توضیح پیاده سازی Task تعیین شده

محمدرضا رنجبر

فهرست

به نام خدا

اجرا توسط Docker

به منظور اجرای برنامه ابتدا باید دستور زیر را در پوشه ریشه پروژه اجرا نمایید (تمامی دستورات در فایل Commands.txt در همین پوشه قرار دارد)

```
docker build -f Pishtazan.Salaries\Dockerfile -t web_api  .
```

سپس از طریق دستور cd Pishtazan.Salaries به پوشه پروژه webApi رفته و دستور زیر را اجرا نمایید

```
docker-compose up
```

در انتها نیز از طریق http://localhost:5000/swagger/index.html می توانید به SwaggerUI دسترسی پیدا کنید.

Api های پیاده سازی شده

Api های در نظر گرفته شده برای برنامه به شرح زیر می باشند

۱-۱ درج یک رکورد جدید

برای این منظور باید از متد POST در مسیر /api/v1/{datatype}/Salaries اقدام شود. فرمت داده ی ارسالی بصورت JSON بوده و به فرمت زیر می باشد

```
{
    “data”: “”,
    “OverTimeCalculator”: “CalculatorA”
}
```

در پروژه اجرا شده تمامی Calculator ها یک interface واحد را پیاده سازی کرده اند و برنامه در شروع کار بصورت اتوماتیک تمامی کلاس هایی که این اینترفیس را پیاده سازی کرده اند را لود می کند لذا نیازی به رجیستر کردن دستی محاسبات مختلف نمی باشد و تنها کافیست در پروژه Pishtazan.Salaries.OvertimePolicies یک کلاس که اینترفیس IovertimePolicyCalculator را پیاده سازی کرده است را تعریف نمایید. با اجرای پروژه این کلاس بصورت اتوماتیک رجیستر می شود و

validation مربوط به نام این کلاس جدید هم در لایه کنترلر انجام می شود یعنی از این به بعد کاربر مجاز می باشد که این نوع محاسبه را نیز تعیین کند

Datatype طبق تسک تعیین شده ، می تواند یکی از نوع های Csv ، Custom و ... باشد که از طریق آن ، فرمت فیلد data به منظور Parse کردن تعیین می شود. فرمت Custom طبق سند پیاده سازی شده است اما فرمت Csv در سند به درستی بیان نشده است . در این فرمت عملا فیلد ها با ‘,’ از یکدیگر جدا می شوند به همین دلیل در این پیاده سازی فرض بر اینست که فیلد ها با کاما از هم جدا شده اند.

همانند IovertimePolicyCalculator ، فرمت های متفاوت نیز در سیستم بصورت اتوماتیک رجیستر می شوند و به منظور اضافه کردن فرمت جدید ، تنها کافیست اینترفیس IinputProvider پیاده سازی شود و در پروژه Pishtazan.Salaries قرار گیرد. پس از پیاده سازی ، فرمت جدید بصورت اتوماتیک در سیستم رجیستر می شود.

۱-۲ بهنگام سازی یک رکورد

این Api همانند درج رکورد می باشد با این تفاوت که با متد Put انجام می شود. مسیر همانند Api مربوط به درج رکورد برابر است با /api/v1/{datatype}/Salaries

۱-۳ حذف یک رکورد

برای این منظور باید نام و نام خانوادگی به همراه تاریخ دقیق اعلام شود. مسیر /api/v1/Salaries و متد Delete می باشد.

۱-۴ پرس و جو

برای پرس و جو با توجه به تسک تعیین شده دو حالت زیر وجود دارد

- گرفتن اطلاعات یک ماه. برای این منظور از طریق مسیر /api/v1/Salaries و ارسال درخواست GET و اعلام اطلاعات نام ،نام خانوادگی و تاریخ دقیق می توان اقدام کرد
- گرفتن اطلاعات یک بازه. برای این منظور از طریق مسیر /api/v1/Salaries/GetRange و اعلام نام ، نام خانوادگی ، تاریخ شروع بازه ،تاریخ پایان بازه ، اندیس صفحه (pagination) و اندازه صفحه می توان اقدام کرد.

اعتبارسنجی

در تسک اعلام شده صحبت زیادی در رابطه با Validation ها و الزاماتی که باید پیاده سازی شوند نشده است به همین دلیل من بعنوان نمونه موارد زیر را پیاده سازی کردم

- نام و نام خانوادگی باید حداقل دو کاراکتر و حداکثر ۲۰۰ کاراکتر باشند
- موارد مثل حقوق پایه و ... باید حداقل یک و حداکثر ۹۹۹۹۹۹۹۹۹ باشند
- در هر ماه تنها یک حقوق می توان ثبت کرد. بعنوان مثال اگر برای تاریخ ۱۴۰۲/۵/۱ حقوقی ثبت شده باشد ، و در درخواست تعریف رکورد دیگری در هر روز ماه ۵ وجود داشته باشد ، با خطا مواجه خواهد شد
- برای دریافت یک رکورد خاص ، تاریخ باید دقیق وارد شود
- برای بهنگام سازی یک رکورد ، نیازی نیست تاریخ دقیق باشد و تنها کافیست در همان ماه باشد.
- پروژه با زبان های فارسی و انگلیسی می باشد و این تنظیم از طریق SwaggerUI قابل انجام می باشد
- API Versioning

لیست پروژه ها

پروژه هایی که در این Solution وجود دارد به شرح زیر می باشند

- Pishtazan.Salaries به همراه پروژه تست. این پروژه حاوی پیاده سازی Web api می باشد. در این پروژه InputProvider ها تعریف شده اند که کار تبدیل فیلد Data و Parse آن را انجام می دهند و پیاده سازی نوع جدید داده باید در این پروژه انجام شود.
- Pishtazan.Salaries.Application به همراه تست. این پروژه لایه Application را تشکیل می دهد و Controller ها ، متد های مربوط به کلاس های سرویسی که در این پروژه تعریف شده اند را فراخوانی می کنند. این پروژه حاوی کلاس های Contract مربوط به Command و Query می باشد. علاوه بر این تعریف اینترفیس های Repository در این پروژه انجام شده است. پیاده سازی Repository هم بصورت InMemory و هم از طریق EF Core و Dapper انجام شده است
- Pishtazan.Salaries.Domain به همراه تست. این کلاس شامل ValueType ها (نام ، نام خانوادگی ، حقوق و ...) و Entity مربوط به Employee می باشد.

- Pishtazan.Salaries.Domain.Common به همراه تست. این پروژه شامل کلاس های Domain مشترک می باشد و برای جلوگیری از ارجاع حلقوی بین پروژه ها ایجاد شده است.
- Pishtazan.Salaries.Framework: کلاس پایه ValueType و DomainException در این پروژه تعریف شده است
- Pishtazan.Salaries.Infrastructure به همراه تست. این پروژه سرویس های عمومی مثل لود کردن اتوماتیک کلاس هایی که اینترفیس خاصی را پیاده سازی کرده اند و اساسا هر سرویس عمومی دیگری می باشد(Validation)
- Pishtazan.Salaries.OvertimePolicies به همراه تست. این پروژه طبق موارد بیان شده در تسک ایجاد شده است
- Pishtazan.Salaries.Persistence. پیاده سازی DbContext و همچنین Repository های تعریف شده در این پروژه وجود دارد. این پروژه شامل EF core و Dapper می باشد