# Code Snippet - Creating an ASP.NET Web API Service as a reliable service

**bower.json:**

{

"name": "asp.net",

"private": true,

"dependencies": {

"bootstrap": "3.3.7",

"jquery": "2.2.0",

"jquery-validation": "1.14.0",

"jquery-validation-unobtrusive": "3.2.6",

"angular": "v1.6.5",

"angular-bootstrap": "v1.1.0"

}

}

**site.js:**

var app = angular.module('VotingApp', ['ui.bootstrap']);

app.run(function () { });

app.controller('VotingAppController', ['$rootScope', '$scope', '$http', '$timeout', function ($rootScope, $scope, $http, $timeout) {

$scope.refresh = function () {

$http.get('api/Votes?c=' + new Date().getTime())

.then(function (data, status) {

$scope.votes = data;

}, function (data, status) {

$scope.votes = undefined;

});

};

$scope.remove = function (item) {

$http.delete('api/Votes/' + item)

.then(function (data, status) {

$scope.refresh();

})

};

$scope.add = function (item) {

var fd = new FormData();

fd.append('item', item);

$http.put('api/Votes/' + item, fd, {

transformRequest: angular.identity,

headers: { 'Content-Type': undefined }

})

.then(function (data, status) {

$scope.refresh();

$scope.item = undefined;

})

};

}]);

**Site.js:**

**wwwroot/js/site.js:**

var app = angular.module('VotingApp', ['ui.bootstrap']);

app.run(function () { });

app.controller('VotingAppController', ['$rootScope', '$scope', '$http', '$timeout', function ($rootScope, $scope, $http, $timeout) {

$scope.refresh = function () {

$http.get('api/Votes?c=' + new Date().getTime())

.then(function (data, status) {

$scope.votes = data;

}, function (data, status) {

$scope.votes = undefined;

});

};

$scope.remove = function (item) {

$http.delete('api/Votes/' + item)

.then(function (data, status) {

$scope.refresh();

})

};

$scope.add = function (item) {

var fd = new FormData();

fd.append('item', item);

$http.put('api/Votes/' + item, fd, {

transformRequest: angular.identity,

headers: { 'Content-Type': undefined }

})

.then(function (data, status) {

$scope.refresh();

$scope.item = undefined;

})

};

}]);

**Index.cshtml:**

Views/Home/Index.cshtml

@{

ViewData["Title"] = "Service Fabric Voting Sample";

}

<div ng-controller="VotingAppController" ng-init="refresh()">

<div class="container-fluid">

<div class="row">

<div class="col-xs-8 col-xs-offset-2 text-center">

<h2>Service Fabric Voting Sample</h2>

</div>

</div>

<div class="row">

<div class="col-xs-8 col-xs-offset-2">

<form class="col-xs-12 center-block">

<div class="col-xs-6 form-group">

<input id="txtAdd" type="text" class="form-control" placeholder="Add voting option" ng-model="item" />

</div>

<button id="btnAdd" class="btn btn-default" ng-click="add(item)">

<span class="glyphicon glyphicon-plus" aria-hidden="true"></span>

Add

</button>

</form>

</div>

</div>

<hr />

<div class="row">

<div class="col-xs-8 col-xs-offset-2">

<div class="row">

<div class="col-xs-4">

Click to vote

</div>

</div>

<div class="row top-buffer" ng-repeat="vote in votes.data">

<div class="col-xs-8">

<button class="btn btn-success text-left btn-block" ng-click="add(vote.key)">

<span class="pull-left">

{{vote.key}}

</span>

<span class="badge pull-right">

{{vote.value}} Votes

</span>

</button>

</div>

<div class="col-xs-4">

<button class="btn btn-danger pull-right btn-block" ng-click="remove(vote.key)">

<span class="glyphicon glyphicon-remove" aria-hidden="true"></span>

Remove

</button>

</div>

</div>

</div>

</div>

</div>

</div>

### \_Layout.cshtml:

### Views/Shared/\_Layout.cshtml

<!DOCTYPE html>

<html ng-app="VotingApp" xmlns:ng="http://angularjs.org">

<head>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>@ViewData["Title"]</title>

<link href="~/lib/bootstrap/dist/css/bootstrap.min.css" rel="stylesheet" />

<link href="~/css/site.css" rel="stylesheet" />

</head>

<body>

<div class="container body-content">

@RenderBody()

</div>

<script src="~/lib/jquery/dist/jquery.js"></script>

<script src="~/lib/bootstrap/dist/js/bootstrap.js"></script>

<script src="~/lib/angular/angular.js"></script>

<script src="~/lib/angular-bootstrap/ui-bootstrap-tpls.js"></script>

<script src="~/js/site.js"></script>

@RenderSection("Scripts", required: false)

</body>

</html>

### VotingWeb.cs

Add the using System.Net.Http; directive to the top of the file.

Replace the CreateServiceInstanceListeners() function with the following, then save your changes.

protected override IEnumerable<ServiceInstanceListener> CreateServiceInstanceListeners()

{

return new ServiceInstanceListener[]

{

new ServiceInstanceListener(serviceContext =>

new WebListenerCommunicationListener(serviceContext, "ServiceEndpoint", (url, listener) =>

{

ServiceEventSource.Current.ServiceMessage(serviceContext, $"Starting WebListener on {url}");

return new WebHostBuilder().UseWebListener()

.ConfigureServices(

services => services

.AddSingleton<StatelessServiceContext>(serviceContext)

.AddSingleton<HttpClient>())

.UseContentRoot(Directory.GetCurrentDirectory())

.UseStartup<Startup>()

.UseApplicationInsights()

.UseServiceFabricIntegration(listener, ServiceFabricIntegrationOptions.None)

.UseUrls(url)

.Build();

}))

};

}

### VotesController.cs

Add a controller which defines voting actions. Right-click on the **Controllers** folder, then select **Add->New item->Class**. Name the file "VotesController.cs" and click **Add**.

using System;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Mvc;

using System.Collections.Generic;

using Newtonsoft.Json;

using System.Text;

using System.Net.Http;

using System.Net.Http.Headers;

namespace VotingWeb.Controllers

{

[Produces("application/json")]

[Route("api/Votes")]

public class VotesController : Controller

{

private readonly HttpClient httpClient;

public VotesController(HttpClient httpClient)

{

this.httpClient = httpClient;

}

// GET: api/Votes

[HttpGet]

public async Task<IActionResult> Get()

{

List<KeyValuePair<string, int>> votes= new List<KeyValuePair<string, int>>();

votes.Add(new KeyValuePair<string, int>("Pizza", 3));

votes.Add(new KeyValuePair<string, int>("Ice cream", 4));

return Json(votes);

}

}

}

### Configure the listening port

VotingWeb/PackageRoot/ServiceManifest.xml

<Resources>

<Endpoints>

<!-- This endpoint is used by the communication listener to obtain the port on which to

listen. Please note that if your service is partitioned, this port is shared with

replicas of different partitions that are placed in your code. -->

<Endpoint Protocol="http" Name="ServiceEndpoint" Type="Input" Port="80" />

</Endpoints>

</Resources>