­Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Базы Данных (БД)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту на тему

Веб-Форум

БГУИР КП 1-40 01 01 020 ПЗ

Выполнил

студент: гр. 151004 Король Н. А.

Руководитель: Новицкая Н. В.

Минск 2024

содержание

[Введение 3](#_Toc165933640)

[1 Аналитический обзор литературы 4](#_Toc165933641)

[1.1 Обзор существующих аналогов 4](#_Toc165933642)

[1.2 Требования к проектируемому программному средству 6](#_Toc165933643)

[1.3 Постановка задачи 8](#_Toc165933644)

[2 Анализ требований к программному средству и разработка функциональных требований 9](#_Toc165933645)

[2.1 Обоснование выбора СУБД и иных средств разработки 9](#_Toc165933646)

[2.2 Спецификация функциональных требований 10](#_Toc165933647)

[2.3 Учитываемые функции и бизнес-процессы 10](#_Toc165933648)

[2.4 Ключевые особенности базы данных 10](#_Toc165933649)

[2.5 Пользователи системы и их роли 11](#_Toc165933650)

[2.6 Параметры поиска информации 11](#_Toc165933651)

[2.7 Бизнес-процессы, связанные с реализацией информационной системы 11](#_Toc165933652)

[3 Инфологическая модель предметной области 12](#_Toc165933653)

[3.1 Описание моделей 12](#_Toc165933654)

[3.2 Сущности и связи 13](#_Toc165933655)

[3.3 Особенности нормализации 16](#_Toc165933656)

[4 Подробное описание бизнес-логики 17](#_Toc165933657)

[4.1 Триггеры 17](#_Toc165933658)

[4.2 Основные алгоритмы ПС 18](#_Toc165933659)

[5 Тестирование базы данных и приложений системы 21](#_Toc165933660)

[5.1 Тестирование триггеров 21](#_Toc165933661)

[5.2 Тестирование программного средства 21](#_Toc165933662)

[Заключение 25](#_Toc165933663)

[Список использованной литературы 26](#_Toc165933664)

[Приложение А 27](#_Toc165933665)

# Введение

В настоящее время сфера веб-форумов стремительно развивается, предоставляя пользователям огромное количество площадок для общения, обмена опытом и обсуждения различных тем. Однако, с ростом популярности и активности пользователей, возникают важные вопросы, связанные с безопасностью, модерацией контента и архитектурой форумов. В этом контексте создание надежных и гибких веб-форумов становится ключевой задачей для разработчиков и администраторов.

Можно назвать ряд причин, почему полезно будет рассмотреть такой проект, как Веб-Форум. Вот некоторые причины:

1. **Обмен знаниями и опытом:** Веб-форумы собирают людей с общими интересами. Здесь можно задавать вопросы, делиться своим опытом и узнавать новое от других участников.
2. **Решение проблем:** Если у вас возникли технические или другие вопросы, веб-форумы – отличное место для поиска ответов. Участники помогут вам разрешить проблемы и поделиться своими находками.
3. **Создание сообщества:** Форумы объединяют людей, которые разделяют общие интересы. Здесь можно найти единомышленников, обсудить темы и даже завести новых друзей.
4. **Анонимность и конфиденциальность:** Веб-форумы позволяют общаться анонимно, что может быть удобно, если вы не хотите раскрывать свою личность.
5. **Бесплатный доступ к информации:** Форумы предоставляют доступ к обширным знаниям и ресурсам без необходимости платить за подписку или контент.

**Особенности предметной области:**

Веб-Форум также имеет свои особенности и требования. Одним из ключевых аспектов является обеспечение безопасности и защиты данных пользователей. В наше время, когда цифровые технологии сталкиваются с угрозами кибератак и нарушениями конфиденциальности, обеспечение безопасности личной информации и данных пользователей становится неотъемлемой частью разработки и поддержки веб-форумов.

Кроме того, производительность и гибкость архитектуры сервиса также играют важную роль. В условиях растущего числа пользователей и потока данных, необходимо обеспечить плавный и быстрый доступ к информации, а также возможность масштабирования сервиса для удовлетворения потребностей растущей аудитории.

В данном проекте основное внимание и усилия будут направлены на разработку надежной базы данных, обеспечивающей безопасное хранение пользовательской информации, а также на создание гибкой и производительной архитектуры Веб-Форума.

# Аналитический обзор литературы

## Обзор существующих аналогов

### BMW Club

BMW Club - это Веб-Форум основной темой которого является автомобили.

Изображение выглядит как текст, программное обеспечение, веб-страница, Значок на компьютере

Автоматически созданное описание

Рисунок 1.1 – BMW Club

Достоинства:

* простой и понятный интерфейс;
* общедоступная статистика сайта;
* разбиение форумов на категории.

Недостатки:

* узконаправленный форум.

### 4PDA

4PDA — это один из самых популярных веб-форумов, занимает 2-е место поп популярности по данным сайта Infoselection. Основная тема форума “Смартфоны”

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание

Рисунок 1.2 – 4PDA

Достоинства:

* простой и понятный интерфейс;
* популярный форум;
* удобная статистика пользователей;
* разбиение форумов на категории.

Недостатки:

Явных недостатков обнаружено не было.

## Требования к проектируемому программному средству

### Назначение разработки

Назначение разработки базы данных для веб-форума, состоит в обеспечении удобного и многофункционального способа доступа к добавляемым пользователями постами.

Веб-форум предоставляет пользователям возможность создавать посты, а также комментировать уже существующие. Веб-форум включает возможность подписываться на пользователей и просматривать написанные ими комментариями, постами и их подписей на других пользователей.

База данных, в свою очередь, играет ключевую роль в организации и хранении информации о постах, пользователей, форумах, тем, комментариев информации о пользователе и других связанных данных. Она предназначена для эффективного учета данных и их данных.

Целью разработки сервиса и базы данных является обеспечение удобства и функциональности для пользователей, а также эффективного управления и доступа к информации.

### Состав выполняемых функций

В рамках данного курсового проекта планируется разработка базы данных для Веб-Форума. Основные функциональные возможности базы данных:

* Авторизация и регистрация пользователей;
* Редактирования профиля пользователя;
* Просмотр списка форумов;
* Просмотр списка тем выбранного форума;
* Просмотр списка постов выбранной темы;
* Добавления поста в выбранную тему;
* Добавление комментариев посту;
* Просмотр комментариев поста;
* Просмотр списка пользователей;
* Добавления закладок для тем;
* Просмотр постов, комментариев и закладок выбранного пользователя;
* Подпись на выбранного пользователя.

### Входные данные

Информация о:

* Пользователях;
* Постах;
* Форумах;
* Тем;
* Информации о пользователе;
* Комментарии;
* Закладки.

### Выходные данные

Информация о пользователе, ФИО, страна, город, возраст, написанные пользователем посты и комментарии, закладки пользователя и его подписки.

Написанные пользователями посты их содержание и название.

Комментарии, написанные пользователями для конкретного поста.

### Требования к составу и параметрам технических и программных средств

В состав технических средств входит сервер, включающий в себя:

* процессор Pentium 3 и выше;
* 256 Мбайт оперативной памяти;
* 512 Мбайт свободного пространства на жестком диске.

### Требования к информационной и программной совместимости

Программа рассчитывается на работу на сервере и требует иметь .NET 8 на устройстве сервера. А для клиентской стороны достаточно наличия браузера.

### Обоснование выбора языка и сред разработки

Для разработки программного средства были выбраны язык C# и среда разработки Visual Studio 2022, так как — это мощный и популярный инструментарий для разработки приложений. C# - современный язык программирования с понятным синтаксисом, а Visual Studio 2022 предлагает удобную интегрированную среду разработки с множеством функций. Они обеспечивают разработчикам доступ к богатой экосистеме, высокую производительность и надежность приложений, а также поддержку и обновления со стороны Microsoft.

## Постановка задачи

Разработать Веб-сервис для написания постов с использованием базы данных. Веб-сервис должен предоставлять пользователям возможность просматривать посты, написанные другими пользователями и добавлять им комментарии. База данных должна содержать информацию о постах, включая название, текс, список комментариев и другие связанные данные для обеспечения полноценного функционирования веб-сервиса.

# Анализ требований к программному средству и разработка функциональных требований

## Обоснование выбора СУБД и иных средств разработки

Выбор в качестве системы управления базами данных (СУБД) был сделан в пользу MySQL, свободной реляционной СУБД, разрабатываемой корпорацией Oracle. Этот продукт распространяется под лицензией GNU General Public License и коммерческой лицензией, и обычно применяется в малых и средних приложениях, включая серверы WAMP и LAMP.

MySQL может действовать как сервер, обслуживающий локальных или удаленных клиентов, и включает в себя внутренний сервер для встраивания в автономные программы. СУБД обладает гибкостью благодаря поддержке различных типов таблиц, таких как MyISAM и InnoDB, с различными характеристиками, включая поддержку полнотекстового поиска и транзакций.

MySQL имеет ограничения по размеру таблиц, зависящие от их типа. Например, MyISAM ограничен размером файловой системы, тогда как InnoDB может использовать несколько файлов для одной таблицы, увеличивая общее табличное пространство.

Созданы различные форки MySQL, включая MariaDB. MySQL поддерживает разнообразные типы данных, операторы и функции SQL, а также обеспечивает функциональности, такие как репликация, транзакции, работа с внешними ключами и другие. СУБД пользуется популярностью благодаря своей простоте, обширным функциональным возможностям, системам безопасности и поддержке SQL.

MySQL также реализует функцию разграничения доступа, позволяя определить уровни доступа для разных пользователей. Эта особенность усиливает безопасность и дает возможность организовать управление данными в соответствии с ролями.

Важным аспектом является также активное сообщество разработчиков и поддержка со стороны Oracle, что гарантирует постоянное обновление и совершенствование СУБД. MySQL остается одним из ведущих решений для различных проектов, включая веб-приложения, корпоративные системы и приложения с невысокой нагрузкой.

Таким образом, MySQL представляет собой оптимальное решение для курсового проекта, обеспечивая легкость в настройке, надежность и масштабируемость при необходимости.

## Спецификация функциональных требований

Хранение информации о постах: База данных должна предоставлять возможность хранения информации о постах, включая название, текс, список комментариев и другие связанные данные для обеспечения полноценного функционирования веб-сервиса.

Управление пользователями: База данных должна позволять хранить информацию о пользователях, включая их ФИО, город, страну, возраст и другие связанные данные. Также должна быть возможность связывать пользователей с темами, на которые они подписаны.

Управление комментариями: База данных должна поддерживать хранение комментариев на определённые посты. Это позволит пользователю ответить на пост, написанный другим человеком.

## Учитываемые функции и бизнес-процессы

База данных для веб-форума должна иметь возможность хранить информацию о постах, включая название, текст, список комментариев, дата написания поста и другие характеристики. Она также должна поддерживать управление пользователями, включая хранение информации о пользователях, таких как их ФИО, город, страну, возраст и другие связанные данные. База данных должна также поддерживать хранение комментариев это позволит пользователю ответить на пост, написанный другим человеком.

Программное средство должно поддерживать возможность просматривать постов, оставлять комментарии, добавлять посты, добавлять темы в закладки и подписываться на других пользователей.

## Ключевые особенности базы данных

Программное средство должно позволять работать одновременно большому числу пользователей, что требует хорошей оптимизации как самого программного средства, так и базы данных. База данных должна содержать в себе сущности для хранения таких важных данных, как посты, комментарии, информация о пользователях. Для обеспечения эффективной работы программного средства и базы данных в условиях большой нагрузки можно применить следующие оптимизационные подходы:

1. Индексирование: Создание индексов на часто используемых полях в базе данных позволяет ускорить поиск и фильтрацию данных. Например, можно создать индексы для полей, используемых при поиске постов по названию, а также для полей, связанных с пользователями, таких как имена, города, страны или возраст пользователя.
2. Оптимизация запросов: Анализ и оптимизация запросов к базе данных могут значительно повысить производительность системы. Использование индексов, правильное написание запросов с использованием оптимальных операторов и избегание излишних запросов могут сократить время выполнения запросов и уменьшить нагрузку на базу данных.

## Пользователи системы и их роли

Авторизованный пользователь: пользователь может просматривать информацию о постах, создавать собственные посты и редактировать информацию о себе. Пользователь может оставлять комментарии к посту, а также может добавлять темы в свои закладки и подписываться на других пользователей.

## Параметры поиска информации

Данные поступают в базу данных через API, который реализован с использованием контроллеров в ASP.NET Core. Контроллеры служат посредниками между клиентскими запросами и сервисами, которые выполняют операции с базой данных SQL Server.

Когда клиент отправляет запрос через API, соответствующий контроллер принимает этот запрос и передает его соответствующему сервису. Сервис содержит логику обработки запроса и взаимодействует с базой данных, отправляя соответствующие SQL-запросы для получения или изменения данных.

После получения данных из базы данных сервис возвращает их контроллеру, который в свою очередь формирует ответ и отправляет его обратно клиенту. Клиентский интерфейс отображает полученные данные пользователю.

## Бизнес-процессы, связанные с реализацией информационной системы

Множество запросов на чтение: предполагается, что пользователи базы данных часто выполняют запросы на чтение данных, например, для аналитических целей или отображения информации, это может потребовать оптимизации процесса чтения данных.

Большое количество параллельных запросов: база данных получает большое количество параллельных запросов от разных пользователей. Поэтому следует разработать план оптимизации как запросов к базе данных, так и саму базу данных. Также для программного средства тоже следует оптимизировать архитектуру, для более быстрого выполнения запросов.

# Инфологическая модель предметной области

Изображение выглядит как текст, диаграмма, План, Параллельный

Автоматически созданное описание

Рисунок 3.1 – Инфологическая модель базы данных

## Описание моделей

База данных должна содержать в себе информацию о постах, форумах, темах, комментарии, также пользователей и информацию о них.

При проектировании базы данных на инфологическом уровне можно выделить следующие сущности и их атрибуты:

1. Topic: Таблица для содержания id темы и id форума к которому относится, названия темы и тегов к этому темы.
2. Forum: Таблица для содержания id форума, названия форума и тегов к этому форуму.
3. Post: Таблица для содержания id поста и id темы к которой относится, названия поста, содержание этого поста и пользователя, кто написал этот пост.
4. Most-popular-forum: Таблица для содержания самого популярного форума.
5. Most-popular-topic: Таблица для содержания самой популярной темы.
6. Most-popular-post: Таблица для содержания самого популярного поста.
7. Post\_Like: Таблица для хранения лайков и дизайнов к посту, а также id поста.
8. Forum\_Tags: Таблица для хранения тегов форумов.
9. Topic\_Tags: Таблица для хранения тегов тем.
10. Forum\_Tag\_Bans: Таблица для хранения заблокированных тегов форума.
11. Topic\_Tag\_Bans: Таблица для хранения заблокированных тегов тем.
12. Bookmarks: Таблица для хранения тем которые пользователь добавил в закладки.
13. User: Таблица для хранения ника пользователя
14. Comments: Таблица для хранения комментария написанного к определённому посту.
15. Comments\_Like: Таблица для хранения лайков и дизлайков комментария.
16. Profile: Таблица для хранения информации о пользователе включающую ФИО, страну, город, возраст и т.д..
17. Subscriptions: Таблица для хранения подписок пользователя на других пользователей.
18. Gender: Таблица для хранения гендеров.
19. Country: Таблица для хранения стран.
20. City: Таблица для хранения городов.

## Сущности и связи

Таблица 1 – Сущности и связи

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Отношение (имя на русском языке и латиницей) | Описание | Основные атрибуты | Краткое описание связей с другими отношениями | Особенности |
| Forum (Форум) | Содержит информацию о форумах | For\_id (INT): идентификатор таблицы.  (первичный ключ)  For\_Name (Varchar(55)) названия форума.  For\_Tags (INT) id тега к форуму. | Связано с отношениями:  Most-popular-forum, Topic, Forum\_Tags | For\_id является уникальным идентификатором |
| Topic (Тема) | Содержит информацию о теме | Top\_id (INT): идентификатор таблицы.  (первичный ключ)  Top\_Name (Varchar(55)) названия темы.  Top\_Tags (INT) id тега к теме.  Top\_Forum (INT) id форума данной темы. | Связано с отношениями:  Most-popular-topic, Forum, Topic\_Tags, Post | Top\_id является уникальным идентификатором |
| Post (Пост) | Содержит информацию о посте | Post \_id (INT): идентификатор таблицы.  (первичный ключ)  Post \_User (INT) id пользователя.  Post \_Text (Text) текст поста.  Post \_Topic (INT) id топика данного поста. | Связано с отношениями:  Post\_Like, Topic, User, Comment | Post\_id является уникальным идентификатором |
| Most-popular-forum (Самый популярный форум) | Содержит id самого популярного форума | MPF\_id (INT): идентификатор таблицы.  (первичный ключ)  MPF\_Forumы (INT) id самого популярного форума | Связано с отношениями:  Forum | MPF\_id является уникальным идентификатором |
| Most-popular-topic (Самый популярный топик) | Содержит id самой популярной темы | MPT\_id (INT): идентификатор таблицы.  (первичный ключ)  MPT\_Topicы (INT) id самой популярной темы | Связано с отношениями:  Topic | MPT\_id является уникальным идентификатором |
| Post\_Like (Лайки поста) | Содержит информацию о лайках поста | PL\_id (INT): идентификатор таблицы.  (первичный ключ)  PL\_Post (INT) id поста  PL\_Likt\_Count (INT) количество лайков  PL\_Dislikt\_Count (INT) количество лайков | Связано с отношениями:  Post | PL\_id является уникальным идентификатором |
| Forum\_Tags (Теги форума) | Содержит информацию о тегах форума | FTag\_id (INT): идентификатор таблицы.  (первичный ключ)  FTag\_Name (Varchar(55)) Название тега | Связано с отношениями:  Forum, Forum-Tags-Bans | FTag\_id является уникальным идентификатором |
| Topic\_Tags (Теги тем) | Содержит информацию о тегах тем | TTag\_id (INT): идентификатор таблицы.  (первичный ключ)  TTag\_Name (Varchar(55)) Название тега | Связано с отношениями:  Topic, Topic-Tags-Bans | TTag\_id является уникальным идентификатором |
| Bookmarks (Закладки) | Содержит id тем добавленных в закладки | Bo\_Profile (INT): id профиля пользователя.  Bo \_Topic (INT) id темы | Связано с отношениями:  Topic, Profile | Bo\_Profile, Bo \_Topic являются ключами |
| User (Пользователь) | Содержит информацию о пользователе | U\_id (INT): id пользователя.  (первичный ключ)  U\_Profile (INT) id профиля пользователя  U\_Nickname (Varchar(55)) ник пользователя | Связано с отношениями:  Topic, Profile | U\_id является уникальным идентификатором |
| Comment (Комментарий) | Содержит информацию о комментарии | Com\_id (INT): идентификатор таблицы.  (первичный ключ)  Com\_Profile (INT) id профиля пользователя  Com\_Text (Text) текст комментария  Com\_Post (INT) id поста | Связано с отношениями:  Post, Commen- Like, Profile | Com\_id является уникальным идентификатором |
| Comment\_ Like (Лайки комментария) | Содержит информацию о лайках комментария | CL\_id (INT): идентификатор таблицы.  (первичный ключ)  CL\_Comment (INT) id комментария  CL\_Likes\_Count (INT) колличество лайков  CL\_Dislikes\_Count (INT) колличество дизлайков | Связано с отношениями:  Post, Commen- Like, Profile | CL\_Profile является уникальным идентификатором |
| Profile (Профиль) | Содержит информацию о профиле пользователя | P\_id (INT): идентификатор таблицы.  (первичный ключ)  P\_Name (Varchar(55)) имя пользователя  P\_Country (INT): id страны.  P\_Surname (Varchar(55)) фамилия пользователя  P\_Patrinymic (Varchar(55)) фамилия пользователя  P\_Login (Varchar(55)) логин пользователя  P\_Password (Varchar(55)) пароль пользователя  P\_Gender (Varchar(55)) пол пользователя  P\_Age (Varchar(55)) возраст пользователя | Связано с отношениями:  Gender, City, Country, Subscription, Comment, User, BookMark, Topic\_Tag\_Bans, Forum\_Tag\_ Bags | P\_id является уникальным идентификатором |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Subscription (Подписка) | Содержит информацию о подписях пользователя | Sub\_Profile (INT): id пользователя.  Sub\_ Subscription\_ Profile (INT) id пользователя подписки. | Связано с отношениями:  Profile | Sub\_Profile, Sub\_ Subscription\_ Profile является ключами |
| Gender (Пол) | Содержит информацию о полах программы | Gen\_id (INT): идентификатор таблицы.  (первичный ключ)  Gen\_Name (Varchar(55)) Название пола | Связано с отношениями:  Profile | Gen\_id является уникальным идентификатором |
| City (Пол) | Содержит информацию о городах программы | City\_id (INT): идентификатор таблицы.  (первичный ключ)  City\_Name (Varchar(55)) Название города | Связано с отношениями:  Profile, Country | City\_id является уникальным идентификатором |
| Country (Пол) | Содержит информацию о странах программы | C\_id (INT): идентификатор таблицы.  (первичный ключ)  C\_Name (Varchar(55)) Название страны | Связано с отношениями:  Profile, City | C\_id является уникальным идентификатором |

## Особенности нормализации

Разбиение таблицы с информацией о профиле пользователя на Gender, City и Country позволяет контролировать добавление и изменение данных, так как теперь нельзя записать данные, не находящиеся в данных таблицах.

# Подробное описание бизнес-логики

Изображение выглядит как текст, диаграмма, Шрифт, рукописный текст

Автоматически созданное описание

## Триггеры

### Заполнение профиля при создании

Триггер add\_empty\_values срабатывает при создании пользователя. Триггер заполняет базу данных профиля пустыми полями после регистрации пользователя.

DELIMITER $$

CREATE TRIGGER add\_empty\_values BEFORE INSERT ON Profile

FOR EACH ROW

BEGIN

SET NEW.P\_Name = '';

SET NEW.P\_Surname = '';

SET NEW.P\_Country = -1;

SET NEW.P\_City = -1;

SET NEW.P\_Patronymic = '';

SET NEW.P\_Gender = -1;

SET NEW.P\_Age = '';

END $$

DELIMITER ;

## Основные алгоритмы ПС

* + 1. **Алгоритмы пагинации**

Алгоритм buttonNextPage\_Click меняет страницу листа, выводящего на окно тот или иной список. Для проверки на возможность перейти на следующую страницу, алгоритм делает запрос в базу данных на получения количества объектов, если колличество объектов позволяет перейти на следующую страницу, то наращивается переменная page.

Алгоритм buttonPrevPage\_Click выполняет обратное действие, и проверяет на то, что page не будет меньше еденицы.

**Алгоритм buttonNextPage\_Click:**

private static void buttonNextPage\_Click(object sender, EventArgs e)

{

string query = $"Select count(\*) from Comment where Com\_Post = {PostId}";

MySqlCommand command = new MySqlCommand(query, connection);

object result = command.ExecuteScalar();

int Num = Convert.ToInt32(result);

double i = (Num - 1) / 2;

if (i + 1 > page)

{

page++;

drawPanel();

}

}

**Алгоритм buttonPrevPage\_Click:**

private static void buttonPrevPage\_Click(object sender, EventArgs e)

{

if (page - 1 > 0)

{

page--;

drawPanel();

}

}

* + 1. Алгоритмы отрисовки списка

Алгоритм drawPanel выполняет отрисовку панели с элементами бд, для вывода списка данных элементов. В алгоритме производится запрос в бд на получение необходимых данных, создаётся панель для каждого элемента, которая в последствии будет добавлена на основную панель.

**Алгоритм drawPanel:**

private static void drawPanel()

{

panel.Controls.Clear();

int i = 0;

string queryCount = $"Select count(\*) from Post where Post\_Topic = {TopicId};";

MySqlCommand commandCount = new MySqlCommand(queryCount, connection);

object result = commandCount.ExecuteScalar();

commandCount.Dispose();

int count = Convert.ToInt32(result);

int countUs = 0;

string query = $"SELECT Post\_Title FROM Post where Post\_Topic = {TopicId};";

MySqlCommand command = new MySqlCommand(query, connection);

MySqlDataReader reader = command.ExecuteReader();

string name = "";

while (reader.Read() && countUs < 7 && countUs < count - (7 \* (page - 1)))

{

name = reader.GetString("Post\_Title");

if (name != "")

{

Panel innerPanel = new Panel();

innerPanel.BorderStyle = BorderStyle.Fixed3D;

innerPanel.Location = new System.Drawing.Point(0, 26 \* countUs);

innerPanel.Size = new System.Drawing.Size(panel.Size.Width, 26);

panel.Controls.Add(innerPanel);

Label label = new Label();

label.Text = $"Post: {name}";

label.Size = new System.Drawing.Size(120, label.Size.Height);

innerPanel.Controls.Add(label);

Button button = new Button();

button.Location = new System.Drawing.Point(panel.Size.Width - 80, 0);

button.Text = $"Watch";

button.Name = $"{name}";

button.Click += buttonPost\_Click;

innerPanel.Controls.Add(button);

countUs++;

}

i++;

}

command.Dispose();

reader.Close();

}

# Тестирование базы данных и приложений системы

## Тестирование триггеров

### Триггер: add\_empty\_values

Исходные данные: Добавление пользователя

Ожидаемый результат: Добавление пустых полей.

Полученный результат: Добавление пустых полей.

## Тестирование программного средства

### Создание пользователя

Исходный набор данных: запрос на добавление пользователя в бд с исходными данными логина eucalypt и паролем 12345.

Ожидаемый результат: пользователь создан

Полученный результат:

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 5.2.1 – создание пользователя

### Обновление информации пользователя

Исходный набор данных:

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 5.2.2 – список добавляемых данных

Ожидаемый результат: обновление данных пользователя

Полученный результат:

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 5.2.3 – полученный результат

### Добавление поста

Исходный набор данных:

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 5.2.4 – написание поста

Ожидаемый результат: пост добавлен

Полученный результат:

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 5.2.5 – полученный результат добавление поста

### Добавление комментария

Исходный набор данных:

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 5.2.6 – написание комментария

Ожидаемый результат: пост добавлен

Полученный результат:

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 5.2.7 – комментария добавлен к посту

### Регистрация

Исходный набор данных: Запрос на регистрацию.

Ожидаемый результат: Успешная регистрация пользователя.

Полученный результат: Успешная регистрация.

### Авторизация

Исходный набор данных: Запрос на авторизацию.

Ожидаемый результат: Успешная авторизация пользователя.

Полученный результат: Успешная авторизация.

# Заключение

В рамках курсового проекта была разработана база данных для веб-форума. Данная разработка позволяет хранить и обеспечивать управление информацией о постах, пользователей, комментариев, форумах, тем, и других связанных с форумом сущностях. Была обеспечена функциональность по вводу, изменению и извлечению данных, а также поддерживать связи и целостность данных.

Также было разработано приложение для взаимодействия с базой данных. Пользователи могут искать посты, просматривать информацию о них, оставлять комментарии, добавлять темы в закладки. Приложение обеспечивает удобный пользовательский интерфейс для навигации по постам и взаимодействия с ними.

В процессе разработки исследовались различные варианты решения задач, связанных с веб-форумом. Была изучена модель базы данных, специфичная для веб-форума, а также язык запросов для работы с базой данных. В данном проекте в качестве системы управления базами данных (СУБД) был применен MySQL Server, который обеспечивает надежное хранение и эффективное извлечение данных для сервиса просмотра фильмов. SQL Server предоставляет широкие возможности для моделирования, хранения и запросов данных, что позволяет эффективно работать с информацией о постах и связанных с ними сущностях в базе данных.

В результате были достигнуты поставленные цели, и приложение взаимодействует успешно с базой данных, обеспечивая надежное хранение и управление информацией, а также удобный интерфейс для пользователей. Проект прошел все тестирования и готов для использования.

# Список использованной литературы

1. Куликов, Святослав. Реляционные базы данных в примерах. EPAM Systems, 2020–2023. – 424 с.
2. Куликов, Святослав. Работа с MySQL, MS SQL Server и Oracle в примерах. EPAM Systems, 2020–2023. – 590 с.
3. Б. Новиков, Е. Горшкова, Н. Графеева. Основы технологий баз данных: учебное пособие. ДМК Пресс, 2020. – 582 с.
4. Руководство по Entity Framework 8 [Электронный ресурс] – Режим доступа: <https://metanit.com/sharp/efcore>
5. Руководство по ASP.NET Core 8 [Электронный ресурс] – Режим доступа: <https://metanit.com/sharp/aspnet6/>
6. Руководство по ASP.NET Core MVC [Электронный ресурс] – Режим доступа: <https://metanit.com/sharp/aspnetmvc/>

Приложение А

(обязательное)

Схема базы данных на языке SQL

SET FOREIGN\_KEY\_CHECKS=0

;

/\* Drop Tables \*/

DROP TABLE IF EXISTS `City` CASCADE

;

DROP TABLE IF EXISTS `Comment` CASCADE

;

DROP TABLE IF EXISTS `Comment\_Like` CASCADE

;

DROP TABLE IF EXISTS `Country` CASCADE

;

DROP TABLE IF EXISTS `Forum` CASCADE

;

DROP TABLE IF EXISTS `Forum\_Tag\_Bans` CASCADE

;

DROP TABLE IF EXISTS `Forum\_Tags` CASCADE

;

DROP TABLE IF EXISTS `Gender` CASCADE

;

DROP TABLE IF EXISTS `m2m\_Bookmarks` CASCADE

;

DROP TABLE IF EXISTS `Most\_popular\_forum` CASCADE

;

DROP TABLE IF EXISTS `Most\_popular\_topic` CASCADE

;

DROP TABLE IF EXISTS `Post` CASCADE

;

DROP TABLE IF EXISTS `Post\_Like` CASCADE

;

DROP TABLE IF EXISTS `Profile` CASCADE

;

DROP TABLE IF EXISTS `Subscriptions` CASCADE

;

DROP TABLE IF EXISTS `Topic` CASCADE

;

DROP TABLE IF EXISTS `Topic\_Tag\_Bans` CASCADE

;

DROP TABLE IF EXISTS `Topic\_Tags` CASCADE

;

DROP TABLE IF EXISTS `User` CASCADE

;

/\* Create Tables \*/

CREATE TABLE `City`

(

`City\_id` INT NOT NULL,

`City\_Name` VARCHAR(50) NULL,

CONSTRAINT `PK\_Country` PRIMARY KEY (`City\_id` ASC)

)

;

CREATE TABLE `Comment`

(

`Com\_id` INT NOT NULL,

`Com\_Profile` INT NULL,

`Com\_Post` INT NULL,

`Com\_Text` VARCHAR(50) NULL,

CONSTRAINT `PK\_Comment` PRIMARY KEY (`Com\_id` ASC)

)

;

CREATE TABLE `Comment\_Like`

(

`CL\_id` INT NOT NULL,

`CL\_Comment` INT NULL,

`CL\_Likes\_Count` INT NULL,

`CL\_Dislikes\_Count` INT NULL,

CONSTRAINT `PK\_Comment\_Like` PRIMARY KEY (`CL\_id` ASC)

)

;

CREATE TABLE `Country`

(

`C\_id` INT NOT NULL,

`C\_Country\_Name` VARCHAR(50) NULL,

CONSTRAINT `PK\_Country` PRIMARY KEY (`C\_id` ASC)

)

;

CREATE TABLE `Forum`

(

`For\_id` INT NOT NULL,

`For\_Name` VARCHAR(50) NULL,

`For\_Tags` INT NULL,

CONSTRAINT `PK\_Forum` PRIMARY KEY (`For\_id` ASC)

)

;

CREATE TABLE `Forum\_Tag\_Bans`

(

`FTB\_Profile` INT NULL,

`FTB\_Tag` INT NULL

)

;

CREATE TABLE `Forum\_Tags`

(

`FTag\_id` INT NOT NULL,

`FTag\_Name` VARCHAR(50) NULL,

CONSTRAINT `PK\_Forum\_Tags` PRIMARY KEY (`FTag\_id` ASC)

)

;

CREATE TABLE `Gender`

(

`Gen\_id` INT NOT NULL,

`Gen\_Name` VARCHAR(50) NULL,

CONSTRAINT `PK\_Gender` PRIMARY KEY (`Gen\_id` ASC)

)

;

CREATE TABLE `m2m\_Bookmarks`

(

`Bo\_Profile` INT NOT NULL,

`Bo\_Topic` INT NOT NULL,

CONSTRAINT `PK\_m2m\_Bookmarks` PRIMARY KEY (`Bo\_Topic` ASC, `Bo\_Profile` ASC)

)

;

CREATE TABLE `Most\_popular\_forum`

(

`MPF\_id` INT NOT NULL,

`MPF\_Forum` INT NULL,

CONSTRAINT `PK\_Most\_popular\_forum` PRIMARY KEY (`MPF\_id` ASC)

)

;

CREATE TABLE `Most\_popular\_topic`

(

`MPT\_id` INT NOT NULL,

`MPT\_Topic` INT NULL,

CONSTRAINT `PK\_Most\_popular\_topic` PRIMARY KEY (`MPT\_id` ASC)

)

;

CREATE TABLE `Post`

(

`Post\_id` INT NOT NULL,

`Post\_Text` VARCHAR(50) NULL,

`Post\_User` INT NULL,

`Post\_Topic` INT NULL,

CONSTRAINT `PK\_Post` PRIMARY KEY (`Post\_id` ASC)

)

;

CREATE TABLE `Post\_Like`

(

`PL\_id` INT NOT NULL,

`PL\_Likes\_Count` INT NULL,

`PL\_Post` INT NULL,

`PL\_dislikes\_Count` INT NULL,

CONSTRAINT `PK\_Post\_Like` PRIMARY KEY (`PL\_id` ASC)

)

;

CREATE TABLE `Profile`

(

`P\_id` INT NOT NULL,

`P\_Name` VARCHAR(50) NULL,

`P\_Country` INT NULL,

`P\_Surname` VARCHAR(50) NULL,

`P\_Patronymic` VARCHAR(50) NULL,

`P\_Login` VARCHAR(50) NULL,

`P\_Password` VARCHAR(50) NULL,

`P\_City` INT NULL,

`P\_Gender` INT NULL,

`P\_Age` VARCHAR(50) NULL,

CONSTRAINT `PK\_Profile` PRIMARY KEY (`P\_id` ASC)

)

;

CREATE TABLE `Subscriptions`

(

`Sub\_Profile` INT NULL,

`Sub\_Subsciption\_profile` INT NULL,

CONSTRAINT `PK\_Subscriptions` PRIMARY KEY ()

)

;

CREATE TABLE `Topic`

(

`Top\_id` INT NOT NULL,

`Top\_Name` VARCHAR(50) NULL,

`Top\_Tags` INT NULL,

`Top\_Forum` INT NULL,

CONSTRAINT `PK\_Topic` PRIMARY KEY (`Top\_id` ASC)

)

;

CREATE TABLE `Topic\_Tag\_Bans`

(

`qwe` VARCHAR(50) NULL,

`TTB\_Tag` INT NULL,

`TTB\_Name` INT NULL

)

;

CREATE TABLE `Topic\_Tags`

(

`TTag\_id` INT NOT NULL,

`TTag\_Name` VARCHAR(50) NULL,

CONSTRAINT `PK\_Topic\_Tags` PRIMARY KEY (`TTag\_id` ASC)

)

;

CREATE TABLE `User`

(

`U\_id` INT NOT NULL,

`U\_NickName` VARCHAR(50) NULL,

`U\_Profile` INT NOT NULL,

CONSTRAINT `PK\_User` PRIMARY KEY (`U\_id` ASC)

)

;

/\* Create Primary Keys, Indexes, Uniques, Checks \*/

ALTER TABLE `Comment`

ADD INDEX `IXFK\_Comment\_Post` (`Com\_Post` ASC)

;

ALTER TABLE `Comment`

ADD INDEX `IXFK\_Comment\_Profile` (`Com\_Profile` ASC)

;

ALTER TABLE `Comment\_Like`

ADD INDEX `IXFK\_Comment\_Like\_Comment` (`CL\_Comment` ASC)

;

ALTER TABLE `Forum`

ADD INDEX `IXFK\_Forum\_Forum\_Tags` (`For\_Tags` ASC)

;

ALTER TABLE `Forum\_Tag\_Bans`

ADD INDEX `IXFK\_Forum\_Tag\_Bans\_Forum\_Tags` (`FTB\_Tag` ASC)

;

ALTER TABLE `Forum\_Tag\_Bans`

ADD INDEX `IXFK\_Forum\_Tag\_Bans\_Profile` (`FTB\_Profile` ASC)

;

ALTER TABLE `m2m\_Bookmarks`

ADD INDEX `IXFK\_m2m\_Bookmarks\_Profile` (`Bo\_Profile` ASC)

;

ALTER TABLE `m2m\_Bookmarks`

ADD INDEX `IXFK\_m2m\_Bookmarks\_Topic` (`Bo\_Topic` ASC)

;

ALTER TABLE `Most\_popular\_forum`

ADD INDEX `IXFK\_Most\_popular\_forum\_Forms` (`MPF\_Forum` ASC)

;

ALTER TABLE `Most\_popular\_topic`

ADD INDEX `IXFK\_Most\_popular\_topic\_Topic` (`MPT\_Topic` ASC)

;

ALTER TABLE `Post`

ADD INDEX `IXFK\_Post\_Topic` (`Post\_Topic` ASC)

;

ALTER TABLE `Post`

ADD INDEX `IXFK\_Post\_User` (`Post\_User` ASC)

;

ALTER TABLE `Post\_Like`

ADD INDEX `IXFK\_Post\_Like\_Post` (`PL\_Post` ASC)

;

ALTER TABLE `Profile`

ADD INDEX `IXFK\_Profile\_City` (`P\_City` ASC)

;

ALTER TABLE `Profile`

ADD INDEX `IXFK\_Profile\_Country` (`P\_Country` ASC)

;

ALTER TABLE `Profile`

ADD INDEX `IXFK\_Profile\_Gender` (`P\_Gender` ASC)

;

ALTER TABLE `Subscriptions`

ADD INDEX `IXFK\_Subscriptions\_Profile` (`Sub\_Subsciption\_profile` ASC)

;

ALTER TABLE `Subscriptions`

ADD INDEX `IXFK\_Subscriptions\_Profile\_02` (`Sub\_Profile` ASC)

;

ALTER TABLE `Topic`

ADD INDEX `IXFK\_Topic\_Forum` (`Top\_Forum` ASC)

;

ALTER TABLE `Topic`

ADD INDEX `IXFK\_Topic\_Topic\_Tags` (`Top\_Tags` ASC)

;

ALTER TABLE `Topic\_Tag\_Bans`

ADD INDEX `IXFK\_Topic\_Tag\_Bans\_Profile` (`TTB\_Name` ASC)

;

ALTER TABLE `Topic\_Tag\_Bans`

ADD INDEX `IXFK\_Topic\_Tag\_Bans\_Topic\_Tags` (`TTB\_Tag` ASC)

;

ALTER TABLE `User`

ADD INDEX `IXFK\_User\_Profile` (`U\_Profile` ASC)

;

/\* Create Foreign Key Constraints \*/

ALTER TABLE `Comment`

ADD CONSTRAINT `FK\_Comment\_Post`

FOREIGN KEY (`Com\_Post`) REFERENCES `Post` (`Post\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `Comment`

ADD CONSTRAINT `FK\_Comment\_Profile`

FOREIGN KEY (`Com\_Profile`) REFERENCES `Profile` (`P\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `Comment\_Like`

ADD CONSTRAINT `FK\_Comment\_Like\_Comment`

FOREIGN KEY (`CL\_Comment`) REFERENCES `Comment` (`Com\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `Forum`

ADD CONSTRAINT `FK\_Forum\_Forum\_Tags`

FOREIGN KEY (`For\_Tags`) REFERENCES `Forum\_Tags` (`FTag\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `Forum\_Tag\_Bans`

ADD CONSTRAINT `FK\_Forum\_Tag\_Bans\_Forum\_Tags`

FOREIGN KEY (`FTB\_Tag`) REFERENCES `Forum\_Tags` (`FTag\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `Forum\_Tag\_Bans`

ADD CONSTRAINT `FK\_Forum\_Tag\_Bans\_Profile`

FOREIGN KEY (`FTB\_Profile`) REFERENCES `Profile` (`P\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `m2m\_Bookmarks`

ADD CONSTRAINT `FK\_m2m\_Bookmarks\_Profile`

FOREIGN KEY (`Bo\_Profile`) REFERENCES `Profile` (`P\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `m2m\_Bookmarks`

ADD CONSTRAINT `FK\_m2m\_Bookmarks\_Topic`

FOREIGN KEY (`Bo\_Topic`) REFERENCES `Topic` (`Top\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `Most\_popular\_forum`

ADD CONSTRAINT `FK\_Most\_popular\_forum\_Forms`

FOREIGN KEY (`MPF\_Forum`) REFERENCES `Forum` (`For\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `Most\_popular\_topic`

ADD CONSTRAINT `FK\_Most\_popular\_topic\_Topic`

FOREIGN KEY (`MPT\_Topic`) REFERENCES `Topic` (`Top\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `Post`

ADD CONSTRAINT `FK\_Post\_Topic`

FOREIGN KEY (`Post\_Topic`) REFERENCES `Topic` (`Top\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `Post`

ADD CONSTRAINT `FK\_Post\_User`

FOREIGN KEY (`Post\_User`) REFERENCES `User` (`U\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `Post\_Like`

ADD CONSTRAINT `FK\_Post\_Like\_Post`

FOREIGN KEY (`PL\_Post`) REFERENCES `Post` (`Post\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `Profile`

ADD CONSTRAINT `FK\_Profile\_Country`

FOREIGN KEY (`P\_Country`) REFERENCES `Country` (`C\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `Profile`

ADD CONSTRAINT `FK\_Profile\_Gender`

FOREIGN KEY (`P\_Gender`) REFERENCES `Gender` (`Gen\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `Subscriptions`

ADD CONSTRAINT `FK\_Subscriptions\_Profile`

FOREIGN KEY (`Sub\_Subsciption\_profile`) REFERENCES `Profile` (`P\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `Subscriptions`

ADD CONSTRAINT `FK\_Subscriptions\_Profile\_02`

FOREIGN KEY (`Sub\_Profile`) REFERENCES `Profile` (`P\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `Topic`

ADD CONSTRAINT `FK\_Topic\_Forum`

FOREIGN KEY (`Top\_Forum`) REFERENCES `Forum` (`For\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `Topic`

ADD CONSTRAINT `FK\_Topic\_Topic\_Tags`

FOREIGN KEY (`Top\_Tags`) REFERENCES `Topic\_Tags` (`TTag\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `Topic\_Tag\_Bans`

ADD CONSTRAINT `FK\_Topic\_Tag\_Bans\_Profile`

FOREIGN KEY (`TTB\_Name`) REFERENCES `Profile` (`P\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `Topic\_Tag\_Bans`

ADD CONSTRAINT `FK\_Topic\_Tag\_Bans\_Topic\_Tags`

FOREIGN KEY (`TTB\_Tag`) REFERENCES `Topic\_Tags` (`TTag\_id`) ON DELETE Restrict ON UPDATE Restrict

;

ALTER TABLE `User`

ADD CONSTRAINT `FK\_User\_Profile`

FOREIGN KEY (`U\_Profile`) REFERENCES `Profile` (`P\_id`) ON DELETE Restrict ON UPDATE Restrict

;

INSERT INTO `Country` (`C\_id`, `C\_Country\_Name`)

VALUES

(0, 'Belarus'),

(1, 'Russia'),

(2, 'England'),

(3, 'Germany'),

(4, 'Poland');

INSERT INTO `City` (`City\_id`, `City\_Name`, `City\_Country\_id`)

VALUES

(0, 'Minsk', 0),

(1, 'Moskow', 1),

(2, 'London', 2),

(3, 'Berlin', 3),

(4, 'Warsaw', 4),

(5, 'Grodno', 0),

(6, 'Brest', 0),

(7, 'Vitebsk', 0),

(8, 'Gomel', 0),

(9, 'Saint Petersburg', 1),

(10, 'Novosibirsk', 1),

(11, 'Yekaterinburg', 1),

(12, 'Kazan', 1),

(13, 'London', 2),

(14, 'Manchester', 2),

(15, 'Birmingham', 2),

(16, 'Liverpool', 2),

(17, 'Berlin', 3),

(18, 'Munich', 3),

(19, 'Hamburg', 3),

(20, 'Cologne', 3),

(21, 'Warsaw', 4),

(22, 'Krakow', 4),

(23, 'Wroclaw', 4),

(24, 'Poznan', 4);

INSERT INTO `Gender` (`Gen\_id`, `Gen\_Name`) VALUES (0, 'Male'), (1, 'Female');

INSERT INTO `Forum` (`For\_id`, `For\_Name`) VALUES (0, 'TV'), (1, 'Games'), (2, 'Books');

INSERT INTO `Topic` (`Top\_id`, `Top\_Name`, `Top\_Forum`) VALUES (0, 'Films' ,0), (1, 'Series' ,0), (2, 'What to watch' ,0),

(3, 'Help' ,1), (4, 'Discussions' ,1), (5, 'News' ,1),

(6, 'What to read' ,2), (7, 'Discussions' ,2), (8, 'News' ,2);

SET FOREIGN\_KEY\_CHECKS=1

;