**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

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

Факультет автоматизированных и информационных систем Кафедра «Информационные технологии»

Специальность 1-40 05 01-01 Информационные системы и технологии (в проектировании и производстве)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту

по дисциплине «Разработка приложений баз данных для информационных систем»

на тему: **«РАЗРАБОТКА WEB­**-**ПРИЛОЖЕНИЯ БАЗ ДАННЫХ «РАДИОСТАНЦИЯ»»**

Исполнитель: студент гр. ИТП-31

Расшивалов Н.И.

Руководитель: доцент

Асенчик О.Д.

Дата проверки:

Дата допуска к защите:

Дата защиты:

Оценка работы:

Подписи членов комиссии

по защите курсового проекта:

Гомель 2021

Оглавление

[**ВВЕДЕНИЕ** 4](#_Toc89568030)

[**1** **ЛОГИЧЕСКАЯ И ФИЗИЧЕСКАЯ СТРУКТУРА БАЗЫ ДАННЫХ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ «РАДИОСТАНЦИЯ»** 5](#_Toc89568031)

[**1.1** **Логическая структура базы данных программного обеспечения «Радиостанция»** 5](#_Toc89568032)

[**1.2** **Физическая структура базы данных программного обеспечения «Радиостанция»** 9](#_Toc89568033)

[**2** **АППАРАТНОЕ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ** 11](#_Toc89568034)

[2.1 Общие сведения. 11](#_Toc89568035)

[2.1 Настройка приложения. 12](#_Toc89568036)

[**3** **СТРУКТУРА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ «РАДИОСТАНЦИЯ»** 13](#_Toc89568037)

[3.1 Архитектура программного обеспечения «Радиостанция» 13](#_Toc89568038)

[**3.2** **Структура контроллеров приложения «Радиостанция»** 14](#_Toc89568039)

[**3.3** **Структура представлений приложения «Радиостанция»** 15](#_Toc89568040)

[3.4 Структура моделей и моделей представлений приложения «Радиостанция» 17](#_Toc89568041)

[3.5 Структура системы аутентификации и авторизации пользователей приложения «Радиостанция» 19](#_Toc89568042)

**ВВЕДЕНИЕ**

На сегодняшний день развитие информационных технологий играет огромную роль в мировом обществе. Объем информации, который увеличивается с каждым днем, вынуждает тратить на обработку данных большое количество временных и иных затрат. По этой причине становится все более необходимым создание приложений, которые позволяют быстро обрабатывать информацию и предоставлять её пользователю в удобном для него виде.

Для эффективной работы любой из организаций важно наличие какой-либо базы или несколько баз данных. Они позволяют хранить большие объемы информации о различных сведениях.

Для рациональной обработки и предоставления информации, которая хранится в базах данных, необходимы информационные системы. Они упрощают обработку больших объемов данных и предоставляют возможность корректно их предоставить.

Наибольшую популярность занимают *web*-приложения. Благодаря их использованию, пользователю нужен только компьютер с браузером и соединение с интернетом. Для того чтобы совершить обновление *web*-приложения, его необходимо обновить только на сервере, и все сразу же смогут работать с новой версией. Так же большинство *web*-приложений являются кроссбраузерными, что позволяет использовать любую операционную систему.

Целью курсового проекта является разработка *web*-приложения баз данных, которое позволит предоставлять и редактировать различную информацию о музыкальных композициях, альбомах, жанрах, трансляциях и т.д. Зарегистрированному пользователю должны предоставляться данные о исполнителях и их композициях, о группах и составе групп, о жанрах и их описанию, о радиотрансляциях. Сотрудники должны иметь возможность редактировать элементов приложения: добавлять, удалять и изменять различную информацию. Администратору необходимо иметь возможность управления штатом сотрудников.

Задачами курсового проекта являются:

* рассмотрение программного обеспечения необходимого для разработки *web*-приложения;
* проектирование базы данных;
* разработка проекта *web*-приложения;
* проектирование пользовательского интерфейса средствами .*NET Core MVC.*

Для разработки был выбран язык программирования *C#*, с использованием технологий *ASP.NET Core* и *Entity Framework Core*. В качестве источника данных была выбрана система управления базами данных (СУБД) *MS SQL Server*.

1. **ЛОГИЧЕСКАЯ И ФИЗИЧЕСКАЯ СТРУКТУРА БАЗЫ ДАННЫХ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ «РАДИОСТАНЦИЯ»**

**1.1 Логическая структура базы данных программного обеспечения «Радиостанция»**

На логическом уровне база данных представлена следующими сущностями:

* сущность *Records* содержит информацию о названии композиции. Имеет связь «многие к одному» с сущностью *Performers* и «один к одному» с сущностью *Genres*. Подробная информация о сущности представлена в таблице 1.1.

Таблица 1.1 – Описание атрибутов сущности *Records*

|  |  |  |
| --- | --- | --- |
| **Название атрибута** | **Тип данных** | **Примечание** |
| 1 | 2 | 3 |
| *Id* | Числовой | Первичный ключ, идетификатор |
| *CompositionName* | Текстовый | Название композиции |
| *PerformerId* | Числовой | Внешний ключ для связи с  сущностью *Performers* |
| *GenreId* | Числовой | Внешний ключ для связи с  сущностью *Genres* |
| *Album* | Текстовый | Название альбома |
| *RecordDate* | Дата | Дата выхода композиции |
| *Lasting* | Длительность | Длительность композиции |
| *Rating* | Числовой | Рейтинг композиции |

* Сущность *Performers* содержит информацию о имени и фамилии исполнителя, его группе. Имеет связь «многие к одному» с сущностью *Groups.* Подробная информация о сущности представлена в таблице 1.2.

Таблица 1.2 – Описание атрибутов сущности *Performers*

|  |  |  |
| --- | --- | --- |
| **Название атрибута** | **Тип данных** | **Примечание** |
| 1 | 2 | 3 |
| *Id* | Числовой | Первичный ключ,  идетификатор |
| *Name* | Текстовый | Имя исполнителя |
| *Surname* | Текстовый | Фамилия исполнителя |
| *GroupId* | Числовой | Внешний ключ для связи с  сущностью *Groups*. Может быть пустым если исполнитель не состоит в группе |

* Сущность *Groups* содержит описание группы. Имеет связь «один ко многим» с сущностью *Performers.* Информация о сущности представлена в таблице 1.3.

Таблица 1.3 – Описание атрибутов сущности *Groups*

|  |  |  |
| --- | --- | --- |
| **Название атрибута** | **Тип данных** | **Примечание** |
| 1 | 2 | 3 |
| *Id* | Числовой | Первичный ключ,  идетификатор |
| *Description* | Текстовый | Описание группы |

* Сущность *Employees* содержит информацию о сотрудниках радиостанции и стоит на стороне отношения «многие» с сущностью *AspNetUsers* и на стороне отношения «один» с сущностью *Positions*. Подробная информация о сущности представлена в таблице 1.4.

Таблица 1.4 – Описание атрибутов сущности *Employees*

|  |  |  |
| --- | --- | --- |
| **Название атрибута** | **Тип данных** | **Примечание** |
| 1 | 2 | 3 |
| *Id* | Числовой | Первичный ключ,  идетификатор |
| *WorkTime* | Числовой | Время работы сотрудника |
| *Education* | Текстовый | Образование сотрудника |
| *PositionId* | Числовой | Внешний ключ для связи с сущностью *Positions* |

Продолжение таблицы 1.4

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| *AspNetUserId* | Текстовый | Внешний ключ для связи с  сущностью *AspNetUsers* |

* Сущность *Positions* содержит информацию о должностях сотрудников радиостанции, стоит на стороне отношения «один» с сущностью *Employees.* Подробная информация о сущности представлена в таблице 1.5.

Таблица 1.5 – Описание атрибутов сущности *Positions*

|  |  |  |
| --- | --- | --- |
| **Название атрибута** | **Тип данных** | **Примечание** |
| 1 | 2 | 3 |
| *Id* | Числовой | Первичный ключ,  идетификатор |
| *Name* | Текстовый | Название должности |

* Сущность *Genres* содержит информацию о жанрах, стоит на стороне отношения «один» с сущностью *Records*.Подробная информация о сущности представлена в таблице 1.6.

Таблица 1.6 – Описание атрибутов сущности *Genres*

|  |  |  |
| --- | --- | --- |
| **Название атрибута** | **Тип данных** | **Примечание** |
| 1 | 2 | 3 |
| *Id* | Числовой | Первичный ключ,  идетификатор |
| *GenreName* | Текстовый | Название жанра |
| *Discription* | Текстовый | Описание жанра |

* Сущность *Broadcasts* содержит информацию о трансляциях на радиостанции*.* Имеет связь «один к одному» с сущностью *Records.* Подробная информация о сущности представлена в таблице 1.7.

Таблица 1.7 – Описание атрибутов сущности *Broadcasts*

|  |  |  |
| --- | --- | --- |
| **Название атрибута** | **Тип данных** | **Примечание** |
| 1 | 2 | 3 |
| *Id* | Числовой | Первичный ключ,  идетификатор |
| *DateAndTime* | Дата и время | Дата и время трансляции |

Продолжение таблицы 1.7

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| *EmployeeId* | Числовой | Внешний ключ для связи с  сущностью *Employees* |
| *RecordId* | Числовой | Внешний ключ для связи с  сущностью *Records* |

* Сущность *HomePageImages* содержит информацию о популярных альбомах*.* Подробная информация о сущности представлена в таблице 1.8.

Таблица 1.8 – Описание атрибутов сущности *HomePageImages*

|  |  |  |
| --- | --- | --- |
| **Название атрибута** | **Тип данных** | **Примечание** |
| 1 | 2 | 3 |
| *Id* | Числовой | Первичный ключ,  идетификатор |
| *SrcImg* | Текстовый | Путь к картинке альбома |
| *ImgCaption* | Текстовый | Описание альбома |

Также имеется ряд встроенных сущностей *ASP.NET Core Identity*, прямое проектирование которых не производилось, за исключением добавления ряда атрибутов для сущности *AspNetUsers*:

*Name* – текстовый тип, имя пользователя;

*Surname* – текстовый тип, фамилия пользователя;

*MiddleName* – текстовый тип, отчество пользователя.

Разбиение данных на сущности соответствует требованиям как минимум третьей нормальной формы:

* все поля являются простыми и содержат только скалярные значения, каждое поле хранит одно единственное значение;
* соответствует второй нормальной форме так как приведена к первой нормальной форме, отсутствуют частичные зависимости. Это означает, что каждый неключевой атрибут (поле) неприводимо зависит от первичного ключа (ключа отношения).
* Соответствует третьей нормальной форме так, как находится во второй нормальной форме, и отсутствуют транзитивные функциональные зависимости неключевых атрибутов от ключевых.

Диаграмма базы данных представлена на рисунке 1.1.

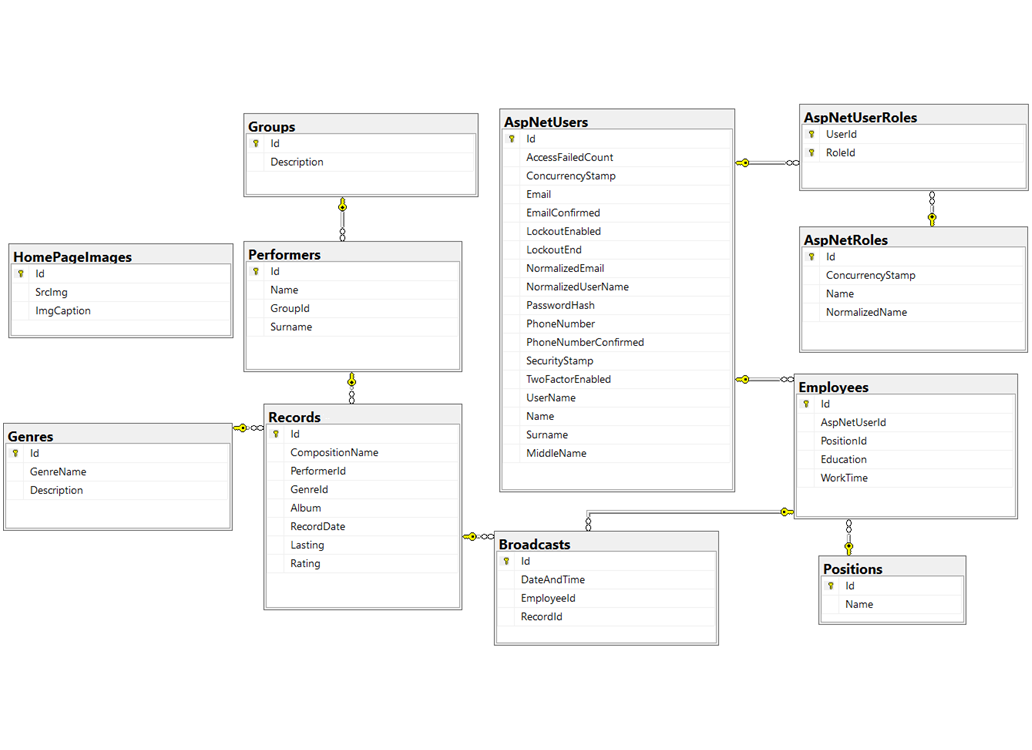


Рисунок 1.1 – Диаграммы базы данных

* 1. **Физическая структура базы данных программного обеспечения «Радиостанция»**

База данных была разработана в СУБД *MS SQL Server* и состоит из двух файлов:

* первичный – содержит сведения, необходимые для запуска базы данных, и ссылки на другие файлы в базе данных, имеет расширение *MDF* и начальный размер 8 мегабайт с функцией автоматического увеличения до 64 мегабайт;
* журнал транзакций – содержит информацию для восстановления базы данных, имеет расширение *LDF* и начальный размер 8 МБ с функцией автоматического увеличения до 64 МБ.

Ниже представлена структура таблиц разработанной базы данных.

Структура таблицы *Records*:

* *Id int not null identity(1,1) primary key,*
* *СompositionName varchar(450) not null,*
* *PerformerId int not null,*
* *GenreId int not null,*
* *Album varchar(450) not null,*
* *RecordDate date not null,*
* *Lasting int not null,*
* *Rating decimal(2,1) not null.*

Структура таблицы *Positions*:

* *Id int not null Primary key,*
* *Name varchar(50) not null.*

Структура таблицы *Performers*:

* *Id int not null identity(1,1) primary key,*
* *Name varchar(50) not null,*
* *Surname varchar(50) not null,*
* *GroupId int.*

Структура таблицы *Groups*:

* *Id int not null identity(1,1) primary key,*
* *Description varchar(50).*

Структура таблицы *Groups*:

* *Id int not null identity(1,1) primary key,*
* *GenreName varchar(50) not null,*
* *Description varchar(450).*

Структура таблицы *Employees*:

* *Id int not null identity(1,1) primary key,*
* *AspNetUserId nvarchar(450) not null,*
* *PositionId int not null,*
* *Education nvarchar(450),*
* *WorkTime int.*

Структура таблицы *Broadcasts*:

* *Id int not null identity(1,1) primary key,*
* *DateAndTime Datetime,*
* *EmployeeId int not null,*
* *RecordId int not null.*

Структура таблицы *HomePageItems*:

* *Id int not null Primary key identity,*
* *srcImg nvarchar(450),*
* *ImgCaption nvarchar(450).*

Также при разработке базы данных добавлены ограничения внешнего ключа для обеспечения целостности данных, с запретом на удаление при существовании связанных данных.

1. **АППАРАТНОЕ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ**

## 2.1 Общие сведения.

Разработанное приложение предназначено для работы с большим потоком данных и использует СУБД *MS SQL Server* для хранения данных, для корректной работы данной СУБД необходимо соблюдение следующих требований:

* рекомендуется минимум 4 ГБ свободного места на диске, требования к месту на диске определяются набором устанавливаемых компонентов *SQL Server* и могут отличаться в зависимости от выбранных компонентов;
* для экспресс-выпуска минимальный объем оперативной памяти: 512 МБ, для всех остальных: 1 ГБ, рекомендованный: 1 ГБ и 4 ГБ соответственно;
* процессор x64 с тактовой частотой не ниже 2 ГГц.

Сервер приложения реализован с помощью технологии *ASP.NET Core* и имеет следующие аппаратные требования:

* минимум 4 ГБ свободного места на диске;
* 1 ГБ оперативной памяти (рекомендуется 4 ГБ и выше);
* процессор x86/x64 с тактовой частотой не ниже 2 ГГц.

В качестве клиента может выступать любой браузер, требования могут варьироваться в зависимости от выбранного браузера, в общем случае необходимо соблюдение следующих требований:

* минимум 1ГБ свободного места на диске;
* 1 ГБ оперативной памяти;
* процессор x86/x64 с тактовой частотой не ниже 1 ГГц.

Требования к программному обеспечению для работы *SQL* *Server*:

* операционная система Windows 8 и выше, поддерживаемые операционные системы для *SQL* *Server* содержат встроенное сетевое программное обеспечение необходимое для работы *SQL* *Server*;
* наличие .*NET* *Framework* версии 4.1 и выше.

Для работы сервера приложения необходимо наличие следующего программного обеспечения:

* 32 или 64-разрядные системы, для *Linux*: начиная с *Ubuntu* 16.01, для *macOS* начиная с версии 10.1;
* доступность сетевых протоколов: *TCP/IP*, поддерживающих именованные каналы.

## Настройка приложения.

Перед началом установки приложения и базы данных на сервер необходимо убедиться в том, что они удовлетворяют минимальным системным требованиям.

Если база данных для приложения отсутствует воспользоваться проектом базы данных поставляемым вместе с приложением и опубликовать его на необходимом сервере в СУБД *MS SQL Server*, либо использовать заранее подготовленный скрипт базы данных, содержащий частичную инициализацию данных. Если база данных уже существует никаких дополнительных действий не требуется.

После настройки базы данных необходимо получить строку подключения и добавить ее в конфигурационный файл приложения *appsettings.json*, как показано на рисунке 2.1

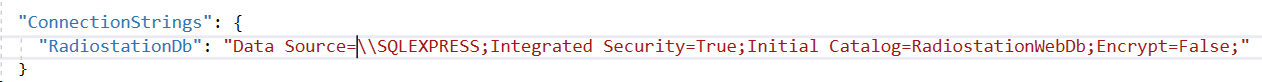


Рисунок 2.1 – Добавление строки подключения к базе в

конфигурационный файл

После настройки строки подключения, приложение может быть размещено на любом удобном сервисе, поддерживающем *ASP.NET Core*. Например, можно воспользоваться сервисом *Azure* от *Microsoft*, для публикации приложения достаточно иметь подписку на сервис *Azure*.

Открытие домашней страницы приложения после его настройки свидетельствует о его корректной работе.

Дополнительные возможности в приложении отсутствуют.

При возникновении ошибок подключения к базе данных проверить правильность указанных данных в конфигурационном файле и перезапустить приложение.

1. **СТРУКТУРА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ «РАДИОСТАНЦИЯ»**

## 3.1 Архитектура программного обеспечения «Радиостанция»

Общая архитектура разработанного приложения организуется при использовании *MVC* паттерна при помощи взаимодействия моделей, контроллеров и представлений и представлена тремя составными частями:

* модели – описывают используемые данные на уровне представления;
* представления – отвечают за пользовательский интерфейс, отображая данные представленные в моделях;
* контроллеры – содержат логику обработки запросов пользователя.

Архитектура уровня представления представлена на рисунке 3.1.

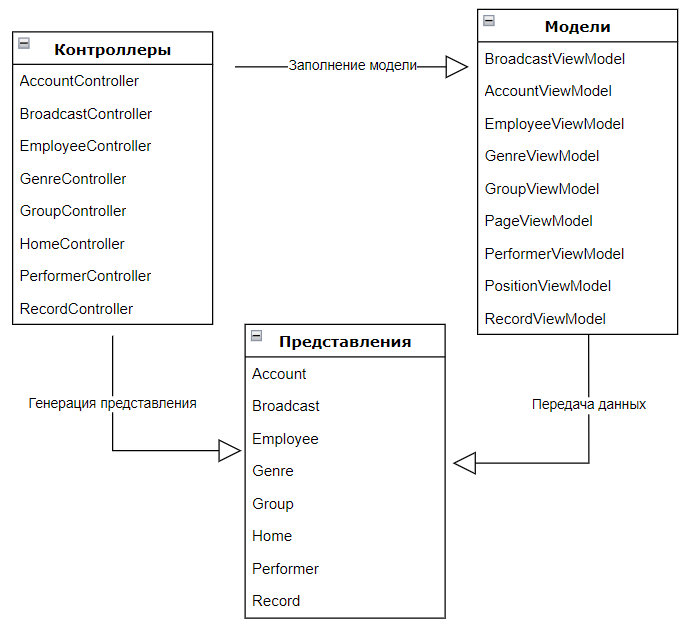


Рисунок 3.1 – Архитектура уровня представления

Общая архитектура приложения заключается в том, что действия пользователя производятся на представлении, после этого представления передают вызов к контроллерам, контроллеры в свою очередь обращаются к изменению модели, а модели сообщают об изменениях в представление.

* 1. **Структура контроллеров приложения «Радиостанция»**

В приложении имеются следующие контроллеры:

* *AccountController –* разработан для предоставления возможности регистрации, авторизации, отображения, удаления, редактирования пользователей, а также для управления ролями пользователей;
* *BroadcastController* разработан для предоставления возможности отображения, создания, удаления, редактирования радиотрансляций;
* *EmployeeController* разработан для предоставления возможности отображения сотрудников;
* *HomeController* разработан для отображения домашней страницы и страницы об ошибке;
* *GenreController* разработан для предоставления возможности отображения, создания, удаления, редактирования жанров;
* *GroupController* разработан для предоставления возможности отображения, создания, удаления, редактирования музыкальных групп;
* *PerformerController* разработан для предоставления возможности отображения, создания, удаления, редактирования музыкальных исполнителей;
* *RecordController* разработан для предоставления отображения, создания, удаления, редактирования музыкальных композиций.

*UML*-диаграмма контроллеров представлена на рисунке 3.2.

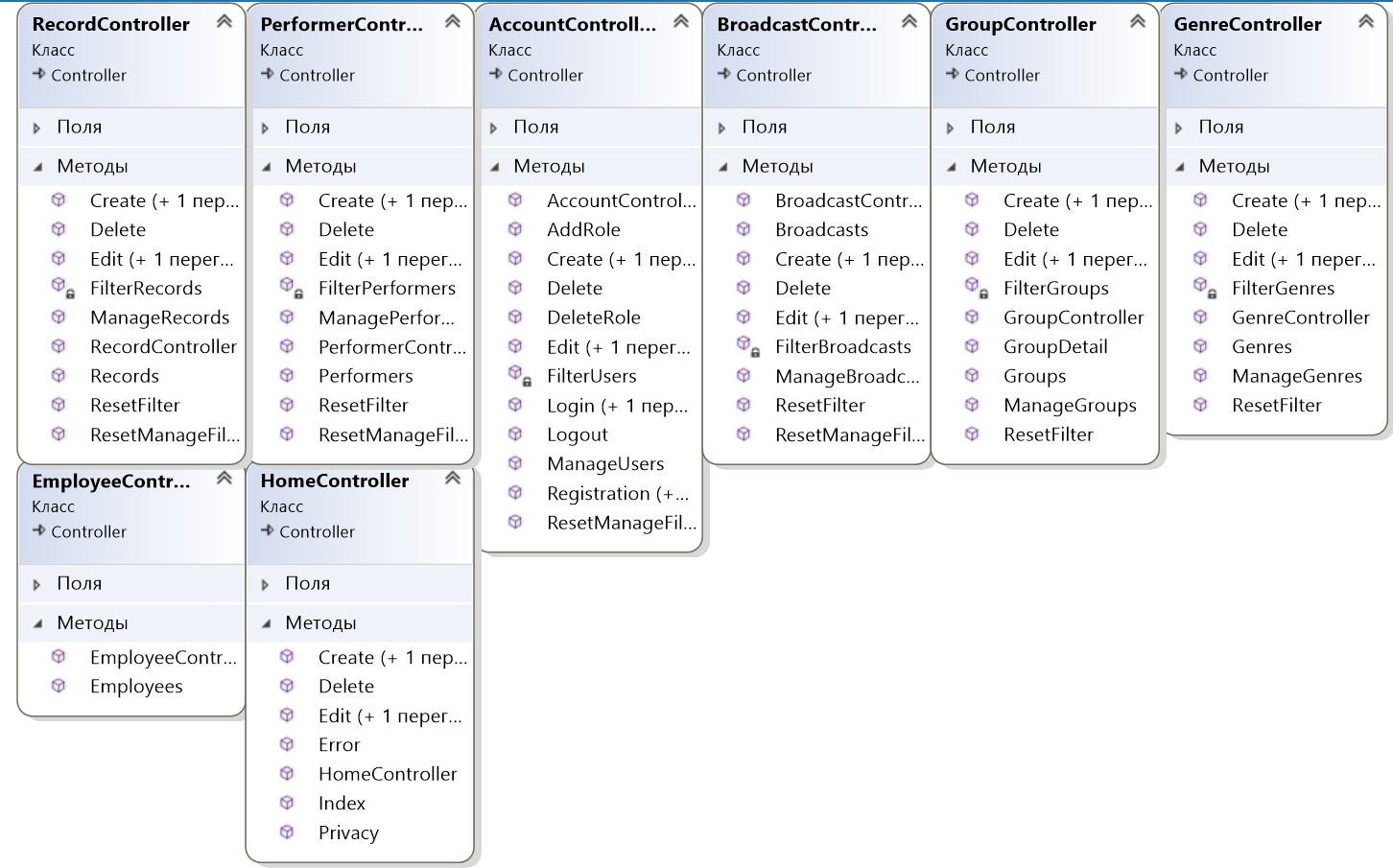


Рисунок 3.2 – *UML*-диаграмма контроллеров

* 1. **Структура представлений приложения «Радиостанция»**

Для контроллера *AccountController* в папке *Account* присутствуют следующие представления:

* *Login* представляет собой представление, которое отображает страницу для авторизации пользователя;
* *Registration* представляет собой представление, которое отображает страницу для регистрации пользователя;
* *Create* представляет собой представление, которое отображает страницу для создания пользователя администратором приложения;
* *ManageUsers* представляет собой представление, которое отображает страницу для просмотра пользователей приложения, с возможностью редактирования;

Для контроллера *BroadcastController* в папке *Broadcast* присутствуют следующие представления:

* *Create* представляет собой представление, которое отображает страницу для создания радиотрансляции;
* *Edit* представляет собой представление, которое отображает страницу для редактирования информации о радиотрансляции;
* *ManageBroadcasts* представляет собой представление, которое отображает страницу для просмотра и редактирования радиотрансляции;
* *Broadcasts* представляет собой представление, которое отображает страницу для просмотра радиотрансляции;

Для контроллера *EmployeeController* в папке *Employee* присутствуют следующие представления:

* *Employees* представляет собой представление, которое отображает страницу для просмотра сотрудников.

Для контроллера *HomeController* в папке *Home* присутствуют следующие представления:

* *Index* представляет собой представление, которое отображает домашнюю страницу приложения, с информацией о популярных альбомах и их описанию;
* *Error* представляет собой представление, которое отображает страницу ошибки.
* *Edit* представляет собой представление, которое отображает страницу для редактирования информации о популярных альбомах с возможностью смены картинки;
* *Create* представляет собой представление, которое отображает страницу для добавления информации популярных альбомах с возможностью выбора картинки;

Для контроллера *GenreController* в папке *Genre* присутствуют следующие представления:

* *Create* представляет собой представление, которое отображает страницу для создания жанра;
* *Edit* представляет собой представление, которое отображает страницу для редактирования информации о жанре;
* *ManageGenres* представляет собой представление, которое отображает страницу для просмотра и редактирования жанров;
* *Genres* представляет собой представление, которое отображает страницу для просмотра жанров;

Для контроллера *GroupController* в папке *Group* присутствуют следующие представления:

* *Create* представляет собой представление, которое отображает страницу для создания группы;
* *Edit* представляет собой представление, которое отображает страницу для редактирования информации о группе;
* *ManageGroups* представляет собой представление, которое отображает страницу для просмотра и редактирования групп;
* *Groups* представляет собой представление, которое отображает страницу для просмотра групп;
* *GroupDetail* представляет собой представление, которое отображает страницу для просмотра исполнителей групп;

Для контроллера *PerformerController* в папке *Performer* присутствуют следующие представления:

* *Create* представляет собой представление, которое отображает страницу для создания музыкального исполнителя;
* *Edit* представляет собой представление, которое отображает страницу для редактирования информации о музыкальном исполнителе;
* *ManagePerformers* представляет собой представление, которое отображает страницу для просмотра и редактирования музыкальных исполнителей;
* *Performers* представляет собой представление, которое отображает страницу для просмотра музыкальных исполнителей;

Для контроллера *RecordController* в папке *Record* присутствуют следующие представления:

* *Create* представляет собой представление, которое отображает страницу для создания музыкальной композиции;
* *Edit* представляет собой представление, которое отображает страницу для редактирования информации о музыкальной композиции;
* *ManageRecords* представляет собой представление, которое отображает страницу для просмотра и редактирования музыкальных композиций;
* *Records* представляет собой представление, которое отображает страницу для просмотра музыкальных композиций;

Также в папке *Shared* находятся представления, используемые как макет:

* \_*Layout* представляет собой представление, которое содержит макет страницы с навигационным меню сайта и подвалом сайта;

## Структура моделей и моделей представлений приложения «Радиостанция»

Приложение представлено рядом моделей, которые проецируют сущности базы данных:

* *Broadcast* представляет собой информацию о радиотрансляции;
* *Record* представляет собой информацию о музыкальной композиции;
* *Employee* представляет собой информацию о сотруднике;
* *Position* представляет собой информацию о должности сотрудника радиостанции;
* *Genre* представляет собой информацию о музыкальном жанре;
* *Group* представляет собой информацию о музыкальной группе;
* *Performer* представляет собой информацию о музыкальном исполнителе;

По той причине, что данные хранимые в базе данных и отображаемые данные имеют разный вид, большинство представлений имеет собственные модели. Модели представлений, имеющиеся в приложении:

* *UserEmployeeEditViewModel* содержит свойства с атрибутами валидации, которые необходимо заполнить при редактировании пользователя если он является сотрудником, а также коллекции класса *SelectListItem* для выбора данных из выпадающего списка;
* *LoginViewModel* содержит свойства с атрибутами валидации, которые необходимо заполнить при авторизации в приложении;
* *RegistrationViewModel* содержит свойства с атрибутами валидации, которые необходимо заполнить при регистрации пользователя, а также коллекции класса *SelectListItem* для выбора данных из выпадающего списка;
* *СreateBroadcastViewModel* содержит свойства с атрибутами валидации, которые необходимо заполнить при добавлении радиотрансляции, а также коллекцию класса *SelectListItem* для выбора данных из выпадающего списка;
* *EditBroadcastViewModel* содержит свойства с атрибутами валидации, которые необходимо заполнить при редактировании радиотрансляции, а также коллекцию класса *SelectListItem* для выбора данных из выпадающего списка;
* *BroadcastViewModel* содержит данные для отображения подробной информации о радиотрансляциях, с заменой внешних ключей на данные из связанных таблиц, с возможностью постраничного просмотра;
* *RecordViewModel* содержит данные для отображения информации о музыкальных композициях, с заменой внешних ключей на данные из связанных таблиц, с возможностью постраничного просмотра;
* *СreateRecordViewModel* содержит свойства с атрибутами валидации, которые необходимо заполнить при добавлении музыкальной композиции, а также коллекцию класса *SelectListItem* для выбора данных из выпадающего списка;
* *EditRecordViewModel* содержит свойства с атрибутами валидации, которые необходимо заполнить при редактировании музыкальной композиции, а также коллекцию класса *SelectListItem* для выбора данных из выпадающего списка;
* *PerformerViewModel* содержит данные для отображения информации о музыкальных исполнителях, с заменой внешних ключей на данные из связанных таблиц, с возможностью постраничного просмотра;
* *СreatePerformerViewModel* содержит свойства с атрибутами валидации, которые необходимо заполнить при добавлении музыкального исполнителя, а также коллекцию класса *SelectListItem* для выбора данных из выпадающего списка;
* *EditPerformerViewModel* содержит свойства с атрибутами валидации, которые необходимо заполнить при редактировании музыкального исполнителя, а также коллекцию класса *SelectListItem* для выбора данных из выпадающего списка;
* *GroupDetailViewModel* содержит данные для отображения информации о музыкальных группах и их исполнителях, с заменой внешних ключей на данные из связанных таблиц, с возможностью постраничного просмотра;
* *EmployeeViewModel* содержит данные для отображения информации о сотрудниках, с заменой внешних ключей на данные из связанных таблиц, с возможностью постраничного просмотра;
* *PageViewModel* представляет данные для реализации разделения на страницы;
* *PageItemsViewModel* представляет обобщенный класс, который хранит модель страницы и коллекцию элементов заданного типа.
* *SortViewModel* представляет обобщенный класс, который хранит модель страницы и коллекцию отсортированных элементов.

## Структура системы аутентификации и авторизации пользователей приложения «Радиостанция»

В качестве системы авторизации и аутентификации выбрана имеющаяся встроенная система: *ASP.NET Core Identity*. Для реализации данной системы создан класс *ApplicationDbContext* который наследуется от *IdentityDbContext<ApplicationUser>*, также создана дополнительная модель *ApplicationUser* для расширения информации которая будет храниться о пользователе.

Для обеспечения удобства работы с данными, в качестве места хранения таблиц необходимых *ASP.NET Core Identity* выбрана основная база данных приложения, для этого в основную базу был добавлен ряд встроенных таблиц и созданы связи между таблицами *ASP.NET Core Identity* и таблицами приложения.

В классе *Startup* настроено сохранение авторизационных данных пользователя в куках браузера, с помощью данной функции в течении некоторого времени пользователь может начать новую сессию без необходимости заново проходить авторизацию.

На некоторые из маршрутов контроллеров добавлен авторизационный атрибут, который позволяет защитить приложение от несанкционированного доступа и выполнить разделение функционала приложения по ролям.

Для удобства работы с ролями добавлен класс *Roles* который содержит константные значения ролей. Роли, имеющиеся в приложении:

* *Admin* представляет пользователя, который имеет полный доступ к приложению и возможностью редактирования полной информации;
* *HR\_Employee(Human Resources)* представляет пользователя, который может производить редактирование персонала радиостанции;
* *User* представляет пользователя, который может видеть подробную информацию о радиостанции;
* *Employee* представляет пользователя, который является сотрудником радиостанции и может просматривать, и редактировать информацию на радиостанции;

Исходный код программного обеспечения представлен в приложении А. Структура *web*-приложения представлена в приложении Б.