**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

**ЛАБОРАТОРНАЯ РАБОТА №5**

по дисциплине: «Разработка приложений баз данных для информационных систем»

на тему: «Разработка интерфейса приложения баз данных с использованием с использованием аутентификации и авторизации»

Выполнил: студент гр. ИТП-31

Бондарев Е.Ю.

Принял: доцент

Асенчик О.Д.

Гомель 2023

**Цель работы:** получить навыки использования *ASP.NET MVC Сore* для создания интерфейса типовых *web*-приложений для работы с информацией из реляционных баз данных.

**Задание:**

Используя разработанный ранее слой доступа к базе данным согласно своему варианту, спроектировать и создать интерфейс *Web*-приложения на основе *ASP*.*NET* *Core* *MVC* *Framework* и *Entity* *Framework* *Core*.

*Web*-приложение должно удовлетворять следующим требованиям:

1. Осуществлять ввод, редактирование, добавление и просмотр данных не менее чем из трех таблиц реляционной базы согласно варианту. Не менее, чем одна из таблиц должна находится на стороне отношения «многие» в схеме базы данных.

2. Иметь единое стилевое оформление, основанное на использовании мастер-страниц.

3. Иметь удобную систему навигации (строка меню, гиперссылки, кнопки), которая обеспечивает оптимальный путь перехода между двумя произвольно выбранными страницами в соответствии с логикой приложения.

4. Пользователь для работы с приложением должен пройти аутентификацию.

5. Должно поддерживать реализацию не менее двух ролевых политик.

6. Администратор должен иметь возможность управлять пользователями: просматривать, создавать, удалять и редактировать данные учетных записей.

7. Представления для просмотра данных из таблиц должны предусматривать разбиение данных на страницы, фильтрацию по одному или нескольким полям.

8. Осуществить кэширования данных для отображения с помощью встроенного инструмента кэширования – объекта *ImemoryCache*. Выводить кэшированные данные таблиц *MemoryCache* на соответствующие страницы на сайт, генерируемые с использованием представлений (*Views*). Данные в кэше хранить неизменными до проведения операций вставки, изменения или удаления данных. После проведения этих операций кэш должен формироваться заново.

9. Реализовать сохранение состояния (значений) элементов представлений, предназначенных для осуществления фильтрации, с использованием куки и (или) с объекта *Session*. Осуществить заполнение элементов представлений, предназначенных для осуществления фильтрации, при их загрузке данными, ранее сохранёнными в объекте куки и (или) *Session*.

Для проверки преподавателем следует разместить код разработанного проекта на *GitHub*.

**Ход работы**

В процессе выполнения лабораторной работы были созданы классы модели, контекста для взаимодействия с этими моделями, инициализации для создания и заполнения базы данных в случае ее отсутствия, а также *Middleware* для интеграции класса инициализации в проект.

Затем были созданы контроллеры и представления для доступа к данным из базы данных. Вся информация из таблицы была организована по страницам для более удобного просмотра. В процессе были внедрены все операции *CRUD*, такие как просмотр, удаление, обновление и создание записей. Также был реализован механизм фильтрации с использованием технологии кук, что позволяет сохранять данные из предыдущих операций фильтрации. Пример страницы с данными трех таблиц указан на рисунке 1.

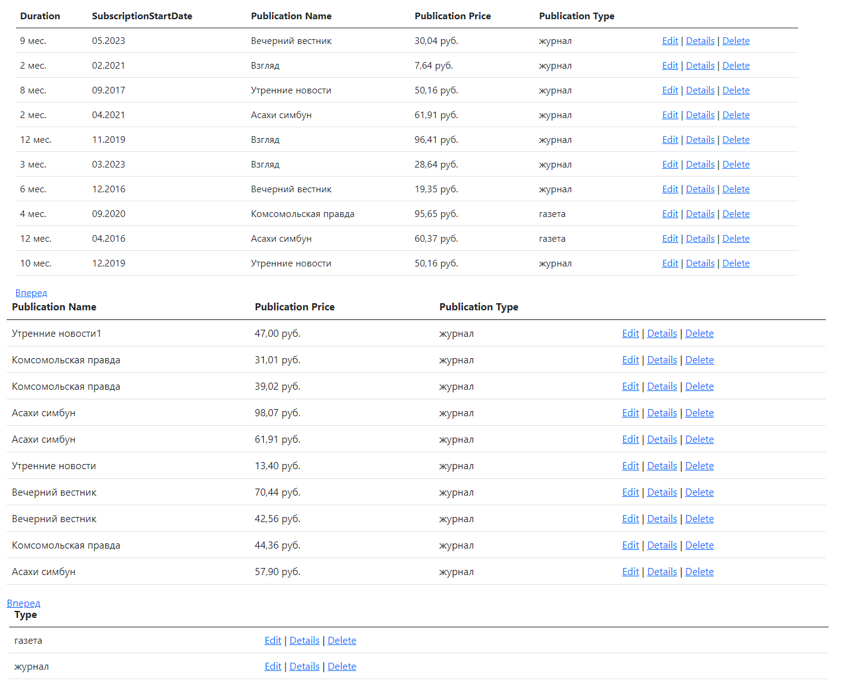


Рисунок 1 – Пример страницы с данными трех таблиц

Пример запроса с фильтрацией и сохранением полей в куки указана на рисунке 2.

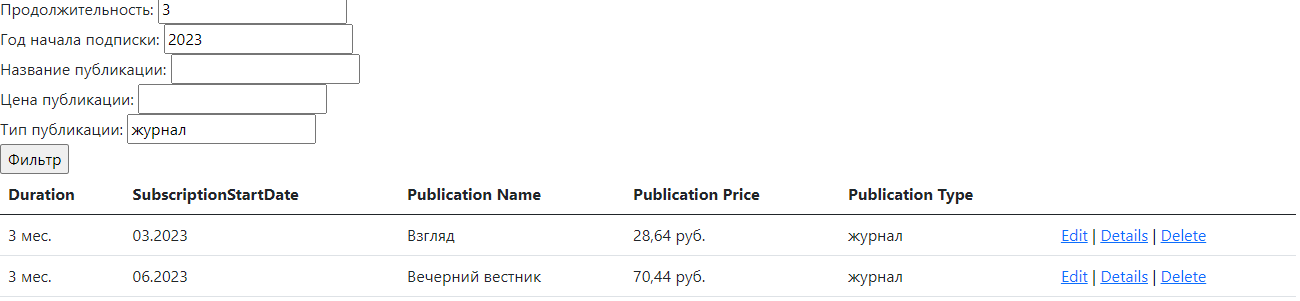


Рисунок 2 – Пример запроса с фильтрацией и сохранением полей в куки

Пример создания нового объекта для одной из таблиц указан на рисунке 3.

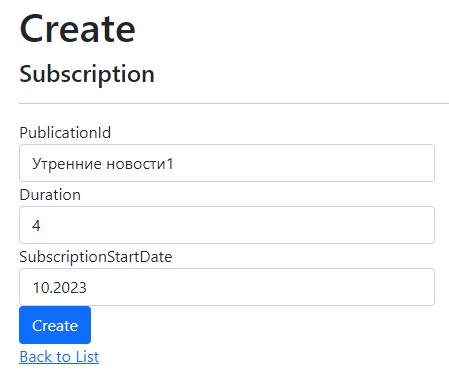


Рисунок 3 – Пример создания нового объекта для одной из таблиц

Пример удаления и записи из таблицы указана на рисунке 4.

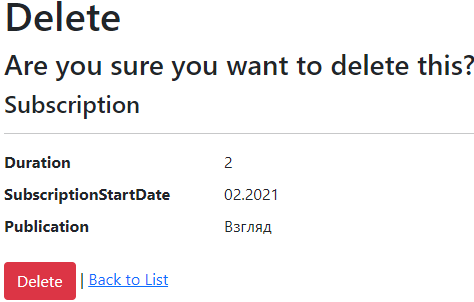


Рисунок 4 – Пример удаления и записи из таблиц

Пример редактирования и записи в таблице указана на рисунке 5.

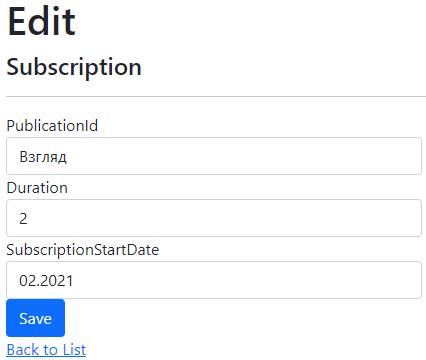


Рисунок 5 – Пример удаления и записи из таблиц

Затем была внедрена система аутентификации и авторизации пользователей. С использованием технологии *ASP.NET Identity* были добавлены дополнительные таблицы в базу данных, предназначенные для хранения информации о пользователях. Также были разработаны контроллеры и представления для регистрации и входа в систему пользователей с различными ролями. Пример окна для регистрации указан на рисунке 6.

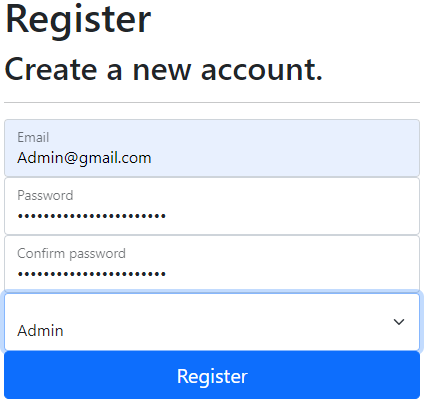


Рисунок 6 – Пример страницы для регистрации

Пример окна для входа в систему указана на рисунке 7.

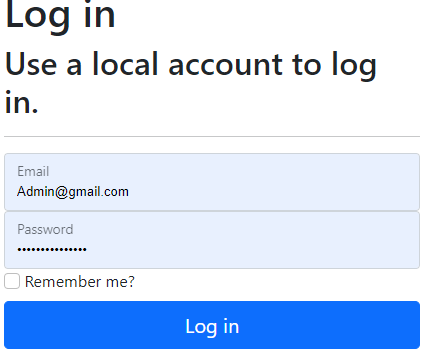


Рисунок 7 – Пример страницы для входа в систему

Для каждой страницы был реализован уровень доступа при помощи атрибута *Authorize*. Если на определенную страницу попытается получить доступ пользователь без необходимых прав будет выведено сообщение, указанное на рисунке 8.



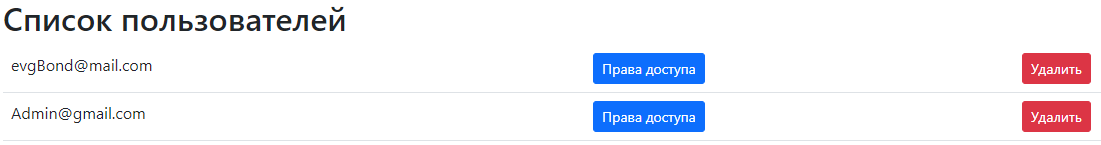
Рисунок 8 – Пример сообщения о запрете доступа

Были реализованы две роли пользователь и администратор. Пользователь может просматривать всю информацию из таблиц бизнес-логики. Администратор может редактировать роли и пользователей. Пример станицы для редактирования ролей указана на рисунке 9.



Рисунок 9 – Страница для редактирования ролей в системе

Пример станиц для редактирования пользователей указана на рисунке 10.



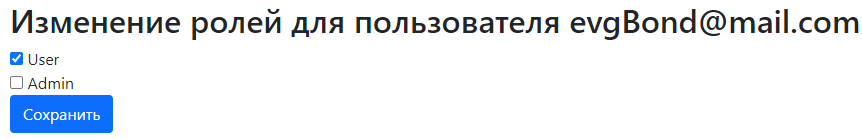


Рисунок 10 – Страниц для редактирования пользователей в системе

Также был разработан класс, предназначенный для кэширования данных. Кэширования производится при помощи интерфейса *IMemoryCache*. Листинг всех этих классов указан в приложении А. Пример скорости запроса без кэширования указан на рисунке 1.

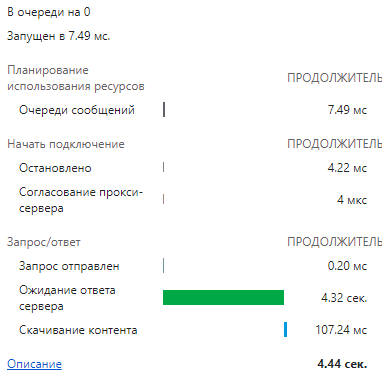


Рисунок 1 – Скорость запроса без кэширования

Далее была замерена скорость загрузки страницы данные в которую загружаются из кэша. Пример этих замеров указаны на рисунке 2.

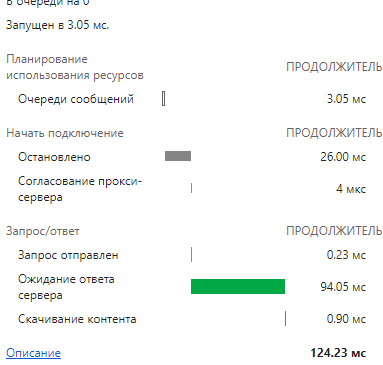


Рисунок 2 – Скорость запроса с кэшированием

После выполнения лабораторной работы созданные проект был добавлен в локальный *git* репозиторий а потом перенесен в *GitHub* репозиторий своего аккаунта. Чтобы ознакомится с созданным проектом можно по ссылке <https://github.com/EvgeniBondarev/DDBAISE/tree/main/Laba5>

**Вывод**: в процессе выполнения лабораторной работы была освоена технология *ASP .NET Core MVC*. Созданы модельные и контекстные классы, предназначенные для взаимодействия с данными. Разработаны контроллеры для связи моделей с представлениями, а также классы представлений, предназначенные для работы с данными из базы данных. Был создан унифицированный дизайн для всех страниц. С использованием технологии *ASP .NET Identity* реализован вход в систему с различными ролями. Для улучшения удобства работы с данными каждая выборка была разбита на страницы.