Дисциплина «Программирование»
Практическое задание №11
26 мая – 7 июня
«Сервис сообщений»

Процесс выполнения этого задания состоит из трёх частей:
1) реализация программы, согласно описанным в условии требованиям;
2) оценивание работ других студентов;
3) период «споров».

Период реализации программы:

После выдачи задания Вам необходимо выполнить его и загрузить архив (*.zip) с решением задачи (полностью заархивировать решение, созданное средой разработки) до крайнего срока. Работа должна выполняться студентом самостоятельно. В работе строго запрещается указывать ФИО, а также любую другую информацию, которая может выдать авторство работы. В случае выявления факта деанонимизации работы, работа может быть аннулирована.

Период взаимного оценивания:

После окончания срока, отведенного на реализацию программы, начинается период взаимного оценивания. Вам будет необходимо проверить пять работ других студентов, также выполнявших данное задание, согласно критериям оценивания. Проверка осуществляется анонимно: Вы не знаете, чью работу Вы проверяете, также, как и человек, кому принадлежит решение, не знает, кем была проверена его работа. Помимо оценки Вам необходимо указать комментарий к каждому из критериев. В случае, если Вы снижаете балл, необходимо подробно описать, за что именно была снижена оценка. Также рекомендовано писать субъективные комментарии, связанные с тонкостями программной реализации, предлагать автору работу более оптимальные на ваш взгляд решения. Снимать баллы за субъективные особенности реализации запрещено.

Период споров:

По окончании периода взаимного оценивания, в случае если Вы не согласны с оценкой, выставленной одним из проверяющих, Вы можете вступить с этим студентом в анонимный диалог с целью уточнения причин выставления оценки по тому или иному критерию. В случае, если проверяющий не ответил Вам, или вы не пришли к обоюдному решению об изменении оценки, Вы можете поставить флаг, и работа будет рассмотрена одним из преподавателей.

Флаги, поставленные без предварительного обсуждения с проверяющим или после окончания периода выставления флагов, будут отклонены.

Также допустима перепроверка Вашей работы преподавателем или ассистентом. В таком случае оценки других проверявших работу не учитываются.

Дедлайн загрузки работы: 7 июня 23:59
Дедлайн проверки: 11 июня 23:59
Дедлайн обсуждения оценок: 13 июня 23:59
Дедлайн выставления флагов: 14 июня 23:59

Возможность поздней сдачи работы в этом задании предоставляться не будет.

Оценивание:
$О_{итог} = 0{,}8 * О_{задание} + 0{,}2 * О_{проверки}$, где $О_{задание}$ – неокруглённая десятибалльная оценка за решение задания выставленная проверяющими с учётом возможной перепроверки преподавателем, а $О_{проверки}$ неокруглённая десятибалльная оценка, выставленная студентами, чьи работы вы проверяли. Также в случае, если преподаватель обнаружит, что Вы проверили работы некачественно к Вам могут быть применены санкции в виде штрафа до 3 десятичных баллов от $О_{итог}$ (в таком случае оценка вычисляется по формуле $О_{итог} = О_{задание} – Штраф$).

Задание:

Разработать ASP.Net Core приложение для работы с сервисом сообщений.

Данная программа должна реализовывать функционал для просмотра информации о пользователях и их сообщениях (в случае реализации дополнительных критериев – и их добавления в систему).

Пользователь характеризуется именем и адресом электронной почты. Электронная почта используется в качестве идентификатора, поэтому является уникальной (не может быть двух пользователей с одинаковым почтовым адресом).

Сообщение характеризуется темой, текстовым содержанием и, конечно же, отправителем и получателем.

Программа должна обладать следующей функциональностью:

1. Механизм чтения и записи списка пользователей и сообщений из/в соответствующих JSON-файлов. Список пользователей хранится в упорядоченном виде, т.е. сортируется лексикографически по **Email** (по

возрастанию).

Пользователи обладают двумя свойствами – string **UserName**, string **Email**.

Сообщения обладают четырьмя свойствами: **Subject**, **Message**, **SenderId**, **ReceiverId**.

Данный функционал должен использоваться обработчиками, перечисленными ниже.

2. Реализовать обработчик, доступный через метод POST для инициализации (т.е. первоначального заполнения) списка пользователей и сообщений с использованием Random.
3. Реализовать два обработчика, доступных только через метод GET:
   a) для получения информации о пользователе по его идентификатору (**Email**), учитывая, что при отсутствии пользователя код ответа должен быть HTTP 404 (not found);
   b) для получения всего списка пользователей.
4. Реализовать обработчик GET для получения списка сообщений по идентификатору отправителя и получателя.
5. Реализовать обработчики GET для получения списка сообщений:
   a) по идентификатору отправителя (получатель – любой);
   b) по идентификатору получателя (отправитель – любой).
6. Использовать Swagger для получения автоматически генерируемой документации по реализованным обработчикам.
7. [Дополнительный функционал] Предусмотреть возможность регистрации новых пользователей: создать обработчик (POST), добавляющий информацию о новом пользователе в систему (через поля формы или JSON – решать вам).
8. [Дополнительный функционал] Предусмотреть возможность отправки сообщений: создать обработчик (POST), добавляющий информацию о новом сообщении. Предусмотреть проверку того, что отправитель и получатель сообщения – зарегистрированные пользователи (т.е. существуют в списке пользователей). Вернуть сообщение об ошибке, если такая ситуация возникает (сообщение при этом не сохраняется).
9. [Дополнительный функционал] Доработать обработчик для получения списка всех пользователей (п. 2b), реализовав поддержку функционала для постраничной выборки – добавить поддержку параметров **Limit** и **Offset**:
   - int **Limit** – количество пользователей, которое необходимо вернуть (максимальное);
   - int **Offset** – порядковый номер пользователя, начиная с которого необходимо получать информацию (другими словами – количество пользователей, которые необходимо пропустить с начала списка).

При отрицательных значениях **Offset** или неположительных значениях **Limit** – возвращать HTTP 400 (bad request).

Список пользователей для определения порядкового номера сортируется лексикографически по **Email** (по возрастанию).

Пример: для условного списка {a, b, c, d, e, f, g}:
- применение параметров Limit=2 и Offset=3 вернет список {d, e};
- применение параметров Limit=3 и Offset=5 вернет список {f, g}.

Дополнительные требования (критерии оценивания):

1. Для проверки в систему PeerGrade должен быть загружен архив с решением. Ожидается, что проверка работы будет проводиться, в среде разработки Microsoft Visual Studio 2019, поэтому в случае выполнения задания с использованием другой среды разработки настоятельно рекомендуется проверить возможность открытия и запуска проекта в MS VS 2019.
2. Программа не должна завершаться аварийно (или уходить в бесконечный цикл) при любых входных данных (заметим, что статус ответа 500 является аварийным завершением). Недопустимо сохранять некорректные данные в файлы. Механизм уведомления о некорректных данных остается на ваше усмотрение.
3. Программа должна быть декомпозирована. Каждый из логических блоков должен быть выделен в отдельный метод. Не строго, но желательно, чтобы каждый метод по длине не превышал 40 строк.
4. Ответы обработчиков должны быть понятны. Разрешается добавить в ответ обработчиков дополнительное поле с информацией об ошибке, если таковая возникла. Ответы должны соответствовать запросам – ни при каких обстоятельствах нельзя выводить некорректный ответ (например, информацию не о запрашиваемом, а о другом пользователе).
5. Текст программы должен быть документирован. Необходимо писать, как комментарии перед методами, так и комментарии, поясняющие написанный внутри метода код. Названия переменных и методов должны быть на английском языке и отражать суть хранимых значений / выполняемых действий.
6. Работа должна быть выполнена в соответствии с заданием, критерии оценивания (“Rubric”) доступны для предпросмотра в системе “PeerGrade”.
7. Для получения отличной оценки более 8 баллов студенту необходимо реализовать дополнительный функционал, описанный в п.7 – п.9 задания.
8. Также оценивается общее впечатление, которое производит работа (как с точки зрения удобства обращения к обработчикам, так и с точки зрения написания кода программы). Эта часть оценки выставляется по усмотрению проверяющего.