4Create

C# .NET developer Test

Version 3. Last update 1. October 2021

Create API service which has 3 models: Employee, Company and SystemLog. Employees might belong to multiple Companies. Company might have multiple employees inside (NOTE: many-to-many relation).

Employee fields

- id
- title (enum: developer, manager, tester)
- email
- created at

Company fields

- id
- name
- created at

SystemLog fields

- resource type and its identifier (eg. Employee or Company)
- created at
- event (eg. “create”, “update”)
- resource attributes aka. changeset (for Employee - employee attributes, for company - company attributes)
- comment (eg. “new employee %employee.email% was created”)

You need to create 2 endpoints with the following logic:

<u>1) POST /api/employees - creates a new employee.</u>

Payload:
It accepts ONLY employee email, employee title AND company ids which new employee needs to be added to. Note that we should protect “created at” and “created by” fields to be set up from UI and we should fulfill it on server side.

Endpoint logic:

- New Employee record is created
- New SystemLog is created (“new employee was created”)

Validation:

- Employee email must be unique
- Employee title must be unique within a company (there could not be 2 developers or two managers inside one company)

<u>2) POST /api/companies - creates a new Company</u>

Payload:
The endpoint accepts ONLY company name AND array of employees to be created (if not exist) and added to the company. It could be an array of employee attributes (email, title) and Employee ids (if they are already created in the system).

Example:
```
employees: [
   { email: "new_employee@gmail.com" },
   { id: EXISTING_EMPLOYEE_ID }
]
```

Endpoint logic:

- New Company is created
- Employees specified in the request body are created (if they don’t already exist) and added to the company
- System logs for new company and new employees

Validation:

- Company name is unique
- Validation for employees records are same with endpoint # 1

Note: For endpoint #2 try reuse the code related to Employee creation written for endpoint#1

API specification:

- Use latest version of C# and .NET (C# 10 and .NET 6)
- Errors response must contain problematic key and message which says what's wrong with sent data for the corresponding key
- Feel free to use any libraries you want
- No frontend needed
- Use Mysql, Postgres or Mongo

PS. Use best practices, such as clean/onion/hexagonal architecture for project structure, create shareable components, imagine that the project has 100+ models and other project developers will follow your architecture for their endpoints.