

Федеральное государственное автономное образовательное учреждение
высшего образования
«Омский государственный технический университет»

Кафедра «Автоматизированные системы обработки информации и
управления»

ОТЧЕТ
по лабораторной работе №4
на тему
«Повышение качества программной документации»

Выполнил: ст. гр. ПИН-202

Горшенин Л.И.

Проверила: ст. преподаватель

Фатеева А.С.

Омск 2023

При выполнении лабораторной работы были исправлены следующие критерии в документации:

1. Полнота: была описана вся функциональность, избегая пропусков в описании.
2. Указание на необратимость действий: были описаны необратимые последствия при выполнении определенных действий.
3. Подтверждение ожидаемого: указан ожидаемый результат после выполнения определенных действий
4. Описание последствий отсутствия действий: добавлена информация о том, что может произойти, если пользователь не выполнит требуемые действия
5. Термины и их значение: добавлено более подробное описание терминов
6. Доступность пользователю: обеспечена максимальная понятность документации для любой целевой аудитории
7. Адаптивность к поиску: в данной программе это не предусмотрено

```

/// <summary>
/// Создание экземпляра класса <see cref="Comment"/> и добавление его в базу данных
/// </summary>
/// <remarks>
/// Создает комментарий <see cref="Comment"/>, добавляет его в базу данных и отображает в представлении
/// <para>
/// <c>Ожидается:</c>
/// <para>Данные о комментарии сохраняются в БД, сервер генерирует и отправляет клиенту ответ.</para>
/// </para>
/// <para>
/// <c>Последствия:</c>
/// <para>Клиент отображает комментарий в представлении.</para>
/// </para>
/// </remarks>
/// <param name="text">Текст комментария</param>
/// <param name="userId">Уникальный идентификатор пользователя, написавшего текущий комментарий</param>
/// <param name="postId">Уникальный идентификатор поста, под которым был написан текущий комментарий</param>
Ссылка: 1
public async void CreateComment(string text, int userId, int postId)
{
    var comment = new Comment {
        Text = text,
        UserId = userId,
        PostId = postId
    };
    await InsertComment(comment);
    OnShowComments?.Invoke();
}

/// <summary>
/// Удаление экземпляра
/// </summary>
/// <param name="currentId">Уникальный идентификатор комментария</param>
/// <param name="selectedComment">экземпляр класса <see cref="Comment"/>, который будет удален</param>

```

void CommentsCollection.CreateComment(string text, int userId, int postId)
Создание экземпляра класса Comment и добавление его в базу данных

Создает комментарий Comment, добавляет его в базу данных и отображает в представлении

Ожидается:

Данные о комментарии сохраняются в БД, сервер генерирует и отправляет клиенту ответ.

Последствия:

Клиент отображает комментарий в представлении.

Рисунок 1 – Пример измененной документации метода

```

/// <summary>
/// Класс комментария
/// </summary>
/// <remarks>
/// <para>Экземпляр класса комментария.</para>
/// <para>Модель, которая будет сохранена в базе данных.</para>
/// <para>
/// <c>Содержит в себе:</c>
/// <list type="table">
/// <item><see cref="Text"/> - <see langword="string"/></item>
/// <item><see cref="Id"/> - <see langword="int"/></item>
/// <item><see cref="Time"/> - <see langword="dateTime"/></item>
/// <item><see cref="UserId"/> - <see langword="int"/></item>
/// <item><see cref="PostId"/> - <see langword="int"/></item>
/// </list>
/// </para>
/// </remarks>
/// <param name="currentId">Уникальный идентификатор комментария</param>
/// <param name="text">Текст комментария</param>
/// <param name="time">Время написания комментария</param>
/// <param name="userId">Уникальный идентификатор пользователя, написавшего текущий комментарий</param>
/// <param name="postId">Уникальный идентификатор поста, под которым был написан текущий комментарий</param>
Ссылка: 2
public Comment(int currentId, string text, DateTime time, int userId, int postId)
{
    Id = currentId;
    Text = text;
    Time = time;
    UserId = userId;
    PostId = postId;
}

/// <summary>
/// Текст комментария
/// </summary>
Ссылка: 5
public string Text { get; set; }

/// <summary>
/// Уникальный идентификатор комментария

```

Comment.Comment(int currentId, string text, DateTime time, int userId, int postId)
Класс комментария

Экземпляр класса комментария.

Модель, которая будет сохранена в базе данных.

Содержит в себе:

Comment.Text - string

Comment.Id - int

Comment.Time - dateTime

Comment.UserId - int

Comment.PostId - int

Рисунок 2 – Пример измененной документации класса

```

/// <value>
/// Текст комментария - <see langword="string"/>
/// </value>
/// <remarks>
/// Текст комментария, который будет сохранен в базе данных.
/// <para>В дальнейшем может быть изменен путем вызова метода <see cref="CommentsCollection.UpdateComment(int, string, Comment)"/></para>
/// </remarks>
Ссылка: 5
public string Text { get; set; }

/// <summary>
/// Уникальный идентификатор
/// </summary>
Ссылка: 9
public int Id { get; set; }
/// <summary>

```

string Comment.Text { get; set; }

Текст комментария, который будет сохранен в базе данных.

В дальнейшем может быть изменен путем вызова метода [CommentsCollection.UpdateComment\(int, string, Comment\)](#)

Значение:

Текст комментария - string

Рисунок 3 – Пример измененной документация свойства