Министерство образования и молодежной политики

Свердловской области

Государственное автономное профессиональное образовательное учреждение

Свердловской области «Ирбитский политехникум»

Специальность 09.02.03 “Программирование в компьютерных системах”

Квалификация: техник-программист

ДИПЛОМНАЯ РАБОТА

РАЗРАБОТКА И ВНЕДРЕНИЕ ПРОГРАММНОГО КОМПЛЕКСА «ФАБРИКА НОВОСТЕЙ» ПО АВТОМАТИЗАЦИИ БИЗНЕС-ПРОЦЕССОВ РЕДАКЦИИ ТЕЛЕКАНАЛА «НТС-ИРБИТ» С ИСПОЛЬЗОВАНИЕМ СУБД MONGODB.

ДОПУСКАЕТСЯ К ЗАЩИТЕ зам. директора по УПР

А. В. Свинкин

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2023

Исполнитель:

Останин Иван Васильевич

Руководитель ВКР:

Вишнякова Наталья Викторовна

Ирбит, 2023

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 4](#_Toc138233317)

[ГЛАВА 1. ТЕХНОЛОГИЧЕСКИЕ АСПЕКТЫ РАЗРАБОТКИ ПРОГРАММНОГО КОМПЛЕКСА 5](#_Toc138233318)

[1.1. Словарь терминов предметной области 5](#_Toc138233319)

[1.2. Архитектура программного комплекса 5](#_Toc138233320)

[1.2.1 Общие сведения о программном комплексе 5](#_Toc138233321)

[1.2.2 Композитор сюжетов (КС) 6](#_Toc138233322)

[1.2.3 Телевизионный штаб 6](#_Toc138233323)

[1.2.4 Виртуальный интерактивный агент 7](#_Toc138233324)

[1.3. Использованные технологии, библиотеки и архитектурные шаблоны 7](#_Toc138233325)

[1.3.1 Язык программирования C# и платформа .NET 7](#_Toc138233326)

[1.3.2 Windows Presentation Foundation 8](#_Toc138233327)

[1.3.3 Архитектурный шаблон Model-View-ViewModel - MVVM 9](#_Toc138233328)

[1.3.4 СУБД MongoDB 9](#_Toc138233329)

[1.3.5 Библиотека VkNet 10](#_Toc138233330)

[1.3.6 Пакет алгоритмов FuzzySharp 10](#_Toc138233331)

[1.3.7 Extended WPF Toolkit 10](#_Toc138233332)

[1.4 Организация хранения информации в базе данных 11](#_Toc138233333)

[1.4.1 Концептуальная модель базы данных 11](#_Toc138233334)

[1.4.2 Логическая модель базы данных 12](#_Toc138233335)

[ГЛАВА 2. РАЗРАБОТКА И ВНЕДРЕНИЕ ПРОГРАММНОГО КОМПЛЕКСА 14](#_Toc138233336)

[2.1 Разработка программного комплекса 14](#_Toc138233337)

[2.1.1 Разработка физической модели базы данных 14](#_Toc138233338)

[2.1.2 Разработка проекта NfModels 16](#_Toc138233339)

[2.1.3 Разработка композитора сюжетов (КС) 17](#_Toc138233340)

[2.1.4 Разработка телевизионного штаба 21](#_Toc138233341)

[2.1.5 Разработка виртуального интерактивного агента 23](#_Toc138233342)

[2.2. Внедрение программного комплекса 26](#_Toc138233343)

[2.2.1 План внедрения программного комплекса 26](#_Toc138233344)

[2.2.2 Отчёт по внедрению программного комплекса в редакции «Ирбитский вестник» 27](#_Toc138233345)

[ГЛАВА 3. РУКОВОДСТВА ОПЕРАТОРА И ПРОГРАММИСТА 29](#_Toc138233346)

[3.1 Руководство оператора 29](#_Toc138233347)

[3.2 Руководство программиста 30](#_Toc138233348)

[ГЛАВА 4. ЭКОНОМИЧЕСКАЯ ЧАСТЬ 33](#_Toc138233349)

[4.1 Затраты на заработную плату и социальные выплаты 33](#_Toc138233350)

[4.2 Материальные затраты 34](#_Toc138233351)

[4.3 Затраты на электроэнергию 34](#_Toc138233352)

[4.4 Амортизационные отчисления 35](#_Toc138233353)

[4.5 Расчет производственной себестоимости 36](#_Toc138233354)

[ГЛАВА 5. Охрана труда 37](#_Toc138233355)

[5.1 Общие требования охраны труда 37](#_Toc138233356)

[5.2. Требования охраны труда перед началом работы 38](#_Toc138233357)

[5.3. Требования охраны труда во время работы 39](#_Toc138233358)

[5.4. Требования охраны труда в аварийных ситуациях 39](#_Toc138233359)

[5.5. Требования охраны труда по окончании работы 39](#_Toc138233360)

[ЗАКЛЮЧЕНИЕ 41](#_Toc138233361)

[Список использованной литературы 42](#_Toc138233362)

[ПРИЛОЖЕНИЕ 44](#_Toc138233363)

[*Приложение 1 Программный код DBProvider.cs (Статический провайдер методов работы с БД)* 44](#_Toc138233364)

[*Приложение 2 Программный код FileDialogs.cs (Набор файловых диалогов)* 49](#_Toc138233365)

[*Приложение 3 Программный код MaterialViewModel.cs (Модель представления материала)* 56](#_Toc138233366)

# ВВЕДЕНИЕ

Современные телеканалы сталкиваются с множеством бизнес-задач, которые необходимо эффективно решать для успешной работы и предоставления качественных услуг. Управление процессами, контроль персонала, анализ данных - все эти аспекты играют важную роль в оперативности и эффективности работы телеканала. Для решения этих задач разработка соответствующего программного комплекса становится неотъемлемой необходимостью. Основными целями такого комплекса являются автоматизация бизнес-процессов, повышение производительности и эффективности, упрощение управления и улучшение предоставления услуг.

Целью дипломной работы является проектирование, разработка и тестирование программного комплекса "Фабрика новостей" с использованием технологий WPF и СУБД MongoDB.

Данный комплекс направлен на решение бизнес-задач редакции телеканала, предоставляя средства для систематизации, каталогизации и хранения оперативных материалов, организации архивного хранения и аналитики данных.

Для достижения поставленной цели необходимо выполнить следующие задачи:

* Проанализировать бизнес-процессы, сформировавшиеся в редакции телеканала, для определения основных потребностей и требований к программному комплексу;
* Спроектировать архитектуру программного комплекса "Фабрика новостей", учитывая особенности работы и потребности редакции телеканала;
* Разрабатывать базу данных в СУБД MongoDB, обеспечивая надежное хранение информации о процессах редакции;
* Создавать пользовательский интерфейс программного комплекса с использованием технологии WPF и языка программирования C#, чтобы обеспечить удобство и интуитивно понятный доступ к функциональности комплекса;
* Провести тестирование и отладку программного комплекса для обеспечения его стабильной работы, и соответствия требованиям редакции телеканала;
* Оформить необходимую документацию к дипломной работе, включая техническое описание, инструкции и руководства пользователя.

Перейдём к рассмотрению технологических аспектов разработки программного комплекса.

# ГЛАВА 1. ТЕХНОЛОГИЧЕСКИЕ АСПЕКТЫ РАЗРАБОТКИ ПРОГРАММНОГО КОМПЛЕКСА

## 1.1. Словарь терминов предметной области

Люфт *(нем. Luft: 1) воздух; 2) зазор, просвет)* — включение в сюжет живой реальной сценки, «картинки» с соответствующим интершумом, но без закадрового текста. Нередко люфт дается до стендапа для того, чтобы зритель сразу погрузился в атмосферу события.

Синхрон – синоним интервью.

Интершум – синоним фоновый шум.

Закадровый голос – голос корреспондента, идущий за кадром во время новостного сюжета.

## 1.2. Архитектура программного комплекса

### 1.2.1 Общие сведения о программном комплексе

Программный комплекс "Фабрика новостей" состоит из нескольких основных компонентов:

* *База данных*: Хранит информацию о проектах, материалах, съемках, метках и работниках, используя СУБД MongoDB. База данных обеспечивает доступ, сохранение и управление данными, необходимыми для работы программного комплекса.
* *Композитор Сюжетов (КС):* Редактор сюжетов, предоставляющий блочный редактор для создания и редактирования сюжетов.
* *Телевизионный Штаб (ТШ):* Менеджер проектов, обеспечивающий контроль над телевизионными проектами, съемками и материалами.
* *Виртуальный Интерактивный Агент (ВАИ)*: Бот, предоставляющий доступ к запланированным съемкам, ходу работы над проектами и отчетности через социальную сеть ВКонтакте.

Архитектура программного комплекса "Фабрика новостей" представляет собой распределенную систему, где каждый компонент выполняет определенные функции и взаимодействует с другими компонентами. Модульность комплекса позволяет легко масштабировать и модифицировать систему, а также облегчает поддержку и разработку каждого модуля независимо от остальных. Взаимодействие между компонентами осуществляется посредством стандартного для всех интерфейса, что стандартизирует и упрощает связи между ними. Централизованное хранилище данных обеспечивает целостность и доступность информации.

### 1.2.2 Композитор сюжетов (КС)

Композитор Сюжетов (КС) играет ключевую роль в комплексе "Фабрика новостей" и является неотъемлемой частью процесса создания и редактирования телевизионных сюжетов. Он предоставляет мощный инструментарий для структурирования, организации и визуализации информации, связанной с сюжетами.

Работа композитора сюжетов включает в себя различные аспекты, начиная от создания новых сюжетов и заканчивая редактированием уже существующих. Композитор сюжетов предоставляет блочный редактор, который позволяет создавать сюжеты с использованием разных элементов и компонентов. Он предоставляет возможность структурировать информацию в виде блоков, которые можно легко перемещать, редактировать и комбинировать.

Композитор сюжетов облегчает процесс создания сюжетов путем предоставления гибкого и интуитивно понятного интерфейса, который позволяет редакторам легко манипулировать контентом и расставлять акценты в сюжетах. Он также обеспечивает визуализацию сюжетов, позволяя редакторам видеть структуру и последовательность событий в сюжете.

Роль композитора сюжетов в комплексе заключается не только в создании и редактировании сюжетов, но и в обеспечении эффективной работы всей системы "Фабрика новостей". Он является инструментом, который помогает редакторам телевизионных сюжетов в их творческом процессе, упрощает организацию информации и повышает эффективность работы всей команды.

### 1.2.3 Телевизионный штаб

Телевизионный Штаб (ТШ) в программном комплексе "Фабрика новостей" играет важную роль в организации и управлении телевизионными проектами. Он выполняет функции центра управления и координации различных процессов, связанных с созданием сюжетов и управлением съемками.

ТШ является основным инструментом для планирования и контроля хода работы над телевизионными проектами. Он обеспечивает возможность управлять проектами, назначать задачи и контролировать сроки выполнения. Таким образом, ТШ способствует эффективной организации работы и соблюдению установленных сроков.

Кроме того, ТШ обеспечивает централизованный доступ к материалам, связанным с телевизионными проектами. Он предоставляет возможность управлять базой данных, где хранятся съемки, материалы и другая информация, необходимая для создания сюжетов. ТШ позволяет удобно организовать доступ к этим материалам для работников, связанных с проектами.

Таким образом, роль Телевизионного Штаба в комплексе заключается в организации и управлении телевизионными проектами, планировании и контроле хода работы, а также в обеспечении удобного доступа к материалам базы данных проекта.

### 1.2.4 Виртуальный интерактивный агент

Виртуальный Интерактивный Агент (ВИА) является одним из компонентов программного комплекса "Фабрика новостей". Он представляет собой бота, интегрированного с социальной сетью ВКонтакте, который обеспечивает доступ к запланированным съемкам, проектам, а также информации о материалах.

Роль ВИА в комплексе заключается в предоставлении удобного и интерактивного интерфейса для сотрудников телеканала. Благодаря интеграции с ВКонтакте, ВИА позволяет работникам получать информацию о запланированных съемках, следить за ходом работы над проектами и получать оперативные уведомления о новых материалах и съемках.

Использование платформы ВКонтакте в качестве основной среды общения с ВИА позволяет сотрудникам телеканала взаимодействовать с программным комплексом через уже существующую популярную платформу социальной сети. Это означает, что не требуется разработка и поддержка дополнительного веб-сайта или мобильного приложения, что в свою очередь экономит время и ресурсы команды разработчиков.

ВИА упрощает и ускоряет процессы взаимодействия с программным комплексом. Работники телеканала могут получать доступ к необходимой информации и выполнять определенные задачи, используя функциональность ВИА через интерфейс социальной сети ВКонтакте, находясь на выездных мероприятиях или дома. Это позволяет им работать удаленно и обмениваться данными с фабрикой новостей даже вне рабочего места.

## 1.3. Использованные технологии, библиотеки и архитектурные шаблоны

### 1.3.1 Язык программирования C# и платформа .NET

Выбор языка программирования C# и платформы .NET для разработки программного комплекса "Фабрика новостей" был обусловлен несколькими факторами и потребностями проекта. Эти технологии предоставляют широкий набор инструментов и функциональности, позволяющих создать мощное и масштабируемое приложение для управления телевизионными сюжетами и проектами.

C# является объектно-ориентированным языком программирования, который обладает мощными средствами для разработки сложных и структурированных систем. Он предоставляет разработчикам удобный и понятный синтаксис, обширную стандартную библиотеку классов и инструменты для управления памятью и выполнения операций. Это позволяет эффективно создавать и поддерживать код, улучшая его читаемость, надежность и производительность.

Одним из основных преимуществ использования C# и .NET является существование огромной базы пользовательских пакетов, благодаря которым была обеспечена совместимость и интеграция с сервисами ВКонтакте и базой данных MongoDB.

Платформа .NET предлагает обширный набор инструментов и функций для разработки приложений любой сложности. Она обеспечивает удобное развертывание, масштабируемость и безопасность приложений.

Выбор C# и .NET для разработки "Фабрики новостей" обусловлен их возможностью создать мощное, гибкое и эффективное программное решение, которое способно управлять сложными процессами создания и управления телевизионными сюжетами. Эти технологии предоставляют все необходимые инструменты и функциональность для разработки программного обеспечения любого уровня.

### 1.3.2 Windows Presentation Foundation

Для разработки программного комплекса "Фабрика новостей" был выбран Windows Presentation Foundation (WPF) как технология для создания пользовательского интерфейса. Одной из основных причин выбора WPF была его способность обеспечить богатый и гибкий пользовательский интерфейс с возможностью создания сложных и красивых визуальных элементов.

WPF предлагает мощный инструментарий для разработки интерактивных приложений, основанных на привлекательных и интуитивно понятных пользовательских интерфейсах. Он позволяет разработчикам создавать интерфейсы с привлекательными графическими эффектами, анимацией, стилями и адаптивным дизайном. Такой подход помогает создавать современные и профессионально выглядящие приложения, что является важным аспектом в разработке программного комплекса для телеканала.

Другим важным фактором при выборе WPF была его интеграция с языком программирования C#. WPF предоставляет богатый набор инструментов и возможностей для разработки пользовательского интерфейса с использованием C#. Благодаря этому, разработчики могут использовать мощные функциональные возможности языка программирования и легко интегрировать их в интерфейс приложения.

WPF также обладает хорошей поддержкой стилизации и шаблонизации элементов пользовательского интерфейса, что позволяет создавать переиспользуемые компоненты и упрощает процесс разработки и поддержки интерфейса. Это особенно полезно при создании сложных редакторов и управляющих элементов, которые используются в программном комплексе "Фабрика новостей".

Таким образом, выбор WPF для разработки программного комплекса "Фабрика новостей" был обусловлен его способностью предоставить богатый и гибкий пользовательский интерфейс, интеграцию с языком программирования C#, поддержку стилизации и шаблонизации элементов. Эти возможности позволяют создать профессионально выглядящее и удобное в использовании приложение.

### 1.3.3 Архитектурный шаблон Model-View-ViewModel - MVVM

При разработке программного комплекса "Фабрика новостей" был использован архитектурный шаблон MVVM (Model-View-ViewModel) для обеспечения модульности, удобства сопровождения и разделения ответственности между компонентами системы.

Основной причиной выбора шаблона MVVM является его способность эффективно разделять логику приложения и пользовательский интерфейс. Модель представляет собой состояние данных и бизнес-логику, которая управляет этими данными. View (представление) отвечает за отображение данных пользователю и обработку пользовательского ввода. ViewModel служит связующим звеном между моделью и представлением, обеспечивая передачу данных и команд между ними.

Выбор MVVM позволяет создать гибкую и масштабируемую архитектуру, где каждый компонент выполняет свою специфическую функцию. Модель содержит основную логику приложения и не зависит от конкретного представления, что обеспечивает ее повторное использование и тестируемость. Представление отвечает только за отображение данных и реагирует на изменения в ViewModel, что облегчает разработку пользовательского интерфейса. ViewModel обеспечивает связь между моделью и представлением, предоставляет данные для отображения и обрабатывает пользовательский ввод.

Выбор шаблона MVVM для разработки программного комплекса "Фабрика новостей" был обусловлен его преимуществами в разделении ответственности, обеспечении модульности и удобства сопровождения, что способствует эффективной разработке и поддержке системы на протяжении всего ее жизненного цикла.

### 1.3.4 СУБД MongoDB

При выборе СУБД для разработки программного комплекса "Фабрика новостей" было принято решение в пользу MongoDB в связи с рядом факторов.

Во-первых, MongoDB обладает высокой масштабируемостью и производительностью, что является важным аспектом при работе с большим объемом данных и высокими нагрузками. Фабрика новостей предполагает обработку и хранение большого количества информации о проектах и съемках, а MongoDB позволяет эффективно масштабировать систему и обеспечивать высокую производительность при выполнении запросов к базе данных.

Во-вторых, MongoDB предлагает гибкую схему данных, что означает, что структура данных может меняться со временем без необходимости проведения миграций схемы, что значительно упрощает разработку и сопровождение системы. Это особенно актуально для проекта "Фабрика новостей", где возможны изменения и доработки функционала со временем.

И наконец, MongoDB имеет обширную документацию и активное сообщество пользователей, что облегчает внедрение и поддержку СУБД. Разработчики программного комплекса могут полагаться на ресурсы и опыт сообщества для решения вопросов и получения поддержки.

В целом, выбор MongoDB для разработки "Фабрики новостей" обусловлен его способностью эффективно работать с гибкими структурами данных, обеспечивать высокую масштабируемость и производительность, а также гибкую схему данных и поддержку открытого сообщества.

### 1.3.5 Библиотека VkNet

При разработке программного комплекса "Фабрика новостей" было принято решение выбрать библиотеку VkNet для интеграции с социальной сетью ВКонтакте. Это решение было обусловлено несколькими факторами и требованиями проекта.

Прежде всего, VkNet предоставляет удобный и простой интерфейс для взаимодействия с API ВКонтакте. Библиотека обеспечивает высокий уровень абстракции и инкапсуляции, что позволяет разработчикам легко и эффективно осуществлять различные операции, такие как отправка сообщений, получение информации о пользователях, работа с группами и многое другое.

Кроме того, VkNet активно поддерживается сообществом разработчиков и обновляется с учетом изменений в API ВКонтакте. Это позволяет быть уверенным в том, что библиотека будет соответствовать текущим требованиям и функциональности социальной сети.

Еще одним преимуществом выбора VkNet является его распространенность и популярность среди разработчиков. Существует множество документации, учебных материалов и сообществ, где можно получить поддержку и помощь в случае возникновения проблем или вопросов.

В результате, выбор VkNet для разработки программного комплекса "Фабрика новостей" был обусловлен его удобством, поддержкой, совместимостью и широкими возможностями для работы с API ВКонтакте. Это позволило эффективно реализовать функциональность Виртуального Интерактивного Агента и обеспечить комфортное взаимодействие с социальной сетью ВКонтакте в рамка программного комплекса.

### 1.3.6 Пакет алгоритмов FuzzySharp

Пакет FuzzySharp был использован в процессе разработки с целью реализации нечеткого сравнения и поиска текстовых данных. Он предоставил функциональность и алгоритмы, которые позволили сравнивать и оценивать схожесть между строками текста на основе различных метрик и показателей. Использование FuzzySharp помогло решить задачи, связанные с распознаванием и классификацией текстовых данных, а также определением степени схожести между ними. Это обеспечило более гибкую и точную обработку текстовых элементов в проекте, улучшая точность поиска и анализа данных.

### 1.3.7 Extended WPF Toolkit

В разработке программного комплекса "Фабрика новостей" использовано Extended WPF Toolkit для расширения возможностей пользовательского интерфейса. Эта библиотека предоставляет дополнительные элементы управления, которые не входят в стандартный набор WPF, позволяя создавать более интерактивные и функциональные пользовательские интерфейсы.

Один из примеров использования элементов Extended WPF Toolkit в проекте был связан с созданием формы редактирования проектов телевизорного штаба. Для удобства пользователей и более эффективного взаимодействия с фабрикой новостей, было необходимо включить дополнительный функциональный элемент – DateTimePicker, который позволяет выбрать на календаре конкретную дату и время.

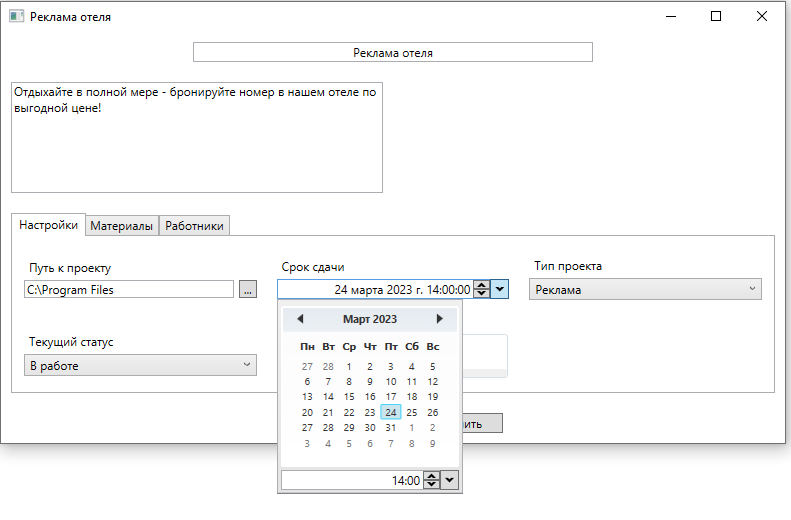


Рис. 1. Пример использования элемента Extended WPF Toolkit.

Благодаря использованию Extended WPF Toolkit, удалось значительно расширить возможности пользовательского интерфейса и создать более удобное и функциональное окружение для работы с фабрикой новостей.

## 1.4 Организация хранения информации в базе данных

### 1.4.1 Концептуальная модель базы данных

Концептуальная модель базы данных при использовании NoSQL СУБД MongoDB отличается от модели реляционных баз данных, так как MongoDB является документноориентированной СУБД, использующий использует формат документов BSON (Binary JSON), который позволяет хранить документы в виде иерархических структур. При проектировании базы данных необходимо учитывать эту структуру и определять иерархию данных.

MongoDB не требует определения схемы данных заранее, что позволяет гибче работать с данными и вносить изменения в структуру базы данных без необходимости пересоздания схемы. Также, это может приводить к неожиданным ошибкам в работе программного комплекса из-за отсутствия жесткой структуры данных.

Таблица 1. Концептуальная модель базы данных.

|  |  |
| --- | --- |
| **Сущности** | **Атрибуты сущностей** |
| Съёмка | Название, описание  Адрес  Время съёмки  Информация о работниках и выполняемых ими обязанностях на съёмке  Присвоенные метки |
| Материал | Название, описание  Путь к материалу в файловой системе  Тип материала  Присвоенные метки  Информация о работниках ответственных за материал |
| Проект | Название, описание  Использованные материалы  Путь к проекту в файловой системе  Срок сдачи проекта  Тип проекта  Информация о работниках и выполняемых ими обязанностях в проекте  Присвоенные метки |
| Работник | Имя, Фамилия  Должность |
| Метка (Тег) | Название, описание  Тег-родитель  Цвет |

### 1.4.2 Логическая модель базы данных

В MongoDB для хранения данных используется документоориентированная модель, что позволяет хранить связанные данные внутри одного документа или в нескольких документах с использованием ссылок. В случае использования нескольких коллекций, для программной реализации соединения таблиц в MongoDB можно использовать ссылки на документы.

В данном случае, связанные данные хранятся в разных коллекциях, а в документе коллекции, который ссылается на другой документ, хранится только идентификатор (ObjectId) этого документа.

Таблица 2. Связи между таблицами базы данных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Связь | Главный документ | Дочерний документ | Поле для связи главного документа | Поле для связи дочернего документа |
| Метки съёмки | Filmings | Tags | Элемент поля «Tags» | \_id |
| Метки материала | Materials | Tags | Элемент поля «Tags» | \_id |
| Метки проекта | Projects | Tags | Элемент поля «Tags» | \_id |
| Работники съёмки | Filmings | Persons | Элемент поля «MyCrew» | \_id |
| Работники материала | Materials | Persons | Элемент поля «MyCrew» | \_id |
| Работники проекта | Projects | Persons | Элемент поля «MyCrew» | \_id |
| Материалы проекта | Projects | Materials | Элемент поля «Materials» | \_id |
| Родитель тега | Tags | Tags | «ParentTagId» | \_id |

# ГЛАВА 2. РАЗРАБОТКА И ВНЕДРЕНИЕ ПРОГРАММНОГО КОМПЛЕКСА

## 2.1 Разработка программного комплекса

### 2.1.1 Разработка физической модели базы данных

В СУБД MongoDB, данные хранятся в виде BSON-документов (Binary JSON), каждый из которых содержит поля и значения, а также ссылки на другие документы.

Физическая модель базы данных программного комплекса "Фабрика новостей" по автоматизации бизнес-процессов редакции телеканала может быть описана следующим образом:

Таблица 3. Коллекции базы данных «NewsFactory»

|  |  |
| --- | --- |
| Коллекция БД | Индексы |
| Filmings | \_id\_ - уникальность поля «\_id» |
| Materials | \_id\_ - уникальность поля «\_id» |
| Persons | \_id\_ - уникальность поля «\_id» |
| Projects | \_id\_ - уникальность поля «\_id» |
| Tag | \_id\_ - уникальность поля «\_id»  Title\_ - уникальность поля «Title» |

Таблица 4. Структура документов коллекции «Filmings»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Описание |
| \_id | ObjectId | Идентификатор съёмки |
| Title | String | Название съёмки |
| Description | String | Описание съёмки |
| CreationTime | Date | Дата создания объекта |
| Tags | Array | Метки |
| Status | Int32 | Текущий статус съёмки |
| MyTime | Date | Время съёмки |
| Address | String | Адрес съёмки |

Таблица 5. Структура документов коллекции «Materials»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Описание |
| \_id | ObjectId | Идентификатор материала |
| Title | String | Название материала |
| Description | String | Описание материала |
| CreationTime | Date | Дата создания объекта |
| Tags | Array | Метки |
| MyCrew | Array | Работники и выполняемые ими обязанности для материала |
| Path | String | Путь к материалу в файловой системе |
| MyType | Int32 | Тип материала |

Таблица 6. Структура документов коллекции «Persons»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Описание |
| \_id | ObjectId | Идентификатор сотрудника |
| FirstName | String | Имя сотрудника |
| LastName | String | Фамилия сотрудника |
| CreationTime | Date | Дата создания объекта |
| Post | String | Должность |

Таблица 7. Структура документов коллекции «Projects»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Описание |
| \_id | ObjectId | Идентификатор проекта |
| Title | String | Название проекта |
| Description | String | Описание проекта |
| CreationTime | Date | Дата создания объекта |
| Tags | Array | Метки |
| MyCrew | Array | Работники и выполняемые ими обязанности в проекте |
| Path | String | Путь к проекту в файловой системе |
| MyType | Int32 | Тип проекта |
| DeadLine | Date | Срок сдачи проекта |
| Status | Int32 | Текущий статус работы над проектом |
| Materials | Array | Массив идентификаторов материалов, использованных в проекте |

Таблица 8. Структура документов коллекции «Tags»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Описание |
| \_id | ObjectId | Идентификатор метки |
| Title | String | Название метки |
| Description | String | Описание метки |
| CreationTime | Date | Дата создания объекта |
| ParentTagId | ObjectId | Метка-родитель |
| MyColor | Object | Цвет метки |

### 2.1.2 Разработка проекта NfModels

Проект NfModels (News Factory Models) является общим для всех программ в комплексе "Фабрика новостей". Он содержит код, который обеспечивает основные функциональности, необходимые для работы с базой данных, реализации шаблона MVVM (Model-View-ViewModel), заранее заготовленных диалогов работы с операционной системой и содержит некоторую общую логику, используемую другими программами комплекса.

В рамках проекта NfModels реализована связь с базой данных, позволяющая программам получать доступ к данным, хранящимся в базе данных MongoDB. Код, отвечающий за взаимодействие с базой данных, обеспечивает чтение, запись и обновление данных, а также выполнение запросов и фильтрацию данных с использованием MongoDB .NET Driver.

Для реализации шаблона MVVM в программном комплексе используется модели представления (ViewModel) различных объектов, которые отвечают за представление данных и обработку логики взаимодействия с пользователем. Проект NfModels предоставляет как базовые классы и интерфейсы для создания моделей представления, так и классы объектов фабрики, обеспечивая единообразную структуру и методы для связи данных с пользовательским интерфейсом.

Также проект NfModels содержит заранее заготовленные диалоги работы с файловой системой (FileDialogs), которые используются программами для предоставления пользователю возможности выбора места сохранения, загрузки файлов или экспорта данных.

Одна из важных особенностей проекта NfModels заключается в том, что он содержит часть общей логики, которая используется различными программами комплекса. Это позволяет избежать дублирования кода и обеспечивает единообразие функциональности между программами.

В целом, проект NfModels играет неотъемлемую роль в комплексе "Фабрика новостей", обеспечивая общую функциональность, связь с базой данных, модели представления для шаблона MVVM и другие необходимые компоненты, способствуя более эффективной разработке и поддержке программного комплекса.

Рассмотрим основу файловую структуры данного проекта.

Таблица 9. Основа файловой структуры проекта NfModels

|  |  |
| --- | --- |
| Файл / Директория | Описание |
| /Infrastructure | Дополнительные файлы инфраструктуры решения |
| /Infrastructure/Commands | Директория, которая содержит классы команд, используемые для обработки пользовательских действий, таких как нажатие кнопки или выбор элемента из списка |
| /Infrastructure/Converters | директории размещаются классы конвертеров, которые позволяют преобразовывать данные из одного формата в другой |
| /Services | Директория, где располагаются классы, предоставляющие различные службы и функциональности, которые могут быть использованы в моделях представления |
| /Services/DBProvider.cs | Класс, провайдера статических методов работы с базой данных |
| /Services/FileDialogs.cs | Класс, провайдера статических методов работы с файловой системы |
| /ViewModels | Директория, где размещаются классы моделей представления |
| /ViewModels/NewsFactoryObjects | Директория, моделей представления объектов фабрики |
| /Views | Директория, содержащая XAML-файлы, которые определяют внешний вид пользовательского интерфейса |
| /NfResources.xaml | Файл разметки, в котором содержатся ресурсы, наследуемые в другие проекты решения |

### 2.1.3 Разработка композитора сюжетов (КС)

Разработка композитора сюжетов, как части программного комплекса "Фабрика новостей", была одним из ключевых этапов проекта. Целью разработки было создание мощного инструмента для создания и редактирования сюжетов новостных программ. Композитор сюжетов должен предоставлять удобный интерфейс для визуального составления сюжетов из блоков информации, а также обеспечивать возможность редактирования и сохранения сюжетов.

Для реализации шаблона проектирования MVVM были созданы соответствующие классы модели представления (ViewModel) и представления (View). Модели представления отображены в следующей таблице.

Таблица 10. Модели представления композитора сюжетов

|  |  |
| --- | --- |
| Модель представления | Описание |
| StartupViewModel | Модель представления приветственного диалога |
| MainWindowViewModel | Модель представления главного окна |
| SettingsViewModel | Модель представления окна настроек |
| DictorViewModel | Модель представления части новостного сюжета «Диктор» |
| LuftViewModel | Модель представления части новостного сюжета «Люфт» |
| SynchroneViewModel | Модель представления части новостного сюжета «Синхрон» |
| VoiceViewModel | Модель представления части новостного сюжета «Закадровый голос» |
| ScriptViewModel | Модель представления новостного сюжета |
| MaterialViewModel | Модель представления материала сюжета |
| MaterialPlayerViewModel | Модель представления проигрывателя материалов сюжета |

В композиторе сюжетов реализована возможность создания и редактирования сюжетов с использованием блочного редактора. Он позволяет пользователям визуально компоновать и структурировать информацию, представленную в виде блоков. Каждый блок содержит определенную часть сюжета, такую как дикторская программа, закадровый текст, люфт и снихрон.

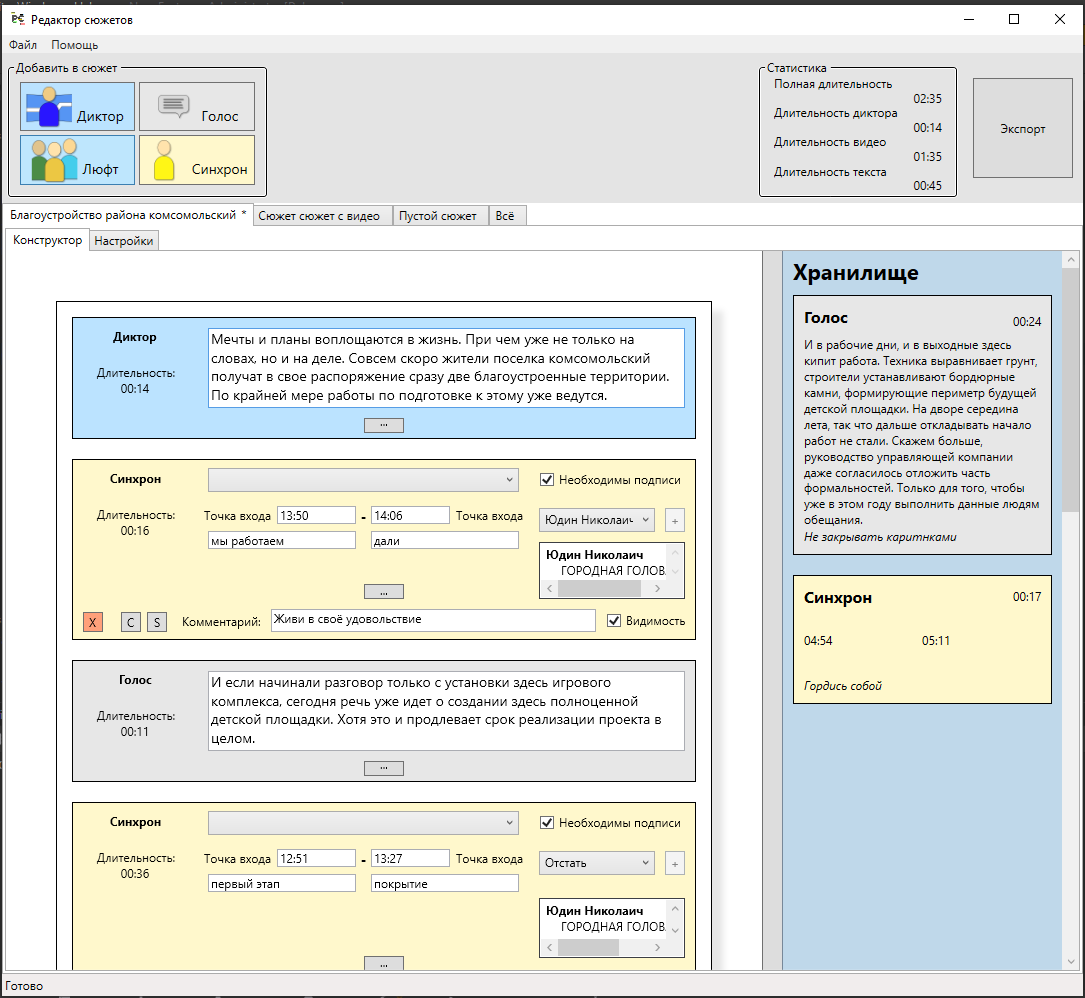


Рис. 2 Главное окно композитора сюжетов.

Пользователи могут свободно перемещать блоки, а также изменять их содержимое. Композитор сюжетов предоставляет различные инструменты и функции для работы с блоками. Так части сюжета, которым пока нет места в сюжете можно отложить в хранилище (Stash), а при необходимости вернуть обратно.

Также в рамках композитора сюжетов был реализован проигрыватель материалов, который предоставляет возможность вставлять видеофрагменты прямо в сюжеты. Этот проигрыватель значительно облегчает работу редакторов сюжетов, позволяя им непосредственно просматривать и выбирать нужные фрагменты видеоматериалов для включения в сюжеты.

Одной из важных функций проигрывателя является возможность ставить маркеры на таймлайне. Маркеры позволяют редакторам отметить интересующие моменты видеофрагментов, которые они хотят использовать в сюжете. Редакторы могут легко устанавливать маркеры на нужных временных позициях в проигрывателе, чтобы запомнить эти моменты и быстро вернуться к ним в дальнейшем.

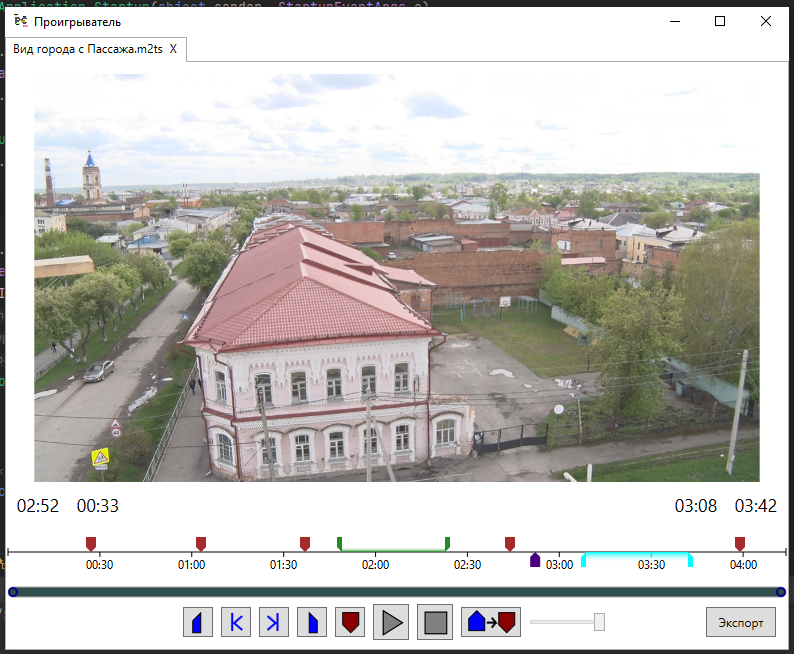


Рис. 3. Окно проигрывателя материалов.

Интересно то, что уже использованные видеофрагменты отображаются в проигрывателе как маркеры. Это позволяет редакторам легко отслеживать и визуально представлять, какие фрагменты уже были использованы в составлении сюжета. Благодаря этому, редакторы могут легко контролировать и организовывать свою работу, избегая повторного использования одних и тех же фрагментов.

Кроме того, композитор сюжетов обеспечивает возможность сохранения и загрузки созданных сюжетов в собственный файл с расширением «.nscr» (скор. от news script – новостной сюжет), а также экспорта на печать их в формате HTML. Это позволяет пользователям легко обмениваться и распространять свои сюжеты с коллегами и другими заинтересованными сторонами.

Важным аспектом разработки композитора сюжетов была также обеспечение удобного пользовательского интерфейса с интуитивно понятными элементами управления. Были разработаны различные функциональные панели, инструменты и контекстные меню, чтобы пользователи могли легко осуществлять необходимые действия и настройки при работе с композитором сюжетов.

В итоге, разработка композитора сюжетов позволила создать мощный инструмент, способствующий эффективной и креативной работе редакторов сюжетов телевизионных программ. Благодаря его функциональности и удобному интерфейсу, редакторы могут быстро и легко создавать сюжеты, а также вносить необходимые изменения в уже существующие сюжеты, улучшая качество и привлекательность телевизионных программ.

### 2.1.4 Разработка телевизионного штаба

В начале процесса разработки были проведены анализ и определение требований к функциональности ТШ. Определены основные задачи, которые должна выполнять программа, такие как контроль над телевизионными проектами, управление съемками и материалами, планирование и отслеживание выполнения задач.

Затем был разработан дизайн и архитектура ТШ, определены основные компоненты и модули, необходимые для реализации требуемой функциональности. В рамках архитектуры были определены различные классы, интерфейсы и связи между ними.

Большую часть главного окна занимает интерфейс браузера объектов фабрики, позволяющий искать проекты, съёмки и материалы по множеству настраиваемых условий. Данное условия формулируются в запрос, который необходимо ввести в поисковую строку, находящуюся в верхней части браузера. Запрос формулируется на специальном языке. На данный момент поддерживается иерархический поиск по метке, поиск по пути материала или проекта, поиск по типу объекта, поиск по статусу объекта, поиск по дате создания объекта, поиск по сотруднику и исполняемыми им обязательствам. По двойному клику на результате поиска, откроется окно редактирования данного объекта.

В верхней части главного окна расположено меню, состоящие из двух разделов: «Файл», «Помощь». В меню «Файл» находятся команды для создания новых объектов фабрики: «Проект», «Материал», «Съёмка», «Тег», «Работник». В разделе меню «Помощь» помещены команды открытия окон справочной информацией: «Синтаксис поисковой строки» и «О программе».

В левой части главного окна расположено боковое меню, расширяющееся при наведении курсора. В данном меню находятся кнопки, с помощью которых можно перейти в браузер проектов с предустановленным запросом.

Всего для редактирования доступно 5 объектов: «Проект», «Материал», «Съёмка», «Тег», «Работник». Вызвать окно редактирования объекта можно двойным кликом по нему из браузера или из контекстного меню представления данного объекта.

В верхней части окон редактирования можно изменить заголовок объекта. Описание объекта изменяется ниже. В нижней части окна находятся две кнопки: «Отмена» и «Сохранить». По нажатию кнопки отмены все изменения, внесённые локально будут сброшены и состояние объекта будет сброшено до последнего сохранённого в БД. При нажатии кнопки сохранения все локальные изменения будут сохранены в БД.

Если объект содержит ссылки на другие объекты, например, проект ссылается на материалы, использованные в нём, и на работников, которые ответственны за данный проект, то редактирование подобных ссылок вынесено на отдельную вкладку, как показано на данном рисунке:

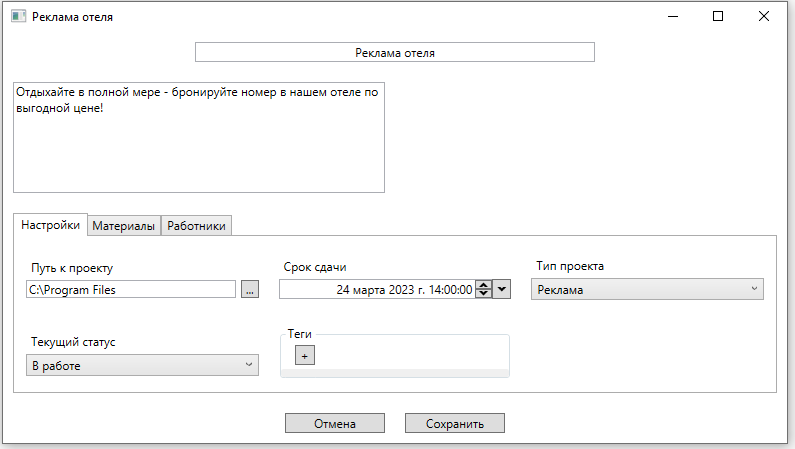


Рис. 4 Окно редактирования проекта.

Исключением являются метки (теги). Их редактирование вынесено в область основных настроек. И так как теги имеют свойства наследовать друг друга, то при попытке добавить к объекту родителя или потомка уже присвоенного тега, будет появляться сообщение об особенности данного действия, как показано на рисунках:

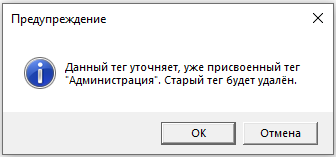


Рис. 5 Предупреждение при попытке добавить тег, являющийся потомком уже присвоенному.

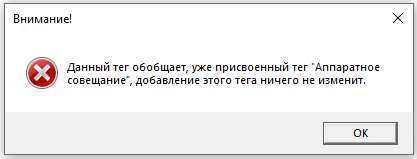


Рис. 6 Сообщение об ошибке при попытке добавить тег, являющийся родителем уже присвоенному.

В процессе разработки ТШ были реализованы основные функциональные возможности, включая создание и редактирование проектов, управление съемками, контроль за материалами, планирование задач и другие операции, необходимые для эффективной работы телевизионного штаба.

Также в процессе разработки ТШ были проведены тестирование и отладка программы для обеспечения ее стабильной работы и соответствия требованиям.

В итоге разработки Телевизионного штаба получилась программа, которая позволяет управлять телевизионными проектами, контролировать съемки и материалы, планировать задачи и обеспечивать эффективную работу штаба.

### 2.1.5 Разработка виртуального интерактивного агента

В рамках разработки программного комплекса "Фабрика новостей" был создан компонент под названием "Виртуальный Интерактивный Агент" (ВИА). Этот компонент был разработан с целью предоставления доступа к различным функциональным возможностям фабрики новостей через социальную сеть ВКонтакте. Он представляет собой бот, который обеспечивает взаимодействие пользователей с комплексом через интерфейс ВКонтакте

Процесс разработки ВИА начался с определения требований и функциональности, которые должны были быть реализованы. В ходе анализа и планирования были учтены особенности работы редакции телеканала и потребности их сотрудников.

В процессе настройки сообществ в социальной сети ВКонтакте для работы бота были выполнены несколько шагов. В начале было создано сообщество, в котором планировалось развернуть бота. Работникам телеканала были разосланы приглашения на вступление.

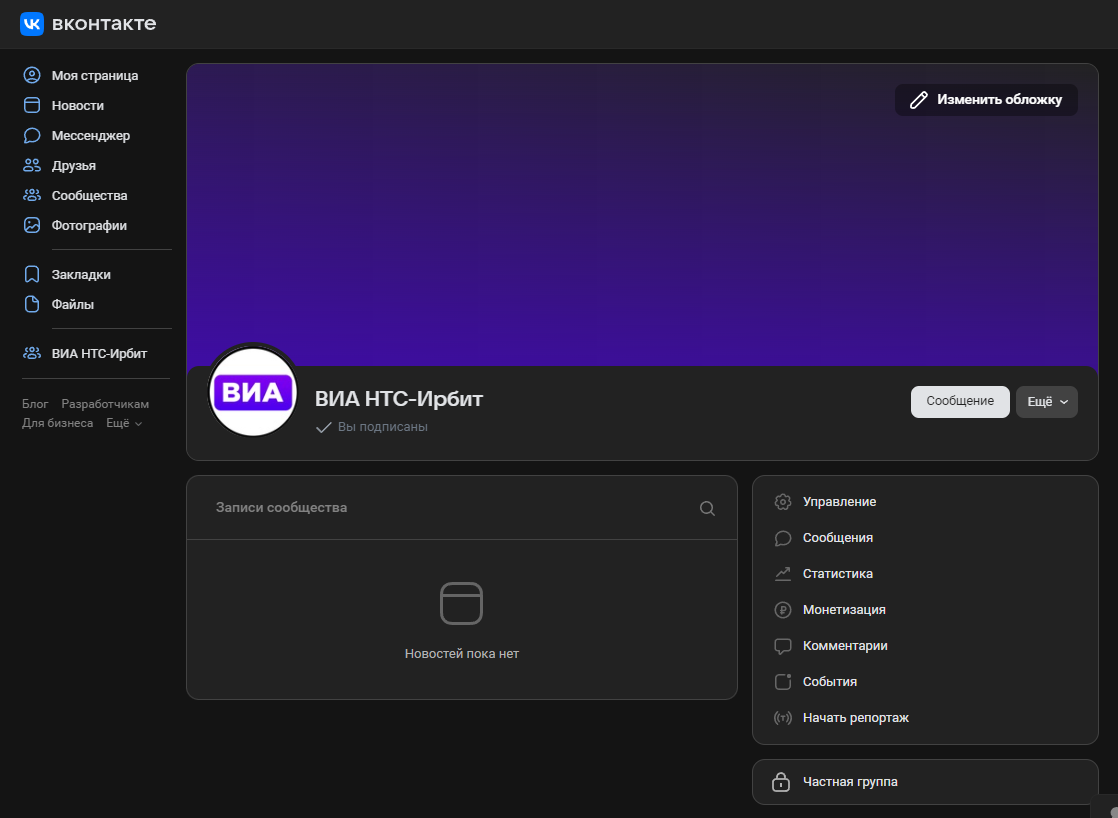


Рис. 7. Сообщество ВИА ВКонтакте

Затем были пройдены следующие этапы настройки интеграции:

* Установка необходимых разрешений: в настройках сообщества были предоставлены необходимые разрешения для работы с API ВКонтакте. Это включало разрешения на управление сообщениями, получение информации о пользователях и доступ к статистике.
* Создание ключа доступа: Был создан ключ доступа (access token), который предоставлял боту авторизацию для взаимодействия с API ВКонтакте. Этот ключ доступа использовался для аутентификации бота и выполнения различных операций, таких как отправка и получение сообщений.
* Настройка Long Poll API: Для обработки входящих сообщений и событий в режиме реального времени был настроен Long Poll API. Это позволяло боту моментально реагировать на новые сообщения и события в сообществе.

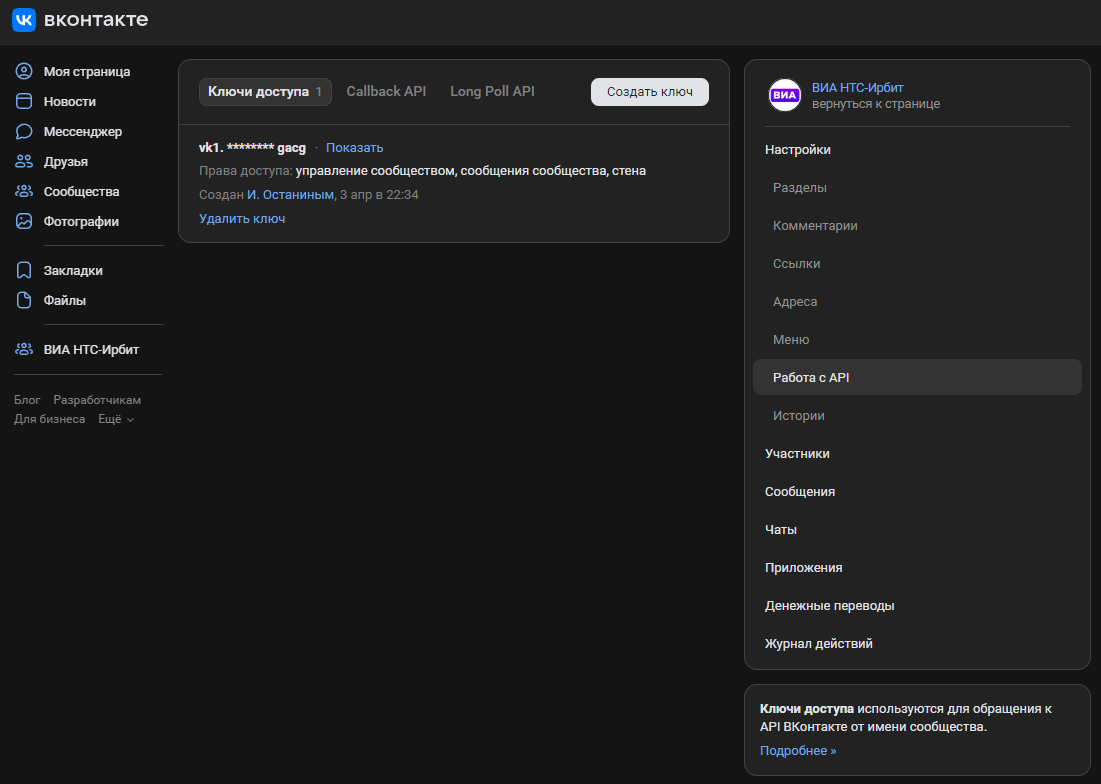


Рис. 8. Натройка API сообщества ВИА Вконтакте

Одной из основных проблем разработки ВИА было создание гибкой архитектуры приложения, которая бы обеспечивала удобный интерфейс для взаимодействия с функциональностью фабрики новостей, и в тоже время позволял в будущем расширять структуру диалога. Был проведен процесс проектирования и определения структуры бота, а также визуального представления информации и возможностей, доступных пользователям.

Затем был осуществлен программирование ВИА с использованием языка программирования C# и библиотеки VkNet для интеграции с социальной сетью ВКонтакте. Разработка включала создание классов и модулей, обеспечивающих функциональность бота, включая доступ к запланированным съемкам, ходу работы над проектами и отчетности.

В процессе разработки ВИА также проводились тестирования для обнаружения и исправления возможных ошибок и неполадок. Тестирование осуществлялось с помощью отладчика .NET и проведением различных сценариев использования, чтобы убедиться в корректной работе и соответствии функциональности требованиям.

ВИА представляет собой инструмент, который позволяет работникам телеканала получать доступ к важной информации и функциональности фабрики новостей, облегчая их трудовые процессы и повышая эффективность работы всей редакции.

## 2.2. Внедрение программного комплекса

### 2.2.1 План внедрения программного комплекса

Внедрение программного комплекса "Фабрика новостей" на предприятии включает несколько этапов, которые позволяют эффективно развернуть и интегрировать комплекс в рабочую среду. Ниже представлен общий план внедрения:

Этап 1. Анализ требований и подготовительные работы

Проведение встреч с представителями предприятия для определения требований и целей, которые должен решать программный комплекс.

Изучение имеющейся информации об инфраструктуре предприятия, существующих системах и процессах работы.

Определение функциональных и технических требований к программному комплексу.

Подготовка плана внедрения, включающего этапы, ресурсы, сроки и бюджет.

Этап 2. Установка и настройка серверной инфраструктуры

Подготовка серверной инфраструктуры, включая установку необходимых операционных систем, баз данных и других зависимостей.

Установка программного комплекса на сервер и его настройка в соответствии с требованиями предприятия.

Проверка и тестирование работоспособности серверной инфраструктуры и комплекса.

Этап 3. Установка и настройка клиентского приложения

Установка клиентского приложения на компьютеры пользователей согласно системным требованиям.

Настройка клиентского приложения для подключения к серверу и взаимодействия с программным комплексом.

Обучение пользователей работе с клиентским приложением и ознакомление с его функциональностью.

Этап 4. Тестирование и отладка

Проведение тестирования функциональности и производительности программного комплекса в контролируемой среде.

Выявление и устранение ошибок, дефектов и несоответствий требованиям.

Проведение интеграционного тестирования с другими системами и процессами предприятия.

Этап 5. Внедрение и обучение

Перенос программного комплекса в рабочую среду предприятия.

Проведение обучающих мероприятий для пользователей, включающих демонстрацию функциональности и инструктаж по использованию программного комплекса.

Обеспечение поддержки и консультаций пользователей на начальном этапе использования комплекса.

Этап 6. Мониторинг и поддержка

Организация системы мониторинга работы программного комплекса и его инфраструктуры.

Предоставление технической поддержки пользователей, включая решение возникающих проблем и вопросов.

Постоянный анализ работы комплекса, выявление улучшений и внесение необходимых изменений.

Только после прохождения всех 6 этапов процесс внедрения комплекса можно считать оконченным. Весь процесс должен проводиться в тесном сотрудничестве с представителями предприятия, чтобы учесть его особенности, потребности и гарантировать успешное внедрение и использование комплекса.

### 2.2.2 Отчёт по внедрению программного комплекса в редакции «Ирбитский вестник»

Внедрение программного комплекса "Фабрика новостей" в редакции «Ирбитский вестник» завершилось неудачей на первом этапе из-за несоответствия системных требований. Редакция располагает только сервером с операционной системой Windows 7, и обновление этой системы оказалось невозможным из-за риска потери данных. К сожалению, данный факт представляет значительное препятствие для развертывания базы данных, необходимой для функционирования программного комплекса.

Системные требования программного комплекса "Фабрика новостей" предусматривают использование операционных систем, более современных, чем Windows 7. Это обусловлено не только необходимостью обеспечить поддержку новых функций и технологий, но и обеспечить безопасность и стабильность работы системы в целом.

Несмотря на значимость данной проблемы, есть положительные перспективы на её решение. В рамках переезда редакции в новую студию на ул. Советская, который запланирован на вторую половину 2023 г., предполагается возможность установки нового сервера, соответствующего требованиям программного комплекса.

Учитывая перспективы переезда редакции и создания новой инфраструктуры, необходимо провести дополнительное исследование и планирование для обеспечения безопасного и эффективного внедрения программного комплекса "Фабрика новостей" в редакции «Ирбитский вестник».

# ГЛАВА 3. РУКОВОДСТВА ОПЕРАТОРА И ПРОГРАММИСТА

## 3.1 Руководство оператора

В этом руководстве мы подробно рассмотрим основные функции, инструменты и задачи, с которыми вы будете работать.

Композитор Сюжетов (КС):

Композитор Сюжетов является одним из ключевых инструментов в программном комплексе "Фабрика новостей". С его помощью вы можете создавать и редактировать сюжеты в виде блочного редактора. КС позволяет вам структурировать сюжеты, добавлять различные элементы, такие как текст, изображения, видео и аудио, а также управлять их порядком. Это помогает вам создавать динамичные и информативные новостные сюжеты, соответствующие требованиям вашего телевизионного канала.

Телевизионный Штаб (ТШ):

Телевизионный Штаб является мощным инструментом для управления проектами в программном комплексе "Фабрика новостей". Он предоставляет вам полный контроль над телевизионными проектами, съемками и материалами. С помощью ТШ вы можете создавать новые проекты, устанавливать сроки выполнения задач, добавлять съемки и назначать ответственных. Вы также можете отслеживать ход работы над проектами, проверять выполнение задач и контролировать доступ к материалам и ресурсам. ТШ помогает вам эффективно организовывать и управлять всеми аспектами вашего телевизионного производства.

Инспектор Сюжетной Точности (ИСТ):

Инспектор Сюжетной Точности играет важную роль в обеспечении точности и качества телевизионных сюжетов. Он позволяет в реальном времени отслеживать работу над сюжетами и выявлять возникающие ошибки. С помощью ИСТ вы можете обнаруживать опечатки, несоответствия, неправильное форматирование или другие проблемы в сюжетах и оперативно информировать редакторов и операторов о необходимости исправлений. ИСТ помогает вам поддерживать высокий уровень точности и актуальности ваших телевизионных новостей.

Планировщик Кинопроизводства (ПК):

Планировщик Кинопроизводства является незаменимым инструментом для планирования съемок в программном комплексе "Фабрика новостей". С помощью ПК вы можете создавать расписания съемок, устанавливать время, место и другие детали. ПК позволяет вам эффективно организовывать работу съемочной группы, согласовывать расписания с участниками проекта и контролировать выполнение плана. Он обеспечивает точное планирование и координацию всех этапов процесса кинопроизводства, помогая вам достигать результатов в установленные сроки.

Виртуальный Интерактивный Агент (ВАИ):

Виртуальный Интерактивный Агент, или бот ВАИ, предоставляет удобный доступ к запланированным съемкам, ходу работы над проектами и отчетности из любой точки с доступом в интернет. Бот ВАИ интегрирован с социальной сетью Вконтакте, позволяя операторам работать с программным комплексом "Фабрика новостей" даже удаленно. Через бота ВАИ вы можете получать уведомления о новых съемках, получать информацию о проектах, просматривать отчеты и взаимодействовать с другими участниками команды. Бот ВАИ обеспечивает удобство и мобильность в использовании программного комплекса, позволяя операторам быть в курсе происходящего и эффективно взаимодействовать со всей командой.

Каждая программа в составе комплекса "Фабрика новостей" предоставляет свои уникальные возможности и инструменты, которые помогают операторам редакции телеканала эффективно работать с сюжетами, управлять проектами, обеспечивать точность и качество материалов, планировать съемки и иметь доступ к необходимой информации в удобной форме.

## 3.2 Руководство программиста

Установка программного комплекса "Фабрика новостей" включает несколько этапов, каждый из которых требует определенных действий для успешного развертывания системы.

В начале процесса установки необходимо подготовить серверное окружение, на котором будет работать СУБД. Это должен быть физический сервер с прямым сетевым соединением с клиентами. Важно убедиться, что на сервере установлены все необходимые компоненты для работы комплекса. Далее следует установка и настройка операционной системы Windows на сервере. На этом этапе необходимо установить все необходимые пакеты и обновления операционной системы.

После этого начинается установка и настройка СУБД MongoDB. Это требует установки самой базы данных и ее настройки, включая параметры безопасности, доступа и конфигурации хранилища данных. Установщик стабильной версии СУБД поставляется вместе с комплексом, но при желании её можно обновить, скачав последнюю версию с официального сайта MongoDB.

После развёртывания СУБД следует установка и настройка самого программного комплекса. Это включает распаковку на компьютерах клиентов и настройку доступа к базе данных. Все клиентские программы поставляются в портативном издании и не требуют установки. Для начала работы необходимо только распаковать необходимые компоненты фабрики и провести их соединение с сервером.

В случае, если это не удаётся по адресу по умолчанию (127.0.0.1: 27017), пользователь должен ввести эти данные в представленной форме:

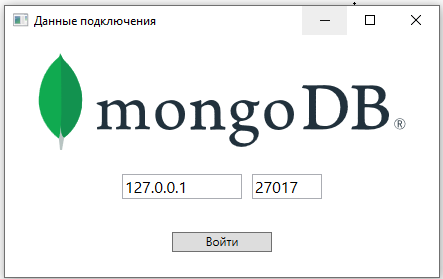


Рис. 9 Форма соединения с MongoDB сервером.

По нажатию кнопки «Войти» приложение попытается связаться с сервером. Если соединение будет установлено, пользователь попадает в главное окно, иначе, форма соединения с сервером будет показана снова.

После завершения начальной конфигурации программного комплекса следует выполнить тестирование и отладку системы. Это позволяет проверить правильность установки, обнаружить и устранить возможные ошибки или проблемы в работе комплекса.

В конечном итоге, после успешной установки и отладки программного комплекса "Фабрика новостей", система готова к использованию. На этом этапе можно приступить к регистрации пользователей, настройке прав доступа и использованию функционала комплекса для работы с сюжетами, проектами, съемками и другими элементами телевизионного процесса.

Системные требования для сервера:

* Процессор: Многоядерный процессор с тактовой частотой от 2.5 ГГц или выше;
* Оперативная память: 8 ГБ или больше;
* Жесткий диск: Рекомендуется использовать SSD-накопитель, 20 ГБ свободного пространства;
* Операционная система: Windows 10 или новее;
* Сетевое соединение: Стабильное высокоскоростное соединение с доступом в Интернет;
* Доп. компоненты: .NET Framework 4.8 или новее.

Системные требования для клиента:

* Процессор: Многоядерный процессор с тактовой частотой от 1.8 ГГц или выше;
* Оперативная память: 4 ГБ или больше;
* Жесткий диск: 4 ГБ свободного пространства;
* Операционная система: Windows 7 или новее;
* Сетевое соединение: Стабильное соединение с доступом в Интернет;
* Доп. компоненты: .NET Framework 4.8 или новее.

Важно отметить, что конкретные системные требования могут варьироваться в зависимости от конфигурации программного комплекса, его компонентов и требований к производительности.

# ГЛАВА 4. ЭКОНОМИЧЕСКАЯ ЧАСТЬ

В ходе работы над дипломом был разработан программный комплекс «Фабрика новостей» для телеканала «НТС-Ирбит» в экономической части дипломной работы производится расчет себестоимости программного комплекса.

Все исследования и разработка проводились на предприятии редакция телерадиовещания «Ирбитский вестник»

Для принятия решения по целесообразности реализации проекта необходимо рассчитать смету затрат на разработку проекта и произвести расчет экономического эффекта.

Затраты на разработку включают:

* затраты на заработную плату разработчикам проекта;
* затраты на социальные выплаты;
* материальные затраты;
* амортизационные отчисления;
* затраты на электроэнергию.

Работа выполнялась в течение двух с половиной месяцев: в период с 18.04.2023 по 18.06.2023.

## 4.1 Затраты на заработную плату и социальные выплаты

Затраты на заработную плату разработчикам проекта складываются из затрат на заработную плату программистам. Заработная плата – вознаграждение за труд в зависимости от квалификации работника. Оплачивается исходя из отработанного времени и тарифной ставки рабочего в соответствии с его квалификацией.

Затраты определяются исходя из количества нормо-часов, необходимых для выполнения отдельных работ и тарифных часовых ставок.

Оклад техника – программиста на предприятии в 2023 году составил 22500 руб. При расчете тарифной ставки () будем исходить от 8-ми часового рабочего дня, 22-х дневного рабочего месяца. Учтен Уральский коэффициент, составляющий 15%,

В таблице 4.1 приведены общие затраты на инженерно-технические работы по проекту.

Таблица 11. Общие затраты на инженерно-технические работы по проекту

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Этап разработки | Количество дней | Количество часов | Ставка, руб/час | Основная заработная плата, руб. |
| Обзор существующих алгоритмических и программных решений по тематике исследования | 5 | 30 | 147,02 | 4410,60 |
| Разработка функциональной модели | 5 | 30 | 4410,60 |
| Разработка алгоритма работы программного комплекса | 15 | 90 | 13231,80 |
| Разработка программного комплекса | 20 | 120 | 17642,40 |
| Тестирование программного комплекса | 5 | 30 | 4410,60 |
| Итого | 45 | 270 | 147,02 | 39695,40 |

Для расчета итоговой суммы заработной платы за разработку пакета программного обеспечения следует вычислить сумму основной () Таким образом, основная заработная плата составит:

Социальные выплаты включают в себя затраты на страховые взносы - 30 %

Отчисления на обязательное страхование от несчастных случаев на производстве и профессиональных заболеваний составляет 0,2% (I класс профессионального риска).

Норматив начисления составит:

Суммарные затраты на страховые взносы составят:

## 4.2 Материальные затраты

Материальные затраты включают в себя стоимость расходных материалов (канцелярские принадлежности, бумага) и средств разработки. Данные взяты по факту (нормативу) расхода на предприятии. В таблице 4.2 приведен расчет материальных затрат.

Таблица 12 – Расчёт материальных затрат.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Наименование | Количество | Стоимость единицы, руб | Сумма затрат, руб |
| 1 | Бумага (формат А4), упаковка | 1 | 210,00 | 210,00 |
| 2 | Канцелярские принадлежности: | | | |
| - ручка, шт. | 2 | 20,00 | 40,00 |
| - ластик, шт. | 1 | 10,00 | 10,00 |
| - карандаш, шт. | 2 | 10,00 | 20,00 |
| - скоросшиватель, шт. | 2 | 40,00 | 80,00 |
| 3 | Тонер картридж для принтера (заправка) | 1 | 150,00 | 250,00 |
| Итого: | | | | 510,00 |

Таким образом, общая сумма затрат по статье «Материальные затраты» составила 510 руб.

## 4.3 Затраты на электроэнергию

Расчет суммы затрат на электроэнергию производится исходя из стоимости одного киловатт-часа электроэнергии; количества единиц установленного оборудования (берется из технической документации) и количества часов работы каждого вида оборудования. В таблице 4. приведен расчет затрат на электроэнергию.

Расчет затрат на электроэнергию производится на основе данных о стоимости   
1 кВт/ч, суммарной потребляемой мощности оборудования за время работы этого оборудования.

Таблица 13 – Расчёт затрат на электроэнергию.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Наименование оборудования | Кол-во единиц, шт. | Потребляемая мощность, кВт/час | Кол-во часов, час | Стоимость одного Квт/час | Затраты, руб |
| Системный блок | 1 | 0,5 | 270 | 5,15 | 695,25 |
| Монитор | 1 | 0,016 | 270 | 5,15 | 22,25 |
| Источник бесперебойного питания (ИБП) | 1 | 0,24 | 270 | 5,15 | 333,75 |
| Принтер | 1 | 0,304 | 1 | 5,15 | 1,57 |
| Лампа освещения | 10 | 0,007 | 270 | 5,15 | 97,34 |
| Итого: | | | | | 1150,16 |

Таким образом, общие затраты на электроэнергию потребляемого оборудования составили 1150,16 руб.

## 4.4 Амортизационные отчисления

Амортизация – это процесс накопления средств путем перечисления их стоимости на вновь созданный продукт. Амортизационные накопления включаются в состав себестоимости продукции.

Объектами для начисления амортизации выступают основные средства, находящиеся у предприятия на правах собственности, хозяйственного ведения и оперативного управления, а также нематериальные активы.

Начисление амортизации начинается с 1-го числа месяца, следующего за месяцем принятия объекта к учету. Начисляется амортизация в течение срока полезного использования объекта. Сумма начисленной амортизации отражается в том отчетном периоде, к которому она относится и начисляется вне зависимости от результатов хозяйственной деятельности предприятия в отчетном году.

Для расчета общей суммы амортизационных отчислений необходимо рассчитать первоначальную стоимость приобретенного основного оборудования и определить срок его полезного использования.

Первоначальная стоимость амортизируемого оборудования определяется по данным бухгалтерского учета и включает в себя: стоимость приобретения; транспортные расходы по доставке основных фондов; затраты на монтаж, установку и окончательное доведение до состояния, в котором оборудование пригодно для эксплуатации.

Сроком полезного использования признается период, в течение которого объект служит для выполнения целей деятельность налогоплательщика. В таблице 5.5 приведен перечень оборудования, подлежащего амортизации.

Годовая сумма амортизационных отчислений вычисляется по формуле:

где – годовая сумма амортизационных отчислений, руб; – первоначальная стоимость оборудования, руб; – срок полезного использования.



В данном случае срок полезного использования составляет 60 месяцев. Результаты расчетов приведены в Таблица .

Таблица 14 – Расчет амортизационных отчислений.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Наименование  оборудования | Кол-во | Первоначальная стоимость, руб. | Срок полезного использования, мес. | Сумма амортизационных отчислений, руб., | |
| Годовая | В месяц |
| 1 | Системный блок | 1 | 30 000,00 | 60 | 6000,00 | 500,00 |
| 2 | Монитор | 2 | 6 500,00 | 60 | 2600,00 | 216,67 |
| 3 | ИБП | 1 | 4 200,00 | 60 | 840,00 | 70,00 |
| 4 | Принтер | 1 | 18 000,00 | 60 | 3 600,00 | 300,00 |
| Итого: | | 4 | 47 700,00 |  | 10043,00 | 1086,67 |

Сумма годовых амортизационных отчислений составила 10043 руб. (1086,67 руб. в месяц), таким образом, за сроки выполнения проекта (2 месяца) амортизационные отчисления составят 2173,34 руб.

## 4.5 Расчет производственной себестоимости

Затраты на проектирование и производство продукта иначе называют производственной себестоимостью. Расчет полной себестоимости выполнения дипломного проекта производится путем суммирования всех видов затрат. Статьи затрат и итоговая себестоимость приведены в таблице 5.7**Ошибка! Источник ссылки не найден.**.

Таблица 15 – Смета затрат на проектирование и реализацию дипломного проекта

|  |  |  |
| --- | --- | --- |
| № | Статьи затрат | Сумма, руб. |
| 1 | Материальные затраты | 510,00 |
| 2 | Выплата за проект штатного сотрудника | 39695,40 |
| 3 | Обязательные социальные отчисления для штатного сотрудника | 11988,01 |
| 4 | Затраты на электроэнергию | 1150,16 |
| 5 | Амортизационные отчисления | 2173,34 |
| Итого цена разработки проекта | | 55516,91 |

Общая сумма затрат на разработку и изготовление данного дипломного проекта составила 55516,91 руб.

Экономический анализ работы показал, что затраты на разработку составляют 55516,91 руб., причем основной статьей расходов является оплата труда техника-программиста;

# ГЛАВА 5. Охрана труда

## 5.1 Общие требования охраны труда

К работе в качестве оператора ПК допускаются лица не моложе 18 лет, не имеющие противопоказаний по состоянию здоровья, прошедшие вводный и первичный на рабочем месте инструктажи по охране труда, обученные безопасным методам и приемам ведения работ, прошедшие стажировку на рабочем месте и проверку знаний требований охраны труда.

В течение трудовой деятельности оператор ПК обязан проходить повторные инструктажи на рабочем месте – не реже 1 раза в 6 месяцев, периодический медосмотр – в соответствии с Приказом Минздрава № 302н, очередную проверку знаний требований охраны труда – не реже 1 раза в год.

Оператор ПК проходит внеплановый инструктаж: при изменении технологического процесса или правил по охране труда, замене или модернизации оборудования, изменении условий и организации труда, при нарушениях инструкций по охране труда, перерывах в работе более чем на 60 календарных дней (для работ, к которым предъявляются повышенные требования безопасности – 30 календарных дней).

Оператор ПК обязан:

— соблюдать Правила внутреннего трудового распорядка;

— соблюдать требования настоящей инструкции, инструкции о мерах пожарной безопасности, инструкции по электробезопасности;

— соблюдать правила личной гигиены, перед приемом пищи необходимо мыть руки с мылом;

— уметь оказывать первую помощь пострадавшему, знать местонахождение аптечки, а также уметь пользоваться средствами пожаротушения и знать место их нахождения.

Курить разрешается только в специально отведенном и оборудованном для этого месте, принимать пищу разрешается в комнате отдыха и приема пищи. Пить воду только из специально предназначенных для этого установок.

В помещении, где проводятся работы на ПК, необходимо создать оптимальные условия зрительной работы. Освещенность рабочего места при смешанном освещении (в горизонтальной плоскости в зоне размещения клавиатуры и рабочих документов) должна быть в пределах от 300 до 500 Лк. Основной поток естественного света должен быть слева, солнечные лучи и блики не должны попадать в поле зрения работающего и на экраны видеомониторов.

Монитор должен находиться на расстоянии 50-70 см от глаз оператора ПК и иметь антибликовое покрытие. Покрытие должно также обеспечивать снятие электростатического заряда с поверхности экрана, исключать искрение и накопление пыли.

Нельзя загораживать заднюю стенку системного блока или ставить ПК вплотную к стене, это приводит к нарушению охлаждения системного блока и его перегреву.

Режим работы и отдыха должен зависеть от характера выполняемой работы. При вводе данных, редактировании программ, считывании информации с экрана непрерывная продолжительность работы с ПК не должна превышать 4 часа за рабочий день при 8-ми часовом рабочем дне. Через каждый час работы необходимо делать перерывы на отдых по 5-10 минут или по 15-20 минут каждые два часа работы.

Для снятия общего утомления во время перерывов необходимо проводить физкультпаузы, включающие упражнения общего воздействия, улучшающие функциональное состояние нервной, сердечно-сосудистой, дыхательной систем, а также улучшающих кровообращение, снижающих мышечное утомление.

В процессе работы на оператора ПК могут воздействовать следующие опасные и вредные факторы:

— повышенные уровни электромагнитного излучения;

— повышенный уровень статического электричества;

— повышенный уровень прямой блескости;

— неравномерность распределения яркости в поле зрения;

— напряжение зрения;

— длительные статические нагрузки;

— монотонность труда;

— большой объем информации;

— интеллектуальные нагрузки.

Оператор ПК должен извещать своего непосредственного руководителя о любой ситуации, угрожающей жизни и здоровью людей, о каждом несчастном случае, происшедшем на производстве, об ухудшении состояния своего здоровья, в том числе о проявлении признаков острого заболевания.

Курить следует только в специально отведенных и оборудованных для этого местах. Запрещается употребление спиртных напитков и появление на работе в нетрезвом состоянии, в состоянии наркотического или токсического опьянения.

За невыполнение данной инструкции виновные привлекаются к ответственности согласно действующему законодательству Российской Федерации.

## 5.2. Требования охраны труда перед началом работы

Подготовить свое рабочее место к работе, убрать посторонние предметы. Произвести визуальный осмотр ПК, убедиться в исправности электророзеток, штепсельных вилок, питающих электрошнуров.

Включить ПК в сеть 220В, при этом штепсельную вилку держать за корпус.

Обо всех обнаруженных неисправностях оборудования, электропроводки и других неполадках сообщить своему непосредственному руководителю и приступить к работе только после их устранения.

## 5.3. Требования охраны труда во время работы

Во время работы быть внимательным, не отвлекаться на посторонние дела и разговоры, не отвлекать других.

Рабочее место должно быть оборудовано так, чтобы исключать неудобные позы и длительные статические напряжения тела.

При работе на ПК должна быть исключена возможность одновременного прикосновения к оборудованию и к частям помещения или оборудования, имеющим соединение с землей (радиаторы батарей, металлоконструкции).

Во время работы нельзя класть на монитор бумаги, книги и другие предметы, которые могут закрыть его вентиляционные отверстия.

Оператору ПК во время работы запрещается:

— касаться одновременно экрана монитора и клавиатуры;

— прикасаться к задней панели системного блока при включенном питании;

— переключение разъемов интерфейсных кабелей периферийных устройств при включенном питании;

— загромождать верхние панели устройств бумагами и посторонними предметами;

— производить частые переключения питания;

— допускать попадание влаги на поверхность системного блока, монитора;

— производить самостоятельно вскрытие и ремонт оборудования.

## 5.4. Требования охраны труда в аварийных ситуациях

При возникновении неисправности необходимо отключить ПК от сети. ЗАПРЕЩАЕТСЯ пытаться самостоятельно устранить причину неисправности, об этом необходимо сообщить в службу технического обслуживания.

В случае возгорания электропроводки или ПК немедленно отключить его от сети, сообщить об этом в пожарную часть по телефону 101 или 112 и приступить к тушению пожара углекислотным или порошковым огнетушителем.

Запрещается применять пенные огнетушители для тушения электропроводок и оборудования под напряжением, так как пена — хороший проводник электрического тока.

При несчастном случае немедленно освободить пострадавшего от действия травмирующего фактора, соблюдая собственную безопасность, оказать пострадавшему первую помощь, при необходимости вызвать бригаду скорой помощи по телефону 103 или 112. По возможности сохранить обстановку, при которой произошел несчастный случай, если это не угрожает жизни и здоровью окружающих и не нарушает технологического процесса, для проведения расследования причин возникновения несчастного случая, или зафиксировать на фото или видео. Сообщить руководству предприятия.

## 5.5. Требования охраны труда по окончании работы

Отключить ПК от сети, штепсельную вилку при этом держать за корпус. Запрещается отключать ПК за электропровод. При отключении ПК со съемным шнуром питания сначала необходимо отключить вилку от розетки, а затем отключить питающий шнур от ПК.

Привести в порядок рабочее место.

Чистку ПК от пыли необходимо производить только после его полного отключения от сети.

# ЗАКЛЮЧЕНИЕ

В рамках данной дипломной работы был разработан программный комплекс "Фабрика новостей", который представляет собой комплексное решение для модернизации условий работы редакции телеканала. Целью работы над проектом было упростить и оптимизировать процессы создания, редактирования, планирования и контроля телевизионных сюжетов.

В результате проведенного исследования и разработки был создан программный комплекс, состоящий из нескольких взаимосвязанных программных компонентов. Композитор Сюжетов (КС) предоставляет редактор сюжетов с возможностью создания и редактирования блоков информации. Телевизионный Штаб (ТШ) выполняет функции менеджера проектов, обеспечивая контроль над телевизионными проектами, съемками и материалами. Виртуальный Интерактивный Агент (ВАИ) предоставляет доступ к запланированным съемкам, ходу работы над проектами и отчетности через социальную сеть ВКонтакте.

Одним из ключевых достижений данного программного комплекса является его модульность и гибкость. Каждый компонент выполняет свою специфическую функцию, что позволяет разделить задачи и обеспечить более эффективную работу всей редакции. Благодаря использованию отдельных классов и модулей, расширение и изменение функциональности программного комплекса становится более удобным и простым.

Основной принцип разработки программного комплекса заключался в использовании современных технологий и подходов. Для разработки был выбран язык программирования C# с использованием библиотеки VkNet для реализации Виртуального Интерактивного Агента (ВАИ). Кроме того, была использована СУБД MongoDB и официальный MongoDB .NET Driver для работы с базой данных, в которой хранятся сущности проекта.

В конце можно отметить, что разработанный программный комплекс "Фабрика новостей" может внести значимый вклад в область модернизации условий работы редакции телеканала. Он позволяет упростить процессы создания и редактирования телевизионных сюжетов, планирования и контроля съемок, а также обеспечивает доступ к информации и отчетности через социальную сеть ВКонтакте. Дальнейшее развитие и совершенствование программного комплекса может включать в себя внедрение механизмов мониторинга и управления, оптимизацию работы и расширение функциональности.

**Список использованной литературы**

1. The Official C# .NET Driver for MongoDB [Электронный ресурс] // GitHub : Хостинг IT-проектов. URL : https://github.com/mongodb/mongo-csharp-driver (дата обращения: 22.02.2023)
2. MongoDB Documentation [Электронный ресурс] // MongoDB : Сайт разработчиков СУБД MongoDB. URL : https://www.mongodb.com/docs/ (дата обращения: 27.02.2023)
3. MongoDB .NET Driver API Documentation [Электронный ресурс] // MongoDB .NET Driver : Документация по драйверу .NET MongoDB. URL: https://mongodb.github.io/mongo-csharp-driver/2.18/apidocs/html/N\_MongoDB\_Driver.htm (дата обращения: 24.02.2023)
4. Official .NET driver for MongoDB [Электронный ресурс] // NuGet : Система управления пакетами для .NET. URL : https://www.nuget.org/packages/mongodb.driver (дата обращения: 21.02.2023)
5. MongoDB Driver Quick Tour [Электронный ресурс] // MongoDB .NET Driver: Документация по драйверу .NET MongoDB. URL: https://mongodb.github.io/mongo-csharp-driver/2.18/getting\_started/quick\_tour/ (дата обращения: 28.02.2023)
6. Serialization Tutorial [Электронный ресурс] // MongoDB .NET Driver : Документация по драйверу .NET MongoDB. URL: https://mongodb.github.io/mongo-csharp-driver/1.11/serialization/ (дата обращения: 27.01.2023)
7. C# Class MongoDB.Bson.IO.BsonDocumentWriter [Электронный ресурс] // Hot Examples : Агрегатор примеров использования исходного кода. URL : https://csharpdoc.hotexamples.com/class/MongoDB.Bson.IO/BsonDocumentWriter# (дата обращения: 24.01.2023)
8. Using C# MongoDB Driver, how to serialize a collection of object referements? [Электронный ресурс] // Stack Overflow : Система вопросов и ответов о программировании. URL : https://stackoverflow.com/questions/27537543/using-c-sharp-mongodb-driver-how-to-serialize-a-collection-of-object-referement (дата обращения: 21.01.2023)
9. Inserting .NET object into MongoDB [Электронный ресурс] // Stack Overflow : Система вопросов и ответов о программировании. URL : https://stackoverflow.com/questions/15224048/inserting-net-object-into-mongodb (дата обращения: 26.02.2023)
10. Full text search in mongodb in .net [Электронный ресурс] // Stack Overflow : Система вопросов и ответов о программировании. URL : https://stackoverflow.com/questions/41356544/full-text-search-in-mongodb-in-net (дата обращения: 28.01.2023)
11. In c# what does 'where T : class' mean? [Электронный ресурс] // Stack Overflow : Система вопросов и ответов о программировании. URL : https://stackoverflow.com/questions/3786774/in-c-sharp-what-does-where-t-class-mean (дата обращения: 22.01.2023)
12. MongoDb custom collection serializer [Электронный ресурс] // Stack Overflow : Система вопросов и ответов о программировании. URL : https://stackoverflow.com/questions/30421379/mongodb-custom-collection-serializer (дата обращения: 25.02.2023)
13. How to get File Created Date and Modified Date [Электронный ресурс] // Stack Overflow : Система вопросов и ответов о программировании. URL : https://stackoverflow.com/questions/23243336/how-to-get-file-created-date-and-modified-date (дата обращения: 23.01.2023)
14. BsonExtraElements with custom serializer [Электронный ресурс] // MongoDB : Сайт разработчиков СУБД MongoDB. URL : https://www.mongodb.com/community/forums/t/bsonextraelements-with-custom-serializer/16742 (дата обращения: 01.03.2023)
15. Словари ресурсов [Электронный ресурс] // METANIT : Сайт о программировании. URL : https://metanit.com/sharp/wpf/9.3.php (дата обращения: 20.02.2023)
16. Generic type constraint [Электронный ресурс] // Microsoft Learn : онлайн-платформа для обучения по технологиям Microsoft. URL : https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/where-generic-type-constraint?source=recommendations (дата обращения: 26.01.2023)
17. Constraints on type parameters [Электронный ресурс] // Microsoft Learn : онлайн-платформа для обучения по технологиям Microsoft. URL : https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/generics/constraints-on-type-parameters?redirectedfrom=MSDN (дата обращения: 20.02.2023)

# ПРИЛОЖЕНИЕ

***Приложение 1 Программный код DBProvider.cs (Статический провайдер методов работы с БД)***

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Threading;

using System.Windows;

using System.Windows.Media;

using System.Xml.Serialization;

using MongoDB.Bson;

using MongoDB.Bson.Serialization;

using MongoDB.Driver;

using MongoDB.Driver.Core.Clusters;

using NfModels.Services.Serializers;

using NfModels.ViewModels.Base;

using NfModels.ViewModels.NewsFactoryObjects;

using NfModels.Views.Dialogs;

namespace NfModels.Services;

/// <summary>

/// Статический провайдер методов работы с БД

/// </summary>

public static class DBProvider

{

private const string RecentConnectionPath = "recent.d";

private static readonly IMongoCollection<ProjectViewModel> ProjectsCollection;

private static readonly IMongoCollection<FilmingViewModel> FilmingsCollection;

private static readonly IMongoCollection<MaterialViewModel> MaterialsCollection;

private static readonly IMongoCollection<PersonViewModel> PersonsCollection;

private static readonly IMongoCollection<TagViewModel> TagsCollection;

static DBProvider()

{

BsonSerializer.RegisterSerializer(typeof(Color), new ColorSerializer());

BsonSerializer.RegisterSerializer(typeof(TagList), new TagListSerializer());

BsonSerializer.RegisterSerializer(typeof(CrewList), new CrewListSerializer());

BsonSerializer.RegisterSerializer(typeof(MaterialList), new MaterialSerializer());

var client = Connect();

var db = client.GetDatabase("NewsFactory");

ProjectsCollection = db.GetCollection<ProjectViewModel>("Projects");

FilmingsCollection = db.GetCollection<FilmingViewModel>("Filmings");

MaterialsCollection = db.GetCollection<MaterialViewModel>("Materials");

PersonsCollection = db.GetCollection<PersonViewModel>("Persons");

TagsCollection = db.GetCollection<TagViewModel>("Tags");

}

/// <summary>

/// Наладка соединения с БД

/// </summary>

/// <returns>Клиент MongoDB</returns>

private static MongoClient Connect()

{

var firstTry = true;

var settings = LoadConnectionSettings();

while (true)

{

MongoClient client;

if (!firstTry) {

settings = DBEnterDialog.PickSettings();

if (settings is null) {Application.Current.Shutdown();

return null!;

}

}

client = new MongoClient(settings);

if (CheckConnection(client))

{

SaveConnectionSettings((settings.Server.Host, settings.Server.Port));

return client;

}

firstTry = false;

}

}

/// <summary>

/// Сохранение настроек соединения в файл

/// </summary>

/// <param name="settings">Настройки соединения</param>

private static void SaveConnectionSettings((string, int) settings)

{

var xml = new XmlSerializer(typeof((string, int)));

using var fs = new FileStream(RecentConnectionPath, FileMode.Create, FileAccess.Write);

xml.Serialize(fs, settings);

fs.Close();

}

/// <summary>

/// Загрузка настроек соединения из файла

/// </summary>

/// <returns>Настройки соединения</returns>

private static MongoClientSettings LoadConnectionSettings()

{

if (File.Exists(RecentConnectionPath))

{

var xml = new XmlSerializer(typeof((string, int)));

using var fs = new FileStream(RecentConnectionPath, FileMode.Open);

var newSettings = ((string, int))xml.Deserialize(fs);

fs.Close();

return new MongoClientSettings {

Server = new MongoServerAddress(newSettings.Item1, newSettings.Item2),

ConnectTimeout = TimeSpan.FromSeconds(5),

};

}

var settings = new MongoClientSettings {

Server = new MongoServerAddress("localhost", 27017),

ConnectTimeout = TimeSpan.FromSeconds(5),

};

return settings;

}

/// <summary>

/// Проверка статуса соединения с БД

/// </summary>

/// <param name="client">Клиент соединения</param>

/// <returns>Есть ли соединение</returns>

private static bool CheckConnection(IMongoClient client)

{

var count = 0;

while (client.Cluster.Description.State == ClusterState.Disconnected)

{

Thread.Sleep(100);

if (count++ >= 50) return false;

}

return true;

}

/// <summary>

/// Сохранение объекта фабрики в БД

/// </summary>

/// <param name="instance">Экземпляр объекта фабрики</param>

public static void Save(NewsFactoryObjectViewModel instance)

{

switch (instance)

{

case ProjectViewModel pr:

Save(pr);

break;

case FilmingViewModel pr:

Save(pr);

break;

case MaterialViewModel pr:

Save(pr);

break;

case TagViewModel pr:

Save(pr);

break;

case PersonViewModel pr:

Save(pr);

break;

}

}

/// <summary>

/// Загрузка сохранённых данных об объекте в БД

/// </summary>

/// <param name="viewModel">Экземпляр объекта фабрики</param>

/// <returns>Последние сохранённые данные объекта фабрики</returns>

/// <exception cref="ArgumentOutOfRangeException">Неизвестный тип объекта</exception>

public static NewsFactoryObjectViewModel? LoadSavedData(NewsFactoryObjectViewModel viewModel)

{

return viewModel switch

{

ProjectViewModel pr => ProjectsCollection.FindSync(GetVmFindFilter(pr)).FirstOrDefault(),

FilmingViewModel pr => FilmingsCollection.FindSync(GetVmFindFilter(pr)).FirstOrDefault(),

MaterialViewModel pr => MaterialsCollection.FindSync(GetVmFindFilter(pr)).FirstOrDefault(),

TagViewModel pr => TagsCollection.FindSync(GetVmFindFilter(pr)).FirstOrDefault(),

PersonViewModel pr => PersonsCollection.FindSync(GetVmFindFilter(pr)).FirstOrDefault(),

\_ => throw new ArgumentOutOfRangeException()

};

}

/// <summary>

/// Сохранение модели представления проекта в БД

/// </summary>

/// <param name="instance">Экземпляр проекта</param>

private static void Save(ProjectViewModel instance)

{

SaveObject(ProjectsCollection, instance);

}

/// <summary>

/// Сохранение модели представления съёмки в БД

/// </summary>

/// <param name="instance">Экземпляр съёмки</param>

private static void Save(FilmingViewModel instance)

{

SaveObject(FilmingsCollection, instance);

}

/// <summary>

/// Сохранение модели представления материала в БД

/// </summary>

/// <param name="instance">Экземпляр материала</param>

private static void Save(MaterialViewModel instance)

{

SaveObject(MaterialsCollection, instance);

}

/// <summary>

/// Сохранение модели представления тега в БД

/// </summary>

/// <param name="instance">Экземпляр тега</param>

private static void Save(TagViewModel instance)

{

SaveObject(TagsCollection, instance);

}

/// <summary>

/// Сохранение модели представления работника в БД

/// </summary>

/// <param name="instance">Экземпляр работника</param>

private static void Save(PersonViewModel instance)

{

SaveObject(PersonsCollection, instance);

}

/// <summary>

/// Универсальное сохранение модели представления объекта фабрики в БД

/// </summary>

/// <param name="collection">Ссылка на коллекцию для сохранения</param>

/// <param name="vm">Экземпляр модели представления</param>

/// <typeparam name="T">Тип объекта фабрики</typeparam>

private static void SaveObject<T>(IMongoCollection<T> collection, T vm) where T : NewsFactoryObjectViewModel

{

try

{

var r = collection.FindOneAndReplace(GetVmFindFilter(vm), vm);

if (r is not null) return;

collection.InsertOne(vm);

if (typeof(T) == typeof(TagViewModel)) TagManager.Refresh();

if (typeof(T) == typeof(PersonViewModel)) PersonManager.Refresh();

}

catch (TimeoutException)

{

LostConnection();

}

}

/// <summary>

/// Вывод сообщения и прекращение работы приложения при разрыве соединения с БД

/// </summary>

private static void LostConnection()

{

MessageBox.Show("Потеряно соединение с сервером", "Внимание!", MessageBoxButton.OK,

MessageBoxImage.Error);

Application.Current.Shutdown();

}

/// <summary>

/// Получение фильтра для нахождения данных о модели представления в БД

/// </summary>

/// <param name="vm">Искомая модель представления</param>

/// <typeparam name="T">Тип модели представления</typeparam>

/// <returns>Фильтр для нахождения данных о модели представления в БД</returns>

private static FilterDefinition<T> GetVmFindFilter<T>(T vm) where T : NewsFactoryObjectViewModel

{

return Builders<T>.Filter.Eq(r => r.Id, vm.Id);

}

/// <summary>

/// Поиск объектов фабрики по запросу поисковой строки

/// </summary>

/// <param name="search">Запросу поисковой строки</param>

/// <returns>Объекты фабрики, удовлетворяющие запросу</returns>

public static IEnumerable<NewsFactoryObjectViewModel> LoadObjects(Search search)

{

try

{

var projects = search.Filter.ProjectFilter is null

? Enumerable.Empty<NewsFactoryObjectViewModel>()

: ProjectsCollection.FindSync(search.Filter.ProjectFilter).ToEnumerable();

var materials = search.Filter.MaterialFilter is null

? Enumerable.Empty<NewsFactoryObjectViewModel>()

: MaterialsCollection.FindSync(search.Filter.MaterialFilter).ToEnumerable()

.Cast<CrewedObjectViewModel>();

var filmings = search.Filter.FilmingFilter is null

? Enumerable.Empty<NewsFactoryObjectViewModel>()

: FilmingsCollection.FindSync(search.Filter.FilmingFilter).ToEnumerable();

return projects.Concat(materials).Concat(filmings);

}

catch (MongoCommandException e)

{

return Enumerable.Empty<NewsFactoryObjectViewModel>();

}

catch (TimeoutException e)

{

LostConnection();

return Enumerable.Empty<NewsFactoryObjectViewModel>();

}

}

/// <summary>

/// Загрузка всех тегов

/// </summary>

/// <returns>Все теги</returns>

public static IQueryable<TagViewModel> LoadTags()

{

return TagsCollection.AsQueryable();

}

/// <summary>

/// Загрузка всех работников

/// </summary>

/// <returns>Все работники</returns>

public static IQueryable<PersonViewModel> LoadPersons()

{

return PersonsCollection.AsQueryable();

}

/// <summary>

/// Загрузка всех материалов по набору их идентификаторов

/// </summary>

/// <param name="ids">Набору искомых идентификаторов </param>

/// <returns>Набор материалов</returns>

public static IEnumerable<MaterialViewModel> LoadMaterials(IEnumerable<ObjectId> ids)

{

return MaterialsCollection.AsQueryable().Where(m => ids.Contains(m.Id));

}

}

***Приложение 2 Программный код FileDialogs.cs (Набор файловых диалогов)***

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Windows.Forms;

using System.Xml.Serialization;

using NfModels.ViewModels.NewsFactoryObjects;

using NfModels.ViewModels.ScriptParts.Base;

using OpenFileDialog = Microsoft.Win32.OpenFileDialog;

using SaveFileDialog = Microsoft.Win32.SaveFileDialog;

namespace NfModels.Services;

public static class FileDialogs

{

/// <summary>

/// Путь по умолчанию для сохранения сюжетов

/// </summary>

public static string DefaultScriptPath =

Path.Join(Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments), "Сюжеты");

/// <summary>

/// Событие, вызываемое при изменении файлов

/// </summary>

public static event EventHandler<FileManagementEventArgs> FileManagement = null!;

/// <summary>

/// Сохранение сюжета

/// </summary>

/// <param name="script">Сюжет</param>

/// <param name="saveAs">Сохранить по новому пути</param>

/// <returns>Удалось ли сохранить</returns>

public static bool Save(ScriptViewModel script, bool saveAs = false)

{

string path;

if (string.IsNullOrWhiteSpace(script.Path) || saveAs)

{

string? newPath;

if (script.IsDefaultName && script.ScriptParts.Count > 0 && script.ScriptParts[0] is TextPart textPart &&

textPart.Content.Length > 10)

newPath = GetNewScriptPath(script.Path,

textPart.Content[..Math.Min(textPart.Content.Length, 50)]);

else newPath = GetNewScriptPath(script.Path, script.Title);

if (newPath is null) return false;

path = newPath;

script.Path = path;

if (script.IsDefaultName)

{

script.Title = Path.GetFileNameWithoutExtension(path);

script.IsDefaultName = false;

}

}

else

{

path = script.Path;

}

try

{

var xml = new XmlSerializer(typeof(ScriptViewModel));

using var fs = new FileStream(path, FileMode.Create, FileAccess.Write);

xml.Serialize(fs, script);

}

catch

{

OnFileManagement(new FileManagementEventArgs

{

ActionType = FileManagementType.Save,

IsSuccesses = false

});

return false;

}

script.IsNotSaved = false;

OnFileManagement(new FileManagementEventArgs

{

Script = script,

ActionType = FileManagementType.Save,

IsSuccesses = true

});

return true;

}

/// <summary>

/// Загрузка сюжета по его пути

/// </summary>

/// <param name="path">Путь к сюжету</param>

/// <returns>Сюжет</returns>

public static ScriptViewModel? Load(string path)

{

try

{

var xml = new XmlSerializer(typeof(ScriptViewModel));

using var fs = new FileStream(path, FileMode.Open);

var newScript = (ScriptViewModel)xml.Deserialize(fs);

newScript.Path = path;

newScript.OnLoaded();

OnFileManagement(new FileManagementEventArgs

{

Script = newScript,

ActionType = FileManagementType.Load,

IsSuccesses = true

});

return newScript;

}

catch (Exception e)

{

return null;

}

}

/// <summary>

/// Вызов диалога получения нового пути сюжета

/// </summary>

/// <param name="previousPath">Предыдущий путь сюжета</param>

/// <param name="title">Заголовок окна</param>

/// <param name="filter">Фильтр</param>

/// <returns>Новый путь сохранения</returns>

public static string? GetNewScriptPath(string? previousPath, string title,

string filter = "Сюжеты|\*.nscr|All files|\*.\*")

{

var dialog = new SaveFileDialog

{

Filter = filter,

RestoreDirectory = true,

OverwritePrompt = true,

Title = title

};

if (!string.IsNullOrWhiteSpace(previousPath))

{

dialog.FileName = Path.GetFileName(previousPath);

dialog.InitialDirectory = Path.GetDirectoryName(previousPath);

}

else

{

dialog.InitialDirectory = DefaultScriptPath;

dialog.FileName = title;

}

var isSuccesses = dialog.ShowDialog();

switch (isSuccesses)

{

case null:

case false:

return null;

default:

return dialog.FileName;

}

}

/// <summary>

/// Вызов диалога открытия материалов из файлов

/// </summary>

/// <param name="previousPath">Предыдущий путь</param>

/// <param name="filter">Фильтр</param>

/// <param name="multiselect">Возможность множественного выбора</param>

/// <returns>Набор материалов</returns>

public static IEnumerable<Material>? OpenMaterial(string? previousPath = null, string filter = "Видео|\*.\*",

bool multiselect = true)

{

var dialog = new OpenFileDialog

{

Filter = filter,

RestoreDirectory = true,

Title = "Открыть материал",

Multiselect = multiselect

};

if (!string.IsNullOrWhiteSpace(previousPath))

{

dialog.FileName = Path.GetFileName(previousPath);

dialog.InitialDirectory = Path.GetDirectoryName(previousPath);

}

// TODO: Настройки пути по умолчанию поиска видео

// dialog.InitialDirectory = SettingsManager.CurrentSettings.DefaultScriptPath;

var isSuccesses = dialog.ShowDialog();

switch (isSuccesses)

{

case null:

case false:

return null;

default:

return dialog.FileNames.Select(m => new Material { Path = m });

}

}

/// <summary>

/// Вызов диалога открытия файлов

/// </summary>

/// <param name="previousPath">Предыдущий путь</param>

/// <param name="filter">Фильтр</param>

/// <param name="multiselect">Возможность множественного выбора</param>

/// <returns>Набор путей</returns>

public static IEnumerable<string>? OpenFiles(string? previousPath = null, string filter = "Файлы |\*.\*",

bool multiselect = true)

{

var dialog = new OpenFileDialog

{

Filter = filter,

RestoreDirectory = true,

Title = "Открыть материал",

Multiselect = multiselect

};

if (!string.IsNullOrWhiteSpace(previousPath))

{

dialog.FileName = Path.GetFileName(previousPath);

dialog.InitialDirectory = Path.GetDirectoryName(previousPath);

}

// TODO: Настройки пути по умолчанию поиска видео

// dialog.InitialDirectory = SettingsManager.CurrentSettings.DefaultScriptPath;

var isSuccesses = dialog.ShowDialog();

switch (isSuccesses)

{

case null:

case false:

return null;

default:

return dialog.FileNames;

}

}

/// <summary>

/// Вызов диалога получения пути для сохранения

/// </summary>

/// <param name="previousPath">Предыдущий путь</param>

/// <param name="filter">Фильтр</param>

/// <param name="title">Заголовок окна</param>

/// <returns>Путь сохранения</returns>

public static string GetNewFilePath(string? previousPath = null, string filter = "Файлы |\*.\*", string title = "Экспорт")

{

var dialog = new SaveFileDialog

{

Filter = filter,

RestoreDirectory = true,

Title = title,

};

if (!string.IsNullOrWhiteSpace(previousPath))

{

dialog.FileName = Path.GetFileName(previousPath);

dialog.InitialDirectory = Path.GetDirectoryName(previousPath);

}

var isSuccesses = dialog.ShowDialog();

switch (isSuccesses)

{

case null:

case false:

return "";

default:

return dialog.FileName;

}

}

/// <summary>

/// Вызов диалога открытия папки

/// </summary>

/// <param name="initialDirectory">Начальная директория</param>

/// <param name="title">Заголов окна</param>

/// <returns>Путь к директории</returns>

public static string? GetDirectory(string? initialDirectory, string? title)

{

using var dialog = new FolderBrowserDialog();

dialog.Description = title;

dialog.SelectedPath = initialDirectory;

var result = dialog.ShowDialog();

switch (result)

{

case DialogResult.Yes:

case DialogResult.OK:

return dialog.SelectedPath;

case DialogResult.None:

case DialogResult.Cancel:

case DialogResult.Abort:

case DialogResult.Retry:

case DialogResult.Ignore:

case DialogResult.No:

default:

return null;

}

}

/// <summary>

/// Вызов диалога выбора файла пути сюжета

/// </summary>

/// <returns>Путь к файлу сюжета</returns>

public static string? GetExistsScriptFileName()

{

var dialog = new OpenFileDialog

{

CheckFileExists = true,

Filter = "Сюжет|\*.nscr",

InitialDirectory = DefaultScriptPath

};

var result = dialog.ShowDialog();

switch (result)

{

case null:

case false:

return null;

}

return dialog.FileName;

}

/// <summary>

/// Вызов события при изменении файлов

/// </summary>

/// <param name="e">Аргументы события</param>

private static void OnFileManagement(FileManagementEventArgs e)

{

FileManagement?.Invoke(null, e);

}

/// <summary>

/// Запись данных во временны файл

/// </summary>

/// <param name="data">Данные для записи</param>

public static void WriteTempFile(string data)

{

File.WriteAllText(Path.Join(Path.GetTempPath(), "ScriptEditorTempFile.dat"), data);

}

/// <summary>

/// Чтение данных из временного файла

/// </summary>

/// <returns>Данные временного файла</returns>

public static string? ReadTempFile()

{

var path = Path.Join(Path.GetTempPath(), "ScriptEditorTempFile.dat");

if (!File.Exists(path)) return null;

var data = File.ReadAllText(path);

File.Delete(path);

return data;

}

}

/// <summary>

/// Аргумент события изменения файлов

/// </summary>

public class FileManagementEventArgs : EventArgs

{

public ScriptViewModel? Script { get; set; }

public FileManagementType ActionType { get; set; }

public bool IsSuccesses { get; set; }

}

/// <summary>

/// Тип действия

/// </summary>

public enum FileManagementType

{

Save,

Load

}

***Приложение 3 Программный код MaterialViewModel.cs (Модель представления материала)***

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.IO;

using System.Linq;

using System.Windows.Input;

using NfModels.Infrastructure.Commands;

using NfModels.Services;

using NfModels.ViewModels.Base;

namespace NfModels.ViewModels.NewsFactoryObjects;

/// <summary>

/// Модель представления материала

/// </summary>

public class MaterialViewModel : CrewedObjectViewModel

{

/// <summary>

/// Массив известных расширений видеофайлов

/// </summary>

public static readonly string[] VideoFileExtensions = new[]

{

".webm", ".mkv", ".flv", ".flv", ".vob", ".ogv", ".ogg", ".drc", ".gif", ".gifv", ".mng", ".avi",

".MTS", ".M2TS", ".TS", ".mov", ".qt", ".wmv", ".yuv", ".rm", ".rmvb", ".viv", ".asf", ".amv",

".mp4", ".m4p", ".m4v", ".mpg", ".mp2", ".mpeg", ".mpe", ".mpv", ".mpg", ".mpeg", ".m2v", ".m4v",

".svi", ".3gp", ".3g2", ".mxf", ".roq", ".nsv", ".flv", ".f4v", ".f4p", ".f4a", ".f4b"

}.Select(t => t.ToLower()).ToArray();

/// <summary>

/// Массив известных расширений аудиофайлов

/// </summary>

public static readonly string[] AudioFileExtensions = new[]

{

".3gp", ".aa", ".aac", ".aax", ".act", ".aiff", ".alac", ".amr", ".ape", ".au", ".awb", ".dss", ".dvf", ".flac",

".gsm", ".iklax", ".ivs", ".m4a", ".m4b", ".m4p", ".mmf", ".mp3", ".mpc", ".msv", ".nmf", ".ogg", ".oga",

".mogg", ".opus", ".ra", ".rm", ".raw", ".rf64", ".sln", ".tta", ".voc", ".vox", ".wav", ".wma", ".wv", ".webm",

".8svx", ".cda"

}.Select(t => t.ToLower()).ToArray();

/// <summary>

/// Массив известных расширений изображений

/// </summary>

public static readonly string[] PictureFileExtensions = new[]

{

".006", ".007", ".3DMF", ".3DX", ".8PBS", ".ABM", ".ABR", ".ADI", ".AEX", ".AI", ".AIS", ".ALBM", ".AMU",

".ARD", ".ART", ".ARW", ".ASAT", ".B16", ".BIL", ".BLEND", ".BLKRT", ".BLZ", ".BMC", ".BMC", ".BMP", ".BOB",

".BR4", ".BR5", ".C4", ".CADRG", ".CATPART", ".CCX", ".CDR", ".CDT", ".CDX", ".CGM", ".CHT", ".CM2", ".CMX",

".CMZ", ".COMICDOC", ".CPL", ".CPS", ".CPT", ".CR2", ".CSF", ".CV5", ".CVG", ".CVI", ".CVI", ".CVX", ".DAE",

".DCIM", ".DCM", ".DCR", ".DCS", ".DDS", ".DESIGN", ".DIB", ".DJV", ".DJVU", ".DNG", ".DRG", ".DRW", ".DRWDOT",

".DT2", ".DVL", ".DWB", ".DWF", ".DXB", ".EASM", ".EC3", ".EDP", ".EDRW", ".EDW", ".EMF", ".EPRT", ".EPS",

".EPSF", ".EPSI", ".EXR", ".FAC", ".FACE", ".FBM", ".FBX", ".FC2", ".FCZ", ".FD2", ".FH11", ".FHD", ".FIT",

".FLIC", ".FLM", ".FM", ".FPF", ".FS", ".FW.PNG", ".FXG", ".GIF", ".GRAFFLE", ".GTX", ".HD2", ".HDZ", ".HPD",

".HPI", ".HR2", ".HTZ4", ".ICL", ".ICS", ".IDW", ".IEF", ".IGES", ".IGR", ".ILBM", ".ILM", ".IMA", ".IME",

".IMI", ".IMS", ".INDD", ".INDT", ".IPJ", ".IRF", ".ITC2", ".ITHMB", ".J2K", ".JIFF", ".JNG", ".JPEG", ".JPF",

".JPG", ".JPG2", ".JPS", ".JPW", ".JT", ".JWL", ".JXR", ".KDC", ".KODAK", ".KPG", ".LDA", ".LDM", ".LET",

".LT2", ".LTZ", ".LVA", ".LVF", ".LXF", ".MAC", ".MACP", ".MCS", ".MCZ", ".MDI", ".MGS", ".MGX", ".MIC", ".MIP",

".MNG", ".MPF", ".MPO", ".MTZ", ".MUR", ".MUR", ".NAV", ".NCR", ".NEU", ".NFF", ".NJB", ".NTC", ".NTH", ".ODI",

".ODIF", ".OLA", ".OPD", ".ORA", ".OTA", ".OTB", ".OTC", ".OTG", ".OTI", ".OVW", ".P21", ".P2Z", ".PAT", ".PC6",

".PC7", ".PCD", ".PCT", ".PCX", ".PDF", ".PDN", ".PEF", ".PI2", ".PIC", ".PIC", ".PICNC", ".PICTCLIPPING",

".PL0", ".PL2", ".PLN", ".PMB", ".PNG", ".POL", ".PP2", ".PPSX", ".PRW", ".PS", ".PS", ".PSB", ".PSD", ".PSF",

".PSG", ".PSP", ".PSPIMAGE", ".PSQ", ".PVL", ".PWD", ".PWS", ".PX", ".PXR", ".PZ2", ".PZ3", ".QTIF", ".QTZ",

".QXD", ".RIC", ".RLC", ".RLE", ".RW2", ".SDK", ".SDR", ".SEC", ".SFW", ".SIG", ".SKP", ".SLDASM", ".SLDDRW",

".SLDPRT", ".SNX", ".SRF", ".SST", ".SUN", ".SVG", ".SVGZ", ".TARGA", ".TCW", ".TCX", ".TEX", ".TGA", ".TIF",

".TIFF", ".TJP", ".TN", ".TPF", ".TPX", ".TRIF", ".TRX", ".U3D", ".UPX", ".URT", ".UTX", ".V00", ".V3D", ".VFS",

".VGA", ".VHD", ".VIS", ".VRL", ".VTX", ".WB1", ".WBC", ".WBD", ".WBZ", ".WEBP", ".WGS", ".WI", ".WMF", ".WNK",

".XDW", ".XIP", ".XSI", ".X\_B", ".X\_T", ".ZDL", ".ZIF", ".ZNO", ".ZPRF", ".ZT"

}.Select(t => t.ToLower()).ToArray();

/// <summary>

/// Массив известных расширений текстовых файлов

/// </summary>

public static readonly string[] TextFileExtensions = new[]

{

".txt", ".doc", ".docx"

}.Select(t => t.ToLower()).ToArray();

/// <summary>

/// Словарь связывающий тип материала и его расширения

/// </summary>

public static readonly Dictionary<MaterialType, string[]> TypesExtensions = new()

{

{ MaterialType.Video, VideoFileExtensions },

{ MaterialType.Audio, AudioFileExtensions },

{ MaterialType.Picture, PictureFileExtensions },

{ MaterialType.Text, TextFileExtensions }

};

public MaterialViewModel()

{

ChangePathCommand = new LambdaCommand(OnChangePathCommandExecute, CanChangePathCommandExecute);

}

public override string TypeName => "Материал";

protected override void Reload(NewsFactoryObjectViewModel other)

{

base.Reload(other);

var viewModel = (MaterialViewModel)other;

Path = viewModel.Path;

Duration = viewModel.Duration;

MyType = viewModel.MyType;

}

/// <summary>

/// Обновления типа материала исходя из его расширения

/// </summary>

private void UpdateType()

{

foreach (var typeExtension in TypesExtensions.Where(typeExtension =>

typeExtension.Value.Contains(System.IO.Path.GetExtension(\_path).ToLower())))

{

MyType = typeExtension.Key;

return;

}

MyType = MaterialType.Other;

}

#region Path

private string \_path = string.Empty;

/// <summary>

/// Путь к материалу

/// </summary>

public string Path

{

get => \_path;

set

{

if (File.Exists(value)) CreationTime = new FileInfo(value).CreationTime;

if (!string.IsNullOrWhiteSpace(value) &&

(string.IsNullOrWhiteSpace(Title) || Title == System.IO.Path.GetFileName(\_path)) &&

System.IO.Path.GetFileName(value) is { } dn)

Title = dn;

Set(ref \_path, value);

UpdateType();

}

}

#endregion

#region Duration

/// <summary>

/// Длительность материала

/// </summary>

public TimeSpan? Duration

{

get => \_duration;

set => Set(ref \_duration, value);

}

private TimeSpan? \_duration;

#endregion

#region MyType

/// <summary>

/// Тип материала

/// </summary>

public MaterialType MyType

{

get => \_myType;

set => Set(ref \_myType, value);

}

private MaterialType \_myType;

#endregion

#region ChangePathCommand

/// <summary>

/// Команда изменения пути материала

/// </summary>

public ICommand ChangePathCommand { get; }

private bool CanChangePathCommandExecute(object p)

{

return true;

}

private void OnChangePathCommandExecute(object p)

{

if (FileDialogs.OpenFiles(Path, multiselect: false) is { } pa) Path = pa.First();

}

#endregion

}

/// <summary>

/// Перечисление возможных типов материала

/// </summary>

public enum MaterialType

{

[Description("Видео")] Video,

[Description("Аудио")] Audio,

[Description("Изображение")] Picture,

[Description("Текст")] Text,

[Description("Другое")] Other

}