Web-сервис

Реализовать WCF-сервис, хостящийся под IIS.   
API сервиса имеет 2 метода:

1. Принимает чек.
2. Принимает int значение, в ответ возвращает коллекцию чеков.

Interface репозитория для общения с БД:

1. Сохраняет чек в БД.
2. Запрашивает от БД коллекцию из n-последних чеков.

Реализация Interface репозитория для общения с БД MsSQL:

\* Считать, что реализацией процедур занимается SQL-разработчик, нужно только реализовать их вызов из Net.  
\*\* Connection String должна храниться в конфиге: *”data source=some\_server;initial catalog=some\_db;integrated security=SSPI;Application Name=TestApp"*

*[custom].[save\_cheques]*Входные параметры:

|  |  |  |
| --- | --- | --- |
| Параметр | Тип | Примечение |
| cheque\_id | uniqueidentifier |  |
| cheque\_number | nvarchar(50) |  |
| summ | money |  |
| discount | money |  |
| articles | nvarchar(max) | артикли чека, сджойненные символом “;”. |

*[custom].[get\_cheques\_pack]*Входные параметры:

|  |  |  |
| --- | --- | --- |
| Параметр | Тип | Примечение |
| pack\_size | integer |  |

Возвращаемые данные:   
коллекция объектов модели чека.

***Дополнительно***

***1.Фейковая реализация интерфейса репозитория для общения с БД:***

*1 метод принимает чек и сохраняет его в App\_Data.  
2 метод возвращает N чеков с рандомным ID и номером от 1 до N (остальные поля по-умолчанию).  
  
Есть возможность указать в конфиге сервиса какую реализацию репозитория использовать.*

***2. Для Senior+******уровня:*** *реализовать рядом ASP Web API сервис с прикрученным Swagger’ом.*

Windows-сервис

1. Мониторит папку на событие появление нового файла \*.txt файла (путь к папке указывается в конфиге).
2. Файл в себе должен содежать JSON чека.
   1. *(дополнительно)* Если в папке оказывается некорректный файл (не верное расширение, не десериализуется, etc), то он перемещается в папку Garbage (путь указывается в конфиге).
3. Если файл корректный, то чек отправлется в 1й метод WCF-сервиса.
4. *(дополнительно)* Файл перемещается в папку Complete
5. *(дополнительно)* Пути к Garbage и Complete могут быть и относительными, и абсолютными.

Модель чека

* Guid **Id**
* string **Number**
* decimal **Summ**
* decimal **Discount**
* string[] **Articles**

Требования

* Общение с БД через Dapper, БД MsSQL.
* Работа с JSON через NewtonSoft.JSON.
* Логирование через log4net, по логам должно быть понятно, что происходило (*логирование для wcf-сервиса – дополнительно*).
* Сторонние библиотеки подключаются через NuGet
* Функционал с пометкой «*дополнительно*» не обязателен к реализации.

К ознакомлению:

* <https://docs.microsoft.com/ru-ru/dotnet/csharp/programming-guide/inside-a-program/coding-conventions>
* <https://github.com/ktaranov/naming-convention/blob/master/C%23%20Coding%20Standards%20and%20Naming%20Conventions.md>

Критерии оценки:

На оценку тестового задания влияют срок выполнения, качество кода, возможность к масштабированию и расширению функционала, наличие проверок недопустимых значений, а так же реализация функционала с пометкой «*дополнительно*».