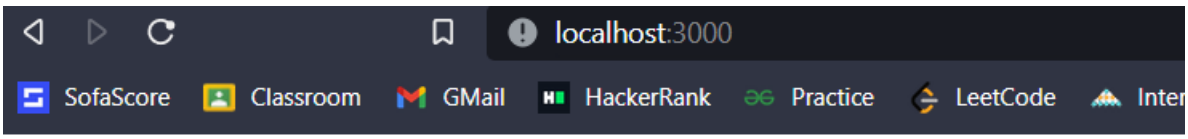
Material-UI (MUI) is a CSS framework that provides React Components out-of the-

box. MUI makes it possible to use different components to create a UI for a company's web and mobile apps. Google uses Material Design to guarantee that no matter how users interact with the products they use, they will have a consistent experience.

Output:

4A

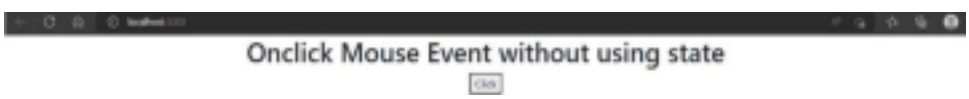
```
function App() {
  return (
    <div className="App">
      <header className="App-header">

        <p>
         <h1> Hello !! This is Basic React App. </h1>
        </p>
      </header>
    </div>
  );
}

function Welcome(props) {
  return <h1>Hello, {props.name}</h1>;
}

export default App;
```

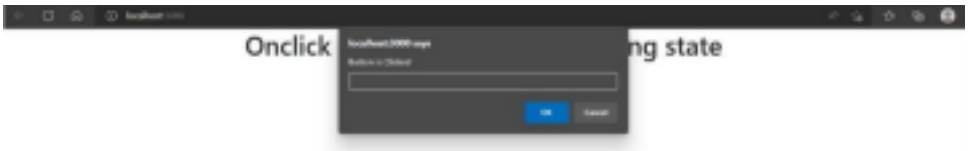


Hello !! This is Basic React App.

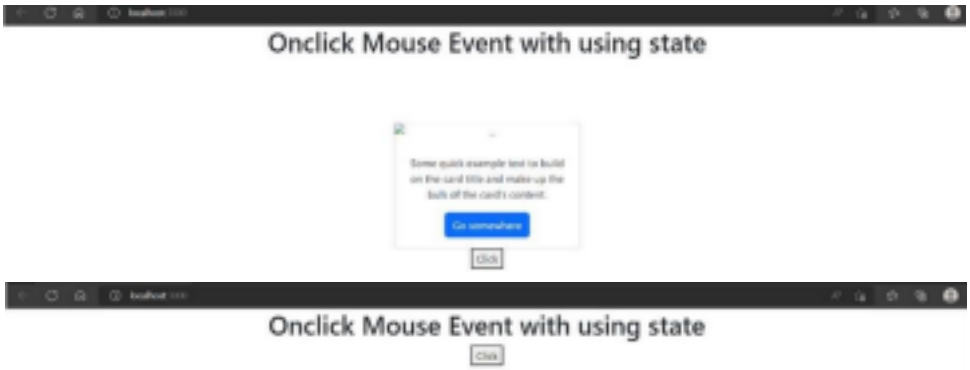
4B

Using Onclick Mouse Event without using state:





Using Onclick Mouse Event along with state:



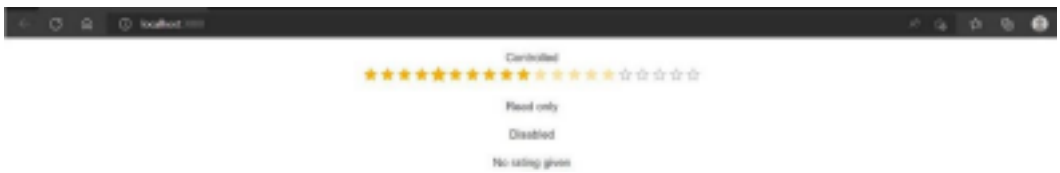
4C

4D

Controlled Form:

Uncontrolled Form:

4E



Conclusion:

Thus we,

a. Installed and configured React Js and implemented and learnt about basic props, components.
b. Learnt about Basic States and Events in React.
c. Developed a single page application in react using Router.
d. Learnt about Controlled and Uncontrolled forms using React JS.
e. Created a simple interactive UI using Material UI.

Aim:

a.
   1. Installation and Configuration of Node Js.
   2. Learning about Call Backs and Event Loops in Node Js

b. Creating an Express Application.

Theory:

NODE:

Node.js (Node) is an open-source development platform for executing JavaScript code server-side. Node is useful for developing applications that require a persistent connection from the browser to the server and is often used for real-time applications such as chat, news feeds and web push notifications.

Node.js is intended to run on a dedicated HTTP server and to employ a single thread with one process at a time. Node.js applications are event based and run asynchronously. Code built on the Node platform does not follow the traditional model of receive, process, send, wait, receive. Instead, Node processes incoming requests in a constant event stack and sends small requests one after the other without waiting for responses. It interprets JavaScript code via Google's V8 JavaScript engine. It interprets JavaScript code via Google's V8 JavaScript engine.

Steps To Install Node Js:

1. Download the Node.js '.msi' installer from: https://nodejs.org/en/download/
2. Run the Node.js installer. Install it. Verify By typing 'node -v' in CMD

Event Loops –
The event loop is what allows Node.js to perform non-blocking I/O operations (despite the fact that JavaScript is single-threaded) by offloading operations to the system kernel whenever possible.

Callback –
A callback is a function which is called when a task is completed, thus helps in preventing any kind of blocking and a callback function allows other code to run in the meantime. Callback is called when task get completed and is asynchronous equivalent for a function. Using Callback concept, Node.js can process a large number of requests without waiting for any function to return the result which makes Node.js highly scalable.

EXPRESS:

It's a web framework that lets you structure a web application to handle multiple different http requests at a specific URL. Express is a minimal, open source and

flexible Node.js web app framework designed to make developing websites; web apps, & APIs much easier.

- Express helps to respond to requests with route support so that you may write responses to specific URLs
- It Supports multiple templating engines to simplify generating HTML

Step to Install Express:

1. Install Node JS first. Check For npm package.
2. Open Vs Code & in terminal type ‘npm install express ‘
3. Express will be installed on the respective device

Code & Output:

5A

HelloWorld.js:

```
setTimeout(()=>{console.log('Força Barça')},0);

console.log('Hola!');
```

```
Hola!
Força Barça
```

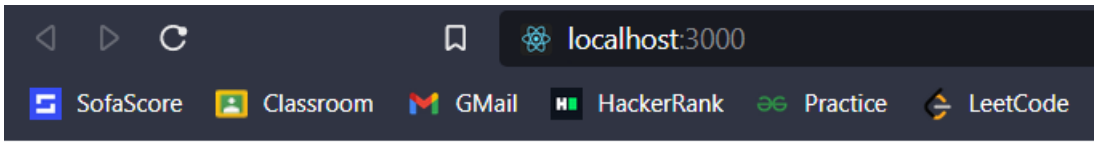
CallBack.js:

```
cal = function (x, y) {
  console.log("x=" + x, "y=" + y);
  let sum = add(x, y, function (result) {
    console.log("sum= " + result);
    console.log("End");
  });
};
add = function (x, y, callback) {
  setTimeout(function () {
    console.log("Addition Performed after 100ms");
    callback(x + y);
  }, 100);
};
cal(3, 6);
```

```
x=3 y=6
Addition Performed after 100ms
sum= 9
End
```

5B

```
const express = require('express')
const app = express()
const port = 3000

app.get('/', (req, res) => {
  res.send('<h1>Hello Subrato!</h1>')
})

app.listen(port, () => {
  console.log(`Example app listening on port ${3000}`)
})
```



Hello Subrato!

Conclusion:

Thus, we learnt:

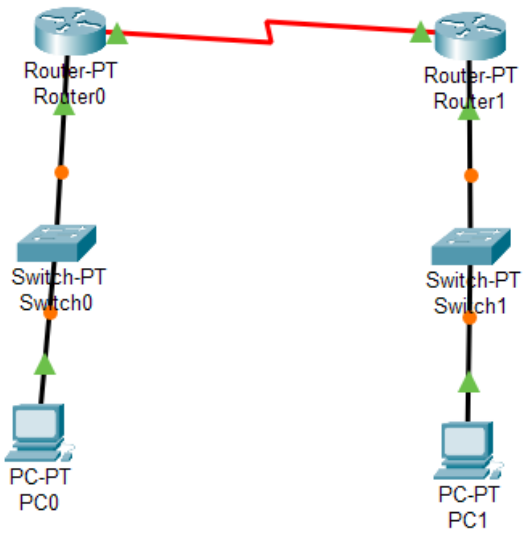
a. To Install and Configure Node Js.
b. About Event Loops and Call Backs in Node Js.
c. To Install and Configure Express.
d. To Create Express Application

Aim: To design and simulate the environment for Dynamic routing using **Cisco** packet tracer/ GNS3

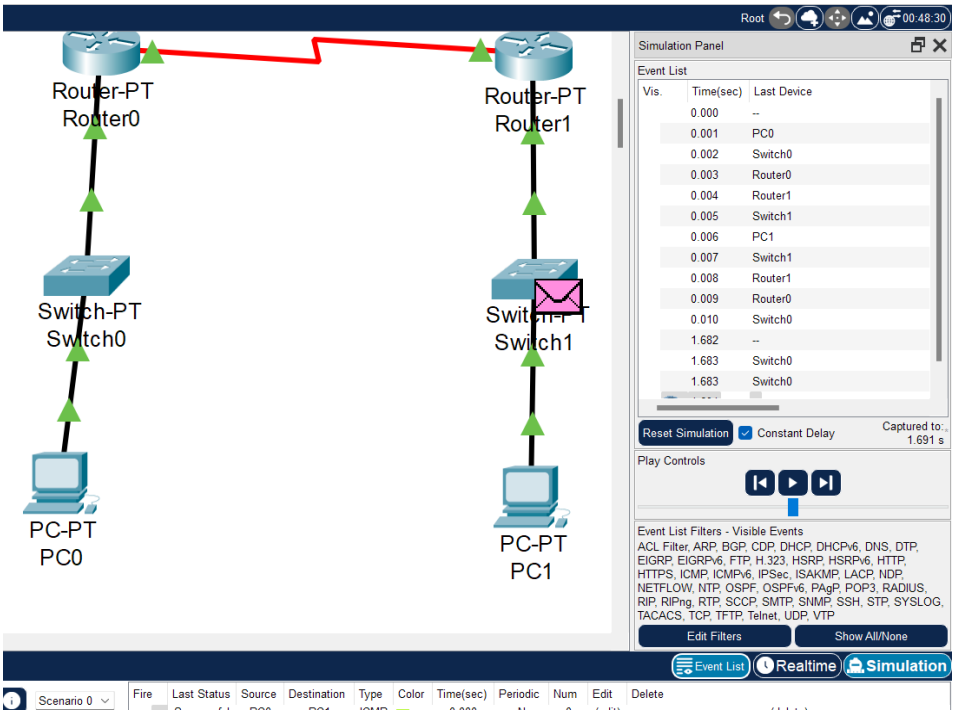
Theory:

1. Packet Tracer is a cross-platform visual simulation tool designed by Cisco Systems that allows users to create network topologies and imitate modern computer networks.
2. The software allows users to simulate the configuration of Cisco routers and switches using a simulated command line interface.
3. Packet Tracer makes use of a drag and drop user interface, allowing users to add and remove simulated network devices as they see fit.
4. Packet Tracer can be run on Linux, Microsoft Windows, and macOS. Similar Android and iOS apps are also available.
5. Packet Tracer allows users to create simulated network topologies by dragging and dropping routers, switches and various other types of network devices.
6. A physical connection between devices is represented by a 'cable' item.
7. Packet Tracer supports an array of simulated Application Layer protocols, as well as basic routing with RIP, OSPF, EIGRP, BGP etc.
8. Packet Tracer allows students to design complex and large networks, which is often not feasible with physical hardware, due to costs

Output:



Sending a Packet



Conclusion: Dynamic routing has been implemented using Cisco packet tracer

Aim: To design and Simulate VLANs on the switch/router using Cisco packet tracer/ GNS3

Theory:

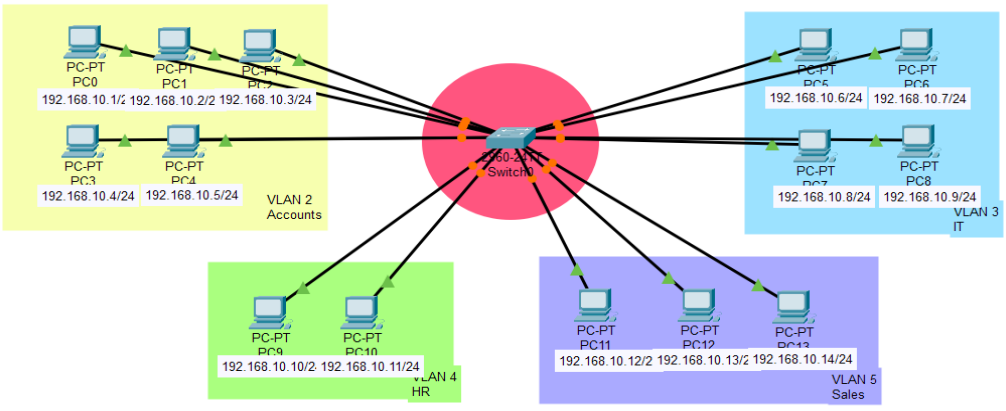
VLANs (Virtual LANs) are logical grouping of devices in the same broadcast domain. VLANs are usually configured on switches by placing some interfaces into one broadcast domain and some interfaces into another. Each VLAN acts as a subgroup of the switch ports in an Ethernet LAN.

VLANs can spread across multiple switches, with each VLAN being treated as its own subnet or broadcast domain. This means that frames broadcasted onto the network will be switched only between the ports within the same VLAN.

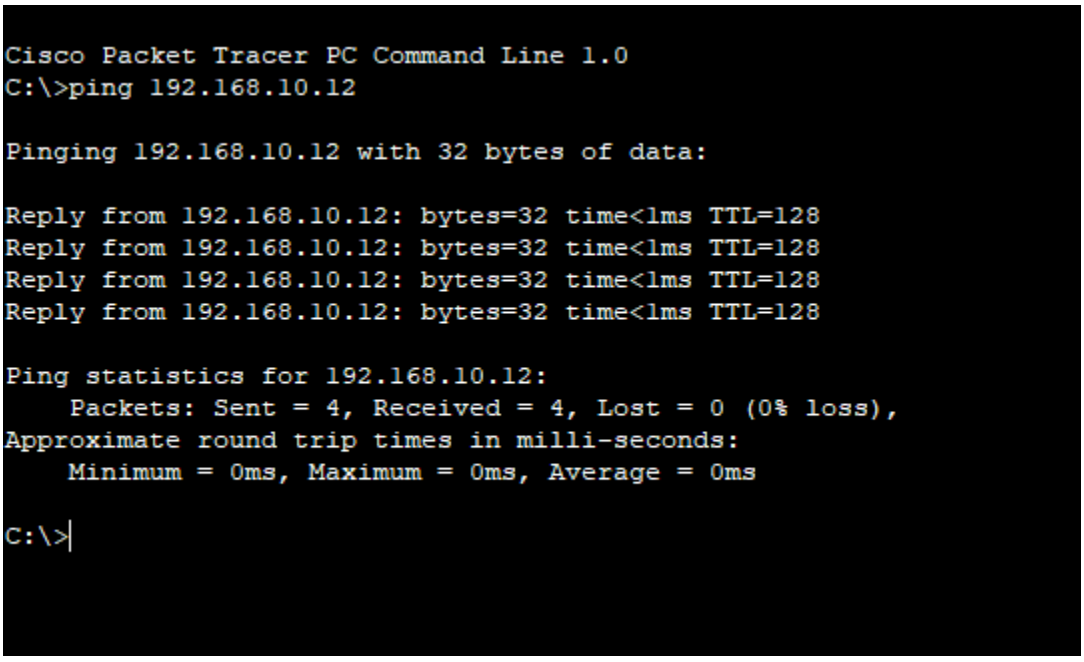
A VLAN acts like a physical LAN, but it allows hosts to be grouped together in the same broadcast domain even if they are not connected to the same switch. Here are the main reasons why VLANs are used:

- VLANs increase the number of broadcast domains while decreasing their size.
- VLANs reduce security risks by reducing the number of hosts that receive copies of frames that the switches flood.
- you can keep hosts that hold sensitive data on a separate VLAN to improve security.
- you can create more flexible network designs that group users by department instead of by physical location.
- network changes are achieved with ease by just configuring a port into the appropriate VLAN.

Output:



Pinging from PC10 to PC11



Conclusion: VLANs have been simulated using Cisco packet tracer

Aim: To Design and Simulate NAT on router using Cisco Packet Tracer.

Software Used: Cisco Packet Tracer

Requirements: 2 Routers, 2 Switches, 1 server, 2 Pcs

Theory:

NAT stands for network address translation. It's a way to map multiple local private addresses to a public one before transferring the information.Organizations that want multiple devices to employ a single IP address use NAT, as do most home routers.

A NAT works by selecting gateways that sit between two local networks: the internal network, and the outside network. Systems on the inside network are typically assigned IP addresses that cannot be routed to external networks (e.g., networks in the 10.0.0.0/8 block). A few externally valid IP addresses are assigned to the gateway. The gateway makesoutbound traffic from an inside system appear to be coming from one ofthe valid external addresses. It takes incoming traffic aimed at a valid external address and sends it to the correct internal system. This helps ensure security. Because each outgoing or incoming request must go through a translation process that offers the opportunity to qualify or authenticate incoming streams and match them to outgoing requests, for example. NAT conserves the number of globally valid IP addresses a company needs.

Procedure:

1. Take 2 Pt-Routers, 2 PCS, 1 Server and 1 switch. Arrange them inthe following configuration. (As per output)
2. Connect the 2 Router using Serial DCE cable. Connect the PCS,Switches and Server using Copper Straight Through Cables.
3. Assign Ip Addresses to Pcs, Server and router.
4. Open Pt Router 0, go to CLI mode and type the following:

```
Router(config)#ip nat inside source static 10.10.10.2 50.50.50.2
Router>enable                                                    3
Router#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#ip nat inside source static 20.20.20.2 60.60.60.2
Router(config)#interface fastEthernet 0/0
Router(config-if)#ip nat inside
Router(config-if)#exit
Router(config)#interface serial 2/0
Router(config-if)#ip nat outside
Router(config-if)#exit
Router(config)# ip route 50.0.0.0 255.0.0.0 192.162.10.1
Router(config)#exit
Router#
%SYS-5-CONFIG_I: Configured from console by console
```

5. Do the same in another router.

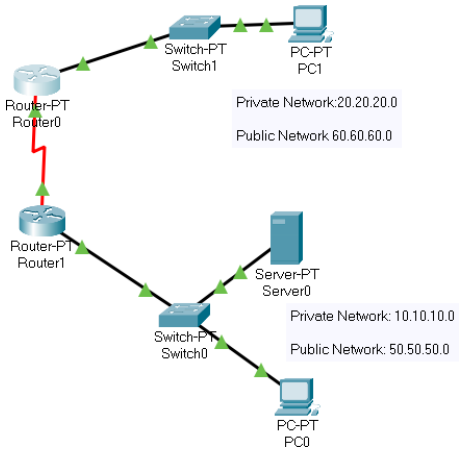
```
Router(config)#ip route 60.0.0.0 255.0.0.0 192.162.10.2
Router(config)#exit
Router#
%SYS-5-CONFIG_I: Configured from console by console
```

6. Check Your Configuration using following command:(For Both Routers)

```
Router#show ip route
```

Output:

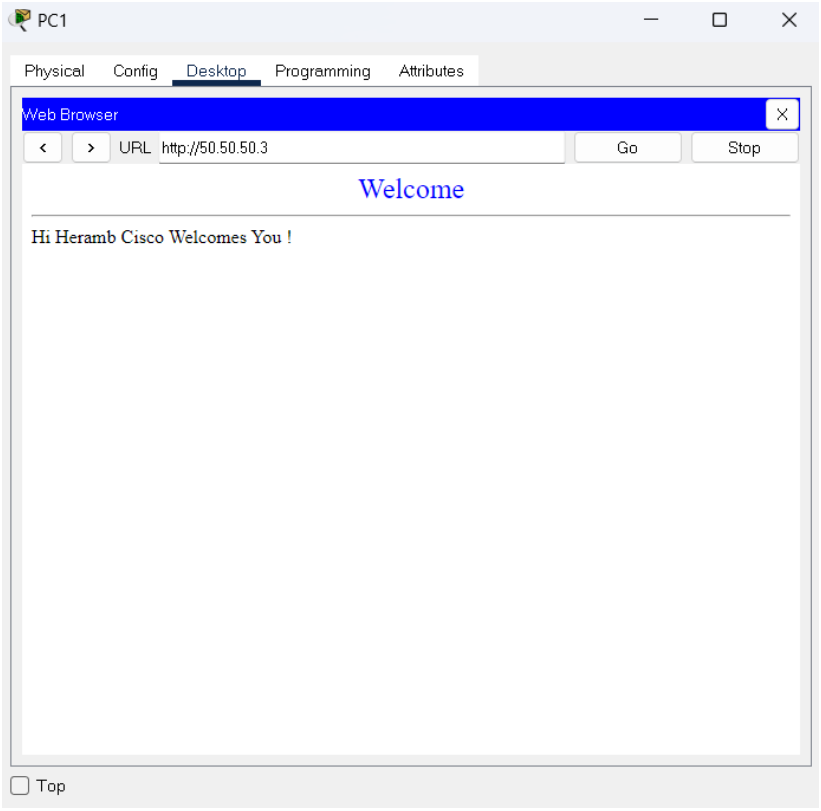
1. Network Design



2. Pinging To Pc 1 From Pc 0 using Public as well as Private IP address

```
Packet Tracer PC Command Line 1.0
C:\>ping 60.60.60.2

Pinging 60.60.60.2 with 32 bytes of data:

Request timed out.
Reply from 60.60.60.2: bytes=32 time=2ms TTL=126
Reply from 60.60.60.2: bytes=32 time=5ms TTL=126
Reply from 60.60.60.2: bytes=32 time=5ms TTL=126

Ping statistics for 60.60.60.2:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 5ms, Average = 4ms

C:\>ping 20.20.20.2

Pinging 20.20.20.2 with 32 bytes of data:

Reply from 10.10.10.1: Destination host unreachable.
Request timed out.
Reply from 10.10.10.1: Destination host unreachable.
Reply from 10.10.10.1: Destination host unreachable.

Ping statistics for 20.20.20.2:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

C:\>
```

3. Accessing server through Pc 1.



Conclusion:

Thus, we successfully designed and simulated NAT on router using ciscopack.