БУ ВО

«СУРГУТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ХАНТЫ-МАНСИЙСКОГО АВТОНОМНОГО ОКРУГА - ЮГРЫ»

ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Кафедра информатики и вычислительной техники

УТВЕРЖДАЮ:

Заведующий кафедрой ИВТ

Канд. физ.-мат. наук, доцент

\_\_\_\_\_\_\_\_\_\_\_\_\_С.А.Лысенкова

" \_\_ " \_\_\_\_\_\_\_\_\_\_\_ 2024г.

**КУРСОВОЙ ПРОЕКТ**

по дисциплине «**Методы и средства проектирования**

**информационных систем**»

бакалавра по направлению

09.03.02 - Информационные системы и технологии

на тему ***Программное обеспечение для записи экрана рабочего стола***

Выполнил: бакалавр группы: 607-01

Абзалимов Айдар Амирович

(фамилия, имя, отчество)

по специальности 09.03.02 - Информационные системы и технологии

Проверил:

преподаватель Столбов Д.А. \_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись

Сургут – 2024

**Реферат**

Курсовой проект содержит 45 страниц, 22 рисунка, 4 таблицы, 8 источников, 1 приложение.

***Программное обеспечение для записи экрана рабочего стола***

**Ключевые слова:** программное обеспечение, запись экрана, рабочий стол, visual studio 2022, c# wpf, легкость использования, опытные пользователи, бесплатная версия, водяной знак, ограничение по времени, obs, bandicam, fraps, трудный интерфейс, вес видеофайлов, таблица сравнения, открытый код, настройки медиа, горячие клавиши, инструкция по эксплуатации, горячие клавиши, интерфейс программы.

**Цель работы:** разработать эффективное, доступное и простое программное обеспечение, предназначенное для записи экрана рабочего стола компьютера.

**Основные результаты:**

* проведен анализ предметной области и созданы диаграммы для лучшего представление о характеристиках и взаимосвязях элементов в этой области.
* сформулированы требования к программе в соответствии с особенностями предметной области, чтобы обеспечить функциональность, производительность, безопасность и удобство использования программного продукта.
* реализован программный продукт, соответствующий заданным требованиям.
* готовый продукт протестирован, а также сравнен с его аналогами.

Оглавление

[Словарь терминов 5](#_Toc164074649)

[Введение 7](#_Toc164074650)

[Основная часть 8](#_Toc164074651)

[1. Постановка задачи 8](#_Toc164074652)

[2. Описание предметной области 9](#_Toc164074653)

[2.1. Заинтересованные лица 9](#_Toc164074654)

[2.2. Модель предметной области 10](#_Toc164074655)

[3. Обзор аналогов 13](#_Toc164074656)

[4. Требования к системе 15](#_Toc164074657)

[4.1. Функциональные требования 15](#_Toc164074658)

[4.2. Требования по видам обеспечения 15](#_Toc164074659)

[4.2.1 Требования к информационному обеспечению системы 15](#_Toc164074660)

[4.2.2. Требования к алгоритмическому обеспечению 15](#_Toc164074661)

[4.2.3. Требования к лингвистическому обеспечение 16](#_Toc164074662)

[4.2.4. Требования к программному обеспечению системы 17](#_Toc164074663)

[4.2.5. Требования к программному обеспечению 17](#_Toc164074664)

[4.2.6. Требования к техническому обеспечению 17](#_Toc164074665)

[4.2.7. Требования к методическому обеспечению 18](#_Toc164074666)

[5. Проектирование системы 19](#_Toc164074667)

[5.1. SWOT – анализ 19](#_Toc164074668)

[5.2. Диаграмма вариантов использования 19](#_Toc164074669)

[5.3. Анализ прецедентов 20](#_Toc164074670)

[5.4. Диаграммы процессов BPMN 21](#_Toc164074671)

[5.5. Диаграмма компонентов 22](#_Toc164074672)

[5.6. Прототип UI 23](#_Toc164074673)

[6. Разработка информационной системы 27](#_Toc164074674)

[6.1. Backlog 27](#_Toc164074675)

[6.2. Функциональные блоки программы 28](#_Toc164074676)

[6.3. Пользовательский интерфейс 34](#_Toc164074677)

[6.4. Настройка системы 37](#_Toc164074678)

[Заключение 38](#_Toc164074679)

[Список источников 39](#_Toc164074680)

[Приложения 40](#_Toc164074681)

[Приложение 1. Код программы 40](#_Toc164074682)

# Словарь терминов

Таблица 1. Перечень сокращений

|  |  |  |
| --- | --- | --- |
| Термин | Определение | Синоним |
| OBS (Open Broadcaster Software) | Программа для проведения видеотрансляций и записи экрана |  |
| Fraps | Программа для записи экрана |  |
| Bandicam | Программа для записи экрана |  |
| FPS (frames per second) | Количество сменяемых кадров за одну секунду |  |
| Трей | Часть пользовательского интерфейса, в которой отображаются значки функций системы и программ, которые отсутствуют на рабочем столе, а также время и значок громкости |  |
| NAudio | Библиотека для работы с аудио |  |
| Accord | Библиотека для работы с видео |  |
| Definition Statements | Функционал |  |
| Донат | Добровольное пожертвование денежных средств | Пожертвование |
| Rec | Запись | Захват |
| Visual Studio | Линейка продуктов компании Microsoft, включающая интегрированную среду разработки (IDE) программного обеспечения и ряд других инструментов. |  |
| Фреймворк | Готовый набор инструментов, который помогает разработчику быстро создать продукт: сайт, приложение, интернет-магазин, CMS-систему. | Каркас, структура |
| WPF | Аналог WinForms, система для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем, графическая (презентационная) подсистема в составе .NET Framework, использующая язык XAML. |  |
| WinForms | Интерфейс программирования приложений, отвечающий за графический интерфейс пользователя и являющийся частью Microsoft .NET Framework. |  |
| Taskbar Notification | Библиотека для отображения приложения в панели задач. |  |
| High Definition Audio | Звуковой драйвер |  |
| Водяной знак | Специальный графический элемент, который добавляется поверх видео для защиты авторских прав или для идентификации источника контента. | Знак авторского права |
| Горячие клавиши | Комбинации клавиш на клавиатуре компьютера, которые выполняют определенные функции или команды в программе или операционной системе. | Сочетание клавиш |

# Введение

В современном информационном обществе, где визуальные и мультимедийные материалы играют все более важную роль, создание и обмен видеоконтентом становятся неотъемлемой частью нашей повседневной жизни. От обучения и демонстрации рабочих процессов до разработки обучающих видеоуроков и стриминга игрового процесса, запись экрана рабочего стола становится необходимой задачей. В данной работе представлено программное обеспечение, разработанное на платформе Visual Studio 2022 с использованием фреймворка C# WPF, которое призвано удовлетворить эту потребность.

Данное программное обеспечение спроектировано с учетом возможности его использования широким кругом пользователей, включая как опытных специалистов, так и тех, кто только начинает свой путь в мире записи экрана. Основным преимуществом является легкость использования и доступность для всех.

# Основная часть

## 1. Постановка задачи

Цель курсовой работы: заключается в разработке программы для записи экрана рабочего стола с целью создания обучающих видеоуроков, демонстрации работы программного обеспечения или записи игрового процесса.

Задачи ВКР:

1. Проектирование программы
2. Реализация записи экрана
3. Сравнительный анализ существующих программ
4. Тестирование производительности
5. Добавление функций сравнения с конкурентами

Цель информационной системы: обеспечение эффективного, доступного и простого программного обеспечения, предназначенное для записи экрана рабочего стола компьютера

Задачи информационной системы:

1. Захват и запись видео с экрана рабочего стола компьютера
2. Поддержка различных форматов записи видео (например, MP4, AVI)
3. Поддержка различных форматов записи аудио (например, MP3, WAV)
4. Регулировка качества записываемого видео
5. Возможность выбрать записываемый дисплей, если их несколько
6. Возможность разместить водяной знак
7. Автоматическое сохранение записанного видео после завершения процесса записи
8. Интуитивно понятный интерфейс для пользователей всех уровней навыков

## 2. Описание предметной области

### 2.1. Заинтересованные лица

Таблица 2. Заинтересованные лица

|  |  |  |  |
| --- | --- | --- | --- |
| **Заинтересованное лицо** | **Цель высокого уровня** | **Проблемы, возможности и замечания** | **Текущие решения** |
| Преподаватель | Записать урок | Быстрый старт записи, без настроек и понятный интерфейс | Настройки видео, аудио и качества уже выставлены по умолчанию. Интерфейс имеет очень мало кнопок и настроек, достаточно просто нажать кнопку записи |
| Игрок | Записать лучший момент из игры, без упадка FPS | Не сильная нагрузка на систему, быстрая запись игрового момента | MitREC почти не нагружает систему, а для быстрой записи есть горячие клавиши и доступ в трей |

### 2.2. Модель предметной области

Запись экрана, также известная как захват экрана, представляет собой технологию, позволяющую фиксировать и сохранять видео или изображения текущего содержимого экрана компьютера или мобильного устройства. Эта функциональность нашла широкое применение в различных областях, начиная от создания обучающих материалов до отслеживания действий пользователей на экране.

Процесс записи экрана обычно осуществляется с использованием специализированных программных решений или встроенных функций операционной системы. Пользователь может выбрать, записывать ли весь экран или только определенное окно приложения. Захваченный контент сохраняется в виде видеофайла или серии изображений, что обеспечивает удобство в последующем использовании и распространении.

Одним из основных сценариев использования записи экрана является создание обучающих роликов или демонстраций программного обеспечения. Это позволяет пользователям эффективно делиться информацией, делая уроки более доступными и понятными. Кроме того, запись экрана широко используется в сфере технической поддержки, где пользователи могут передавать информацию о проблемах, происходящих на их устройствах, сотрудникам службы поддержки.

Однако, как и в любой технологии, связанной с захватом информации, есть вопросы безопасности и конфиденциальности. Важно соблюдать соответствующие нормы и правила при использовании записи экрана, чтобы избежать нежелательного доступа к личной информации или конфиденциальным данным.

В современных условиях с развитием технологий и программного обеспечения для записи экрана, пользователи получают все больше возможностей для творческого и продуктивного использования этой функциональности. Улучшения в сфере алгоритмов сжатия видео, добавление функций редактирования и возможности стриминга делают запись экрана эффективным и универсальным инструментом в современной цифровой среде.

Таким образом, программное обеспечение для записи экрана рабочего стола представляет собой инструмент для простой и эффективной записи экрана. Пояснительная модель программы приведена на Рис. 1.

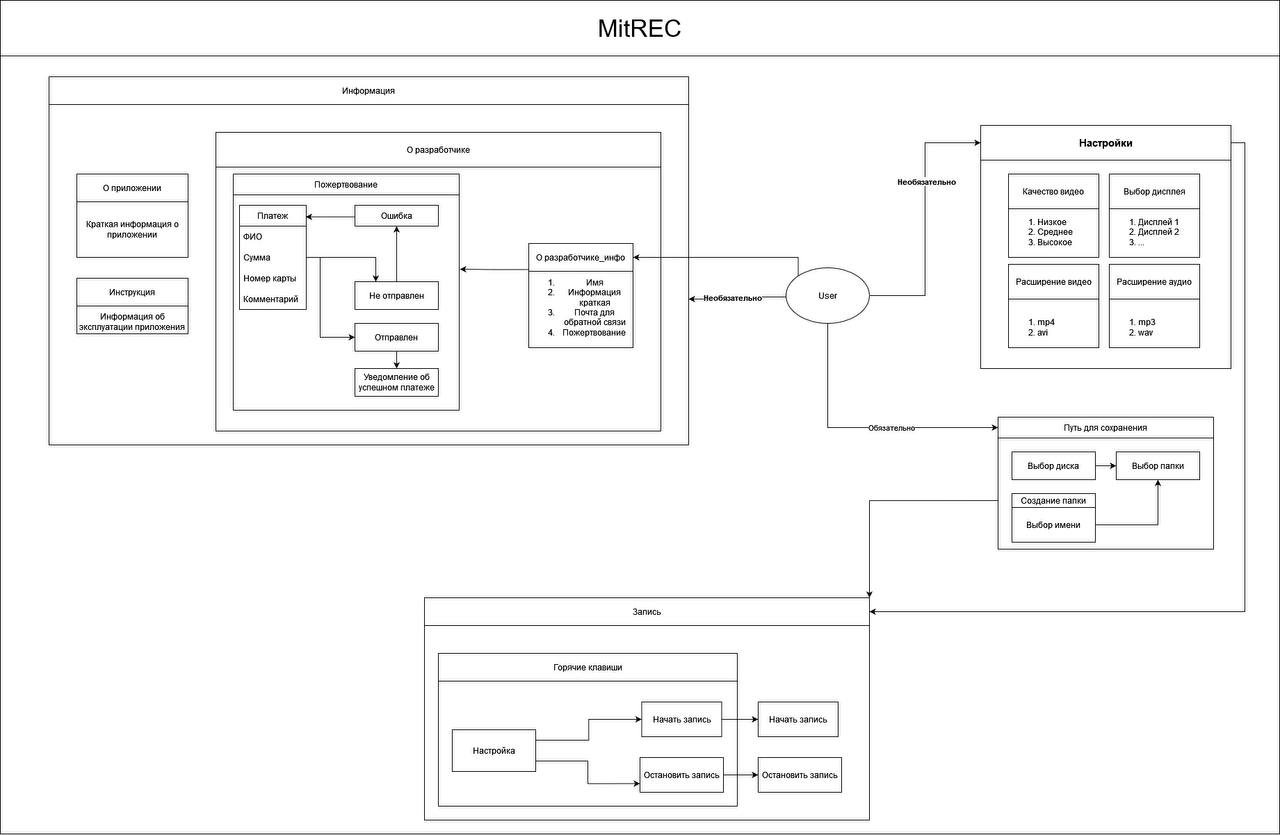


Рис. 1. Пояснительная модель.

На Рис. 1 показаны главные области программы - пояснительная модель принципа работы программы следующая: сначала пользователь выбирает путь сохранения, дальше по своему усмотрению может выбрать параметры для записи, например, качество видео, расширение видео и аудио, а также дисплей, если он не один. Из необязательных функций приложения могу перечислить следующее:

1. Информация о разработчике
2. Информация о приложении
3. Инструкция по использованию
4. Добровольное пожертвование

Концептуальная модель для информационной системы приведена на Рис.2. Она более четко отображает взаимодействия разных компонентов системы, описанных в рамках пояснительной модели.

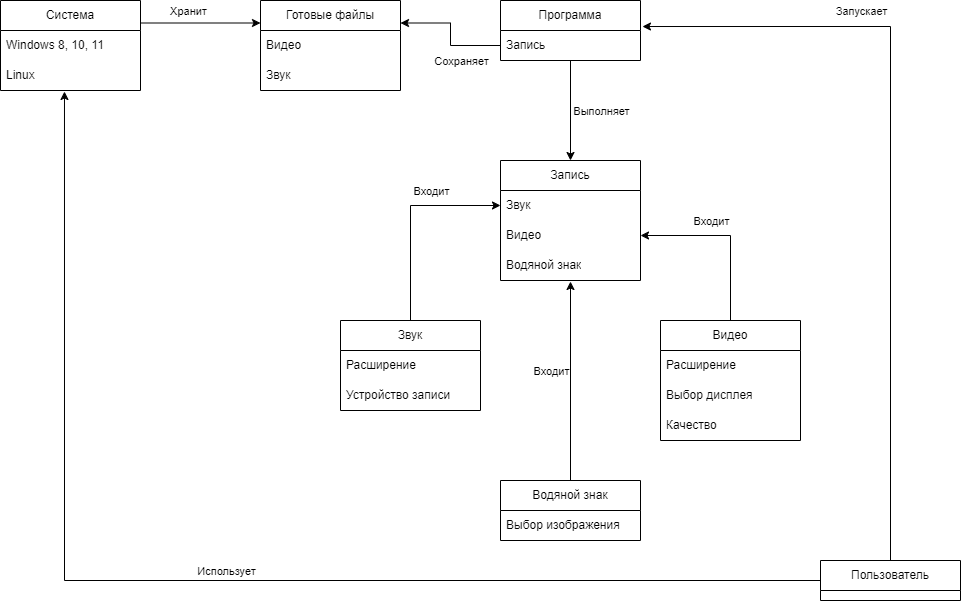


Рис.2. Концептуальная модель приложения

## 3. Обзор аналогов

Сейчас на рынке присутствует несколько программ для записи экрана, например, Bandicam, OBS, Fraps.

У **Bandicam** присутствует бесплатная версия продукта, но ограниченная, то есть присутствует водяной знак на записанном материале, также ограничение записи по времени (10 минут), к тому же, нет возможности купить подписку для полного доступа из-за санкций.

**OBS** не требует подписки, программа является бесплатной, **НО** содержит трудный и непонятный интерфейс. Чтобы начать записывать видео, нужно:

1. Создать сцену
2. Добавить объект
3. Настроить объект
4. Зайти в настройки
5. Настроить параметры записи
6. Указать путь, куда будет сохранен материал

Такая программа не всем подойдет, особенно, если человеку нужно срочно записать видео рабочего стола или он не очень хорошо разбирается в компьютере.

**Fraps** программа довольно старая, последнее обновление было ещё в 2013 году. Основным недостатком этой программы был большой вес записанного материала. Например, вес 10-ти минутного ролика мог достигать от 7-10 Гб, что по сравнению с Bandicam или OBS слишком много (вес у ролика из Bandicam максимум 2-3 Гб).

Дополнительная информация представлена в таблице 3.

Таблица 3. Обзор аналогов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Критерии** | **OBS** | **Bandicam** | **Fraps** | **MitREC** |
| Стоимость | Бесплатно, открытый код | Платно (нельзя купить) | Бесплатно | Бесплатно, открытый код |
| Объем готового файла, мб за 5 сек. | 3.7 Мб (5 секунд) | 3 Мб (5 секунд) | 102 Мб (5 секунд) | 2.90 Мб (5 секунд) |
| Нагрузка на систему | Низкая | Низкая | Высокая | Низкая |
| Обновления | Постоянные обновления | Постоянные обновления | Больше не обновляется | Постоянные обновления |
| ОС | Windows 11/10, macOS 11, Linux | Windows 11/10/8/7 (64-bit) | Windows XP, 2003, Vista и Windows 7 | Windows 11/10  (вскоре Linux) |

## 4. Требования к системе

### 4.1. Функциональные требования

Программа должна включать в себя следующие возможности:

1. Возможность записи рабочего стола
2. Возможность выбирать качество видео (низкое, среднее, высокое)
3. Возможность выбрать расширение для видеофайла
4. Возможность выбрать расширение для аудиофайла
5. Возможность добавить водяной знак
6. Доступ из трея
7. Начать или остановить запись с помощью горячих клавиш
8. Собственный каталог для выбора пути сохранения (возможность создать папку)

### Требования по видам обеспечения

### 4.2.1 Требования к информационному обеспечению системы

**Хранение данных:**

* 1. Система должна обеспечивать хранение всех записанных аудио и видеофайлов.
  2. Данные должны храниться в формате, обеспечивающем целостность и доступность при необходимости последующей обработки или анализа.

### 4.2.2. Требования к алгоритмическому обеспечению

Требования к алгоритмическому обеспечению включают следующее:

1. Эффективность алгоритмов записи и обработки аудио- и видеоданных:
   1. Алгоритмы записи аудио и видео должны обеспечивать высокую производительность и минимальное потребление ресурсов системы.
   2. При обработке аудио- и видеоданных должны использоваться оптимизированные алгоритмы, способные работать с большими объемами данных без потери качества и производительности.
2. Алгоритмы обработки аудио- и видеоданных должны быть стабильными и надежными, минимизируя риск возникновения ошибок и сбоев во время работы программы.
3. Адаптивность алгоритмов под различные условия записи и обработки:
   1. Алгоритмы должны быть адаптивными и способными к автоматической настройке параметров в зависимости от условий записи и характеристик аудио- и видеоданных.
4. Совместимость с форматами и стандартами данных:
   1. Алгоритмы должны быть совместимы с широко используемыми форматами аудио и видео файлов, такими как WAV, MP3, MP4, AVI и другими.
   2. При обработке данных необходимо соблюдать международные стандарты кодирования и сжатия данных для обеспечения совместимости с другими программами и устройствами.

### 4.2.3. Требования к лингвистическому обеспечение

1. Разработка программного обеспечения:

* Использование языка программирования C# WPF.

2. Описание предметной области:

* Использование языка моделирования UML.
* Использование языка моделирования BPMN.
* Использование диаграммы компонентов.
* Использование Use Case диаграммы.

### 4.2.4. Требования к программному обеспечению системы

1. **Производительность**: приложение должно обеспечивать высокую производительность при записи, обработке и воспроизведении аудио- и видеоданных, минимизируя задержки и обеспечивая плавное взаимодействие с пользователем.
2. **Интерфейс пользователя**: пользовательский интерфейс должен быть интуитивно понятным и удобным в использовании, обеспечивая простоту настройки параметров записи и обработки аудио- и видеоданных.

### 4.2.5. Требования к программному обеспечению

1. **Windows**:

Программное обеспечение предназначено для работы на операционных системах семейства Windows (Windows 10, Windows 8, Windows 7), обеспечивая совместимость с основными версиями ОС и их обновлениями.

1. **Linux** (планируемые дальнейшие шаги):

В дальнейших планах предусмотрена поддержка операционных систем на базе ядра Linux, таких как Ubuntu, CentOS и других дистрибутивов. Это позволит расширить охват аудитории пользователей и обеспечить работу программного обеспечения на различных платформах.

### 4.2.6. Требования к техническому обеспечению

**Минимальные требования**:

* Процессор: Intel Core i3 или аналогичный.
* Оперативная память: 4 ГБ.
* Дисковое пространство: 100 МБ на жестком диске.
* Видеокарта: совместимая с DirectX 10.
* Разрешение экрана: 1280x768 пикселей.

**Рекомендуемые требования**:

* Процессор: Intel Core i5 или аналогичный.
* Оперативная память: 8 ГБ или более.
* Дисковое пространство: 500 МБ на жестком диске.
* Видеокарта: совместимая с DirectX 11 или более новой.
* Разрешение экрана: 1920x1080 пикселей.

### 4.2.7. Требования к методическому обеспечению

**Документация**:

* Полная и понятная документация по установке, настройке и использованию программного обеспечения.
* Руководства пользователя с пошаговыми инструкциями.

## 5. Проектирование системы

### 5.1. SWOT – анализ

Таблица 4. SWOT - анализ

|  |  |  |
| --- | --- | --- |
|  | **Возможности** | **Угрозы** |
| **Внешние** | * Рост рынка программ для захвата экрана рабочего стола | * Конкуренция на рынке * Технические проблемы, сбои |
| **Внутренние** | * Улучшение программы с помощью добавления новых функций и возможностей * Улучшение алгоритма записи видео и аудио | * Финансовые ограничения * Ограниченный человеческий ресурс |

### 5.2. Диаграмма вариантов использования

Диаграмма вариантов использования (Use Case Diagram) является одним из видов диаграмм UML (Unified Modeling Language.

Use Case диаграмма для рассматриваемой предметной области приведена на Рис. 3.

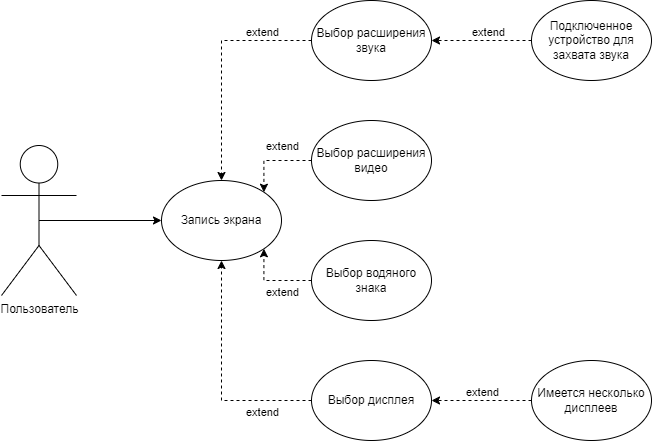


Рис. 4. Use case

На модели представлен главный «актор» - внешний участник или сущность, взаимодействующий с системой – Пользователь (преподаватель, игрок, собеседник и т.д.)

### 5.3. Анализ прецедентов

**Прецедент №1:** Запуск записи экрана

**Участники:**

1. Пользователь

**Описание:** Пользователь запускает процесс записи аудио и видео с выбранного экрана для создания видеофайла с аудиодорожкой.

**Предусловие:** Пользователь успешно запустил приложение и выбрал параметры записи.

**Основной сценарий:**

1. Пользователь открывает приложение для записи аудио и видео.
2. Пользователь выбирает экран, с которого будет осуществляться запись.
3. Пользователь выбирает параметры аудио и видео записи (качество, формат и т. д.).
4. Пользователь запускает запись.

**Постусловия:** Система начинает запись аудио и видео с выбранного экрана с заданными параметрами.

**Альтернативные сценарии:** Если выбранный экран не доступен или происходит ошибка при запуске записи, система выводит сообщение об ошибке и предлагает пользователю повторить попытку или выбрать другой экран.

**Исключения:** Если не удалось инициализировать аудио- или видеозапись, система выводит сообщение об ошибке и прекращает процесс записи.

**Прецедент №2:** Остановка записи экрана

**Участники:**

1. Пользователь

**Описание:** Пользователь останавливает процесс записи аудио и видео и сохраняет полученный файл.

**Предусловие:** Процесс записи аудио и видео уже запущен.

**Основной сценарий:**

1. Пользователь нажимает кнопку "Стоп" или выбирает соответствующую опцию в пользовательском интерфейсе приложения.
2. Система завершает процесс записи аудио и видео.
3. Пользователь выбирает место для сохранения файла записи и подтверждает сохранение.

**Постусловия:** Система завершает процесс записи аудио и видео и сохраняет полученный файл по указанному пользователем пути.

**Альтернативные сценарии:** Если пользователь решает не сохранять файл, система прекращает запись и не сохраняет результаты.

**Исключения:** Если происходит ошибка при остановке записи или сохранении файла, система выводит сообщение об ошибке и предлагает пользователю повторить попытку или обратиться за помощью.

### 5.4. Диаграммы процессов BPMN

BPMN (Business Process Model and Notation) представляет собой стандартную нотацию для моделирования бизнес-процессов.

BPMN диаграмма для программы записи экрана рабочего стола была составлена с учетом Use Case диаграммы, она представлена на Рис. 5.

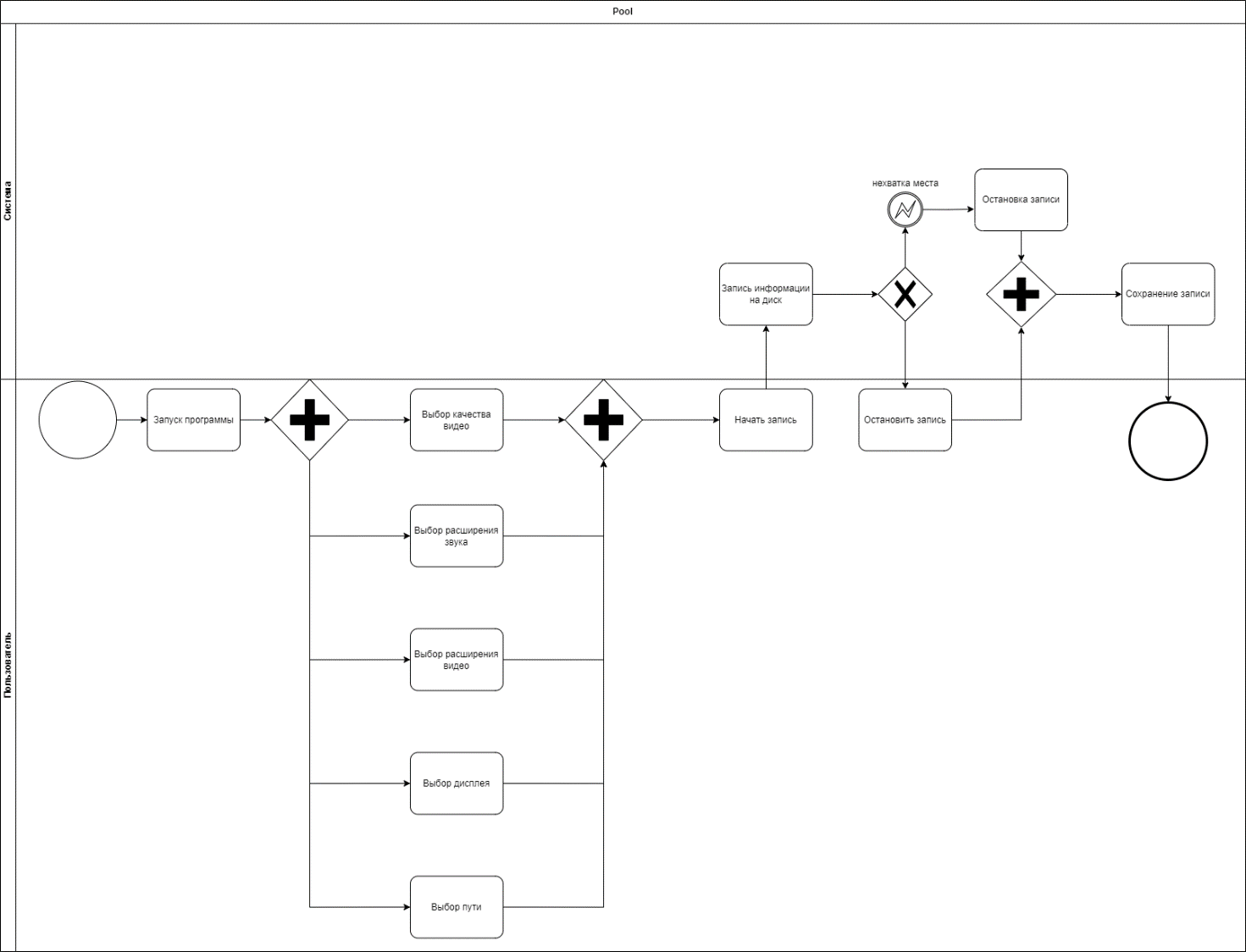


Рис.5. BPMN диаграмма

### 5.5. Диаграмма компонентов

На Рис. 6 приведена диаграмма компонентов.

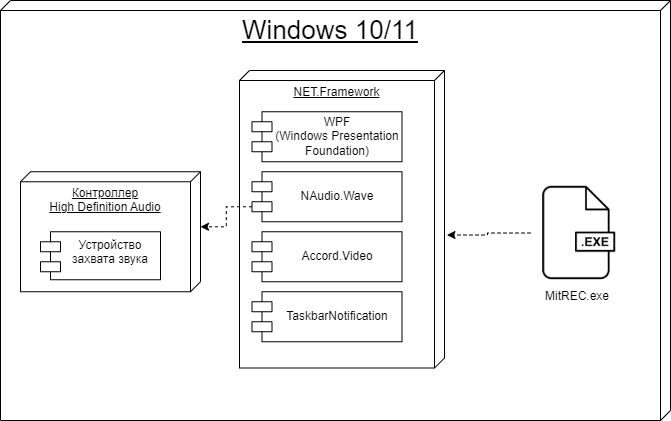


Рис. 6 Диаграмма компонентов

Диаграмма компонентов состоит из Desktop приложения на базе Windows.

На данной модели представлены требования к ПО:

1. **MitRec.exe** – исполняемый файл, при открытии которого запускается приложение.
2. **NAudio.Wave** – библиотека для захвата звука и вывода его в итоговый звуковой файл.
3. **Accord.Video** – библиотека для захвата экрана с последующим выводом в итоговый видеофайл.
4. **Taskbar Notification** – библиотека для отображения приложения в панели задач.
5. **WPF** – framework для создания GUI программы.
6. **High Definition Audio** – звуковой драйвер

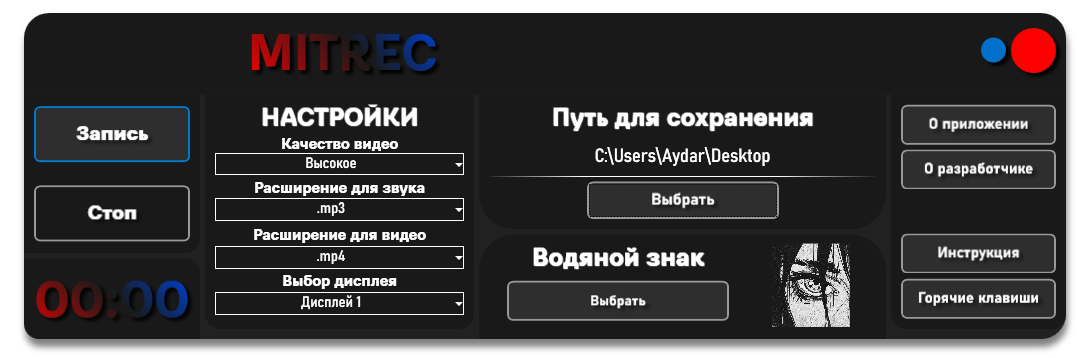
### 5.6. Прототип UI

Данное программное обеспечение спроектировано таким образом, чтобы оно могло быть доступно и использовано широким кругом пользователей, включая как опытных специалистов, так и тех, кто только начинает свой путь в мире записи экрана. При этом основным преимуществом является легкость использования и доступность для всех.

Открыв приложение, пользователь видит основные категории, такие как:

1. Кнопки для начала записи/остановки;
2. Настройки медиа (качество видео, расширение для звука, расширение для видео, выбор дисплея);
3. Выбор пути сохранения;
4. Выбор водяного знака;
5. Информация (О приложении, о разработчике, инструкция по эксплуатации, горячие клавиши).

Наглядно программное обеспечение представлено на рисунке 7.

  
Рис. 7. Оформление программного обеспечения

**Обязательным** действием, которое должен выполнить пользователь, является выбор пути для сохранения итогового файла, выглядит он следующим образом:

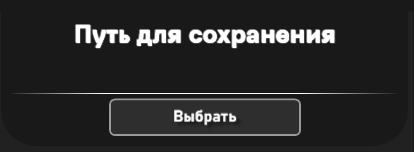
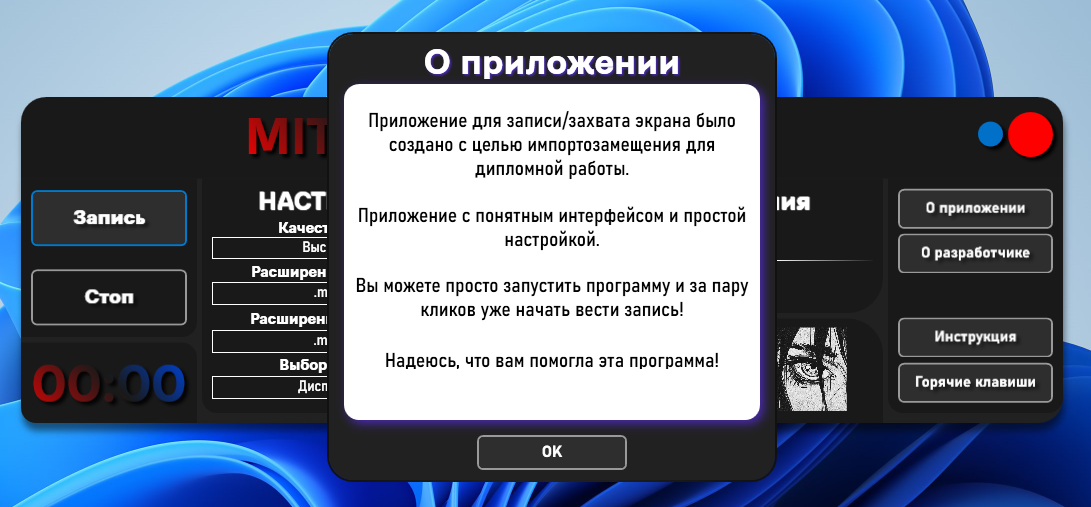
  
Рис. 8. Путь для сохранения

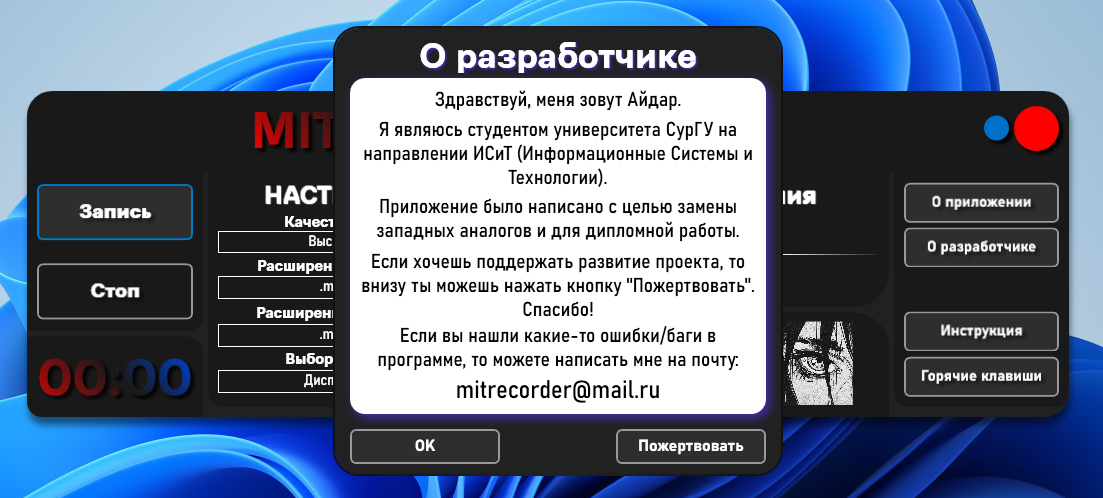
  
Рис. 9. Каталог выбора пути для сохранения

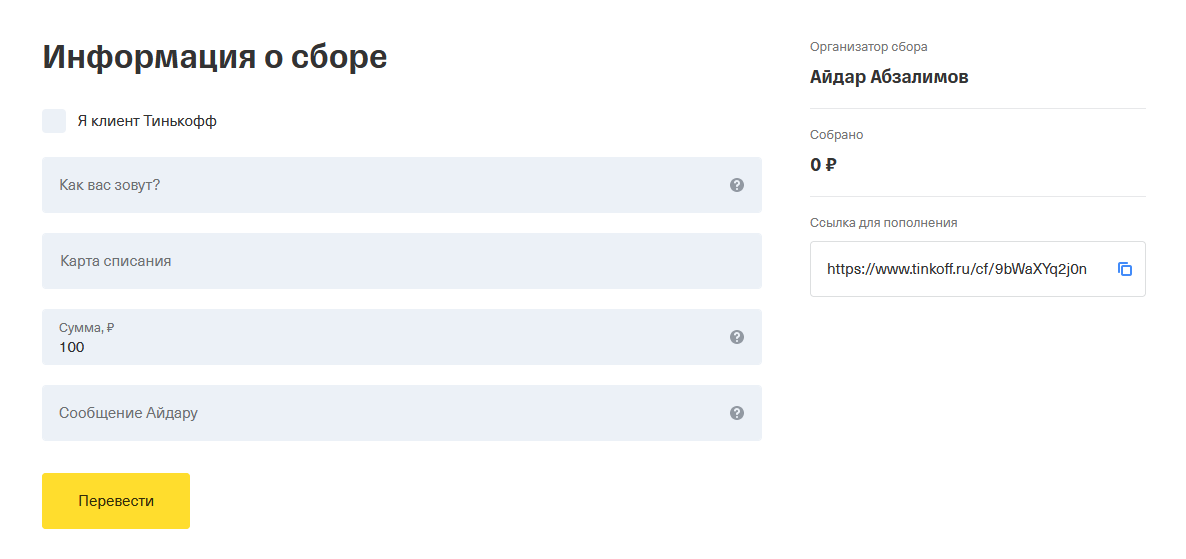
**Необязательные действия:**

1. Кнопка “О приложении” представляет собой всплывающее окно с информацией о приложении:

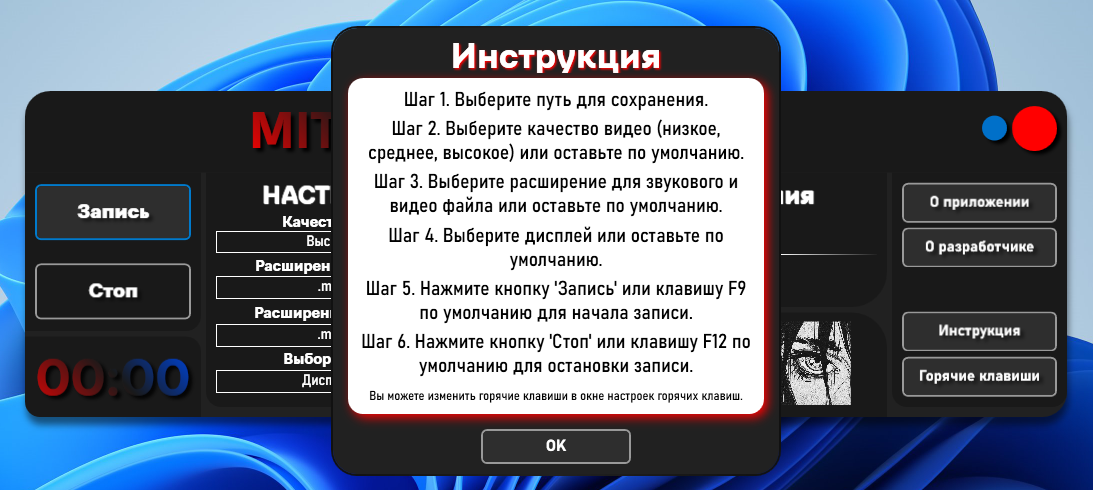
  
Рис. 10. О приложении

1. Кнопка “О разработчике” представляет собой всплывающее окно с информацией о разработчике, почтой для обратной связи и с дополнительной кнопкой для добровольного пожертвования:

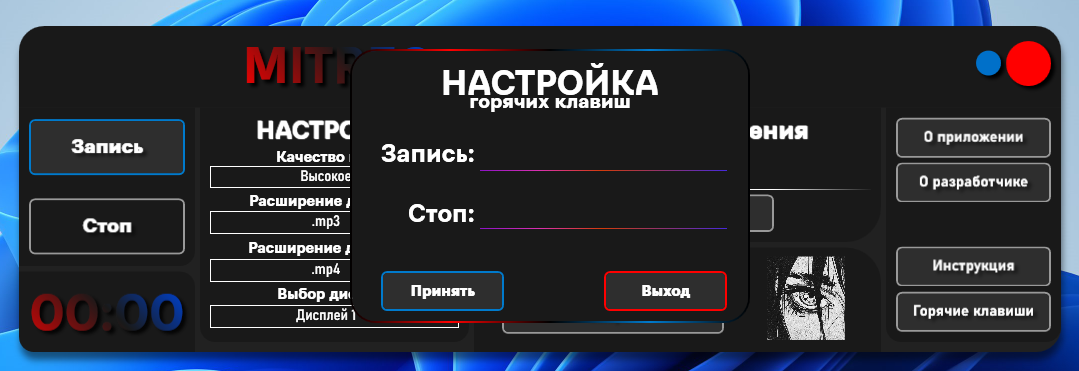
  
Рис. 11. О разработчике

  
Рис. 12. Страница для пожертвования

1. Кнопка “Инструкция” представляет собой всплывающее окно с информацией по эксплуатации программы:

  
Рис. 13. Инструкция

1. Кнопка “Горячие клавиши” представляет собой всплывающее окно, где у пользователя появляется возможность редактировать горячие клавиши для воспроизведения и остановки записи:

  
Рис. 14. Горячие клавиши

1. Категория “Настройки” служит для выбора пользователем предпочитаемых параметров для записи видео и звука:

  
Рис. 15. Настройки

## 6. Разработка информационной системы

### 6.1. Backlog

Список задач по разработке информационной системы может выглядеть следующим образом:

1. Разработка функциональных требований к программе.
   1. Описание функциональности системы, включая возможности, которые должны быть реализованы.
   2. Определение требований к вводу данных пользователем и ожидаемым результатам.
2. Разработка дизайна и реализация интерфейса.
   1. Создание прототипа пользовательского интерфейса (UI), отображающего основные элементы и функции системы.
   2. Разработка дизайна пользовательского интерфейса с учетом принципов удобства использования и эстетики.
   3. Имплементация UI с использованием соответствующих технологий и инструментов, таких как WPF (Windows Presentation Foundation) для ОС Windows и GTK (GIMP Toolkit) для Linux.
3. Реализация функционала программы под ОС Windows и Linux.
   1. Написание кода, реализующего функции, определенные в функциональных требованиях, с учетом особенностей каждой операционной системы.
   2. Тестирование функционала на обеих платформах для обеспечения совместимости и корректной работы на ОС Windows и Linux.

### 6.2. Функциональные блоки программы

Обратимся к диаграмме компонентов, которая явно показывает весь функционал кода и все взаимозависимости в нем. Обратимся к Рис. 6.

Рассмотрим файл MainWindow.xaml.cs, который содержит в себе большинство функций и действий в приложении, в том числе действия в трее, список дисплеев, горячие клавиши, а также запись видео и аудио

public MainWindow()

{

taskbarIcon = (TaskbarIcon)FindResource("MyNotifyIcon");

taskbarIcon.TrayMouseDoubleClick += (s, e) => ShowMainWindow();

//Список для хранения информации о дисплеях

List<DisplayInfo> displayInfoList = new List<DisplayInfo>();

//Получение списка доступных дисплеев

foreach (Screen screen in Screen.AllScreens)

{

displayInfoList.Add(new DisplayInfo

{

DeviceName = screen.DeviceName,

DisplayName = "Дисплей " + (displayInfoList.Count + 1)

});

}

//Источник данных для ComboBox

DisplayBox.ItemsSource = displayInfoList;

DisplayBox.DisplayMemberPath = "DisplayName";

DisplayBox.SelectedValuePath = "DeviceName";

}

// Горячие клавиши

private void SetRecordingHotkey()

{

// Загрузка горячей клавиши из настроек

string hotkeyText = Properties.Settings.Default.RecordingHotkey;

// Парсинг горячей клавиши

KeyGestureConverter converter = new KeyGestureConverter();

recordingHotkey = converter.ConvertFromString(hotkeyText) as KeyGesture;

if (recordingHotkey == null)

{

// Если горячая клавиша не удалось загрузить из настроек, используйте значение по умолчанию

recordingHotkey = new KeyGesture(Key.F9);

}

// Зарегистрируем обработчик горячей клавиши

CommandBinding binding = new CommandBinding(ApplicationCommands.New);

binding.Executed += RecordingHotkey\_Executed;

CommandBindings.Add(binding);

// Привязываем горячую клавишу к окну

InputBinding inputBinding = new InputBinding(ApplicationCommands.New, recordingHotkey);

InputBindings.Add(inputBinding);

}

private void SetStopHotkey()

{

string hotkeyText = Properties.Settings.Default.StopHotkey;

stopHotkey = new KeyGesture(Key.F12);

CommandBinding binding = new CommandBinding(ApplicationCommands.Close);

binding.Executed += StopHotkey\_Executed;

CommandBindings.Add(binding);

InputBinding inputBinding = new InputBinding(ApplicationCommands.Close, stopHotkey);

InputBindings.Add(inputBinding);

}

private void RecordingHotkey\_Executed(object sender, ExecutedRoutedEventArgs e)

{

// Обработка нажатия горячей клавиши для начала записи

RecButton\_Click(sender, e);

}

private void StopHotkey\_Executed(object sender, ExecutedRoutedEventArgs e)

{

// Обработка нажатия горячей клавиши для кнопки "Стоп"

StopButton\_Click(sender, e);

}

// Запись видео и водяной знак

private void RecButton\_Click(object sender, RoutedEventArgs e)

{

RecButton.IsEnabled = false;

StopButton.IsEnabled = true;

if (waveIn == null)

{

waveIn = new WaveInEvent();

waveIn.DataAvailable += WaveIn\_DataAvailable;

// Получить выбранный дисплей из ComboBox

string selectedDeviceName = (DisplayBox.SelectedValue as string);

// Проверка на то, что пользователь выбрал дисплей

if (string.IsNullOrEmpty(selectedDeviceName))

{

System.Windows.MessageBox.Show("Выберите дисплей для записи.");

return;

}

savePath = SaveRoad.Text;

if (!string.IsNullOrEmpty(savePath))

{

if (audioBox.SelectedItem != null)

{

string selectedAudio = (audioBox.SelectedItem as ComboBoxItem).Content.ToString();

switch (selectedAudio)

{

case ".wav":

string fileNameAudioWav = "audio\_" + DateTime.Now.ToString("yyyyMMdd\_HHmmss") + selectedAudio;

writer = new WaveFileWriter(Path.Combine(savePath, fileNameAudioWav), waveIn.WaveFormat);

break;

case ".mp3":

string fileNameAudioMp3 = "audio\_" + DateTime.Now.ToString("yyyyMMdd\_HHmmss") + selectedAudio;

writer = new WaveFileWriter(Path.Combine(savePath, fileNameAudioMp3), waveIn.WaveFormat);

break;

}

}

// Начать запись

waveIn.StartRecording();

// Начать отслеживание времени записи

recordingTime = TimeSpan.Zero;

timer = new Timer

{

Interval = TimeSpan.FromSeconds(1)

};

timer.Tick += Timer\_Tick;

timer.Start();

int width = 1920; // Ширина видео

int height = 1080; // Высота видео

int fps = 30; // Кадры в секунду (FPS)

if (videoQualityBox.SelectedItem != null)

{

string selectedQuality = (videoQualityBox.SelectedItem as ComboBoxItem).Content.ToString();

switch (selectedQuality)

{

case "Низкое":

bitrate = 2000000;

break;

case "Среднее":

bitrate = 8000000;

break;

case "Высокое":

bitrate = 20000000;

break;

}

}

// Определение экрана для записи на основе выбранного имени

Screen selectedScreen = Screen.AllScreens.FirstOrDefault(s => s.DeviceName == selectedDeviceName);

if (selectedScreen != null)

{

// Запускаем захват рабочего стола выбранного дисплея

screenCapture = new ScreenCaptureStream(selectedScreen.Bounds, fps);

screenCapture.NewFrame += ScreenCapture\_NewFrame;

screenCapture.Start();

// Начинаем запись видео

videoWriter = new VideoFileWriter();

if (videoBox.SelectedItem != null)

{

string selectedExtension = (videoBox.SelectedItem as ComboBoxItem).Content.ToString();

switch (selectedExtension)

{

case ".mp4":

videoExtension = ".mp4";

break;

case ".avi":

videoExtension = ".avi";

break;

}

string fileName = "video\_" + DateTime.Now.ToString("yyyyMMdd\_HHmmss") + videoExtension;

videoWriter.Open(Path.Combine(savePath, fileName), width, height, fps, VideoCodec.MPEG4, bitrate);

}

isRecording = true;

}

else

{

System.Windows.MessageBox.Show("Выбранный дисплей не найден.");

}

}

else

{

System.Windows.MessageBox.Show("Выберите путь для сохранения!");

}

}

}

private void ApplyWatermark(Bitmap frame)

{

if (!string.IsNullOrEmpty(watermarkImagePath))

{

// Загружаем изображение водяного знака

Bitmap watermark = new Bitmap(watermarkImagePath);

watermark = new Bitmap(watermark, new System.Drawing.Size(75, 75));

// Размещаем водяной знак на видеокадре

using (Graphics g = Graphics.FromImage(frame))

{

g.DrawImage(watermark, new System.Drawing.Point(frame.Width - watermark.Width - 10, frame.Height - watermark.Height - 10));

}

}

}

private void ScreenCapture\_NewFrame(object sender, NewFrameEventArgs eventArgs)

{

if (IsRecordingValid())

{

Bitmap frame = eventArgs.Frame;

ApplyWatermark(frame);

videoWriter.WriteVideoFrame(frame);

lastFrameTime = DateTime.Now;

}

}

private void WaveIn\_DataAvailable(object sender, WaveInEventArgs e)

{

writer?.Write(e.Buffer, 0, e.BytesRecorded);

}

private void StopButton\_Click(object sender, RoutedEventArgs e)

{

RecButton.IsEnabled = true;

StopButton.IsEnabled = false;

if (waveIn != null)

{

waveIn.StopRecording();

waveIn.Dispose();

waveIn = null;

if (writer != null)

{

writer.Close();

writer.Dispose();

writer = null;

}

if (timer != null)

{

timer.Stop();

timer.Tick -= Timer\_Tick;

timer = null;

}

if (isRecording)

{

isRecording = false;

screenCapture.SignalToStop();

screenCapture.WaitForStop();

videoWriter.Close();

videoWriter.Dispose();

}

}

}

// Параметр для смены дисплея

private void DisplayBox\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

int fps = 30;

if (isRecording)

{

screenCapture.SignalToStop();

screenCapture.WaitForStop();

string newDeviceName = (DisplayBox.SelectedValue as string);

Screen newScreen = Screen.AllScreens.FirstOrDefault(s => s.DeviceName == newDeviceName);

if (newScreen != null)

{

screenCapture = new ScreenCaptureStream(newScreen.Bounds, fps);

screenCapture.NewFrame += ScreenCapture\_NewFrame;

screenCapture.Start();

}

else

{

System.Windows.MessageBox.Show("Выбранный дисплей не найден.");

}

}

}

В файле DisplayInfo.cs хранится класс для информации связанной со списком дисплеев

public class DisplayInfo

{

public string DeviceName { get; set; }

public string DisplayName { get; set; }

}

В файле HotKey.cs хранится класс для реализации действий горячих клавиш

internal class HotKey

{

private static Dictionary<int, Action> RegisteredActions = new Dictionary<int, Action>();

private const int WM\_HOTKEY = 0x0312;

[DllImport("user32.dll")]

private static extern bool RegisterHotKey(IntPtr hWnd, int id, int fsModifiers, int vk);

[DllImport("user32.dll")]

private static extern bool UnregisterHotKey(IntPtr hWnd, int id);

private static int currentId;

public static void Register(Key key, KeyModifier keyModifier, Action action)

{

int id = currentId++;

RegisterHotKey(IntPtr.Zero, id, (int)keyModifier, KeyInterop.VirtualKeyFromKey(key));

RegisteredActions[id] = action; // Сохраните делегат в словаре

}

public static void Unregister()

{

foreach (var id in KeyHandler.RegisteredActions.Keys)

{

UnregisterHotKey(IntPtr.Zero, id);

}

}

public class KeyHandler : MainWindow

{

private HwndSource source;

public static Dictionary<int, Action> RegisteredActions = new Dictionary<int, Action>();

public KeyHandler()

{

try

{

source = HwndSource.FromHwnd(new WindowInteropHelper(this).Handle);

source.AddHook(HwndHook);

}

catch (Exception ex)

{

Console.WriteLine("Error initializing KeyHandler: " + ex.Message);

}

}

private IntPtr HwndHook(IntPtr hwnd, int msg, IntPtr wParam, IntPtr lParam, ref bool handled)

{

if (msg == WM\_HOTKEY)

{

int id = wParam.ToInt32();

if (RegisteredActions.TryGetValue(id, out var action))

{

action.Invoke();

}

handled = true;

}

return IntPtr.Zero;

}

}

}

public enum KeyModifier

{

None = 0,

Alt = 1,

Control = 2,

Shift = 4,

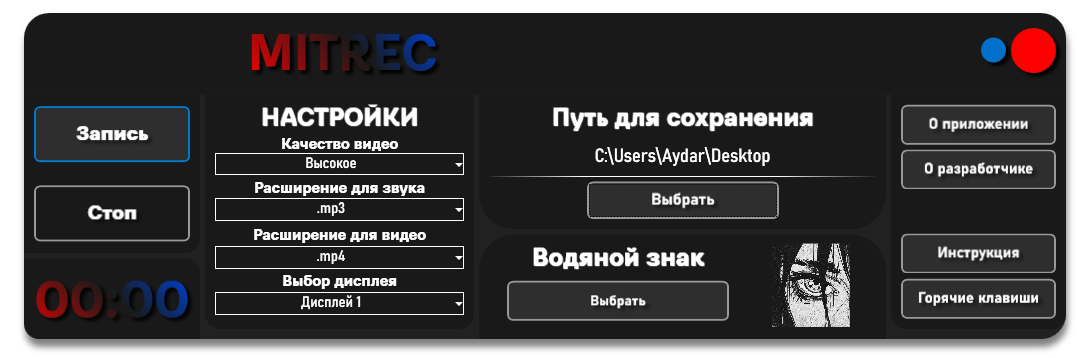
Win = 8

}

### 6.3. Пользовательский интерфейс

В ходе реализации проекта были предприняты некоторые решения, в связи с которыми были внесены изменения в итоговой вариант дизайна пользовательского интерфейса.

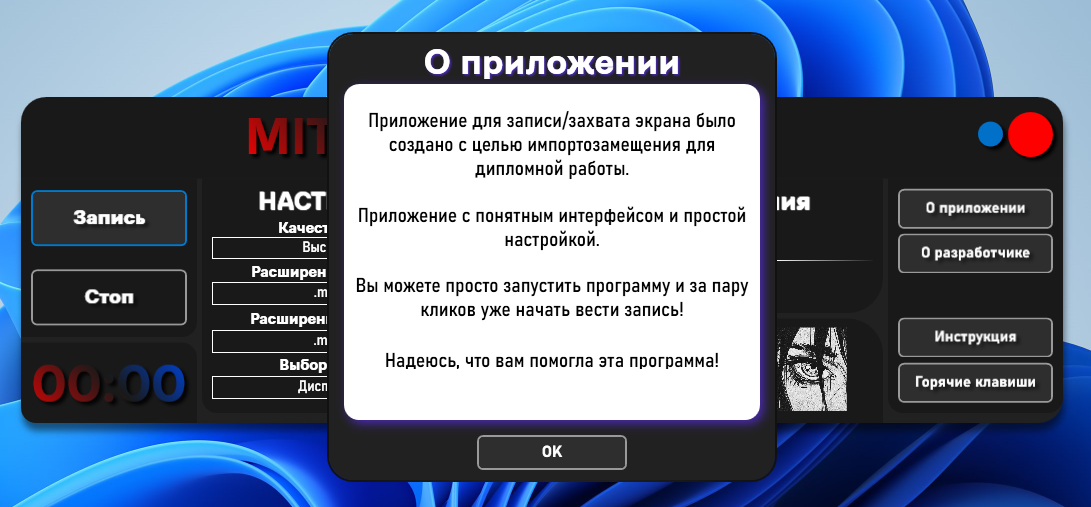
Дизайн окна приложения – открывающегося по умолчанию при запуске – приведен на Рис. 16.

  
Рис. 16. Главный экран

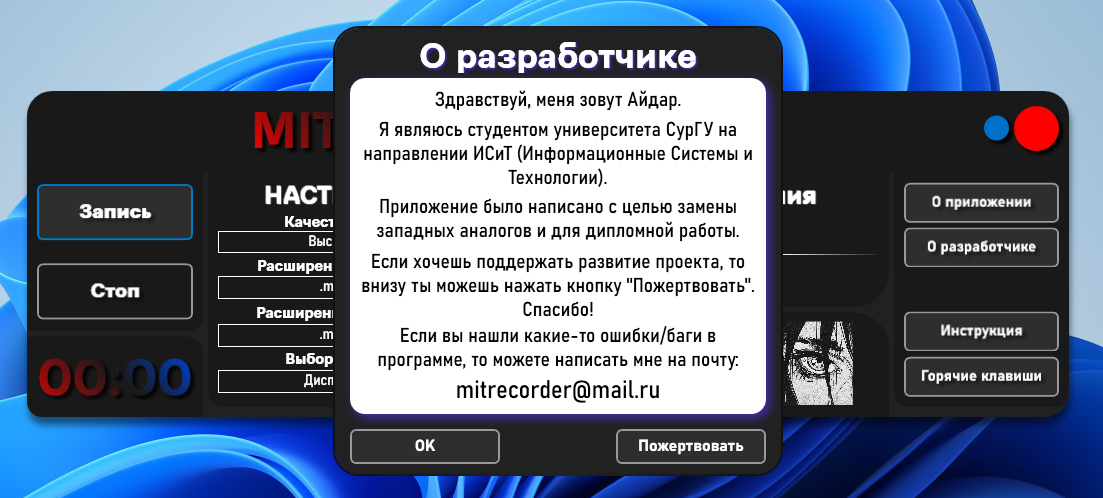
Дизайн окна каталога, где пользователь выбирает путь для сохранения итоговых файлов приведен на Рис. 17.

  
Рис. 17. Каталог

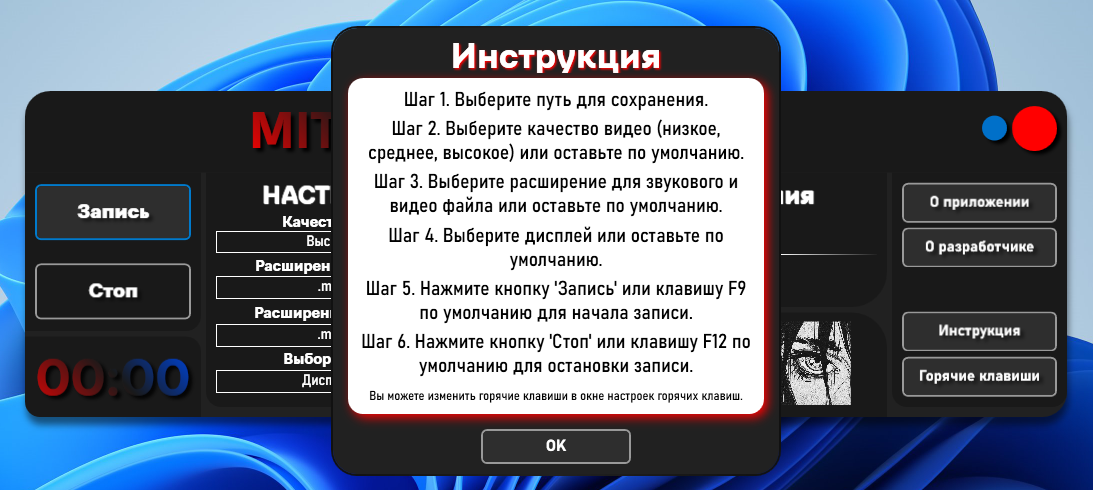
Дизайн окна “О приложении” приведен на Рис. 18.

  
Рис. 18. О приложении

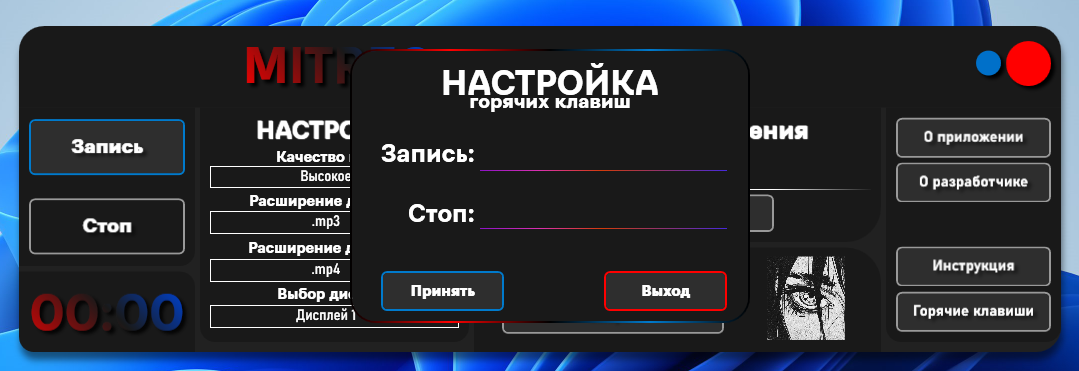
Дизайн окна “О разработчике” приведен на Рис. 19.

  
Рис. 19. О разработчике

Дизайн окна “Инструкция” приведен на Рис. 20.

  
Рис. 20. Инструкция

Дизайн окна “Горячие клавиши” приведен на Рис. 21.

  
Рис. 21. Горячие клавиши

На главном экране расположен блок категории “Настройки”, который служит для выбора пользователем предпочитаемых параметров для записи видео и звука – приведен на Рис. 22.

  
Рис. 22. Настройки

### 6.4. Настройка системы

Для обеспечения удобства пользователей и оптимальной работы приложения предусмотрена возможность настройки основных параметров системы. В разделе "Настройки" пользователь может выбрать предпочтительные параметры для записи видео и звука. Эти параметры могут включать в себя:

1. **Качество видео**:
   * Пользователь может выбрать желаемое качество видеозаписи: низкое, среднее или высокое. Это влияет на битрейт и разрешение видеофайлов.
2. **Качество звука**:
   * Предоставляется возможность выбора качества звукозаписи: стандартное или высокое. Это может повлиять на частоту дискретизации и битрейт аудиофайлов.
3. **Путь сохранения файлов**:
   * Пользователь может указать предпочтительный путь для сохранения итоговых файлов видео и аудио.
4. **Расширение файлов**:
   * Возможность выбора формата файлов видео и аудио для сохранения: MP4, AVI, WAV, MP3.

Пользовательский интерфейс раздела "Настройки" представлен в виде блока категории на главном экране, что обеспечивает простоту доступа к настройкам и интуитивно понятное управление ими.

# Заключение

В ходе выполнения курсового проекта в течение двух семестров по дисциплине "Методы и средства проектирования информационных систем" была разработана проектная модель, охватывающая все необходимые аспекты проекта, подлежащие реализации в течение всего процесса разработки программного обеспечения. Эта модель включает в себя следующие ключевые аспекты:

1. **Анализ требований**: В начальной стадии был проведен анализ требований, включающий сбор и анализ потребностей пользователей, определение функциональных и нефункциональных требований, а также выработку основных целей проекта.
2. **Проектирование системы**: На основе выявленных требований была разработана архитектура системы, включая выбор подходящих технологий, определение структуры базы данных и проектирование пользовательского интерфейса.
3. **Разработка и тестирование**: На этапе разработки были созданы и протестированы основные компоненты системы, включая модули записи аудио и видео, взаимодействие с пользовательским интерфейсом.

# Список источников

1. Язык C# и платформа .NET [Электронный ресурс] URL:  
   <https://metanit.com/sharp> (Дата обращения 10.10.2023)
2. Visual Studio: IDE и редактор кода для разработчиков [Электронный ресурс] URL:  
   <https://visualstudio.microsoft.com/ru/> (Дата обращения 10.10.2023)
3. Screen Recorder – Bandicam [Электронный ресурс] URL:  
   <https://www.bandicam.com/w/ru/> (Дата обращения 12.10.2023)
4. Fraps [Электронный ресурс] URL:  
   <https://fraps.com> (Дата обращения 12.10.2023)
5. Open Broadcaster Software [Электронный ресурс] URL:  
   <https://obsproject.com> (Дата обращения 12.10.2023)
6. NAudio [Электронный ресурс] URL:  
   <https://github.com/naudio/NAudio?ysclid=lq4zzr1vci796471987> (Дата обращения 13.10.2023)
7. Accord.Video.FFMPEG [Электронный ресурс] URL:  
   <https://github.com/accord-net/framework/issues/428?ysclid=lq5031gx51887443662> (Дата обращения 13.10.2023)
8. AudioCodec Enumeration [Электронный ресурс] URL:  
   <http://accord-framework.net/docs/html/T_Accord_Video_FFMPEG_AudioCodec.htm> (Дата обращения 13.10.2023)

# Приложения

## Приложение 1. Код программы

public MainWindow()

{

taskbarIcon = (TaskbarIcon)FindResource("MyNotifyIcon");

taskbarIcon.TrayMouseDoubleClick += (s, e) => ShowMainWindow();

//Список для хранения информации о дисплеях

List<DisplayInfo> displayInfoList = new List<DisplayInfo>();

//Получение списка доступных дисплеев

foreach (Screen screen in Screen.AllScreens)

{

displayInfoList.Add(new DisplayInfo

{

DeviceName = screen.DeviceName,

DisplayName = "Дисплей " + (displayInfoList.Count + 1)

});

//Источник данных для ComboBox

DisplayBox.ItemsSource = displayInfoList;

DisplayBox.DisplayMemberPath = "DisplayName";

DisplayBox.SelectedValuePath = "DeviceName";

}

// Горячие клавиши

private void SetRecordingHotkey()

{

// Загрузка горячей клавиши из настроек

string hotkeyText = Properties.Settings.Default.RecordingHotkey;

// Парсинг горячей клавиши

KeyGestureConverter converter = new KeyGestureConverter();

recordingHotkey = converter.ConvertFromString(hotkeyText) as KeyGesture;

if (recordingHotkey == null)

{

// Если горячая клавиша не удалось загрузить из настроек, используйте значение по умолчанию

recordingHotkey = new KeyGesture(Key.F9);

}

// Зарегистрируем обработчик горячей клавиши

CommandBinding binding = new CommandBinding(ApplicationCommands.New);

binding.Executed += RecordingHotkey\_Executed;

CommandBindings.Add(binding);

// Привязываем горячую клавишу к окну

InputBinding inputBinding = new InputBinding(ApplicationCommands.New, recordingHotkey);

InputBindings.Add(inputBinding);

}

private void SetStopHotkey()

{

string hotkeyText = Properties.Settings.Default.StopHotkey;

stopHotkey = new KeyGesture(Key.F12);

CommandBinding binding = new CommandBinding(ApplicationCommands.Close);

binding.Executed += StopHotkey\_Executed;

CommandBindings.Add(binding);

InputBinding inputBinding = new InputBinding(ApplicationCommands.Close, stopHotkey);

InputBindings.Add(inputBinding);

}

private void RecordingHotkey\_Executed(object sender, ExecutedRoutedEventArgs e)

{

// Обработка нажатия горячей клавиши для начала записи

RecButton\_Click(sender, e);

}

private void StopHotkey\_Executed(object sender, ExecutedRoutedEventArgs e)

{

// Обработка нажатия горячей клавиши для кнопки "Стоп"

StopButton\_Click(sender, e);

}

// Запись видео и водяной знак

private void RecButton\_Click(object sender, RoutedEventArgs e)

{

RecButton.IsEnabled = false;

StopButton.IsEnabled = true;

if (waveIn == null)

{

waveIn = new WaveInEvent();

waveIn.DataAvailable += WaveIn\_DataAvailable;

// Получить выбранный дисплей из ComboBox

string selectedDeviceName = (DisplayBox.SelectedValue as string);

// Проверка на то, что пользователь выбрал дисплей

if (string.IsNullOrEmpty(selectedDeviceName))

{

System.Windows.MessageBox.Show("Выберите дисплей для записи.");

return;

}

savePath = SaveRoad.Text;

if (!string.IsNullOrEmpty(savePath))

{

if (audioBox.SelectedItem != null)

{

string selectedAudio = (audioBox.SelectedItem as ComboBoxItem).Content.ToString();

switch (selectedAudio)

{

case ".wav":

string fileNameAudioWav = "audio\_" + DateTime.Now.ToString("yyyyMMdd\_HHmmss") + selectedAudio;

writer = new WaveFileWriter(Path.Combine(savePath, fileNameAudioWav), waveIn.WaveFormat);

break;

case ".mp3":

string fileNameAudioMp3 = "audio\_" + DateTime.Now.ToString("yyyyMMdd\_HHmmss") + selectedAudio;

writer = new WaveFileWriter(Path.Combine(savePath, fileNameAudioMp3), waveIn.WaveFormat);

break;

}

}

// Начать запись

waveIn.StartRecording();

// Начать отслеживание времени записи

recordingTime = TimeSpan.Zero;

timer = new Timer

{

Interval = TimeSpan.FromSeconds(1)

};

timer.Tick += Timer\_Tick;

timer.Start();

int width = 1920; // Ширина видео

int height = 1080; // Высота видео

int fps = 30; // Кадры в секунду (FPS)

if (videoQualityBox.SelectedItem != null)

{

string selectedQuality = (videoQualityBox.SelectedItem as ComboBoxItem).Content.ToString();

switch (selectedQuality)

{

case "Низкое":

bitrate = 2000000;

break;

case "Среднее":

bitrate = 8000000;

break;

case "Высокое":

bitrate = 20000000;

break;

}

}

// Определение экрана для записи на основе выбранного имени

Screen selectedScreen = Screen.AllScreens.FirstOrDefault(s => s.DeviceName == selectedDeviceName);

if (selectedScreen != null)

{

// Запускаем захват рабочего стола выбранного дисплея

screenCapture = new ScreenCaptureStream(selectedScreen.Bounds, fps);

screenCapture.NewFrame += ScreenCapture\_NewFrame;

screenCapture.Start();

// Начинаем запись видео

videoWriter = new VideoFileWriter();

if (videoBox.SelectedItem != null)

{

string selectedExtension = (videoBox.SelectedItem as ComboBoxItem).Content.ToString();

switch (selectedExtension)

{

case ".mp4":

videoExtension = ".mp4";

break;

case ".avi":

videoExtension = ".avi";

break;

}

string fileName = "video\_" + DateTime.Now.ToString("yyyyMMdd\_HHmmss") + videoExtension;

videoWriter.Open(Path.Combine(savePath, fileName), width, height, fps, VideoCodec.MPEG4, bitrate);

}

isRecording = true;

}

else

{

System.Windows.MessageBox.Show("Выбранный дисплей не найден.");

}

}

else

{

System.Windows.MessageBox.Show("Выберите путь для сохранения!");

}

}

}

private void ApplyWatermark(Bitmap frame)

{

if (!string.IsNullOrEmpty(watermarkImagePath))

{

// Загружаем изображение водяного знака

Bitmap watermark = new Bitmap(watermarkImagePath);

watermark = new Bitmap(watermark, new System.Drawing.Size(75, 75));

// Размещаем водяной знак на видеокадре

using (Graphics g = Graphics.FromImage(frame))

{

g.DrawImage(watermark, new System.Drawing.Point(frame.Width - watermark.Width - 10, frame.Height - watermark.Height - 10));

}

}

}

private void ScreenCapture\_NewFrame(object sender, NewFrameEventArgs eventArgs)

{

if (IsRecordingValid())

{

Bitmap frame = eventArgs.Frame;

ApplyWatermark(frame);

videoWriter.WriteVideoFrame(frame);

lastFrameTime = DateTime.Now;

}

}

private void WaveIn\_DataAvailable(object sender, WaveInEventArgs e)

{

writer?.Write(e.Buffer, 0, e.BytesRecorded);

}

private void StopButton\_Click(object sender, RoutedEventArgs e)

{

RecButton.IsEnabled = true;

StopButton.IsEnabled = false;

if (waveIn != null)

{

waveIn.StopRecording();

waveIn.Dispose();

waveIn = null;

if (writer != null)

{

writer.Close();

writer.Dispose();

writer = null;

}

if (timer != null)

{

timer.Stop();

timer.Tick -= Timer\_Tick;

timer = null;

}

if (isRecording)

{

isRecording = false;

screenCapture.SignalToStop();

screenCapture.WaitForStop();

videoWriter.Close();

videoWriter.Dispose();

}

}

}

// Параметр для смены дисплея

private void DisplayBox\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

int fps = 30;

if (isRecording)

{

screenCapture.SignalToStop();

screenCapture.WaitForStop();

string newDeviceName = (DisplayBox.SelectedValue as string);

Screen newScreen = Screen.AllScreens.FirstOrDefault(s => s.DeviceName == newDeviceName);

if (newScreen != null)

{

screenCapture = new ScreenCaptureStream(newScreen.Bounds, fps);

screenCapture.NewFrame += ScreenCapture\_NewFrame;

screenCapture.Start();

}

else

{

System.Windows.MessageBox.Show("Выбранный дисплей не найден.");

}

}

}

public class DisplayInfo

{

public string DeviceName { get; set; }

public string DisplayName { get; set; }

}

internal class HotKey

{

private static Dictionary<int, Action> RegisteredActions = new Dictionary<int, Action>();

private const int WM\_HOTKEY = 0x0312;

[DllImport("user32.dll")]

private static extern bool RegisterHotKey(IntPtr hWnd, int id, int fsModifiers, int vk);

[DllImport("user32.dll")]

private static extern bool UnregisterHotKey(IntPtr hWnd, int id);

private static int currentId;

public static void Register(Key key, KeyModifier keyModifier, Action action)

{

int id = currentId++;

RegisterHotKey(IntPtr.Zero, id, (int)keyModifier, KeyInterop.VirtualKeyFromKey(key));

RegisteredActions[id] = action;

}

public static void Unregister()

{

foreach (var id in KeyHandler.RegisteredActions.Keys)

{

UnregisterHotKey(IntPtr.Zero, id);

}

}

public class KeyHandler : MainWindow

{

private HwndSource source;

public static Dictionary<int, Action> RegisteredActions = new Dictionary<int, Action>();

public KeyHandler()

{

try

{

source = HwndSource.FromHwnd(new WindowInteropHelper(this).Handle);

source.AddHook(HwndHook);

}

catch (Exception ex)

{

Console.WriteLine("Error initializing KeyHandler: " + ex.Message);

}

}

private IntPtr HwndHook(IntPtr hwnd, int msg, IntPtr wParam, IntPtr lParam, ref bool handled)

{

if (msg == WM\_HOTKEY)

{

int id = wParam.ToInt32();

if (RegisteredActions.TryGetValue(id, out var action))

{

action.Invoke();

}

handled = true;

}

return IntPtr.Zero;

}

}

}

public enum KeyModifier

{

None = 0,

Alt = 1,

Control = 2,

Shift = 4,

Win = 8

}