Slip 1A

Write an AngularJS script for addition of two numbers using ng-init, ng-model & ng-bind. And also demonstrate ng-show, ng-disabled, ng-click directives on button component.

```
<!DOCTYPE html>
<html ng-app>
    <head>
        <title>Demo</title>
        <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
        <!-- <script src="angular.min.js"></script> Local path to AngularJS -->
    </head>
    <body ng-init="n1=null; n2=null; show=false;">
        <input type="number" ng-model="n1">
        <input type="number" ng-model="n2">
        <button ng-click="result = n1+n2; show=true;" ng-disabled="!n1||!n2"
>Calculate</button>
        <div ng-show="show">Result <span ng-bind="result"></span></div>
    </body>
</html>
```

Slip 1B 7A

Create a Node.js application that reads data from multiple files asynchronously using promises and async/await.

```
//Create a Node.js application that reads data from multiple files asynchronously
//using promises and async/await.

const fs = require('fs').promises;

async function readFiles(files){
    try{
        const fileread = files.map(file => fs.readFile(file,'utf-8'));
        const contents = await Promise.all(fileread);

        console.log(contents);

        // contents.forEach((content, index) => {
        //     console.log(`Content of ${files[index]}:\n${content}\n`);
        // });
    }
    catch (error){
        console.error(error);
    }
}
```

```
const f=['a2.txt','a1.txt']
readFiles(f)
```

Slip 2A 12A

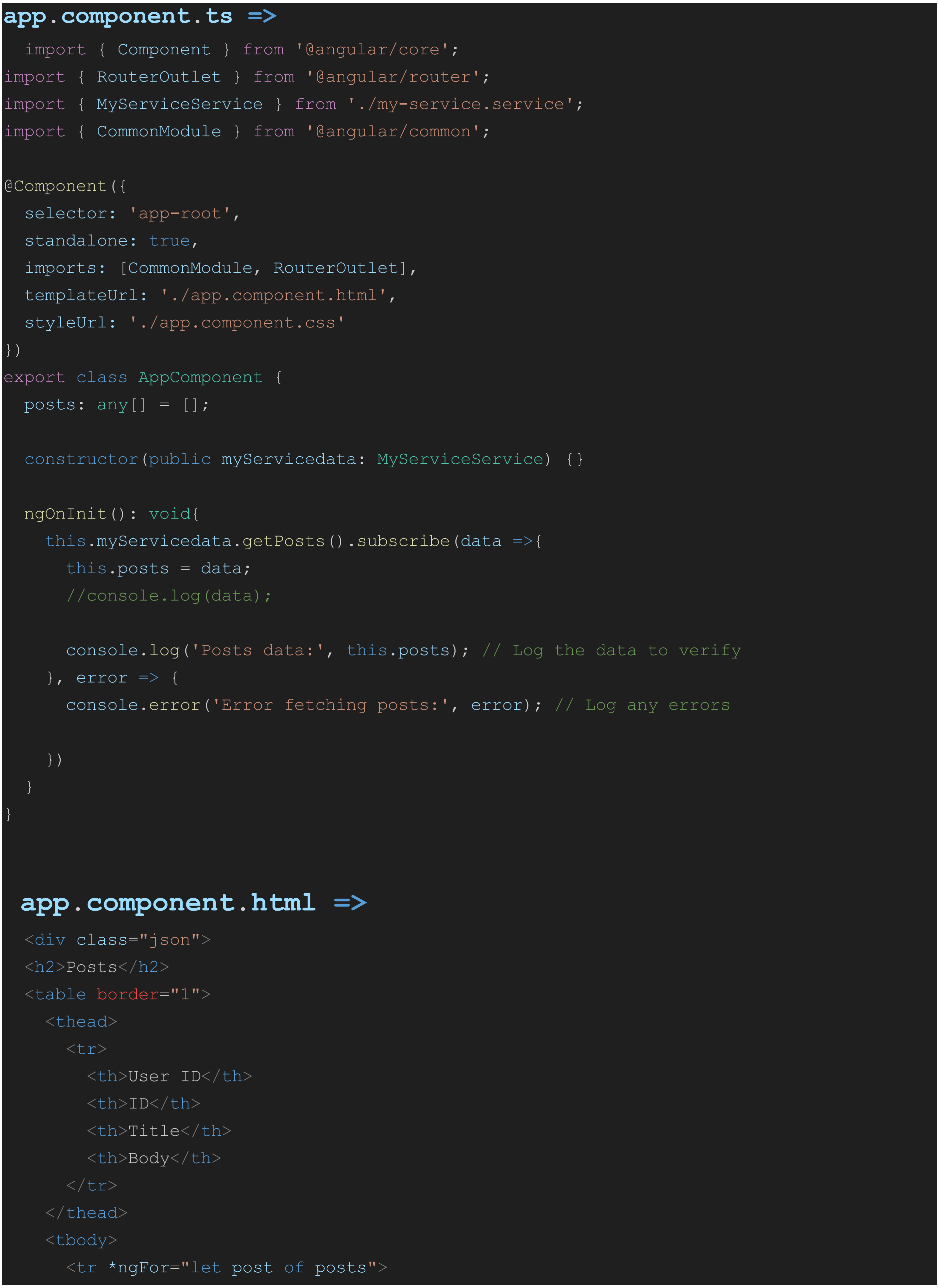
Write an AngularJS script to print details of bank (bank name, MICR code, IFC code, address etc.) in tabular form using ng-repeat.

Write an AngularJS script to print details of Employee (employee name, employee Id,Pin code, address etc.) in tabular form using ng-repeat.

```
<!DOCTYPE html>
<html ng-app="bank">
    <head>
        <title>Bank App</title>
        <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
    </head>
    <body ng-controller="BankController">
        <table border="1">
            <thead>
                <tr>
                    <th>bank name</th>
                    <th>micr</th>
                    <th>IFC Code</th>
                    <th>Address</th>
                </tr>
            </thead>
            <tbody>
                <tr ng-repeat="bank in banks">
                    <td>{{bank.name}}</td>
                    <td>{{bank.micr}}</td>
                    <td>{{bank.ifc}}</td>
                    <td>{{bank.addr}}</td>
                </tr>
            </tbody>
            <script>
                angular.module('bank',[]).controller('BankController',function($scope){
                    $scope.banks=[
                        {name:"Rushi",micr:"1234567",ifc:"RLS000143",addr:"Fitness
club"},
                        {name:"Shubham Bank",micr:"N/A",ifc:"N/A",addr:"N/A"}
                    ]
                })
            </script>
        </table>
    </body>
</html>

<!-- Write an AngularJS script to print details of Employee (employee name, employee
Id,Pin code, address etc.) in tabular form using ng-repeat.  -->
```

Slip 2B 8A
Create a simple Angular application that fetches data from an API using HttpClient. Implement an Observable to fetch data from an API endpoint.

```
my-service.service.ts =>
  import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';
import { Observable } from 'rxjs';

@Injectable({
  providedIn: 'root'
})
export class MyServiceService {

  public apiUrl = 'https://jsonplaceholder.typicode.com/posts';
  data:any;
  constructor(public http: HttpClient) { }

  getPosts():Observable<any[]>{
    console.log(this.apiUrl)
    return this.http.get<any[]>(this.apiUrl);
  }
}


   app.config.ts =>
import { ApplicationConfig, provideZoneChangeDetection } from '@angular/core';
import { provideRouter } from '@angular/router';

import { routes } from './app.routes';
import { provideHttpClient } from '@angular/common/http';

export const appConfig: ApplicationConfig = {
  providers: [provideZoneChangeDetection({ eventCoalescing: true }),
provideRouter(routes), provideHttpClient()]
};
```



app.component.ts =>

```
  import { Component } from '@angular/core';
import { RouterOutlet } from '@angular/router';
import { MyServiceService } from './my-service.service';
import { CommonModule } from '@angular/common';

@Component({
  selector: 'app-root',
  standalone: true,
  imports: [CommonModule, RouterOutlet],
  templateUrl: './app.component.html',
  styleUrl: './app.component.css'
})
export class AppComponent {
  posts: any[] = [];

  constructor(public myServicedata: MyServiceService) {}

  ngOnInit(): void{
    this.myServicedata.getPosts().subscribe(data =>{
      this.posts = data;
      //console.log(data);

      console.log('Posts data:', this.posts); // Log the data to verify
    }, error => {
      console.error('Error fetching posts:', error); // Log any errors

    })
  }
}
```

app.component.html =>

```
<div class="json">
<h2>Posts</h2>
<table border="1">
  <thead>
    <tr>
      <th>User ID</th>
      <th>ID</th>
      <th>Title</th>
      <th>Body</th>
    </tr>
  </thead>
  <tbody>
    <tr *ngFor="let post of posts">
```

```
        <td>{{ post.userId }}</td>
        <td>{{ post.id }}</td>
        <td>{{ post.title }}</td>
        <td>{{ post.body }}</td>
      </tr>
    </tbody>
  </table>
</div>
```

Slip 3A 4A
Write an AngularJS script to display list of games stored in an array on click of button using ng-click and also demonstrate ng-init, ng-bind directive of AngularJS.

```
<!-- Write an AngularJS script to display list of games stored in an array on click of
button using ng-click and also demonstrate ng-init, ng-bind directive of AngularJS. -->

<!DOCTYPE html>
<html ng-app="gamesApp">
<head>
    <title>List Of Games</title>
    <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
</head>
<body ng-controller="gameController">
    <button ng-click="fetchGames()"> Show </button>

    <p ng-repeat=" game in games" ng-bind="game"></p>

    <script>
        angular.module('gamesApp', [])
        .controller('gameController', function($scope) {
            $scope.fetchGames = function() {
                $scope.games = ['Football', 'Basketball', 'Tennis', 'Cricket', 'Hockey'];
            };
        });
    </script>
</body>
</html>
```

Slip 3B 6B 9A 15A

Find a company with a workforce greater than 30 in the array (use find by id method)

```
const companies = [
    { id: 1, name: "Company 1", desc: "Description 1", workForce: 10 },
    { id: 2, name: "Company 2", desc: "Description 2", workForce: 450 },
    { id: 3, name: "Company 3", desc: "Description 3", workForce: 29 },
  ];

  const matchedCompany = companies.find(company => company.workForce > 30 && company.id);

  console.log(matchedCompany);

// if using ts then



// interface Company {
//     name: string;
//     desc: string;
//     workForce: number;
//   }

//   const companies: Company[] = [
//     { name: "GeeksforGeeks", desc: "A Computer Science Portal.", workForce: 200 },
//     { name: "Company 2", desc: "Description 1", workForce: 30 },
//     { name: "Company 3", desc: "Description 2", workForce: 10 },
//   ];

//   const matchedCompany = companies.find(company => company.workForce > 30);
//   console.log(matchedCompany);




//   to eun it use :
//   tsc --target ES6 a.ts
//   node a.js
```

Slip 4B 9B 10B 13A

Express.js application to include middleware for parsing request bodies (e.g., JSON, form data) and validating input data.

```
const express = require('express');
const bodyParser = require('body-parser');
const { body, validationResult } = require('express-validator');

const app = express();
const port = 3000;


app.use(bodyParser.json()); // To parse JSON bodies
app.use(bodyParser.urlencoded({ extended: true })); // To parse form data

app.post('/submit', [  body('name').isString().withMessage('Name must be a string'),
                       body('age').isInt({ min: 18 }).withMessage('Age must be at least
18'),], (req, res) => {

  const errors = validationResult(req);
  if (!errors.isEmpty()) {
    return res.status(400).json({ errors: errors.array() });
  }

  // If validation passes, proceed with the logic
  const { name, age } = req.body;
  res.status(200).json({ message: 'Data received successfully', name, age });
});

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



// Invoke-WebRequest -Uri http://localhost:3000/submit -Method POST -Headers
@{"Content-Type"="application/json"} -Body '{"name": "John Doe", "age": 26}'
// curl -X POST http://localhost:3000/submit -H "Content-Type: application/json" -d
'{"name": "John Doe", "age": 25}'
```

Slip 5A 13B

Create a simple Angular component that takes input data and displays it.

```
<!DOCTYPE html>
<html ng-app="display">
    <head>
        <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
    </head>
    <body ng-controller="disCon" >

        <input type="text" ng-model="input" ng-change="reset()">
        <button ng-click="showResult()" ng-disabled="!input">Show</button>

        <p ng-if="show && input" ng-bind="input"></p>


        <script>
            angular.module("display",[]).controller("disCon",function($scope){
                $scope.show = false;
                $scope.input = '';

                $scope.showResult = ()=> $scope.show=true;
                $scope.reset = ()=> $scope.show=false;

            })
        </script>
    </body>
</html>
```

Slip 5B 10A
Implement a simple server using Node.js.

```
const express = require('express');
const app = express();



app.get('/', (req, res) => {
  res.send('Hello, World! This is a simple Node.js server.');
});

const PORT = 3000;
app.listen(PORT, () => {
  console.log(`Server is running on http://localhost:${PORT}`);
});
```

Slip 6A 7B 8B 11A 12B 14B

Develop an Express.js application that defines routes for Create and Read operations on a resource (products).
Develop an Express.js application that defines routes for Create and Read operations on a resource (User).
Develop an Express.js application that defines routes for Create, Update operations on a resource (Employee).
Develop an Express.js application that defines routes for Create operations on a resource (Movie).

```
const express = require('express');
const bodyParser = require('body-parser');

const app = express();
const PORT = 3000;

app.use(bodyParser.json())
let products=[];

//CREATE
app.post('/products',(req,res)=>{
  const {id, name, price} = req.body;

  if(!id || !name || !price){
    return res.status(400).json({error:'mendatory id,name and price'})
  }
  const newItem = {id,name,price};
  products.push(newItem);
});

//READ
app.get('/products',(req,res) => {
  res.status(200).json(products);
});

//UPDATE
app.put('/products/:id',(req,res)=>{
  const {id} = req.params;
  const {name,price} = req.body;

  const item = products.find((p) =>p.id === parseInt(id));

  if(!item){
    return res.status(404).json({error:"not a product id"})
  }

  if(name) item.name = name;
  if(price) item.price = price;

});

app.listen(PORT,()=>{
  console.log("localhost:3000 is running...")
})

//curl -X POST http://localhost:3000/products -H "Content-Type: application/json"
-d '{"id":1 ,"name":"shubhams","price":150}'
//curl -X GET http://localhost:3000/products
//curl -X PUT http://localhost:3000/products/1 -H "Content-Type: application/json"
-d '{"id":1 ,"name":"shubhams","price":150}'
```

```


const express = require('express');
const bodyParser = require('body-parser');

const app = express();
const PORT = 3000;

app.use(bodyParser.json())
let employees=[];

//CREATE
app.post('/employee',(req,res)=>{
  const {id, name, salary} = req.body;

  if(!id || !name || !salary){
    return res.status(400).json({error:'mendatory id,name and salary'})
  }
  const newItem = {id,name,salary};
  employees.push(newItem);
});

//READ
app.get('/employee',(req,res) => {
  res.status(200).json(employees);
});

//UPDATE
app.put('/employee/:id',(req,res)=>{
  const {id} = req.params;
  const {name,salary} = req.body;

  const item = employees.find((p) =>p.id === parseInt(id));

  if(!item){
    return res.status(404).json({error:"not a employee id"})
  }

  if(name) item.name = name;
  if(salary) item.salary = salary;

});

app.listen(PORT,()=>{
  console.log("localhost:3000 is running...")
})

//curl -X POST http://localhost:3000/employee -H "Content-Type: application/json" -d
'{"id":1 ,"name":"shubhams","salary":150}'
//curl -X GET http://localhost:3000/employee
//curl -X PUT http://localhost:3000/employee/1 -H "Content-Type: application/json" -d
'{"id":1 ,"name":"shubhams","salary":150}'
```

Slip 11B 14A 15B

Create Angular application that print the name of students who play basketball using filter and map method.

```
<!DOCTYPE html>
<html ng-app="Basketball">
    <head>
        <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
    </head>
    <body ng-controller="basketCon">
        <p ng-repeat="stud in Basketball">{{stud}}</p>

        <script>
            angular.module('Basketball',[]).controller('basketCon',function($scope){

                $scope.students = [ {name:'shubham',sports:['football','cricket'],perc:91},
                                 {name:'rushikesh', sports:['kabaddi','basketball'], perc:85},
                                 {name:'pankaj', sports:['basketball','racing'], perc:35}
```

```
                               ];

                $scope.Basketball = $scope.students
                                    .filter(s => s.sports.includes('basketball'))
                                    .map(s => s.name)

                // $scope.Basketball = $scope.students
                //                     .filter(s => s.perc == 85)
                //                     .map(s => s.name)
            });
        </script>
    </body>
</html>
```