# How to post custom field JSON to server and map it to C# model in ASP.NET Core

## Introduction

For naming convention, interface upgrade, or other reason, the format of JSON data can’t be kept consistency with C# model, so it can’t map to C# model anticipated either.

This sample will demonstrates how to post and map the custom field JSON data.

**Note:** make sure the data posted from client is JSON format and the Content-Type is “application/json” in request header.

## Sample prerequisites

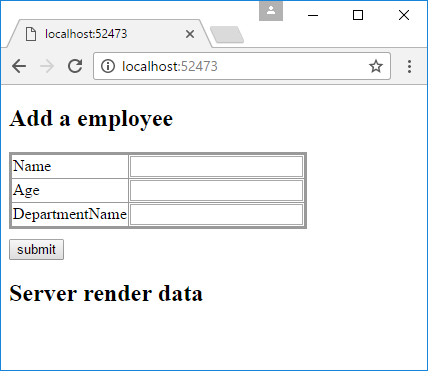
* Visual Studio 2017 or above. [[Visual Studio Home Page](https://www.visualstudio.com/)]
* Visual Studio enabled ASP.NET Core develop component.

## Building the sample

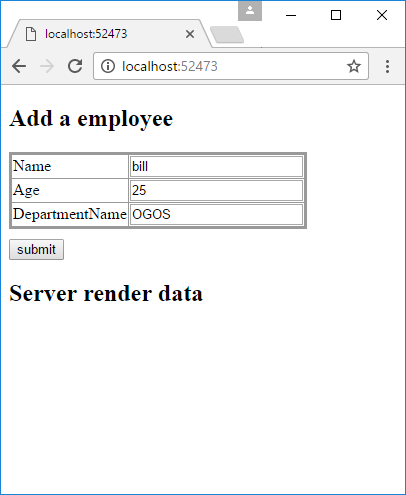
Use Visual Studio to open the sample solution, then press **F6 Key** to build the sample project.

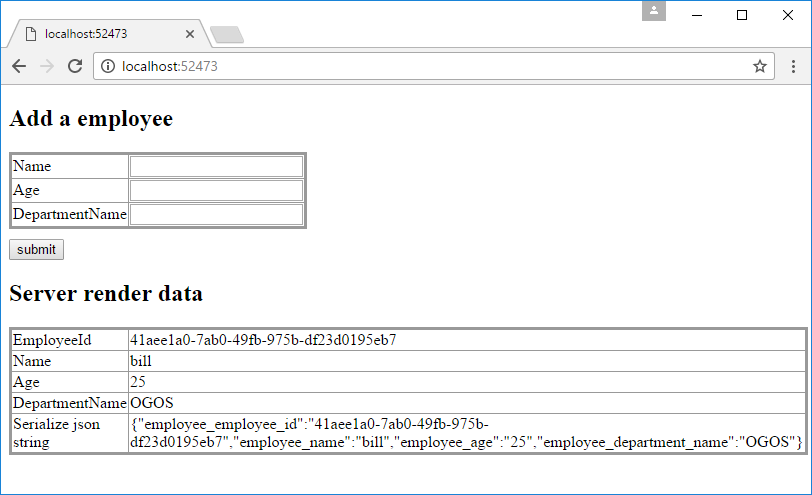
## Running the sample

* Use Visual Studio to open the sample solution, then press F5 Key or select **Debug -> Start Debugging** from the menu.
* When the project is running, you will see below page in browser.



* Fill all fields and click **submit** button.





## Using the code

**The Model**

public class Employee

{

[JsonProperty(propertyName: "employee\_employee\_id")]

public Guid EmployeeId { get; set; }

[JsonProperty(propertyName: "employee\_name")]

public string Name { get; set; }

[JsonProperty(propertyName: "employee\_age")]

public string Age { get; set; }

[JsonProperty(propertyName: "employee\_department\_name")]

public string DepartmentName { get; set; }

}

**Front end**

var jsonData = {

employee\_name: $("#name").val(),

employee\_age: $("#age").val(),

employee\_department\_name: $("#department\_name").val()

};

$.ajax({

url: "/Home/Index",

type: "POST",

contentType: "application/json",

data: JSON.stringify(jsonData),

success: function (data) {

if (data.state == 0) {

document.location.reload();

}

}

});

**Controller**

public class HomeController : Controller

{

private static List<Employee> \_employees = new List<Employee>();

[HttpGet]

public IActionResult Index()

{

return View(\_employees);

}

[HttpPost]

public JsonResult Index([FromBody]Employee employee)

{

if (employee != null) {

employee.EmployeeId = Guid.NewGuid();

\_employees.Add(employee);

}

return Json(new {

state = 0,

msg = string.Empty

});

}

}

## More information

Json.NET

<http://www.newtonsoft.com/json>

Model binding JSON POSTs in ASP.NET Core

<https://andrewlock.net/model-binding-json-posts-in-asp-net-core/>