Controllers:

Employees\_Controller:

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.Entity;

using System.Data.Entity.Infrastructure;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

using System.Web.Http.Description;

using EmployeeService.Data;

using EmployeeService.Models;

namespace EmployeeService.Controllers

{

public class EmployeesController : ApiController

{

private EmployeeServiceContext db = new EmployeeServiceContext();

// GET: api/Employees

public IQueryable<Employee> GetEmployees()

{

return db.Employees;

}

// GET: api/Employees/5

[ResponseType(typeof(Employee))]

public IHttpActionResult GetEmployee(int id)

{

Employee employee = db.Employees.Find(id);

if (employee == null)

{

return NotFound();

}

return Ok(employee);

}

// PUT: api/Employees/5

[ResponseType(typeof(void))]

public IHttpActionResult PutEmployee(int id, Employee employee)

{

if (!ModelState.IsValid)

{

return BadRequest(ModelState);

}

if (id != employee.Id)

{

return BadRequest();

}

db.Entry(employee).State = EntityState.Modified;

try

{

db.SaveChanges();

}

catch (DbUpdateConcurrencyException)

{

if (!EmployeeExists(id))

{

return NotFound();

}

else

{

throw;

}

}

return StatusCode(HttpStatusCode.NoContent);

}

// POST: api/Employees

[ResponseType(typeof(Employee))]

public IHttpActionResult PostEmployee(Employee employee)

{

if (!ModelState.IsValid)

{

return BadRequest(ModelState);

}

db.Employees.Add(employee);

db.SaveChanges();

return CreatedAtRoute("DefaultApi", new { id = employee.Id }, employee);

}

// DELETE: api/Employees/5

[ResponseType(typeof(Employee))]

public IHttpActionResult DeleteEmployee(int id)

{

Employee employee = db.Employees.Find(id);

if (employee == null)

{

return NotFound();

}

db.Employees.Remove(employee);

db.SaveChanges();

return Ok(employee);

}

protected override void Dispose(bool disposing)

{

if (disposing)

{

db.Dispose();

}

base.Dispose(disposing);

}

private bool EmployeeExists(int id)

{

return db.Employees.Count(e => e.Id == id) > 0;

}

}

}

Department\_controller:

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.Entity;

using System.Data.Entity.Infrastructure;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

using System.Web.Http.Description;

using EmployeeService.Data;

using EmployeeService.Models;

namespace EmployeeService.Controllers

{

public class DepartmentsController : ApiController

{

private EmployeeServiceContext db = new EmployeeServiceContext();

// GET: api/Departments

public IQueryable<Department> GetDepartments()

{

return db.Departments;

}

// GET: api/Departments/5

[ResponseType(typeof(Department))]

public IHttpActionResult GetDepartment(int id)

{

Department department = db.Departments.Find(id);

if (department == null)

{

return NotFound();

}

return Ok(department);

}

// PUT: api/Departments/5

[ResponseType(typeof(void))]

public IHttpActionResult PutDepartment(int id, Department department)

{

if (!ModelState.IsValid)

{

return BadRequest(ModelState);

}

if (id != department.Id)

{

return BadRequest();

}

db.Entry(department).State = EntityState.Modified;

try

{

db.SaveChanges();

}

catch (DbUpdateConcurrencyException)

{

if (!DepartmentExists(id))

{

return NotFound();

}

else

{

throw;

}

}

return StatusCode(HttpStatusCode.NoContent);

}

// POST: api/Departments

[ResponseType(typeof(Department))]

public IHttpActionResult PostDepartment(Department department)

{

if (!ModelState.IsValid)

{

return BadRequest(ModelState);

}

db.Departments.Add(department);

db.SaveChanges();

return CreatedAtRoute("DefaultApi", new { id = department.Id }, department);

}

// DELETE: api/Departments/5

[ResponseType(typeof(Department))]

public IHttpActionResult DeleteDepartment(int id)

{

Department department = db.Departments.Find(id);

if (department == null)

{

return NotFound();

}

db.Departments.Remove(department);

db.SaveChanges();

return Ok(department);

}

protected override void Dispose(bool disposing)

{

if (disposing)

{

db.Dispose();

}

base.Dispose(disposing);

}

private bool DepartmentExists(int id)

{

return db.Departments.Count(e => e.Id == id) > 0;

}

}

}

Model Classes:

Department.cs:

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.Linq;

using System.Web;

namespace EmployeeService.Models

{

public class Department

{

public int Id { get; set; }

[Required]

public string Name { get; set; }

public List<Employee> Employees { get; set; }

}

}

Employee.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.ComponentModel.DataAnnotations;

namespace EmployeeService.Models

{

public class Employee

{

public int Id { get; set; }

[Required]

public string FirstName { get; set; }

[Required]

public string LastName { get; set; }

[Required]

public float Salary { get; set; }

public int DepartmentId { get; set; }

public Department Department { get; set; }

}

}