Q2 Web API

**Model**

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.Linq;

using System.Web;

namespace WebApiForProductTable.Models

{

public class Product

{

[Required(ErrorMessage = "ProudctId is Required ! ")]

[Display(Name = "Product Id")]

public int ProductId { set; get; }

[Required(ErrorMessage = "ProductName is Required ! ")]

[Display(Name = "Product Name")]

public string ProductName { set; get; }

[Required(ErrorMessage = "ProductName is Required ! ")]

[Display(Name = "Rate")]

public decimal Rate { set; get; }

[Required(ErrorMessage = "Discription is Required ! ")]

[Display(Name = "Discription")]

public string Discription { set; get; }

[Required(ErrorMessage = "CategoryName is Required ! ")]

[Display(Name = "Category Name")]

public string CategoryName { set; get; }

}

}

**ApiController**

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.Entity;

using System.Data.Entity.Infrastructure;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

using System.Web.Http.Description;

using WebApiForProductTable;

using WebApiForProductTable.Models;

namespace WebApiForProductTable.Controllers

{

public class ProductsController : ApiController

{

private JkJan22Entities db = new JkJan22Entities();

// GET: api/Products

public IQueryable<Product> GetProducts()

{

return db.Products;

}

// GET: api/Products/5

[ResponseType(typeof(Product))]

public IHttpActionResult GetProduct(int id)

{

Product product = db.Products.Find(id);

if (product == null)

{

return NotFound();

}

return Ok(product);

}

// PUT: api/Products/5

[ResponseType(typeof(void))]

public IHttpActionResult PutProduct(int id, Product product)

{

if (!ModelState.IsValid)

{

return BadRequest(ModelState);

}

if (id != product.ProductId)

{

return BadRequest();

}

db.Entry(product).State = EntityState.Modified;

try

{

db.SaveChanges();

}

catch (DbUpdateConcurrencyException)

{

if (!ProductExists(id))

{

return NotFound();

}

else

{

throw;

}

}

return StatusCode(HttpStatusCode.NoContent);

}

// POST: api/Products

[ResponseType(typeof(Product))]

public IHttpActionResult PostProduct(Product product)

{

if (!ModelState.IsValid)

{

return BadRequest(ModelState);

}

db.Products.Add(product);

db.SaveChanges();

return CreatedAtRoute("DefaultApi", new { id = product.ProductId }, product);

}

// DELETE: api/Products/5

[ResponseType(typeof(Product))]

public IHttpActionResult DeleteProduct(int id)

{

Product product = db.Products.Find(id);

if (product == null)

{

return NotFound();

}

db.Products.Remove(product);

db.SaveChanges();

return Ok(product);

}

protected override void Dispose(bool disposing)

{

if (disposing)

{

db.Dispose();

}

base.Dispose(disposing);

}

private bool ProductExists(int id)

{

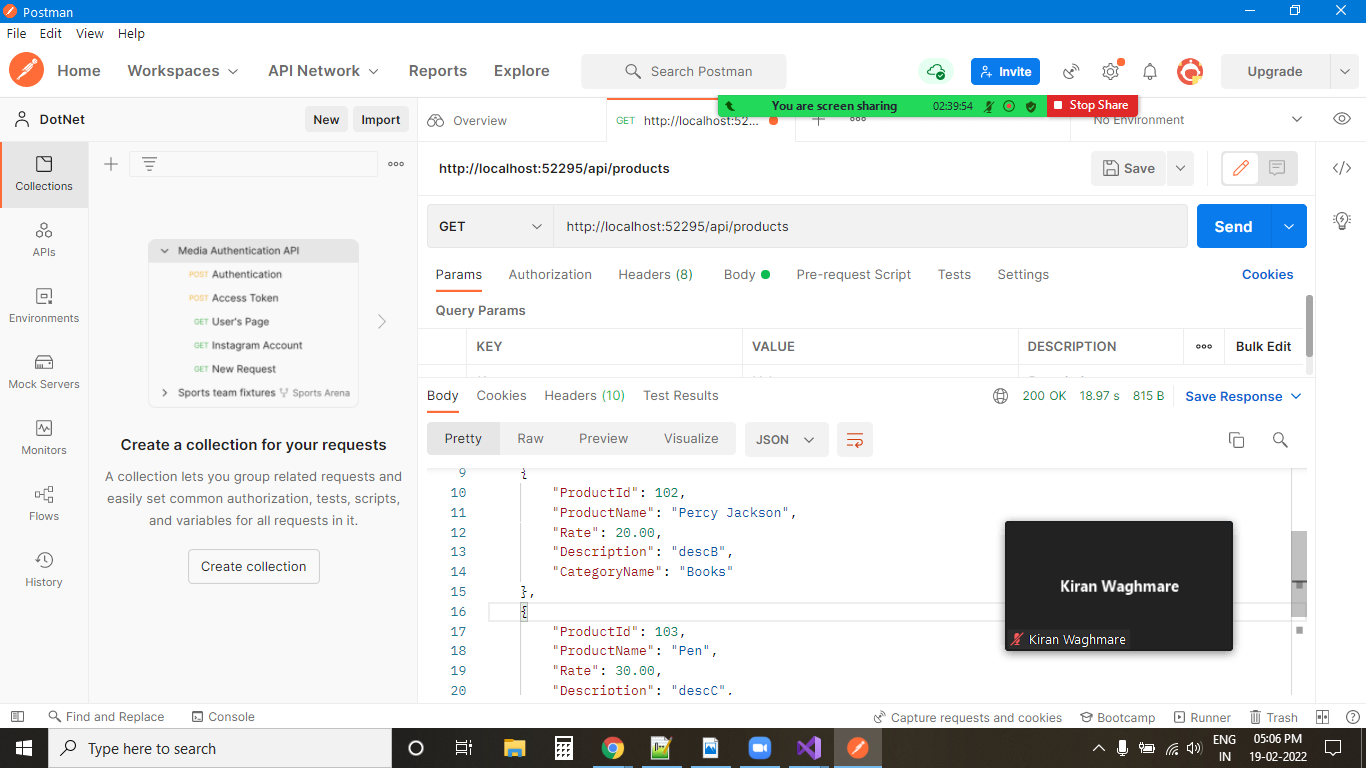
return db.Products.Count(e => e.ProductId == id) > 0;

}

}

}

**Postman-GET**



**Postman-Post**

