

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Threading.Tasks;
5 using Microsoft.AspNetCore.Authorization;
6 using Microsoft.AspNetCore.Mvc;
7 using PDLERP.Models;
8 using PDLERP.ServiceInterfaces.Emp;
9
10 namespace PDLERP.Controllers
11 {
12     [AllowAnonymous]
13     [Route("emp/")]
14     //[ApiController]
15     public class ApiController : ControllerBase
16     {
17         private readonly IEmployee _employee;
18
19         public ApiController(IEmployee employee)
20         {
21             _employee = employee;
22         }
23
24         [HttpGet("GetAll")]
25         public async Task<IActionResult> GetAll()
26         {
27             try
28             {
29                 var employees = await _employee.GetAll();
30
31                 if (employees.Any())
32                     return Ok(employees);
33                 return NoContent();
34             }
35             catch (Exception)
36             {
37                 return BadRequest();
38             }
39         }
40
41         [HttpGet("Find/{id}")]
42         public async Task<IActionResult> FindById(int id)
43         {
44             try
45             {
46                 var employees = await _employee.All();
47                 var firstOrDefault = employees.FirstOrDefault(e => e.Id.Equals
48                     (id));
```

```
49         if (firstOrDefault == null)
50             return NoContent();
51
52         return Ok(firstOrDefault);
53     }
54     catch (Exception)
55     {
56         return BadRequest();
57     }
58 }
59
60 [HttpPost("Post")]
61 // [Route("/Post")]
62 public async Task<IActionResult> PostEmployee(Employee employee)
63 {
64     try
65     {
66         await _employee.GetInsertedObjByAsync(employee);
67         return CreatedAtAction(nameof(FindById), new { id = employee.Id }, employee);
68     }
69     catch (Exception)
70     {
71         return BadRequest();
72     }
73 }
74
75 [HttpPut("Update")]
76 public async Task<IActionResult> UpdateEmployee(Employee employee)
77 {
78     try
79     {
80         var findByIdAsync = await _employee.FindByIdAsync(employee.Id);
81
82         if (findByIdAsync == null)
83             return NotFound();
84
85         findByIdAsync.FirstName = employee.FirstName;
86         findByIdAsync.MiddleName = employee.MiddleName;
87         findByIdAsync.LastName = employee.LastName;
88
89         if (await _employee.Update(findByIdAsync))
90             return CreatedAtAction(nameof(FindById), new { id = findByIdAsync.Id }, findByIdAsync);
91         return NoContent();
92     }
93     catch (Exception)
94     {
95         return BadRequest();
96     }
97 }
```

```
96         }
97     }
98
99     [HttpDelete("Delete/{id}")]
100     public async Task<IActionResult> DeleteEmployee(int id)
101     {
102         try
103         {
104             var findByIdAsync = await _employee.FindByIdAsync(id);
105
106             if (findByIdAsync == null)
107                 return NotFound();
108
109             await _employee.Delete(findByIdAsync);
110             return NoContent();
111         }
112         catch (Exception)
113         {
114             return BadRequest();
115         }
116     }
117 }
118 }
119
```