Weather Stack Net Core

Sample .NET Core Application for consuming Weather Stack API

Table of Contents

1 Symbol Reference

1.1 Files

The following table lists files in this documentation.

Files

Name	Description
AutoComplete.cs (see page 1)	This is file AutoComplete.cs.
AutoCompleteController.cs (see page 2)	This is file AutoCompleteController.cs.
AutoCompleteViewModel.cs (see page 4)	This is file AutoCompleteViewModel.cs.
Current.cs (see page 4)	This is file Current.cs.
CurrentWeather.cs (see page 5)	This is file CurrentWeather.cs.
CurrentWeatherController.cs (see page 5)	This is file CurrentWeatherController.cs.
CurrentWeatherViewModel.cs (see page 8)	This is file CurrentWeatherViewModel.cs.
Error.cs (see page 9)	This is file Error.cs.
ErrorResult.cs (see page 9)	This is file ErrorResult.cs.
ErrorViewModel.cs (see page 9)	This is file ErrorViewModel.cs.
GeneralRequest.cs (see page 9)	This is file GeneralRequest.cs.
HomeController.cs (see page 10)	This is file HomeController.cs.
ItemList.cs (see page 10)	This is file ItemList.cs.
Location.cs (see page 11)	This is file Location.cs.
LocationRequest.cs (see page 11)	This is file LocationRequest.cs.
site.js (see page 11)	Please see documentation at https://docs.microsoft.com/aspnet/core/client-side/bundling-and-minification for details on configuring this project to bundle and minify static web assets.
WeatherStackNetCore.csproj (see page 12)	This is file WeatherStackNetCore.csproj.
WeatherStackNetCore.sln (see page 12)	This is file WeatherStackNetCore.sln.

1.1.1 AutoComplete.cs

This is file AutoComplete.cs.

Body Source

```
?namespace WeatherStackNetCore.Utils;

/// <summary>
/// API Request Result Class
/// </summary>
public class AutoComplete
{
    public LocationRequest request { get; set; }
```

```
        public List<Location> results { get; set; }
        public string success { get; set; }
        public Error error { get; set; }
    }
```

1.1.2 AutoCompleteController.cs

This is file AutoCompleteController.cs.

Body Source

```
    ?using System.Runtime.Serialization.Json;
    using System.Text;
    using Microsoft.AspNetCore.Mvc;
    using WeatherStackNetCore.Models;
    using WeatherStackNetCore.Utils;

    namespace WeatherStackNetCore.Controllers;

    /// <summary>
    /// This controller is used to
    /// Call API function which is about location searching
    /// </summary>
    public class AutoCompleteController : Controller
    {
        /// <summary>
        /// Config element
        /// </summary>
        private readonly IConfiguration _configuration;

        /// <summary>
        /// This is the constructor of controller
        /// </summary>
        /// <param name="config"></param>
        public AutoCompleteController(IConfiguration config)
        {
            // this is used to get some key values from appSettings.Json
            _configuration = config;
        }

        /// <summary>
        /// This is used to show Location search page
        /// </summary>
        /// <returns>Location search page</returns>
        public IActionResult IndexWithJQuery()
        {
            var model = new AutoCompleteViewModel();
            return View(model);
        }

        /// <summary>
        /// This is used to show Location search page
        /// </summary>
        /// <returns>Location search page</returns>
        public IActionResult IndexWithModel(AutoCompleteViewModel? model)
        {
            model ??= new AutoCompleteViewModel();
            return View(model);
        }

        /// <summary>
        /// This method is used to make WeatherStack Current API call
        /// And bring the result to the screen
        /// </summary>
        /// <param name="placeName">General Parameter (Like City Name, County Name etc)</param>
        /// <returns>Location Info</returns>
```

```
    [HttpPost]
    public async Task<AutoComplete?> GetLocations(string placeName)
    {
        try
        {
            var apiKey = _configuration.GetValue<string>("API_Key");

            var httpClient = new HttpClient();
            httpClient.Timeout = new TimeSpan(0, 0, 30);

            var requestString = "http://api.weatherstack.com/" +
$"autocomplete?access_key={apiKey}&query={placeName}";

            var response = await httpClient.GetAsync(requestString);
            var result = await response.Content.ReadAsStringAsync();

            var serializer = new DataContractJsonSerializer(typeof(AutoComplete));
            var memoryStream = new MemoryStream(Encoding.UTF8.GetBytes(result));
            var autoComplete = (AutoComplete)serializer.ReadObject(memoryStream)!;

            return autoComplete;
        }
        catch (Exception e)
        {
            return null;
        }
    }

    /// <summary>
    /// This method is used to make WeatherStack Current API call
    /// And bring the result to the screen
    /// But in this method we use the whole model
    /// </summary>
    /// <param name="model">Location Search Page Elements</param>
    /// <returns>Result Page</returns>
    [HttpPost]
    public async Task<IActionResult> GetLocationsWithModel(AutoCompleteViewModel model)
    {
        try
        {
            if (!ModelState.IsValid) return View("IndexWithModel", model);
            var apiKey = _configuration.GetValue<string>("API_Key");

            var httpClient = new HttpClient();
            httpClient.Timeout = new TimeSpan(0, 0, 30);

            var requestString = "http://api.weatherstack.com/" +
$"autocomplete?access_key={apiKey}&query={model.PlaceName}";

            var response = await httpClient.GetAsync(requestString);
            var result = await response.Content.ReadAsStringAsync();

            var serializer = new DataContractJsonSerializer(typeof(AutoComplete));
            var memoryStream = new MemoryStream(Encoding.UTF8.GetBytes(result));
            var autoComplete = (AutoComplete)serializer.ReadObject(memoryStream)!;

            model = new AutoCompleteViewModel
            {
                AutoComplete = autoComplete
            };

            if (model.AutoComplete.error != null)
            {
                ViewBag.MessageType = "error";
                ViewBag.BoxType = "large";
                ViewBag.Message = model.AutoComplete.error.info;
            }
            else
            {
```

```
                ViewBag.MessageType = "success";
                ViewBag.BoxType = "normal";
                ViewBag.Message = "Operation is successful";
            }

            return View("IndexWithModel", model);
        }
        catch (Exception e)
        {
            model = new AutoCompleteViewModel
            {
                AutoComplete = null
            };

            ViewBag.MessageType = "error";
            ViewBag.BoxType = "large";
            ViewBag.Message = "Something has gone wrong";

            return View("IndexWithModel", model);
        }
    }
}
```

1.1.3 AutoCompleteViewModel.cs

This is file AutoCompleteViewModel.cs.

Body Source

```
using System.ComponentModel.DataAnnotations;
using WeatherStackNetCore.Utils;

namespace WeatherStackNetCore.Models;

/// <summary>
/// This model class is used to create the search page
/// And stores the result info too
/// </summary>
public class AutoCompleteViewModel
{
    #region Fields

    [Required(ErrorMessage = "Please Enter A Place Name")]
    [Display(Name = "Place Name:")]
    public string? PlaceName { get; init; }

    public AutoComplete? AutoComplete { get; init; }

    #endregion
}
```

1.1.4 Current.cs

This is file Current.cs.

Body Source

```
?namespace WeatherStackNetCore.Utils;

/// <summary>
/// Query Place Current Weather Information Class
/// </summary>
```

```
public class Current
{
    public string observation_time { get; set; }
    public decimal temperature { get; set; }
    public decimal weather_code { get; set; }
    public List<string> weather_icons { get; set; }
    public List<string> weather_descriptions { get; set; }
    public decimal wind_speed { get; set; }
    public decimal wind_degree { get; set; }
    public string wind_dir { get; set; }
    public decimal pressure { get; set; }
    public decimal precip { get; set; }
    public decimal humidity { get; set; }
    public decimal cloudcover { get; set; }
    public decimal feelslike { get; set; }
    public decimal uv_index { get; set; }
    public decimal visibility { get; set; }
}
```

1.1.5 CurrentWeather.cs

This is file CurrentWeather.cs.

Body Source

```
?namespace WeatherStackNetCore.Utils;

/// <summary>
/// API Request Result Class
/// </summary>
public class CurrentWeather
{
    public GeneralRequest request { get; set; }
    public Location location { get; set; }
    public Current current { get; set; }
    public string success { get; set; }
    public Error error { get; set; }
}
```

1.1.6 CurrentWeatherController.cs

This is file CurrentWeatherController.cs.

Body Source

```
?using System.Runtime.Serialization.Json;
using System.Text;
using Microsoft.AspNetCore.Mvc;
using WeatherStackNetCore.Models;
using WeatherStackNetCore.Utils;

namespace WeatherStackNetCore.Controllers;

/// <summary>
/// This controller is used to
/// Call API function which is about Current Weather Info searching
/// </summary>
public class CurrentWeatherController : Controller
{
    /// <summary>
    /// Config element
    /// </summary>
```

```
    private readonly IConfiguration _configuration;

    /// <summary>
    /// This is the constructor of controller
    /// </summary>
    /// <param name="config"></param>
    public CurrentWeatherController(IConfiguration config)
    {
        // this is used to get some key values from appSettings.Json
        _configuration = config;
    }

    /// <summary>
    /// This is used to show Current Weather search page
    /// </summary>
    /// <returns>Current Weather search page</returns>
    public IActionResult IndexWithJQuery()
    {
        var model = new CurrentWeatherViewModel();
        return View(model);
    }

    /// <summary>
    /// This is used to show Current Weather search page
    /// </summary>
    /// <returns>Current Weather search page</returns>
    public IActionResult IndexWithModel(CurrentWeatherViewModel? model)
    {
        model ??= new CurrentWeatherViewModel();
        return View(model);
    }

    /// <summary>
    /// This method is used to make WeatherStack Current API call
    /// And bring the result to the screen
    /// </summary>
    /// <param name="placeName">General Parameter (Like City Name, County Name etc)</param>
    /// <param name="unit">Unit Parameter</param>
    /// <param name="language">Language Parameter (You shouldn't fill this parameter if
your key is free version)</param>
    /// <returns>Current Weather Info</returns>
    [HttpPost]
    public async Task<CurrentWeather?> GetCurrentWeather(string placeName, string unit,
string language)
    {
        try
        {
            var apiKey = _configuration.GetValue<string>("API_Key");

            var httpClient = new HttpClient();
            httpClient.Timeout = new TimeSpan(0, 0, 30);

            var requestString = "http://api.weatherstack.com/" +
$"current?access_key={apiKey}&query={placeName}";
            if (!string.IsNullOrEmpty(unit))
                requestString += $"&unit={unit}";
            if (!string.IsNullOrEmpty(language))
                requestString += $"&language={language}";

            var response = await httpClient.GetAsync(requestString);
            var result = await response.Content.ReadAsStringAsync();

            var serializer = new DataContractJsonSerializer(typeof(CurrentWeather));
            var memoryStream = new MemoryStream(Encoding.UTF8.GetBytes(result));
            var currentWeather = (CurrentWeather)serializer.ReadObject(memoryStream)!;

            return currentWeather;
        }
        catch (Exception e)
```

```
        {
            return null;
        }
    }

    /// <summary>
    /// This method is used to make WeatherStack Current API call
    /// And bring the result to the screen
    /// But in this method we use the whole model
    /// </summary>
    /// <param name="model">Current Weather Search Page Elements</param>
    /// <returns>Result Page</returns>
    [HttpPost]
    public async Task<IActionResult> GetCurrentWeatherWithModel(CurrentWeatherViewModel
model)
    {
        try
        {
            if (!ModelState.IsValid) return View("IndexWithModel", model);
            var apiKey = _configuration.GetValue<string>("API_Key");

            var httpClient = new HttpClient();
            httpClient.Timeout = new TimeSpan(0, 0, 30);

            var requestString = "http://api.weatherstack.com/" +
$"current?access_key={apiKey}&query={model.PlaceName}";
            if (!string.IsNullOrEmpty(model.Unit))
                requestString += $"&unit={model.Unit}";
            if (!string.IsNullOrEmpty(model.Language))
                requestString += $"&language={model.Language}";

            var response = await httpClient.GetAsync(requestString);
            var result = await response.Content.ReadAsStringAsync();

            var serializer = new DataContractJsonSerializer(typeof(CurrentWeather));
            var memoryStream = new MemoryStream(Encoding.UTF8.GetBytes(result));
            var currentWeather = (CurrentWeather)serializer.ReadObject(memoryStream)!;

            model = new CurrentWeatherViewModel
            {
                CurrentWeather = currentWeather
            };

            if (model.CurrentWeather.error != null)
            {
                ViewBag.MessageType = "error";
                ViewBag.BoxType = "large";
                ViewBag.Message = model.CurrentWeather.error.info;
            }
            else
            {
                ViewBag.MessageType = "success";
                ViewBag.BoxType = "normal";
                ViewBag.Message = "Operation is successful";
            }

            return View("IndexWithModel", model);
        }
        catch (Exception e)
        {
            model = new CurrentWeatherViewModel
            {
                CurrentWeather = null
            };

            ViewBag.MessageType = "error";
            ViewBag.BoxType = "large";
            ViewBag.Message = "Something has gone wrong";
```

```
            return View("IndexWithModel", model);
        }
    }
}
```

1.1.7 CurrentWeatherViewModel.cs

This is file CurrentWeatherViewModel.cs.

Body Source

```
using System.ComponentModel.DataAnnotations;
using WeatherStackNetCore.Utils;

namespace WeatherStackNetCore.Models;

/// <summary>
/// This model class is used to create the search page
/// And stores the result info too
/// </summary>
public class CurrentWeatherViewModel
{
    #region Fields

    [StringLength(60, MinimumLength = 3)]
    [Required(ErrorMessage = "Please Enter A Place Name")]
    [Display(Name = "Place Name:")]
    public string? PlaceName { get; init; }

    [Display(Name = "Unit:")] public string? Unit { get; init; }

    [Display(Name = "Language:")] public string? Language { get; init; }

    /// <summary>
    /// This is used to fill dropdownlist for units
    /// </summary>
    public List<ItemList> UnitList { get; init; }

    /// <summary>
    /// This is used to fill dropdownlist for languages
    /// </summary>
    public List<ItemList> LanguageList { get; init; }

    /// <summary>
    /// API Call result
    /// </summary>
    public CurrentWeather? CurrentWeather { get; init; }

    #endregion

    public CurrentWeatherViewModel()
    {
        UnitList = new List<ItemList>
        {
            new() {Text = "Metric", Value = "m"},
            new() {Text = "Scientific", Value = "s"},
            new() {Text = "Fahrenheit", Value = "f"}
        };

        LanguageList = new List<ItemList>
        {
            new() {Text = "English", Value = "en"},
            new() {Text = "Turkish", Value = "tr"},
            new() {Text = "German", Value = "de"}
        };
    }
```

```
}
```

1.1.8 Error.cs

This is file Error.cs.

Body Source

```
?namespace WeatherStackNetCore.Utils;

/// <summary>
/// Error Class
/// </summary>
public class Error
{
    public string type { get; set; }
    public string info { get; set; }
    public int code { get; set; }
}
```

1.1.9 ErrorResult.cs

This is file ErrorResult.cs.

Body Source

```
?namespace WeatherStackNetCore.Utils;

/// <summary>
/// Error Class
/// </summary>
public class ErrorResult
{
    public string success { get; set; }
    public Error error { get; set; }
}
```

1.1.10 ErrorViewModel.cs

This is file ErrorViewModel.cs.

Body Source

```
namespace WeatherStackNetCore.Models;

public class ErrorViewModel
{
    public string? RequestId { get; init; }

    public bool ShowRequestId => !string.IsNullOrEmpty(RequestId);
}
```

1.1.11 GeneralRequest.cs

This is file GeneralRequest.cs.

Body Source

```
?namespace WeatherStackNetCore.Utils;

/// <summary>
/// API Request Class
/// </summary>
public class GeneralRequest
{
    public string type { get; set; }
    public string query { get; set; }
    public string language { get; set; }
    public string unit { get; set; }
}
```

1.1.12 HomeController.cs

This is file HomeController.cs.

Body Source

```
?using System.Diagnostics;
using Microsoft.AspNetCore.Mvc;
using WeatherStackNetCore.Models;

namespace WeatherStackNetCore.Controllers;

public class HomeController : Controller
{
    private readonly ILogger<HomeController> _logger;

    public HomeController(ILogger<HomeController> logger)
    {
        _logger = logger;
    }

    public IActionResult Index()
    {
        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});
    }
}
```

1.1.13 ItemList.cs

This is file ItemList.cs.

Body Source

```
?namespace WeatherStackNetCore.Utils;
```

```
public class ItemList
{
    public string Text { get; set; }
    public string Value { get; set; }
}
```

1.1.14 Location.cs

This is file Location.cs.

Body Source

```
?namespace WeatherStackNetCore.Utils;

/// <summary>
/// Query Place Detailed Location Info Class
/// </summary>
public class Location
{
    public string name { get; set; }
    public string country { get; set; }
    public string region { get; set; }
    public string lat { get; set; }
    public string lon { get; set; }
    public string timezone_id { get; set; }
    public string localtime { get; set; }
    public decimal localtime_epoch { get; set; }
    public string utc_offset { get; set; }
}
```

1.1.15 LocationRequest.cs

This is file LocationRequest.cs.

Body Source

```
?namespace WeatherStackNetCore.Utils;

/// <summary>
/// API Request Class
/// </summary>
public class LocationRequest
{
    public string query { get; set; }
    public int results { get; set; }
}
```

1.1.16 site.js

Please see documentation at https://docs.microsoft.com/aspnet/core/client-side/bundling-and-minification for details on configuring this project to bundle and minify static web assets.

Body Source

```
?// Please see documentation at
https://docs.microsoft.com/aspnet/core/client-side/bundling-and-minification
// for details on configuring this project to bundle and minify static web assets.

// Write your JavaScript code.
```

1.1.17 WeatherStackNetCore.csproj

This is file WeatherStackNetCore.csproj.

1.1.18 WeatherStackNetCore.sln

This is file WeatherStackNetCore.sln.

Index