**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**

**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Образовательная программа бакалавриата «Программная инженерия»

|  |  |  |
| --- | --- | --- |
| СОГЛАСОВАНО  Научный руководитель,  Преподаватель департамента  программной инженерии    \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.Н. Степанов  «15» апреля 2022 г. |  | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия»  профессор департамента  программной инженерии,  канд. техн. наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Шилов  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г. |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** |  | | **Клиент-серверное приложение для обмена навыками и услугами для студентов НИУ ВШЭ. Серверная часть.**  **Пояснительная записка**  **ЛИСТ УТВЕРЖДЕНИЯ**  **RU.17701729.06.02-01 81 01-1-ЛУ** | | |
|  |  | |
| Исполнитель:  студент группы БПИ191  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Царёв К.Д. /  «15» апреля 2022 г. | |
|  | | |
|  | |  |

**Москва 2022**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| УТВЕРЖДЕН  RU.17701729.06.02-01 81 01-1-ЛУ |  | |  | |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** |  | | **Пояснительная записка**  **RU.17701729.06.02-01 81 01-1**  **Листов 33** | | | | |
|  | |  | | |
|  | | |
|  | | | | |
|  | | | |  |

**Москва 2022**

**АННОТАЦИЯ**

В данном программном документе приведена пояснительная записка к серверной части приложения для обмена навыками и услугами для студентов НИУ ВШЭ.

В разделе «Введение» указано наименование программы, краткое наименование программы и документы, на основании которых ведется разработка.

В разделе «Назначение и область применения» указано функциональное назначение программы, эксплуатационное назначение программы и краткая характеристика области применения программы.

В разделе «Технические характеристики» содержатся следующие подразделы:

− постановка задачи на разработку программы;

− описание алгоритма и функционирования программы;

− описание метода организации входных и выходных данных;

− описание и обоснование выбора состава технических и программных средств.

В разделе «Ожидаемые технико-экономические показатели» указана предполагаемая потребность и экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами.

Настоящий документ разработан в соответствии с требованиями:

1) ГОСТ 19.101-77 Виды программ и программных документов;

2) ГОСТ 19.102-77 Стадии разработки;

3) ГОСТ 19.103-77 Обозначения программ и программных документов;

4) ГОСТ 19.104-78 Основные надписи;

5) ГОСТ 19.105-78 Общие требования к программным документам;

6) ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом;

7) ГОСТ 19.404-79 Пояснительная записка. Требования к содержанию и оформлению.

Изменения к Пояснительной записке оформляются согласно ГОСТ 19.603-78, ГОСТ 19.604-78.

Для лучшего понимания написанного в данной пояснительной записке настоятельно рекомендуется прочитать терминологию, представленную в приложении 1 настоящего документа.

ОГЛАВЛЕНИЕ

[1. ВВЕДЕНИЕ 5](#_Toc100011125)

[1.1. Наименование программы 5](#_Toc100011126)

[1.2. Документы, на основании которых ведется разработка 5](#_Toc100011127)

[2. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ 6](#_Toc100011128)

[2.1. Назначение программы 6](#_Toc100011129)

[2.1.1. Функциональное назначение 6](#_Toc100011130)

[2.1.2. Эксплуатационное назначение 6](#_Toc100011131)

[2.2. Краткая характеристика области применения 6](#_Toc100011132)

[3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 7](#_Toc100011133)

[3.1. Постановка задачи на разработку программы 7](#_Toc100011134)

[3.2. Описание процессов и алгоритмов системы 7](#_Toc100011135)

[3.2.1. Описание доступных пользователю функций программы и их связь между собой 7](#_Toc100011136)

[3.2.2. Описание программных классов 7](#_Toc100011137)

[3.2.3. Описание полей, методов и свойств 8](#_Toc100011138)

[3.2.4. Описание серверных ошибок в методах контроллеров 8](#_Toc100011139)

[3.2.5. Описание оптимизации хранения данных 8](#_Toc100011140)

[3.2.6. Описание процесса оценки пользователей 8](#_Toc100011141)

[3.2.7. Описание процесса присваивания пользователю статуса модератора 8](#_Toc100011142)

[3.2.8. Описание алгоритма цензуры текста 9](#_Toc100011143)

[3.2.9. Описание процесса удаления отзывов 9](#_Toc100011144)

[3.2.10. Описание алгоритма получения переписок 10](#_Toc100011145)

[3.2.11. Описание реализации внутреннего чата 10](#_Toc100011146)

[3.2.12. Описание алгоритма получения частичной истории сообщений в чате 10](#_Toc100011147)

[3.2.13. Описание процесса обмена сообщениями в реальном времени 10](#_Toc100011148)

[3.2.14. Описание процесса отправки уведомлений 11](#_Toc100011149)

[3.3. Описание и обоснование выбора метода организации входных и выходных данных 12](#_Toc100011150)

[3.3.1. Описание метода организации входных и выходных данных 12](#_Toc100011151)

[3.3.2. Обоснования выбора метода организации входных и выходных данных 12](#_Toc100011152)

[3.4. Описание ограничений, накладываемых на данные, хранимые в базе данных 12](#_Toc100011153)

[3.5. Описание и обоснование выбора состава технических и программных средств 12](#_Toc100011154)

[3.5.1. Состав технических и программных средств 12](#_Toc100011155)

[3.5.2. Обоснование выбора технических и программных средств 12](#_Toc100011156)

[4. ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ 13](#_Toc100011157)

[4.1. Предполагаемая потребность 13](#_Toc100011158)

[4.2. Экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами 13](#_Toc100011159)

[ПРИЛОЖЕНИЕ 1 14](#_Toc100011160)

[ТЕРМИНОЛОГИЯ 14](#_Toc100011161)

[ПРИЛОЖЕНИЕ 2 15](#_Toc100011162)

[СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 15](#_Toc100011163)

[ПРИЛОЖЕНИЕ 3 16](#_Toc100011164)

[ОГРАНИЧЕНИЯ, НАКЛАДЫВАЕМЫЕ НА ДАННЫЕ, 16](#_Toc100011165)

[ХРАНИМЫЕ В БАЗЕ ДАННЫХ 16](#_Toc100011166)

[ПРИЛОЖЕНИЕ 4 20](#_Toc100011167)

[ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ 20](#_Toc100011168)

[ПРИЛОЖЕНИЕ 5 21](#_Toc100011169)

[ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ПОЛЕЙ, МЕТОДОВ И СВОЙСТВ 21](#_Toc100011170)

[ПРИЛОЖЕНИЕ 6 27](#_Toc100011171)

[ОПИСАНИЕ И НАЗНАЧЕНИЕ МЕТОДОВ В СООТВЕТСТВИИ С REST API 27](#_Toc100011172)

[ПРИЛОЖЕНИЕ 7 29](#_Toc100011173)

[ОПИСАНИЕ СЕРВЕРНЫХ ОШИБОК, КОТОРЫЕ ВОЗВРАЩАЮТ HTTP МЕТОДЫ 29](#_Toc100011174)

[ПРИЛОЖЕНИЕ 8 31](#_Toc100011175)

[АВТОМАТИЧЕСКИ ГЕНЕРИРУЕМЫЕ ПИСЬМА ДЛЯ ПОЛЬЗОВАТЕЛЕЙ 31](#_Toc100011176)

[ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ 33](#_Toc100011177)

# ВВЕДЕНИЕ

## Наименование программы

Наименование программы – Серверная часть приложения для обмена навыками и услугами для студентов НИУ ВШЭ.

Наименование программы на английском языке – Server part of application for sharing skills and services for HSE students.

Краткое наименование программы – «Sharing Skills HSE».

## Документы, на основании которых ведется разработка

Основанием для разработки является учебный план подготовки бакалавров по направлению 09.03.04 "Программная инженерия" и утвержденная академическим руководителем тема курсового проекта.

# НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ

## Назначение программы

### Функциональное назначение

Глобально данная система имеет вид клиент-серверного приложения, где его конечный пользователь имеет возможность: внесения личных данных в своем профиле, поиска других пользователей, предложения обмена навыками и услугами, согласия или отказа от обменов, завершения выполненных обменов, просмотра действующих и завершенных обменов, переписки с другими пользователями во встроенном чате, оставления отзывов и оценок другим пользователям.

У серверной части разработанной системы следующее назначение:

* Организация внутренней бизнес-логики системы
* Реализация CRUD операций для взаимодействия клиентов с базой данных
* Передача своего API для использования клиентами

### Эксплуатационное назначение

Разработанная система может использоваться студентами НИУ ВШЭ, которые хотят произвести обмен навыками/услугами с другими студентами НИУ ВШЭ.

## Краткая характеристика области применения

«Sharing Skills HSE» – клиент-серверное приложение, с помощью которого можно осуществлять поиск людей, которые готовы обмениваться своими навыками и услугами друг с другом. Область применения системы – сфера знакомств и услуг.

# ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

## Постановка задачи на разработку программы

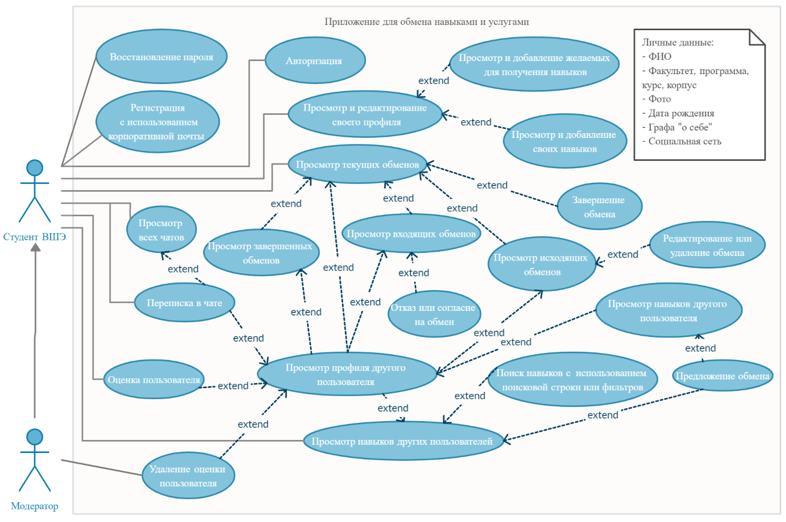
Данная часть системы должна удовлетворять следующим требованиям, изложенным в техническом задании к системе: к составу выполняемых функций, к интерфейсу, к надежности.

## Описание процессов и алгоритмов системы

Так как этот проект является продолжением курсового проекта с прошлого учебного года, то в данной пояснительной записке будут описаны результаты выполнения только за 2021–2022 учебный год. В частности, будет описан нововведенный функционал и некоторые оптимизационные изменения.

### Описание доступных пользователю функций программы и их связь между собой

На рис.1 представлены все доступные пользователю функции приложения. На нем представлен как старый функционал, так и новый, добавленный в текущем учебном году. Это сделано для общего понимания работы всей системы целиком.



*Рисунок 1*

### Описание программных классов

Первым делом рекомендуется ознакомиться с тем, какие программные классы были добавлены и изменены в ходе выполнения данного курсового проекта. С их названиями и функциональным описанием можно ознакомиться в приложении 4. Хочется еще раз обратить внимание на то, что в приложении представлены не все программные классы, а лишь та часть, которая была добавлена в текущем учебном году.

### Описание полей, методов и свойств

Также для более удобного и понятного прочтения следующих пунктов рекомендуется сначала ознакомиться с приложениями 5 и 6. В них описаны назначения полей, методов и свойств всех добавленных классов, а также дано описание методов контроллеров с точки зрения Rest API.

Доступ к этим методам контроллеров со стороны клиента осуществляется с использованием URL адреса (<https://sharingskillsapp.azurewebsites.net>) и приписок к нему, которые описаны в приложении 6.

### Описание серверных ошибок в методах контроллеров

И последнее, с чем стоит ознакомиться, это приложение 7. В нем можно увидеть коды серверных ошибок, возвращаемых различными HTTP-методами контроллеров, и описание того, по какой причине они возникают.

### Описание оптимизации хранения данных

Для того, чтобы данные занимали меньше места, и запросы к серверу от клиента выполнялись быстрее, было принято решение отказаться от хранения непосредственно самих зависимых сущностей в других сущностях. Вместо этого в настоящее время в сущностях хранятся лишь первичные ключи зависимых. По которым, в случае необходимости, можно достать и сам экземпляр зависимой сущности. В частности:

* в сущности навыка (Skill) хранится почта пользователя (UserMail)
* в сущности отзыва (Feedback) хранятся почта отправителя (SenderMail) и почта получателя (ReceiverMail)
* в сущности сообщения (Message) хранятся почта отправителя (SenderMail) и почта получателя (ReceiverMail)

### Описание процесса оценки пользователей

В случае, когда один пользователь хочет оценить другого, происходят следующие действия. Сначала пользователь на клиенте заполняет данные в полях ввода (оценку и комментарий). Не будем вдаваться в подробности, как это происходит, потому что это не имеет отношения к серверной части. Затем, опять же на клиенте, из всех данных формируется экземпляр сущности отзыва (Feedback), и выполняется POST-запрос (PostFeedback).

В этом методе происходит следующая последовательность действий. Сначала по автоматически сгенерированному ID отзыва проверяется, нет ли уже такого отзыва в базе данных. Затем по переданным почтам находятся отправитель и получатель, и проверяется, что они существуют в базе данных, и что это разные экземпляры сущности пользователя. Затем происходит автоматическая цензура комментария, вычисление и обновление новой средней оценки получателя. Отзыв добавляется в базу данных. В случае удовлетворения некоторых условий, пользователю присваивается статус модератора. Происходит уведомление получателя.

### Описание процесса присваивания пользователю статуса модератора

В данном приложении присутствует система отзывов и оценок пользователей, которая включает в себя написание текстовых комментариев. Для борьбы с неприемлемыми комментариями был реализован класс для автоматической цензуры текста. Но так как любую машину можно обойти, было принято решение, в помощь авто цензуре добавить возможность ручной цензуры самими пользователями. Для этого им необходимо получить статус модератора.

Для того, чтобы получить статус модератора, пользователю не нужно ничего делать. Достаточно просто пользоваться приложением и получать отзывы от других пользователей. В POST-запросе (PostFeedback) каждый раз при добавлении нового отзыва для получателя отзыва проверяются условия (количество оценок больше 5 и средняя оценка больше 3.5). При удовлетворении условий, у пользователя меняется флаг (IsModer) на true, и ему становятся доступны функции модератора.

### Описание алгоритма цензуры текста

Для реализации алгоритма цензуры текста был создан соответствующий программный класс (Censorship). Хочется обратить внимание, что цензура работает только на русский язык.

Для начала рассмотрим внутренние поля этого класса. В нем присутствует список запрещенных нецензурных слов, которые будут изменяться на звездочки. Но так как человек существо хитрое, он может изменять некоторые буквы на другие символы, чтобы цензура не срабатывала (например, а = @). Такой вариант тоже предусмотрен нашим алгоритмом, и для этого в классе присутствует словарь с вариантами написания одних и тех же букв.

Теперь перейдем к методу, выполняющему непосредственную цензуру текста (DoCensorship). В качестве входного параметра ему передается строка для цензуры. Сначала производятся подготовительные действия (текст переводится в нижний регистр, и все буквы, имеющие похожее написание, меняются на буквы из русского алфавита). Затем для каждого нецензурного слова из списка проходимся по отрезкам текста, равным длине этого слова. Если отрезок оказался запрещенным словом, то заменяем его на звездочки (\*\*\*).

Но так как люди могут слегка изменять сами слова (например, банан = бонан), необходимо как-то обрабатывать и такие ситуации. Для этого пришлось находить расстояние Левенштейна, которое показывает степень похожести слов. Таким образом, мы считаем, что слово запрещено, если расстояние Левенштейна меньше 25% длины слова.

Затем все измененные символы (за исключением звездочек) возвращаются в первоначальный вариант (заглавные буквы, специальные символы), и зацензуренный текст возвращается.

Данная автоматическая цензура используется в комментариях отзывов и в графе “О себе” личных данных пользователя.

### Описание процесса удаления отзывов

Процесс удаления отзыва реализован в двух видах:

* Удаление отзыва пользователем, оставившим этот отзыв. Данный вид удаления реализован с помощью DELETE-запроса (DeleteFeedbackUser). В этом методе происходит следующая последовательность действий. Сначала по переданному ID в базе данных находится отзыв. Затем происходит проверка на соответствие переданной в запрос почты и хранящейся в отзыве почты отправителя (SenderMail). Это сделано для того, чтобы пользователи не могли удалять чужие отзывы. Затем мы находим получателя отзыва, удаляем сам отзыв, изменяем значение средней оценки и сохраняем изменения в базе данных.
* Удаление отзыва модератором. Данный вид удаления реализован с помощью DELETE-запроса (DeleteFeedbackModer). В этом методе происходит следующая последовательность действий. Сначала по переданному ID в базе данных находится отзыв. Затем происходит проверка на соответствие переданной в запрос почты и хранящейся в отзыве почты получателя (ReceiverMail). Это сделано для того, чтобы модераторы не могли удалять отзывы, оставленные им. Затем мы находим получателя отзыва, удаляем сам отзыв, изменяем значение средней оценки и сохраняем изменения в базе данных.

### Описание алгоритма получения переписок

Для того, чтобы со стороны клиентов на экране чатов отображались все переписки конкретного пользователя, в серверную часть была добавлена сущность переписки (Chat). При первоначальном создании экземпляра переписки выполняется POST-запрос (PostChat), в котором передаются данные о пользователях, необходимые для отображения на экране чатов.

При отправке сообщения в сущности чата автоматически обновляются поля последнего сообщения (LastMessage) и времени отправки (SendTime), без дополнительных PUT-запросов.

Для получения всех чатов конкретного пользователя выполняется GET-запрос (GetChats) с указанием почты пользователя.

### Описание реализации внутреннего чата

Глобально, внутренний чат приложения будет организован следующей последовательностью действий:

* Пользователь заходит в конкретную переписку.
* На клиенте вызывается GET-запрос (GetMessages), который подгружает историю сообщений в данном конкретном чате.
* Клиент подключает пользователя к хабу чатов для непосредственного обмена сообщениями в режиме реального времени.
* Если второй пользователь из чата тоже в данный момент подключен к хабу, то происходит обмен сообщениями, и сообщения добавляются в базу данных POST-запросом (PostMessage).
* Если второй пользователь не подключен, то при отправке сообщений первым, второму будут приходить уведомления о новых сообщениях. А когда второй зайдет в чат, ему подгрузится история сообщений из базы данных.

При отправке любых сообщений они добавляются в базу данных с флагом прочитанности (IsRead) равным false. А затем, когда получатель прочитает их, с его клиента PUT-запросами (PutMessage) флаги прочитанности будут изменяться на true.

### Описание алгоритма получения частичной истории сообщений в чате

Для ускорения получения истории сообщений в чате было принято решение подгружать только 100 последних сообщений, чего должно быть более чем достаточно. А при необходимости подгружать следующие 100 и так далее.

Получение частичной истории сообщений в чате реализовано в GET-запросе (GetMessages). В этом методе в качестве входных параметров передаются почты двух участников чата, и число, означающее количество сотен сообщений, которые необходимо пропустить. Сначала по переданным почтам в базе данных находятся пользователи. (Они должны существовать, и должны быть разными.) Затем среди всех сообщений находятся сообщения только между этими двумя. Потом они сортируются по времени отправки, пропускаются n\*100 сообщений, где n – входной параметр, и возвращаются 100 сообщений.

### Описание процесса обмена сообщениями в реальном времени

Обмен сообщениями в чатах в реальном времени реализован с использованием библиотеки SignalR. Для этого был написан программный класс хаба чатов (ChatHub).

Сначала клиент должен подключиться к хабу с использованием данного URL адреса (<https://sharingskillsapp.azurewebsites.net/chat>). При подключении в список подключенных клиентов добавляется новая сущность подключенного пользователя (UserChatInfo) с заполненным ID подключения, но не заполненным значением почты. Для сопоставления ID подключения и почты пользователя, на клиенте вызывается соответствующий метод хаба (SetMail). Теперь при обмене сообщениями, пользователи ищутся в списке подключенных, и при наличии получателя в этом списке, ему отправляется сообщение. При выходе из чата (при отключении от хаба) пользователь удаляется из списка.

### Описание процесса отправки уведомлений

Изначально планировалось осуществлять отправку уведомлений в виде PUSH-уведомлений с использованием библиотеки PushSharp. Но из-за того, что клиент реализован только под iOS платформу, пришлось отказаться от такой идеи, потому что Apple разрешает отправлять PUSH-уведомления только через свои сервисы. А для этого необходимо завести Apple Developer аккаунт, подписка на который стоит 99 долларов в год, что является очень крупной суммой для учебного проекта.

Таким образом, отправка всех уведомлений происходит с помощью автоматически генерируемых писем, отправляемых с почты ([sharingskills@sharingskills.xyz](mailto:sharingskills@sharingskills.xyz)) через SMTP протокол с использованием библиотеки MailKit.

Уведомления в приложении отправляются в следующих случаях:

* Уведомления о предложении обмена
* Уведомления об одобрении обмена
* Уведомления об отказе в обмене
* Уведомления о завершении обмена
* Уведомления о новом сообщении в чате
* Уведомления о новом отзыве
* Уведомления о присваивании статуса модератора

Примеры автоматически генерируемых писем для отправки на почту можно посмотреть в приложении 8.

## Описание и обоснование выбора метода организации входных и выходных данных

### Описание метода организации входных и выходных данных

HTTP-запросы являются входными данными для сервера со стороны клиентов.

HTTP-ответы на запросы являются выходными данными с сервера для клиентов. Также выходными данными являются автоматически генерируемые электронные письма, отправляющиеся пользователям на корпоративную почту НИУ ВШЭ в домене edu.hse.ru в качестве уведомлений.

### Обоснования выбора метода организации входных и выходных данных

Реализация входных/выходных данных с помощью HTTP-запросов/ответов, проводимых между клиентами и сервером, является самым удобным и общепринятым способом для их взаимодействия.

Отправка уведомлений конечным пользователям с помощью автоматически генерируемых писем, отправляемых с почты ([sharingskills@sharingskills.xyz](mailto:sharingskills@sharingskills.xyz)) через SMTP протокол с использованием библиотеки MailKit обусловлена тем фактором, что для реализации общепринятых уведомлений в виде PUSH-уведомлений, необходимо платить 99 долларов в год, что является очень крупной суммой для учебного проекта.

## Описание ограничений, накладываемых на данные, хранимые в базе данных

В приложении 3 можно ознакомиться со всеми сущностями, хранящимися в базе данных, а также с ограничениями, накладываемыми на поля этих сущностей.

## Описание и обоснование выбора состава технических и программных средств

### Состав технических и программных средств

Для работы серверной части приложения необходим следующий состав технических средств:

* 1. 32-битная операционная система Windows
  2. Минимальный размер хранилища базы данных 32 гб
  3. Минимальный размер оперативной памяти 1 гб
  4. Двухъядерный процессор 2 ггц

### Обоснование выбора технических и программных средств

Описанный выше состав средств позволяет быстро взаимодействовать между собой клиентам и серверу, быстро отправлять письма по протоколу SMTP и работать без перебоев. Небольшой размер базы данных является следствием закрытости приложения только для студентов НИУ ВШЭ.

# ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

## Предполагаемая потребность

В настоящее время не существует инструмента, способного обеспечить студентов НИУ ВШЭ возможностью осуществления поиска людей, которые готовы обмениваться своими навыками и услугами друг с другом, в закрытом сообществе только для студентов НИУ ВШЭ. Этим фактором обуславливается предполагаемая потребность в продукте.

## Экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами

На момент написания пояснительной записки не было выявлено ни одного прямого аналога разработанной системы.

Из непрямых аналогов можно выделить следующие:

* Profi.ru
* Авито
* Бартер.рф

С помощью этих веб-сайтов и приложений также можно искать людей для обмена навыками и услугами. Но они не обеспечивают безопасность аудитории за счет закрытости сообщества только для студентов НИУ ВШЭ, что является главным преимуществом и особенностью разработанной системы (регистрация в ней осуществляется только по корпоративной почте НИУ ВШЭ в домене edu.hse.ru).

* Сообщества в социальной сети ВКонтакте (например, Подслушано ВШЭ)

Тут также можно искать людей для обмена навыками и услугами. И, кажется, что безопасность пользователей тоже обеспечена. Но это на первый взгляд. В реальности же туда совершенно спокойно может войти человек не из НИУ ВШЭ, потому что никакой проверки на принадлежность к ВУЗу не производится.

Еще одним минусом данного непрямого аналога является отсутствие собственного приложения и, как следствие, зависимость от политики самой социальной сети.

Таким образом, разработанная система является уникальным приложением, не только позволяющим людям обмениваться навыками и услугами, но и являющимся связующим звеном для студентов НИУ ВШЭ.

# ПРИЛОЖЕНИЕ 1

## ТЕРМИНОЛОГИЯ

Ниже приведен список необходимых терминов для ознакомления.

**Клиент** – аппаратный или программный компонент вычислительной системы, посылающий запросы серверу.

**Сервер** – программный компонент вычислительной системы, выполняющий сервисные (обслуживающие) функции по запросу клиента, предоставляя ему доступ к определённым ресурсам или услугам.

**Навык** – что-то, что умеет пользователь.

**Обмен** – условная договоренность между двумя пользователями о том, что они готовы поделиться друг с другом своими навыками.

**Корпоративная почта** – это электронная почта на домене вашей компании.

**API** (Application Programming Interface) – описание способов, которыми одна компьютерная программа может взаимодействовать с другой программой.

**HTTP** (HyperText Transfer Protocol) – протокол прикладного уровня передачи произвольных данных.

**SMTP** (Simple Mail Transfer Protocol) – это широко используемый [сетевой протокол](https://ru.wikipedia.org/wiki/IP), предназначенный для передачи [электронной почты](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BF%D0%BE%D1%87%D1%82%D0%B0).

**CRUD** – [акроним](https://ru.wikipedia.org/wiki/%D0%90%D0%BA%D1%80%D0%BE%D0%BD%D0%B8%D0%BC), обозначающий четыре базовые функции, используемые при работе с [базами данных](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85): создание (create), чтение (read), модификация (update), удаление (delete).

# ПРИЛОЖЕНИЕ 2

## СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. ГОСТ 19.101-77 Виды программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
2. ГОСТ 19.102-77 Стадии разработки. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.103-77 Обозначения программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.104-78 Основные надписи. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.105-78 Общие требования к программным документам. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.404-79 Пояснительная записка. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. ГОСТ 19.603-78 Общие правила внесения изменений. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
9. ГОСТ 19.604-78 Правила внесения изменений в программные документы, выполненные печатным способ
10. Клиент-сервер [Электронный ресурс] / Википедия. Режим доступа: <https://ru.wikipedia.org/wiki/Клиент_—_сервер>, свободный. (дата обращения: 20.11.2021).
11. Введение в REST API — RESTful веб-сервисы [Электронный ресурс] / Хабр. Режим доступа: <https://habr.com/ru/post/483202/>, свободный. (дата обращения: 09.10.2021).
12. Deploy an ASP.NET Core App with EF Core and SQL Server to Azure [Электронный̆ ресурс] / medium. Режим доступа: <https://medium.com/net-core/deploy-an-asp-net-core-app-with-ef-core-and-sql-server-to-azure-e11df41a4804>, свободный. (дата обращения: 15.12.2021).
13. Руководство по ASP.NET Core 5 [Электронный ресурс] / metanit. Режим доступа: <https://metanit.com/sharp/aspnet5>, свободный. (дата обращения: 29.12.2021).
14. PushSharp — простая работа с Push Notifications [Электронный ресурс] / Хабр. Режим доступа: <https://habr.com/ru/post/178775/>, свободный. (дата обращения: 09.01.2022).
15. iOS and Android Push Notifications with PushSharp [Электронный ресурс] / Matt Burke. Режим доступа: <https://www.mattburkedev.com/ios-and-android-push-notifications-with-pushsharp/>, свободный. (дата обращения: 12.01.2022).
16. Introduction to SignalR [Электронный ресурс] / Microsoft Docs. Режим доступа: <https://docs.microsoft.com/en-us/aspnet/signalr/overview/getting-started/introduction-to-signalr>, свободный. (дата обращения: 01.02.2022).

# ПРИЛОЖЕНИЕ 3

### ОГРАНИЧЕНИЯ, НАКЛАДЫВАЕМЫЕ НА ДАННЫЕ,

### ХРАНИМЫЕ В БАЗЕ ДАННЫХ

Таблица 3.1

Ограничения полей класса Skill.cs

|  |  |  |  |
| --- | --- | --- | --- |
| **Поля** |  |  |  |
| Имя | Тип | Ограничение | Сообщение при нарушении |
| Id | long | - | - |
| Status | int | Принимает значения от 0 до 2 | Некорректный статус навыка |
| Name | string | Длина не больше 50 символов | Слишком много символов |
| Description | string | Длина не больше 300 символов | Слишком много символов |
| Category | int | Принимает значения от 0 до 1 | Некорректная категория |
| Subcategory | int | Принимает значения от 0 до 21 | Некорректная подкатегория |
| UserMail | string | Состоит из латинских заглавных и строчных букв, цифр, подчеркиваний, и комбинации "@edu.hse.ru" на конце | Некорректный почтовый адрес |
| UserName | string | Длина от 2 до 40 символов | Имя должно быть от 2 до 40 символов |
| UserName | string | Состоит из латинских и кириллических заглавных и строчных букв | В имени могут присутствовать только буквы |
| UserSurname | string | Длина от 2 до 40 символов | Фамилия должна быть от 2 до 40 символов |
| UserSurname | string | Состоит из латинских и кириллических заглавных и строчных букв | В фамилии могут присутствовать только буквы |
| UserPhoto | byte[] | - | - |

Таблица 3.2

Ограничения полей класса Transaction.cs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | |  | | |
| Имя | Тип | | Ограничение | Сообщение при нарушении |
| Id | long | | - | - |
| Skill1 | string | | Длина не больше 50 символов | Слишком много символов |
| Skill2 | string | | Длина не больше 50 символов | Слишком много символов |
| Description | string | | Длина не больше 300 символов | Слишком много символов |
| SenderMail | string | | Состоит из латинских заглавных и строчных букв, цифр, подчеркиваний, и комбинации "@edu.hse.ru" на конце | Некорректный почтовый адрес отправителя |
| ReceiverMail | string | | Состоит из латинских заглавных и строчных букв, цифр, подчеркиваний, и комбинации "@edu.hse.ru" на конце | Некорректный почтовый адрес получателя |
| WhoWantMail | string | | Состоит из латинских заглавных и строчных букв, цифр, подчеркиваний, и комбинации "@edu.hse.ru" на конце | Некорректный почтовый адрес |
| Status | int | | Принимает значения от 0 до 2 | Некорректный статус обмена |
| Users | List<User> | | - | - |

Таблица 3.3

Ограничения полей класса User.cs

|  |  |  |  |
| --- | --- | --- | --- |
| **Поля** | | | |
| Имя | Тип | Ограничение | Сообщение при нарушении |
| Mail | string | Состоит из латинских заглавных и строчных букв, цифр, подчеркиваний, и комбинации "@edu.hse.ru" на конце | Некорректный почтовый адрес |
| Confirmation  CodeServer | int | - | - |
| Confirmation  CodeUser | int | Равенство с ConfirmationCodeServer | Неверный код подтверждения |
| Password | string | Длина от 6 до 40 символов | Пароль должен быть от 6 до 40 символов |
| Name | string | Длина от 2 до 40 символов | Имя должно быть от 2 до 40 символов |
| Name | string | Состоит из латинских и кириллических заглавных и строчных букв | В имени могут присутствовать только буквы |
| Surname | string | Длина от 2 до 40 символов | Фамилия должна быть от 2 до 40 символов |
| Surname | string | Состоит из латинских и кириллических заглавных и строчных букв | В фамилии могут присутствовать только буквы |
| BirthDate | Date  Time | Возраст должен быть от 13 до 100 лет | Некорректная дата рождения |
| Gender | int | Принимает значения от 0 до 2 | Некорректный пол |
| StudyingYearId | int | Принимает значения от 0 до 7 | Некорректный ID курса |
| MajorId | int | Принимает значения от 0 до 53 | Некорректный ID образовательной программы |
| Campus  LocationId | int | Принимает значения от 0 до 5 | Некорректный ID расположения корпуса |
| DormitoryId | int | Принимает значения от 0 до 12 | Некорректный ID общежития |
| About | string | Длина не больше 200 символов | Слишком много символов |
| Contact | string | Состоит из комбинации "vk.com/" или "t.me/" в начале, а также латинских заглавных и строчных букв, цифр, подчеркиваний, тире и точек | Некорректный формат ссылки |
| Photo | byte[] | - | - |
| Transactions | List  <Transaction> | - | - |
| Skills | List  <Skill> | - | - |
| Feedbacks | List<Feedback> | - | - |
| Chats | List  <Chat> | - | - |
| GradesCount | int | - | - |
| GradesSum | int | - | - |
| AverageGrade | double | - | - |
| IsModer | bool | - | - |

Таблица 3.4

Ограничения полей класса Message.cs

|  |  |  |  |
| --- | --- | --- | --- |
| **Поля** |  |  |  |
| Имя | Тип | Ограничение | Сообщение при нарушении |
| Id | long | - | - |
| SendTime | DateTime | - | - |
| Text | string | Длина не больше 300 символов | Слишком много символов |
| SenderMail | string | Состоит из латинских заглавных и строчных букв, цифр, подчеркиваний, и комбинации "@edu.hse.ru" на конце | Некорректный почтовый адрес отправителя |
| ReceiverMail | string | Состоит из латинских заглавных и строчных букв, цифр, подчеркиваний, и комбинации "@edu.hse.ru" на конце | Некорректный почтовый адрес получателя |

Таблица 3.5

Ограничения полей класса Feedback.cs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | |  | | |
| Имя | Тип | | Ограничение | Сообщение при нарушении |
| Id | long | | - | - |
| Grade | int | | Принимает значения от 0 до 5 | Некорректная оценка |
| Comment | string | | Длина не больше 300 символов | Слишком много символов |
| SenderMail | string | | Состоит из латинских заглавных и строчных букв, цифр, подчеркиваний, и комбинации "@edu.hse.ru" на конце | Некорректный почтовый адрес отправителя |
| ReceiverMail | string | | Состоит из латинских заглавных и строчных букв, цифр, подчеркиваний, и комбинации "@edu.hse.ru" на конце | Некорректный почтовый адрес получателя |

Таблица 3.6

Ограничения полей класса Chat.cs

|  |  |  |  |
| --- | --- | --- | --- |
| **Поля** | | | |
| Имя | Тип | Ограничение | Сообщение при нарушении |
| Id | long | - | - |
| LastMessage | string | Длина не больше 300 символов | Слишком много символов |
| SendTime | DateTime | - | - |
| Mail1 | string | Состоит из латинских заглавных и строчных букв, цифр, подчеркиваний, и комбинации "@edu.hse.ru" на конце | Некорректный почтовый адрес собеседника |
| Name1 | string | Длина от 2 до 40 символов | Имя должно быть от 2 до 40 символов |
| Name1 | string | Состоит из латинских и кириллических заглавных и строчных букв | В имени могут присутствовать только буквы |
| Surname1 | string | Длина от 2 до 40 символов | Фамилия должна быть от 2 до 40 символов |
| Surname1 | string | Состоит из латинских и кириллических заглавных и строчных букв | В фамилии могут присутствовать только буквы |
| Photo1 | byte[] | - | - |
| Mail2 | string | Состоит из латинских заглавных и строчных букв, цифр, подчеркиваний, и комбинации "@edu.hse.ru" на конце | Некорректный почтовый адрес собеседника |
| Name2 | string | Длина от 2 до 40 символов | Имя должно быть от 2 до 40 символов |
| Name2 | string | Состоит из латинских и кириллических заглавных и строчных букв | В имени могут присутствовать только буквы |
| Surname2 | string | Длина от 2 до 40 символов | Фамилия должна быть от 2 до 40 символов |
| Surname2 | string | Состоит из латинских и кириллических заглавных и строчных букв | В фамилии могут присутствовать только буквы |
| Photo2 | byte[] | - | - |
| Users | List<User> | - | - |

ПРИЛОЖЕНИЕ 4

## ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ

Таблица 4.1

Описание и функциональное назначение классов

|  |  |
| --- | --- |
| Класс | Назначение |
| MessagesController | Осуществляет взаимодействие клиента с сервером в отношении сущности сообщений. |
| FeedbacksController | Осуществляет взаимодействие клиента с сервером в отношении сущности отзывов. |
| ChatsController | Осуществляет взаимодействие клиента с сервером в отношении сущности переписок. |
| Message | Описывает сущность сообщения. |
| Feedback | Описывает сущность отзыва. |
| Chat | Описывает сущность переписки. |
| UserChatInfo | Предназначен для хранения информации о подключенных к чату клиентах. |
| ChatHub | Осуществляет обмен сообщениями в чатах в режиме реального времени. |
| Censorship | Производит автоматическую цензуру текста. |

# ПРИЛОЖЕНИЕ 5

## ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ПОЛЕЙ, МЕТОДОВ И СВОЙСТВ

Таблица 5.1

Описание полей, методов и свойств класса MessagesController.cs

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Поля** |  | |  | | |  |
| Имя | Модификатор доступа | | Тип | | | Назначение |
| \_context | private | | SharingSkillsContext | | | Контекст базы данных. |
| **Методы** |  | |  | | |  |
| Имя | Модификатор доступа | Тип | | Аргументы | Назначение | |
| MessagesController | public | void | | SharingSkillsContext context | Конструктор. | |
| GetMessages | public | Task<ActionResult  <IEnumerable<Message>>> | | - | Возвращает все сообщения. | |
| GetMessage | public | Task<ActionResult  <Message>> | | long id | Возвращает конкретное сообщение. | |
| GetMessages | public | Task<ActionResult<IEnumerable<Message>>> | | string mail1, string mail2,  int n | Возвращает 100 последних сообщений между двумя конкретными пользователями. | |
| PutMessage | public | Task<IActionResult> | | long id, Message message | Изменение конкретного сообщения. | |
| PostMessage | public | Task<ActionResult  <Message>> | | Message message | Добавление нового сообщения. | |
| DeleteMessage | public | Task<IActionResult> | | long id | Удаление сообщения. | |
| MessageExists | private | bool | | long id | Проверка наличия сообщения. | |

Таблица 5.2

Описание полей, методов и свойств класса FeedbacksController.cs

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Поля** | |  | |  | |  | |
| Имя | | Модификатор доступа | | Тип | | Назначение | |
| \_context | | private | | SharingSkillsContext | | Контекст базы данных. | |
| **Методы** | |  | |  | |  | |
| Имя | Модификатор доступа | | Тип | | Аргументы | | Назначение |
| FeedbacksController | public | | void | | SharingSkillsContext context | | Конструктор. |
| GetFeedback | public | | Task<ActionResult  <Feedback>> | | long id | | Возвращает конкретный отзыв. |
| GetFeedbacks | public | | Task<ActionResult  <IEnumerable  <Feedback>>> | | - | | Возвращает все отзывы. |
| PutFeedback | public | | Task  <IActionResult> | | long id,  Feedback feedback | | Изменение конкретного отзыва. |
| PostFeedback | public | | Task<ActionResult  <Feedback>> | | Feedback feedback | | Добавление нового отзыва. |
| DeleteFeedbackUser | public | | Task  <IActionResult> | | string mail, long id | | Удаление отзыва пользователем. |
| DeleteFeedbackModer | public | | Task  <IActionResult> | | string mail, long id | | Удаление отзыва модератором. |
| FeedbackExists | private | | bool | | long id | | Проверка наличия отзыва. |

Таблица 5.3

Описание новых полей, методов и свойств класса UsersController.cs

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Методы** | |  | |  | |  | |
| Имя | Модификатор доступа | | Тип | | Аргументы | | Назначение |
| GetUserFeedbacks | public | | Task<ActionResult<Feedback>> | | string mail | | Возвращает отзывы конкретного пользователя. |

Таблица 5.4

Описание новых полей, методов и свойств класса SharingSkillsContext.cs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Свойства** |  |  |  |  |
| Имя | Модификатор доступа | Тип | Доступ | Назначение |
| Feedbacks | public | DbSet<Feedback> | get, set | Свойство доступа к отзывам. |
| Messages | public | DbSet<Message> | get, set | Свойство доступа к сообщениям. |
| Chats | public | DbSet<Chat> | get, set | Свойство доступа к перепискам. |

Таблица 5.5

Описание полей, методов и свойств класса Message.cs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Свойства** |  |  |  |  |
| Имя | Модификатор доступа | Тип | Доступ | Назначение |
| Id | public | long | get, set | ID сообщения. |
| SendTime | public | DateTime | get, set | Время отправки. |
| Text | public | string | get, set | Текст. |
| SenderMail | public | string | get, set | Почта отправителя. |
| ReceiverMail | public | string | get, set | Почта получателя. |

Таблица 5.6

Описание полей, методов и свойств класса Feedback.cs

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Свойства** | |  | |  | |  | |
| Имя | Модификатор доступа | | Тип | | Доступ | | Назначение |
| Id | public | | long | | get, set | | ID отзыва. |
| Grade | public | | int | | get, set | | Оценка. |
| Comment | public | | string | | get, set | | Комментарий. |
| SenderMail | public | | string | | get, set | | Почта отправителя. |
| ReceiverMail | public | | string | | get, set | | Почта получателя. |

Таблица 5.7

Описание новых полей, методов и свойств класса User.cs

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Свойства** | |  | |  | |  | |
| Имя | Модификатор доступа | | Тип | | Доступ | | Назначение |
| Feedbacks | public | | List<Feedback> | | get, set | | Отзывы. |
| Chats | public | | List<Chat> | | get, set | | Переписки. |
| GradesCount | public | | int | | get, set | | Количество оценок. |
| GradesSum | public | | int | | get, set | | Сумма оценок. |
| AverageGrade | public | | double | | get, set | | Средняя оценка. |
| IsModer | public | | bool | | get, set | | Является ли пользователь модератором. |

Таблица 5.8

Описание полей, методов и свойств класса UserChatInfo.cs

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Свойства** | | | |  | | | | |  | | |  | | |
| Имя | | | Модификатор доступа | | | Тип | | | | Доступ | | | | Назначение |
| ConnectionID | | | public | | | string | | | | get, set | | | | ID подключения. |
| Mail | | | public | | | string | | | | get, set | | | | Почта пользователя. |
| **Методы** | |  | | | | |  | | | |  | | | |
| Имя | Модификатор доступа | | | | Тип | | | Аргументы | | | | | Назначение | |
| UserChatInfo | public | | | | void | | | string mail, string connectionId | | | | | Конструктор. | |

Таблица 5.9

Описание полей, методов и свойств класса ChatHub.cs

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Поля** |  | |  | |  |
| Имя | Модификатор доступа | | Тип | | Назначение |
| users | public | | List<UserChatInfo> | | Список подключенных  к чату пользователей. |
| **Методы** |  | |  | |  |
| Имя | Модификатор доступа | Тип | | Аргументы | Назначение |
| Send | public | Task | | string mail, Message message | Отправка сообщения. |
| SetMail | public | void | | string mail | Сопоставление почты с ID подключения. |
| OnConnectedAsync | public | Task | | - | Подключение клиента к хабу. |
| OnDisconnectedAsync | public | Task | | Exception e | Отключение клиента от хаба. |

Таблица 5.10

Описание полей, методов и свойств класса Censorship.cs

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Поля** |  | |  | |  |
| Имя | Модификатор доступа | | Тип | | Назначение |
| words | private | | List<string> | | Список нецензурных слов. |
| d | private | | Dictionary<char, List<char>> | | Варианты написания одних и тех же букв. |
| **Методы** |  | |  | |  |
| Имя | Модификатор доступа | Тип | | Аргументы | Назначение |
| LevenshteinDistance | private | int | | string firstWord, string secondWord | Определения расстояния Левенштейна для двух слов. |
| DoCensorship | public | string | | string phrase | Непосредственно зацензуривание. |

Таблица 5.11

Описание полей, методов и свойств класса Chat.cs

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Свойства** | |  | |  | | |  |
| Имя | Модификатор доступа | | Тип | | Доступ | Назначение | |
| Id | public | | long | | get, set | ID переписки | |
| LastMessage | public | | string | | get, set | Последнее сообщение. | |
| SendTime | public | | DateTime | | get, set | Время отправки последнего сообщения. | |
| Mail1 | public | | string | | get, set | Почта первого собеседника. | |
| Name1 | public | | string | | get, set | Имя первого собеседника. | |
| Surname1 | public | | string | | get, set | Фамилия первого собеседника. | |
| Photo1 | public | | byte[] | | get, set | Фотография первого собеседника. | |
| Mail2 | public | | string | | get, set | Почта второго собеседника. | |
| Name2 | public | | string | | get, set | Имя второго собеседника. | |
| Surname2 | public | | string | | get, set | Фамилия второго собеседника. | |
| Photo2 | public | | byte[] | | get, set | Фотография второго собеседника. | |
| Users | public | | List<User> | | get, set | Собеседники. | |

Таблица 5.12

Описание полей, методов и свойств класса ChatsController.cs

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Поля** |  | |  | | |  |
| Имя | Модификатор доступа | | Тип | | | Назначение |
| \_context | private | | SharingSkillsContext | | | Контекст базы данных. |
| **Методы** |  | |  | | |  |
| Имя | Модификатор доступа | Тип | | Аргументы | Назначение | |
| ChatsController | public | void | | SharingSkillsContext context | Конструктор. | |
| GetChat | public | Task<ActionResult  <Chat>> | | long id | Возвращает конкретную переписку. | |
| GetChats | public | Task<ActionResult  <IEnumerable<Chat>>> | | - | Возвращает все переписки. | |
| GetChats | public | Task<ActionResult  <IEnumerable<Chat>>> | | string mail | Возвращает все переписки конкретного пользователя. | |
| PutChat | public | Task<IActionResult> | | long id,  Chat chat | Изменение конкретной переписки. | |
| PostChat | public | Task<ActionResult  <Chat>> | | Chat chat | Добавление новой переписки. | |
| DeleteChat | public | Task<IActionResult> | | long id | Удаление переписки. | |
| ChatExists | private | bool | | long id | Проверка наличия переписки. | |

# ПРИЛОЖЕНИЕ 6

## ОПИСАНИЕ И НАЗНАЧЕНИЕ МЕТОДОВ В СООТВЕТСТВИИ С REST API

Таблица 6.1

Описание методов класса MessagesController.cs

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Методы** | |  | |  | |  |
| Имя | HTTP тип | | Пример URL-адреса | | Назначение | | |
| GetMessage | GET | | api/Messages/1 | | Возвращает конкретное сообщение. | | |
| GetMessages | GET | | api/Messages | | Возвращает все сообщения. | | |
| GetMessages | GET | | api/Messages/kdtsaryov@edu.hse.ru/  eoshtanko@edu.hse.ru/0 | | Возвращает 100 последних сообщений между двумя конкретными пользователями. | | |
| PutMessage | PUT | | api/Messages/1 | | Изменение конкретного сообщения. | | |
| PostMessage | POST | | api/Messages | | Добавление нового сообщения. | | |
| DeleteMessage | DELETE | | api/Messages/5 | | Удаление сообщения. | | |

Таблица 6.2

Описание методов класса FeedbacksController.cs

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Методы** | |  | |  | |  |
| Имя | HTTP тип | | Пример URL-адреса | | Назначение | | |
| GetFeedbacks | GET | | api/Feedbacks | | Возвращает все отзывы. | | |
| GetFeedback | GET | | api/Feedbacks/1 | | Возвращает конкретный отзыв. | | |
| PostFeedback | POST | | api/Feedbacks | | Добавление нового отзыва. | | |
| PutFeedback | PUT | | api/Feedbacks/1 | | Изменение отзыва. | | |
| DeleteFeedbackUser | DELETE | | api/Feedbacks/user/  eoshtanko@edu.hse.ru/1 | | Удаление отзыва пользователем. | | |
| DeleteFeedbackModer | DELETE | | api/Feedbacks/moder/  eoshtanko@edu.hse.ru/1 | | Удаление отзыва модератором. | | |

Таблица 6.3

Описание новых методов класса UsersController.cs

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Методы** | |  | |  | |  |
| Имя | HTTP тип | | Пример URL-адреса | | Назначение | | |
| GetUserFeedbacks | GET | | api/Users/eoshtanko@edu.hse.ru  /feedbacks | | Возвращает отзывы конкретного пользователя. | | |

Таблица 6.4

Описание методов класса ChatsController.cs

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Методы** | |  | |  | |  |
| Имя | HTTP тип | | Пример URL-адреса | | Назначение | | |
| GetChat | GET | | api/Chats/1 | | Возвращает конкретную переписку. | | |
| GetChats | GET | | api/Chats | | Возвращает все переписки. | | |
| GetChats | GET | | api/Chats/kdtsaryov@edu.hse.ru/  user | | Возвращает все переписки конкретного пользователя. | | |
| PutChat | PUT | | api/Chats/1 | | Изменение конкретной переписки. | | |
| PostChat | POST | | api/Chats | | Добавление новой переписки. | | |
| DeleteChat | DELETE | | api/Chats/5 | | Удаление переписки. | | |

# ПРИЛОЖЕНИЕ 7

## ОПИСАНИЕ СЕРВЕРНЫХ ОШИБОК, КОТОРЫЕ ВОЗВРАЩАЮТ HTTP МЕТОДЫ

Таблица 7.1

Описание ошибок в методах класса MessagesController.cs

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Методы** | |  | |  | |  |
| Имя | HTTP тип | | Ошибки | | Описание | | |
| GetMessage | GET | | 404 NOT FOUND | | Сообщения с таким ID нет. | | |
| GetMessages | GET | | - | | - | | |
| GetMessages | GET | | 400 BAD REQUEST | | Нет такого первого пользователя или второго пользователя, или первый равен второму. | | |
| PutMessage | PUT | | 400 BAD REQUEST | | Переданный ID не равен ID сообщения. | | |
| PutMessage | PUT | | 404 NOT FOUND | | Сообщения с таким ID нет. | | |
| PostMessage | POST | | 400 BAD REQUEST | | Такое сообщение уже есть. | | |
| PostMessage | POST | | 400 BAD REQUEST | | Нет такого отправителя или получателя, или отправитель равен получателю. | | |
| PostMessage | POST | | 400 BAD REQUEST | | Нет такой переписки. | | |
| DeleteMessage | DELETE | | 404 NOT FOUND | | Сообщения с таким ID нет. | | |

Таблица 7.2

Описание ошибок в методах класса FeedbacksController.cs

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Методы** | |  | |  | |  |
| Имя | HTTP тип | | Ошибки | | Описание | | |
| GetFeedbacks | GET | | - | | - | | |
| GetFeedback | GET | | 404 NOT FOUND | | Отзыва с таким ID нет. | | |
| PostFeedback | POST | | 400 BAD REQUEST | | Такой отзыв уже есть. | | |
| PostFeedback | POST | | 400 BAD REQUEST | | Нет такого отправителя или получателя, или отправитель равен получателю. | | |
| PutFeedback | PUT | | 400 BAD REQUEST | | Переданный ID не равен ID отзыва. | | |
| PutFeedback | PUT | | 404 NOT FOUND | | Отзыва с таким ID нет. | | |
| DeleteFeedbackUser | DELETE | | 404 NOT FOUND | | Отзыва с таким ID нет. | | |
| DeleteFeedbackUser | DELETE | | 400 BAD REQUEST | | Попытка удаления чужого отзыва. | | |
| DeleteFeedbackModer | DELETE | | 404 NOT FOUND | | Отзыва с таким ID нет. | | |
| DeleteFeedbackModer | DELETE | | 400 BAD REQUEST | | Попытка удаления отзыва о себе. | | |

Таблица 7.3

Описание ошибок в новых методах класса UsersController.cs

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Методы** | |  | |  | |  |
| Имя | HTTP тип | | Ошибки | | Описание | | |
| GetUserFeedbacks | GET | | 404 NOT FOUND | | Пользователя с такой почтой нет. | | |
| GetUserFeedbacks | GET | | 404 NOT FOUND | | У пользователя с такой почтой нет отзывов. | | |

Таблица 7.4

Описание ошибок в методах класса ChatsController.cs

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Методы** | |  | |  | |  |
| Имя | HTTP тип | | Ошибки | | Описание | | |
| GetChat | GET | | 404 NOT FOUND | | Переписки с таким ID нет. | | |
| GetChats | GET | | - | | - | | |
| GetChats | GET | | 400 BAD REQUEST | | Нет такого пользователя. | | |
| PutChat | PUT | | 400 BAD REQUEST | | Переданный ID не равен ID переписки. | | |
| PutChat | PUT | | 404 NOT FOUND | | Переписки с таким ID нет. | | |
| PostChat | POST | | 400 BAD REQUEST | | Такая переписка уже есть. | | |
| PostChat | POST | | 400 BAD REQUEST | | Нет такого первого собеседника или второго собеседника, или собеседники равны. | | |
| DeleteChat | DELETE | | 404 NOT FOUND | | Переписки с таким ID нет. | | |

# ПРИЛОЖЕНИЕ 8

## АВТОМАТИЧЕСКИ ГЕНЕРИРУЕМЫЕ ПИСЬМА ДЛЯ ПОЛЬЗОВАТЕЛЕЙ

1. Письмо при регистрации нового пользователя:

Тема –

Код подтверждения регистрации

Само письмо –

Здравствуйте!

Спасибо за регистрацию в Обмене Навыками

Ваш код подтверждения - 5678

1. Письмо для напоминания забытого пароля

Тема –

Напоминание пароля

Само письмо –

Здравствуйте!

В Обмене Навыками была нажата кнопка "Забыли пароль?"

Ваш пароль – simSIM2237

1. Оповещение о новом обмене

Тема –

Новый обмен

Само письмо –

Екатерина Штанько предложил(а) Вам новый обмен.

Зайдите в приложение "Обмен навыками", чтобы узнать детали.

1. Оповещение об отказе в обмене

Тема –

Отказ в обмене

Само письмо –

Екатерина Штанько отказал(а) Вам в обмене.

Зайдите в приложение "Обмен навыками", чтобы узнать детали.

1. Оповещение о принятии обмена

Тема –

Принятый обмен

Само письмо –

Екатерина Штанько принял(а) Ваш обмен.

Зайдите в приложение "Обмен навыками", чтобы узнать детали.

1. Оповещение о завершении обмена

Тема –

Обмен завершен

Само Письмо –

Екатерина Штанько завершил(а) обмен.

Зайдите в приложение "Обмен навыками", чтобы узнать детали.

1. Оповещение о новом сообщении в чате

Тема –

Новое сообщение

Само Письмо –

Вам отправили новое сообщение.

Зайдите в приложение "Обмен навыками", чтобы узнать детали.

1. Оповещение о присвоении статуса модератора

Тема –

Статус модератора

Само Письмо –

Поздравляем!

Вам стали доступны функции модератора в приложении "Обмен навыками".

Теперь Вы можете удалять неприличные отзывы пользователей приложения.

1. Оповещение о новом отзыве

Тема –

Новый отзыв

Само письмо –

Екатерина Штанько оставил(а) Вам новый отзыв.

Зайдите в приложение "Обмен навыками", чтобы узнать детали.

# ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц в докум.) | № документа | Входящий № сопроводительного докум. и дата | Подп. | Дата |
| Изм. | Измененных | Замененных | Новых | Аннулированных |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |