

**View:**

**Index.cshtml:**

Index.cshtml file is a view file that represents the user interface of an MVC application. It contains an async method to upload images and is responsible for rendering the interface to the user in the web browser.

The **.cshtml** file extension stands for "C# Razor syntax HTML," and it combines Razor syntax with HTML to allow developers to embed C# code in their HTML views. Razor syntax is a lightweight syntax for embedding server-side code in web pages.

Here is an example of an Index.cshtml file that displays a list of products in an MVC application:

public async Task<IActionResult> OnPostUpload(FileUpload fileUpload)

{

string fullPath = \_hostEnvironment.WebRootPath + "/UploadImages/";

if (!Directory.Exists(fullPath))

{

Directory.CreateDirectory(fullPath);

}

var formFile = fileUpload.FormFile;

if (formFile.Length > 0)

{

var filePath = Path.Combine(fullPath, formFile.FileName);

ViewData["ImageUrl"] = formFile.FileName;

using (var stream = new FileStream(filePath, FileMode.Create))

{

await formFile.CopyToAsync(stream);

}

}

}

**Services:**

A service is a component that provides a set of related functions or operations that can be performed on a given domain. Services are typically used to encapsulate business logic or to access external resources, such as databases or third-party APIs.

In CompterVisionService perform the operation on image stream using **‘AnalyzeImageUrl()**’ async method. In this method Creating a list that defines the features to be extracted from the image.

Ex. Categories, Faces, Tags, Color, Description, ImageType, Adult, Brands.

List<VisualFeatureTypes?> features = new List<VisualFeatureTypes?>() {

VisualFeatureTypes.Categories, VisualFeatureTypes.Description,

VisualFeatureTypes.Faces, VisualFeatureTypes.ImageType,

VisualFeatureTypes.Tags, VisualFeatureTypes.Adult,

VisualFeatureTypes.Color, VisualFeatureTypes.Brands,

VisualFeatureTypes.Objects

**Data Access Layer:**

The DataContext class is typically used to represent a data context for a database. It is a class that is responsible for interacting with a database and performing various operations on the data stored in the database.

public class CompVisionDataContext : DbContext

{

public DbSet<dbComputerVision> dbComputerVision { get; set; }

protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

{

optionsBuilder.UseSqlServer(@"Data Source=DESKTOP-G75N66U;Initial Catalog=DocService;Integrated Security=True");

}

}

In CompVisionDataContext derived from the DbContext class , which is part of the Entity framework.

The **DbContext** class is responsible for managing the entity objects in a database and handling the communication with the database.

The DataContext class has one proprietary **dbComputerVision**, which represent the entity sets for the **dbComputerVision** objects, respectively. These properties allow the DataContext to perform various operations on the data stored in the database,