Exp.Name..... Date :.....

Write an angularJs program to binding and perform operations using ng-bind

de : Index.html

```
<html>
<head> <title> ng-bind Directive </title>
        <script src="https://ajax.googleaps.com/ajax/libs/angularj
            /1.6.9/angular.min.js"> </script>
</head>
<body ng-app="gfg" style="text-align:center">
<h1 style="color:blue"> product </h1>
<h2> ng-bind Directive </h2>
<div ng-controller="app">
  number1: <input type="number" ng-model="num1" ng-change=
                "product()"/> </br>
  number2: <input type="number" ng-model="num2"
                ng-change="product()"/> </br>
  <b> Product : </b> <span ng-bind="result"> </span>
</div>
<script>
  var app = angular.module("gfg",[]);
  app.controller('app',['$scope', function($app){
            $app.num1=1;
            $app.num2=1;
```

Exp Name ..... Date .....

```
$app.product = function() {
    $app.result = ($app.num1 * $app.num2);
  }
  }]);
</script>
</br>
</body>
</html>
```

Aim: Write an angularjs program using angularjs directory

Source code:

```
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/angula
1.5.2/angular.min.js/"></script>
</head>
<body ng-app="myapp">
<div ng-controller="HelloController">
<h2>Welcome to {{helloTo.title}} to PACE CSE student <
<script>
angular.module("myapp",[])
.controller("HelloController", function($scope)
{
$scope.helloTo={};
$scope.helloTo.title={"Angular.js"};
});
</script>
</body>
</html>
```

No. 07 Exp.Name ..... Date :

Aim : Create a table and display content of table by using angularjs.

Source code:

```
<html ng-app="myapp">
<title> Student details </title>
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/
    1.5.2/angular.min.js"> </script>
<script>
  var myapp = angular.module("myapp",[]);
    myapp.filter("DisplayPageData", function() {
        return function (import,start) {
            start = start
            return inputslice(start);
        }
    });
  myapp.controller("myappcont", function ($scope) {
      $scope.currentpage = 0,
      $scope.student = [
  { Name: "Akash", Gender: "Male", class: "1std" },
  { Name: "Bablu", Gender: "Male", class: "2std" },
  ]);
</script>
<body ng-controller="myappcont">
<table binder="1" style="width: 80%">
```

```
<caption> Student names list </caption>
<thead>
<tr><th> Name </th>
    <th> Gender </th>
    <th> class </th>
</tr>
</t head>
</t body>
<tr ng-script = "student in students">
  <td> {{ student.Name }} </td>
  <td> {{ student.Gender }} </td>
  <td> {{ student.class }} </td>
</tr>
</t body>
</body>
</html>
```

Exp No 05 Exp Name Date

Aim: Write a program create a form and validation of form on submit using angularjs.

Source code:

```
<html>
<head> <title> Event Registration </title> </head>
<body ng-app=" SampApp ">
 <script src = "https://code.angularjs.org/1.6.9/angular-route.js"> </script
 <script src = "https://code.angularjs.org/1.6.9/angular-min.js "> </script
 <script src = "https://code.angularjs.org/1.6.9/angular.js"> </script>
 <script src = "lib/bootstrap.js"></script>
 <script src = "lib/bootstrap.css"></script>
 <h1> Form Validation </h1>
 <div>g-controller = " Angular controller">
 <form>
  Enter Name : <input type = " text" name="name" required> </br>
  Enter password : <input type = "password"> </br>
  Enter email : <input type = " email" > </br>
  Enter Age : <input type = " number" > </br>
  <input type = " submit" value = "submit" >
 </form>
 </div>
 </body>
</html>
```

Aim: To write a Node.js program to create a Http Server using Http Module.

ource code:

```
Const http = require ('http');
const server = http.createServer ((req, res) =>
{
    if (req.url == '/') {
     res.write ('<h1> Hello Node.js ! </h1>');
    }
res.end();
});
server.listen (5000);
console.log ('The Http Server is running on port 5000');
```

The following starts the Http Server:

```
node server.js
```

07 Exp.Name Date

Aim: To write a node.js program to perform operations on files
(a) Read files (b) Create files (c) Update files (d) Delete files (e) Rename files.
(a) Create a .js file to open input.txt for reaching and writing

Source code:

```
var fs = require("fs");
console.log("Going to open file!");
fs.open('input.txt', 'r+', function(err, fd)
{
  if(err) {
    return console.error(err);
  }
}
console.log("file opened Sucessfully!");
});
```

ing Today's students to address Tomorrow's world....

(b) To write data into file :

source code :

```
var fs = require('fs');
console.log("going to write into existing file");
fs.writeFile('input.txt', 'simply Easy learning!',(err) {
   if(err) {
      return console.error(err);
   }
   console.log("Data written sucessfully!");
   console.log("let's read newly written data");
   fs.readFile('input.txt', function (err, data)
   {
      if(err) {
         return console.error(err);
      }
      console.log("Asynchronous read:"+data.toString());
   });
});
```

(c) To read data from a file

Source code:

```
var fs = require("fs");
var buf = new Buffer(1024);
console.log("Going to open an existing file");
fs.open("input.txt", 'r+', function(err, fd) {
   if (err) {
      return console.error(err);
   }
   console.log("file opened sucessfully!");
   console.log("Going to read the file");
   fs.read(fd, buf, 0, buf.length, 0, function(err, bytes)
      if (err) {
         console.log(err);
      }
      console.log(bytes + " bytes read");
      if (bytes > 0) {
         console.log(buf.slice(0, bytes).toString());
      }
   });
});
```

(d) To delete a file

Source code:

```
var fs = require("fs");
console.log("Going to delete an existing file");
fs.unlink('input.txt', function (err) {
    if (err) {
        return console.error(err);
    }
    console.log("file deleted Sucessfully!");
});
```

No. 08 Exp Name Date

Aim:

To write a blockchain application in JavaScript for the Sample transaction

Source code:

Step1: Create a project and create a Genesis data

The first step is to create a project:

```
mkdir cryptochain
cd cryptochain
npm init -y
// genesis.js
const GENESIS_DATA = {
    timestamp: Data.now(),
    LastHash: '64b7edc786326632700db0e143f',
    Hash: 'c67c81b92b9fd65007d38467',
    data: 'krunal', };
module.exports = { GENESIS_DATA };
```

Step2: Create a Block

```
const { GENESIS_DATA } = require('./genesis.js');
class Block {
    constructor ({ timestamp, lastHash, hash, data }) {
        this.timestamp = timestamp;
        this.time[illegible]Hash = lastHash;
        this.hash = hash;
```

```
        this.data = data;
    }
    static genesis() {
        return new this(GENESIS_DATA);
    }
}
module.exports = Block;
```

Step3: Create a hash based on a previous block.

```
// crypto-hash.js
const crypto = require('crypto');
const cryptoHash = (...inputs) => {
    const hash = crypto.createHash('sha256');
    hash.update(inputs.sort().join(' '));
    return hash.digest('hex');
}
module.exports = cryptoHash;
```

Step4: Mine a new Block based on a previous block

```
const { GENESIS_DATA } = require('./genesis.js');
const cryptoHash = require('./crypto-hash');
class Block {
    constructor({ timestamp, lastHash, hash, data }) {
        this.timestamp = timestamp;
        this.hash = hash;
```

```
        this.data = data;
    }
    static genesis() {
        return new this(GENESIS_DATA);
    }
    static mineBlock({ lastBlock, data }) {
        const timestamp = Date.now();
        const lastHash = lastBlock.hash();
        return new this({
            timestamp,
            lastHash,
            data,
            hash: cryptoHash(timestamp, lastHash, data)
        });
    }
}

module.exports = Block;
```

Exp No. ..... Exp Name ..... Date .....

Step 5: Create a Blockchain

```
const Block = require('./block');
class Blockchain {
    constructor() {
        this.chain = [Block.genesis()];
    }
    addBlock({ data }) {
        const newBlock = Block.mineBlock({
            lastBlock: this.chain[this.chain.length-1],
            data
        });
        this.chain.push(newBlock);
    }
}
module.exports = Blockchain;
```


Exp.No Exp.Name Date

Step 6: Run the project and get the blockchain.

```
//server.js
const Blockchain = require('./blockchain');
const Block = require('./block');
const blockchain = new Blockchain();
for (let i=0; i<5; i++) {
    const newData = 'krunal' + i;
    blockchain.addBlock({data: newData})
}
console.log(blockchain);
```

Go to the terminal and start the node server:

```
node server
```

Then your output will be generated.