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

namespace SimplonaTechEMS.Models
{
    [Table("EmpProfile")]
    public class EmpProfile
    {
        [Key]
        public int EmpCode { get; set; }
        public DateTime DateOfBirth { get; set; }
        public string EmpName { get; set; }
        public string Email { get; set; }
        public int DeptCode { get; set; }
        public virtual DeptMaster DeptMaster { get; set; }
    }
}
using System.ComponentModel.DataAnnotations.Schema;
using System.ComponentModel.DataAnnotations;

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

        public string DeptName { get; set; }

        public virtual ICollection<EmpProfile> EmpProfiles { get; set; }
    }
}
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*",
  "ConnectionStrings": {
    "EmployeeDBMSContext": "Server=LAPTOP-
H3OMMTNN\\SQLEXPRESS;Database=SimplonaTechEMS.Data;Trusted_Connection=True;MultipleA
ctiveResultSets=true;TrustServerCertificate=true;"
  }
}

using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using SimplonaTechEMS.Data;
using SimplonaTechEMS.Models;
```

```
namespace SimplonaTechEMS.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class DeptMastersController : ControllerBase
    {
        private readonly EmployeeDBMSContext _context;

        public DeptMastersController(EmployeeDBMSContext context)
        {
            _context = context;
        }

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

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

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

            return deptMaster;
        }

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

            _context.Entry(deptMaster).State = EntityState.Modified;

            try
            {
```

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

            return NoContent();
        }

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

            return CreatedAtAction("GetDeptMaster", new { id = deptMaster.DeptCode
}, deptMaster);
        }

        // DELETE: api/DeptMasters/5
        [HttpDelete("{id}")]
        public async Task<IActionResult> DeleteDeptMaster(int id)
        {
            if (_context.DeptMaster == null)
            {
                return NotFound();
            }
            var deptMaster = await _context.DeptMaster.FindAsync(id);
            if (deptMaster == null)
            {
                return NotFound();
            }

            _context.DeptMaster.Remove(deptMaster);
            await _context.SaveChangesAsync();

            return NoContent();
        }

        private bool DeptMasterExists(int id)
        {
```

```
            return (_context.DeptMaster?.Any(e => e.DeptCode ==
id)).GetValueOrDefault();
        }
    }
}
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using SimplonaTechEMS.Data;
using SimplonaTechEMS.Models;

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

        public EmpProfilesController(EmployeeDBMSContext 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();
            }

            _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 'EmployeeDBMSContext.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)
        {
            if (_context.EmpProfile == null)
            {
                return NotFound();
```

```
            }
            var empProfile = await _context.EmpProfile.FindAsync(id);
            if (empProfile == null)
            {
                return NotFound();
            }

            _context.EmpProfile.Remove(empProfile);
            await _context.SaveChangesAsync();

            return NoContent();
        }

        private bool EmpProfileExists(int id)
        {
            return (_context.EmpProfile?.Any(e => e.EmpCode ==
id)).GetValueOrDefault();
        }
    }
}
using System;
using Microsoft.EntityFrameworkCore.Migrations;

#nullable disable

namespace SimplonaTechEMS.Migrations
{
    /// <inheritdoc />
    public partial class SimponaTechDb : Migration
    {
        /// <inheritdoc />
        protected override void Up(MigrationBuilder migrationBuilder)
        {
            migrationBuilder.CreateTable(
                name: "DeptMaster",
                columns: table => new
                {
                    DeptCode = table.Column<int>(type: "int", nullable: false)
                        .Annotation("SqlServer:Identity", "1, 1"),
                    DeptName = table.Column<string>(type: "nvarchar(max)", nullable:
false)
                },
                constraints: table =>
                {
                    table.PrimaryKey("PK_DeptMaster", x => x.DeptCode);
                });

            migrationBuilder.CreateTable(
                name: "EmpProfile",
                columns: table => new
                {
                    EmpCode = table.Column<int>(type: "int", nullable: false)
                        .Annotation("SqlServer:Identity", "1, 1"),
                    DateOfBirth = table.Column<DateTime>(type: "datetime2",
nullable: false),
                    EmpName = table.Column<string>(type: "nvarchar(max)", nullable:
false),
```

```
                    Email = table.Column<string>(type: "nvarchar(max)", nullable:
false),
                    DeptCode = table.Column<int>(type: "int", nullable: false),
                    DeptMasterDeptCode = table.Column<int>(type: "int", nullable:
false)
                },
                constraints: table =>
                {
                    table.PrimaryKey("PK_EmpProfile", x => x.EmpCode);
                    table.ForeignKey(
                        name: "FK_EmpProfile_DeptMaster_DeptMasterDeptCode",
                        column: x => x.DeptMasterDeptCode,
                        principalTable: "DeptMaster",
                        principalColumn: "DeptCode",
                        onDelete: ReferentialAction.Cascade);
                });

            migrationBuilder.CreateIndex(
                name: "IX_EmpProfile_DeptMasterDeptCode",
                table: "EmpProfile",
                column: "DeptMasterDeptCode");
        }

        /// <inheritdoc />
        protected override void Down(MigrationBuilder migrationBuilder)
        {
            migrationBuilder.DropTable(
                name: "EmpProfile");

            migrationBuilder.DropTable(
                name: "DeptMaster");
        }
    }
}
```