EmpProfile.cs

```
using System.ComponentModel.DataAnnotations.Schema;
using System.ComponentModel.DataAnnotations;

namespace EmployeeMS.Models
{
    [Table("Employee")]
    public class EmpProfile
    {
        [Key]
        public int EmpCode { get; set; }

        [Required]
        [StringLength(100)]
        public string EmpName { get; set; }

        [Required]
        [EmailAddress]
        [StringLength(100)]
        public string Email { get; set; }

        [Required]
        public DateTime DateOfBirth { get; set; }

        public int DeptCode { get; set; }

        [ForeignKey("DeptCode")]
        public virtual DeptMaster DeptMaster { get; set; }
    }
}
```

DeptMaster.cs

```
using System.ComponentModel.DataAnnotations.Schema;
using System.ComponentModel.DataAnnotations;

namespace EmployeeMS.Models
{
    [Table("Department")]
    public class DeptMaster
    {
        [Key]
        public int DeptCode { get; set; }

        [Required]
        [StringLength(50)]
        public string DeptName { get; set; }
    }
}
```

EmpProfileController.cs

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using EmployeeMS.Data;
using EmployeeMS.Models;

namespace EmployeeMS.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class EmpProfilesController : ControllerBase
    {
        private readonly EMSDbContext _context;

        public EmpProfilesController(EMSDbContext context)
        {
            _context = context;
        }

        // GET: api/EmpProfiles
        [HttpGet]
        public async Task<ActionResult<IEnumerable<EmpProfile>>> GetEmpProfile()
        {
          if (_context.EmpProfile == null)
          {
              return NotFound();
          }
            return await _context.EmpProfile.ToListAsync();
        }

        // GET: api/EmpProfiles/5
        [HttpGet("{id}")]
        public async Task<ActionResult<EmpProfile>> GetEmpProfile(int id)
        {
          if (_context.EmpProfile == null)
          {
              return NotFound();
          }
            var empProfile = await _context.EmpProfile.FindAsync(id);

            if (empProfile == null)
            {
                return NotFound();
            }

            return empProfile;
        }

        // PUT: api/EmpProfiles/5
        // To protect from overposting attacks, see https://go.microsoft.com/fwlink/?linkid=2123754
        [HttpPut("{id}")]
        public async Task<IActionResult> PutEmpProfile(int id, EmpProfile empProfile)
        {
            if (id != empProfile.EmpCode)
            {
                return BadRequest();
            }

            // Check if the Department Code has changed
            if (empProfile.DeptCode != null && empProfile.DeptCode != 0)
            {
                // Get the existing employee
                var existingEmployee = await _context.EmpProfile.FindAsync(id);

                if (existingEmployee == null)
                {
                    return NotFound();
                }

                // Update the employee's properties
                existingEmployee.EmpName = empProfile.EmpName;
                existingEmployee.Email = empProfile.Email;
                existingEmployee.DateOfBirth = empProfile.DateOfBirth;

                // Check if the DeptCode is changing
                if (empProfile.DeptCode != existingEmployee.DeptCode)
                {
                    // Update the EmpProfile entity
                    existingEmployee.DeptCode = empProfile.DeptCode;

                    // Retrieve the related DeptMaster entity
                    var deptMaster = await _context.DeptMaster.FindAsync(empProfile.DeptCode);

                    if (deptMaster != null)
                    {
                        // Update the DeptName
                        deptMaster.DeptName = empProfile.DeptMaster.DeptName; // Assuming the
DeptName is accessible this way
                    }
                }

                try
                {
                    await _context.SaveChangesAsync();
                }
                catch (DbUpdateConcurrencyException)
                {
                    if (!EmpProfileExists(id))
                    {
                        return NotFound();
                    }
                    else
                    {
                        throw;
                    }
                }

                return NoContent();
            }
            else
            {
                // If DeptCode is null or 0, you're not changing the department.
                // Proceed with standard update logic here.

                // Set EntityState to Modified for empProfile
                _context.Entry(empProfile).State = EntityState.Modified;

                try
                {
```

```
            await _context.SaveChangesAsync();
        }
        catch (DbUpdateConcurrencyException)
        {
            if (!EmpProfileExists(id))
            {
                return NotFound();
            }
            else
            {
                throw;
            }
        }

        return NoContent();
    }
}

// POST: api/EmpProfiles
// To protect from overposting attacks, see https://go.microsoft.com/fwlink/?linkid=2123754
[HttpPost]
public async Task<ActionResult<EmpProfile>> PostEmpProfile(EmpProfile empProfile)
{
  if (_context.EmpProfile == null)
  {
      return Problem("Entity set 'EMSDbContext.EmpProfile'  is null.");
  }
    _context.EmpProfile.Add(empProfile);
    await _context.SaveChangesAsync();

    return CreatedAtAction("GetEmpProfile", new { id = empProfile.EmpCode }, empProfile);
}

// DELETE: api/EmpProfiles/5
[HttpDelete("{id}")]
public async Task<IActionResult> DeleteEmpProfile(int id)
{
    var empProfile = await _context.EmpProfile.FindAsync(id);

    if (empProfile == null)
    {
        return NotFound();
    }

    // Store the department code before deletion
    var deptCodeToDelete = empProfile.DeptCode;

    _context.EmpProfile.Remove(empProfile);

    try
    {
        await _context.SaveChangesAsync();
    }
    catch (DbUpdateConcurrencyException)
    {
        if (!EmpProfileExists(id))
        {
            return NotFound();
```

```
                }
                else
                {
                    throw;
                }
            }

            // Check if the department has no more employees
            var employeesInDept = await _context.EmpProfile.CountAsync(e => e.DeptCode ==
deptCodeToDelete);

            if (employeesInDept == 0)
            {
                // If there are no more employees in the department, delete the department
                var deptToDelete = await _context.DeptMaster.FindAsync(deptCodeToDelete);
                if (deptToDelete != null)
                {
                    _context.DeptMaster.Remove(deptToDelete);
                    await _context.SaveChangesAsync();
                }
            }

            return NoContent();
        }

        private bool EmpProfileExists(int id)
        {
            return (_context.EmpProfile?.Any(e => e.EmpCode == id)).GetValueOrDefault();
        }
    }
}
```

EMSDbContext.cs

```
using Microsoft.EntityFrameworkCore;
using EmployeeMS.Models;

namespace EmployeeMS.Data
{
    public class EMSDbContext : DbContext
    {
        public EMSDbContext (DbContextOptions<EMSDbContext> options)
            : base(options)
        {
        }

        public DbSet<EmployeeMS.Models.EmpProfile> EmpProfile { get; set; } = default!;
        public DbSet<DeptMaster> DeptMaster { get; set; }
    }
}
```