МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий   
Кафедра Программной инженерии   
Специальность 1-98 01 03 Программное обеспечение информационной безопасности мобильных систем  
Направление специальности 1-98 01 03 «Программное обеспечение информационной безопасности мобильных систем»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА КУРСОВОГО ПРОЕКТА**

по дисциплине «Современные технологии программирования мобильных систем»

Тема Программное средство «Лайн-ап для концертов»

Исполнитель

студентка 2 курса 7 группы Бобрович Глеб Сергеевич

(Ф.И.О.)

Руководитель работы ассистент Панченко О.Л.

(учен. степень, звание, должность, подпись, Ф.И.О.)

Курсовой проект защищен с оценкой ­

Руководитель Панченко О.Л.

(подпись)

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования   
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий   
Кафедра программной инженерии

Утверждаю

Заведующий кафедрой ПИ

\_\_\_\_\_\_\_\_\_\_\_\_ \_\_ Н.В. Пацей

подпись инициалы и фамилия

“\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2021г.

**ЗАДАНИЕ**

**к курсовому проектированию**

**по дисциплине** "Объектно-ориентированное программирование"

|  |  |
| --- | --- |
| Специальность: 1-98 01 03 Программное обеспечение информационной безопасности мобильных систем  Студент: Бобрович Г.С. | Группа:\_\_7\_\_ |
| **Тема: Программное средство «Клавиатурный тренажёр»** | |

**1. Срок сдачи студентом законченной работы**: "20 мая 2022 г."

**2. Исходные данные к проекту:**

**2.1**. Функционально ПС поддерживает:

* Функции администратора сервиса:
  + Поддерживать работу c базой данных;
  + Регистрировать группы;
  + Устанавливать порядок выступления;
  + Добавлять информацию об участниках группы;
  + Добавлять информацию о релизах;
* Функции клиента:
  + Выполнять регистрацию и авторизацию;
  + Поиск группы по определенным параметрам;

**2.2.** При выполнении курсового проекта необходимо использовать принципы проектирования ООП. Приложение разрабатывается под ОС Windows и представляет собой настольное приложение (desktop). Отображение, бизнес-логика должны быть максимально независимы друг от друга для возможности расширения. Диаграммы вариантов использования, классов реализации задачи, взаимодействия разработать на основе UML. Язык разработки проекта – C#. Управление программой должно быть интуитивно понятным и удобным. При разработке использовать несколько наиболее подходящих шаблонов проектирования ПО.

**3. Содержание расчетно-пояснительной записки**

(перечень вопросов подлежащих разработке)

* Введение
* Постановка задачи и обзор литературы (алгоритмы решения, обзор прототипов, актуальность задачи)
* Проектирование архитектуры проекта (структура модулей, классов).
* Разработка функциональной модели и модели данных ПС (выполняемые функции)
* Тестирование
* Заключение
* Список используемых источников
* Приложения

**4. Форма представления выполненной курсовой работы:**

* + Теоретическая часть курсового проекта должны быть представлены в формате docx. Оформление записки должно быть согласно выданным правилам.
  + Листинги программы представляются частично в приложении.
  + Пояснительную записку, листинги, проект (инсталляцию проекта) необходимо загрузить диск, указанный преподавателем.

#### Календарный план

|  |  |  |  |
| --- | --- | --- | --- |
| №  п/п | Наименование этапов курсового проекта | Срок выполнения этапов проекта | Примечание |
| 1 | Введение | 19.02.2022 |  |
| 2 | Аналитический обзор литературы по теме проекта. Изучение требований, определение вариантов использования | 12.03.2022 |  |
| 3 | Анализ и проектирование архитектуры приложения (построение диаграмм, проектирование бизнес-слоя, представления и данных) | 26.03.2022 |  |
| 4 | Проектирование структуры базы данных. Разработка дизайна пользовательского интерфейса | 2.04.2022 |  |
| 5 | Кодирование программного средства | 23.04.2022 |  |
| 6 | Тестирование и отладка программного средства | 30.04.2022 |  |
| 7 | Оформление пояснительной записки | 7.05.2022 |  |
| 9 | Сдача проекта | 20.05.2022 |  |

**5. Дата выдачи задания \_\_\_\_**14.02.2022**\_\_\_\_**

Руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *О.Л. Панченко*

(подпись)

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (дата и подпись студента)

# СОДЕРЖАНИЕ

445454

[СОДЕРЖАНИЕ 4](#_Toc105340893)

[ВВЕДЕНИЕ 5](#_Toc105340894)

[1 Анализ прототипов и литературных источников 6](#_Toc105340895)

[2 Анализ требований к программному средству и разработка функциональных требований 9](#_Toc105340896)

[3 Проектирование программного средства 11](#_Toc105340897)

[3.1 Проектирование базы данных 11](#_Toc105340898)

[3.2 Проектирование доступа к базе данных 15](#_Toc105340899)

[3.3 Проектирование вариантов использования 15](#_Toc105340900)

[4 Реализация программного средства 17](#_Toc105340901)

[4.1 Реализация сущностей 17](#_Toc105340902)

[4.2 Реализация авторизации и регистрации 17](#_Toc105340903)

[4.3 Реализация основных функций 18](#_Toc105340904)

[5 Тестирование 20](#_Toc105340905)

[5.1 Тестирование регистрации и входа 20](#_Toc105340906)

[5.2 Тестирование вывода данных 23](#_Toc105340907)

[6 Руководство по использованию 25](#_Toc105340908)

[6.1 Общее руководство по использованию программного продукта 25](#_Toc105340909)

[6.2 Поддержка пользовательской настройки интерфейса 32](#_Toc105340910)

[ЗАКЛЮЧЕНИЕ 35](#_Toc105340911)

[СПИСОК ЛИТЕРАТУРЫ 36](#_Toc105340912)

[ПРИЛОЖЕНИЕ А 37](#_Toc105340913)

[ПРИЛОЖЕНИЕ Б 38](#_Toc105340914)

[ПРИЛОЖЕНИЕ В 39](#_Toc105340915)

# ВВЕДЕНИЕ

В наше время довольно часто проходят различные мероприятия и фестивали. Обычно, это музыкальные концерты, на которых есть определенный список приглашенных групп, их информация, состав групп и их трек-листы. Дабы упростить процесс работы с группами, это все объединено в лайн-ап, в котором предусмотрено все вышеперечисленное.

Существует большое количество web-решений, однако многим удобнее пользоваться десктопными приложениями. Хорошим инструментом для разработки такого приложения являются Windows Forms и WPF.

Задачей данного курсового проекта является реализация десктопного приложения с использованием технологий WPF и SQL Server.

Курсовой проект включает в себя 7 разделов: введение, содержание, обзор технических методов и программных средств разработки приложения, последовательность разработки содержания и структуры приложения, описание хода работы программного средства с копиями экрана, заключение, список использованной литературы, приложение.

В процессе работы над проектом были поставлены следующие задачи:

* Реализация системы регистрации музыкальных групп.
* Добавление участников группы.
* Добавление трек-листов.
* Сортировка и поиск групп по определенным параметрам.

Основная цель курсового проекта — разработать пользовательский интерфейс, серверную часть и базу данных. Развить навыки объектно-ориентированного программирования на языке C#.

# 1 Анализ прототипов и литературных источников

Для того чтобы сформировать требования к разрабатываемому приложению и узнать, что пользователь хочет получить от приложения в итоге одним из этапов разработки является анализ прототипов и литературных источников. На данный момент большая часть схожих по теме приложений перешли в web-версии. Вот некоторые из них:

«Ural Music Night» – интернет-ресурс для просмотра информации о концерте, расписания выхода на сцену, списков групп и информации о них.

Ресурс поддерживает функции:

* Просмотр информации об артистах (участники, песни, прочая информация).
* Показ расписания концерта, с временем выхода каждой группы.
* Регистрация и вход пользователей.
* Выбор языка.

На рисунке 1.1 представлен интерфейс программы «Ural Music Night».

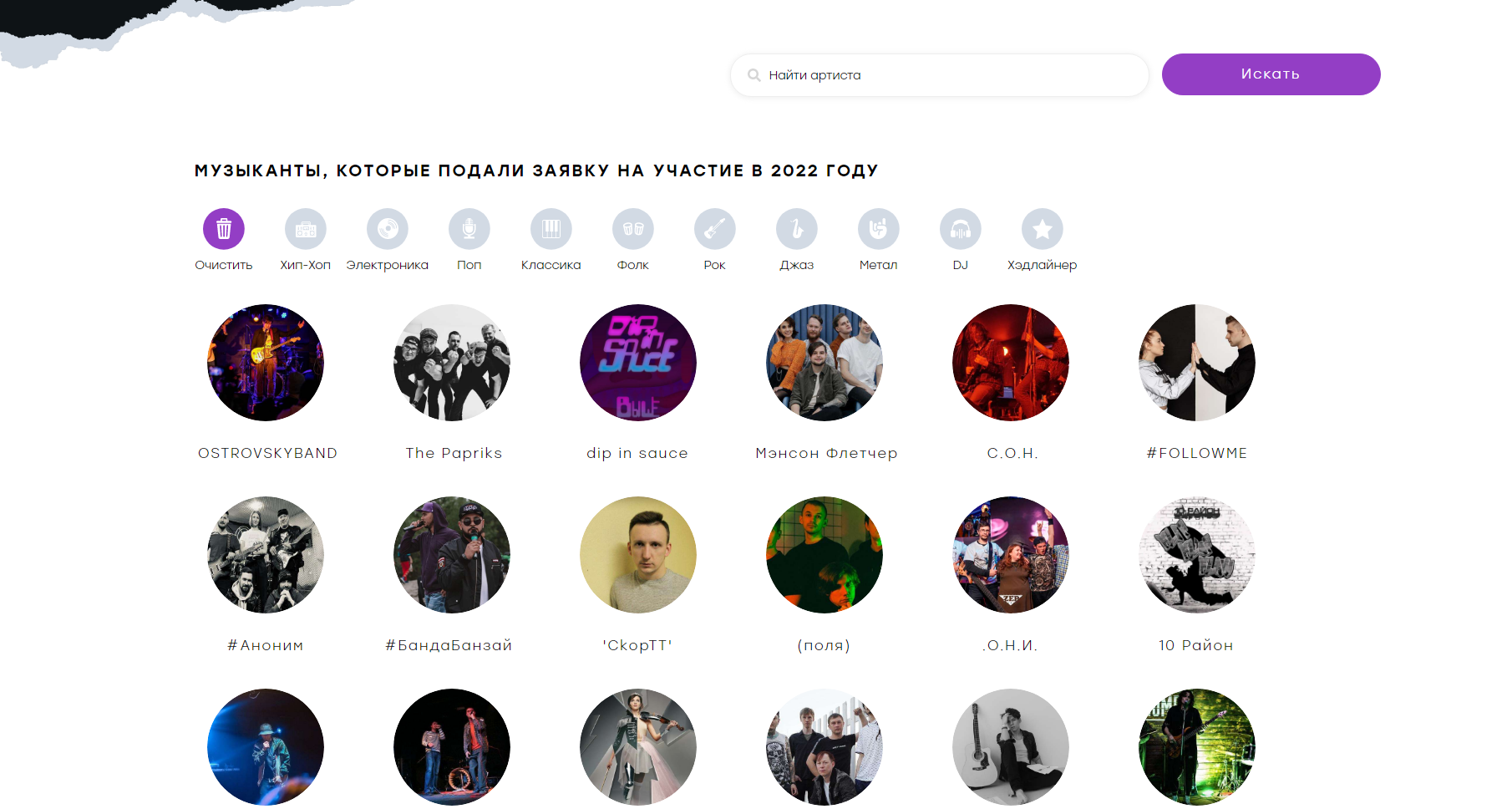


Рисунок 1.1 ­­– Интерфейс «Ural Music Night»

Как видно из приложенного выше скриншота, интерфейс выглядит вполне приемлемо. Также, реализованы все необходимые функции для приложения такого рода.

«Boston Calling» – интернет-ресурс, один из самых популярных сайтов для просмотра информации о концерте за рубежом.

Особенности веб-сайта:

* Поиск групп.
* Фильтр групп по названию, времени выступления.
* Возможность добавить группу в избраное.

На рисунке 1.2 представлен интерфейс приложения.

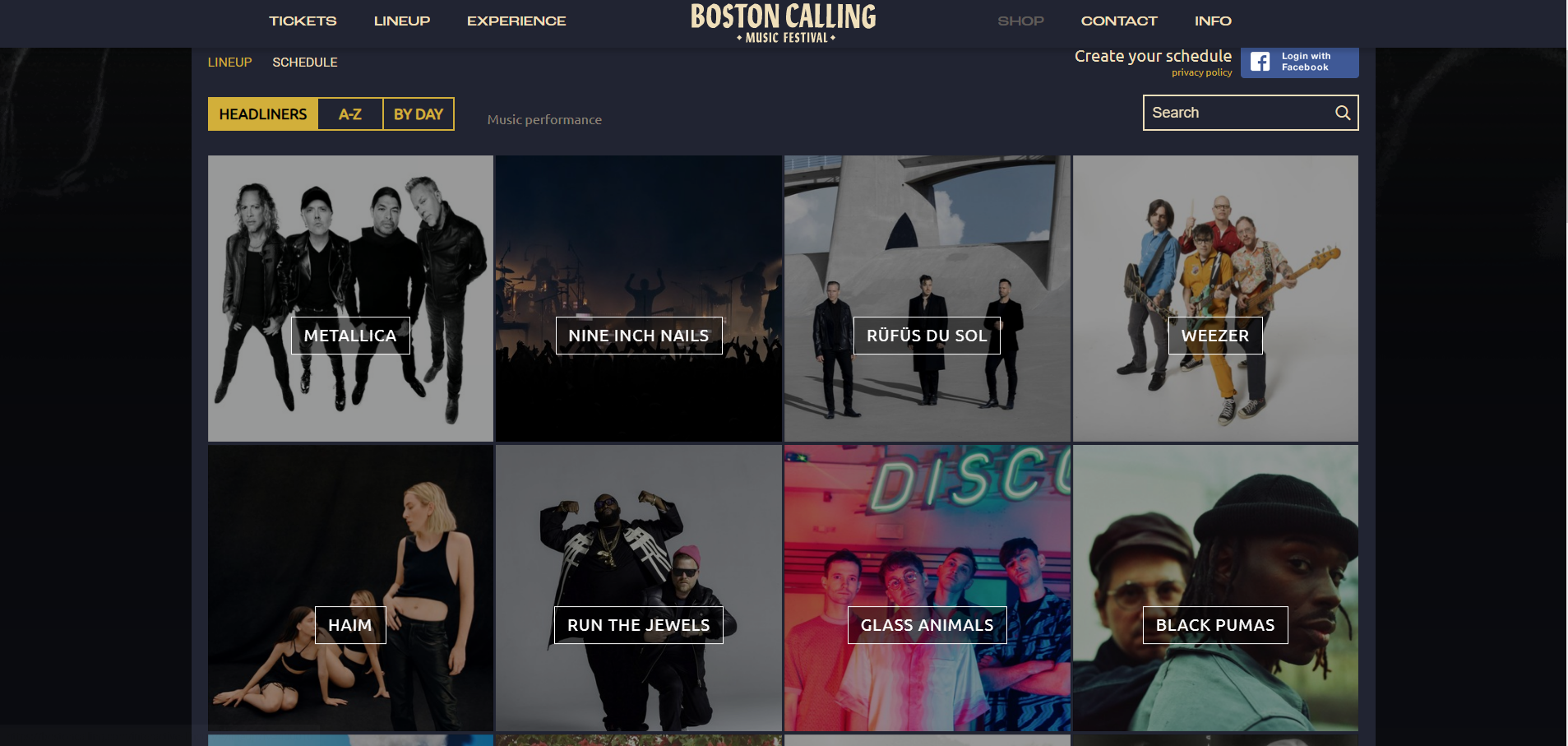


Рисунок 1.2 – Пример интерфейса веб-сайта «Boston Calling»

В разработке приложения были использованы нижеперечисленные технологии.

Windows Presentation Foundation (WPF) — это система следующего поколения для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем. C WPF можно создавать широкий спектр как автономных приложений, так и приложений, размещенных в веб-обозревателе. В основе WPF лежит векторная система визуализации, не зависящая от разрешения и созданная с расчетом на возможности современного графического оборудования. WPF расширяет базовую систему полным набором функций разработки приложений. Именно использование WPF позволило гибко управлять дизайном интерфейса, также стало возможным подключение различных сторонних пакетов и использование паттернов.

ADO.NET предоставляет собой технологию работы с данными, которая основана на платформе .NET Framework. Эта технология представляет нам набор классов, через которые мы можем отправлять запросы к базам данных, устанавливать подключения, получать ответ от базы данных и производить ряд других операций.

Основу интерфейса взаимодействия с базами данных в ADO.NET представляет ограниченный круг объектов: Connection, Command, DataReader, DataSet и DataAdapter. С помощью объекта Connection происходит установка подключения к источнику данных. Объект Command позволяет выполнять операции с данными из БД. Объект DataReader считывает полученные в результате запроса данные. Объект DataSet предназначен для хранения данных из БД и позволяет работать с ними независимо от БД. И объект DataAdapter является посредником между DataSet и источником данных. Главным образом, через эти объекты и будет идти работа с базой данных.

Однако чтобы использовать один и тот же набор объектов для разных источников данных, необходим соответствующий провайдер данных. Собственно через провайдер данных в ADO.NET и осуществляется взаимодействие с базой данных. Причем для каждого источника данных в ADO.NET может быть свой провайдер, который собственно и определяет конкретную реализацию вышеуказанных классов.

2 Анализ требований к программному средству и разработка функциональных требований

Анализ требований — это процесс сбора требований к программному обеспечению, их систематизации, документирования, анализа, выявления противоречий, неполноты, разрешения конфликтов в процессе разработки программного обеспечения.

Цель анализа требований в проектах — получить максимум информации о заказчике и специфике его задач, уточнить рамки проекта, оценить возможные риски. На этом этапе происходит идентификация принципиальных требований методологического и технологического характера, формулируются цели и задачи проекта, а также определяются критические факторы успеха, которые впоследствии будут использоваться для оценки результатов внедрения. Определение и описание требований — шаги, которые во многом определяют успех всего проекта, поскольку именно они влияют на все остальные этапы.

Различают три уровня требований к проекту:

* бизнес-требования;
* пользовательские требования;
* функциональные требования.

Бизнес-требования содержат высокоуровневые цели организации или заказчиков системы. Как правило, их высказывают те, кто финансируют проект, покупатели системы, менеджер реальных пользователей, отдел маркетинга. Курсовой проект не подразумевает наличие заказчика, который мог бы выдвинуть бизнес-требования, поэтому в качестве таких высокоуровневых требований можно рассматривать общие требования к разрабатываемому средству. К их числу относятся:

* простота и лёгкость интерфейса;
* использование принципов объектно-ориентированного программирования;
* использование системы управления базами данных (СУБД);

Весь дальнейший процесс проектирования и разработки программного средства должен находиться в очерченных бизнес-требованиями границах.

Следующими требованиями являются требования пользователей. Данные требования описывают цели и задачи, которые пользователям позволит решить система. Таким образом, в пользовательских требованиях указано, что клиенты смогут делать с помощью системы. Пользователь данного программного решения должен иметь возможность:

* Осуществлять регистрацию.
* Осуществлять вход в приложение.
* Просматривать информацию о группах.
* Видеть оставшееся время до концерта.

Администратор имеет возможность:

* Добавлять, изменять, удалять группы, их трек-листы, списки участников.
* Добавлять новые даты концертов.

После проведения анализа были выявлены следующие функциональные требования:

* При регистрации должна осуществляться валидация: имя пользователя и пароль не могут быть слишком короткими и содержать в себе специальные символы.
* Информация о музыкальных коллективах должна храниться в базе данных.

Таким образом, был проведен тщательный анализ требований к программному средству, который позволил разработать список функциональных требований. Разработка данной программной системы должна проводиться в соответствии с сформированными списком.

3 Проектирование программного средства

Проектирование программного средства — процесс создания проекта программного обеспечения. Целью проектирования является определение внутренних свойств системы и детализации её внешних свойств на основе исходных условий задачи. Исходные условия задачи уже были сформулированы во втором разделе данной пояснительной записки. Этап проектирования подразумевает их анализ.

3.1 Проектирование базы данных

Проектирование баз данных — процесс создания схемы базы данных и определения необходимых ограничений целостности.

Основные задачи проектирования базы данных:

* обеспечение хранения в БД всей необходимой информации;
* обеспечение возможности получения данных по всем необходимым запросам;
* сокращение избыточности и дублирования данных;
* обеспечение целостности базы данных.

Проектирование базы данных проводится в три этапа: концептуальное (инфологическое), логическое (даталогическое) проектирование и физическое проектирование.

Концептуальное (инфологическое) проектирование — построение семантической модели предметной области, то есть информационной модели наиболее высокого уровня абстракции. Такая модель создаётся без ориентации на какую-либо конкретную СУБД и модель данных. Термины «семантическая модель», «концептуальная модель» и «инфологическая модель» являются синонимами. Кроме того, в этом контексте равноправно могут использоваться слова «модель базы данных» и «модель предметной области» (например, «концептуальная модель базы данных» и «концептуальная модель предметной области»), поскольку такая модель является как образом реальности, так и образом проектируемой базы данных для этой реальности.

Конкретный вид и содержание концептуальной модели базы данных определяется выбранным для этого формальным аппаратом. Обычно используются графические нотации, подобные ER-диаграммам.

Основными понятиями ER-модели являются: сущность, связь и атрибут

Сущность – это реальный или представляемый объект, информация о котором должна сохраняться и быть доступна.

Связь – это графически изображаемая ассоциация, устанавливаемая между двумя сущностями. Эта ассоциация обычно является бинарной и может существовать между двумя разными сущностями или между сущностью и ей же самой (рекурсивная связь).

Атрибут сущности − это любая детaль, которая служит для уточнения, идентификации, классификации, числовой характеристики или выражения состояния сущности.

В рамках этого этапа была создана ER-модель, которая включает 5 сущностей:

* группа;
* песни;
* участники;
* картинки;
* концерт;

Также для ER-диаграммы были созданы связи, а именно: сущность группы связана с сущностями песни, картинки, участники и концерт. Для сущностей были разработаны атрибуты, например, для сущности Группа были выделены следующие атрибуты: идентификатор группы, название, жанр, рейтинг, общее количество прослушиваний, время выхода/ухода со сцены.

Логическое (даталогическое) проектирование — создание схемы базы данных на основе конкретной модели данных, например, реляционной модели данных. Для реляционной модели дaнных логическая модель — набор схем отношений, обычно с указанием первичных ключей, а также «связей» между отношениями, представляющих собой внешние ключи. На этапе логического проектирования учитывается специфика конкретной модели данных, но может не учитываться специфика конкретной СУБД.

Логическая модель базы данных представлена на рисунке 3.1

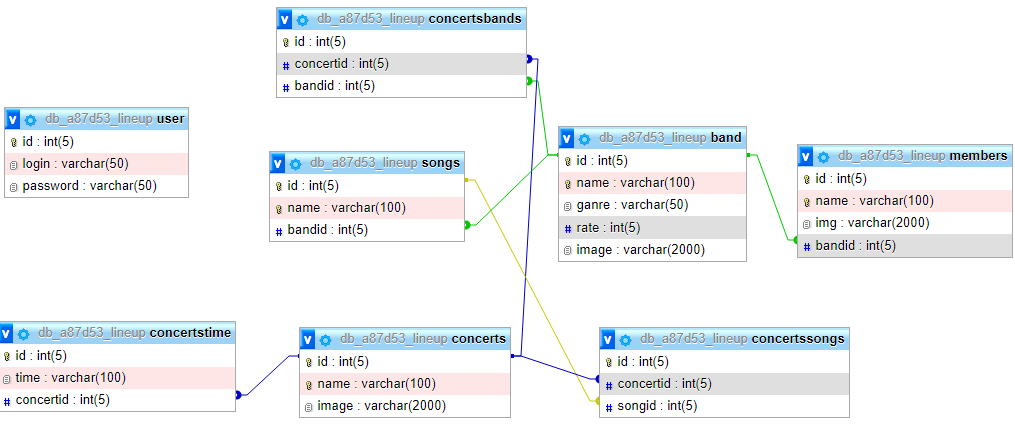


Рисунок 3.1 ­­­­­­– Логическая модель базы данных

Физическое проектирование — создание схемы базы данных для конкретно СУБД. Специфика конкретной СУБД может включать в себя ограничения на именование объектов базы данных, ограничения на поддерживаемые типы данных и т. п. Кроме того, специфика конкретной СУБД при физическом проектировании включает выбор решений, связанных с физической средой хранения данных (выбор методов управления дисковой памятью, разделение БД по файлам и устройствам, методов доступа к данным), создание индексов и т. д.

Так как в качестве СУБД была выбрана MySQL, то база данных была создана с учетом ограничений на именование объектов базы данных для выбранной СУБД.

На рисунке 3.2 представлена схема базы данных в MySQL

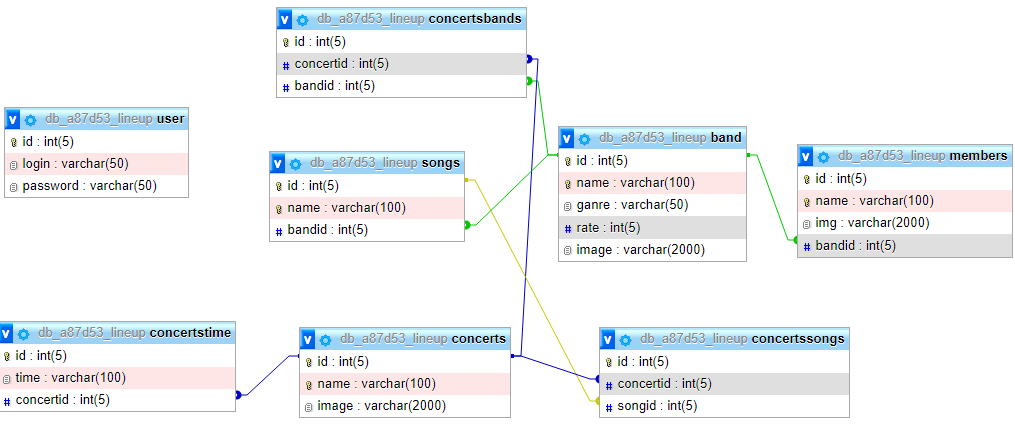


Рис. 3.2 – Схема базы данных в MySQL

База данных состоит из 8 таблиц: Band, Song, Member, User, Concerts, Concerttime, Concertband, Concertsong. Таблица Band хранит всю информацию о группах: название, жанр, рейтинг. Таблица Songs хранит информацию о песнях, которые есть у групп. Таблица Member содержит информацию об участниках группы. Таблица User содержит информацию о пользователях: логин и пароль. Таблица Concertband предназначена для связи между Band и Concert. Таблица Concert содержит идентификатор концерта, название и изображение. Таблица Concertsong предназначена для связи между Song и Concert. Concerttime хранит время концерта.

Таблица Band состоит из 5 столбцов:

* ID
* name
* genre
* rate
* image

Столбец ID содержит идентификатор группы, столбец name хранит название группы, столбец genre хранит жанр группы, столбец rate хранит информацию о рейтинге группы.

Таблица Tracks состоит из 3 столбцов:

* ID
* name
* BandID

Столбец ID содержит идентификатор песни, столбец Track\_name содержит название песни. Столбец BandID является связывающим песню с идентификатором группы

Таблица Members состоит из 4 столбцов:

* ID
* name
* image
* BandID

В столбце ID хранится идентификатор участника, в столбце name хранится имя участника, в столбце image хранится фото участника. BandID является связывающим песню с идентификатором группы

Таблица Users состоит из 3 столбцов:

* ID
* Login
* Password

В столбце ID хранится идентификатор пользователя, в столбце Login хранится логин пользователя для входа в приложение, в столбце Password содержится пароль пользователя.

Таблица Concerts состоит из 3 столбцов:

* ID
* Name
* Image

В столбце ID хранится идентификатор концерта, к нему привязаны даты имеющихся концертов. В name – название концерта, а в image – изображение.

Таблица Concertsbands состоит из 3 столбцов:

* ConcertID
* BandsID
* id

Столбец ConcertID является связующим между идентификатором концерта и данной таблицей, а столбец BandID является связующим между данной таблицей и идентификатором группы.

Таблица Concertssongs состоит из 3 столбцов:

* ConcertID
* SongID
* id

Столбец ConcertID является связующим между идентификатором концерта и данной таблицей, а столбец SongID является связующим между данной таблицей и идентификатором группы.

Таблица Concertsешьу состоит из 3 столбцов:

* ConcertID
* time
* id

Столбец ConcertID является связующим между идентификатором концерта и данной таблицей, а столбец time хранит время концерта.

3.2 Проектирование доступа к базе данных

Для доступа к базе данных была выбрана технология ADO.NET.

ADO.NET (ActiveX Data Object для .NET) — технология, предоставляющая доступ к данным и управление ими, хранящимися в базе данных или других источниках (Microsoft SQL Server, Microsoft Access, Microsoft Excel, Microsoft Outlook, Microsoft Exchange, Oracle, OLE DB, ODBC, XML, текстовые файлы), основанных на платформе .NET Framework и входящая в состав .NET Framework 2.0, представляет собой набор библиотек.

В отличие от технологии ADO, которая была в основном предназначена для тесно связанных клиент-серверных систем, ADO.NET больше нацелена на автономную работу с помощью объектов DataSet. Объекты DataSet представляют локальные копии взаимосвязанных таблиц данных, каждая из которых содержит набор строк и столбцов. Объекты DataSet позволяют вызывающей сборке (наподобие веб-страницы или программы, выполняющейся на настольном компьютере) работать с содержимым DataSet, изменять его, не требуя подключения к источнику данных, и отправлять обратно блоки измененных данных для обработки с помощью соответствующего адаптера данных. Но, пожалуй, самое фундаментальное различие между классической ADO и ADO.NET состоит в том, что ADO.NET является управляемой кодовой библиотекой, и, значит, подчиняется тем же правилам, что и любая управляемая библиотека.

Типы, составляющие ADO.NET, используют протокол управления памятью CLR, принадлежат к той же системе типов (классы, интерфейсы, перечисления, структуры и делегаты), и доступ к ним возможен с помощью любого языка .NET. Классы ADO.NET находятся в сборке System.Data.dll.

Технология ADO.NET построена так, чтобы изолировать программиста от изучения структур баз данных разных производителей, представляя поставщиков баз данных (data provider), которые инкапсулируют механизм работы с конкретной СУБД, что позволяет создавать адаптеры для любой СУБД и полностью использовать её особенности. Сделана такая абстракция для того, чтобы использовать одинаковые типы данных для работы с различными источниками данных, иметь общий подход (универсализацию) для работы с базами данных разных производителей, чтобы технология ADO.NET поддерживалась CLR.

3.3 Проектирование вариантов использования

На рисунке 3.3 представлена диаграмма использования приложения.

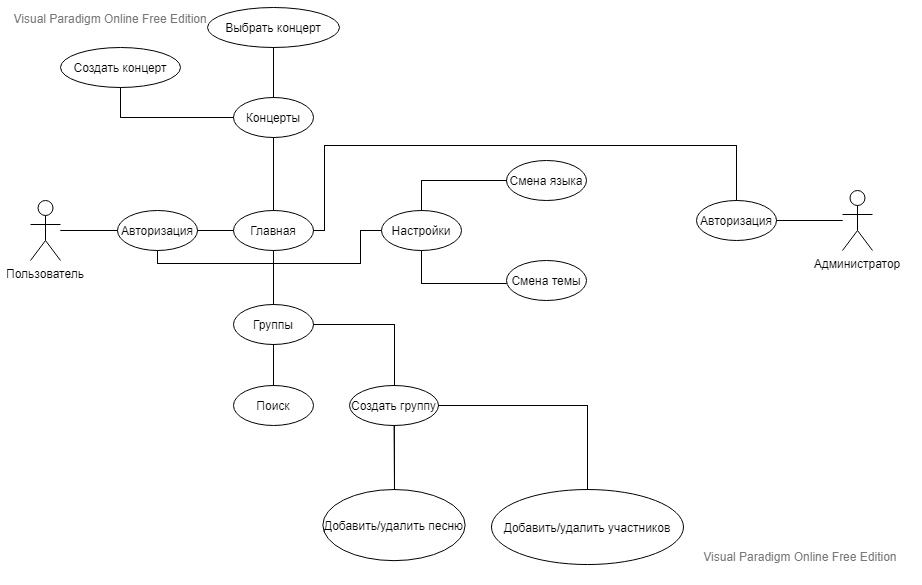


Рисунок 3.3 – Диаграмма вариантов использования для пользователя

На диаграмме видно, что пользователь при запуске приложения будет попадать на страницу входа, на которой чтобы перейти дальше необходимо ввести соответствующие данные или перейти на страницу регистрации и ввести данные для регистрации.

После входа пользователь переходит в основное окно, где ему доступен просмотр информации о группах, песнях и участниках. Пользователь может найти нужную ему группу, введя ее название. Также, пользователь может воспользоваться функционалом настроек, где ему доступны смена языка (3 на выбор) и выбор цветовой темы приложения (светлая и темная). После всех настроек и поисков, пользователь может выбрать одну из доступных дат концертов, чтобы посмотреть список выступающих групп, их состав и трек-листы. В специальном поле начнет отсчет таймер.

Если же приложением воспользуется администратор, для него будет следующий алгоритм. Вход по логину и паролю администратора, после чего открывается соответствующее окно. Тут администратор может просмотреть уже добавленные группы, песни и участников. Как и обычный пользователь, может воспользоваться окном настроек. Основной функционал окна администратора – добавление, удаление и изменение данных в приложении. Для этого отведены кнопки Группы, Песни, Участники. По нажатию одной из кнопок будут доступны поля для ввода соответствующей информации. Если необходимо удалить или изменить данные, то нужно ввести ID изменяемого объекта и заполнить форму.

4 Реализация программного средства

Следующим этапом разработки приложения является непосредственная реализация программного решения в соответствии с уже сформированными требованиями и шаблонами.

4.1 Реализация сущностей

В функциональных требованиях значилось хранение информации в базе данных. Соответственно, нужно осуществить упомянутое выше хранение информации. А для реализации хранения данных в базе данных была выбрана технология ADO.NET вместе с СУБД MySQL.

Диаграмма классов UML для сущностей классов, на основе которых была создана база данных, представлена на рисунке 4.1

На диаграмме классов видно, как связаны между собой сущности

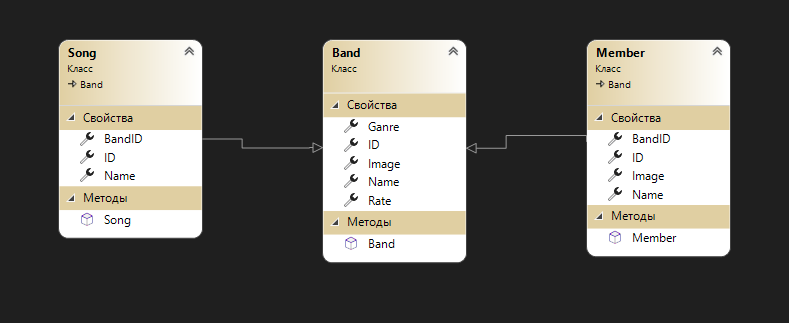


Рисунок 4.1 − Диаграмма классов

**4.2 Реализация авторизации и регистрации**

Для использования приложения пользователю необходимо создать аккаунт. При создании аккаунта используется валидация на основе регулярных выражений, которая не позволяет пользователю создать слишком короткое имя пользователя (не менее 5 символов) и запрещает использование специальных символов. Также валидация не позволяет создать слишком короткий пароль (не менее 6 символов).

При входе в систему пользователю необходимо указать имя пользователя и пароль, в случае если введен неверны пароль или несуществующее имя пользователя, пользователь получит сообщение об ошибке.

**4.3 Реализация основных функций**

Основной задачей приложения является предоставление расписания концертов и информации об тех, кто будет участвовать. Также, отображение таймера до начала концерта.

На рисунке 4.2 представлены основные компоненты, необходимые для реализации основной задачи

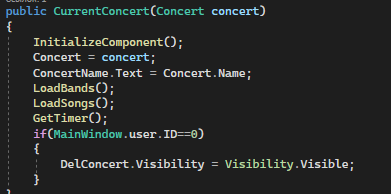


Рис 4.2 – Основные компоненты

DispatcherTimer необходим для отсчета времени до даты концерта.

LoadBands() загружает группы с базы данных.

LoadTracks() загружает песни с базы данных.

Функция GetTimer(), представленная на рисунке 4.3 отвечает за показ информации в DataGrid в зависимости от выбранной даты концерта.

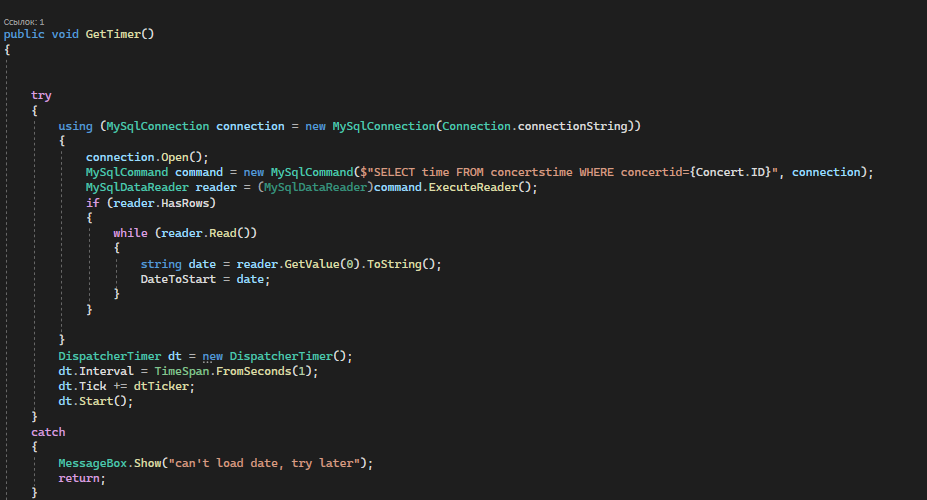


Рис. 4.3 – Функция GetTimer()

Функция Settings(), представленная на рисунке 4.4 нужна для открытия окна с настройками, где содержится выбор языка и цветовой темы.

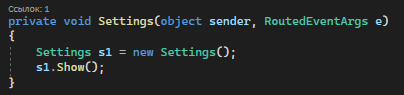


Рис. 4.4 – Функция Settings

Функция позволяет занести введенные данные в таблицу базы данных. При успешно занесенных данных будет выведено соответствующее сообщение. Если какие-либо данные введены неправильно, будет показано сообщение об ошибке.

Данная функция работает схожим образом и с песнями и с участниками. Отличие лишь в заносимых данных и таблицах.

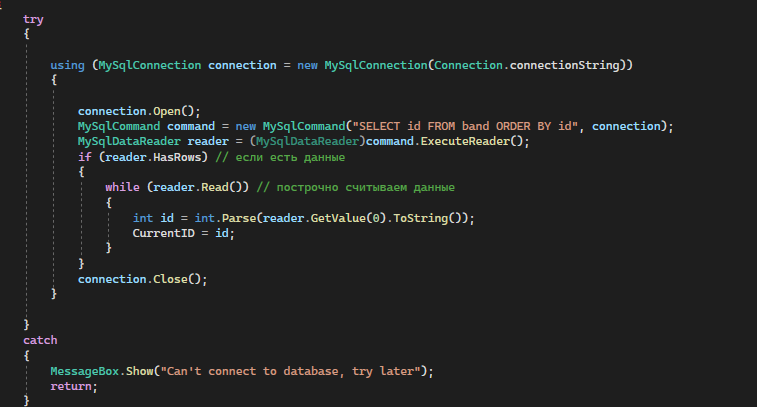


Рис. 4.5 – Функция AddBand

5 Тестирование

Для того чтобы проверить работоспособность программного продукта, было проведено тестирование всех мест, где пользователь может совершить ошибку и вызвать неправильную работу приложения.

5.1 Тестирование регистрации и входа

Так как при регистрации пользователю предоставлена возможность ввода данных, окно регистрации может стать местом появления ошибок. Для предотвращения этого была использована валидация на основе регулярных выражений.

Так, если попытаться оставить пустым Логин, то будет получено следующее сообщение об ошибке (рисунок 5.1):

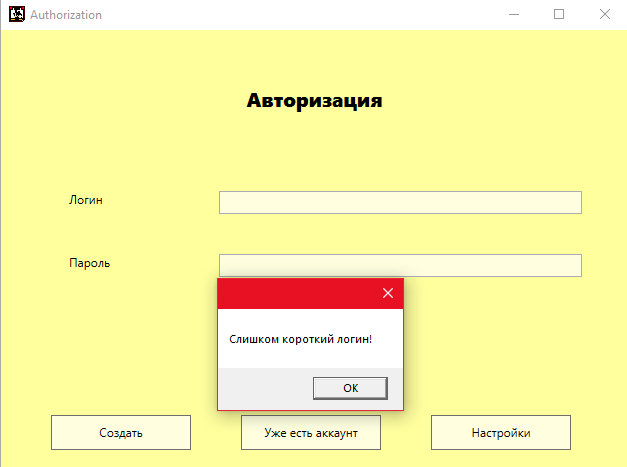


Рис. 5.1 – Сообщение об ошибке

Теперь рассмотрим ситуацию, когда логин соответствует нормам, а пароль нет (рисунок 5.2):

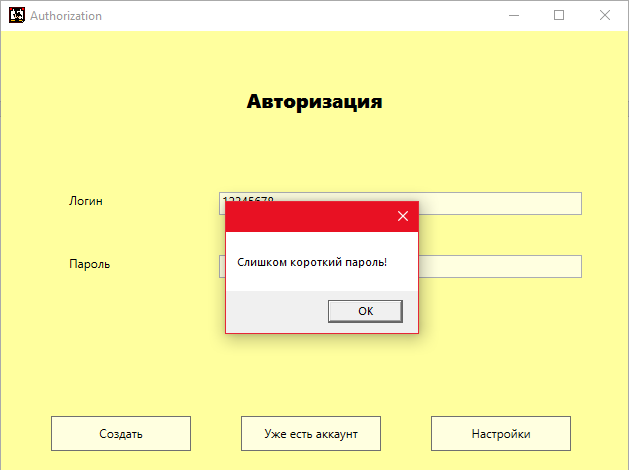


Рис. 5.2 – Сообщение об ошибке

Теперь попробуем войти без регистрации (рисунок 5.3):

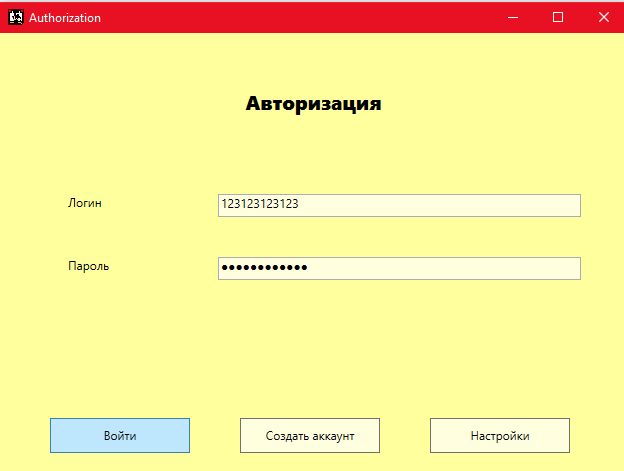


Рис. 5.3 – Сообщение об ошибке

Ожидаемо, войти не получилось, так как такого пользователя не существует.

5.2 Тестирование вывода данных

На этот раз попробуем с аккаунта администратора добавить новую группу. Нажимаем на кнопку «Группы» и видим форму для заполнения. Однако, заполнять мы ничего не будем, попробуем добавить пустоту (рисунок 5.4).

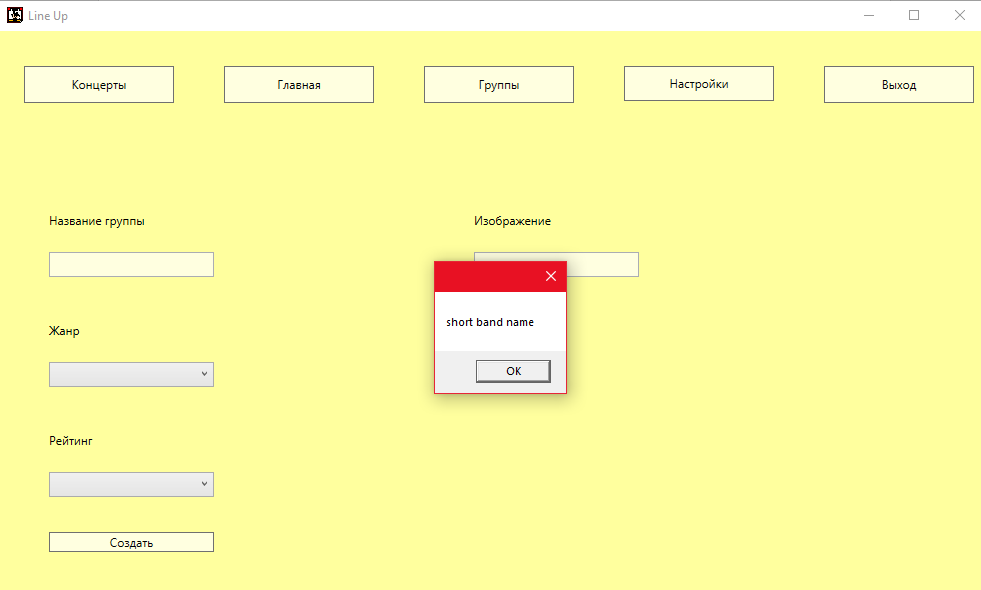


Рисунок 5.4 – Сообщение об ошибке при смене пароля

И тут предусмотрен такой вариант событий. При попытке добавить незаполненную форму, выскакивает ошибка с указанием на первое пустое поле. Если мы заполним поле «Название» и оставим остальные пустыми, нам предложат ввести «Жанр» и т.д.

Аналогично работает и с выводом данных о песнях и участниках.

6 Руководство по использованию

6.1 Общее руководство по использованию программного продукта

При запуске приложения, пользователь попадает в окно с входом и регистрацией, представленным на рисунке 6.1

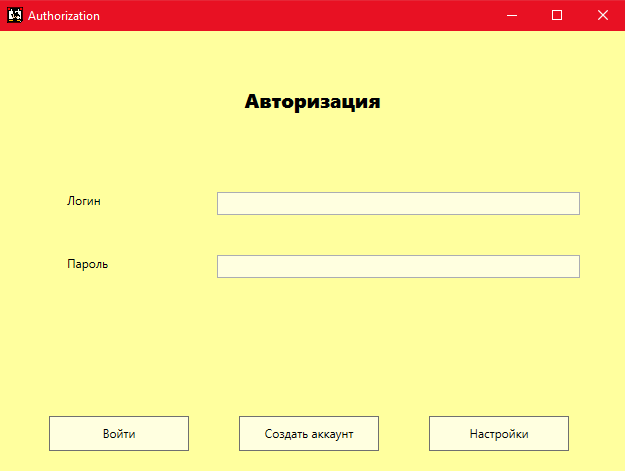


Рисунок 6.1 – Окно «Авторизация»

Если же пользователь пользуется приложением в первый раз, то ему необходимо зарегистрироваться, открыв страницу «Регистрации» (рисунок 6.2)

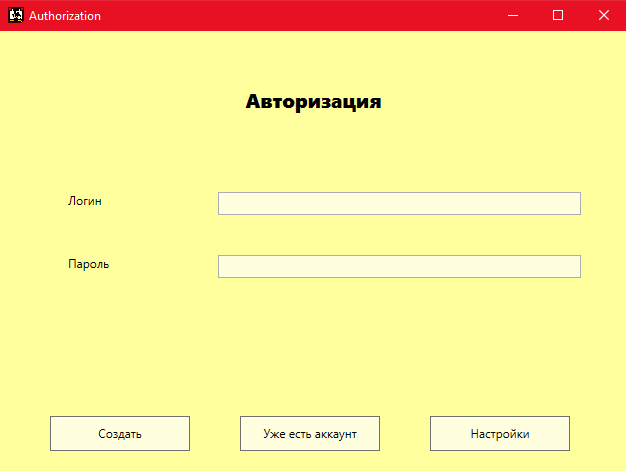


Рисунок 6.2 – Страница «Регистрация»

После окончания процедуры авторизации пользователя, он заходит и попадает на главную страницу. Тут показаны три табличных окошка, в которых располагаются все имеющиеся группы, песни, участники. Снизу есть два окна взаимодействия, одно из которых – выбор даты концерта, а второе – поиск и настройки (рисунок 6.3).

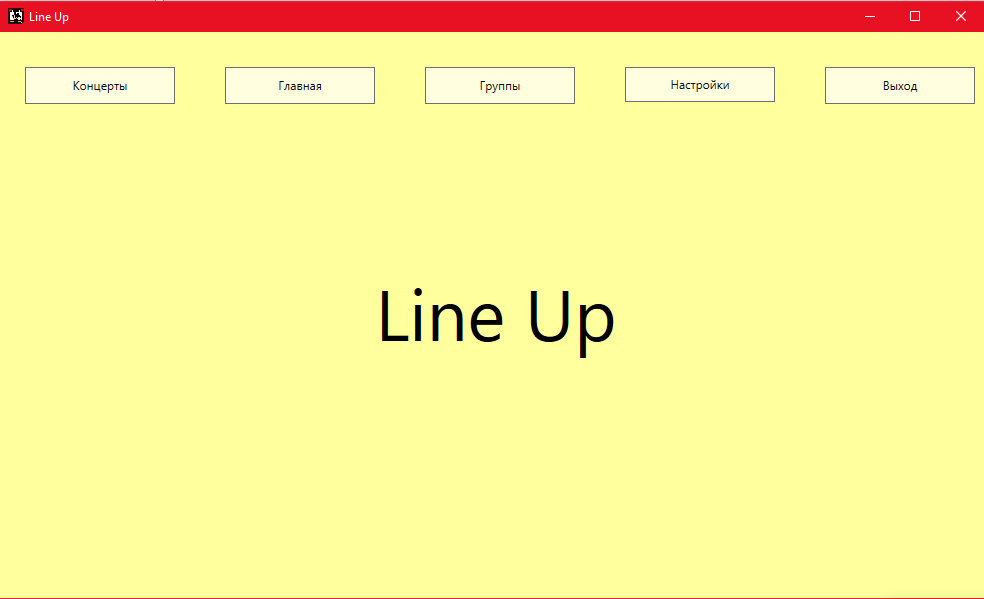


Рисунок 6.3 – Главное окно

Начнем со второго. Для начала посмотрим, что есть в настройках. Тут можно выбрать язык и цветовую тему. Показано на рисунке 6.4. В моем приложении доступно три языка на выбор: Беларуский, Русский, Английский. По умолчанию стоит русский (рисунок 6.5).

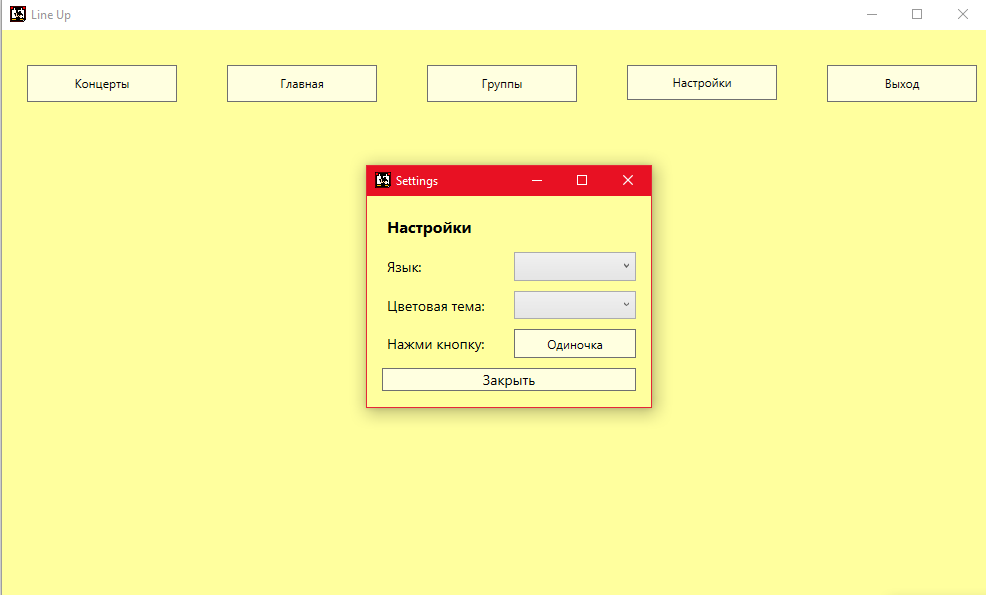


Рисунок 6.4 – Настройки

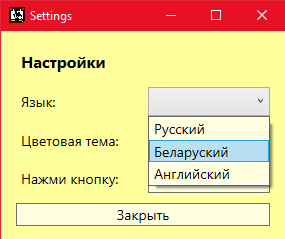


Рисунок 6.5 – Настройки (язык)

Далее цветовая тема. Тут все просто: светлая (по умолчанию) и темная (рисунок 6.6).

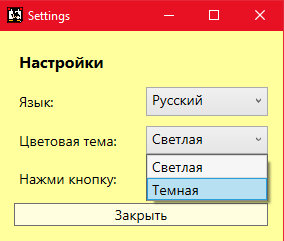
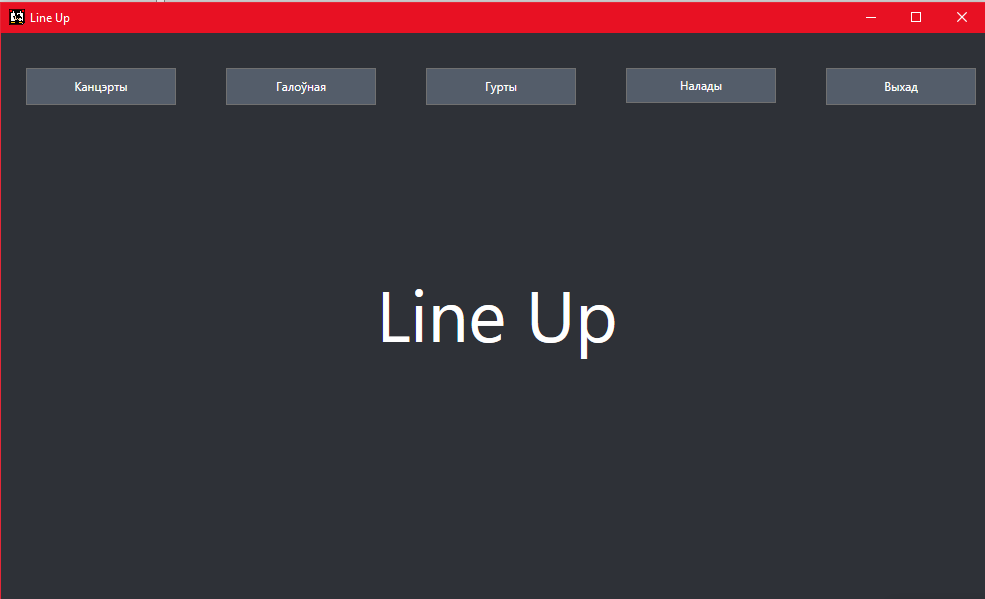


Рисунок 6.6 – Настройки (цветовая тема)

Также есть небольшое дополнение в виде кликабельной кнопки. При срабатывании выводит сообщение, ничего особенного.

Допустим, пользователь захотел продолжить пользоваться приложением на беларуском языке и темной темой (рисунок 6.7)

Рис. 6.7 – Применили настройки

Далее, попробуем найти какую-нибудь группу в строке поиска. Например, популярную инди-рок группу dysania! В итоге, в окнах осталась информация только о группе dysania! Теперь мы можем ознакомится с их группой, составом и трек-листом (рисунок 6.8)

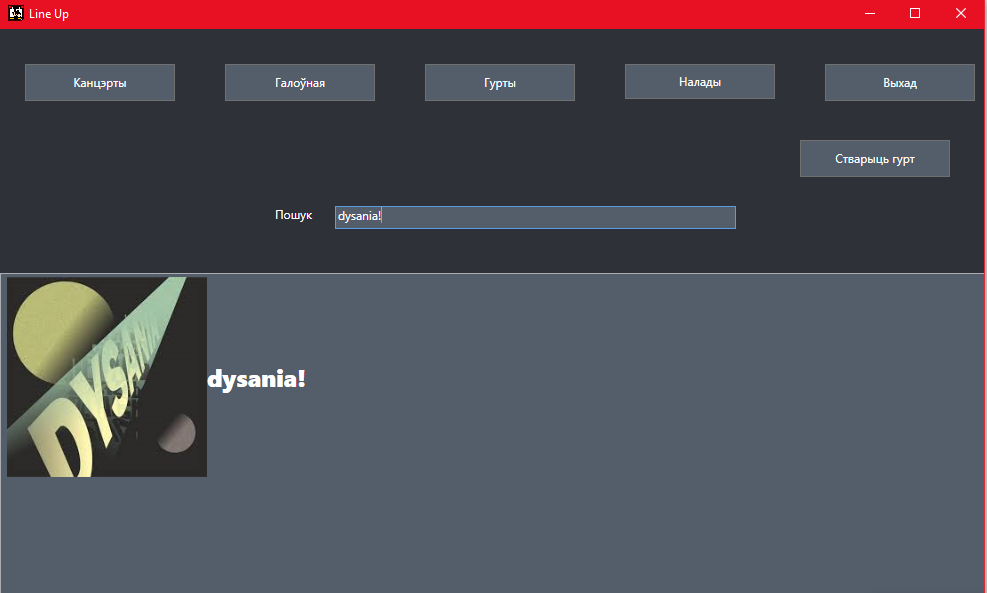


Рис. 6.8 – Результаты поиска

Вернем светлую тему и попробуем узнать лайн-ап на ближайший концерт (рисунок 6.9). Заодно узнаем, через сколько он начнется. Итог на рисунке 6.10

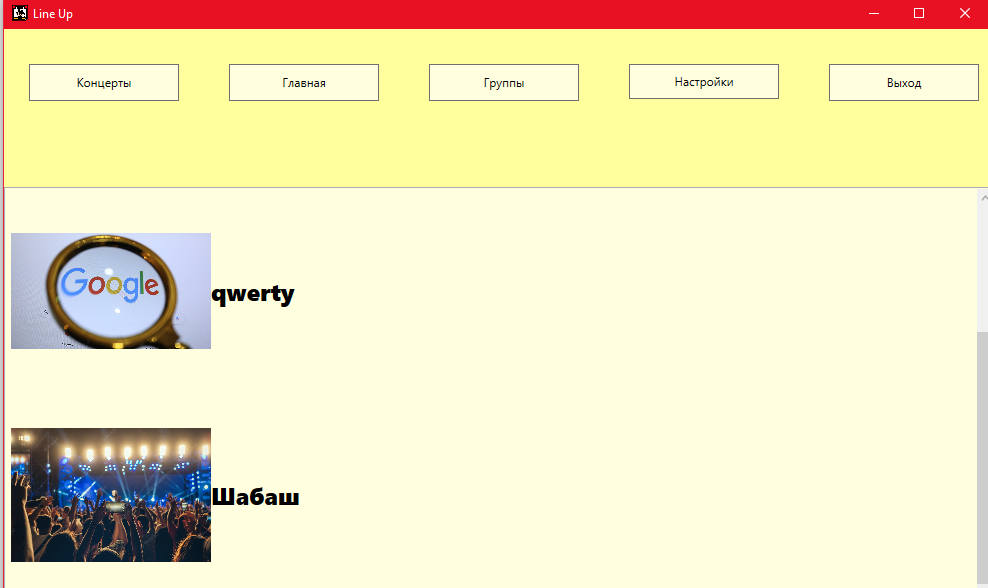


Рис. 6.9 – Выбираем дату

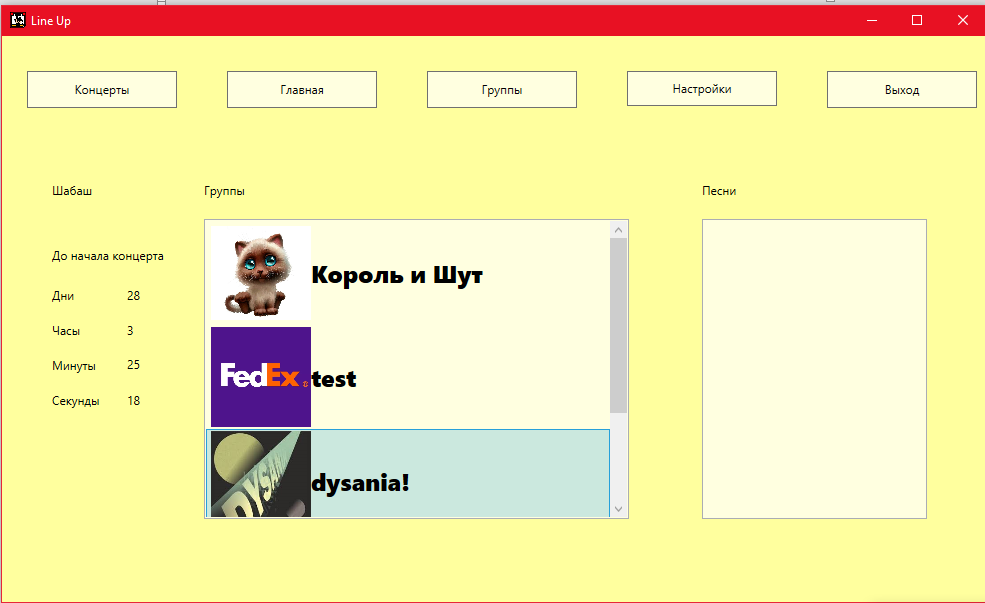


Рис. 6.10 – Выведенная информация

Таким образом, мы узнали список выступающих групп, песни этих групп, а также составы. Вся информация расположена в порядке хронологии, это же относится к песням и составу. А еще, мы узнали, что концерт начнется примерно через 5 дней и 22 часа.

6.2 Поддержка пользовательской настройки интерфейса

Как было сказано ранее, в приложении на выбор доступно три языка (Беларуский, Английский и Русский) и две цветовые темы (Светлая и Темная). Рисунки 6.11 – 6.14 демонстрируют все виды настроек.

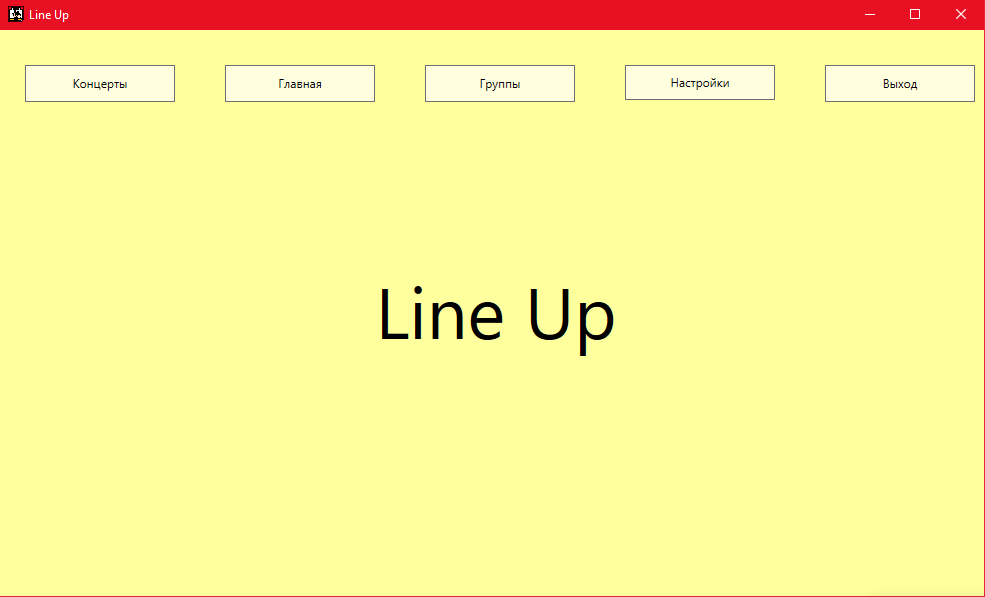


Рис. 6.11 – Главное окно со светлой темой и русским языком

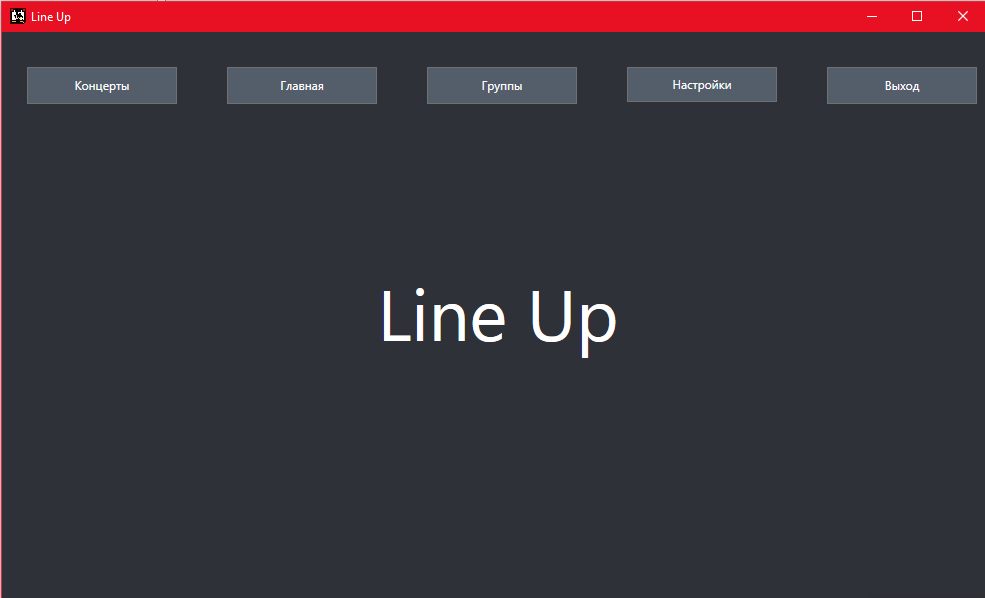


Рис. 6.12 – Главное окно с темной темой и русским языком

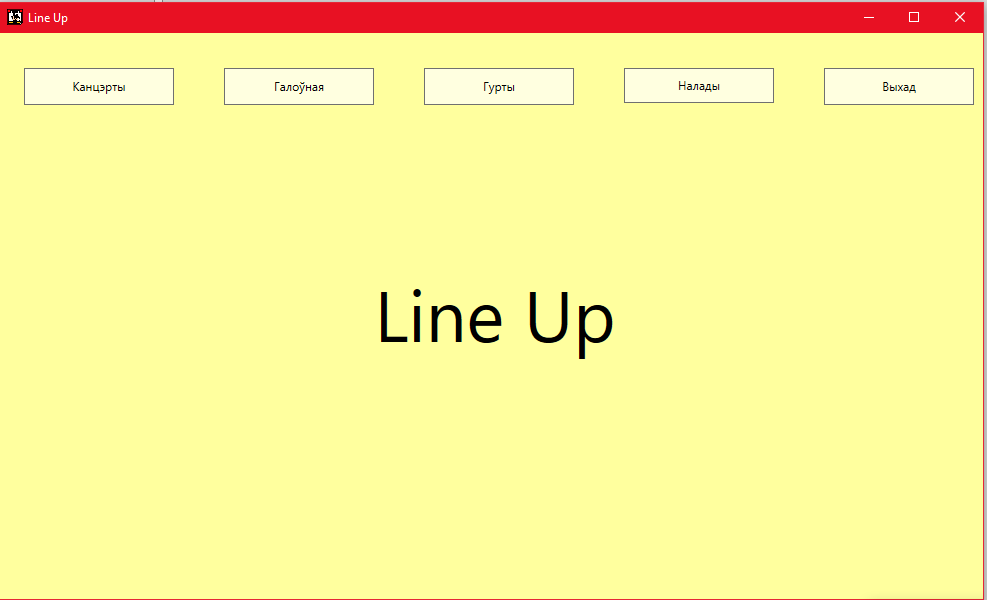


Рис. 6.13 – Главное окно на беларуском языке

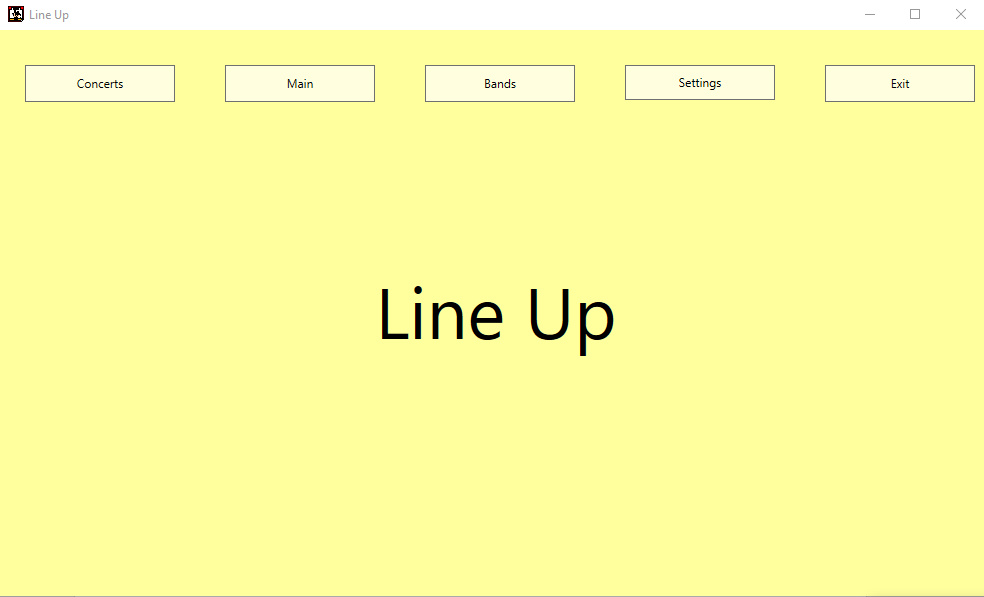


Рис. 6.14 – Главное окно на английском языке

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсового проекта было разработано приложение «Лайн-ап для концертов», учитывая все пункты возможного функционала. В конечном счете были реализованы:

Функции администратора сервиса:

* + Поддерживать работу c базой данных;
  + Регистрировать группы;
  + Устанавливать порядок выступления;
  + Добавлять информацию об участниках группы;
  + Добавлять информацию о релизах;

Функции клиента:

* + Выполнять регистрацию и авторизацию;
  + Поиск группы по определенным параметрам;

В итоге, на выходе мы имеем совершенно готовое приложение, полностью работоспособное, с проработанным дизайном и функционалом. Данное приложение могло бы быть востребовано, учитывая практически полное отсутствие десктопных приложений на данную тему.

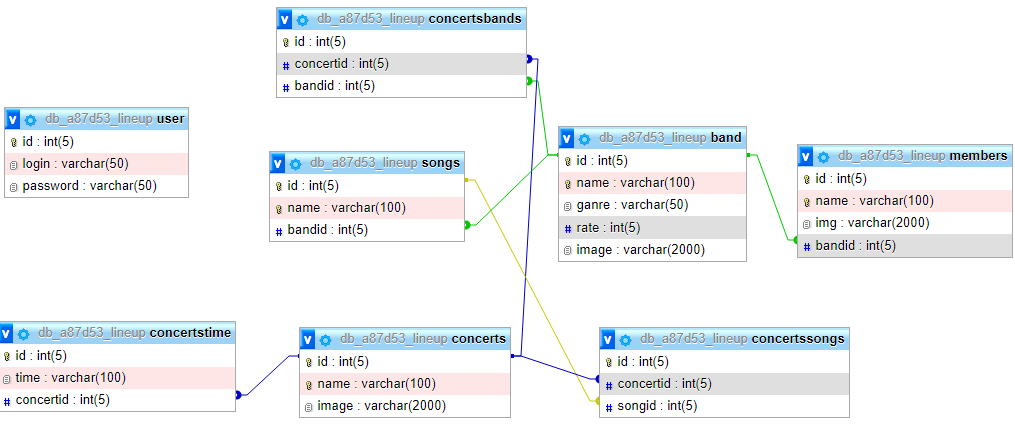
СПИСОК ЛИТЕРАТУРЫ

1. METANIT.COM Сайт о программировании [Электронный ресурс] / Режим доступа: <https://metanit.com> . Дата доступа: 20.04.2020

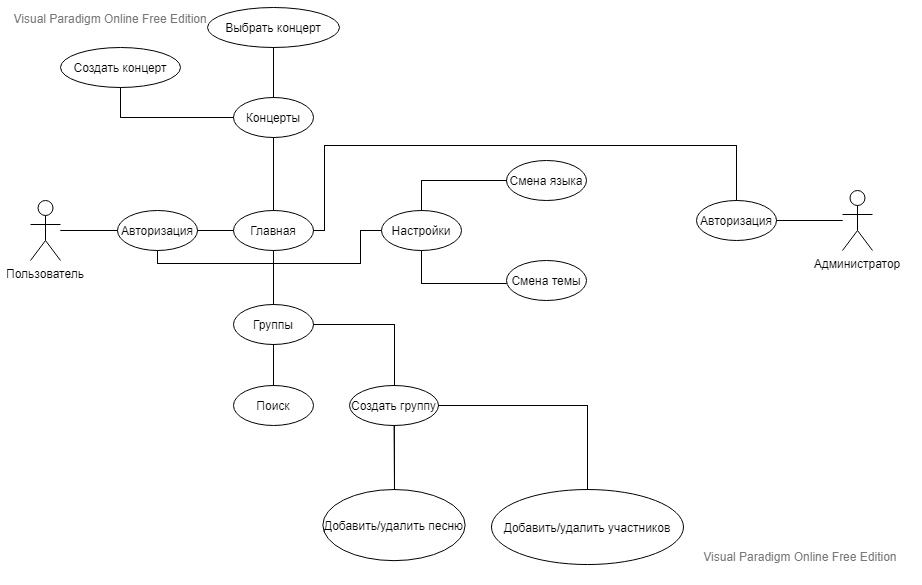
2. ProfessorWeb .NET & Web Programming [Электронный ресурс] / Режим доступа: <https://professorweb.ru> Дата доступа: 13.04.2020

3. Форум для программистов или разработчиков [Электронный ресурс] – <https://stackoverflow.com/> – Дата доступа: 5.05.2020

ПРИЛОЖЕНИЕ А

****

ПРИЛОЖЕНИЕ Б



ПРИЛОЖЕНИЕ В

