**APEX COLLEGE**

**POKHARA UNIVERSITY**

**LAB REPORT OF**

**DOT NET TECHNOLOGY**

**LAB -2**

**Submitted By: Submitted To:**

**Remisha Maharjan Bishwomitra Adhikari**

**BCIS ‘Adware’ Ananta Pant**

**Roll no: 17**

**OBJECTIVE:-**

AN INTRODUCTION TO WEB API (PART 2)

**THEORY:-**

The term API stands for Application Programming Interface. ASP.NET Web API is a framework for building Web APIs, i.e. HTTP based services on top of .NET Framework.that can be accessed in different applications on different platforms such as web, windows, mobile etc. It works more or less the same way as ASP.NET MVC web application except that it sends data as a response instead of html view. It is like a webservice or WCF service but the exception is that it only supports HTTP protocol. A server-side **web API** is a programmatic interface consisting of one or more publicly exposed endpoints to a defined request–response message system, typically expressed in JSON or XML, which is exposed via the **web**—most commonly by means of an HTTP-based **web** server.  
  
The most common case of using Web API is building RESTful services. These services can then be consumed by a broad range of clients like,

* Browsers
* Mobile Applications
* Desktop Applications
* IOT’s.

## ASP.NET Web API Characteristics:-

1. ASP.NET Web API is an ideal platform for building RESTful services.
2. ASP.NET Web API is built on top of ASP.NET and supports ASP.NET request/response pipeline
3. ASP.NET Web API maps HTTP verbs to method names.
4. ASP.NET Web API supports different formats of response data. Built-in support for JSON, XML, BSON format.
5. ASP.NET Web API can be hosted in IIS, Self-hosted or other web server that supports .NET 4.0+.
6. ASP.NET Web API framework includes new HttpClient to communicate with Web API server. HttpClient can be used in ASP.MVC server side, Windows Form application, Console application or other apps.

**CODE:-**

1.Make a model

**Check.cs**

**using System;**

**namespace lab1\_mvc.Models**

**{**

**public class Check**

**{**

**public int a { get; set; }**

**}**

**}**

2. In controller make an object to display odd or even number..

**HomeController.cs**

**using System;**

**using System.Collections.Generic;**

**using System.Diagnostics;**

**using System.Linq;**

**using System.Threading.Tasks;**

**using Microsoft.AspNetCore.Mvc;**

**using lab1\_mvc.Models;**

**namespace lab1\_mvc.Controllers**

**{**

**public class HomeController : Controller**

**{**

**[HttpPost]**

**public IActionResult Check(int a)**

**{**

**var r ="even";**

**if(a%2==0)**

**// return View("even");**

**r="even";**

**else**

**r="odd";**

**ViewBag.Result=r;**

**return View();**

**}**

**[HttpGet]**

**public IActionResult Check()**

**{**

**return View();**

**}**

**public IActionResult Privacy()**

**{**

**return View();**

**}**

**[ResponseCache(Duration = 0, Location = ResponseCacheLocation.None, NoStore = true)]**

**public IActionResult Error()**

**{**

**return View(new ErrorViewModel { RequestId = Activity.Current?.Id ?? HttpContext.TraceIdentifier });**

**}**

**}**

**}**

3. Showing the output in the view

**Check.cshtml**

**@model lab1\_mvc.Models.Check**

**@{**

**ViewData["Title"] = "Check Page";**

**}**

**@using (Html.BeginForm())**

**{**

**<input type="text" name="a">**

**<input type="submit" name="submit" value="Check Even or Odd" class="btn btn-info">**

**}**

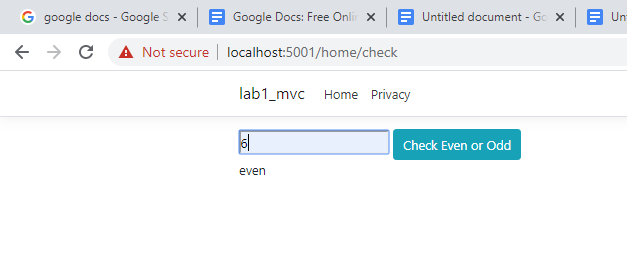
**@{**

**var result =ViewBag.Result;**

**<p>@result</p>**

**}**

**OUTPUT**

****

**CONCLUSION**

In lab 2, we learned to use basics of web api to find the odd or even number.