| **Experiment No. – 5** | | | | |
| --- | --- | --- | --- | --- |
| **Date of Performance:** | **5/2/25** | | | |
| **Date of Submission:** | **12/2/25** | | | |
| Program Execution/  formation/  correction/  ethical practices  (06) | Timely  Submission  (01) | Viva  (03) | Experiment  Total (10) | Sign with Date |
|  |  |  |  |  |

**Experiment No. 5**

.**5.1 Aim:**  Design Web services using AngularJS Framework

**5.2 Course Outcome:** CO2: Understand how TypeScript and AngularJS framework can build dynamic, responsive single-page web applications

**5.3 Learning Objectives:** To learn how to effectively use AngularJS services to create modular and maintainable applications, understand the use of built-in services like $http, $timeout, $interval, and $location for handling data fetching, delays, timers, and routing, and implement custom services using .service(), .factory(), or .provider() to encapsulate and reuse business logic across controllers and components.

**5.4 Requirement:**

* Visual Studio Code / Any IDE
* AngularJS Library (CDN)
* Browser (Chrome/Edge/Firefox)
* Basic HTML/CSS knowledge

5.5 Related Theory:

AngularJS is a JavaScript-based open-source front-end web framework maintained by Google. In AngularJS, a Service is a reusable singleton object used to organize and share code across the application. It allows separation of concerns by moving business logic out of controllers and into service components.

Types of AngularJS Services::

1. $http: Performs AJAX calls to interact with backend APIs.
2. **$location:** Provides access to the browser's URL. Useful for routing.
3. **$timeout:** Executes a function after a specified time (like setTimeout).
4. **$interval:** Executes a function repeatedly at specified intervals (like setInterval).
5. **$route:** Handles routing in SPAs using ngRoute.
6. **$window:** Gives access to the browser's window object.
7. **Custom Services:** Defined using .service(), .factory(), or .provider(). explain it

5.6 Procedure:

Step 1: Include AngularJS library via CDN in the HTML file.

eg. <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>

Step 2: Define a module, Service, and a controller in the script.

Step 3: Use AngularJS Service in the View.

Step 4: Run the application in a browser and test the SPA behavior.

**5.7 Program and Output:**

<!DOCTYPE html>

<html ng-app="numConverterApp">

<head>

<title>Number Base Converter</title>

<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>

</head>

<body ng-controller="BaseConverterController" style="font-family: Arial, sans-serif; padding: 40px; max-width: 800px; margin: auto;">

<h1 style="text-align: center;">Base Converter App</h1>

<p style="text-align: center;"><strong>Live Time:</strong> {{ liveTime }}</p>

<p style="text-align: center;"><em>{{ headerMessage }}</em></p>

<section style="margin-top: 40px; text-align: center;">

<input type="number" ng-model="inputDecimal" placeholder="Enter decimal number" />

<button ng-click="runConversion()">Convert</button>

<div style="margin-top: 20px;">

<p><strong>Binary:</strong> {{ binaryResult }}</p>

<p><strong>Octal:</strong> {{ octalResult }}</p>

<p><strong>Hexadecimal:</strong> {{ hexResult }}</p>

<p><strong>Time Converted:</strong> {{ timeConverted | date: 'mediumTime' }}</p>

</div>

</section>

<section style="margin-top: 50px;">

<h3>Conversion History</h3>

<ul>

<li ng-repeat="item in history track by $index">

{{item.decimal}} ➡️ Bin: {{item.binary}}, Oct: {{item.octal}}, Hex: {{item.hex}} at {{item.time | date: 'shortTime'}}

</li>

</ul>

</section>

<footer style="margin-top: 40px; text-align: center; font-size: large;">

Page Link: <span style="color: darkblue;">{{ pageUrl }}</span>

</footer>

<script>

var app = angular.module('numConverterApp', []);

app.service('conversionService', function() {

this.toBinary = function(num) {

return parseInt(num).toString(2);

};

this.toOctal = function(num) {

return parseInt(num).toString(8);

};

this.toHex = function(num) {

return parseInt(num).toString(16);

};

});

app.controller('BaseConverterController', function($scope, $location, $interval, conversionService) {

$scope.inputDecimal = null;

$scope.binaryResult = '';

$scope.octalResult = '';

$scope.hexResult = '';

$scope.timeConverted = null;

$scope.history = [];

$scope.pageUrl = $location.absUrl();

$scope.headerMessage = "Convert Decimal to Binary, Octal, and Hex!";

let toggle = true;

$interval(() => {

$scope.liveTime = new Date().toLocaleTimeString();

}, 1000);

$interval(() => {

toggle = !toggle;

$scope.headerMessage = toggle ? "Convert Decimal to Binary, Octal, and Hex!" : "Quick & Easy Converter!";

}, 3000);

$scope.runConversion = function() {

if ($scope.inputDecimal === null || $scope.inputDecimal === "") {

alert("Please enter a decimal number.");

} else {

const num = parseInt($scope.inputDecimal);

$scope.binaryResult = conversionService.toBinary(num);

$scope.octalResult = conversionService.toOctal(num);

$scope.hexResult = conversionService.toHex(num);

$scope.timeConverted = new Date();

$scope.history.unshift({

decimal: num,

binary: $scope.binaryResult,

octal: $scope.octalResult,

hex: $scope.hexResult,

time: new Date()

});

}

};

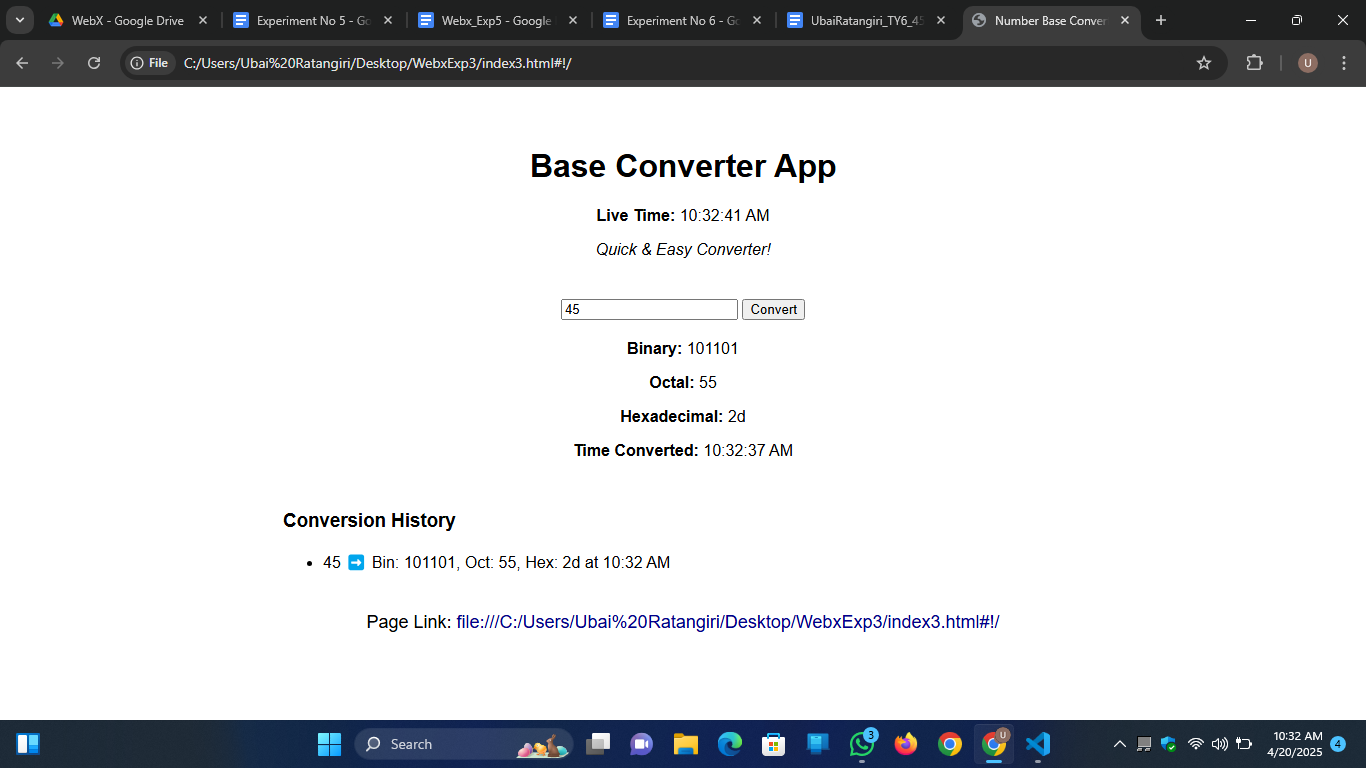
});

</script>

</body>

</html>

Output:



**5.8 Conclusion:** In this experiment, we explored how to effectively use **built-in services** and **custom services** to create dynamic and interactive web applications. By incorporating services like $http, $timeout, $interval, and $location, we were able to handle tasks such as fetching external data, managing time delays, and controlling the application’s behavior based on the URL or timer.

**5.9 Questions:**

1. **What is the role of services in AngularJS?**

Services in AngularJS organize and share logic/data across controllers, making code reusable and modular.

1. **What is the difference between $http and $location services in AngularJS?**

$http handles HTTP requests, while $location manages the browser’s URL and routing.

1. **How can services help in achieving code reusability in AngularJS?**

Services allow you to define common logic that can be reused across multiple controllers, reducing code duplication.