Тестовое задание на бэк разработчика .NET

# Легенда

Представим себе систему, обрабатывающую входящий http трафик, запрашивая бэкэнд и отдающая его результат в ответе. Общение с бэкэндом нужно спроектировать через брокера сообщений - т.е. апи бэкэнда не имеет синхронного апи типа “request-reply”.

# Задача

Нужно реализовать две версии, реализующие решение задачи.

В “наивной” (primitive) реализации системы все входящие запросы поступают в брокер и ожидают ответа от него, который и передают вызывающему.

В продвинутой (advanced) реализации требуется схлопывать идентичные запросы в один запрос брокеру. Идентичность определяется через функцию, извлекающую из запроса ключ. После процедуры схлопывания (т.е. отсылки ответов вызывающим), следующая пачка “таких же” запросов вновь порождает запрос брокеру.

# Реализация (замечания по имлементации)

В качестве брокера использовать директорию, где запрос (method, path) кладется в файл с именем “ключ запроса.req”

Ответ брокера ожидается в файле “ключ запроса.resp”,где первой строкой будет http код, а остальное - тело ответа для вызывающего. После вычитки ответа файлы ответа и запроса должны удаляться с диска сервисом.

Продумать как будет осуществляться смена реализации на “настоящие” брокеры, обладают возможностью уведомления клиентов о сообщении в канале (например - любой amqp брокер).

Продумать различные варианты ошибок, например: недоступность брокера (=нет каталога в нашей реализации), таймауты вызывающих (сценарий - два вызывающих с одним запросом с разницей во времени 1 минута, ответ через 1.5минуты от первого запроса, а таймаут вызывающих - 1 минута), падение сервиса и его рестарт (что делать с очередью?), ошибка в формате ответа, невозможность удаления ответа и/или запроса.

Расчет ключа для сохранения файла производить по формуле md5(http method + http path). Не путать этот ключ с ключем, используемым для схлопывания запросов - он остается на усмотрение разработчика.

Ответы могут быть достаточно сильно разнесены по времени с ответом, так, что какой-то из вызывающих, ожидающих ответа, может принудительно со своей стороны разорвать соединение не дождавшись ответа.

Настройки брокера в конфиг файле (директория хранилища).

# Как будет тестироваться, на что будет обращаться внимание

* Будет создаваться нагрузка в 10-20 тыс запросов, пачками по идентичности ключа, также в фоне будут создаваться различные ответы на пачки запросов и анализироваться результаты.
* Логи читаются с stdout
* Автоматически снимаем метрики по кол-ву потоков (системных) и по деградации по ним и памяти исходя из повышения нагрузки
* Использование примитивов синхронизации
* Общее исполнение задачи в коде, применение ООП, возможность расширения/замены “малой кровью” функционала (смена брокера, функций ключей, хранилища настроек и тп)
* (не критично) Наличие юнит тестов
* (не критично) наличие нагрузочных тестов и бенчмарков
* наличие мусора в коде